UFPR – Bacharelado em Ciência da Computação
CI210 – Projetos Digitais e Microprocessadores
Primeira Prova — 2012-2

1) Projete um circuito deslocador combinacional que multiplica a sua entrada de 4 bits por qualquer número do conjunto $\{1, 4, 16, 64, 256, 1024, 4096, 16.384\}$. [5 pontos]

2) Projete um circuito sequencial síncrono que produz em sua saída a sequência de pulsos mostrada abaixo. Seu projeto deve empregar FFs tipo D, e a resposta deve mostrar claramente todos os passos da solução. [10 pontos]

0000
1001
1111
0011
1010
1100
0000 → repete

3) Projete um circuito sequencial síncrono que produz em sua saída a sequência de pulsos mostrada abaixo. Quando a entrada I fica ativa, a sequência se inverte. Seu projeto deve empregar um microcontrolador. A resposta deve mostrar claramente todos os passos da solução. [10 pontos]

I	PE	I	PE
0	0000	1	1100
0	1001	1	0000
0	1111	1	1001
0	0011	1	1111
0	1010	1	0011
0	1100	1	1010

Segunda Prova — 2012-2

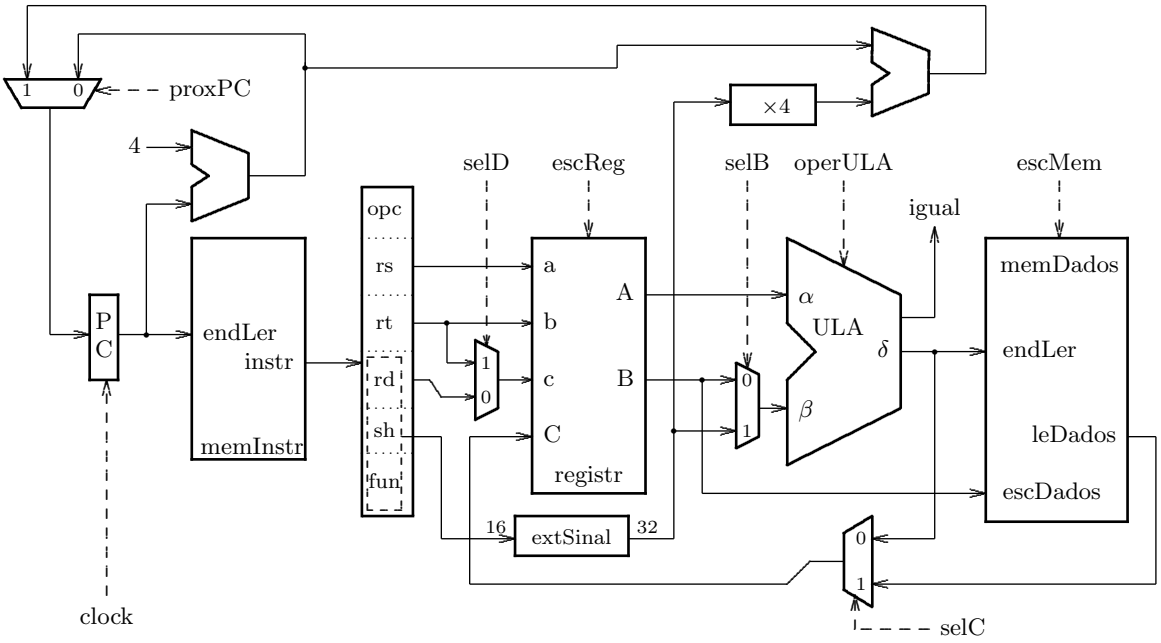
2) Traduza para *assembly* do MIPS o trecho de programa ao lado. Seu código *assembly* deve empregar as convenções de programação do MIPS. *Para facilitar a correção indique os registradores como* `re`, `rp`, *etc.* [15 pontos]

```
typedef elem {
   elem *next;
   int vet[4];
} elem;

elem *head, *x;
elem strut[256];

...
   x = insert( head, &(strut[j]) );
   x->vet[3] = 512;
...
```

```
elem * insert(elem *h, elem *e) {
   elem *p;

   p = h;
   while (p != NIL) {
      p = p->next;
   }
   p->next = e;
   e.next = NIL;
   return e;
}
```

1) Mostre como implementar a instrução BRANCH-AND-LINK definida abaixo. Sua resposta deve conter: [10 pontos]
(i) indicação clara das modificações no circuito;
(ii) a tabela com os sinais de controle;
(iii) um diagrama de tempos completo da execução desta instrução.

`bal desl` `# $31←PC+8 , PC←(PC+4)+(ext(desl)≪2)` (formato I)



Prova Substitutiva — 2012-2

1) 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: [15 pontos]
(i) indicação clara da 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)

2) Escreva um programa em C que copia uma cadeia (*string*). O endereço da cadeia fonte é apontado por `char *f`, e o endereço de destino por `char *d`. A função retorna o número de caracteres copiados, e o protótipo de `cpyCad()` é mostrado abaixo.

```
int cpyCad(char *f, char *d);
```

Seu código *assembly* deve empregar as convenções de programação do MIPS. Lembre que as instruções para operações com caracteres são `lb` (*load-byte*) e `sb` (*store-byte*). [10 pontos]

Exame Final — 2012-2

1) Projete um circuito combinacional que é um 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, e que também repete a entrada na saída sem deslocamento. Dependendo do sinal de controle, o bit de saída s_i pode mostrar qualquer um dentre $\{e_{i-1}, e_i, e_{i+1}\}$.
(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_0 e s_7.
(ii) Estenda o circuito de forma a manter o sinal de números representados em complemento de dois nos deslocamentos. [30 pontos]

2) Traduza para *assembly* do MIPS o trecho de código C. O operador `%` é o módulo da divisão inteira. [40 pontos]

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

```
int a, i; int x[1024], y[2048];

i=0;
a=0;
while (i < 1024) {
   a = a + x[i] + y[ (x[i] % 2048) ];
   i = i + 1;
}
```

3) 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: [30 pontos]
(i) indicação clara da 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]` $\leftarrow$ `mem[ R[rs] + (` $2^{\texttt{R[rt]}}$ $\times$ `extS(desl) ) ]` (formato I)