Test Code

```
SET R1, 0x0384          # lower 16-bit
SET R8, 0x1234          # Lower half of R8
SSET R8, 0x5678         # Upper half of R8: R8 = 0x12345678
ADDI R5, R1, 20         # R5 = R1 + 20
XOR R3, R1, R5          # R3 = R1 ^ R5
ADD R4, R8, R3          # R4 = R8+ R3
LW R1, 0(R0)            # R1 = Mem[0] = 0x00000001
LW R2, 1(R0)            # R2 = Mem[1] = 0x00000001
LW R3, 2(R0)            # R3 = Mem[2] = 0x0000000A
SUB R4, R4, R4          # R4 = 0
Loop1: ADD R4, R2, R4
SLT R6, R2, R3          # R6 = (R2 < R3) ? 1 : 0
BEQ R6, R0, done        # If R6 == 0, go to “done” label
ADD R2, R1, R2          # R2 = R1 + R2 (may be skipped)
BEQ R0, R0, Loop1             # Loop back 5 instructions
done: SW R4, 0(R0)
MUL R10, R2, R3         # R10 = R2 * R3
SRL R14, R10, R4        # R14 = R10 >> R4[4:0]
SRA R15, R10, R4        # R15 = R10 >> R4 (arithmetic)
```

```
RORI R26, R14, 5        # R26 = R14 rotate right 5
JALR R7, R0, func       # Call func, save PC+1 in R7
SET R9, 0x4545              # R9 = 0x00004545
SET R10, 0x4545             # R10 = 0x00004545
BGE R10, R9, L1         # Branch if R10 >= R9 (taken)
ANDI R23, R1, 0xFFFF        # R23 = R1 & 0x0000FFFF
L1: BEQ R0, R0, L1      # Infinite loop to terminate
func: OR R5, R2, R3       # R5 = R2 | R3
LW R1, 0(R0)            # R1 = Mem[0]
LW R2, 5(R1)            # R2 = Mem[R1 + 5]
LW R3, 6(R1)            # R3 = Mem[R1 + 6]
AND R4, R2, R3          # R4 = R2 & R3
SW R4, 0(R0)            # Mem[0] = R4
JALR R0, R7, 0          # Return to caller (JR R7)
```

Memory starts from zero address contain:

Mem[0] = 0x00000001

Mem[1] = 0x00000001

Mem[2] = 0x0000000A

Mem[60] = 0x128945AC

Mem[61] = 0x05007342