AMD OPTERON 64-BİT İŞLEMCİ MİMARİSİ

BİLGİSAYAR MİMARİSİNDE YENİ YAKLAŞIMLAR

PROF. DR. BÜLENT ÖRENCİK

ÖZER ÖZAYDIN

704051012

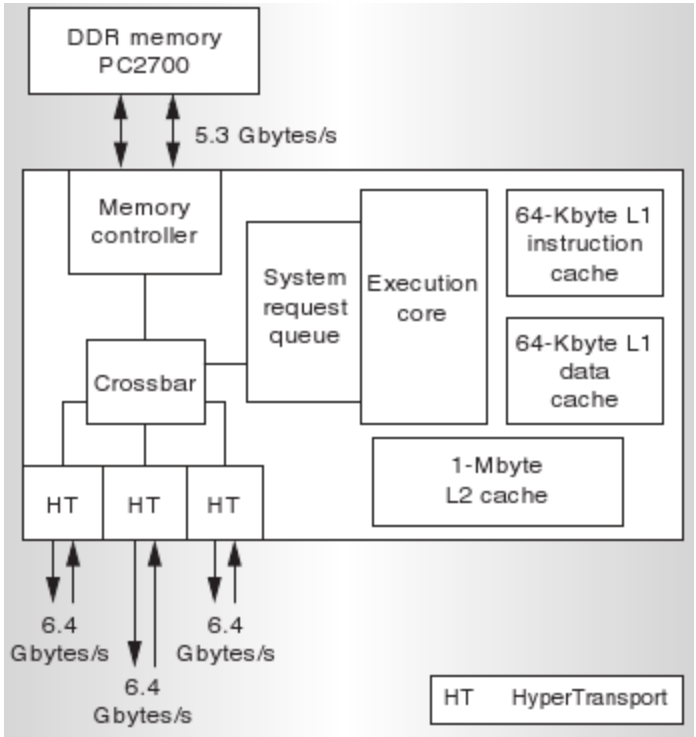
01.01.2008

İçindekiler

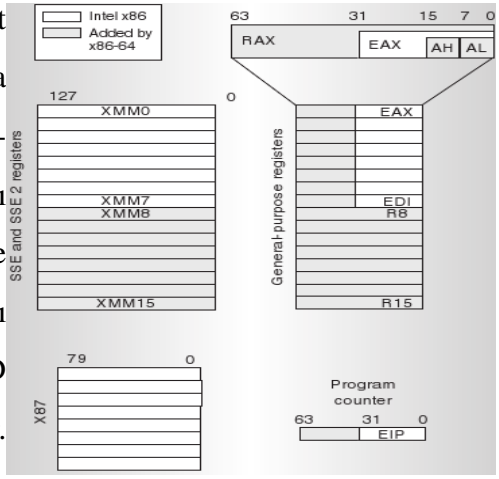
AMD Opteron
64-bit İşlemci Mimarisi

Giriş

Advanced Micro Devices (AMD)'nin üretmiş olduğu Opteron işlemcisi, 64bitlik, x86 tabanlı, çip üzerinde bellek yöneticisine sahip (on-chip memory controller) ve çok işlemcili uygulamalar için kullanılan üç HyperTransport bağlantılı bir işlemcidir. AMD işlemcisi standart x86 mimarisinin gelişmiş bir versiyonu olan geri-uyumluluğa sahip x86-64 mimarisi üzerine kurulmuştur. Bu şekilde AMD Opteron hem 32bitlik x86 komutlarını hem de 64bitlik yeni komutları destekler. Opteron çekirdeği sırasız işlem yapan (out-of-order), çip üzerinde bulunan (on-chip) L1 ve L2 cep bellekleriyle desteklenen üç-yollu bir süperskaler işlemcidir. Opteron'un önemli özelliklerinden birisi de düşük bellek gecikmeli, yüksek performanslı tamsayı, kayan nokta ve çoğulortam komutları işlenebilmesine olanak veren, çipin üzerinde bulunan DDR (double-data-rate) bellek yöneticisidir. Çok işlemcili uygulamalar için ayrıca destek çiplerine ihtiyaç duymadan HyperTransport bağlantı yolları kullanılabilir. Ayrıca Opteron komut ön-çözme(predecoding), dallanma öngörüsü (branch prediction), sanal adres dönüşümü gibi alanlarda getirdiği yeniliklerde çevrim başına düşen komut sayısı (instructions per cycle) performansını kendisinden önceki Athlon-64 işlemcisine göre geliştirmiştir.

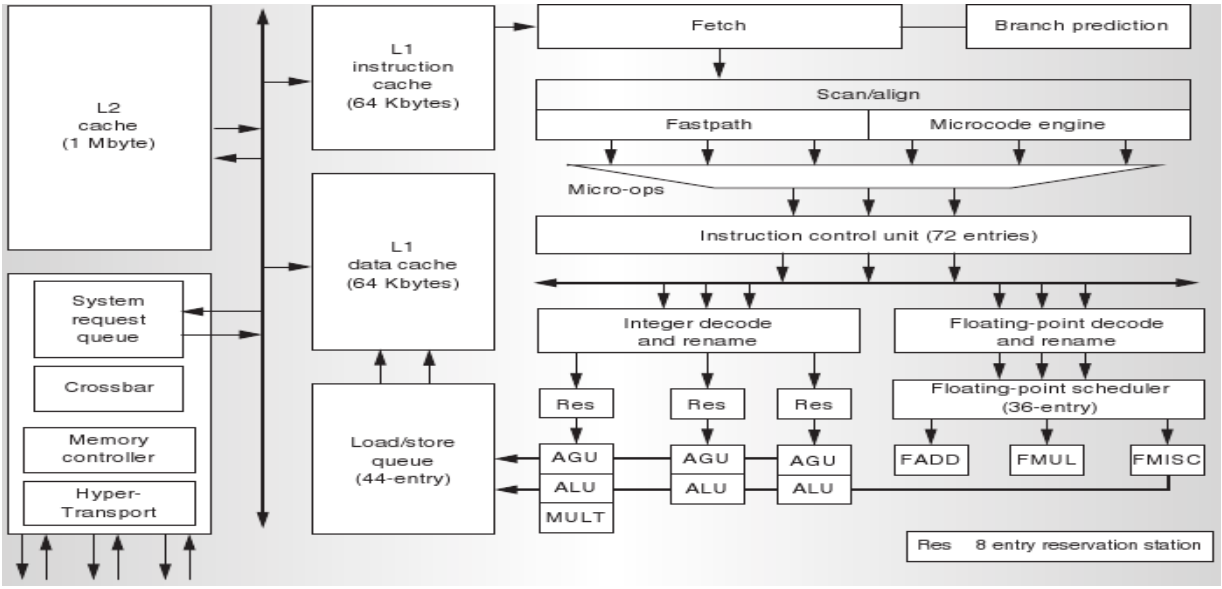


AMD'nin x86-64 mimarisi 64-bit sanal ve 52-bit fiziksel adresi destekler ancak tüm uygulamalar bu koşula uymak zorunda değildir. Opteron işlemcisi 48-bit sanal ve 40-bit fiziksel adresi destekler. x86-64 mimarisi tüm x86 tamsayı aritmetik işlem ve mantıksal tamsayı komutlarını 64bit'e genişletir. Çekirdekteki tamsayı genel-kullanım saklayıcıları (general-puspose registers-GPR) ve SSE (streaming SIMD extension) saklayıcıları sayıları 8'den 16'ya çıkarılmıştır. GPR'ların boyutları da 32bitten 64bit'e genişletilmiştir.



Çekirdek Mimarisi

Opteron sırasız işlem yapan (out-of-order), üç-yollu bir süperskaler işlemcidir. Aşağıdaki şekilde Opteron işlemcisinin çekirdek blok diyagramı gösterilmiştir. Opteron 72 mikro-iş (micro-ops)i yeniden sıralayabilir, her çevrimde üç x86-64 komutunu komut cep belleğinden çekme ve çözme işlemini gerçekleştirebilir. Kendisinden önceki seri AMD Athlon'a benzer bir şekilde değişik boyutlarda bulunabilen x86-64 komutlarını sabit boyutlu mikro-işlere dönüştürür ve bu mikro-işleri sıralayıcılara gönderir. Tamsayı ve kayan-noktalı iş-sıralayıcı birimleri ayrıdır. Aynı zamanda yükle/sakla mikro-işleri için yükleme/saklama birimine mikro-işleri gönderir. İş-sıralayıcılar ve yükleme/saklama birimleri her çevrimde şu yürütme birimlerine 11 mikro-iş dağıtabilirler: üç tam sayı yürütme birimi, üç adres hesaplama birimi, üç kayan-noktalı ve çoğulortam yürütme birimi ve iki yükleme/saklama birimi.



Opteron'un Performans Arttırıcı Özellikleri

Gelişmiş Dallanma Öngörüsü

Dallanma öngörüsü iş-hatlı mimarilerin en önemli özelliklerinden birisidir. Öngörü yapılarak, bir dallanma durumu oluştuğunda tüm iş hattının boşaltılması yerine doğru öngörü yapılmışsa gecikmesiz veya az bir gecikmeyle yola devam edilebilir. Opteron'un tamsayı işlemleri için 12, kayan-noktalı işlemler için 17 kademeli olan iş hattı düşünüldüğünde dallanmanın önemi büyüktür. Koşullu dallanma olması durumunda genellikle dallanmanın sonucu iş-hattının sonlarına doğru belli olur ki bu bilginin iş-hattının başlarında bilinmesi istenir; çünkü dallanmanın sonucuna göre bir sonraki komut satırı komut cep belleğinden dallanma adresine göre çekilmelidir. Bir satır 16byte'lık komutun yüklenmesi iki çevrim gerektirir. Eğer daha fazla zaman kaybedilmek istenmiyorsa komut satırının yüklendiği çevrimin sonunda bir sonraki komut işaretçisinin belirli olması gerekir. Ancak bir taraftan da çekilen komut satırını incelemek çok zaman kaybettirecektir. Bu yüzden Opteron'un dallanma öngörüsü donanımı çekilen komut satırının içeriği ile ilgilenmez. Bunun yerine çeşitli veri yapıları kullanarak geçmişe ait bilgilerin ışığında hızlı bir şekilde yeni adresi hesaplar. Bu iş için 2048 giriye sahip bir Dallanma Hedef Tamponu (Branch Target Buffer-BTB) ve 12 girili Dönüş Yığını (Return Stack) kullanır. Ayrıca dallanmaya karar verme işi için iki dallanma geçmiş bilgisini kullanır, birisi yerel geçmiş bilgisini kullanan dallanma seçicileri (branch selectors), diğeri ise daha genel bir geçmiş bilgisini saklaran genel geçmiş sayaçlarıdır (global history counters).

Dallanma Seçicileri

Dallanma seçicileri yerel geçmiş bilgisini içerir. Buradaki yerelin anlamı dallanma öngörüsünün dallanmanın kendi geçmişinden yararlanılarak yapılacak olmasıdır. Neredeyse her zaman aynı yöne doğru olan koşullu dallanmalar kolaylıkla dallanma seçicileri tarafından tahmin edilebilir, bunun yanında koşulsuz dallanmalar da dallanma seçicileri tarafından kotarılabilir. Bir çok dallanma komutunda dallanmanın olacağı büyük bir olasılıkla belirlidir, ancak eğer dallanmanın olasılığı düşükse dallanma seçicileri dallanma öngörüsünü genel geçmiş sayaçlarına bırakabilir. Dallanma seçicileri dallanmanın olacağı yönünde karar verir ancak genel bayrağını birleyerek son kararı genel geçmiş sayaçlarına bırakır.

16 baytlık komut satırı															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
BS0		BS1		BS2		BS3		BS4		BS5		BS6		BS7	

Dallanma Seçicileri	K8 Athlon 64
3	3. dallanmayı seç (ya da dön)
2	2. dallanmayı seç (ya da dön)
1	1. dallanmayı seç (ya da dön)
0	sonraki komut satırınından devam et

Her 16baytlık kod satırı sekiz adet 2 bitlik dallanma seçicisi ile ilişkilendirilir. Satırın içerisindeki dallanma seçici komut çekme adresinin [3:1] bitleri ile belirlenir. Dallanma seçici 16baytlık satırın neresinde ise orasının durumuna göre bir sonraki çevrimde hangi satırın yüklenmesi gerektiğini söyler. Bir komut satırında değişik sayıda atlama(jump), fonksiyon çağırma (call)veya dönme komutları(returns) bulunabilir. Satıra bu komutların herhangi birisinin bulunduğu yerden girilmiş olunabilir. Dallanma seçicileri kod satırına giriş yerine bağlı olarak ne yapılması gerektiğini söyler. Her 16baytlık komut satırı için olan dallanma seçicileri komut cep belleğinden komut satırı çekilir çekilmez 1MB boyutundaki L2 cep belleğine komut satırı ile beraber yazılır.

Genel Geçmiş Sayaçları (Global History Bimodal Counters -GHBC)

Opteron işlemcisinde yerel geçmiş bilgisi yanında genel geçmiş bilgilerinin bulunduğu 16384 dallanma geçmişinin saklanabildiği bir genel geçmiş sayaçları tablosu bulunmaktadır. Buradaki iki bitlik sayaçlar bir dallanmanın gerçekleşip gerçekleşmeyeceğinin olasılığı anlamına gelir. En büyük olasılık üçtür ve sıfıra doğru azalır.

2 bitlik Dallanma Geçmişi Sayaçlarının Tanımı	
Sayaç Değeri	**Dallanma Tahmini**
3	Yüksek İhtimalle Dallan
2	Düşük İhtimalle Dallan
1	Düşük İhtimalle Dallanma
0	Yüksek İhtimalle Dallanma

Tabloya erişim 8 bitlik geçmiş dallanma örüntüsü + dallanma adresinin 4 biti = 12 bitlik bir indis ile gerçekleşir. Unutulmamalıdır ki her 16baytlık kod satırı için dört dallanma öngörüsü yapılabilir, ancak işlemcide bir kısıtlama olarak her komut satırı için en fazla üç dallanma komutu bulunabileceğinden bunlardan sonuncusu kullanılmaz. Genel geçmiş tablosu kullanılırken sadece genel bayrağı birlenmiş olan dallanmalar hesaba katılır. Böylece sabit davranışlı olan dallanmaların genel geçmiş tablosunu kirletmesi engellenmiş olur. Buradaki genel bayrağı dallanma seçiciler tarafından eğer dallanama değişken bir davranışa sahipse birlenir.

Dallanma Hedef Tamponu (Branch Target Buffer- BTB)

Dallanmayla ilgili diğer önemli birim olan BTB ise yerel veya genel dallanma geçmiş tablolarından gelen öngörüye göre dallanma kararı verilmişse dallanılacak olan adresin çekilebildiği yerdir. Bu adres dallanma komutunun içerisinde olabileceği gibi bir hesaplama sonucunda da bulunabilir ki hesaplama yapılarak bulunan dallanma adresleriyle daha sık karşılaşılır. Böyle bir durumda dallanma ve dallanmaya karşı düşen adres daha sonra kullanılmak üzere BTB'de saklanır. Aynı dallanma tekrar gerçekleşirse tekrar adres hesabına gerek kalmadan dallanma adresi BTB'den alınır. BTB'de 2048 komut çekilme adresi girisi vardır. BTB, Dallanma Hedefi Adres Hesaplayıcısı (Branch Target Address Calculator-BTAC) denilen bir birim tarafından yedeklenir, bir bakıma BTB'den gelen adresler kontrol edilir. BTAC eğer BTB dallanma durumunda dallanmanın yanlış bir adresini tutuyorsa bunu kontrol ederek yanlış dallanma cezası süresini kısaltır. Bazı olası durumlarda BTB'nin yanlış dallanma adres türetebilir. BTAC dallanma hedef adresini hesaplar ve BTB ile karşılaştırır, eğer adresler birbirlerini tutmuyorsa dallanma öngörücü hemen yanlış dallanma uyarısı verir ve yanlış dallanma ceza çevrimleri azalır. BTAC tüm dallanmalarla çalışamaz, ancak 8 veya 32bitlik kayıklık adreslerine sahip olan çoğu atlama (jump) ve fonksiyon çağırma (call) komutlarıyla çalışır.

Kestirme Dönüşüm Tamponu Boşaltma Filtresi(Translation Lookaside Buffer Flush Filter)

Günümüz bilgisayar sistemlerinde belleğin etkin ve esnek kullanımı için kullanılan bellek yönetim modellerinde belleğe erişebilmek için sanal-fiziksel adres dönüşümleri yapmak gerekir. Bu dönüşümlerle ilgili en büyük problemlerden birisi gecikmedir. Dönüşüm bilgisini bellekten almadaki kaybedilen zamanın yanında bu dönüşümü gerçekleştirmek için bir çok işlemci çevrimi harcanır. Bu zaman kaybı sık bellek erişimine ihtiyaç duyan programlarda kabul edilemeyecek düzeylerde olabilir. Bu dönüşümü hızlandırmak için sık kullanılan dönüşüm bilgilerinin tutulduğu yonga üzerinde bulunan hızlı erişimli kestirme dönüşüm tamponu denilen bir tampon bellek kullanılır. Belleğe erişmek isteyen bir program ilk önce kestirme dönüşüm tamponuna bakar ve eğer dönüşüm buradaysa hızlı bir şekilde sonucu elde eder. Böylelikle her seferinde dönüşüm bilgisi ana bellekten ya da sabit diskten alınarak zaman kaybedilmemiş olur.

TLB'nin bu getirisine rağmen halen iyileştirmelere ihtiyaç duyulan durumlar vardır. TLB kullanan bir çok işlemci proseslerin bağlam değiştirmelerinde tüm TLB içeriğini boşaltır. TLB'nin boşaltılma sebebi eski bağlamın TLB'deki adres dönüşümlerinin alındığı bellek bölgesinde dönüşüm bilgilerin değiştirmiş, yani TLB'deki bilgilerin artık geçersiz olabilme olasılığıdır. Bağlam değiştirildikten sonraki yeni proses için bir çok bellek erişimine, dolayısıyla adres dönüşümlerine ihtiyaç duyulur. Bu yüzden TLB'nin tekrar doldurulması gerekir, bu da oldukça yüksek gecikmelerle karşılaşılması demektir. Sık bağlam değiştirilen işletim sistemlerinde bu durum yüksek performans kayıplarına sebep olabilir. Bir başka taraftan bakıldığında bazı durumlarda TLB'nin boşaltılmasının gereksiz olduğu görünebilir. TLB'nin içerisinde bulunan adres dönüşümleri bellekte bulunan bazı veri yapılarının kullanılmasının yardımıyla hesaplanabilir. Örneğin sayfalama yönteminin uygulandığı sistemlerde adres dönüşüm bilgilerinin saklandığı veri yapılarını saklayan sayfa tabloları kullanılır ve bu veri yapıları TLB'nin içeriğini oluşturur. Bağlam değiştirme sırasında TLB'nin içeriğini oluşturan bu veri yapılarında değişiklik yapılmış olabileceği gibi yüksek bir olasılıkla eski bağlamı kullanan proses herhangi bir değişiklik yapmamıştır. Ancak bu bilinmesine rağmen tutarlılık adına TLB boşaltılır ve yeniden doldurulur. Neticede genellikle gerekmemesine rağmen sistem fazladan gecikmelere maruz kalır.

Opteron'un mimarisinde TLB'nin gereksiz boşaltılmasını engellemeye yönelik bir Kestirme Dönüşüm Tamponu Boşaltma Filtresi (Translation Lookaside Buffer Flush Filter -TLB-FF) bulunmaktadır. TLB-FF, adres dönüşümlerinin alınıp TLB'ye yazıldığı bellek bölgelerini izler ve bellekteki dönüşümlerde bir değişiklik olup olmadığını belirler. Eğer bellekteki adres dönüşümünde

bir değişiklik yapılmadıysa işlemcinin bağlamı değiştiği durumda işlemcinin TLB'yi boşaltmasını önler. Bellekteki adres dönüşümlerinde bir değişiklik olması durumunda ise TLB eskiden de olduğu gibi boşaltılır ve tekrar doldurulur.

TLB-FF, adres dönüşümlerinin alındığı bellek bloklarını 32 girili bir bölge tablosu (region table) kullanarak tutar. TLB-FF sayfa tabloları veya sayfa dizinlerinindeki adres dönüşümlerinin değişip değişmediğini bu tablo üzerinden takip edebilir. İşlemcideki ilk bağlam değiştirme bir boşaltma işlemi ile sonuçlanır ve bu bağlam değiştirmeden sonra TLB-FF'i aktif hale gelir. Bundan sonraki boşaltma işlemleri eğer bölge tablosundaki bellek bölgelerinde bulunan adres dönüşümlerinde bir değişiklik yoksa engellenir, aksi takdirde izin verilir.

TLB-FF aynı zamanda bağlam değiştirme işlemlerini de takip eder. Bu işlem belirli bir bağlamın sayfa tablosunun taban adresinine karşılık gelen etiketlerle gerçeklenir. Bu etiketler ve etiketlere karşılık düşen taban adresler bölge tablosunun içeriğini oluşturur. Bağlamların etiketleri diğer bilgilerle beraber TLB'de saklanır. Neticede bağlamların hangi sayfa tablolarına eriştikleri TLB'deki etiketler vasıtasıyla takip edilir. Böylelikle farklı bağlamların kullandığı adres dönüşümleri TLB'de aynı anda etiketlerle ayrılarak tutulabilir. İşlemci belirli bir bağlamdayken sadece o bağlamın etiketiyle uyuşan TLB girileri bağlamın hizmetinde olur, etiketi uyuşmayan giriler göz ardı edilir.

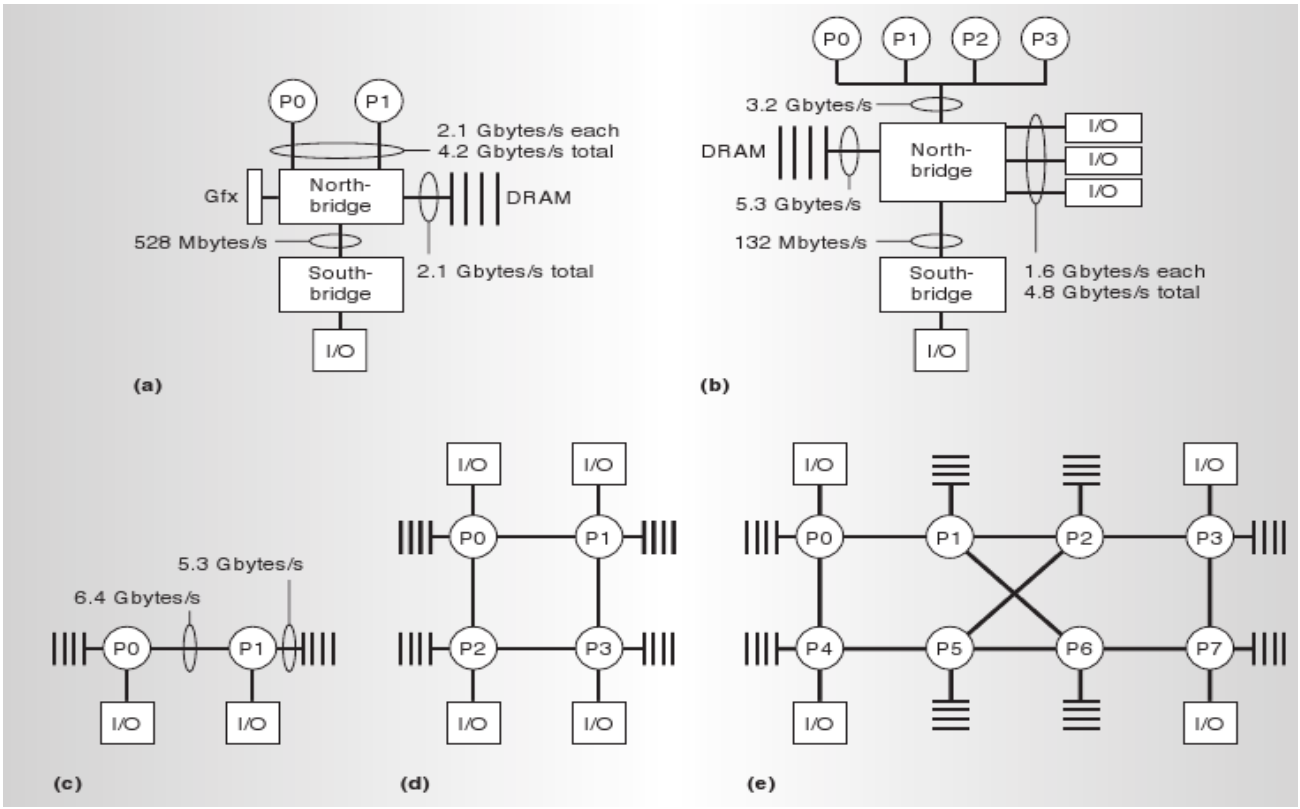
Opteron'da kullanılan bölge tablosu bir İçerik Adreslemeli Bellek (Content Addressable Memory-CAM) ve bir Rastgele Erişimli Bellek (Random Access Memory-RAM) ile gerçeklenir. CAM'de adres dönüşümlerinin TLB'ye yüklendiği bellek yapısıyla ilgili bilgi (sayfa tablosu ve sayfa dizini gibi) tutulur. RAM'de ise etiketler ve karşılığındaki sayfa tablosunun taban adresi tutulur. Bunlarla beraber taban adresi girilerinin sayısını ve CAM'deki girilerin sayısını tutan iki sayaç mevcuttur. Bu sayaçlar belirli bir değeri aştığında TLB-FF pasif hale getirilir ve bölge tablosu temizlenir. Bu aşamadan sonraki bir bağlam değiştirme işlemi TLB'nin boşaltılmasıyla sonuçlanır, bu boşaltılmadan sonra ise TLB-FF tekrar aktif halde getirilir.

Sonuç olarak TLB-FF uzun süreli gecikmelere sebep olan bir çok gereksiz TLB boşaltma işlemini engeller. Boşaltılmayan TLB'de bulunan ve doğru olduğu bilinen dönüşüm bilgisini yeni bağlam kullanabilir (önceden kendisi yüklemiş ve sıra tekrar kendisine gelmiş). Birçok durumda tekrarlanması gereken sanal-fiziksel adres dönüşümleri ve bunların TLB'ye yazılması işleminin sayısı azaltılmış olur, bu da işlemcinin performansında önemli bir artışa sebep olur.

Yerleşik Bellek Yöneticisi ve HyperTransport Veri Yolu

Opteron üzerindeki yerleşik bellek yöneticisi PC2700 tipli 333 MHz hızında çalışan DDR (double data rate) DIMM'lere (dual in-line memory modules) çift kanallı 128 bitlik geniş bir arayüz sağlar. Bellek yöneticisinin en yüksek bant genişliği 5.3 Gbayt/sn'dir. Sekiz adet DIMM'e kadar destekler. Bellek yöneticisi işlemci çekirdeğinden ayrı bir saatle tetiklense de işlemciyle aynı saat frekansında çalışır. İşlemcinin çalışma frekansında yapılacak olan hızlanmalar bellek gecikmesine de yansıyacaktır. Çekirdekten farklı bir saatle sürülen bellek yöneticisi işlemci çekirdeğinin saatinin durdurulabildiği düşük güç durumlarında (low-power mode) grafik veya diğer cihazlara belleğe erişimi devam ettirebilir.

Bellek yönetimi işleri daha önceleri Northbridge ismindeki işlemci çekirdeğinden ayrı olan birimle yapılmaktaydı, ancak Opteron'da bulunan yerleşik bellek birimi eski yöntemi kullanan Athlon işlemcisine göre %20-%25 arasında bir performans artışı sağlamıştır. Yerleşik Northbridge birimi aynı zamanda PCI (peripheral component interconnect) konfigürasyon isteklerinden, aktarım sıralamalarınından ve cep bellek tutarlılığından sorumludur. Çoklu işlemcili sistemlerde de yerleşik bellek kontrolörü sayesinde her işlemciye kendi belleği ve bu belleğe bir arayüz sağlanır. Eski yöntemin ölçekleme problemi de her işlemcinin ayrı bellek kontrolörü sayesinde aşılmıştır.



a) Çift çekirdekli AMD Athlon b) Dört çekirdekli Inter Xenon c-d-e) Çok çekirdekli Opteron

Opteron'un bir diğer özelliği de üzerindeki üç HyperTransport bağlantı birimidir. Her HT bağlantısı 16bit genişliğinde çift yönlü (her yön için 3.2Gbayt/s) tutarlı (coherent-HT) veya tutarsız (noncoherent-cHT) olarak ayarlanabilen bir veri yolu sağlar. Tutarlı olan cHT veriyolu işlemcilerin birbirleri arasındaki bağlantı için, daha basit olan tutarsız veri yolu HT ise işlemcinin G/Ç birimleriyle bağlanması için kullanılır. Noktadan-noktaya (point-to-point) HyrperTransport bağlantıları esnek, ayarlanabilen ve ölçeklenebilir çok işlemcili-G/Ç birimli uygulamalar için olanak verir.

Tamsayı Çekirdeği

Opteron işlemcisi 3 yollu bir süperskaler işlemcidir. Yani tek çevrimde 3 tane x86 komutunu çözüp yürütüp bırakabilir (retire). Bu komutlar ikiden fazla işlenen (operand) içeren CISC komutları da olabilir. Genel olarak bir x86 komutu şu şekillerde olabilir; F(saklayıcı,saklayıcı), F(saklayıcı,bellek) veya F(bellek,saklayıcı). İlk iki durum genellikle tamsayı, MMX ve SSE(2) komutları için kullanılır. Son durum ise temel olarak tamsayı komutları içindir: şöyle ki bellekten bir kaynak işleneni çekilir ve sonuç aynı yere yazılır. Kayan-noktalı ve çoklu-ortam komutları için gereken yükleme ve saklama işlerini Tamsayı İş-hattı yapısı kotarır.

Çift Dağıtmalı İşlemler (Double Dispatch Operations)

AMD'nin Opteron'dan önce üretilmiş olan işlemcisi 32 bitlik Athlon komutları Doğrudan Yollu (Direct Path) ve Vektör Yollu (Vector Path) olarak iki sınıfa ayırır. Doğru yollu olan komutlar daha az karmaşık işlemler içeren ve donanımla tek işlemde kotarılabilen komutlardır. Daha karmaşık olan vektör yollu komutların kotarılması işi doğrudan donanımla değil de bir mikro kod program ardışımlayıcısı çağrılarak bu ardışımlayıcının bir mikro kodlu programı yürütmesiyle gerçekleştirilir. Komutlar mikro kod salt okunur belleğinden (Rom) okunur ve 3 yollu iş hattına sürülerek komut kotarılır.

Opteron işlemcisinde yukarıda bahsedilen iki komut türünün yanı sıra üçüncü bir komut türü olarak Çift Dağıtmalı (Double Dispatch) komutları mevcuttur. Kısaca çiftler denilen bu komutlar komut çözme iş hattının sonunda üretilirler. Doğru yollu veya vektör yollu olarak iş hattına gelen komutlar iş hattının sonunda bölünerek iki bağımsız komut haline gelirler. Bu şekilde 3-yollu olan komut çözme iş hattı tek çevrimde altı komut üretebilir.

Opteron'un En İyileme (optimization) Rehberi'nin Ek-3 kısmında hangi komutun hangi sınıfa ait olduğu belirtilmiştir. Çoğu 128 bit SSE ve SSE2 komutları çift komut olarak gerçeklenmiştir.

Sadece 64 bir olarak ikiye ayrılamayan komutlar vektör yollu olarak gerçeklenir. Aynı zamanda 128 bitlik saklayıcıların sadece bir yarısında işlem yapan SSE2 komutları doğru yollu komut olarak gerçeklenmiştir. SSE2 komutları için çift komutlar kullanmak Opteron işlemcisinde Pentium 4 işlemcisine göre %25 lik bir gecikme azalmasına sebep olur. Pentium 4 işlemcisinde bir SSE2 komutu kayan-nokta biriminde sonlara doğru ikiye ayrılır. Kayan-nokta birimi 128 bitlik kaynak veriyi ilk aşamada alır, işlemi ikiye böler ve son aşamada işlem sonuçlarını birleştirerek tek bir 128 bitlik sonuç elde eder. Son aşamada işlem sonuçlarının birleştirilmesi fazladan bir çevrim anlamına gelir. Sonuç olarak Pentium 4 işlemcisindeki x87 FADD ve FMUL komutları 5 ve 7 çevrim, 128 bitlik SSE2 eşlenikleri ise 6 ve 8 çevrim gerektirir. Opteron her iki x87 işlemini 4 çevrimde kotarır. Aynı şekilde SSE2 eşlenikleri de 4 çevrimlik gecikmeyle kotarılırlar. Çift komutlar sadece SSE ve SSE2 komutları için değil, aynı zamanda POP ve PUSH gibi x86 komutları, bazı tamsayı çarpma işlemleri ve LEAVE komutu için kullanılır.

SSE ve SSE2'de kullanılan 128bitlik bellek referansları da benzer şekilde ikiye bölünerek tamsayı çekirdeği tarafından kotarılan iki bağımsız 64bit referansa dönüştürülür. Sonuçlar Veri Cep Belleği (Data Cache) Veri Yükleme (Load Data) veri yollarından kayan-nokta çekirdeği tarafından alınır. Opteron'da Athlon'daki 32bitlik tamsayı saklayıcıları 64bit'e çıkarılmıştır ve 128bitlik komutlar da 64bit'e bölünmektedir, bunun sonucunda her şeyi 64bit olan bir mikro mimari ortaya çıkar.

Saklayıcı Tekrar-İsimlendirme ve Sırasız İşlem Yapma (Register Renaming and Out-of- Order Processing

AMD'nin Athlon işlemcilerinde olduğu gibi Opteron işlemcisinde de Saklayıcı Tekrar-İsimlendirme ve O-O (out-of-order) işlemlerde iş hattının boyunu %25 kısaltan bir takım teknikler kullanılmıştır. Yapılan tasarım basit ve hızlı olduğu gibi, aynı zamanda da cep-ıskalarından kaynaklanan yanlış sıralama (miss-scheduling) hatalarını kotarmak için herhangi bir fazladan donanıma ihtiyaç duymaz.

Saklayıcı tekrar isimlendirme, çevrim başına işlemcinin yürütebileceği komut sayısını (IPC-Instructions Per Cycle) sınırlandıran Yanlış Bağımlılık hatalarını ortadan kaldırmak için kullanılır. Yanlış bağımlılıklar saklayıcıların saysının sınırlı olmasından dolayı kaynaklanır. Daha önce kullanılacak olan bir saklayıcı değerine yazmak üzere olan bir komut, mutlaka saklayıcının yazılmadan önceki değerini kullanması gereken komutu beklemek zorunda kalmaktadır. Bu durum komut yürütme işleminin sırayla yapılmasını gerektirir ve IPC'yi sınırlar. Bu durum özellikle

oldukça az sayıda saklayıcıya sahip olan x86 mimarilerinde ortaya çıkar. Aşağıdaki örnekte saklayıcı tekrar isimlendirmenin yanlış bağımlılıkları nasıl ortadan kaldırabileceği gösterilmiştir: rC saklayıcısı üçüncü komut tarafından üzerine yazılmıştır, öyleyse üçüncü komut ikinci komutun sonlanmasını beklemelidir, yani yanlış bir bağımlılık... Saklayıcı tekrar isimlendirmesi sayesinde hali hazırda bulunan ama kullanılmayan saklayıcılar da gerektiği gibi kullanabilinir. Örnekteki rC saklayıcısını tekrar kullanmaya gerek yoktur, bunun yerine yeni bir saklayıcı kullanılarak bağımlılık çözülmüş olur (r7 gibi). Buradaki temel kural o anlık yürütülen tüm komutların farklı hedef saklayıcılarına yönlendirilmesidir.

tekrar-isimlendirmesiz: rC=rA+rB; rF=rC&rD; rC=rA-rB;

tekrar-isimlendirilmiş : r3=r1+r2; r6=r3&r4; r7=r1-r2;

Opteron'un 16x64bit tamsayı saklayıcısı vardır. Bunların yanında karmaşık x86 komutlarını kotaran mikro kod rutinlerinde kullanılmak üzere programcıya görünür olmayan 8x64bit saklayıcısı mevcuttur. Toplam tekrar isimlendirilmiş saklayıcı sayısı anlık yürütülen komut sayısı artı saklayıcı sayısı kadar veya daha fazla olmalıdır. Yani toplamda 72+16+8=96 tekrar isimlendirilmiş saklayıcıyı yönetmek gerekir. Bu saklayıcıları iki yapı yönetir; anlık yürütülen komutların sonuçları (instructions-in-flight results) içerisinde 72 giri bulunan Yeniden-Sıralama Tamponu (re-order buffer) (ROB)'nun sonuç bitleriyle, saklayıcılar ise Tamsayı Gelecek Dosyası ve Kayıt Dosyası (Integer Future File and Register File) (IFFRF) ile yönetilir.

Yeniden-Sıralama Tamponu Etiket Tanımı							
Wrap bit	Anlık Yürütülen Komut Numarası (Instruction In Flight Number)						
	Yeniden-sıralama tamponu indisi 0...23					alt-indisi 0..2	
bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0

Bu yapılanış oldukça basit olan ve sıfır çevrimde tamamlanabilen bir tekrar isimlendirme düzenlemesine olanak verir. Üç kod çözücünün birinden çıkan her komut bir “Tekrar İsimlendirme Tampon Etiketi” veya “Anlık Yürütülen Komut Etiketi” alır. Buradaki etket; komutun üç kod çözücünün hangisinden geldiğini belirten bir bölüm, komutun kaçıncı çevrimde salındığını belirten bir bölüm (0-23 arasında değer alır ve 23ten sonra 0'a döner) ve son olarak daha önce bir 0'a dönme işlemi gerçekleştiğinin belirtildiği bitten oluşur (iki komutun bu biti farklıysa çevrim sayacı 23'ten 0'a dönmüş ve devam etmiştir demektir).

Tamsayi Gelecek Dosyası ve Saklayıcı Dosyası (Integer Future File and Register File – IFFRF)

Saklayıcı dosyası 16 saklayıcıyı ve 8 geçici işlemler için kullanılan saklayıcı yönetiminde kullanılır. 16 saklayıcının her biri için iki tane girisi bulunur. Bu girilerden birisi programcının görebildiği gerçek saklayıcı değerine sahiptir, şöyle ki bu saklayıcının değeri kendisini yaratan komut sonlandığında (retired) yazılır. Bir komut ancak şu koşullarda sonlanabilir; komut herhangi bir olağan dışı durum veya yanlış dallanma öngörüsü hatasına sebep olmamıştır ve kendisinden önceki tüm komutlar sonlanmıştır. Bu komut sonucunda değişen saklayıcı "gerçek" (non-speculative) şeklinde isimlendirilir.

40 girili Tamsayı Gelecek Dosyası ve Saklayıcı Dosyası: IFFRF	
16 giri	Sonlanan Saklayıcı Değerleri
16 giri	Güvenilmeyen Saklayıcı Değerleri: "Gelecek Dosyası"
8 giri	Geçici Saklayıcılar

Anlık yürütülen komutlar sonlanmadıkları sürece iptal edilebilir veya göz ardı edilebilirler. İptal etmenin sebebi komutun olağan dışı bir duruma veya yanlış bir dallanma öngörüsü hatasına sebep olması olabilir. Anlık yürütülen her komut her zaman spekülatif-güvenilmeyen (speculative) olarak kabul edilir. Komutun yürütülmesinin bitmesi sonucun spekülatifliğini değiştirmez. Sonuçlar sadece komutlar sonlandığında (retired) sonlanma mantıksal birimi herhangi bir olağan dışı durum olmadığına karar verirse gerçek sonuç (non-speculative) olarak değerlendirilirler.

Gelecek Dosyası

Her 16 saklayıcı için bulunan iki giriden diğeri ise "gelecek" değerini taşıyan giridir. Bu girilerin tümü Gelecek Dosyası'nı oluştururlar. Buradaki giriler herhangi bir komut tarafından kendisine atanmış olan saklayıcının en güncel halini taşırlar. Gelecek dosyasının belirli bir girisi, o giriyi yaratan komut sonlanmadığı sürece spekülatif olarak değerlendirilir. Kendisini yaratan komut temiz bir şekilde sonlandığında girinin değeri spekülatiflikten çıkar.

Komutlar sonuç üretilir üretilmez gelecek dosyasına sonucu yazarlar ancak; gelecek dosyası sonucu eğer belirli bir saklayıcının en güncel sonucu değilse kabul etmez. Eğer sonradan gelen bir komut erken biter ve sonucunu gelecek dosyasına yazarsa, gelecek dosyası buradaki saklayıcı için daha eski olan komutlardan sonuç kabul etmez. Sonuçlanmış olan komutlar gelecek dosyasını komut

kodu saklayıcı numarası(0-15 arasında bir sayı) ile adreslerler. Saklayıcıdaki sonucun üzerine yazılıp yazılmadığı “tekrar sıralama tamponu etiketi” ile anlaşılır, her gelecek dosyası girisinin kendisine karşılık düşen bir etiketi vardır. Daha sonra anlatılacağı gibi komutlar gelecek dosyasına eğer hala kendi etiketi ile girideki etiket tutuyorsa, yani gelecek dosyasındaki giriye o komut hala sahipse yazabilir.

Tüm spekülatif olan komutlar IFFRF'in sonlanmış komutlar tarafından yazılan değerleri gelecek dosyasına kopyalanarak iptal edilebilir. Spekülatif olan değerler sonlanma mantıksal birimi anlık veya önceki bir komuttan kaynaklanan olağan dışı bir durum tespit ettiğinde iptal edilmelidirler.

Yeniden-sıralama Tamponu

Sonlanma işlemi yeniden-sıralama tamponu yardımıyla kotarılır, tampon isminden de anlaşılabileceği gibi komutları tekrar sıralar. Komutlar yeniden-sıralama tamponu vasıtasıyla olağan program akış sırasına dönerler. Sıralayıcılar sıralamasız (O-O) yürütme ile ilgili işlerden sorumludurlar. Sıralayıcılar komutları yürütme birimlerine kaynak operantları hazır ve yürütme birimi müsait olduğu zaman gönderirler. Bu işlem sırasında sıralayıcıdan çıkan komutların sırası değişebilir, yeniden-sıralama tamponunun görevi bu sıralamayı olağan program akışı sırasına dönüştürmektir.

Yeniden-Sıralama Tamponu İşlemi

indeks	1	2	3	4	5	6	7	8	9	10	11	12
şerit 0												
şerit 1												
şerit 2												

	= Sırasız sonlanmış komutlar, sonuçlar halen güvenilir değil
	= Şu anda sonlanan komutlar.
	= Sonlanan komutlar, sonuçlar güvenilir, doğru.

Opteron işlemcisinin yeniden-sıralama tamponunda her birinde 24 giri bulunan 3 satır mevcuttur. Buradaki her satır ve giri her komuta atanmış olan yeniden-sıralama etiketine karşılık

düşer. Biten her yürütme işleminde komut sonucunu tampon etiketini adres olarak kullanarak buraya yazar. Komutlar sonuçlarla beraber yürütme sırasında olağan dışı bir duruma sebep olabilecek durumları da bu tampona kaydederler. Komutlar aynı zamanda yeniden-sıralama işinin yapılabilmesi için gerekli olan komutu türü, hedef saklayıcısının donanımsal saklayıcı adresi (0-15), komutun komut belleğindeki adresi gibi bilgileri kaydederler. yeniden-sıralama tamponu tüm komutlar tarafından ortak kullanılır. Kayan-nokta, SSE(2) ve MMX komutlarını da yeniden-sıralama işlemi için de kullanılır ancak bu komutlar sonuç verilerini tampona değil de 120 giri içeren tekrar-isimlendirilmiş kayan-nokta saklayıcı dosyasına (renamed floating-point register file) yazarlar. Sonuç verisi haricindeki bilgiler ise halen tamponda saklanır, tüm komutlar yeniden-sıralama tamponunun aracılığı ile sonuçlanırlar.

Sıralamasız-Yürütme (out-of-order processing)

Öncelikle komut dağıtma (instruction dispatching) aşamasından bahsetmek gerekir. Komut dağıtma işlemi komutların doğrudan yürütme birimlerine gönderilmesi yerine sıralayıcılara gönderilmesi anlamına gelir. Komutlar sıralayıcı biriminden önce saklayıcı dosyasına erişebilir. Üç komut her çevrimde toplam dokuz saklayıcı değerini IFFRF biriminden okuyabilir. Bununla birlikte gelecek dosyasına da erişim mümkündür. Buradaki değerler daha önce de bahsedildiği gibi en son spekülatif saklayıcı değerlerini (toplam 16 saklayıcı için) içermektedir. Tüm saklayıcı değerleriyle birlikte bu üç komut üç ayrı sıralama birimine yerleştirilir.

Önceki tüm komutların bitmediğini, yani saklayıcı değerlerinin önceki komutlardan kalan eski değerler olduğunu düşünmek yüksek olasılıklı bir varsayım olacaktır. Dağıtılan (dispatched) her komut içeriğini değiştireceği hedef saklayıcısının geçerlilik bayrağını sıfırlar, aynı zamanda kendi etiketini de bu saklayıcı için kullanır. Başarıyla sonuçlanmış olan komutlar bu yolla gelecek dosyası girisinin artık geçerli olmadığını anlarlar, aynı zamanda daha sonra gereken veriyi kendilerine sağlayacak olan komutun etiketini de bilirler. Bu etiketi kullanarak sonucu yaratıldığında doğrudan sonuç veri yollarından alabilirler. Saklayıcıdaki değeri geçersiz kılan komut daha sonra sonlanır ve eğer hala gelecek dosyasındaki giriye sahipse sonucu buraya yazar ve geçerlilik bayrağını birler. Komutun gelecek dosyasındaki giriye sahip olması komutun etiketinin girideki etiketle aynı olması demektir. Komut dağıtıldığında bu sahipliği kazanır ve eğer daha sonra aynı hedef saklayıcıya sahip başka bir komut dağıtılırsa bu sahipliği kaybeder. Komut yalnız gelecek dosyasındaki giriye halen sahipse sonucunu buraya yazabilir. Takip eden komutlar sonucu buradan alabilirler. Eğer komut sahipliğini yitirmişse giriyi değiştirmez; saklayıcıdaki değere ihtiyaç duyan herhangi başka bir

komut zaten dağıtılmış ve sonucu sonuç veri yollarından doğrudan almıştır.

Komut Cep Belleği

Opteron işlemcisi toplamda 102 kByte bir komut cep belleğine sahiptir. Komut cep belleğine erişim 128 bittir. Her çevrimde 16byte'lık komut satırı okunabilir. Komut bitleriyle beraber 76 bit bilgi de komutlarla beraber okunur. Bu bitlerle beraber komut cep belleğine toplam erişim 204 bit olur. Komut satırlarına eklenen bu bilgi bitlerinden en önemlileri ön-çözme (pre-decode) bitleridir. Bu bitler değişken boyutlu olan karmaşık x86 komutlarının başlangıç ve son noktalarını belirler ve komut hakkında bir takım fonksiyonel bilgi sağlarlar. Diğer bilgi bitlerinden birisi de eşlik (parity) bitleridir. Her 16bit için 1 eşlenik biti eklenir. Son bilgi bitleri de dallanma seçicileridir ve bu bitler de her 2bytelık komut satırı için 2bit olarak düzenlenmişlerdir.

	Ram Boyutu	Veri Yolu Boyutu	
Komut Kodu	64 kBayt	128 bit	16 bayt komut kodu
Eşlik biti	4 kBayt	8 bit	Her 16 bit için bir eşlik biti
Ön-çözme bitleri	26 kBayt	52 bit	Her bayt için 3 bit (başla, bitir, fonksiyon) + her 16 baytlık komut satırı için 4bit
Dallanma Seçicileri	8 kBayt	16 bit	2 bayt komut kodu için 2bit
TOPLAM	102 kBayt	204 bit	

Ön-çözme (predecode) bitleri

Komut cep belleğindeki her sekizli ön-çözücü birimi tarafından atanmış olan üç ön-çözme bitine sahiptir. Her komut sekizlisinde o sekizlinin bir komut başlangıcı olduğunda birlenen bir başlangiç biti ve benzer şekilde eğer sekizli komutun son sekizlisiyle son sekizli oldıuğunu belirten bir son biti vardır. Eğer tek sekizli boyutunda bir komut varsa iki bit de birlenir. Son ön-çözme biti ise komut hakkında biraz daha fazla bilgi veren fonksiyon bitidir. Komut çözücü ilk olarak komutun en son sekizlisindeki fonksiyon bitine bakar, eğer buradaki fonksiyon biti sıfırsa komut doğrudan-yollu bir komuttur ve bu komut doğrudan yütürme birimleriyle kotarılabilir. Fonksiyon biti eğer 1 ise komut vektör-yollu bir komuttur ve yürütülmesi biraz daha karmaşık olan mikro programlarla kotarılırlar.

Opteron işlemcisinin sadece iki çevrimde tüm 16bytelık bir komutu ön-çözebilme yetisi vardır. Uzunlukları çeşitli olan komutların çözülmesindeki en büyük problem ilk komutun boyutu

öğrenilinceye kadar ikinci komutun başlangıç sekizlisinin bilinememesidir. Opteron'un paralel ön-çözücüsü bu problemi olabilecek olan tüm 16 konumu çözerek aşar. Her komut 16byte'ın bir baytından başlamış gibi kabul edilir ve çözülür. Daha sonra gerçek olan komutlar, bir sonraki komutun adresini gösteren program sayacı sayesinde bulunur. Aslında bu ön-çözücü Athlon işlemcisinde bulunan ön-çözücünün geliştirilmiş versiyonudur. Önceki çözücüde bir tane çözme bloku aynı anda dört olası komutu çözebilmekteydi ve 16bytelık komut satırını çözmesi için 4 çevrim gerekiyordu. Şimdi ise Opteron'da aynı ön-çözme blokundan 4 adet vardır ve iki çevrimde 16bytelık komut satırını çözebilmektedir. Dallanma ile ilgili açıklamalar önceki bölümlerde yapılmıştır.

Referanslar

[1] Understanding the detailed Architecture of AMD's 64 bit Core,
http://chip-architect.com/news/2003_09_21_Detailed_Architecture_of_AMDs_64bit_Core.html

[2] US-PATENT 6510508, Translation lookaside buffer flush filter,
http://www.freepatentsonline.com/US6510508.html

[3] Software Optimization Guide for AMD64 Processors,
http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/25112.PDF

[4] Keltcher, C.N., McGrath K.J., Ahmed, A. & Conway, P. (2003). The AMD Opteron Processor for Multiprocessor Servers. IEEE Computer Society.

[5] Weber, F. (2002). Past, Present and Future Hammer in Context. Presented in AMD Developer Symposium in 2002.

[6] http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_8796_15223,00.html