

Primeira Prova

1) Projete um circuito que executa a divisão inteira de dois números positivos de 8 bits. Este circuito deve implementar a divisão por subtrações repetidas. Seu projeto deve conter o circuito de dados e a máquina de estados que controla o circuito de dados, além de uma descrição textual breve da operação do circuito, que é especificado abaixo. Cada uma das variáveis de 8 bits deve ser implementada como um registrador: *ddndo* e *dvsor* são carregados pelo sinal de entrada da interface externa *inicia*, e os valores finais do quociente e do resto são disponibilizados nos registradores *quoc* e *resto* quando o sinal *pronto* for ativado pela máquina de estados que controla o divisor. [15 pontos]

$$\begin{aligned} inicia, pronto &: \mathbb{B} \\ ddndo, dvsor &: \mathbb{B}_8 \\ quoc, resto &: \mathbb{B}_8 \\ div8 : \beta \times (\beta_8 \times \beta_8) &\mapsto (\beta_8 \times \beta_8) \times \beta \\ div8(inicia, ddndo, dvsor, quoc, resto, pronto) &\equiv \\ &inicia \wedge [pronto \Rightarrow (ddndo = (dvsor \cdot quoc) + resto)] \end{aligned} \tag{1}$$

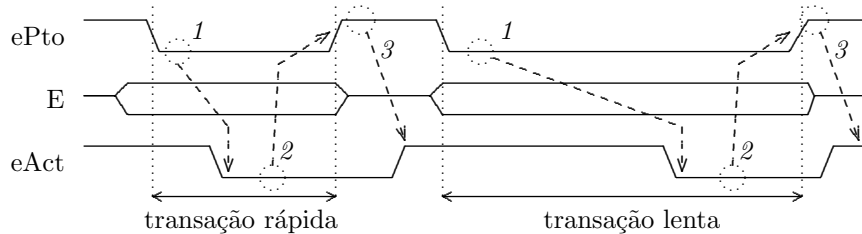
2) Traduza para *assembly* do MIPS o trecho de programa abaixo: [15 pontos]
Para facilitar a correção indique os registradores como *ri*, *rd*, etc.

```
int main(...) {
    ...
    s = reverte(fte, dst, N);
    ...
}

int reverte(char *f, char *d, int n) {
    int i;
    d = (char *) ( (int)d+n );
    for (i=0; (i < n) & (*f != '\0') ; i++)
        *d-- = *f++;
    return i;
}
```

Segunda Prova

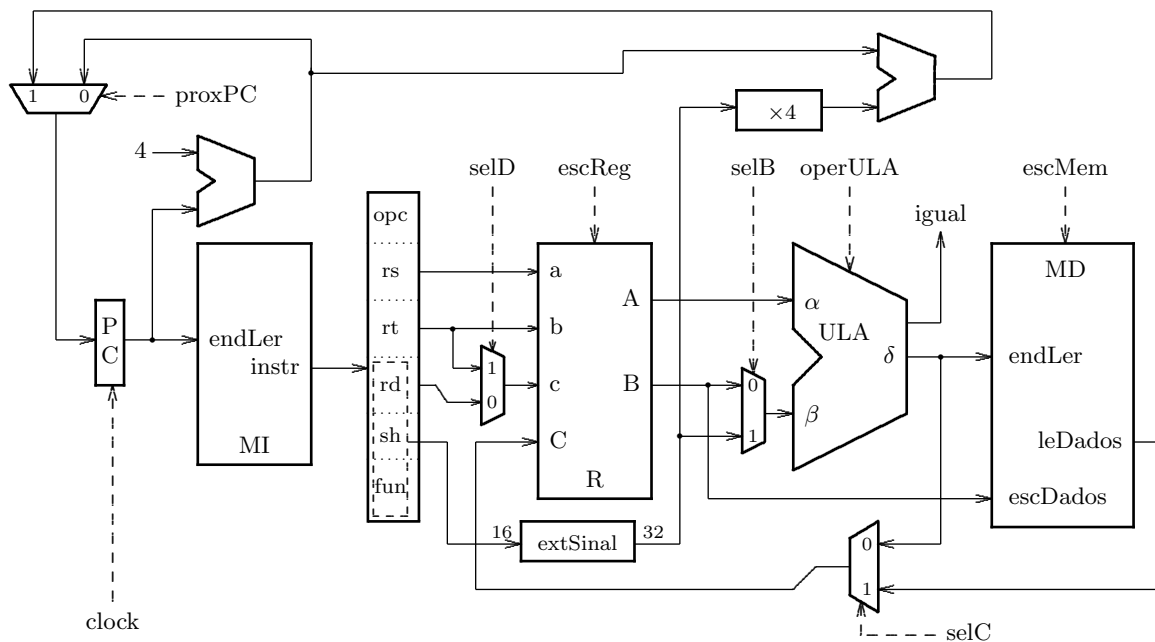
0) O diagrama de tempos abaixo mostra o comportamento dos sinais da interface assíncrona entre um computador e um periférico, numa transação rápida e numa lenta. Os sinais **ePto** e **E** são emitidos pelo mestre, e **eAct** é emitido pelo escravo. Projete as máquinas de estado do circuito que inicia uma transação (mestre) e do circuito que aceita uma transação iniciada pelo mestre (escravo). Mostre como implementar a máquina de estados do escravo, com *flip-flops* do tipo D. [10 pontos]



1) Mostre como implementar a instrução **JUMP THROUGH MEMORY** definida abaixo. Sua resposta deve conter: (i) uma tabela de sinais de controle indicando os sinais ativos durante a execução desta instrução; e (ii) um diagrama de tempos completo da execução desta instrução. [10 pontos]

`jtm desl(rs) # PC ← M[desl + rs]`

(formato I)



2) Traduza para *assembly* do MIPS o trecho de programa ao lado. Seu código *assembly* deve empregar as convenções de programação do MIPS. [10 pontos]

```
int log2(int n) {
    if (n < 2) then
        return 0;
    else
        return (1 + log2(n/2));
}
```

Prova Substitutiva

0) Projete um circuito deslocador com 8 bits na entrada ($e_i, i \in \{0..7\}$) e que permite deslocamentos de uma posição para a esquerda e para a direita — atenção com os bits de saída s_0 e s_7 . (i) Sua resposta deve conter o circuito que efetua os deslocamentos do bit s_i , bem como a composição de 8 destes circuitos para implementar o deslocador de 8 bits. (ii) Estenda o circuito de forma a manter o sinal de números representados em complemento de dois nos deslocamentos. [10 pontos]

1) Traduza para *assembly* do MIPS o trecho de código C. [10 pontos]

Para facilitar a correção indique os registradores como *ra*, *rb*, etc.

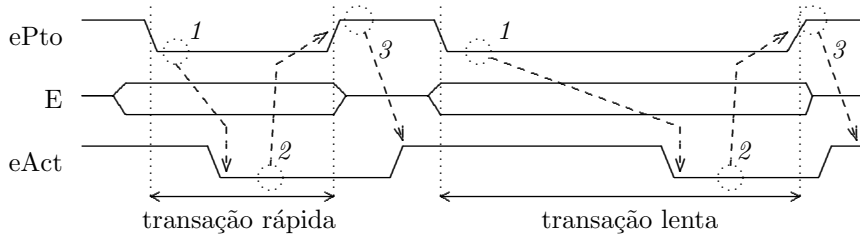
```
int a,b,i;
int x[NNN], y[MMM], z[KKK];
a = x[10] + x[ y[3] ] + x[ y[ z[5] ] ];
i = a/4;
b = x[i] + x[ y[2*i] ];
```

2) Mostre como implementar a instrução LOAD WITH POST-INCREMENT definida abaixo. Sua resposta deve conter: (i) uma tabela de sinais de controle indicando os sinais ativos durante a execução desta instrução; e (ii) um diagrama de tempos completo da execução desta instrução. No comentário a vírgula significa “execução simultânea”. [10 pontos]

`lwpi rt,desl(rs) # rt ← M[rs+ext(desl)] , rs ← rs+ext(desl) (formato I)`

Exame Final

1) O diagrama de tempos abaixo mostra o comportamento dos sinais da interface assíncrona entre um computador e um periférico, numa transação rápida e numa transação lenta. Os sinais **ePto** e **E** são emitidos pelo mestre, e **eAct** é emitido pelo escravo. Projete as máquinas de estado do circuito que inicia uma transação (mestre) e do circuito que aceita uma transação iniciada pelo mestre (escravo). Mostre como implementar a máquina de estados do mestre, com *flip-flops* do tipo D. [30 pontos]



2) Mostre como implementar a instrução **BRANCH-AND-LINK** definida abaixo. Sua resposta deve conter: (i) uma lista dos valores dos sinais de controle; (ii) um diagrama de tempos completo da execução desta instrução; (iii) explique por que o endereço de retorno é em **PC+8**? [30 pontos]

bal desl # $\$31 \leftarrow PC+8$, $PC \leftarrow (PC+4) + (\text{ext}(\text{desl}) \ll 2)$

(formato I)

3) Traduza para *assembly* do MIPS o trecho de programa ao lado. Seu código *assembly* deve empregar as convenções de programação do MIPS. Para facilitar a correção indique os registradores como **re**, **rn**, etc. [40 pontos]

```
...
x = power(y,z);
...
int power(int n,int exp) {
    if (exp > 1) return (n * power(n,exp-1));
    else      return (n);
}
```