Universidade Federal de Pernambuco

RELATÓRIO DE PROJETO

Henrique César Higino Holanda Cordeiro – HCHHC
João Pedro Marinho de Souza – JPMS3
Luiz Gustavo Pinheiro dos Santos Silva – LGPSS

Recife
Agosto de 2024

ÍNDICE

INTRODUÇÃO

Esse relatório visa como objetivo principal a implementação de instruções pendentes em um simulador de dispositivo Risc-V. Nos foi fornecido um repositório com todas as orientações e implementações básicas já funcionais: BEQ, LW, SW, ADD e AND. Seguindo as orientações, ainda devíamos implementar outras 21 instruções: JAL, JALR, BNE, BLT, BGE, LB, LH, LBU, SB, SH, SLTI, ADDI, SLLI, SRLI, SRAI, SUB, SLT, XOR, OR, LUI e HALT. Em nosso projeto, todas as instruções requisitadas foram implementadas, conforme o previsto e seguindo as orientações do repositório.

DESCRIÇÃO DA IMPLEMENTAÇÃO DAS INSTRUÇÕES POR CLASSE DE INSTRUÇÕES

- Instruções Aritméticas e Lógicas (R-type):

Instruções: add, sub, and, or, xor, slt, sll, srl, sra

Formato: opcode (7 bits) | funct3 (3 bits) | rs1 (5 bits) | rs2 (5 bits) | funct7 (7 bits) | rd (5 bits)

Implementação:

Controller: Decodifica opcode (7'b0110011) para identificar instruções R-type. Gera sinais de controle para operações ALU, registros de leitura e escrita.

ALUController: Usa funct7 e funct3 para determinar a operação específica da ALU (ADD, SUB, AND, OR, etc.).

ALU: Executa a operação específica determinada pelo funct7 e funct3. Os operandos são os valores lidos dos registradores rs1 e rs2.

Datapath: Lê os valores dos registradores rs1 e rs2, envia para a ALU, e escreve o resultado no registrador de destino rd.

- Instruções Imediatas (I-type)

Instruções: addi, andi, ori, xori, slli, srli, srai, lw, jalr

Formato: opcode (7 bits) | funct3 (3 bits) | rs1 (5 bits) | imm (12 bits) | rd (5 bits)

Implementação:

Controller: Decodifica opcode (7'b0010011 para aritméticas imediatas, 7'b0000011 para lw, 7'b1100111 para jalr). Gera sinais de controle específicos.

ALUController: Determina a operação da ALU usando funct3 e funct7 (se aplicável).

ALU: Executa a operação aritmética ou lógica usando o valor do registrador rs1 e o valor imediato imm.

Datapath: Lê o valor do registrador rs1, aplica a operação imediata na ALU, e escreve o resultado no registrador de destino rd.

- Instruções de Carga e Armazenamento (Load/Store)

Instruções: lw, sw

Formato: opcode (7 bits) | funct3 (3 bits) | rs1 (5 bits) | imm (12 bits) | rd (5 bits) para lw; rs2 (5 bits) para sw

Implementação:

Controller: Decodifica opcode (7'b0000011 para lw, 7'b0100011 para sw). Gera sinais de controle para leitura/escrita de memória.

ALU: Calcula o endereço de memória usando o valor do registrador rs1 e o valor imediato imm.

Datapath:

Load (lw): A ALU calcula o endereço, e os dados são lidos da memória para o registrador rd.

Store (sw): A ALU calcula o endereço, e os dados do registrador rs2 são escritos na memória.

- Instruções de Ramificação (Branch)

Instruções: beq, bne, blt, bge

Formato: opcode (7 bits) | funct3 (3 bits) | rs1 (5 bits) | rs2 (5 bits) | imm (12 bits)
Implementação:
Controller: Decodifica opcode (7'b1100011). Gera sinais de controle para operações de comparação.
ALU: Compara os valores dos registradores rs1 e rs2 usando a operação especificada (beq, bne, etc.).
BranchUnit: Determina o novo valor do PC com base na comparação realizada pela ALU. Se a condição for verdadeira, calcula o novo endereço PC (PC + imm).

- Instruções de Salto (Jump)

Instruções: jal, jalr
Formato: opcode (7 bits) | imm (20 bits) para jal; rs1 (5 bits) | imm (12 bits)para jalr`
Implementação:
Controller: Decodifica opcode (7'b1101111 para jal, 7'b1100111 para jalr). Gera sinais de controle específicos para operações de salto.
ALU: Calcula o endereço de destino para jalr (somando rs1 e imm).
BranchUnit: Atualiza o valor do PC para o endereço de destino (PC + imm para jal, ou rs1 + imm para jalr).
Datapath: Escreve o valor PC + 4 no registrador de destino rd.

- Instrução de Parada (Halt)

Instrução: halt
Formato: opcode (7 bits) | 7'b1111111
Implementação:
Controller: Decodifica opcode (7'b1111111). Gera um sinal de controle para parar a execução.
BranchUnit: Usa o sinal de halt para parar a execução quando o PC atinge um valor específico

DESCRIÇÃO DOS SINAIS DE CONTROLE PARA CADA CLASSE DE INTRUÇÕES

- Instruções Aritméticas e Lógicas (R-type)

Instruções: add, sub, and, or, xor, slt, sll, srl, sra
Formato: opcode (7 bits) | funct3 (3 bits) | rs1 (5 bits) | rs2 (5 bits) | funct7 (7 bits) | rd (5 bits)
Sinais de Controle:

ALUSrc: 0 (O segundo operando da ALU vem do segundo registrador)
MemtoReg: 0 (O valor a ser escrito no registrador vem da ALU)
RegWrite: 1 (Habilita a escrita no registrador de destino)
MemRead: 0 (Não realiza leitura de memória)
MemWrite: 0 (Não realiza escrita de memória)
ALUOp: 10 (Indica que a operação é do tipo R-type)
Branch: 0 (Não é uma instrução de ramificação)
jal: 0
jalr: 0
halt: 0

- Instruções Imediatas (I-type)

Instruções: addi, andi, ori, xori, slli, srli, srai, lw, jalr
Formato: opcode (7 bits) | funct3 (3 bits) | rs1 (5 bits) | imm (12 bits) | rd (5 bits)
Sinais de Controle:

ALUSrc: 1 (O segundo operando da ALU vem do valor imediato)
MemtoReg: 0
RegWrite: 1
MemRead: 0
MemWrite: 0
ALUOp: 10 (Indica operação I-type)
Branch: 0
jal: 0
jalr: 0
halt: 0

Para lw:

ALUSrc: 1
MemtoReg: 1 (O valor a ser escrito no registrador vem da memória)
RegWrite: 1
MemRead: 1 (Realiza leitura de memória)

MemWrite: 0
ALUOp: 00 (Indica operação de carga)
Branch: 0
jal: 0
jalr: 0
halt: 0

Para jalr:

ALUSrc: 1
MemtoReg: 0
RegWrite: 1
MemRead: 0
MemWrite: 0
ALUOp: 11 (Indica operação de salto)
Branch: 0
jal: 0
jalr: 1 (Instrução jalr ativa)
halt: 0

- Instruções de Carga e Armazenamento (Load/Store)

Instruções: lw, sw
Formato: opcode (7 bits) | funct3 (3 bits) | rs1 (5 bits) | imm (12 bits) | rd (5 bits) para lw; rs2 (5 bits) para sw
Sinais de Controle:

Para lw:

ALUSrc: 1
MemtoReg: 1
RegWrite: 1
MemRead: 1
MemWrite: 0
ALUOp: 00
Branch: 0
jal: 0
jalr: 0
halt: 0

Para sw:

ALUSrc: 1

MemtoReg: 0
RegWrite: 0
MemRead: 0
MemWrite: 1 (Habilita escrita na memória)
ALUOp: 00
Branch: 0
jal: 0
jalr: 0
halt: 0

- Instruções de Ramificação (Branch)

Instruções: beq, bne, blt, bge
Formato: opcode (7 bits) | funct3 (3 bits) | rs1 (5 bits) | rs2 (5 bits) | imm (12 bits)
Sinais de Controle:

ALUSrc: 0
MemtoReg: 0
RegWrite: 0
MemRead: 0
MemWrite: 0
ALUOp: 01 (Indica operação de ramificação)
Branch: 1 (Indica que é uma instrução de ramificação)
jal: 0
jalr: 0
halt: 0

- Instruções de Salto (Jump)

Instruções: jal, jalr
Formato: opcode (7 bits) | imm (20 bits) para jal; rs1 (5 bits) | imm (12 bits) para jalr
Sinais de Controle:

Para jal:

ALUSrc: 1
MemtoReg: 0
RegWrite: 1
MemRead: 0
MemWrite: 0
ALUOp: 11
Branch: 0
jal: 1 (Instrução jal ativa)
jalr: 0

halt: 0
Para jalr:

ALUSrc: 1
MemtoReg: 0
RegWrite: 1
MemRead: 0
MemWrite: 0
ALUOp: 11
Branch: 0
jal: 0
jalr: 1 (Instrução jalr ativa)
halt: 0

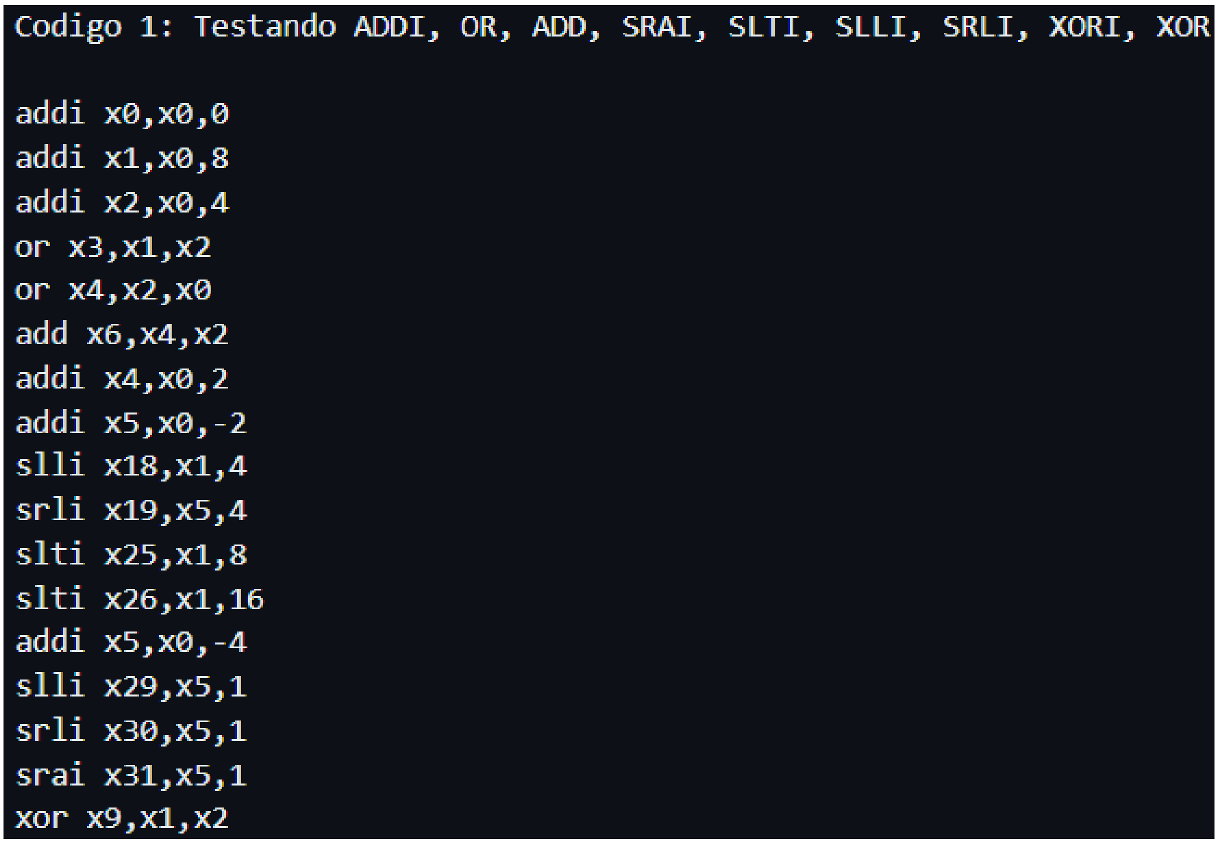
- Instrução de Parada (Halt)

Instrução: halt
Formato: opcode (7 bits) | 7'b1111111
Sinais de Controle:

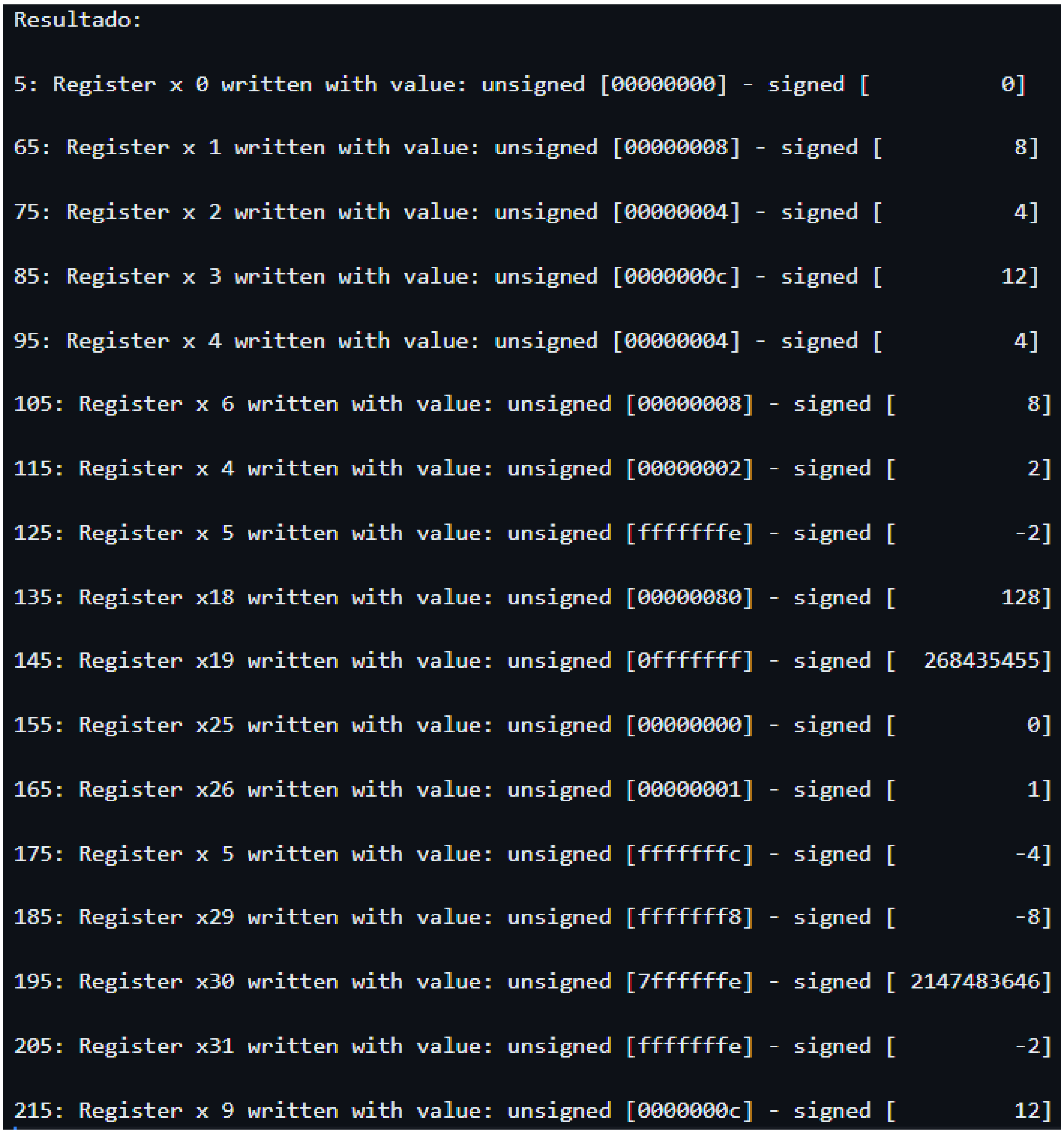
ALUSrc: 0
MemtoReg: 0
RegWrite: 0
MemRead: 0
MemWrite: 0
ALUOp: 00
Branch: 0
jal: 0
jalr: 0
halt: 1 (Instrução de parada ativa)

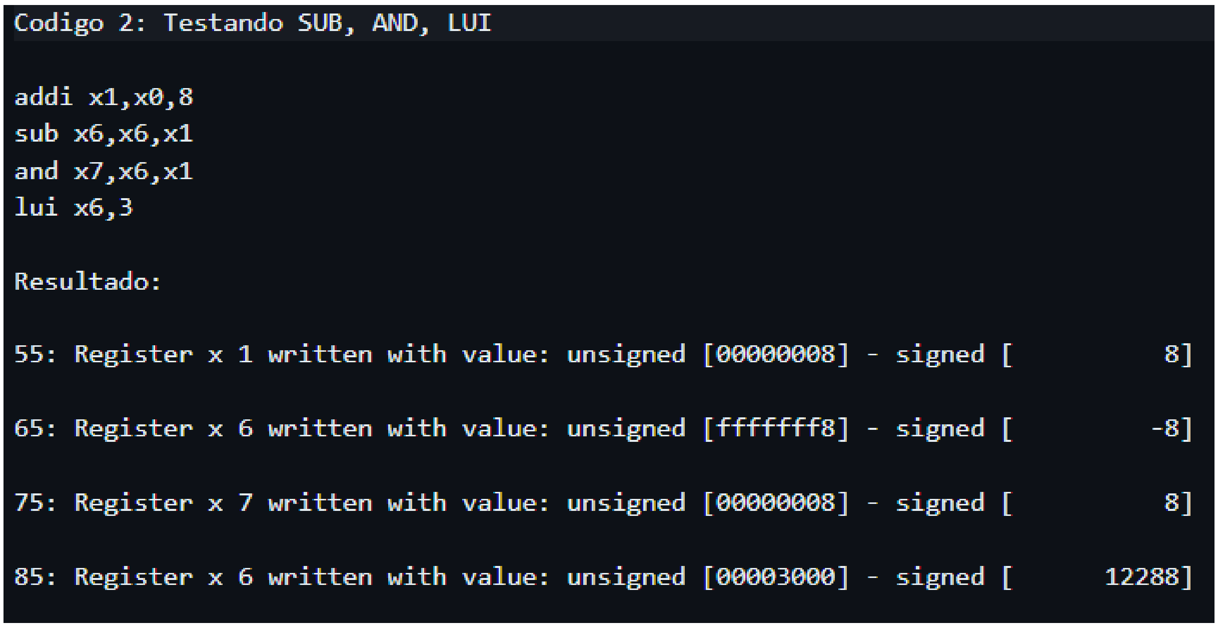
DESCRIÇÃO DAS SIMULAÇÕES REALIZADAS

Nessa seção, iremos apresentar os resultados das simulações aplicando cada um dos 13 testes disponíveis no repositório fornecido.



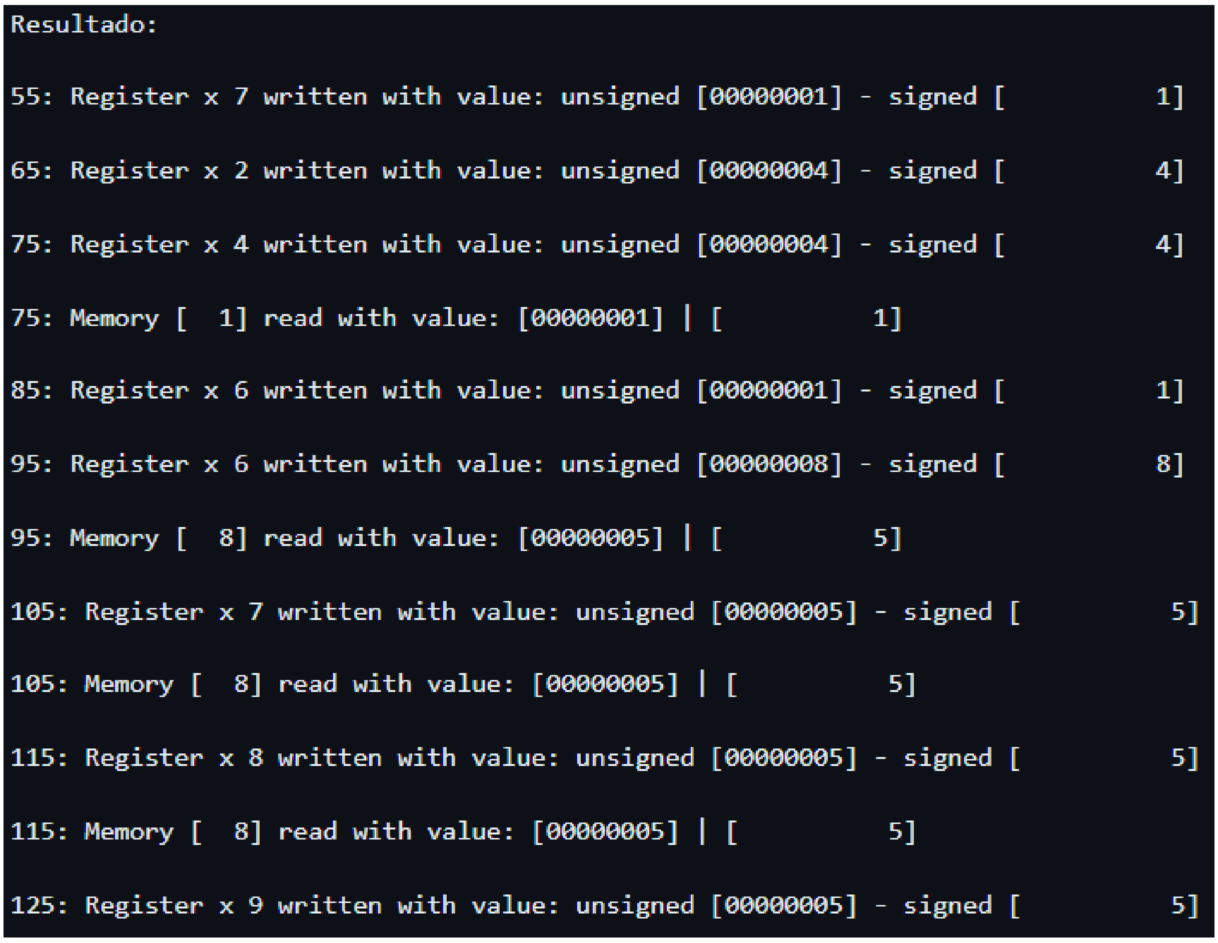
```
Codigo 1: Testando ADDI, OR, ADD, SRAI, SLTI, SLLI, SRLI, XORI, XOR

addi x0,x0,0
addi x1,x0,8
addi x2,x0,4
or x3,x1,x2
or x4,x2,x0
add x6,x4,x2
addi x4,x0,2
addi x5,x0,-2
slli x18,x1,4
srli x19,x5,4
slti x25,x1,8
slti x26,x1,16
addi x5,x0,-4
slli x29,x5,1
srli x30,x5,1
srai x31,x5,1
xor x9,x1,x2
```


```
Resultado:

5: Register x 0 written with value: unsigned [00000000] - signed [          0]

65: Register x 1 written with value: unsigned [00000008] - signed [          8]

75: Register x 2 written with value: unsigned [00000004] - signed [          4]

85: Register x 3 written with value: unsigned [0000000c] - signed [         12]

95: Register x 4 written with value: unsigned [00000004] - signed [          4]

105: Register x 6 written with value: unsigned [00000008] - signed [          8]

115: Register x 4 written with value: unsigned [00000002] - signed [          2]

125: Register x 5 written with value: unsigned [fffffffe] - signed [         -2]

135: Register x18 written with value: unsigned [00000080] - signed [        128]

145: Register x19 written with value: unsigned [0fffffff] - signed [  268435455]

155: Register x25 written with value: unsigned [00000000] - signed [          0]

165: Register x26 written with value: unsigned [00000001] - signed [          1]

175: Register x 5 written with value: unsigned [fffffffc] - signed [         -4]

185: Register x29 written with value: unsigned [fffffff8] - signed [         -8]

195: Register x30 written with value: unsigned [7ffffffe] - signed [ 2147483646]

205: Register x31 written with value: unsigned [fffffffe] - signed [         -2]

215: Register x 9 written with value: unsigned [0000000c] - signed [         12]
```

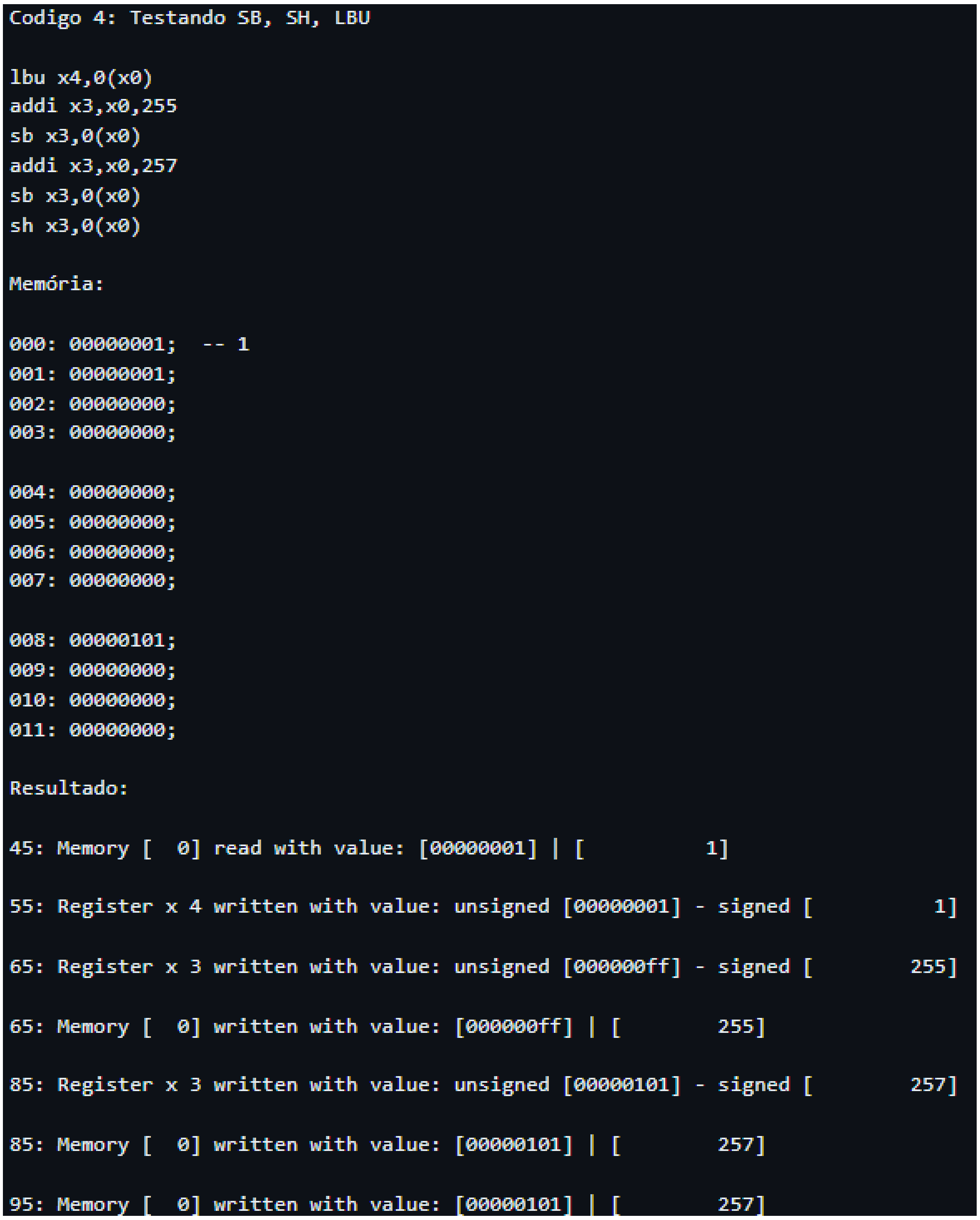


```
Codigo 2: Testando SUB, AND, LUI

addi x1,x0,8
sub x6,x6,x1
and x7,x6,x1
lui x6,3

Resultado:

55: Register x 1 written with value: unsigned [00000008] - signed [          8]

65: Register x 6 written with value: unsigned [fffffff8] - signed [         -8]

75: Register x 7 written with value: unsigned [00000008] - signed [          8]

85: Register x 6 written with value: unsigned [00003000] - signed [      12288]
```

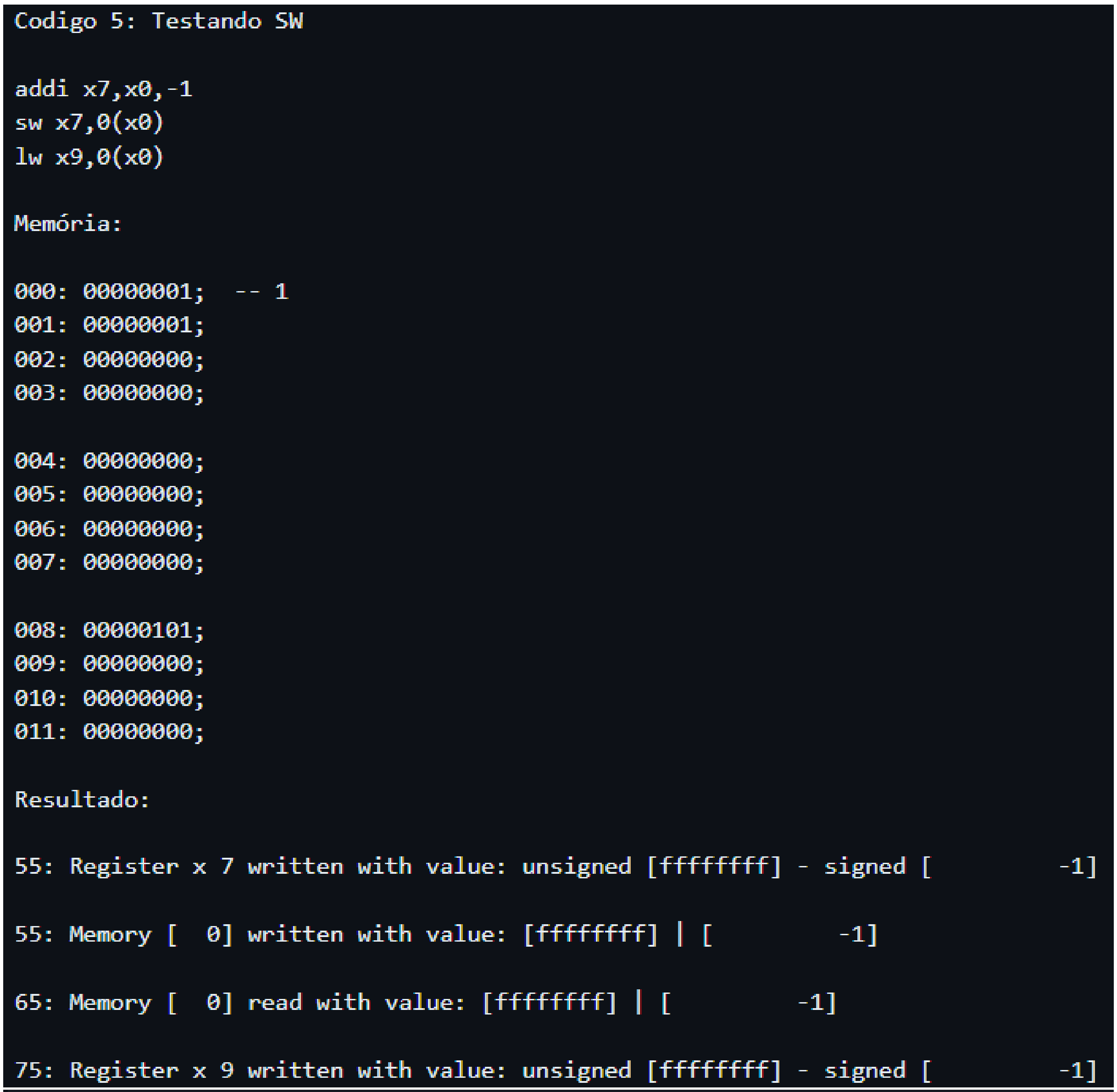
```
Codigo 3: Testando LB, LH, LW

addi x7,x0,1
addi x2,x0,4
or x4,x2,x0
lb x6,0(x7)
add x6,x4,x2
lb x7,0(x6)
lh x8,0(x6)
lw x9,0(x6)

Memória:

000: 00000001;  -- 1
001: 00000001;
002: 00000000;
003: 00000000;

004: 00000000;
005: 00000000;
006: 00000000;
007: 00000000;

008: 00000101;
009: 00000000;
010: 00000000;
011: 00000000;
```



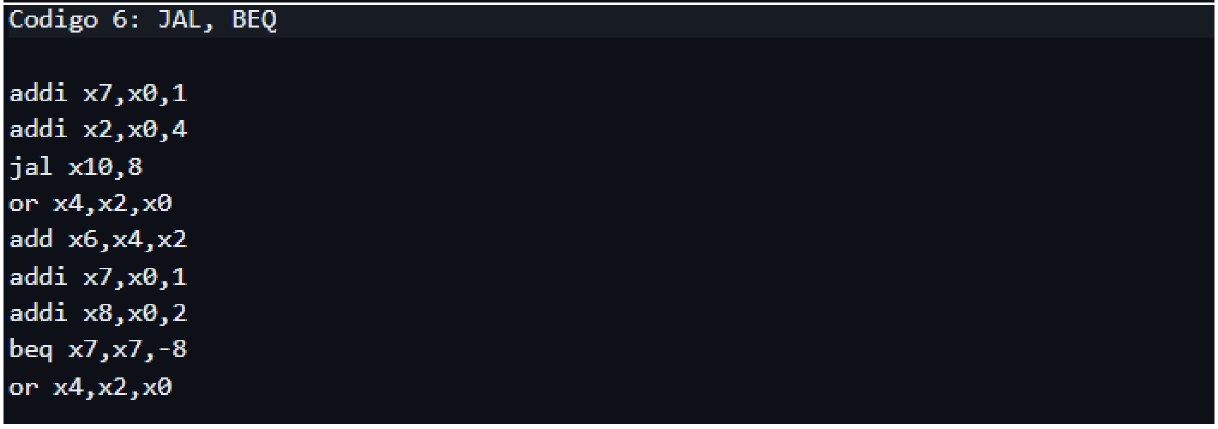
```
Resultado:

55: Register x 7 written with value: unsigned [00000001] - signed [          1]

65: Register x 2 written with value: unsigned [00000004] - signed [          4]

75: Register x 4 written with value: unsigned [00000004] - signed [          4]

75: Memory [  1] read with value: [00000001] | [          1]

85: Register x 6 written with value: unsigned [00000001] - signed [          1]

95: Register x 6 written with value: unsigned [00000008] - signed [          8]

95: Memory [  8] read with value: [00000005] | [          5]

105: Register x 7 written with value: unsigned [00000005] - signed [          5]

105: Memory [  8] read with value: [00000005] | [          5]

115: Register x 8 written with value: unsigned [00000005] - signed [          5]

115: Memory [  8] read with value: [00000005] | [          5]

125: Register x 9 written with value: unsigned [00000005] - signed [          5]
```



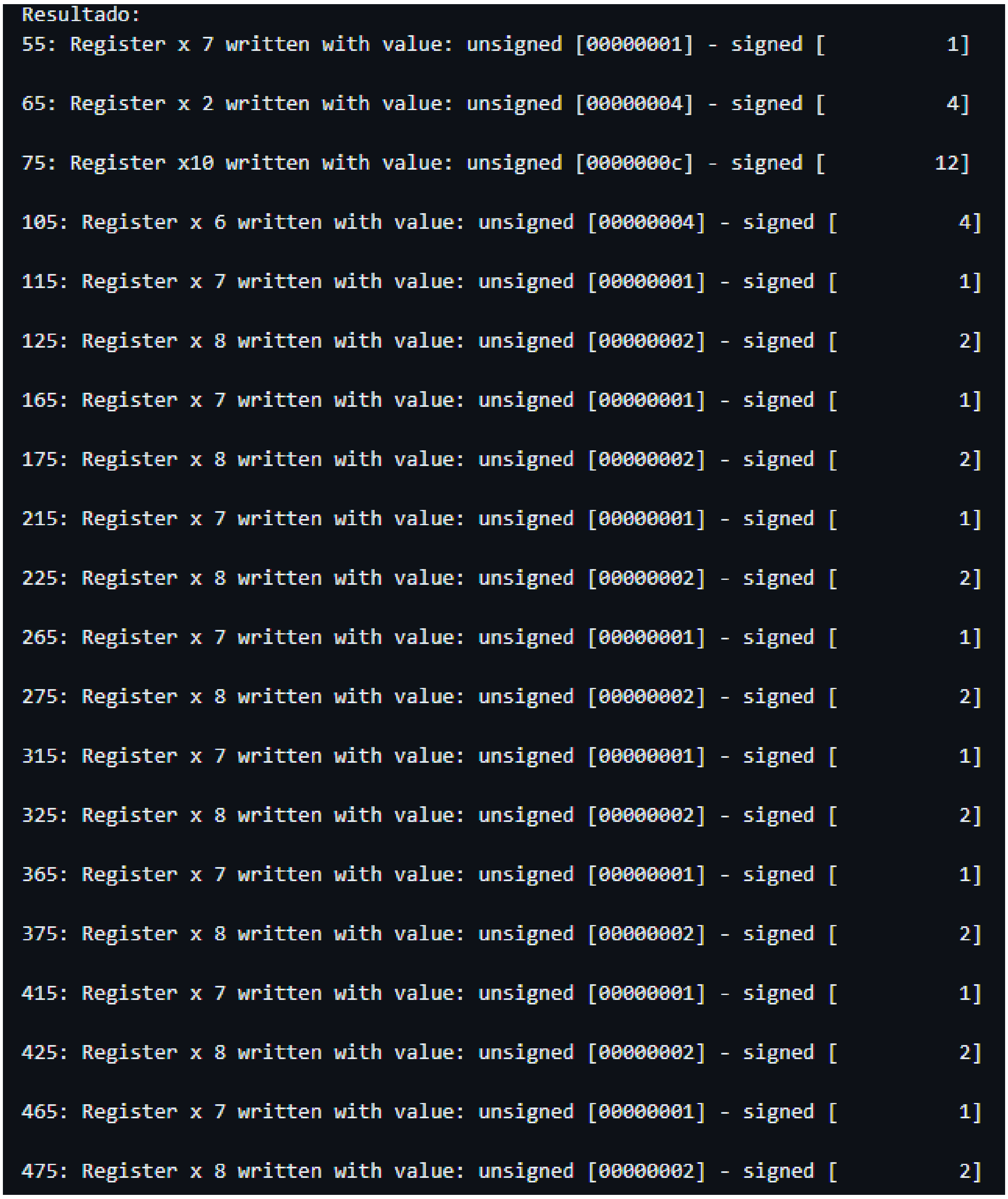
```
Codigo 4: Testando SB, SH, LBU

lbu x4,0(x0)
addi x3,x0,255
sb x3,0(x0)
addi x3,x0,257
sb x3,0(x0)
sh x3,0(x0)

Memória:

000: 00000001;  -- 1
001: 00000001;
002: 00000000;
003: 00000000;

004: 00000000;
005: 00000000;
006: 00000000;
007: 00000000;

008: 00000101;
009: 00000000;
010: 00000000;
011: 00000000;

Resultado:

45: Memory [  0] read with value: [00000001] | [          1]

55: Register x 4 written with value: unsigned [00000001] - signed [          1]

65: Register x 3 written with value: unsigned [000000ff] - signed [        255]

65: Memory [  0] written with value: [000000ff] | [        255]

85: Register x 3 written with value: unsigned [00000101] - signed [        257]

85: Memory [  0] written with value: [00000101] | [        257]

95: Memory [  0] written with value: [00000101] | [        257]
```



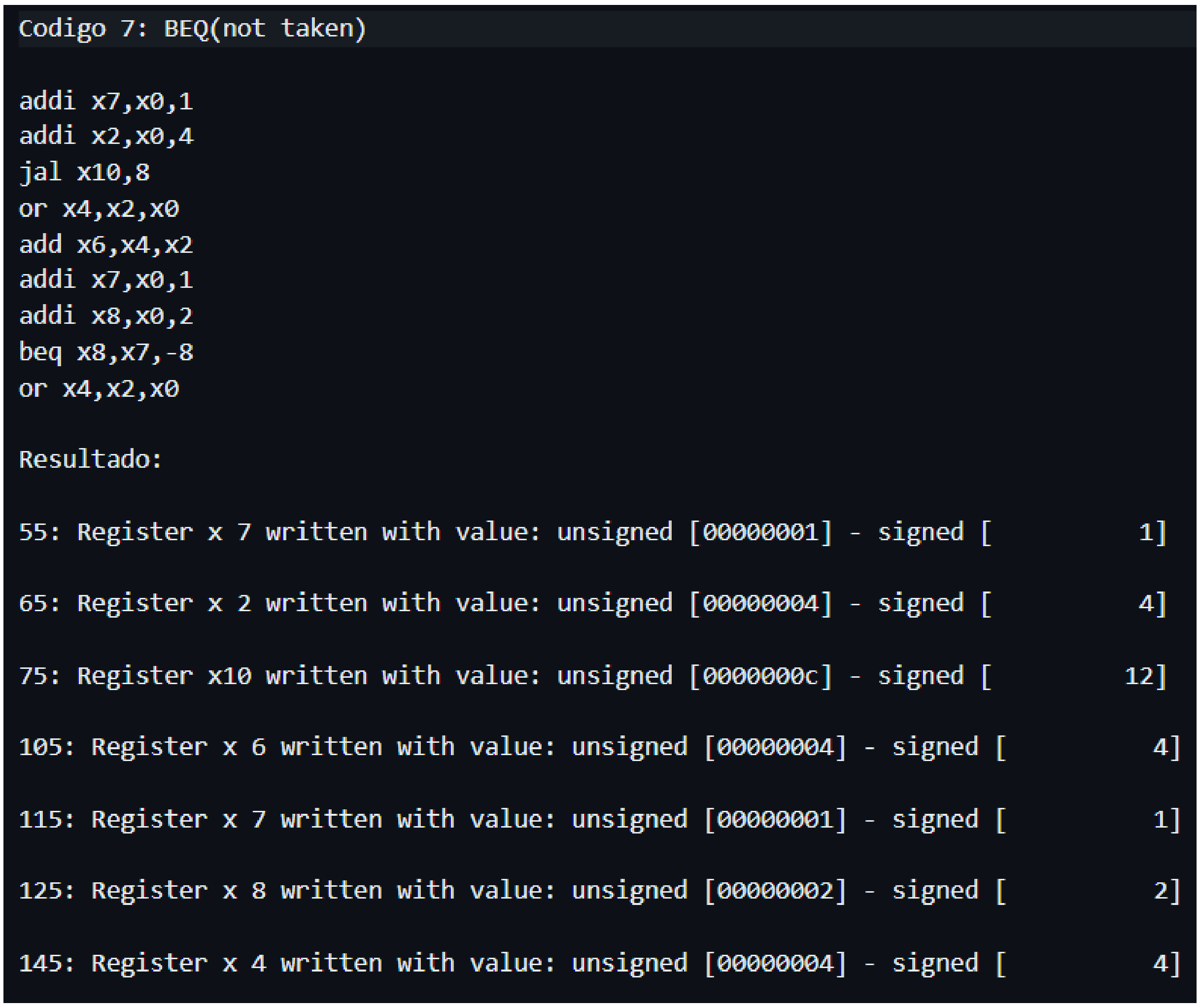
```
Codigo 5: Testando SW

addi x7,x0,-1
sw x7,0(x0)
lw x9,0(x0)

Memória:

000: 00000001;  -- 1
001: 00000001;
002: 00000000;
003: 00000000;

004: 00000000;
005: 00000000;
006: 00000000;
007: 00000000;

008: 00000101;
009: 00000000;
010: 00000000;
011: 00000000;

Resultado:

55: Register x 7 written with value: unsigned [ffffffff] - signed [         -1]

55: Memory [  0] written with value: [ffffffff] | [         -1]

65: Memory [  0] read with value: [ffffffff] | [         -1]

75: Register x 9 written with value: unsigned [ffffffff] - signed [         -1]
```



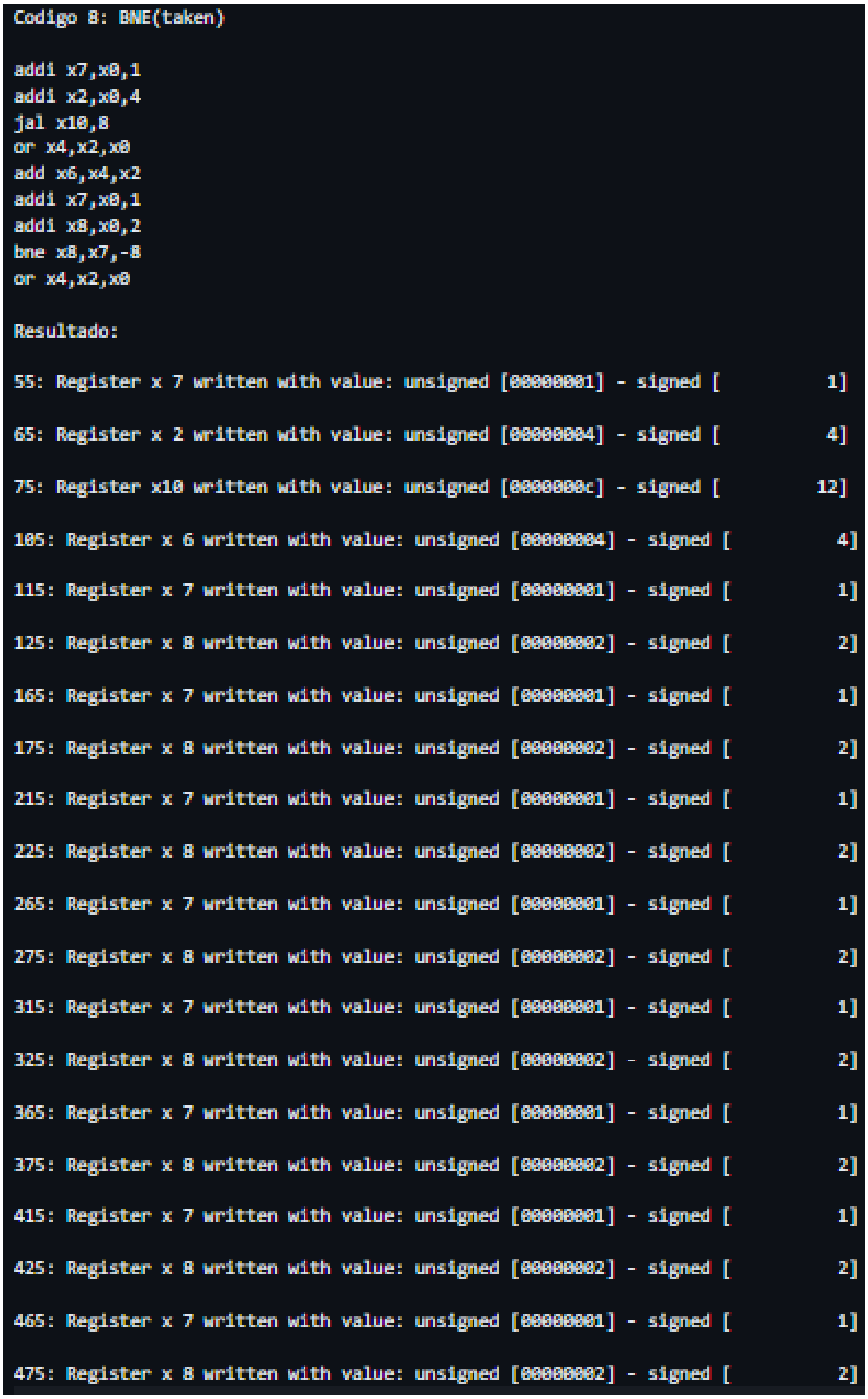
```
Codigo 6: JAL, BEQ

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,1
addi x8,x0,2
beq x7,x7,-8
or x4,x2,x0
```



```
Resultado:
55: Register x 7 written with value: unsigned [00000001] - signed [          1]

65: Register x 2 written with value: unsigned [00000004] - signed [          4]

75: Register x10 written with value: unsigned [0000000c] - signed [         12]

105: Register x 6 written with value: unsigned [00000004] - signed [          4]

115: Register x 7 written with value: unsigned [00000001] - signed [          1]

125: Register x 8 written with value: unsigned [00000002] - signed [          2]

165: Register x 7 written with value: unsigned [00000001] - signed [          1]

175: Register x 8 written with value: unsigned [00000002] - signed [          2]

215: Register x 7 written with value: unsigned [00000001] - signed [          1]

225: Register x 8 written with value: unsigned [00000002] - signed [          2]

265: Register x 7 written with value: unsigned [00000001] - signed [          1]

275: Register x 8 written with value: unsigned [00000002] - signed [          2]

315: Register x 7 written with value: unsigned [00000001] - signed [          1]

325: Register x 8 written with value: unsigned [00000002] - signed [          2]

365: Register x 7 written with value: unsigned [00000001] - signed [          1]

375: Register x 8 written with value: unsigned [00000002] - signed [          2]

415: Register x 7 written with value: unsigned [00000001] - signed [          1]

425: Register x 8 written with value: unsigned [00000002] - signed [          2]

465: Register x 7 written with value: unsigned [00000001] - signed [          1]

475: Register x 8 written with value: unsigned [00000002] - signed [          2]
```

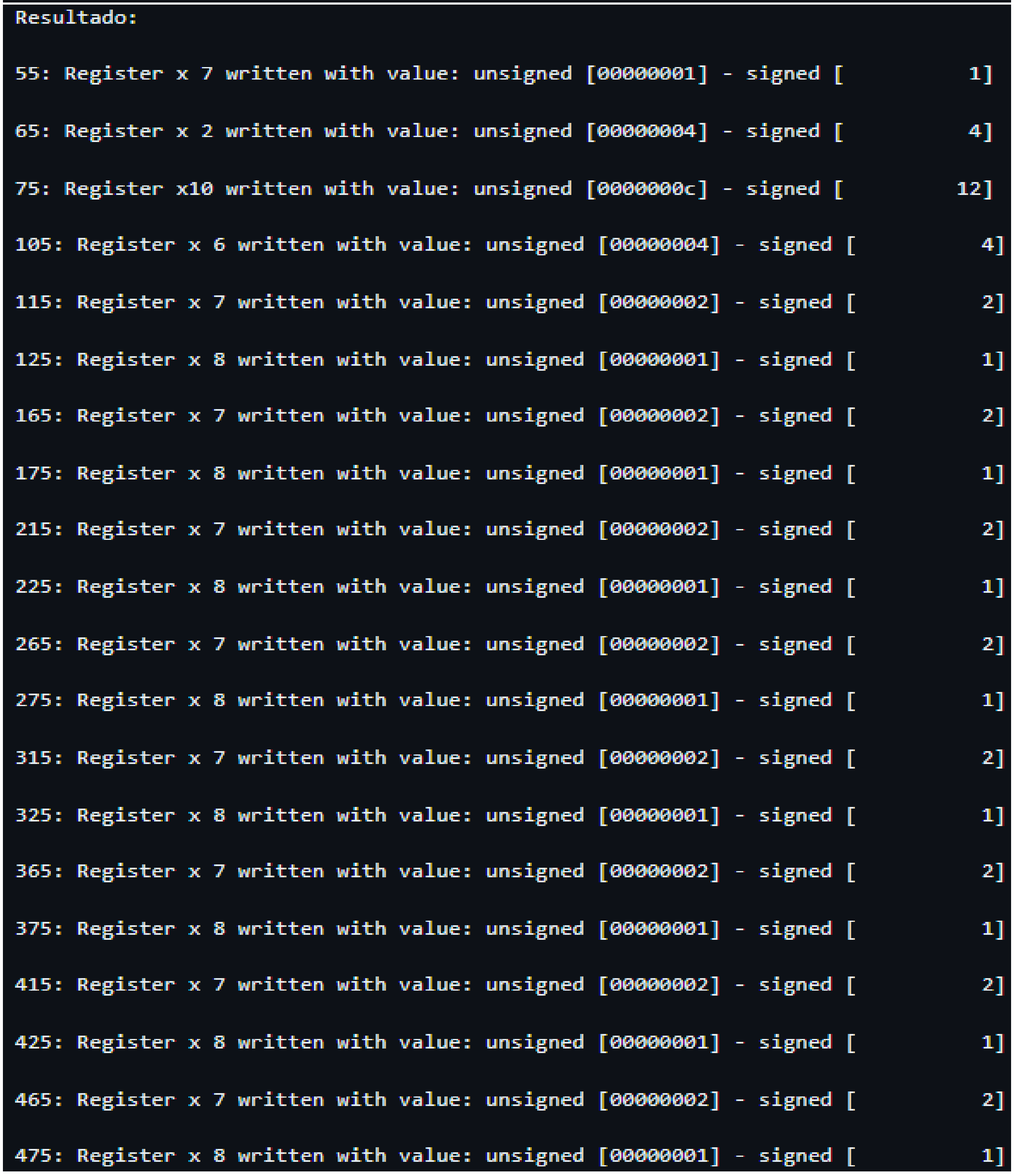


```
Codigo 7: BEQ(not taken)

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,1
addi x8,x0,2
beq x8,x7,-8
or x4,x2,x0

Resultado:

55: Register x 7 written with value: unsigned [00000001] - signed [          1]

65: Register x 2 written with value: unsigned [00000004] - signed [          4]

75: Register x10 written with value: unsigned [0000000c] - signed [         12]

105: Register x 6 written with value: unsigned [00000004] - signed [          4]

115: Register x 7 written with value: unsigned [00000001] - signed [          1]

125: Register x 8 written with value: unsigned [00000002] - signed [          2]

145: Register x 4 written with value: unsigned [00000004] - signed [          4]
```

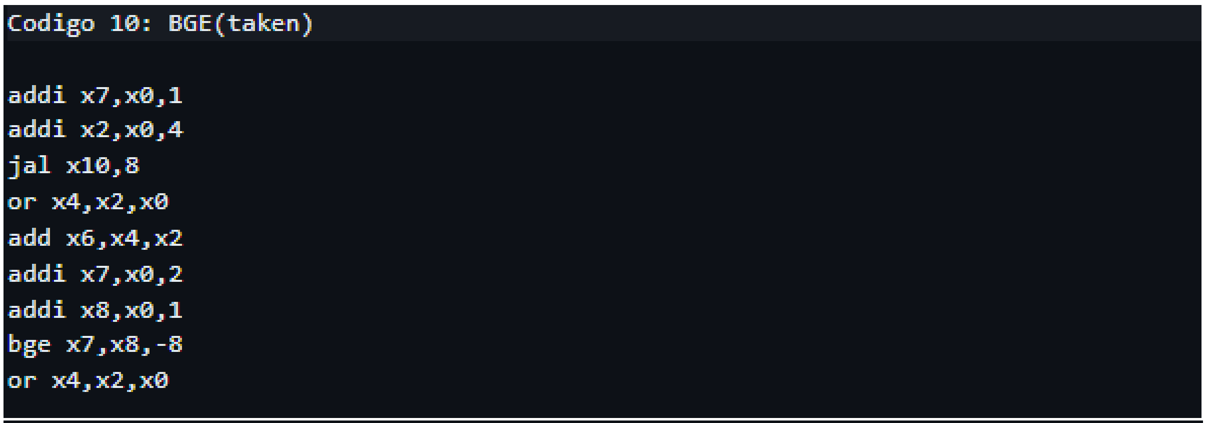


```
Codigo 8: BNE(taken)

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,1
addi x8,x0,2
bne x8,x7,-8
or x4,x2,x0

Resultado:

55: Register x 7 written with value: unsigned [00000001] - signed [          1]

65: Register x 2 written with value: unsigned [00000004] - signed [          4]

75: Register x10 written with value: unsigned [0000000c] - signed [         12]

105: Register x 6 written with value: unsigned [00000004] - signed [          4]

115: Register x 7 written with value: unsigned [00000001] - signed [          1]

125: Register x 8 written with value: unsigned [00000002] - signed [          2]

165: Register x 7 written with value: unsigned [00000001] - signed [          1]

175: Register x 8 written with value: unsigned [00000002] - signed [          2]

215: Register x 7 written with value: unsigned [00000001] - signed [          1]

225: Register x 8 written with value: unsigned [00000002] - signed [          2]

265: Register x 7 written with value: unsigned [00000001] - signed [          1]

275: Register x 8 written with value: unsigned [00000002] - signed [          2]

315: Register x 7 written with value: unsigned [00000001] - signed [          1]

325: Register x 8 written with value: unsigned [00000002] - signed [          2]

365: Register x 7 written with value: unsigned [00000001] - signed [          1]

375: Register x 8 written with value: unsigned [00000002] - signed [          2]

415: Register x 7 written with value: unsigned [00000001] - signed [          1]

425: Register x 8 written with value: unsigned [00000002] - signed [          2]

465: Register x 7 written with value: unsigned [00000001] - signed [          1]

475: Register x 8 written with value: unsigned [00000002] - signed [          2]
```

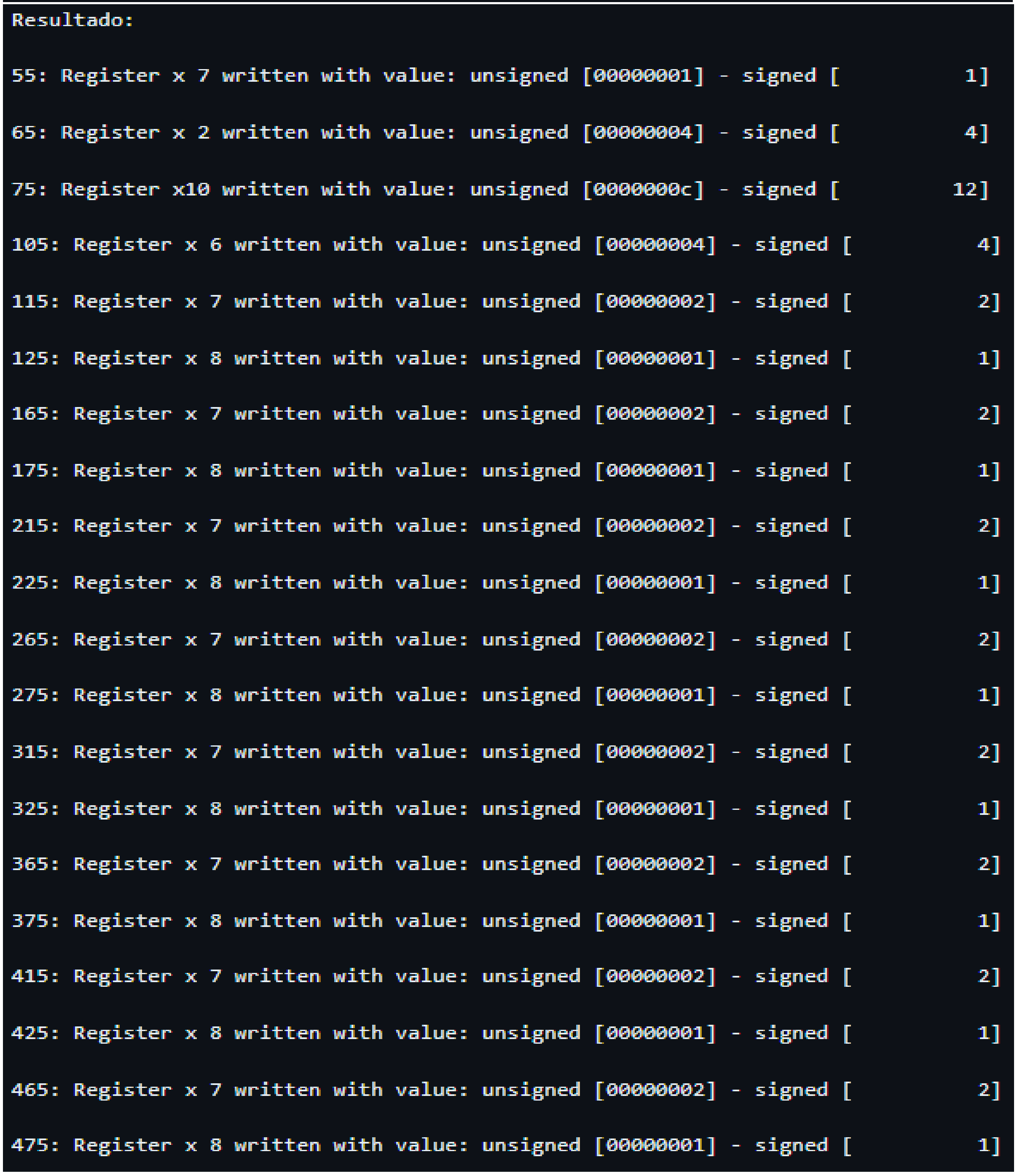
```
Codigo 9: BLT(taken)

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,2
addi x8,x0,1
blt x8,x7,-8
or x4,x2,x0
```



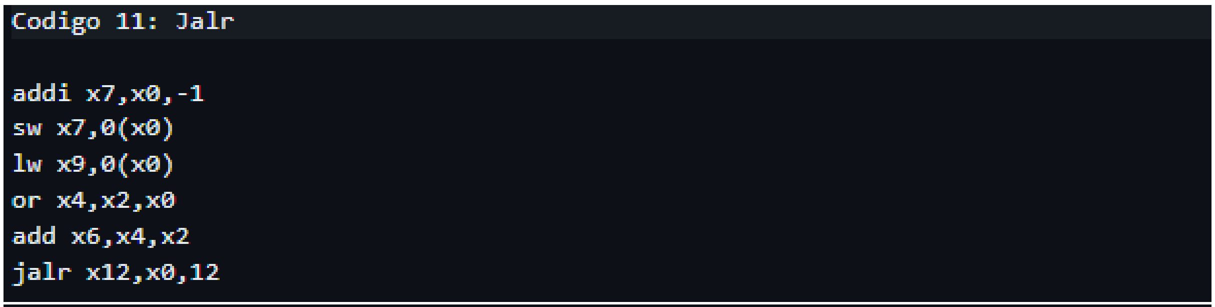
```
Resultado:

55: Register x 7 written with value: unsigned [00000001] - signed [          1]

65: Register x 2 written with value: unsigned [00000004] - signed [          4]

75: Register x10 written with value: unsigned [0000000c] - signed [         12]

105: Register x 6 written with value: unsigned [00000004] - signed [          4]

115: Register x 7 written with value: unsigned [00000002] - signed [          2]

125: Register x 8 written with value: unsigned [00000001] - signed [          1]

165: Register x 7 written with value: unsigned [00000002] - signed [          2]

175: Register x 8 written with value: unsigned [00000001] - signed [          1]

215: Register x 7 written with value: unsigned [00000002] - signed [          2]

225: Register x 8 written with value: unsigned [00000001] - signed [          1]

265: Register x 7 written with value: unsigned [00000002] - signed [          2]

275: Register x 8 written with value: unsigned [00000001] - signed [          1]

315: Register x 7 written with value: unsigned [00000002] - signed [          2]

325: Register x 8 written with value: unsigned [00000001] - signed [          1]

365: Register x 7 written with value: unsigned [00000002] - signed [          2]

375: Register x 8 written with value: unsigned [00000001] - signed [          1]

415: Register x 7 written with value: unsigned [00000002] - signed [          2]

425: Register x 8 written with value: unsigned [00000001] - signed [          1]

465: Register x 7 written with value: unsigned [00000002] - signed [          2]

475: Register x 8 written with value: unsigned [00000001] - signed [          1]
```



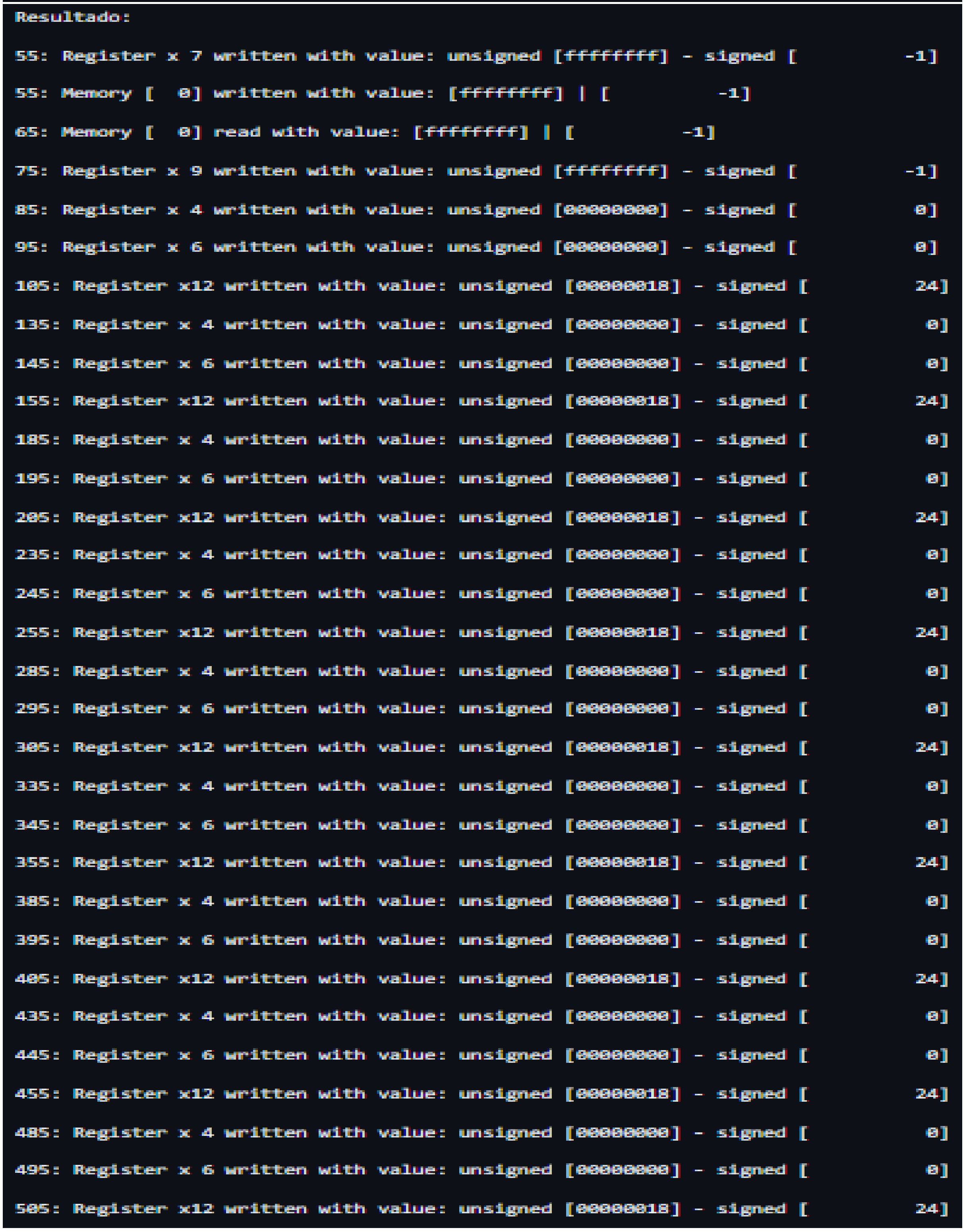
```
Codigo 10: BGE(taken)

addi x7,x0,1
addi x2,x0,4
jal x10,8
or x4,x2,x0
add x6,x4,x2
addi x7,x0,2
addi x8,x0,1
bge x7,x8,-8
or x4,x2,x0
```



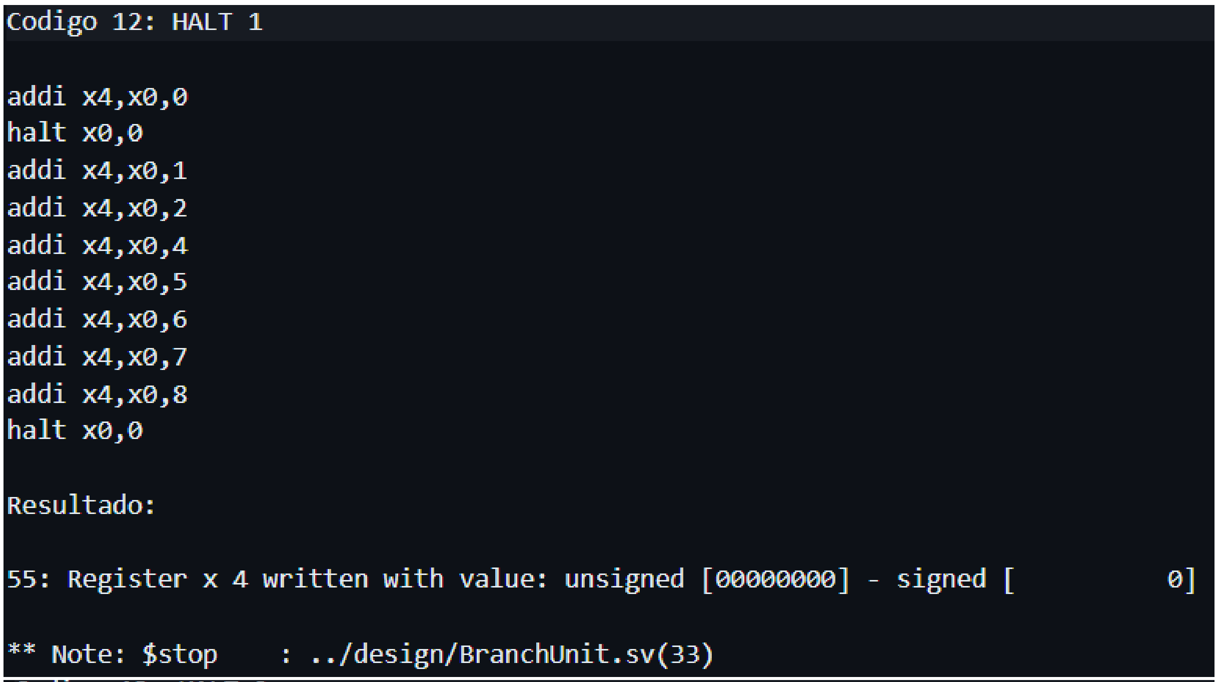
```
Resultado:

55: Register x 7 written with value: unsigned [00000001] - signed [          1]

65: Register x 2 written with value: unsigned [00000004] - signed [          4]

75: Register x10 written with value: unsigned [0000000c] - signed [         12]

105: Register x 6 written with value: unsigned [00000004] - signed [          4]

115: Register x 7 written with value: unsigned [00000002] - signed [          2]

125: Register x 8 written with value: unsigned [00000001] - signed [          1]

165: Register x 7 written with value: unsigned [00000002] - signed [          2]

175: Register x 8 written with value: unsigned [00000001] - signed [          1]

215: Register x 7 written with value: unsigned [00000002] - signed [          2]

225: Register x 8 written with value: unsigned [00000001] - signed [          1]

265: Register x 7 written with value: unsigned [00000002] - signed [          2]

275: Register x 8 written with value: unsigned [00000001] - signed [          1]

315: Register x 7 written with value: unsigned [00000002] - signed [          2]

325: Register x 8 written with value: unsigned [00000001] - signed [          1]

365: Register x 7 written with value: unsigned [00000002] - signed [          2]

375: Register x 8 written with value: unsigned [00000001] - signed [          1]

415: Register x 7 written with value: unsigned [00000002] - signed [          2]

425: Register x 8 written with value: unsigned [00000001] - signed [          1]

465: Register x 7 written with value: unsigned [00000002] - signed [          2]

475: Register x 8 written with value: unsigned [00000001] - signed [          1]
```



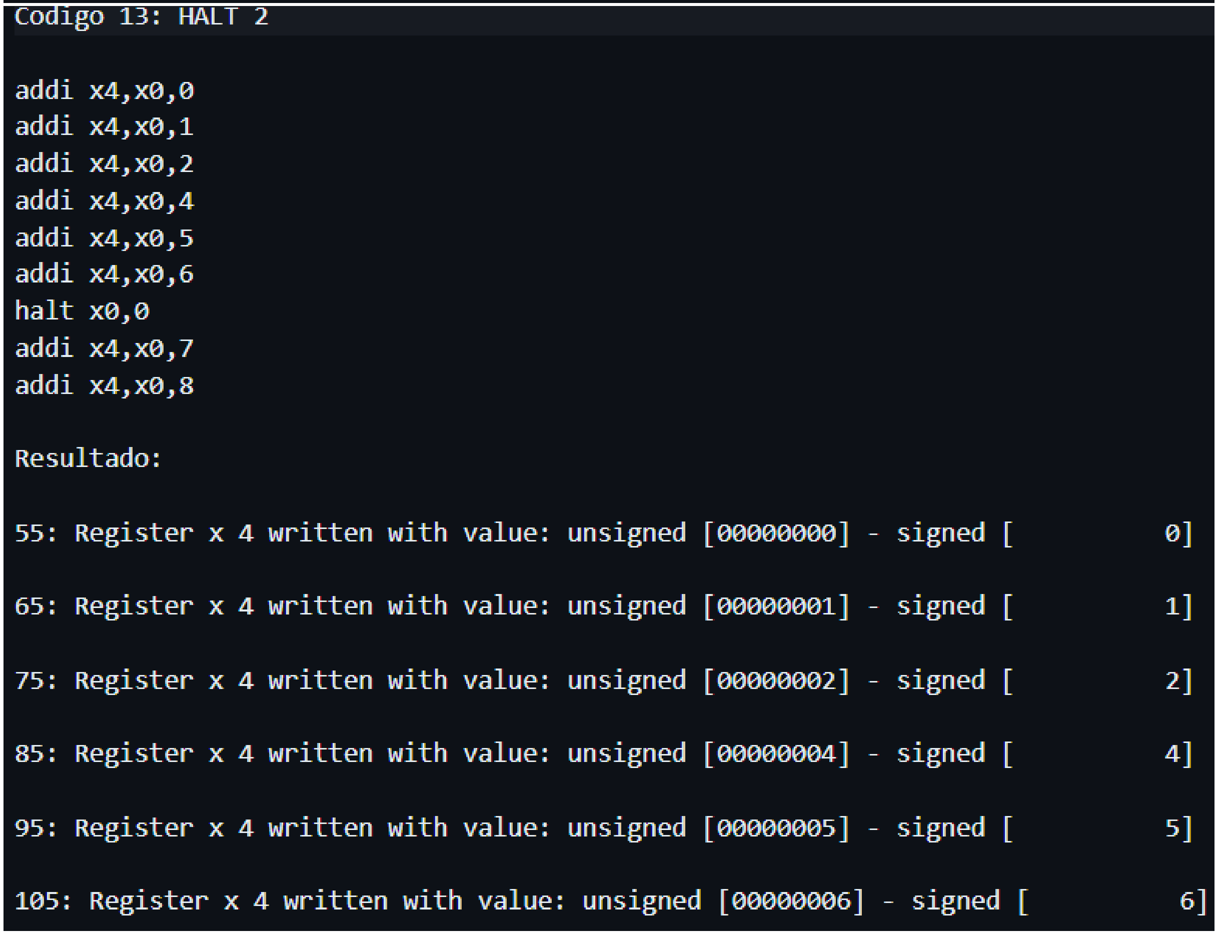
```
Codigo 11: Jalr

addi x7,x0,-1
sw x7,0(x0)
lw x9,0(x0)
or x4,x2,x0
add x6,x4,x2
jalr x12,x0,12
```



```
Resultado:

55: Register x 7 written with value: unsigned [ffffffff] - signed [          -1]

55: Memory [  0] written with value: [ffffffff] | [          -1]

65: Memory [  0] read with value: [ffffffff] | [          -1]

75: Register x 9 written with value: unsigned [ffffffff] - signed [          -1]

85: Register x 4 written with value: unsigned [00000000] - signed [           0]

95: Register x 6 written with value: unsigned [00000000] - signed [           0]

105: Register x12 written with value: unsigned [00000018] - signed [          24]

135: Register x 4 written with value: unsigned [00000000] - signed [           0]

145: Register x 6 written with value: unsigned [00000000] - signed [           0]

155: Register x12 written with value: unsigned [00000018] - signed [          24]

185: Register x 4 written with value: unsigned [00000000] - signed [           0]

195: Register x 6 written with value: unsigned [00000000] - signed [           0]

205: Register x12 written with value: unsigned [00000018] - signed [          24]

235: Register x 4 written with value: unsigned [00000000] - signed [           0]

245: Register x 6 written with value: unsigned [00000000] - signed [           0]

255: Register x12 written with value: unsigned [00000018] - signed [          24]

285: Register x 4 written with value: unsigned [00000000] - signed [           0]

295: Register x 6 written with value: unsigned [00000000] - signed [           0]

305: Register x12 written with value: unsigned [00000018] - signed [          24]

335: Register x 4 written with value: unsigned [00000000] - signed [           0]

345: Register x 6 written with value: unsigned [00000000] - signed [           0]

355: Register x12 written with value: unsigned [00000018] - signed [          24]

385: Register x 4 written with value: unsigned [00000000] - signed [           0]

395: Register x 6 written with value: unsigned [00000000] - signed [           0]

405: Register x12 written with value: unsigned [00000018] - signed [          24]

435: Register x 4 written with value: unsigned [00000000] - signed [           0]

445: Register x 6 written with value: unsigned [00000000] - signed [           0]

455: Register x12 written with value: unsigned [00000018] - signed [          24]

485: Register x 4 written with value: unsigned [00000000] - signed [           0]

495: Register x 6 written with value: unsigned [00000000] - signed [           0]

505: Register x12 written with value: unsigned [00000018] - signed [          24]
```



```
Codigo 12: HALT 1

addi x4,x0,0
halt x0,0
addi x4,x0,1
addi x4,x0,2
addi x4,x0,4
addi x4,x0,5
addi x4,x0,6
addi x4,x0,7
addi x4,x0,8
halt x0,0

Resultado:

55: Register x 4 written with value: unsigned [00000000] - signed [          0]

** Note: $stop    : ../design/BranchUnit.sv(33)
```



```
Codigo 13: HALT 2

addi x4,x0,0
addi x4,x0,1
addi x4,x0,2
addi x4,x0,4
addi x4,x0,5
addi x4,x0,6
halt x0,0
addi x4,x0,7
addi x4,x0,8

Resultado:

55: Register x 4 written with value: unsigned [00000000] - signed [          0]

65: Register x 4 written with value: unsigned [00000001] - signed [          1]

75: Register x 4 written with value: unsigned [00000002] - signed [          2]

85: Register x 4 written with value: unsigned [00000004] - signed [          4]

95: Register x 4 written with value: unsigned [00000005] - signed [          5]

105: Register x 4 written with value: unsigned [00000006] - signed [          6]
```

CONCLUSÃO

Ao analisarmos o que se foi pedido e o que conseguimos implementar, concluímos que fomos capazes de conseguir os resultados esperados ao realizar as implementações das instruções pedidas. Quando analisamos nossos resultados ao se executar os casos testes presentes no repositório e em outros dispositivos de simulação, percebemos que os valores estão conforme o esperado em todos eles. Todos os objetivos foram alcançados.