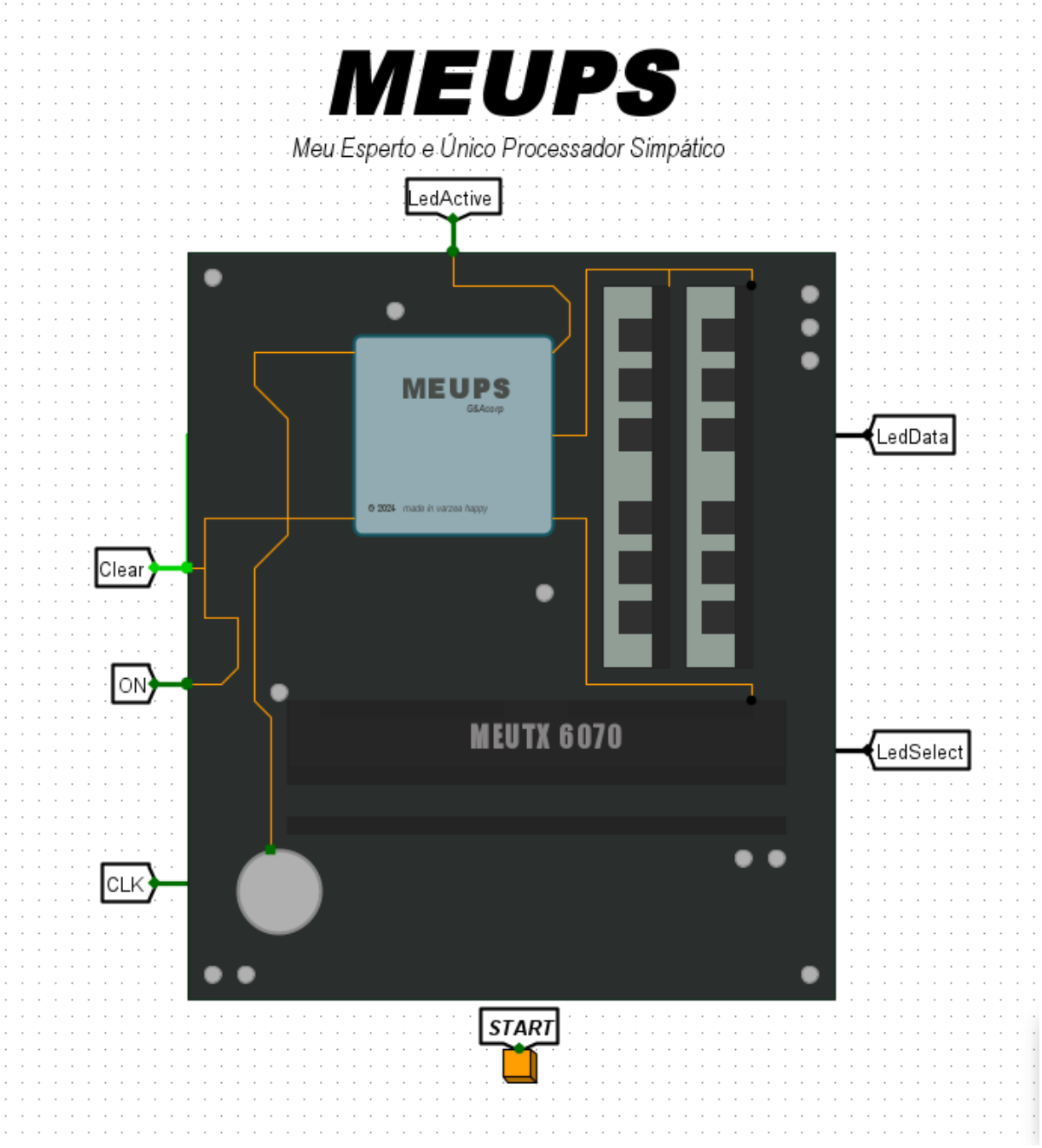
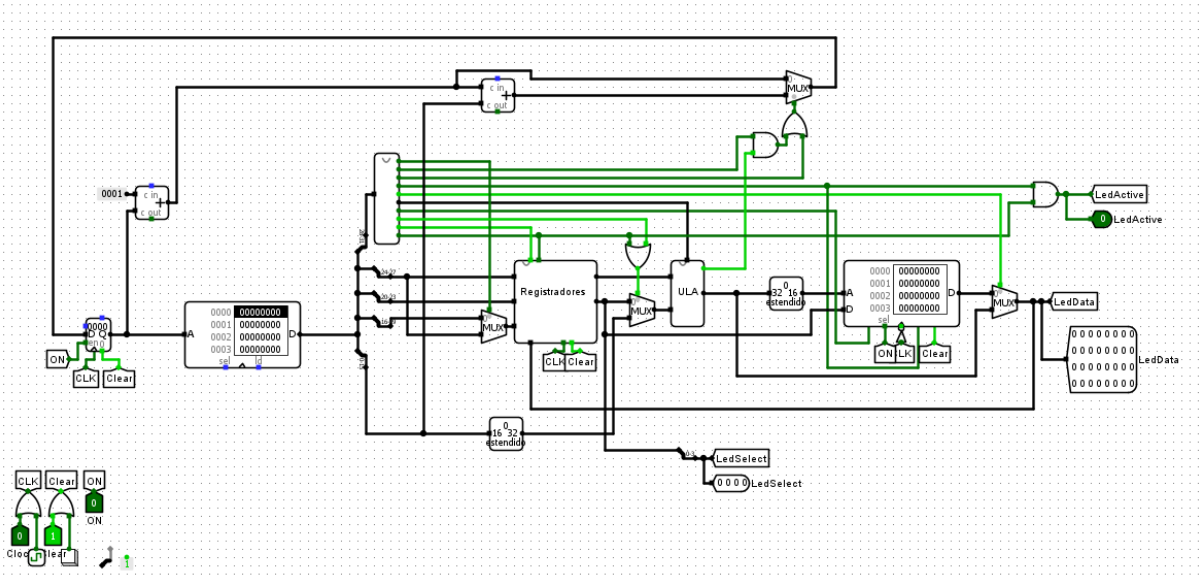
RELATÓRIO MEUPS



MEUPS é o projeto de conclusão da cadeira de Arquitetura e Organização de Computadores do semestre 2023.2 da UFCA (Universidade Federal do Cariri). Foi desenvolvido por Guilherme Viana Batista e Antônio José Monteiro Neto. Consiste em um processador monociclo com 16 registradores capaz de executar códigos simples, inclusive fazer uma multiplicação de matrizes NxN.



O MEUPS possui 14 funções que permitem a execução do código escrito em hexadecimal.

Funções criadas para o MEUPS:

- addn ⇒ Acumulador de um Imediato à um registrador
- lwn ⇒ Load similar ao do MIPS porém usa a soma de dois registradores para gerar o endereço
- print ⇒ Envia um valor da memória de dados para um pequeno display 4×4 presente no HUB
- Todas as outras funções são similares as do MIPS porém com a ordem dos registradores diferente (usa o registrador de destino por último no código).

Função	Tipo	Codigo	Registrador	Codigo
addi	R0 0 RD I	0	v0	0
sw	R0 RD 0 I	1	a0	1
lwi	R0 0 RD I	2	a1	2
move	R0 0 RD 0	3	a2	3
addn	R0 0 0 I	4	t0	4
lwn	R0 R1 RD 0	5	t1	5
li	0 0 RD I	6	t2	6
multi	R0 0 RD I	7	t3	7
mult	R0 R1 RD 0	8	t4	8

j	0 0 0 I	9	t5	9
beq	R0 R1 0 I	a	t6	a
sub	R0 R1 RD 0	b	t7	b
add	R0 R1 RD 0	c	s0	c
div	R0 R1 RD 0	d	s1	d
print	R0 RD 0 I	e	s2	e
			s3	f

CÓDIGO EM ASSEMBLY:

Cria duas matrizes 4×4 e multiplica as duas.

```
.eqv SIZE 4
.globl main

main:
    #===================================== FOR DA MATRIZ 1
    addi $sp, $sp, -64
    move $s0, $sp

    li $a0, 1
    li $t0, SIZE
    li $t1, 0 #i=0
    for_ia:
    bge $t1, $t0, fim_ia
    li $t2, 0 #j=0
    for_ja:
    bge $t2, $t0, fim_ja

    sll $s1, $t1, 4 #i*4*4
    sll $s2, $t2, 2 #j*4
    add $s3, $s1, $s2 #Valor da soma
    add $s3, $s3, $s0

    sw $a0, 0($s3)
    addi $a0, $a0, 1
```

```
    addi $t2, $t2, 1
    j for_ja
    fim_ja:
    addi $t1, $t1, 1
    j for_ia
    fim_ia:

    move $a1, $s0
    #========================================== FOR DA MAT

    addi $sp, $sp, -64
    move $s0, $sp

    li $a0, 1
    li $t0, SIZE
    li $t1, 0 #i=0
    for_ib:
    bge $t1, $t0, fim_ib
    li $t2, 0 #j=0
    for_jb:
    bge $t2, $t0, fim_jb

    sll $s1, $t1, 4 #i*4*4
    sll $s2, $t2, 2 #j*4
    add $s3, $s1, $s2 #Valor da soma
    add $s3, $s3, $s0

    sw $a0, 0($s3)
    addi $a0, $a0, 1
    addi $t2, $t2, 1
    j for_jb
    fim_jb:
    addi $t1, $t1, 1
    j for_ib
    fim_ib:

    move $a2, $s0
    #======================================== FOR DA MULTIP
```

```
addi $sp, $sp, -64
move $s0, $sp

li $a0, 1
li $t0, SIZE
li $t1, 0 #i=0
for_ic:
bge $t1, $t0, fim_ic
li $t2, 0 #j=0
for_jc:
bge $t2, $t0, fim_jc

sll $s1, $t1, 4 #i*4*4
sll $s2, $t2, 2 #j*4

li $t3, 0 #Retorno
li $t4, 0 #X=0
for_x:
bge $t4, $t0, fim_x

sll $s3, $t4, 2 #j*4 p/ X
add $s3, $s3, $s1 #Valor da soma
add $s3, $s3, $a1

lw $t5, 0($s3)

sll $s3, $t4, 4 #i*4*4 p/ X
add $s3, $s3, $s2 #Valor da soma
add $s3, $s3, $a2

lw $t6, 0($s3)

mul $t7, $t5, $t6
add $t3, $t3, $t7


add $s3, $s1, $s2 #Valor da soma
```

```
    add $s3, $s3, $s0

    sw $t3, 0($s3)
    addi $t4, $t4, 1
    j for_x
    fim_x:
    addi $t2, $t2, 1
    j for_jc
    fim_jc:
    addi $t1, $t1, 1
    j for_ic
    fim_ic:

    move $a3, $s0

    #=====================================================PR

    li $a0, 1
    li $t0, SIZE
    li $t1, 0 #i=0
    for_id:
    bge $t1, $t0, fim_id
    li $t2, 0 #j=0
    for_jd:
    bge $t2, $t0, fim_jd

    sll $s1, $t1, 4 #i*4*4
    sll $s2, $t2, 2 #j*4
    add $s3, $s1, $s2 #Valor da soma
    add $s3, $s3, $s0

    lw $t3, 0($s3)

    li $v0, 11
    li $a0, 32                                  #print (' ')
    syscall

    li $v0, 1
```

```
    move $a0, $t3
    syscall

    sw $a0, 0($s3)
    addi $a0, $a0, 1
    addi $t2, $t2, 1
    j for_jd
    fim_jd:
    addi $t1, $t1, 1
    j for_id
    fim_id:
```

CÓDIGO EM MEUPS

O código em MEUPS é capaz de exibir a matriz resultante no HUB do projeto

```
v2.0 raw

600c0000

60010001
60040004
60050000

a540000c
60060000

a6400008

750d0004
760e0001
cdef0000
cffc0000

1f100000
41000001
46000001
9000fff7
```

```
45000001
9000fff3

60020000



600c0010

60010001
60040004
60050000

a540000c
60060000

a6400008

750d0004
760e0001
cdef0000
cfcf0000

1f100000
41000001
46000001
9000fff7
45000001
9000fff3

60030010


600c0020

60010001
60040004
60050000
```

```
a540001a
60060000


a6400016

750d0004
760e0001

60070000
60080000


a840000f

780f0001
cfdf0000
cf2f0000

5f090000
780f0004
cfef0000
cf3f0000

5f0a0000

8a9b0000
c7b70000

cdef0000
cfcf0000

1f700000
08080001
9000fff0
```

```
06060001
9000ffe9


05050001
9000ffe5

60060010
60070000
60080000

a8600004

ec800000
4c000001
48000001

9000fffb
```