**6.3 - a)**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CT | | | | | | | | CI | | CO | |

# of lines per set, E = 2; Block Size in Bytes, B = 4; # of Sets, S = 4

# of memory bytes = m = 2Max = 12 slots; Set index bits (CI), s = log2(S) = 2;

Offset (CO), b = log2(B) = 2; Tag (CT), t = m – (s + b) = 8;

**6.3 b)**

|  |  |  |  |
| --- | --- | --- | --- |
| Operation | Address | Hit? | Read Value |
| Read | 0x834 | Miss | None |
| Write | 0x836 | Hit | Unknown |
| Read | 0xFFD | Hit | 0xC0 |

0x834 = 1000 0011 0100 => 10000011 | 01 | 00 => Index = 01 = 1, Offset = 00 = 0,

Tag = 10000011 = 83 🡪 Miss, the properties of the address are now placed into the cache

0x836 = 1000 0011 0110 => 10000011 | 01 | 10 => Index = 01 = 1, Offset = 10 = 2,

Tag = 10000011 = 83 🡪 Hit, but Unknown value read, since that part of the cache is now whatever was previously placed into the cache.

0xFFD = 1111 1111 1101 => 11111111 | 11 | 01 => Index = 11 = 3, Offset = 01 = 1,

Tag = 11111111 = FF 🡪 Hit, 0xC0

**8.23)**

The bug occurs within the for loop. At the start of the for loop, the child sends the first SIGUSR2 signal to the parent, which the parent then uses the handler to increment the counter once. However, the parent then has to sweep the child. This leads to an issue where the child then sends another signal to the parent, but the signal is not receive, due to the fact that signals can’t be queued. The parent sweep all the children, then at the end of the loop receives another signal to increment the counter, thus explaining why the counter only increments up to two.

**9.11 - a)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |

Virtual Address = 0x027c = 0000 0010 0111 1100 => 00 0010 0111 1100

**9.11 - b)**

Address Translation, View Page 796 for the tables (Section 9.6.4). Convert First Table Values into hex:

Bit Blocks 6 to 13 (from a) are the VPN: 00001001 = 0000 1001 = 0x09

Bit Blocks 6 to 7 are the TLBI: 01 = 1

Bit Blocks 8 to 13 are the TLBT: 000010 = 00 0010 = 2

Looking at the table a, set 1, tag 2 has no valid value, so it’s a miss

Looking at page table (b), value 9 has a PPN of 17, which means there is no page fault.

|  |  |
| --- | --- |
| Parameter | Value |
| VPN | 0x9 |
| TLB index | 0x1 |
| TLB tag | 0x2 |
| TLB hit? (Y/N) | N |
| Page fault? (Y/N) | N |
| PPN | 0x17 |

**9.11 - c)**

Physical Address Format

PPN = 17 => 0001 0111 => Goes into Bits 6 to 11

Bits 0 to 5 should be the VPO (Offset) = 111100

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |

**9.11 - d)**

Physical Memory Reference, View Diagram C

Bits 0 to 1 are the Offset => 00 = 0x0; Bits 2 to 5 are the Index => 1111 = 0xF; Tag is PPN = 0x17

There is a miss, since there isn’t a tag 17 on table C; which means no byte is returned

|  |  |
| --- | --- |
| Parameter | Value |
| Byte Offset | 0x0 |
| Cache Index | 0xF |
| Cache Tag | 0x17 |
| Cache Hit? (Y/N) | N |
| Cache Byte Returned | None |