Instituto Tecnológico y de Estudios Superiores de Monterrey
(Campus Guadalajara)

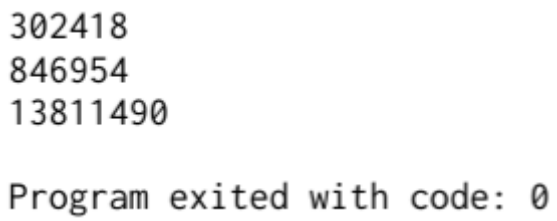
Diseño de sistemas en chip (Gpo 501)

Matriz_mult_RISCV

Alumna:
Elizabeth Jáuregui Zárate (A01253381)

Fecha de entrega:
22/04/2025

Captura de pantalla del resultado (memoria con matriz C):



Código fuente en archivo Simulador Ripes:

```
.data
   A: .word 1, 2, 3
      .word 4, 5, 6
      .word 7, 8, 9

   B: .word 9, 8, 7
      .word 6, 5, 4
      .word 3, 2, 1

   C: .word 0, 0, 0
      .word 0, 0, 0
      .word 0, 0, 0

   newline: .asciz "\n"   # Salto de línea

.text
main:
   la a0, A
   la a1, B
   la a2, C
   li t0, 0            # i = 0

loop_i:
   li t1, 0            # j = 0
loop_j:
   li t2, 0            # k = 0
   li t3, 0            # acumulador = 0

loop_k:
   # Cálculo de A[i][k]
   li t4, 3
   mul t5, t0, t4
   add t5, t5, t2
```

```
    slli t5, t5, 2
    add t5, a0, t5
    lw t6, 0(t5)

    # Cálculo de B[k][j]
    mul t5, t2, t4
    add t5, t5, t1
    slli t5, t5, 2
    add t5, a1, t5
    lw t5, 0(t5)

    mul t5, t6, t5
    add t3, t3, t5

    addi t2, t2, 1        # k++
    blt t2, t4, loop_k

    # Guardar C[i][j]
    mul t5, t0, t4
    add t5, t5, t1
    slli t5, t5, 2
    add t5, a2, t5
    sw t3, 0(t5)

    addi t1, t1, 1        # j++
    blt t1, t4, loop_j

    addi t0, t0, 1        # i++
    blt t0, t4, loop_i

    la a2, C
    li t0, 0              # i = 0

print_loop_i:
    li t1, 0              # j = 0

print_loop_j:
    # Calcular dirección de C[i][j]
    li t4, 3
    mul t5, t0, t4
    add t5, t5, t1
    slli t5, t5, 2
    add t5, a2, t5
    lw a0, 0(t5)          # Cargar valor de C[i][j] en a0

    # Imprimir entero
```

```
    li a7, 1
    ecall

    addi t1, t1, 1        # j++
    blt t1, t4, print_loop_j

    # Imprimir salto de línea
    la a0, newline
    li a7, 4
    ecall

    addi t0, t0, 1        # i++
    blt t0, t4, print_loop_i

    # Final
    li a7, 10
    ecall
```

Explicación de código:

El código presentado realiza una multiplicación de dos matrices de 3x3 en lenguaje ensamblador para la arquitectura RISC-V 32I:

Se reservan los espacios en la memoria para números de 32 bits, para 3 matrices (A,B y C). Se cargan las direcciones de los primeros elementos de las matrices en registros (a0, a1, a2). Se asignan valor constantes (valor = 0) a registros temporales (t0=i, t1=j, t2=k, t3=suma de las multiplicaciones). Se obtiene el offset de la fila (t5 = i * 3), se suma k para obtener el offset total en A, se convierte en bytes y se asigna a t5 (dirección de A[i][k]). Se carga el valor de A[i][k] desde memoria a t6. Se realiza el mismo proceso para B[k][j]. A continuación se multiplican los valores cargados en t5 y t6 y se suma el resultado al acumulador t3. Se incrementa k y se verifica que sea menor que 3. Se guarda t3 en C[i][j], y se avanza a la siguiente columna (j) y fila (i). Por último se imprime la matriz usando syscall y se termina el programa.