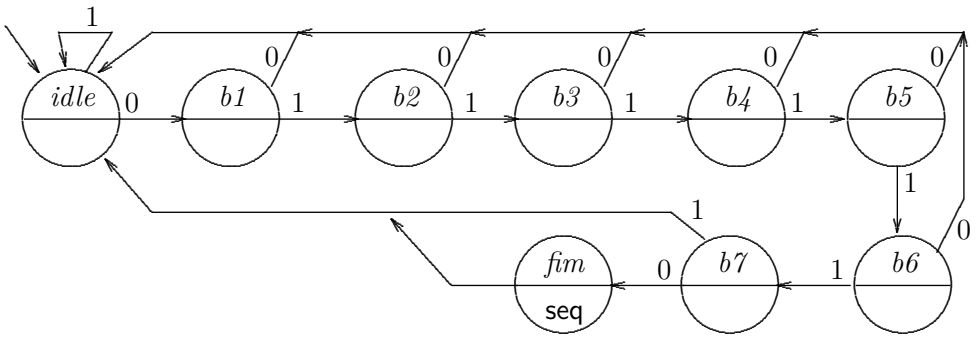
UFPR – Bacharelado em Ciência da Computação
CI210 – Projetos Digitais e Microprocessadores — 2014-1

Primeira Prova

1. Projete o microcontrolador que implementa a máquina de estados representada no diagrama abaixo. O circuito deve reconhecer a sequência com um zero, seguido de seis 1's, seguido de um zero (`01111110`). Quando a sequência é reconhecida, a saída seq fica em 1.

Sua resposta deve conter o diagrama do circuito do controlador e o conteúdo da memória ROM (μROM). [15 pontos]



2. Projete um circuito sequencial síncrono que produz em sua saída a sequência mostrada ao lado. Seu projeto deve empregar quatro FFs tipo T (T de "tatu"), e a resposta deve mostrar claramente todos os passos da solução. [10 pontos]

$0001 \rightarrow 0011 \rightarrow 0101 \rightarrow 0111 \rightarrow 1000 \rightarrow 1010 \rightarrow 1100 \rightarrow 1110$ (repete)

3. Projete um circuito combinacional que é um deslocador com 8 bits na entrada ($e_i, i \in \{7..0\}$) e que permite deslocamentos de nenhuma, uma, ou duas posições para a esquerda. Dependendo do sinal de controle, o bit de saída s_i pode mostrar qualquer um dentre $\{e_i, e_{i+1}, e_{i+2}\}$.
(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. Use multiplexadores. Atenção com os bits de saída s_7 e s_0.
(ii) Estenda o circuito de forma a manter o sinal de números representados em complemento de dois nos deslocamentos. [10 pontos]

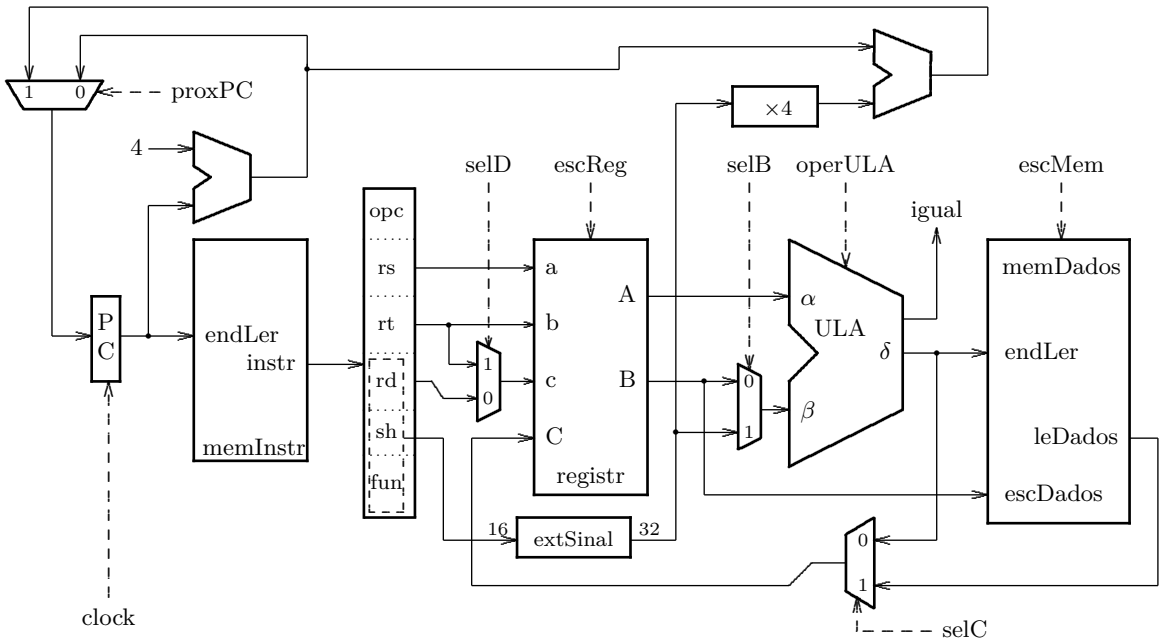
Segunda Prova

4. Traduza para *assembly* do MIPS o trecho de programa abaixo. Seu código *assembly* deve empregar as convenções de programação do MIPS. [15 pontos]

```
int a, i;
int x[2048], y[64];
...
i=1;
a=0;
while (i < 2048) {
   a = a + x[i] + x[ y[i%64] % 2048 ]; // MOD, MOD
   i = i * 2;
}
```

5. Mostre como implementar a instrução LOAD-WORD-SCALED (`lws`), que emprega um novo modo de endereçamento, chamado de *base-deslocamento escalado* e definido como
$\text{Ender}_{\text{efetivo}} = \text{regBase} + 2^{\text{escala}} \times \text{desloc}$.
O registrador base (R[rs]) e o deslocamento são os mesmos campos que na instrução LOAD-WORD. A *escala* é obtida do registrador R[rt], e é sobrescrita ao final da instrução. Sua resposta deve conter: [20 pontos]
(i) projeto e implementação do circuito deslocador;
(ii) indicação clara das modificações ao circuito do processador;
(iii) a tabela com os sinais de controle para a intrução `lws`;
(iv) um diagrama de tempos completo da execução desta instrução.

`lws rt,rs(desl)` `# R[rt] ← mem[ R[rs] + ( 2^R[rt] × extS(desl) ) ]` (formato I)



Exame Final

6. Traduza para *assembly* do MIPS o trecho de programa abaixo. Seu código *assembly* deve empregar as convenções de programação do MIPS. [35 pontos]

```
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;
}
```

7. Mostre como implementar a instrução ABSOLUTE VALUE definida abaixo. Sua resposta deve conter: (i) indicação clara de quaisquer modificações necessárias ao circuito do processador, *que podem ser desenhadas na folha de perguntas*; (ii) uma tabela de sinais de controle indicando os sinais ativos durante a execução desta instrução; e (iii) um diagrama de tempos completo da execução desta instrução. [35 pontos]

`abs rd, rs` `# R[rd] ← ( R[rs] >= 0 ? R[rs] : compl2(R[rs]) )` (formato I)

8. Mostre como projetar um registrador de deslocamento com 4 bits de largura ($Q_0Q_1Q_2Q_3$), com carga paralela (sinal `load`) nos sinais $D_0D_1D_2D_3$, inicialização assíncrona (`reset`), e deslocamento de uma posição a cada pulso do relógio em que não ocorra uma carga, a partir da entrada D_0. Sua resposta deve conter um diagrama de blocos com *flip-flops* e outros componentes que sejam necessários. [30 pontos]