PCS3115 - Sistemas Digitais II - Trabalho 1

por Bruno de Carvalho Albertini

15/03/2021

Neste trabalho você montará o processador PoliStack a partir da UC e do DF desenvolvidos em SD1.

Introdução

Na disciplina de Sistemas Digitais 1 você desenvolveu todo arcabouço necessário para projetar um processador de pilha, começando com circuitos combinatórios simples, passando por circuitos aritméticos, sequenciais (registradores) e terminando com a descrição da Unidade de Controle (UC) e do Fluxo de Dados (DF).

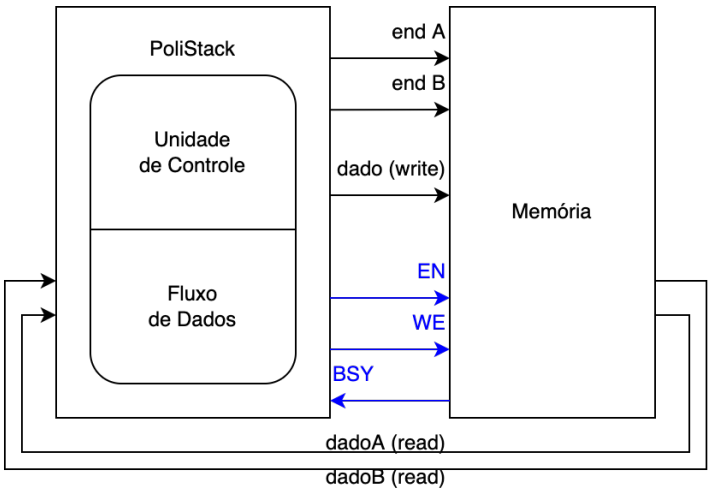

Figura 1: Conexões entre o processador e a memória no PoliStack. Em preto os dados e em azul o controle.

Neste trabalho, você deverá juntar a UC e o DF desenvolvidos para montar o processador PoliStack. A entidade do processador está descrita na Figura **??**.

O processador é parametrizável, tanto na largura do barramento de endereços (número de bits que o processador usa para acessar os endereços de memória) quanto no tamanho da palavra de memória (quantidade de bits por operação). Por padrão, o processador assume que o barramento de endereços é de 16b, ou seja, pode endereçar 2^{16} endereços, e as palavras são de 32b, ou seja, qualquer leitura ou escrita é realizada com palavras de 32b. Lembramos que o PoliStack é um processador de 32b (4B), ou seja, todas as operações acontecem com 32b (e.g. uma soma sempre é de 32b + 32b = 32b (e possivelmente *emph*)). No entanto, o PoliStack tem tamanho de instrução fixo em 8b (1B).

A memória é única e organizada em palavras de 8b (1B). Não há distinção entre memória de dados e de instruções, então o processador busca instruções na mesma memória que opera dados. No caso

da instância com parâmetros padrão, a memória seria de 2^{16} posições de 8b (1B). Note que a memória tem duas portas de acesso, porém a porta A só permite leitura e a porta B permite tanto leitura quanto escrita, esta última habilitada pelo sinal `mem_we` em alto e com canal de dados a serem escritos separado (`memB_wrd`).

```
entity polistack is
  generic(
    addr_s : natural := 16; -- addr_s size in bits
    word_s : natural := 32  -- word size in bits
  );
  port (
    clock, reset: in bit;
    halted: out bit;
    -- memory interface
    mem_we, mem_enable    : out bit;
    memA_addr, memB_addr : out bit_vector(addr_s-1 downto 0);
               memB_wrd  : out bit_vector(word_s-1 downto 0);
    memA_rdd,  memB_rdd  : in  bit_vector(word_s-1 downto 0);
    busy                 : in  bit
  );
end entity;
```

Figura 2: Entidade VHDL para o PoliStack.

Arquivos Iniciais

Junto com este enunciado, também está publicado o enunciado do último trabalho de SD1, que contém detalhes sobre a UC e o DF recomendados e também uma descrição das instruções suportadas. Não é obrigatório reutilizar as suas soluções e você pode resolver o problema usando outras abordagens, apesar de recomendarmos fortemente que atenha-se ao paradigma de UC-DF.

Além do enunciado do último trabalho, este trabalho possui um espaço de endereçamentos. O espaço de endereçamento começa com a ROM, onde é carregado o programa a ser executado (incluindo constantes), seguido pela RAM, inicialmente zerada, e todos os espaços que sobram são considerados Entrada/Saída (ES). Ao escrever em um endereço de ES, você de fato está escrevendo em uma interface serial e pode ver o que o seu processador está escrevendo. O processo de leitura é idêntico: ao ler um endereço que não está mapeado na ROM ou RAM, você estará lendo da serial.

Para facilitar o trabalho, também está anexado a este trabalho um arquivo compactado. Este arquivo contém um *testbench* que implementa o mapeamento de espaço de endereçamento citado, com os seguintes parâmetros: ROM 2^{12} endereços, totalizando 4KiB, seguida por uma RAM de 2^{17} endereços, totalizando 128KiB, em um espaço de endereçamento de no mínimo 18b (2^{18} endereços ou 256KiB).

O software fornecido no arquivo `hello_world.mif` já está compilado e cada 8b corresponde a uma instrução do PoliStack (e.g. as primeiras 4 instruções são `0b`, que significam `NOP`).

Atividades

T1A1 Implemente o processador PoliStack conforme a entidade da Figura **??**. Use os documentos em anexo como apoio (enunciado de SD1 e arquivo compactado com *testbench*).

Trabalho 1, Atividade 1, 10 envios, maior nota, 10 pontos

Instruções para uso dos arquivos de suporte

Os binários disponíveis no arquivo compactado foram gerados com o GHDL 3.0.0-dev, *backend* `llvm`. Você pode usá-los em um sistema compatível, mas a maneira mais fácil é usando o Docker. Com o Docker instalado na sua máquina, execute `docker pull ghdl/ext` para baixar a imagem e em seguida, no diretório onde descompactou o arquivo:

`https://www.docker.com/`

```
docker run -rm -it -name ghdl -v $PWD:/workspace -w /workspace ghdl/ext /bin/bash
```

para abrir um terminal com o o GHDL disponível. Para executar a solução do professor, execute:

```
ghdl -link toplevel
./toplevel
```

E finalmente para testar com a sua implementação, execute:

```
ghdl -a polistack.vhd
./toplevel
```

Lembrando que o arquivo `polistack.vhd` deve conter a sua solução em um único arquivo. Esta operação irá sobrescrever o `polistack.o` fornecido, portanto se você executar novamente o passo de *link* (o mesmo para a solução do professor), você executará o mesmo *testbench*, mas agora com o seu processador.

Caso decida desenvolver o próprio software, o *testbench* espera que a escrita seja realizada no endereço `0x080A000C`.

Instruções para Entrega

Para este trabalho você pode usar todas as bibliotecas de VHDL disponíveis por padrão no GHDL, exceto a `textio`. A violação destas restrições acarreta nota zero automaticamente, sem direito a revisão.

Como boa prática de engenharia, faça seus *testbenches* ou use o fornecido e utilize o GHDL para validar suas soluções antes de postá-las no juiz.

Há um *link* específico no e-Disciplinas para a atividade deste trabalho. Acesse-o somente quando estiver confortável para enviar sua solução. Você pode enviar apenas um único arquivo com sua descrição VHDL em UTF-8 para cada atividade. O nome do arquivo não importa, mas sim a descrição que está dentro. As entidades devem ser como as especificadas ou o juiz te atribuirá nota zero.

Quando acessar o *link* no e-Disciplinas, o navegador abrirá uma janela para envio do arquivo. Selecione-o e envie para o juiz. Jamais

Quando você clicar no *link* tem 1 minuto para enviar o arquivo ou fechar a janela, caso contrário uma submissão será contabilizada.

recarregue a página de submissão pois seu navegador pode enviar o arquivo novamente, o que vai ser considerado pelo juiz como um novo envio e pode prejudicar sua nota final. Caso desista do envio, simplesmente feche a janela antes do envio.

Depois do envio, a página carregará automaticamente o resultado do juiz, quando você poderá fechar a janela. Se não quiser esperar o resultado, feche a janela após o envio e verifique sua nota no e-Disciplinas posteriormente. A nota dada pelo juiz é somente para a submissão que acabou de fazer. Sua nota na atividade poderá ser vista no e-Disciplinas e pode diferir da nota dada pelo juiz dependendo da estratégia de atribuição de notas utilizada pelo professor que montou o problema.

Este trabalho demora cerca de 4os para ser analisado pelo juiz.

Atenção: não atualize a página de envio e não envie a partir de conexões instáveis (e.g. móveis) para evitar que seu arquivo chegue corrompido no juiz.