

YILDIZ TEKNİK ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

Hpc I 6003 Mikr. İle X-Y-Z Koor.
Sist. Kon.

YÜKSEK LİSANS TEZİ

Can Özkan

1990

YILDIZ TEKNİK ÜNİVERSİTESİ
KÜTÜPHANE DOKÜMANTASYON
DAİRE BAŞKANLIĞI

Kot	: R 368 : 58
Alındığı Yer	: Fen Bil. Ens. :
Tarih	: 13.4.95
Fatura	:
Fiyatı	: 65.000.-
Ayniyat No	: 1-4
Kayıt No	: 51036
UDC	:
Ek	:

YILDIZ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HPC 16003 Mikrodenetleyicisi ile X-Y-Z Koordinat Sisteminin Kontrolü



YÜKSEK LİSANS TEZİ

Elektrik Mühendisi
Can ÖZKAN

Tez yöneticisi
Prof. M. Yahya KARSLIGİL

İSTANBUL - 1990

İ Ç İ N D E K İ L E R

ABSTRACT	1
ÖNSÖZ	11
GİRİŞ	111
1. Bölüm GENEL TANIMLAMA.....	1
1.1 GELİŞMİŞ TEK YONGA CMOS MİKRODENETLEYİCİ.....	1
1.1.1 Özellikler.....	1
2. Bölüm MİMARİ ÖZELLİKLERİ	4
2.1 VON NEUMANN MİMARISI VE BELLEK HARİTASI.....	4
2.2 ELDE KULLANIMI	5
2.3 ELDENİN TEST EDİLMESİ	5
2.4 8/16 BITLİK VERİ KULLANIMI	6
2.4.1 Byte ve word'lerin adreslenmesi	6
2.4.2 Çeşitli veri uzunluklarının birlikte kullanılması	6
2.4.3 Örnekler	8
2.5 DURUMSAL KOMUTLAR VE ATLAMALAR	10
3. Bölüm PROGRAMLAMA MODELİ	12
3.1 SAKLAYICILAR	12
4. Bölüm ADRESLEME MODLARI	15
4.1 GİRİŞ	15
4.2 BİR ADRESLİ GENEL MODLAR	19
4.2.1 Dolaylı B saklayıcılı adresleme	19
4.2.2 Dolaylı X saklayıcılı adresleme	19
4.2.3 Doğrudan adresleme	19

4.2.4	Dolaylı adresleme	20
4.2.5	Sıralı adresleme	20
4.3	İKİ ADRESLİ GENEL MODLAR	21
4.3.1	İvedi adresleme	21
4.3.2	Bellekten belleğe doğrudan adresleme	21
4.3.3	Belleğe ivedi adresleme	21
4.4	ÖZEL MODLAR	22
4.4.1	X saklayıcılığı	22
4.4.2	B saklayıcılığı	22
4.4.3	Çift saklayıcılığı dolaylı adresleme	24
5. Bölüm	KOMUT SETİ	25
5.1	GİRİŞ	25
5.2	KOMUTLAR	25
6. Bölüm	FONKSİYON PINLERİNİN TANITIMI	47
7. Bölüm	DONANIM BAĞLANTILARI	50
7.1	BESLEME VE TOPRAKLAMA	50
7.2	SAAT	50
7.3	RESET	50
8. Bölüm	BELLEK TASARIMI VE SAKLAYICILAR	54
8.1	KONTROL SAKLAYICILARI	54
8.2	İŞLEMCİ DURUM SÖZCÜĞÜ	54
9. Bölüm	PORT YAPILARI VE İŞLEMLERİ	59
9.1	PORT A	59
9.1	PORT B	61
9.1	PORT I	62

9.1	PORT D	63
9.1	PORT P	64
10.	Bölüm KESMELER	66
11.	Bölüm MICROWIRE / PLUS	71
11.1	GİRİŞ	71
11.2	SAKLAYICILAR	74
11.3	İŞLEMİ GERÇEKLEŞTİRME	77
12.	Bölüm YOL FONKSİYONLARI	80
12.1	YOL MODLARI VE SEÇİMLERİ	80
12.1.1	Tek yonga veya genişletilmiş mod	80
12.1.2	Normal veya ROM'suz mod	81
12.1.3	8 bit veya 16 bit mod	81
13.	Bölüm ASENKRON SERİ HABERLEŞME ARA BİRİMİ(UART)	87
13.1	GİRİŞ	87
13.2	UART'IN GİRİŞ ÇIKIŞ PINLERİ	87
13.3	UART'IN ÇALIŞMASI	88
13.4	ÇERÇEVE FORMATLARI	90
13.5	HATA BAYRAKLARI	91
13.6	KESMELER	93
13.7	SAATLEME	94
13.8	WAKE-UP MODU	95
13.9	UART'IN KONTROL SAKLAYICILARI	97
14.	Bölüm RESET DURUMU	100
15.	Bölüm GENEL ÇEVRE ARA BİRİMİ (UPI)	102
15.1	GİRİŞ	102

15.2	İÇ ORGANİZASYON	102
15.3	UPI KONTROL SAKLAYICISI	105
15.4	PIN FONKSİYONLARININ DETAYLARI	107
15.4.1	UPI yazma işlemi	107
15.4.2	UPI okuma işlemi	107
15.5	EL SIKIŞMA	110
15.6	UPI MODUNU AYARLAMA	117
16.	Bölüm ZAMANLAYICILAR	118
16.1	GİRİŞ	118
16.2	ZAMANLAYICI İŞLEMLERİ	119
16.2.1	T0 ve T8 zamanlayıcıları	119
16.2.2	T1 zamanlayıcısı	122
16.2.3	T2 ve T3 zamanlayıcıları	122
16.2.4	Darbe/boşluk ayarlamalı zam. (T4-T7)	124
16.3	ZAMANLAYICI ÇIKIŞLARI	124
16.4	ZAMANLAYICI SAKLAYICILARI	126
16.5	ZAMANLAYICILARI KURMAK VE AYARLAMAK	134
16.6	ZAMANLAYICI UYGULAMALARI	136
17.	Bölüm HPC 46003 İLE XYZ KOORDİNAT SİSTEM KONTROLU ...	142
17.1	SİSTEMDE KULLANILAN BİRİMLER	142
17.2	SİSTEMİN ADRES ALANI	142
17.3	SİSTEMİN TANITIMI	143
18.	Bölüm VIDEO GRAFİK KARTI	144
18.1	GÖRÜNTÜNÜN OLUŞUMU	144

18.2 VIDEO KARTININ ÇALIŞMA PRENSİBİ	148
18.3 DONANIM	150
19. Bölüm STEP MOTOR KARTI	158
19.1 STEP MOTOR DEVRESİ VE ÇALIŞMASI	158

In this thesis we aimed to develop the systems and provide new features them.

In a CNC, it must be converted the physical information to digital information that the computers understand. One of them is to measure the position. For that here we developed a position sensor and a motor control circuit to be able to give the new position to system.

ABSTRACT

It is common to use CNC Machines in industry because they provide a lot of features like that reliability , quality , easy to use etc... But in our country it has just began the studies about this area .

In this thesis we aimed to develop the systems and provide new features them .

In a CNC , it must be converted the physical information to digital information that the computers understand. One of them is to measure the position. For that here we developed a position sensor and a motor control circuit to be able to give the new position to system.

Günümüzde CNC makinalar sağladıkları hassasiyet , iş kalitesi , zaman tasarrufu , vb. gibi avantajlarla iş hayatının vazgeçilmez birer parçası haline gelmişlerdir.Fakat ülkemizde bu sahadaki çakışmalar daha yeni başlamıştır ve geliştirilmeyi beklemektedir.

Bu tezin amacı bir CNC makinanın temelini teşkil eden mesafe ve buna bağlı olarak motor kontrollerini gerçekleştirerek , geliştirilmeye açık olan bir sistem teşkil etmektir.Bunu yaparken bir CNC tezgahta bulunması gereken kontrolör , giriş çıkış birimleri , güç katı ve mesafe ölçüm devreleri gibi bütün donanımlar gözönünde bulundurulmuş , sistem yazılım yoluyla geliştirilmeye açık hale getirilmiştir.

Sitem 5 kısımdan müteşekkildir.Bunlar sırasıyla mikroişlemci kısmı , CRT kontrol , tuş takımı , encoder arabirimi ve motor kontrol kısımlarıdır.Sistemde mesafe ölçüleri için incremental encoder kullanılmıştır.Kullanım nedeni ucuzluğu ve prototipe adaptasyonunun kolaylığıdır.Bunun yerine donanımda hiçbir değişiklik yapılmadan sayısal cetveller kullanılabilir.

G İ R İ Ş

Kontrol bilimi özellikle 19. yy da ortaya çıkan endüstri devrimi ile beraber önem kazanmıştır. Geçmiş çağlarda işlemlerin pratik olarak yapmış oldukları otomatik kontrolli sistemler bir kenara bırakılacak olunursa James Watt 'ın buhar makinasının hızını kontrol için geliştirdiği Watt Regülatörü ile birlikte otomatik kontrol , bir bilim dalı haline gelmeye başladı.

Başlangıçta mekanik düzeyde kalan otomatik kontrol elektronik ilminin gelişmesiyle birlikte elektronik düzenleme vasıtasıyla sağlanır olmuştur. Bunda endüstride katı ve sıvı yakıtlardan elektrige doğru olan geçiş önemli rol oynamıştır.

1950 yıllarının sonlarına doğru programlanabilen genel amaçlı mikroişlemcilerin ortaya çıkışı ile otomatik kontrolde yeni bir çığır açılmış oldu. Çünkü mikroişlemciler ile çok karışık uzun hesap gerektiren işlemler çok kısa sürede yapılabilmektedir. Yeni mikroişlemcili kontrol sistemlerinde ölçme düzeninden kaynaklanan hatalarda oldukça azdır. Böylece mikroişlemcilerin kontrol sistemlerine girmesiyle birlikte kontrol kurallarının hızlı ve esnek bir şekilde gerçekleştirilmesi sağlanmış oldu.

1980 lerden sonra mikroişlemcilerin geliştirilmesiyle elde edilen ve kontrolde birçok kolaylıklar sağlayarak mikrodenetleyiciler bu sahada kullanılmaya başlandı. Mikrodenetleyiciler , bir mikrobilgisayar için gerekli

olan bütün sistem zamanlayıcılarına , dahili belleğe , ROM ve RAM belleklerine ve kontrol fonksiyonlarını gerçekleştirmek üzere tasarlanan giriş/çıkış pinlerine sahiptir. Bu imkanlar kullanıcıya tek bir entegre paketi halinde sunulmaktadır.

Bu mikrodenetleyicilerden birisi de 1987 yılının sonlarına doğru National Semiconductor firmasının piyasaya çıkardığı HPC(High Performance Controller , yüksek performanslı kontrolör) serisidir.

Bu tezde , HPC mikrodenetleyicisinin incelenmesi ve XY koordinat sistem kontrolü amacıyla kullanılmasına ilişkin çalışmalar yapılmıştır.

HPC16083 tek yongada tek entegre bir mikrodenetleyicidir. Aşağıdaki seri Giriş/Çıkış arabirimi (UART), dokuz tane 16 bitlik zamanlayıcı ve MICROWIRE/PLUS™ arabirimi gibi yonga fonksiyonları, yüksek seviyede bir sistem gerçekleştirmeye yardımcı olur. (Şekil 1-1)

1.1.1 Özellikler

- CMOS Teknoloji: çok düşük güç harcaması
- HIZ-Saat frekansının frekansı 30 MHz. olduğunda, saatlayıcı komutlarının işlem süresi 134 ns.dir.
- Veriyolu, Arithmetik İşlem Birimi ve saatlayıcılar 16 Bitlidir.
- Komut seti oldukça geliştirilmiştir.

* MICROWIRE/PLUS National Semiconductor firmasının tescilli markasıdır.

1B Ö L Ü M

GENEL TANIMLAMA

1.1 GELİŞMİŞ TEK YONGA CMOS MİKRODENETLEYİCİ :

HPC16083 bir mikrobilgisayarın gerektirdiği bütün sistem zamanlayıcıları, dahili lojikleri, ROM'u, RAM'ı ve Giriş/Çıkış birimlerini içine alan tek yonga mikrodeneleyicidir. HPC ailesinin her elemanı NATIONAL firmasının geliştirdiği mikroCMOS teknolojisi ile üretilmiştir. Bu teknoloji HPC ailesini karakterize eden, hızlı, esnek Giriş/Çıkış kontrolünü, etkili veri işlemeyi ve yüksek hızda işlem yapmayı sağlar.

HPC16083 tek yongada tam anlamıyla bir mikrodeneleyicidir. Asenkron seri Giriş/Çıkış arabirimi (UART), dokuz tane 16 bitlik zamanlayıcı ve MICROWIRE/PLUSTM arabirimi gibi yonga fonksiyonları, yüksek seviyede bir sistem gerçekleştirmeye yardımcı olur. (Bkz. Şekil 1-1)

1.1.1 Özellikler

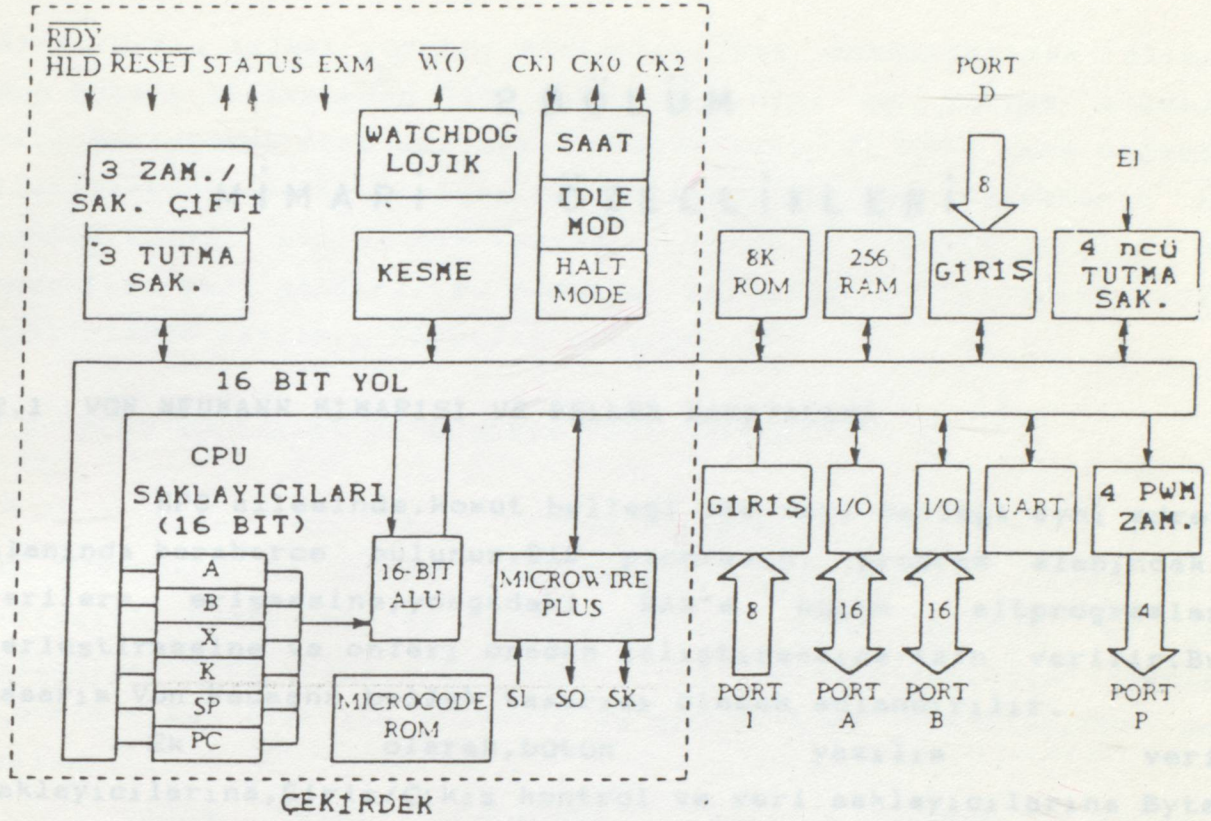
- CMOS Teknoloji-çok düşük güç harcaması
- HIZ-Saat işaretinin frekansı 30 KHz. olduğunda, saklayıcı komutlarının işlem süresi 134 ns.dir.
- Veriyolu, Aritmetik İşlem Birimi ve saklayıcılar 16 Bitlidir.
- Komut seti oldukça geliştirilmiştir.

* MICROWIRE/PLUS Natioal Semiconductor firmasının tescilli markasıdır.

- 16 Bitlik Word* ve 8 Bitlik Byte* erişimini mümkün kılan gelişmiş mimari
- Tek Byte'lık atlama (jump), altprogram çağırma ve RAM erişimi yapma kabiliyeti
- 16*16'lık Çarpma ve 32*16'lık Bölme işlemi yapma kabiliyeti
- Yonga içinde 8K*8 ROM ve 256*8 RAM imkanı
- Yonga dışında 64K'lık adresleme imkanı
- 16 Bitlik Yığın Göstergesi saklayıcısı (Stack Pointer)
- Temel ve Giriş/Çıkış saklayıcılarına bellek adresleme yöntemi ile erişme imkanı
- 52 tane Giriş/Çıkış hattı
- Microwire/Plus özelliği kullanılarak senkron Giriş/Çıkış yapabilme özelliği
- Programlanabilir Asenkron Seri Giriş/Çıkış Arabirimi
- Dokuz tane 16 Bitlik zamanlayıcı
- 10 zamanlayıcısı 4 tane giriş tutma saklayıcısı
- Watchdog özelliği ile işlemciyi kontrol etme özelliği

* Word : 16 Bitlik veri, bellek yada saklayıcı gruplarına verilen isim.

* Byte : 8 Bitlik veri, bellek yada saklayıcı gruplarına verilen isim.



ŞEKİL 1-1. HPC Blok Diyagramı

2. BÖLÜM

MİMARİ ÖZELLİKLERİ

2.1 VON NEUMANN MİMARISI VE BELLEK HARİTALAMA

HPC ailesinde, komut belleği ile veri belleği aynı adres alanında beraberce bulunur. Bir programın, program alanındaki verilere erişmesine, yongadaki RAM'e küçük altprogramlar yerleştirmesine ve onları oradan çalıştırmasına izin verilir. Bu tasarım Von Neumann bellek tasarısı olarak adlandırılır.

Ek olarak, bütün yazılım veri saklayıcılarına, Giriş/Çıkış kontrol ve veri saklayıcılarına Byte ve/veya Word düzeyinde erişilebilir. Bu tasarım da Bellek Haritalama olarak adlandırılır.

Von Neumann ve Bellek Haritalama düşüncelerinin birleşimi HPC'ye çok güçlü bir komut seti kazandırır.

Örnek vermek gerekirse:

X A,K ; Akümülatör ile K saklayıcısının içeriklerini değiştir.

Bu komut, Akümülatör'ün içeriği ile her hangi bir bellek alanının içeriğini değiştiren X komutunun basit bir durumdur. K saklayıcısı haritalanmış bellekte bir adrese (OOCA Hex) sahip olduğundan: bu komut, komuttaki 8 bitlik adres alanını kullanarak K saklayıcısına bir bellek gözü gibi erişir. Bu nedenle, HPC için özel olarak ayrılmış saklayıcı kaynaklı komutlara gerek yoktur.

SBIT 7, SIO ; Tasdik bitini birleyerek, karakteri geri gönder. Bu komut, genel amaçlı bit birleme (SBIT) komutunu kullanan bir

Giriş/Çıkış işlemi yürütür. Microwire/Plus arabiriminden alınan son Byte'ı ihtiva eden SIO saklayıcısının, en yüksek anlamlı biti birleştirilerek (set edilerek) değiştirilir. Bu etki yeni değerin iletilmesini sağlar. Böylece bu tek komut alınan karakterin en yüksek anlamlı bitini bir testik işaretçisi olarak birler ve karakteri geri gönderir. Bu nedenle, Giriş/Çıkış işlemleri için özel olarak ayrılmış komutlar gereksizdir.

2.2 ELDE KULLANIMI VE ELDENİN TEST EDİLMESİ

Elde (C) biti (PSW saklayıcısının 5. biti) ADC, SUBC, DADC ve DSUBC komutlarının yürütülmesi sırasında, Aritmetik Lojik Birime (ALU) doğrudan elde girişi olarak kullanılır. Elde biti, bu komutların yürütülmesinden sonra ALU'dan çıkan elde ile yüklenir ve ADD ve ADDS komutlarında da kullanılır.

Elde biti doğrudan ikili düzende eldeyi gösterdiğinden, elde bitinin işlevleri toplama ve çıkarma işlemleri sırasında farklılık gösterir.

Elde girişi kabulüne ek olarak, eğer elde bitinin değeri sıfırsa iki verinin toplamı doğrudur. Toplamadan sonra elde bitinin 1 değerini alması, işaretli taşma durumu olduğunu gösterir. Eğer işaretli taşma yoksa elde biti 0 değerini alacaktır.

HPC'nin -ileride açıklanacak olan- iki çıkarma komutu da elde kullanır. Ancak bu elde; çıkarılacak iki verinin arasındaki farkı doğru olarak hesaplamak için baştan 1 değerine getirilmelidir. Eğer elde bitinin değeri 0 olursa, sonuç gerçek sonuçtan bir eksik olacaktır. Çıkarma işleminden sonra elde bitinin değeri 0 ise işaretli taşma (borç durumu) oluşmuştur, 1 ise sonuç doğrudur.

2.4 8/16 BITLİK VERİ KULLANIMI

HPC 8bitlik (Byte) ve 16 bitlik (Word) verileri aynı kolaylıkta kullanılabilecek şekilde tasarlanmıştır.

2.4.1 Byte ve Word'lerin Adreslenmesi

HPC ailesinde kullanılan bellek byte düzeninde adreslenir. Word 16 bitlik bir alan kapsar. Dolayısıyla ardarda gelen iki byte word olarak düşünülebilir. Bir word'ün adresi daima çift numaralı adres olmalıdır. Bu word'ün düşük anlamlı byte'ının adresidir. Word'ün yüksek anlamlı byte'ının adresi, düşük anlamlı byte'ının adresinin bir fazlasıdır. HPC'nin mimarisi word'lerin düşük anlamlı byte'larının tek numaralı adreslere yerleştirilmesine izin vermez.

HPC yongasındaki bütün 16 bitlik saklayıcılara word olarak erişilebildiği gibi byte olarak da erişilebilir. Örneğin B saklayıcısının sadece düşük anlamlı yarısı, onun bellek haritasındaki yeri kullanılarak arttırılabilir veya A portunun Giriş/Çıkış veri saklayıcısının sadece yüksek anlamlı byte'ı, yine bellek adresi kullanılarak değiştirilebilir.

2.4.2 Çeşitli Veri Uzunluklarının Birlikte Kullanılması

HPC aritmetik işlemleri 16 bit üzerinden yapar. Bununla beraber, diğer mikrobilgisayarlardan farklı olarak HPC, farklı uzunluktaki veriler üzerinde doğrudan işlem yapabilir. Örneğin, bir dönüştürme işlemi yapmadan tek byte uzunluğundaki bir veriyi bir word uzunluğundaki veri ile toplayabilir. HPC bu işlemden önce, programcının biruzunluk çevirme işlemi yapmasını gerektirmeden otomatik olarak 8 bit (byte) verileri -üst yarısına sıfır yazarak- 16 bit verilere dönüştürür. Bu işleme sıfır genişletme denmektedir (Yani 8 bitlik byte düzenindeki bilgi 16 bitlik word düzeyine dönüştürüldüğünde, word'ün yüksek anlamlı byte'ına sıfır yazılır).

HPC başlangıçta işaretli değerler kullandığından, HPC için sıfır genişletme tasarımı seçilmiştir. Sıfır genişletme, 0-255 sınırları arasındaki pozitif sayılar kullanıldığı zaman doğru bir yaklaşımdır. Örneğin tek byte'lık veri 255 (FF Hex) değerine sahipse ve işaretli sayı ise; sıfır genişletme işleminden sonra aynı değerde kalır (00FF Hex).

Bununla beraber tek byte'lık bir veri ikiye tülenmiş sayı (sınırları -128 den +127 ye kadar olan) olarak dönüştürülürse, değişim işlemi için farklı bir yaklaşım kullanılmalıdır. Eğer verinin işaret bitinin (7. bit) değeri 1 ise, dönüştürülecek verinin üst yarısına 0 yerine 1 yazılmalıdır. HPC'nin bir verinin işaretli mi işaretli değil mi olduğunu anlaması mümkün olmadığından işaretli sayılar üzerindeki dönüştürmeler programcının sorumluluğu altındadır. Fakat uzunluk dönüştürme işlemi zor ve karışık bir işlem değildir. Bir işaretli tek byte'lık verinin dönüştürülmesi şöyle yapılabilir (Burada otomatik sıfır genişletmeden de yararlanılmaktadır):

```
LD      TAMPWD, BVER      ;Byte veriyi otomatik sıfır
                           genişletmeyle bir tampon word alana
                           yükle
```

```
IFBIT 7, TAMPWD.B ;Verinin negatif (işaret biti=1) olup
                  olmadığını test et.
```

```
DECSZ  TAMPWDH      ;Eğer negatif ise üst byte'ı bir artır
                           (00 dan FF Hex e)
```

Burada BVER dönüştürülecek işaretli (ikiye tülenmiş) veriyi, TAMPWD word düzeyinde tampon alanı ve TAMPWDH bu tampon alanın yükecek anlamdaki byte'ını göstermektedir. Bu işlemden sonra BVER'in orjinal değerinden, uygun işaret genişletme ile TAMPWD'ye dönüştürülmüş değer yüklenmiştir. Burada TAMPWD yerine

akümülatör de kullanılabilirdi.

16 bitlik işaretli veriler dönüştürülmeye ihtiyaç olmadan toplama ve çıkarma da kullanılabilirler. Eger işaretli veriler uygun sınırlarda tutulmuşsa, bu işlemlerden sonra doğru sonuç elde edilir.

-- Akümülatör için veri uzunluk ayarı

Akümlatörün her iki byte'ı (A saklayıcısı) bir hedef adresi olduğu zaman, kendiliğinden işleme sokulur. Bununla birlikte akümülatör üzerinde işlem yapma, byte'larının tek tek adreslenmesiyle -bellek haritasındaki adresleri kullanılarak gerçekleştirilebilir. Örneğin bu akümülatörün yüksek anlamlı byte'ını etkilemek için sadece düşük anlamlı byte'ını bir arttırmak için şu komut kullanılabilir:

INC H'C8.B ;C8 akümülatörün adresidir.

Adres vermeden INC A komutu kullanılsaydı, akümülatörün yüksek anlamlı byte'ı düşük anlamlı byte'ından bir elde gelmesi halinde değişecekti.

-- Elde bitinin uygun yerden yüklenmesi

Akümlatörü otomatik olarak değiştiren bütün etkiler daima 16 bitlik eldeyi gösteren elde bayrağını içine almaz. Akümülatörde 8 bitlik verilerin kolayca işlenebilmesini sağlamak için, elde bayrağı toplama ve çıkarma işlemi esnasında verilerin uzunluğuna uygun olan bit yerlerinden (Byte veriler için 7.bit, Word veriler için 15.bit) çıkan elde ile yüklenir. Eldenin yeri iki veriden birinin uzunluğuna göre saptanır ve verinin kontrol ettiği elde yerinin seçimi, verilere erişmek için kullanılan adresleme modundan belirlenir.

2.4.3 Örnekler

İlk üç örnek akümülatörün hedef adresi olduğu durumları gösterir.

Örnek 1:

```
LD    A,#H'EF FE           ; A ← EF FE Hex
ADD   A,#4                  ; A ← F002 Hex ve C ← 0
```

Bu durumda elde üretilmedi. Çünkü akümülatör, kaynak veri ivedi olduğundan elde ve verinin her ikisi için 16 bitlik hedef adresi olarak davrandı.

Örnek 2:

```
LD    A,#H'EF FE           ; A ← EF FE Hex
LD    23.B,#4               ; 23 No.lu bellek byte'ına 4 alındı.
ADD   A,23.B                ; A ← F002 Hex ve C ← 1
```

Bu durumda elde biti byte taşması olduğunu gösteriyor. Çünkü kaynak veri byte düzeninde. Bu işlemden akümülatörün tamamı etkilendi.

Örnek 3:

```
LD    A,#H'C823             ; A ← C823 Hex
ADD   A,#H'8900             ; A ← 5123 Hex ve C ← 1
```

Bu durumda elde biti bir word taşması olduğunu gösteriyor.

Aşağıdaki örnekler hedef adresi olarak akümülatör yerine doğrudan adreslenmiş bellek alanlarına sahiptir. Bu durumda hedef adresi ya byte yada word olarak belirlenebilir. Elde yerini belirleyen hedef adresidir.

LOCB bellekte bir byte ve LOCW bellekte bir word alandır.

Örnek 4:

```
LD    LOCB,#H'23            ; LOCB ← 23 Hex
ADD   LOCB,#H'E0            ; LOCB ← 03 Hex ve C ← 1
```

Bu durumda hedef adresi byte olduğundan, bir byte eldesi üretildi.

Örnek 5:

```
LD    LOCW,#H'56            ; LOCW ← 56 Hex
ADD   LOCW,#H'FEC0          ; LOCW ← FF16 Hex ve C ← 0
```

Bu durumda elde bitinin sıfırlanması bir byte eldesi oluşmasına rağmen word eldesi oluşmadığını gösteriyor.



Örnek 6:

LD LOCB,#H'45 ; LOCB ← 45 Hex

LD LOCW,#H'OOF5 ; LOCW ← OOF5 Hex

ADD LOCW,LOCB ; LOCW ← 013A Hex ve C ← 0

Bu durumda elde oluşmadı; bir byte eldesi oluşmasına rağmen word eldesi üretilmedi.

2.5 DURUMSAL KOMUTLAR VE ATLAMALAR

Birçok mikroişlemciden farklı olarak, HPC durumsal dallanma komutlarına sahip değildir. Bunun yerine, herhangi bir durumsal komutu yürütmeyi gerçekleştirecek bir atlama tekniği kullanılmıştır. Bazı komutlar, bazı koşullar altında kendisinden sonraki komutu yürüten atlama vasfı sahiptir. Bu özellik denetleyici çevre birimlerinde birçok uygulamaya sahiptir. Örneğin aşağıdaki blok sadece üç tane tek byte'lık komutla bir çevrim yürütür.

ÇEVİRİM : LD A,[X+].B ; A'ya X üzerinden bir değer yükle ve x'i bir artır.

XS A,[B+].B ; A ile B'nin byte olarak adreslenmiş kısmının içeriklerini değiştir ve B'nin içeriği K'ninkinden büyük ise aşağıdaki komutu atla.

JS ÇEVİRİM ; ÇEVİRİM'e dallen (Komut atlama yoke).

Atlama özelliğine sahip komutlar şunlardır:

- | | |
|-------|---|
| IFGT | İlk veri ikinciden daha büyükse bir sonraki komut yürütülür, değilse atlanır. |
| IFEQ | Veriler eşitse bir sonraki komut yürütülür, değilse atlanır. |
| IFBIT | Belirlenen bitin değeri 1 ise bir sonraki komut yürütülür, değilse atlanır. |

IFC PSW saklayıcısındaki elde bitinin değeri 1 ise, bir sonraki komut yürütülür, değilse atlanır
IFNC PSW saklayıcısındaki elde bitinin değeri 0 ise bir sonraki komut yürütülür, değilse atlanır.
DECSZ. Bu komut bir değeri azaltır ve sonuç sıfırsa bir sonraki komutu atlar.

LDS, XS AKLAYT Otomatik bir artırmalı/azaltmalı adresleme modu kullanan bu komutlar, B saklayıcısının değeri K saklayıcısında bulunan adres sınırını aşarsa, bir sonraki komutu atlar. B saklayıcısı arttırıldığında sonuç K'nin değerinden büyük olursa, bir sonraki komut atlanır; B saklayıcısı azaltıldığında sonuç K'nin değerinden küçük olursa bir sonraki komut atlanır.

RETSK Bu komut bir altprogramdan geri dönüş yürütür. Fakat karşılaşılan ilk komut koşulsuz olarak atlanır.

Atlamaya ek olarak bir başka karar verme kabiliyeti de sağlanmıştır: Çoklu dallanma. JID ve JIDW komutları bu amaç içindir. Bu komutları atlanılacak yerin PC'ye bağlı uzaklıklarını ihtiva eden byte (JID) yada word'lerin (JIDW) bir tablosu takip eder. Dallanılacak yerin PC'ye uzaklığını gösteren tablo yeri akümülatörün içeriği tarafından seçilir.

3. BÖLÜM

PROGRAMLAMA MODELİ

3.1 SAKLAYICILAR

HPC'nin programlama modeli altı tane 16 bitlik saklayıcı ve elde 32 bitinden oluşur. Bunlar aşağıda açıklanmıştır. Bütün saklayıcılar bellek haritasında adreslenmiştir. Saklayıcıların byte'ları da adreslenebilir ve bunlara tek tek erişilebilir.

AKÜMÜLATÖR (A) SAKLAYICISI : -0008 Adresinde-
16 Bitlik A saklayıcısı, birçok komut için hedef saklayıcısı olarak davranır. Bellekte istenilen herhangi bir adrese yerleştirilebilir. Yığının uzunluğu belleğe bağlıdır.

ADRES (B) SAKLAYICISI : -000C Adresinde-
16 Bitlik B saklayıcısı dolaylı adresleme için kullanılan ön adres saklayıcısıdır. Belirli komutlarda, bellek tablolarını ve dizilerini işaret etmek için otomatik olarak bir artırılır veya bir azaltılabilir. B saklayıcısı bu komutlarda K saklayıcısı ile karşılaştırılır ve bir sınıra ulaştığında çevrimin dışına çıkmak için bir komut atlama işlemi yürütülebilir.

ADRES (X) SAKLAYICISI : -000E Adresinde-
16 bitlik X saklayıcısı, B saklayıcısı ile aynı tarzda dolaylı adresleme için kullanılabilir. B saklayıcısındaki gibi, bir sınır testi olmaksızın, belirli komutlarda otomatik olarak bir artırılır veya azaltılabilir.

SINIR SABİTİ (K) SAKLAYICISI :

-00CA Adresinde-

16 Bitlik K saklayıcısı, B saklayıcısının otomatik artırılmasında veya azaltılmasında, bir sınır değeri tutmak için kullanılır. Bunu gerçekleştiren komutlarda, her ne zaman B saklayıcısının içerdigi adres K saklayıcısının içerdigi adresin ötesine ilerlemişse, bir komut atlama işlemi yürütülür.

YIĞIN GÖSTERGESİ (SP) SAKLAYICISI :

-00C4 Adresinde-

16 Bitlik SP saklayıcısı yığını adresleyen, yığın göstergesidir. SP saklayıcısı, PUSH (Yığına eleman atma) ve CALL (Alt program çağırma) komutları tarafından bir artırılır ve POP (Yığından eleman çekme) ve RETURN (Alt programdan geri dönme) komutları tarafından bir azaltılır. Kesmeler de 16 bitlik bir geri dönüş adresini yığına atarlar. SP saklayıcısı, bellek haritasındaki adresi kullanılarak okunabilir. Bu saklayıcıya aynı yöntem ile yazma da yapılabilir. Yığın 16 Bitlik bellekte istenilen herhangi bir alana yerleştirilebilir. Yığının uzunluğu belleğe bağlıdır.

PROGRAM SAYICI (PC) SAKLAYICISI : -00C6 Adresinde-

16 Bitlik PC saklayıcısı, 64K adresleme sınırının her yerindeki program belleğini adresler. Bu saklayıcı sadece okunabilir.

ELDE (C) BITİ: -00C6 Adresindeki PSW saklayıcısında-

Elde biti PSW saklayıcısının 5. bitinde tutulur. Elde biti, bir toplama veya çıkarma işlemlerinden sonra işaretli taşıma veya borç durumu olup olmadığını gösterir. Elde bitini etkilemek için byte ve word eldelerinin her ikisi de seçilebilir. Toplama komutları bir işaretli taşıma durumunda elde bitini birler. Çıkarma komutları ise, bir borç durumunda elde bitini sıfırlar. ADC, SUBC, DADC ve DSUBC komutlarında, bir önceki komuttan gelen elde yada borç girişi eldesi olarak

kullanılır. MULT ve DIV komutları elde bitini daima sıfırlar. DIVD komutu bir taşıma olursa veya sıfıra bölmeyle karşılaşılırsa elde bitini birler. Bu durumlar dışında elde bitini sıfırlar. Elde biti SC ve RC komutlarıyla doğrudan birleşir veya sıfırlanır. Böylece elde biti genel amaçlı bir bayrak olarak kullanılabilir. Elde biti SHL, SHR, RLC ve RRC gibi öteleme komutlarında, ötelenmiş bitin kaynağı veya hedef yeri olarak da kullanılır.

RPC'nin adresleme modları, verilerin daha etkin bir şekilde kullanılmasına sağlar. Adresleme modları, iki ayrı kategoriye ayrılır: Genel ve Özel. Genel adresleme modu birçok komutta kullanılır. Bunlar Tablo 4-1'de listelenmiştir. Özel adresleme modu birkaç komutta kullanılır. Bu mod 4.4 bölümünde açıklanmıştır.

Bir RPC komutu, yerleri adresleme moduyla tanımlanmış bir veya iki veriye sahip olabilir. Hedef veri, işlem sonucunun yerleştirildiği veridir. Kaynak veri, komut tarafından işlenen veya hareket ettirilen bir değerdir. Bir ifadede komutun önce yazılan ilk veri hedef, ikinci veri kaynak veridir.

Adresleme modlarının, belirlenen hedef ve kaynak verilerin yerlerini nasıl belirleyeceği aşağıda açıklanmıştır. Bunlar üç kategoriye bölünmüştür:

BİR ADRESLİ GENEL MOD

Bu adresleme modu, genel adresleme modu kullanılarak bütün komutlarda kullanılabilir. İki adresli komutlar (örneğin, Tablo 4-2'de bu mod hedef veri olarak A saklayıcısına ve kaynak veri olarak da bir tek adres ifadesini kullanır. Örneğin: "LD A, [B].W" komutu B saklayıcısına gösterdiği verdeki word'u A saklayıcısına yükler. Tek adresli komutlar arasında (örneğin, Tablo 4-3'te) tek adres ifadesi, genellikle hedef veri olan, bellekteki bir tek verinin adresini belirtir. Örneğin: "INC [B].W" komutu B saklayıcısına gösterdiği word'u bir artırır.

TABLO 4-1. 4. BÖLÜM ADRESLEME MODLARI

MODLAR	ADRESLEME	MODLARI
Döleyli B Saklayıcı Adr	(B).B (B).W	Byte'in adresi B'dedi Word'un adresi B'dedi
4.1 GİRİŞ	(X).B (X).W	Byte'in adresi X'dedi Word'un adresi X'dedi

HPC'nin adresleme modları verilerin daha etkin bir şekilde kullanılmasını sağlar. Adresleme modları iki ayrı kategoriye ayrılır: Genel ve Özel. Genel adresleme modu birçok komutta kullanılır. Bunlar tablo 4-1 de listelenmiştir. Özel adresleme modu birkaç komutta kullanılır. Bu mod 4.4 ncü kısımda açıklanmıştır.

Bir HPC komutu, yerleri adresleme moduyla tanımlanmış bir veya iki veriye sahip olabilir. Hedef veri, işlem sonucunun yerleştirildiği veridir. Kaynak veri komut tarafından işlenen veya hareket ettirilen bir değerdir. Bir ifadede komuttan sonra yazılan ilk veri hedef, ikinci veri kaynak veridir.

Adresleme modlarının, belirlenen hedef ve kaynak verilerin yerlerini nasıl belirleyeceği aşağıda açıklanmıştır. Bunlar üç kategoriye bölünmüştür:

BİR ADRESLİ GENEL MOD

Bu adresleme modu, genel adresleme modu kullanan bütün komutlarda kullanılabilir. İki adresli komutlar sınıfında (Bkz. Tablo 4-2), bu mod hedef veri olarak A saklayıcısını ve kaynak veri olarak da bir tek adres ifadesini kullanır. (Örneğin ; "LD A, [B].W" komutu B saklayıcısının gösterdiği yerdeki word'u akümülatöre yükler.) Tek adresli komutlar sınıfında (Bkz. Tablo 4-3), tek adres ifadesi, genellikle hedef veri olan, bellekteki bir tek verinin adresini belirler (Örneğin ; "INC [B].W" komutu B saklayıcısının gösterdiği word'u bir artırır.)

TABLO 4-1. GENEL ADRESLEME MODLARI

MODLAR	SENTAKS		VERİLER
	BİR ADRESLİ	İKİ ADRESLİ	
Dolaylı B Saklayıcı Adr	[B].B [B].W	A, [B].B A, [B].W	Byte'ın adresi B'dedi Word'ün adresi B'dedi
Dolaylı X Saklayıcı Adr	[X].B [X].W	A, [X].B A, [X].W	Byte'ın adresi X'dedi Word'ün adresi X'dedi
Doğrudan Adresleme	adres.B adres.W	A, adres.B A, adres.W	Byte verilen adrested Word verilen adrested
Dolaylı Adresleme	[pntr].B [pntr].W	A, [pntr].B A, [pntr].W	Byte'ın adresi pntr a word'ün işaret ettiği yerdir. Word'ün adresi pntr a word'ün işaret ettiği yerdir.
Sıralı Adresleme	dis[pnt].B dis[pnt].W	A, dis[pnt].B A, dis[pnt].W	Byte'ın adresi pnt ad word'ün işaret ettiği yerin içerisinden dis kadar uzaktadır. Word'ün adresi pnt ad word'ün işaret ettiği yerin içerisinden dis kadar uzaktadır.
İvedi Adresleme	—	A, #değer	Veri değer'in kendisi
Bellekten Belleğe Doğrudan adresleme	— — — — —	h.B, k.B h.W, k.B h.W, k.W h.B, k.W	Hedef :h adresindeki by Kaynak:k " wo Hedef :h " wo Kaynak:k " by Hedef :h " wo Kaynak:k " wo Hedef :h " by Kaynak:k " wo (Not:Bu mod tavsiye edilmez)
Belleğe İvedi Adresleme	— —	h.B, #değer h.W, #değer	Hedef :h adresindeki by Kaynak:değer'in kendisi Hedef :h adresindeki wo Kaynak:değer'in kendisi
* İvedi veri daima 8 bitlik olmalıdır			

İKİ ADRESLİ GENEL MOD

Bu adresleme modu iki adres kullanan komutların (Bkz. Tablo 4-2) büyük bir kısmında kullanılır. Bu adresleme modu; bellekten belleğe doğrudan adresleme modu ile bellekten belleğe, belleğe ivedi adresleme modu ile ivedi veri bellek, ivedi adresleme modu ile ivedi veri akümülatör işlemlerine izin verir.

ÖZEL MOD

Bu mod, komutların küçük bir kısmıyla sınırlıdır. Bu nedenle, herhangi bir genel mod gibi davranmaz. Bununla beraber, bu mod özellikle veri aktarma komutlarına büyük etkinlik ve güç ekler. Özel modlar LD/LDS (Yükleme) ve X/XS (Değiştirme) komutlarında X veya B saklayıcılarının otomatik artırılması veya azaltılması işlemlerine de sahiptir.

Aşağıdaki satırlarda adres modlarının hiçbirisini kullanmayan bazı komutlardan bahsedilmiştir; bunlar atlamalar, alt programdan geri dönüşler ve sadece akümülatör veya sadece elde biti üzerinde işlem yapan komutlardır. Bu komutlar, adresiz komutlar olarak adlandırılır.

Bir verinin uzunluğu, 8 bitlik byte (.B modu) veya 16 bitlik word (.W modu) olabilir ve bir iki adresli mod kullanıldığı zaman iki verinin farklı uzunlukta olması da mümkündür. (Örneğin "ADD 16.W, 21.B") Word uzunluğundaki veriler belleğin iki byte'ını kullanır ve adresleri daima çift olmalıdır; Byte uzunluğundaki veriler çift veya tek adreslerde tutulabilir.

NOT: Reset de seçilen işlem moduna (BKz. Bölüm 10.1.3) bağlı olarak, Word erişimine yongadaki 0000-001F ve E000-FFFF adres aralığının dışında izin verilir. Ayrıntılı bilgi için Bölüm 10.9'a bakabilirsiniz.

Adres alanının ilk 256 byte'ındaki ("TEMELSAYFA" verileri: 0000- 00FF adreslerindedir.) veriler ile diğer

Sonra : A'da 0105 Hex

F200'daki Word'de 0105 Hex

4.2.4 Dolaylı Adresleme

Bu adres modunu kullanan komutlar 8 bit adres alanını kapsarlar. Bu adres alanı, verinin efektif adresini içeren temelsayfa'daki word'e işaret eder. Word işaretçisinin adresi çift olmalıdır.

Örnek : LD A, [H'1E.W].B Önce : A'da 5555 Hex

001E'deki Word'de 2CFF Hex

2CFF'deki Byte'da 78 Hex

Sonra : A'da 0078 Hex

001E'deki Word'de 2CFF Hex

2CFF'deki Byte'da 78 Hex

4.2.5 Sıralı Adresleme

Sıralı adresleme modu kullanan komutlar 8 bit adres alanı ve 8 bit veya 16 bit kayıklık* alanı içerirler. 8 Bit adres (çift olmalı) tarafından adreslenen temelsayfadaki Word'ün içeriği, verinin efektif adresini elde etmek için kayıklığa eklenir

Örnek : LDA, -2[H'C4.W].W Önce : A'da 0000 Hex

00C4'deki Word'de 01C8 Hex

01C6'daki Word'de 2344 Hex

Sonra : A'da 2344 Hex

00C4'deki Word'de 01C8 Hex

01C6'daki Word'de 2344 Hex

Not: 00C4 yığın göstergesinin bellek haritasındaki adresidir. Bu komut yığının en üstündeki Word'ü akümülatöre kopyalama etkisine

*Kayıklık : Bellekteki bir tablonun herhangi bir elemanına erişmek için tablonun başlangıç adresine eklenen değer (Displacement, offset)

4.4 ÖZEL MODLAR

4.4.1 X Saklayıcısı Kaynaklı Otomatik Artırmalı/Azaltmalı Dolaylı

Bu adresleme modu sadece LD ve X komutlarında kullanılır. Bu mod aslında X saklayıcılı dolaylı modu işletir. Ancak X saklayıcısı, veriye erişildikten sonra bir artırılır veya azaltılır. Bir artırma veya azaltma özelliği programcı tarafından ya [X-] yada [X+] gösterimi ile seçilir. Veri byte uzunluğunda ise X saklayıcısının içeriği bir artırılır; Word uzunluğunda ise iki artırılır. Bu özellikler çevrimlerde bellek blokları üzerinde işlem yapmayı kolaylaştırır.

```
Örnek : LD A,[X+].W      Önce : A'da 0000 Hex  
                                X'de FE24 Hex  
                                FE24'deki Word'de FCFC Hex  
Sonra : A'da FCFC Hex  
                                X'de FE26 Hex  
                                FE24'deki Word'de FCFC Hex
```

```
LD  A,[X-].B      Önce : A'da FFFF  Hex
                        X'de F088  Hex
                        F088'deki Byte'ta 11  Hex
Sonra : A'da 0011  Hex
                        X'de F087  Hex
                        F088'deki Byte'ta 11  Hex
```

4.4.2 B Saklayıcı Kaynaklı Otomatik Artırmalı/Azaltmalı Dolaylı

Bu adresleme modu sadece LDS ve XS komutları ile kullanılır. B saklayıcısı veriyi adreslemek için kullanılır. Veriye erişildikten sonra B saklayıcısının içeriği artırılır veya azaltılır. (Byte veriler için bir, word veriler için iki artırma ve azaltma uygulanır.) Sonra B saklayıcısı, K saklayıcısı ile karşılaştırılır. Eğer artırılmış B saklayıcısı K

saklayıcısından büyükse veya azaltılmış B saklayıcısı K saklayıcısından küçükse, aşağıdaki komut atlanır. Bu adresleme modu veri bloklarının işlem gördüğü yerdeki çevrimlerde kullanılan LDS ve XS komutlarının etkin bir şekilde kullanılmasını sağlar. K saklayıcısı bir sınır ile başlatılır ve bu sınır degere doğru ardarda artırılan veya azaltılan B saklayıcısı sınır değeri geçtiğinde çevrimin dışına çıkılır.

Örnek : LDS A,[B+].B Önce : A'da 5555 Hex

B'de 00F5 Hex

K'da 00F6 Hex

00F5'deki Byte'ta 22 Hex

Sonra : A'da 0022 Hex

B'de 00F6 Hex

K'da 00F6 Hex

00F5'deki Byte'ta 22 Hex

Bir sonraki komut atlanmaz, çünkü B'nin içeriği K'nın içeriğini geçmedi.

LDS A,[B+].W Önce : A'da 0000 Hex

B'de 00F6 Hex

K'da 00F6 Hex

00F6'daki Word'de FDDF Hex

Sonra : A'da FDDF Hex

B'de 00F8 Hex

K'da 00F6 Hex

00F6'daki Word'de FDDF Hex

Bir sonraki komut atlanır çünkü B'nin içeriği K'nın içeriğinden daha büyüktür.

LDS A,[B-].W Önce : A'da 0000 Hex

B'de FFE8 Hex

K'da FFE7 Hex

FFE8'deki Word'de 3300 Hex

Sonra : A'da 3300 Hex

B'de FFE6 Hex

K'da FFE7 Hex

FFE8'deki Word'de 3300 Hex

Bir sonraki komut atlanır,

çünkü B'nin içeriği K'nın

içeriginden daha küçüktür.

4.4.3 Çift Saklayıcılı Dolaylı Adresleme

Bu adresleme modu, sadece RBIT, SBIT, ve IFBIT komutlarında kullanılan saklayıcı kaynaklı dolaylı adreslemenin özel bir durumudur. Bu modda, B saklayıcısı ve (X saklayıcısı)+8, bellekteki byte'ın adresini elde etmek için toplanır. Test edilecek veya değiştirilecek belirli bit, X saklayıcısının en düşük anlamlı üç biti tarafından adreslenir. Böylece, sadece X'in arttırılmasıyla, Başlangıç adresi B'de tutulan herhangi bir bellek gözünden 64k bit'e kadar bit dizilerini adresleyen çevrimler yapılabilir.

Örnek : SBIT X,[B].B

Önce : B'de F000 Hex

X'de 0802 Hex

F100'deki Byte'ta 0000 Hex

Sonra : B'de F000 Hex

X'de 0802 Hex

F100'deki Byte'ta 0004 Hex

RBIT X,[B].B

Önce : B'de F000 Hex

X'de 0002 Hex

F000'deki Byte'ta 0564 Hex

Sonra : B'de F000 Hex

X'de 0002 Hex

F000'deki Byte'ta 0560 Hex

5. BÖLÜM

KOMUT SETİ

5.1 GİRİŞ

Bu bölüm HPC ailesinin komut setini açıklamaktadır. Her komutun adı, sentaksı, işlevi ve tanımı belirtilmiştir.

5.2 KOMUTLAR

ADC Elde ile Topla

Sentaks : [etiket:] ADC hedef, kaynak

İşlem : hedef $\leftarrow$ hedef + kaynak + C^{*}
C $\leftarrow$ elde^{**}.

Tanım : Kaynak ve hedef verileri toplanır, sonuc hedef veriye yerleştirilir. Elde biti bu sonuca eklenir: Eger elde biti 1 ise sonuç verilerin toplamından bir fazla, elde biti 0 ise sonuç verilerin toplamıdır. Elde biti toplanmadan gelen elde ile yüklenir.

ADD Topla

Sentaks : [etiket:] ADD hedef, kaynak

İşlem : hedef $\leftarrow$ hedef + kaynak
C $\leftarrow$ elde

* C : Elde bitini gösterir.

** elde : Herhangi bir işlemden sonra ortaya çıkan eldeyi gösterir

Tanım : Kaynak ve hedef verileri toplanır, sonuç hedef veriye yerleştirilir.
Sentaks : Elde biti toplanmadan gelen elde ile
İşlem : yüklenir.

ADDS Kısa İvedi Veri ile Topla
Sentaks : [etiket:] ADDS A, #değer
İşlem : $A \leftarrow A + \text{değer}$
Tanım : Bu komut 8 bitlik ivedi veri ile akümülatörün içeriğini toplar. İvedi veri sıfır genişletme ile 16 bitlik hale getirilir. Bununla beraber ADD komutunun ivedi adresleme modundan farklı olarak elde biti 15. bit eldesinden çok 7. bit eldesini gösterir.

AND Lojik VE
Sentaks : [etiket:] AND hedef, kaynak
İşlem : $\text{hedef} \leftarrow \text{hedef} \text{ VE } \text{kaynak}$
Tanım : Hedef veri, kaynak veri ile lojik olarak VE lenir ve sonuç hedef veriye yerleştirilir. Eğer kaynak byte, hedef word uzunluğunda ise, hedefin yüksek anlamlı byte'ı sıfırlanır.

CLR A Akümülatörü Sıfırla
Sentaks : [etiket:] CLR A
İşlem : $A \leftarrow 0$
Tanım : Akümülatörün bütün bitleri sıfırlanır.

COMP A Akümülatörü Tümler

DIV Bölme

Sentaks : [etiket:] COMP A

İşlem : $A \leftarrow \bar{A}$

Tanım : Akümülatörün bütün bitleri tümlenir. Yani değeri 1 olan bütün bitler 0, değeri 0 olan bütün bitler 1 değerini alır.

Bu komutu izleyen bir INC A komutu, A'nın önceki değerinin ikiye tümlenmişini elde etmeği sağlar.

DADC Eldeli Desimal Toplama

Sentaks : [etiket:] DADC hedef,kaynak

İşlem : hedef $\leftarrow$ hedef + kaynak + C
C $\leftarrow$ elde

Tanım : Kaynak ve hedef verileri BCD değerler olarak toplanır ve BCD düzenindeki sonuç hedef veriye yerleştirilir. Eğer veriler geçerli BCD değerlere sahip degillerse, bu komut tarafından üretilen sonuç ve elde tanımsızdır.

DEC A Akümülatörü Bir Azalt

Sentaks : [etiket:] DEC A

İşlem : $A \leftarrow A - 1$

Tanım : Akümülatörün içeriği bir azaltılır.

DECSZ Bir Azalt ve Sonuç Sıfırsa Komut Atla

Sentaks : [etiket:] DECSZ hedef

İşlem : hedef $\leftarrow$ hedef - 1

hedef=0 ise sonraki komutu atla

Tanım : Hedef veri bir azaltılır. Sonuç

sıfırsa, bir sonraki komut atlanır.

DIV

Bölme

Sentaks : [etiket:] DIV hedef, kaynak

İşlem : hedef $\leftarrow$ hedef + kaynak

X $\leftarrow$ kalan

C $\leftarrow$ 0, K $\leftarrow$ 0

Tanım : Hedef veri, kaynak veriye bölünür ve sonuç hedef veriye yerleştirilir. Kalan X saklayıcısına yerleştirilir. K saklayıcısı ve elde biti bu komut tarafından sıfırlanır. Her iki veri de işaretli sayı olarak yorumlanır. Eğer sıfıra bölme durumu ile karşılaşırsa, hedef verinin bütün bitleri 1 değeri alır, X saklayıcısı hedef verinin orjinal içeriğini alır, bölen değişmeden kalır, elde biti birleşir ve K saklayıcısı sıfırlanır.

DIVD

Çift Word ile Bölme

Sentaks : [etiket:] DIVD hedef, kaynak

İşlem : hedef $\leftarrow$ (X, hedef) + kaynak

C $\leftarrow$ Taşma bayrağı, K $\leftarrow$ 0

Tanım : Bölünecek sayı 32 bitliktir. Bu 32 bitin yüksek anlamlı 16 biti X saklayıcısında, düşük anlamlı 16 biti hedef veride bulunur. Bölen kaynak veridedir. Bölüm hedef veriye yerleştirilir. Kalan X saklayıcısına yerleştirilir. Yapılan işlem

IFB1T Bit Testi

Sentaks

İşlem

Tanım

dogruysa elde biti ve K saklayıcısının her ikisi de sıfırlanır. Her iki veri işaretsiz olarak yorumlanır ve 16 bit degerler kullanılmadan önce sıfır genişletme işlemine tabi tutulur. X saklayıcısı bu komut yürütülmeden önce bölünecek sayının üst 16 bitini ihtiva etmelidir. Eger sıfıra bölmeyle yada bölümün 16 bitten daha uzun olduğu durumlarla karşılaşırsa, elde biti birleşir ve X saklayıcısı, hedef ve kaynak verileri değişmez. K saklayıcısı anlamsız bir deger alır.

DSUBC

Eldeli Desimal Çıkarma

Sentaks : [etiket:] DSUBC hedef, kaynak

İşlem : hedef $\leftarrow$ hedef - kaynak - $\bar{C}$

C $\leftarrow$ elde

Tanım : Kaynak ve hedef veriler BCD olarak çıkarılır. BCD sonuç hedef veriyeyerleştirilir. Elde biti bu sonuca eklenir: Eger 0 ise sonuç verilerin farkından bir eksiktir; 1 ise sonuç verilerin tam farkıdır. Elde biti çıkarmadan gelen elde ile yüklenir. Veriler geçerli BCD degerlere sahip degillerse bu komut tarafından üretilen sonuç ve elde tanımsızdır.

IFBIT Bit Testi

Sentak : [etiket:] IFBIT n,kaynak

[etiket:] IFBIT X,[B].B

İşlem : Kaynağın n.bit=0 ise bir sonraki komutu atla.

Tanım : Bu komut bellekteki bir biti test

eder.Eğer bit 1 değerindeyse bir

sonraki komut yürütülür, 0

değerindeyse bir sonraki komut

atlanır ve atlanılan komuttan

sonraki yürütülür.Yukarıdaki

birinci sentaksta "n" 0-7 sınırları

arasında bir sabittir.Erişilen

bit,byte düzeyindeki kaynak verinin

"n". bitidir.Yukarıdaki ikinci

İşlem : sentaksta Byte adresi ve bit yeri

Tanım : aşağıdaki gibi hesaplanır:

Adres = B + (X + 8)

Bit no = Kalan(X + 8)

B ve X ,B ve X saklayıcılarının
içerigini gösterir.

IFC Elde=1 ise

Sentak : [etiket:] IFC

İşlem : C=1 ise bir sonraki komutu yürüt.

Tanım : Elde bitinin değeri 1 ise,IFC

komutunu takip eden komut

yürütülür. Değilse atlanır.

IFEQ Eşit ise

Sentak : [etiket:] IFEQ kaynak1,kaynak2

İşlem : kaynak1 ≠ kaynak2 ise bir sonraki
komutu atla.

Tanım : İki veri karşılaştırılır.Eşit
degiller ise takip eden komut
atlanır.

IFGT Daha Büyük ise

Sentaks : [etiket:] IFGT kaynak1,kaynak2

İşlem : Kaynak1 ≤ kaynak2 ise bir sonraki
komutu atla.

Tanım : İki veri işaretliiz olarak
karşılaştırılır.Kaynak2 verisi
kaynak1 verisinden daha büyük ise
bir sonraki komut atlanır.

IFNC Elde=0 ise

Sentaks : [etiket:] IFNC

İşlem : C=0 ise bir sonraki komutu yürüt.

Tanım : Elde biti sıfırsa IFNC yi takip
eden komut yürütölür, değilse
atlanır.

INC Bir Artır

Sentaks : [etiket:] INC hedef

İşlem : hedef ← hedef + 1

Tanım : Hedef verisi bir artırılır.

INC A Akümülatörü Bir Artır

Sentaks : [etiket:] INC A

İşlem : A ← A + 1

Tanım : Akümülatörün içeriği bir artırılır.

JID Dolaylı Dallen

Sentaks : [etiket:] JID

[etiket:] .PT hedef0,hedef1,hedef2...

İşlem : $PC^* \leftarrow PC + \text{Akümülatörün içeriği}$
tarafından seçilen dallanma tablosu

İşlem : elemanının içeriği

Tanım : Bu tek byte'lık komut bir çok yönlü dallanma yürütür. (C programlama dilindeki "switch" komutuna benzer olarak.) Bu komutunu, bir dallanma tablosunu belirten .PT assembler direktifi takip eder. Tablonun her bir elemanı 8 bit uzunluğundadır dallanma için ayrı bir yeri işaret eder. Dallanma tablosunun hangi elemanının seçileceği, akümülatörün içeriği tarafından belirlenir. Yani akümülatörün içeriği sıfır ise ilk eleman, bir ise ikinci eleman, iki ise üçüncü eleman seçilir. Her bir tablo elemanı iki işaretsiz bir kayıklık değere sahiptir. Bu kayıklık dallanılacak komutun adresi ile kendi adresi arasındaki uzaklığı içerir. Tablo elemanları işaretsiz olduğundan, JID komutu sadece ileriye doğru dallanma yapmaya izin verir. Ayrıca dallanılacak komutlar en çok 255 byte ileriye yerleştirilebilir.

JIDW Dolaylı Dallan (Word)

Sentaks : .ODD

[etiket:] JIDW

*PC : PC (Program sayacı) saklayıcısının içeriği

[etiket:] .PTW hedef0,hedef1,hedef2..

İşlem : $PC \leftarrow PC + \text{Akümülatörün içeriği}$
tarafından seçilen dallanma tablosu
elemanının içeriği

Tanım : Bu tek byte'lık komut JID komutu
gibi çok yönlü bir dallanma işlemi
yürütür.JID komutundan farkı
dallanma tablosu içeriklerinin 16
bitlik olmasıdır.Bu nedenle JIDW
komutundan sonra .PTW assembler
direktifi bulunmaktadır.Tablo
elemanlarının akümülatör tarafından
seçiminde de bir fark
vardır.Akümlatörün içeriği çift
sayı olmalıdır.Bu durumda
akümülatörün değeri sıfırsa birinci
eleman,iki ise ikinci eleman ,dört
ise üçüncü eleman gibi tablo
elemanlarına giriş yapılır.Uygun
akümülatör içeriği, istenilen
değerin akümülatöre girilmesi ve
akümülatör içeriğinin bir bit sola
ötelenmesiyle elde edilir.Herbir
tablo elemanı işaretli (ikiye
tümlemiş) bir kayıklıktan
oluşturur.Bu kayıklık dallanılacak
komutun ile kendi yerinin adresi
arasındaki uzaklığa sahiptir.

NOT : Tablo elemanları word
uzunluğunda olduğundan,bunlar
sadece çift adreslere
yerleştirilebilirler. Bu nedenle
tek byte'lık JIDW komutubir tek

adrese yerleştirilmelidir. Buyüzden JIDW komutundan önce .ODD assembler direktifi kullanılmıştır.

JMP Bağımlı Dallen

Sentaks : [etiket:] JMP hedef

İşlem : $PC \leftarrow PC + \text{hedef'e göre kayıklık}$ (8 bitlik)

Tanım : Bu iki byte'lık komut, JMP komutunun ilk byte'ından -255 - +257 byte uzaklıktaki yerlere dallanma yapar.

JMPL Bağımlı Dallen (Uzun)

Sentaks : [etiket:] JMPL hedef

İşlem : $PC \leftarrow PC + \text{hedef'e göre kayıklık}$ (16 bitlik)

Tanım : Bu üç byte'lık komut HPC'nin 16 bitlik adres alanındaki herhangi bir yere dallanmak için kullanılır.

JP Bağımlı Dallen (Kısa)

Sentaks : [etiket:] JP hedef

İşlem : $PC \leftarrow PC + \text{hedef'e göre kayıklık}$

Tanım : Bu tek byte'lık komut, JP komutundan -31 - + 32 byte uzaklıktaki yerlere dallanma yapar.

JSR Altprograma Dallen

Sentaks : [etiket:] JSR hedef

İşlem : $SP.W^* \leftarrow \text{Bir sonraki komutun adresi}$

$SP \leftarrow SP + 2$

$PC \leftarrow PC + \text{hedef'e göre kayıklık}$

LD

Yukle

Tanım : Bu iki byte'lık komut, JSR komutunun ilk byte'ından -1023 - +1025 byte uzaklıktaki hedef tarafından belirlenen altprograma dallanmayı sağlar.

Sentaks : [etiket:] LD-A, [X-].B

JSRL Altprograma Dallan (Uzun)

Sentaks : [etiket:] JSRL hedef

İşlem : SP.W ← Bir sonraki komutun adresi

İşlem : SP ← SP + 2

PC ← PC + hedefe göre kayıklık

Tanım : Bu üç bytelik komut, HPC'nin 16 bitlik adres alanındaki herhangi bir yerdeki altprograma dallanmayı sağlar.

[etiket:] LD X, #değer

(JSRP) - Tablodan Altprograma Dallan

Sentaks : .SPT hedef

[etiket:] JSR hedef

yada

[etiket:] JSRL hedef

İşlem : SP.W ← Bir sonraki komutun adresi

SP ← SP + 2

PC ← PC + Komut tarafından seçilen altprogram tablosu elemanı

Tanım : "Bu tek bytelik komut, ".spt hedef" assembler direktifi ile tanımlanan altprogram tablosundaki elemanın gösterdiği altprograma dallanmayı sağlar.

*SP.W : SP (Yığın Göstergesi) saklayıcısının işaret ettiği Word

LD

Yükle

Sentaks : [etiket:] LD hedef,kaynak

İşlem : hedef $\leftarrow$ kaynak

Sentaks : [etiket:] LD A,[X+].B

[etiket:] LD A,[X+].W

[etiket:] LD A,[X-].B

[etiket:] LD A,[X-].W

İşlem : A $\leftarrow$ X tarafından işaret edilen
değer

X $\leftarrow$ X + 1 veya 0

Sentaks : [etiket:] LD B,#değer

[etiket:] LD X,#değer

[etiket:] LD K,#değer

[etiket:] LD BK,#değer

İşlem : hedef $\leftarrow$ kaynak

Tanım : Hedef veri kaynak veriden yüklenir.
Hedef veri kaynak veriden uzunsa,
kaynak veri sıfır genişletme ile
uygun hale getirilir.
Saklayıcılara, bellek haritasındaki
adresleri kullanılarak yükleme
yapılabilir.

LDS Otomatik Artırmalı/Azaltmalı ve Durumsal Atlamalı A'ya
Yükleme

Sentaks : [etiket:] LDS [B+].B

[etiket:] LDS [B+].W

[etiket:] LDS [B-].B

[etiket:] LDS [B-].W

İşlem : $A \leftarrow \text{Veri (B'nin işaret ettiği)}$

Tanım : $B \leftarrow B \pm \text{Veri genişliği (1 veya 2)}$

gereklilik Artırılmış / Azaltılmış B'nin içeriği, K'daki adresi geçerse, bir sonraki komut atlanır.

Tanım : B saklayıcısı tarafından adreslenen yerin içeriği akümülatöre yüklenir. Sonra B saklayıcısı, verideki byte sayısı kadar otomatik olarak artırılır veya azaltılır. ("+" veya "-" tarafından seçilerek.) B'nin artırılmasından sonra sonuç K saklayıcısındaki değerden daha büyük olursa, bir sonraki komut atlanır. B'nin azaltılmasından sonra sonuç K saklayıcısındaki değerden daha küçük olursa, bir sonraki komut atlanır.

Not : HPC bir işaretsiz makine olduğundan, FFFF (Hex) 'den daha büyük ve 0 'dan daha küçük değer düşünülmez; bu nedenle komut atlamayı yasaklamak için K saklayıcısına bu değerlerden biri yazılabilir. Bu komut çevrimlerde kullanılır. Çevrimler belleğin ilk ve son byte'larını (yada word) gerektirdiğinden, bu komut dikkatlice kullanılmalıdır.

MULT Çarpma

Sentaks : [etiket:] MULT hedef, kaynak

İşlem : $X \text{ ve hedef} \leftarrow \text{hedef} * \text{kaynak}$

$K \leftarrow 0, C \leftarrow 0$

Tanım : Kaynak ve hedef verileri okunur ve gerekliyse verileri 16 bit yapmak için sıfır genişletme yapılır. Sonra bu veriler işaretli olarak çarpılır ve 32 bitlik sonucun yüksek anlamlı 16 biti X saklayıcısına, alçak, anlamlı 16 biti hedef veriye yerleştirilir. K saklayıcısı ve elde biti sıfırlanır. Hedef veri bir byte uzunluğunda olduğu zaman sonucun 8-15. bitleri kaybolur.

NOP İşlem Yok

Sentaks : [etiket:] NOP

İşlem : $PC \leftarrow PC + 1$

Tanım : Bu tek byte'lık komut bir işlem yürütmeksizin bir sonraki komuta geçiş yapmayı sağlar. "JP .+1" komutu ile aynı etkiye sahiptir.

OR Lojik Veya

Sentaks : [etiket:] OR hedef, kaynak

İşlem : $\text{hedef} \leftarrow \text{hedef VEYA kaynak}$

Tanım : Bu komut hedef ve kaynak verilerinin bitleri arasında, bit düzeyinde lojik veya işlemi yapılmasını sağlar. Sonuç hedef veriye yerleştirilir.

POP Yığından Deger Çekme

Sentaks : [etiket:] POP hedef

İşlem : $SP \leftarrow SP - 2$
hedef $\leftarrow$ SP'nin işaret ettiği Word
Tanım : SP (Yığın Göstergesi) saklayıcısı
iki azaltılır. Sonra hedef veri, SP
saklayıcısının içeriği tarafından
adreslenen bellek yerinden
yüklenir. Veri doğrudan adresleme
modu kullanılarak adreslenir ve
bellegin ilk 256 byte'ındaki bir
word olmalıdır.

PUSH Yığına Deger At

Sentak : [etiket:] PUSH kaynak

İşlem : SP'nin işaret ettiği Word $\leftarrow$ kaynak
 $SP \leftarrow SP + 2$

Tanım : Kaynak verinin içeriği, SP
saklayıcısının içeriğinin
adreslediği bellek gözüne
yerleştirilir. Sonra SP saklayıcısı
iki artırılır. Veri doğrudan
adresleme modu kullanılarak
adreslenir ve bellegin ilk 256
byte'ındaki bir word olmalıdır.

RBIT Biti Sıfırla.

Sentak : [etiket:] RBIT n, kaynak

[etiket:] RBIT X, [B].B

İşlem : hedefin n. biti $\leftarrow$ 0

Tanım : Bu komut bellekteki bir biti
sıfırlar. Yukarıdaki birinci
sentakta n 0-7 sınırları arasında
bir sabittir. Hedefin (Byte
düzeyinde) n. yerindeki bite

erişilir.İkinci sentakata,IFBIT, RBIT ve SBIT komutlarında kullanılabıliren özel bir adresleme kullanılmıştır.B saklayıcısı bellekteki bir byte'a işaret eder. X saklayıcısı 0-65535 sınırları arasındaki bit offsetini içerir.Byte adresi ve bit yeri şu şekilde hesaplanır:
Adres = B + (X + 8)
Bit no. = Kalan (X + 8)
B ve X ,B ve X saklayıcılarının içeriklerini göstermektedir.

RC

Eldeyi Sıfırla

Sentak : [etiket:] RC

İşle : C $\leftarrow$ 0

Tanı : Elde biti sıfırlanır.

RET

Altprogramdan Geri Dönüş

Sentak : [etiket:] RET

İşle : SP $\leftarrow$ SP - 2

PC $\leftarrow$ SP'nin gösterdiği Word

Tanı : SP saklayıcısı iki azaltılır. Sonra program sayacı,SP saklayıcısının içeriği tarafından adreslenen word düzeyindeki bellekten yüklenir ve program bu noktadan itibaren çalışmaya başlar.

RETI

Kesmeden Geri Dönüş

Sentak : [etiket:] RETI

İşle : SP $\leftarrow$ SP - 2

PC $\leftarrow$ SP'nin gösterdiği Word

ENIR* 'daki GIE** biti $\leftarrow$ 1

Tanım : SP saklayıcısı iki azaltılır. Sonra
işlem : program sayacı SP saklayıcısının
içerisinin adreslediği word
düzeyindeki bellekten yüklenir.
Kesme izin saklayıcısındaki (ENIR)
maskeleye biti (GIE) birleşir
(Maskelenebilir kesmelere izin
verilir.) ve program, program
sayacının yeni içeriğince
belirlenen adresten yürütülmeye
başlanır.

RETSK Altprogramdan Geri Dönüş ve Komut Atlama

Sentaks : [etiket:] RETSK

İşlem : SP $\leftarrow$ SP - 2

Tanım : PC $\leftarrow$ SP'nin gösterdiği Word

PC'nin işaret ettiği komutu atla

Tanım : SP saklayıcısı iki azaltılır. Sonra
program sayacı SP'nin içeriğinin
adreslediği word düzeyindeki
bellekten yüklenir. PC'nin
gösterdiği komut atlanır ve bu
komutu takip eden ilk komut
yürütülür.

* ENIR : Kesme izin saklayıcısı

** GIE : Maske biti (Maskelenebilir kesmeleri yasaklamak için)

RLC A Akümülatörü Elde Üzerinden Sola Döndür

Sentaks : [etiket:] RLC A

İşlem : $C \leftarrow A[15] \leftarrow \dots \leftarrow A[0] \leftarrow C$

Tanım : Akümülatörün içeriği bir bit sola ötelenir. Akümülatörün en düşük anlamlı biti elde bitinden yüklenir. Elde biti de akümülatörün en yüksek anlamlı bitinden yüklenir.

Örneğin :

Önce : 0011 1111 0000 0000 C=1 ;

Sonra: 0111 1110 0000 0001 C=0

RRC A Akümülatörü Elde Üzerinden Sağa Döndür

Sentaks : [etiket:] RRC A

İşlem : $C \rightarrow A[15] \rightarrow \dots \rightarrow A[0] \rightarrow C$

Tanım : Akümülatörün içeriği bir bit sağa ötelenir. Akümülatörün en yüksek anlamlı byte'ı elde bitinden yüklenir. Elde biti de akümülatörün en düşük anlamlı bitinden yüklenir.

Örneğin :

Önce : 0011 1111 0011 1111 C = 0

Sonra: 0001 1111 1001 1111 C = 1

SBIT Biti Birle

Sentaks : [etiket:] SBIT n, hedef

[etiket:] SBIT X, [B].B

İşlem : hedefin n. biti $\leftarrow 1$

Tanım : Bu komut bellekteki bir biti birler Yukarıdaki birinci sentaksta n 0-7 sınırları arasında bir sabittir. Hedefin (Byte düzeyinde)

n. yerindeki bite erişilir. İkinci sentaksta, IFBIT, RBIT ve SBIT komutlarında kullanılabılina özel bir adresleme kullanılmıştır. B saklayıcısı bellekteki bir byte'a işaret eder. X saklayıcısı 0-65535 sınırları arasındaki bit offsetini içerir. Byte adresi ve bit yeri şu şekilde hesaplanır:

$$\text{Adres} = B + (X + 8)$$

$$\text{Bit no.} = \text{Kalan} (X + 8)$$

B ve X, B ve X saklayıcılarının içeriklerini göstermektedir.

SC Eldeyi Birle

Sentak : [etiket:] SC

İşlem : $C \leftarrow 1$

Tanım : Elde biti birlenir.

SHL A Akümülatörü Sola Ötele

Sentak : [etiket:] SHL A

İşlem : $C \leftarrow A[15] \leftarrow \dots \leftarrow A[0] \leftarrow 0$

Tanım : Akümülatörün içeriği bir bit sola ötelenir. Akümülatörün en düşük anlamlı biti sıfırlanır ve elde biti akümülatörün en yüksek anlamlı bitinden yüklenir.

Örneğin:

Önce : 0011 1111 0000 0000 C = 1/0

Sonra: 0111 1110 0000 0000 C = 0

SHR A Akümülatörü Sağa Ötele

Sentak : [etiket:] SHR A

İşlem : $0 \rightarrow A[15] \rightarrow \dots \rightarrow A[0] \rightarrow C$
Tanım : Akümülatörün içeriği bir bit sağa
ötelenir. Akümülatörün en anlamlı
biti sıfırlanır ve elde biti
akümülatörün en düşük anlamlı
bitinden yüklenir.

Örneğin:

Önce : 0011 1111 0011 1111 C $\leftarrow$ 1/0

Sonra: 0001 1111 1001 1111 C $\leftarrow$ 1

ST Akümülatörü Sakla

Sentaks : [etiket:] ST A, hedef

İşlem : $A \rightarrow \text{hedef}$

Tanım : Akümülatördeki değer hedef veriye
saklanır.

SUBC Eldeli Çıkarma

Sentaks : [etiket:] SUBC hedef, kaynak

İşlem : $\text{hedef} \leftarrow \text{hedef} - \text{kaynak} - \bar{C}$
 $C \leftarrow \text{elde}$

Tanım : Kaynak ve hedef veriler çıkarılır
ve sonuç hedef veriye
yerleştirilir. Elde biti bu sonuç
ile birleştirilir; elde biti bir
ise sonuç verilerin gerçek
farkıdır, sıfır ise sonuç verilerin
farkından bir eksiktir. Elde biti
çıkarmadan gelen elde ile yüklenir.

SWAP A Akümülatör'ün Bit Gruplarını Birbiri ile Değiştir

Sentaks : [etiket:] SWAP A

İşlem : $A[11-15] \leftrightarrow A[8-11] \leftrightarrow A[4-7] \leftrightarrow A[0-3]$

Tanım : Akümülatör'ün dört bitlik grupları

sole  telenir ve d    k anlamlı d rt
bit, 4-7. bitlerle y  klenir.

 rne in:

 nce : 0101 0000 1111 1100

Sonra: 0000 1111 1100 1111

X Ak    lat r ile De i tir.

Sentaks : [etiket:] X A, hedef

     : $A \leftarrow \rightarrow \text{hedef}$

Sentaks : [etiket:] X A, [X+].B

[etiket:] X A, [X+].W

[etiket:] X A, [X-].B

[etiket:] X A, [X-].W

     : $A \leftarrow \rightarrow X$ 'in g    rdi i yer

$X \leftarrow X \pm (1 \text{ veya } 2)$

Tanı  : Ak    lat r n i  eri i hedef veriyle de i tirilir. Ek olarak bir adresli genel adresleme modunda, X komutu, X saklayıcısının otomatik artırma/azaltma ile dolaylı adreslemede kullanılmasına izin verir.

XOR Dar Veya

Sentaks : [etiket:] XOR hedef, kaynak

     : $\text{hedef} \leftarrow \text{hedef} \oplus \text{kaynak}$

Tanı  : Bu komut, hedef ve kaynak verilerinin bitleri arasında, bit d zeyinde dar veya     me yapmayı sa lar.

XS Otomatik Artırmalı/Azaltmalı ve Durumsal Atlamalı A ile De i tirme

Sentak : [etiket:] XS [B+].B

[etiket:] XS [B+].W

[etiket:] XS [B-].B

[etiket:] XS [B-].W

İşlem : $A \leftarrow B$ 'nin gösterdiği veri

$B \leftarrow B +$ Verinin uzunluğu (1 veya Artırılmış veya azaltılmış B'n içeriği K'daki adresi geçerse, b sonraki komutu atla

Tanım : B'nin adreslediği yerdeki içeriği akümülatörün içeriği ile değiştir

Sonra B saklayıcısı verini

uzunluğuna bağlı olarak 1 veya

artırılır. B artırıldıktan sonra

sonuç K saklayıcısındaki değerde

daha büyük olursa bir sonraki komut

atlanır. B azaltıldıktan sonra

sonuç K saklayıcısındaki değerde

daha küçük olursa bir sonraki komut

atlanır.

Not : HPC bir işaretsiz makine

olduğundan, FFFF (Hex) 'den daha

büyük ve 0 'den daha küçük değere

düşünülmez; bu nedenle komut

atlamayı yasaklamak için K

saklayıcısına bu değerlerden biri

yazılabilir. Bu komut çevrimlerde

kullanılır. Çevrimler belleğin ilk

ve son byte'larını (yada word)

gerektirdiğinden, bu komut

dikkatlice kullanılmalıdır.

6. BÖLÜM

FONKSİYON PINLERİNİN TANITIMI

MOS transistörlerle tasarlanan HPC , elektrostatik yüklerin zararlarına karşı korunmuştur . Bütün pinlerde , diyod koruması vardır . HPC , kullanıcı ile iletişimini 68 pin ile yapar . (bkz . şekil 6-1)

G/Ç PINLERİ :

AO - A15 : Port A

BO - B15 : Port B

IO - I7 : Port I

DO - D7 : Port D

PO - P3 : Port P

BESLEME KAYNAĞI PINLERİ :

VCC1 : Pozitif kaynak pini

VCC2 : " " "

GND : Yonga lojigi için referans

DGND : Çıkış pinlerine bağlanan yonga sürücüler için referans

VCC1 ve VCC2 pinleri , yonga içerisinde bağlanmışlardır , fakat GND ve DGND pinleri elektriksel olarak yalıtılmışlardır .

SAAT PINLERİ :

CKI : Osilator girişi

CKO : Osilator çıkışı

CK2 : Saat çıkışı (CK2=CKI/2)

CKI ve CKO pinleri , genelde bir dış kristale bağlanırlar .

DİĞER PINLER :

$\overline{WO}$: Watchdog lojik pini

ST1 : Yeni komutun ilk işlem kodunu almak için okuma çevriminin başlatıldığını gösterir .

ST2 : ALE işareti ile birlikte komutun durumunu gösterir .

$\overline{RESET}$: Yonganın başlangıç durumuna (reset) gelmesi için aktif lojik 0 giriştir .

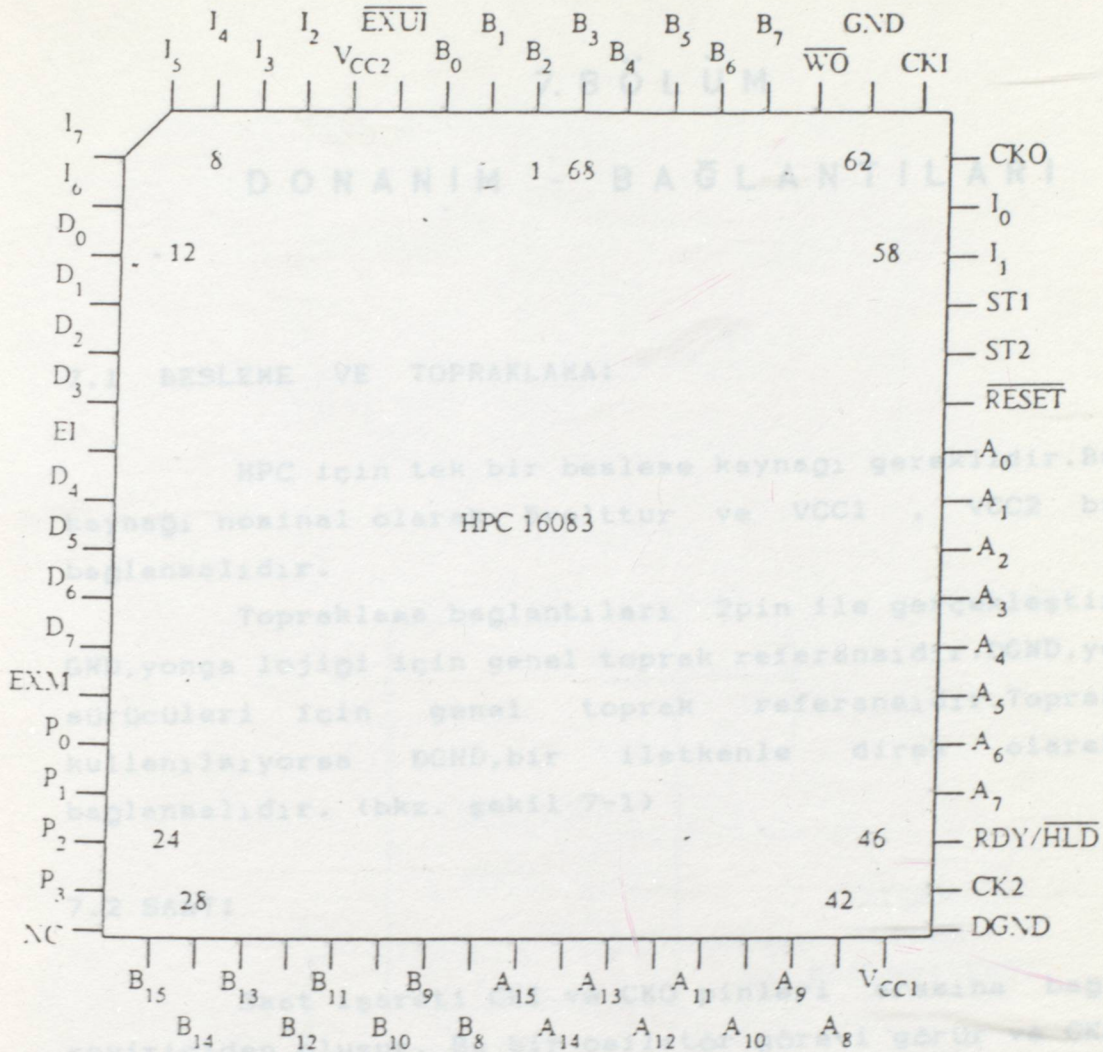
RDY/ $\overline{HLD}$: 2 şekilde kullanılır . ENIR saklayıcısının 0. biti tarafından seçilir . READY girişi , yavaş belleklerin yol çevrimi genişletmekte kullanılır . $\overline{HOLD}$ istek girişi , DMA amaçları için yolu yüksek empedans durumuna getirir .

EXM : Dış bellek izin pini . (lojik 1 ' de aktif) . Bu pin , yongadaki ROM ve haritaları yasaklar. ROMsuz modda dış bellek yerleştirmede kullanılır .

EI : Dış maskelenebilir kesme giriş pini . 1/0 seviye duyarlıklılı veya çıkan/inen kenar duyarlıklılı olarak programlanabilir . EI , EICR saklayıcısının 4.biti için giriş olarak kullanılır .

$\overline{EXUI}$: Dış UART kesme giriş pini . Seviye duyarlıklılı ve bir iç pull-up devresiyle lojik 0'da aktif hale gelir .

NC : Bağlantı olmayan (boş) pin .



7.3 RESET (SIFIRLANA):

RESET pin'i yonga içişigini başlangıca (ilk duruma) getirir. Yani yonga içişisindeki birimler (sayıcı, sayıcı v.a.) sıfırlanır. CK2 saatin son 16 periyodu boyunca RESET pinini SEKİL 6-1. HPC 'nin Pin Çıkış Diyagramı

Yongadaki cihazlar devresinin gerekli olmasını sağlayarak içinebaşlangıca kaynağının yongasına zamanının bug kaba kadar bir süre RESET sıfırda tutulmalıdır. Power-on reset üretmek için

7. BÖLÜM

DONANIM - BAĞLANTILARI

7.1 BESLEME VE TOPRAKLAMA:

HPC için tek bir besleme kaynağı gereklidir. Bu besleme kaynağı nominal olarak 5 voltur ve VCC1 , VCC2 bu kaynağa bağlanmalıdır.

Topraklama bağlantıları 2pin ile gerçekleştirilmiştir. GND, yonga lojigi için genel toprak referansıdır. DGND, yonga çıkış sürücüleri için genel toprak referansıdır. Toprak yüzeyi kullanılmıyorsa DGND, bir iletkenle direk olarak GND'a bağlanmalıdır. (bkz. şekil 7-1)

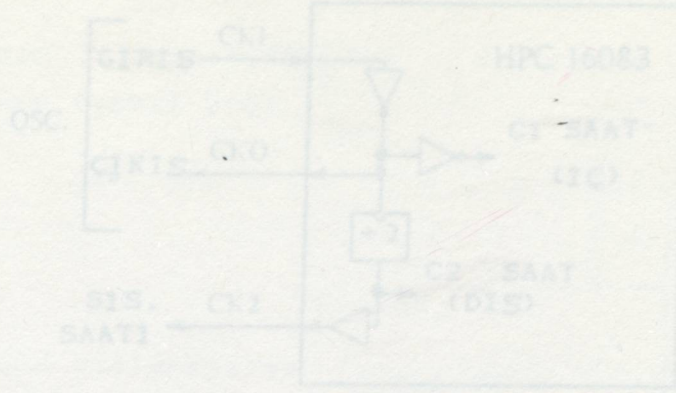
7.2 SAAT:

Saat işareti CKI ve CKO pinleri arasına bağlanan faz çeviriciden oluşur. Bu bir osilatör görevi görür ve CK2 pininden (clock/2) 2'ye bölünmüş olarak sağlanır. (bkz. şekil 7-2 , şekil 7-3 ve şekil 7-4)

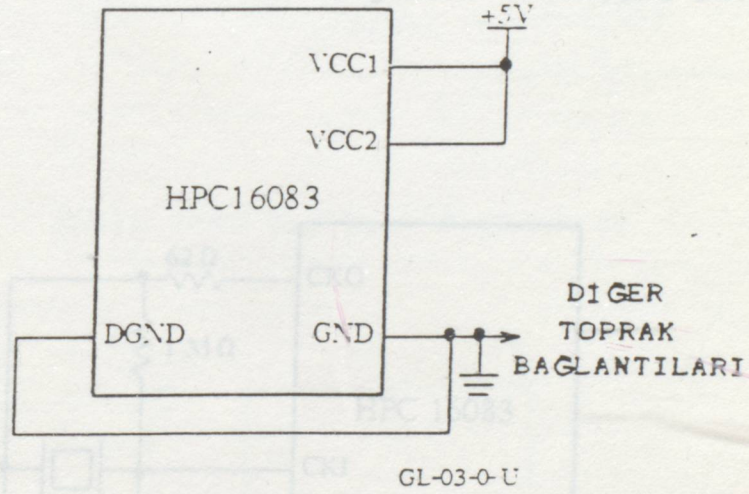
7.3 RESET (SIFIRLAMA):

RESET pini yonga lojigini başlangıca (ilk duruma) getirir. Yani yonga içerisindeki birimler (saklıyıcı, sayıcı v.s) sıfırlanır. CK2 saatin son 16 periyodu boyunca RESET pininin sıfırda kalması HPC'yi sıfırlayacaktır.

Yongadaki osilatör devresinin kararlı olmasını sağlamak için besleme kaynağının yükselme zamanının beş katı kadar bir süre RESET sıfırda tutulmalıdır. Power-on reset üretmek için

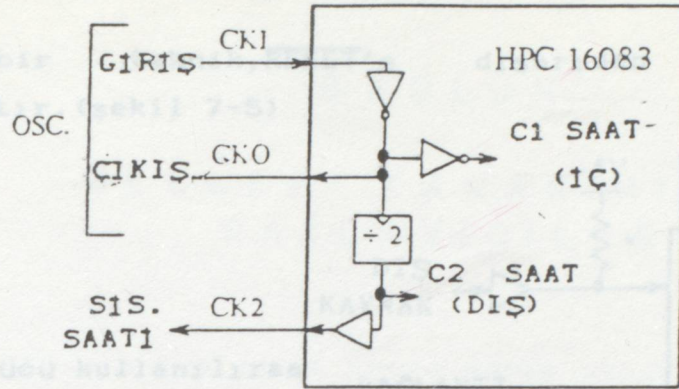


ŞEKİL 7-2. Yonga'daki Zamanlama Devresi

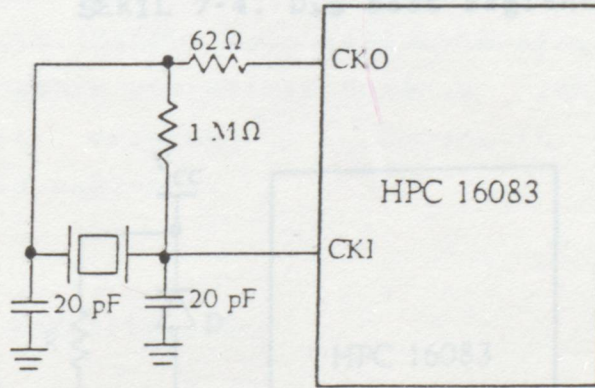


ŞEKİL 7-1. Güç Kaynağı Bağlantıları

ŞEKİL 7-3. Tipik Kristal Osilatör Bağlantıları

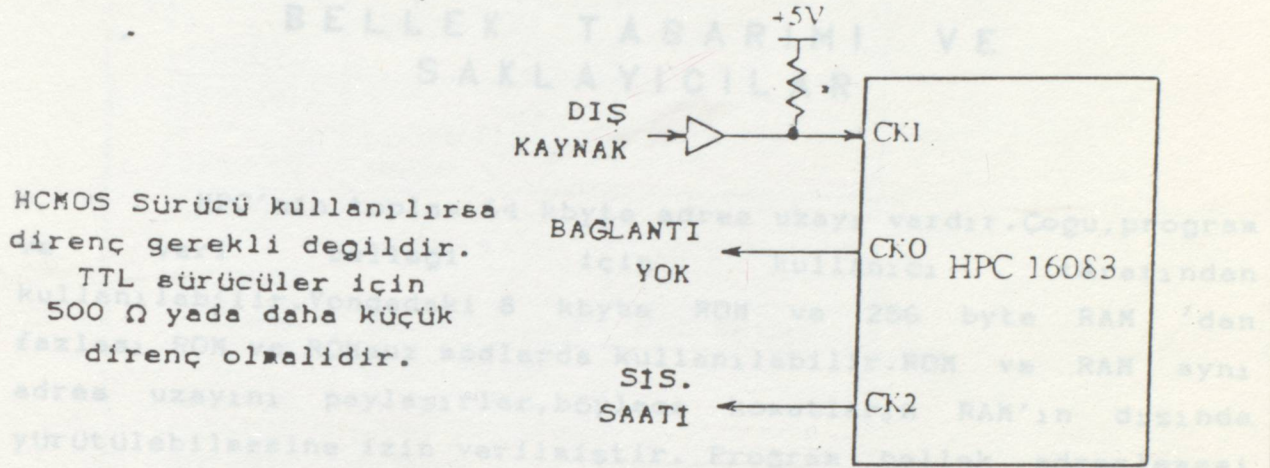


ŞEKİL 7-2. Yongadaki Zamanlama Devresi

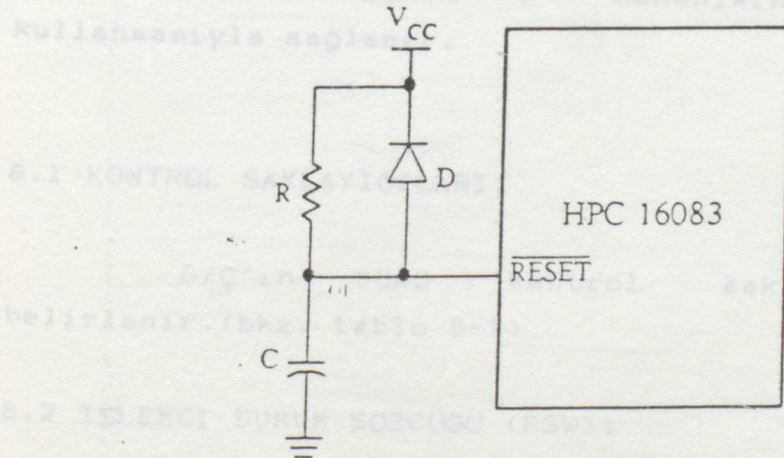


ŞEKİL 7-3. Tipik Kristal Osilatör Bağlantıları

basit bir teknik, RESET'e dışarıdan RC elemanlarını bağlamaktır. (Şekil 7-5)



ŞEKİL 7-4. Dış Saat Bağlantısı



$$RC \geq 5 \cdot \text{GÜÇ KAYNAĞI YÜKSELME ZAMANI}$$

ŞEKİL 7-5. Güç Açıldığında Reset Devresi

8. BÖLÜM

BELLEK TASARIMI VE
SAKLAYICILAR

HPC'nin toplam 64 kbyte adres uzayı vardır. Çoğu, program ve veri belleği için kullanıcı tarafından kullanılabilir. Yongadaki 8 kbyte ROM ve 256 byte RAM 'dan fazlası ROM ve ROMsuz modlarda kullanılabilir. ROM ve RAM aynı adres uzayını paylaşırlar, böylece komutların RAM'in dışında yürütülebilmesine izin verilmiştir. Program bellek adreslemesi 16-bit PC tarafından temel byte alınmıştır. Bellek komutlarla direk veya B, X, SP saklayıcılarının içerikleri ile dolaylı olarak adreslenebilir. Veri belleği, sözcüklerle veya bytelerle adreslenebilir. (Sözcük için çift-byte sınırlaması vardır.)

Yonganın çevrebirlik donanım fonksiyonları , G/Ç ve saklayıcılara erişmek , donanımın bellek haritasını kullanmasıyla sağlanır.

8.1 KONTROL SAKLAYICILARI:

G/Ç'in tümü kontrol saklayıcıları tarafından belirlenir. (bkz. tablo 8-1)

8.2 İŞLEMCİ DURUM SÖZCÜĞÜ (PSW):

PSW , saklayıcısı 8-bittir. Formatı aşağıdaki gibidir:

**	CGIE	C	EA	WT	WZ	H/I	EHI
X	XR	X	O	O	O	O	O

00C0 adresinde

TABLE 8-1. MEMORY MAP OF HPC16083

FFFF:FFF0	Interrupt Vectors in on-chip ROM	USER MEMORY
FFEF:FFD0	JSRP Vectors in on-chip ROM	
FFCF:FFCE : E001:E000	On-Chip ROM	
DFFF:DFFE : 0201:0200	External Expansion Memory	
01FF:01FE : 01C1:01C0	On-Chip RAM	USER RAM
0195:0194	Watchdog Register	Watchdog Logic
0192 0191:0190 018F:018E 018D:018C 018B:018A 0189:0188 0187:0186 0185:0184 0183:0182 0181:0180 015F:015E 015C	TOCON Register TMMODE Register DIVBY Register T3 Timer R3 Register T2 Timer R2 Register I2CR Register/ R1 I3CR Register/ T1 I4CR Register EICR Register EICON Register	Timer Block T0:T3
0153:0152 0151:0150 014F:014E 014D:014C 014B:014A 0149:0148 0147:0146 0145:0144 0143:0142 0141:0140	PORTP Register PWMODE Register R7 Register T7 Timer R6 Register T6 Timer R5 Register T5 Timer R4 Register T4 Timer	Timer Block T4:T7

TABLE 8-1. (Cont)

0128	INUR Register	UART
0126	TBUF Register	
0124	RBUF Register	
0122	ENU Register	
0120	ENU Register	
0104	PORTD Input Register	PORT D
00F5:00F4 00F3:00F2 00F1:00F0	BFLN Register DIRB Register DIRA Register / IBUF	PORTS A & B CONTROL
00E6	UPIC register	UPI CONTROL
00E3:00E2 00E1:00E0	PORTB PORTA Register / OBUF	PORTS A & B
00DE 00DD:00DC 00D8 00D6 00D4 00D2 00D0	Microcode ROM Dump Halt Enable Register PORTI Input Register SIO Register IRCD Register IRPD Register ENIR Register	PORT I, CONTROL & INTERRUPT CONTROL REGISTERS
00CF:00CE 00CD:00CC 00CB:00CA 00C9:00C8 00C7:00C6 00C5:00C4 00C3:00C2 00C0	X Register B Register K Register A Register PC Register SP Register (reserved) PSW Register	HPC CORE REGISTERS
00BF:00BF : 0001:0000	On-Chip RAM	USER RAM

O = Reset durumunda sıfırlanır.

X = Reset durumunda durumu belirsiz.

R = Sadece okuma.

Bit Tanımları:

** :Tanımsız bit.Belirsiz veriyi içerebilir.

CGIE :Bir kesme hizmete başladığında ENIR saklayıcısının GIE biti tarafından yüklenir.Maskelenemez kesmenin amaçları için kesmelerin , hizmet programlarından döndükten sonra tekrar iznin verildiğini gösterir.Sadece okunabilen bittir.

C :Bir toplama veya çıkartma komutu sırasında bir elde veya borç içerildiğini gösterir. Toplamada '1' eldeyi ve '0' elde olmadığını gösterir. Çıkartmada '0' borcu ve '1' borç olmadığını gösterir. Öteleme komutlarında , C biti akümülatörden dışarı ötelenen biti alır. Bölme komutlarında C biti hata bayrağı olarak kullanılır.

EA :Bu bit , dış erişimde adres alanının iznini tanımlar. Böylece tek yonga veya genişletilmiş moda HPC'nin işlem yapması belirlenir. Bu biti sıfırlamak , sadece yongadaki adres alanının adreslenmesinin iznini gerçekleştirir , birlemek ise tüm 64 kbyte adres uzayının kullanılmasına izin verir.

WAIT0/WAIT1:Bu ikibit, programlanabilen bekleme durumlarının sayısına izin verir . Kodlama şöyle :

00=4 Bekleme durumu

01=2 Bekleme durumu

10=1 Bekleme durumu

11=X Bekleme durumu yok

HLT/IDL :HLT/IDL modu seçme bitidir . Ya HALT ya da IDLE güç koruma modu seçiminde EHI PSW biti ile bağlantılı olarak çalışır . Halt modu , bu bitin birleşmesiyle ve Idle modu , sıfırlanmasıyla seçilir .

EHI :Bu kontrol biti , HLT/IDL kontrol biti ile

9. BÖLÜM

PORT YAPILARI VE İŞLEMLERİ

bağlantılı olarak çalışır . Bu bitin birleşmesi, HLT/IDLE bitinin seçimine göre yazılım yoluyla HPC ' nin HALT veya IDLE moda girmesini sağlar .

HPC ' nin kullanıcının programlayabileceği pinleri 3 porta bölünür . A , B , D , I ve P olarak atamışlardır .

9.1 PORT A :

Port A , 16-bit çift yönlü (her alıcı , her verici olarak üzere) G/C pinlerinden oluşmaktadır . 3 evetten birine hizmet eder . Şöyle ki:

- Yonga dışı bellek veya çevre birimleri ile haberleşme için çift yönlü adres/veri yolu olarak .

- 8-bit veya 16-bit UPI portu olarak . Bir dış yardımcı işlemci HPC ' deki programlara komutlar verebilir . Port A durma göstergelerini alır ve veri iletişimini yapar .

- 16-bit programlanabilen G/C pinleri : başlangıç girişi veya çıkışı olarak tanımlanabilir .

Port A ' nin bu 3 fonksiyonu özetler . Her pin yoluyla PSW EA biti veya ROMaux moda Dış Erişim Kontrolü ile çalışır . Port A adres/veri yolu olarak atanır . UPIC saklayıcısının UPINW bitinin birleşmesi Port A ' nin bir veya her iki byte 'ının UPI fonksiyonu için veri yolu olarak atanması sağlanır . Port A ile , yazılımla erişilebilen iki okuma/yazma kontrol saklayıcısı birleştirilmiştir . Bunlar : 16-bit veri saklayıcısı (Port A , adresi:00E0 ve 00E1) ve 16-bit yönlendirici saklayıcısı (Port A , adresi:00F0 ve 00F1) . AO pin bu saklayıcıların 0. bitine (en düşük anlamlı bit) , AI bit 1. bitine . karşılık düşmektedir .

Bir A saklayıcısı , Port A ' nin her bir pinine programla giriş veya çıkışı olarak izin verir .

Bir A ' nin her bitine 1 yazma ile veri saklayıcısı

9. BÖLÜM

PORT YAPILARI VE İŞLEMLERİ

HPC ' nin kullanıcının programlayabileceği pinleri 5 porta bölünür . A , B , D , I ve P olarak atanmışlardır .

9.1 PORT A :

Port A , 16-bit çift yönlü (hem alıcı , hem verici olmak üzere) G/Ç pinlerinden oluşmuştur . 3 amaçtan birine hizmet eder . Şöyle ki:

- * Yonga dışı bellek veya çevre birimlerle haberleşme için çift yönlü adres/veri yolu olarak .

- * 8-bit veya 16-bit UPI portu olarak . Bir dış yardımcı işlemci HPC ' deki programlara komutlar verebilir ; Port A durum göstergelerini alır ve veri iletimini yapar .

- * 16-bit programlanabilen G/Ç pinleri bağımsız giriş veya çıkış olarak tanımlanabilir .

Port A ' nın bu 3 fonksiyonu özeldir . EXM pini yoluyla PSW EA biti veya ROMsuz modda Dış Erişim Modunun seçilmesiyle ; Port A adres/veri yolu olarak atanır . UPIC saklayıcısının UPIEN bitinin birleşmesi Port A ' nın bir veya her iki byte 'ının UPI fonksiyonu için veri yolu olarak atanması sağlanır . Port A ile , yazılımla erişilebilen iki okuma/yazma kontrol saklayıcısı birleştirilmiştir . Bunlar : 16-bit veri saklayıcısı (Port A , adresi:OOE0 ve OOE1) ve 16- bit yönlendirici saklayıcısı (Dir A , adresi:OOF0 ve OOF1) . A0 pini bu saklayıcıların 0. bitine (en düşük anlamlı biti) , A1 biti 1. bitine .. karşılık düşmektedir.

Dir A saklayıcısı , Port A ' nın her bir pinine programla giriş veya çıkış olarak izin verir .

Dir A ' nın her bitine 1 yazmakla veri saklayıcısı

çıkış olarak koşullanır . Okumada veri saklayıcısı ' nın değeri bu pinlerde gözükür .

Dir A ' nın her bitine 0 yazmakla veri saklayıcısı giriş olarak koşullanır . Bu durumda pinlerden veri girişi yapılabilir .

Port A ve Dir A saklayıcılarının bit düzenleri aşağıda verilmiştir :

A15	A14	A13	A12	A11	A10	A 9	A 8	A 7	A 6	A 5	A 4	A 3	A 2	A 1	A 0
X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

OOE1

OOEO

PORT A

A15	A14	A13	A12	A11	A10	A 9	A 8	A 7	A 6	A 5	A 4	A 3	A 2	A 1	A 0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OOF1

OOFO

DIR A

0 = Resette sıfırlanır

X = Resette durumu belirsiz

NOT : Port A ' nın bütün pinlerine pull-up dirençleri bağlanmalıdır .

9.2 PORT B :

Port B , 16 pinden oluşur . Bunların 12 ' si Port A ' nın yapısına benzer çift yönlü G/Ç pinleridir . Diğer 4 pin (B10,B11,B12,veB15) sadece çıkış olarak kullanılır . Yazılımla bu pinlerin durumlarının değiştirilmesine izin verilmez .

Port B ' de veri saklayıcısı (Port B , adresi:00E2) , yönlendirici saklayıcısı (Dir B , adresi:00F2) ve fonksiyon saklayıcısı (Bfun , adresi:00F4) bulunmaktadır .

Port B veri ve Dir B saklayıcılarının işlemleri ,Port A ' dakilerle aynıdır .

Bfun saklayıcı bitine 1 yerleştirmek değişik fonksiyon seçilebilmesini sağlar .

Port B pinlerinden , aşağıdaki fonksiyonlar sağlanır :

B0 - TDX.....UART veri çıkışı

B1 - Boş

B2 - CKX.....UART saati (giriş ya da çıkış)

B3 - T2IO....2.zamanlayıcının giriş veya çıkışı

B4 - T3IO....3. " " " "

B5 - SO.....MICROWIRE/PLUS veri çıkışı

B6 - SK..... " " saati (giriş ya da çıkış)

B7 - HLDA....Tutma isteği alındı çıkışı

B8 - TSO.....Zamanlayıcı senkronlama çıkışı

B9 - TS1..... " " "

B10- UAO.....UPI portunun adres 0 girişi

B11- WRRDY...UPI portunun yazma hazır çıkışı

B12- Boş

B13- TS2.....Zamanlayıcı senkronlama çıkışı

B14- TS3..... " " "

B15- RDRDY...UPI portu okuma hazır çıkışı

Genişletilmiş veya ROMsuz modlarda işleme girildiğinde 3 Port B pinleri B10 , B11 ve B15 , kontrol veri sinyalleri $\overline{ALE}$, $\overline{WR}$, ve $\overline{RD}$ olur .

Resette , seçilen veri genişliği 16 bit ise , B12 pini
 daıma HBE işareti olarak tanır . Diğer bir deyişle bu pin ,
 programlanabilen bir çıkıştır .

Port B ve Dir B saklayıcılarının bit düzenleri aşağıda
 verilmiştir :

B15	B14	B13	B12	B11	B10	B 9	B 8	B 7	B 6	B 5	B 4	B 3	B 2	B 1
X0	X	X	X0	X0	X0	X	X	X	X	X	X	X	X	X

OOE3

OOE2

PORT B

B15	B14	B13	B12	B11	B10	B 9	B 8	B 7	B 6	B 5	B 4	B 3	B 2	B 1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

OOF3

OOF2

DIR B

X = Resette belirarız.

0 = Resette sıfırlanır

0 = Sadece çıkıştır . Giriş olarak kullanılamaz

NOT : Port B ' nin bütün pinlerine pull-up dirençleri
 bağlanmalıdır .

9.3 PORT I :

Port I ' nin 8 pini giriştir . Bu porttan ,yonganın
 özelliklerini desteklemek için gerekli fonksiyonlar elde edilir
 . Port I için bir fonksiyon saklayıcısı yoktur . Bu port aktif

hale getirildiğinde otomatik olarak giriş için uygun olarak kullanılacaktır . I1 giriş pini , maskelenemez kesme için ayrılmıştır . Aşağıdaki listede Port I pinlerinin fonksiyonları görülmektedir :

- I0 - Fonksiyonu yok
- I1 - Maskelenemez kesme (NMI)
- I2 - 2. kesme/giriş tutucusu/ $\overline{URD}$
- I3 - 3. kesme/giriş tutucusu/ $\overline{UWR}$
- I4 - 4. kesme/giriş tutucusu
- I5 - SI (MICROWIRE PLUS seri veri girişi)
- I6 - RDX (UART seri veri girişi)
- I7 - Fonksiyonu yok

Port I ' nin bit düzeni aşağıda verilmiştir :

I7	I6	I5	I4	I3	I2	I1	I0
XR	RDX XR	SI XR	INT4 XR	INT3 XR	INT2 XR	NMI XR	XR

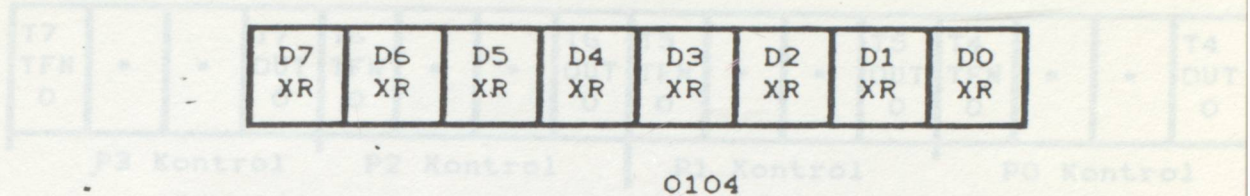
OOD8

X = Resette durumu belirsiz .

R = Sadece okuma . Yazmada etkisizdir .

9.4 PORT D :

Port D , genel amaçlı sayısal giriş olarak kullanılabilen 8 adet pinden oluşmuştur . Port D ' nin pin düzeni aşağıda verilmiştir :



X = Resette durumu belirsiz .

R = Sadece okuma . Yazmada etkisizdir .

* = İsaretsiz bit

0 = Resette sıfırlanır

9.5 PORT P :

Port P , genel amaçlı sayısal çıkış olarak kullanılabilen 4adet çıkış pininden oluşmuştur . Bu pinlerden herhangi biri bir zamanlayıcıya bağlanabilir . T4 P0 pinine , T5 P1 pinine , T6 P2 pinine ve T7 P3 pinine bağlanabilir .

16-bit Port P saklayıcısı , hem pinlerdeki değerleri tutar , hem de her pinin fonksiyonunu belirler . Bir pin , zamanlayıcıya bağlandığında , Port P ' nin veri pini yazılım tarafından sadece okuma ve ilgili zamanlayıcı alttan taşıdığına tümleyen olarak çalışır .

Port P pinleri , 3 durumlu olamaz ve resette lojik 0-a çekilir .

Bu bitler , TFN bitinin birleşmesiyle tümlene moduna girildiğinde sadece okuma olarak çalışır .

TFN = 1.....Zamanlayıcının tümlenmede alttan taşmasına izin verir

TFN = 0.....Zamanlayıcının tümlenmede çalışması yasaklanır

Aşağıda Port P ' nin bit düzeni verilmiştir .

T7 TFN 0	*	*	T7 OUT 0	T6 TFN 0	*	*	T6 OUT 0	T5 TFN 0	*	*	T5 OUT 0	T4 TFN 0	*	*	T4 OUT 0
P3 Kontrol				P2 Kontrol				P1 Kontrol				P0 Kontrol			

0153

0152

* = İşaretsiz bit

0 = Resette sıfırlanır

VEKTÖR

KESME

ADRESİ

KAYNAĞI

ÖNCELİK

FFFF	Reset:Reset girişine sıfır seviyesi uygulandığında kesme tetiklenir.	0
FFFFC	Maskelenemez kesme: Dış 11 pininin yüksek kenarında tetiklenir.	1
FFFFA	Dış 12 pininin yüksek kenarında tetiklenir.	2

*Tablonun devamı başka sayfada.

10. BÖLÜM

KESMELER

HPC , 8 kaynaktan vektörlenmiş kesmeleri sağlar .Bu kaynaklar : RESET pini , Port I ' nın I2 , I3 ve I4 pinleri , Eİpini, zamanlayıcı bölümü ve UART ' dır . HPC , aynı anda iki veya daha fazla kesme geldiğinde hangi kesmeye hizmet verileceğine karar vermek için karar lojigi içermektedir . Aşağıdaki tabloda , kesmelerin öncelikleri verilmiştir:

VEKTÖR ADRESİ	KESME KAYNAGI	ÖNCELİK
FFFF	Reset:Reset girişine sıfır seviyesi uygulandığında kesme tetiklenir .	0 (en öncelikli)
FFFC	Maskelenemez kesme:Dış , I1 pininin yükselen kenarında tetiklenir .	1
FFFA	Dış , I2 pininin yükselen/düşen kenarında tetiklenir.	2

*)Tablonun devamı arka sayfada

FFF8	Dış , I3 pininin yükselen/ düşen kenarında tetiklenir.	3
FFF6	Dış , I4 pininin yükselen/ düşen kenarında tetiklenir.	4
FFF4	İç , zamanlayıcılardan	5
FFF2	İç , UART'dan veya dış ise EXUI pininden	6
FFFO	Dış , yükselme/düşme veya EI pininin I/O seviyesi	7

Kesme Öncelik tablosu

RESET ve NMI maskelenemeyen kesmelerdir . Maskelenebilir kesmeler , ayrı ayrı izin verilebilir veya yasaklanabilir . ENIR kontrol saklayıcısı her maskelenebilir kesme kaynağına karşılık düşecek bitler içerir . İlgili bit birleştirilerek kesme kaynağı izin veya sıfırlanarak yasak edilebilir . HPC aynı zamanda ENIR saklayıcısı içindeki Global Kesme İzni (GIE) sayesinde maskelenebilir kesmelerin , global olarak izin veya yasak edilmelerini sağlar . GIE bitinin

sıfırlanmasıyla bütün maskelenebilir kesmeleri yasak eder . Birleşmesiyle ilgili kaynağın ENIR biti de 1 durumunda ise kaynağa izin verilmiş olur . Reset anında GIE biti de sıfırlanır .

Reset aktif sıfır seviye duyarlıklı kesmedir . NMI (I1), I2 , I3 ve I4 ise kenar duyarlıklı kesmelerdir . NMI yükselen kenar , I2,I3 ve I4 ise IRCD kontrol saklayıcısındaki karşılık düşen bitlerle çıkan veya inen kenar olmak üzere programlanabilir . EI ise EICOM saklayıcısını kullanarak yüksek veya düşük seviye , inen veya çıkan kenar olmak üzere programlanabilir .

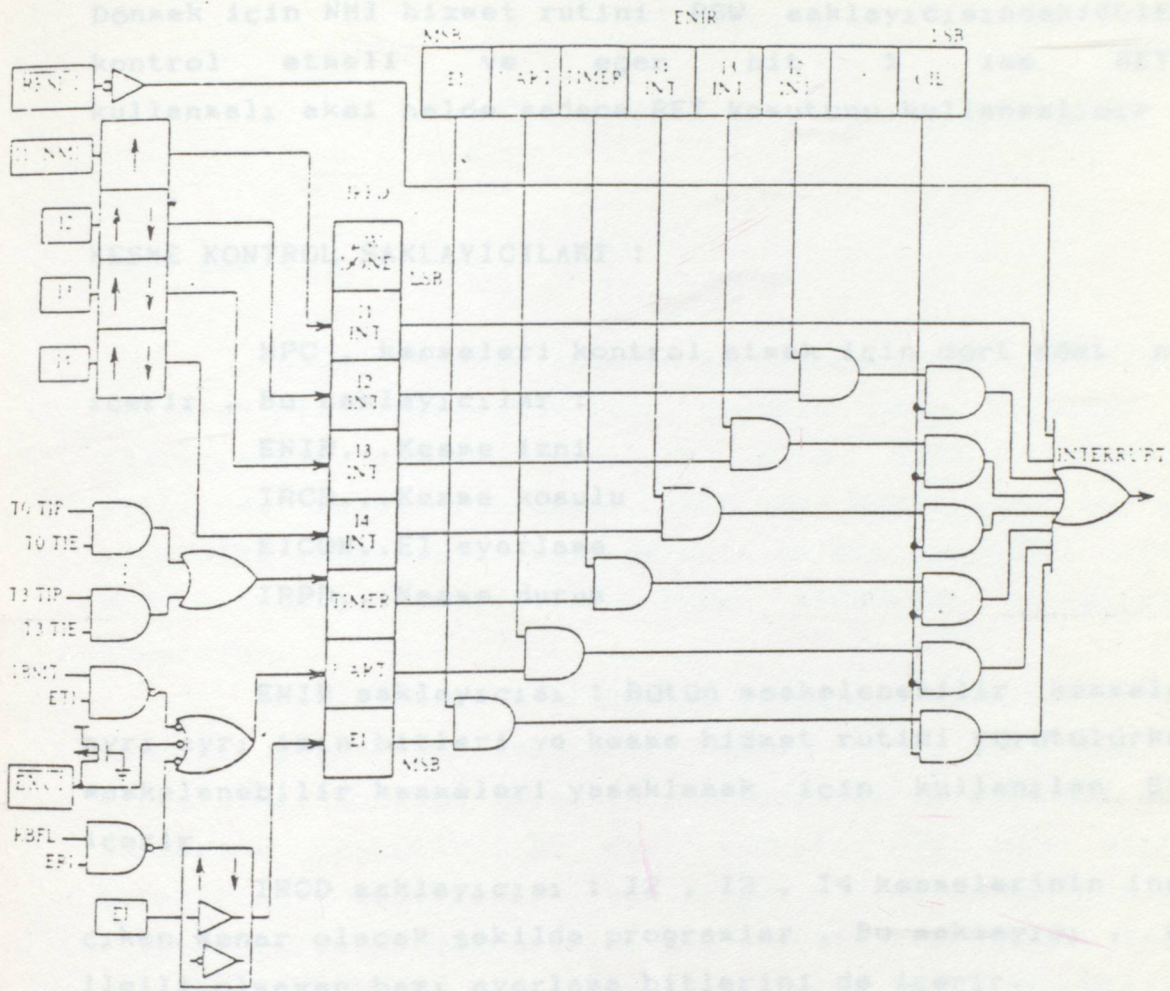
HPC , belleğin son taraflarında yerleştirilmiş sekiz adet kesme kaynağını tanır . Her bir vektör 16 bitliktir ve programcı tarafından her bir vektör için hizmet programının adresi ile doldurulabilir . Kesme hizmet rutinleri 64 Kbyte'lik adres alanının herhangi bir yerine yerleştirilebilir .

KESME İŞLEMİ :

Reset dışındaki bütün kesmelerde HPC , yürütmekte olduğu komutu tamamladıktan sonra hizmet verir .

Bir kesme alındıktan sonra HPC , ENIR saklayıcısının global kesme iznini (GIE) bitini PSW saklayıcısının CGIE bitine kopyalar . Daha sonra GIE biti reset edilerek bütün maskelenebilir kesmeler yasak edilir . Dönüş adresi yığın üzerine Push edilerek yığın göstergesi iki artırılır . Daha sonra ilgili kesme vektörünün içeriği PC ' ye yüklenir .

Kesmeler kesme hizmet rutininde iken GIE biti birleşerek işler hale getirilebilir . Aksi takdirde RETI (kesmeden geri dönüş) komutu yürütüldükten sonra otomatik olarak GIE biti birleşir ve kesmelere izin verilmiş olur . Bu komutla aynı zamanda dönüş adresi yığından Pop edilir . PSW saklayıcısındaki CGIE biti NMI kesmeden dönüşte kullanılır .



Sekil 10.1 Keşme Lojigini için Blok Diyagramı

Dönmek için NMI hizmet rutini PSW saklayıcısındaki CGIE bitini kontrol etmeli ve eğer bit 1 ise RETI 'yi kullanmalı aksi halde sadece RET komutunu kullanmalıdır .

KESME KONTROL SAKLAYICILARI :

HPC , kesmeleri kontrol etmek için dört adet saklayıcı içerir . Bu saklayıcılar :

ENIR...Kesme izni

IRCD...Kesme kosulu

EICON..EI ayarlama

IRPD...Kesme durum

ENIR saklayıcısı : Bütün maskelenebilir kesmeler için ayrı ayrı izin bitleri ve kesme hizmet rutini yürütülürken bütün maskelenebilir kesmeleri yasaklamak için kullanılan GIE biti içerir .

IRCD saklayıcısı : I2 , I3 , I4 kesmelerinin inen veya çıkan kenar olacak şekilde programlar . Bu saklayıcı , kesmeyle ilgili olmayan bazı ayarlama bitlerini de içerir .

EICON saklayıcısı : EI pini için mümkün dört kesme çeşidini programlar (inen kenar , çıkan kenar , 0 seviye , 1 seviye) . Bu aynı zamanda herhangi bir kenar duyarlıklı kesme alındıktan sonra IRPD saklayıcısındaki EI bitini sıfırlamak için bir tasdik biti içerir .

IRPD saklayıcısı : Hangi kesmeye hizmet verildiğini gösterir . Herbir kesme vektörü bir bit ile temsil edilir . Bu bitler kesme durumunda birleşir . IRPD saklayıcısının NMI UART , Zamanlayıcı ve EI bitleri sadece okunabilir .

11 BÖLÜM

MICROWIRE / PLUS

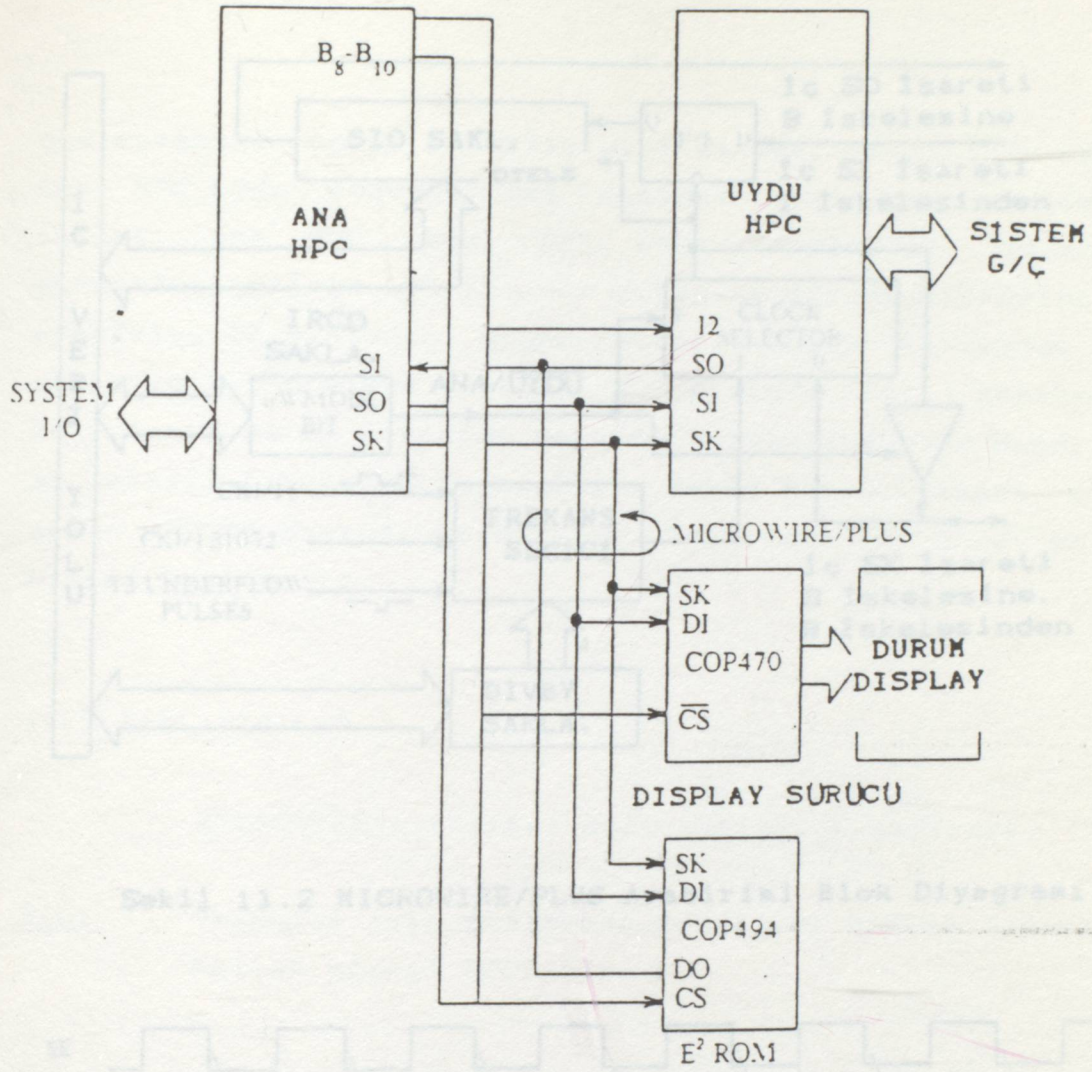
11.1 GİRİŞ

MICROWIRE bir senkron seri haberleşme sistemidir. Çevre birimleriyle haberleşmek için gerekli olan bağlantı sayısını (dolayısıyla fiyatı) azaltmak için COPS mikrodeneleyici ailesinde temel vasıta dır. HPC de kullanılan bu yol COPS MICROWIRE arabirimine katkıda bulunan bir kısımdır. MICROWIRE / PLUS OLARAK isimlendirilir. Bütün MICROWIRE çevrebirimleriyle uyumludur.

Sekil 11.1 de MICROWIRE cihazları arasındaki tipik bir bağlantı şeması görülmektedir. Herbir cihaz S1 veya D1 seri giriş pini, S0 veya D0 seri çıkış piniyle SK veya CLK seri saat pinlerine sahiptir. Bu saat işareti bir ana mikroişlemci tarafından hazırlanır ve diğer uydu mikroişlemcilerle çevrebirimlerine gönderilir. HPC hem ana hem de uydu tipi olarak kullanılabilir. Ek bir işaret olan Yonga Seçme işareti ise çevre birimi seçmek için kullanılır ve ana HPC nin herhangi bir iskelesinden elde edilir.

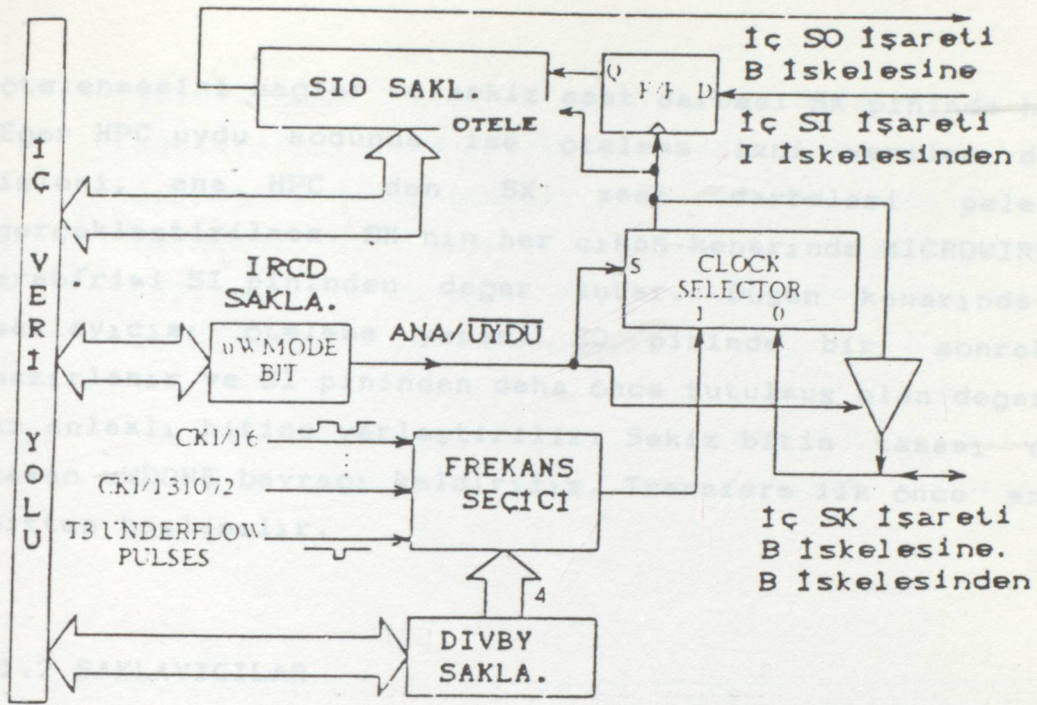
MICROWIRE / PLUS yolu, S1 , S0, ve SK hatlarına sahiptir ve SIO ötelemeli saklayıcısıyla beraber kullanılır. Sekil 11.2 de MICROWIRE / PLUS arabirim devresinin blok diyagramı görülmektedir.

Sekil 11.3 de MICROWIRE transferinde izlenilen sıra görülmektedir. Transfer, SIO saklayıcısına yazma ile başlatılır. Bu işlem uWDONE bayrağını (IRPD saklayıcısının 0. biti) indirir. Bu bayrak SIO saklayıcısına öteleme izni vermek için kullanılır. Eger HPC ana modda ise bu yükleme işlemi aynı zamanda SIO saklayıcısının otomatik olarak sekiz kez sola

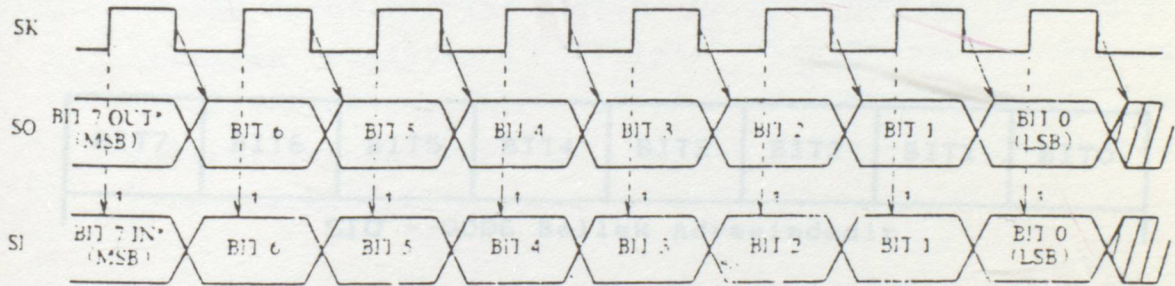


GL-37-0-U

Şekil 11.1 Bir MICROWIRE Uygulaması Örneği



Şekil 11.2 MICROWIRE/PLUS Arabirimi Blok Diyagramı



Şekil 11.3 MICROWIRE/PLUS Transferi

ötelenmesini sağlar ve sekiz saat darbesi SK pininde hazırlanır. Eger HPC uydu modunda ise öteleme izni verilse de öteleme işlemi, ana HPC den SK saat darbeleri gelene kadar gerçekleştirilmez. SK nın her çıkan kenarında MICROWIRE / PLUS arabirimi SI pininden değer tutar. Düşen kenarında ise SIO saklayıcısı öteleme yapar, SO pininde bir sonraki değer hazırlanır ve SI pininden daha önce tutulmuş olan değer onun en az anlamlı bitine yerleştirilir. Sekiz bitin tamamı ötelendiği zaman uWDONE bayrağı kaldırılır. Transfere ilk önce en anlamlı bitten başlanılır.

11.2 SAKLAYICILAR

MICROWIRE arabirimi dört saklayıcı kullanır.

SIO: Transfer için ötelemeli saklayıcı olarak hizmet verir. Bellek adresi OOD6 dir.

BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
------	------	------	------	------	------	------	------

SIO - OOD6 Bellek Adresindedir

NOT: Bazı komutlar (ADD gibi), hedef alanına hem okuma hem de yazmak için erişir. MICROWIRE, transfer işlemini sadece yazma erişiminde yapar. Bunun için SIO saklayıcısına okuma-değişiklik yapma-yazma erişimleri son yazılan değer transferinden etkilenir. Veriyi ötelerken SIO saklayıcısına yazmaya dikkat edilmelidir. Böyle bir işlem belirsiz bir veri

yazılmasına sebep olabilir. Uydu modunda SIO saklayıcısı ana mikroişlemciden SK saat darbeleri gelinceye kadar öteleme yapmaz. Bu bakımdan SIO ya herhangi bir sayıda erişim (okuma veya yazma) yapılabilir. Bu işlem ana mikroişlemci SK darbelerini göndermeden tamamlanmalıdır.

IRCD: Kesme durumlarını saklar. 1. bit uWMODE bitidir. HPC nin ana MICROWIRE veya uydu olarak işlem görmesini sağlar.

**	**	**	I4 POL	I3 POL	I2 POL	uWMODE	RDY/HD
0	0	0	0	0	0	0	0

IRCD - OOD4 Bellek Adresindedir

IRPD: Kesme önceliklerini saklar. 0. biti uWDONE bayrağıdır. Transfer başlayınca silinir (indirilir). Transfer tamamlanınca birleşir (kaldırılır). uWDONE bayrağı IRPD saklayıcısında bir bit işgal etmesine rağmen kesme işleminden etkilenmez.

EI	UART	TIMERS	INT4	INT3	INTR	NMI	uWDONE
XR	XR	XR	XC	XC	XC	XR	XR

IRPD - OOD2 Bellek Adresindedir

0: Resette silinir.

X: Resette belirsiz durum

R: Sadece okunabilir.

C: Sıfır yazılarak silinebilir. Bir yazılınca içeriği değişmez.

DIVBY: Saat işareti belirler. 0-3 cü bitleri ana moddaki HPC tarafından üretilen SK saatinin oranını belirlemek için kullanılır. SK kristal frekansının (CKI) önceden belirlenen 14 durumunun herhangi birinden veya T3 zamanlayıcısının alttan taşmasıyla elde edilebilir.

T3	T2	UART	MICROWIRE
018F Bellek Adresindedir		018E Bellek Adresindedir	

DIVBY - 018E Bellek Adresindedir

BIT3	BIT2	BIT1	BIT0	MICROWIRE SAATI
0	0	0	0	İzin Verilmez
0	0	0	1	T3'ün alttan taşmasıyla
0	0	1	0	CKI/16
0	0	1	1	CKI/32
0	1	0	0	CKI/64
0	1	0	1	CKI/128
0	1	1	0	CKI/256
0	1	1	1	CKI/512
1	0	0	0	CKI/1024
1	0	0	1	CKI/2048
1	0	1	0	CKI/4096
1	0	1	1	CKI/8192
1	1	0	0	CKI/16384
1	1	0	1	CKI/32768
1	1	1	0	CKI/65536
1	1	1	1	CKI/131072

*: T3 zamanlayıcısının her bir alttan taşması SK'nın bir

çevrimine karşılık düşer. T3 zamanlayıcısına ve R3 saklayıcısına n değerini yerleştirmek T3 giriş saatinin $n+1$ e bölmek işlemini gerçekleştirir. T3 den elde edilen SK işareti genelde tamamıyla kare dalga değildir. Bu nedenle varolan birçok MICROWIRE çevrebirimleriyle uyumluluk sağlanamaz. Bununla birlikte HPC ler arasındaki haberleşmede kullanılır. SK darbeleri arasındaki boşluk süreleri SKI saatinin sekiz çevriminde sabitleştirilir. Bu işlem herbir çevrimin kalanı için Lojik 1 işaretiyle gerçekleştirilir. T3 , yonga dışındaki bir simetrik saat kaynağını bire bölse bile asimetriklik aynen kalır.

11.3 İŞLEMİ GERÇEKLEŞTİRME

SO çıkış işareti B iskelesinin B5 pininin diğer fonksiyonudur. Bu fonksiyona izin vermek için BFUN ve DIRB saklayıcılarının 5. bitlerini birlemek gerekir.

SI girişi I iskelesinin I5 pininden otomatik olarak alınır. Bunun için hiçbir koşullamaya gerek yoktur.

SK saati B iskelesinin B6 pininin diğer fonksiyonudur. Bununla beraber B iskelesinin B6 pinini koşullamak için iki yerine üç bite gerek vardır. Bu bitler BFUN ve DIRB saklayıcılarının 6. bitleriyle IRCD saklayıcısının uWMODE bitidir. Bu bitlerin izin verilen kombinasyonları için SEÇME TABLOSU na bakınız.

Tabloda DIKKAT yazılan kombinasyonlar her an seçilemez. Çünkü bu durumlar yongadaki iki tamponda karışıklıklara yol açar. HPC, kendi kendisine hasar vermez. Fakat bu seçim fazla güç harcamasına ve/veya önceden tahmin edilemeyen durumlara yol açabilir.

BFUN 6.BIT	DIRB 6.BIT	uWMODE IRCD 1.BITI	B6 Pini İçin Seçilen Fonks.
0	0	0	SK saat girişi (MICROWIRE uydu modu). Bu mod B6'nın B iskelesi girişi olarak kullanılmasına izin verir
0	0	1	DIKKAT
0	1	0	Iskele Çıkışı
0	1	1	DIKKAT
1	0	0	Geçersiz Seçme (Geçersiz SK saat işaretli uydu mod)
1	0	1	Sadece B Iskelesi girişi olarak. (SK saat çıkışı olmayan ana mod)
1	1	0	Geçersiz Seçme (Geçersiz SK saat işaretli uydu mod)
1	1	1	SK saat çıkışı (MICROWIRE Ana Mod)

SEÇME TABLOSU

Bu üç bit reset durumunda otomatik olarak silinir.

uWDONE bayrağı reset durumunda indirilmez (silinmez).

Çünkü transfer isteğini gösterebilir.

MICROWIRE / PLUS arabirimini koşullamak için şu prosedür kullanılabilir:

1. BFUN ve DIRB saklayıcılarının 5. bitleri birlenir. Bu işlem B5 pinini S0 olarak kullanmayı sağlar.
2. BFUN saklayıcısının 6. bitleri birlenir. Bu işlemle B6 pinin SK olarak koşullanması sağlanır.

3. DIVBY saklayıcısının en az anlamlı dört bitine 0010H sayısı yerleştirilerek arabirim için zaman tablosundan CKI/16 saatı seçilir.
4. IRCD saklayıcısının 1. biti (uWMODE) birleştirir. Bu ana birimi ana moda geçirir. Eger uWDONE biti hala birleşmişse o anda öteleme yapacak ve onu kaydedecektir.
5. uWDONE bayrağı kaldırılana kadar beklenir ve sonra arabirim koşullanır. Eger ana mod seçilmek isteniliyorsa SK yı çıkış yapmak için DIRB saklayıcısının 6. bitini birleştirir. Ayrıca DIVBY saklayıcısının en az anlamlı dört bit yardımıyla zaman tablosundan istenilen saat işareti seçilir. Eger uydu mod isteniliyorsa önce uWMODE biti sonra da BFUN saklayıcısının 6. bitini silinir.

NOT: Reset durumunda SO ve SK bitleri, B iskelesinin diğer bitleri gibi belirsiz durumdadır. MICROWIRE çevrebirimlerinden istenmeyen darbeler alınmadığına emin olmak için SK pini pull-down direnciyle beraber kullanılmalıdır.

12. BÖLÜM

YOL FONKSİYONLARI

12.1 YOL MODLARI VE SEÇİMLERİ :

Port A ' nın 16 biti ve Port B ' nin 4 biti , yonga dışı bellek ve/veya çevrebirimlere erişmek için kullanılır .

Port A , 16-bit adres/veri yolu olarak seçilebilir ve Port B pinleri , B10 , B11 , B12 ve B15 sırasıyla yol kontrol işaretleri ALE , $\overline{WR}$, $\overline{HBE}$ ve $\overline{RD}$ olarak kullanılır .

HPC ' nin kullanılmasında 3 ayrı mod söz konusudur :

- * Tek yonga veya Genişletilmiş mod
- * Normal veya ROMsuz mod
- * 8-bit veya 16-bit mod

12.1.1 TEK YONGA VEYA GENİŞLETİLMİŞ MOD :

Bellek adres alanı , 2 mod ile belirlenir . Tek yonga modu , 8K adres alanını kullanabilir ve 8K adres alanı dışındaki adresleri belirleyen watchdog lojigine izin verir .

Genişletilmiş mod , HPC ' nin 64 Kbyte adres alanında herhangi bir yerde bellek erişimi yapmasına izin verir .

Tek yonga modu , EA bitinin sıfırlanmasıyla seçilir. Tek yonga modunda , sadece yonga içindeki ROM ve RAM alanlarının kullanılabilmesine izin verir . Yonga dışı bellek kullanımına izin vermez .

Genişletilmiş mod , EA bitinin bire çekilmesiyle seçilir . Genişletilmiş mod , herhangi bir yerde (yonga içi veya yonga dışı) 64 Kbyte adres alanında bellek erişimine izin verir.

Genişletilmiş modda tanımsız adresi bulan watchdog lojigini yasaklar .

12.1.2 NORMAL MOD VEYA ROMsuz MOD :

Normal modda yongadaki 8 Kbyte ROM kullanılabilir . ROMsuz modda ise bu adres alanı , yonga dışında kullanılabilir .

ROMsuz mod , tek yonga ve genişletilmiş işlemlerinde ilk modeli sağlar . Bir dış bellek , yongadaki ROM ile yer değiştirmek için kullanılır . ROMsuz mod seçildiğinde bütün erişimler , yonga içi ve dışındaki bellek adreslerinde yol transferi sağlar .

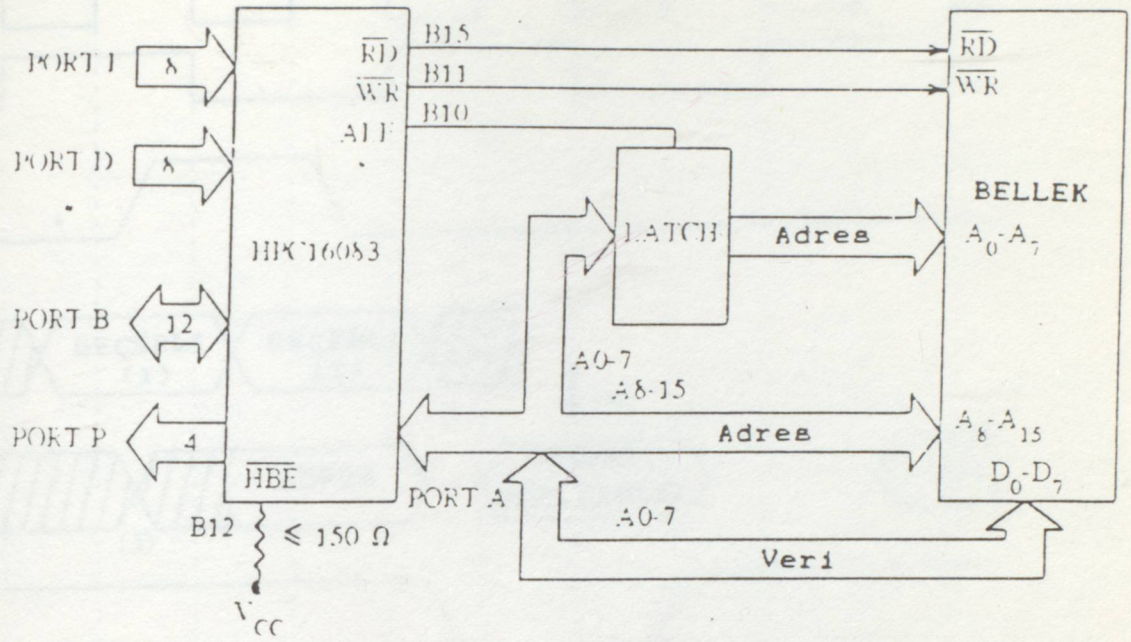
12.1.3 8-BİT VEYA 16-BİT MODU :

16-bit modunda Port A ' nın bütün pinleri , veri veya adres transferinde kullanılır . İki sekizlik tutucu , veri transferi sırasında adres tutmaları için gereklidir . Bu tutucular , HPC ' nin ürettiği ALE işareti ile senkronlanırlar .

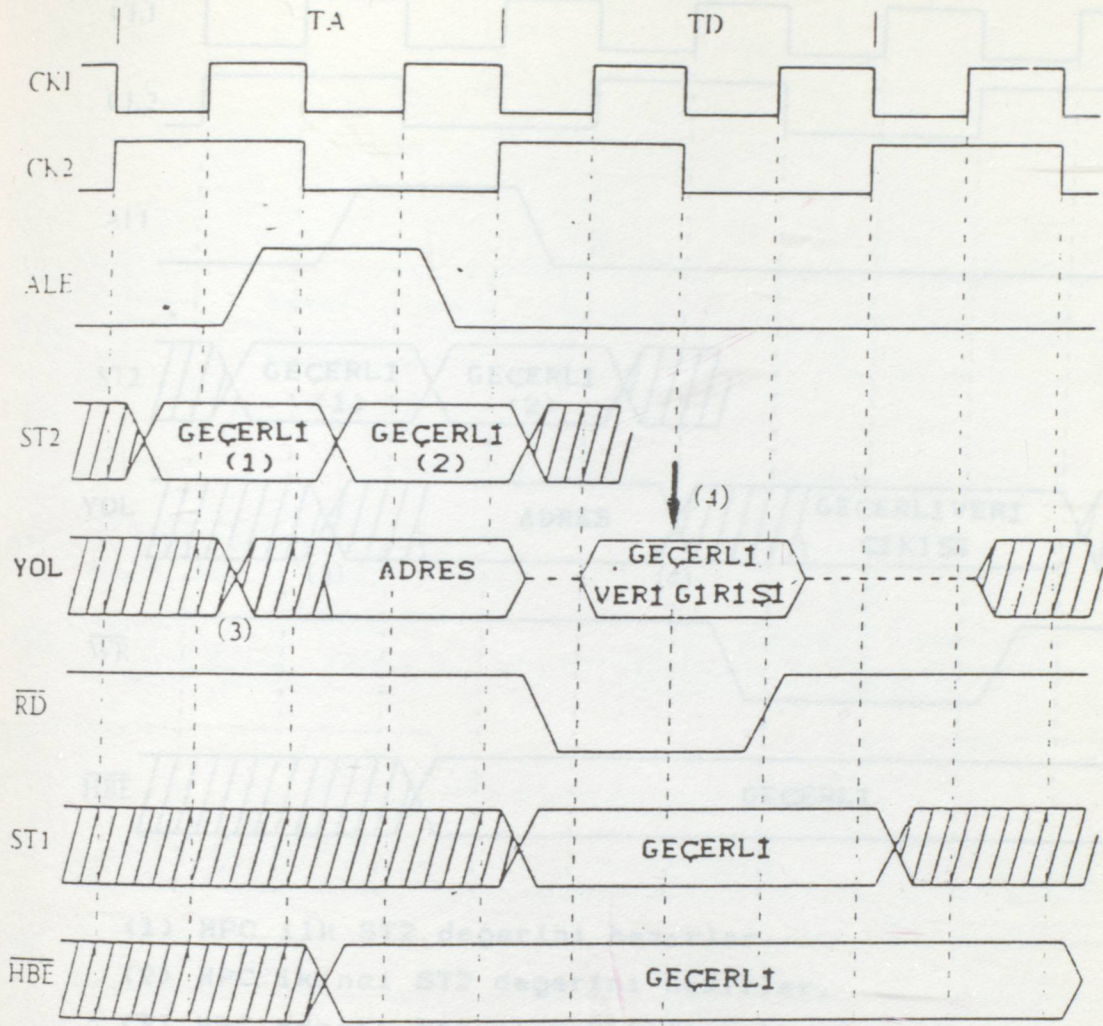
8-bit modunda , sadece A0 - A7 pinleri , veri transferinde kullanılır . A8 - A15 pinleri , adresin en anlamlı sekizlisini tutar . Adresin diğer sekizlisini de bir adet sekizli tutucu tutar .

$\overline{\text{HBE}}(\text{B12})$ pini sıfıra çekilirse 16-bit modu , bire çekilirse 8-bit modu seçilmiş olur .

ŞEKİL 12-1 Sistem Konfigürasyonu : 8 Bit Modu

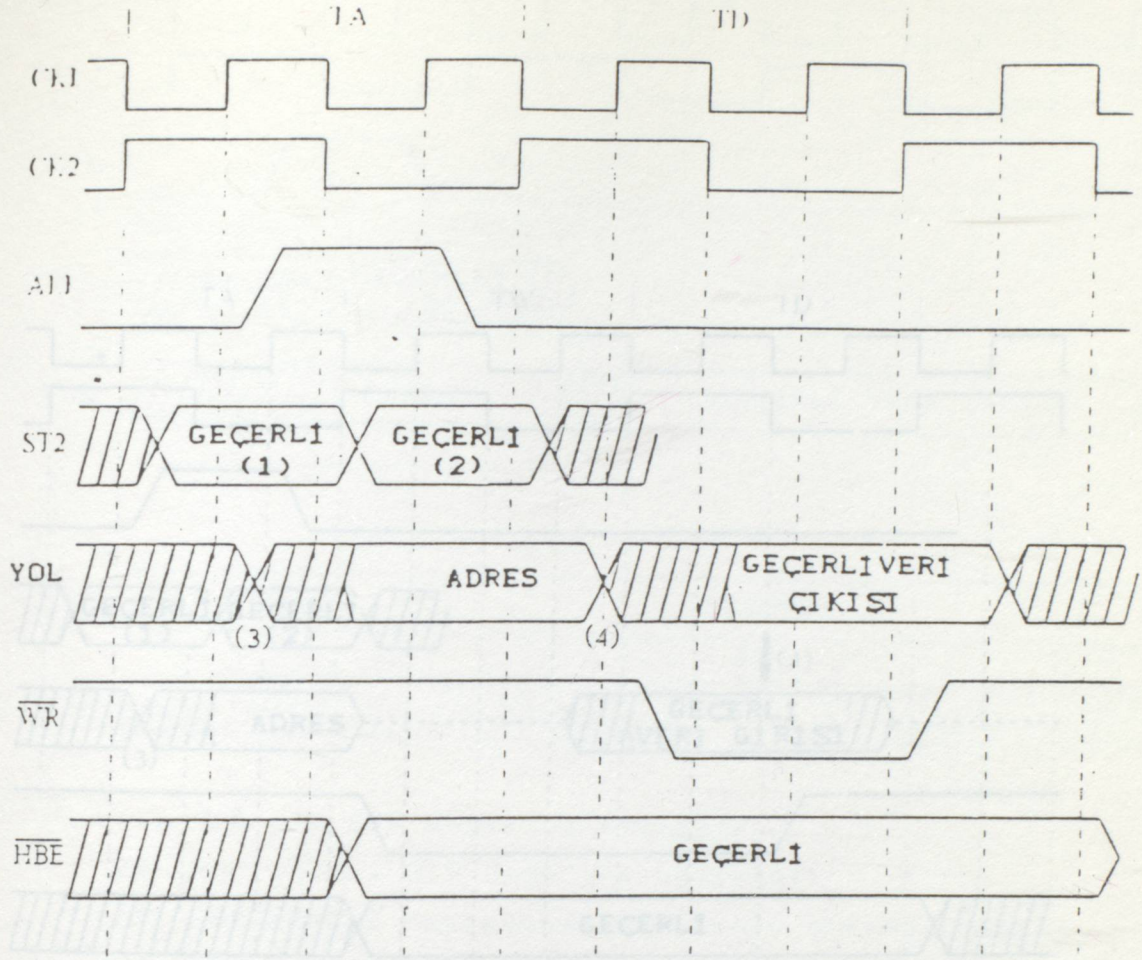


ŞEKİL 12-1 Sistem Konfigürasyonu : 8 Bit Modu



- (1) HPC ilk ST2 deęerini hazırlar.
- (2) HPC ikinci ST2 deęerini hazırlar.
- (3) HPC adresi hazırlamak için yolu sürmeye başlar.
- (4) HPC giriş verisini örnekler.

ŞEKİL 12-2 Yol Zamanlaması : Okuma Çevrimi; Bekleme Durumu Yok

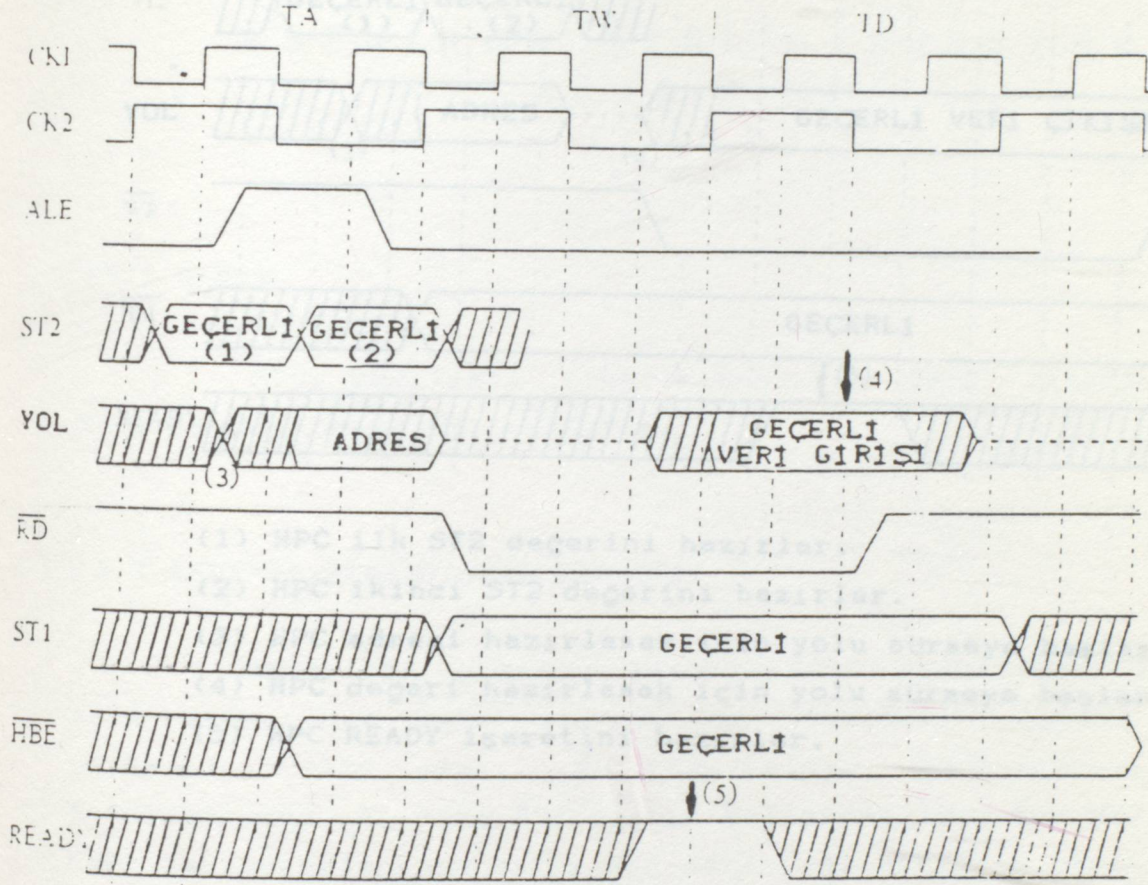


- (1) HPC ilk ST2 değerini hazırlar.
- (2) HPC ikinci ST2 değerini hazırlar.
- (3) HPC adresi hazırlamak için yolu sürmeye başlar.
- (4) HPC değeri hazırlamak için yolu sürmeye başlar.

- (1) HPC ilk ST2 değerini hazırlar.
- (2) HPC ikinci ST2 değerini hazırlar.
- (3) HPC adresi hazırlamak için yolu sürmeye başlar.
- (4) HPC giriş verisinin örnekler.
- (5) HPC READY işlevini hazırlar.

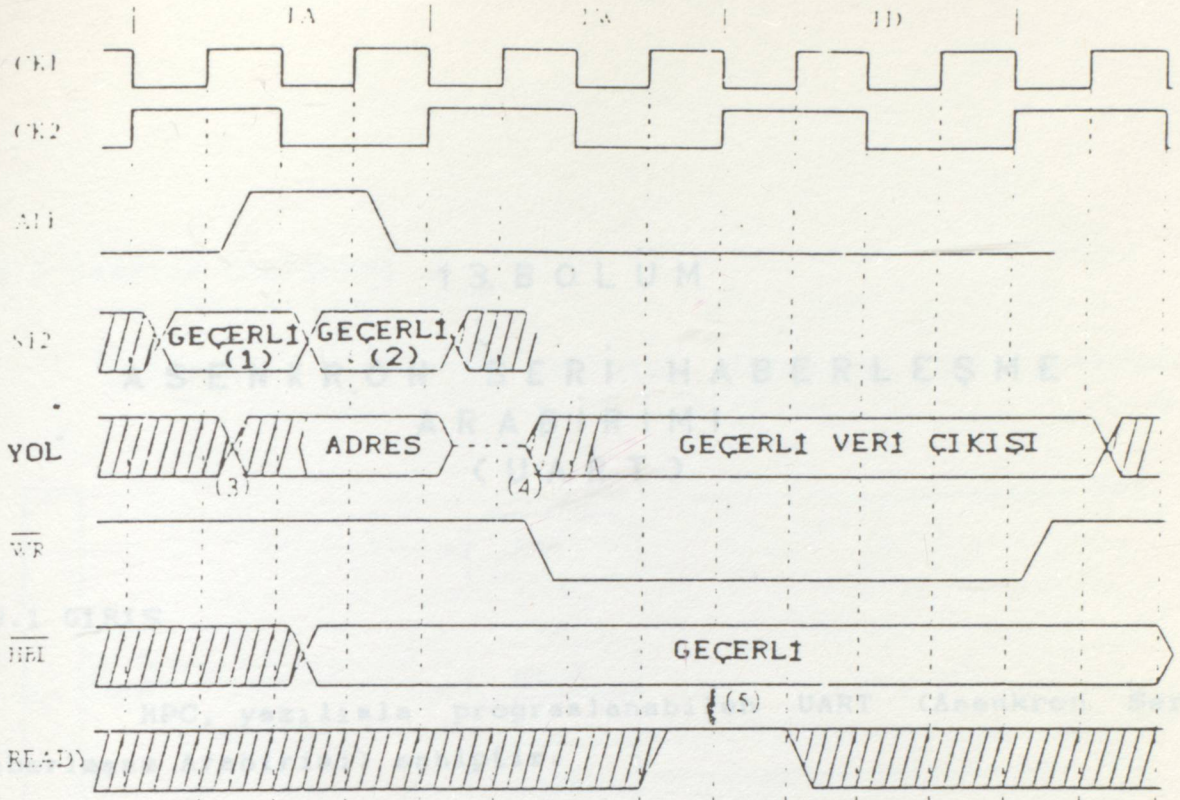
ŞEKİL 12-3 Yol Zamanlaması : Yazma Çevrimi;Bekleme Durumu Yok

ŞEKİL 12-4 Yol Zamanlaması : Okuma Çevrimi;Bekleme Durumu Yok



- (1) HPC ilk ST2 degerini hazirlar.
- (2) HPC ikinci ST2 degerini hazirlar.
- (3) HPC adresi hazirlamak için yolu sürmeye başlar.
- (4) HPC giriş verisini örnekler.
- (5) HPC READY işaretini hazirlar.

ŞEKİL 12-4 Yol Zamanlaması : Okuma Çevrimi; Bir Bekleme Durumlu



(1) HPC ilk ST2 değerini hazırlar.

(2) HPC ikinci ST2 değerini hazırlar.

(3) HPC adresi hazırlamak için yolu sürmeye başlar.

(4) HPC değeri hazırlamak için yolu sürmeye başlar.

(5) HPC READY işaretini hazırlar.

ŞEKİL 12-3 Yol Zamanlaması : Yazma Çevrimi; Bir Bekleme Durumlu

13.3 UART'IN ÇALIŞMASI 13. BÖLÜM

ASENKRON SERİ HABERLEŞME ARABİRİMİ (UART)

13.1 GİRİŞ

HPC, yazılımla programlanabilen UART (Asenkron Seri Haberleşme Arabirimi) sahiptir.

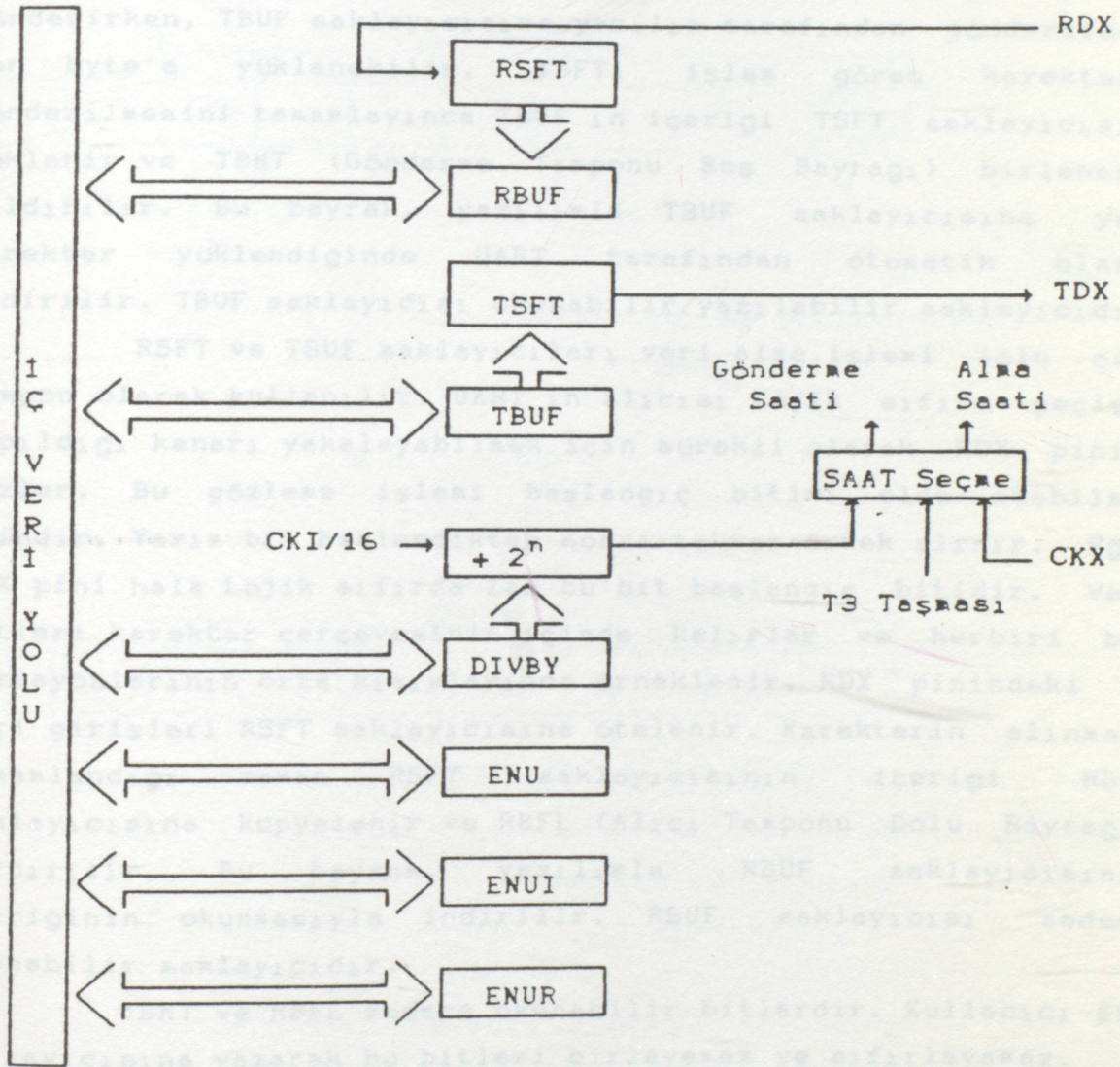
13.2 UART'IN GİRİŞ/ÇIKIŞ PINLERİ:

UART, TDX piniyle haberleşmede bulunduğu kaynaklara veri aktarırken RDX, piniyle de verileri kabul ederken kullanılır. TDX pini, B iskelesinin BO pininin diğer fonksiyonudur. Bu pini RDX olarak kullanmak için hem BFUN hem de DIRB saklayıcılarının sıfırıncı bitlerini birleştirmek gerekir. RDX pini de I iskelesinin doğal fonksiyonuyla belirlenir. RDX için herhangi bir saklayıcıda önceden birleştirmeye gerek yoktur. RDX pininin Lojik seviyesi veri kabul edilmediği sürece Lojik 1 dir.

UART ın haberleşmesinde saatın baud hızı devrede üretilebilir veya bir dış kaynaktan temin edilebilir. Giriş veya çıkış olarak belirlenmesi ise DIRB saklayıcısının 2. bitinin birleşmesiyle belirlenir. Eğer giriş olarak seçilirse veri almak veya göndermek için bir saat işareti kabul edebilir. Eğer çıkış olarak seçilirse UART tarafından kullanılan saat işaretini hazırlar.

13.3 UART'IN ÇALIŞMASI

Sekil 11.1 de UART ın blok diyagramı görülmektedir.



Sekil 13.1 UART ın Blok Diyagramı

UART, gönderme ve alma bölümlerini içerir. TBUF, RBUF, ENU ve ENUI olmak üzere beş adet adreslenebilir saklayıcıya sahiptir. Bu saklayıcılar 8 bitlik saklayıcılardır. Ayrıca RSFT

ve TSFT olarak bilinen iki adet adreslenemeyen saklayıcıya da sahiptir. ENU, ENUR ve ENUI saklayıcıları, UART için kontrol ve durum bitlerini içerir.

TSFT ve TBUF saklayıcıları gönderme için çift tampon olarak kullanılır. TSFT işlem gören karakteri TDX pinine gönderirken, TBUF saklayıcısına yazılım tarafından gönderilecek son byte'a yüklenebilir. TSFT, işlem gören karakterin gönderilmesini tamamlayınca TBUF'ın içeriği TSFT saklayıcısına yüklenir ve TBMT (Gönderme Tamponu Boş Bayrağı) birleştirilerek kaldırılır. Bu bayrak, yazılımla TBUF saklayıcısına yeni karakter yüklendiğinde UART tarafından otomatik olarak indirilir. TBUF saklayıcısı okunabilir/yazılabilir saklayıcıdır.

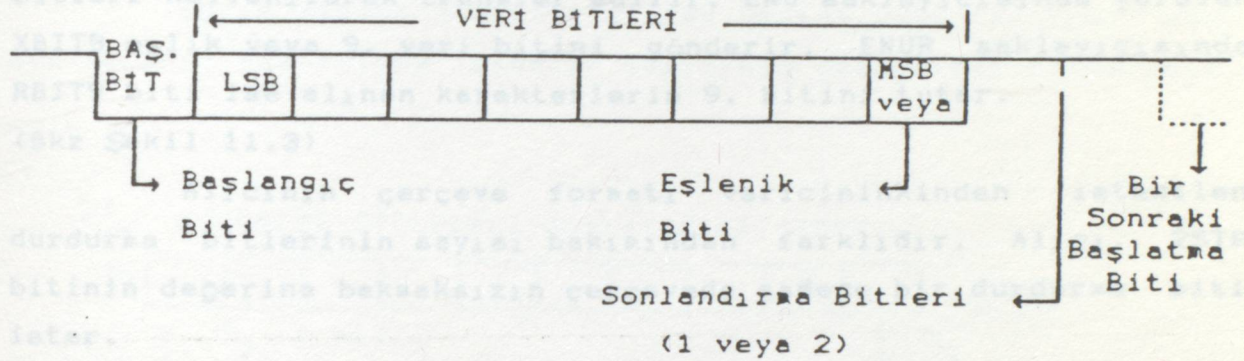
RSFT ve RBUF saklayıcıları veri alma işlemi için çift tampon olarak kullanılır. UART'ın alıcısı Lojik sıfıra geçişin yapıldığı kenarı yakalayabilmek için sürekli olarak RDX pinini gözler. Bu gözleme işlemi başlangıç bitini elde edebilmek içindir. Yarım bit beklendikten sonra tekrar örnek alınır. Eğer RDX pini hala Lojik sıfırda ise bu bit başlangıç bitidir. Veri bitleri karakter çerçevesinin içinde kalırlar ve herbiri bit pozisyonlarının orta kısımlarında örneklenir. RDX pinindeki bu veri girişleri RSFT saklayıcısına ötelenir. Karakterin alınması tamamlandığı zaman RSFT saklayıcısının içeriği RBUF saklayıcısına kopyelenir ve RBFL (Alıcı Tamponu Dolu Bayrağı) kaldırılır. Bu bayrak, yazılımla RBUF saklayıcısının içeriğinin okunmasıyla indirilir. RBUF saklayıcısı sadece okunabilir saklayıcıdır.

TBMT ve RBFL sadece okunabilir bitlerdir. Kullanıcı ENU saklayıcısına yazarak bu bitleri birleyemez ve sıfırlamaz.

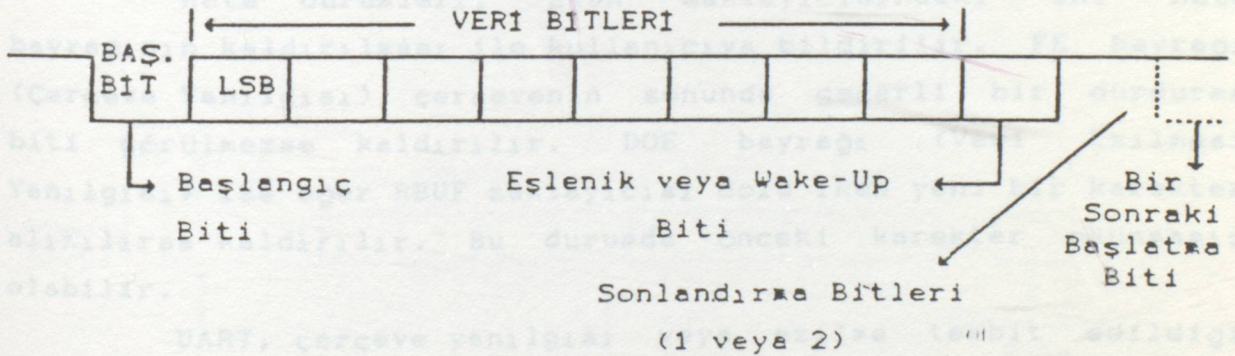
NOT: UART, HALT veya IDLE güç koruma modları girildiği zaman otomatik olarak silinir. Gönderme çıkışları işlem gören durumlarında dondurulur ve bütün kontrol saklayıcılarının tekrardan başlatma durumlarında olduğu farzedilir.

13.4 ÇERÇEVE FORMATLARI

UART iki çeşit çerçeve formatı kullanabilir. Bu formatlar ENU saklayıcısındaki 80R9 bitiyle belirlenir.



8 BITLİK MOD



9 BITLİK MOD

Birinci format, transfer edilen veri için bir başlangıç bit, 8 veri bit ve bir veya iki sonlandırma bit içerir. Durdurma bitinin sayısı ENUI saklayıcısının 2STP bitiyle belirlenir.

İkinci format, transfer edilen veri için bir başlangıç

1. FE ve DOE bayrakları kontrol edilmeden önce değeri
biti, 9 veri biti ve bir veya iki sonlandırma biti içerir. Bu format eşlik bitli 8 bitlik transferi gerçekleştirir ve UART ın "Wake Up" modunda kullanılır. Bu formatla işlem yapılarak TBUF ve RBUF saklayıcılarının 8. biti veri için kullanılır. 9. veri biti (veya eşlik biti) kontrol saklayıcılarındaki XBIT9 ve RBIT9 bitleri kullanılarak transfer edilir. ENU saklayıcısında yer alan XBIT9 eşlik veya 9. veri bitini gönderir. ENUR saklayıcısında RBIT9 biti ise alınan karakterlerin 9. bitini tutar.
(Bkz Şekil 11.3)

Alıcının çerçeve formatı vericinininkinden istenilen durdurma bitlerinin sayısı bakımından farklıdır. Alıcı, 2STP bitinin değerine bakmaksızın çerçevede sadece bir durdurma bitini ister.

13.5 HATA BAYRAKLARI

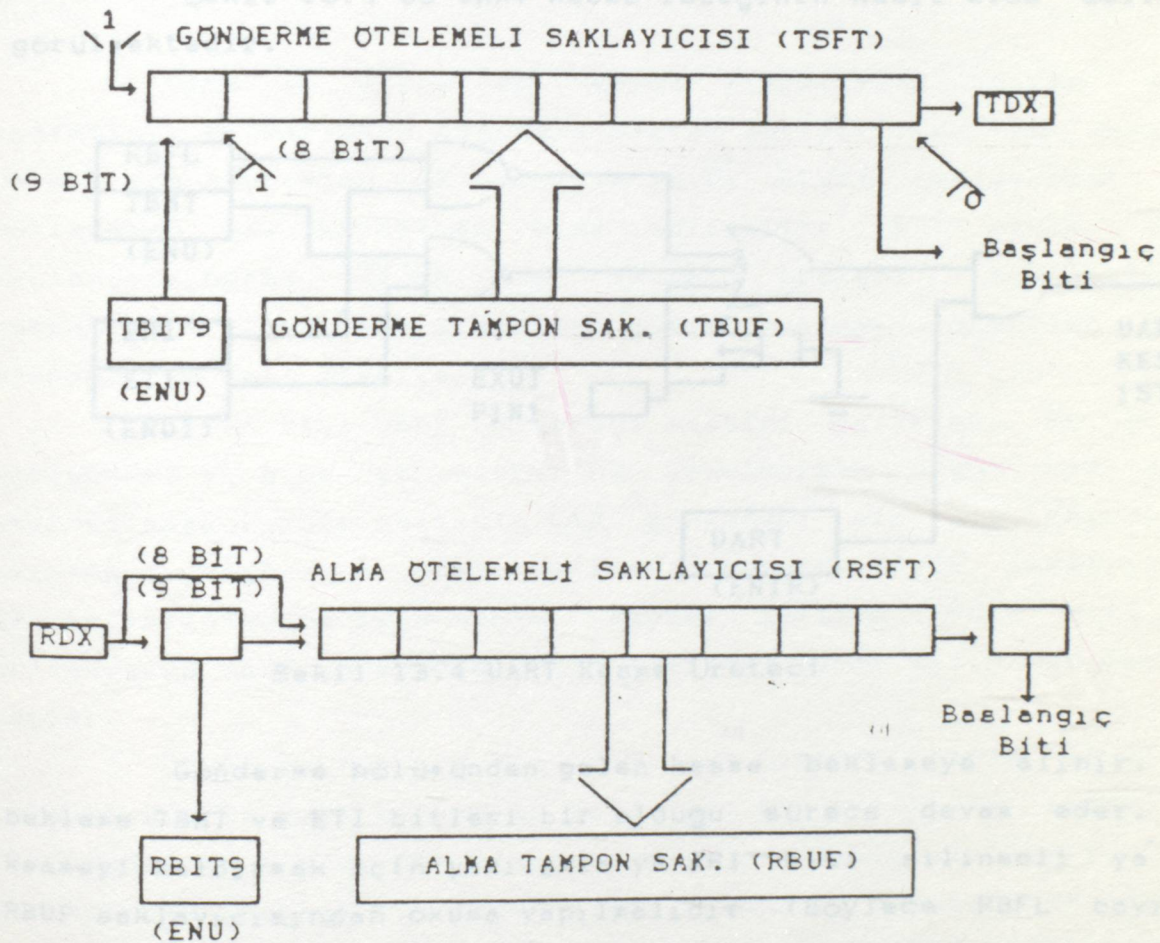
Hata durumları, ENUR saklayıcısındaki iki hata bayrağının kaldırılması ile kullanıcıya bildirilir. FE bayrağı (Çerçeve Yanılgısı), çerçevenin sonunda geçerli bir durdurma biti görülmezse kaldırılır. DOE bayrağı (Veri Ezilmesi Yanılgısı) ise eğer RBUF saklayıcısı dolu iken yeni bir karakter alınılırsa kaldırılır. Bu durumda önceki karakter okunmamış olabilir.

UART, çerçeve yanılgısı veya ezilme tesbit edildiği zaman bile RBUF saklayıcısının aldığı herbir karakteri güncelleştirmeye devam eder.

FE ve DOE bitleri kullanıcı için sadece okunabilir. Bu bitler, yazılımla birkez okunduğu zaman veya UART reseti yapıldığı zaman silinir.

Bu bayrakların kontrol edilmesiyle, onların silinmesi arasında zaman olarak fark yoktur. Bu nedenle ENUR saklayıcısına ulaşmak için şu iki yol tercih edilebilir.

1. FE ve DOE bayrakları kontrol edilmeden veya deęerleri kaydedilmeden ENUR saklayıcısı okunmamalıdır.
 2. ENUR saklayıcısına hiçbir zaman okuma-düzeltilme-yazma işlem sırası doğrudan gerçekleştirilmemelidir. Çünkü bu işlem DOE ve FE bayraklarını herhangi bir yere kaydetmeksiz okur ve indirir.
- Eşlik bitini otomatik olarak üreten veya kontrol eden bir donanım yoktur. Bu işlemler yazılımla gerçekleştirilmelidir.

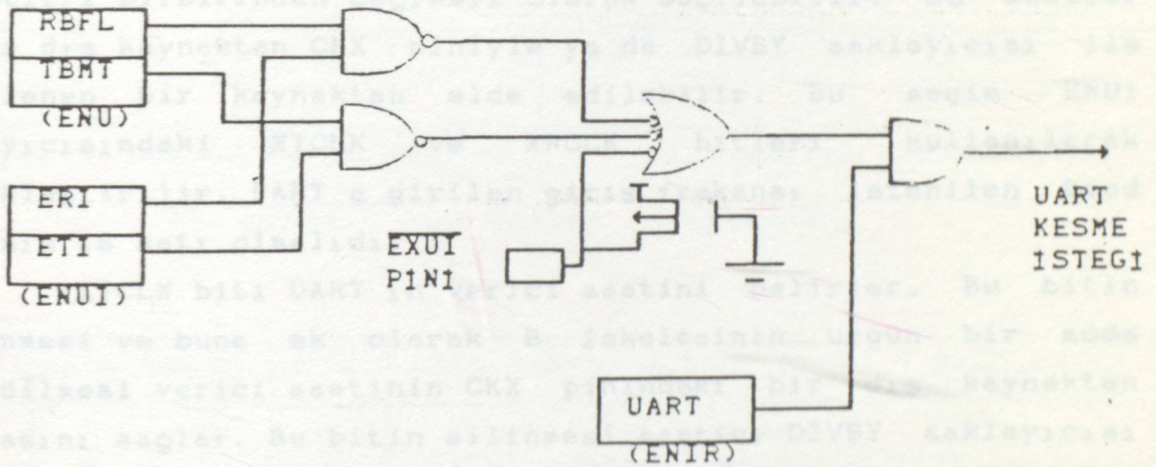


Şekil 13.3 TSFT/RSFT ve Diğer Saklayıcılar Arasındaki Veri Akışı

13.6 KESMELER

UART kesme istegi üretme kapasitesine sahiptir. Alma veya gönderme kesmelerine birbirinden bağımsız olarak izin verilebilir veya bu izin kaldırılabilir. Bu izinler ENUI saklayıcısının ETI ve ERI bitleriyle belirlenir. Eğer bu iki kesmeye izin verilirse kesme hizmet programları ENU saklayıcısındaki RBFL ve TBMT bayrakları yardımıyla kesme kaynağını belirlemelidir.

Şekil 13.4 de UART kesme isteginin nasıl elde edildiği görülmektedir.



Şekil 13.4 UART Kesme Üretici

Gönderme bölümünden gelen kesme bekleme alınır. Bu bekleme TBMT ve ETI bitleri bir olduğu sürece devam eder. Bu kesmeyi kaldırmak için yazılımla ya ERI bitini silinmeli ya da RBFL saklayıcısından okuma yapılmalıdır (Böylece RBFL bayrağı indirilir). UART kesmelerinin izinleri ENIR saklayıcısındaki UART bitleri silinerek kaldırılabilir.

UART kesmesi EXUI pini kullanılarak da elde edilebilir. Bu pin düzey duyarlı pin olup kesme istegi

girişidir. Lojik sıfırda etkindir. EXUI pini, HPC nin UART a ihtiyacı olmadığı fakat ek bir kesme girişine ihtiyacı olduğu uygulamalarda basitlik sağlar. EXUI den gelen kesme isteği, pini Lojik bire çekerek veya ENIR saklayıcısının UART bitini silerek kaldırılabilir. Yonga'daki pull-up devresi, eğer dışarıdan bir bağlama yapılmamışsa bu pini Lojik bir yani etkin olmayan seviyede tutmaya yarar.

13.7 SAATLEME

UART in alıcı ve verici kısımlarına giren saat işaretleri birbirinden bağımsız olarak seçilebilir. Bu saatler ya bir dış kaynaktan CKX piniyle ya da DIVBY saklayıcısı ile belirlenen bir kaynaktan elde edilebilir. Bu seçim ENUI saklayıcısındaki XTCLK ve XRCLK bitleri kullanılarak gerçekleştirilir. UART a girilen giriş frekansı istenilen Baud oranının 16 katı olmalıdır.

XTCLK biti UART in verici saatini belirler. Bu bitin birleşmesi ve buna ek olarak B iskelesinin uygun bir moda kaydedilmesi verici saatinin CKX pinindeki bir dış kaynaktan alınmasını sağlar. Bu bitin silinmesi saatin, DIVBY saklayıcısı ile belirlenmesine izin verir. Benzer şekilde XRCLK biti de kullanıcıya UART in alıcısı için saat kaynağını belirleme imkanı sağlar.

DIVBY saklayıcısının 4-9 cu bitleri tablo 13.1 de görüldüğü gibi saat kaynağını belirler. Bu belirleme işleminin yapılabilmesi için saat CKX pininden elde edilmemiş olmalıdır. HPC nin kristal frekansının önceden belirlenen 14 durumuna ek olarak T3 zamanlayıcısının alttan taşmasıyla da saat elde edilebilir. Herbir taşmada bir çevrim elde edilir. Bununla birlikte her saat işareti CKX pininde çıkış olarak seçilirse varolan saat işareti asimetriktir. İşaretin Lojik sıfır seviyesi CKI saatinin sekiz

çevriminde sabitleştirilir.

UART ın alıcı ve vericisinin farklı oranlardaki baud hızında işlem yapması mümkündür. Bu, birini CKX pininden diğerini de DIVBY saklayıcısı ile belirlenen bir iç kaynaktan elde ederek gerçekleştirilebilir. T3 ü bir dış kaynaktan saatleyerek (T3IO, B4 pini) ve DIVBY saklayıcısında T3 ü bir saat kaynağı olarak tanımlayarak iki UART bölümünü birbirinden bağımsız dış frekanslarla çalıştırma imkanı sağlanabilir. T3 ve R3 saklayıcılarına sıfır yerleştirme, T3IO frekansını bire bölmeyi gerçekleştirir. Başka bir değer yerleştirme, örneğin n , dış kaynağın $n+1$ değeriyle önceden belirlenmesine neden olur.

T3	T2	UART	MICROWIRE
018F Bellek Adresindedir		018E Bellek Adresindedir	

DIVBY - 018E Bellek Adresindedir

Bit7	Bit6	Bit5	Bit4	BAUD SAATI	BAUD ORANI	BAUD ORANI
0	0	0	0	< izin verilmez >		
0	0	0	1	<T3'un Altından Taşmasıyla Elde Edilir>		
0	0	1	0	CKI/16	38400	65536
0	0	1	1	CKI/32	19200	32768
0	1	0	0	CKI/64	9600	16384
0	1	0	1	CKI/128	4800	8192
0	1	1	0	CKI/256	2400	4096
0	1	1	1	CKI/512	1200	2048
1	0	0	0	CKI/1024	600	1024
1	0	0	1	CKI/2048	300	512
1	0	1	0	CKI/4096	150	256
1	0	1	1	CKI/8192	75	128
1	1	0	0	CKI/16384	38	64
1	1	0	1	CKI/32768	19	32
1	1	1	0	CKI/65536	9.4	16
1	1	1	1	CKI/131072	4.7	8

TABLO 13.1- DIVBY Saklayıcısından Elde Edilen Saat Kaynakları

13.8 WAKE-UP MODU

UART ın alıcısı başka bir mod işlemi de gerçekleştirir. Bu mod Wake-Up modudur. Bu modda işlem yapmak için ENUR saklayıcısının WAKEUP biti birleşmelidir. Transfer için veri formatı dokuz veri biti bir veya iki sonlandırma biti şeklinde seçilmelidir.

Wake-Up modu işlemleri HPC ile diğer prosesörler arasındaki network haberleşmelerinde kullanılır. Böyle haberleşmelerde "mesajlar", "header" (cihaz adresi) ve bunu izleyen "body" (gerçek veri) yi içerir. Bu modda her bir cihazın bir numarası olmalıdır. "Header" kısmı mesajın "body" kısmındaki veriyi hangi cihazın alacağını belirler. Wake-Up modu şöyle bir yol izler. Dokuz bitlik veri alanını birleyerek sekiz bitlik "header" alanının güncelleştirilmesini sağlar. "Mesajın" "body" kısmı sekiz byte'lık veriler içerir. Veri alanlarının dokuzuncu bitleri sıfırlanır.

Wake-Up modda iken HPC nin UART ı haberleşme akışını kontrol eder. Fakat adres karakteri alınana kadar gelen bütün karakterleri önemsemez. Bir adres karakteri alındığında, UART RBFL bayrağını kaldırarak karakterin hazır olduğunu bildirir. Eğer UART alıcısının kesmelerine izin verilirse bu işaret prosesörlere kesme olarak döner. WAKEUP biti bu noktada otomatik olarak silinebilir. Bunun için veri karakterleri adres karakterlerinde olduğu gibi iyi bir şekilde tanınmalıdır.

Yazılımla RBUF saklayıcısının içeriği kontrol edilir ve buna göre ya bir sonraki verinin kabul edilmesine (WAKEUP biti silinmiş bırakılarak) ya da gelecek adres karakterinin görünmesine kadar beklemeye (WAKEUP biti tekrar birleşerek) karar verilebilir.

UART vericisinin işlemleri Wake-Up modunun seçilmesinden etkilenmez. Gönderilen dokuzuncu bitin değeri, ENU saklayıcısındaki XBIT9 bitine uygun değer kaydedilerek

programlanabilir. Alınan dokuzuncu bitin değeri ENUF saklayıcısının RBIT9 biti okunarak elde edilebilir.

13.9 UART'IN KONTROL SAKLAYICILARI

UART in işlemleri ENU , ENUR ve ENUI saklayıcılarını programlanmasıyla gerçekleştirilir. Bu saklayıcıların bitlerini fonksiyonları şu şekildedir.

ENU: UART in kontrol ve durum saklayıcısıdır.

*	*	XBIT9 0	8OR9 0	*	*	RBFL OR	TBMT 1R
---	---	------------	-----------	---	---	------------	------------

0120- Bellek Adresinde Bulunur

ENUR: UART in alıcı kontrol ve durum saklayıcısıdır.

DOE OD	FE OD	*	*	RBIT9 OR	WAKEUP 0	*	*
-----------	----------	---	---	-------------	-------------	---	---

0128- Bellek Adresinde Bulunur

ENUI: UART in kesme ve saat kaynağı saklayıcısıdır.

2STP 0	*	*	*	XRCLK 0	XTCLK 0	ERI 0	ETI 0
-----------	---	---	---	------------	------------	----------	----------

0122- Bellek Adresinde Bulunur

1:Reset durumunda birleşir.

0:Reset durumunda sıfırlanır.

R:Sadece okunabilir bittir yazılımcı tarafından yazılmaz. Bu bit UART karakterin alması işlemini tamamlayıp onu

D:Okuma işlemiyle silinir. Yazılımla bir kez okundugunda silinir. Bu bite yazma durumunu etkilemez. Uygulanması

•:Kullanılmayan bit. Belirsiz veri içerebilir.

TSMT Bu bit, UART gönderilecek Veri TSMT saklayıcısından

NOT: Bu saklayıcıların bitlerinin içerikleri, Reset durumunda veya HPC Halt veya Idle moda girdiğinde başlangıç durumlarında olarak kabul edilir.

ENI Alıcının kabul üretilmesine izin verir veya bu izin

13.9.1 UART Saklayıcı Bitlerinin Detaylı Olarak Tanıtımı:

ENI=0: Alıcının kabul üretilmesi yasaklanır.

FE Çerçeve yanlışlığı bayrağı. Bu kabul üretilmesine izin
FE=1 : Çerçeve yanlışlığının olduğunu gösterir.

FE=0 : ENUR saklayıcısının en son kez okunmasından bu yana çerçeve yanlışlığının olmadığını gösterir.

DOE Veri ezilme yanlışlığı bayrağı. kabul üretilmesine izin
DOE=1 : Veri Veri ezilme yanlışlığı olduğunu gösterir.

DOE=0 : ENUR saklayıcısının en son kez okunmasından bu yana veri ezilmesi hatasının olmadığını gösterir.

RBIT9 UART, her çerçevede dokuz bitlik veri içeren modunda işlem yaptığından, alınan dokuzuncu biti içerir.

XBIT9 UART, her çerçevede dokuz bitlik veri içeren modunda işlem yaptığından gönderilecek olan dokuzuncu biti ayarlar.

BOR9 Karakter çerçevesinin formatını belirler

BOR9=1: Çerçeve sekiz bit veri içerir.

BOR9=0: Çerçeve dokuz bit veri içerir.

WAKEUP Bu bitin birleşimiyle Wake-Up modu seçilir. Alınan bir karakterde dokuzuncu bite değer kaydedilmesiyle bu bit

- otomatik olarak silinir.
- RBFL** Bu bit UART karakterin alma işlemini tamamlayıp onu RBUF saklayıcısına kopyelediği zaman birlenir. HPC, RBUF saklayıcısından karakteri okudugu zaman otomatik olarak silinir.
- TBMT** Bu bit, UART gönderilmek üzere TBUF saklayıcısından TSFT saklayıcısına veri aktardığı zaman birlenir. Yazılımla TBUF saklayıcısına veri yazılmasıyla otomatik olarak silinir.
- ERI** Alıcının kesme üretmesine izin verir veya bu izni kaldırır.
ERI=0: Alıcının kesme üretmesi yasaklanır.
ERI=1: Alıcının kesme üretmesine izin verilir.
- ETI** Vericinin kesme üretmesine izin verir veya bu izni kaldırır.
ETI=0: Vericinin kesme üretmesi yasaklanır.
ETI=1: Vericinin kesme üretmesine izin verilir.
- XRCLK** Alıcının saat kaynağını belirler.
XRCLK=0: Saat kaynağı, DIVBY saklayıcısı tarafından belirlenir.
XRCLK=1 CKX pinindeki işaret saat olarak kullanılır.
- XTCLK** Vericinin saat kaynağını belirler.
XTCLK=0 Saat kaynağı, DIVBY saklayıcısı tarafından belirlenir.
XTCLK=1 CKX pinindeki işaret saat olarak kullanılır.
- 2STP** Gönderilecek olan sonlandırma bitlerinin sayısını belirler.
2STP=0: Sonlandırma biti bir bittir.
2STP=1: Sonlandırma biti iki bittir.

14. BÖLÜM

RESET DURUMU

RESET pininin lojik 0 ' a bağlanmasıyla HPC ' nin içerdigi lojik ,işleminin durumunu başlangıç haline getirir . CK2 saatinin 16 periyodu süresince bu pin , lojik 0 'da bağlı kalmalıdır .

RESET pini , yüksek empedans Schmitt tetikleme giriş pinidir . HPC, resetlendiğinde bir kesme üreterek işleminin FFFE adresine dallanması sağlanır .

Resetin dallanma hedefi , yonga içindeki adres uzayında olmalıdır . Çünkü HPC , tek yonga modunda çalışabilir .

HBE pini (B12) , RESET 'in yükselen kenarında örneklenir ve HPC , 16-bit veya 8-bit moduna girer . Eğer bu pin bire çekilmiş ise HPC , 8-bit moduna girecektir .

Resette , HPC aşağıdaki ilk durumları alır :

- * Yığın göstergesinin içeriği sıfırdır . Push komutunun yürütülmesinden sonra SP ' nin içeriği 0002 dir .(Push , 0000 ve 0001 adresine yazmıştır) .

- * PSW saklayıcısındaki EHI , HLT/IDL , EA , WAIT0 ve WAIT1 bitleri sıfırlanır .

- * Port A ve B ' nin yönlendirici saklayıcıları sıfırlanır .

- * BFUN saklayıcısının bütün pinleri sıfırlanır , Port B pinlerinin hepsi G/Ç fonksiyonu olarak seçilir .

- * UPIC saklayıcısındaki UPIEN ve WRRDY bitleri sıfırlanır . RDRDY biti ise birleşir .

- * ENIR saklayıcısındaki , Global Kesme İzni (GIE) biti sıfırlanır . Bütün maskelenebilir kesmeler yasaklanır . Diğer kesme izin bitlerinin hiçbiri etkilenez .

15. BÖLÜM

* IRCD saklayıcısındaki RDY/HLD bitleri sıfırlanır . RDY/HLD pini için HOLD fonksiyonu seçilir . uWMODE bitleri sıfırlanır , MICROWIRE arabirimi için uydu modu seçilir . Diğer bitler etkilenmez .

* OODC adresindeki HALT izin saklayıcısı sıfırlanır .

* Zamanlayıcı TO sıfıra resetlenir . Watchdog lojisi resetlenir . Diğer zamanlayıcılar etkilenmez .

* Port P saklayıcısı sıfırlanır .

* UART saklayıcıları , ENU , ENUI ve ENUR sıfırlanır .

doğrudan bir çevre birimi olarak kullanılabileceğine izin verir. Diğer işlemci bütün veri değişimlerini başlattığından, burada Ana İşlemci olarak adlandırılacaktır.

UPI arabirimi bir Veri Yolu (UDS:8 veya 16 bit genişliğinde), iki veri belirtme girişi (URD ve URW), iki çıkış çıkışı (URRDY ve URRDY) ve bir Adres girişi (UAD) içerir.

UPI Nedir? Port A UPI Veri Yolu'dur. Port B'nin B10, B11 ve B15 pinleri sırasıyla UAD girişi ve URRDY ve URRDY çıkışlarıdır. Port P'nin P2 ve P3 pinleri sırasıyla URD ve URW girişleri olur ve MPC'nin işlemlerini koordine kabiliyetine sahiptirler.

15.2 UPI ORGANİZASYONU

Sekil 15-1 UPI arabiriminin iç organizasyonunu gösterir. Arabirim bir kontrol saklayıcısı (UPIC) içerir ve PORTA ve DIRA'yı tampon saklayıcılar olarak kullanır. PORTA saklayıcısı (0020 adresinde) Ana İşlemci tarafından okunması için verinin tutulduğu çıkış tampon saklayıcısı olarak kullanılır. Burada PORTA saklayıcısı 0020 olarak adlandırılacaktır. DIRA saklayıcısı (0020 adresinde) Ana

15.BÖLÜM

GENEL ÇEVRE ARABİRİMİ (UPI)

15.1 GİRİŞ

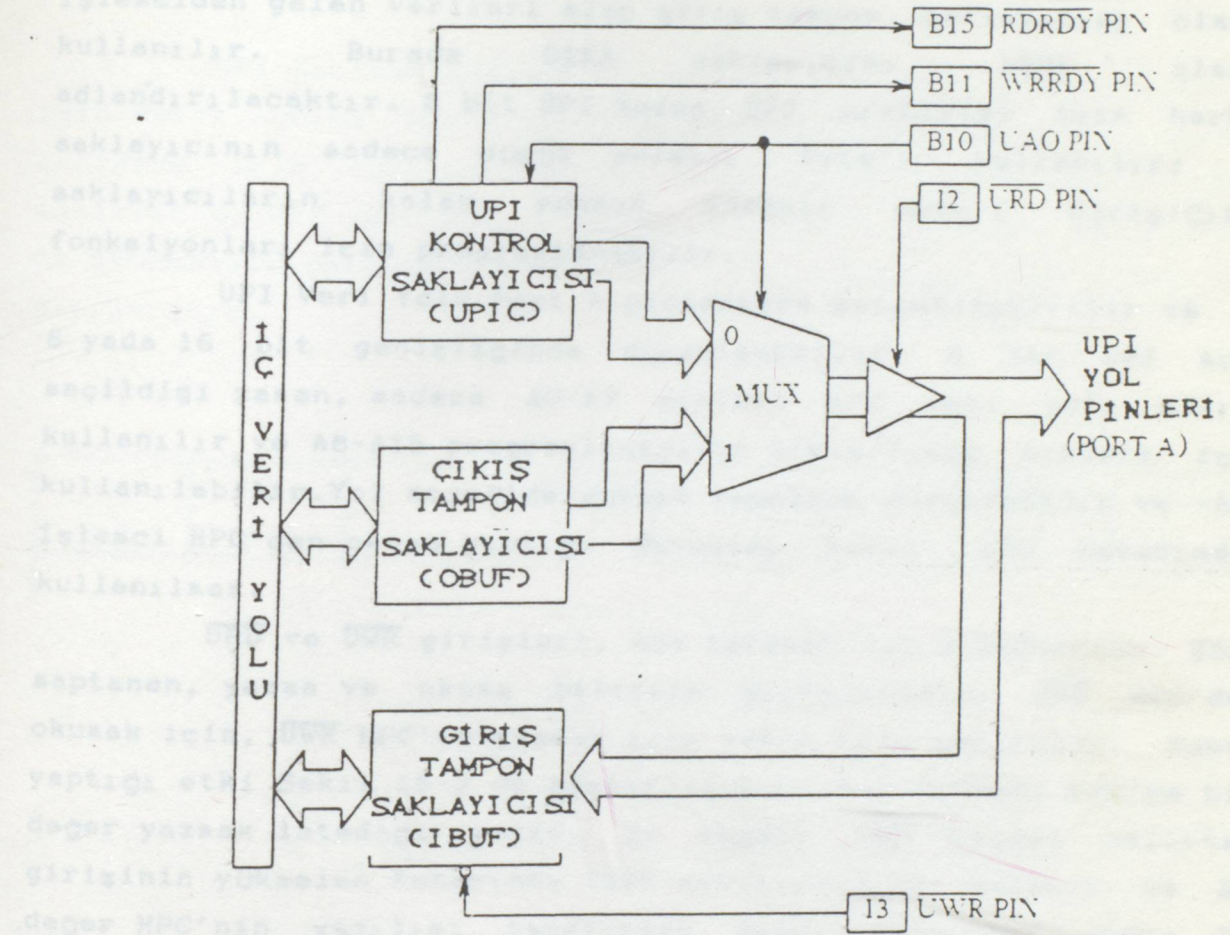
Genel Çevre Arabirimi (UPI) özelliği; HPC'nin diğer işlemci sistemleri tarafından, bir paralel yol üzerinden doğrudan bir çevre birimi olarak kullanılmasına izin verir. Diğer işlemci bütün veri değişimlerini başlattığından, burada Ana İşlemci olarak adlandırılacaktır.

UPI arabirimi bir Veri Yolu (UDB:8 veya 16 bit genişliğinde), iki veri belirtme girişi (URD ve UWR), iki el sıkışma çıkışı (RDRDY ve WRRDY) ve bir Adres girişi (UAO) içerir.

UPI Modunda, Port A UPI Veri Yolu olur. Port B'nin B10, B11 ve B15 pinleri sırasıyla UAO girişi ve RDRDY ve WRRDY çıkışları olur. Port I'nın I2 ve I3 pinleri sırasıyla URD ve UWR girişleri olur ve HPC'nin işlemlerini kesme kabiliyetine sahiptirler.

15.2 İÇ ORGANİZASYONU

Şekil 15-1 UPI arabiriminin iç organizasyonunu gösterir. Arabirim bir kontrol saklayıcısı (UPIC) içerir ve PORTA ve DIRA'yi tampon saklayıcılar olarak kullanır. PORTA saklayıcısı (OOFO adresinde) Ana İşlemci tarafından okunması için verinin tutulduğu çıkış tampon saklayıcısı olarak kullanılır. Burada PORTA saklayıcısı OBUF olarak adlandırılacaktır. DIRA saklayıcısı (OOEO adresinde) Ana



Şekil 15.1 UPI Blok Diyagramı

15.3 UPI KONTROL SAKLAYICISI

İşlemciden gelen verileri alan giriş tampon saklayıcısı olarak kullanılır. Burada DIRA saklayıcısı IBUF olarak adlandırılacaktır. 8 bit UPI Modda, UPI arabirimi için her bir saklayıcının sadece düşük anlamlı byte'ı kullanılır; bu saklayıcıların kalan yüksek anlamlı byte'ı Giriş/Çıkış fonksiyonları için programlanabilir.

UPI Veri Yolu, Port A pinlerinde gerçekleştirilir ve ya 8 yada 16 bit genişliğinde düzenlenebilir. 8 bit UPI modu seçildiği zaman, sadece AO-A7 pinleri UPI veri yolu olarak kullanılır ve A8-A15 programlanabilir Giriş/Çıkış bitleri için kullanılabilir. Yol normalde yüksek empedans durumundadır ve -Ana İşlemci HPC'den okuma yaptığı durumlar hariç- HPC tarafından kullanılmaz.

URD ve UWR girişleri, Ana İşlemci'nin standardına göre saptanan, yazma ve okuma belirtme girişleridir: URD HPC'den okumak için, UWR HPC'ye yazmak için etkin hale getirilir. Bunun yaptığı etki Şekil 15-2 de gösterilmiştir. Ana İşlemci HPC'ye bir değer yazmak istediği zaman, bu değeri UWR yazma belirtme girişinin yükselen kenarında IBUF saklayıcısına aktarır ve Bu değer HPC'nin yazılımı tarafından değerlendirilmek üzere bu saklayıcıda kalır. HPC'nin yazılımı OBUF saklayıcısına bir değer yazdığı zaman, Bu değer Ana İşlemci için orada tutulur. Ana İşlemci URD pinini low seviyesine getirdiği zaman, veri yola çıkarılır. Bu veri ya OBUF yada UPIC saklayıcısından gelebilir. Hangi saklayıcıdan geldiği Ana İşlemcinin UAO pini kullanılarak belirlenir.

UPI arabirimi, HPC'nin adres seçiminin Ana İşlemci tarafından yapılmasına izin vermek için eleman seçme lojigi içermez.

15.3 UPI KONTROL SAKLAYICISI

UPI Kontrol saklayıcısı (UPIC) UPI modunun girilmesine izin verir. Ayrıca kesme-sürülmüş el sıkışma işlemleri kadar seçimine izin verilmiş durum bilgilerini de içerir. UPIC saklayıcısı HPC yazılımıyla olduğu kadar Ana İşlemci tarafından da okunabilir. Aşağıda bitlerin tek tek işlevsel tanımlamaları yapılmıştır.

0	0	0	8/16	UPIEN	LAO	RDRDY	WRDY
X R	X R	X R	X	0	X R	1 R	0 R

UPIC : OOE6'daki byte'te

- 0 Bit resette sıfırlanır.
- 1 Bit resette birleşir.
- X Bitin durumu resette tanımsızdır.
- R Bit yazılım için sadece okunabilir.

BIT

ISLEV

WRDY

Durum biti. Bu bit UPI arabiriminden aktif seviyesi low olan bir çıkıştır. Ana İşlemcinin Giriş Tampon Saklayıcısına (IBUF) bir veri yazabileceğini gösterir. Bu bit normalde low seviyededir ve Ana İşlemci IBUF'a veri yazdığı zaman high seviye çıkar. HPC IBUF'tan veriyi okuduğu zaman, bit tekrar low seviyeye iner. Bu bit Ana İşlemci için durum işareti olarak Port B'nin B11 pininde de görülür.

RDRDY

Durum biti. Bu bit UPI arabiriminden aktif seviyesi low olan bir çıkıştır. Ana İşlemcinin Çıkış Tampon Saklayıcısından bir veri okuyabileceğini gösterir. Bu bit normalde high seviyededir, ve HPC çıkış tamponuna veri yazdığı zaman low seviyeye iner. Ana İşlemci çıkış tamponunu okudugunda tekrar high seviyeye döner. Bu bit Ana İşlemci için bir durum işareti olarak Port B'nin B15 pininde de görülür.

LAO

Bu bit UWR'nin yükselen kenarında Adres Giriş pini UAO'ı tutar. Bu Ana İşlemci tarafından yazılan fazladan bir veri biti etkisindedir ve örneğin HPC'ye gönderilen veri ve komutları ayırmak için kullanılabilir.

UPIEN

UPI Modunun etkin hale getirilmesinde kullanılır.

UPIEN = 1

UPI Modu etkinleştirilir.

UPIEN = 0

UPI Modu kaldırılır.

8 / 16

8 yada 16 bitlik UPI modunu seçer.

8 / 16 = 1 UPI veri yolu 8 bit genişliğinde.

8 / 16 = 0 UPI veri yolu 16 bit genişliğinde.

0

Bu bit yerleri OBUF saklayıcısından ilgili bitlerle mevcuttur. UPIC saklayıcısı üzerinden erişildiği zaman bu bitler sadece okunabilir.

UPIC saklayıcısı 16 bitlik bir değer olarak ya HPC yada Ana İşlemci tarafından okunduğu zaman, bütün fazlalık bitler OBUF'tan gelir. Böylece karışık veri ve durum aktarımının aynı anda yapılması mümkündür. Bununla birlikte UPIC saklayıcısına erişim **WRDY** ve **RDRDY** bayraklarına etki yapmayacağından dikkatli olmak gerekir.

15.4 PIN FONKSİYONLARININ DETAYLARI

Şekil 15-2 UPI okuma ve yazma işlemleri sırasında oluşan zamanlama sekanslarını gösterir.

15.4.1 UPI Yazma İşlemi

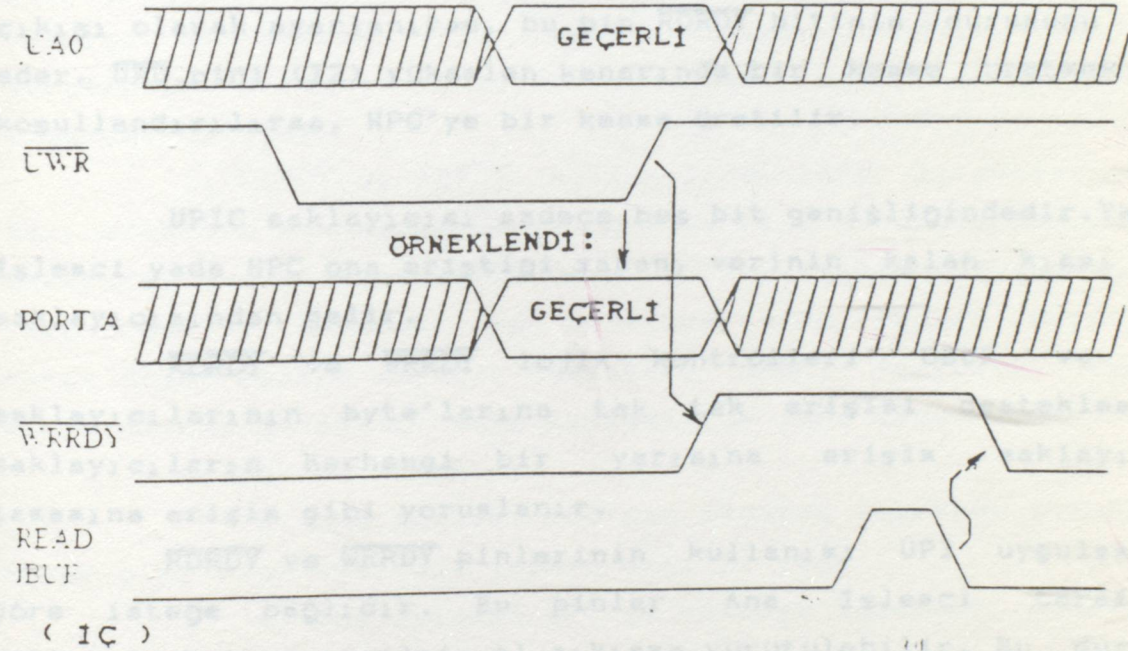
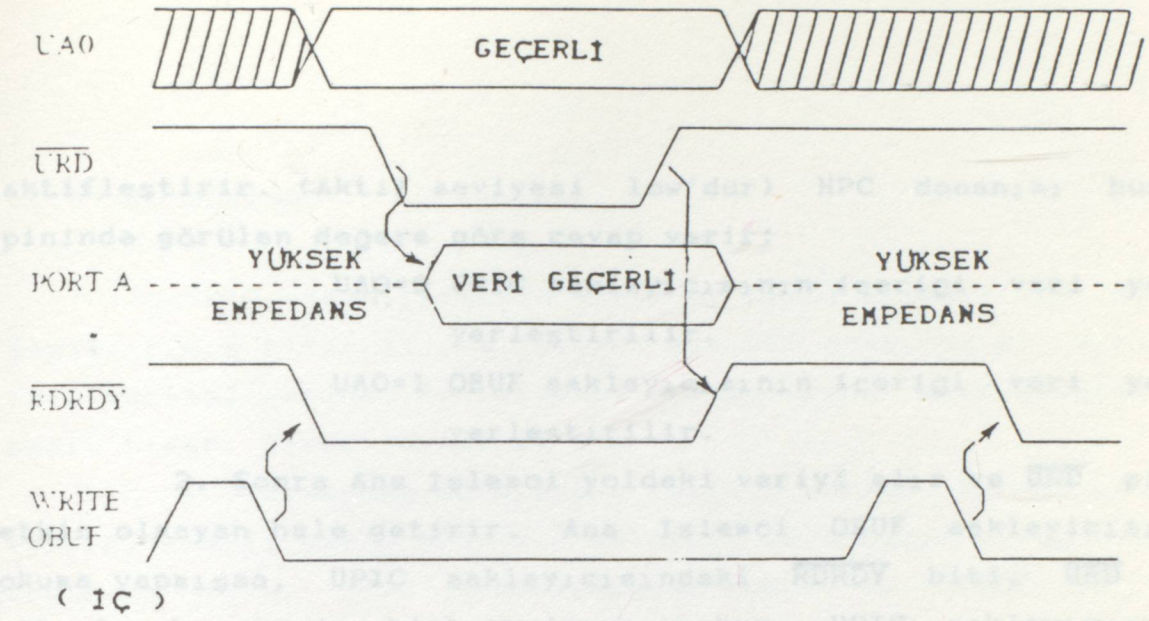
Ana işlemci HPC'ye veriyi yola koyarak ve UWR hattını low seviyeye getirerek yazma yapar. UWR'nin yükselen kenarı sırasında dört olay meydana gelir:

1. Veri yolunun içeriği IBUF saklayıcısında saklanır.
2. UAO (B10) pini örneklenir ve değeri UPIC saklayıcısının LAO bitine yerleştirilir. UAO pini UPI yazma transferlerinde başka bir amaç için kullanılmaz. Örneğin HPC'deki yazılımla komutlarla verileri ayırmak için UPIC saklayıcısındaki LAO biti kullanılabilir.
3. UWR pini (I3) yükselen kenarda kesme tetiklemek için koşullandırılırsa, HPC'ye bir kesme üretilir.
4. UPIC saklayıcısındaki WRRDY biti normalde low seviyededir. IBUF saklayıcısının dolu olduğunu göstermek için high seviyeye çıkar. Eğer Port B'nin B11 pini bir WRRDY çıkışı olarak düzenlenirse, bu anda bu çıkışta high seviyeye çıkar. WRRDY biti, HPC IBUF saklayıcısını okuduğu zaman tekrar low seviyeye iner.

15.4.2 UPI Okuma İşlemi

Bir UPI okuması, genellikle HPC yazılımının OBUF saklayıcısına bir değer yazmasıyla başlar. Normalde high seviyesinde olan UPIC saklayıcısındaki RDRDY biti, OBUF saklayıcısının dolu olduğunu göstermek üzere low seviyeye iner. Eğer B15 biti RDRDY pini olarak etkinleştirilirse, bu anda B15 pini de low seviyeye iner. Ana işlemci aşağıdaki adımları yürüten bir UPI okuma çevrimi başlatır:

1. Ana işlemci UAO bitini birler, sonra URD pinini



GL-45-6-U

Şekil 15-2. UPI Zamanlama Diyagramı

aktifleştirir. (Aktif seviyesi low'dur) HPC donanımı buna UAO pininde görülen değere göre cevap verir:

UAO=0 UPIC saklayıcısının içeriği veri yoluna yerleştirilir.

UAO=1 OBUF saklayıcısının içeriği veri yoluna yerleştirilir.

2. Sonra Ana İşlemci yoldaki veriyi alır ve URD pinini etkin olmayan hale getirir. Ana İşlemci OBUF saklayıcısından okuma yapmışsa, UPIC saklayıcısındaki RDRDY biti, URD 'nin yükselen kenarında, high seviyeye çıkar. UPIC saklayıcısından yapılan okuma RDRDY bitini etkilemez. Port B B15 biti RDRDY çıkışı olarak ayarlanırsa, bu pin RDRDY bitinin durumunu takip eder. URD pini (I2) yükselen kenarında bir kesme üretmek için koşullandırılırsa, HPC'ye bir kesme üretilir.

UPIC saklayıcısı sadece beş bit genişliğindedir. Ya Ana İşlemci yada HPC ona eriştiği zaman, verinin kalan kısmı OBUF saklayıcısından gelir.

RDRDY ve WRRDY lojik kontrolleri OBUF ve IBUF saklayıcılarının byte'larına tek tek erişimi desteklemez. Bu saklayıcıların herhangi bir yarısına erişim saklayıcının tamamına erişim gibi yorumlanır.

RDRDY ve WRRDY pinlerinin kullanımı UPI uygulamasına göre isteğe bağlıdır. Bu pinler Ana İşlemci tarafından kullanılmak üzere seçilmiş el sıkışma yürütülebilir. Bu durumda, bu pinlerin programlanabilir Giriş/Çıkış biti olarak kullanılabilir.

- UAO ismi, Ana İşlemcinin adres yolunun en düşük anlamlı bitine bağlanan pini anlatır. Bununla birlikte, bu sadece Ana İşlemci'nin veri yolu bir byte genişliğinde ise doğrudur. UAO, sadece yoldaki verinin dizilişini belirleyen herhangi bir adres hattına bağlanmalıdır.

15.5 EL SIKIŞMA

UPIC saklayıcısının WRRDY ve RDRDY bitleri ve onların ilgili çıkış bitleri, Ana İşlemci ile HPC arasında el sıkışmaya izin vermeyi sağlar. Ana İşlemci erişimlerini senkronlamayı üç basit tasarı gerçekleştirir:

1. Tamamında el sıkışma işareti yoktur. Burada en basit tasarı, Ana İşlemci, HPC veri aktarımı için hazır olduğu zamanı saptamak için, UPIC saklayıcısını okur ve WRRDY ve RDRDY bitlerini test eder. (Bkz. Şekil 15-3) Bazı uygulamalarda UAO pini gerekli değildir; örneğin, Ana İşlemcinin sadece veri yazdığı ve sadece UPIC saklayıcısını okuduğu yerlerdeki durumlarda. Başka durumlarda, UAO bir çıkış olarak programlanabilir.

2. WRRDY ve RDRDY kullanılarak Ana İşlemciyi kesme. Bazı ararbirim örnekleri Şekil 15-4 ve 15-5 te gösterilmiştir. Bu tasarıda, Ana İşlemci WRRDY veya RDRDY pini low seviyeye gittiği zaman kesilir. Kesmeler ya seviyeye yada kenara duyarlıdır, fakat düşen kenara duyarlı kesmeler, WRRDY işaretini normalde etkin olması gerektiği bazı sistemlerde daha uygun olabilir.

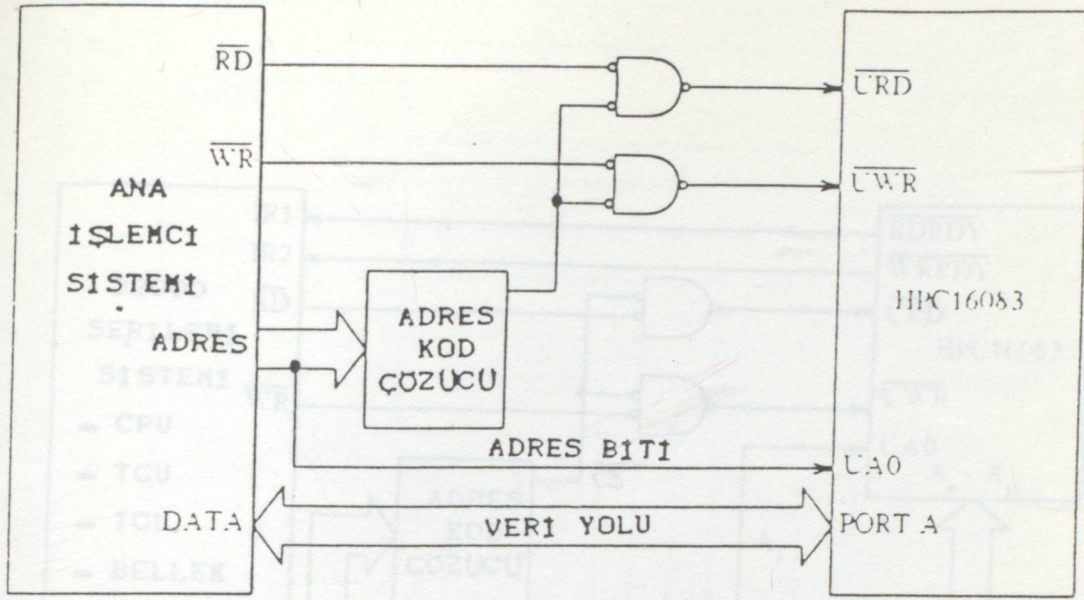
3. Her ne zaman

a. Ana İşlemci HPC'den okumaya teşebbüs etmekte ve RDRDY pini high seviyede (etkin değil), yada

b. Ana İşlemci HPC'ye yazmaya teşebbüs etmekte ve WRRDY pini high seviyede (etkin değil)

ise Ana İşlemci Bekleme durumunda tutulur.

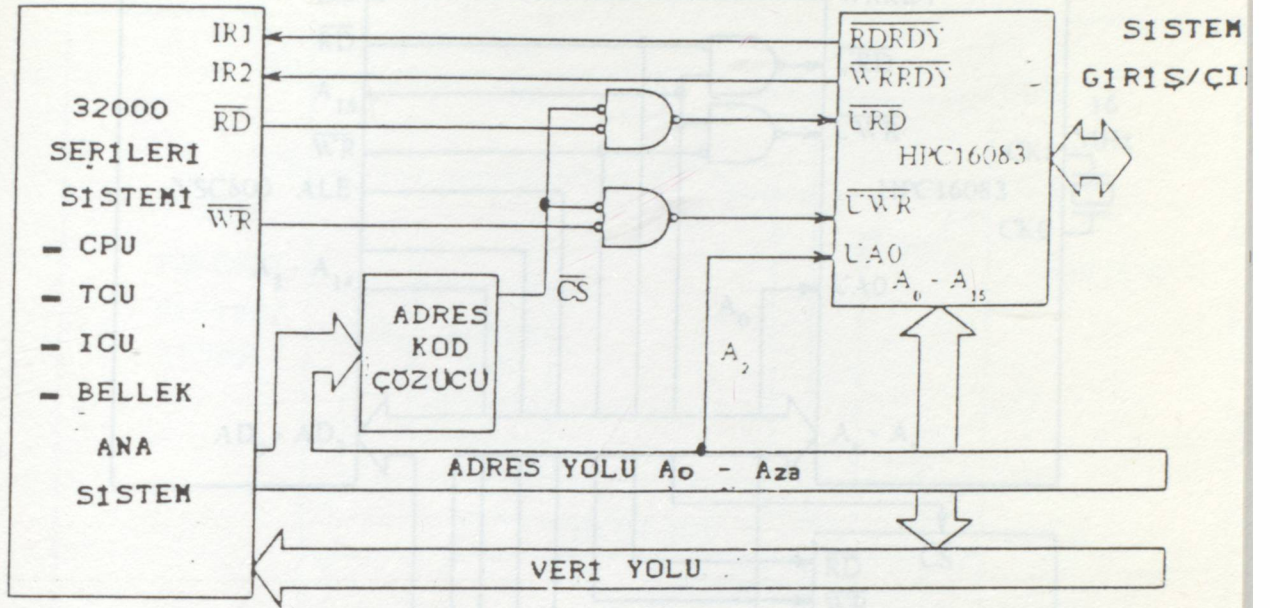
Bu tasarı haberleşmede en yüksek verimi meydana çıkarır, fakat daha pahalı olabilir. Bunu desteklemek için, HPC URD pininin RDRDY high iken etkinleştirilmesine izin verilir. Bununla beraber, UWR pini, sadece WRRDY etkin olduğu zaman HPC'ye sürülsün diye kapılmalıdır. Şekil 15-6 bu tasarımı kullanmak için gerekli olan bağlantıları gösterir. Şekil 15-7 ve 15-8 herhangi bir sistemde okuma ve yazma için gerekli zamanlama



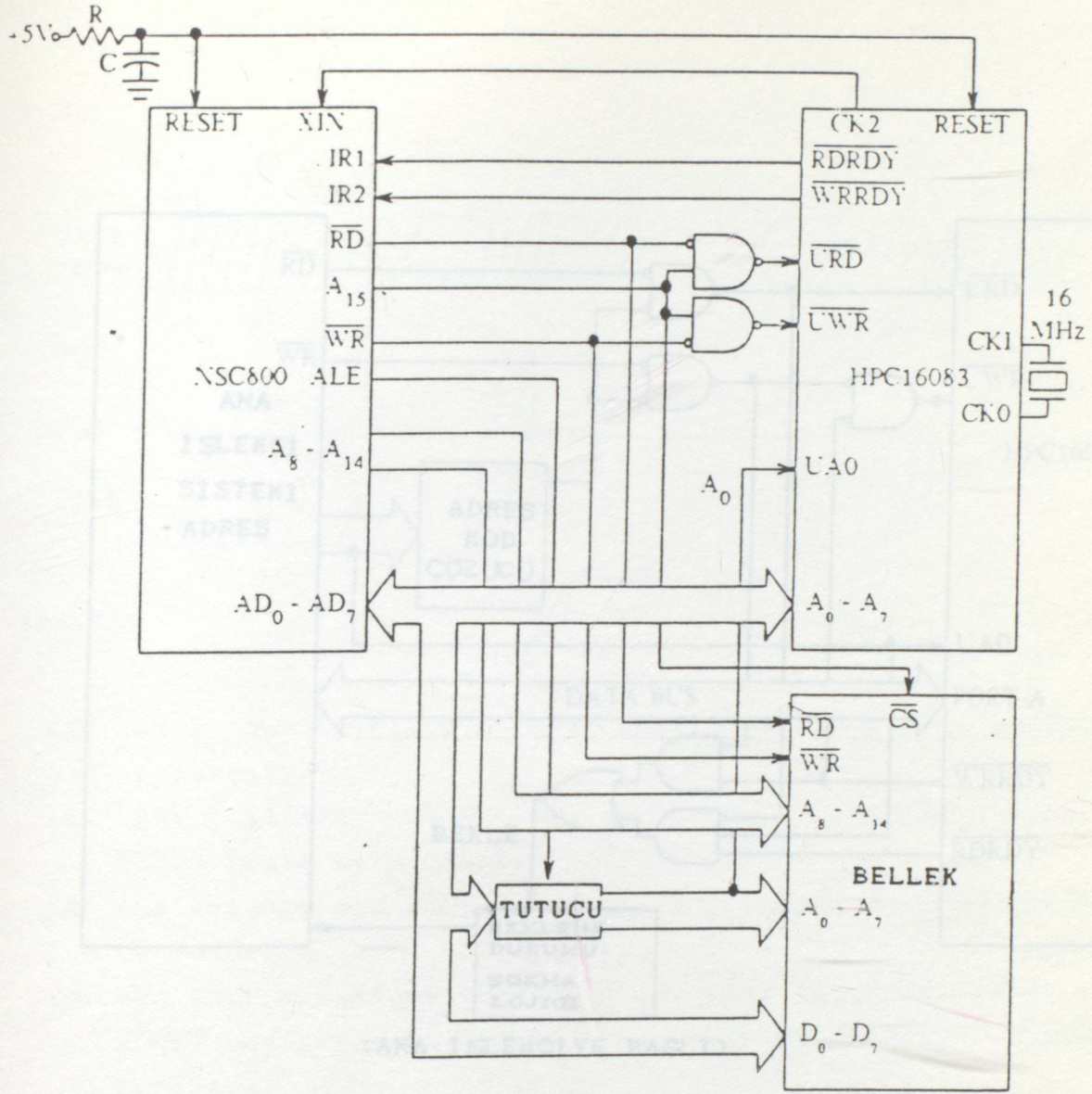
GL-46-0-U

The name "UA0" implies that the pin should be connected to the least-significant bit of the Host Processor's address bus. This, however, is correct only if the Host Processor's data bus is one byte wide. UA0 should not be connected to any address line that specifies only the alignment of data on the bus. For example, the least-significant address bit from a host that addresses bytes on a 16-bit bus should not be used.

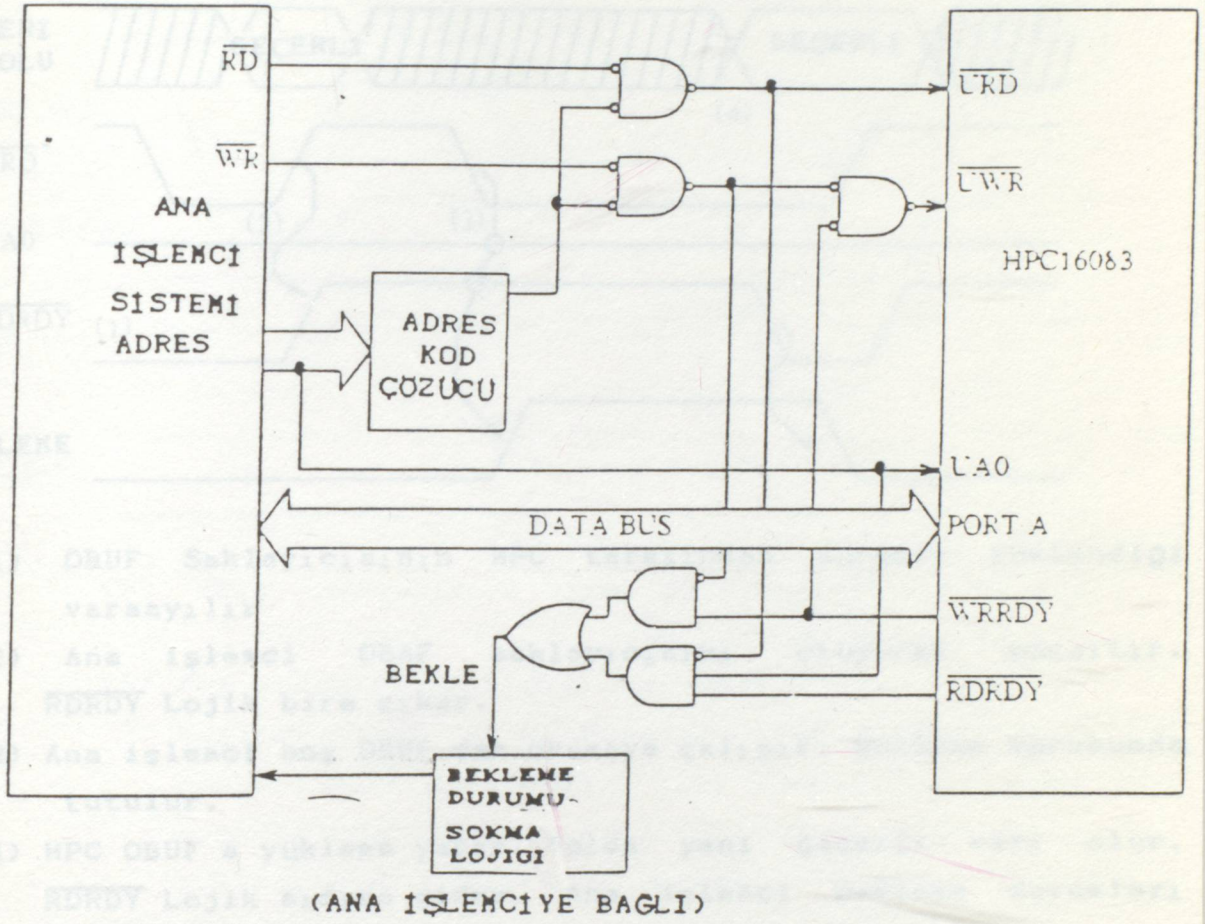
Şekil 15.3 Basit UPI Uygulaması



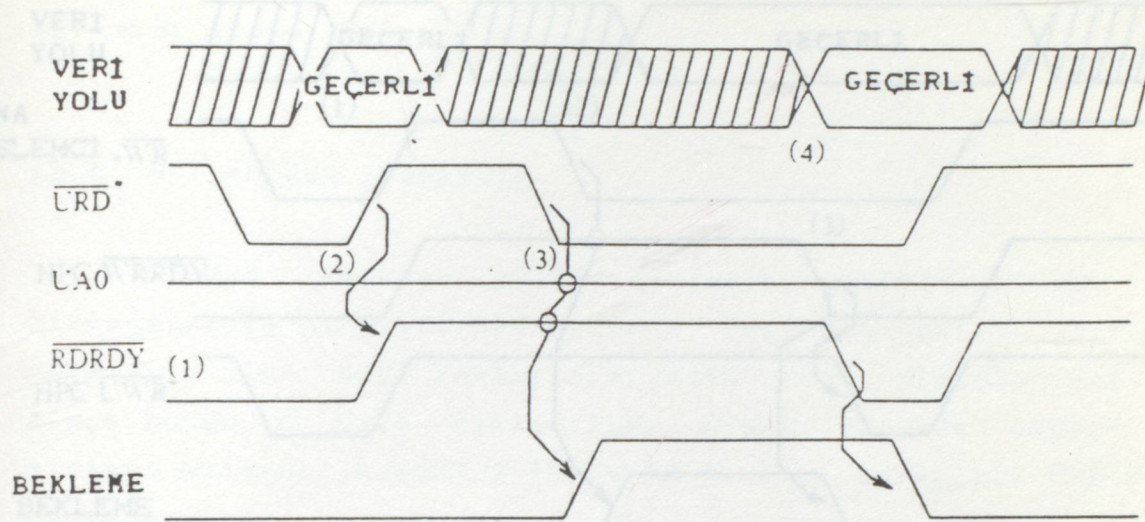
Şekil 15-4. Kesme Süren UPI Uygulaması: UPI 32000 Serileri Sistemi İçin Arabirim



Sekil 15-5. Kesme Süren UPI Uygulaması: UPI NSC800 için arabirim

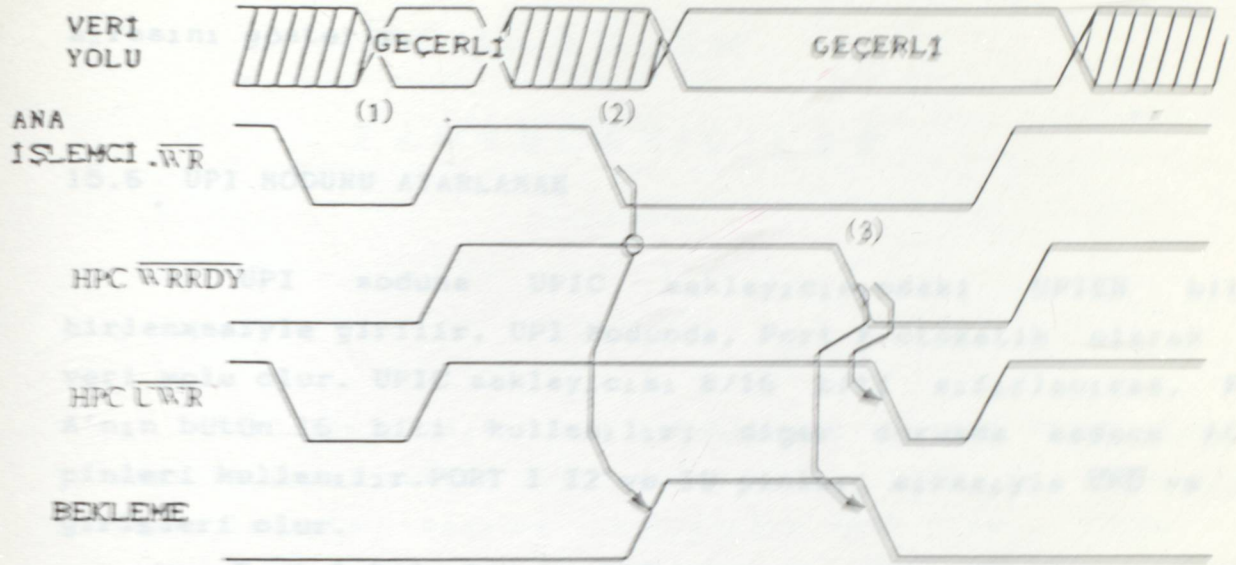


ŞEKİL 15-6. UPI Uygulaması Araya Bekleme Durumları İlave Eder



- (1) OBUF Saklayıcısının HPC tarafından sürekli yüklendiği varsayılır
- (2) Ana işlemci OBAF saklayıcısını okuyarak boşaltır. RDRDY Lojik bire çıkar.
- (3) Ana işlemci boş OBUF dan okumaya çalışır. Bekleme durumunda tutulur.
- (4) HPC OBUF a yükleme yapar. Yolda yeni geçerli veri olur. RDRDY Lojik sıfıra gider. Ana işlemci bekleme durumları bitirilir.

Sekil 15.7 Okuma İşlemlerinin Sırasına Ana İşlemci Bekleme Durumu Yerleştirme



(1) OBUF Saklayıcısı boş; Bu yazma transferi tam hızla gerçekleştirilir

(2) Ana mikroislemci HPC ye ikinci kez yazmaya başlar; IBUF hala doludur, UWR etkin hala getirilmemelidir. Çünkü IBUF saklayıcısındaki değerın üzerine yazılmasına neden olur, Ana mikroislemci bekleme durumunda tutulur,

(3) HPC IBUF Saklayıcısını boşaltır, WRDY aktif duruma geçer ve UWR yi etkinleştirir, Ana mikroislemci bekleme durumları sonlandırılır,

Sekil 15.8 Okuma İşlemleri Anesine Ana İşlemci Bekleme Durumu Yerleştirme

sirasını gösterir.

10. BÖLÜM

ZAMANLAYICILAR

15.6 UPI MODUNU AYARLAMAK

UPI moduna UPIC saklayıcısındaki UPIEN bitinin birleşmesiyle girilir. UPI Modunda, Port A otomatik olarak UPI veri yolu olur. UPIC saklayıcısı 8/16 biti sıfırlanırsa, Port A'nın bütün 16 biti kullanılır; diğer durumda sadece AO-A7 pinleri kullanılır. PORT I 12 ve 13 pinleri sırasıyla URD ve UWR girişleri olur.

Port B B10, B11 ve B15 pinleri sırasıyla UAO girişi ve RDRDY ve WRRDY çıkışları olarak kullanılabilir. Bunları UPI işlemi için ayarlamak için, bazı başlatmalar Port B de yürütülür. WRRDY pini için uygun modu seçmek için, DIRB ve BFUN saklayıcılarının onbirinci bitleri birleşmelidir. RDRDY pini de DIRB ve BFUN saklayıcılarının onbeşinci bitlerinin birleşmesiyle, düzenlenebilir.

UAO pin fonksiyonunun seçimi otomatiktir; bunu seçmek için BFUN saklayıcısının onuncu bitini birlemek gereksizdir. Bununla beraber, UAO pini DIRB saklayıcısının onuncu bitinin sıfırlanmasıyla bir giriş pini olarak tasarlanabilir. Uygulamada UAO pininin fonksiyonu gereksiz ise, bu pin bir çıkış olarak tasarlanabilir.

16. BÖLÜM

ZAMANLAYICILAR

16.1 GİRİŞ

HPC, 16 bitlik dokuz zamanlayıcı (T0'dan T8'e) içerir. T0 zamanlayıcısı yukarı sayıcı olup I2CR, I3CR ve I4CR tutma saklayıcılarıyla beraber kullanılır. I2CR ve I3CR, T1 zamanlayıcısı ve onun giriş saklayıcısı olan R1 gibi hizmet verirler. T8 zamanlayıcısı her zaman T0 ile aynı değerleri içerir. T8 zamanlayıcısı dördüncü giriş tutma saklayıcısı (EICR) için zaman tabanı olarak hizmet verir. T1 den T7 ye kadar olan zamanlayıcılar, R1 den R7 ye kadar olan saklayıcılar ile beraber kullanılan aşağı sayıcılardır. R1 den R7 ye kadar olan giriş saklayıcıları geri sayma işlemi bitince otomatik olarak tekrar yüklenirler. T2 den T7 ye kadar olan zamanlayıcılar kendi çıkış işaretlerine sahiptirler (T2 zamanlayıcısının çıkışı beş adet olabilir). T2 ve T3 zamanlayıcıları iki ayrı giriş vasıtasıyla dışarıya saat işareti verebilirler.

HPC, T0 ve T8 dışındaki bütün zamanlayıcıların herhangi bir zamanda başlatılmasına veya durdurulmasına izin verir. T0 ve T8 dışındaki bu zamanlayıcılar ve onların giriş saklayıcıları herhangi bir zamanda okunabilir. Zamanlayıcı içerikleri, zamanlayıcı asenkron bir dış işaret ile saatlenmiş olsa dahi, yazılımla okunduğu zaman gerçek değerleri gösterir. Bir giriş saklayıcısına veya bir zamanlayıcıya onun zamanlayıcısı ister çalışıyor olsun isterse çalışıyor olmasın, herhangi bir anda geçerli olarak yazılabilir. Bir zamanlayıcıya otomatik olarak veya yazılımla yükleme yapmak, giriş saklayıcısının

içerigini deęiřtirmez. Zamanlayıcıyı bařlatıp giriř saklayıcısını yalnız bırakmak, giriř saklayıcısının ierięi tarafından belirlenen alttan tařmaya ve bu da dzenli araların oluřmasına neden olur. Zamanlayıcı 0000 dan FFFF deęerine alttan tařmayla getięi anda otomatik olarak yklenir. Bir zamanlayıcının n ye blmnden kalanı almak iin giriř saklayıcısına $n-1$ deęeri yklenmelidir.

Btn zamanlayıcılar, zamanlayıcı vektrleri vasıtasıyla kesme retebilme kapasitesine sahiptirler.

16.2 ZAMANLAYICI İřLEMLERİ

16.2.1 TO ve T8 Zamanlayıcıları

TO serbest alıřtırma zamanlayıcısı olup CKI/16 sabit oranıyla saatlenir. Yazılım tarafından basit olarak zaman aralıęı lmede kullanılır. Bununla beraber TO zamanlayıcısı WATCHDOG LOJİK iin referans zamanı belirlemede ve bořta g koruma modu(IDLE) ndan ıkımede kullanılır. Bu iřlemlerden dolayı TO zamanlayıcısını g koruma modu(HALT) dıřında durdurmak veya ona yazmak mmkn deęildir. Buna raęmen, I4CR saklayıcısını kontrol ederek dolaylı olarak okunabilir. Bu iřlem ondan srekli ykleme yapmak iin kullanılır. TO zamanlayıcısının ıkıř pinleriyle baęlantısı yoktur. Fakat alttan tařma sırasında kesme retmek zere programlanabilir. TMMODE saklayıcısının 0-3' bitleri TO zamanlayıcısının alttan tařmasını ayarlar ve kesme retimini kontrol eder.

TO zamanlayıcısı 16 bitlik I2CR, I3CR ve I4CR tutma saklayıcıları ile beraber kullanılır. Bu saklayıcılar TO zamanlayıcısından kendilerine ait I2,I3 veya I4 pinlerinde programlanan kenarın grnmesiyle yklenirler. Her bir pin iin kenarın polaritesi IRCD saklayıcı ile belirlenir. Bu  pin aynı kenarlarda kesme retme kabiliyetine de sahiptirler(Bkz. řekil 16.1) Bu  pinin bir tutma saklayıcısını tetikleyebilmesi

için ENIR saklayıcısındaki I2,I3,I4 dan uygun kesme izin biti birleşmelidir. Bununla beraber G/E bitini birleştirmek gerekmez. I2CR ve I3CR tutma saklayıcıları T1 zamanlayıcısı ve onunla beraber kullanılan R1 saklayıcısını koşullamak için kullanılabilir. Bu üç tutma saklayıcısı okunabilir/yazılabilir saklayıcılardır.

I4CR saklayıcısı da I4 giriş pinine bakmaksızın TO zamanlayıcısından sürekli yüklenebilecek şekilde programlanabilir. Sadece TO zamanlayıcısının kontrolünü sağlayan bu özellik TOCON saklayıcısının C4F bitinin birleşmesiyle sağlanır. I4CR saklayıcısı CK2 saatinin her sekiz çevriminde bir kez yüklenir. Bu sebepten dolayı C4F bitini birleştirmeyi ve sonra TO zamanlayıcısını kontrol etmeyi amaçlayan bir program, I4CR saklayıcısının en az bir kez yüklendiğine emin olmak için C4F bitini birleyip ondan sonra bu kadar zaman periyodu beklemelidir. C4F bitini birleyen komuttan sonra iki NOP komutu yerleştirme, gerekli gecikmeyi sağlar.

T8 zamanlayıcısı TO zamanlayıcısı gibi yukarı sayıcı olarak kullanılır. Yine dördüncü giriş saklayıcısı olan EICR için zaman tabanı olarak hizmet verir (Bkz. Şekil16.1). EICR saklayıcısı, EI pininde programlanan kenarın görünmesiyle, T8 zamanlayıcısındaki değerle yüklenir. T8 zamanlayıcısı her zaman TO zamanlayıcısı ile aynı değerleri içerir. Bu nedenle I2CR, I3CR, I4CR ve EICR giriş saklayıcılarının tümüne I2, I3, I4 ve EI pinlerinden kendilerine ait olanında kenar tesbit edildiğinde TO zamanlayıcısındaki değerle yüklenirler. EI pini üretilen kesmenin lojik 1 veya lojik 0 ve düşen veya çıkan kenarda tesbit edilmesi için programlanabilir. Bu dört farklı seçim EICON saklayıcısındaki iki bitin programlanması ile gerçekleştirilir. EI pini, giriş saklayıcısındaki zaman aralıklarını ölçmek için EICR saklayıcısıyla beraber kullanıldığı zaman sadece düşen veya çıkan kenar tesbit edilir. EI pini, giriş saklayıcısı için kullanıldığı zaman, IRPD saklayıcısındaki EI öncelik biti, EICON

saklayıcısındaki ACK bitine sıfır yazarak sıfırlanmalı ve ENIR saklayıcısındaki EI bitini birleyerek EI kesmesine izin verilmelidir.

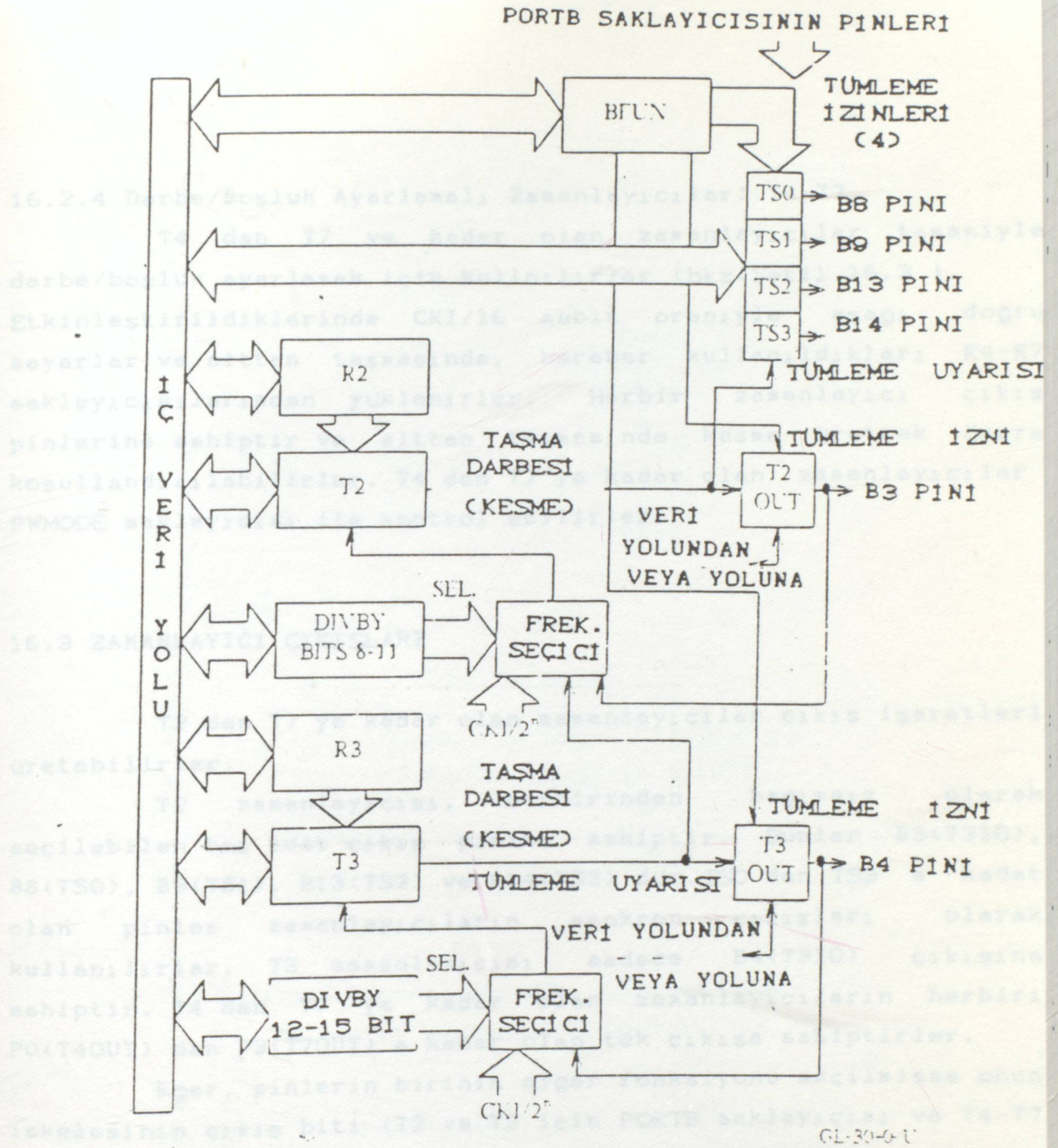
16.2.2 T1 Zamanlayıcısı

T1 zamanlayıcısı ve onunla birlikte kullanılan R1 saklayıcısı, T0 tutma saklayıcıları olan I2CR ve I3CR seçilmediği zaman kullanılırlar (bkz şekil 16.1). Bu seçme işlemi TOCON saklayıcısındaki TOORG bitinin durumuyla belirlenir. Bu bit sıfırsa tutma saklayıcısı fonksiyonları, I3CR ve I2CR, birse zamanlayıcı fonksiyonları seçilmiş demektir. T1 zamanlayıcısı etkinleştirildiğinde CKI/16 sabit oranıyla aşağı doğru sayar. T1 zamanlayıcısının çıkış pini yoktur. Fakat alttan taşmada kesme üretmesi için koşullanabilir. T1 zamanlayıcısı TMMODE saklayıcısının 4-7 ci bitleriyle kontrol edilir.

16.2.3 T2 ve T3 Zamanlayıcıları

T2 ve T3 zamanlayıcıları HPC nin en esnek zamanlayıcılarıdır (bkz. Şekil 16.2). T2 ve T3 zamanlayıcılarına saat işaretleri, CKI saatinin önceden belirlenen 14 durumunun birinden veya DIVBY saklayıcısınca belirlenen bir dış kaynaktan gelebilir. T2 zamanlayıcısı, yine T3 zamanlayıcısının alttan taşmasıyla da saatlenebilir. Bu çift R3 saklayıcısında tutulan, degere bağlı olan, 32 bite kadarki saymanın sonundaki etkiyle şekillendirilir. Eger DIVBY sak ile T2 ve T3 zamanlayıcılarına dış girişler secilmişse bu girişler B iskelesinin B3 ve B4 pinlerinden alınır ve zamanlayıcılar düşen kenarda azaltılırlar. Bu pinler zamanlayıcı çıkışları veya iskele giriş/çıkışları olarak hizmet verebileceğinden BFUN ve DIRB saklayıcılarında doğru koşullandırma yapılmalıdır.

T2 zamanlayıcısı TMMODE saklayıcısının 8-11 ci ve T3 zamanlayıcısı da yine aynı saklayıcının 12-15 ci bitleriyle kontrol edilir.



Şekil 16.2 T2 ve T3 Zamanlayıcıları İçin Blok Diyagramı

16.2.4 Darbe/Boşluk Ayarlanmalı Zamanlayıcılar: T4-T7

T4 den T7 ye kadar olan zamanlayıcılar tamamıyla darbe/boşluk ayarlamak için kullanılırlar (bkz.Şekil 16.3). Etkinleştirildiklerinde CKI/16 sabit oranıyla aşağı doğru sayarlar ve alttan taşmasında, beraber kullanıldıkları R4-R7 saklayıcılarından yüklenirler. Herbir zamanlayıcı çıkış pinlerine sahiptir ve alttan taşmasında kesme üretmek üzere koşullandırılabilirler. T4 den T7 ye kadar olan zamanlayıcılar PWMODE saklayıcısı ile kontrol edilirler.

16.3 ZAMANLAYICI ÇIKIŞLARI

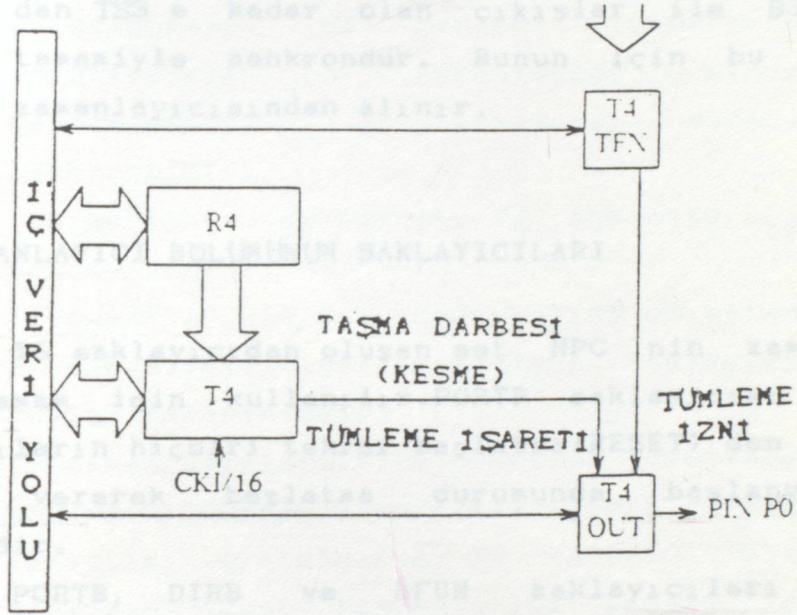
T2 den T7 ye kadar olan zamanlayıcılar çıkış işaretleri üretebilirler.

T2 zamanlayıcısı, birbirinden bağımsız olarak seçilebilen beş adet çıkış pinine sahiptir. Bunlar B3(T2IO), B8(TSO), B9(TS1), B13(TS2) ve B14(TS3) dür.TSO den TS3 e kadar olan pinler zamanlayıcıların senkron çıkışları olarak kullanılırlar. T3 zamanlayıcısı sadece B4(T3IO) çıkışına sahiptir. T4 den T7 ye kadar olan zamanlayıcıların herbiri PO(T4OUT) dan P3(T7OUT) a kadar olan tek çıkışa sahiptirler.

Eğer, pinlerin birinin diğer fonksiyonu seçilmişse onun iskelesinin çıkış biti (T2 ve T3 için PORTB saklayıcısı ve T4-T7 saklayıcısı için POTRP saklayıcısı) ilgili zamanlayıcının her alttan taşmasında tükleyen(toggel) olarak çalışır. Eğer pin, çıkış olarak koşullanmışsa (B iskelesi pinleri için geçerlidir, P iskelesi pinleri sadece çıkıştır) iskele çıkış bitinin değerini izler. Bu bit yine yazılımcı için sadece okunabilir durumdadır. Böylece aynı saklayıcılerdeki kullanılmayan diğer bitlerin programlanması sırasında oluşabilecek müdahalelerden korunulmuş olunur.

NOT:

Farklı zamanlayıcılardan alınan çıkışlar temelde aynı zamanda döndürmeyi yapar. Çünkü zamanlayıcı benzer şekilde çalışır. Zamanlayıcılar, T0-T7, T4-T7 olarak üzere iki ayrı gruba ayrılırlar. Her gruptaki bir zamanlayıcı CK2 PORTB SAKLAYICISI BITLERİ



NOT: Sadece T4 zamanlayıcısı görülmektedir
T5, T6 ve T7 Benzer Şekildedir.

Şekil 16.3 PWM Zamanlayıcıları (T4-T7) Blok Diyagramı

12CR/R1 :16 bitlik tutma saklayıcısıdır. 0186 bellek adresindedir.

NOT: Farklı zamanlayıcılardan alınan çıkışlar tamamıyla aynı zamanda dönüştürmeyi yapmaz. Çünkü zamanlayıcılar benzer şekilde çalışmazlar. Zamanlayıcılar, T0-T7 ve T4-T7 olmak üzere iki ayrı gruba ayrılırlar. Her gruptaki bir zamanlayıcı CK2 saatinin her iki sayma işleminden sonra güncelleştirilir. Bununla beraber TS0 dan TS3 e kadar olan çıkışlar ile B3(T2IO) çıkışı tamamıyla senkronudur. Bunun için bu çıkışlar T2 zamanlayıcısından alınır.

16.4 ZAMANLAYICI BÖLÜMÜNÜN SAKLAYICILARI

16 saklayıcıdan oluşan set HPC nin zamanlayıcılarını programlamak için kullanılır. PORTP saklayıcısı dışındaki bu saklayıcıların hiçbiri tekrar başlatma(RESET) den etkilenmezler ve güç vererek başlatma durumunda başlangıç durumları belirsizdir.

PORTB, DIRB ve BFUN saklayıcıları T2 ve T3 zamanlayıcılarından alınan çıkışları ve bu zamanlayıcılara dışarıdan girilen girişleri kontrol etmek için kullanılır.

I4CR :16 bitlik tutma saklayıcısıdır. 0180 bellek adresindedir. I4 pininde programlanan kenarın görünmesiyle T0 zamanlayıcısından yüklemeyi başlatmak için kullanılır. T0 zamanlayıcısından sürekli yükleme yapmak için de programlanabilir. Bu işlemlerin seçimi TOCON saklayıcısındaki C4F biti ile gerçekleştirilir.

I3CR/TI :0182 bellek adresindedir. 16 bitlik tutma saklayıcısıdır. I3 pininde programlanan kenarın görünmesiyle T0 zamanlayıcısından yüklemeyi başlatmak için kullanılır. TOCON saklayıcısındaki TOORG bitinin birleşmesiyle T1 zamanlayıcısı gibi hizmet verir.

I2CR/R1 :16 bitlik tutma saklayıcısıdır. 0184 bellek adresindedir. I2 pininde programlanan kenarın gözükmesiyle T0 zamanlayıcısından yüklemeyi başlatmak için kullanılır. TOCON saklayıcısının TOORG bitinin birleşmesiyle T1 zamanlayıcısı için R1 giriş saklayıcısı olarak hizmet verir.

EICR :16 bitlik tutma saklayıcısıdır. 015E bellek adresindedir. EI pininde programlanan kenarın gözükmesiyle T8 zamanlayıcısından yüklemeyi gerçekleştirir. Bu saklayıcı sadece okunabilir saklayıcıdır.

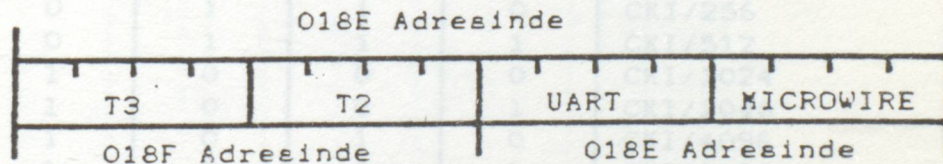
R2 :0186 bellek adresindedir. T2 zamanlayıcısı için giriş saklayıcısıdır.

T2 :T2 zamanlayıcısıdır. 0188 bellek adresindedir.

R3 :T3 zamanlayıcısıdır için giriş saklayıcısı olup 018A bellek adresindedir.

T3 :T3 zamanlayıcısıdır.018C bellek adresindedir.

DIVBY :UART ve MICROWIRE arabirimleri için gerekli olan saat işaretini üretmek ve T2 ve T3 zamanlayıcılarına girilecek saat işaretlerini seçmek için kullanılan saklayıcıdır. DIVBY saklayıcısının adres alanının formatı şu şekildedir:



DIVBY saklayıcısının 12-15 ci bitleri T3 zamanlayıcısına girecek olan saati belirlemede kullanılır.

BIT15	BIT14	BIT13	BIT12	T3'e GİREN SAAT
0	0	0	0	B4 Pininden
0	0	0	1	T3 Çıkışından
0	0	1	0	CKI/16
0	0	1	1	CKI/32
0	1	0	0	CKI/64
0	1	0	1	CKI/128
0	1	1	0	CKI/256
0	1	1	1	CKI/512
1	0	0	0	CKI/1024
1	0	0	1	CKI/2048
1	0	1	0	CKI/4096
1	0	1	1	CKI/8192
1	1	0	0	CKI/16384
1	1	0	1	CKI/32768
1	1	1	0	CKI/65536
1	1	1	1	CKI/131072

DIVBY saklayıcısının 8-11 ci bitleri T2 zamanlayıcısına girecek olan saati belirlemede kullanılır.

BIT11	BIT10	BIT9	BIT8	T2'ye GİREN SAAT
0	0	0	0	B4 Pininden
0	0	0	1	T3 Çıkışından
0	0	1	0	CKI/16
0	0	1	1	CKI/32
0	1	0	0	CKI/64
0	1	0	1	CKI/128
0	1	1	0	CKI/256
0	1	1	1	CKI/512
1	0	0	0	CKI/1024
1	0	0	1	CKI/2048
1	0	1	0	CKI/4096
1	0	1	1	CKI/8192
1	1	0	0	CKI/16384
1	1	0	1	CKI/32768
1	1	1	0	CKI/65536
1	1	1	1	CKI/131072

DIVBY saklayıcısının 4-7 ci bitleri UART ın haberleşme hızını BAUD olarak belirlemede kullanılır.

Bit7	Bit6	Bit5	Bit4	BAUD SAATI	BAUD ORANI	BAUD ORANI
0	0	0	0	< izin verilmez >		
0	0	0	1	<T3'un Alttan Taşmasıyla Elde Edilir>		
0	0	1	0	CKI/16	38400	65536
0	0	1	1	CKI/32	19200	32768
0	1	0	0	CKI/64	9600	16384
0	1	0	1	CKI/128	4800	8192
0	1	1	0	CKI/256	2400	4096
0	1	1	1	CKI/512	1200	2048
1	0	0	0	CKI/1024	600	1024
1	0	0	1	CKI/2048	300	512
1	0	1	0	CKI/4096	150	256
1	0	1	1	CKI/8192	75	128
1	1	0	0	CKI/16384	38	64
1	1	0	1	CKI/32768	19	32
1	1	1	0	CKI/655536	9.4	16
1	1	1	1	CKI/131072	4.7	8

DIVBY saklayıcısının 0-3 cü bitleri ise HPC ana modu iken MICROWIRE sisteminin seri saat frekansını belirlemede kullanılır.

Bit3	Bit2	Bit1	Bit0	MICROWIRE SAATI
0	0	0	0	İzin Verilmez
0	0	0	1	T3'un alttan taşmasıyla
0	0	1	0	CKI/16
0	0	1	1	CKI/32
0	1	0	0	CKI/64
0	1	0	1	CKI/128
0	1	1	0	CKI/256
0	1	1	1	CKI/512
1	0	0	0	CKI/1024
1	0	0	1	CKI/2048
1	0	1	0	CKI/4096
1	0	1	1	CKI/8192
1	1	0	0	CKI/16384
1	1	0	1	CKI/32768
1	1	1	0	CKI/655536
1	1	1	1	CKI/131072

TOCON :TO zamanlayıcısını koşullandırmada kullanılır.

*	*	*	*	TOORG X	TPC4F X	*	*
---	---	---	---	------------	------------	---	---

019E Bellek Adresinde Bulunur.

X:Reset durumunda belirsiz

*:Kullanılmayan bit. Belirsiz veri içerebilir.

TOC4F:I4CR tutma saklayıcısının fonksiyonunu belirlemede kullanılır.

TOC4F=1:TO zamanlayıcısından I4CR ye sürekli yükleme yapmak için.

TOC4F=0:I4 pinindeki durumları I4CR ye saklamak için.

TOORG:TO zamanlayıcısının tutma saklayıcıları olan I2CR ve I3CR nin organizasyonu için kullanılır.

TOORG=1:I2CR yi R1 saklayıcısı ve I3CR yi T1 zamanlayıcısı olarak tekrar tanımlamak için.

TOORG=0:I2CR ve I3CR yi, I2 ve I3 pinlerindeki değerleri tutacak şekilde tanımlamak için.

EICON :EI pinin kesme konumlarını belirlemede kullanılır.

*	*	*	*	*	ACK 1W	MODE 0	POL 0
---	---	---	---	---	-----------	-----------	----------

015C Bellek Adresinde Bulunur

1:Reset durumunda birlenir.

0:Reset durumunda sıfırlanır.

X:Reset durumunda

belirsizdir.

W:Sadece okunabilir bittir ve başlangıç durumu geri okunabilir.

*:Kullanılmayan bit. Belirsiz veri içerebilir.

ACK:EI kesme anlaşıldı biti. Bu biti sıfırlayarak IRPD saklayıcısındaki EI kesme öncelik biti silinir. Bunun

İçin uygun kesme kenarının belirlenmesi gerekir.
 MODE: Bu bit ile EI kesme fonksiyonunun seviyesini veya kenar tipinin belirlenmesi sağlanır.
 POL: Bu bit EI kesmesinin kenarını veya aktif geçişini belirler. MODE bitiyle beraber kullanılır.

EICON SAKLAYICISI 1. ve 2. BITLERİ			
MODE BITİ	POL BITİ	KESME FONKSİYONU	GİRİŞ SAKLAYICISI FONKSİYONU
0	0	Lojik 1	Düşen Kenar
0	1	Lojik 0	Çıkan Kenar
1	0	Düşen Kenar	Düşen Kenar
1	1	Çıkan Kenar	Çıkan Kenar

TMODE : Zamanlayıcı mod saklayıcılarıdır. T0-T3 zamanlayıcılarını kontrol eder.

0190 Bellek Adresinde Bulunur.

T3	T3	T3	T3	T2	T2	T2	T2	T1	T1	T1	T1	T0		T0	T0
ACK	STP	TIP	TIE	ACK	STP	TIP	TIE	ACK	STP	TIP	TIE	ACK	*	TIP	TIE
OW	X	XR	X	OW	X	XR	X	OW	X	XR	X	OW		XR	X

0191 Bellek Adresinde

PWMODE : Darbe/bosluk ayarlamalı saklayıcıdır. T4-T7 zamanlayıcılarını kontrol eder.

0150 Bellek Adresinde Bulunur.

T7	T4	T7	T7	T6	T6	T6	T6	T5	T5	T5	T5	T4	T4	T4	T4
ACK	STP	TIP	TIE	ACK	STP	TIP	TIE	ACK	STP	TIP	TIE	ACK	STP	TIP	TIE
OW	X	XR	X	OW	X	XR	X	OW	X	XR	X	OW	X	XR	X

0151 Bellek Adresinde

TIE:İlgili zamanlayıcının kesmesini etkinleştirir veya izni kaldırır. Bu bit birlenerek ilgili kesme etkinleştirilir.

TIP:İlgili zamanlayıcının kesme bayrağı. TIE ve bu bit birleşmiş olarak kaldığı sürece kesme isteği devam eder. TIP biti sadece okunabilir. Zamanlayıcının alttan veya üstten taşmasıyla birleşir ve beraber kullanıldığı ACK kontrol bitinin birleşmesiyle sıfırlanır. TIP bitinin birleşmesinde TIE bitinin durumuna bakılmaz.

STP:İlgili zamanlayıcıyı başlatmak veya durdurmak için kullanılır

STP=1 :Zamanlayıcıyı durdurmak için.

STP=0 :Zamanlayıcıyı başlatmak için.

ACK:Zamanlayıcı kesme bilgisi kontrol biti. Bu bit birlenerek ilgili zamanlayıcının kesme bayrağı indirilir. ACK biti sadece yazılabilir bittir. Her zaman ondan geriye sıfır okunur.

ACK=1 :TIP bitini sil.

ACK=0 :İşlem yapma.

NOT: Zamanlayıcının alttan veya üstten taşmasının anlaşılmasıyla TIP bayrağının kaldırılması arasında sekiz CK2 saat periyodundan oluşan bir pipeline gecikmesi vardır. Bu sebepten ACK ve STP bitlerinin aynı anda birleşmesiyle TIP bayrağının inmiş olarak kalmasına garanti verilemez. Eğer STP ve ACK bitleri zamanlayıcının alttan taşmasından sonra ivedi olarak birleşirse, TIP biti ACK biti tarafından silindikten sonra donanıma tarafından birlenebilecektir. Özellikle bu durum güç reseti yaptıktan sonra TMMODE ve PWMODE saklayıcılarının başlangıç değerleri yüklendiği zaman söz konusu olur. Bu nedenle böyle durumlarda kontrol saklayıcıları ve zamanlayıcıların durumları belirsizdir. TIP bayrağının inmiş olduğundan emin olmak

için uygulanabilecek iki ayrı erişim vardır: İlk erişim zamanlayıcıyı durdurmak için STP bitini birlemektir. İki NOP komutu (en az sekiz CK2 saat periyodu gecikmeyi sağlamak için) yürütüldükten sonra ikinci erişim, TIP bayrağını indirmek için ACK bitini birlemektir.

PORTP : P iskelesi çıkış saklayıcısıdır. Bitlerin içerikleriyle T4 den T7 ye kadar olan PWM zamanlayıcılarının çıkışlarının programlanması sağlanır.

T7 TFN 0	*	*	T7 OUT 0	T6 TFN 0	*	*	T6 OUT 0	T5 TFN 0	*	*	T5 OUT 0	T4 TFN 0	*	*	T4 OUT 0
P3 Kontrol			P2 Kontrol			P1 Kontrol			P0 Kontrol						

0153

0152

OUT:P iskelesinin çıkış işaretlerinin seviyesini belirlemede kullanılır. Bu bitler TFN bitini birleyerek tımlama (toggle) moduna girildiğinde birkez sadece okunabilir olur.

OUT=1 :İlgili P iskelesi pininin seviyesi lojik birdir.

OUT=0 :İlgili P iskelesi pininin seviyesi lojik sıfırdır.

TFN:PO(T4OUT)-P3(T7OUT) iskele pinlerinin fonksiyonlarını belirler.

TFN=1 :İlgili OUT bitini zamanlayıcının alttan taşmasında tımlayan olarak çalıştırmak için koşullandırılır.

TFN=0 :Tımlama işleminin iznini kaldırır. İlgili P iskelesi pini yazılımla değiştirilebilen OUT bitinden kendi değerini almaya devam eder.

NOT : Bir OUT biti, ilgili TFN bitini birleyen bir erişimle

aynı anda değiştirilemez. Böyle yapmak OUT bitine tahmin edilemeyen etki yapabilir. Bunun yerine, ilk önce OUT bitine istenen değer kaydedilmeli sonra da TFN bitine müteakip bir komutla değeri kaydedilmelidir. OUT bitini değiştiren bir erişimde TFN bitinin silinmesine izin verilir. İki bitteki yapılan değişikliklere güvenilebilir.

16.5 ZAMANLAYICILARI KURMAK VE AYARLAMAK

Uygulamalarda zamanlayıcıları ve giriş çıkış pinlerini koşullamada şu şekildeki işlem sırası tavsiye edilir.

- Zamanlayıcı durdurma ve tekrardan başlatma :

TMODE saklayıcısına 4400H değeri yazılarak T1 ve T3 zamanlayıcıları durdurulur ve T0-T3 zamanlayıcılarının kesme izinleri kaldırılır.

CK2 saat işaretinin en az sekiz çevrimlik gecikmesini sağlamak için iki NOP komutu veya uygun herhangi bir komut yürütülür.

TMODE saklayıcısına CCC8H değeri yazılarak T0-T3 zamanlayıcısının bütün PND bitleri silinir. Bu işlem STP bitlerini birde, tie bitlerini de sıfırda tutarak ACK bitlerinin tümünü birleyerek yapılır.

PWMODE saklayıcısına 4444H değeri yazılarak T4-T7 zamanlayıcıları durdurulur ve kesme izinleri kaldırılır.

CK2 saatinin en az 8 çevriminden oluşan gecikmenin sağlanması için iki NOP komutu veya uygun bir komut yürütülür.

PWMODE saklayıcısına CCCCH değeri yazılarak T4-T7 zamanlayıcılarının bütün PND bitleri silinir. Bu işlem STP bitlerini birde, TIE bitlerini de sıfırda tutarak

ACK bitlerinin tümünü birleyerek yapılır.

. Zamanlayıcı giriş/çıkışlarını koşullama:

16.1 tablosu B3 (T2IO) ve B4 (T3IO) pinleri için mümkün olan işlem modlarını göstermektedir. Çıkış olarak koşullanabilecek olan bu bitlerin herbiri için PORTB saklayıcısının ilgili bitine istenilen başlangıç değeri yazılır. İstenilen işlem modunu seçmek için DIRB ve BFUN saklayıcılarının 3. ve 4. bitleri koşullanmalıdır. Bu anda DIVBY saklayıcısı önemsenmez.

16.2 tablosu B8 (TS0), B9 (TS1), B13 (TS2) ve B14 (TS3) pinleri için mümkün olan işlem modlarını gösteriyor. Çıkış olarak koşullanabilecek olan bu bitlerin herbiri için PORTB saklayıcısının ilgili bitine istenilen ilk değer yazılır. İstenilen işlem modunu seçmek için DIRB ve BFUN saklayıcılarının 8., 9., 13. ve 14. bitleri uygun olarak kaydedilir.

PORTP saklayıcısının T4OUT-T7OUT bitlerine istenilen başlangıç değerleri yazılır. Sonra zamanlayıcıların alttan taşmalarında tümlen olarak çalışması için T4TFN-T7TFN bitleri koşullanır. Tekrardan başlatma (reset) durumunda bu pinlerin başlangıç değerleri sıfırdır.

. Zamanlayıcıları koşullama ve ayarlama

TOCON saklayıcısının TOORG bitine uygun değer yazılarak I2CR/I3CR (R1/T1) saklayıcısının istenilen fonksiyonu seçilir.

TOCON saklayıcısının TOC4F bitine uygun değer yazılarak I4CR saklayıcısı için istenilen mod seçilir.

DIVBY saklayıcısının yüksek anlamlı baytına uygun değer yazılarak T2 ve T3 zamanlayıcıları için istenilen saat kaynağı seçilir.

İstenilen zamanlayıcılara ve onların giriş saklayıcılarına başlangıç değerleri yüklenir. TMMODE

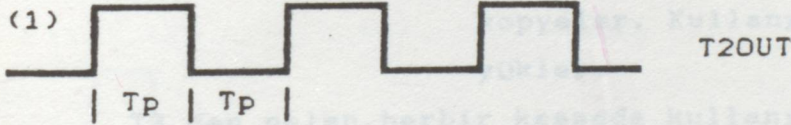
(1) ve PWMODE saklayıcılarında ilgili bitlerin koşullanmasıyla kesme izinleri verilir. Zamanlayıcıları başlatmadan önce ENIR saklayıcısının TIMERS ve CIE bitlerinin birlendigine emin olunmalıdır. Bu işlem zamanlayıcıların kesmelerinin tanınması için gereklidir.

• TMMODE ve PWMODE saklayıcısındaki uygun STP bitleri silinerek zamanlayıcılar başlatılır.

16.6 ZAMANLAYICI UYGULAMALARI

Birçok uygulamada yaygın olarak kullanılan örneklerin zamanlayıcılar ile oluşturulmasını inceleyelim.

KARE DALGA FREKANSI URETME:



(1) Kullanıcı Yazılımı:

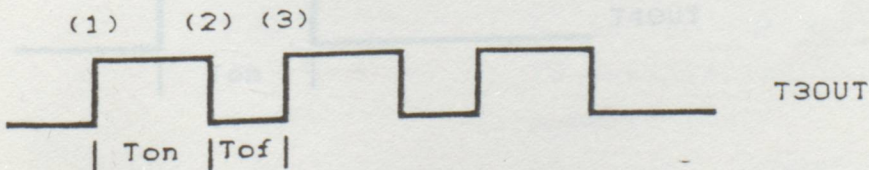
T2 ye TP e yüklenir.

R2 ye TP e yüklenir.

T2 zamanlayıcısı başlatılır.

Çıkışta sabit kare dalgı elde edilir.

SABIT VE DEĞİŞKEN DARBE/BOSLUK ORANLI FREKANS URETME



(1) Kullanıcı Yazılımı:

T3 e Ton yüklenir.

R3 e Tof yüklenir.

T3 zamanlayıcısının kesmesi etkinleştirilir.

B4(T3OUT) pini Lojik 1 e çıkarılır ve tümleyen olarak çalıştırılmasına izin verilir.

T3 zamanlayıcısı başlatılır.

(2) T3 den Kesme Gelmesi Durumu:

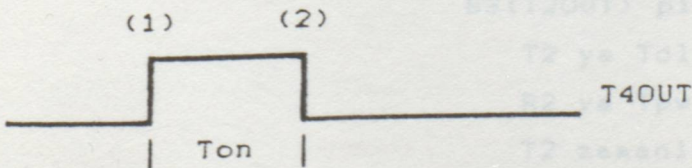
Donanım T3OUT u tümleyen olarak çalıştırır ve R3 ün içeriğini T3 e kopyeler. Kullanıcı yazılımla R3 e Ton yükler.

(3) T3 den Kesme Gelmesi Durumu:

Donanım T3OUT u tümleyen olarak çalıştırır ve R3 ün içeriğini T3 e kopyeler. Kullanıcı yazılımla R3 e Tof yükler.

T3 den gelen herbir kesmede kullanıcı dönüşümlü olarak Ton ve Tof değerlerini R3 e yükler Böylece çıkışta sabit çevrimli işaret elde edilir. Ton ve Tof değerlerini, toplamaları sabit kalacak şekilde, değiştirerek sabit frekansta değişken darbe/boşluk oranlı çevrimler oluşturulabilir.

TEK DARBE ÜRETME:



(1) Kullanıcı Yazılımı:

T4 e Ton yüklenir.

T4 zamanlayıcısına kesme izni verilir.

P0 (T4OUT) biti çıkış olarak koşullanır.

T4OUT un tümleyen olarak çalışmasına izin verilir.

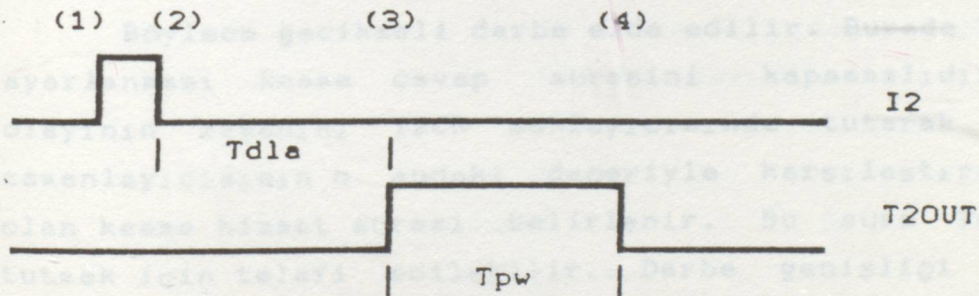
T4 zamanlayıcısı başlatılır.

(2) T4 den Kesme Geldiğinde:

Donanım T4OUT u tümleyen olarak çalıştırır.

Kullanıcı T4 zamanlayıcısını durdurur.

DISARIDAN TETIKLEYEREK GECIKMELI DARBE URETME:



(1) Kullanıcı Yazılımı:

T2 zamanlayıcısı durdurulur ve B3(T2OUT) pini silinir.

T2 ye Tdla yüklenir.

R2 ye Tpw yüklenir.

T2 zamanlayıcısının kesme üretmesine izin verilir.

I2 nin düşen kenarında kesme üretmesine izin verilir

I2CR/R1 tutma saklayıcısı olarak koşullanır.

I4CR TO zamanlayıcısından sürekli yüklenmek üzere koşullanır.

(2) I2 den Kesme Geldiğinde:

Kullanıcı yazılımla I2CR nin içeriginden I4CR nin içerigini çıkarır. T2 deki Tdla değeri bu temel değere göre ayarlanır. Yazılımla T2 başlatılır.

(3) T2 Zamanlayıcısından Kesme Geldiğinde:

Donanım T2OUT u tümleyen olarak çalıştırır ve içerigini T2 ye kopyeler.

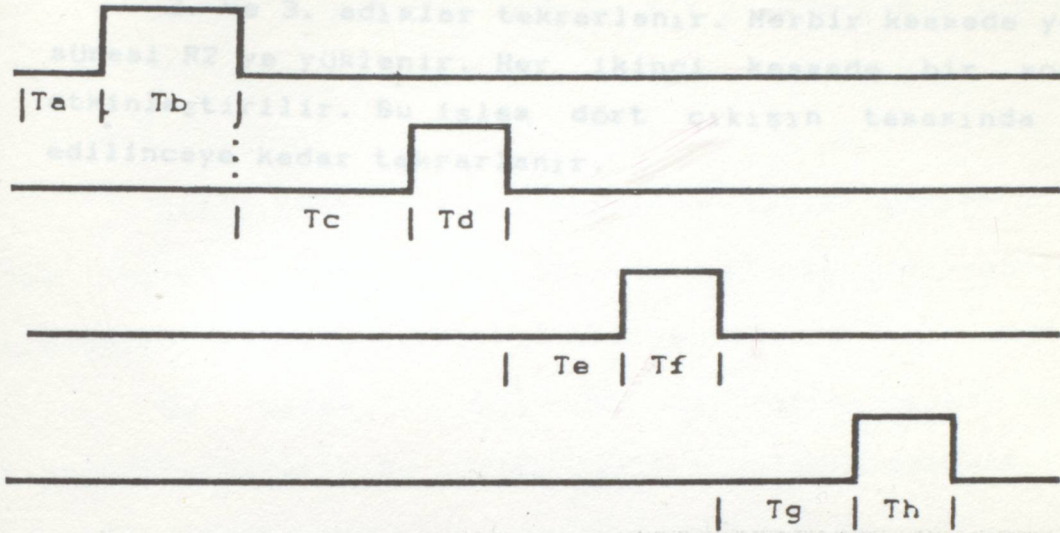
(4) T2 Zamanlayıcısından Kesme Geldiğinde:

Donanım T2OUT u tümleyen olarak çalıştırır.

Kullanıcı değeri yazılımla T2 zamanlayıcısını durdurur.

Böylece gecikmeli darbe elde edilir. Burada Tda değerinin ayarlanması kesme cevap süresini kapsamalıdır. Tetikleme olayının zamanını I2CR saklayıcısında tutarak ve bunu TO zamanlayıcısının o andaki değeriyle karşılaştırarak değişken olan kesme hizmet süresi belirlenir. Bu süre Tdla yı sabit tutmak için telafi edilebilir. Darbe genişliği doğal olarak kesme hizmet programı süresinden bağımsız olup sadece T2 ve R2 ye yerleştirilen değerlere bağlıdır.

SENKRON DARBE ÜRETME:



(1) Kullanıcı Yazılımı:

T2 ye Ta değeri yüklenir.

R2 ye Tb değeri yüklenir.

T2 ye kesme izni verilir.

TS0-TS3 resetlenir.

TS0 in tümleyen olarak çalışmasına izin verilir. Diğerlerinin izinleri kaldırılır. T2 zamanlayıcısı çalıştırılır.

(2) T2 Zamanlayıcısından Kesme Geldiğinde:

Donanım TS0 ı tümleyen olarak çalıştırır ve R2 nin içeriğini T2 ye kopyeler.

Kullanıcı Tc yi R2 ye yükler.

(3) T2 Zamanlayıcısından Kesme Geldiğinde:

Donanım TS0 yu tümleyen olarak çalıştırır ve R2 nin içeriğinin T2 ye kopyeler.

Kullanıcı Td yi R2 ye yükler ve

sadece TS1 in tümleyen olarak çalışmasına izin verir.

2. ve 3. adımlar tekrarlanır. Herbir kesmede yeni gecikme süresi R2 ye yüklenir. Her ikinci kesmede bir sonraki çıkış etkinleştirilir. Bu işlem dört çıkışın tamamında darbe elde edilinceye kadar tekrarlanır.

Giriş Kontrol sistemlerinde sayısal kontrolör ve

çıkış sürücüleri veri işleyici olarak mikrosistemli

mikrokontrolörli sistemler kullanılmaktadır.

Bu tezde bir CNC makinesinin temel fonksiyonları

geliştirilmiştir. Bu sistem yazılım yoluyla geliştirilmeye başlanmıştır.

17.1 Sistemde kullanılan birimler

• HPC 46005 : Mikrokontrolör

• 27256 : EPROM

• 5264 : RAM

• 2818 : EEPROM

Donanımda diğer önemli birimler

• Arayüz entegreli kontrolör

17.2 Sistemin entegre birimleri

• EPROM : 27256

• RAM : 5264

• EEPROM : 2818

• Arayüz entegreli kontrolör

17.3 Sistemin 17. BÖLÜM

H P C . 4 6 0 0 3 İ L E X Y Z K O O R D İ N A T

S İ S T E M K O N T R O L U

GİRİŞ: Kontrol sistemlerinde sayısal kontrolör ve birçok ölçme düzenlerinde veri işleyici olarak mikroişlemci veya mikrokontrolörlü sistemler kullanılmaktadır.

Bu tezde , bir CNC makinanın temel fonksiyonları gerçekleştirilmiştir. Bu sistem yazılım yoluyla geliştirilmeye açık hale getirilmiştir.

17.1 Sistemde kullanılan birimler

- HPC 46003 : Mikrokontrolör
- 27256 : EPROM
- 6264 : RAM
- 2816 : EEPROM

Donanımda adres çözücü olarak PAL (Programable Lojik Array) entegresi kullanılmıştır.

17.2 Sistemin adres alanı

EPROM : FFFF - 8000

RAM : 7FFF - 6000

EEPROM : 5FFF - 5800

ENCODER: 1800 - 1803

17.3 Sistemin tanıtımı

Sistem , genişletilmiş ROM'suz 8-bit (Expanded ROMless 8-bit) modunda tasarlanmıştır.Bu modda PORT A , adres / veri yolu olarak kullanılmaktadır.PORT B pinleri B10 , B11 ve B15 (ALE , RD , WR) , yol kontrolu için kullanılırlar . Pin B12 (HBE) yol kontrolu için kullanılmaz ve Vcc 'ye bağlanmıştır.

Okuma ve yazma işlemleri:

OKUMA : RD işareti , hem yonga içi hem de yonga dışı okuma yapabilmek için üretilir.HPC , bir adrese eriştiğinde PORT A'nın düşük anlamlı 8 bitinden veriyi kabul eder .

Yolun düşük anlamlı , byte'ı okuma işlemi sırasında yüksek empedans gösterir.Yüksek anlamlı byte'ı ise adresi tutar.

YAZMA : WR işareti , yonga içi veya yonga dışına yazma durumunda üretilir. PORT A'nın düşük anlamlı byte'ı üzerinden veri yazılır.Yüksek anlamlı byte'ı adres tutar.

Sistem 5 kısımdan oluşmaktadır.Bunlar sırasıyla :

- Mikroişlemci kısmı
- CRT kontrol kısmı
- Encoder arabirimi
- Step motor kontrol kısmı
- Tuş takımı

18 . B Ö L Ü M

VIDEO GRAFİK KARTI

Bir video görüntü ünitesi, bir bilgisayar tarafından oluşturulan çeşitli harf, sayı ve işaretleri bir ekranda göstermek için kullanılır. Arzu edilen karakterleri, monitörün çalışabileceği video işaretlerine dönüştüren elektronik devreleri de kapsar.

18.1 Görüntünün Oluşumu

Bir monitör gerçekte bir takım devreleri atılmış TV alıcısıdır. Monitör, sadece bir görüntü tübünü ve gerekli sürücü elektronik devreleri içerir ve asıl video işareti tarafından beslenir. Monitörün band genişliği, alışlagelmiş bir TV alıcısından çok daha geniştir. Tipik olarak, iyi bir monitörün band genişliği 20 MHz iken, TV alıcısının band genişliği 5.5 MHz dir (Bu da verici bant genişliğinin sınırıdır).

Televizyonda, video işareti bir taşıyıcı dalgayı modüle etmektedir, bu yüzden alıcıya gelen işarettten saf video işaretini elde edebilmek için bir alıcı ve kod çözücü kısımları gerekmektedir.

Bir görüntü 625 satırdan oluşmakta ve 25 Hz lik frekansta tekrarlanmaktadır. Bu frekans, insan gözünün, rahatsız edici

titremeleri algılamasını önleyecek kadar yüksektir. Her görüntü, raster adı verilen ve herbiri $312 \frac{1}{2}$ satırdan oluşan iki kısma bölünmüştür.

Rasterlerden biri, tüm çift olmayan satırlardan , ve diğeri çift satırlardan oluşur. Rasterler üzerindeki hareketli görüntüler böylece kırpışma olmayan tek bir duran görüntü gibi gözükcektir. Bu teknik , geçmeli tarama olarak bilinir ve prensibi Şekil 1a da gösterilmiştir. Diyagramdan görüleceği gibi , bir raster , yarım satır ile başlarken diğeri yarım satır ile sona ermektedir. Yarım satırla son bulma nedeniyle , raster senkronizasyon darbeleri , son satır senkronizasyon darbesinden sonra satır periyodu (Ekranı tam bir satırı tarayabilmek için geçen süre) süresinin tam sayısı kadar süre sonra ortaya çıkar , aksi halde , raster senkronizasyon darbeleri, satır periyodunun yarısı kadar süre sonra gözükcektir (Şekil 2). Yarım satırdaki bu farklılık , elektron demetinin , geri dönüşünden sonra , bir sonraki satırın hangi yükseklikten başlayarak yazacağını belirler. Bir yarım satır periyodu , tam olarak ekrandaki yarım satır yüksekliğine karşı düştüğünden , sonuç olarak iki rastere ilişkin satırlar kesinlikle birbirinin arasında gözükcektir.

Televizyonda kullanılan sistem budur, fakat elimizde duran bir görüntü (Örneğin sayılarla kaplı bir ekran) varsa , o zaman bu birbirinin içine geçmiş gözükten rasterler , rahatsız

edici bir "Atlama" etkisi oluřturur ki , bilgisayar sistemlerine iliřkin monitörlerde bu durumdan sakınılması gerekir.Bununla beraber bu etkinin oluřmasını önlemek için bir yöntem de mevcuttur.Elimizde ekranda yeterinden fazla bulunan satır sayısının yarısını kullanarak , aynı raster , katod ışınlı tüp üzerine saniyede elli defalık hız ile yazılabilir.

Bu işlem software yardımıyla raster senkronizasyon darbelerinin daima son satır senkronizasyon darbesinden sonra aynı zaman aralığı ile gözükmesini sağlatarak gerçekleştirilebilir. Buna geçmeli olmayan görüntü adı verilir ve bir normal TV alıcısı ile ya da bir monitör ile elde etmek mümkündür. Kırpıřmasız bir görüntü elde etmek için genellikle bu yöntem kullanılır (Şekil 1b).

Herbir karakter için çoğunlukla 5*7 ya da 7*9 luk noktalardan oluřan bir matris kullanılır. Ekrana harflerden ya da sayılardan oluřan bir satır yazma işlemi Şekil 3 deki gibi olmaktadır. Tüm bir sıradaki karakterler için belli bir anda , noktalardan oluřan sıranın biri yazılır. Böylece 5*7 lik bir matrisle , bir sıradaki karakterleri yazabilmek için 7 görüntü satırı gerekecektir. Şekil 3 de gösterilen kelimeyi yazmak için gerekli modülasyon ile donatılmış bir takım video işaretleri gösterilmiştir.Satır senkronizasyon darbesinden sonraki herbir darbe elektron demetinin ekranı aydınlatması anlamına

gelmektedir. Açık olması için darbeler taranmış ve karakterlerin ortaya çıkışını gösterebilmek için satırlar birbirine yaklaştırılmıştır. Karakterlerin birbirinden ayrılabilmesi için ekrandaki her iki satırlık karakterden sonra boş bir satır çizilmektedir. Gerçekte bir satırdaki karakterler için 9 görüntü satırı bulunmaktadır. Video kartı , ekrana normal olarak 24*80 adet karakter yerleştirir. Fakat bu sadece 216=24*9 tane satır gerektiği anlamına gelmelidir. Çünkü bu durumda ilk satır ekranın tam üst tarafında yer alacaktır. Karakterlerin kaybolmasını engellemek üzere ekranın yan taraflarında az bir miktar boş yere gereksinme vardır. Bizim tam olarak istediğimiz şey , ekranın ortasında karakterlerin görüleceği dikdörtgen bir alan parçasıdır. Bunun ekranda nasıl gözükeyeceği Şekil 4 de görülmektedir. Ekran üzerinde toplam olarak 297 satır (33 tane karakter satırı) ve 128 karakter yazılabilir. O halde ortadaki 80 karaktere ilişkin 216 satırlık bir alanı kullanmış oluruz. Diyagramda büyütülmüş olarak gösterilen kısım video kartının gerçek bir karakteri nasıl oluşturduğunu belirtmektedir. Buna göre karakterler için 5*8 lik bir matris , karakterler arasına 3 noktalıkbir boşluk ve herbir karakter satırı arasında boş bir görüntü satırı bulunmaktadır.

Bir monitörün band genişliğinin bir TV alıcısınıninkinden büyük olmasının nedeni : Normal bir TV de satır periyodu 64 mikro

saniyedir. Pratik olarak bu sürenin bir kısmı bile elektronik devrede kaybolmamaktadır. Çünkü böyle bir kayıp kompanze edilmektedir. Bir satırda yer alan 80 karaktere ilişkin herbir noktanın devam ettiği süre $64 \text{ us} / 128$ (satır başına düşen karakterlerin teorik sayısı) $\times 8$ (Karakter başına 5 nokta + 3 noktalık boşluk) = 62.5 ns. Senkronizasyon darbesine ilişkin süre , 128 adet teorik karakter içinde kapsamaktadır. Eger ekrandaki motif siyah-beyaz-siyah-beyaz..... şeklinde ise , en yüksek frekans ortaya çıkacaktır ki bu değer $1/(2 \times 62.5) = 8 \text{ MHz}$ olarak bulunur.

Bulunan bu sayısal değer siyah ve beyaz noktaların keskinliğini bile gözönüne almamıştır. Bu normal bir TV alıcısında 80×24 lük bir matriste yer alan karakterlerin kalitesinin çok iyi olmayacağı anlamına gelmektedir. O halde ya satır başına düşen karakterlerin sayısını azaltmalıyız ya da düzgün bir video girişi olan TV alıcısı kullanmalıyız. Çoğu kez arzu edilen diğer bir ekran görüntüsü grafik sembolleridir. Video kartı 8×8 bir matris üzerinde özel grafik karakterleri kullanmakta ve yatay olarak birbir peşisıra gözükmemektedir. Düşey doğrultuda , bu noktalar birbirleri üzerine de düşebilir. Çünkü boş satır iptal edilmiş ve ekranın en altına kaydırılmıştır. Öyle ki toplam satır sayısı aynı kalır.

18.2 Video kartının çalışma prensibi:

İlkönce video kartının görüntüyü nasıl oluşturduğuna bakmamız gerekir. Çünkü olay tam olarak Şekil 4a daki gibi değildir. Şekil 4b bir parça değişiktir ve 80*24 adet karakterin kartın toplam bellek alanında ne kadar yer kapladığını gösterir. Asli yazılan kısım başlangıçtır. Buna karşılık tüm boş yerler ekranın sağına ve sonlarına doğrudur. Bununla beraber ekranın tüm çevresinde kenarlarda boşluklar kalması istendiğinde bu işlem yatay ve düşey senkronizasyon darbelerini bellek alanında nerede yer alacağını belirtmekle sağlanabilir. Bu gerçekte adres alanının en dip kısmının ekranın üst tarafında olacağı anlamını taşır. Çünkü monitör , raster senkronizasyon darbesinden sonra gene ekranın üstünden itibaren yazmaya başlar. Benzer şeyler ekranın sol ve sağında bırakılacak boşluklar içinde geçerlidir. Fakat bu durumda boşlukla satır senkronizasyon darbesine bağlıdır. Sayısal trafik CRTC tarafından kontrol edilir.

Bu IC nin görevleri şunlardır:

- Ekranda yazılacak olan karakterin adresinin yerini belirlemek,
- Söz konusu karakteri ilgili nokta matrisine dönüştürmek,
- Uygun zamanlarda yatay ve düşey senkronizasyon darbelerini oluşturmak,
- Bir satırın matris noktalarını video girişine göndermek .

Ayrıca yatay ve düşey senkronizasyon darbeleri bir kompozit video işareti oluşturmak üzere birleştirilebilir. Kontrolörün ayrıca arzu edilen noktalar matrisini seçmek , satır başına düşen karakter sayısını ve görüntü başına düşen satır sayısını seçmek , geçmeli tarama ya da geçmeli olamayan tarama seçimini yapmak gibi diğer bazı işlevleride bulunmaktadır. Bu arada ekranda görülmekte olan kursorü sürmek ve lightpen'e gelen bağlantıları kontrol etmek gibi görevleride vardır.

16.3 Donanılar

Şekil 5 de video kartının blok şeması görülmektedir. Kart çok işlevli CRTC yanında bir video RAM i , bir karakter ROM u da içerir. Video RAM'i ekrana yazılabilecek tüm karakterleri saklar. Eğer ekrana 80*24 lük karakterler yazılacaksa o zaman $80*24=1920$ adet bellek gözüne gereksinim vardır. Bu yüzden 2K lık bir RAM kullanılmıştır.

ROM , grafik sembolleride dahil olmak üzere herbir karakteri oluşturan nokta düzeni hakkında bilgi içerir. CRTC adres ve veri yolu üzerinden video kartı ile geri kalan bilgisayar sistemi arasındaki haberleşmeyi kontrol eder. Ekranda çıkması gereken veri CRTC tarafından okunur video RAM indeki uygun bellek gözlerine yerleştirilir. RAM deki veriyi okuyabilmek için CRTC belleğin tüm adres alanını tarar ve böylece bir satırdaki 80 karakter birbirinin peşi sıra okunmuş olur. Veri karakter ROM una gider ve bu karakterlere ilişkin nokta düzenleri belirlenir.

Şekil 3 de bir karakterin 8 satır üzerine yazıldığı görülür. Video katında herbir 80 karakter serisi 8 defa okunur ve her defasında tekbir görüntü satırına ilişkin nokta düzeni okunmuş olur. Tek bir satıra ilişkin noktalar daha sonra bir ötelemeli yazıcıya gönderilir. Bu arada seri olarak çıkış yapılır. Bu çıkışın CRTC tarafından sağlanan senkronizasyon işareti ile birleşerek tam bir video işareti elde edilir.

18.3 Donanım

Video RAM ine ve CRTC ye ilişkin kod çözme işlemi ana karttaki kod çözücü devre tarafından gerçekleştirilir. Seçme işareti verildiğinde video RAM i mikroişlemci tarafından sistem yolu üzerinden adreslenir. Bu şekilde 6116 nın adres girişleri U12 ... U14 çoğullayıcılarının A girişleri üzerinden sistemin adres yoluna bağlanır. Aynı zamanda U10 veri yolu tamponu U02F ve U04A üzerinden yetkilendirilir. Mikroişlemci 6845 in saklayıcılarına erişmek için ana karttaki kod çözücü devreyi kullanır.

Böylece CRTC için gereken ilk değerler yazılmış olur. Mikroişlemci video RAM ine giriş yaptığında ekranda herhangi bir taramanın görünmemesi için ana karttaki kod çözücü devre yardımıyla U17 74LS175 flip-flop entegresini sınırlar. Kartın zamanlaması U03F ve U03E üzerine kurulmuş olan osilator tarafından

kontrol edilir. Bu osilatör nokta frekansı olarak adlandırılan ve değeri burada kullanılmış olan ekran düzeni için 15 MHz lik frekansı sağlar. U21 osilatör frekansını 8 e böler. Bu tüm-devre bir senkron sayıcı olup sayma işlemi 7 ye ulaştığında U06B tarafından sıfırlanmaktadır. Sıfırlama işlemi bir sonraki saat darbesinde yapıldığından U21 8 e kadar saymış olur. Bu 6845 için gerekli olan saat darbelerini sağlar. CRTC bu çıkış işaret frekansında 00 dan 7FF e kadar sürekli olarak sayar (Video RAM in tüm bölgesi). Şimdi işlemcinin video RAM e ulaşması söz konusu olmadığından U12 nin MA0 .. MA10 adres çıkışları 6116 nin adres - girişlerine çoğullayıcılar üzerinden bağlanır.

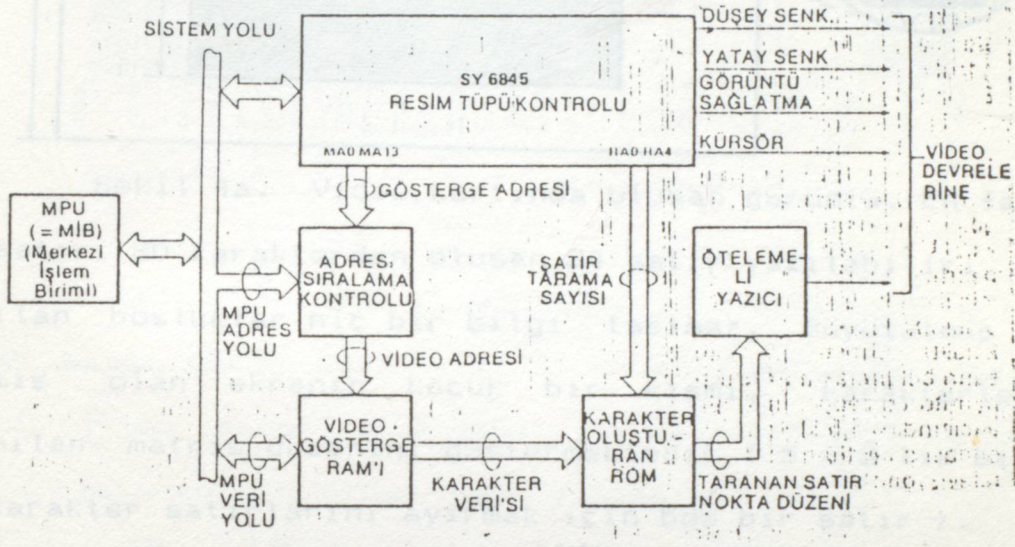
Böylece tüm RAM adreslerine sürekli olarak ulaşılmış olur. Bundan sonra RAM sürekli olarak veri sağlar ki bunlar U18 tutucusuna yerleştirilir. Tutucu ,RAM in çıkışlarında kararlı veri elde etmek için gereklidir. Ve bu koşul sağlanana kadar saat darbeleri uygulanmaz. Daha sonra RAM e diğer bir adres sağlandığı sırada tutucunun çıkış veriside aynı anda kullanılabilir. Tutucu saat darbesi U03D üzerinden sağlanır.

Tutucu içindeki bilgi IC19 karakter üretici için bir adres görevi görür. CRTC 2732 ye görüntülecek olan her karakterin satır adresleri (RA0 .. RA3) aynı anda sağlar öyle ki o anda ekrana yazılacak olan satırın noktaları bir anda video hattına verilmek üzere okunur. U20 nokta bilgisini paralelden seriye dönüştürür.

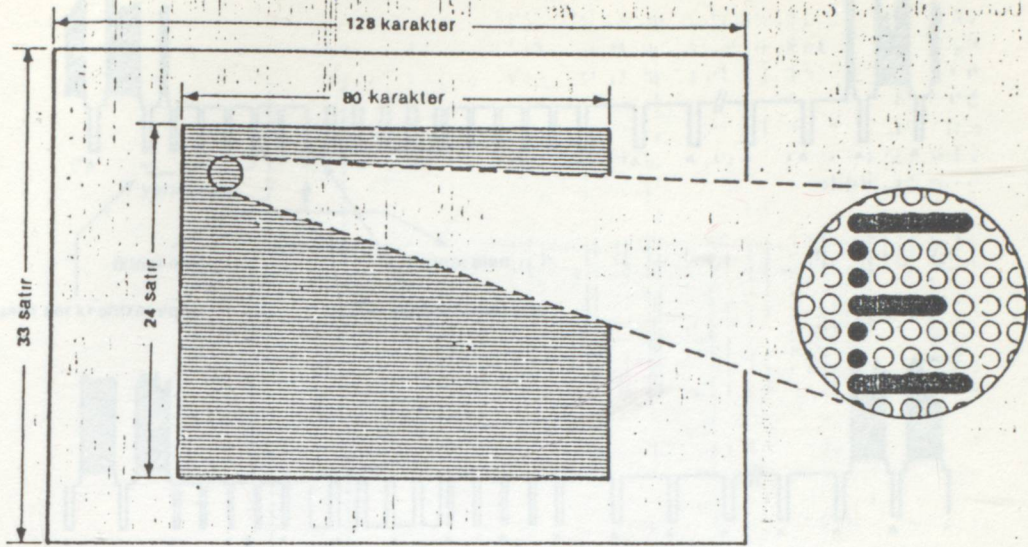
Kullanılan yüksek frekanslar ile herhangi bir zamanlama hatasının önüne geçmek üzere ötelemeli yazıcılar senkron olarak çalıştırılmışlar ve bunların saat işareti U03A ve U03B üzerinden osilatörden alınmıştır. Seri nokta bilgisi U20 nin Y çıkışında gözükür. U05A U06A U03C ve Q1 ile Q2 etrafında ki devreden oluşan video karıştırıcı katı U20 den gelen Y işareti ile CRTC tarafından sağlanan satır ve raster senkronizasyon darbelerini birleştirir. Önceden ayarlanan T1 ve T2 potansiyometreleri senkronizasyon darbelerinin büyüklüğünü ve nokta genliğini ayarlamakta kullanılır. CRTC nin ele alınması gereken iki önemli işareti daha mevcuttur. Bunlar DEN ve CUR olarak gösterilen işaretlerdir. CUR çıkışı ekrandaki kursorün konumunu ve DEN çıkışı CRTC nin ne zaman ekranın aktif bölgesinde olduğunu belirler. İkinci işaret ekranı aktif bölge dışında tamamen karanlık tutmak için gereklidir. Her iki işaret video işareti ile birleştirilmelidir. Fakat bu iş derhal yapılamaz. Çünkü RAM adres sağlanması ile nokta bilgisinin EPROM çıkışlarında görülmesi arasında da bir süre farkı bulunmaktadır. Bu gecikme birkaç yüz nano saniyedir ve bu yüzden CUR ve DEN işaretleri nokta işaretine göre oldukça erken ortaya çıkmış olacaktır.

Bu problemi halletmek için DEN ve CUR işaretleri nokta işareti ile karıştırılmadan önce iki bütün karakter süresi kadar geciktirilir. U05D in 12 nolu ucundaki köprüler kullanıcının

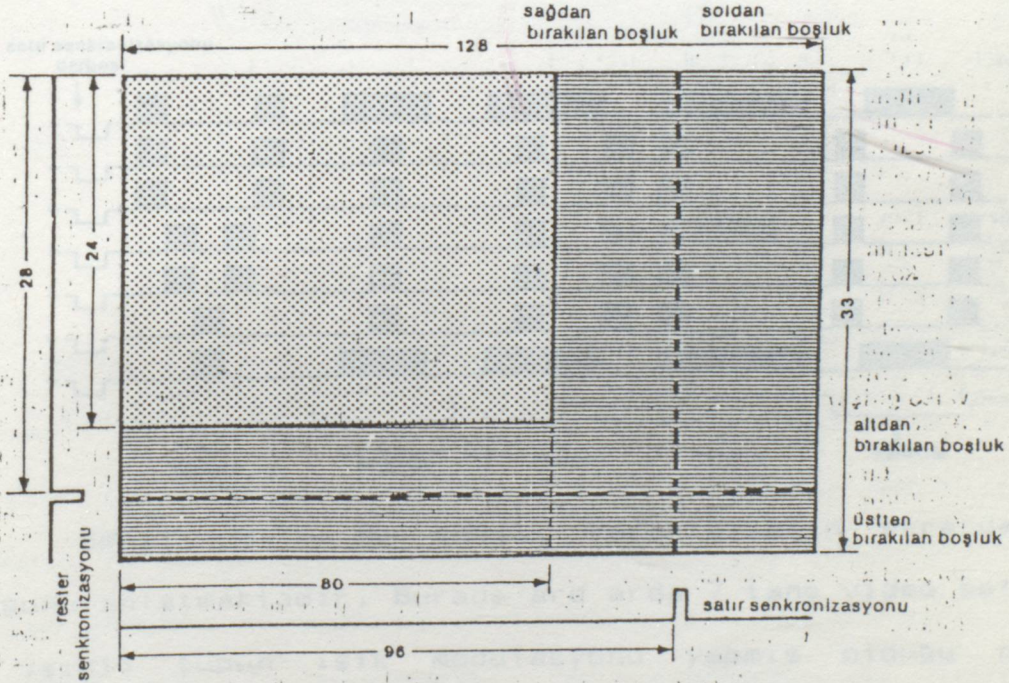
sü evrilmiş bir görüntü aydınlık fon için kullanılır



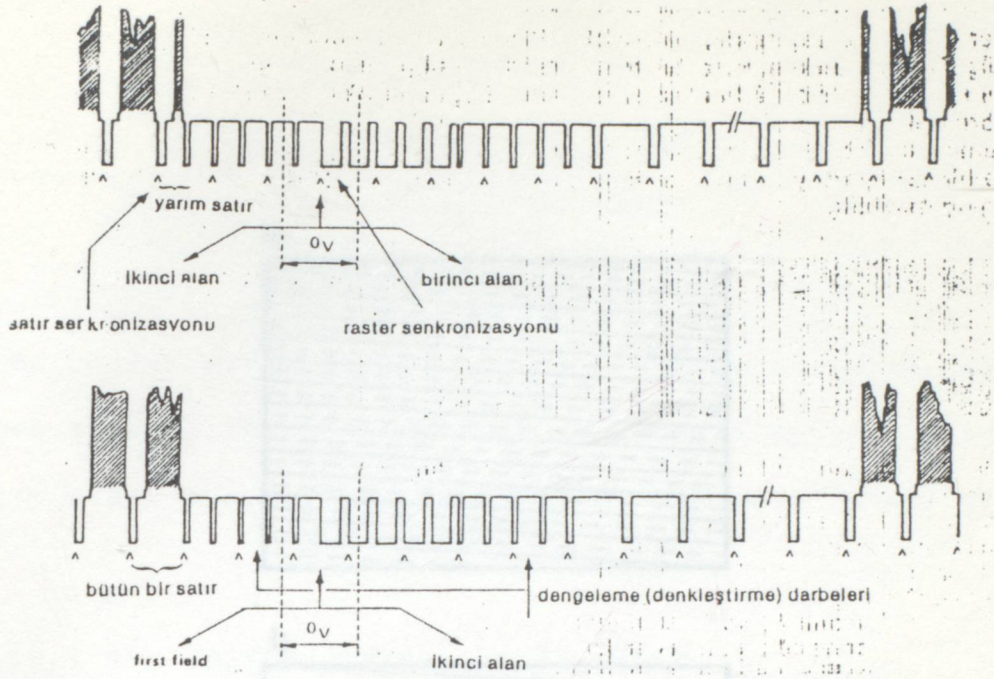
Şekil 5. Video kartının blok diyagramı. En önemli kısımlar kontrolör (CRTC), video RAM'i ve karakter ROM'u dur.



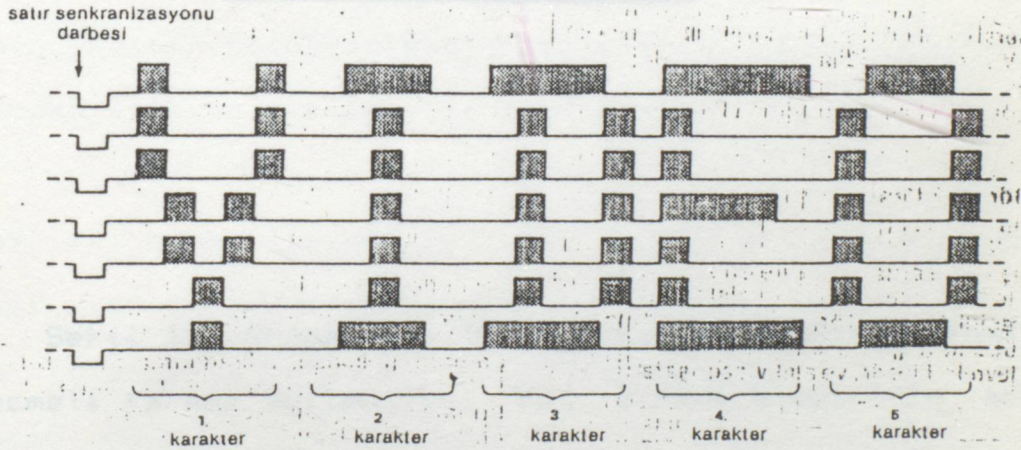
Sekil 4a. Video kartında oluşan görüntü. En fazla, her bir satırı 80 karakterden oluşan 24 satır yazılabilir. Yanlarda bırakılan boşluklar hiç bir bilgi taşımaz. Büyütülmüş biçimde çizilmiş olan ekranın küçük bir kısmı, karakterler için kullanılan matris düzenini göstermektedir (5 x 8 lik bir matris ile, karakter satırlarını ayırmak için boş bir satır).



Sekil 4b. CRTC kartının ekranı nasıl gördüğü anlatılmak isteniyor. Yazılacak kısım sol üst köşedeki kısımdır, diğer taraftan, görüntü yerinin geri kalan kısmı sağ alt tarafa doğru yer alır.



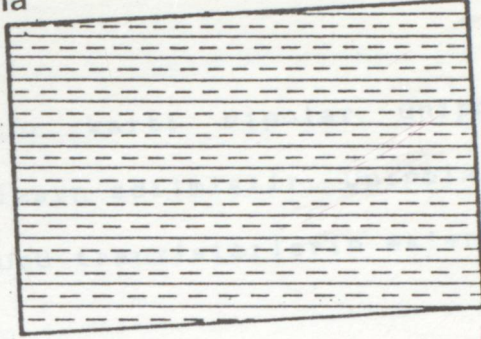
Şekil 2. Bir video işaretinin başlaması ve bitmesi (raster silme olarak da bilinir). Şekil a da birinci rasterin son buluşu (bu raster yarım satır ile bitmektedir) ve Şekil b de ikinci rasterin son buluşu görülmektedir.



Şekil 3. Bu şekil, karakterlerin ekranda nasıl oluştuğunu anlatmaktadır. Burada ard arda 7 tane video satırı ile katod ışıklı tüpün ışık modülasyonu yapmış olduğu noktalar görülmektedir. Satırları birbirine yakın yerleştirerek ve ışık modülasyonlarını karalayarak, ekrana yazılmak istenen kelimenin ne olduğunu anlayabiliyoruz.

19. BÖLÜM STEP MOTOR KARTI

1a



b



Sekil 1. Normal bir TV alıcısında görüntüyü oluşturmak için geçmeli tarama kullanılır. Bu, 2 tane birbirinin arasına geçen rasterin, ekrana birbirinin peşisıra yazılması demektir. Bu durum Sekil 1a da gösterilmiştir. Sekil 1b de, geçmeli tarama kullanmadan görüntü oluşmasını göstermektedir, burada aynı raster saniyede 50 defa yazılmaktadır. Bu şekilde, bir monitörde kırışmasız bir görüntü elde edilir.

19 . B Ö L Ü M

STEP MOTOR KARTI

Bu step motor kartı bipolar unipolar step motorlarını sürececek şekilde dizayn edilmiştir. Unipolar motorları sürmek için devreye diğer sürücü tranzistorlerin eklenmesi yeterlidir.

19.1 STEP MOTOR DEVRESİ VE ÇALIŞMASI

Bir adımlı motor, isminden de anlaşılabilceği gibi, sadece tekil dönme adımları atabilir. Bu amaçla, ayrı ayrı sürülebilmeleri için bobin uçlarının tümü dışarıya çıkartılmıştır. Bu arada iki tip adımlı motor bulunmaktadır, ve bu iki motor bacaklarının sayılarının farklı olması ile birbirinden kolayca ayrılabilir (Şekil 1).

Her iki motor tipinde de, tahrik edilen eksene bağlanmış olan bir çubuk mıknatıs -rotor- iki çapraz elektromıknatısın alanına göre yönelir. B tipi motorda (A tipinin aksine) bobinlerin birer orta ucuda vardır; bu orta uç magnetik alanın yönünün basit devreler yardımıyla değiştirilebilmesine olanak verir. Bir adımlı motorun çalışma ilkesini açıklamak amacıyla A tipi motoru ele alalım: Her bobinin ucuna birer komütatör konur.

Bu komütatör belirli bir konumda iken, motorun rotoru da yine belirli bir konumda sabit olarak durur. Buından sonra elektromagnetik alanın hareket yönüne göre motor bir adım sağa veya bir adım sola atabilir. 90 derece sağa dönmek için soldaki mıknatısın akım yönünü değiştirmek gerekir, bunun için de S1/S2'nin durumu değiştirilmelidir. Sonraki çeyrek tur için S3/S4'ün konumunu değiştirmek gerekir. Daha sonraki çeyrek tur için yeniden S1/S2 karşı konuma alınır v.s.

Besleme kaynağının pozitif ucu ile bağlantı sağlayan bir anahtar konumu "H" (high) besleme kaynağının negatif ucu ile bağlantı sağlayan bir anahtar konumu da "L" (low) ile belirtilirse, birbirini izleyen dört dönme adımı (360 derece) için şu doğruluk tablosu elde edilir:

Adım (onlu)	S1	S2	S3	S4
0	L	H	L	H
1	H	L	L	H
2	H	L	H	L
3	L	H	H	L

Bu tabloya dikkatlice baktığımızda /S1=S2 ve /S3=S4 olduğu hemen görülmektedir. O halde sadece S1 ve S3 göz önüne alınarak tablo basitleştirilebilir:

Adım (ikili)	S1	S3
0 0	L	L
0 1	H	L
1 0	H	H
1 1	L	H

Söz konusu dört adımı saymanın en basit yolu 2 Bit'lik bir ikili sayıcı kullanmaktır (Şekil 2). S3 anahtarı bu sayıcının en yüksek anlamlı biti ile senkron olarak çalıştırılır. S1 ise her iki sayıcı çıkışının IC 2a'da EXOR'lanması ile kazanılan işaret yardımıyla sürülür. IC 2'nin içindeki diğer kapılar da S2 ve S4 için evrilmiş işaretlemleri oluşturmada kullanılmaktadır. LS-TTL serisi lojik devre elemanları motoru çalıştırabilecek kadar çıkış akımı veremediklerinden, araya dört adet güç sürücü katının bağlanması gerekmektedir. A tipi motor için bobin başına bir köprü devre kullanılmalıdır, B tipi motor için basit anahtarlama katları yeterli olmaktadır.

Sayıcının U girişindeki TTL seviyesindeki darbeler motoru bir yönde, D girişindeki darbeler de diğer yönde çevirir. Dönme aralarında (boşluklarda) son olarak etkinleştirilen bobinlerden akım akmaya devam ettiğinden, motor erişilen konumda sabit tutulur. U+ noktası, motor besleme geriliminin bağlanacağı yerdir (A-tipi); B-tipi motorda ise besleme gerilimi alan sargılarının

orta noktalarına verilir. U_+ gerilimi 40 V'dan daha büyük olmamalıdır, 5 V'luk bir gerilim çoğunlukla yeterli olmaktadır. D1...D8 diyotları ile tranzistorlar, anahtarlama sırasında oluşan endüktif gerilim tepelerine karşı korunurlar; bu diyotlardan hiç bir zaman vazgeçilmemelidir. Ya doymada ya da kesimde çalıştırılan tranzistorların soğutulması için dörtlü grup başına 50 X 20 X 2 mm'lik Al-sac yeterlidir. Bu sırada tranzistor kılıflarına bağlı olan kolektörlerin birer mika parçası ile soğutucudan yalıtılmalarına dikkat edilmelidir.

Şekil 3'de genel amaçlı sürücüsünün baskı devreyerleştirme planı görülmektedir. B-tipi motorlarda devrenin kesikli çizgilerle ayrılmış sağdaki bölümünü yapmak gerekmez.

Adımlı motorların profesyonel uygulama alanları arasında basicıları (printer), çizicileri (plotter), ve endüstriyel robotları sayabiliriz.

20. BÖLÜM

KONTROL PROGRAMI

```

_bprflg:      .dsb      1
_othflg:      .dsb      1
_cntflg:      .dsb      1
_param_flg:   .dsb      1
_temp:        .dsb      1
_convflg:     .dsb      1
_bzper:       .dsb      1
_bzcnt:       .dsb      1
_bzflg:       .dsb      1
_sys_status:  .sect     base,rom16,abs=0x0
               .dsw      14
_lookup2:     .dsb      6
_lookup1:     .dsw      11
_multiplier:  .dsb      2
_kyb_table:   .dsb      2
_val:         .dsb      2
_col_buf:     .dsb      2
dispntr:      .dsb      2
_kats:        .dsw      1
_hiz:         .dsw      1
_kalipson:    .dsw      1
_Delay:       .dsw      1
_grup_max:    .dsb      2
_grup_min:    .dsb      2
_second:      .dsb      2
_msecond:     .dsb      2
_minute:      .dsb      2
_timer1:      .dsb      2
_timer2:      .dsb      2
_timer3:      .dsb      2
_timer4:      .dsb      2
_mengene_min: .dsb      2
_mengene_max: .dsb      2
_intcnt:      .dsb      2
_parca:       .dsb      2
_parsa:       .dsb      2
               .sect     data1,rom16,abs=0xe000
_keybuf:      .dsb      6
_latches1:    .dsb      1
_latches2:    .dsb      1
_latches3:    .dsb      1
_messcount:   .dsb      1
_messnum:     .dsb      1
_dignum:      .dsb      1
_shiftbuf:    .dsb      1
_digbuf:      .dsb      1
_diglim:      .dsb      1
_anykey:      .dsb      1
_dispflg:     .dsb      1
_tempflg:-    .dsb      1

```



```

oprflg:      .dsb      1
othflg:      .dsb      1
cntflg:      .dsb      1
param_flag:  .dsb      1
temp:        .dsb      1
convflg:     .dsb      1
bzper:       .dsb      1
bzcnt:       .dsb      1
bzflg:       .dsb      1
otoflg:      .dsb      1
ydrk:        .dsb      1
mdrk:        .dsb      1
sen_var:     .dsb      1
senbuf:      .dsb      1
malbuf:      .dsb      1
divbuf:      .dsb      1
row_pntr:    .dsw      1
col_pntr:    .dsb      1
key:         .dsb      1
funkey:      .dsb      1
kalipsayisi: .dsb      1
kalipflg:    .dsb      1
message:     .dsb      1
StpFlg:      .dsb      1
TylFlg:      .dsb      1
Mod_switch:  .dsb      1
SignFlg:     .dsb      1
V2:          .byte     0x3f,0x0
V3:          .byte     0x3,0x0
V4:          .byte     0xf,0x0
V5:          .byte     0x1c,0x0
V6:          .byte     0x34,0x0,0x0
V8:          .byte     0x0
stepileri:   .byte     0x04      ;B0100
stepgeri:    .byte     0x09      ;B1001
stepstop:    .byte     0x00      ;B0000

XFlg:        .dsw      1
YFlg:        .dsw      1
ZFlg:        .dsw      1

_kalipbas    =0x7100
_kaliplib    =0x7000
_valflatch1  =0x140A
_valflatch2  =0x140B
_mallatch    =0x1408
_steplatchX  =0x1410
_steplatchY  =0x1412

```



```

_steplatchZ      =0x1414
_optoaddr        =0x1416

manuel15: .endsect
           .extrn  PROGRAM_start
           .incld  HPC16083.MAP
manuel11: .sect   prog,rom16,abs=0xf000
_main:
           LD      PORTP.W,#0x11
           JSRL    ClrScr
           LD      _funkey.B,#0x0
           LD      _anykey.B,#0x0
           LD      _messnum.B,#0x0
           JSRL    _Mod_switch_rd
           IFEQ    Mod_switch.B,#0x1
           JMPL    Manuel_mod
manuel10: IFEQ    Mod_switch.B,#0x2
manuel16: JMPL    Oto_mod
           JMPL    _main

Manuel_mod:
           JSRL    ClrScr ;Clear display
           LD      dispntr.W,#_hizmod ;Hizmod mesajinin displayi
           JSRL    _dispmess ;
           JSRL    _keypress ;Keyboard .Scanning
           JSRL    _i2 ;Keyboard programi
manuel17: IFEQ    _key.B,#0x1
           LD      _Delay.W,#0x0
           IFEQ    _key.B,#0x2
           LD      _Delay.W,#0x3ff
manuel12: IFEQ    _key.B,#0x3
           LD      _Delay.W,#0x1ff
           LD      _funkey.B,#0x0
           LD      _key.B,#0x0
           LD      _anykey,#0x0
manuel13: LD      _funkey.B,#0x0
           LD      _key.B,#0x0
           LD      _anykey,#0x0
           JSRL    _dspdst
           LD      PORTP.W,#0x11 ;P1=0,P0=0
manuel16: JSRL    _keypress
manuel14: JSRL    _i2
           IFEQ    _funkey.B,#0x22 ;Stop tusu fonksiyonu
           JMPL    _main ;simdilik Escape tusu
           IFEQ    _funkey.B,#0xf2 ;Referans tusu fonksiyonu
           JMP     manuel5
           IFEQ    _funkey.B,#0xf3 ;X+ tusuna basilmis mi
           JMP     manuel1

```



```

IFEQ      _funkey.B,#0xf1          ;X- tusuna basilmis mi
JMP       manuel2
JMP       manuel3
manuel5:LD  _lookup1+0x0.W,#0X0     ;Referans tusu
LD        SignFlg.B,#0x0
JMP       manuel3
manuel1:JSRL _dspdst
LD        PORTP.W,#0X10           ;P1=0,P0=1 (X+)
JSRL      _delay
LD        PORTP.W,#0X11           ;P1=0,P0=0
IFEQ      SignFlg.B,#0x0          ;Onceden -X tusuna basilmis
JMP       man10
DECSZ     _lookup1+0x0.B
JMP       manuel8
LD        SignFlg.B,#0x0
JMP       manuel8
man10:INC  _lookup1+0x0.B
manuel8:JSRL _keydetect
IFEQ      _anykey.B,#0x0
JMP       manuel3
; JSRL     _i2
; IFEQ     _funkey.B,#0x22          ;Stop tusu fonksiyonu
; JMPL     _main                   ;simdilik Escape tusu
LD        _funkey.B,#0x0
LD        _anykey,#0x0
JMP       manuel1
manuel7:DECSZ _lookup1+0x0.B
JMP       manuel8
LD        SignFlg.B,#0x0
JMP       manuel8
manuel2:JSRL _dspdst
LD        PORTP.W,#0X01           ;P1=1,P0=0 (X-)
JSRL      _delay
LD        PORTP.W,#0X11           ;P1=0,P0=0
IFEQ      SignFlg.B,#0x1
JMP       manuel6
DECSZ     _lookup1+0x0.B
JMP       manuel4
LD        SignFlg.B,#0x1
JMP       manuel4
manuel6:INC  _lookup1+0x0.B
manuel4:JSRL _keydetect
IFEQ      _anykey.B,#0x0
JMP       manuel3
; JSRL     _i2
; IFEQ     _funkey.B,#0x22          ;Stop tusu fonksiyonu
; JMPL     _main                   ;simdilik Escape tusu
LD        _funkey.B,#0x0

```



```

main7: LD      _anykey,#0x0
      JMP      manuel2
Oto_mod: LD      dispntr.W,#_mesaj18
      JSRL     _Param_al
      JSRL     _HomeSystem          ;Mekanizmayı Origine
      JSRL     _dspdst              ;Mesafeleri displayd
      LD      _steplatchX.W,#stepileri ;X Step motorunu ile
      JSRL     _Encoder
      LD      _steplatchY.W,#stepileri ;Y Step motorunu ile
main6: *****
      JSRL     _VT_disp_con
      LD      _funkey.B,#0x0
      LD      _anykey.B,#0x0
      JSRL     _keydetect
      IFEQ     _anykey,#0x0
      JMP      CLM70
      LD      _anykey,#0x0
      JSRL     _keypress
      JSRL     _i2
      IFEQ     _funkey.B,#0x22      ;Stop tuşu fonksiyonu
      JMPL     _main               ;simdilik Escape tuşu
      *****
CLM70: JSRL     _VT_disp_con
      IFGT     _lookup1+0x0.B,_sys_status+0x0.B;X motoru istenen me
      JMP      main1                ;Evet
      IFGT     _lookup1+0x2.B,_sys_status+0x2.B;Y motoru istenen me
      JMP      main2                ;Evet
main1: LD      _steplatchX.W,#stepstop ;X motorunu durdur.
      JMP      main3
main2: LD      _steplatchY.W,#stepstop ;Y motorunu durdur
      JMP      main4
main3: LD      XFlg.B,#0x1
      IFEQ     YFlg.B,#0x1
      JMP      main5
      LD      _steplatchY.W,#stepileri;
      JMP      main6
main4: LD      YFlg.B,#0x1
      IFEQ     XFlg.B,#0x1
      JMP      main5
      LD      _steplatchX.W,#stepileri;
      JMP      main6
main5: LD      XFlg.B,#0x0
      LD      YFlg.B,#0x0
      LD      _steplatchZ.W,#stepileri; ;Z motoru ileri yonde
      IFGT     _lookup1+0x4.B,_sys_status+0x4.B;Z motoru istenen mes
      JMP      main7                ;Evet
      JMP      main5                ;Havin

```



```

main7: LD      ZFlg.B, #0x1
      LD      _steplatchZ.W, #stepstop;           ;Z motorunu durdur
      LD      _steplatchZ.W, #stepgeri;          ;Z motoru geri yonder
main9: IFEQ    _lookup1+0x4.B, #0x0
      JMP     main8
      JMP     main9
main8: JSRL    ClrScr
      JMP     _main

;      IFEQ    ProcFlg.B, #0x1
;      JSRL    _Encoder
;      JMP     _main
;      NOP

_Mod_switch_rd:
CC241: LD      A, #0x1
      JSRL    _wr_disp_com
      LD      A, #0x14
      JSRL    _wr_disp_com
      LD      dispntr.W, #_modmess
      JSRL    _dispmess
      JSRL    _keypress
      JSRL    _i2
      IFGT    _key.B, #0x3
      JP      CC241
      IFEQ    _key.B, #0x0
      JP      CC241
      LD      A, _key.B
      ST      A, Mod_switch.B
      LD      A, #0x1
      JSRL    _wr_disp_com
      LD      A, #0x14
      JSRL    _wr_disp_com
      LD      _key.B, #0x0
      JSRL    _debounce1
      RET

_HomeSystem:
      LD      _steplatchX.W, #stepgeri
      LD      _steplatchY.W, #stepgeri
HMS1:  IFBIT   1, 0x104.B
      JMP     HMS2
      JMP     HMS1
HMS2:  LD      _steplatchX.W, #stepstop
      LD      _steplatchY.W, #stepstop
      RET

      .public _Manuel_key
_Manuel_key:

```


; switch has 6 cases - min 20 max 25 range 6

```
CC242: LD      A, _funkey.B
      RC
      SUBC    A, #0x13
      IFGT    A, #0x5
CC247: JMP     CC62
      JID
      .pt     CJ64
      .pt     CJ65
CC246: .pt     CJ66
      .pt     CJ67
      .pt     CJ68
      .pt     CJ69
CJ64:  JMPL    CC64
CJ65:  JMPL    CC65
CJ66:  JMPL    CC66
CJ67:  JMPL    CC67
CJ68:  JMPL    CC68
CJ69:  JMPL    CC69
      JP      CC62
CC64:
      LD      _mdrk.B, #0x1
      JMPL    CC62
CC65:
      LD      _mdrk.B, #0x2
      JMPL    CC62
CC66:
      LD      _mdrk.B, #0x3
      JMPL    CC62
CC67:
      LD      _mdrk.B, #0x4
      JMPL    CC62
CC68:
      LD      _mdrk.B, #0x5
      JMPL    CC62
CC69:
      LD      _mdrk.B, #0x6
CC62:
      RET
ClrScr: LD      A, #0x1
      JSRL    _wr_disp_com
      LD      A, #0x14
      JSRL    _wr_disp_com
      RET
      .public _keydetect
_keydetect:
```



```

CC248:  IFBIT    7,0x104.B
        JMP     CC242
        JMP     CC251
CC242:
        LD      _intent.W,#0x0
CC247:  IFBIT    7,0x104.B
        JMPL    CC246
        JMPL    CC248
CC246:  LD      A,_intent.W
        IFEQ    A,#0x14
        JMPL    CC249
        JMPL    CC247
CC249:  LD      _anykey.B,#0x1
CC251:  NOP
        RET
        .public Param_place
Param_place:
        LD      A,_kaliplib.B
        ST      A,_kalipsayisi.B
        MULT    A,#100
        ADD     A,#_kalipbas
        ST      A,_kalipson
        LD      X,_kalipbas
        LD      A,_sys_status.B
CN130:  IFEQ    A,[X].B
        JP      CN120
        ADD     X,#100
        IFEQ    X,_kalipson
        JP      CC293
        JP      CN130
CN120:  LD      B,#00
CC292:  LD      A,[X.W].B
        ST      A,_sys_status[B].B
        INC     X
        INC     B
        LD      _kalipflg,#0x1
        IFGT    B,#0x1A
        JP      CC293
        JP      CC292
CC293:  NOP
        RET
        .public lsi_sens_oku
lsi_sens_oku:
        LD      _sen_var.B,#0x0
;        JSRL    _Sensor_oku
        LD      A,_senbuf.B
        ST      A,_lookup1+0xE.B

```



```

;      LD      _sen_var.B, #0x1
;      JSRL    _Sensor_oku
;      LD      A, _senbuf.B
;      ST      A, _lookup1+0x10.B

```

```

;      LD      _sen_var.B, #0x2
;      JSRL    _Sensor_oku
;      LD      A, _senbuf.B
;      ST      A, _lookup1+0x12.B
;      RET

```

```

;      .public _Sensor_oku
;_Sensor_oku:
;      LD      X, #_sensadr
;      ADD     X, _sen_var.B
;      LD      A, _sen_var.B
;      ST      A, [X.W].B
;      LD      BK, #0x1FF, #0x1FF
;wait10: DECSZ B
;      JMP     wait10
;      LD      A, [X.W].B
;      ST      A, _senbuf.B
;      RET

```

```

*****
reset:

```

```

      NOP
      LD      SP.W, #0x80
      LD      PSW.W, #0x10

```

```

*****

```

```

;      LD      _Que_In1, #_QueTop1
;      LD      _Que_Out1, #_QueTop1
;      LD      _Que_In2, #_QueTop2
;      LD      _Que_Out2, #_QueTop2
;      LD      _Que_In3, #_QueTop3
;      LD      _Que_Out3, #_QueTop3

```

```

;      LD      0xf4.W, #0x0000
;      LD      0xf2.W, #0xffff
;      LD      0x186.W, #0x1f4
;      LD      0x188.W, #0x1f4
;      LD      0xd2.B, #0x0
;      LD      0xd4.B, #0x4
;      LD      _key.b, #0
;      LD      _funkey.b, #0
;      LD      _intcnt.w, #0
;      LD      _ydrk.B, #0x0

```



```

LD      _param_flag.B, #0x1
LD      _dispflg.B, #0x0
LD      _othflg.b, #0x0
CN110: LD      _cntflg.B, #0x0
LD      _messnum.B, #0x0
LD      _messcount.B, #0x2
LD      _dignum.B, #0x0
LD      _shiftbuf.B, #0x0
LD      _diglim.B, #0x4
dispne LD      _kats.B, #0x1
;      LD      dispntr.W, #_mesaj1
CC1125: LD      _timer1.W, #0x0
LD      _timer2.W, #0x0
LD      _timer3.W, #0x0
LD      _oprflg.B, #0x0
LD      _tempflg.B, #0x0
CC1126: LD      _otoflg.B, #0x0
LD      SignFlg.B, #0x0
LD      _anykey.B, #0x0
CC1127: LD      _parsa.W, #0x0
LD      _mengene_min.W, #0x5
LD      _mengene_max.W, #0xf0
LD      _grup_max.W, #0x9
LD      _grup_min.W, #0x5
JSRL    _init_table
; dspost JSRL    _rslatch1
JSRL    _d_init
SBIT    3, BFUN.W
LD      PORTP.W, #0x0
JSRL    _resetall
LD      DIVBY.W, #00200      ;CKI/16
LD      TMMODE.W, #0xCDCC
LD      PWMODE.W, #0xCCCC
LD      ENIR.B, #021
AND     TMMODE.W, #0FBFF
JMP     _main

_resetall:
RBIT    0, 0xe3.B
RBIT    1, 0xe3.B
RBIT    5, 0xe3.B
RBIT    6, 0xe3.B
RET

_init_table:
LD      A, #0x0
LD      B, #0x0
CN100:  ST      A, _sys_status[B].W
INC     B
INC     B

```



```

        IFEQ      B,#0x1C
        JP        CN110
        JP        CN100
CN110:
        LD        _lookup1+0x0.W,#0x0
        LD        _lookup1+0x2.W,#0x0
        LD        _lookup1+0x4.W,#0x0
        LD        _lookup1+0x6.W,#0x0
        RET
_dispmess:
        LD        X,dispntr.w
CC1125:
        LD        A,[X].B
        IFEQ      A,#0x0
        JMPL      CC1127
        JSRL      _wr_disp_dta
CC1126:
        INC       X.W
        JMPL      CC1125
CC1127:
        INC       X.W
        LD        A.W,X.W
        ST        A.w,dispntr.W
        RET
        .public _dspdst
_dspdst:
        LD        A,#0x1
        JSRL      _wr_disp_com
        LD        A,#0x14
        JSRL      _wr_disp_com
        LD        dispntr.W,#_mesaj21
        JSRL      _dispmess
;
        LD        A,_sys_status+0x0.B
;
        JSRL      _convert
;
        JSRL      _dispkeybuf
        LD        A,#0x14
        JSRL      _wr_disp_com
        IFEQ      SignFlg.B,#0x1
        JSRL      Negatif
        LD        A,_lookup1+0x0.B
        JSRL      _convert
        JSRL      _dispkeybuf
        LD        A,#0x14

```



```

        JSRL      _wr_disp_com

        LD        dispntr.W, #_mesaj22
        JSRL      _dispmess

; E240:      LD        A, _sys_status+0x2.B
;           JSRL      _convert
;           JSRL      _dispkeybuf

        LD        A, #0x14
        JSRL      _wr_disp_com

        LD        A, _lookup1+0x2.B
        JSRL      _convert
        JSRL      _dispkeybuf

        LD        A, #0x14
        JSRL      _wr_disp_com

        LD        dispntr.W, #_mesaj23
        JSRL      _dispmess

;           LD        A, _sys_status+0x4.B
;           JSRL      _convert
;           JSRL      _dispkeybuf

        LD        A, #0x14
        JSRL      _wr_disp_com
        ;increment keybuf add,
        ;digit number

        LD        A, _lookup1+0x4.B
        JSRL      _convert
        JSRL      _dispkeybuf
        LD        _dispflg.B, #0x1
        RET

Negatif:
        LD        A, #0x2d
        JSRL      _wr_disp_dta
        RET

        .public _dispkeybuf
_dispkeybuf:
        LD        _digbuf.B, _dignum.B
        LD        X, #_keybuf
CE219:    LD        A, [X].B
        ADD        A, #0x30
        JSRL      _wr_disp_dta
        INC        X
        DECSZ     _digbuf.B

```



```

        JMP      CE219
        RET
        .public  _reskeybuf
_reskeybuf:
        LD       B, #0x6
        LD       X, #_keybuf
CE240:   LD       A, #0xCE
        ST       A, [X].B
        INC     X
        DECSZ   B.B
        JMP     CE240
        RET
        .public  _convert
_convert:
        LD       _convflg.B, #0x0
        LD       _dignum.B, #0x0
        LD       B.W, #0x0
        IFGT    A, #999
        JMP     CC226
CE227:   IFGT    A, #99
        JMP     CC228
        IFEQ    _convflg.B, #0
        JMP     CC229
        PUSH    A
        CLR     A
        ST      A, _keybuf[B].b
        POP     A
        INC     B ;Increment keybuf add.
        INC     _dignum ; " digit number
CC229:   IFGT    A, #9
        JMP     CC230
        IFEQ    _convflg.B, #0
        JMP     CC231
        PUSH    A
        CLR     A
        ST      A, _keybuf[B].b
        POP     A
        INC     B ;Increment keybuf add.
        INC     _dignum ; " digit number
        JMP     CC231
CC226:   DIV     A, #1000
        ST      A, _keybuf[B].b
        LD      A.W, X.W
        INC     B ;Increment keybuf add.
        INC     _dignum ; " digit number
        LD      _convflg.B, #0x1
        JMP     CC227
CC228:   DIV     A, #100

```



```

        ST      A,_keybuf[B].B
        LD      A,W,X.W
CD109:   INC      B                ;Increment keybuf add.
        INC      _dignum          ;      "      digit number
        LD      _convflg.B,#0x1
        JMP     CC229
CC230:   DIV     A,#10
        ST      A,_keybuf[B].B
        LD      A,W,X.W
        INC      B                ;Increment keybuf add.
        INC      _dignum          ;      "      digit number
        LD      _convflg.B,#0x1
CC231:   ST      A,_keybuf[B].B
        INC      _dignum          ;      "      digit number
CC232:   .public _Column
        .public _Keyboard
        .setall
CC98:   JSRL     _setall
; switch has 4 cases - min 48 max 51 range 4
        LD      A,_row_pntr.B
        IFEQ    A,#0x0
        JP      CC99
        IFEQ    A,#0x1
        JMW     CC101
        IFEQ    A,#0x2
        JMW     CC103
        IFEQ    A,#0x3
        JMW     CC105
        IFEQ    A,#0x4
        JMW     CC109
        JP      CC109
CC99:   RBIT     0,0xe3.B
        JMW     CC97
CC101:   RBIT     1,0xe3.B
        JMW     CC97
CC103:   RBIT     5,0xe3.B
        JMW     CC97
CC105:   RBIT     6,0xe3.B
CC97:   LD      _key.B,#0xff
        JSRL     _Column
        INC      _row_pntr.B
        IFEQ    _key.B,#0xff

```



```

CC122:    JP      CD109
          JMPL    CC109
CD109:    LD      _key.B, #0x0
          IFEQ    _funkey.B, #0x0
          JMPL    CC98
CC109:    RET
          .public _Column
_setall:
          SBIT    0, 0xe3.B
          SBIT    1, 0xe3.B
          SBIT    5, 0xe3.B
          SBIT    6, 0xe3.B
          RET
          .public _Column
_Column:
          LD      _col_pntr.B, #0x0
          LD      _col_buf.W, 0x104.B
          LD      A, _col_buf.W
          COMP    A
          AND     A, #0x007F
          IFEQ    A, #0x0
          JMPL    CC129
          LD      A, _col_buf.W
          COMP    A ;fonksiyon tuslari devreye girdigine
          AND     A, #0x007F ;003f ile AND islemi yapılacak
          ST      A, _col_buf.W ; ve son iki bit maskelenecek
; switch has 6 cases - min 49 max 13106 range 13058
          LD      A, _col_buf.W
          IFEQ    A, #0x1
          JMPL    CC122
          IFEQ    A, #0x2
          JMPL    CC123
          IFEQ    A, #0x4
          JMPL    CC124
          IFEQ    A, #0x8
          JMPL    CC125
          IFEQ    A, #0x10
          JMPL    CC126
          IFEQ    A, #0x20
          JMPL    CC127
          IFEQ    A, #0x40
          JMPL    CC233
          IFEQ    A, #0x80
          JMPL    CC234
          JMPL    CC129

```



```

CC122:      LD      _col_pntr.B,#0x1
            JMPL    CC120
CC123:      LD      _col_pntr.B,#0x2
            JMPL    CC120
CC124:      LD      _col_pntr.B,#0x3
            JMPL    CC120
CC125:      LD      _col_pntr.B,#0x4
            JMPL    CC120
CC126:      LD      _col_pntr.B,#0x5
            JMPL    CC120
CC127:      LD      _col_pntr.B,#0x6
            JMPL    CC120
CC233:      LD      _col_pntr.B,#0x7
            JMPL    CC120
CC234:      LD      _col_pntr.B,#0x8
CC120:      LD      A,_row_pntr.B
            MULT    A,#0x7
            ADDS    A,_col_pntr.B
            ST      A,_kyb_table.W
; switch has 22 cases - min 56 max 13105 range 13050
            LD      A,_kyb_table.W
            IFEQ    A,#0x1
            JMPL    CC131
            IFEQ    A,#0x2
            JMPL    CC132
            IFEQ    A,#0x3
            JMPL    CC133
            IFEQ    A,#0x4
            JMPL    CC134
            IFEQ    A,#0x5
            JMPL    CC135
            IFEQ    A,#0x6
            JMPL    CC136
            IFEQ    A,#0x7
            JMPL    CC137
            IFEQ    A,#0x8
            JMPL    CC138
            IFEQ    A,#0x9
            JMPL    CC139

```



```

CC134:  IFEQ    A, #0xa
        JMPL   CC140
CC135:  IFEQ    A, #0xb
        JMPL   CC141
CC136:  IFEQ    A, #0xc
        JMPL   CC142
        IFEQ    A, #0xd
        JMPL   CC143
CC137:  IFEQ    A, #0xe
        JMPL   CC144
        IFEQ    A, #0xf
        JMPL   CC145
        IFEQ    A, #0x10
        JMPL   CC146
CC138:  IFEQ    A, #0x11
        JMPL   CC147
        IFEQ    A, #0x12
        JMPL   CC148
CC139:  IFEQ    A, #0x13
        JMPL   CC149
        IFEQ    A, #0x14
        JMPL   CC150
CC140:  IFEQ    A, #0x15
        JMPL   CC151
        IFEQ    A, #0x16
        JMPL   CC152
CC141:  IFEQ    A, #0x17
        JMPL   CC153
        IFEQ    A, #0x18
        JMPL   CC154
CC142:  IFEQ    A, #0x19
        JMPL   CC155
        IFEQ    A, #0x1a
        JMPL   CC156
CC143:  IFEQ    A, #0x1b
        JMPL   CC157
        IFEQ    A, #0x1c
        JMPL   CC158
CC131:  LD      _key.B, #0x1           ;Key = 1
        JMPL   CC129
CC132:  LD      _key.B, #0x2           ;Key = 2
        JMPL   CC129
CC133:  LD      _key.B, #0x3           ;Key = 3
        JMPL   CC129

```



```

CC134: LD      _funkey.B, #0x20      ; ENTER
      JMPL   CC129

CC135: LD      _funkey.B, #0x5E      ; X
      JMPL   CC129

CC136: LD      _funkey.B, #0x14      ; F1
      JMPL   CC129

CC137: LD      _funkey.B, #0x15      ; F2
      JMPL   CC129

CC138: LD      _key.B, #0x4          ; Key = 3
      JMPL   CC129

CC139: LD      _key.B, #0x5          ; Key = 4
      JMPL   CC129

CC140: LD      _key.B, #0x6          ; Key = 5
      JMPL   CC129

CC141: LD      _funkey.B, #0x21      ; DELETE
      JMPL   CC129

CC142: LD      _funkey.B, #0xF1      ; Y
      JMPL   CC129

CC143: LD      _funkey.B, #0x16      ; F3
      JMPL   CC129

CC144: LD      _funkey.B, #0x17      ; F4
      JMPL   CC129

CC145: LD      _key.B, #0x7          ; Key = 6
      JMPL   CC129

CC146: LD      _key.B, #0x8          ; Key = 7
      JMPL   CC129

CC147: LD      _key.B, #0x9          ; Key = 8
      JMPL   CC129

CC148: LD      _funkey.B, #0x22      ; ESCAPE
      JMPL   CC129

CC149: LD      _funkey.B, #0xF3      ; P
      JMPL   CC129

```



```

CC150:      LD      _funkey.B,#0x1B      ; F5
           JMPL    CC129
CC151:      LD      _funkey.B,#0x19      ; F6
           JMPL    CC129
CC152:      LD      _key.B,#0x0          ;Key = 0
           JMPL    CC129
CC153:      LD      _key.B,#0x0          ;_funkey.B,#0x24
           JMPL    CC129
CC154:      LD      _funkey.B,#0x25      ;
           JMPL    CC129
CC155:      LD      _funkey.B,#0x23      ;STORE
           JMPL    CC129
CC156:      LD      _funkey.B,#0xF2      ; Q
           JMPL    CC129
CC157:      LD      _funkey.B,#0x1A      ; F7
           JMPL    CC129
CC158:      LD      _funkey.B,#0x1B      ; F8
           JMPL    CC129
CC129:      RET
_wr_disp_com:
LD      BK,#0x1FF,#0x1FF
wait1:  DECSZ    B
           JMP     wait1
CC159:      NOP
           NOP
           NOP
           NOP
           NOP
;         IFBIT   7,0x1802.B
;         JMPL    CC159
CC160:      ST      A,0x1000.B
           RET
_wr_disp_dta:
LD      BK,#0x1FF,#0x1FF
wait2:  DECSZ    B
           JP      wait2

```



```

CC166:    NOP      _dispmem
          NOP      _scrflg.B, #0x0
          NOP      _wr_bld_val
CC173:    NOP      _keypress
CD171:    NOP      _i2
;         IFBIT    7, 0x1802.B, #0x0
;         JMWPL    CC166
CC167:    ST      A, 0x1001.B
          RET
          JMP      CC172
_d_init: LD      A, V2.W
          JSRL     _wr_disp_com
          LD      A, V2.W
          JSRL     _wr_disp_com
CC172:    LD      A, V2.W
          JSRL     _wr_disp_com
          LD      A, V3.W
          JSRL     _wr_disp_com
CC221:    LD      A, V4.W
          JSRL     _wr_disp_com
          LD      A, V5.W
          JSRL     _wr_disp_com
          LD      A, V6.W
          JSRL     _wr_disp_com
CD178:    LD      A, #0x1
          JSRL     _wr_disp_com
          RET
_keypress:
;         JSRL     _rslatch1
CC290:    IFBIT    7, 0x104.B
;         IFEQ     0x0e.W, #0x0
          JMWPL    CC174
          JMWPL    CC290
CC174:    LD      _intent.W, #0x0
CD192:    IFBIT    7, 0x104.B
          JMWPL    CD190
          JMP      CC290
CD190:    LD      A, _intent.W
          IFEQ     A, #0x14
          JMWPL    CC240
          JMWPL    CD192
CC240:    JSRL     _debounce
;         RET

```



```

_Param_al:
    JSRL    _dispmess
    IFEQ    _othflg.B,#0x0
    JSRL    _wr_old_val
CC173:    JSRL    _keypress
CD191:    JSRL    _i2
    IFGT    _funkey.B,#0x0
    JMPL    CD178
;    IFEQ    _shiftbuf.B,#0x0
;    JMP     CC710
;    JSRL    _shift
;    JMP     CC712
CC710:    LD      X,#_keybuf
    ADD     X,_dignum.B
    LD      A,B,_key.B
    ST      A,[X].B
    INC     _dignum.B
CC712:    IFGT    _dignum.b,_diglim.b
    JMP     CC221
    JMP     CC222
CC221:    JSRL    _digcont
;    LD      _key.B,#0x0
    JMP     CD178
CC222:    LD      A,_key.B
    ADD     A,#0x30
    JSRL    _wr_disp_dta
;    LD      _key.B,#0x0
CD178:    IFEQ    _funkey.B,#0x20      ; Enter
    JP      CC181
    IFEQ    _funkey.B,#0x21      ; Delete
    JP      CD182
    IFEQ    _funkey.B,#0x22      ; Escape
    JP      CD183
    IFEQ    _funkey.B,#0x23      ; Store
    JP      CD184
;    IFEQ    _funkey.B,#0x5E      ; X Fonk.
;    JP      CD185
;    IFEQ    _funkey.B,#0xF2      ; Y Fonk.
;    JP      CD186
;    IFEQ    _funkey.B,#0xF3      ; P Fonk.
;    JP      CD187
;    IFEQ    _funkey.B,#0xF1      ; Q Fonk.
;    JP      CD188
    NOP
    JSRL    _debounce1
    JMPL    CC173
CD182:    IFEQ    _dignum.B,#0x0

```



```

        JMP      CC213
;       IFEQ     _shiftbuf.B,#0x0
;       JMP      CC213
        LD       A.B,#0x4          ;Cursor Decrement yonunde
        JSRL     _wr_disp_com
        LD       A.B,#0x20         ;Cursorun oldugu yere boslu
        JSRL     _wr_disp_dta
        LD       A.B,#0x20         ;Cursorun oldugu yere boslu
        JSRL     _wr_disp_dta.B
CD130:  LD       A.B,#0x6          ;Cursor Increment yonunde
        JSRL     _wr_disp_com
        LD       A.B,#0x14
        JSRL     _wr_disp_com
        DECSZ    _dignum.B
        NOP
CD70:   JSRL     _debounce1
        LD       _funkey.B,#0x0
        JMPL     CC173
CC213:  PUSH
        LD       _funkey.B,#0x0
        JSRL     _debounce1
        JMPL     CC173
CD183:  JSRL
CD20:   IFEQ     _dignum.B,#0x0
        JMPL     CC214
        LD       A.B,#0x4
        JSRL     _wr_disp_com
CC212:  LD       A.B,#0x20
        JSRL     _wr_disp_dta
        DECSZ    _dignum
        JP       CC212
        LD       A.B,#0x20
        JSRL     _wr_disp_dta
        LD       A.B,#0x6
        JSRL     _wr_disp_com
        LD       A.B,#0x14
        JSRL     _wr_disp_com
CC214:  LD       _shiftbuf.B,#0x0
        LD       _funkey.B,#0x0
        JSRL     _debounce1
        JMPL     CC173
CD184:  JSRL     _store
        JP       CC183
        .public _store
_store:
str:    LD       _funkey.B,#0x0
        LD       _key.B,#0x0

```



```

LD      A,_kaliplib.B
IFEQ    _kaliplib.B,#0x0
CS60:   JMP      CS70
CS30:   ST       A,_kalipsayisi.B
MULT    A,#100
ADD     A,#_kalipbas
ST      A,_kalipson
LD      X,#_kalipbas
LD      A,_sys_status.B
CS130:  IFEQ    A,[X].B
JP      CS120
ADD     X,#100
CS185:  IFEQ    X,_kalipson
JP      CS70
JP      CS130
CS70:   PUSH    X
INC     _kaliplib.B
JMP     CS10
CS120:  PUSH    X
LD      A.B,#0x1
JSRL    _wr_disp_com
CS186:  LD      A.B,#0x14
JSRL    _wr_disp_com
CS20:   LD      dispntr.W,#_errmess3
JSRL    _dispmess
JSRL    _debounce
JSRL    _keypress
JSRL    _i2
IFGT    _funkey.B,#0x0
CS187:  JP      CS60
IFEQ    _key.B,#0x0
JP      CS60
IFEQ    _key.B,#0x1
JP      CS10
IFGT    _key.B,#0x1
JP      CS60
JP      CS20
CS10:   LD      A.B,#0x1
JSRL    _wr_disp_com
LD      A.B,#0x14
JSRL    _wr_disp_com
POP     X
LD      B,#00
CC294:  LD      A,_sys_status[B].W
ST      A,[X.W].B
INC     X
INC     B
IFGT    B,#0x1A

```



```

        JF      CS30
        JF      CC294
CS60:   POP     X
CS30:   LD      A,B,#0x1
        JSRL   _wr_disp_com
        LD      A,B,#0x14
        JSRL   _wr_disp_com
        LD      _dispflg.B,#0x0
;        JSRL   _disptemp
        NOP
        RET
        B, _messnum.B
        B, #0x2
CD185:  ST      A.W, _sys_status[B].W
        LD      _funkey.B, #0x0
        IFEQ    _param_flag.B, #0x1      ; İlk parametre alinmada ipt
        JMWPL   CC173
        IFEQ    _messnum.B, #0x0
        JMWPL   CC184
        DECSZ   _messnum.B
        NOP
        JMWPL   CC184
CD186:  LD      _funkey.B, #0x0
        IFEQ    _param_flag.B, #0x1      ; İlk parametre alinmada ipt
        JMWPL   CC173
        IFGT    _messnum.B, _messcount
        JMWPL   CC184
        INC     _messnum.B
        JMWPL   CC184
CD187:  LD      _funkey.B, #0x0
        IFGT    _shiftbuf.B, _dignum.B
        JMP     CC173
        INC     _shiftbuf.B
        LD      A, #0x14
        JSRL   _wr_disp_com
        JMWPL   CC173
CD188:  LD      _funkey.B, #0x0
        IFEQ    _shiftbuf.B, #0x0
        JMWPL   CC173
        DECSZ   _shiftbuf.B
        NOP
        LD      A, #0x10
        JSRL   _wr_disp_com
        JMWPL   CC173
CD189:  LD      _funkey.B, #0x0

```



```

LD      _val.W,#0x0
CC270: LD      _shiftbuf.B,#0x0
IFEQ    _dignum.B,#0x0
JMP      CC173
_delay: DECSZ   _dignum
NOP
bekle1: JSRL    _valcmpt
LD      A.W,_val.W
IFEQ    _othflg.b,#0x1
CC270: JMPL    CC182
LD      B,_messnum.B
MULT    B,#0x2
ST      A.W,_sys_status[B].W
CC180:
; LD      _val.W,#0x0
; LD      _shiftbuf.B,#0x0
IFEQ    _ydrk.B,#0x1
JMPL    CC182
_shift: INC      _messnum.B
LD      A,_messnum.B
IFGT    A,_messcount.B
JMPL    CC182
CC183:
LD      _funkey.B,#0x0
LD      A,#0x1
_wr_01: JSRL    _wr_disp_com
LD      A,#0x14
JSRL    _wr_disp_com
CC184: JSRL    _debounce1
JSRL    _Param_a1
CC182:
LD      _funkey.B,#0x0
LD      _param_flag.B,#0x0
RET
.public _debounce
_debounce:
LD      _intcnt.W,#0x0
bekle1: IFEQ    _intcnt.W,#0x3ff
JP      CC250
JP      bekle1
CC250: NOP
RET
.public _debounce1
_debounce1:
LD      _intcnt.W,#0x0
bekle2: IFEQ    _intcnt.W,#0xff
JP      CC270
JP      bekle2

```



```

CC270:  NOP
        RET
        .public _delay
_delay:  LD      _intcnt.W,#0x0
bekle3:  IFGT    _intcnt.W,_Delay.W
        JP      CLC270
        JP      bekle3
CLC270:  NOP
        RET

        .public _resetbuzz
_resetbuzz:
NOP
RET

        .public _shift
_shift:  LD      X,#_keybuf
        ADD     X,_shiftbuf.B
        LD      A,B,_key.B
        ST      A,[X].B
        RET
        .public _wr_old_val
_wr_old_val:
LD      B,_messnum.B
MULT    B,#0x2
LD      A,_sys_status[B].W
IFEQ    A,#0x0
JMP     CE220
JSRL    _convert
; IFEQ    _shiftbuf.B,#0x0
; JP      CC702
; JP      CC704
; CC702: LD      _shiftbuf.B,_dignum.B
CC704:  JSRL    _dispkeybuf
CE220:  RET
        .public _digcont
_digcont:
DECSZ   _dignum.B
NOP
; IFEQ    _shiftbuf.B,#0x0
; JP      CC722
; LD      _digbuf.B,_shiftbuf.B
; JP      CC720
; CC722: LD      _digbuf.B,_dignum.B
CC720:  LD      K,#_keybuf

```



```

        ADD     K, _diglim
        ADD     K, #0x1
        LD      X, #_keybuf
        LD      B, #_keybuf
        ADD     B, #1
CC215:   LDS     A, [B+].B
        JMP     CC216
        JMP     CC220
CC216:   ST      A, [X].B
        INC     X
        JP      CC215
CC220:   LD      A.B, #0x4
        JSRL    _wr_disp_com
CC218:   LD      A.B, #0x10
        JSRL    _wr_disp_com
        DECSZ   _digbuf.B
        JMP     CC218
        LD      A.B, #0x6
        JSRL    _wr_disp_com
        LD      _digbuf.B, _dignum.B
        LD      X, #_keybuf
CC219:   LD      A, [X].B
        ADD     A, #0x30
        JSRL    _wr_disp_dta
        INC     X
        DECSZ   _digbuf.B
        JMP     CC219
        RET
        .public _valcmpt
_valcmpt:
        LD      _multiplier.W, #0x1
        LD      _val.W, #0x0
CC211:   LD      X, #_keybuf
        ADD     X, _dignum.B
        LD      A.W, [X].B
        MULT    A, _multiplier.W
        ADD     A.W, _val.W
        ST      A.W, _val.W
        LD      A, _multiplier.W
        MULT    A, #0xa
        ST      A, _multiplier.W
        IFEQ    _dignum.B, #0x0
        JP      CC210
        DECSZ   _dignum.B
        NOP
        JMP     CC211
CC210:   RET

```



```

        .public _i2
_i2:
        .IPT      2, _i2
;        function_entry CC185
        PUSH      PSW.W
        PUSH      K
        PUSH      X
        PUSH      B
        PUSH      A
;        LD        0xd0.B, #0x0
;        LD        A, 0xd2.B
;        AND       A, #0x4
;        SHR       A
;        SHR       A
;        PUSH      A
;        IFEQ      _intcnt.W, #0x0; 14
;        JF        CC187
;        CLR       A
;        JF        CC188
CC187:
;        LD        A, #0x1
CC188:
;        POP       K
;        AND       A, K
;        IFEQ      A, #0x0
;        JMPL      CC186
;        LD        _intcnt.W, #0x0
CC192:
;        LD        _col_pntr.B, #0x0
;        LD        _row_pntr.B, #0x0
;        JSRL      _Keyboard
;        JSRL      _resetall
;        LD        0xd2.B, #0x0
CC186:
;        LD        0xd0.B, #0x21
;        POP       A
;        POP       B
;        POP       X
;        POP       K
;        POP       PSW.W
;        function_exit CC185
;        RET; I
        .public _intserv
_intserv:
;        function_entry CC190, CC191
;        PUSH      PSW.W
;        PUSH      K
;        PUSH      X
;        PUSH      B

```



```

PUSH    A
LD      TMMODE.W, #0xCECC
INC     _intcnt.W
IFBIT   2, 0xd2.B ; INT2 Int. girisinden
LD      StpFlg.B, #0x2
IFBIT   3, 0xd2.B ; INT3 Int. girisinden
LD      StpFlg.B, #0x3
IFBIT   4, 0xd2.B ; INT4 Int. girisinden
LD      StpFlg.B, #0x4
LD      0XD2.B, #0X0 ; Clear all int pending flag
JSRL    IncHome
JSRL    _PutQue
LD      0xd0.B, #0x0
LD      A, 0xd2.B
AND     A, #0x20
LD      K, #0x20
DIV     A, K
PUSH    A
LD      A, 0x190.W
AND     A, #0x200
LD      K, #0x200
DIV     A, K
POP     K
AND     A, K
IFEQ    A, #0x0
JMPL    CC192
SBIT    3, 0x191.B
JSRL    _Real_time
CC192:
LD      TMMODE.W, #0xCBCC
LD      PSW.B, #0x10
LD      0xd0.B, #0x21
POP     A
POP     B
POP     X
POP     K
POP     PSW.W
; function_exit CC190, CC191
RETI

IncHome:
IFEQ    StpFlg.B, #0x2
INC     _lookup1+0x0.B
IFEQ    StpFlg.B, #0x3
INC     _lookup1+0x2.B
IFEQ    StpFlg.B, #0x4
INC     _lookup1+0x4.B
LD      StpFlg.B, #0x0
RET

```



```

        .public _Real_time
_Real_time:
;      function_entry CC194, CC195
        INC      _msecond.W
        INC      _timer1.W
        IFEQ     _msecond.W,#0x3c
        JP      CC197
        JP      CC196
CC197:
        LD      _msecond.W,#0x0
        INC      _second.W
        INC      _timer3
        INC      _timer4
CC196:
        IFEQ     _second.W,#0x3c
        JP      CC200
        JP      CC198
CC200:
        LD      _second.W,#0x0
        INC      _minute.W
        LD      _second.W,#0x0
CC198:
;      function_exit CC194, CC195
        RET

        .public _mesaj18      ;X Mesafesini veriniz
_mesaj18:
        .byte 058,020,04d,065,073,061,066,065
        .byte 073,069,06e,069,020,076,065,072
        .byte 069,06e,069,07a,03d,00
        .public _mesaj19      ;(Y Mesafesini veriniz)
_mesaj19:
        .byte 059,020,04d,065,073,061,066,065
        .byte 073,069,06e,069,020,076,065,072
        .byte 069,06e,069,07a,03d,00
        .public _mesaj20      ;(Z Mesafesini veriniz)
_mesaj20:
        .byte 05a,020,04d,065,073,061,066,065
        .byte 073,069,06e,069,020,076,065,072
        .byte 069,06e,069,07a,03d,00
        .public _mesaj21      ;(X = )
_mesaj21:
        .byte 058,020,03d,020,00
        .public _mesaj22      ;(Y = )
_mesaj22:
        .byte 059,020,03d,020,00
        .public _mesaj23      ;(Z = )
_mesaj23:
        .byte 05a,020,03d,020,00

        .public _errmess1
_errmess1:
        .byte 053,069,063,061,06b,06c,069,06b
        .byte 020,069,073,074,065,06e,065,06e
        .byte 020,073,065,076,069,079,065,064
        .byte 065,020,064,065,067,069,06c,00

```



```

_errmess2:    .byte    041,072,061,06e,061,06e,020,06b
               .byte    061,06c,069,070,020,062,075,06c
               .byte    075,06e,061,06d,069,073,074,069
               .byte    072,02e,00
_errmess3:    .byte    042,075,020,06b,061,06c,069,070
               .byte    020,06d,065,076,063,075,074,074,075,072
               .byte    028,030,080,044,065,076,061,06d,02f
               .byte    031,0b0,020,04b,061,0f9,069,074,029,00
               .public   _isitici1
_isitici1:    .byte    049,073,069
               .byte    031,03d,00
               .public   _isitici2
_isitici2:    .byte    049,073,069
               .byte    032,03d,00
               .public   _isitici3
_isitici3:    .byte    049,073,069
               .byte    033,03d,00
               .public   _modmess
_modmess:     .byte    04d,06f,064,020,028,031,03a,04d
               .byte    061,06e,075,065,06c,02c,032,03a,04f,074
               .byte    06f,06d,061,074,069,06b,029,020,00,00
_negatif:     .byte    02d,00
               .public   _hizmod
               ;{"1-H i z l i ,2-Y a v a s ,3-N o r m a l "}
_hizmod:     .byte    031,02d,048,020,069,020,07a,020
               .byte    06c,020,069,020,02e,032,02d,059
               .byte    020,061,020,076,020,061,020,073
               .byte    020,02c,033,02d,04e,020,06f,020
               .byte    072,020,06d,020,061,020,06c,020
               .byte    00
               .SECT     INTSECT,ROM16,ABS=0FFF4
               .WORD     _intserv
               .end       reset
               .SECT     INTSECT,ROM16,ABS=0FFFA
               .WORD     _i2
               .end

```


REFERANSLAR

1 - John B. Peatman , " Design with Microcontrollers "

(McGraw-Hill 1988)

2 - John Uffenbeck , " Microcomputers and Microprocessors "

(Prentice-Hall 1985)

3 - National Semicon. Co. , " The HPC User's Manuel "

(1987)

4 - National Semicon. Co. , " Microcontrollers Databook "

(1988)

5 - Fink and Christiansen , " Electronics Engineers' Handbook "

(McGraw-Hill 1987)

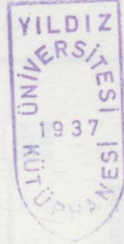
6 - Benjamin Kuo , " Automatic Control Systems "

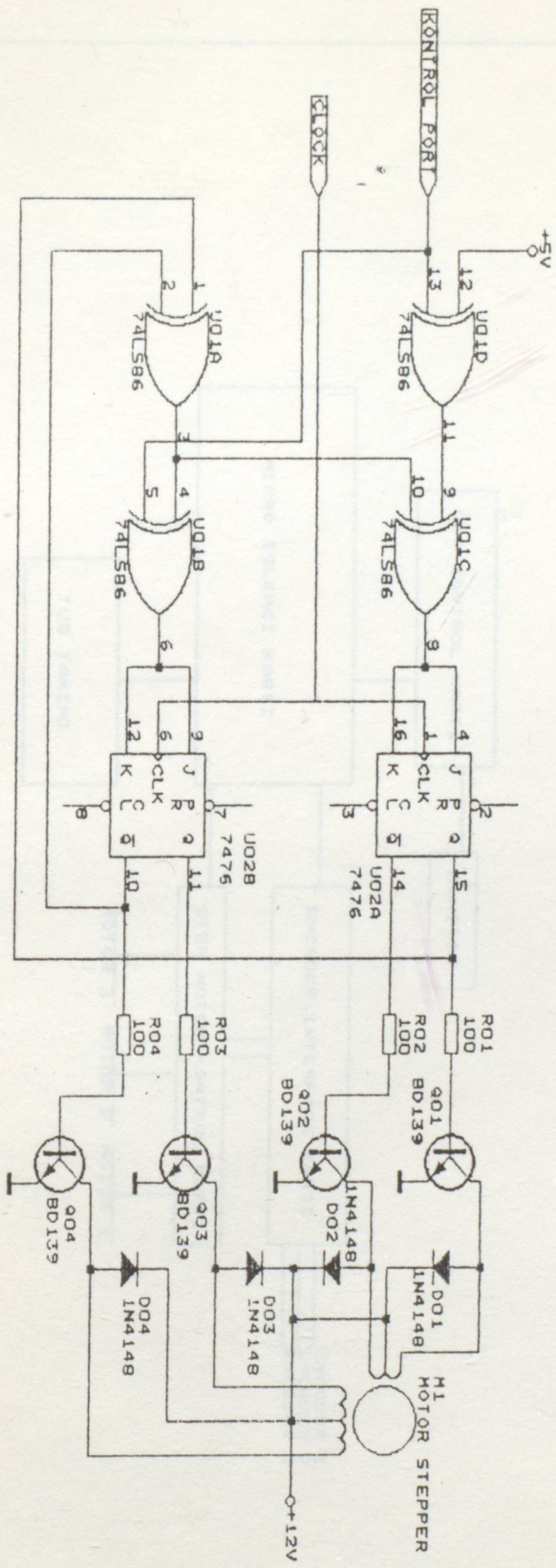
(Prentice-Hall 1987)

Ö Z G E Ç M İ Ş

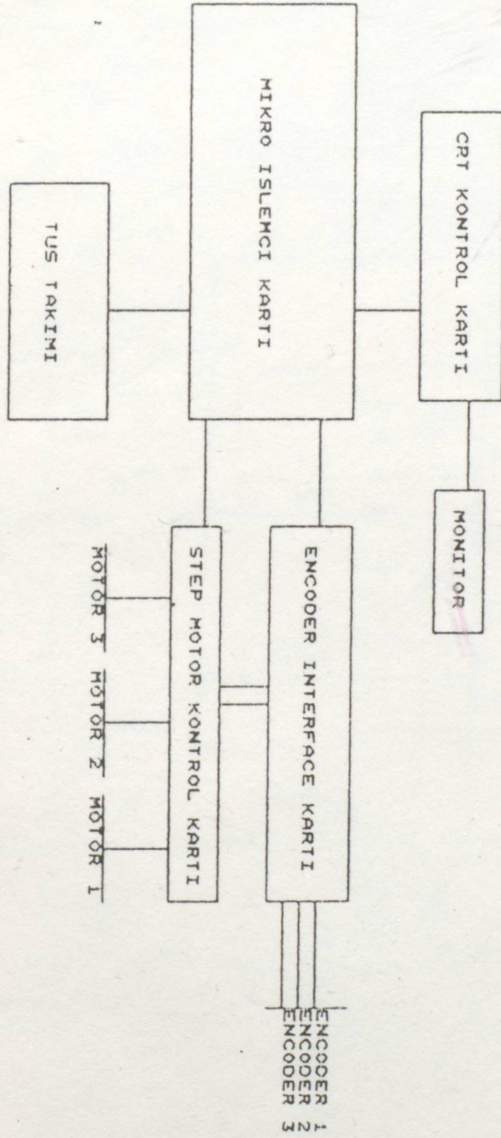
CAN ÖZKAN , 1965 yılında YOZGAT ilinde doğdu. İlk öğrenimini ANKARA 'da orta ve meslek lisesi öğrenimini İstanbulda tamamlamıştır. 1982 yılında Yıldız Üniversitesi Elektrik Bölümüne girmiştir. 1987 yılında bu bölümden mezun olmuştur.

Universite yıllarında çalışma hayatına atılmış, özel bir şirkette ARGE sorumlusu olarak çalışmıştır. 1989 yılında elektronik cihaz üretimine başlamıştır ve halen bu işe devam etmektedir.





HPC 46003 İLE XYZ KOORDİNAT SİSTEM KONTROLU
 Title
 STEP MOTOR INTERFACE DEVRESİ



HPG 46003 İLE XYZ KOORDİNAT SİSTEM KONTROLU
İtite
KONTROL SİSTEMİ BLOK SEMASI
Size Document Number
A 900113
Date: January 14, 1990 Sheet
REV

