

SSC0114

Arquitetura de Computadores

27ª Aula – Arquitetura ARM

Profa. Sarita Mazzini Bruschi

sarita@icmc.usp.br

Parte dos slides foram preparados pela estagiária PEEG Laíse Aquino Cardoso e Silva
Graduanda em Engenharia da Computação

Arquitetura ARM

- ARM (**Advanced RISC Machine**) é uma família de arquiteturas que originou o uso da tecnologia RISC em aplicações comerciais
- Microprocessadores ARM visam a **simplificação** das instruções, buscando atingir a máxima eficiência por ciclo
- Possuem alta **flexibilidade** e capacidade de **customização**
- Atualmente são bastante utilizados em **sistemas embarcados**, principalmente **telefones celulares**

Arquitetura ARM

- Por que são ideais para aplicações embarcadas?
 - **Tamanho:** a estrutura de um processador ARM é simples e enxuta, o que diminui o espaço físico ocupado pelo chip
 - **Consumo de potência:** esta arquitetura opera com menos potência do que processadores x86 convencionais
 - **Custo:** uma área reduzida e conjunto de instruções simples - que facilita seu aprendizado, otimização e compilação - resultam na redução do seu custo

Arquitetura ARM

- Histórico
- Características
- Principais Componentes
- Conjunto de instruções

Arquitetura ARM

- Desenvolvimento iniciado em **1980** como parte de um projeto da BBC, que envolvia a criação de uma nova arquitetura para um microcomputador destinado ao ensino de computação
- Empresa contratada:
Acorn Digital Computers



Microcomputador BBC Micro

Histórico

- O conceito de arquitetura **RISC**, que estava se consolidando na época, influenciou fortemente o projeto. Por isso, o nome adotado para o microprocessador desenvolvido foi **Acorn RISC Machine (ARM)**
- Sua primeira versão comercial (ARM-1) foi lançada em **1985** e possuía cerca de 25 mil transistores



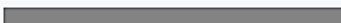














Histórico

- Como os modelos de microprocessadores ARM apresentavam alta velocidade, tamanho pequeno do circuito integrado e baixo consumo, sua aplicação cresceu na área de **sistemas embarcados**
- O sucesso dos projetos da Acorn e o crescente interesse pela tecnologia RISC chamou a atenção da **Apple Computers**. Formou-se uma parceria entre Acorn, Apple e VLSI, originando a companhia **Advanced RISC Machines Ltd** em **1990**

Histórico

- Com o desenvolvimento de novas versões do ARM e a revolução dos dispositivos móveis, essa tecnologia foi amplamente utilizada e se tornou padrão para uso em celulares

Sales of chips containing Arm cores^{[131][132][133]}

Year	Billion units	Relative size
2015	15	
2014	12	
2013	10	
2012	8.7	
2011	7.9	
2010	6.1	
2009	3.9	
2008	4.0	
2007	2.9	
2006	2.4	
2005	1.662	
2004	1.272	
2003	0.782	
2002	0.456	
2001	0.420	
2000	0.367	
1999	0.175	
1998	0.051	
1997	0.009	
Total	78.094	

Crédito da Imagem - [Wikipedia](#)

Histórico

- Apesar da grande quantidade de dispositivos que possuem um processador ARM, a ARM Holding (nome alterado em 1998) não produz nenhum desses chips
 - Diferente de todas as fabricantes de chips, a ARM trabalha usando um modelo de licenciamento baseado em direitos de propriedade intelectual:
 - *Core licence*: venda de IP cores (bloco de lógica ou dados que é usado para produzir um FPGA ou um SOC)
 - *Architectural licence*: baseada nas necessidades dos clientes, as empresas podem obter uma licença arquitetural para projetar a própria CPU baseada no conjunto de instruções ARM

Arquitetura ARM

- Histórico
- Características
- Principais Componentes
- Conjunto de instruções

Características

- Os processadores ARM são projetos para atender às necessidades de 3 categorias:
 - Sistemas embarcados de tempo real (R): sistemas para aplicações de armazenamento, automotivas, industriais e de rede
 - Plataformas de aplicação (A): dispositivos executando sistemas operacionais abertos
 - Microcontroladores (M): smart cards, placas SIM e terminais de pagamento

Características

Tabela 2.8 Evolução da ARM

Família	Recursos notáveis	Cache	MIPS típico @ MHz
ARM1	RISC 32 bits	Nenhuma	
ARM2	Instruções de multiplicação e swap; unidade de gerenciamento de memória integrada, processador gráfico e de E/S	Nenhuma	7 MIPS @ 12 MHz
ARM3	Primeira a usar cache de processador	4 KB unificada	12 MIPS @ 25 MHz
ARM6	Primeira a aceitar endereços de 32 bits: unidade de ponto flutuante	4 KB unificada	28 MIPS @ 33 MHz
ARM7	SoC integrado	8 KB unificada	60 MIPS @ 60 MHz
ARM8	Pipeline de 5 estágios; previsão estática de desvio	8 KB unificada	84 MIPS @ 72 MHz
ARM9		16 KB/16 KB	300 MIPS @ 300 MHz
ARM9E	Instruções DSP melhoradas	16 KB/16 KB	220 MIPS @ 200 MHz
ARM10E	Pipeline de 6 estágios	32 KB/32 KB	
ARM11	Pipeline de 9 estágios	Variável	740 MIPS @ 665 MHz
Cortex	Pipeline superescalar de 13 estágios	Variável	2 000 MIPS @ 1 GHz
XScale	Processador de aplicações; pipeline de 7 estágios	32 KB/32 KB L1 512KB L2	1 000 MIPS @ 1,25 GHz

DSP = processador de sinal digital (do inglês *digital signal processor*)

SoC = sistema em um chip (do inglês *system on a chip*)

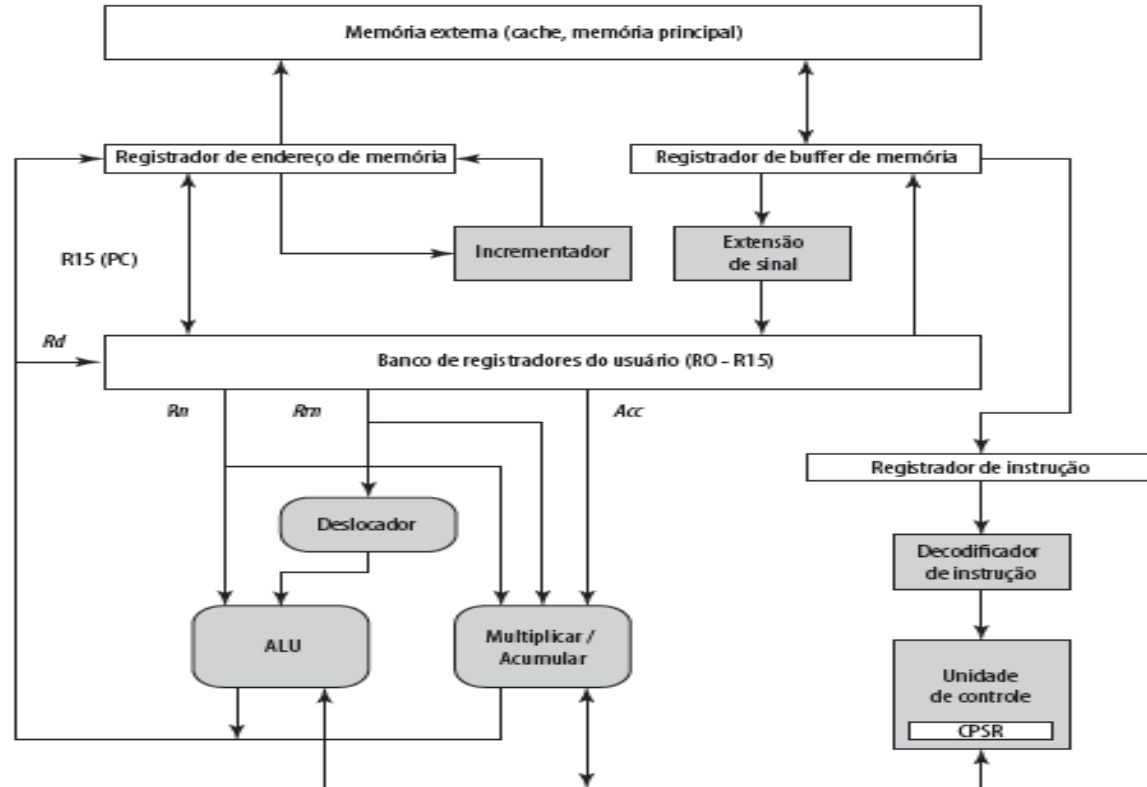
Características

- Dois conjuntos de instruções de tamanho fixo: ARM (32 bits) e Thumb (16 bits)
- 16 registradores de 32 bits
- Arquitetura Load-Store: as instruções somente operam sobre valores que já estejam nos registradores (ou imediatos) e sempre armazenam resultados em algum registrador. O acesso à memória é feito apenas através das instruções load e store
- Dispositivos de E/S são mapeados em memória
- Suporte a interrupções e execução condicional de instruções

Arquitetura ARM

- Histórico
- Características
- Principais Componentes
- Conjunto de instruções

Arquitetura ARM



CPSR = registrador de estado de programa corrente (do inglês *current program status register*)

Principais Componentes

- **Unidade de decodificação de instruções e controle lógico:** decodifica as instruções ARM e Thumb e organiza a sequência de exceções e outros eventos irregulares
- **Registrador de endereço de memória:** associado ao incrementador de endereço, mantém o controle da posição de PC
- **Registradores de memória de dados:** controlam o fluxo de entrada e saída
- **ULA (Unidade Lógica Aritmética):** Realiza as operações lógicas e aritméticas requisitadas

Principais Componentes

- **Banco de Registradores**
 - 1 porta de leitura
 - 2 portas de escrita
 - Portas de leitura e escrita do PC
- **Barrel Shifter**: realiza o deslocamento de uma palavra de dados em uma quantidade específica de bits

Modelos

Architecture ↕	Core bit-width ↕	Cores		Profile ↕	References ↕
		ARM Holdings ↕	Third-party ↕		
ARMv1	32 ^[a 1]	ARM1		Classic	
ARMv2	32 ^[a 1]	ARM2, ARM250, ARM3	Amber, STORM Open Soft Core ^[39]	Classic	
ARMv3	32 ^[a 2]	ARM6, ARM7		Classic	
ARMv4	32 ^[a 2]	ARM8	StrongARM, FA526, ZAP Open Source Processor Core ^[40]	Classic	
ARMv4T	32 ^[a 2]	ARM7TDMI, ARM9TDMI, SecurCore SC100		Classic	
ARMv5TE	32	ARM7EJ, ARM9E, ARM10E	XScale, FA626TE, Feroceon, PJ1/Mohawk	Classic	
ARMv6	32	ARM11		Classic	
ARMv6-M	32	ARM Cortex-M0, ARM Cortex-M0+, ARM Cortex-M1, SecurCore SC000		Microcontroller	
ARMv7-M	32	ARM Cortex-M3, SecurCore SC300		Microcontroller	
ARMv7E-M	32	ARM Cortex-M4, ARM Cortex-M7		Microcontroller	
ARMv8-M	32	ARM Cortex-M23, ^[41] ARM Cortex-M33 ^[42]		Microcontroller	^[43]
ARMv7-R	32	ARM Cortex-R4, ARM Cortex-R5, ARM Cortex-R7, ARM Cortex-R8		Real-time	
ARMv8-R	32	ARM Cortex-R52		Real-time	^{[44][45][46]}
ARMv7-A	32	ARM Cortex-A5, ARM Cortex-A7, ARM Cortex-A8, ARM Cortex-A9, ARM Cortex-A12, ARM Cortex-A15, ARM Cortex-A17	Qualcomm Krait, Scorpion, PJ4/Sheeva, Apple Swift	Application	
ARMv8-A	32	ARM Cortex-A32		Application	
ARMv8-A	64/32	ARM Cortex-A35, ^[47] ARM Cortex-A53, ARM Cortex-A57, ^[48] ARM Cortex-A72, ^[49] ARM Cortex-A73 ^[50]	X-Gen, Nvidia Project Denver, Cavium Thunder X ^{[51][52][53]} , AMD K12, Apple Cyclone/Typhoon/Twister/Hurricane/Zephyr, Qualcomm Kryo, Samsung M1 and M2 ("Mongoose") ^[54]	Application	^{[55][56]}
ARMv8.1-A	64/32	TBA		Application	
ARMv8.2-A	64/32	ARM Cortex-A55, ^[57] ARM Cortex-A75, ^[58] ARM Cortex-A76 ^[59]		Application	^[60]
ARMv8.3-A	64/32	TBA	Apple A12 Bionic	Application	
ARMv8.4-A	64/32	TBA		Application	

1. ^{a b} Although most datapaths and CPU registers in the early ARM processors were 32-bit, addressable memory was limited to 26 bits; with upper bits, then, used for status flags in the program counter register.

2. ^{a b c} ARMv3 included a compatibility mode to support the 26-bit addresses of earlier versions of the architecture. This compatibility mode optional in ARMv4, and removed entirely in ARMv5.

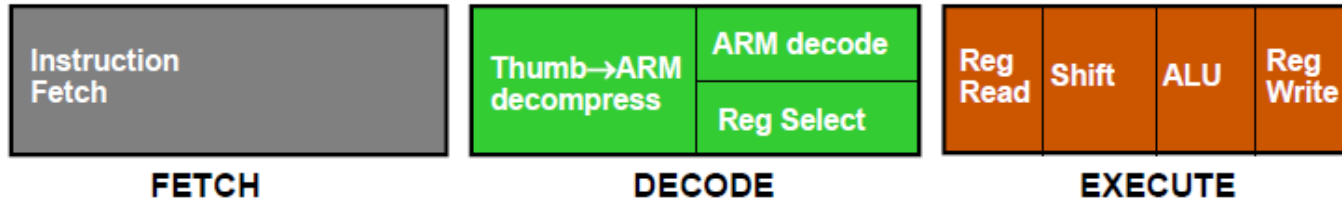
ARM Holdings provides a list of vendors who implement ARM cores in their design (application specific standard products (ASSP), microprocessor and microcontrollers).^[61]

Arquitetura ARM

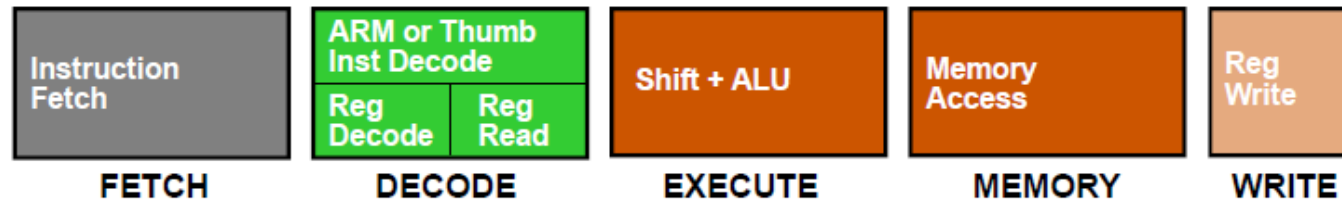
- A arquitetura ARM7 e seus antecessores possuem um **pipeline** de três estágios: *Fetch, Decode e Execute*
 - **Fetch**: a instrução é trazida da memória e colocada no pipeline
 - **Decode**: os registradores usados na instrução são decodificados
 - **Execute**: o banco de registradores é lido, as operações lógico aritméticas são executadas sobre os operandos, o resultado da operação é gerado e escrito no registrador de destino

Arquitetura ARM

ARM7TDMI

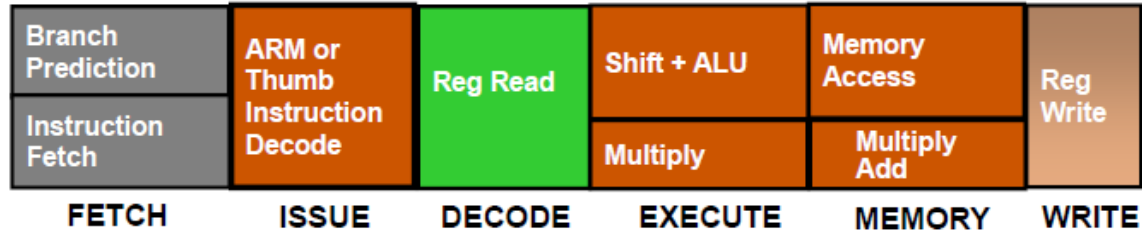


ARM9TDMI

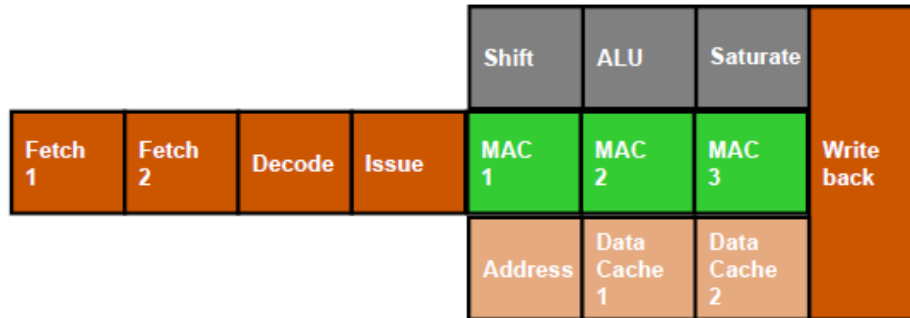


Arquitetura ARM

ARM10

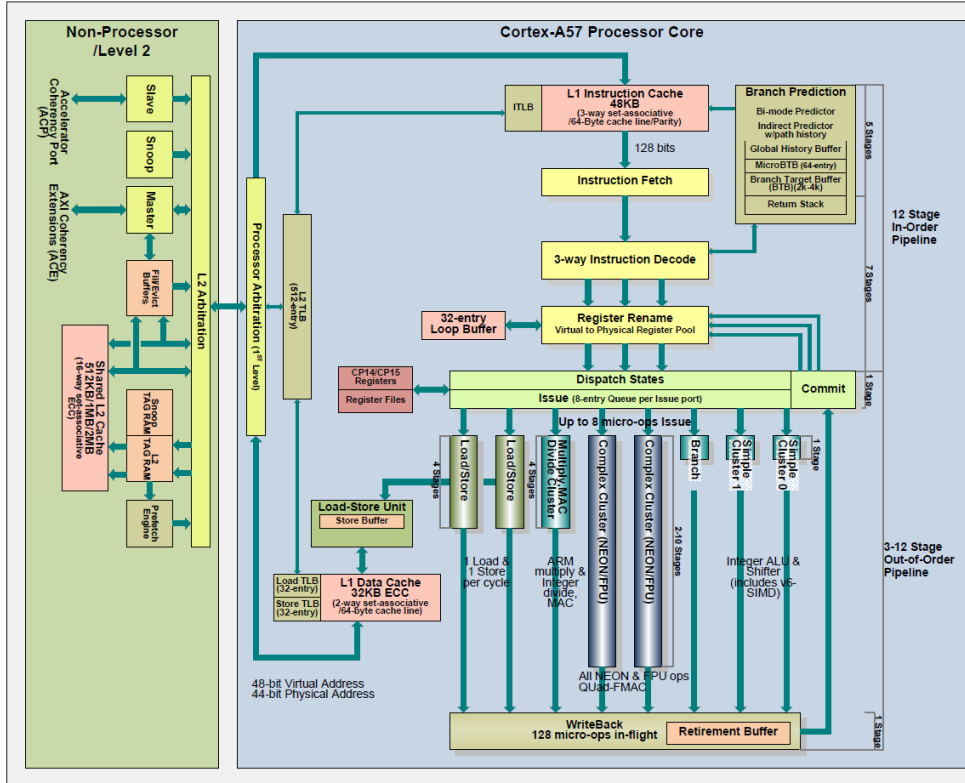


ARM11

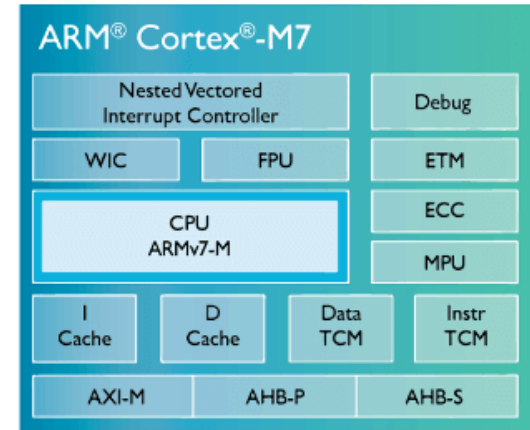
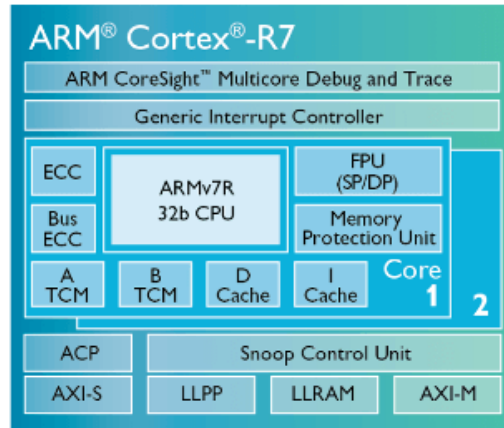
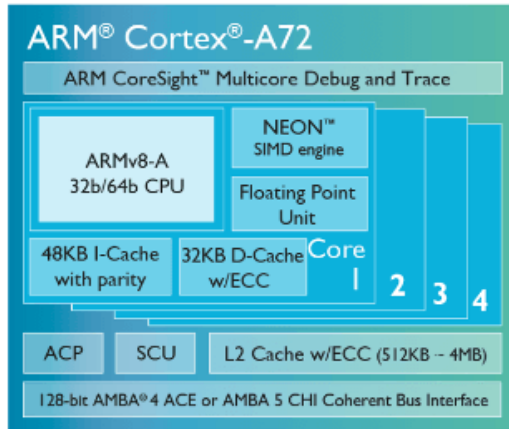


Arquitetura ARM

ARM Cortex-A57 Block Diagram



Arquitetura ARM



Arquitetura ARM

- 7 modos de operação:
 - Modo usuário: execução da maioria das aplicações
 - Modo supervisor: modo em que se executa o SO
 - Modo de abortamento: ativado quando se tem falha de memória
 - Modo indefinido: ativado quando o processador tenta executar uma instrução que não é suportada nem pelo núcleo nem pelos coprocessadores
 - Modo de interrupção rápida (FIQ – *fast interrupt*): ativada quando o processador recebe um sinal de interrupção a partir de uma fonte designada de interrupção rápida
 - Modo de interrupção (IRQ – *interrupt request*): ativado sempre que o processador recebe um sinal de interrupção
 - Modo de sistema: usado para executar certas tarefas privilegiadas do SO

Arquitetura ARM

- Histórico
- Características
- Principais Componentes
- Conjunto de instruções

Conjunto de instruções

- A arquitetura ARM possui dois conjuntos de instruções: **ARM** e **Thumb**
- O conjunto Thumb consiste em um subconjunto das instruções ARM, porém comprimidas de 32 bits para 16 bits. Essa medida reduz a quantidade de **memória** utilizada pelo código, além de diminuir o **tamanho** do hardware final e consequentemente seu custo
- Toda instrução Thumb possui uma instrução ARM equivalente, porém o contrário não é válido
- É possível **alternar** entre os estados Thumb ou ARM do processador por meio de instruções específicas SUB_BRANCH e SUB_RETURN, respectivamente

Conjunto de Instruções

- A arquitetura ARM possui 37 registradores no total
 - 1 dedicado para o contador de programa
 - 1 dedicado para o registrador de estado do programa corrente (cprs)
 - 5 dedicados para os registradores de estado do programa salvos (sprs)
 - 30 registradores de propósito geral
- Os registradores são arrançados em vários bancos, com o banco acessível dependendo do modo de operação. Cada modo pode acessar:
 - Um conjunto particular de registradores de propósito geral (r0 – r12)
 - Um particular r13 (ponteiro para a pilha) e r14 (registrador de ligação)
 - r15 (contador de programa)
 - cpsr

Conjunto de Instruções

Current Visible Registers

Abort Mode

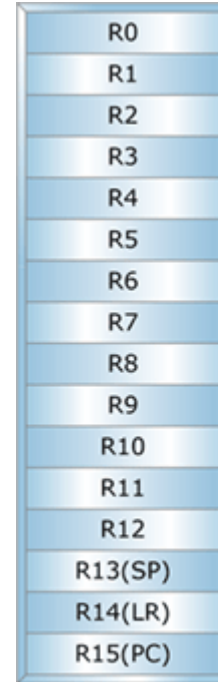
r0
r1
r2
r3
r4
r5
r6
r7
r8
r9
r10
r11
r12
r13 (sp)
r14 (lr)
r15 (pc)
cpsr
spsr

Banked out Registers

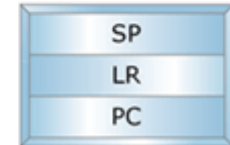
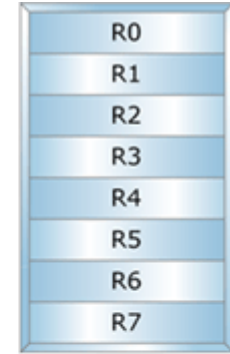
User	FIQ	IRQ	SVC	Undef
	r8			
	r9			
	r10			
	r11			
	r12			
r13 (sp)	r13 (sp)	r13 (sp)	r13 (sp)	r13 (sp)
r14 (lr)	r14 (lr)	r14 (lr)	r14 (lr)	r14 (lr)
	spsr	spsr	spsr	spsr

Conjunto de instruções

- Registradores acessíveis ao programador nos dois estados de operação
 - R0 - R12: uso geral
 - R13: stack pointer
 - R14: link
 - R15: PC

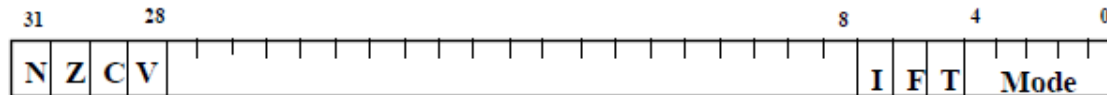


ARM



Thumb

Conjunto de instruções



- * **Condition Code Flags**
 - N = **N**egative result from ALU flag.
 - Z = **Z**ero result from ALU flag.
 - C = ALU operation **C**arried out
 - V = ALU operation **o**Verflowed
- * **Mode Bits**

M[4:0] define the processor mode.
- * **Interrupt Disable bits.**
 - I** = 1, disables the IRQ.
 - F** = 1, disables the FIQ.
- * **T Bit (Architecture v4T only)**
 - T = 0, Processor in ARM state
 - T = 1, Processor in Thumb state

Conjunto de instruções

- Tipos de instruções:
 - Processamento de dados (operações lógico-aritméticas e movimentação)
 - Fluxo de controle (instruções de desvio)
 - Carregamento/armazenamento (load e store)

Conjunto de instruções

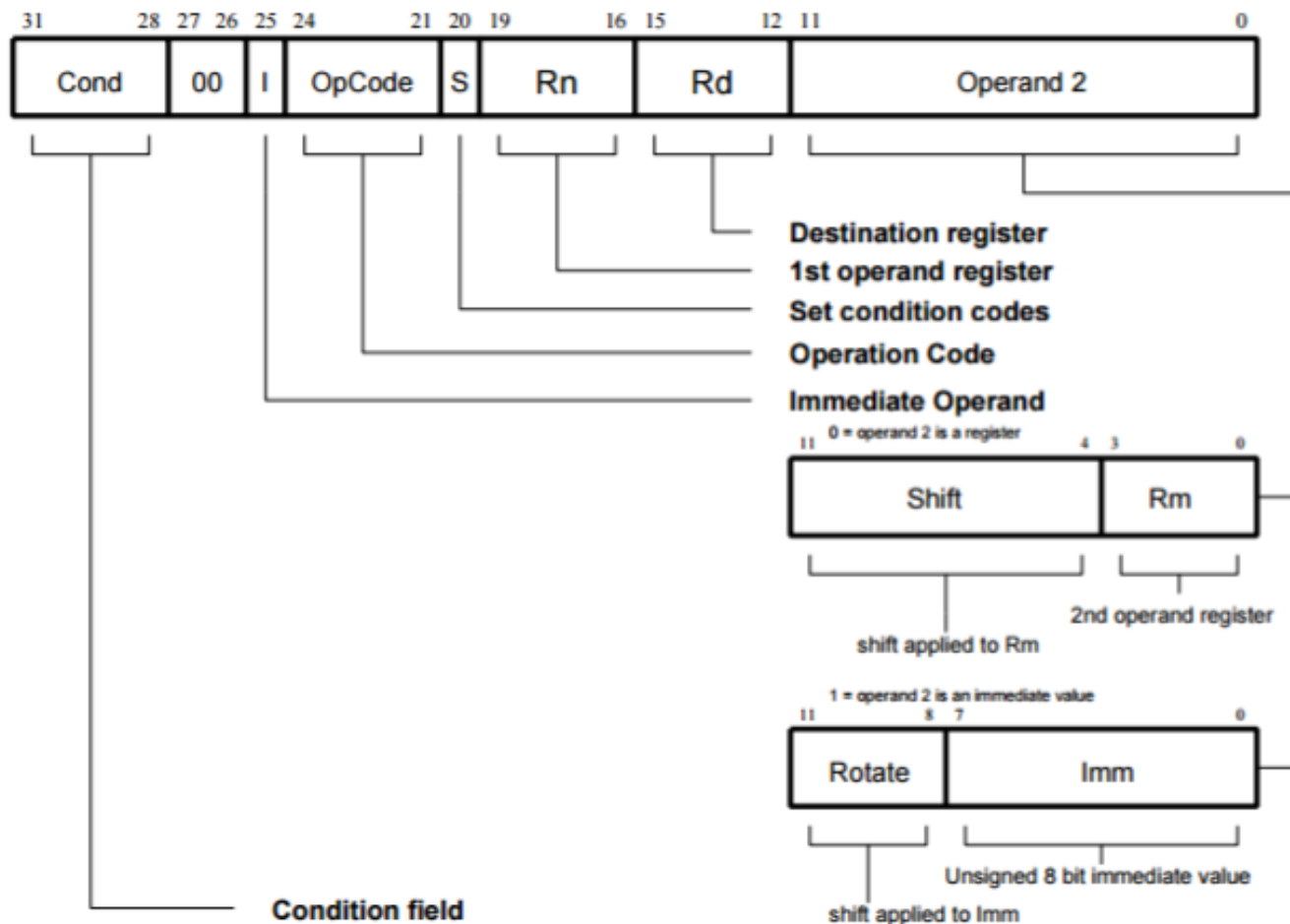
Processamento de Dados

- As instruções de processamento de dados são constituídas por até três endereços: um ou dois registradores como operandos (ou valores imediatos) e outro para armazenar o resultado
- Valores imediatos são precedidos pelo símbolo #
- Exemplos de instruções:

ADD R1, R2, #4 $R1 \leftarrow (R2 + 4)$

MOV R1, #0 $R1 \leftarrow 0$

Assembler Mnemonic	OpCode	Action
AND	0000	operand1 AND operand2
EOR	0001	operand1 EOR operand2
SUB	0010	operand1 - operand2
RSB	0011	operand2 - operand1
ADD	0100	operand1 + operand2
ADC	0101	operand1 + operand2 + carry
SBC	0110	operand1 - operand2 + carry - 1
RSC	0111	operand2 - operand1 + carry - 1
TST	1000	as AND, but result is not written
TEQ	1001	as EOR, but result is not written
CMP	1010	as SUB, but result is not written
CMN	1011	as ADD, but result is not written
ORR	1100	operand1 OR operand2
MOV	1101	operand2 (operand1 is ignored)
BIC	1110	operand1 AND NOT operand2 (Bit clear)
MVN	1111	NOT operand2 (operand1 is ignored)

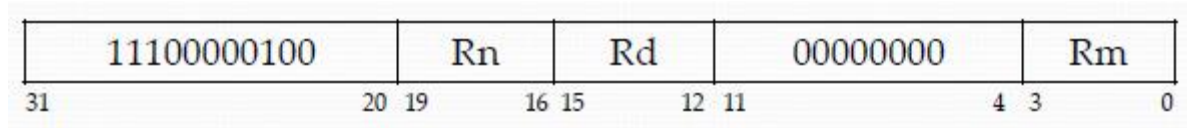


Conjunto de instruções

Processamento de Dados

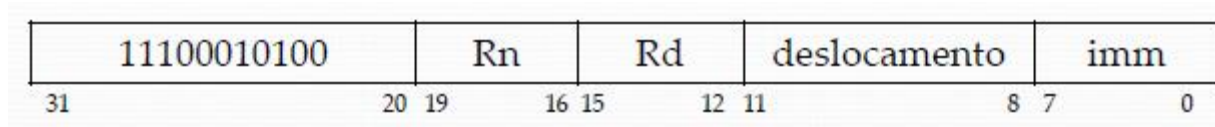
- Endereçamento:
 - Operação com registradores

Ex: ADD Rd, Rn, Rm



- Operação com valores imediatos

Ex: ADD Rd, Rn, #imm



Conjunto de instruções

Fluxo de Controle

- As instruções de fluxo de controle são divididas em dois tipos:

- Desvio incondicional (ocorre de qualquer forma)

B <rótulo>

- Desvio condicional (ocorre de acordo com uma condição *cond*)

B*cond* <rótulo>

Ex:

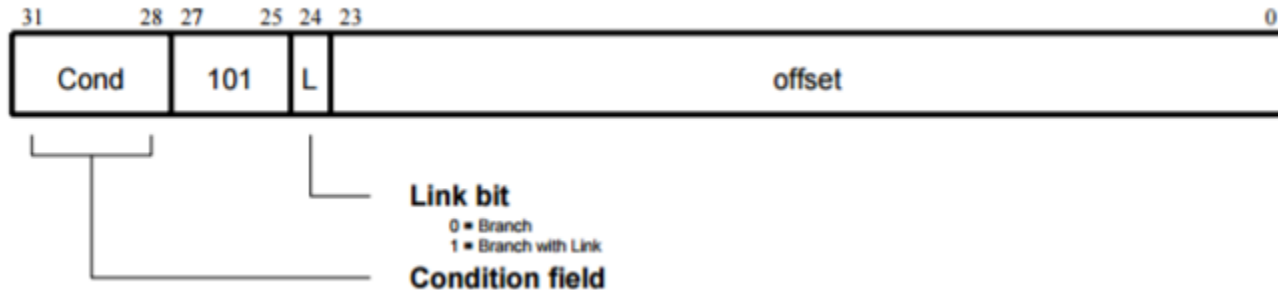
```
loop:  CMP    Ri, Rj    ; compara i e j
       BNE    loop     ; se i != j (Not Equal), vá para loop
```

Conjunto de instruções

Fluxo de Controle

- Endereçamento:

O intervalo de memória até o local onde está o rótulo de destino é armazenado no campo offset e, quando houver a mudança de fluxo, adicionado ao PC. Assim, uma instrução pode especificar um desvio de até +/- 32Mbytes



Mnemonic	Meaning after ARM data processing instruction	Meaning after VFP VCMP instruction
EQ	Equal	Equal
NE	Not equal	Not equal, or unordered
CS or HS	Carry set or Unsigned higher or same	Greater than or equal, or unordered
CC or LO	Carry clear or Unsigned lower	Less than
MI	Negative	Less than
PL	Positive or zero	Greater than or equal, or unordered
VS	Overflow	Unordered (at least one NaN operand)
VC	No overflow	Not unordered
HI	Unsigned higher	Greater than, or unordered
LS	Unsigned lower or same	Less than or equal
GE	Signed greater than or equal	Greater than or equal
LT	Signed less than	Less than, or unordered
GT	Signed greater than	Greater than
LE	Signed less than or equal	Less than or equal, or unordered
AL	Always (normally omitted)	Always (normally omitted)

3 1	3 0	2 9	2 8	2 7	2 6	2 5	2 4	2 3	2 2	2 1	2 0	1 9	1 8	1 7	1 6	1 5	1 4	1 3	1 2	1 1	1 0	9	8	7	6	5	4	3	2	1	0	Instruction Type
Condition				0	0	I	OPCODE					S	Rn				Rs				OPERAND-2								Data processing			

0000 = EQ - Z set (equal)

0001 = NE - Z clear (not equal)

0010 = HS / CS - C set (unsigned higher or same)

0011 = LO / CC - C clear (unsigned lower)

0100 = MI -N set (negative)

0101 = PL - N clear (positive or zero)

0110 = VS - V set (overflow)

0111 = VC - V clear (no overflow)

1000 = HI - C set and Z clear (unsigned higher)

1001 = LS - C clear or Z (set unsigned lower or same)

1010 = GE - N set and V set, or N clear and V clear (>or =)

1011 = LT - N set and V clear, or N clear and V set (>)

1100 = GT - Z clear, and either N set and V set, or N clear and V set (>)

1101 = LE - Z set, or N set and V clear, or N clear and V set (<, or =)

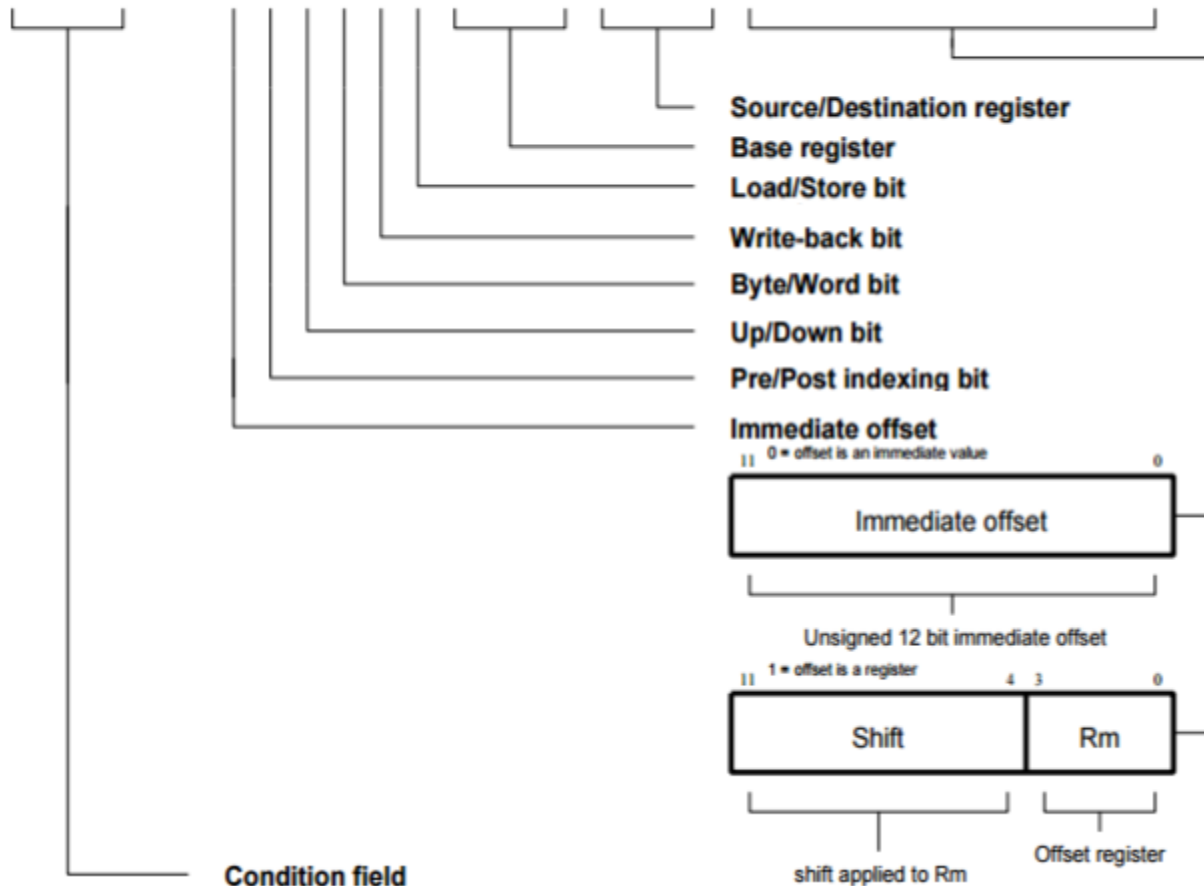
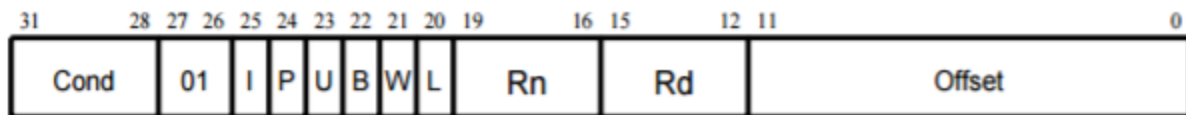
1110 = AL - always

1111 = NV - reserved.

Conjunto de instruções

Carregamento/Armazenamento

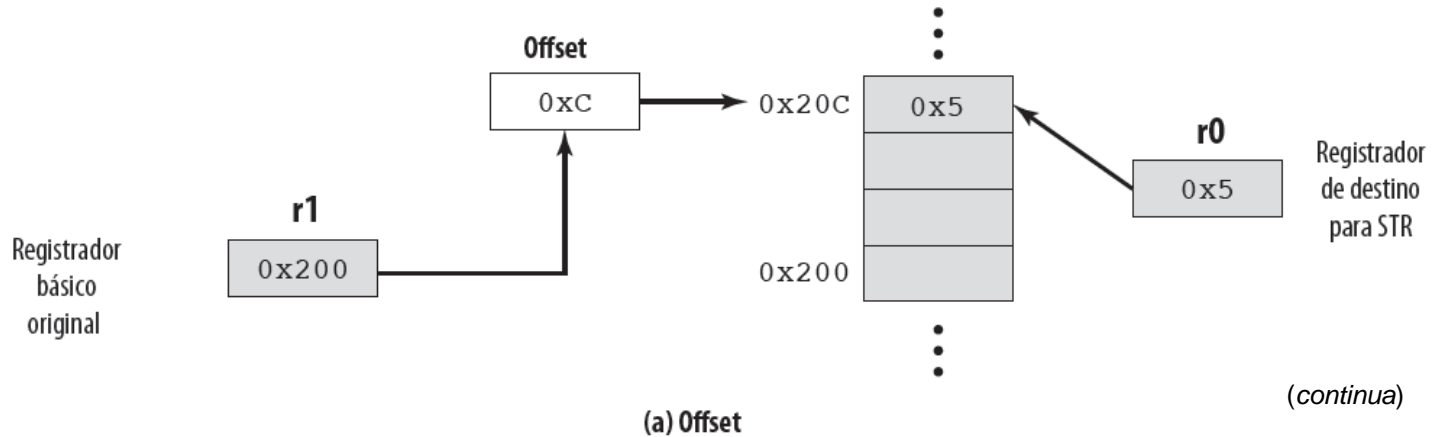
- Há duas instruções principais para acesso à memória:
 - LDR (register load): carrega em um registrador o conteúdo de uma posição de memória
 - STR (register store): coloca o conteúdo de um registrador em uma determinada posição de memória
- 3 Modos de endereçamento (offset, pré-indexado, pós-indexado)



Métodos de Indexação

Figura 11.3 Métodos de indexação ARM

STRB r0, [r1, #12]

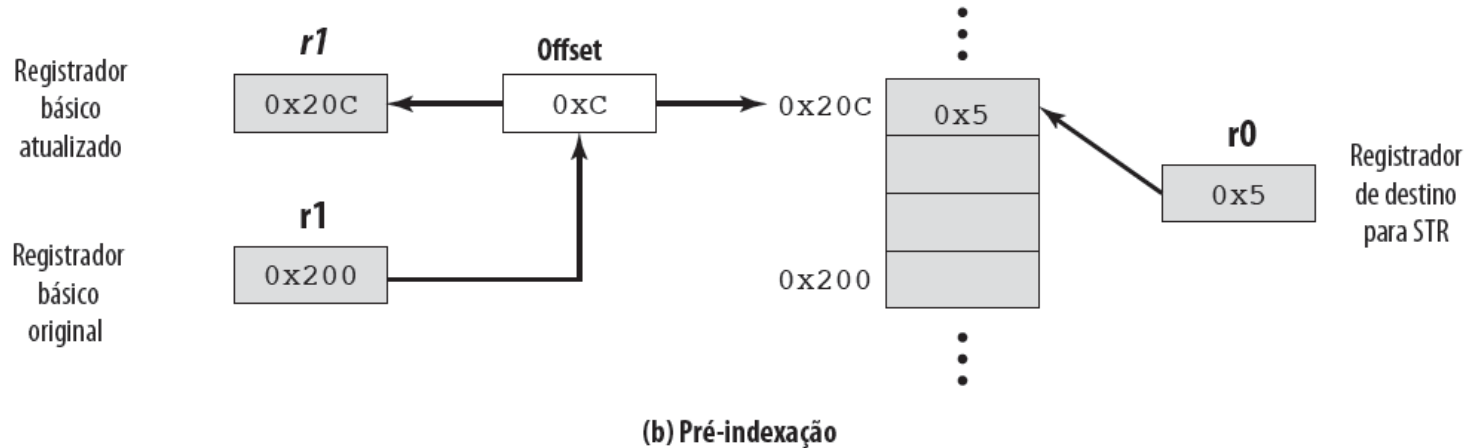


Métodos de Indexação

(continuação)

Figura 11.3 Métodos de indexação ARM

`STRB r0, [r1, #12]!`



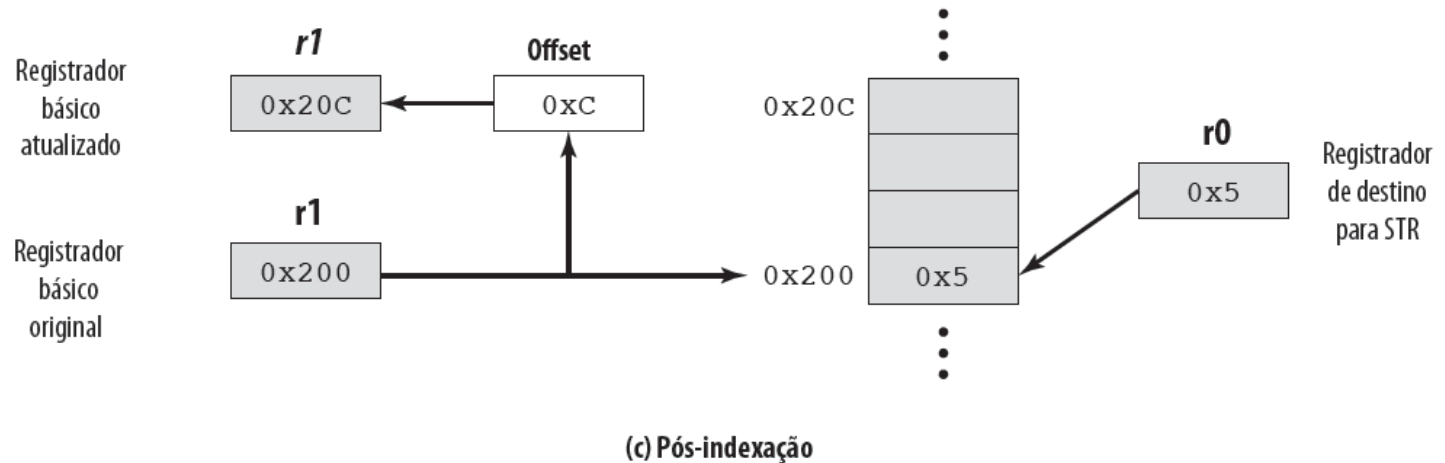
(continua)

Métodos de Indexação

(continuação)

Figura 11.3 Métodos de indexação ARM

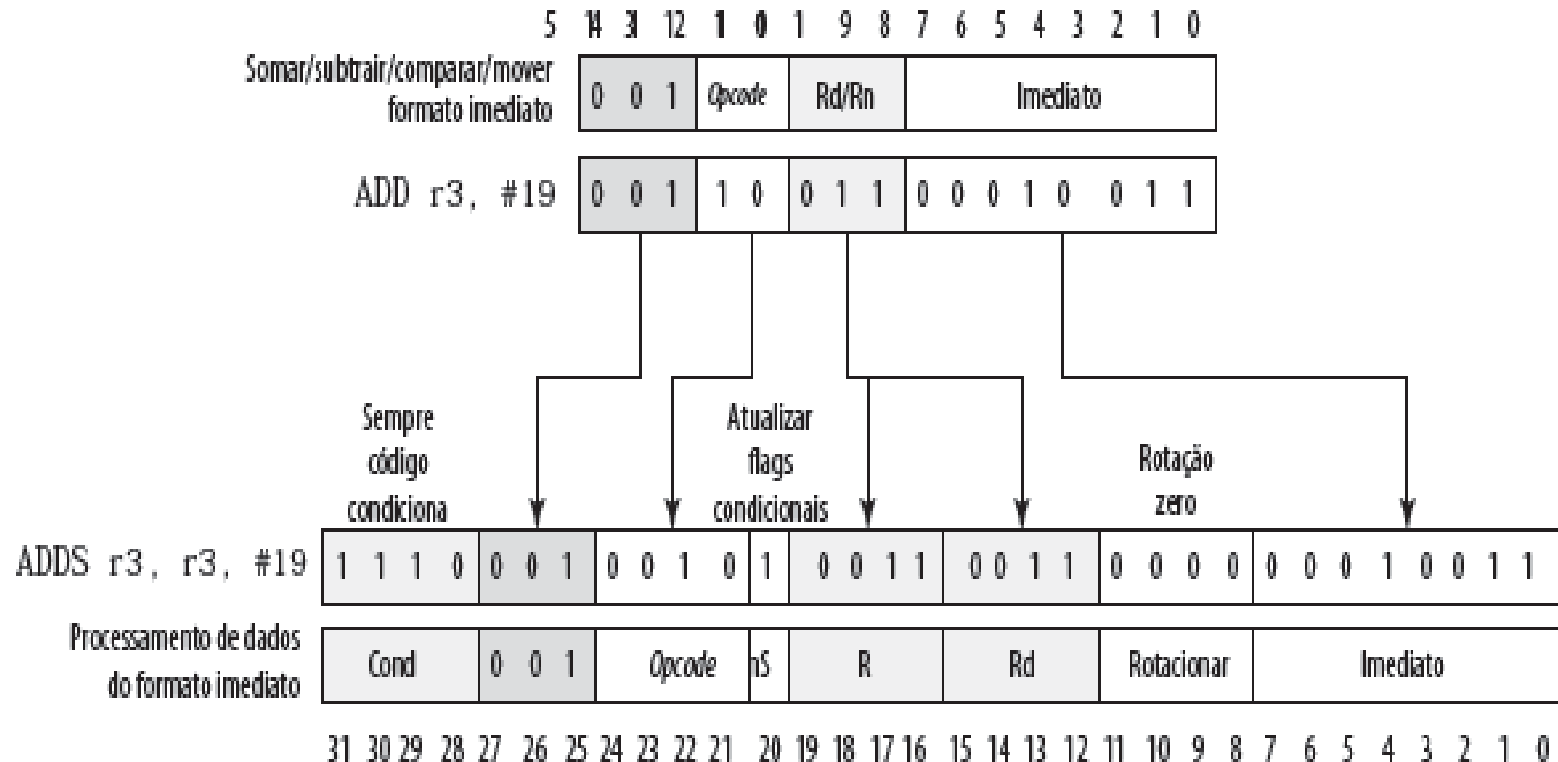
STRBv r0, [r1], #12



Conjunto de instruções Thumb

- Subconjunto recodificado do conjunto de instruções do ARM
- Aumenta desempenho em barramento de dados de 16 bits ou menos
- Incondicional (economia de 4 bits)
- Sempre atualiza flags de condição
 - Atualiza flag de não usado (economia de 1 bit).
- Subconjunto de instruções:
 - Opcode de 2 bits, campo de tipo de 3 bits (economia de 1 bit)
 - Especificações de operando reduzidas (economia de 9 bits)

Expandindo instrução ADD do Thumb para equivalente ARM



Exemplos de Códigos

- Como executar uma instrução condicional

```
if ( R0 > 30 ) {  
    R1 = 3;  
}else {  
    R2 = 4;  
}
```

```
CMP R0,#30  
MOVGT R1,#3  
MOVLE R2,#4
```

Exemplos de Códigos

- Instrução ADD normal
 - ADD r0, r1, r2 ; $r0 = r1 + r2$
- Instrução ADD que será executada somente se o flag de zero foi setado
 - ADDEQ r0, r1, r2 ; se flag de zero está setado então $r0 = r1 + r2$

Exemplos de Códigos

- Por padrão, as instruções de processamento de dados não afetam os flags de condição. Para fazer com que os flags sejam atualizados o bit S da instrução precisa estar setado adicionando um “S” na instrução
 - ADDS r0, r1, r2 ; $r0 = r1 + r2$ e os flags são setados

Exemplos de Códigos

- Comparação entre não usar as condições nas instruções e usar as condições

CMP r3, #0

BEQ skip

ADD r0, r1, r2

CMP r3, #0

ADDNE r0, r1, r2

loop

...

SUBS r1, r1, #1

BNE loop

Exemplos de Códigos

- C

```
while (i != j)    // We enter the loop when i<j or i>j, not when i==j
{
    if (i > j)      // When i>j we do this
        i -= j;
    else // When i<j we do that (no if(i<j) needed since i != j is checked in while condition)
        j -= i;
}
```

- ARM

```
loop:
// Compare i and j
GT = i > j;
LT = i < j;
NE = i != j;
// Perform operations based on flag results
if(GT) i -= j; // Subtract *only* if greater-than
if(LT) j -= i; // Subtract *only* if less-than
if(NE) goto loop; // Loop *only* if compared values were not equal
```

```
loop:    CMP Ri, Rj                ; set condition "NE" if (i != j),
                                           ; "GT" if (i > j),
                                           ; or "LT" if (i < j)
        SUBGT Ri, Ri, Rj           ; if "GT" (Greater Than), i = i-j;
        SUBLT Rj, Rj, Ri           ; if "LT" (Less Than), j = j-i;
        BNE loop                  ; if "NE" (Not Equal), then loop
```

Referências

<https://www.embarcados.com.br/breve-historico-da-arm/>

http://www.dca.fee.unicamp.br/~lbocato/topico_3.1_processador_ARM.pdf

<http://www.embedded.com/electronics-blogs/beginner-s-corner/4024632/Introduction-to-ARM-thumb>

https://web.eecs.umich.edu/~prabal/teaching/eecs373-f10/readings/ARMv7-M_ARM.pdf

<https://www.design-reuse.com/articles/11767/bringing-arm7-tm-to-the-masses.html>

<http://infocenter.arm.com/help/index.jsp>

SSC0114

Arquitetura de Computadores

27ª Aula – Arquiteturas M68000 (Motorola), PowerPC (IBM) e XeonPhi (Intel)

Profa. Sarita Mazzini Bruschi
sarita@icmc.usp.br

Motorola / Freescale / NXP / Qualcomm

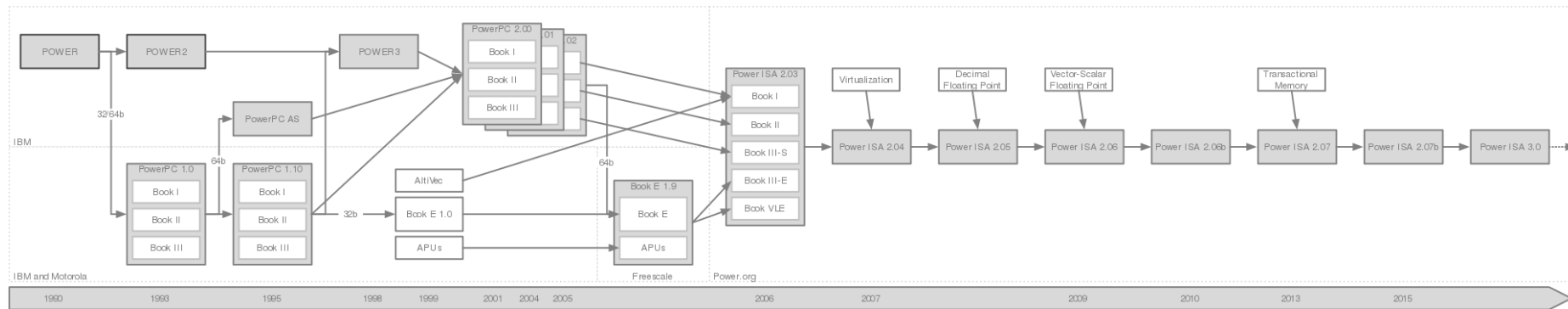
- Freescale foi uma das primeiras fabricantes de semicondutores (Autin / TX)
 - Começou como uma divisão da Motorola em 1948
 - Se tornou independente da Motorola em 2004
 - Em Dez/2015, NXP (Holandesa) e Freescale se uniram
 - Em Out/2016: Qualcomm anunciou que compraria a NXP
 - Em Abr/2017: Qualcomm recebeu aprovação do setor *antitrust* americano para concretizar a compra da NXP
- Primeiro microprocessador: MC6800 (1974 – 8 bits)
- Próxima geração: microprocessador 32 bits MC68000
- Power Series (Power PC): em parceria com a IBM
- ARM: i.MX (Ford Sync, Amazon Kindle, Sony Reader)

Motorola 68000

- Introduzido em 1979
- Também conhecido como **m68k**
- Arquitetura CISC
- 16/32 bits
 - Registradores internos de 32 bits
 - Barramento de endereço de 24 bits
 - Barramento de dados de 16 bits
- Conjunto de instruções era adequado para desenvolvimento do UNIX
 - CPU dominante em sistemas UNIX-based, tais como estações de trabalho (SUN), terminais gráficos (VAXstation 100) e computadores pessoais (Apple Lisa, Macintosh, Amiga e Atari)

Power Architecture

Conjunto de Instruções (ISA)



Power Architecture

Algumas nomenclaturas

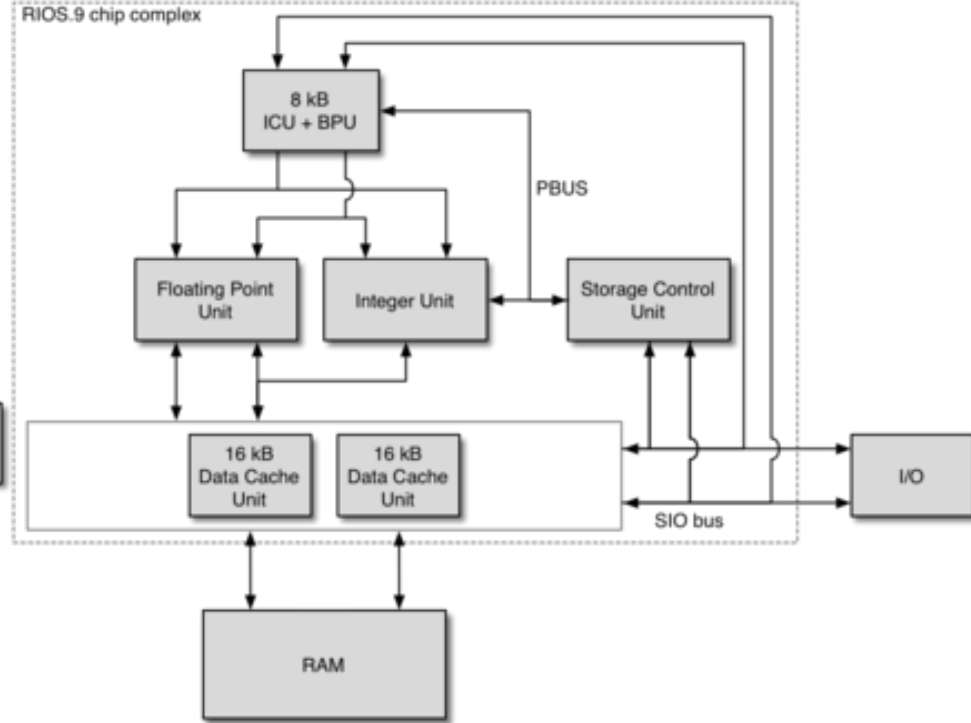
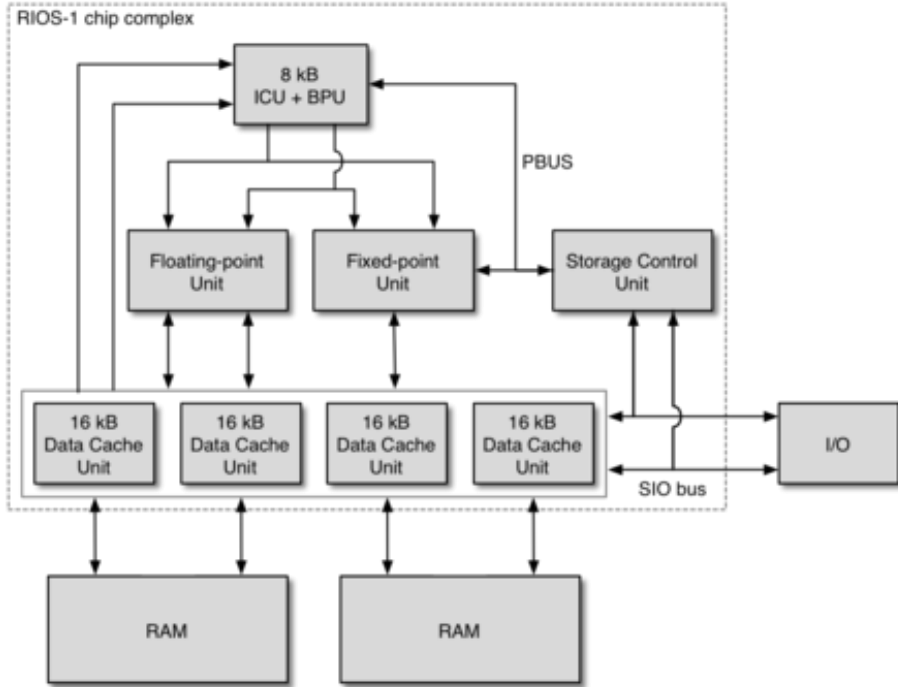
- POWER: **P**erformance **O**ptimization **W**ith **E**nhanced **R**ISC. Um *conjunto de instruções* de microprocessador antigo da IBM
- PowerPC: Power **P**erformance **C**omputing. Um *conjunto de instruções* de 32/64-bit para microprocessadores, incluindo alguns novos elementos. Projetado pela aliança AIM: Apple, IBM and Motorola. Renomeado para POWER ISA em 2006
- PowerPC-AS: **P**ower**P**C-**A**dvanced **S**eries. Apelido "Amazon". Um *conjunto de instruções* que é uma variação pura de 64-bits do PowerPC, incluindo alguns elementos da versão POWER2
- POWER*n*: Onde **n** é um número de 1 a 9. Uma *série de microprocessadores* de alta performance construído pela IBM usando diferentes combinações dos conjuntos de instruções POWER, PowerPC e PowerPC-AS

Processadores POWER (ISA)

- *America Project* (1982): projeto de pesquisa da IBM em uma segunda geração (a primeira foi o projeto do IBM 801) de arquiteturas RISC
 - Série RS/6000: primeiro computador a ter um processador POWER usando a primeira POWER ISA
 - Início do desenvolvimento em 1986
 - Lançamento em Fev/1990
 - Tinha duas configurações (ambas conhecidas como POWER1 CPU):
 - RIOS-1: 10 unidades independentes: 1 cache de instruções, unidades de ponto fixo e ponto flutuante, 4 chips de cache de dados, 1 unidade de controle de armazenamento, unidades de E/S e clock
 - RIOS-9: 8 unidades independentes: igual anterior só que com 2 unidades de cache de dados
 - Primeiro microprocessador a usar renomeamento de registradores e execução fora de ordem
- Outros processadores: POWER1+, POWER1++ e RSC (RIOS-1 single chip)

POWER

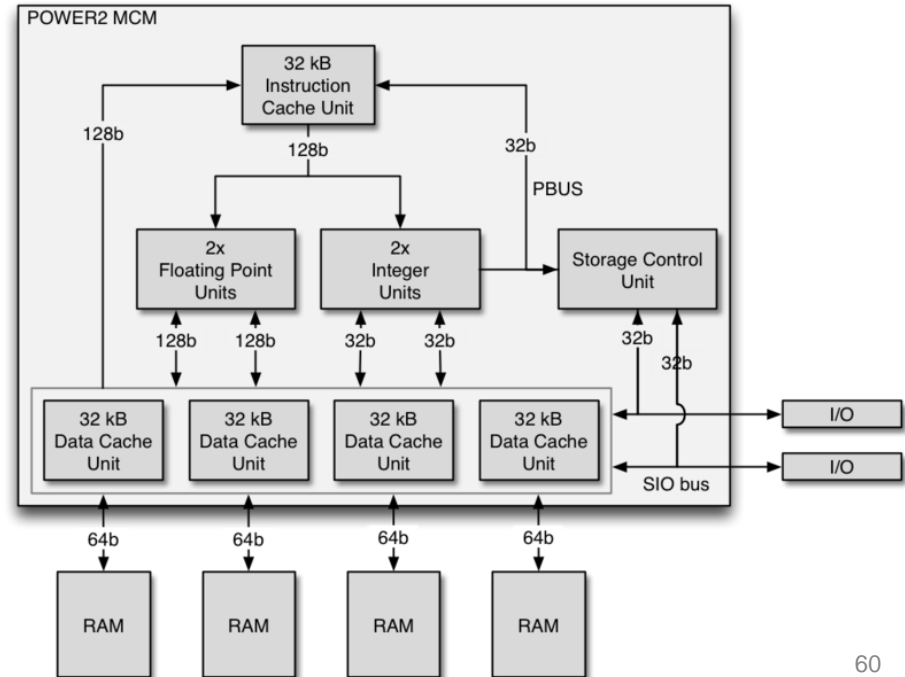
RIOS-1 e RIOS-9



ICU = Instruction Cache / BPU = Branch Processing Unit

Processadores POWER2 (ISA)

- Melhorias no conjunto de instruções
- Microprocessadores:
 - POWER2
 - POWER2+
 - P2SC (Super Chip)
 - POWER3



Processadores PowerPC

- **POWERPC: *Performance Optimization With Enhanced RISC – Performance Computing***
 - Criado em 1991 pela aliança Apple-IBM-Motorola, conhecida como *AIM*
 - Baseado na arquitetura do conjunto de instruções POWER, da IBM, com algumas características do processador 88000 da Motorola
 - Versões da ISA: PowerPC 1.1, PowerPC 2.01, PowerPC 2.02
 - O conjunto de instruções foi renomeado para *Power ISA* em 2006
 - Power ISA 2.03, Power ISA 2.04, Power ISA 2.05, Power ISA 2.06, Power ISA 2.06b, Power ISA 2.07, Power ISA 2.07b, Power ISA 3.0

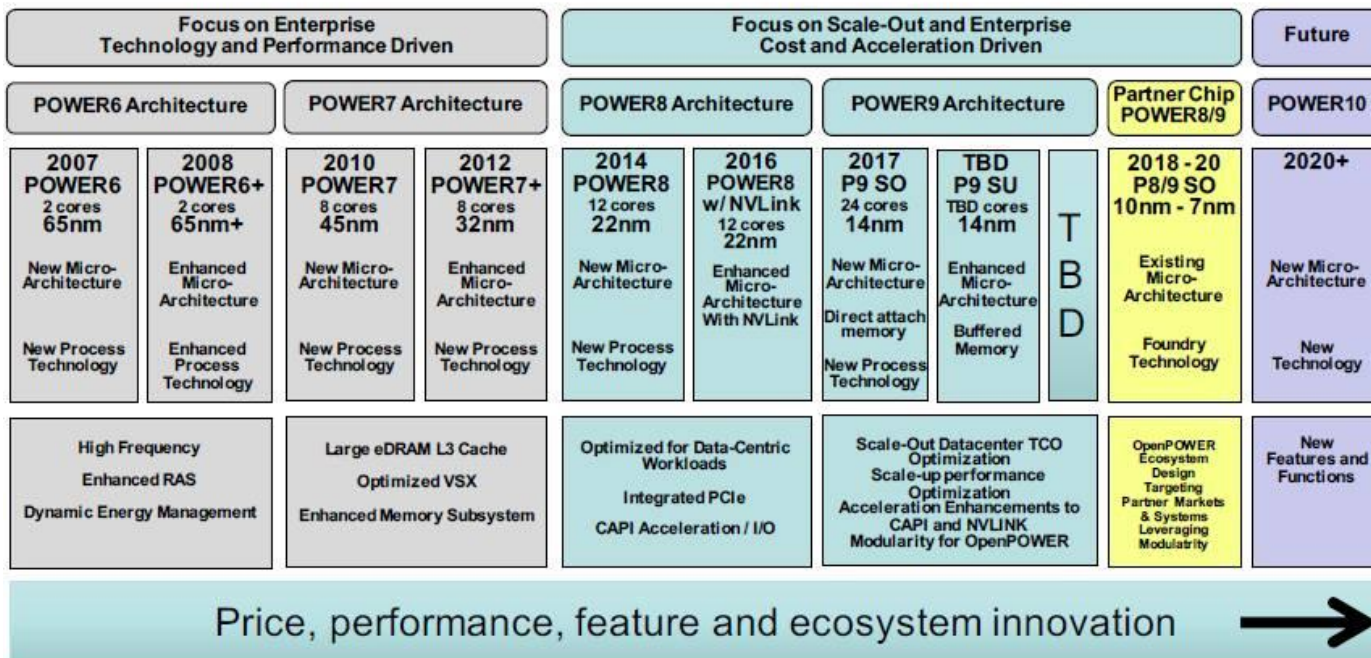
Processadores PowerPC

- PPC601, ou MPC601, ou PowerPC 601: primeiro geração a suportar o conjunto de instruções (ISA) PowerPC em 1993
 - Usado na workstation IBM/RS6000 (1993) e no primeiro computador Power Macintosh (1994)
 - É uma CPU superescalar de 64 bits que podia eficientemente executar até 3 instruções por ciclo de clock
- PPC 603e: chip com baixo consumo de energia, foi usado no primeiro PPC PowerBooks

Power



POWER Processor Roadmap

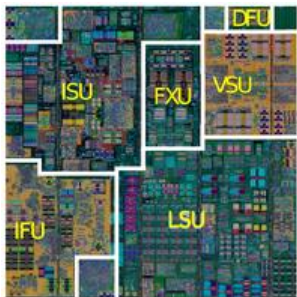


POWER9

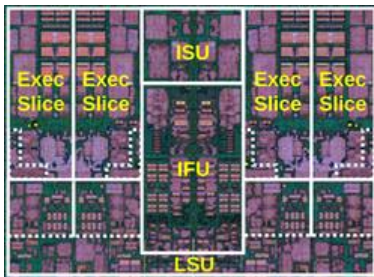
- Supercomputador que é o primeiro no TOP500:
 - Summit
 - Oak Ridge Leadership Computing Facility's (OLCF's)
 - 3400 nós, cada um com 2 CPUs POWER9 e 6 NVIDIA VOLTA conectados por NVLink e um link com a memória de 512GB de alta bandwidth entre a CPUs e GPUs e 800GB de NVRAM (*Non-Volatile RAM*)

POWER9

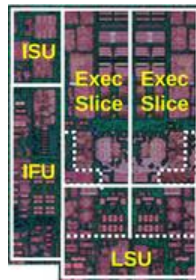
Power8



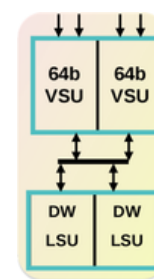
P9 SMT8 (4x SuperSlice)



P9 SMT4 (2x SuperSlice) SuperSlice

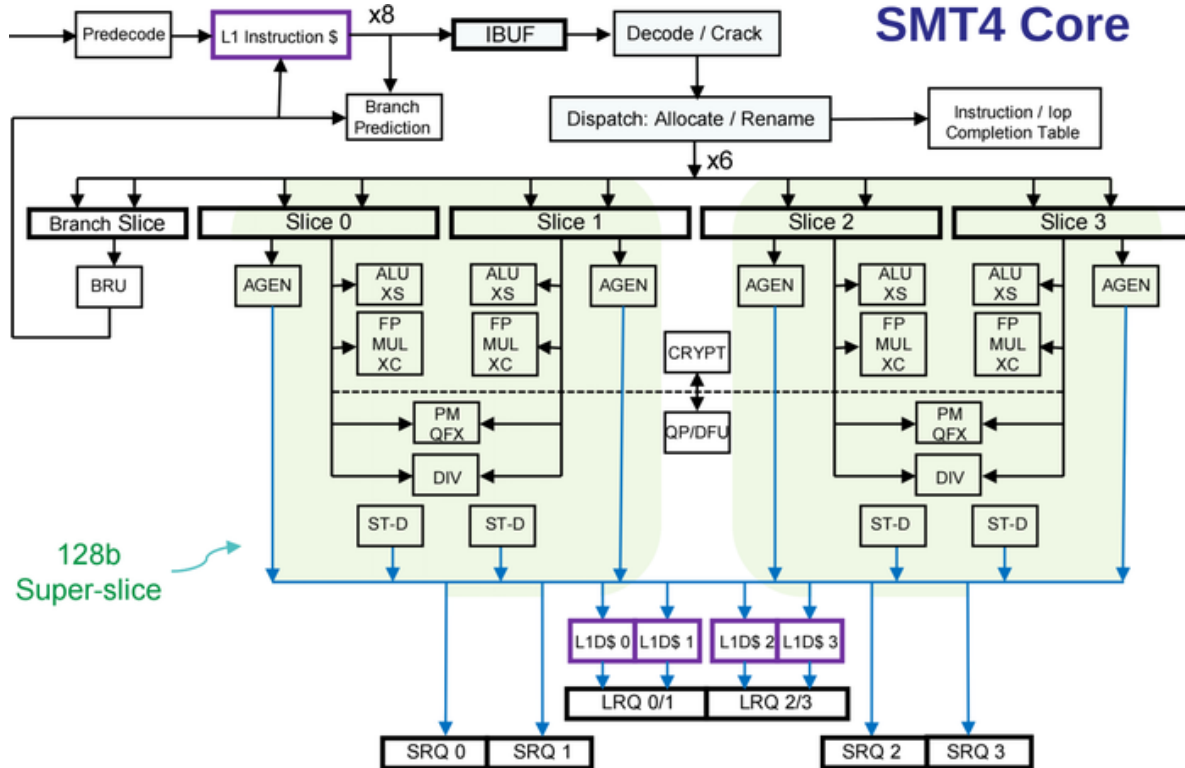


Slice



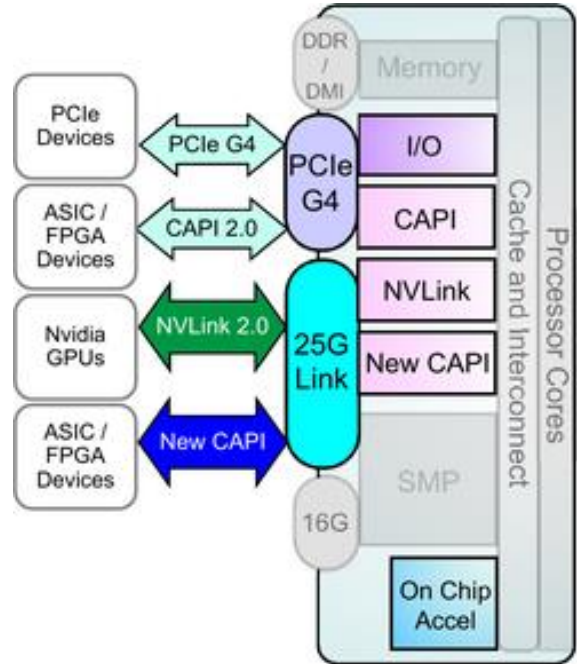
VSU: Vector Scalar Unit
LSU: Load Store Unit

POWER9



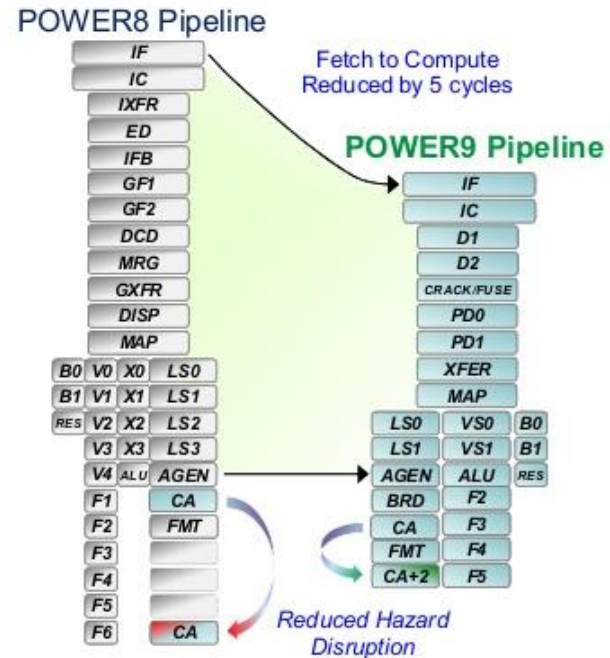
POWER9

- PCIe Gen4: 48 lanes a 192 GiB/s
- CAPI (Coherence Accelerator Processor Interface)
- New CAPI: projetada para acelerar a comunicação entre CPU e aceleradores (FPGAs)
- NVLink: ligação do processador com GPUs



POWER9

- Pipeline:
 - 12 estágios (5 a menos do que a versão anterior – POWER8)

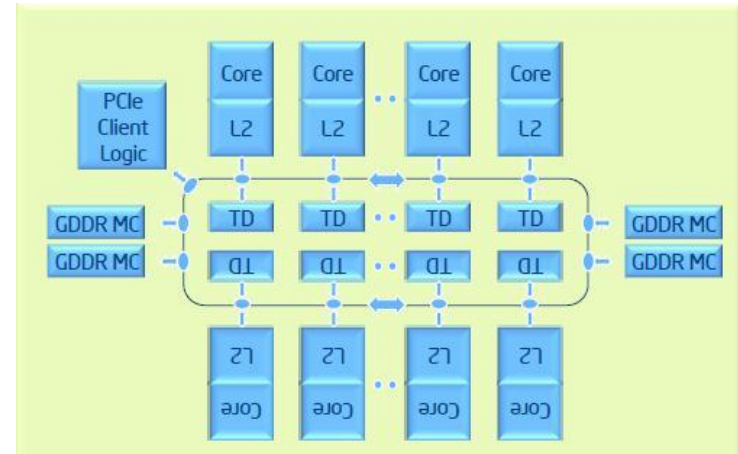


Processadores da Apple

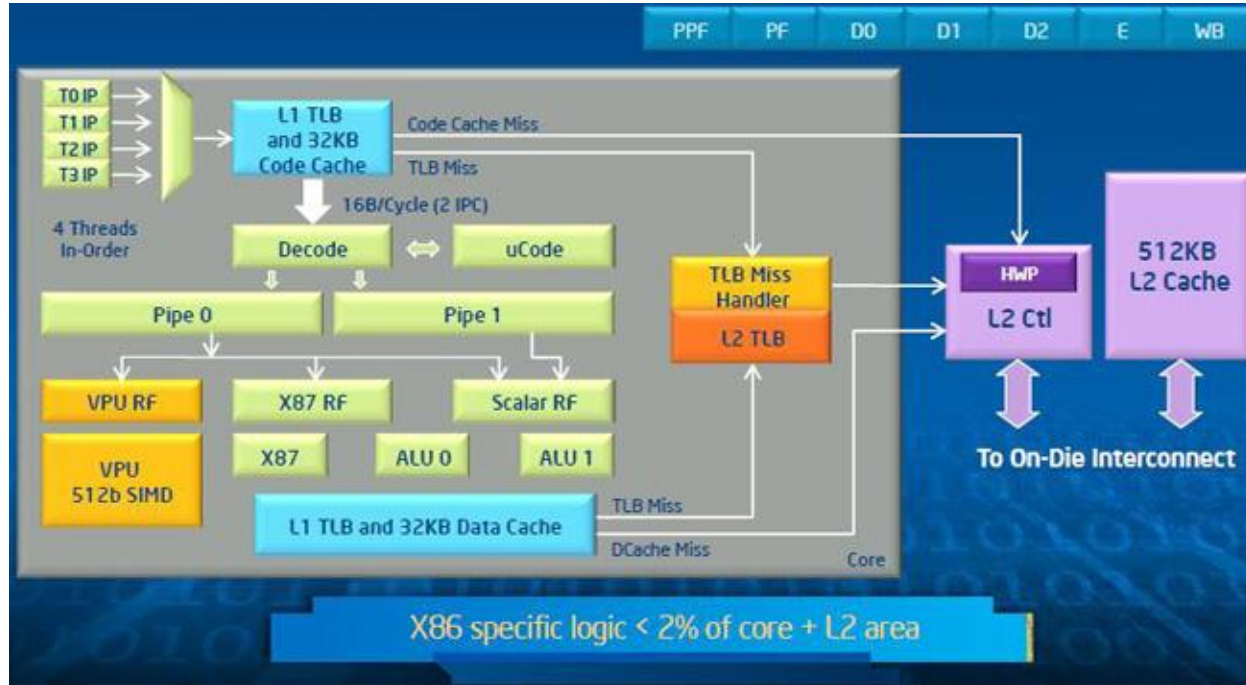
- Motorola 68000 (e variações)
 - 1984 a 1996
- PowerPC 601 (602, 603, 604, G3, G4 e G5)
 - 1995 a 2006
- Intel (P6, Core, Penryn, Nehalem, Westmere, Sandy Bridge, Haswell, Broadwell, Skylake)
- Motivo da mudança do uso dos processadores PowerPC:
 - Freescale e IBM decepcionaram a Apple ao não produzir chips com maiores clocks

Xeon Phi

- Série de processadores manicores x86 (MIC – *Many Integrated Core*)
- Utilizados em supercomputadores, servidores, e estações de trabalho de alto desempenho



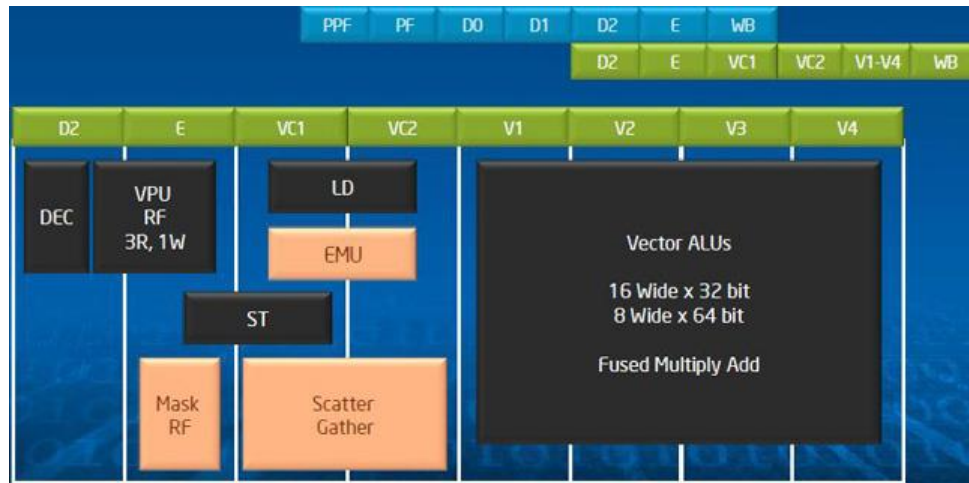
Xeon Phi Core



Xeon Phi

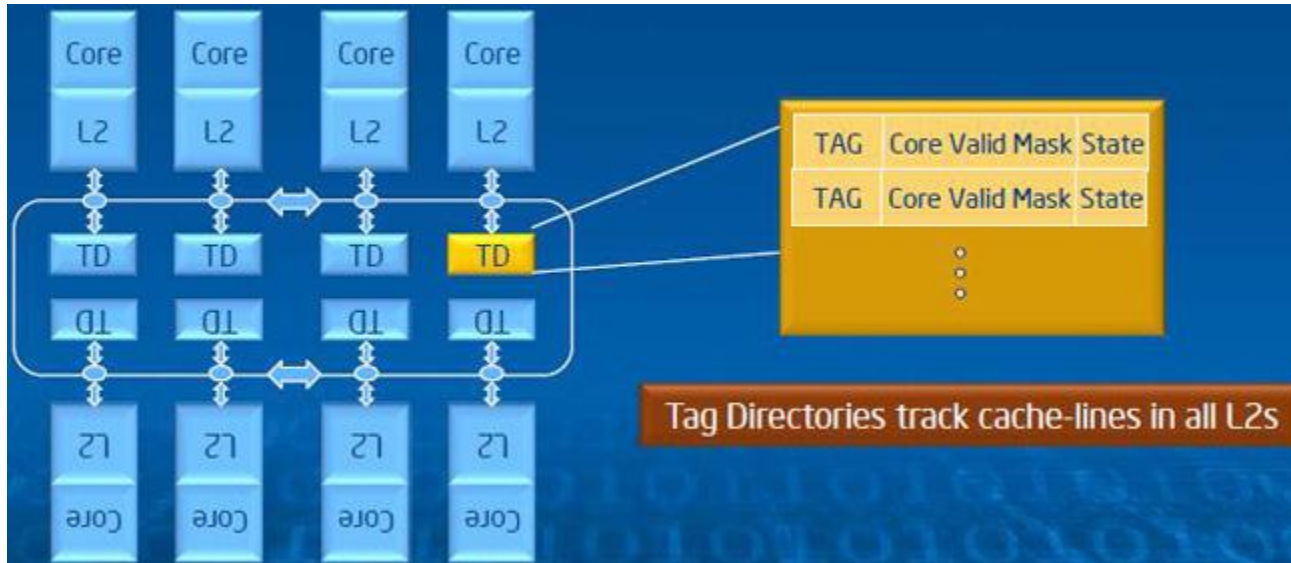
Vector Processing Unit

- Intel Initial Many Core Instruction (Intel IMCI)
 - Conjunto de instruções SIMD de 512 bits
 - Pode executar 16 operações com precisão simples e 8 com precisão dupla por ciclo de clock

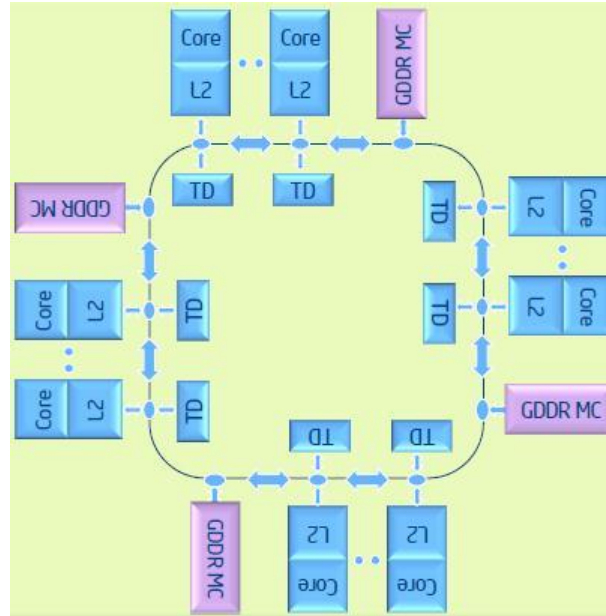


Xeon Phi

Diretório de Tags Distribuído



Xeon Phi Memory



The 11 Most Influential Microprocessors of All Time

By Benj Edwards, PCWorld | Aug 31, 2009

- Intel Pentium (1993): Brand-name processor
- Motorola 68000 (1980): Apple Macintosh (1984)
- AIM PowerPC 601 (1992): Apple Power Macintosh 6100 (1994)
- RCA COSMAC CDP 1802 (1976): NASA Voyager 1 (1977)
- AMD Opteron 240 (2003): IBM server hardware
- Zilog Z80 (1976): Radio Shack TRS-80 Model I
- MOS Technology 6502 (1975): Apple II (1977)
- Intel 8088 (1979): IBM PC 5150 (1981)
- Acorn Computers ARM2 (1986): Acorn Archimedes (1987)
- Intel 8080 (1974): MITS Altair 8800 (1975)
- Intel 4004 (1971): Busicom 141-PF Calculator (1971)