 # Author : Muhammad Imran
 # Date: 12-02-2026
 # Module : RISCV Arch Test
 # Section: RISCV Arch Test
 # Task Name: Task 2

Github link: RISCV Arch Test- Task **2**

Task Description:

This test creates and configures two memory regions with different constraints and verifies them. One follows the TOR addressing method with **'X'** permissions and other follows the NAPOT addressing method with **'R'** permissions. PMP checks are also applied on the existing memory regions created in the previous tasks. Load , Stores and Executes accesses are tested from the lower privilege modes on these configured memory regions and verified that configured PMP restrictions are working correctly. Each access gives the appropriate access fault. At the end the same restrictions are also set for Machine mode setting the lock bit of each PMP region and appropriate access faults are again tested and verified for Machine mode. Trap handler for corresponding access faults is also added to the trap vector to address the access faults.

Memory Regions:

The linker script defines multiple memory sections. Two new PMP test regions are created:

TOR Section

Label: .tor_section
Start: _tor_start (4KB aligned)
End: _tor_end (4KB aligned)
Contains a single nop instruction and a ret instruction.

NAPOT Section

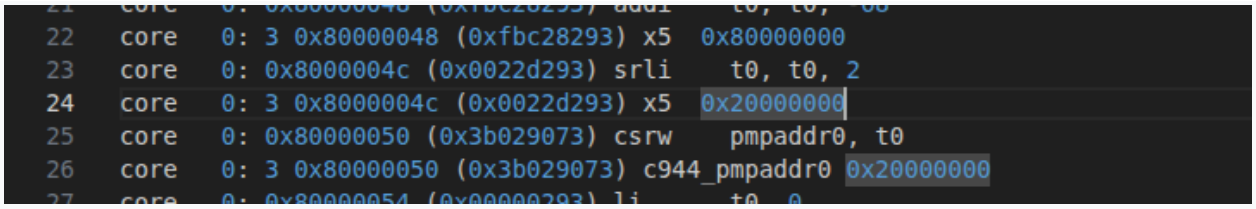
Label: .napot_section
4KB aligned
Contains a single word .word 0x0.

Other memory sections exist as usual among them stack is not configured for PMP PMP checks:

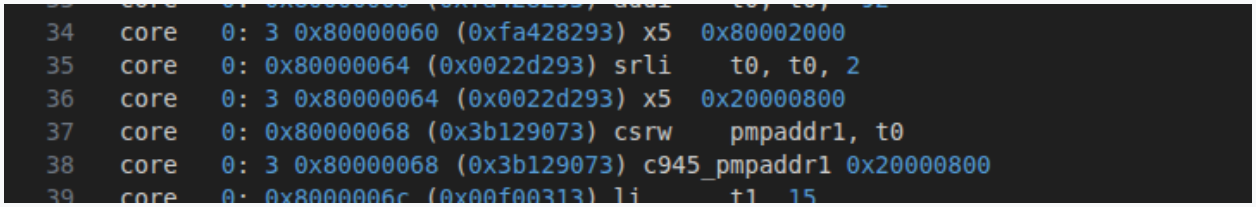
PMP_configurations:

The **_start** function calls **pmp_config** to initialize PMP entries in which at first existing memory region is configured as follows:

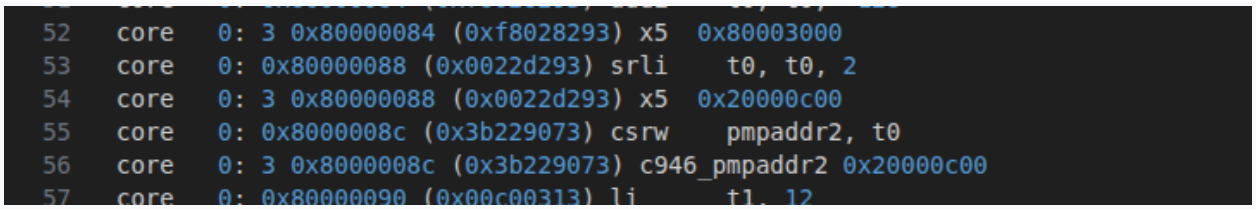
- **Region 0 (pmpaddr0):** .**text.init** configured as **TOR**, initially OFF. for this region the lower bound is addresses towards 0 and upper bound is the address of the **.text.init** section. It is shifted 2 bit right because pmpaddr reg stores word aligned address.



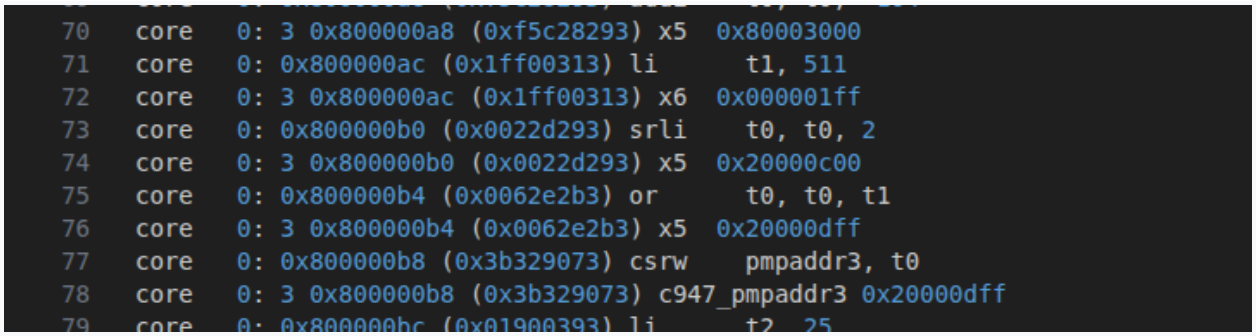
- **Region 1 (pmpaddr1):** Mapped from **_end of .bss** to top of memory as **TOR** with all permissions.



- **Region 2 (pmpaddr2): TOR SECTION** Configured as TOR region with execute-only permission



- **Region 2 (pmpaddr3):** NAPOT Section Configured as NAPOT region with read-only permission



Actual Output:

After switching to the supervisor mode

Performing RWX in TOR Region

READ/LOAD access fault

```
150   core   0: 1 0x8000010c (0xef828293) x5  0x80002000
151   core   0: 0x80000110 (0x0002a303) lw      t1, 0(t0)
152   core   0: exception trap_load_access_fault, epc 0x80000110
153   core   0:           tval 0x80002000
154   core   0: >>>>  trap_vector
155   core   0: 0x80000014 (0x342022f3) csrr    t0, mcause
156   core   0: 3 0x80000014 (0x342022f3) x5  0x00000005
157   core   0: 0x80000018 (0x00800313) li      t1, 8
```

WRITE/STORE access FAULT

```
186   core   0: 1 0x8000011c (0xee828293) x5  0x80002000
187   core   0: 0x80000120 (0x0072a223) sw      t2, 4(t0)
188   core   0: exception trap_store_access_fault, epc 0x80000120
189   core   0:           tval 0x80002004
190   core   0: >>>>  trap_vector
191   core   0: 0x80000014 (0x342022f3) csrr    t0, mcause
192   core   0: 3 0x80000014 (0x342022f3) x5  0x00000007
193   core   0: 0x80000018 (0x00800313) li      t1, 8
```

NO INSTRUCTION Access FAULT

```
core   0: 0x80000124 (0x6dd010ef) jal     pc + 0x1edc
core   0: 1 0x80000124 (0x6dd010ef) x1  0x80000128
core   0: 0x80002000 (0x00000013) nop
core   0: 1 0x80002000 (0x00000013)
core   0: 0x80002004 (0x00008067) ret
core   0: 1 0x80002004 (0x00008067)
core   0: 0x80000128 (0x00003297) auipc   t0, 0x3
core   0: 1 0x80000128 (0x00003297) x5  0x80003128
```

Performing RWX in NATOP Region

NO READ/LOAD access fault

```
core   0: 0x80000128 (0x00003297) auipc   t0, 0x3
core   0: 1 0x80000128 (0x00003297) x5  0x80003128
core   0: 0x8000012c (0xed828293) addi    t0, t0, -296
core   0: 1 0x8000012c (0xed828293) x5  0x80003000
core   0: 0x80000130 (0x0002a303) lw      t1, 0(t0)
core   0: 1 0x80000130 (0x0002a303) x6  0x00000000 mem 0x80003000
core   0: 0x80000134 (0x00500393) li      t2, 5
core   0: 1 0x80000134 (0x00500393) x7  0x00000005
```

WRITE/STORE access FAULT

```
core   0: 1 0x8000013c (0xec828293) x5  0x80003000
core   0: 0x80000140 (0x0072a223) sw      t2, 4(t0)
core   0: exception trap_store_access_fault, epc 0x80000140
core   0:           tval 0x80003004
core   0: >>>>  trap_vector
core   0: 0x80000014 (0x342022f3) csrr    t0, mcause
core   0: 3 0x80000014 (0x342022f3) x5  0x00000007
```

INSTRUCTION Access FAULT

```
core   0: 0x80000144 (0x6bd020ef) jal     pc + 0x2ebc
core   0: 1 0x80000144 (0x6bd020ef) x1  0x80000148
core   0: exception trap_instruction_access_fault, epc 0x80003000
core   0:           tval 0x80003000
core   0: >>>>  trap_vector
core   0: 0x80000014 (0x342022f3) csrr    t0, mcause
core   0: 3 0x80000014 (0x342022f3) x5  0x00000001
```

Going back to Machine mode using ECALL in S-mode

```
core   0: 3 0x80000224 (0x30200073) c768_mstatus 0x00000080 c784_mstatush 0x00000000
core   0: 0x80000148 (0x00000073) ecall
core   0: exception trap_supervisor_ecall, epc 0x80000148
core   0: >>>>  trap_vector
core   0: 0x80000014 (0x342022f3) csrr    t0, mcause
core   0: 3 0x80000014 (0x342022f3) x5  0x00000009
core   0: 0x80000018 (0x00800313) li      t1, 8
```

Switching back to Machine mode, the same PMP restrictions apply. Appropriate access faults are generated when attempting unauthorized access.

Trap occured in machine mode too

```
core   0: 0x80000150 (0xeb428293) addi     t0, t0, -332
core   0: 3 0x80000150 (0xeb428293) x5  0x80002000
core   0: 0x80000154 (0x0002a303) lw       t1, 0(t0)
core   0: exception trap_load_access_fault, epc 0x80000154
core   0:           tval 0x80002000
core   0: >>>>  trap_vector
core   0: 0x80000014 (0x342022f3) csrr     t0, mcause
core   0: 3 0x80000014 (0x342022f3) x5  0x00000005
```

```
core   0: 3 0x80000160 (0xea428293) x5  0x80002000
core   0: 0x80000164 (0x0072a223) sw       t2, 4(t0)
core   0: exception trap_store_access_fault, epc 0x80000164
core   0:           tval 0x80002004
core   0: >>>>  trap_vector
core   0: 0x80000014 (0x342022f3) csrr     t0, mcause
core   0: 3 0x80000014 (0x342022f3) x5  0x00000007
core   0: 0x80000018 (0x00800313) li       t1, 8
```

```
12   core   0: 3 0x8000023c (0x30200073) c768_mstatus 0x00000080 c784_mstatush 0x00000
13   core   0: 0x80000168 (0x699010ef) jal      pc + 0x1e98
14   core   0: 3 0x80000168 (0x699010ef) x1  0x8000016c
15   core   0: 0x80002000 (0x00000013) nop
16   core   0: 3 0x80002000 (0x00000013)
17   core   0: 0x80002004 (0x00008067) ret
18   core   0: 3 0x80002004 (0x00008067)
19   core   0: 0x8000016c (0x0f80006f) j        pc + 0xf8
```

Reference:

Physical Memory Protection: Privileged Architecture v1.12, Sec 3.7