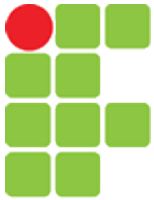


Instituto Federal de Santa Catarina
Área de Telecomunicações

MIC29004 – Microprocessadores

Prof. Roberto de Matos

roberto.matos@ifsc.edu.br

São José, maio de 2019.

Aviso de direitos Autorais:
Transparências baseadas no trabalho do
Prof. Eduardo Augusto Bezerra

Microcontroladores

Microcontroladores

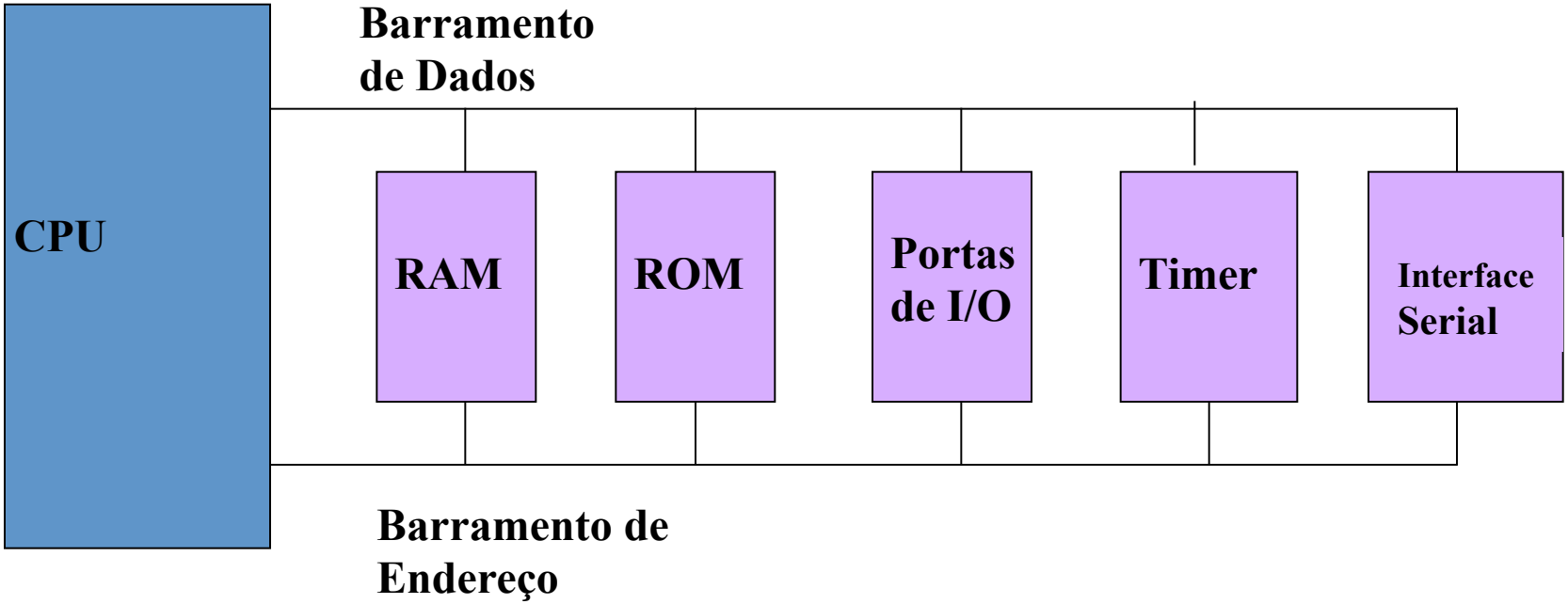
Componentes básicos de sistemas computacionais processados:

– CPU
– Memória de dados e programa
– Sistema de entrada/saída

Microcontroladores

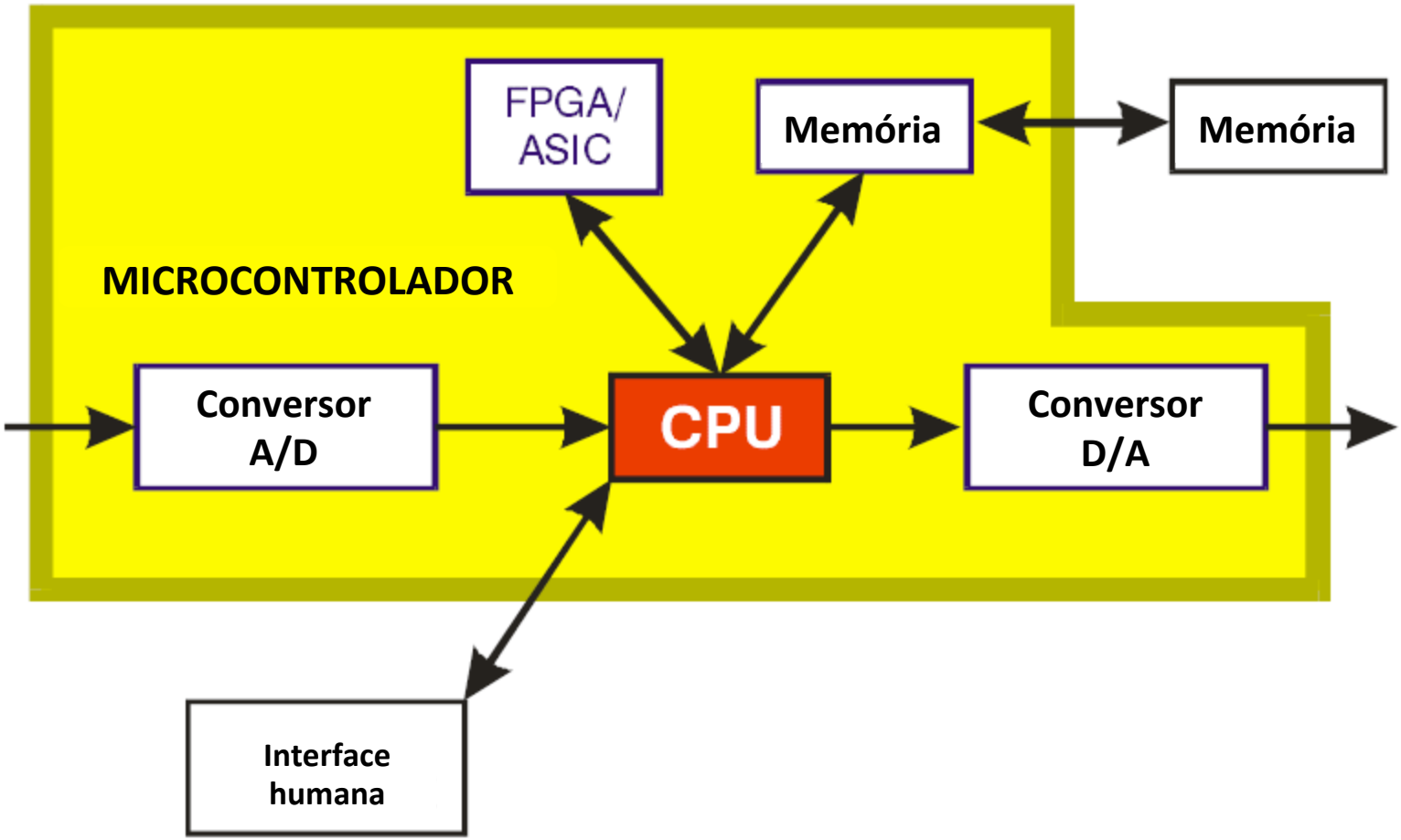
Microprocessadores de Propósito Geral

Sem RAM, ROM ou dispositivos de I/O



Vantagem: flexibilidade, sistema expansível ;
Desvantagem: custo, roteamento de placa e dimensões do circuito.

Microcontroladores



Microcontroladores

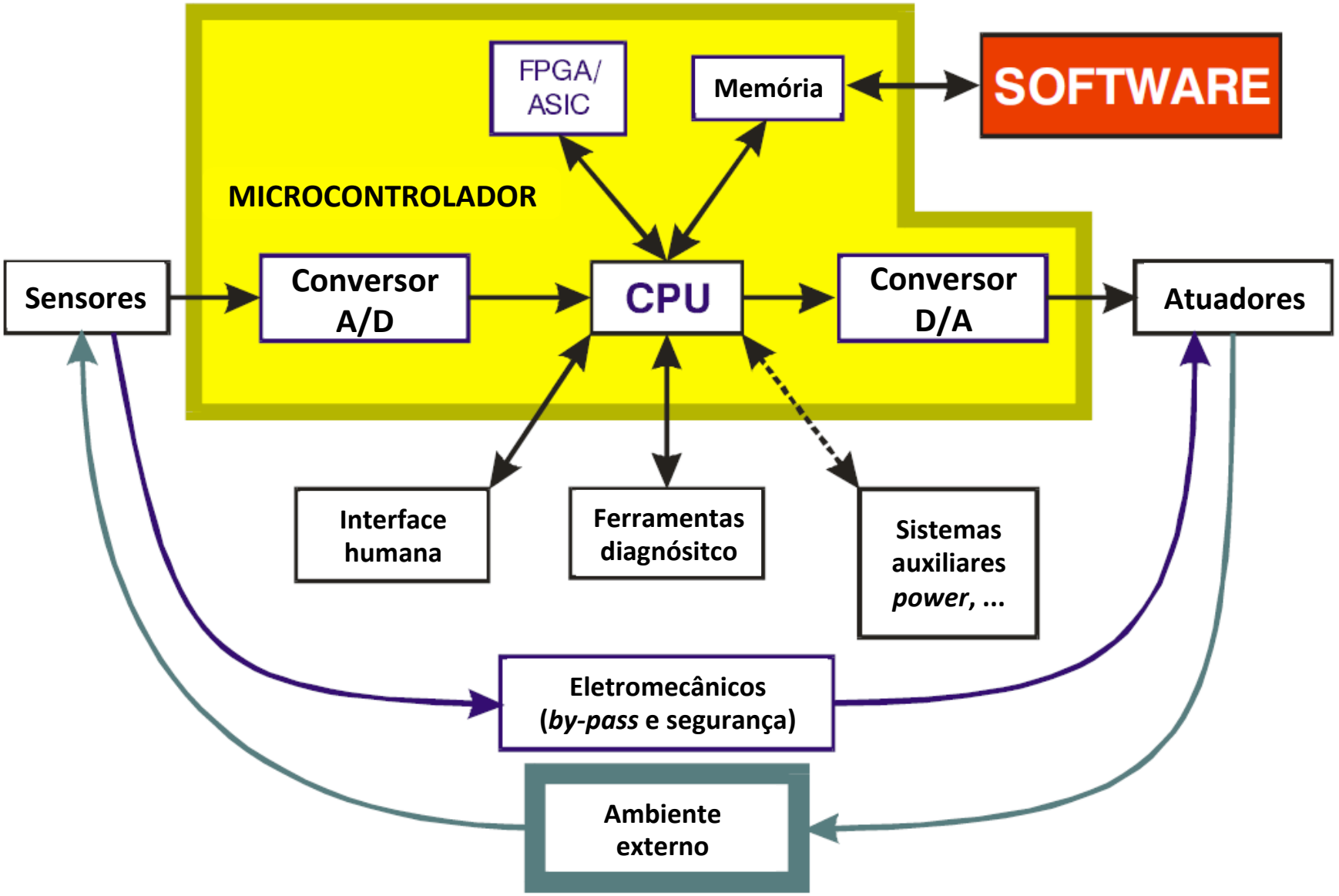
Microprocessadores são computadores de propósito geral

– São necessários componentes periféricos, externos, para apoio à execução das aplicações

Microcontroladores são computadores em um único chip

– Os periféricos estão embarcados no mesmo chip da CPU
– Algumas características, tamanho e custo reduzidos, alto desempenho com baixo consumo de energia, uso eficiente de espaço no PCB, baixo clock, endereçamento bit-a-bit

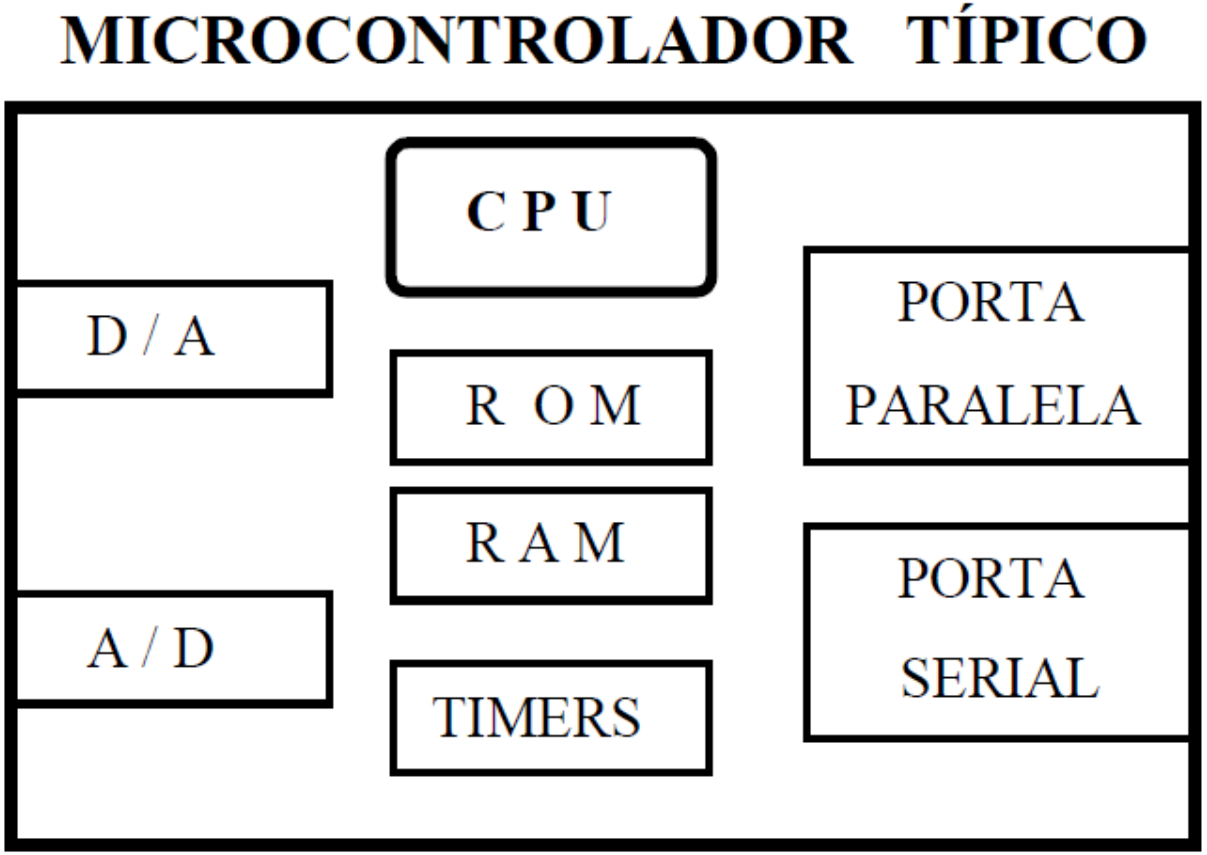
Microcontroladores



Microcontroladores

MCU – *Microcontroller Unit*
Composta por CPU e periféricos no mesmo encapsulamento

- Memória de Dados e Programa; Portas de Entrada e Saída (I/O); Temporizadores (Timers); EEPROM; Conversores AD/DA; USB.

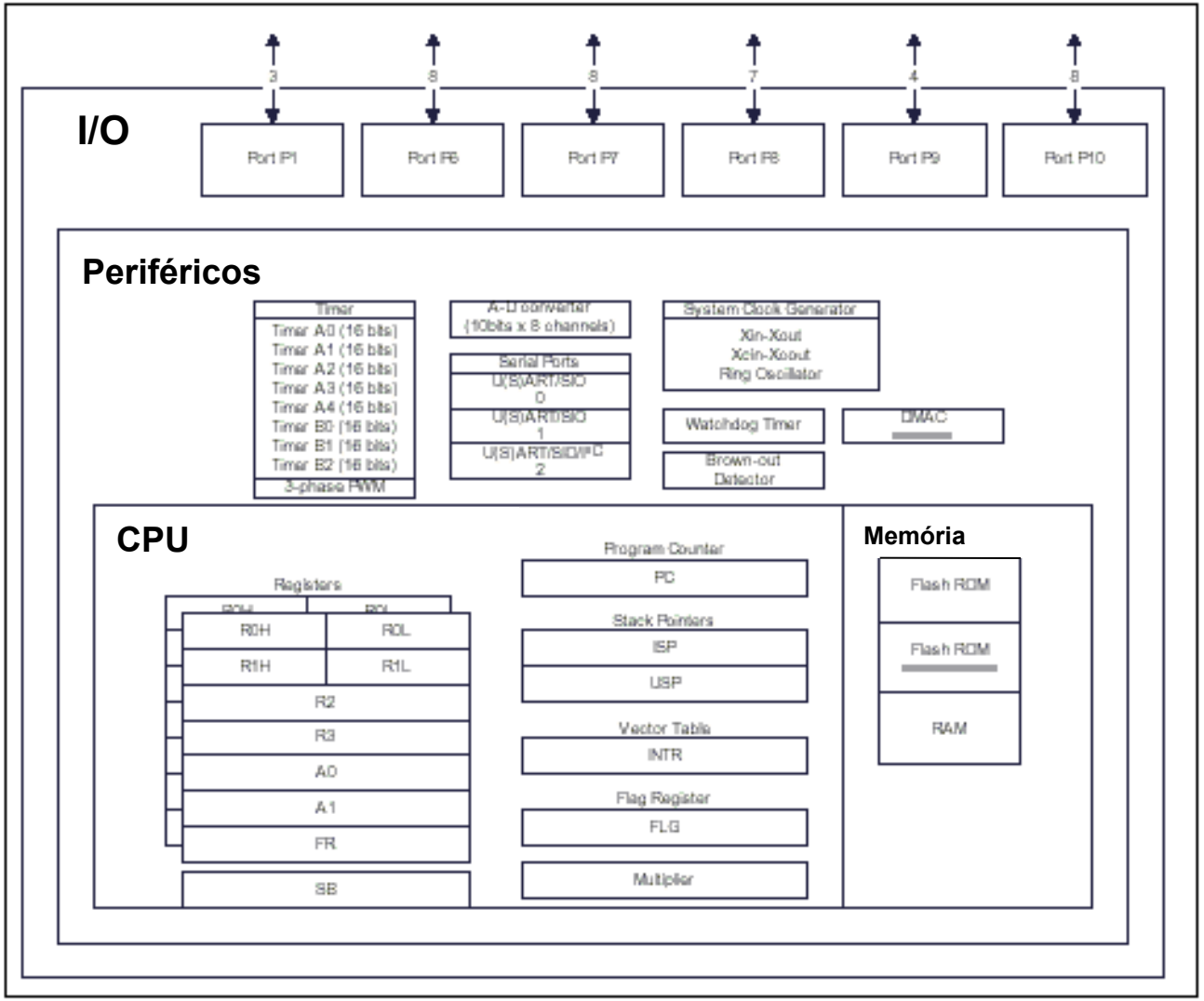


Microcontroladores

MCU – *Microcontroller Unit*

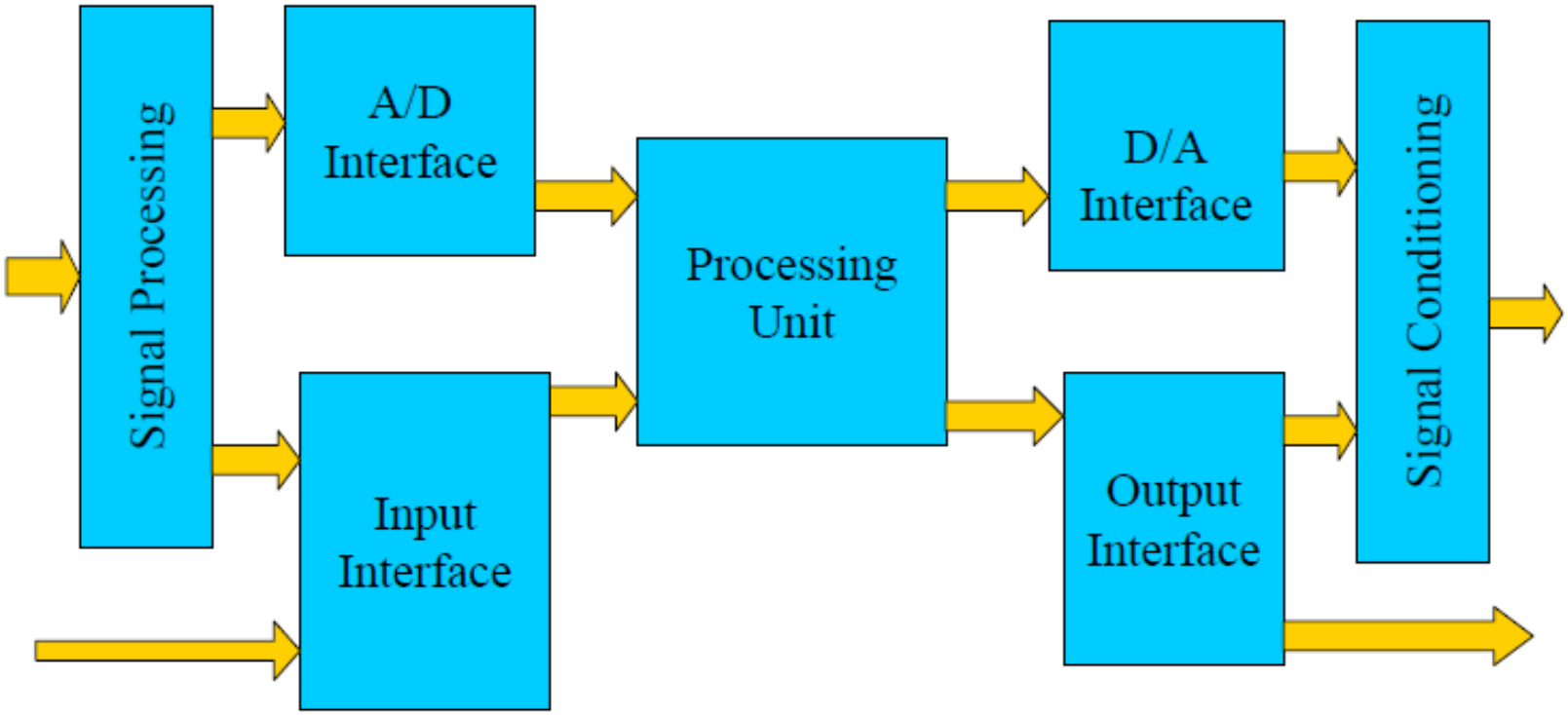
Composta por CPU e periféricos no mesmo encapsulamento

- Registradores
- RAM
- Flash
- EEPROM
- Portas digitais
- Portas Analógicas
- Timers
- Gerador de relógio
- DMA



Microcontroladores

Fluxo de dados



Microcontroladores

Diversidade de fabricantes e modelos

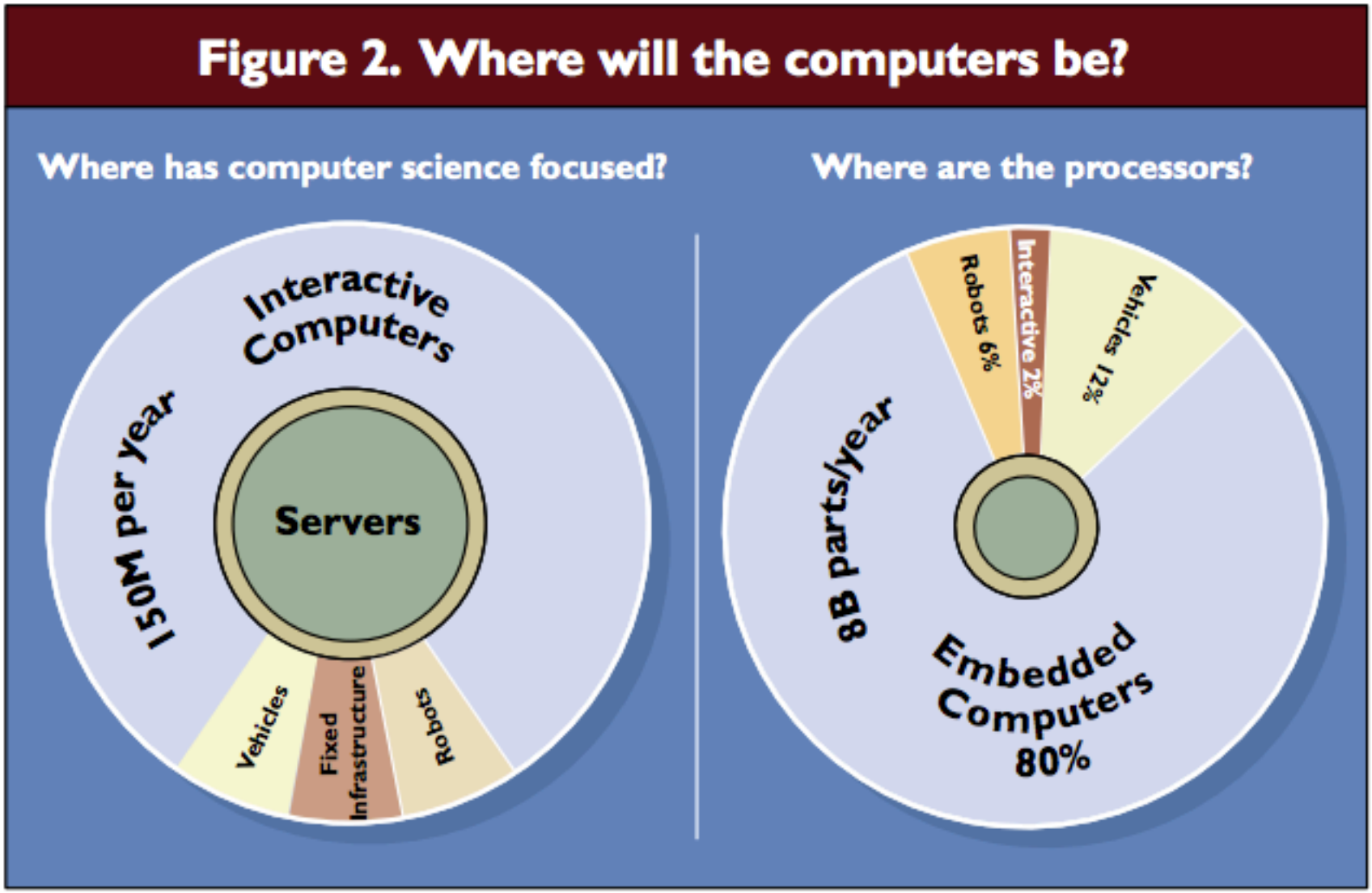
- LINHA PIC (Microchip)
- LINHA AVR (Atmel)
- LINHA 8051 (Philips, Dallas, Intel, Cygnal, Texas, TDK, Siemens ...)
- Z8 Encore (Zilog)
- HC08 (Motorola)
- ...

Escolha do dispositivo

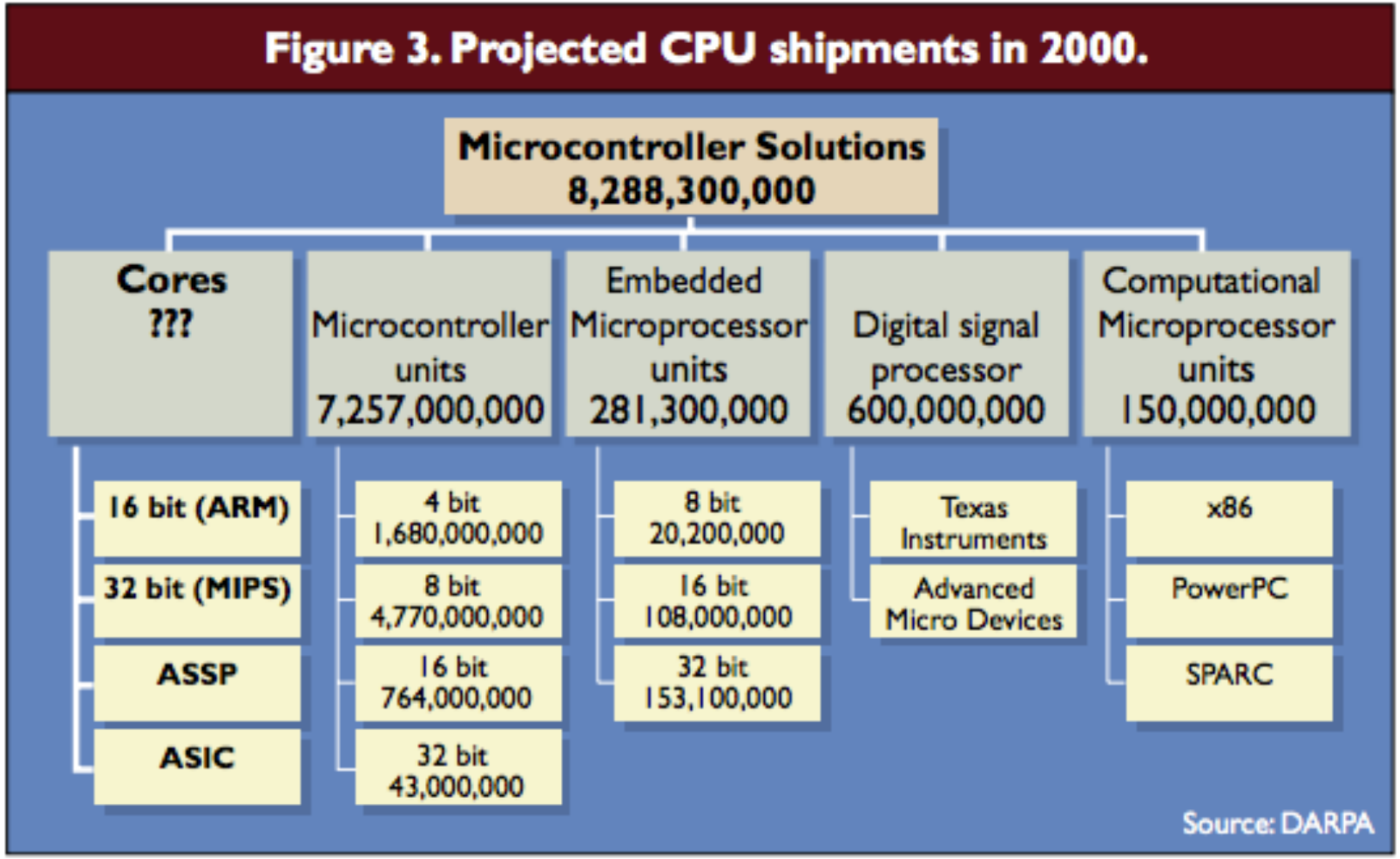
- Capacidade de processamento
  - 8 bits, 16 bits, 32 bits
  - Clock, 4MHz, 40Mhz, ...
- Periféricos necessários
- Capacidade de memória
  - Programa
  - Dados
- Outros fatores
  - Ferramentas disponíveis
  - Formato físico
  - Continuidade / Reaproveitamento de projeto

Microcontroladores: aplicações

Microcontroladores: aplicações



Microcontroladores: aplicações



Microcontroladores: aplicações

Microcontroladores: aplicações

Microcontroladores: aplicações

- Área Automobilística
- Automação
- Segurança
- Controle de Tráfego
- Médica
- Entretenimento
- Robótica

Microcontroladores: aplicações

- Embarcados em:
  - Sistemas automotivos
  - Aviônicos
  - Brinquedos
  - Dispositivos médicos
  - Eletrodomésticos
- Bilhões de unidades

Microcontroladores: aplicações

- Produtos de uso pessoal: Celulares, pagers, relógios, gravadores portáteis, calculadoras, câmeras fotográficas
- Laptops: mouse, teclado, modem, fax, placa de som, carregador de bateria
- Domótica: tranca eletromagnética, despertador, termostato, ar condicionado, controle remoto de TV, secador de cabelo, aparelho de DVD, geladeira, lavadora de roupa/louça, forno de microondas

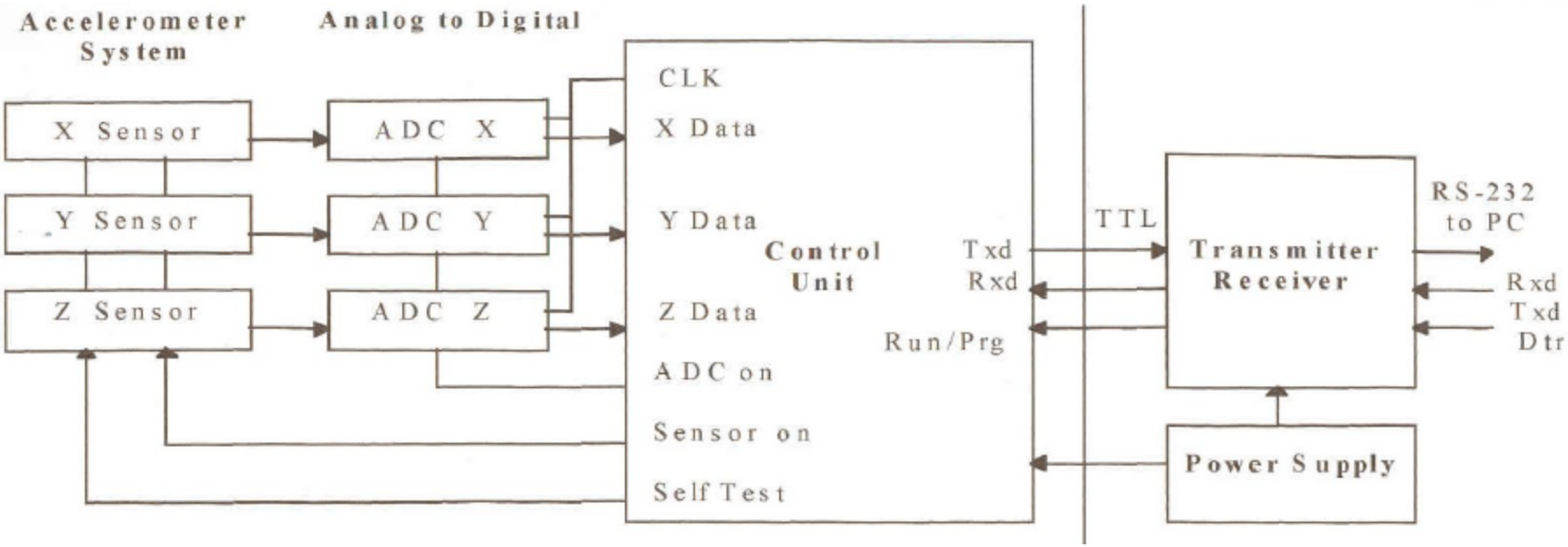
Microcontroladores: aplicações

Outro exemplo corriqueiro é o despertador

Outro exemplo é a parte de segurança

Microcontroladores: aplicações

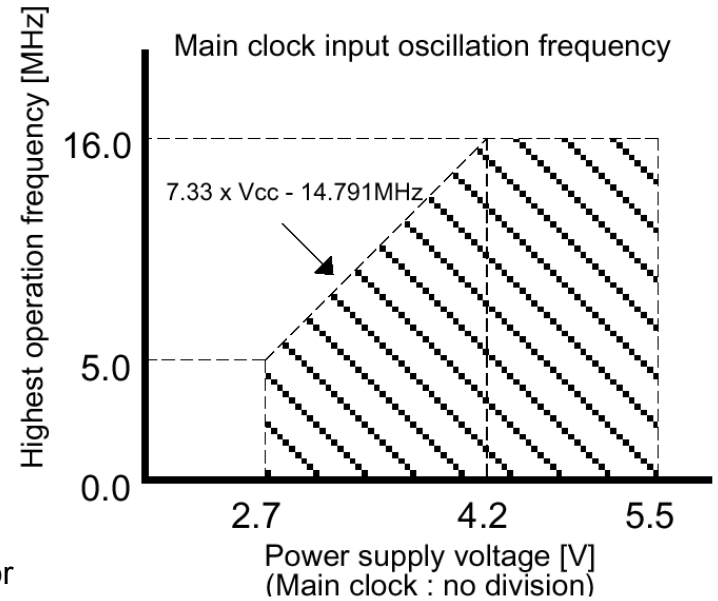
Microcontroladores: aplicações



Microcontroladores: Limitações

Limitações

- **Limitação importante: velocidade de processamento**
- **Não adequados para aplicações com tempo de resposta abaixo de poucos microsegundos**
- **Ambiente de desenvolvimento (compiladores, montadores, linkers, bibliotecas, plataformas de software e hardware, ...) – pode ser de uso complexo e custo elevado**
- **Tamanho dos programas e dados (recursos de memória escassos)**
- **Programas sequenciais**



Limitações

PIC

- Disponibilidade em encapsulamento DIP para uso direto em placas de prototipação
- Valores na ordem de US$1 a US$9
- Limitação: Custo das ferramentas – Compilador ~US$200; Debug ~US$150.

AVR

- Ferramentas gratuitas (gcc)
- IDE disponível para Windows, Mac e Linux, incluindo debug
- AVR-Dragon da Atmel custa em torno de US$50 e pode ser utilizado para programação e depuração
- Limitação: poucas famílias de dispositivos disponíveis (pouca variedade) ao se comparar com o PIC

Limitações

Microcontrolador:

Vantagens:

- **Mais versátil que CPLD, especialmente para aplicações analógicas (A/D, D/A).**
- **Facilidade para implementar algoritmos complexos e funções densas**

Desvantagens:

- **Temporização difícil de ser determinada para aplicações mais complexas (em C)**
- **Normalmente, menos desempenho em tempo de execução do que CPLD**

CPLD:

Vantagens:

- **Temporização eficiente e precisa**
- **Normalmente, melhor desempenho em tempo de execução do que microcontroladores**

Desvantagens:

- **Limitação para aplicações complexas e lógicas densas**

Limitações

Programa Exemplo: Loop

```
/* pulses pin PORTB<3>
eight times */

pulse:
movlw 0x08
movwf counter

pulse_lp0:
bsf       PORTB, 3
bcf       PORTB, 3
decfsz counter, F
goto      pulse_lp0
return
```

Assembly

```
/* pulses pin PORTB<3>
 eight times */

void pulse()
{
   int i;
   for (i=0; i<8; i++)
     { output_high(PIN_B3);
          output_low(PIN_B3);
          }
  return;
}
```

C

Limitações

Compilador Ineficiente

```
/* pulses pin PORTB<3>
eight times */

0000: movlw  0x8
0001: movwf  0x20
0002: bsf    0x6,0x3
0003: bcf    0x6,0x3
0004: decfsz 0x20
```

Assembly escrito pelo desenvolvedor

```
/* pulses pin PORTB<3> eight
   times */
   0005:  CLRF   21
   0006:  MOVF   21,W
   0007:  SUBLW  07
   0008:  BTFSS  03,0
   0009:  GOTO   014
   000A:  BSF    03,5
   000B:  BCF    06,3
   000C:  BCF    03,5
   000D:  BSF    06,3
   000E:  BSF    03,5
   000F:  BCF    06,3
   0010:  BCF    03,5
   0011:  BCF    06,3
   0012:  INCF   21,F
   0013:  GOTO   006
```

Assembly gerado pelo compilador

RISC e CISC

Microprocessadores

Componentes básicos de sistemas computacionais processados:

– CPU
– Memória de dados e programa
– Sistema de entrada/saída

Microcontroladores são computadores em um único chip

– Os periféricos estão embarcados no mesmo chip da CPU
– Algumas características: tamanho e custo reduzidos, alto desempenho com baixo consumo de energia, uso eficiente de espaço no PCB, baixo clock, endereçamento bit-a-bit

Microprocessadores são computadores de propósito geral

– São necessários componentes periféricos, externos, para apoio à execução das aplicações

CISC – "*Complex Instruction Set Computer*"

- **Arquiteturas projetadas para facilitar a programação (assembly), e com acesso eficiente a memória**
- **Memória cara e lenta representava na época situação ideal para CISC**
- **Exemplos de arquiteturas da época incluem o PDP-11 e o DEC system 10 e 20**
- **Por razões semelhantes, arquiteturas de microprocessadores largamente utilizados no passado tais como o Intel 80x86 e o Motorola 68K também seguiram a filosofia CISC**
- **Avanços na tecnologia de software e hardware levaram a uma reavaliação na filosofia CISC, resultando em novas arquiteturas híbridas implementando princípios RISC**
- **CISC foi desenvolvido para facilitar o desenvolvimento de compiladores. Por exemplo, o compilador não precisa gerar longas seqüências de instruções para calcular uma raiz quadrada, uma vez que existe no hardware das arquiteturas CISC instruções com essa funcionalidade.**

CISC – "*Complex Instruction Set Computer*"

- **Restrições de projeto/tecnológicas que direcionaram o desenvolvimento da arquitetura CISC (programas em assembly e memória lenta, escassa e cara) resultaram em algumas características marcantes.**
- **Formato de instruções com dois operandos (fonte, destino). Instruções do tipo Registrador/Registrador, Registrador/ Memória e Memória/Registrador.**
- **Diversos modos de endereçamento a memória, incluindo modos especiais para acesso a arrays indexados.**
- **Instruções de tamanho variável, de acordo com o modo de endereçamento.**
- **Instruções que necessitam diversos ciclos de clock.**
- **O Pentium é um exemplo de arquitetura CISC da atualidade.**

CISC – "*Complex Instruction Set Computer*"

- **Arquiteturas CISC compartilham diversas características.**
- **Lógica de decodificação de instruções complexa devido a necessidade de suporte a instruções com vários modos de endereçamento.**
- **Conjunto reduzido de registradores de uso geral, devido a existência de instruções que acessam diretamente a memória.**
- **Área reduzida no chip para lógica de decodificação de instruções, execução e armazenamento de microcódigo.**
- **Diversos registradores de uso especial – ponteiros para pilha, manipulação de interrupções, strings, entre outros.**
- **Isso facilita o projeto do hardware, porém o conjunto de instruções se torna mais complexo.**
- **Registrador de "condição" para armazenar o resultado da última operação (informando se foi igual a zero, se menor ou igual a, ...).**

CISC – "*Complex Instruction Set Computer*"

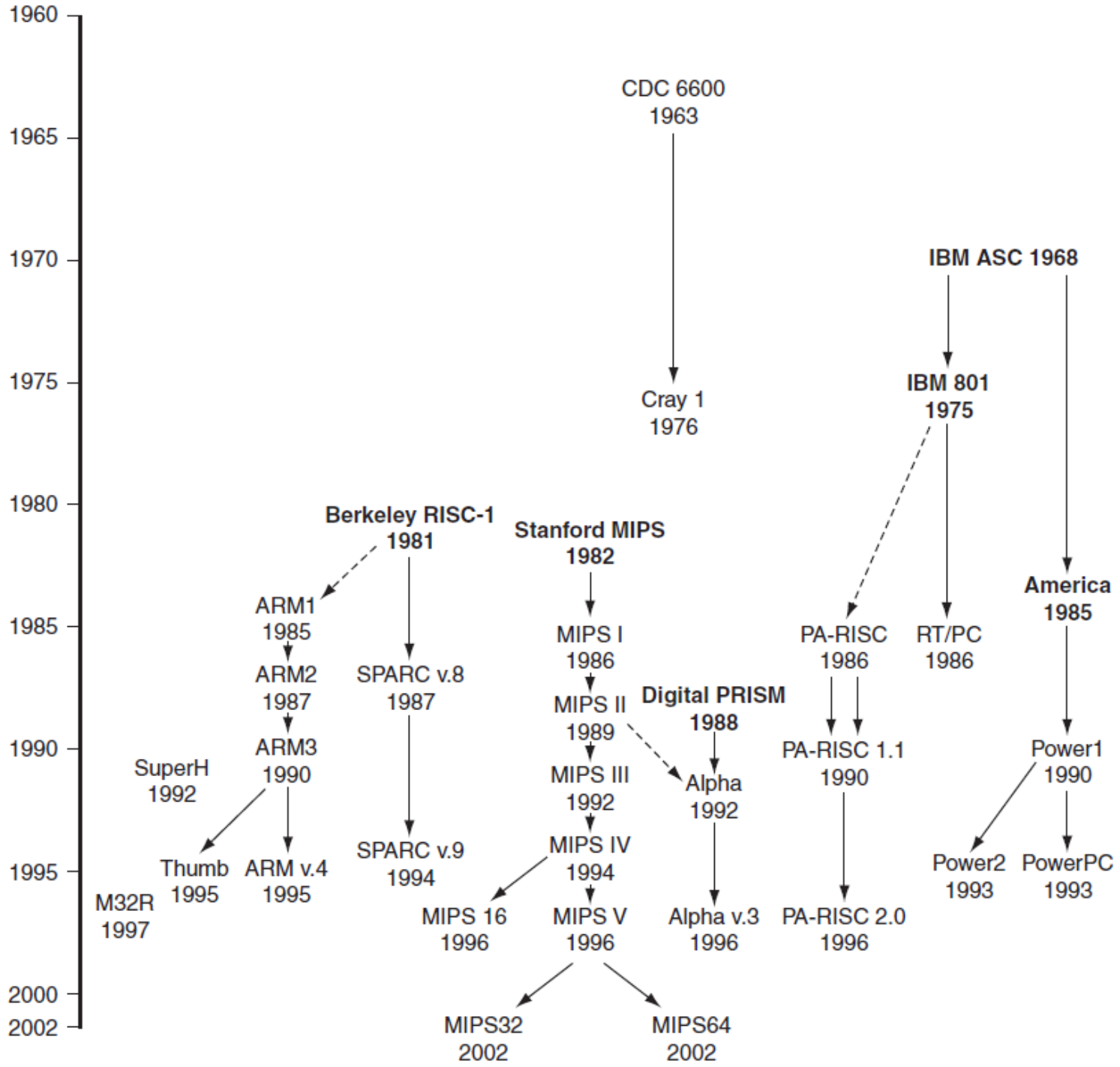
Desvantagens das arquiteturas CISC

- **Aumento na complexidade do conjunto de instruções e hardware de novas gerações de processadores, que incluem as gerações anteriores na forma de um subconjunto por questões de compatibilidade binária.**
- **Devido aos requisitos de memória, arquiteturas CISC tendem a armazenar o máximo possível de instruções, de tamanhos diferentes, evitando qualquer desperdício. Dessa forma, instruções diferentes irão necessitar de número de ciclos de clocks diferentes para execução, reduzindo a velocidade de processamento.**
- **Instruções "especializadas" não são executadas com frequência suficiente para justificar sua existência. Apenas 20% do total de instruções são utilizadas em um programa.**
- **A atualização dos flags de condição realizada por diversas instruções representa custo de processamento, e o programador precisa lembrar de verificar esses flags antes que a próxima instrução seja os altere.**

RISC – "*Reduced Instruction Set Computer*"

- **Processadores RISC possuem um número reduzido de instruções, e altamente otimizadas**
- **Primeiros projetos RISC foram desenvolvidos pela IBM (IBM 801), Stanford (MIPS) e Berkeley (RISC 1 e 2) no final dos anos 70 e início dos anos 80.**
- **Uma instrução por ciclo: processadores RISC possuem CPI = 1, devido a otimização das instruções na CPU e também pelo uso de pipeline.**
- **Pipeline: técnica que possibilita a execução em paralelo de parte (ou estágios) das instruções.**
- **Aumento na quantidade de registradores: utilizados, por exemplo, para evitar acessos seguidos a memória.**

RISC


1960
1965
1970
1975
1980
1985
1990
1995
2000
2002
CDC 6600
1963
Cray 1
1976
IBM ASC 1968
IBM 801
1975
America
1985
Berkeley RISC-1
1981
Stanford MIPS
1982
ARM1
1985
ARM2
1987
ARM3
1990
SuperH
1992
Thumb
1995
ARM v.4
1995
M32R
1997
SPARC v.8
1987
SPARC v.9
1994
MIPS I
1986
MIPS II
1989
MIPS III
1992
MIPS IV
1994
MIPS 16
1996
MIPS V
1996
MIPS32
2002
MIPS64
2002
Digital PRISM
1988
Alpha
1992
Alpha v.3
1996
PA-RISC
1986
PA-RISC 1.1
1990
PA-RISC 2.0
1996
RT/PC
1986
Power1
1990
Power2
1993
PowerPC
1993

RISC – "*Reduced Instruction Set Computer*"

Processadores CISC:

- **Número considerável de instruções**
- **Instruções complexas e eficientes**
- **Diversos modos de endereçamento para operações na memória**
- **Poucos registradores**

Processadores RISC possuem características opostas:

- **Quantidade reduzida de instruções**
- **Instruções simples, menos complexas**
- **Poucas opções de endereçamento a memória, basicamente por meio de instruções LOAD e STORE**
- **Quantidade considerável de registradores simétricos, organizados em uma tabela de registradores**

RISC – "*Reduced Instruction Set Computer*"

<u>Desvantagens do RISC</u>:

- **Comunidade RISC defende que a arquitetura é rápida e econômica, sendo a escolha ideal para os computadores do futuro**
- **Porém, ao simplificar o hardware, arquiteturas RISC transferem uma grande responsabilidade para o software**
- **Com os avanços tecnológicos, arquiteturas não RISC acabam se tornado também rápidas e econômicas, vale a pena o esforço a nível de software imposto pelas arquiteturas RISC?**

CISC e RISC

- **Implementações CISC e RISC vem se tornando cada vez mais similares**
- **Arquiteturas RISC da atualidade possuem um número de instruções equivalente as arquiteturas CISC de gerações anteriores**
- **Com o aumento da velocidade da tecnologia atual, arquiteturas CISC passaram a executar mais de uma instrução por ciclo, utilizando pipeline**
- **Com o aumento da densidade de transistores em um chip, arquiteturas RISC passaram a incorporar instruções mais complexas, semelhantes as CISC**
- **Com esses avanços tecnológicos, CISC e RISC passaram a possuir diversas similaridades, e a distinção entre as mesmas deixa de ser tão relevante**
- **Porém, apesar do aumento no conjunto de instruções, RISC continua utilizando instruções de um ciclo, com um grande número de registradores. Além disso, continua utilizando apenas instruções LOAD/STORE para acesso a memória.**

CISC e RISC

CISC	RISC
Ênfase no hardware	Ênfase no software
Instruções complexas multi-ciclo	Instruções simples de um ciclo (pipeline)
Memória para memória: "LOAD" e "STORE" incorporados nas instruções	Registrador para registrador: "LOAD" e "STORE" são instruções independentes
Binários (executáveis) reduzidos, alta taxa de ciclos por segundo	Binários (executáveis) longos, baixa taxa de ciclos por segundo
Transistores usados para armazenar instruções complexas	Transistores utilizados na implementação de registradores

CISC e RISC

Equação de desempenho:

$$\text{Tempo de CPU} = \frac{\text{segundos}}{\text{programa}} = \frac{\text{instruções}}{\text{programa}} \times \frac{\text{ciclos}}{\text{instrução}} \times \frac{\text{segundos}}{\text{ciclo}}$$

CISC e RISC

Equação de desempenho:

$$\text{Tempo de CPU} = \frac{\text{segundos}}{\text{programa}} = \frac{\text{instruções}}{\text{programa}} \times \frac{\text{ciclos}}{\text{instrução}} \times \frac{\text{segundos}}{\text{ciclo}}$$

- **Arquitetura RISC diminui tempo de execução ao reduzir o número de *ciclos por instrução* (instruções simples são decodificadas mais rapidamente)**

CISC e RISC

Equação de desempenho:

$$\text{Tempo de CPU} = \frac{\text{segundos}}{\text{programa}} = \frac{\text{instruções}}{\text{programa}} \times \frac{\text{ciclos}}{\text{instrução}} \times \frac{\text{segundos}}{\text{ciclo}}$$

- Arquitetura RISC diminui tempo de execução ao reduzir o número de *ciclos por instrução* (instruções simples são decodificadas mais rapidamente)
- **Arquitetura CISC diminui tempo de execução ao reduzir o número de instruções em um programa**

CISC e RISC

CISC

```
mov ax, 10
mov bx, 5
mul bx, ax
```

RISC

```
        mov ax, 0
        mov bx, 10
        mov cx, 5
Inicio: add ax, bx
        loop Inicio
```

CISC:

(2 movs x 1 ciclo) + (1 mul x 30 ciclos) = 32 ciclos

RISC:

(3 movs x 1 ciclo) + (5 adds x 1 ciclo) + (5 loops x 1 ciclo) = 13 ciclos

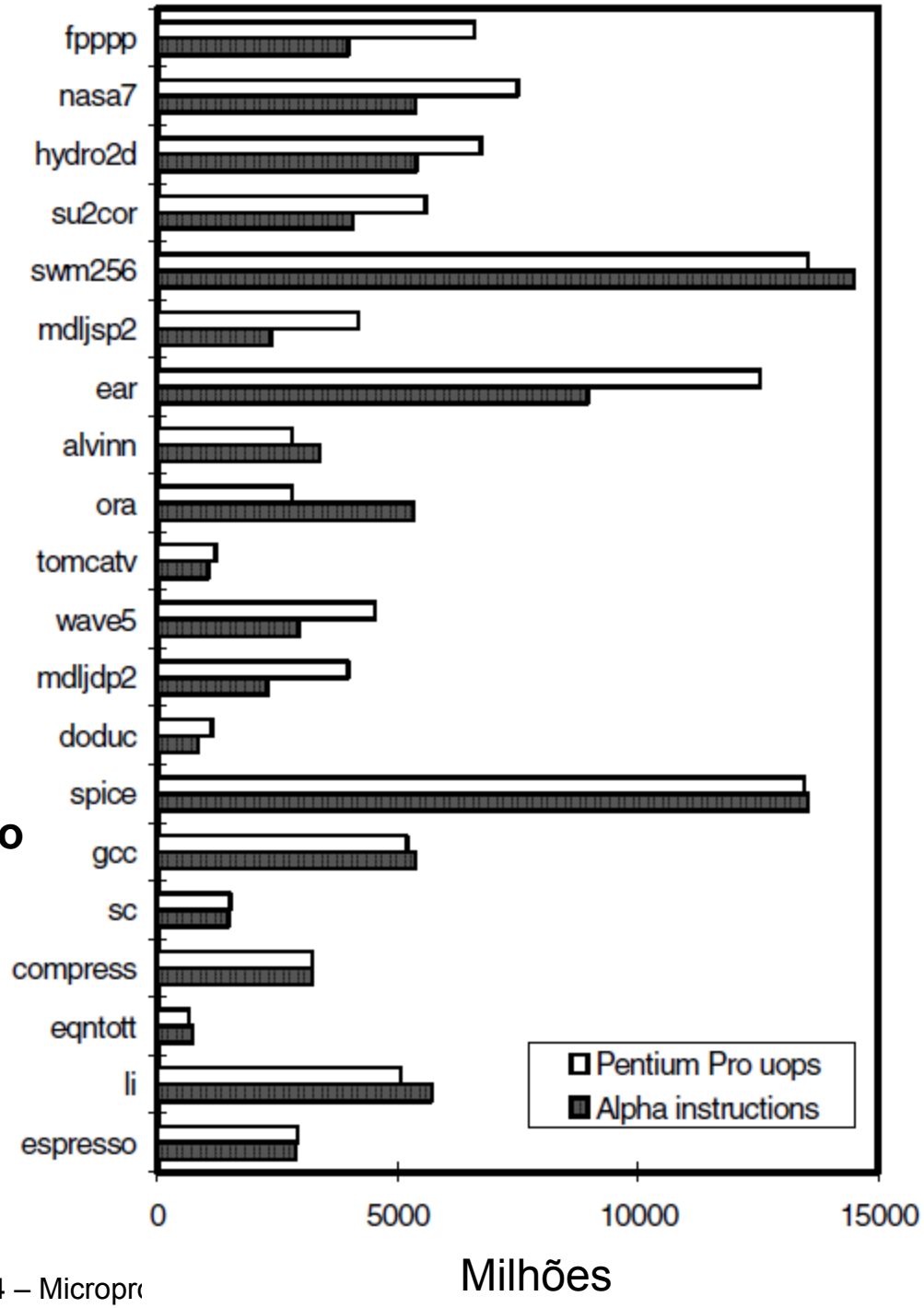
CISC e RISC

- **Arquitetura Intel IA32 – CISC de sucesso**
- **Alto volume de fabricação de chips**
- **Compatibilidade binária com enorme quantidade de software legado padrão IBM-PC**
- **Conversão interna CISC para RISC – aumenta eficiência do pipeline**
- **Escala suficiente para suportar todo o hardware extra**

CISC e RISC

Comparação entre CISC (Alpha) e RISC (Pentium Pro) no SPEC

- **Pentium Pro converte instruções CISC para RISC, *on the fly*, gerando *uops*.**
- **Para esse tipo de conversão em hardware, e "por instrução", espera-se um número maior de uops do que o gerado por um compilador.**
- **Para benchmarks de inteiros e para o spice (menor conteúdo de FP), o número de uops é próximo ao de instruções RISC.**
- **Em benchmarks FP, RISC gera menos instruções, exceto para *ora* onde Alpha precisa de diversas instruções para calcular SQRT.**



Arquitetura ideal?

- Soluções híbridas
  - Core RISC com interface CISC
- ISA desejado
  - Meio termo entre RISC e CISC
  - Poucas instruções complexas, cuidadosamente escolhidas e úteis