Projeto de circuito

MAC0329 – Álgebra booleana e aplicações (DCC / IME-USP — 2019)

– Todas as etapas do projeto deverão ser feitas com o *Logisim* (http://www.cburch.com/logisim/) –

– O projeto poderá ser feito em grupo de até três pessoas –

Parte 4: Processador final – entrega no PACA, até 10/07

O objetivo deste EP é concluir o processador parcialmente desenvolvido até agora. Para isso, o repertório de instruções deverá ser ampliado conforme especificado adiante. Com o repertório ampliado, será possível escrever e simular programas simples.

1 Conjunto de instruções

O conjunto de instruções que devem ser adicionadas ao processador está especificado a seguir.

Código		Descrição
base 10	base 16	
00	00	NOP (no operation)
01	01	Copie [EE] para o AC
02	02	Copie [AC] para a posição de endereço EE
03	03	Some [EE] com [AC] e guarde o resultado em AC
04	04	Subtraia [EE] de [AC] e guarde o resultado em AC
07	07	Leia um número e guarde-o na posição de endereço EE
08	08	Imprima [EE]
09	09	Pare
10	0A	Desvie para EE (desvio incondicional)
11	0B	Desvie para EE se [AC] > 0
13	0D	Desvie para EE se [AC] $= 0$
15	0F	Desvie para EE se [AC] < 0

Observação: A instrução 09 (Pare) deverá fazer com que o processador volte ao estado correspondente ao do início de execução de um programa. Isto é, deve zerar o PC (*Program Counter*) e voltar à configuração de início de um ciclo de execução.

2 Tarefa, entrega e avaliação

Tarefa: concluir o processador, tornando-o capaz de executar as instruções da tabela acima.

Entrega: entregar via PACA um arquivo `cpu.circ`, contendo o processador descrito acima, e um relatório sucinto sobre como está organizado o processador implementado.

Avaliação: Serão **avaliados os seguintes aspectos**:

- corretude dos circuitos

- clareza na organização do circuito: espera-se que o circuito esteja organizado de tal forma que a localização de suas diferentes partes assim como a simulação e teste do circuito sejam fáceis.
- cumprimento do prazo

Neste EP podem ser usados os componentes disponíveis no `Logisim`, exceto a ULA, que deve ser a desenvolvida por vocês.

3 Exemplos para teste do processador

Para testar o circuito, crie pequenas sequências de instruções e gere os pulsos do *clock* manualmente, de forma que seja possível acompanhar as alterações que ocorrem em diferentes partes do circuito a cada pulso. Lembre-se que iremos supor que a primeira instrução a ser executada estará sempre no endereço 0x00.

A seguir estão três exemplos de programas, que podem ser usados para testar o processador. Para cada linha do exemplo, estão presentes o endereço na RAM, o código da instrução, e o significado da instrução (os números estão em notação hexadecimal).

Primeiro exemplo: teste de leitura e impressão. O dado de entrada deve ser representado por um pino de entrada. Antes de executar uma instrução de leitura, insira o valor a ser lido no pino de entrada (podemos supor que esse é o número que foi digitado pelo usuário). No caso da impressão, o valor a ser impresso pode ser enviado para um pino de saída ou então mostrado usando os displays de 7 segmentos.

```
00: 0710 -- Leia um número e guarde-o na posição de endereço 0x10
01: 0810 -- Imprima [0x10]
02: 0900 -- Pare
```

Segundo exemplo: teste de adição e subtração

```
00: 0710 -- Leia um número e guarde-o na posição de endereço 0x10
01: 0711 -- Leia um número e guarde-o na posição de endereço 0x11
02: 0110 -- Copie [0x10] para o AC
03: 0311 -- Some [0x11] com [AC] e guarde o resultado em AC
04: 0212 -- Copie [AC] para a posição de endereço 0x12
05: 0812 -- Imprima [0x12]
06: 0110 -- Copie [0x10] para o AC
07: 0410 -- Subtraia [0x11] de [AC] e guarde o resultado em AC
07: 0213 -- Copie [AC] para a posição de endereço 0x13
08: 0813 -- Imprima [0x13]
09: 0900 -- Pare
```

Terceiro exemplo: teste de laço (desvios)

```
00: 0110 -- Copie [0x10] para o AC
01: 0211 -- Copie [AC] para a posição de endereço 0x11
02: 0811 -- Imprima [0x11]
```

```
03: 0712 -- Leia um número e guarde-o na posição de endereço 0x12
04: 0110 -- Copie [0x12] para o AC
05: 0D0A -- Desvie para 0x0A se [AC] = 0
06: 0311 -- Some [0x11] com [AC] e guarde o resultado em AC
07: 0211 -- Copie [AC] para a posição de endereço 0x11
08: 0811 -- Imprima [0x11]
00: 0A02 -- Desvie para 0x03
0A: 0900 -- Pare
    ...
0F: ...
10: 0000 -- Zero
11:      -- Soma
12:      -- Num
```

Estes programas podem ser gravados em um arquivo txt e carregados para a RAM. Por exemplo, o conteúdo do arquivo txt correspondente ao terceiro exemplo deve ser

```
v2.0 raw
0110 0211 0811 0712 0110 0d0a 0311 0211 0811 0a02 0900 0000 0000 0000 0000 0000 0000
```

Para carregar essas instruções na RAM, basta clicar sobre ela e usar a opção `Carregar imagem...`. Para ver todo o conteúdo da RAM ou editá-los, basta usar a opção `Editar conteúdos...`

Postem as dúvidas no Fórum de discusssões no PACA.