BİLGİSAYAR MİMARİSİ DERSİ

ARAŞTIRMA ÖDEVİ II

Ayben GÜLNAR-191180041

EKİM 2022

İçindekiler Tablosu

1.Özet

Elektrik Interruptı gibi anormal olaylara tepki vermek için genellikle bir programın normal akışının kesilmesi gerekir. Bilgisayara giriş kaydındaki karakterleri yazdırmayı tamamladığını ve diğer karakterleri almaya hazır olduğunu belirten bir yazıcı gibi, belirli bir eylem planının tamamlandığını onaylamak için bir kesme de kullanılabilir. CPU zamanını farklı programlar arasında tahsis etmek için zaman paylaşımlı sistemlerde bir kesme de kullanılabilir. Modern CPU'ların talimat setleri genellikle donanım Interruptlarının eylemlerini taklit eden talimatlar içerir.

CPU Interrupta uğradığında, mevcut faaliyetini durdurması, Interrupt durumuna dikkat etmesi (interrupt'a hizmet etmesi) ve ardından faaliyetine durduğu yerden devam etmesi gerekir. İşlemcinin mevcut etkinliğinin süreksizliği, mevcut talimatın yürütülmesinin tamamlanmasını ve işlemci durumunun kaydedilmesini gerektirir. Interruptlar hem donanımsal hem de yazılımsal olabilir. [5]

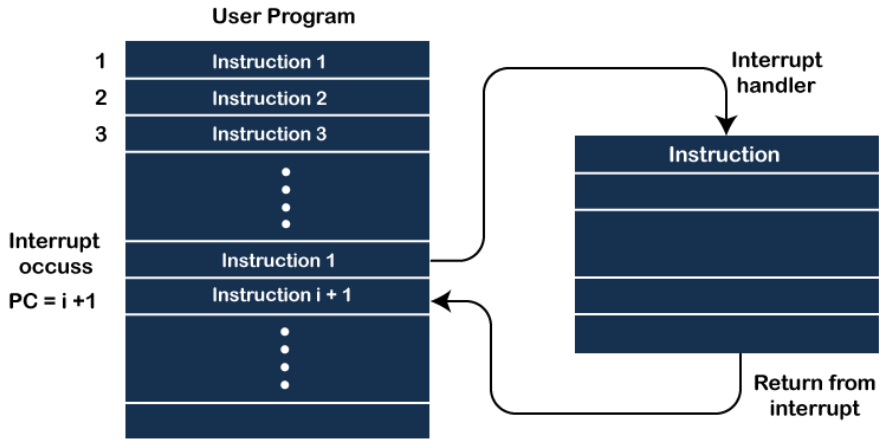
Bu çalışmamda 2. Bölümde interruptların tanımıyla başlayıp sonrasında çeşitlerine, çeşitlerinin alt başlıklarına, interruptların çalışmalarının detaylıca incelenmesine ve son olarak farklı mimarilerde ve mikroişlemcilerdeki çalışmasına yer verdim.

2.Interrupt Tanımı

Interrupt kelimesi genel olarak arada devam eden işin ilerlemesini durdurmak veya işin devamını kesmek anlamına gelir. Erken dijital hesaplamada, sistem işlemcisinin sinyalin işlenmesi için uzun süre beklemesi gerekir. Dahili olarak CPU, onlardan işlenecek herhangi bir sinyal almak için her donanım ve yazılım programını kontrol etmelidir ve sistemdeki bu sinyal kontrol yöntemi, polling yöntemi olarak bilinir.

Dezavantajı, sistemdeki sinyali aramak için sistemin birkaç saat döngüsünü boşa harcamasıdır. İşlem için herhangi bir sinyal gelirse, yoklama işlemiyle meşgul olduğu için işlemcinin işlemi yürütmesi zaman alır ve bu da sistemin performansını ve yanıt süresini düşürür. Bu nedenle, bu sorunu iyileştirmek veya üstesinden gelmek için, bir mühendis tarafından donanım ve yazılımdan gelen sinyali aramak yerine, işlemcinin talimatı tamamlama sürecini yürütmek için onlardan bir sinyal alacağı yeni bir mekanizma tanıtıldı. Sisteme Interrupt adı verilen bir işlem için donanım ve yazılım tarafından bir sinyal gönderme mekanizmasıdır.

Interruptler, şu anda çalışan bir programdan işlemci/kontrolün alınması, talimatın yürütülmesi için üçüncü taraf programa verilmesi ve ardından kontrolün önceki programa aktarılmasıdır. Bu durum bazı hata durumları, kullanıcı gereksinimleri, yazılım ve donanım gereksinimleri nedeniyle ortaya çıkmaktadır. CPU tüm kesmeleri çok dikkatli bir şekilde ele alır ve her komutun yürütülmesi için tüm kesmelere öncelik verilir.



Şekil 2.1 [1]

Yukarıdaki şemada, birçok komut içeren bir programa sahip bir bellek olduğunu varsayalım. (I1, I2, I3……In). Program bir dizi talimattır ve komutlar birer birer yürütülmektedir. Diyelim ki, I talimatının yürütülmesi sırasında bir Interrupt meydana geldi. Önce programı işlemek için harici veya dahili bir istek oluşturulur (Interrupt oluşturulan program).

Genelde işlemcinin kontrolü i+1 komutuna gelirken, interrupt nedeniyle interrupt komutuna geçer; kesme talimatının yürütülmesinden sonra, kontrol kesmeden sonraki talimata (i+1) döner. Servis programı yürütüldükten sonra kontrol orijinal programa döner.

Ancak Interrupt gerçekleşmeden önce CPU bir talimatı işliyordu. Bu nedenle, yürütmeyi kesmeye geçmeden önce, CPU (i) talimatını tamamlar ve talimatı tamamladıktan sonra adrese geri dönmesi gerektiğinden talimat durumunu kaydeder. CPU, kesme talimatına kontrol vermeden önce aşağıdaki bilgileri kaydeder.

(i) Program counter değeri (PC)

PC => i+1

(ii) Tüm CPU kayıtlarının değeri

(Before moving to i+1 instruction, all the values of instruction (I) in registers were saved.)

(iii) Durum bit koşullarının içeriği.

Tüm durum biti koşulları, program status wordde tutulur. (PSW). Genellikle kullanılan dört bayrak vardır, Durum biti, Taşıma biti, Taşma biti, Sıfır biti, bunlar belleğin yığın kısmına kaydedilir. [1]

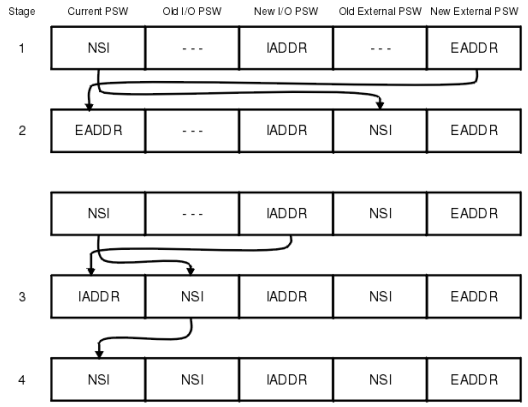
Interrupt mekanizması, bir I-stream engine ile bir channel subsystem engine arasındaki çoklu programlamayı koordine etmek için bir araçtır. Interrupt, bir I-stream motoru(engine) içinde donanım tarafından zorlanan bir kontrol aktarımıdır. Interrupt genellikle bir talimat tamamlandıktan sonra ve bir sonraki talimatın compile edilmesine başlamadan önce gerçekleşir. Ayrıca, aynı türdeki Interruptler genellikle z/TPF sistemi tarafından, en azından I-stream motorunun durumunu korumaya ve kontrol bilgilerini ve verilerini kaydetmeye yetecek kadar uzun süre engellenir. Sonuçta, veri kaybı olmadan kesilen koda dönüş yapılır. Bir I-stream motorunda engellenen Interrupt sınıfları, cihaz kontrolörlerinde ve cihazlarında ayarlanacak Interrupt üreten sinyalleri engellemez. Bu sinyaller, esas olarak, sinyalleri Interruptı kabul etmeye istekli herhangi bir I-stream motoruna sunan kanal channel subsystem içinde yığılır.

Program Status Word (PSW), talimat adresini ve talimat sıralamasını kontrol etmek ve I-stream motorunun durumunu belirlemek için kullanılan diğer bilgileri içerir. Bir PSW, Interruptleri engellemek veya izin vermek için kullanılan bitleri de içerir. Bir I-stream motorunun kontrolünde olan PSW olan mevcut PSW'ye ek olarak, her bir kesme sınıfıyla ilişkili PSW'ler vardır. Olası altı kesme sınıfı vardır:

- External
- Machine check
- I/O
- Program
- Restart
- Supervisor call (SVC) [2]

2.1 Eşzamanlı Interruptların Donanımsal Olarak İşlenmesi

Bir I-stream motorunda meydana gelen iki eşzamanlı Interrupt işleme örneğini düşünün: biri bir giriş/çıkış (G/Ç) Interruptsi ve diğeri bir harici (EXT) Interruptı olsun. Kesilen programın NSI-1 konumunda olduğunu varsayın, burada NSI bir sonraki sıralı talimat anlamına gelir. Architecture donanımı tarafından gerçekleştirilen donanım kesme işlemi, eşzamanlı kesme zorlama sinyalleri bir I-stream motoruna sunulduğunda PSW'lerde yapılan sıralı işlemeyi vurgulamak için gözden geçirilir. Gerçekleşen işleme için zaman sırası Şekil 2'de verilmiştir:



Şekil 2.1 Gerçekleşen işleme sıralaması

```
IADDR = Address of I/O Interrupt Handler
EADDR = Address of External Interrupt Handler
NSI = Next Sequential Instruction
___ = Not relevant
```

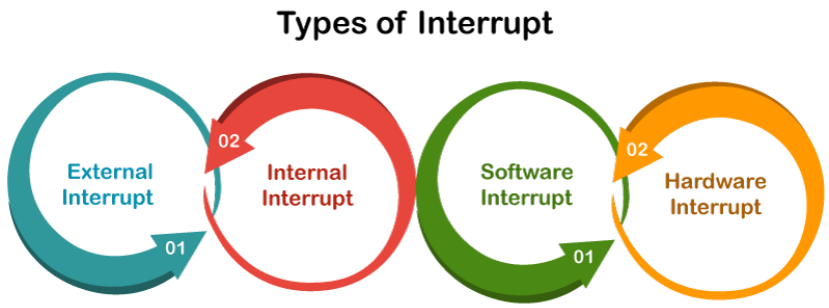
Eşzamanlı Interruptler meydana geldiğinde, PSW'lerin Şekil 1'de SIP'de gösterildiği gibi ayarlandığını varsayalım. Donanım, PSW değiş tokuşunu gerçekleştirdiğinde, geçerli PSW, denetimi, harici kesme işleyicisi olarak adlandırılan z/TPF sistem programı olan EADDR konumuna geçirecek şekilde ayarlanır. Bu noktada hem G/Ç hem de harici kesmeler devre dışı bırakılır. Eşzamanlı G/Ç Interruptsi donanım tarafından istiflenir, çünkü yüklendiğinde harici yeni PSW'de G/Ç Interruptleri devre dışı bırakılır ve G/Ç Interruptsinin şu anda karşılanmasını önler. Harici kesme işleyicisi tamamlandığında, eski harici PSW, geçerli PSW olarak yüklenir. Şu anda, NSI ile ilişkili PSW'de G/Ç Interruptleri etkinleştirildiğinden, donanım tarafından yığınlanan bekleyen G/Ç Interruptsi dikkate alınır. Kontrol, PSW takas mekanizması tarafından IADDR konumuna geçirilir. (Şekil 1'deki 3. Aşama)

G/Ç Interrupt işleyicisi tamamlandığında, eski G/Ç PSW'si mevcut PSW'ye yüklenir ve bu, kontrolü tüm Interruptler etkinleştirilmiş olarak kesilen programdaki NSI konumuna geri döndürür. (Şekil 1'de 4. Aşama)

Açıkça, harici kesme G/Ç kesmesinden daha yüksek önceliğe sahiptir. z/TPF sistemi kesme işleme kodu, işlenmemiş Interruptleri yığınlamak için donanıma bağlıdır; z/TPF sistemi ise verilerin kaybolmamasını sağlamalıdır. Bu, maske uçlarını dikkatlice ayarlayarak yapılır. Donanımda devre dışı bırakılmış (maskelenmiş) bir Interrupt durumu korunur; ve Interrupt olayının yazılım işlemi tamamlandığında, herhangi bir ek Interruptnin gerçekleşmesine izin

vermek için yeniden etkinleştirme gerçekleşir. Bir LOAD PSW veya LOAD PSW EXTENDED komutu, kesilen programın sonraki sıralı talimatını (NSI) yükleyebilir ve aynı anda Interrupt etkinleştirme gerçekleştirebilir. [2]

3.Interrupt Çeşitleri



Şekil 2.1 Sınıflandırma

Interruptları öncelikli olarak yazılımsal ve donanımsal olarak ikiye ayırabiliriz.

1. Hardware interrupt
2. Software interrupt

3.1 Hardware Interrupt

Bilgisayar sisteminin donanımı tarafından oluşturulan Interruptlere donanım Interruptleri denir. Örneğin: Çalışan bir programın devamında klavyeden herhangi bir tuşa basarsak işlemciye kesme adı verilen ve ilk olarak sistem tarafından yürütülen bir sinyal üretilir. Donanım Interruptları harici ve dahili olmak üzere iki kategoriye ayrılır. Her ikisi de CPU donanımında oluşan sinyalden başlatılır. Harici kesmeler, donanım tarafından dahili olarak oluşturulur. Bunlar eş zamansız (hiçbir şeye bağlı değildir) ve öngörülebilir Interruptlerdır. Harici kesmelere örnek olarak Giriş/çıkış cihazlarının veri aktarımı istediklerinde zamanlamaları örnek verilebilir. Dahili kesmeler, bir 'talimatta hata' meydana geldiğinde üretilir. Talimatın hatalı kullanımından kaynaklanır. Bunlar öngörülemeyen Interruptlerdir. Dahili Interruptlerın nedenlerinden birisi sıfıra bölündüğünde oluşan hata örnek verilebilir. [3]

Maskable: İşlemcilerin, donanım kesintilerinin etkinleştirilmesine ve devre dışı bırakılmasına izin veren maske kaydını kesmesi gerekir. Her sinyalin maske kaydına yerleştirilmiş bir biti vardır. Bu bit ayarlanırsa, bir bit ayarlanmadığında bir kesme etkinleştirilir ve devre dışı bırakılır veya bunun tersi de geçerlidir. Bu maskeler aracılığıyla işlemcileri kesintiye uğratan sinyallere maskeli kesintiler denir.

Non Maskable(NMI): MI'ler, bir watchdog zamanlayıcısından oluşturulan bir zaman aşımı sinyali gibi, hemen ve her durumda işlenmesi gereken en yüksek öncelikli faaliyetlerdir. [1]

3.2 Software Interrupt

Bir alt program çağrısı yerine kullanıcı tarafından oluşturulan bir kesme gibi davranan özel bir talimat türüdür. Yazılım Interruptı olarak bilinen, kullanıcı oluşturmak istediğinde kullanıcı tarafından oluşturulan bir hata üreten programdır. Programın istenilen herhangi bir noktasında bir talimat veya bir kesme prosedürü yürütülerek başlatılır. Örneğin, programın düzgün çalışıp çalışmadığını kontrol etmek için bazı hata talimatları oluşturulur. Süpervizör çağrı talimatı, kullanıcı modundan süpervizör moduna geçmek için bir yazılım Interruptı oluşturur. Yazılım Interruptlerinin meydana geldiği iki durum vardır. Normal interrupt: Talimat tarafından oluşturulan bir kesme veya kullanıcı kasıtlı olarak bir hata programı yaptığında, normal yazılım Interruptlerdır. İstisnai Interrupt: Bir programda bir sayı gibi oluşturulan planlanmamış talimatın istisnai bir durumu, sıfıra bölünebilen bir talimatın yürütülmesi sırasında oluşturulur, tanımsız bir değer verir ve bir kesme oluşturur. [3]

Level-triggered Interrupt: Bu tipte, bunun hizmet seviyesi belirtilmişse, giriş modülü bir kesme başlatır. Firmware interrupt işleyicisi tarafından işlendiğinde bir kesme kaynağı belirtilmeye devam ederse, bu modül işleyiciyi yeniden çağırmak için yeniden oluşturur ve tetikler. Seviye tetiklemeli girdiler, daha uzun bir süre için iddialı kalırsa iyi değildir.

Edge-triggered Interrupt: Kenar tetiklemeli bir kesme giriş modülü, bir yükselen veya düşen bir kenar iddia eden bir kenar tanımlar tanımlamaz bir kesmeyi başlatır. Kaynak seviyesi değiştiğinde kenar fark edilir. Bu tür tetikleme, kaynağın etkinliğinden bağımsız olarak acil eylem gerektirir.

Hybrid: Hibrit bir sistem uygulaması türü, hem kenardan tetiklenen hem de seviye tetiklenen sinyallemenin bir kombinasyonuna sahiptir. Donanım bir kenar arayacak ve ayrıca bir sinyalin belirli bir süre aktif olup olmadığını da doğrulayacaktır. Maskelenemeyen kesinti (NMI) girişi için yaygın olarak karma bir tür kullanılır; bu, yanlış kesintilerin sistemi etkilememesini sağlar. [4]

3.Interruptlara Detaylı Bakış

İşlemcinin hizmet vermeye devam etmeden önce bir Interruptnin meydana geldiğini fark ettiğini varsaydık. Bilgisayarlara, işlemciye özel kesme hatları şeklinde kesme donanımı yeteneği sağlanır. Bu satırlar, işlemciye kesme sinyalleri göndermek için kullanılır. G/Ç

durumunda, birden fazla G/Ç cihazı vardır. İşlemciye, eşzamanlı kesme isteklerini işlemesini sağlayan bir mekanizma sağlanmalıdır. Bir kesme talebinin geldiğini algılayan işlemci, zincirleme bir hibe hattı (GL) aracılığıyla, işlemciyle iletişimi başlatmak için iznini talep eden cihaza gönderir. GL, işlemciye en yakın olan ilk cihazdan başlayarak bir sonraki cihaza giden tüm cihazlardan geçer. Son cihaza ulaşır. Cihaz #1 bir istek koyduysa, hibe sinyalini tutacak ve işlemci ile iletişimi başlatacaktır. Öte yandan, Cihaz #1'in kesme talebi yoksa, hibe sinyalini aynı prosedürü tekrarlayacak olan cihaz #2'ye iletir ve bu böyle devam eder. Birden fazla talep olması durumunda, DCBA düzenlemesi, işlemciye fiziksel olarak daha yakın olan aygıta en yüksek önceliği verir. İşlemciden en uzaktaki cihaz en düşük önceliğe sahiptir. ISBA'ya göre (bkz. Şekil 8.6b), her I/O cihazının kendi kesme talebi vardır.

Yazılımsal olarak ise bir Interrupt meydana geldiğinde, işletim sistemi kontrolü ele alır. İşletim sistemi, Interruptye uğrayan işlemin durumunu kaydeder, Interruptyi analiz eder ve kesmeyi işlemek için denetimi uygun rutine geçirir. G/Ç Interruptleri de dahil olmak üzere çeşitli Interrupt türleri vardır. Bir G/Ç Interruptsi, işletim sistemine bir G/Ç cihazının çalışmasını tamamladığını veya askıya aldığını ve CPU'dan bazı servislere ihtiyacı olduğunu bildirir. Bir kesmeyi işlemek için, mevcut işlemin bağlamı kaydedilmeli ve kesme işleme rutini çağrılmalıdır. Bu işleme bağlam değiştirme denir. Bir süreç bağlamının iki bölümü vardır: işlemci bağlamı ve bellek bağlamı. İşlemci bağlamı, program sayacı (PC), program durumu sözcükleri (PSW'ler) ve diğer kayıtlar dahil olmak üzere CPU kayıtlarının durumudur. Bu bellek bağlamı, program ve verileri içeren programın belleğinin durumudur. Kesme işleyicisi, her farklı kesme türünü işleyen bir rutindir. İşletim sistemi, programlara bağlamları için kaydetme alanı sağlamalıdır. Aynı zamanda, bellek tahsisi ve tahsisi için organize bir yol sağlamalıdırInterrupt işleme rutini Interruptyi işlemeyi bitirdiğinde, CPU ya Interruptye uğrayan sürece ya da en yüksek önceliğe gönderilir. [5]

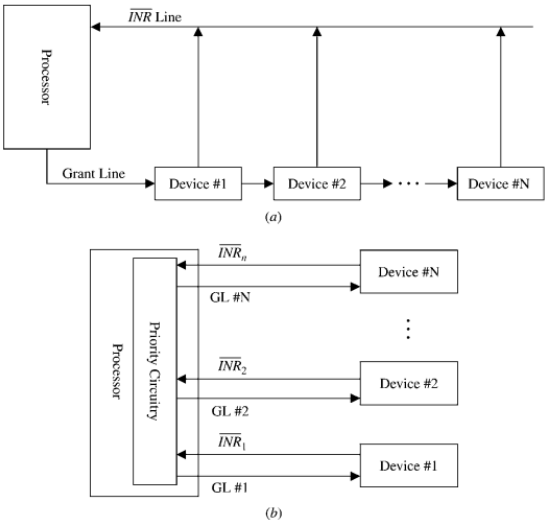


Figure 8.6 Interrupt hardware schemes. (*a*) Daisy chain interrupt arrangement (*b*) Independent interrupt arrangement

Şekil 3.2 Hardware Şeması

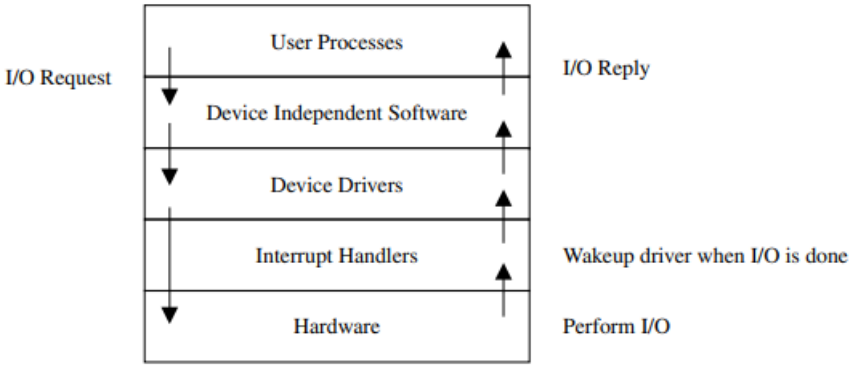


Figure 8.7 Layered I/O software

Şekil 3.3 G/Ç işlemlerinde yer alan yazılım katmanlarını gösterir. İlk olarak, program bir G/Ç çağrısı yoluyla bir G/Ç isteği gönderir. İstek, G/Ç aygıtına iletilir. Cihaz G/Ç'yi tamamladığında bir kesme gönderilir ve kesme işleyicisi çağrılır. Sonunda kontrol, G/Ç'yi başlatan sürece geri bırakılır. [5]

4.Farklı Mimarilerde Interruptlar

Örnek 1: 80386 Kesme Mimarisi 80 86 işlemcilerinde yalnızca iki donanım kesme pimi bulunur. Bunlar INTR ve NMI olarak etiketlenir. NMI maskelenemez.

Kesme, yani engellenemez ve işlemcinin buna yanıt vermesi gerekir. NMI girişi genellikle kritik sistem işlevleri için ayrılmıştır. INTR girişi, CPU ile programlanabilir Interrupt kontrolörü arasında maskelenebilir bir Interrupt talep hattıdır.

(8259A PIC). INTR'deki kesmeler, sırasıyla STI (kesme bayrağını ayarla) ve CLI (kesme bayrağını temizle) komutları kullanılarak etkinleştirilebilir ve devre dışı bırakılabilir. Kesme işleyicileri, kesme hizmeti rutinleri (ISR) olarak adlandırılır. Her bir kesme servis rutininin

adresi, kesme vektör tablosunda (IVT) birbirini takip eden dört bellek konumunda saklanır. IVT, her bir kesme türü için işaretçileri ISR'ye depolar. Bir Interrupt meydana geldiğinde, işlemciye 8 bitlik bir tür numarası verilir, bu tablodaki uygun girişi tanımlar. Bir cihaz tarafından bir kesme oluşturulduğunda, PIC'ye gider. Çoklu Interruptler aynı anda oluşturulabilir. Ancak, hepsi PIC tarafından arabelleğe alınır. PIC, bu Interruptlerden hangisinin CPU'ya iletileceğine karar verir. Bekleyen bir Interruptnin işlenmeyi beklediğini CPU'ya bildirmek için, PIC CPU'ya bir kesme isteği (INTR) gönderir ve ardından uygun zamanda bir kesme bildirimi (INTA) ile yanıt verir. Şu anda, PIC, veri yoluna aygıtla ilişkili 8 bitlik bir kesme tipi numarası koyacaktır, böylece CPU, hangi kesme işleyicisinin çağrılacağını belirleyebilir. Birkaç kesmenin beklemede olması durumunda, PIC yalnızca mevcut ISR'den bir kesme sonu komutu aldıktan sonra CPU'ya bir sonraki kesme isteğini gönderir. Şekil 8.8, hangi ISR'nin çağrılacağını belirlemek için kullanılan basit protokolü göstermektedir. Tek bir PIC (PC ve XT) kullanılan bilgisayar tasarımlarında sekiz farklı kesme isteğine izin verilir (IRQ0–IRQ7). [5]

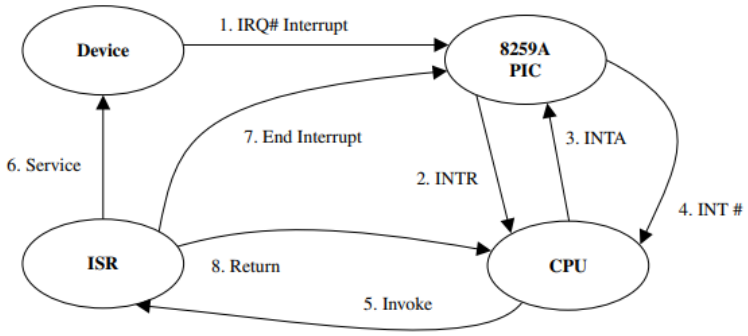


Figure 8.8 Interrupt handling in 80×86

Şekil 3.1 Interrupt Handling

Örnek 2: ARM Interrupt mimarisi ARM, Advanced RISC Machines anlamına gelir. ARM, taşınabilir aygıtlar için kullanılan 16/32 bitlik bir mimaridir, çünkü düşük güç tüketimi ve makul performansı. ARM çekirdeğine yapılan kesme istekleri, kesme denetleyicisi tarafından toplanır ve kontrol edilir.

ATIC. Interrupt denetleyicisi, çekirdeğe bir arabirim sağlar ve 64 adede kadar kesme isteği toplayabilir. Interruptler için olağan olay sırası aşağıdaki gibidir. Interruptler kaynakta (çevre birimi gibi) etkinleştirilir, ardından kesme denetleyicisinde etkinleştirilir ve son olarak, çekirdeğe etkinleştirildi. Kaynakta bir Interrupt meydana geldiğinde, sinyali kesme denetleyicisine ve ardından ARM çekirdeğine yönlendirilir. Interrupt denetleyicisinde kesme

çekirdeğe etkinleştirilebilir veya devre dışı bırakılabilir ve bir öncelik düzeyi atanabilir. Kesme isteği çekirdeğe ulaştığında, çekirdeği normal konumundan durduracaktır. [5]

4.1 CISC & RISC & EPIC Mimarilerinde Interruptlar

CISC, Karmaşık Komut Seti Bilgisayarının kısaltmasıdır. CISC mimarisi, bir programın sahip olduğu Talimat sayısını azaltmaya çalışır, böylece yukarıdaki denklemin Program Başına Talimat bölümünü optimize eder. Bu, birçok basit talimatı tek bir karmaşık talimatta birleştirerek yapılır.

İndirgenmiş Komut Seti Bilgisayarı veya RISC mimarileri daha fazla talimata sahiptir, ancak bir talimatın gerçekleştirmesi gereken döngü sayısını azaltır. Genel olarak, bir RISC makinesindeki tek bir talimat, yalnızca bir CPU döngüsü alacaktır. Bir RISC mimarisinde çarpma, tek bir MUL benzeri komutla yapılamaz. Bunun yerine, önce LOAD komutunu kullanarak verileri bellekten yüklememiz, ardından sayıları çarpmamız ve sonucu belleğe kaydetmemiz gerekir. Bu çok iş gibi görünebilir, ancak gerçekte, bu talimatların her biri yalnızca bir saat döngüsü aldığından, tüm çarpma işlemi daha az saat döngüsünde tamamlanır.

Bununla birlikte, zaman avantajı dezavantajları olmadan değildir. RISC daha basit komut setlerine sahip olduğundan, karmaşık Yüksek Düzeyli Talimatların derleyici tarafından birçok talimata bölünmesi gerekir. Talimatlar basit olsa da ve kodu çözmek için karmaşık mimarilere ihtiyaç duymazken, karmaşık yüksek seviyeli programları birçok basit talimata bölmek derleyicinin işidir. Bu, donanım tarafından yapılması gereken işi azaltırken yazılım ve yazılım tasarımcıları üzerinde çok fazla stres yaratır. Gerçekten 'daha iyi' bir mimari yoktur, her birinin onu farklı uygulamalarda faydalı kılan kendi avantajları ve dezavantajları vardır. CISC çoğunlukla otomasyon cihazlarında kullanılırken, RISC video ve görüntü işleme uygulamalarında kullanılır.[6]

İki mimari Interruptlar açısından kıyaslandığında CISC mimarisinin diğer iki mimariye göre daha düşük performanslı olduğu görülmüştür. Geliştirildiği yıllarda, önemli olan işlemlerin paralel olarak gerçekleşmesi olmadığından ve aynı anda birkaç işlemin çalıştırılması önem taşımadığından interruptların performansı da daha düşük olmuştur. Ancak CISC avantajları da bulunmaktadır. Yoğun ağ trafiğinin yönetildiği işlemcilerde kesme işlemlerinin sık olması sisteme yük teşkil ederek sistemin oyalanmasına sebep olmaktadır. Bu yüzden bu iş için CISC mimarisinin olduğu işlemciler kullanılmaktadır. İş istasyonlarında gibi hesaplama işlemlerinin çok olduğu alanlarda RISC mimarili işlemcilerin kullanılması performansı arttırmaktadır. İşlemci üzerinde her iş parçacığı (thread) işlenip bittikten sonra interruptın varlığı kontrol

edilmektedir. CISC mimarisi ile kıyaslandığında daha yüksek performans sağlamaktadır. Explicitly Parallel Instruction Computing yani Belirtilmiş Paralel Komutlarla Hesaplama olarak isimlendirilen EPIC mimarisi, diğer iki mimariden sonra geliştirilmiştir. CISC ve RISC mimarisine göre daha başarılı bir yapıya sahiptir. Bilgisayarda aynı anda gerçekleştirilen işlem sayısı arttıkça interrupt sayısı da artmaktadır. Bu yüzden üç mimarinin de yapısına bakıldığında interrupt performansının en yüksek olduğu mimarinin EPIC mimarisi olduğu söylenebilir. Paralelliği en çok destekleyen ve kullanan mimaridir. Bu paralel işlemlerin gerçekleştirilebilmesi için de interruptların sıkça kullanıldığı görülmüştür. [7]

5.Sonuç

Yaptığım araştırmalara göre en basit şekilde Interruptları; çalışan bir programın işlemci kontrolünün alınıp, talimatın yürütülmesi için üçüncü taraf programa verilmesi ve ardından kontrolün önceki programa aktarılması olarak tanımladık. Yazılım ve Donanım Interruptları olarak temelde iki çeşitinin olduğunu söyledik. Ancak bu iki çeşitte kendi içinde maskable, nonmaskable ve level-triggered, edge-triggered gibi daha da alt başlıklara parçalandığını detaylıca inceledik. 80386 Interrupt Mimarisi 80 86 işlemcilerinde ve ARM Interrupt mimarisinde detaylıca ele alırken aynı zamanda RISC, CISC VE EPIC mimarilerinde interruptların çalışmalarının kıyaslamasını son bölümde inceledik. EPIC mimarisinin CISC ve RISC mimarisine göre daha başarılı bir yapıya sahip olduğu sonucuna ulaştım. Bilgisayarda aynı anda gerçekleştirilen işlem sayısı arttıkça interrupt sayısı da arttığından üç mimarinin de yapısına bakıldığında interrupt performansının en yüksek olduğu mimarinin EPIC mimarisi olduğu sonucuna ulaştım. Sonuç olarak Interruptların kullanım alanlarını detaylıca araştırdım.

6.Kaynakça

[1] Internet: https://www.tutorialandexample.com/interrupt-and-its-types

[2] Internet: https://docs.oracle.com/javase/tutorial/essential/concurrency/interrupt.html

[3] Internet:https://www.ibm.com/docs/en/ztpf/1.1.0.15?topic=processing-summary-interrupts

[4] Internet: https://www.elprocus.com/basics-of-interrupt-types-and-its-applications/

[5] Abd-El-Barr, El-Rewini, Fundamentals of Computer Organizations and Architecture

[6]Internet:https://medium.com/@csoham358/a-beginners-guide-to-risc-and-cisc-architectures-fc9af424db3b

[7]Internet:https://bilgisayarmuhendisleri.blogspot.com/2020/05/farkli-islemci-mimarilerinde-interrupt.html