

Aluna: Jaqueline Ferreira de Brito

Atividade 1:

a)

```
xcelium> run
Value of x = 1 y = 3 by position
Value of s1 = '{x:2, y:4}' by name
Simulation complete via $finish(1) at time 5 NS + 0
./testbench.sv:13 #1 $finish;
xcelium> exit
```

b) Justificação dos resultados:

- Dentro do typedef struct é definido um conjunto de informações com diferentes identificadores de referência, os elementos inteiros, x e y.
- Em seguida é criado dois elementos, myStruct s1 e int K = 1.
- Dentro de um **initial begin** é iniciado as operações que vai nos trazer os resultados de saída mostrados na letra a).
- No primeiro \$display, diz que $s1 = \{1, 2+k\}$, e pede para apresentar o valor de s1 na posição x e y, representado por s1.x e s1.y, como mostra a saída acima.
- Logo o elemento s1 recebe na posição x = 1, e y igual a 3. Porque na definição do elemento k foi definido que ele teria 1 na definição do elemento, mas na posição y para s1 pede-se $2 + k$, logo a soma tem como resultado 3, resultando $y = 3$.
- No segundo \$display ele pede o nome nas posições x e y para s1 e não só o que tá lá, $s1 = \{x:2, y:3+k\}$, logo o resultado apresentado na saída de s1 na posição x = x:2 e y = y:4, no caso da posição y em s1 tem uma operação de $3+k$ e $k = 1$ por definição do código, logo o \$display apresenta y + o valor real da posição.
- Finalizando com \$finish terminará a operação.

Atividade 2:

a)

```
xcelium> run
a = 00001010 b = 0 c = 0064
a = 00001011 b = 1 c = 0065
Simulation complete via $finish(1) at time 1 NS + 0
./testbench.sv:18 #1 $finish;
xcelium> exit
```

b)

- Dentro do typedef struct um conjunto de informações com diferentes identificadores de referência, os elementos byte a, reg b e shortint unsigned c, chamado de myStruct.
- Dentro do module struct_data, é instanciado elementos decimais para o struct, um elemento é chamado de myLocalStruct = `{11,1,101}, e myStruct tobject = `{10,0,100}.
- Dentro de um **initial begin** é iniciado as operações que vai nos trazer os resultados de saída mostrados na letra a).
- No primeiro \$display, pede que seja apresentado na saída os valores em forma de binário para os elementos decimais em a e b, e hexadecimal para c no struct myStruct Object definido dentro do module.
- No segundo \$display, pede que seja apresentado na saída os valores em forma de binário para os elementos decimais em a e b, e hexadecimal para c no struct myStructLocal definido dentro do module.
- Finalizando com \$finish terminará a operação.

Atividade 3:

a)

```
xcelium> run
Ctrl: '{opcode:'h23}'
Ctrl atualizado: '{opcode:'h2}'
xmsim: *W,RNQUIE: Simulation is complete.
xcelium> exit
```

b)

- Dentro do typedef union é definido uma estrutura de dados, onde opcode tem 6 bit e funct tem 3 bit e é chamado de control_unit.
- Dentro de um **initial begin** é definido o control_unit como ctrl, e em seguida começa as requisições com o \$display.
- No primeiro \$display, ctrl.opcode = b'100011 é apresentado na saída o %p de ctrl, que tem como resultado o nome da operação que representa ctrl.opcode mais seu valor em hexadecimal.
- No segundo \$display, ele atualiza o valor de ctrl.funct = "b010 para ctrl.opcode.

Atividade 4:

a)

```
xcelium> run
Fraction: 000000, Exponent: c8
Fraction: ffffff, Exponent: 38
xmsim: *W,RNQUIE: Simulation is complete.
xcelium> exit
```

b)

- Dentro do typedef union packed é definido uma estrutura de dados, onde wordView tem 32 bit dentro dessa estrutura de union packed, os 32bits de wordView foi distribuídos dentro de uma struct packed, 24 bits para a fraction e 8 bits para o exponent.
- O struct packed é chamado de floatView e o union packed é chamado de MyUnion.

- Dentro de um **initial begin** no module Testbench começa às operações com MyUnion e myData.
- No primeiro \$display, myData.wordView = 200, vai trazer como resultado em forma de hexadecimal a quantidade de bits do myData para o struct floatView da fraction e em seguida do exponent.
- No segundo \$display, ele faz a mesma operação do primeiro só que agora ele trará o resultado da representação negativa do -200, em hexadecimal tanto para fraction, quanto para exponent.