**RV DV: RISC-V Arch Test**

**RISC-V Arch Test – Task: 1**

**Following is the link to the github repository for this task:**

<https://github.com/daniyalahmed-10xe/RISC-V_Arch_Test-Task_1.git>

**Test Description:**

This test switched between different privilege modes including Machine Mode, Supervisor Mode, and User Mode. It calls a function ‘switchMode’ with an argument of either ‘0’ or ‘1’ to idnetify which mode to switch into from Supervisor or User respectively. It also defines a trap handler function ‘trapVector’ that identifies the current mode and switeches the user up one privilege level i.e. if current mode is ‘User’, after an ecall, the mode will be switched to ‘Supervisor’. The ‘mstatus’ and ‘sstatus’ bits of the ‘CSR’ register are used to verify the modes. Finally the program ends in machine mode by calling the ‘writeToHost’ function in an infinite loop. Note that since the provided linker file does not allocate any space for the stack, the ‘PROLOGUE’ and ‘EPILOGUE’ can not be used here and are commented out.

**Whats’s the Actual Output? (Screenshots Attatched at the End of File):**

According to the logfile (screenshot provided at the end) the program starts in machine mode as can be seen by the log, it then switches to supervisor mode by calling the ‘switchMode’ function with an argument of ‘0’. Then an ecall returns it back to machine mode. The ‘switchMode’ function is called again, this time with an argument of ‘1’, to change the privilege level to user mode and finally an ecall is called twice to return to machine mode before terminating the program with a ‘SUCCESS’ message. The log shows the privilege level chaning from to different modes i.e. ‘M to S’, ‘S to M’, ‘M to U’, ‘U to S’, etc.

**Following are the answers to the questions asked in this task:**

All programs start in machine mode if the privilege level is not changed, hence there is no need to explicitly start the program in machine mode as it already starts in machine mode.

**Following is the code written for this task:**

* **task1.S**

#define RVTEST\_DATA\_BEGIN \

.pushsection .tohost,"aw",@progbits; \

.align 6; .global tohost; tohost: .dword 0; .size tohost, 8; \

.align 6; .global fromhost; fromhost: .dword 0; .size fromhost, 8; \

.popsection; \

.align 4; .global begin\_signature; begin\_signature:

#define RVTEST\_CODE\_BEGIN \

.section .text.init; \

.align 6; \

.global \_start; \

\_start: \

j main; \

RVTEST\_CODE\_BEGIN

main:

# PROLOGUE

#addi sp, sp, -16

#sw ra, 0(sp)

#sw fp, 4(sp)

# END PROLOGUE

la t0, trapVector # Setup Trap Vector

csrw mtvec, t0

csrr a0, mstatus # Check Machine Mode Status

li a0, 0 # Switch to Supervisor Mode

call switchMode

csrr a0, sstatus # Check Supervisor Mode Status

ecall # Call Trap Vector to Increment Mode to Machine (Supervisor Mode --Trap--> Machine Mode)

csrr a0, mstatus # Check Machine Mode Status

li a0, 1 # Switch to User Mode

call switchMode

ecall # Call Trap Vector twice to Increment Mode twice to Machine (User Mode --Trap--> Supervisor Mode --Trap--> Machine Mode)

ecall

csrr a0, mstatus # Check Machine Mode Status

# EPILOGUE

#lw fp, 4(sp)

#lw ra, 0(sp)

#addi sp, sp, 16

# END EPILOGUE

end\_main: call writeToHost

writeToHost:

# PROLOGUE

#addi sp, sp, -16

#sw ra, 0(sp)

#sw fp, 4(sp)

# END PROLOGUE

li gp, 1

sw gp, tohost, t5

# EPILOGUE

#lw fp, 4(sp)

#lw ra, 0(sp)

#addi sp, sp, 16

# END EPILOGUE

end\_writeToHost: call writeToHost

switchMode:

# PROLOGUE

#addi sp, sp, -16

#sw ra, 0(sp)

#sw fp, 4(sp)

# END PROLOGUE

mv t0, a0

csrr t1, mstatus

li t6, 0x1800

not t6, t6

and t1, t1, t6

if1: bnez t0, else1

li t6, 0x0800

j end\_if1

else1:

li t6, 0x0000

j end\_if1

end\_if1:

or t1, t1, t6

csrw mstatus, t1

# EPILOGUE

#lw fp, 4(sp)

#lw ra, 0(sp)

#addi sp, sp, 16

csrw mepc, ra

# END EPILOGUE

end\_switchMode: mret

trapVector:

# PROLOGUE

#addi sp, sp, -32

#sw ra, 0(sp)

#sw fp, 4(sp)

#sw s0, 8(sp)

#sw s1, 12(sp)

#sw s2, 16(sp)

# END PROLOGUE

csrr s0, mcause

li s1, 9

li s2, 8

if2: bne s0, s1, elseif2

li s1, 0x1800

j end\_if2

elseif2: bne s0, s2, end\_if2

li s1, 0x0800

j end\_if2

end\_if2:

csrr s0, mstatus

or s0, s0, s1

csrw mstatus, s0

# EPILOGUE

#lw s2, 16(sp)

#lw s1, 12(sp)

#lw s0, 8(sp)

#lw fp, 4(sp)

#lw ra, 0(sp)

#addi sp, sp, 32

addi ra, ra, 4

csrw mepc, ra

# END EPILOGUE

end\_trapVector: mret

.data

base:

.word 0xcafebeef

RVTEST\_DATA\_BEGIN

* **link.ld**

OUTPUT\_ARCH( "riscv" )

ENTRY(\_start)

SECTIONS

{

. = 0x80000000;

.text.init : { \*(.text.init) }

. = ALIGN(0x1000);

.tohost : { \*(.tohost) }

. = ALIGN(0x1000);

.text : { \*(.text) }

. = ALIGN(0x1000);

.data : { \*(.data) }

.bss : { \*(.bss) }

\_end = .;

}

**Following are the screenshots of the output for this task:**













