

Acima é realizada a operação lógica OR gerando o output observado na imagem, e mantendo inativas as saídas zero, “negative”, “carry” e “overflow”.

- **Operação ADD (0010):**

1º caso) ADD com output positivo:

	Msgs	
+ /ula_tb/S_input1	000000000000...	000000000000000000000000000011
+ /ula_tb/S_input2	000000000000...	00000000000000000000000000001101
+ /ula_tb/S_operation	0010	0010
+ /ula_tb/S_output	000000000000...	00000000000000000000000000000000
/ula_tb/S_zero	0	
/ula_tb/S_negative	0	
/ula_tb/S_carry	0	
/ula_tb/S_overflow	0	

Input 1 (19 em decimal) + Input 2 (13 em decimal) = Output (32 em decimal). Observe que as saídas zero, “negative”, “carry” e “overflow” estão inativas.

2º caso) ADD com output zero:

	Msgs			
+ /ula_tb/S_input1	0000000000000...	0000000000000000000000000000000000000011		
+ /ula_tb/S_input2	1111111111111...	111111111111111111111111111111111111101101		
+ /ula_tb/S_operation	0010	0010		
+ /ula_tb/S_output	0000000000000...	00		
/ula_tb/S_zero	1			
/ula_tb/S_negative	0			
/ula_tb/S_carry	0			
/ula_tb/S_overflow	0			

Input 1 (19 em decimal) + Input 2 (-19 em decimal) = Output (0 em decimal). Observe que a saída zero está ativa, indicando que output é nulo. Não ocorre número negativo, “carry” ou “overflow” nesse caso.

3º caso) ADD com output negativo:

[illegible]

Input 1 (19 em decimal) + Input 2 (-30 em decimal) = Output (-11 em decimal). Observe que a saída “negative” está ativa, indicando que output é negativo. Além disso, observamos que também ocorre “carry” nessa soma, ativando a saída “carry”. Não ocorre output zero, ou “overflow” nesse caso.

4° caso) ADD com “overflow”:

[illegible]

Input 1 (-1073741824 em decimal) + Input 2 (-1342177280 em decimal) = Output (1879048192 em decimal). Observe que a saída “overflow” está ativa devido ao fato de somarmos dois números negativos e obtermos um número positivo. Além disso, observamos que também ocorre “carry” nessa soma, ativando a saída “carry”. Não ocorre output zero, ou negativo nesse caso.

- **Operação SUB (0011):**

1º caso) SUB com output positivo:

[illegible]

Input 1 (19 em decimal) - Input 2 (13 em decimal) = Output (6 em decimal). Observe que as saídas zero, “negative”, “carry” e “overflow” estão inativas.

2° caso) SUB com output zero:

[illegible]

Input 1 (19 em decimal) - Input 2 (19 em decimal) = Output (0 em decimal). Observe que a saída zero está ativa, indicando que output é nulo. Não ocorre número negativo, “carry” ou “overflow” nesse caso.

3º caso) SUB com output negativo:

	Msgs	
+ /ula_tb/S_input1	00000000000000...	000000000000000000000000000011
+ /ula_tb/S_input2	00000000000000...	00000000000000000000000000001110
+ /ula_tb/S_operation	0011	0011
+ /ula_tb/S_output	11111111111111...	11111111111111111111111110101
/ula_tb/S_zero	0	
/ula_tb/S_negative	1	
/ula_tb/S_carry	0	
/ula_tb/S_overflow	0	

Input 1 (19 em decimal) - Input 2 (30 em decimal) = Output (-11 em decimal). Observe que a saída “negative” está ativa, indicando que output é negativo. Não ocorre output zero, “carry” ou “overflow” nesse caso.

4º caso) SUB com “overflow”:

[illegible]

Input 1 (-536870912 em decimal) - Input 2 (1879048192 em decimal) = Output (1879048192 em decimal). Observe que a saída “overflow” está ativa devido ao fato de subtrairmos um número positivo de um negativo e obtermos um número positivo. Não ocorre output zero ou negativo, nem “carry” nesse caso.

- **Operação SLT (0100):**

1º caso) SLT não válido setando Output com zero:

	Msgs		
/ula_tb/S_input1	0000000000000000...	000000000000000000000000000010011	
/ula_tb/S_input2	0000000000000000...	00000000000000000000000000001101	
/ula_tb/S_operation	0100	0100	
/ula_tb/S_output	0000000000000000...	00000000000000000000000000000000	
/ula_tb/S_zero	1		
/ula_tb/S_negative	0		
/ula_tb/S_carry	0		
/ula_tb/S_overflow	0		

Acima verificamos se Input 1 (19 em decimal) < Input 2 (13 em decimal). Como a inequação não é válida, output é setado com zero.

2º caso) SLT válido setando Output com um:

	Msgs	
+ /ula_tb/S_input1	00000000000000... 0	0000000000000000000000000000000000000011 0
+ /ula_tb/S_input2	00000000000000... 0	000000000000000000000000000000000000110101 0
+ /ula_tb/S_operation	0100 0	0100 0
+ /ula_tb/S_output	00000000000000... 0	00000000000000000000000000000000000001 0
/ula_tb/S_zero	0	
/ula_tb/S_negative	0	
/ula_tb/S_carry	0	
/ula_tb/S_overflow	0	

Acima verificamos se Input 1 (19 em decimal) < Input 2 (53 em decimal). Como a inequação é válida, output é setado com um.

- **Operação NOR (0101):**

[illegible]

Acima é realizada a operação lógica NOR gerando o output observado na imagem. Observe que a saída “negative” está ativa pois o output gerado é negativo.

- **Operação XOR (0110):**

	Msgs	
+ /ula_tb/S_input1	0000000000000000...	000000000000000000000000000010011
+ /ula_tb/S_input2	0000000000000000...	0000000000000000000000000000110101
+ /ula_tb/S_operation	0110	0110
+ /ula_tb/S_output	0000000000000000...	0000000000000000000000000000100110
/ula_tb/S_zero	0	
/ula_tb/S_negative	0	
/ula_tb/S_carry	0	
/ula_tb/S_overflow	0	

Acima é realizada a operação lógica XOR gerando o output observado na imagem. O output gerado é positivo mantendo assim as saídas zero e “negative” inativas.

- **Operação SLL (0111):**

	Msgs	
/ula_tb/S_input1	10 100 100000000...	10 100 1000000000000 100000000 100 11
/ula_tb/S_input2	5	5
/ula_tb/S_operation	0111	0111
/ula_tb/S_output	10000000000001...	1000000000000 100000000 100 1100000
/ula_tb/S_zero	0	
/ula_tb/S_negative	1	
/ula_tb/S_carry	0	
/ula_tb/S_overflow	0	

Acima executamos “shift left” lógico do Input 1 em 5 bits (valor de Input 2). Observe que o output gerado é negativo, ativando a saída zero.

- **Operação SRL (1000):**

	Msgs	
/ula_tb/S_input1	10 100 100000000...	10 100 1000000000000 100000000 100 11
/ula_tb/S_input2	5	5
/ula_tb/S_operation	1000	1000
/ula_tb/S_output	00000 10 100 1000...	00000 10 100 1000000000000 100000000
/ula_tb/S_zero	0	
/ula_tb/S_negative	0	
/ula_tb/S_carry	0	
/ula_tb/S_overflow	0	

Acima executamos o “shift right” lógico do Input 1 em 5 bits (valor de Input 2). Não é gerado output zero ou negativo nesse caso da imagem.

- **Operação SRA (1001):**

	Msgs	
/ula_tb/S_input1	10 100 100000000...	10 100 1000000000000 100000000 100 11
/ula_tb/S_input2	5	5
/ula_tb/S_operation	1001	1001
/ula_tb/S_output	11111 10 100 1000...	11111 10 100 1000000000000 100000000
/ula_tb/S_zero	0	
/ula_tb/S_negative	1	
/ula_tb/S_carry	0	
/ula_tb/S_overflow	0	

Acima executamos o “shift right” aritmético (com extensão de sinal) do Input 1 em 5 bits (valor de Input 2). Observe que o output gerado mantém o mesmo sinal negativo do Input 1, ativando a saída “negative”.

**Programas Utilizados para gerar o código e simulação:**

- Quartus II 13.0sp1 (64-bit) Web Edition;
- ModelSim-Altera.