

O arquivo **.cpp** seguiu a seguinte estrutura:

```
#include <iostream>

using namespace std;

int multiply(int a, int b){
    int acumula = 0;
    int vezes = b;

    // se B é negativo, troca o sinal
    if(b < 0)
        vezes = 0 - vezes;

    // acumula A, B vezes
    while(vezes!=0){
        acumula = acumula + a;
        vezes = vezes - 1;
    }

    // se B é negativo, trocamos o sinal do resultado
    if(b < 0)
        acumula = 0 - acumula;
    return acumula;
}

int main() {
    // Definindo os vetores A e B
    int A[] = {10, -2, 3, -5, 12, 6, -7, 8};
    int B[] = {11, -3, 5, 22, -4, 9, -8, 19};

    // Tamanho dos vetores
    int n = 8;

    // Inicializando os vetores C e D
    int C[n] = {};
    int D[n] = {};

    //inicializando variáveis
    int max = C[0];
    int SM = 0;
```

```

// Calculando os vetores C e D
for (int i = 0; i < n; ++i) {
    C[i] = A[i] + B[i];
    D[i] = A[i] - B[i];
}

// verificando o valor máximo entre os elementos de C e D
for(int i = 0; i < n; i++){
    if(C[i] > max)
        max = C[i];

    if(D[i] > max)
        max = D[i];
}

// Calculando o somatório dos elementos de C e D
for(int i = 0; i < n; i++){
    SM += C[i] + D[i];
}

//cálculo final
SM = multiply(SM, max);

// Imprimindo os resultados (não será traduzido para assembly)
cout << "C = {";
for (int i = 0; i < n; ++i) {
    cout << C[i];
    if (i < n - 1) {
        cout << ", ";
    }
}
cout << "}\n";

cout << "D = {";
for (int i = 0; i < n; ++i) {
    cout << D[i];
    if (i < n - 1) {
        cout << ", ";
    }
}
}

```

```

cout << "}\n";

cout << "max:" << max << endl;
cout << "SM = " << SM << "\n";

return 0;
}

```

VARIÁVEIS ASSEMBLY	VARIÁVEIS C++ (inicialmente)
\$t0	n
\$t1	A
\$t2	B
\$t3	C
\$t4	D
\$t5	i
\$t6	A[ i ]
\$t7	B[ i ]
\$t8	C [ i ]
\$t9	D [ i ]

Conforme a sequência do código, as variáveis em assembly assumem novas definições. Todas as mudanças estão explicadas como comentário no arquivo **tp4.asm** para melhor entendimento. Contudo, ressalto que o resultado final está contido no registrador \$t9 (SM). A fórmula da especificação 3 é:

$$SM = \max(C, D) \cdot \left( \sum_{l=0}^n C_l + \sum_{l=0}^n D_l \right) = \max(C, D) \cdot (C_0 + C_1 + C_2 + \dots + C_{n-1} + D_0 + D_1 + D_2 + \dots + D_{n-1})$$

Vetor A: {10, -2, 3, -5, 12, 6, -7, 8}

Vetor B:{11, -3, 5, 22, -4, 9, -8, 19}

Vetor C (A + B): 21, -5, 8, 17, 8, 15, -15, 27

Vetor D (A - B): -1, 1, -2, -27, 16, -3, 1, -11

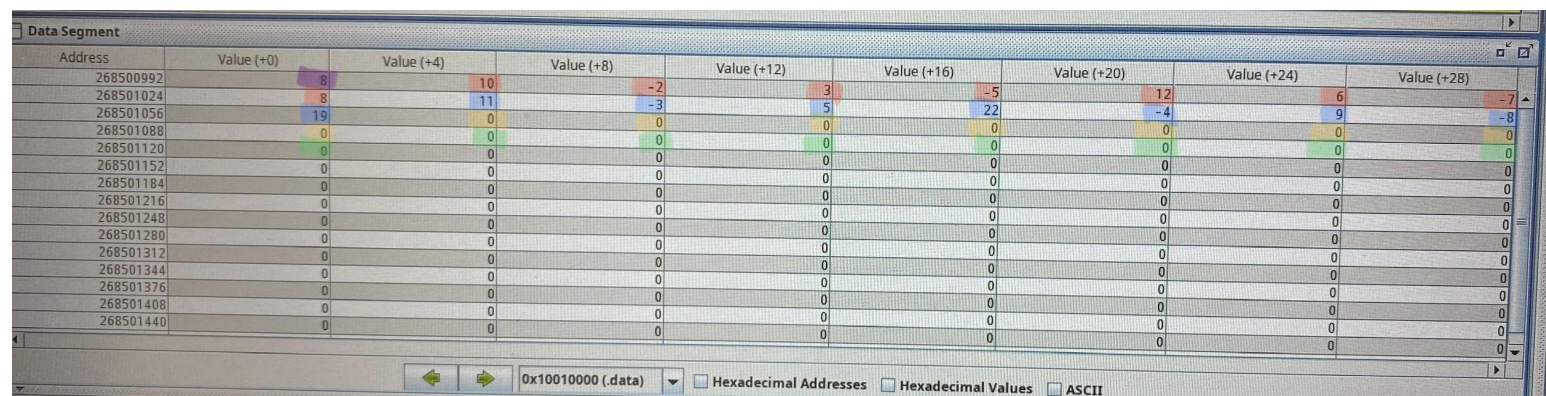
n: 8

max = 27

SM = 1350

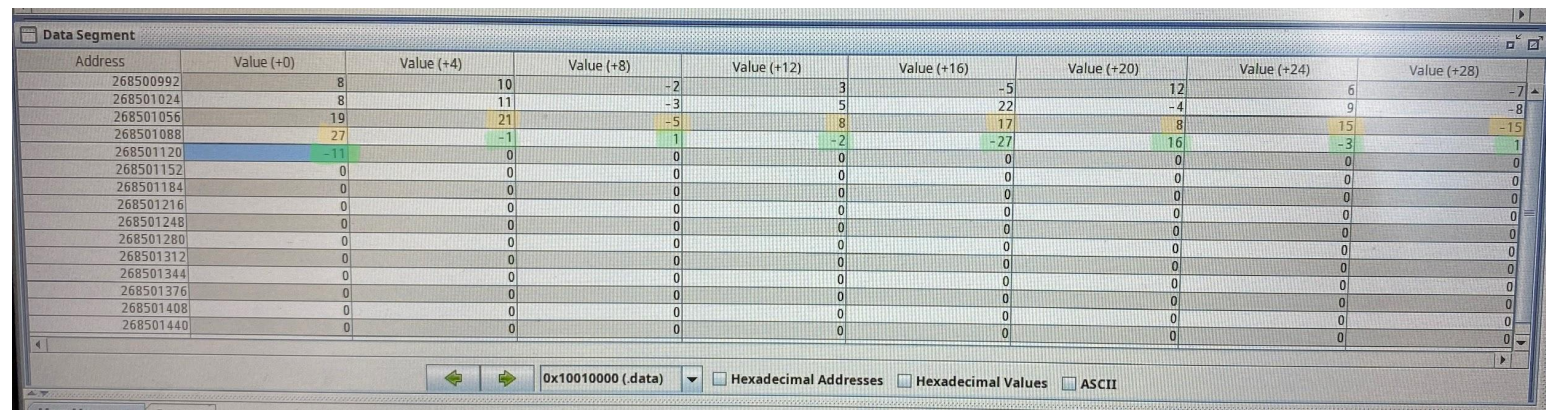
Os valores acima estão destacados nas imagens da seguinte forma: **n** - roxo, **A** - vermelho, **B** - azul, **C** - amarelo, **D** - verde.

Imagem 1, consta os vetores A e B com seus respectivos valores:



Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
268500992	8	10	-2	3	-5	12	6	-7
268501024	8	11	-3	5	22	-4	9	-8
268501056	19	0	0	0	0	0	0	0
268501088	0	0	0	0	0	0	0	0
268501120	0	0	0	0	0	0	0	0
268501152	0	0	0	0	0	0	0	0
268501184	0	0	0	0	0	0	0	0
268501216	0	0	0	0	0	0	0	0
268501248	0	0	0	0	0	0	0	0
268501280	0	0	0	0	0	0	0	0
268501312	0	0	0	0	0	0	0	0
268501344	0	0	0	0	0	0	0	0
268501376	0	0	0	0	0	0	0	0
268501408	0	0	0	0	0	0	0	0
268501440	0	0	0	0	0	0	0	0

Imagem 2, consta os vetores C e D após a realização das operações:



Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)
268500992	8	10	-2	3	-5	12	6	-7
268501024	8	11	-3	5	22	-4	9	-8
268501056	19	21	-5	8	17	8	15	-15
268501088	27	-1	1	-2	-27	16	-3	1
268501120	-11	0	0	0	0	0	0	0
268501152	0	0	0	0	0	0	0	0
268501184	0	0	0	0	0	0	0	0
268501216	0	0	0	0	0	0	0	0
268501248	0	0	0	0	0	0	0	0
268501280	0	0	0	0	0	0	0	0
268501312	0	0	0	0	0	0	0	0
268501344	0	0	0	0	0	0	0	0
268501376	0	0	0	0	0	0	0	0
268501408	0	0	0	0	0	0	0	0
268501440	0	0	0	0	0	0	0	0

Imagem 3, início da simulação no modelsim com a criação dos vetores C e D:



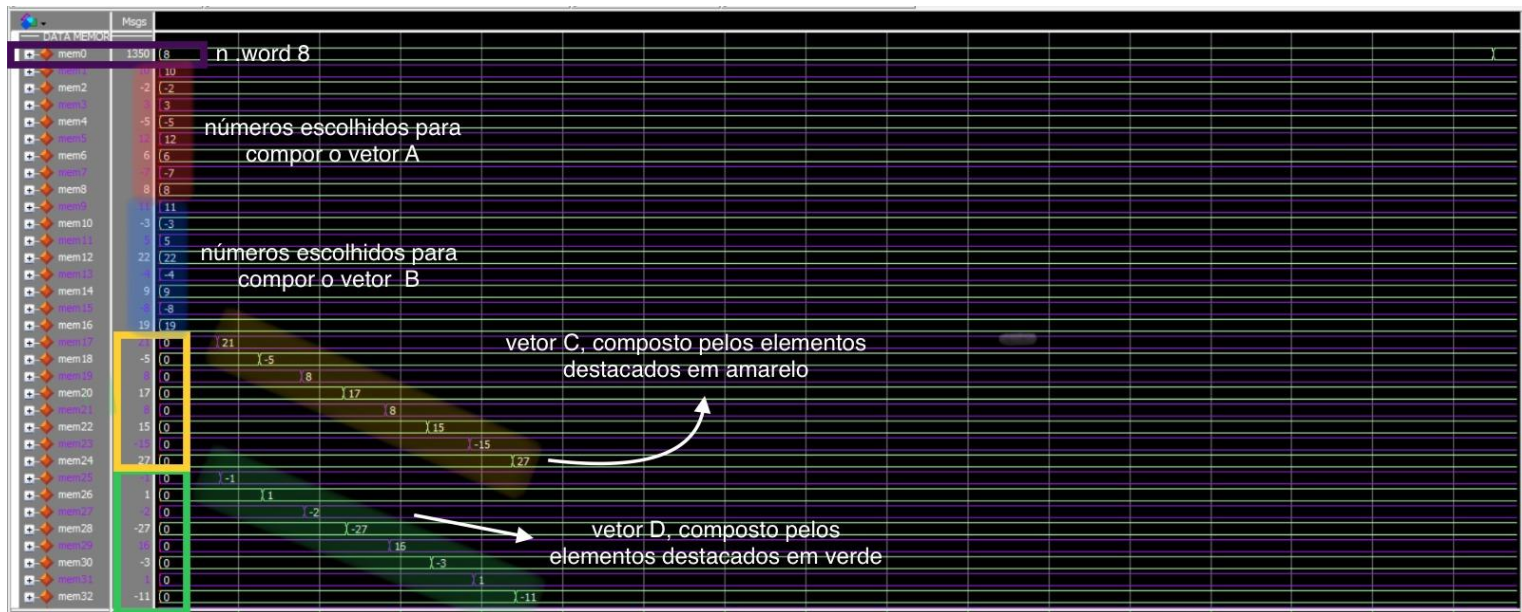


Imagem 4, simulação completa no modelsim:

