# Processador Kraken

DÉBORA BIANCA
EWELLY FABIANE

# KRAKEN

## O PROCESSADOR

1. **CONJUNTO DE INSTRUÇÕES**

O processador **Kraken** possui 8 bits, e suporta instruções de tipo R (Aritmético), tipo I (Transferência), tipo J (Jump), cada uma consequentemente com 8 bits.

| Tipo de instrução | código de operação | Reg1 | Reg2 |
|---|---|---|---|
| Tipo R | opcode | rs | rt |
| | 4 bits | 2 bits | 2 bits |
| Tipo I | opcode | rs | rt (endereço) |
| | 4 bits | 2 bits | 2 bits |
| Tipo J | opcode | rt (endereço) | |
| | 4 bits | 4 bits | |

# KRAKEN
## O PROCESSADOR

### 1.2 OPCODE

O **Opcode** das instruções possui 4 bits, sendo assim obtemos um total de 16 Opcodes (0-3), entretanto, apenas **8 instruções** são distribuídas entre as instruções.

| Opcode | Nome | Formato | Descrição | Exemplo |
|---|---|---|---|---|
| **0000** | add | R | Soma | **add** $s1, $s1,$s2 |
| **0001** | sub | R | Subtração | **sub** $s1, $s1, $s2 |
| **0010** | mult | R | Multiplicação | **mult** $s1,$s2 |
| **0011** | lw | I | Load | **lw** $s1,100 |
| **0100** | sw | I | Store | **sw** $s1,100 |
| **0101** | j | J | Jump | **j** 100 |
| **0110** | beq | I | Beq | **beq** $s1, 100 |
| **0111** | and | R | And | **and** $s1, $s1,$s2 |

# KRAKEN
## O PROCESSADOR

**2. COMPONENTES**

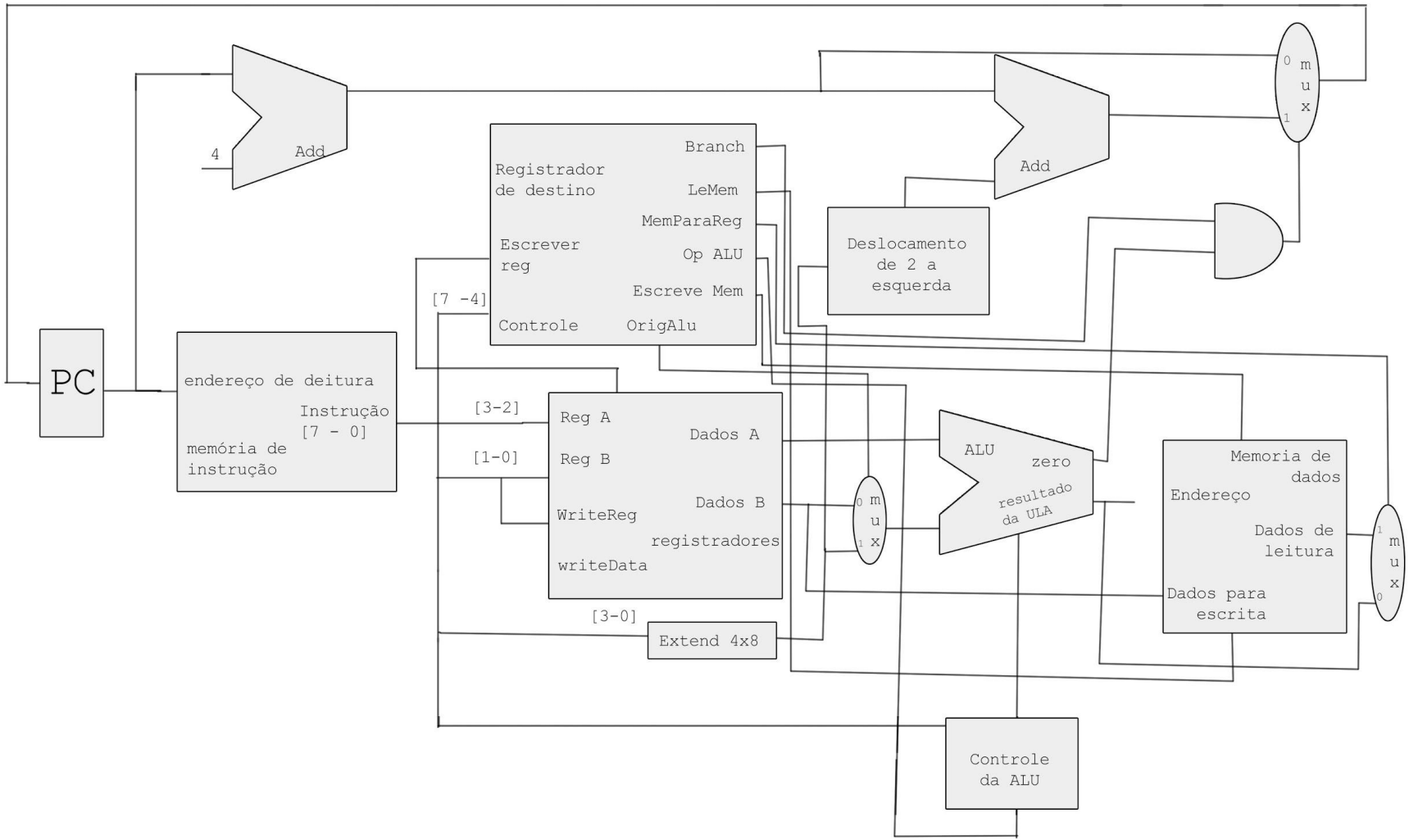
O processador é composto pelos seguintes componentes:

- ULA (Unidade de Controle Aritmético)
- Banco de Registradores
- PC (Program Counter)
- Controle
- Memória de Dados
- Memória de Instrução
- Somador
- Controle da ULA
- Multiplexador de 2 entradas
- Extensor de sinal
    - dois extensores de 2 para 8 bits
    - um extensor de 2 para 4 bits

# KRAKEN

## O PROCESSADOR

### 3. DATAPATH


4
Add
Add
0
m
u
x
1
Branch
Registrador
de destino
LeMem
MemParaReg
Escrever
reg
Op ALU
Escreve Mem
Controle
OrigAlu
Deslocamento
de 2 a
esquerda
[7 -4]
PC
endereço de deitura
Instrução
[7 - 0]
memória de
instrução
[3-2]
[1-0]
Reg A
Reg B
Dados A
Dados B
WriteReg
registradores
writeData
[3-0]
Extend 4x8
ALU
zero
resultado
da ULA
Memoria de
dados
Endereço
Dados de
leitura
Dados para
escrita
Controle
da ALU

# KRAKEN
## O PROCESSADOR

**Caminho de dados para uma instrução tipo R**

1. A instrução é buscada e o PC é incrementado
2. Dois registradores $t1 e $t2, são lidos do banco de registradores..
3. A ALU opera nos dados lidos do banco de registradores usando código de função (bits 3:0, que é o campo funct, da instrução) para gerar a função da ALU.
4. O resultado da ALU é escrito no banco de registradores usando os bits (1:0) da instrução para selecionar o registrador de destino.

# KRAKEN
## O PROCESSADOR

**Caminho de dados para uma instrução tipo Load**

1. A instrução é buscada e o PC é incrementado
2. Um valor de registrador $t2 é lido do banco de registradores.
3. A ALU calcula a soma do valor lido do banco de registradores com os 4 bits menos significativos com o sinal estendido da instrução (offset).
4. A soma da ALU é usada como endereço para memória de dados.
5. Os dados da unidade de memória são escritos no banco de registradores;o registrador de destino é fornecido pelos bits (1:0) da instrução $t1.

# KRAKEN
## O PROCESSADOR

**Caminho de dados para uma instrução branch equal**

1. Uma instrução é buscada da memória de instruções e o PC é incrementado.
2. Dois registradores $t1 e $t2, são lidos do banco de registradores e a unidade de controle principal calcula a definição das linhas de controle também durante essa etapa.
3. A ALU realiza uma subtração dos valores de dados lidos do banco de registradores. O valor de PC + 4 é somado aos 4 bits menos significativos com sinal estendido (offset) deslocados de dois para a esquerda; o resultado é o endereço de destino do desvio.
4. O resultado Zero da ALU é usado para decidir o resultado de que o somador deve ser armazenado no PC.