Arquitetura de Computadores
EEL580

Trabalho Prática Somador

Fernando Bruzzi – DRE: 123360519
Hugo Antunes – DRE: 123143543
Pedro Lima – DRE: 123492138
Raphaela Barbosa – DRE: 123258055
Vivian Souza – DRE: 123205793

Engenharia de Computação e Informação
Universidade Federal do Rio de Janeiro - UFRJ
Julho 2024

Sumário

1 Introdução

Um somador vetorial é um componente essencial para realizar adições de números binários de forma simultânea. Este tipo de circuito é extremamente útil na computação, pois ele permite a soma rápida e eficiente de vetores de bits. A principal vantagem de um somador vetorial é a sua capacidade de operar de maneira paralela, o que significa que ele pode processar várias adições ao mesmo tempo, resultando em um aumento significativo na velocidade de processamento, quando comparado aos somadores sequenciais.

Todos os códigos utilizados nesse trabalho estão neste repositório do Github.

2 Projeto

O projeto do somador vetorial tem como objetivo desenvolver um código capaz de realizar operações de soma e subtração em vetores de diferentes tamanhos (4, 8, 16 e 32 bits). Utiliza-se a técnica de Carry-Lookahead para aumentar a eficiência das operações, pois ela permite calcular os *carries* antecipadamente, ao contrário do somador de Carry Ripple, que calcula junto a soma.

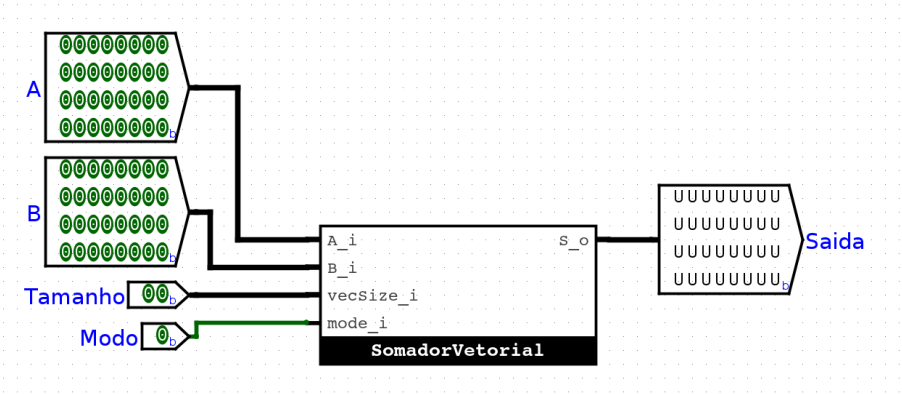

Figura 1: Módulo do somador vetorial

3 Funcionamento

O módulo possui quatro entradas:

- Dois vetores de 32 bits para os operandos, nomeados *A_i* e *B_i*.
- Um vetor de 2 bits que indica o tamanho dos números, chamado *vecSize_i*: 4 bits = 00, 8 bits = 01, 16 bits = 10 e 32 bits = 11.
- Um bit que representa a operação a ser realizada, chamado *mode_i* : Soma = 0 e Subtração = 1.

A saída será um vetor de 32 bits, chamado *S_o*, contendo o resultado da operação.

3.1 Blocos

Para que a operação seja realizada de forma correta, é necessário considerar a divisão dos vetores operandos em blocos com base no tamanho dos números, pois isso influencia a quantidade de operações e a propagação do *carry*. Especificamente, para vetores de 4 bits, serão realizadas 8 somas; para vetores de 8 bits, serão realizadas 4 somas; para vetores de 16 bits, serão realizadas 2 somas; e para vetores de 32 bits, será realizada 1 soma.

3.2 Operações

Para realizar a adição e subtração dos vetores, as entradas A e B foram combinadas bit a bit seguindo a lógica abaixo:

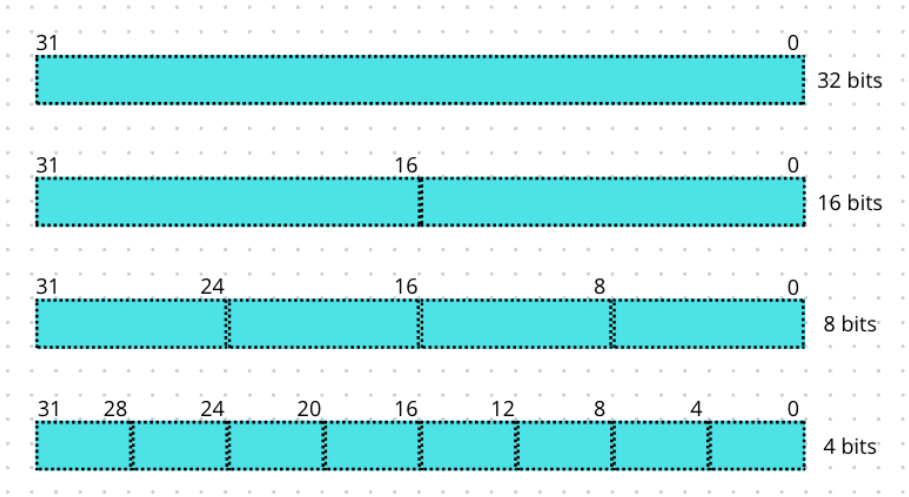

Figura 2: Divisão de blocos

A	B	Cin	Soma	Cout
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Tabela 1: Somador 1 bit

$$
\begin{aligned}
Soma &= A \cdot \bar{B} \cdot \bar{C}_{in} + \bar{A} \cdot B \cdot \bar{C}_{in} + \bar{A} \cdot \bar{B} \cdot C_{in} + A \cdot B \cdot C_{in} \\
&= \bar{C}_{in} \cdot (A \cdot \bar{B} + \bar{A} \cdot B) + C_{in} \cdot (\bar{A} \cdot \bar{B} + A \cdot B) \\
&= \bar{C}_{in} \cdot (A \oplus B) + C_{in} \cdot (A \odot B) \\
&= (A \oplus B) \oplus C_{in}
\end{aligned}
$$

$$
\begin{aligned}
C_{out} &= A \cdot B \cdot \bar{C}_{in} + \bar{A} \cdot B \cdot C_{in} + A \cdot \bar{B} \cdot C_{in} + A \cdot B \cdot C_{in} \\
&= A \cdot B(\bar{C}_{in} + C_{in}) + C_{in} \cdot (\bar{A} \cdot B + A \cdot \bar{B}) \\
&= A \cdot B + (A \oplus B) \cdot C_{in}
\end{aligned}
$$

3.2.1 Adição

Na operação de adição, para cada bit i dos vetores de entrada A e B, e o vetor auxiliar de *carry_in*, é aplicada a seguinte expressão:

$$S(i) = A_i \oplus B_i \oplus C_i$$

Além disso, para a operação ser realizada de forma correta, o início de cada bloco do nosso vetor terá *carry_in* 0.

3.2.2 Subtração

Na subtração, é necessário calcular o complemento a 2 do vetor B. Então, os bits do segundo vetor são invertidos, e o valor 1 é somado ao resultado. Para realizar essa soma na posição correta, foi definido que o *carry_in* do início de cada bloco tenha o valor 1. Por fim, a subtração é realizada de forma semelhante à adição, utilizando a expressão abaixo:

$$S_i = A_i \oplus (-B)_i \oplus C_i$$

3.2.3 Carry

O cálculo do *carry* é o ponto central para o funcionamento do projeto, pois a partir dele conseguiremos manipular se ele será propagado ou não em determinado segmento do vetor.

No início de cada bloco, o *carry* deve ter um valor pré-definido com base na operação que estamos realizando: 0 para soma e 1 para subtração. Os demais *carries* internos devem levar em consideração a propagação dos *carries* anteriores.

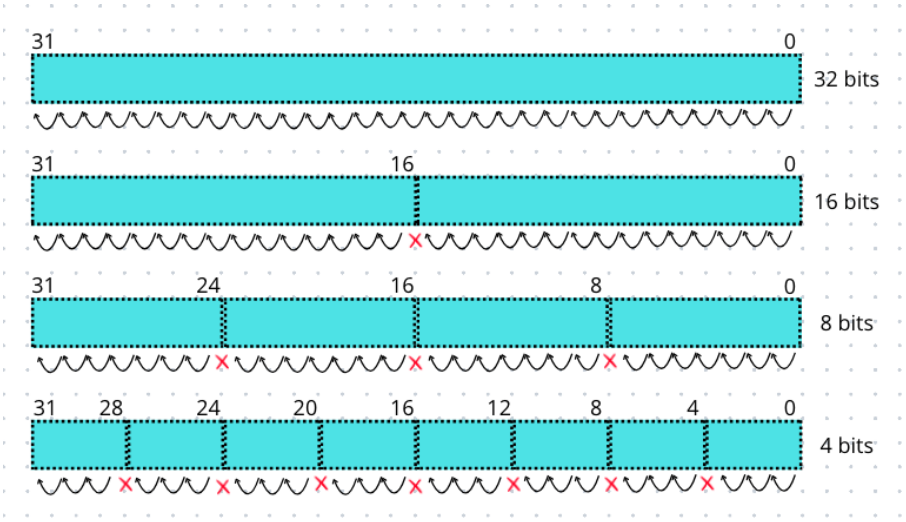

Figura 3: Propagação do carry

Para calcular os *carries* internos, utilizamos o método *Carry-Lookahead* (CLA), que calcula os *carries* de forma mais eficiente e paralela, em vez de sequencialmente. Esse método utiliza a ideia de gerar e propagar *carries* de modo que a expressão resultante dependa apenas do primeiro *carry* daquele bloco.

O CLA utiliza duas funções principais para calcular os carries de cada bit: Generate (G) e Propagate (P).

$$C_{i+1} = A_i \cdot B_i + (A_i \oplus B_i) \cdot C_i$$

- **Gerar (G)**: Um bit gera um *carry* se tanto o bit do número A quanto o bit do número B são 1. Formalmente, $G_i = A_i \cdot B_i$.
- **Propagar (P)**: Um bit propaga um *carry* se pelo menos um dos bits do número A ou do número B é 1. Formalmente, $P_i = A_i + B_i$.

$$C_{i+1} = G_i + P_i \cdot C_i$$

Com essas funções, o *carry* para cada bit C_i pode ser calculado usando as seguintes fórmulas:

$$C_0 = carry\_in$$
$$C_1 = G_0 + (P_0 \cdot C_0) =$$
$$C_2 = G_1 + (P_1 \cdot C_1) = G_1 + (P_1 \cdot (G_0 + (P_0 \cdot C_0)))$$
$$C_3 = G_2 + (P_2 \cdot C_2) = G_2 + (P_2 \cdot G_1 + (P_1 \cdot (G_0 + (P_0 \cdot C_0))))$$
$$\vdots$$

Para generalizar, o *carry* para o i-ésimo bit pode ser expresso como:

$$C_i = G_{i-1} + (P_{i-1} \cdot C_{i-1})$$

Portanto, criamos três vetores para esse processo:

- Os vetores *c_G* e *c_P*, que armazenam os valores de Carry Generate e Carry Propagate, respectivamente.
- O vetor *c_C*, que armazena os valores de *carry* para cada índice do vetor.

Atribuímos a cada índice do vetor *c_C* a expressão equivalente para o tamanho atual dos números contidos no vetor, utilizando os valores de *c_G*, *c_P* e o *"carry_0"* do bloco ao qual ele pertence.

4 Código

```
entity SomadorVetorial is
    Port ( A_i : in  STD_LOGIC_VECTOR (31 downto 0);
           B_i : in  STD_LOGIC_VECTOR (31 downto 0);
           vecSize_i : in  STD_LOGIC_VECTOR (1 downto 0); -- Tamanho: 00 -> 4,01 -> 8,
           10 -> 16 e 11 -> 32
           mode_i : in  STD_LOGIC; -- Operacao: Soma -> 0 e Subtracao -> 1
           S_o : out  STD_LOGIC_VECTOR (31 downto 0));
end SomadorVetorial;

architecture arq_SomadorVetorial of SomadorVetorial is

signal c_G      : std_logic_vector (31 downto 0); -- Carry Generate
signal c_P      : std_logic_vector (31 downto 0); -- Carry Propagate
signal c_C      : std_logic_vector (31 downto 0); -- Carry
signal aux_B    : std_logic_vector (31 downto 0); -- B auxiliar
signal aux_Soma : std_logic_vector (31 downto 0); -- Soma auxiliar

begin

aux_B <= B_i when (mode_i = '0') else
         not(B_i) when (mode_i = '1');

generate_GP: for i in 0 to 31 GENERATE
        c_G(i) <= (A_i(i) and aux_B(i));
        c_P(i) <= (A_i(i) xor aux_B(i));
end GENERATE;

--Inicio do bloco 4,8,16,32
c_C(0) <= '0' when (mode_i = '0') else
           '1' when (mode_i = '1'); --4,8,16,32

c_C(1) <= c_G(0) or (c_P(0) and c_C(0)); --4,8,16,32

c_C(2) <= c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and c_C(0)))); --4,8,16,32

c_C(3) <= c_G(2) or(c_P(2) and (c_G(1) or
        (c_P(1) and (c_G(0) or (c_P(0) and c_C(0)))))); --4,8,16,32

--Inicio do bloco 4
c_C(4) <= '0' when (mode_i = '0' and vecSize_i = "00") else
    '1' when (mode_i = '1' and vecSize_i = "00") else -- 4
    c_G(3) or (c_P(3) and
    (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1)
    and (c_G(0) or (c_P(0) and c_C(0))))))); --8,16,32

c_C(5) <=c_G(4) or (c_P(4) and c_C(4)) when(vecSize_i = "00") else      -- 4
    c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and
    (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and c_C(0)))))))))); --8,16,32

c_C(6) <=c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and c_C(4))))
    when(vecSize_i = "00") else -- 4
    c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or
    (c_P(2) and (c_G(1) or (c_P(1) and (c_G(0)
    or (c_P(0) and c_C(0))))))))))); --8,16,32

c_C(7) <=c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and
c_C(4)))))) when(vecSize_i = "00") else    --4
```

```
    c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or
    (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or
    (c_P(1) and (c_G(0) or (c_P(0) and c_C(0))))))))))))); --8,16,32

--Inicio do bloco 4,8
c_C(8) <= '0' when (mode_i = '0' and (vecSize_i = "00" or vecSize_i = "01")) else
    '1' when (mode_i = '1' and (vecSize_i = "00" or vecSize_i = "01")) else --4,8
    c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and
    (c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or
    (c_P(1) and (c_G(0) or (c_P(0) and c_C(0))))))))))))); --16,32

c_C(9) <= c_G(8) or (c_P(8) and c_C(8)) when (vecSize_i = "00" or vecSize_i =
"01") else --4,8
    c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or
    (c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1)
    or (c_P(1) and (c_G(0) or (c_P(0) and c_C(0))))))))))))); -- 16,32

c_C(10) <=c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and c_C(8))))
    when (vecSize_i = "00" or vecSize_i = "01") else --4,8
    c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and
    (c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3)
    or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and
    (c_G(0) or (c_P(0) and c_C(0))))))))))))); -- 16,32

c_C(11) <=c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or
    (c_P(8) and c_C(8)))))) when (vecSize_i = "00" or vecSize_i = "01")else--4,8
    c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and
    (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or
    (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1)
    and (c_G(0) or (c_P(0) and c_C(0))))))))))))); -- 16,32

--Inicio do bloco 4
c_C(12) <='0' when (mode_i = '0' and vecSize_i = "00") else
    '1' when (mode_i = '1' and vecSize_i = "00") else-- 4
    c_G(11) or(c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8)
    and c_C(8)))))))) when (vecSize_i = "01") else --8
    c_G(11) or(c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8)
    or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3)
    or (c_P(1) and (c_G(0) or (c_P(0) and c_C(0))))))))))))))));--16,32


c_C(13) <=c_G(12) or (c_P(12) and c_C(12)) when(vecSize_i = "00") else --4
    c_G(12) or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10) or(c_P(10) and
    (c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and c_C(8))))))))) when (vecSize_i = "01")
    else--8
    c_G(12) or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9)
    and (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or
    (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or
    (c_P(2) and (c_G(1) or (c_P(1) and
    (c_G(0) or (c_P(0) and c_C(0))))))))))))))));--16,32

c_C(14) <=c_G(13) or(c_P(13) and(c_G(12) or (c_P(12) and c_C(12))))
when(vecSize_i = "00")
    else        --4
    c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10) or
    (c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8)
    and c_C(8)))))))))))) when (vecSize_i = "01") else--8
    c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10) or
    (c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6)
    or (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or
    (c_P(1) and (c_G(0)
```

```
    or (c_P(0) and c_C(0))))))))))))))))))))))))))))); --16,32

c_C(15) <=c_G(14) or(c_P(14) and(c_G(13) or(c_P(13) and
    (c_G(12) or (c_P(12) and c_C(12)))))) when(vecSize_i = "00") else   --4
    c_G(14) or(c_P(14) and(c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or
    (c_P(11)
    and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and c_C(8)))))))))
    when (vecSize_i = "01") else--8
    c_G(14) or(c_P(14) and(     c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and
    (c_G(11) or(c_P(11)
    and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7)
    or (c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and
    (c_G(3) or (c_P(3) and
    (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))); --16,32

--inicio do bloco 4,8,16
c_C(16) <='0' when (mode_i = '0' and ((vecSize_i = "00" or vecSize_i = "01")
    or vecSize_i = "11" )) else
    '1' when (mode_i = '1' and ((vecSize_i = "00" or vecSize_i = "01") or vecSize_i = "11"
    else --4,8,16
    c_G(15) or(c_P(15) and(c_G(14) or(c_P(14) and(      c_G(13) or(c_P(13) and
    (c_G(12) or(c_P(12)
    and(c_G(11) or(c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9)
    and(c_G(8) or (c_P(8)
    and (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or
    (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and
    (c_G(1) or (c_P(1) and (c_G(0)
    or (c_P(0) and c_C(0))))))))))))))))))))))))))))); --32

c_C(17) <=c_G(16) or (c_P(16) and c_C(16)) when(vecSize_i = "00"
    or (vecSize_i = "01" or vecSize_i = "10" )) else--4,8,16
    c_G(16) or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or(c_P(14)
    and(       c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or
    (c_P(11) and(c_G(10) or
    (c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6)
    or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3)
    and (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))); --32

c_C(18) <=c_G(17) or(c_P(17) and(c_G(16) or (c_P(16) and c_C(16))))
    when(vecSize_i = "00" or (vecSize_i = "01" or vecSize_i = "10" )) else--4,8,16
    c_G(17) or(c_P(17) and(     c_G(16) or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or
    (c_P(14) and(       c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or
    (c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or
    (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or
    (c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or
    (c_P(2) and (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))); --32

c_C(19) <=c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or
    (c_P(16) and c_C(16)))))) when(vecSize_i = "00" or (vecSize_i = "01" or vecSize_i = "1
    c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and(      c_G(16) or(c_P(16)
    and(c_G(15) or(c_P(15) and(c_G(14) or(c_P(14) and(  c_G(13) or(c_P(13)
    and(c_G(12) or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10) or(c_P(10)
    and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6)
    or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or
    (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and (c_G(0) or
    (c_P(0) and c_C(0))))))))))))))))))))))))))))))); --32

--Inicio do bloco 4
```

```
c_C(20) <='0' when (mode_i = '0' and vecSize_i = "00") else '1'
    when (mode_i = '1' and vecSize_i = "00") else --4
    c_G(19) or(c_P(19) and
    (c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or (c_P(16) and
    c_C(16)))))))) when(vecSize_i = "01" or vecSize_i = "10" )else--8,16
    c_G(19) or(c_P(19) and(c_G(18) or(c_P(18) and(      c_G(17) or(c_P(17) and
    (   c_G(16) or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or(c_P(14)
    and(        c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or(c_P(11)
    and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or
    (c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4)
    and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or
    (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))));--32

c_C(21) <=c_G(20) or (c_P(20) and c_C(20)) when(vecSize_i = "00") else--4
    c_G(20) or(c_P(20) and(c_G(19) or(c_P(19) and(c_G(18) or(c_P(18) and(c_G(17)
    or(c_P(17) and(c_G(16) or (c_P(16) and c_C(16))))))))) when(vecSize_i = "01"
    or vecSize_i = "10" )else--8,16
    c_G(20) or(c_P(20) and(     c_G(19) or(c_P(19) and(c_G(18) or(c_P(18) and(  c_G(17)
    or(c_P(17) and(     c_G(16) or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14)
    or(c_P(14) and(     c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11)
    or(c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and
    (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or
    (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and (c_G(0
    (c_P(0) and c_C(0))))))))))))))))))))))))))))))));--32

c_C(22) <=c_G(21) or(c_P(21) and(c_G(20) or (c_P(20) and c_C(20))))
    when(vecSize_i = "00") else--4
    c_G(21) or(c_P(21) and(     c_G(20) or(c_P(20) and(c_G(19) or(c_P(19) and(c_G(18)
    or(c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or (c_P(16) and c_C(16))))))))))
    when(vecSize_i = "01" or vecSize_i = "10" )else--8,16
    c_G(21) or(c_P(21) and
    (   c_G(20) or(c_P(20) and( c_G(19) or(c_P(19) and(c_G(18) or(c_P(18)
    and(        c_G(17) or(c_P(17) and( c_G(16) or(c_P(16) and(c_G(15) or(c_P(15)
    and(c_G(14) or(c_P(14) and( c_G(13) or(c_P(13) and(c_G(12) or(c_P(12)
    and(c_G(11) or(c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and
    (c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5)
    and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or
    (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))));--32

c_C(23) <=c_G(22) or(c_P(22) and(c_G(21) or(c_P(21) and(c_G(20)
    or (c_P(20) and c_C(20)))))) when(vecSize_i = "00") else--4
    c_G(22) or(c_P(22) and
    (c_G(21) or(c_P(21) and(    c_G(20) or(c_P(20) and(c_G(19) or(c_P(19)
    and(c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or (c_P(16)
    and c_C(16)))))))))))))) when(vecSize_i = "01" or vecSize_i = "10" )else--8,16
    c_G(22) or(c_P(22) and(     c_G(21) or(c_P(21) and( c_G(20) or(c_P(20) and( c_G(19)
    or(c_P(19) and(c_G(18) or(c_P(18) and(      c_G(17) or(c_P(17) and( c_G(16)
    or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or(c_P(14) and(       c_G(13)
    or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10)
    or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or
    (c_P(7) and(c_G(6)
    or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and
    (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))));--32

--Inicio do bloco 4,8
c_C(24) <='0' when (mode_i = '0' and
    (vecSize_i = "00" or vecSize_i = "01")) else
    '1' when (mode_i = '1' and (vecSize_i = "00" or vecSize_i = "01")) else --4,8
```

```
    c_G(23) or(c_P(23) and(c_G(22) or(c_P(22) and(      c_G(21) or(c_P(21) and( c_G(20)
    or(c_P(20) and(c_G(19) or(c_P(19) and(c_G(18) or(c_P(18) and(c_G(17) or
    (c_P(17) and(c_G(16) or (c_P(16) and c_C(16))))))))))))))))
    when(vecSize_i = "10" )else--16
    c_G(23) or(c_P(23) and(c_G(22) or(c_P(22) and(      c_G(21) or(c_P(21) and( c_G(20)
    or(c_P(20) and(     c_G(19) or(c_P(19) and(c_G(18) or(c_P(18) and(  c_G(17)
    or(c_P(17) and(     c_G(16) or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or
    (c_P(14) and(       c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or
    (c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and
    (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or
    (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or
    (c_P(1) and
    (c_G(0) or (c_P(0)
    and
    c_C(0))))))))))))))))))))))))))))))))))))))))))))))))))))));--32

    c_C(25) <=c_G(24) or(c_P(24) and(c_C(24)))when(vecSize_i = "00" or vecSize_i = "01")els
    c_G(24) or(c_P(24) and(     c_G(23) or(c_P(23) and(c_G(22) or(c_P(22) and(  c_G(21)
    or(c_P(21) and(     c_G(20) or(c_P(20) and(c_G(19) or(c_P(19) and(c_G(18) or
    (c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or (c_P(16) and
    c_C(16))))))))))))))))) when(vecSize_i = "10" )else--16
    c_G(24) or(c_P(24) and(     c_G(23) or(c_P(23) and(c_G(22) or(c_P(22) and(  c_G(21)
    or(c_P(21) and(     c_G(20) or(c_P(20) and( c_G(19) or(c_P(19) and(c_G(18)
    or(c_P(18) and(     c_G(17) or(c_P(17) and( c_G(16) or(c_P(16) and(c_G(15)
    or(c_P(15) and(c_G(14) or(c_P(14) and(      c_G(13) or(c_P(13) and(c_G(12)
    or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or
    (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5)
    or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and
    (c_G(1)
    or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))))))))))))))))))))))))));--32

c_C(26) <=c_G(25) or(c_P(25) and(c_G(24) or(c_P(24) and(c_C(24)))))when
    (vecSize_i = "00" or vecSize_i = "01")else--4,8
    c_G(25) or(c_P(25) and(c_G(24) or(c_P(24) and(      c_G(23) or(c_P(23)
    and(c_G(22) or(c_P(22) and
    (c_G(21) or(c_P(21) and(c_G(20) or(c_P(20) and(c_G(19) or(c_P(19)
    and(c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or (c_P(16)
    and c_C(16))))))))))))))))) when(vecSize_i = "10" )else--16
    c_G(25) or(c_P(25) and(     c_G(24) or(c_P(24) and( c_G(23) or(c_P(23) and(c_G(22)
    or(c_P(22) and(     c_G(21) or(c_P(21) and( c_G(20) or(c_P(20) and( c_G(19)
    or(c_P(19) and(c_G(18) or(c_P(18) and(      c_G(17) or(c_P(17) and( c_G(16)
    or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or(c_P(14) and(       c_G(13)
    or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10) or
    (c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and
    (c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3)
    and (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))))))))))))))))))))))))));--32

c_C(27) <=c_G(26) or(c_P(26) and(c_G(25) or(c_P(25) and(c_G(24) or(c_P(24)
    and(c_C(24)))))))when(vecSize_i = "00" or vecSize_i = "01")else--4,8
    c_G(26) or(c_P(26) and(c_G(25) or(c_P(25) and(c_G(24) or(c_P(24) and(       c_G(23)
    or(c_P(23) and(c_G(22) or(c_P(22) and(      c_G(21) or(c_P(21) and( c_G(20) or
    (c_P(20) and(c_G(19) or(c_P(19) and(c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and
    (c_G(16) or (c_P(16) and
    c_C(16))))))))))))))))))))))))))))))))) when(vecSize_i = "10" )else--16
    c_G(26) or(c_P(26) and(     c_G(25) or(c_P(25) and( c_G(24) or(c_P(24) and
    (c_G(23) or(c_P(23)
    and(c_G(22) or(c_P(22) and( c_G(21) or(c_P(21) and( c_G(20) or(c_P(20) and( c_G(19)
    or(c_P(19) and(c_G(18) or(c_P(18) and(      c_G(17) or(c_P(17) and( c_G(16) or
    (c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or(c_P(14) and( c_G(13) or(c_P(13)
```

```
    and(c_G(12) or(c_P(12) and(c_G(11) or(c_P(11) and(c_G(10) or(c_P(10) and
    (c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6)
    and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3)
    and (c_G(2) or(c_P(2)
    and (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))))))))))))))))))))))))))));--32

--Inicio do bloco 4
c_C(28) <='0' when (mode_i = '0' and vecSize_i = "00") else
    '1' when (mode_i = '1' and vecSize_i ="00") else --4
    c_G(27) or(c_P(27) and(c_G(26) or(c_P(26) and(c_G(25) or(c_P(25) and(c_G(24)
    or(c_P(24) and(c_C(24))))))))when(vecSize_i = "01" )else--8   c_G(27) or(c_P(27)
    and (c_G(26) or(c_P(26) and(c_G(25) or(c_P(25) and(c_G(24) or(c_P(24) and
    (c_G(23) or(c_P(23)
    and(c_G(22) or(c_P(22) and( c_G(21) or(c_P(21) and( c_G(20) or(c_P(20) and(c_G(19)
    or(c_P(19) and(c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or
    (c_P(16) and c_C(16))))))))))))))))))))))))) when(vecSize_i = "10" )else--16
    c_G(27) or(c_P(27) and(c_G(26) or(c_P(26) and(      c_G(25) or(c_P(25) and( c_G(24)
    or
    (c_P(24) and(       c_G(23) or(c_P(23) and(c_G(22) or(c_P(22) and(  c_G(21) or
    (c_P(21) and(       c_G(20) or
    (c_P(20) and(       c_G(19) or(c_P(19) and(c_G(18) or(c_P(18)
    and(        c_G(17) or(c_P(17) and( c_G(16) or(c_P(16) and(c_G(15) or(c_P(15) and
    (c_G(14) or(c_P(14) and(c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or
    (c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and
    (c_G(7) or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4)
    and (c_G(3) or (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and
    (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))))))))))))))))))))))))))));--32

c_C(29) <=c_G(28) or(c_P(28) and(c_C(28)))when(vecSize_i = "00" )else--4
    c_G(28) or(c_P(28) and(c_G(27)
    or(c_P(27) and(c_G(26) or(c_P(26) and(c_G(25) or(c_P(25) and(c_G(24) or
    (c_P(24) and(c_C(24))))))))))when(vecSize_i = "01" )else--8
    c_G(28) or(c_P(28) and(c_G(27) or(c_P(27) and(c_G(26) or(c_P(26) and(c_G(25) or
    (c_P(25) and(c_G(24) or(c_P(24) and(       c_G(23) or(c_P(23) and(c_G(22) or
    (c_P(22) and(       c_G(21) or(c_P(21) and( c_G(20) or(c_P(20) and
    (c_G(19) or(c_P(19)
    and(c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or (c_P(16) and
    c_C(16))))))))))))))))))))))))))) when(vecSize_i = "10" )else--16
    c_G(28) or(c_P(28) and(     c_G(27) or(c_P(27) and(c_G(26) or(c_P(26) and
    (c_G(25) or
    (c_P(25) and(c_G(24) or(c_P(24) and(       c_G(23) or(c_P(23) and(c_G(22) or
    (c_P(22) and(       c_G(21) or(c_P(21) and( c_G(20) or(c_P(20) and( c_G(19) or
    (c_P(19) and(c_G(18) or(c_P(18) and(       c_G(17) or(c_P(17) and( c_G(16) or
    (c_P(16) and(c_G(15) or(c_P(15)
    and(c_G(14) or(c_P(14) and( c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11)
    or(c_P(11) and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or
    (c_P(8) and (c_G(7)
    or(c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and
    (c_G(3)
    or (c_P(3) and (c_G(2) or(c_P(2) and
    (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))))))))))))))))))))))))))));--32


c_C(30) <=c_G(29) or(c_P(29) and(c_G(28) or(c_P(28) and(c_C(28)))))
    when(vecSize_i = "00" )else--4
    c_G(29) or(c_P(29) and(c_G(28) or(c_P(28) and(c_G(27) or(c_P(27) and(c_G(26) or
    (c_P(26) and(c_G(25) or(c_P(25) and(c_G(24) or(c_P(24) and
    (c_C(24))))))))))))))))))))))))when(vecSize_i = "01" )else--8
```

```
    c_G(29) or(c_P(29) and(c_G(28) or(c_P(28) and(c_G(27) or(c_P(27) and(c_G(26) or
    (c_P(26) and(c_G(25) or(c_P(25) and(c_G(24) or(c_P(24) and( c_G(23) or(c_P(23) and
    (c_G(22) or(c_P(22) and(c_G(21) or(c_P(21) and(     c_G(20) or(c_P(20) and
    (c_G(19) or(c_P(19)
    and(c_G(18) or(c_P(18) and(c_G(17) or(c_P(17) and(c_G(16) or (c_P(16) and
    c_C(16))))))))))))))))))))))))))))))))
    when(vecSize_i = "10" )else--16
    c_G(29) or(c_P(29) and(c_G(28) or(c_P(28) and(      c_G(27) or(c_P(27) and(c_G(26) or
    (c_P(26) and(       c_G(25) or
    (c_P(25) and(       c_G(24) or(c_P(24) and( c_G(23) or
    (c_P(23) and(c_G(22) or(c_P(22) and(        c_G(21) or(c_P(21) and( c_G(20) or
    (c_P(20) and(       c_G(19) or(c_P(19) and(c_G(18) or(c_P(18) and(  c_G(17) or
    (c_P(17) and(       c_G(16) or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or(c_P(14)
    and(        c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and
    (c_G(11) or(c_P(11) and(c_G(10)
    or(c_P(10) and(c_G(9) or
    (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or(c_P(7) and(c_G(6)
    or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and (c_G(3) or (c_P(3) and
    (c_G(2)
    or(c_P(2) and
    (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0) and
    c_C(0))))))))))))))))))))))))))))))))))))))))))))))))))));--32

c_C(31) <=c_G(30) or(c_P(30) and(c_G(29) or(c_P(29) and(c_G(28) or(c_P(28)
    and(c_C(28)))))))when(vecSize_i = "00" )else--4
    c_G(30) or(c_P(30) and(c_G(29) or(c_P(29) and(c_G(28) or(c_P(28) and(c_G(27)
    or(c_P(27) and(c_G(26) or(c_P(26) and(c_G(25) or(c_P(25) and(c_G(24) or
    (c_P(24) and(c_C(24))))))))))))))
    when(vecSize_i = "01" )else--8
    c_G(30) or(c_P(30) and(c_G(29)
    or(c_P(29) and(c_G(28) or(c_P(28)
    and(c_G(27)
    or(c_P(27) and(c_G(26) or
    (c_P(26)
    and(c_G(25) or(c_P(25) and(c_G(24) or
    (c_P(24) and(c_G(23) or(c_P(23)
    and(c_G(22) or(c_P(22) and(c_G(21) or
    (c_P(21) and(c_G(20) or(c_P(20) and(c_G(19) or(c_P(19) and(c_G(18) or(c_P(18)
    and(c_G(17) or(c_P(17) and(c_G(16) or (c_P(16) and
    c_C(16))))))))))))))))))))))))))))))))
    when(vecSize_i = "10" )else--16
    c_G(30) or(c_P(30) and(c_G(29) or(c_P(29) and(c_G(28) or(c_P(28) and
    (    c_G(27) or
    (c_P(27) and(c_G(26) or(c_P(26) and(c_G(25) or(c_P(25) and(c_G(24) or(c_P(24)
    and(c_G(23) or(c_P(23) and(c_G(22) or(c_P(22) and(  c_G(21) or(c_P(21) and
    (c_G(20) or(c_P(20) and(c_G(19) or(c_P(19) and(c_G(18) or(c_P(18) and(c_G(17)
    or(c_P(17) and(c_G(16) or(c_P(16) and(c_G(15) or(c_P(15) and(c_G(14) or
    (c_P(14) and(c_G(13) or(c_P(13) and(c_G(12) or(c_P(12) and(c_G(11) or(c_P(11)
    and(c_G(10) or(c_P(10) and(c_G(9) or (c_P(9) and(c_G(8) or (c_P(8) and (c_G(7) or
    (c_P(7) and(c_G(6) or(c_P(6) and(c_G(5) or(c_P(5) and(c_G(4) or (c_P(4) and
    (c_G(3) or
    (c_P(3) and (c_G(2) or(c_P(2) and (c_G(1) or (c_P(1) and (c_G(0) or (c_P(0)
    and c_C(0))))))))))))))))))))))))))))))))))))))))))))))))))
    ))))))))));--32

generate_soma: for j in 0 to 31 GENERATE
        aux_soma(j) <= ((A_i(j) xor aux_B(j))xor c_C(j));
end GENERATE;

S_o <= aux_soma;
```

```
end arq_SomadorVetorial;
```

Referências

[1] David A. Patterson e John L. Hennessy. *Computer Organization and Design RISC-V Edition: The Hardware Software Interface*. 2nd. Morgan Kaufmann, 2021. ISBN: 978-0-12-820331-6.