**Exercício 1: Função do PC e impacto de sua atualização incorreta**

O **PC (Program Counter)** é um registrador que armazena o endereço da próxima instrução a ser buscada na memória. Em cada ciclo de instrução, ele é atualizado para apontar para a próxima instrução a ser executada.

Se o **PC não for atualizado corretamente**, pode ocorrer:

* **Loop infinito**: Se o PC ficar preso em um mesmo endereço, a CPU continuará executando a mesma instrução indefinidamente.
* **Execução incorreta**: Se o PC apontar para um endereço errado, a CPU pode buscar e executar uma instrução inválida.
* **Erro de segmentação**: Se o PC apontar para uma área inválida da memória, pode causar falhas no sistema.

**Exercício 2: Papel do MBR no processamento de dados**

O **MBR (Memory Buffer Register)** atua como um intermediário entre a memória e o processador. Ele armazena temporariamente os dados que estão sendo transferidos entre a CPU e a memória.

Funções do **MBR**:

* **Armazena instruções e dados** lidos da memória antes de serem processados.
* **Envia dados** da CPU para a memória quando há uma operação de escrita.
* **Facilita a comunicação** entre o barramento de dados e os registradores internos do processador.

**Exercício 3: Ciclo de instrução para uma operação de multiplicação**

Suponha que o PC contém o endereço da instrução de multiplicação:

1. **Busca da Instrução**
   * O PC envia o endereço da instrução para o **MAR (Memory Address Register)**.
   * A memória transfere a instrução para o **MBR**.
   * O MBR passa a instrução para o **IR (Instruction Register)**.
   * O PC é atualizado para a próxima instrução.
2. **Decodificação**
   * A **UC (Unidade de Controle)** interpreta a instrução e identifica que é uma **multiplicação**.
   * Determina quais registradores contêm os operandos.
3. **Busca dos operandos**
   * Se os operandos estão na memória, a CPU acessa os endereços indicados.
   * Se os operandos estão nos registradores, a ULA os recebe diretamente.
4. **Execução da multiplicação**
   * A **ULA (Unidade Lógica e Aritmética)** realiza a operação de multiplicação.
5. **Armazenamento do resultado**
   * O resultado da multiplicação é armazenado no registrador de destino ou na memória.

**Exercício 4: Tempo de ciclo de clock**

A frequência do processador é de **4 GHz**, ou seja, **4.000.000.000 Hz**. O tempo de ciclo de clock é dado por:

Tempo de ciclo = 1/Frequência do clock

Tempo de ciclo= 1/4.000.000.0001 = 0,25ns (nanosegundos)

**Influência no desempenho:**

* **Menor tempo de ciclo** = **Maior desempenho**, pois o processador executa mais instruções por segundo.
* **Eficiência depende da arquitetura** = Mesmo com um clock alto, fatores como o número de ciclos necessários por instrução afetam o desempenho.

### ****Exercício 5: Execução da instrução**** ADD R1, [A3]

#### **(a) Etapas do ciclo de instrução**

1. **Busca da Instrução**
   * O PC aponta para o endereço da instrução ADD R1, [A3].
   * O **MAR** recebe o endereço e solicita a instrução da memória.
   * A instrução é carregada no **MBR** e transferida para o **IR**.
2. **Decodificação**
   * A UC interpreta a instrução ADD R1, [A3], que significa **somar o valor armazenado na posição A3 ao registrador R1**.
   * Identifica que A3 é um endereço de memória.
3. **Busca do operando**
   * O endereço **A3** é lido e contém o valor **50**.
   * A CPU acessa a memória no endereço **50**, que contém o valor **25**.
4. **Execução**
   * A **ULA** realiza a operação:   
     R1=R1+Mem[50] R1=10+25=35R1 = 10 + 25 = 35R1=10+25=35
5. **Armazenamento**
   * O novo valor de **R1** (**35**) é atualizado no registrador.

#### **(b) Microoperações**

1. **Busca da instrução**

MAR ← PC

Ler Memória

MBR ← Mem[PC]

IR ← MBR

PC ← PC + 1

1. **Decodificação**

Decodificar IR

1. **Busca do operando**

MAR ← A3

Ler Memória

MBR ← Mem[A3] (MBR recebe 50)

MAR ← MBR

Ler Memória

MBR ← Mem[50] (MBR recebe 25)

1. **Execução**

R1 ← R1 + MBR (R1 recebe 10 + 25)

1. **Armazenamento**

Armazenar R1 no registrador

#### **(c) Valor final em R1**

* **Inicialmente:** R1 = **10**
* **Valor lido da memória:** Mem[50] = **25**
* **Soma realizada:** 10+25=35
* **Valor final armazenado em R1:** **35**