UNIVERSIDADE DE SÃO PAULO

Escola de Artes, Ciências e Humanidades

Relatório Técnico PPgSI-001/2021

Descrição da Arquitetura Intel Core i7 - 900 series

Maurício Ryuzo Tocura
Pedro Augusto Santos Giorgi
João Marcos Garcia Fagundes
Norton Trevisan Roman

Fevereiro - 2021

Série de Relatórios Técnicos

PPgSI-EACH-USP
Rua Arlindo Béttio, 1000 – Ermelino Matarazzo
03828-000 – São Paulo, SP.
TEL: (11) 3091-8197 http://www.each.usp.br/ppgsi

Descrição da Arquitetura Intel Core i7 - 900 series

Maurício Ryuzo Tocura[1], **Pedro Augusto Santos Giorgi**[1], **João Marcos Garcia Fagundes**[1], **Norton Trevisan Roman**[1]

[1]Escola de Artes, Ciências e Humanidades – Universidade de São Paulo
São Paulo – SP, Brasil

mauriciotocura@usp.br,

pedro.giorgi@usp.br,

joaofagundes@usp.br,

norton@usp.br

Resumo. *Neste relatório, são descritas as características do Intel Core i7, da primeira geração do processador (900 series), com a microarquitetura Nehalem.*

1. Introdução

O primeiro microchip comercial foi lançado em 1971 pela Intel. Este foi nomeado Intel4004, sendo um processador de 4 bits e pouco mais de 2000 transistores. A partir disso a Intel se dedicou ao segmento de fabricação de microprocessadores, se tornando uma das maiores responsáveis pelas tecnologias utilizadas atualmente [Almeida 2009].

No fim de 2008 a Intel lança seu novo processador quad-core, o Intel Core i7, sendo o primeiro processador baseado na microarquitetura Nehalem. Este processador traz consigo uma série de inovações em desempenho quad-core e conta com tecnologias avançadas de processador [Teles 2009].

Sendo este processador mais rápido e possuindo uma tecnologia de multi-core mais inteligente, ele implementa melhorias importantes, como:

- Controladora de memória integrada.
- Intel Quick Path Interconnectio (QPI).
- Barramento ponto-a-ponto baseado no Intel QPI.
- Retomada do HyperThreading.
- Utilização do Turbo Boost.

2. Microarquitetura Nehalem

A microarquitetura Nehalem da Intel permitiu a criação de muitas características inovadoras no processador. Construído com a microarquitetura Intel Core 45nm, é possível listar algumas de suas principais características [Thomadakis 2011, Intel 2016a]:

- Melhoria no núcleo do processador
  - Unidade de previsão de desvios[1].

[1]O previsor de desvios agora conta com dois estágios. O segundo estágio possui um histórico maior, portanto é mais lento, mas pode realizar previsões mais precisas. Desta forma diminui-se ainda mais as chances de uma previsão incorreta e da necessidade de retornar até o desvio para continuar a execução, desperdiçando vários ciclos e ainda impondo uma penalização de outros tantos ciclos para reorganizar o pipeline. O que, por coincidência, também foi acelerado com a ajuda do "Renamed Return Stack Buffer", que guarda cópias dos dados já calculados Dessa maneira, em caso de previsão incorreta, menos dados deverão ser recalculados. [Teles 2009].

  - Melhora no streaming de loops para aumentar o desempenho do front-end e reduzir o consumo de energia.
  - Maior profundidade nos buffers[2] de retenção para poder suportar maiores níveis de instruções em paralelo.
- Tecnologia de Hyper-Threading
  - Suporte para duas threads de hardware (processadores lógicos) por núcleo.
  - O cache L3 está maior, e também há uma maior largura de banda.
- Acesso à Memória
  - Controlador de memória integrado, no chipset.
  - Nova organização hierárquica de cache com compartilhamento, inclusive L3.
  - Aumento do tamanho da TLB (Translation Look-a-side Buffer)[3] e ganho de mais um nível.
  - Rápido acesso desalinhado à memória[4].
- Gerenciamento Dedicado de Energia
  - Micro controle integrado com firmware que gerencia o consumo de energia embutido.
  - Sensores de tempo real para temperatura e energia embutidos.
  - Versatilidade para reduzir o consumo de energia na memória e no QPI.
- Execução especulativa, fora de ordem e superescalar [Bruschi 2017]

2.1. Instruction Set Architecture (ISA)

A Arquitetura do Conjunto de Instruções (ISA) é a visão lógica de um computador. Um ISA especifica precisamente todas as instruções da máquina, objetos que podem ser diretamente manipulados pelo processador e sua representação binária. As instruções da máquina determinam quais operações elementares o processador pode executar. Esse conjunto de instruções pode ser classificado como CISC (Complex Instruction Set Computer) ou RISC (Reduced Instruction Set Computer) [Thomadakis 2011].

A Intel definiu o Intel64 como o seu 64-bit ISA, o qual derivou do IA-32. Esse conjunto de instruções suporta o conjunto IA-32 e tem suporte nativo para aplicações de 64-bit e sistemas operacionais também de 64-bit [Thomadakis 2011].

O espaço de endereçamento físico pode chegar a 48 bits, o que leva a 256 Tera-binary-Bytes (TiB) que podem ser diretamente endereçados pelo hardware. O tamanho do endereçamento lógico do Intel64 é de 64-bit que permite um endereçamento linear (flat linear address) de 64-bit [Thomadakis 2011].

---

[2]Aumento dos buffers de instruções para acomodar o maior número de instruções devido à implementação do Hyper-Threading

[3]A TLB (tabela para consulta rápida de endereços de memória) agora possui um segundo nível, com 512 entradas [Teles 2009].

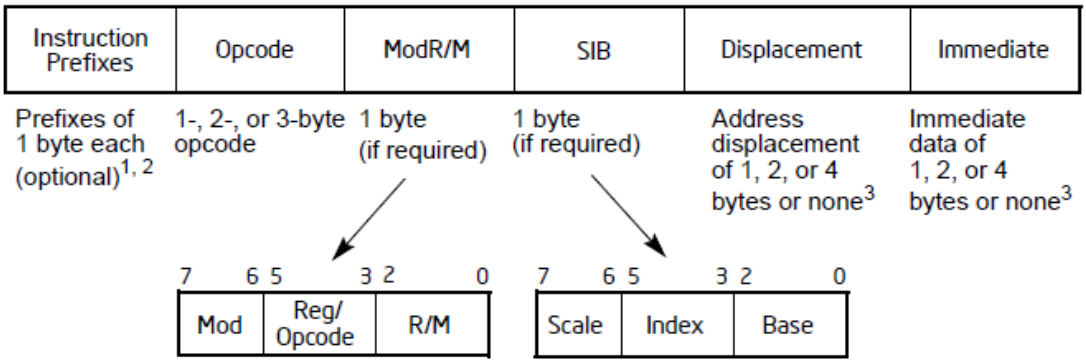
[4]De acordo com Drake e Berg [2021] o acesso desalinhado à memória ocorre quando tenta-se ler um dado de N bytes que começa em um endereço que não é divisível por N (ou seja, endereço % N != 0). Segundo o manual da arquitetura para desenvolvedores, [Intel 2016a], as words, doublewords, e quadwords não precisam estar alinhadas aos limites naturais na memória, isto é, elas não precisam estar em endereços pares, divisíveis por quatro, oito e assim por diante. Porém sempre que possível as words, doublewords e quadwords devem estar alinhadas a estes limites naturais para que os programas possam ter um melhor desempenho. Quando isto não ocorre, é o caso de um acesso desalinhado à memória, onde são necessários dois acessos à memória, enquanto, quando alinhado, é preciso somente de um acesso.

No modo de 64-bit do Intel64, um programa pode acessar aplicações, registradores e ponteiros de 64-bit. De modo geral, mesmo que o Intel64 seja construído com base no conjunto de instruções do tipo CISC, esta microarquitetura da Intel compartilha muitos mecanismos em comum com o conjunto de instruções do tipo RISC [Thomadakis 2011].

O Intel64 implementa o MMX (Multimedia Extensions), streaming SIMD extensions (SSE), streaming SIMD extensions 2 (SSE2), streaming SIMD extensions 3 (SSE3), supplemental streaming SIMD extensions 3 (SSSE3), streaming SIMD extensions 4 (SSE4), que são conjuntos de instruções capazes de processar vários dados em um único ciclo de clock. O conjunto de instrução SSE é um conjunto de 128-bit [Grover e Agrawal 2014].

De acordo com o manual da arquitetura para desenvolvedores, [Intel 2016b], e conforme a Figura 1, uma instrução do Intel64 consiste em:

- Prefixo de Instrução (opcional e em qualquer ordem).
- Bytes do opcode (no máximo 3 Bytes).
- Um especificador de endereçamento (Addressing-form specifier) (se necessário), que consiste no ModR/M Byte e algumas vezes o SIB (Scale-Index-Base) byte.
- Deslocamento (se necessário).
- Campo de dados imediato (se necessário).



1. The REX prefix is optional, but if used must be immediately before the opcode; see Section 2.2.1, "REX Prefixes" for additional information.
2. For VEX encoding information, see Section 2.3, "Intel® Advanced Vector Extensions (Intel® AVX)".
3. Some rare instructions can take an 8B immediate or 8B displacement.

Figura 1. Formato da instrução do Intel64 [Intel 2016b]

2.2. Visão Geral do Chip do Processador

Segundo Thomadakis [2011], um chip com a microarquitetura Nehalem possui os seguintes componentes, como pode ser visto na Figura 2.

- Quatro núcleos idênticos.
- UnCore Interface Unit (conecta os 4 núcleos ao cache L3, além do controle de memória integrado ao QPI).
- Cache L3.
- Controle de memória integrado com 3 conexões DDR3 com a memória.

- 2 portas QPI.
- Além de um circuito auxiliar para a coerência de cache, controle de energia, gerência do sistema e monitoramento de desempenho.

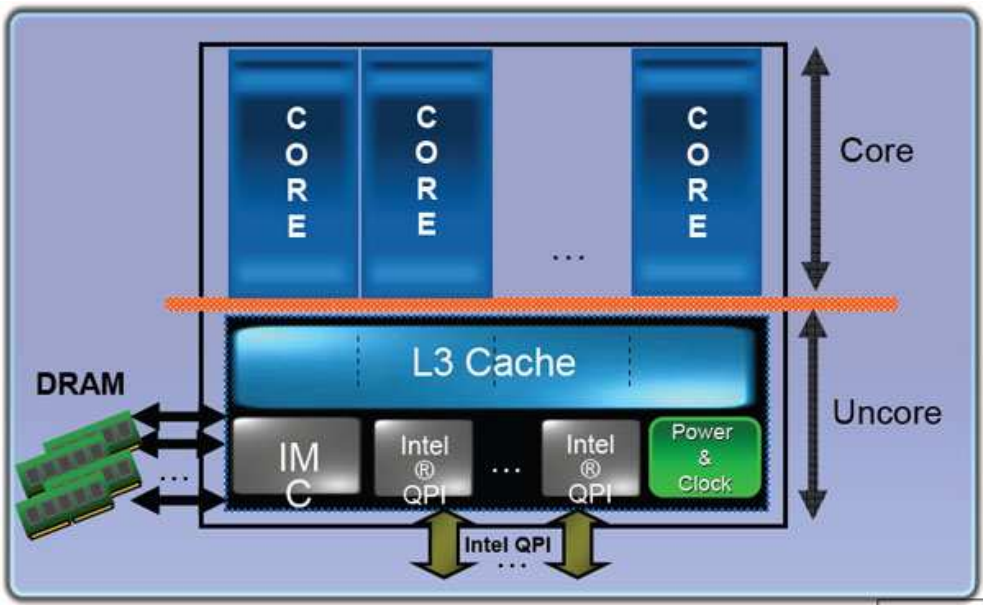


Figura 2. Modularização Intel Core i7 [Grover e Agrawal 2014]

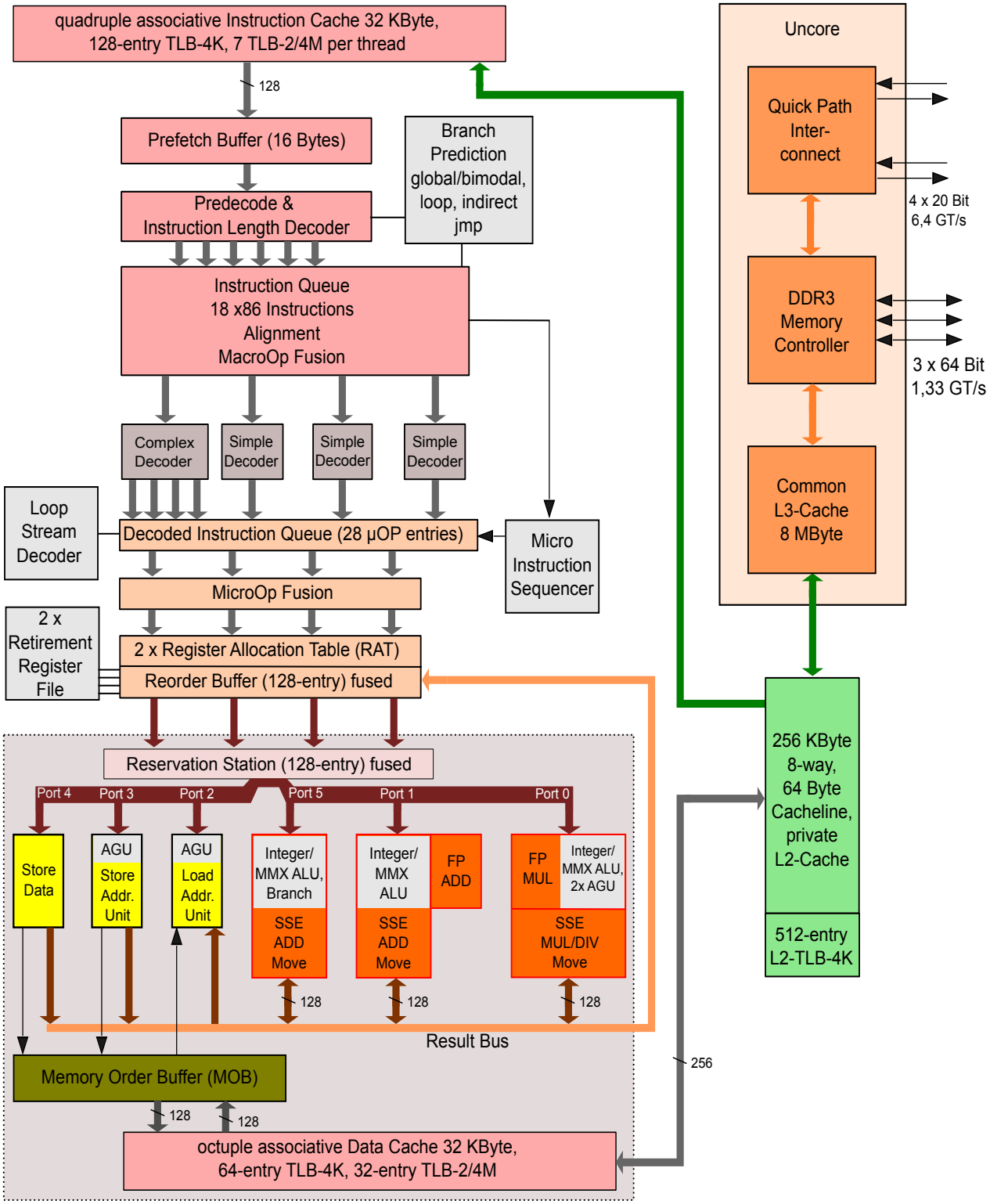
A microarquitetura Nehalem foi construída com foco na modularidade. O chip é dividido em dois módulos principais, o Core e o Uncore (Figura 2). Os componentes que se encontram no Core operam com a mesma frequência de clock, já os componentes que estão no Uncore em uma frequência diferente, o que permite que cada módulo rode independentemente dos demais [Grover e Agrawal 2014].

De modo geral, um chip Nehalem pode ter de dois a oito núcleos, quantos QPI's forem necessários, diferentes tamanhos de cache L3, assim como diferentes bandas de DRAM. Fora do chip, porém ainda muito próximo a ele, encontramos a DRAM, que está acessível através dos 3 canais DDR3 de 8 bytes, cada um sendo capaz de uma transferência de 1,333 GigaTransfers/segundo [Thomadakis 2011].

Em resumo, o módulo Uncore contém o controlador de memória e lógica de cache que antigamente costumava ser implementada pelo chip North-Bridge de modo separado. O alto desempenho do Nehalem deve-se, principalmente, pelo fato de o controlador da DRAM, o L3 e as conexões QPI estarem todos na mesma matriz de silício que os núcleos [Thomadakis 2011].

Segundo o manual da arquitetura para desenvolvedores, [Intel 2016a], os processadores Core i7 suportam as tecnologias quad-core, Hyper-Threading, além de fornecer o QPI (Quick Path Interconnect) para que possa se conectar à placa-mãe, e possui um controlador de memória integrado que suporta três canais de memória DDR3, conforme ilustram as Figuras 3. Nela, vemos de modo mais detalhado a microarquitetura Nehalem, onde o QPI e os três canais de memória DDR3 se encontram no canto superior direito, na região identificada como Uncore. O restante da figura será explicado de modo mais detalhado mais à frente, na seção 2.9.1 deste relatório.

Intel Nehalem microarchitecture



GT/s: gigatransfers per second

Figura 3. Microarquitetura Intel Nehalem [Bruschi 2017]

2.3. Organização de Memória

O controle de memória integrado suporta 3 canais DDR3 de 8 bytes que operam até no máximo 1,333 GigaTransfers/segundo. Teoricamente com uma largura de banda entre o DRAM e o controle integrado de 31,992 GB/s. Cada canal de memória opera independentemente, e cada núcleo suporta até no máximo 10 data cache missing e 16 outstanding missing [Thomadakis 2011].

O Nehalem divide a memória física em blocos com o tamanho de 64 bytes, sendo estes blocos chamados de blocos de cache. A hierarquia de cache é dividida em 3 níveis. Cada núcleo contém o primeiro (L1) e o segundo (L2) níveis de cache, e, o terceiro nível (L3) se encontra no módulo Uncore do processador, como pode ser observado na Figura 4.

- Cache L1 (nível 1): 32KB para dados e 32 KB para instruções. Contém apenas uma porta de conexão e tem o tamanho de 64 bytes. A latência de acesso para retornar um dado que está em L1 é de 4 clocks e o período da vazão é de 1 ciclo de clock [Thomadakis 2011].
- Cache L2 (nível 2): cada núcleo contém um, com a capacidade de 256KB, o tamanho do seu bloco é de 64 bytes e a latência para retornar um dado que está em L2 é de 10 ciclos de clock. A política de escrita é o write-back e o cache é não inclusivo [Thomadakis 2011].
- Cache L3: compartilhado por todos os núcleos no processador, com 8MB, sua latência varia devido as diferenças entre as frequências de clock entre o Core e Uncore, logo sua latência é de aproximadamente 35 $\sim$ 40+ ciclos de clock [Thomadakis 2011].

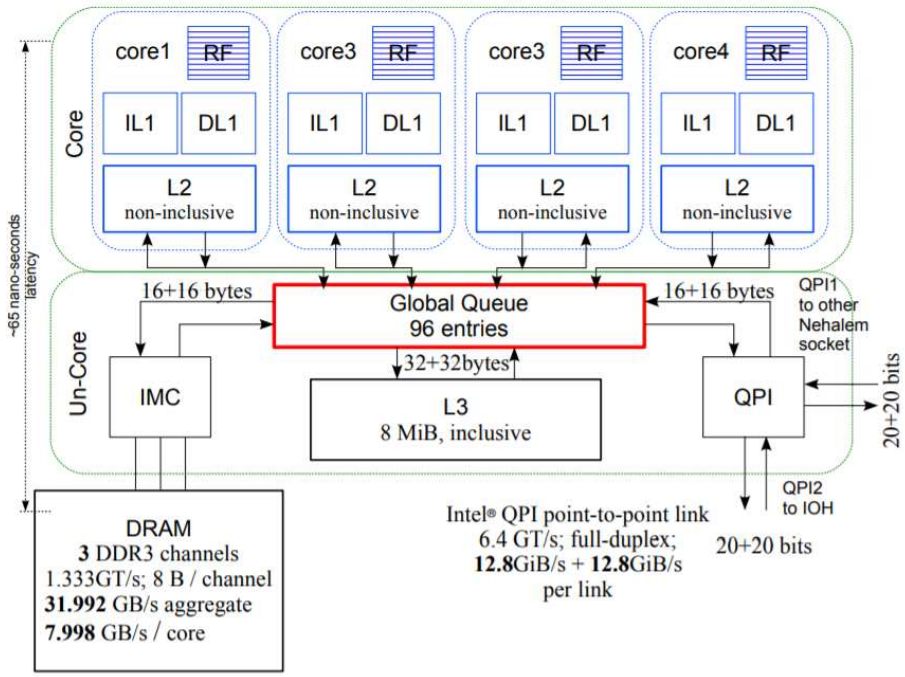


Figura 4. Hierarquia de memória no Intel Core i7 [Thomadakis 2011]

Os caches dos Core2 e Nehalem são organizados de forma inclusiva. Assim, cada nível superior guarda uma cópia do nível anterior. O cache L2 de cada núcleo possui uma

cópia do cache L1 e o cache L3 guarda uma cópia de cada cache L2; portanto, dos 8MB, sobram efetivamente 7MB (já que 1MB é reservado para cópia dos quatro caches L2 de 256KB). Este sistema requer cuidados para que seja mantida a consistência dos dados; pois cada vez que um cache é atualizado, suas cópias também devem ser atualizadas. Porém, facilita o compartilhamento de dados entre os núcleos, já que todos os dados presentes nos caches L1 e L2 de todos os núcleos são encontrados no cache L3 [Teles 2009].

2.4. Consumo de Energia

Considerando uma máquina com dois processadores quad-core e oito módulos de memória FB-DIMM para comparação, cada processador consome até 120 W e cada módulo de memória 12 W. Somados aos quase 40 W do North-Bridge tem-se aproximadamente 375 W, sem considerar o restante da máquina [Teles 2009].

Em uma máquina semelhante, baseada em processadores Nehalem, o consumo dos processadores deve permanecer o mesmo, mas o consumo do North-Bridge cai para níveis desprezíveis (10 a 15 W) e mesmo aumentando o número de módulos de memória para 12 (totalizando apenas 60 W, contra 96 W dos oito módulos FB-DIMM do caso anterior) o consumo do ”conjunto-matriz”deve cair para cerca de 315 W. Neste caso já pode-se constatar uma redução de pelo menos 50 W no consumo, que vem acompanhado de um sensível aumento no desempenho [Teles 2009].

2.5. Hyper-Threading

A tecnologia Hyper-Threading, desenvolvida pela própria Intel, é mais uma técnica criada para oferecer maior eficiência na utilização dos recursos de execução do processador. Esta tecnologia simula em um único processador físico dois processadores lógicos. Cada processador lógico recebe seu próprio controlador de interrupção programável (APIC) e conjunto de registradores. Os outros recursos do processador físico, tais como cache de memória, unidade de execução, unidade lógica e aritmética, unidade de ponto flutuante e barramentos, são compartilhados entre os processadores lógicos. Em termos de software, significa que o sistema operacional pode enviar tarefas para os processadores lógicos como se estivesse enviando para processadores físicos em um sistema de multiprocessamento [Teles 2009].

Deste modo, com o Hyper-Threading dois núcleos virtuais são criados a partir de um núcleo físico e como o Core i7 é quad-core, então tem-se um total de 8 núcleos virtuais, como pode ser visto na Figura 5. É importante ressaltar que o Hyper-Threading ou SMT (Simultaneous Multi-Threading) trata-se de núcleo 4-issue wide (a cada ciclo cada núcleo faz o fetch de 4 instruções para processar), conforme ilustra a Figura 6. Nela cada bloco representa uma unidade de execução, sendo os blocos de cor azul a primeira thread e os de cor verde da segunda thread [Teles 2009].

A Figura 6 mostra que, ao invés de esperar que um bloco de instruções seja executado, para então passar ao seguinte, o núcleo físico pode receber mais instruções simultaneamente com dois núcleos virtuais. Se estas forem de tipos diferentes, sua execução pode ser agendada ao mesmo tempo em que outras instruções são executadas, em outras unidades, proporcionando um significativo ganho em desempenho [Teles 2009].

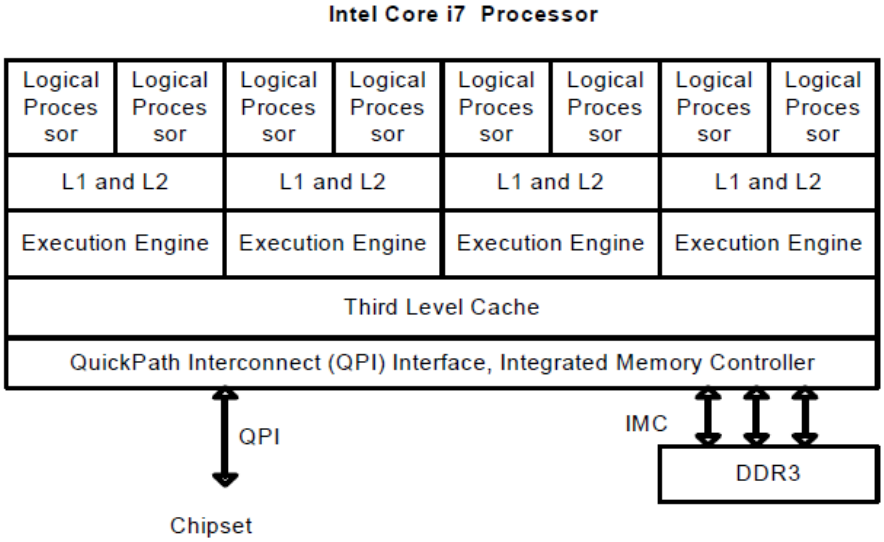


Figura 5. Processador Intel Core i7 [Intel 2016a]

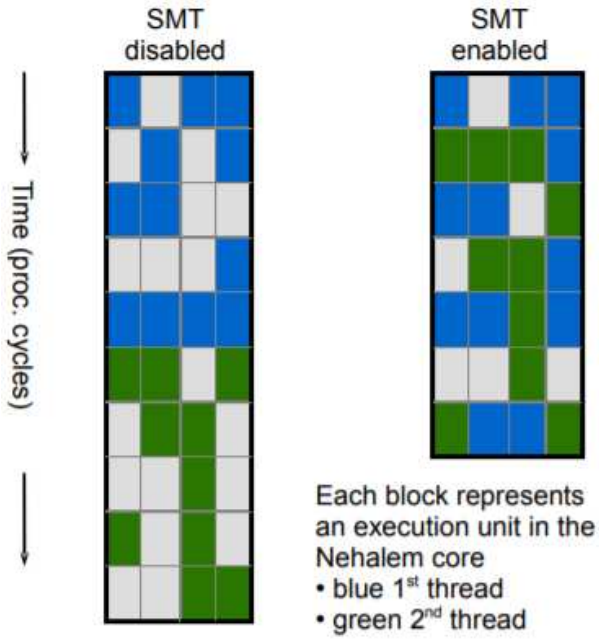


Figura 6. Hyper-Threading no Nehalem [Teles 2009]

2.6. Intel Turbo Boost

A microarquitetura Nehalem possui diversas características que contribuem para a economia de energia. Uma dessas características é o TBT (Turbo Boost Technology), que não apenas "desliga" núcleos ociosos como também aumenta a velocidade do clock dos núcleos que estão sendo utilizados [Thomadakis 2011].

A frequência máxima da tecnologia Intel Turbo Boost depende do número de núcleos ativos. O tempo que o processador gasta no estado da tecnologia Turbo Boost depende da carga de trabalho e do ambiente operacional, proporcionando o desempenho de que

o usuário precisa, quando e onde for necessário. Os elementos que podem definir o limite superior da tecnologia Turbo Boost em uma determinada carga de trabalho são os seguintes: número de núcleos ativos, consumo estimado de corrente, consumo estimado de energia e temperatura do processador [Teles 2009].

Quando o processador estiver operando abaixo desses limites, e a carga de trabalho do usuário exigir desempenho adicional, a frequência do processador aumentará dinamicamente 133 MHz em intervalos curtos e regulares até ser alcançado o limite superior ou o máximo upside possível para o número de núcleos ativos. Por outro lado, quando algum desses limites for alcançado ou ultrapassado, a frequência do processador cairá automaticamente 133 MHz até que ele volte a operar dentro dos seus limites [Teles 2009].

Por exemplo, com 3 núcleos ativos, o processador de 2,26 GHz pode rodar os núcleos a uma frequência de 2,4 GHz. Quando os núcleos se fazem necessários novamente, eles voltam ao seu estado original dinamicamente, ajustando também a frequência do processador [Thomadakis 2011].

2.7. Quick Path Interconnect

Uma importante mudança no projeto da CPU foi a troca do antigo barramento FSB (Front Side Bus), que compartilhava acessos entre a memória e a I/O, pelo novo barramento QPI (QuickPath Interconnection), que é projetado para aumentar a largura de banda e diminuir a latência [Teles 2009].

O QPI utiliza dois caminhos para a comunicação entre a CPU e o chipset, como pode ser visto na Figura 7, onde é possível notar que cada quadrado identificado como "processor" seria um core da CPU, e entre o "processor" e o "chipset" existe uma comunicação de dois caminhos, que seria de modo simplificado, o QPI. Isto permite que a CPU faça a operação de transmissão e recepção dos dados de I/O ao mesmo tempo, isto é, os datapaths de leitura e escrita para esta função são separados. Cada um destes caminhos transfere 20 bits por vez. Destes 20 bits, 16 são utilizados para dados e os restantes são usados para correção de erro CRC (Cyclical Redundancy Check), que permite ao receptor verificar se os dados recebidos estão intactos [Teles 2009].

Além disso, o QPI trabalha com uma frequência de 3,2 GHz transferindo dois dados por ciclo (uma técnica chamada DDR, Double Data Rate), fazendo o barramento trabalhar como se estivesse operando a uma taxa de 6.4GHz. Como 16 bits são transmitidos por vez, tem-se uma taxa teórica máxima de 12,8 GB/s em cada um dos caminhos. Comparado ao FSB, o QPI transmite menos bits por ciclo de clock mas opera a taxas muito maiores. Outra vantagem em ralação ao FSB é que, como o FSB atende a requisições de memória e de I/O, há sempre mais dados sendo transferidos neste barramento comparado ao QPI, que atende apenas às requisições de I/O. Por isso, o QPI fica menos ocupado, e assim há uma maior largura de banda disponível. Por último, o QPI utiliza menos ligações do que o FSB [Teles 2009].

Uma característica incorporada ao QPI são os modos de energia que ele pode assumir chamados de L0, L0s e L1. O L0 é o modo no qual o QPI está em funcionamento pleno. O estado L0s indica que os dados de conexão[5] e os circuitos que os controlam

[5]Também conhecidos como wire data, são os dados que definem as comunicações entre os dispositivos cliente e servidor.

estão desativados para economia de energia. Em L1 todo o barramento está desativado, economizando ainda mais energia. Naturalmente, o estado L1 necessita de um tempo maior para reativação do que o L0s [Teles 2009].

Teoricamente, o barramento QPI deveria ser chamado de conexão ponto-a-ponto, pois conecta apenas dois dispositivos. Entretanto, vale ressaltar que os dados são enviados em paralelo através das várias conexões ponto-a-ponto existentes [Teles 2009].

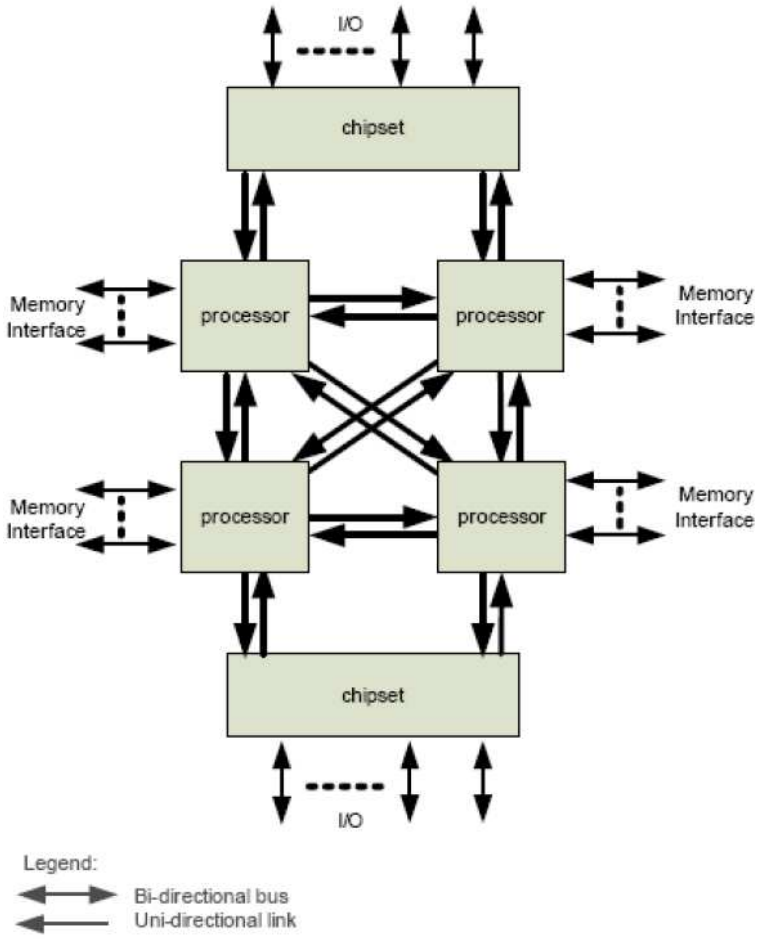


Figura 7. QPI da Intel [Teles 2009]

2.8. Coerência de Cache - Protocolo MESI+F

Pelo fato da microarquitetura Nehalem ser quad-core, e cada core poder ler e/ou escrever em um mesmo endereço, podem haver casos de incoerência entre os caches. Para lidar com tal situação a microarquitetura faz uso do protocolo MESIF (Modified, Exclusive, Shared, Invalid, Forwarding). [Qian e Yan 2008, Thomadakis 2011].

Este protocolo é uma versão adaptada do protocolo MESI, onde foi incluído um novo estado, o Forwarding. Antes de começar a falar sobre o MESIF, é preciso entender seu antecessor. O protocolo MESI, de acordo com Qian e Yan [2008], possui quatro estados, sendo eles:

- Modified (M): O cache line[6] fica exclusivamente neste cache apenas, e o seu

[6]Blocos de tamanho fixo que contêm os dados que são transferidos entre a memória e o cache.

conteúdo é modificado em relação à memória.

- Exclusive (E): O cache line fica exclusivamente neste cache apenas, e o seu conteúdo é igual ao da memória.
- Shared (S): O cache line que fica neste cache é compartilhado com outros caches, e seu conteúdo é o mesmo da memória.
- Invalid (I): O cache line contém uma cópia não válida da memória.

O cache é dono das linhas modified/exclusive e seus conteúdos podem ser alterados sem precisar avisar outros caches. Se um cache local tentar escrever uma linha da qual ele não é dono (estado Shared), ele precisa transmitir uma mensagem para todos os outros caches remotos. Agora, se um cache local tentar ler o conteúdo que é tido por outros caches, o dono precisa abrir mão da propriedade deste conteúdo [Qian e Yan 2008].

Como dito anteriormente, a microarquitetura Nehalem utiliza o protocolo MESI adaptado, onde é acrescido mais um estado, o de Forwarding, e é alterado o papel do estado Shared, este protocolo é conhecido como MESIF. Neste protocolo somente uma instância de cache line pode estar no estado de Forwarding, assim como somente ela pode ser duplicada. Outros caches podem conter os dados, mas eles estarão no estado Shared e não poderão ser copiados. Em outras palavras, o cache line no estado Forwarding é utilizado para responder à requisições de leitura. Fazendo com que um único cache line responda às requisições, o tráfego de coerência é reduzido drasticamente quando existem várias dos dados [Qian e Yan 2008].

Segundo Qian e Yan [2008], quando um cache line no estado Forwarding é copiado, a nova cópia fica no estado Forwarding enquanto a versão anterior volta para o estado Shared. A Tabela 1 mostra as características do protocolo MESIF.

	Clean /Dirty	**Unique?**	**Can Write?**	**Can Forward?**	**Comments**
Modified	Dirty	Yes	Yes	Yes	Must write back To share or Replace
Exclusive	Clean	Yes	Yes	Yes	Transition to M on write
Shared	Clean	No	No	No	Does not forward
Invalid	NA	NA	NA	NA	Can not read
Forwarding	Clean	Yes	No	Yes	Must invalid other copies to write

Tabela 1. Protocolo MESIF [Qian e Yan 2008]

2.9. Fluxo de Instruções

A microarquitetura Nehalem implementa várias técnicas para processar de forma eficiente a stream do ISA CISC no código do usuário. O núcleo consiste em um grande número de unidades funcionais (UF), cada uma sendo capaz de executar uma micro-operação (instrução do tipo RISC) [Thomadakis 2011].

As macro-instruções do tipo CISC são traduzidas nos estágios iniciais do núcleo para uma ou mais micro-operações. As micro-operações atingem as unidades de execução, são despachadas para as unidades funcionais e depois são "retiradas", tendo seu resultado salvo em algum registrador visível ou na memória. Quando todas as micro-operações das macro-instruções terminam, a macro-instrução também termina (é "retirada"), ficando assim claro que o objetivo do processador é maximar a taxa de "retirada" das macro-instruções [Thomadakis 2011].

Segundo Thomadakis [2011], a principal forma com que o Nehalem faz para maximizar esta taxa é permitindo que as micro-operações possuam o máximo de instruções possível e procedam em paralelo com outras micro-operações, ocupando diferente unidades funcionais a cada ciclo de clock. O fluxo de instrução do Intel64 pode ser resumido da seguinte forma:

- São buscadas várias macro-instruções (em bloco de cache, por exemplo).
- Essas são transformadas (pré-decodificadas) em sequências de micro-operações.
- As micro-operações são colocadas em buffers próprios para serem tratadas pelas unidades funcionais.
- Elas são então despachadas para as unidades funcionais disponíveis logo que os operandos de que necessitam estão disponíveis.
- Por fim, as micro-operações são retiradas e seus resultados são guardados.

Todo este processo é executados em estágios da pipeline, mais precisamente em 14 estágios. Para que se tenha sempre a pipeline preenchida, a microarquitetura Nehalem permite que as micro-operações continuem a serem executadas fora de ordem. Para tanto, ela utiliza o escalonamento dinâmico de instruções (para decidir quais micro-operações podem prosseguir mantendo a semântica do programa) e a busca especulativa de instruções (carrega as instruções do programa para além dos saltos condicionais, dispondo então, também, de uma unidade de previsão de desvios que tenta antecipar de forma mais precisa possível o resultado dos desvios condicionais de um programa) [Thomadakis 2011].

Uma vez executadas, as micro-operações são retiradas e colocadas em um buffer de reordenação e/ou colocam seus resultados em outras micro-operações que estão à espera destes resultados. Quando terminam de ser reordenadas essas micro-operações são retiradas do buffer e são guardadas de acordo com o programa. Todo este processo tem como principal objetivo aumentar a taxa de retirada de micro-operações por ciclo de clock [Thomadakis 2011, SabercomLógica 2012].

2.9.1. Pipeline de Busca e Descodificação Ordenada e Fora de Ordem

O funcionamento da pipeline de busca de instruções pode ser visto na Figura 8. Nela a execução ordenada começa pela Unidade de Busca de Instruções (1), que faz a busca na cache L1 e carrega sempre blocos de 16 bytes de instruções alinhadas. Esta unidade faz a busca de instruções especulativa, isto é, antecipando o carregamento de instruções sem ainda ter definido o caminho do programa. Para que a busca de instruções especulativa resulte em instruções no caminho correto do programa, esta unidade recebe o apoio da Unidade de Previsão de Desvios (2) [Thomadakis 2011, SabercomLógica 2012].

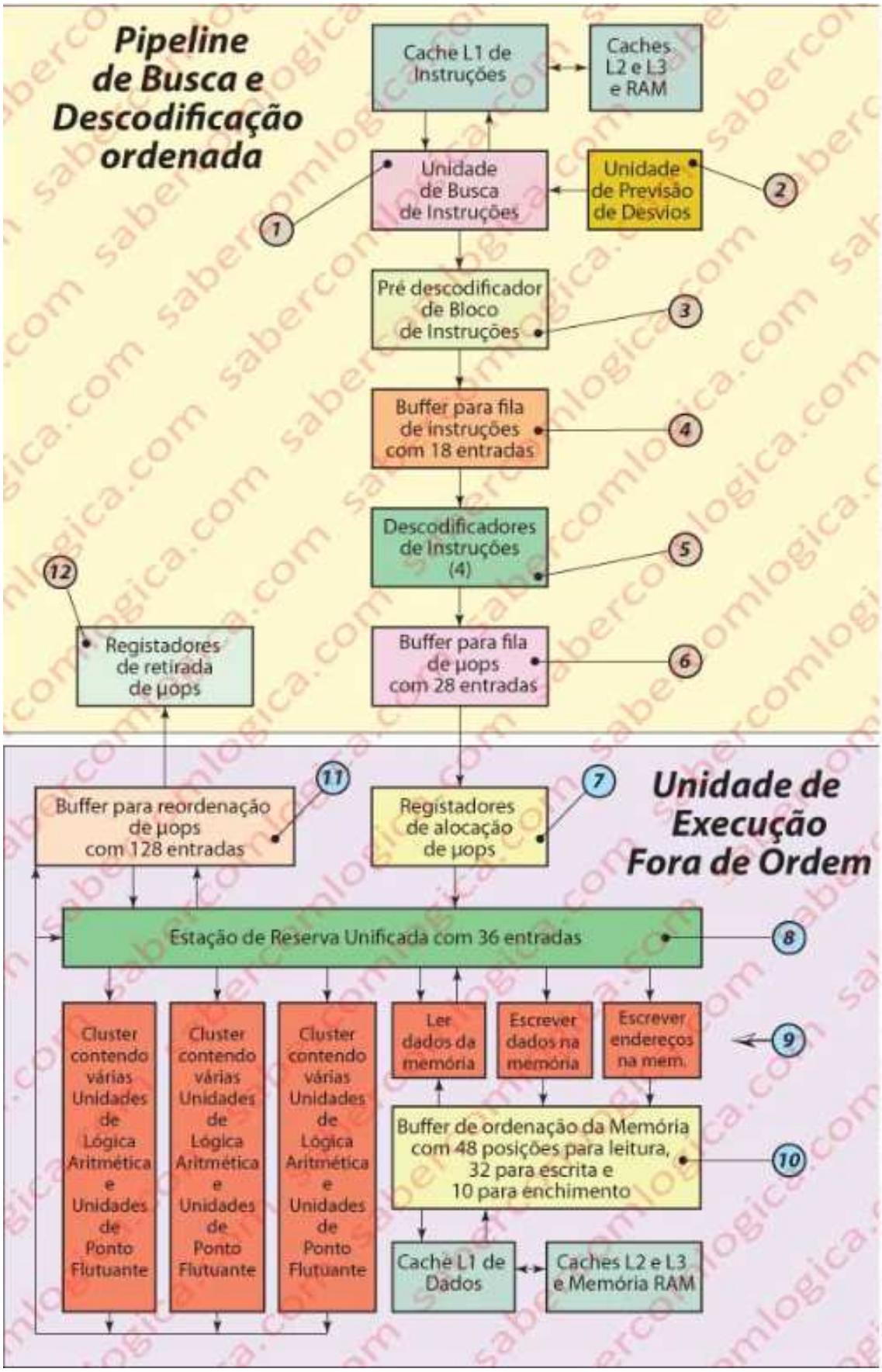


Figura 8. Pipeline de Busca e Descodificação Ordenada [SabercomLógica 2012]

O bloco de 16 bytes de instruções alinhadas obtido é enviado para um Pré-decodificador de Bloco de Instruções (3), que identifica e separa as instruções individuais do bloco e as coloca em um Buffer de Fila de Instrução (4). Este dispõe de 18 entradas, onde as instruções aguardam na sua ordem inicial de leitura para serem recebidas pela Unidade de Descodificação (5), que tem quatro decodificadores e é capaz de enviar para o Buffer de Fila de Micro-Operações (6) até quatro micro-operações por cada ciclo de clock [Thomadakis 2011, SabercomLógica 2012].

As micro-operação que foram obtidas estão ordenadas e prontas para serem executadas. Porém se tivermos de esperar todos os elementos necessários para a sua execução na sua sequência natural, sendo esta uma pipeline de 14 estágios, certamente teremos bolhas e entraremos frequentemente em stall [Thomadakis 2011, SabercomLógica 2012].

Como solução para esta questão, a microarquitetura Nehalem utiliza a execução de micro-operações fora de ordem, ou seja, por antecipação. Várias micro-operações de fases mais avançadas, que não necessitam de elementos de micro-operações anteriores, são executadas primeiro, e seus resultados são armazenadas em um buffer à espera das micro-operações que vão necessitar, conseguindo assim preencher a pipeline [Thomadakis 2011, SabercomLógica 2012].

Assim, e ainda segundo a Figura 8, as micro-operações que ficaram no Buffer de Fila de Micro-Operações (6) vão ser agora selecionadas para a Unidade de Execução pelos Registradores de Alocação (7), que desempenha tarefas como alocar recursos para cada micro-operação, ligar ao canal de despacho apropriado, preparar a arquitetura interna da CPU para trabalhar os resultados intermédios e providenciar os operandos que cada micro-operação necessita. [Thomadakis 2011, SabercomLógica 2012]

Após isso, as micro-operações passam para a Estação de Reserva Unificada (8), que pode conter até 36 micro-operações aguardando por seus operandos para serem despachadas através de seis canais para seis diferente Unidades de Execução (9), conseguindo assim despachar seis micro-operações por ciclo de clock. Três canais e suas respectivas Unidades de Execução são destinados à leitura e escrita de dados e endereços na memória, e os outros três são compostos por Unidades de Lógica e Aritmética (ULA) em quantidade variável. [Thomadakis 2011, SabercomLógica 2012]

Uma Unidade de Execução pode contar várias Unidades Funcionais. Estas, principalmente as mais complexas, podem funcionar em pipeline, permitindo assim que contenham várias micro-operações em simultâneo e produzam até um resultado por ciclo de clock. As Unidades Funcionais de acesso à memória o fazem através do Buffer de Ordenação da Memória (10), que apoiada nos buffers de retenção garante a consistência das leituras e escritas em memória. [Thomadakis 2011, SabercomLógica 2012]

Os resultados das micro-operações processadas pelas Unidades de Execução podem ser enviados de volta para a Unidade de Reserva Unificada, como operandos para micro-operações que se encontram em espera, ou para o Buffer de Reordenação (11), cuja função, essencial para a execução fora de ordem, consiste na retenção das micro-operações e sua liberação exclusivamente na ordem sequencial da arquitetura das macro-instruções do programa em execução, enviando-as assim tratadas para os Registradores de Retirada de micro-operações (12), já funcionando pela ordem do programa. [Thomadakis 2011, SabercomLógica 2012]

2.10. Uso Pretendido e Atual

Segundo Grover e Agrawal [2014], os processadores Intel Core i7 são recomendados principalmente para:

- Multitarefas, como rodar diversas aplicações ao mesmo tempo.
- Aplicações em multithreading.
- Gaming.
- Criação e edição gráfica.

Eles são encontrados atualmente em computadores desktop, notebooks, e são muito presentes no dia a dia das pessoas, e continuam sendo processadores do topo de linha da Intel.

3. Comparação entre o Intel Core i7-950 e AMD Phenom II X6 1090T

3.1. Intel Core i7-950

A versão escolhida do Core i7 para fazer o benchmark foi o Intel Core i7-950, com as características mostradas na Tabela 2

Intel Processor	Intel Core i7-965 Processor Extreme Edition
Date Introduced	2008
Micro-architecture	Intel micro-architecture code name Nehalem; Quadcore; HyperThreading Technology; Intel QPI; Intel 64 Architecture; Intel Virtualization Technology.
Highest Processor Base Frequency at Introduction	3.20 Ghz
Transistors	731 M
Register Sizes	GP: 32,64 FPU: 80 MMX: 64 XMM: 128
SystemBus/QPI Link Speed	QPI: 6.4 GT/s Memory: 25 GB/s
Max External Address Space	64 GB
On-Die Caches	L1: 64 KB L2: 256 KB L3: 8 MB

Tabela 2. Intel Core i7-950 [Intel 2016a]

De acordo com o site UserBenchmark [2019a], o benchmarking geral do Intel Core i7, a partir de 23155 amostras pode ser visto na Figura 9. Nela, podemos ver que a média do desempenho de todos os benchamarkings realizados com este processador está em

48.5%, sendo classificado com um desempenho regular visto que ficou na região amarela. Vemos abaixo do gráfico na imagem que o benchmark indicou que este processador é cerca de 48% apto para ser usado em gaming, 51% para ser usado como desktop, ou seja, em aplicações comuns do dia a dia, e, apenas 33% para ser usado como workstation.

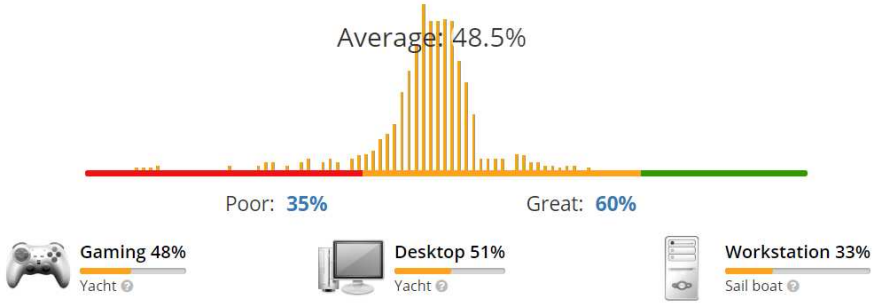


Figura 9. Benchamark do site UserBenchmark [2019a]

3.2. Intel Core i7-950 vs AMD Phenom II X6 1090T

No primeiro momento será feita uma comparação entre os modelos Intel Core i7-950 e AMD Phenom II X6 1090T, que foram lançados basicamente no mesmo ano, por volta de 2008 a 2010. Novamente o benchmark é baseado no site [UserBenchmark 2019b].

Pode ser observado na Tabela 3 que mesmo com o processador da AMD tendo dois cores a mais que o processador da Intel, este ultimo leva vantagem, como pode ser visto na Figura 10, onde o processador da Intel tem uma média de benchmarking que fica na região de desempenho regular, enquanto que o processador da AMD fica muito próximo da região de desempenho ruim. Além disso, pode ser notado que em todos os aspectos abaixo do gráfico da imagem o processador da Intel tem resultados melhores.

Esse resultado se deve principalmente pela suas tecnologias inovadoras para a época, como a execução fora de ordem e Hyper-Threading, que são seu pontos diferenciais. Apesar da Intel também trazer a tecnologia de Turbo Boost, a AMD competiu com a tecnologia do Turbo CORE.

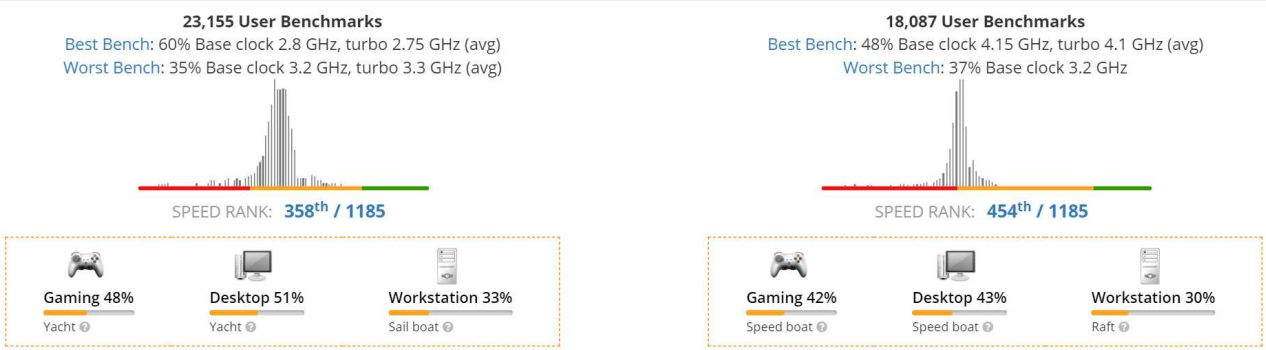


Figura 10. Benchmarking Intel Core i7-950 vs AMD Phenom II X6 1090T do site UserBenchmark [2019b]

Intel Core i7-950	*AMD Phenom II X6 1090T*
Microarquitetura Nehalem	Microarquitetura K10
45nm	45nm
Quad-core	Hexa-core
Turbo Boost	Turbo CORE
64 KB de cache L1, 256 KB de cache L2 e 8 MB de cache L3	Seis caches L2 com 512 KB, e um cache L3 com 6 MB

Tabela 3. Comparação Intel Core i7-950 vs AMD Phenom II X6 1090T [Cruz 2016]

4. Conclusão

Como foi observado, a primeira geração do Intel Core i7 trouxe diversas inovações para a época em que foi lançada, no ano de 2008. Exemplos que demonstram este ponto estão nas tecnologias que traz consigo, como o Turbo Boost, Hyper-Threading, controlador de memória integrado, Quick Path Interconnect, entre outros pontos que contribuíram para que o processador se destacasse.

Referências

Almeida, R. B. (2009). A evolução dos processadores. comparação das famílias dos processadores intel e amd. 2009. instituto de computação unicamp. http://www.ic.unicamp.br/∼ducatte/mo401/1s2009/T2/089065-t2.pdf. Acesso: 03 Novembro 2019.

Bruschi, S. M. (2017). Ssc0611 arquitetura de computadores, aula 15 - evolução arquitetura intel - parte 2. icmc-usp. https://edisciplinas.usp.br/pluginfile.php/3418115/mod_resource/content/16/15aula%20-%20Evolucao%20Arq%20Intel%20-%20parte%202.pdf. Acesso: 03 Novembro 2019.

Cruz, S. I. O. (2016). Comparación de la arquitectura de microprocesadores intel y amd. 2016. departamento de sistemas y computación, instituto tecnológico de tijuana. https://www.academia.edu/22942882/Comparación_de_la_arquitectura_de_microprocesadores_Intel_y_AMD. Acesso: 03 Novembro 2019.

Drake, D. e Berg, J. (2021). Unaligned memory accesses. https://www.kernel.org/doc/html/latest/core-api/unaligned-memory-access.html?highlight=unaligned. Acesso: 14 Janeiro 2021.

Grover, H. e Agrawal, D. (2014). Design and architecture of intel's core i7 processor. 2014. dronacharya college of engineering, farrukhnagar, gurgaon, india. https://www.ijraset.com/fileserve.php?FID=1034. Acesso: 03 Novembro 2019.

Intel (2016a). Intel64 and ia-32 architectures software developer's manual, volume 1: Basic architecture. https://www.intel.com.br/content/www/br/pt/architecture-and-technology/64-ia-32-architectures-software-developer-vol-1-manual.html. Acesso: 03 Novembro 2019.

Intel (2016b). Intel64 and ia-32 architectures software developer's manual, volume 2: Instruction set reference, a-z. https://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-instruction-set-reference-manual-325383.pdf. Acesso: 03 Novembro 2019.

Qian, B. e Yan, L. (2008). The research of the inclusive cache used in multi-core processor. *2008 International Conference on Electronic Packaging Technology & High Density Packaging*, páginas 1–4.

SabercomLógica (2012). Caso de estudo - intel i7 nehalem. http://sabercomlogica.com/pt/caso-de-estudo-intel-i7-nehalem/. Acesso: 03 Novembro 2019.

Teles, B. (2009). O processador intel core i7. 2009. instituto de computação universidade estadual de campinas. http://www.ic.unicamp.br/∼ducatte/mo401/1s2009/T2/042348-t2.pdf. Acesso: 03 Novembro 2019.

Thomadakis, M. E. (2011). The architecture of the nehalem processor and nehalem-ep smp platforms. *Resource*, 3(2):30–32.

UserBenchmark (2019a). Benchmark intel core i7-950. https://cpu.userbenchmark.com/Intel-Core-i7-950/Rating/617. Acesso: 03 Novembro 2019.

UserBenchmark (2019b). Benchmark intel core i7-950 vs amd phenom ii x6 1090t. https://cpu.userbenchmark.com/Compare/Intel-Core-i7-950-vs-AMD-Phenom-II-X6-1090T/617vsm417. Acesso: 03 Novembro 2019.