MICROARQUITETURA DO WR80

1. **Busca da Instrução (1ª estágio do pipeline)**
    - **1.1. Contador de Estágios (2 bits)**
        - 1.1.1. Contabilizar 4 números binários de estágio
        - 1.1.2. Estouro de 4 estágios completa 1 ciclo de CPU
    - **1.2. Contador de Programa (12 bits)**
        - 1.2.1. A cada ciclo, contabilizar +1 em PC
        - 1.2.2. Contabilizar até 4096 bytes sequencialmente
        - 1.2.3. Alterar dinamicamente em saltos (Desvios)
    - **1.3. Seletor de memória**
        - 1.3.1. Seleciona memória no endereço <PC>
        - 1.3.2. Ler instrução e armazena em IR
2. **Decodificador de instrução (2ª estágio do pipeline)**
    - ***2.1. Decodificador de operações***
        - 2.1.1. 1ª Multiplexador – 3 bits + 1 bit:
            - Identificar “parte” da instrução (código de operação - opcode) do registrador IR
            - 8 opcodes/operações possíveis
            - Selecionar circuito de operação
    - ***2.2. Decodificador de dados/registradores***
        - 2.2.1. 2ª Multiplexador – 2 bits:
            - Identificar “parte” da instrução (endereço da instrução) do registrador IR
            - Endereçar 4 registradores possíveis
3. **Operações Lógico/Aritméticas (3ª estágio do pipeline)**
    - **3.1. Operações lógicas**
        - 3.1.1. Operação AND - 0000
        - 3.1.2. Operação OR - 0001
        - 3.1.3. Operação NOT - 0010
        - 3.1.4. Operação XOR - 0011
    - **3.2. Operações aritméticas**
        - 3.2.1. Operação ADD - 0100
        - 3.2.2. Operação SUB - 0101
    - **3.3. Operações de movimento**
        - 3.3.1. Operação ST - 0110
        - 3.3.2. Operação LD - 0111
        - 3.3.3. Operação IN - 1000
        - 3.3.4. Operação OUT - 1001
    - **3.4. Operações de Deslocamento**
        - 3.4.1. Operação SHR - 1010
        - 3.4.2. Operação SHL - 1011
    - **3.5. Operações de Comparação & Saltos**
        - 3.5.1. Operação BT - 1100
        - 3.5.2. Operação JC - 1101
        - 3.5.3. Operação JZ - 1110
        - 3.5.4. Operação JP - 1111

4. Registradores/Armazenamento (4ª estágio do pipeline)

4.1. Registradores de dados/usuários

4.1.1. R0 – Registrador 0 – 00
4.1.2. R1 – Registrador 1 – 01
4.1.3. R2 – Registrador 2 – 10
4.1.4. R3 – Registrador 3 – 11

4.2. Registradores de portas E/S

4.2.1. P0 – Porta 0 – 00 (Endereço)
4.2.2. P1 – Porta 1 – 01 (Endereço)
4.2.3. P2 – Porta 2 – 10 (Dados)
4.2.4. P3 – Porta 3 – 11 (Dados)

4.3. Registradores Internos (Overhead)

4.3.1. IR – Registrador de Instrução (Instrução)
    4.3.1.1. OR – Registrador de Opcode
    4.3.1.2. AR – Registrador de Endereço/Dado
4.3.2. DR – Registrador Acumulador
4.3.3. SR – Registrador de STATUS (4 bits)
- Bit Carry (0) = Bit C
- Bit Zero (1) = Bit Z
- +2 bits reservados (Interrupção, Modos, etc. etc..)

4.3.4. PC – Contador de Programa
4.3.5. Offset – Registro de deslocamento de saltos
4.3.6. RR – Registrador de Resultados da ULA

PRIMEIROS ESTÁGIOS

1ª estágio – buscar instrução – fetching
- Ler um byte da memória = instrução – 0100 0001

IBR = 01000001

2ª estágio – dividir em duas partes a instrução
- IBR -> IR -> 0100
- IBR -> AR -> 0001
- Selecionar Operação do IR
- Operar o conteúdo apontado por AR

R1 = 9

3ª – estágio – R1 = 9 + DR = (R1 + DR) = 12
- Executando operação dos dados

4ª estágio – Escrita dos dados/resultado
- Na memória RAM -> RAM = 12
- Nos registradores -> DR
- Estados, (Ex.: 18 -> CO = 1), BIT<0> de STATUS

SR = 0001

- Saltos condicionais – Verificar se é **maior que, menor que, igual, diferente**, etc..
- IFs, ELSEs, WHILEs, FORs,...

Saltos ->

JC -> Utiliza o Bit Carry (C)

JZ -> Utiliza o Bit Zero (Z)

PROGRAMA RISC EXEMPLO DO PROCESSADOR

ST 9 -> Copia 9 (Literal) para DR, DR = 9
LD R1 -> Copia de DR para R1 (Endereço), R1 = 9
ADD R1 -> Soma DR com R1 e salva em DR, DR = DR + R1

ST 3 -> Copia 3 para DR, DR = 3
LD R1 -> Copia de DR para R1, R1 = 3
ST 9 -> Copia 9 para DR, DR = 9
ADD R1 -> Soma DR com R1 e salva em DR, DR = DR + R1

Instrução monádicas -> 1 operando

DR = 9 + 3 = 12 (1100)

SUB 1 -> Subtrai DR (12) por 1, DR = DR – 1
LD R3 -> R3 = DR = 11
....
...
...

CISC – Complexo – muitas instruções (Intel, AMD...) – 400 instruções – ISA (Set de instruções do CISC, evoluções históricas, programação Assembly), STATUS -> Eflags

RISC – Simples – poucas instruções (PIC, Motorolla) – 35 instruções – Robôs, indústrias, aparelhos eletrodomésticos, aplicações científicas acadêmicas, dispositivos em geral. STATUS -> STATUS

ARM – CISC + RISC (Arquitetura de RISC Avançado – Arduino, MIPs) – 80 a 150 instruções -> STATUS