Cycle Calculation

1. Overall Structure

- **Outer Loop (Row Loop):** Iterates over the rows of matrix A, totaling 20 rows.
- **Middle Loop (Column Loop):** Iterates over the columns of matrix B, totaling 50 columns.
- **Inner Loop (Shared Dimension Loop):** Processes the shared dimension using vector instructions with a maximum vector length of 32.

2. Iteration Counts for Each Loop

- **Outer Loop:** 20 iterations.
  - Each iteration takes 7 cycles.
- **Middle Loop:** Each outer loop iteration has 50 iterations.
  - Includes 6 scalar instructions, each iteration takes 13 cycles.
- **Inner Loop (for each (i, j)):**
  - **Total Number of Shared Dimension Elements:** 46
  - **Number of Iterations:**
    - **First Iteration:** VL = 32
    - **Second Iteration:** VL = 14 (46 - 32 = 14)
  - **Total:** 2 iterations

3. Instruction Cycle Estimation

3.1 Inner Loop (`k_loop`) Cycle Count

```
    bge x16, x6, end_k_loop
    sub x26, x6, x16
    mul t4, x16, x4
    add t6, x3, t4
    vsetvl t5, x26, x4
    mul t4, x16, x4
    add t6, t3, t4
    add x16, x16, t5
    j k_loop
```

A total of 9 scalar instructions, counted as 9 cycles. (Using Data Forwarding; the source code has been reordered to avoid pipeline bubbles; this is an approximation.)

First Iteration (VL = 32)

```
    vlw.v v0, 0(t6)   # Executes in 32 cycles, delayed by 32 cycles
    vlw.v v1, 0(t6)   # Executes in 32 cycles, delayed by 32 cycles
    vmul.vv v2, v0, v1    # Executes in 32 cycles, delayed by 4 cycles
    vredsum x21, v2, x21    # Executes in 32 cycles
```

Total vector instruction cycles for the first iteration: 64 + 64 + 36 + 32 = **196 cycles**

Second Iteration (VL = 14)

```
    vlw.v v0, 0(t6)   # Executes in 14 cycles, delayed by 32 cycles
    vlw.v v1, 0(t6)   # Executes in 14 cycles, delayed by 32 cycles
    vmul.vv v2, v0, v1    # Executes in 14 cycles, delayed by 4 cycles
    vredsum x21, v2, x21    # Executes in 14 cycles
```

Total vector instruction cycles for the second iteration: 46 + 46 + 18 + 14 = **124 cycles**

Total Cycle Count

- **First Iteration Total Cycles:** 9 (scalar) + 196 (vector) = **205 cycles**
- **Second Iteration Total Cycles:** 9 (scalar) + 124 (vector) = **133 cycles**

4.1 Total Cycle Count for Each (i, j)

- **Inner Loop Cycles:** 338 cycles
- **Middle Loop Cycles:** 13 cycles
- **Total:** 338 + 13 = **351 cycles**

4.2 Total Number of (i, j) Combinations

- Inner and middle loop counts: 20 (rows) × 50 (columns) = 1,000
- **Total Inner and Middle Loop Cycles:** 351 × 1,000 = **351,000 cycles**

4.4 Outer Loop Cycle Count

- **Cycles per Outer Loop (for each i):** 7 cycles
- **Total:** 7 × 20 = **140 cycles**

4.5 Total Cycle Count

- **Total Matrix Multiplication Cycles:** 351,000 cycles
- **Outer Loop Cycles:** 140 cycles
- **Total:** 351,000 + 140 = **351,140 cycles**

Test Code and Result

1. Test Code `./test/cli/matrix/program.S`

Note: Custom system calls have been added to qtrvsim to enable console printing of integers and characters.

Test command: `./target/qtrvsim_cli --os-emulation --asm --pipelined --dump-registers ./tests/cli/matrix/program.S`

Added `--os-emulation` to enable system calls, `--asm` to output assembly code, `--pipelined` to simulate pipelined execution, and `--dump-registers` to output register information (already added output for vector registers).

Source Code

```
.globl _start

    .text
    _start:
            # 初始化向量寄存器
            li x11, 32                 # 向量长度（最大为32元素）
            li x4, 4                   # 数据单元大小（4字节/整数）

            # 矩阵维度
            li x6, 46                  # 中间维度
            li x7, 20                  # 矩阵1行数
            li x8, 50                  # 矩阵2列数
            # li x6, 4                   # 假设中间维度为4
            # li x7, 3                   # 假设矩阵1行数为2
            # li x8, 3                   # 假设矩阵2列数为3

            # 打印矩阵维度
            mv a0, x7                  # 矩阵1行数
            li a7, 244                 # syscall 编号: 1 (print integer)
            ecall                     # 调用系统打印值

            li a0, 32                  # ASCII 32 = 空格
            li a7, 245                 # syscall 编号: 11 (print char)
            ecall

            mv a0, x8                  # 矩阵2列数
            li a7, 244                 # syscall 编号: 1 (print integer)
            ecall                     # 调用系统打印值

            li a0, 10                  # ASCII 10 = 换行
            li a7, 245                 # syscall 编号: 11 (print char)
            ecall

            # 矩阵地址
            la x9, matrix1             # 矩阵1基址
            la x10, matrix2            # 矩阵2基址
            la x12, result             # 结果矩阵基址
            la x18, temp               # 暂存矩阵基址

            # 初始化结果矩阵为0
            li x13, 0
            li x14, 0                  # 行偏移
    zero_loop:
            li x15, 0                  # 列偏移
    col_zero_loop:
            sw x13, 0(x12)             # 写入0到结果矩阵
            addi x12, x12, 4           # 移动到下一个元素
            addi x15, x15, 1
            blt x15, x8, col_zero_loop # 处理每列
            addi x14, x14, 1
```

```
        blt x14, x7, zero_loop     # 处理每行

    la x12, result             # 结果矩阵基址

    # ----------------------- 矩阵乘法 -----------------------
    #  (矩阵B已经转置)
    # 只使用vmul.vv 与 vadd.vv 实现矩阵乘法。都是逐元素的操作，使用vector reg作为rd。
    # 使用vsetvl设置向量长度
    # 大致思路:
    # 1. 初始化行索引 i = 0
    # 2. 计算矩阵A当前行的起始地址，并读取A的第j行。 (使用vlw.v)
    # 3. 初始化列索引 j = 0
    # 4. 计算矩阵B转置后当前行 (原矩阵B的列) 的起始地址，并读取B的第j行。 (使用vlw.v)
    # 5. 使用vmul.vv 计算A的第i行与B的第j列的乘积，暂存起来。
    # 6. 使用add累加乘积向量，得到结果矩阵的[i][j]元素。
    # 7. 重复4-6，直到B的第j行处理完毕。
    # 8. 重复2-7，直到A的每一行处理完毕。

    # 初始化行索引 i = 0
    li x14, 0                # 行索引 i

row_loop:
    # 检查是否处理完所有行
    bge x14, x7, end_matrix_mult

    # 计算矩阵 A 当前行的起始地址
    mul t4, x14, x6          # t4 = i * shared_dim

    # 初始化列索引 j = 0
    li x15, 0                # 列索引 j

    slli t4, t4, 2           # t4 = t4 * 4 (字节偏移)

    add x3, x9, t4           # x3 = matrix1 + i * shared_dim * 4, 矩阵A的第i行的起
始地址

col_loop:
    # 检查是否处理完所有列
    bge x15, x8, next_row

    # 计算矩阵 B 转置后当前行 (原矩阵 B 的列) 的起始地址
    mul t4, x15, x6          # t4 = j * shared_dim
    slli t4, t4, 2           # t4 = t4 * 4 (字节偏移)
    add t3, x10, t4          # t3 = matrix2_T + j * shared_dim * 4, 矩阵B转置后的第
j行的起始地址

    # 初始化共享维度索引 k = 0 和累加器
    li x16, 0                # 共享维度索引 k
    li x21, 0
k_loop:
    # 检查是否处理完所有共享维度
    bge x16, x6, end_k_loop

    # 计算剩余元素数量
    sub x26, x6, x16          # t4 = shared_dim - k

    # 加载矩阵 A 的第 i 行的部分向量，长度为t5
    mul t4, x16, x4           # t6 = k * 4，矩阵A的第i行第k个元素的地址偏移量
    add t6, x3, t4           # A[i][k]，矩阵A的第i行第k个元素的地址
    vsetvl t5, x26, x4         # 设置处理向量长度为t5，元素宽度 32 位 (4字节)
    vlw.v v0, 0(t6)             # v0 = A[i][k..k+VL-1]

    # 加载矩阵 B 转置后的第 j 行的部分向量
    mul t4, x16, x4           # t7 = k * 4，矩阵B转置后的第j行第k个元素的地址偏移量
    add t6, t3, t4           # B_T[j][k]，矩阵B转置后的第j行第k个元素的地址
    vlw.v v1, 0(t6)             # v1 = B_T[j][k..k+VL-1]

    # 增加共享维度索引 k += VL
    add x16, x16, t5

    # 向量乘法
    vmul.vv v2, v0, v1        # v2 = A[i][k..] * B_T[j][k..]

    # 向量加法，累加到向量累加器。x18: 累加器地址，x19: 累加器索引，x20: 读取向量值的地
址，x21: 累加器值
    # vsw.v v2, 4(x18)           # 将v2写入x18 + 4
    # li x19, 0                  # 累加器索引
    # mv x20, x18                # 读取向量值的地址
    # lw x21, 0(x20)           # 读取累加器值
    # cumsum:
    #     lw x22, 4(x20)         # 读取累加器值
    #     add x21, x21, x22
    #     addi x20, x20, 4
    #     addi x19, x19, 1
    #     blt x19, t5, cumsum
    #     sw x21, 0(x18)         # 写回累加器值

    # 使用vredsum计算累加和: x21 = sum(v2) + x21

    vredsum x21, v2, x21

    j k_loop

end_k_loop:
    # 将x21写入矩阵C的[i][j]位置

    # 将结果存储到矩阵 C 的 C[i][j] 位置
    # 计算 C[i][j] 的地址
    # mul t6, x14, x8             # t6 = i * cols_B  当前处理行数 * 矩阵B的列数 (结果
矩阵的列数)
    # add t3, t6, x15             # t6 = i * cols_B + j  当前处理行数 * 矩阵B的列数
 (结果矩阵的列数)  + 当前处理列数
    # slli t6, t3, 2              # t6 = (i * cols_B + j) * 4
    # add t5, x12, t6             # t5 = matrix3 + (i * cols_B + j) * 4
    # sw x21, 0(t5)               # C[i][j] = sum
    # # 增加列索引 j += 1
    # addi x15, x15, 1
    # 修改计算逻辑为：(i * cols_B + j) * 4 = i * cols_B * 4 + j * 4
    mul t6, x14, x8           # t6 = i * cols_B
    slli t4, x15, 2           # t7 = j * 4

    slli t6, t6, 2            # t6 = i * cols_B * 4

    addi x15, x15, 1          # 提前更新列索引 j += 1，减少依赖延迟
    add t6, t6, t4            # t6 = i * cols_B * 4 + j * 4

    add t5, x12, t6           # t5 = matrix3 + (i * cols_B + j) * 4
    sw x21, 0(t5)             # C[i][j] = sum

    j col_loop

next_row:
    # 增加行索引 i += 1
    addi x14, x14, 1
    j row_loop

end_matrix_mult:
        ebreak


# ----------------------- 矩阵乘法 -----------------------

        # 打印结果矩阵
        li x14, 0                   # 行索引
print_loop_row:
        li x15, 0                   # 列索引
print_loop_col:
        mul x16, x14, x8            # 行偏移 = 当前行 * 列数
        add x16, x16, x15           # 列偏移 = 行偏移 + 当前列
        slli x16, x16, 2            # 偏移量 = 偏移 * 4 (字节)
        add x16, x16, x12           # 计算结果地址

        lw a0, 0(x16)               # 加载结果矩阵的值到 a0
        li a7, 244                    # syscall 编号：1 (print integer)
        ecall                      # 调用系统打印值

        # 打印空格
        li a0, 32                   # ASCII 32 = 空格
        li a7, 245                   # syscall 编号：11 (print char)
        ecall

        addi x15, x15, 1            # 更新列索引
        blt x15, x8, print_loop_col

        # 打印换行
        li a0, 10                   # ASCII 10 = 换行
        li a7, 245                   # syscall 编号：11 (print char)
        ecall

        addi x14, x14, 1            # 更新行索引
        blt x14, x7, print_loop_row
```

```
exit:
        nop
        nop

.data
    matrix1:
        .word 3, 2, 6, 8, 5, 9, 3, 1, 8, 0, 4, 7, 4, 6, 7, 2, 7, 4, 9, 1, 4, 2, 7,
4, 9, 3, 7, 4, 7, 2, 2, 9, 0, 6, 6, 1, 1, 3, 1, 3, 6, 9, 2, 9, 3, 0
        .word 8, 4, 8, 7, 0, 4, 4, 9, 4, 4, 4, 2, 9, 9, 6, 0, 5, 7, 7, 6, 5, 5, 7,
3, 9, 4, 6, 8, 7, 7, 2, 2, 2, 2, 4, 1, 6, 5, 3, 8, 4, 7, 2, 6, 4, 2
        .word 7, 2, 8, 2, 1, 4, 4, 1, 8, 6, 8, 2, 9, 6, 4, 8, 4, 0, 3, 2, 8, 2, 1,
0, 0, 2, 6, 5, 2, 2, 5, 3, 2, 8, 7, 0, 6, 1, 3, 2, 5, 0, 2, 0, 2, 3
        .word 5, 5, 5, 2, 8, 0, 9, 0, 0, 1, 1, 1, 9, 3, 2, 1, 0, 6, 5, 9, 7, 3, 9,
8, 1, 9, 2, 9, 5, 4, 4, 1, 1, 0, 1, 8, 9, 0, 1, 1, 6, 4, 4, 9, 6, 7
        .word 8, 5, 1, 4, 4, 2, 1, 1, 4, 4, 1, 5, 6, 1, 6, 5, 6, 8, 4, 7, 1, 1, 5,
2, 1, 5, 9, 2, 3, 1, 2, 9, 0, 6, 4, 9, 8, 2, 3, 0, 4, 7, 7, 1, 4, 1
        .word 8, 0, 7, 0, 9, 7, 5, 5, 4, 6, 7, 5, 7, 0, 4, 2, 8, 5, 4, 4, 8, 9, 8,
2, 5, 5, 7, 1, 1, 7, 9, 9, 2, 4, 9, 0, 8, 1, 5, 7, 6, 0, 0, 8, 5, 8
        .word 2, 6, 2, 5, 2, 8, 3, 9, 0, 9, 2, 2, 9, 5, 9, 0, 4, 6, 2, 9, 8, 3, 4,
0, 9, 5, 9, 4, 4, 1, 6, 6, 5, 0, 7, 7, 5, 4, 8, 9, 5, 5, 8, 3, 3, 4
        .word 0, 2, 6, 6, 2, 8, 4, 1, 1, 6, 1, 9, 1, 8, 1, 2, 2, 9, 7, 4, 3, 0, 6,
4, 6, 4, 5, 4, 6, 9, 7, 0, 8, 7, 6, 5, 9, 8, 9, 2, 7, 6, 4, 0, 6, 2
        .word 6, 3, 7, 6, 3, 0, 4, 2, 6, 5, 2, 6, 9, 9, 0, 5, 6, 4, 1, 0, 5, 4, 0,
4, 6, 3, 4, 1, 1, 2, 7, 9, 8, 1, 3, 7, 1, 2, 2, 9, 6, 8, 7, 3, 1, 0
        .word 8, 5, 6, 3, 5, 9, 8, 2, 6, 0, 3, 8, 9, 1, 7, 6, 6, 4, 2, 7, 2, 5, 2,
5, 5, 6, 1, 0, 7, 9, 4, 1, 7, 7, 1, 2, 4, 2, 5, 8, 5, 0, 5, 3, 0, 4
        .word 0, 1, 5, 4, 9, 3, 5, 3, 2, 2, 6, 5, 4, 4, 0, 4, 7, 2, 9, 2, 2, 0, 9,
0, 5, 6, 7, 7, 3, 2, 7, 5, 6, 5, 0, 0, 1, 5, 3, 7, 9, 2, 4, 3, 5, 1
        .word 4, 4, 2, 8, 1, 2, 5, 2, 6, 3, 6, 2, 6, 8, 6, 3, 6, 7, 0, 8, 9, 9, 6,
3, 0, 0, 9, 6, 8, 2, 5, 9, 7, 7, 3, 7, 2, 8, 5, 6, 3, 0, 1, 5, 3, 0
        .word 1, 4, 1, 7, 8, 3, 8, 5, 0, 6, 6, 4, 4, 4, 6, 2, 0, 8, 5, 2, 6, 2, 0,
3, 0, 7, 6, 6, 5, 3, 2, 3, 4, 6, 3, 0, 8, 4, 3, 0, 3, 6, 9, 0, 2, 7
        .word 0, 1, 6, 4, 8, 3, 2, 2, 2, 0, 5, 6, 7, 5, 7, 7, 4, 5, 0, 8, 3, 1, 6,
4, 4, 1, 7, 0, 2, 1, 8, 4, 8, 1, 0, 5, 9, 1, 8, 3, 9, 0, 9, 6, 1, 1
        .word 5, 3, 1, 7, 2, 9, 3, 5, 0, 0, 9, 3, 0, 1, 2, 9, 8, 9, 8, 2, 6, 7, 4,
2, 7, 3, 2, 1, 9, 4, 7, 9, 1, 2, 3, 7, 8, 1, 6, 0, 0, 2, 0, 7, 3, 9
        .word 3, 2, 3, 2, 3, 0, 4, 0, 7, 6, 9, 5, 1, 3, 8, 1, 6, 0, 5, 8, 7, 6, 3,
9, 3, 8, 7, 0, 8, 5, 2, 6, 2, 8, 3, 0, 0, 9, 1, 4, 4, 6, 8, 9, 8, 1
        .word 3, 3, 3, 5, 1, 0, 1, 1, 2, 0, 0, 8, 8, 1, 6, 5, 8, 9, 4, 7, 9, 5, 9,
0, 7, 4, 8, 0, 7, 4, 0, 3, 2, 9, 2, 9, 9, 8, 0, 5, 7, 6, 9, 6, 8, 6
        .word 8, 5, 8, 7, 7, 3, 6, 0, 6, 8, 8, 0, 6, 3, 6, 8, 1, 3, 5, 5, 6, 9, 5,
7, 5, 5, 9, 2, 9, 5, 1, 2, 5, 4, 3, 9, 6, 5, 5, 1, 5, 3, 1, 8, 0, 8
        .word 9, 6, 5, 6, 2, 3, 3, 6, 6, 8, 3, 0, 0, 6, 3, 4, 2, 4, 5, 6, 3, 9, 8,
9, 9, 3, 8, 8, 5, 8, 0, 4, 7, 0, 1, 2, 4, 2, 5, 6, 8, 1, 7, 7, 3, 2
        .word 6, 6, 1, 4, 2, 1, 6, 2, 5, 6, 2, 9, 5, 2, 8, 9, 8, 4, 5, 7, 8, 6, 6,
7, 2, 9, 7, 7, 3, 0, 1, 0, 1, 2, 0, 2, 1, 5, 1, 3, 0, 0, 2, 7, 2, 3

        # 测试矩阵
        # .word 1, 2, 3, 4
        # .word 5, 6, 7, 8
        # .word 9, 10, 11, 12

    matrix2:
        .word 6, 1, 4, 1, 7, 1, 2, 0, 8, 4, 3, 5, 5, 5, 1, 0, 9, 5, 5, 7, 5, 9, 8,
8, 8, 4, 1, 7, 8, 9, 6, 6, 0, 0, 1, 3, 1, 1, 6, 4, 9, 8, 6, 0, 3, 8
        .word 4, 9, 2, 2, 5, 0, 8, 5, 8, 6, 5, 4, 6, 2, 3, 9, 6, 2, 0, 2, 7, 9, 2,
3, 1, 4, 6, 3, 9, 4, 8, 1, 9, 3, 0, 6, 6, 2, 9, 0, 2, 0, 1, 0, 6, 4
        .word 4, 3, 1, 2, 7, 9, 8, 2, 7, 2, 1, 5, 3, 4, 7, 6, 1, 6, 5, 7, 4, 7, 3,
3, 1, 2, 0, 3, 3, 6, 1, 1, 6, 8, 4, 1, 8, 6, 4, 7, 1, 2, 7, 7, 4, 7
        .word 4, 4, 5, 4, 4, 2, 7, 0, 2, 5, 5, 1, 7, 6, 9, 8, 7, 8, 9, 6, 6, 3, 2,
5, 3, 6, 8, 4, 0, 4, 3, 5, 3, 3, 7, 0, 8, 3, 2, 2, 5, 6, 2, 1, 4, 4
        .word 5, 9, 7, 2, 9, 0, 0, 4, 2, 7, 8, 0, 7, 4, 2, 9, 0, 9, 6, 5, 3, 8, 7,
8, 9, 0, 1, 1, 9, 9, 2, 3, 1, 8, 1, 9, 4, 2, 6, 3, 4, 8, 4, 0, 0, 4
        .word 1, 5, 3, 4, 4, 3, 4, 3, 1, 1, 6, 3, 7, 9, 9, 2, 4, 4, 5, 7, 9, 8, 8,
8, 5, 9, 5, 9, 7, 3, 0, 0, 7, 8, 0, 6, 2, 0, 6, 3, 5, 2, 8, 1, 2, 7
        .word 1, 1, 3, 5, 1, 7, 1, 6, 8, 6, 3, 9, 5, 5, 1, 4, 0, 5, 9, 1, 7, 0, 5,
3, 7, 8, 9, 2, 1, 6, 5, 4, 5, 7, 4, 1, 6, 2, 1, 4, 8, 8, 9, 9, 8, 5
        .word 3, 6, 2, 8, 4, 3, 5, 1, 9, 4, 2, 5, 3, 9, 2, 2, 2, 2, 5, 0, 9, 5, 9,
1, 0, 3, 5, 7, 7, 3, 2, 1, 2, 2, 4, 1, 5, 0, 0, 7, 5, 3, 6, 8, 2, 6
        .word 0, 8, 3, 7, 5, 8, 4, 4, 0, 0, 4, 5, 1, 1, 7, 4, 6, 6, 1, 2, 8, 9, 8,
3, 4, 1, 1, 5, 2, 5, 9, 3, 9, 2, 1, 2, 6, 7, 6, 3, 8, 0, 9, 2, 4, 7
        .word 4, 2, 5, 6, 5, 1, 5, 3, 4, 0, 2, 3, 9, 1, 7, 1, 5, 4, 2, 0, 1, 7, 8,
7, 1, 6, 6, 9, 7, 4, 0, 0, 8, 9, 1, 2, 1, 3, 4, 7, 4, 1, 5, 9, 9, 7
        .word 9, 7, 0, 0, 7, 1, 8, 5, 5, 1, 7, 3, 8, 9, 2, 5, 0, 4, 9, 6, 5, 7, 8,
2, 9, 4, 1, 7, 6, 1, 8, 6, 1, 9, 8, 4, 5, 6, 6, 8, 1, 1, 7, 8, 6, 2
        .word 4, 3, 0, 1, 9, 9, 9, 7, 6, 9, 4, 1, 6, 8, 0, 6, 2, 3, 7, 8, 0, 3, 4,
0, 1, 8, 5, 5, 2, 2, 4, 7, 8, 1, 1, 4, 8, 9, 5, 9, 9, 2, 9, 8, 6, 2
        .word 7, 1, 3, 7, 2, 1, 7, 8, 6, 8, 1, 0, 3, 3, 4, 4, 2, 8, 7, 0, 3, 3, 5,
5, 2, 3, 3, 0, 8, 2, 6, 0, 0, 0, 8, 4, 9, 6, 7, 3, 2, 8, 8, 8, 8, 6
        .word 2, 9, 5, 6, 0, 3, 6, 9, 6, 9, 0, 0, 2, 4, 3, 3, 4, 1, 7, 7, 3, 1, 1,
6, 8, 5, 1, 6, 6, 2, 0, 7, 9, 5, 8, 4, 8, 8, 4, 4, 2, 3, 8, 4, 1, 2
        .word 2, 4, 7, 3, 8, 7, 1, 3, 2, 6, 2, 2, 9, 0, 7, 1, 6, 3, 5, 1, 8, 5, 1,
0, 3, 4, 2, 6, 4, 1, 2, 6, 7, 5, 9, 5, 7, 2, 1, 3, 8, 0, 2, 2, 5, 9
        .word 2, 9, 1, 7, 7, 7, 6, 3, 8, 8, 0, 5, 5, 2, 2, 5, 3, 8, 9, 0, 8, 4, 8,
9, 2, 4, 1, 6, 3, 6, 9, 6, 7, 7, 8, 8, 6, 5, 3, 3, 5, 2, 7, 8, 4, 3
        .word 6, 1, 2, 0, 3, 1, 0, 9, 8, 6, 8, 5, 5, 0, 5, 8, 6, 4, 4, 4, 3, 8, 2,
5, 5, 9, 3, 5, 2, 7, 1, 2, 7, 7, 7, 8, 5, 5, 5, 8, 1, 7, 9, 8, 0, 0
        .word 1, 1, 8, 6, 7, 5, 8, 0, 9, 3, 7, 0, 2, 5, 7, 7, 4, 5, 8, 2, 2, 9, 4,
6, 8, 1, 0, 0, 0, 6, 4, 0, 0, 1, 8, 3, 5, 2, 6, 2, 0, 0, 2, 6, 8, 4
        .word 1, 2, 6, 9, 5, 0, 4, 2, 7, 6, 7, 3, 5, 5, 4, 8, 9, 5, 5, 4, 1, 1, 8,
3, 1, 7, 2, 8, 9, 5, 2, 9, 6, 6, 9, 7, 4, 8, 9, 7, 4, 5, 3, 5, 4, 5
        .word 9, 7, 6, 4, 0, 2, 3, 2, 7, 2, 3, 4, 5, 7, 2, 5, 7, 8, 9, 4, 6, 4, 5,
9, 8, 0, 5, 0, 4, 8, 3, 9, 7, 0, 4, 9, 0, 2, 5, 2, 2, 9, 6, 5, 8, 2
        .word 1, 2, 3, 0, 6, 1, 1, 3, 0, 5, 1, 9, 1, 9, 5, 7, 6, 0, 4, 3, 9, 6, 5,
9, 8, 8, 0, 2, 7, 5, 3, 0, 1, 2, 9, 7, 6, 4, 5, 5, 7, 9, 7, 0, 4, 3
        .word 5, 5, 7, 4, 7, 5, 7, 0, 1, 1, 3, 1, 0, 9, 8, 0, 4, 8, 6, 0, 7, 6, 5,
6, 8, 1, 3, 9, 3, 2, 0, 9, 9, 8, 6, 2, 9, 3, 2, 2, 7, 8, 1, 5, 5, 3
        .word 3, 8, 8, 5, 2, 0, 2, 0, 1, 4, 0, 5, 3, 2, 1, 0, 3, 2, 9, 0, 1, 5, 7,
9, 9, 6, 1, 0, 6, 4, 5, 9, 5, 6, 5, 9, 4, 9, 0, 5, 6, 4, 1, 2, 6, 5
        .word 0, 4, 1, 9, 5, 4, 3, 3, 4, 4, 3, 2, 6, 7, 9, 8, 4, 9, 7, 9, 6, 1, 5,
7, 7, 8, 6, 8, 9, 2, 1, 7, 9, 5, 5, 8, 7, 6, 7, 3, 6, 9, 1, 5, 3, 4
        .word 9, 6, 0, 4, 1, 1, 0, 2, 1, 2, 9, 8, 7, 7, 0, 0, 8, 5, 2, 8, 3, 0, 0,
7, 2, 4, 5, 4, 9, 3, 1, 5, 6, 9, 9, 2, 6, 9, 2, 1, 2, 3, 0, 6, 2, 1
        .word 1, 6, 1, 7, 1, 3, 4, 5, 8, 3, 5, 6, 2, 0, 8, 3, 3, 0, 2, 0, 4, 8, 0,
4, 6, 8, 5, 6, 5, 5, 4, 8, 4, 2, 8, 5, 6, 4, 2, 0, 1, 9, 3, 1, 4, 6
        .word 9, 3, 9, 3, 4, 7, 7, 1, 1, 2, 0, 1, 1, 3, 3, 9, 5, 1, 3, 7, 9, 6, 4,
8, 8, 2, 4, 9, 8, 6, 5, 1, 6, 1, 9, 9, 6, 7, 3, 1, 1, 0, 7, 9, 2, 6
```

```
        .word 8, 0, 3, 0, 3, 7, 5, 9, 2, 4, 9, 4, 5, 5, 6, 8, 5, 7, 9, 4, 5, 1, 2, 0, 9, 1, 1, 5, 9, 9, 0, 4, 2, 9, 6, 4, 1, 1, 5, 6, 3, 7, 2, 6, 9, 8
        .word 9, 2, 9, 3, 9, 2, 9, 0, 9, 1, 6, 6, 8, 7, 0, 9, 6, 1, 4, 4, 7, 1, 8, 6, 0, 9, 9, 5, 0, 1, 3, 6, 5, 5, 1, 1, 5, 9, 1, 4, 4, 5, 6, 2, 6, 6
        .word 6, 5, 4, 1, 8, 0, 1, 8, 0, 4, 6, 7, 4, 9, 9, 5, 1, 0, 8, 2, 8, 9, 8, 4, 6, 1, 3, 1, 7, 1, 1, 6, 7, 4, 6, 3, 6, 9, 8, 3, 6, 4, 1, 1, 3, 2
        .word 3, 2, 8, 0, 2, 2, 5, 2, 8, 4, 6, 3, 8, 9, 5, 1, 2, 6, 5, 5, 5, 2, 7, 8, 3, 4, 6, 9, 0, 7, 0, 3, 9, 3, 7, 7, 7, 3, 9, 5, 7, 1, 5, 6, 3, 0
        .word 8, 2, 6, 5, 3, 9, 7, 2, 6, 8, 3, 8, 5, 6, 2, 1, 8, 5, 0, 0, 2, 5, 4, 9, 7, 6, 2, 2, 1, 6, 6, 9, 2, 2, 1, 4, 1, 7, 6, 2, 9, 5, 7, 9, 6, 6
        .word 8, 2, 7, 3, 1, 8, 1, 0, 1, 6, 7, 8, 1, 2, 9, 4, 0, 2, 5, 4, 0, 4, 1, 8, 0, 3, 6, 8, 7, 1, 7, 3, 0, 6, 3, 5, 3, 1, 1, 4, 2, 7, 9, 1, 2, 2
        .word 7, 3, 3, 9, 6, 6, 4, 1, 7, 3, 7, 9, 6, 3, 4, 9, 9, 3, 2, 4, 3, 6, 9, 6, 7, 5, 2, 3, 1, 5, 1, 1, 1, 5, 7, 1, 4, 7, 3, 3, 8, 1, 8, 3, 0, 8
        .word 5, 6, 5, 4, 5, 0, 5, 3, 6, 8, 2, 4, 9, 4, 4, 5, 4, 0, 5, 2, 1, 8, 5, 4, 9, 8, 1, 0, 4, 9, 6, 5, 9, 4, 8, 8, 8, 5, 7, 8, 8, 5, 1, 8, 0, 1
        .word 7, 8, 7, 3, 1, 7, 7, 8, 6, 2, 4, 7, 1, 4, 8, 7, 7, 7, 8, 5, 6, 3, 7, 9, 6, 5, 9, 9, 9, 7, 1, 7, 3, 5, 6, 6, 3, 2, 2, 9, 9, 2, 3, 4, 0, 2
        .word 3, 1, 6, 8, 8, 5, 0, 7, 7, 9, 1, 3, 6, 9, 4, 0, 6, 3, 8, 4, 2, 3, 5, 3, 5, 1, 2, 0, 3, 0, 4, 8, 7, 8, 1, 0, 9, 7, 6, 2, 4, 3, 9, 7, 7, 9
        .word 9, 1, 0, 3, 6, 3, 2, 3, 1, 3, 3, 7, 6, 1, 4, 0, 6, 5, 4, 7, 7, 9, 5, 9, 8, 9, 8, 9, 9, 9, 5, 2, 2, 3, 9, 0, 8, 2, 6, 7, 1, 7, 0, 4, 0, 1
        .word 6, 2, 4, 2, 9, 2, 0, 5, 0, 4, 1, 7, 7, 5, 5, 1, 8, 2, 8, 7, 2, 5, 7, 4, 8, 8, 3, 4, 9, 3, 2, 1, 9, 7, 6, 6, 7, 1, 2, 7, 5, 9, 7, 5, 5, 9
        .word 3, 2, 2, 0, 8, 3, 1, 8, 0, 2, 5, 1, 4, 2, 3, 0, 2, 6, 9, 0, 9, 0, 3, 9, 9, 8, 9, 1, 8, 9, 0, 3, 2, 2, 1, 0, 0, 4, 4, 6, 5, 3, 1, 7, 5, 6
        .word 2, 9, 0, 8, 4, 6, 7, 4, 1, 2, 9, 1, 2, 4, 7, 1, 8, 5, 9, 3, 8, 0, 3, 7, 8, 4, 7, 0, 6, 0, 8, 9, 7, 9, 4, 3, 8, 9, 0, 4, 0, 1, 1, 6, 8, 9
        .word 4, 1, 9, 1, 6, 1, 4, 4, 7, 4, 6, 0, 0, 9, 3, 0, 5, 9, 6, 2, 3, 2, 5, 3, 1, 6, 0, 9, 5, 5, 0, 9, 7, 4, 7, 5, 3, 3, 7, 8, 0, 1, 1, 9, 5, 5
        .word 3, 0, 5, 1, 3, 1, 5, 1, 2, 5, 5, 4, 1, 2, 4, 1, 7, 8, 1, 2, 7, 8, 6, 8, 2, 9, 3, 1, 3, 6, 3, 7, 2, 5, 0, 0, 8, 8, 4, 6, 3, 1, 0, 7, 6, 2
        .word 3, 0, 9, 4, 4, 3, 3, 1, 1, 5, 8, 6, 4, 1, 8, 8, 8, 7, 8, 5, 9, 7, 7, 2, 5, 7, 4, 4, 5, 3, 5, 0, 2, 9, 9, 4, 5, 4, 8, 9, 4, 7, 6, 2, 1, 2
        .word 9, 4, 5, 0, 2, 1, 2, 0, 7, 1, 4, 1, 3, 8, 5, 4, 5, 0, 5, 7, 1, 4, 8, 4, 2, 0, 2, 0, 7, 0, 6, 6, 0, 8, 7, 7, 9, 3, 7, 5, 0, 4, 8, 9, 4, 6
        .word 1, 8, 4, 7, 4, 0, 1, 2, 1, 6, 1, 2, 8, 3, 1, 9, 5, 8, 2, 2, 7, 6, 5, 9, 8, 5, 3, 4, 9, 5, 8, 1, 0, 2, 8, 1, 7, 9, 1, 2, 0, 2, 1, 5, 8, 3
        .word 4, 9, 9, 9, 8, 9, 0, 1, 1, 0, 0, 7, 3, 2, 1, 0, 6, 5, 3, 1, 7, 3, 0, 1, 6, 9, 8, 8, 8, 5, 5, 2, 5, 5, 8, 8, 8, 5, 3, 1, 1, 6, 0, 0, 0, 7
        .word 6, 6, 6, 0, 1, 9, 1, 4, 9, 7, 7, 2, 7, 9, 8, 8, 6, 7, 3, 2, 9, 2, 2, 0, 8, 2, 3, 3, 7, 5, 2, 7, 6, 5, 4, 2, 9, 1, 2, 8, 6, 8, 8, 4, 6, 4
        .word 6, 6, 4, 2, 0, 1, 8, 2, 4, 2, 7, 9, 4, 0, 0, 9, 7, 7, 3, 9, 9, 0, 4, 2, 2, 5, 5, 2, 2, 0, 8, 8, 5, 2, 3, 7, 5, 5, 9, 6, 8, 8, 8, 4, 8, 1
        .word 2, 5, 9, 3, 6, 9, 0, 2, 0, 4, 6, 3, 6, 6, 7, 5, 0, 6, 0, 9, 0, 7, 9, 2, 0, 1, 3, 2, 1, 7, 2, 3, 1, 7, 8, 9, 0, 9, 1, 1, 0, 9, 2, 2, 0, 0

        # 测试小矩阵
        # .word 1, 2
        # .word 5, 6
        # .word 9, 10
        # .word 1, 5, 9, 3
        # .word 2, 6, 10, 1
        # .word 3, 7, 11, 2
```

```
result:
        .space 4000
temp:
        .space 4000
vector_temp:
        .space 500
```

Code Execution Result `./test/cli/matrix/stdout.txt`

```
root@31b7786b65c2:/home/deploy/workspace/qtrvsim# ./cmake-build-release-
3050_proj3_qt/target/qtrvsim_cli --os-emulation --asm --pipelined
./tests/cli/matrix/program.S
[INFO]  machine.BranchPredictor:        Initialized branch predictor: None
20 50
968 671 828 929 872 911 1079 889 809 881 1034 912 856 861 825 1069 888 863 1080
1051 873 1076 890 1162 843 870 911 1031 996 924 920 1052 818 999 976 1214 993 986
1021 853 1051 886 771 1041 872 836 932 1089 905 806
1069 847 919 1033 1050 1082 1078 952 908 1003 1188 1072 1037 1035 883 1075 1045
892 1135 1119 947 1091 911 1238 939 881 1057 1158 1032 1042 1110 1057 830 1012
1126 1303 1054 1175 1155 955 1057 1008 876 1128 924 1003 979 1210 971 875
687 781 695 841 719 755 788 698 645 663 866 787 669 694 771 825 763 688 862 717
655 746 555 818 700 652 779 816 933 751 837 738 675 821 802 898 764 728 720 525
765 687 595 895 738 680 702 968 799 620
964 799 832 893 916 1022 862 837 815 922 1036 974 879 774 807 1059 793 747 933 889
846 924 805 1081 718 691 1021 828 1002 776 973 889 691 836 931 1010 812 968 1031
788 877 837 786 887 828 882 843 845 924 695
806 731 724 880 858 802 853 646 696 708 883 872 805 772 730 940 861 603 944 922
753 811 740 1043 781 751 776 806 902 765 810 836 761 834 846 987 838 844 920 592
849 683 662 901 838 728 811 926 968 751
1105 963 997 1055 982 960 1091 895 1026 973 1198 1125 978 859 1069 1184 1058 1004
1120 1019 933 1055 902 1071 867 892 1078 1128 1129 1044 1053 1162 793 1114 1187
1220 1090 1193 1137 957 1131 1020 1031 1198 981 969 973 1164 1071 834
965 901 901 1000 964 1086 1065 847 1021 869 1125 1172 993 1076 987 1117 1038 775
1092 1030 982 978 848 1277 839 915 1028 1043 915 1025 1052 1036 827 949 1112 1204
1064 1078 1139 878 1101 882 784 1128 885 926 991 1207 1097 869
946 846 917 934 970 970 1100 795 983 831 991 1032 955 957 822 1163 894 827 1108
1008 1007 1031 945 1209 873 808 991 994 942 958 1052 1025 810 949 1055 1115 1024
998 1064 798 1011 878 819 1094 813 933 1039 1042 1013 853
899 787 689 819 853 802 915 751 760 735 927 938 773 828 736 977 913 673 969 1039
846 858 823 976 730 750 828 798 977 820 903 1011 689 852 1046 985 884 809 918 634
811 787 678 900 761 760 783 1033 993 669
952 923 1007 872 953 991 892 758 937 941 1015 987 776 855 830 1027 989 824 989 932
844 856 789 1029 816 768 992 997 992 857 934 1010 792 1055 1081 1167 886 1001 1040
764 922 789 793 1037 836 802 889 1058 951 769
852 711 701 800 746 850 926 741 827 829 954 1012 678 730 737 935 772 667 991 782
742 841 767 960 662 623 747 837 1000 820 842 860 632 858 874 992 884 796 931 760
904 789 703 934 663 707 751 851 899 602
917 984 887 921 924 1050 857 880 951 953 1110 1010 834 908 830 1088 945 773 1157
1022 786 999 818 1208 959 830 999 918 1012 992 1030 953 759 922 999 1173 945 983
```

```
917 716 1086 947 879 1033 918 908 869 1046 1032 866
752 774 831 896 822 934 896 748 836 778 879 923 848 828 787 967 805 657 910 729
774 893 648 1038 724 795 777 845 895 813 800 798 738 824 777 923 889 847 903 733
949 731 690 910 643 758 855 903 820 652
812 810 787 827 835 904 869 682 931 770 887 978 737 731 741 927 827 724 922 839
795 813 652 1036 676 622 866 735 916 853 949 874 683 890 928 947 922 783 916 637
839 699 689 931 810 719 722 937 985 694
881 868 822 860 958 854 867 722 931 724 1002 853 906 789 802 1030 867 872 1011 987
797 905 802 1084 779 842 1002 1068 799 870 735 927 712 913 900 1073 884 934 919
793 1121 837 794 992 859 901 925 997 945 685
961 805 872 902 867 977 1006 799 796 925 1030 966 869 892 730 997 1021 804 1048
984 926 897 863 1067 911 862 915 950 1038 925 912 1019 838 947 960 1090 952 981
1002 876 1060 849 930 1039 903 860 723 1006 980 754
995 857 951 975 1016 1071 1112 879 988 1004 1081 973 963 891 912 1101 1027 762
1101 1072 1008 987 968 1227 907 848 1021 1045 1057 974 955 988 766 1066 1035 1167
1031 1010 1200 846 1090 795 900 1167 966 996 969 1141 1130 844
1034 1075 980 1046 1209 1125 988 936 941 1015 1123 1075 1026 989 956 1176 1078
1002 1190 1094 954 1052 959 1284 905 939 1214 1056 1133 1079 1085 1069 884 1104
1211 1244 1025 1085 1112 919 1102 956 899 1144 996 1015 1026 1117 992 926
1066 892 891 907 1047 1057 1009 918 949 964 1078 1100 939 1034 779 1106 1033 866
1044 1103 900 1026 880 1142 788 824 1099 971 972 983 1094 1069 804 1008 1102 1278
994 1060 1039 896 930 944 841 972 868 895 816 1086 922 735
809 844 802 877 749 945 816 786 767 853 923 852 752 774 681 955 898 707 911 838
797 745 675 1009 756 740 916 795 983 803 818 856 717 941 839 1077 741 933 872 701
880 706 779 945 713 853 741 839 892 640
Machine stopped on BREAK exception.
```

Verification Script `./test/cli/matrix/matrix_check.py` **Result**

```
Matrix multiplication result match: True
```