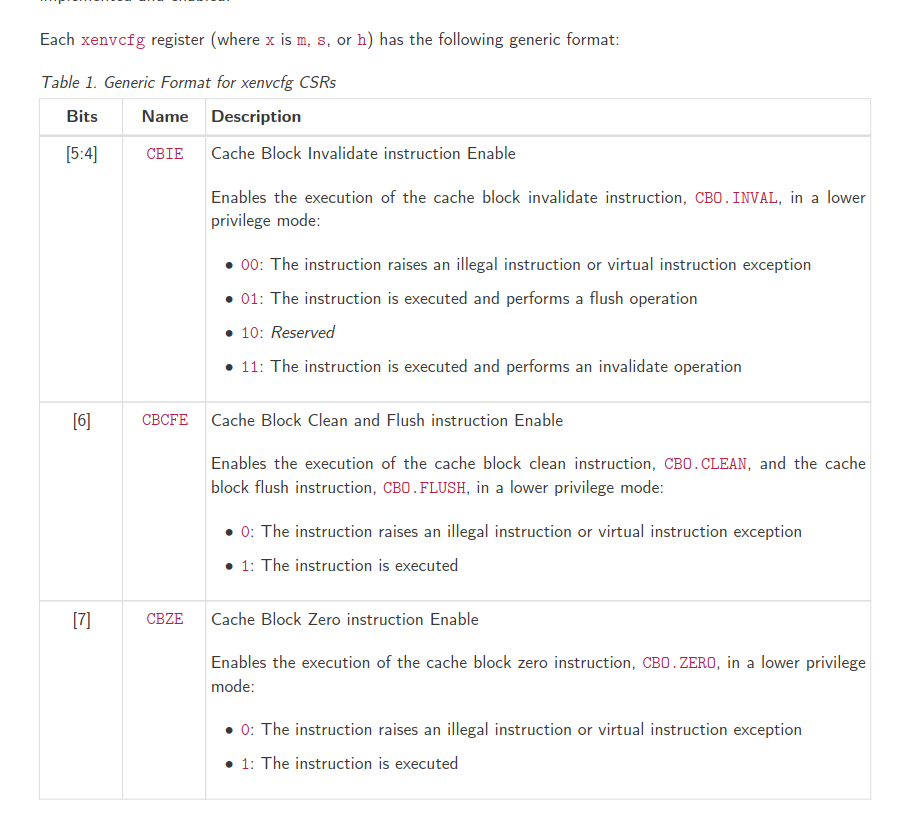
CMO-Testpln - zicbom and zicbop

1)CSR registers that control CMOs:

menvcfg

senvcfg

henvcfg



Test if statuses of these registers are correct, corresponding to the privilege modes

**if (((priv\_mode != M) && (menvcfg.CBIE == 00)) ||**

**((priv\_mode == U) && (senvcfg.CBIE == 00)))**

**{**

**<raise illegal instruction exception>**

**}**

**// virtual instruction exceptions**

**else if (((priv\_mode == VS) && (henvcfg.CBIE == 00)) ||**

**((priv\_mode == VU) && ((henvcfg.CBIE == 00) || (senvcfg.CBIE == 00))))**

**{**

**<raise virtual instruction exception>**

**}**

**// execute instruction**

**else**

**{**

**if (((priv\_mode != M) && (menvcfg.CBIE == 01)) ||**

**((priv\_mode == U) && (senvcfg.CBIE == 01)) ||**

**((priv\_mode == VS) && (henvcfg.CBIE == 01)) ||**

**((priv\_mode == VU) && ((henvcfg.CBIE == 01) || (senvcfg.CBIE == 01))))**

**{**

**<execute CBO.INVAL and perform flush operation>**

**}**

**else**

**{**

**<execute CBO.INVAL and perform invalidate operation>**

**}**

**}**

**Clean and flush:**

**if (((priv\_mode != M) && !menvcfg.CBCFE) ||**

**((priv\_mode == U) && !senvcfg.CBCFE))**

**{**

**<raise illegal instruction exception>**

**}**

**// virtual instruction exceptions**

**else if (((priv\_mode == VS) && !henvcfg.CBCFE) ||**

**((priv\_mode == VU) && !(henvcfg.CBCFE && senvcfg.CBCFE)))**

**{**

**<raise virtual instruction exception>**

**}**

**// execute instruction**

**else**

**{**

**<execute CBO.CLEAN or CBO.FLUSH>**

**}**

**Page faults and other type of faults w/cache:**

**An invalidate operation may change the set of values that can be returned by a load. In particular, an additional**

**condition is added to the Load Value Axiom:**

**• If an invalidate operation i precedes a load r and operates on a byte x returned by r, and no store to x**

**appears between i and r in program order or in the global memory order, then r returns any of the following**

**values for x:**

**1. If no clean or flush operations on x precede i in the global memory order, either the initial value of x or**

**the value of any store to x that precedes i**

**2. If no store to x precedes a clean or flush operation on x in the global memory order and if the clean or**

**flush operation on x precedes i in the global memory order, either the initial value of x or the value of**

**any store to x that precedes i**

**3. If a store to x precedes a clean or flush operation on x in the global memory order and if the clean or**

**flush operation on x precedes i in the global memory order, either the value of the latest store to x that**

**precedes the latest clean or flush operation on x or the value of any store to x that both precedes i and**

**succeeds the latest clean or flush operation on x that precedes i**

**4. The value of any store to x by a non-coherent agent regardless of the above conditions**

**Privilege mode check for all instructions.If fence -> monitor mstatus continuously**

**LR/SC loop:**

**If some other core executes CMO on LR/SC -> CMO will fail**

**Discovery:**

**Cbo.inval -> store address into rs1**

**Every time load takes place -> comapre load address to inval address**

**If same then page fault**