

ELF52 - Sistemas Microcontrolados

Arquitetura

Professor:

Prof. Marcos Eduardo

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Microcontroladores

Microprocessador



- O que é?
- Para que serve?
- Eu já vi algum?
- Qual o tamanho?
- É caro?
- Tem no Netflix?

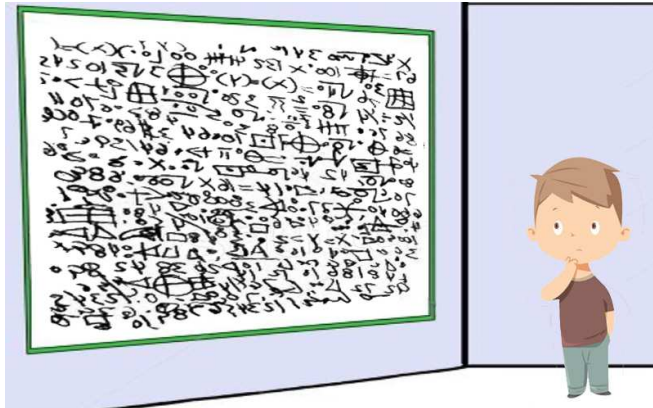
Microprocessador

- Analogia: vamos fazer um bolo!



Microprocessador

- Do que precisamos?



Microprocessador

- Do que precisamos?



- Livro de receitas → passo-a-passo de como fazer (rotina);
- Definir o sabor;
- Acessar a receita do livro;
- Descobrir os ingredientes;
- Ir até a página correspondente da receita (software):
 - "XYZ" → (1kg de bolo)

Microprocessador

- Do que precisamos?



- Livro de receitas → passo-a-passo de como fazer (rotina);
- Definir o sabor;
- Acessar a receita do livro;
- Descobrir os ingredientes;
- Ir até a página correspondente da receita (software):
 - "XYZ" → (1kg de bolo)

Microprocessador

- Qual é a nossa atitude neste instante?
 - Resp.: Ler a receita!
- Analisando o caso, qual elemento corresponde à memória?
 - Memória ROM (*Read-Only Memory*).

Microprocessador

- Qual é a nossa atitude neste instante?
 - Resp.: Ler a receita!
- Analisando o caso, qual elemento corresponde à memória?
 - Memória ROM (*Read-Only Memory*).

Microprocessador

- Receita (Página "XYZ"):
 - 3 xícaras de farinha;
 - 2 xícaras de açúcar;
 - Cobertura: (vide página "ABC").
- **Por que a cobertura não está escrita aqui também?**
 - Resp: Várias receitas utilizam! (corresponde a uma subrotina)

Microprocessador

- Receita (Página "XYZ"):
 - 3 xícaras de farinha;
 - 2 xícaras de açúcar;
 - Cobertura: (vide página "ABC").
- **Por que a cobertura não está escrita aqui também?**
 - **Resp: Várias receitas utilizam! (corresponde a uma subrotina)**

Microprocessador

- Sobre esta "subrotina", o que devemos fazer?
 - Vamos fazer a cobertura e depois voltamos e damos continuidade a nossa receita.
- Sequência:
 - Executar XYZ;
 - Executar ABC;
 - Retornar XYZ e continuar.

Microprocessador

- Sobre esta "subrotina", o que devemos fazer?
 - Vamos fazer a cobertura e depois voltamos e damos continuidade a nossa receita.
- Sequência:
 - Executar XYZ;
 - Executar ABC;
 - Retornar XYZ e continuar.

Microprocessador

- Se se nós quisermos mudar as quantidades como, por exemplo, fazer 2 kg de bolo?
 - Resp: Podemos anotar tais informações em uma folha de rascunho
→ Memória RAM (*Random Access Memory*).
- Ou seja, temos que:
 - Cozinheiro - microprocessador (Hardware);
 - Ingredientes - eletrônica (Hardware);
 - Livro de receitas - ROM (Software);
 - Rascunho - RAM (Software).

Microprocessador

- Se se nós quisermos mudar as quantidades como, por exemplo, fazer 2 kg de bolo?
 - Resp: Podemos anotar tais informações em uma folha de rascunho
→ Memória RAM (*Random Access Memory*).
- Ou seja, temos que:
 - Cozinheiro - microprocessador (Hardware);
 - Ingredientes - eletrônica (Hardware);
 - Livro de receitas - ROM (Software);
 - Rascunho - RAM (Software).

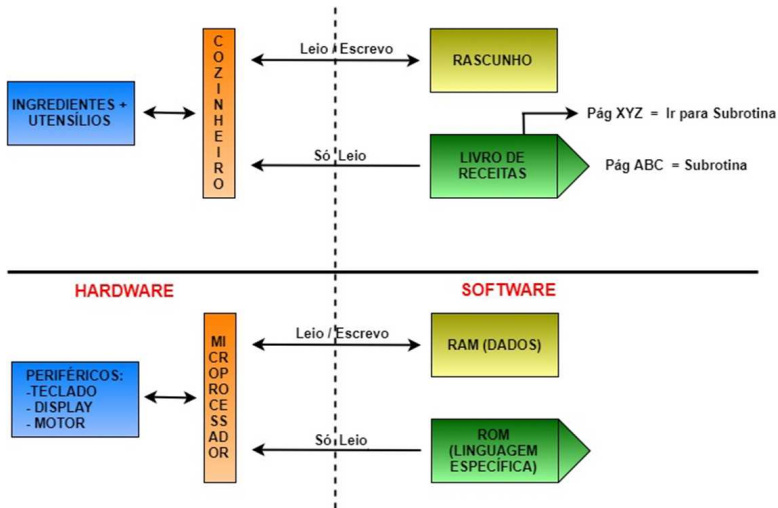
Microprocessador

- Qual a diferença entre o microprocessador e o cozinheiro?

Microprocessador

- Qual a diferença entre o microprocessador e o cozinheiro?
 - Resp: Vocabulário.
- Ou seja, precisamos entender o vocabulário do microprocessador.

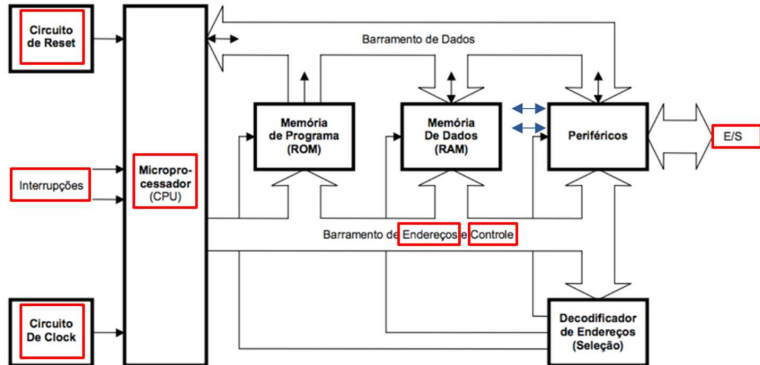
Microprocessador



Microprocessador

- O **microprocessador** é um componente eletrônico capaz de:
 - Ler as **instruções** da ROM que devem ser executadas;
 - Armazenar temporariamente informações na RAM, que são de uso das **instruções**;
 - Lidar com o barramento de dados e endereços.
 - Se comunicar com todos os outros componentes (teclado, impressora, LCD, etc).

Microprocessador



Funções dos Componentes

- **Barramento de Endereços:** Selecionar com qual posição de memória ou periférico deseja se comunicar;
- **Controle:** Permitem o microprocessador acionar a RAM e a ROM em um certo tempo específico e vice-versa (ligar/desligar);
- **Barramento de E/S (I/O):** Comunicação com o mundo externo;
- **CPU (Engenheiro formado e casado):** Se comunicar e acionar todos os barramentos, obedecendo a ROM.

Funções da CPU

- Buscar instruções continuamente na ROM;
- Executar essas instruções;
- Executar funções lógicas;
- Executar funções aritméticas;
- Executar transferências de dados (internas e externas);
- Executar comparação de dados → tomada de decisão;
- *Meio "Tapada", só entende binário!*

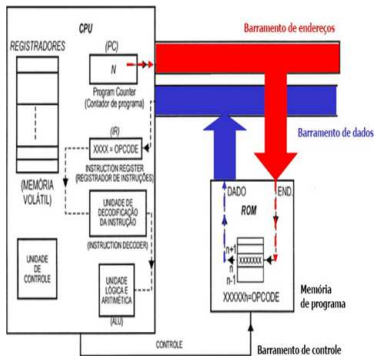
Funções da CPU

- Buscar instruções continuamente na ROM;
- Executar essas instruções;
- Executar funções lógicas;
- Executar funções aritméticas;
- Executar transferências de dados (internas e externas);
- Executar comparação de dados → tomada de decisão;
- *Meio "Tapada", só entende binário!*

Funções dos Componentes

- **Oscilador:** Tarefas internas e externas sincronizadas e com uma velocidade predeterminada;
- **Reset:** Iniciar as rotinas e realizar a leitura no primeiro endereço;
- **Interrupções:** Pinos de acesso externo que interrompem o microprocessador;
- **Registradores:** Armazenamento de alta velocidade dentro do microprocessador;
- **Barramento:** Conjunto de fios utilizados para passar informação entre os módulos.

Microprocessador



- 1 O endereço de PC vai para BUS;
- 2 Ativa o sinal de controle da ROM;
- 3 Ciclo de busca:
 - Lê da ROM;
 - No endereço dado pelo PC;
 - Lê/transmite pelo BUS dados.
- 4 Instrução carregada e armazenada no IR;
- 5 Incrementa o PC (próxima inst.);
- 6 Inicia o ciclo de execução.

Microprocessador x Microcontrolador

- Até então, falamos apenas sobre microprocessadores.
- Mas, e microcontroladores, o que são?
- Qual a diferença entre os dois?
 - Vou ter que pensar, professor?
 - Deixa eu dormir aqui!

Microprocessador x Microcontrolador

- Até então, falamos apenas sobre microprocessadores.
- Mas, e microcontroladores, o que são?
- Qual a diferença entre os dois?
 - Vou ter que pensar, professor?
 - Deixa eu dormir aqui!



Microprocessador x Microcontrolador

- O hardware interno difere:
 - Microprocessador (-)
 - Microcontrolador (+)
- Microprocessador contém:
 - IR, PC, ALU, etc.
- Microcontrolador contém:
 - Tudo o que o microprocessador tem;
 - + Periféricos.

Microprocessador x Microcontrolador

- O hardware interno difere:
 - Microprocessador (-)
 - Microcontrolador (+)
- Microprocessador contém:
 - IR, PC, ALU, etc.
- Microcontrolador contém:
 - Tudo o que o microprocessador tem;
 - + Periféricos.

Periféricos

- Periféricos:
 - Memórias;
 - Temporizadores;
 - Portas de Entrada/Saída;
 - Teclados;
 - *Displays*;
 - Impressoras;
 - Sensores;
 - Motores.

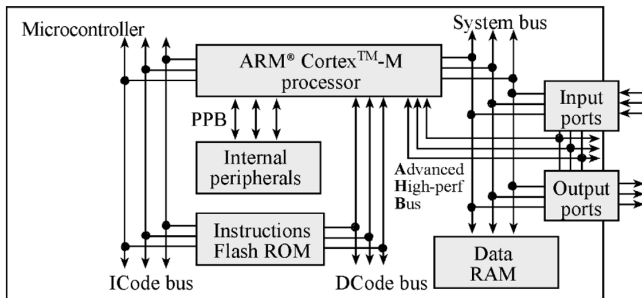
Arquitetura ARM Cortex M4

Arquitetura ARM Cortex M4

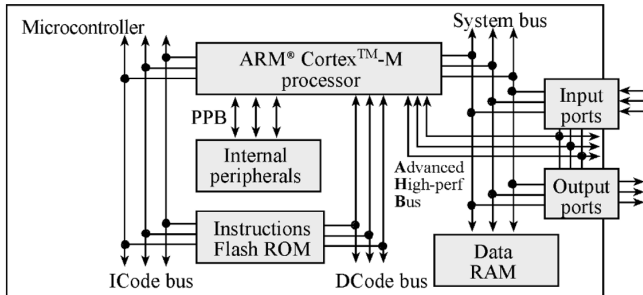
- Processador ARM Cortex-M4:
 - ARMv7-ME.
- Arquitetura Harvard:
 - Diferentes barramentos para instruções e dados.

Arquitetura ARM Cortex M4

- Processador ARM Cortex-M4:
 - ARMv7-ME.
- Arquitetura Harvard:
 - Diferentes barramentos para instruções e dados.



Arquitetura ARM Cortex M4



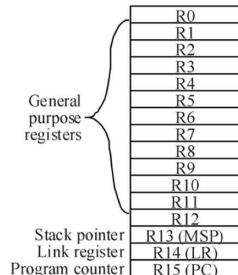
- Barramentos:

- ICode: busca *opcodes* da ROM;
- DCode: lê dados constantes da ROM;
- System: lê/escreve dados da RAM ou I/O, busca *opcode* da RAM;
- PPB: lê/escreve dados de periféricos internos;
- AHB: lê/escreve dados de portas I/O de alta velocidade.

Registradores (32 bits)

- Registradores R0 a R15:

- 13 de uso geral;
- R13 → MSP (*Main Stack Pointer*)
 - Aponta para o topo da pilha.
- R14 → LR (*Link Register*)
 - Guarda o endereço de retorno para funções.
- R15 → PC (*Program Counter*)
 - Aponta para a próxima instrução a ser buscada da memória;
 - Processador busca a instrução que está no PC e incrementa o PC.

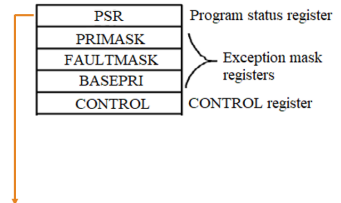


Registradores (32 bits)

• Registradores Especiais:

• PSR (*Program Status Register*):

- APSR → *Application*;
- IPSR → *Interrupt*;
- EPSR → *Execute*.



Bits do APSR	Indicação
N	Resultado é negativo
Z	Resultado é zero
V	<i>Overflow</i> com sinal
C	<u>Carry</u> ou <i>overflow</i> sem sinal
Q	Saturação



Memória

- Endereça até 4GB de memória:
 - 32 bits;
 - Em geral:
 - Flash ROM → começa em 0x0000.0000
 - RAM → começa em 0x2000.0000
 - I/O → entre 0x4000.0000 e 0x5FFF.FFFF
 - I/O PPB Interno → entre 0xE000.0000 e 0xE004.1FFF

Memória

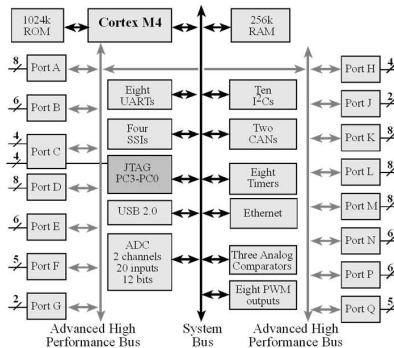
- Endereça até 4GB de memória:
 - 32 bits;
 - Em geral:
 - Flash ROM → começa em 0x0000.0000
 - RAM → começa em 0x2000.0000
 - I/O → entre 0x4000.0000 e 0x5FFF.FFFF
 - I/O PPB Interno → entre 0xE000.0000 e 0xE004.1FFF

Memória

- Para o microcontrolador TM4C1294, a memória tem a seguinte composição:

1024k Flash ROM	0x0000.0000 0x000F.FFFF
256k RAM	0x2000.0000 0x2003.FFFF
I/O	0x4000.0000 0x400F.FFFF
I/O PPB Interno	0xE000.0000 0xE004.1FFF

Periféricos TM4C1294



- Cada *port* tem um número de pinos:
 - Exemplo: PA0, PA1, PA2, PA3, PA4, PA5, PA6, PA7.

Exercícios

- Abrir o datasheet do TM4C1294
 - Verificar a seção dos Registradores (páginas 85 a 99);
 - Verificar a seção das Memórias (seção 2.4).

Paradigma CISC x RISC

CISC

- **Complex Instruction Set Computer:**
 - Conjunto de instruções inicialmente simples;
 - Avanços tecnológicos permitiram a fabricação de computadores com mais transistores e menor custo;
 - Projetistas optaram por conjuntos de instruções cada vez mais complexos;
 - **Intenção:** reduzir a distância semântica entre *Assembly* e linguagens de alto nível.

CISC

- **Complex Instruction Set Computer:**
 - Conjunto de instruções inicialmente simples;
 - Avanços tecnológicos permitiram a fabricação de computadores com mais transistores e menor custo;
 - Projetistas optaram por conjuntos de instruções cada vez mais complexos;
 - **Intenção:** reduzir a distância semântica entre *Assembly* e linguagens de alto nível.

CISC

- **Complex Instruction Set Computer:**

- Instruções com elevado grau semântico;
- Elevado número de modos de endereçamento:
 - Ex: endereçamento indireto em memória.
- Elevado número de ciclos de *clock* por instrução → redução da frequência de *clock*;
- Menor número de instruções por programa → menor uso de memória de código;
- Decodificação através de microcódigo → dificulta/impossibilita o uso de pipeline.

CISC

- **Complex Instruction Set Computer:**

- Instruções com elevado grau semântico;
- Elevado número de modos de endereçamento:
 - Ex: endereçamento indireto em memória.
- Elevado número de ciclos de *clock* por instrução → redução da frequência de *clock*;
- Menor número de instruções por programa → menor uso de memória de código;
- Decodificação através de microcódigo → dificulta/impossibilita o uso de pipeline.

RISC

- **Reduced Instruction Set Computer:**

- Instruções simples que executam rápido;
- Elevado número de registradores de uso geral;
- Decodificação de instruções com lógica combinacional (tabela);
- Execução utilizando pipeline → um ciclo de *clock* por instrução.

RISC

- **Reduced Instruction Set Computer:**
 - Instruções simples que executam rápido;
 - Elevado número de registradores de uso geral;
 - Decodificação de instruções com lógica combinacional (tabela);
 - Execução utilizando pipeline → um ciclo de *clock* por instrução.

RISC

- **Reduced Instruction Set Computer:**

- Regularidade de tempo de execução;
- Regularidade de tamanho de instrução;
- Redução da área de silício e tempo de projeto;
- **Efeito final:** melhor desempenho, apesar do número de instruções ser maior por programa.

RISC

- **Reduced Instruction Set Computer:**
 - Regularidade de tempo de execução;
 - Regularidade de tamanho de instrução;
 - Redução da área de silício e tempo de projeto;
 - **Efeito final:** melhor desempenho, apesar do número de instruções ser maior por programa.

RISC x CISC

RISC	CISC
Conjunto de instruções reduzido	Conjunto de instruções extenso
Instruções semanticamente simples	Instruções semanticamente complexas
Instruções de tamanho fixo	Instruções de tamanho variável
Decodificação simplificada (tabela)	Decodificação complexa (microcódigo)
Execução regular	Cada instrução executa à sua maneira
Instruções requerem o mesmo número de ciclos de <u>clock</u> para executar	Grande variação no número de ciclos de <u>clock</u> por instrução
Possibilita o uso de pipeline	Extremamente difícil/impossível o uso de pipeline

Pipeline (3 Estágios)

- Busca (*Fetch*):
 - Busca da instrução na memória.
- Decodificação (*Decode*):
 - Decodificação dos registradores usados na instrução.
- Execução (*Execute*):
 - Leitura de registradores;
 - Operações lógicas, aritméticas e de deslocamento;
 - Escrita em registradores.

Pipeline (3 Estágios)



Pipeline (3 Estágios)

- Situação Ideal:

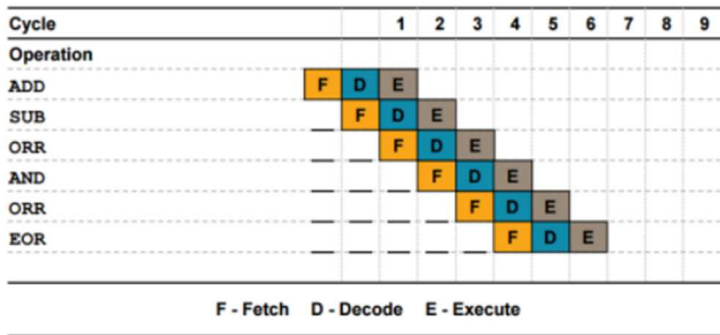
Cycle		1	2	3	4	5	6	7	8	9
Operation										
ADD		F	D	E						
SUB			F	D	E					
ORR				F	D	E				
AND					F	D	E			
ORR						F	D	E		
EOR							F	D	E	

F - Fetch D - Decode E - Execute

- Todas as operações realizadas em registradores → 6 instruções em 6 ciclos de *clock* (ARM Cortex-M4).

Pipeline (3 Estágios)

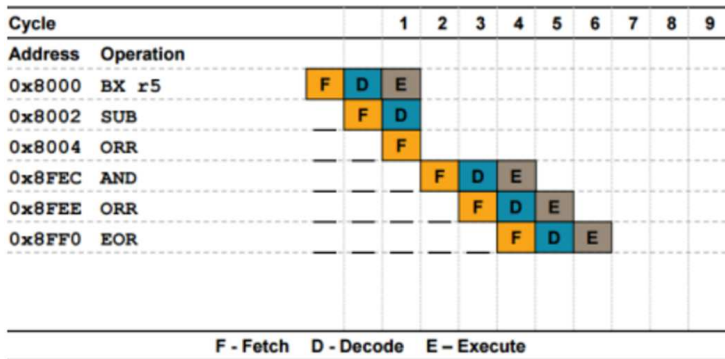
- Situação Ideal:



- Todas as operações realizadas em registradores → 6 instruções em 6 ciclos de *clock* (ARM Cortex-M4).

Pipeline (3 Estágios)

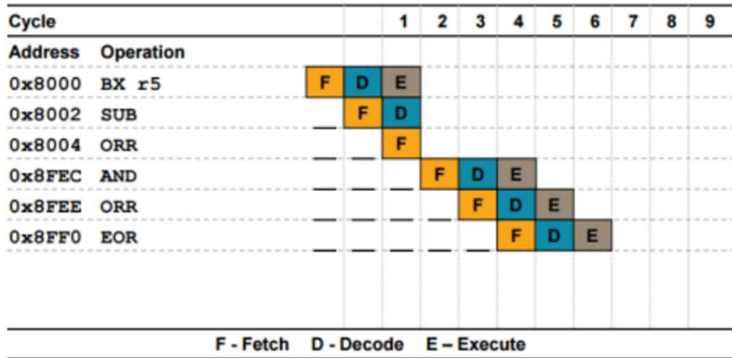
- Efeito de Saltos:



- Pior caso: salto indireto, 3 ciclos de *clock* para completar o salto (ARM Cortex-M4).

Pipeline (3 Estágios)

- Efeito de Saltos:



- Pior caso: salto indireto, 3 ciclos de *clock* para completar o salto (ARM Cortex-M4).

Dúvidas?