

# JoSDC Benchmarks – Phase II

## Instructions:

1. All benchmarks are written in assembly with a brief description or C++ code snippet.
2. Each team is responsible for generating its own machine code and making necessary modifications to the code to fit the design.
3. When the team runs any of the benchmarks below, make sure to show the CPU state before and after execution. In addition, you need to show the memory footprint and the register values such as the program counter and other general-purpose register values before and after.
4. When you add the simulation graph to your report, make sure you add a description of the waveform and a clear list of the signals of interest. Besides that, add another zoomed-in waveform to focus on the state of the signals of interest at a specific time to assist the reviewer in knowing what signals to look at and what to expect regarding the signal's behavior.
5. The result after running the benchmarks and any modifications should be clearly mentioned in the report and reflected in the waveforms, FPGA implementation, or both.
6. Make the necessary RAM and registers initialization for each benchmark.
7. The numbering system used for the benchmarks is decimal.

# 1. Arrays#1:

Description/C code:

- Declare arrays a, b, & c
- b is an array of 13 elements, all initialized with the same value '0x32'
- c is an array of the ASCII values of 'hello', ends in 0
- Initialize your design RAM and registers as needed.

Assembly

#Declare arrays  
.data  
a: .word 0x3, 0x8, 0x5, 0x2  
b: .word 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32, 0x32  
c: .word 'hello', 0  
#Store array values  
.text  
LW R8, 0(R0)  
LW R9, 0x20(R0)  
LW R10, 0x50(R0)  
LW R11, 0x8(R0)

R8 = 3 ✓  
R9 = 32  
R10 = ascii(L)  
R11 = 5

## 2. Arrays#2:

Description/C code:

- Declare an array a of size 5  $\rightarrow a[4] = a[3] = a[2] = a[1] = a[0] = 0x3$
- Initialize your design RAM and registers as needed.
- execute  $a[1] = a[2] * 4$
- execute  $a[3] = a[4] * 2$

Assembly

```
.data  
a: .word 0x3, 0x3, 0x3, 0x3, 0x3
```

```
.text
```

```
LW  R1, 8(R0)  
SLL R1, R1, 2  
SW  R1, 4(R0)  
LW  R2, 4(R0)  
LW  R3, 16(R0)  
SLL R3, R2, 1  
SW  R3, 12(R0)  
LW  R4, 12(R0)
```

$R1 = 8 \cdot 12$

$R2 = 12$

$R3 = R4 = 0x18$

### 3. Arithmetic Operations

Description/C code:

Declare x, y, and z variables  
Calculate  $-(x + y - 2 * z + 1)$

Assembly

```
.data
x: .word 0x23
y: .word 0x2F
z: .word 0x1A
```

+R28

R8 = x

R9 = y

R10 = z

```
.text
```

```
LW R8, 0(R28)
```

```
LW R9, 4(R28)
```

```
ADD R8, R8, R9
```

```
LW R10, 8(R28)
```

```
ADD R10, R10, R10
```

```
SUB R8, R8, R10
```

```
ADDI R8, R8, 1
```

```
SUB R8, R0, R8
```

R8 = x + y

z = 2z

x + y - 2z

x + y - 2z + 1

$-(x + y - 2z + 1)$

at the end  $\rightarrow R8 = -31 = 0xFFFFFFFFE1$

#### 4. Conditional statements#1:

Description/C code:

- Compare R8 and R9 and loop until R8 greater than R9
- When done, store R9 in R13
- Perform some logical operations

Assembly

ADD R8, R8, R0 # assuming R0 has the value of zero.  
ADDI R9, R9, 10 *assume decimal*  
*it's value (don't care) so i changed this to R0*  
#compare R8 and R9 and loop until R8 greater than R9  
LOOP: SUB R10, R8, R9 *R10 = -10*  
SLT R11, R0, R10  
ADDI R12, R0, 1  
BEQ R11, R12, DONE  
ADDI R8, R8, 1  
JUMP LOOP  
  
#when done, store R9 in R13  
DONE: ADD R13, R9, R0  
ADDI R14, R0, 18 *?*  
ANDI R14, R14, 17 *27 ?*  
*23 !*

## 5. Conditional statements#2:

Description/ C code:

```
if (a < b + 3)
```

```
    a = a + 1
```

```
else
```

```
    a = a + 2
```

```
    b = b + a
```

- Assuming R0 is initialized with zero
- Try once with a set to 2 and another time with a set to 6
- Make sure you include the results of both trials in the report.

Assembly

```
→ ADDI R1, R0, 2
```

```
→ ADDI R2, R0, 2
```

```
→ ADDI R3, R2, 3
```

```
SLT R4, R1, R3
```

```
BEQ R4, R0, then
```

```
ADDI R1, R1, 1
```

```
JUMP end
```

```
then: → ADDI R1, R1, 2
```

```
end: → ADD R1, R1, R0
```

$a$   $R1 = 2$   
 $b$   $R2 = 2$   
 $(b+3)$   $R3 = 5$

$a$   $R1 = 6$   
 $b$   $R2 = 2$   
 $(b+3)$   $R3 = 5$

!!! ADD R2, R2, R1

*Note: Use the highlighted instructions are equivalent to BLT pseudo instruction → slt & beq. Modify the assembly code to work as expected with your design*

## 6. While Loop:

Description/C code:

- Initialize R9 with 100.
- Increment R1 by 1 until it equals to R9

Assembly

```
START: ADD R1, R0, R0  
       ADD R2, R0, R0  
       ADDI R9, R0, 100  
LOOP:  BEQ R1, R9, EXIT  
       ADDI R1, R1, 1  
       JUMP START  
EXIT:  ADD R3, R0, R0
```

Handwritten annotations:

- Blue arrows pointing to the first and second instructions.
- Blue arrows pointing to the third and fourth instructions.
- Blue arrow pointing to the fifth instruction.
- Blue arrow pointing to the sixth instruction.
- Blue arrow pointing to the seventh instruction.
- Blue arrow pointing to the eighth instruction.
- Blue arrow pointing to the ninth instruction.
- Blue arrow pointing to the tenth instruction.
- Blue arrow pointing to the eleventh instruction.
- Blue arrow pointing to the twelfth instruction.
- Blue arrow pointing to the thirteenth instruction.
- Blue arrow pointing to the fourteenth instruction.
- Blue arrow pointing to the fifteenth instruction.
- Blue arrow pointing to the sixteenth instruction.
- Blue arrow pointing to the seventeenth instruction.
- Blue arrow pointing to the eighteenth instruction.
- Blue arrow pointing to the nineteenth instruction.
- Blue arrow pointing to the twentieth instruction.
- Blue arrow pointing to the twenty-first instruction.
- Blue arrow pointing to the twenty-second instruction.
- Blue arrow pointing to the twenty-third instruction.
- Blue arrow pointing to the twenty-fourth instruction.
- Blue arrow pointing to the twenty-fifth instruction.
- Blue arrow pointing to the twenty-sixth instruction.
- Blue arrow pointing to the twenty-seventh instruction.
- Blue arrow pointing to the twenty-eighth instruction.
- Blue arrow pointing to the twenty-ninth instruction.
- Blue arrow pointing to the thirtieth instruction.
- Blue arrow pointing to the thirty-first instruction.
- Blue arrow pointing to the thirty-second instruction.
- Blue arrow pointing to the thirty-third instruction.
- Blue arrow pointing to the thirty-fourth instruction.
- Blue arrow pointing to the thirty-fifth instruction.
- Blue arrow pointing to the thirty-sixth instruction.
- Blue arrow pointing to the thirty-seventh instruction.
- Blue arrow pointing to the thirty-eighth instruction.
- Blue arrow pointing to the thirty-ninth instruction.
- Blue arrow pointing to the fortieth instruction.
- Blue arrow pointing to the forty-first instruction.
- Blue arrow pointing to the forty-second instruction.
- Blue arrow pointing to the forty-third instruction.
- Blue arrow pointing to the forty-fourth instruction.
- Blue arrow pointing to the forty-fifth instruction.
- Blue arrow pointing to the forty-sixth instruction.
- Blue arrow pointing to the forty-seventh instruction.
- Blue arrow pointing to the forty-eighth instruction.
- Blue arrow pointing to the forty-ninth instruction.
- Blue arrow pointing to the fiftieth instruction.
- Blue arrow pointing to the fifty-first instruction.
- Blue arrow pointing to the fifty-second instruction.
- Blue arrow pointing to the fifty-third instruction.
- Blue arrow pointing to the fifty-fourth instruction.
- Blue arrow pointing to the fifty-fifth instruction.
- Blue arrow pointing to the fifty-sixth instruction.
- Blue arrow pointing to the fifty-seventh instruction.
- Blue arrow pointing to the fifty-eighth instruction.
- Blue arrow pointing to the fifty-ninth instruction.
- Blue arrow pointing to the sixtieth instruction.
- Blue arrow pointing to the sixty-first instruction.
- Blue arrow pointing to the sixty-second instruction.
- Blue arrow pointing to the sixty-third instruction.
- Blue arrow pointing to the sixty-fourth instruction.
- Blue arrow pointing to the sixty-fifth instruction.
- Blue arrow pointing to the sixty-sixth instruction.
- Blue arrow pointing to the sixty-seventh instruction.
- Blue arrow pointing to the sixty-eighth instruction.
- Blue arrow pointing to the sixty-ninth instruction.
- Blue arrow pointing to the seventieth instruction.
- Blue arrow pointing to the seventy-first instruction.
- Blue arrow pointing to the seventy-second instruction.
- Blue arrow pointing to the seventy-third instruction.
- Blue arrow pointing to the seventy-fourth instruction.
- Blue arrow pointing to the seventy-fifth instruction.
- Blue arrow pointing to the seventy-sixth instruction.
- Blue arrow pointing to the seventy-seventh instruction.
- Blue arrow pointing to the seventy-eighth instruction.
- Blue arrow pointing to the seventy-ninth instruction.
- Blue arrow pointing to the eightieth instruction.
- Blue arrow pointing to the eighty-first instruction.
- Blue arrow pointing to the eighty-second instruction.
- Blue arrow pointing to the eighty-third instruction.
- Blue arrow pointing to the eighty-fourth instruction.
- Blue arrow pointing to the eighty-fifth instruction.
- Blue arrow pointing to the eighty-sixth instruction.
- Blue arrow pointing to the eighty-seventh instruction.
- Blue arrow pointing to the eighty-eighth instruction.
- Blue arrow pointing to the eighty-ninth instruction.
- Blue arrow pointing to the ninetieth instruction.
- Blue arrow pointing to the ninety-first instruction.
- Blue arrow pointing to the ninety-second instruction.
- Blue arrow pointing to the ninety-third instruction.
- Blue arrow pointing to the ninety-fourth instruction.
- Blue arrow pointing to the ninety-fifth instruction.
- Blue arrow pointing to the ninety-sixth instruction.
- Blue arrow pointing to the ninety-seventh instruction.
- Blue arrow pointing to the ninety-eighth instruction.
- Blue arrow pointing to the ninety-ninth instruction.
- Blue arrow pointing to the hundredth instruction.