

Aluno: \_\_\_\_\_

Matrícula: \_\_\_\_\_ Data: \_\_\_\_\_

### Sprint 3 – Banco de registradores – Processador MIPS

**Descrição geral do problema:** Implemente um banco com 8 registradores de 8 bits. Esse banco deverá possuir um barramento de escrita síncrono (dados + endereço) e dois barramentos de leitura combinacionais (dados + endereços)

#### Requisitos mínimos:

Abra o projeto da Sprint2 e edite-o para incluir as funcionalidades dessa sprint. **Obs: “File > Open Project” e NÃO “File > Open”.**

1. Faça a descrição de hardware de um módulo, denominado RegisterFile, que gere o circuito indicado na Figura 1, usando a linguagem Verilog/SystemVerilog.

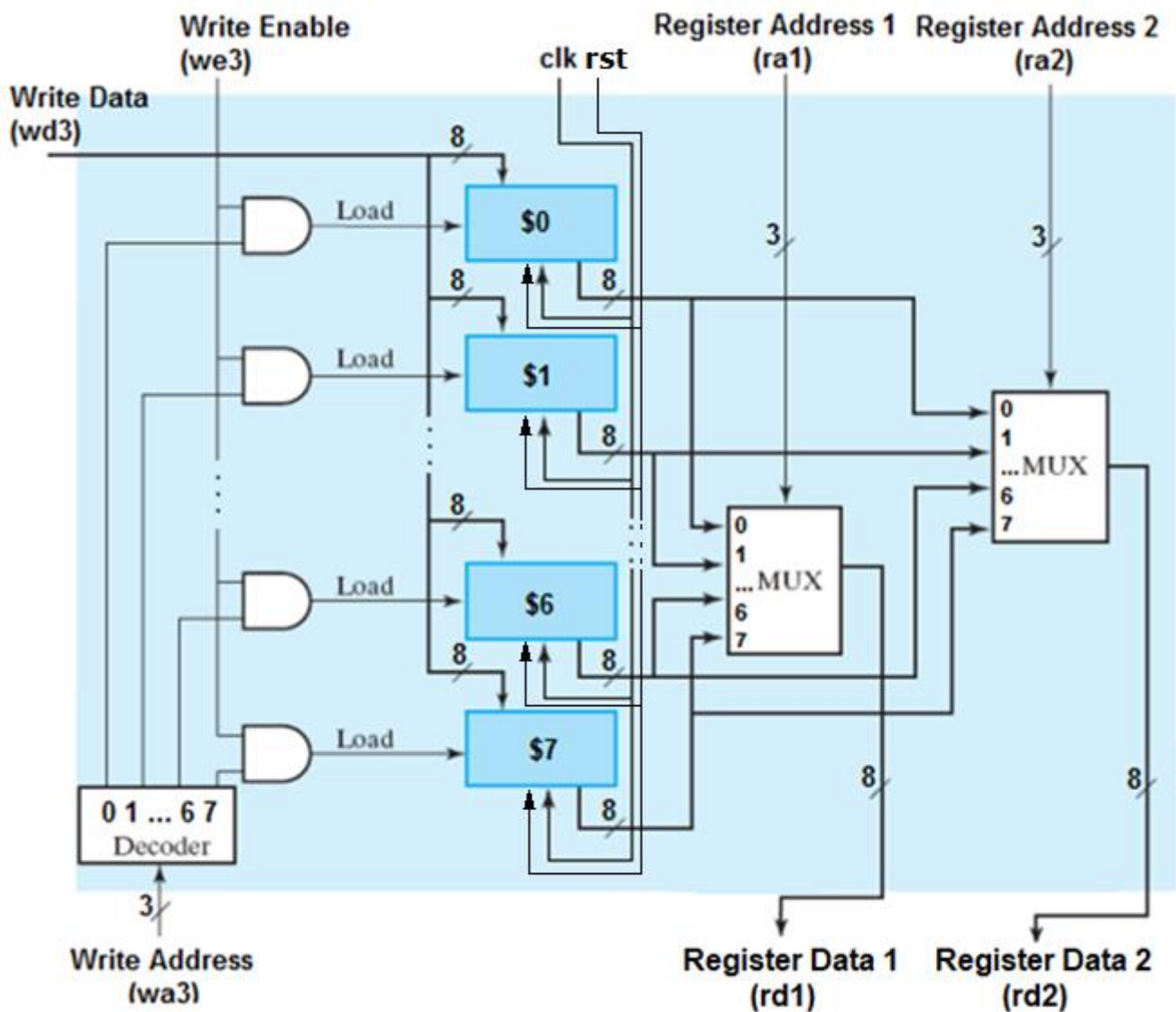


Figura 1 –Diagrama do módulo RegisterFile.

O módulo apresenta as seguintes entradas e saídas:

**Entradas:**

- Write Data (wd3)* (8 bits) – Entrada de dados;
- Write Address (wa3)* (3 bits) – Seleção do registrador que armazenará o dado proveniente de *Write Data (wd3)*;
- Write Enable (we3)* (1 bit) – Habilita (1) ou desabilita (0) a gravação de dados nos registradores de \$1 a \$7;
- clk* (1 bit) – Se o sinal *Write Enable (we3)* estiver ativo (1), na borda de subida do clock, o dado é gravado no registrador selecionado.
- Register Address 1 (ra1)* (3 bits) – Seleção de qual registrador será disponibilizado na saída rd1;
- Register Address 2 (ra2)* (3 bits) – Seleção de qual registrador será disponibilizado na saída rd2;
- Reset (rst)* (1 bit) – Reseta o valor dos registradores, em nível baixo (0);

**Saídas:**

- Register Data 1 (rd1)* (8 bits) – Barramento de saída de 8 bits;
- Register Data 2 (rd2)* (8 bits) – Barramento de saída de 8 bits;

Consideração Importante: O circuito não precisa ficar idêntico à figura, modularizado em decodificadores e portas AND. O funcionamento é que deve ser preservado. Faça a descrição em alto nível, para ganhar produtividade.

Princípio de funcionamento em três partes:

- ESCRITA:** A operação de escrita deve ser sincronizada com a borda de subida do clock (clk). O valor de 8 bits de *Write Data (wd3)* é armazenado no registrador cujo índice é igual a *Write Address (wa3)* (3 bits) se *Write Enable (we3)* for 1. O valor dos registradores se mantém caso *Write Enable (we3)* seja 0. *Circuito sequencial síncrono* → *always @(posedge clock)*. **OBS: O registrador \$0 é somente de LEITURA e seu valor deve ser fixo em ZERO.**
- LEITURA:** o valor do registrador de índice *ra1* é disponibilizado de forma contínua na saída *rd1* assim como o registrador de índice *ra2* na saída *rd2*. Variando qualquer entrada, as saídas são atualizadas. *Circuito Combinacional* → *always @(\*)* ou *assign*.
- RESET:** limpa o valor dos registradores quando a entrada *rst* for 0. Decida se implementará essa funcionalidade de forma síncrona (no momento da subida do clock) ou assíncrona (na descida do rst).

**Os dois multiplexadores que conectam os registradores às saídas rd1 e rd2 são circuitos combinacionais. Os dados gravados nos registradores selecionados, sempre estarão disponíveis nas respectivas saídas.**

2. Durante a instanciação do RegisterFile, no módulo “Mod\_Teste”, faça as seguintes conexões:

Direção da Porta	Nome da porta	Conectar no fio
Entradas	clk	KEY[1]
	we3	SW[17]
	wa3	SW[16:14]
	ra1	SW[13:11]
	ra2	SW[10:8]
	wd3	SW[7:0]
	rst	KEY[2]
Saídas	rd1	w_d0x0[7:0]
	rd2	w_d0x1[7:0]

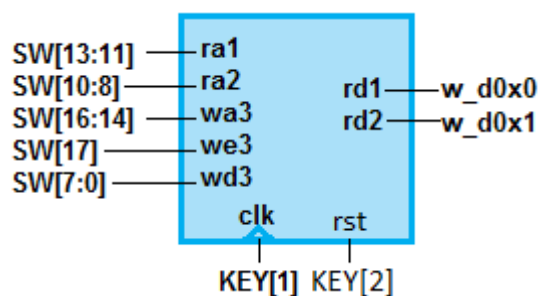
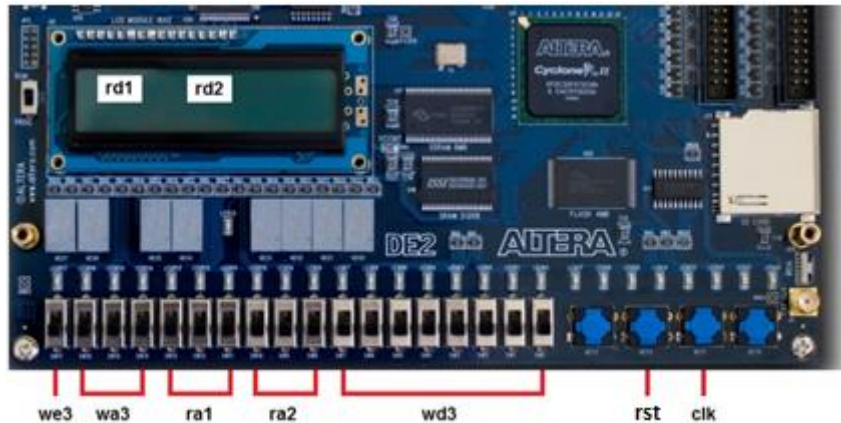


Figura 2 – Interfaces do módulo RegisterFile

Ligações auxiliares:

LEDG[8]	~KEY[1]	Dica: assign
HEX0 e HEX1	SW[7:0]	Dica: conversor hex7seg da sprint 2

3. Visualize a entrada *wd3* nos displays HEX0 e HEX1, assim como conecte as saídas *rd1* e *rd2* respectivamente nas posições *d0x0* e *d0x1* do LCD.
4. Para avaliar o funcionamento do circuito, realize a gravação de um conjunto de dados nos registradores. Após a gravação, verifique se os dados foram de fato gravados, selecionando *ra1* e *ra2* e observando as saídas *rd1* e *rd2*



TESTE: Escrever os seguintes valores nos registradores:

$\$1 \leftarrow CA$

$\$7 \leftarrow FE$

$\$0 \leftarrow DB$

$rst \leftarrow 0$

#### Desafio (Valendo +0,1 na média geral)

- Simule exaustivamente as operações de escrita e leitura em cada um dos registradores do banco
- Crie um testbench para o DUT ou preencha manualmente um arquivo waveform (.vwf) para alimentar a simulação