Computer Architecture Homework 8

Spring 2019, May

1 Virtual memory

1. What are three specific benefits of using virtual memory?

1. Add disks to hierarchy
2. Simplify memory structure for each program
3. Protection between processes

2. What should happen to the TLB when a new value is loaded into the page table address register?

Valid bits are all set to 0 (false), since the new page table are supposed to have a different mapping.

3. A processor has 16-bit addresses, 256 byte pages, and an 8-entry fully associative TLB with LRU replacement (the LRU field is 3 bits and encodes the order in which pages were accessed, 0 being the most recent). At some time instant, the TLB for the current process is the initial state given in the table below. Assume that all current page table entries are in the initial TLB. Assume also that all pages can be read from and written to. Fill in the final state of the TLB according to the access pattern below

Free physical pages: 0x17, 0x18, 0x19

Access pattern:

Read	0x11f0
Write	0x1301
Write	0x20ae
Write	0x2332
Read	0x20ff
Write	0x3415

Initial TLB:

VPN	PPN	Valid	Dirty	LRU
0x01	0x11	1	1	0
0x00	0x00	0	0	7
0x10	0x13	1	1	1
0x20	0x12	1	0	5
0x00	0x00	0	0	7
0x11	0x14	1	0	4
0xac	0x15	1	1	2
0xff	0x16	1	0	3

Final TLB:

VPN	PPN	Valid	Dirty	LRU
0x01	0x11	1	1	5
0x13	0x17	1	1	3
0x10	0x13	1	1	6
0x20	0x12	1	1	1
0x23	0x18	1	1	2
0x11	0x14	1	0	4
0xac	0x15	1	1	7
0x34	0x19	1	1	0

2 Hamming ECC

Recall the basic structure of a Hamming code. Given bits $1, \ldots, m$ the bit at position 2^n is parity for all the bits with a 1 in position n. For example, the first bit is chosen such that the sum of all odd-numbered bits is even.

Bit	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Data	P1	P2	D1	P4	D2	D3	D4	P8	D5	D6	D7	D8	D9	D10	D11
P1	X		X		X		X		X		X		X		X
P2		X	X			X	X			X	X			X	X
P4				X	X	X	X					X	X	X	X
P8								X	X	X	X	X	X	X	X

1. How many bits do we need to add to 0011_2 to allow single error correction?

3 bits

2. Which locations in 0011_2 would parity bits be included? (Use P for parity bits)

PP0P011

3. Which bits does each parity bit cover in 0011_2?

P_1: 1 3 5 7 P_2: 2 3 6 7

P_4: 4 5 6 7

4. Write the completed coded representation for 0011_2 to enable single error correction.

1 0 0 0 0 1 1

5. How can we enable an additional double error detection on top of this?

Add an additional parity bit of all bits

6. Find the original bits given the following Single-Error Correction(SEC) Hamming Code: 0110111_2

P_1, P_4 error $\Rightarrow$ Corrected: 0110011_2

result: 1011_2

7. Find the original bits given the following SEC Hamming Code: 1001000_2

P_1, P_4 error $\Rightarrow$ Corrected: 1001100_2

result: 0100_2

8. Find the original bits given the following SEC Hamming Code: 0100110100001110_2

P_2 P_8 error

$\Rightarrow 0110010011 0_2$

3 Availability

1. In this example, a Warehouse-Scale Computers (WSC) has 55,000 servers, and each server has four disks whose annual failure rate is 4%. How many disks will fail per hour?

# of failed disks / (55000x4) = 4 %

$\Rightarrow$ # of failed disks = 8800

# of failed disks per hour = 8800/8760 $\approx$ 1.00

2. What is the availability of the system if it does not tolerate the failure? Assume that the time to repair a disk is 30 minutes.

MTTF = 1 h

MTTR = 0.5h availability $= \frac{1}{1+0.5} \approx 66.7\%$

3. Which of the following will decrease availability? **Circle** them.

(A.) decrease MTTF

B. increase MTTF

C. decrease MTBF

(D.) increase MTBF

E. decrease MTTR

(F.) increase MTTR

4 Memory Mapped I/O

Certain memory addresses correspond to registers in I/O devices and not normal memory

0xFFFF0000 – Receiver Control:
LSB is the ready bit, there may be other bits set that we don't need right now

0xFFFF0004 – Receiver Data:
Received data stored at lowest byte.

0xFFFF0008 – Transmitter Control:
LSB is the ready bit, there may be other bit set that we don't need right now.

0xFFFF000C – Transmitter Data:
Transmitted data stored at lowest byte

Finish the RISC-V code to read a byte from the receiver and immediately send it to the transmitter.

```
1
2  la t0, 0xFFFF0000
3  receive_wait:
4  lw t1 0(t0)
5
6  andi t4, t1, 0x00000001     # poll on ready of receiver
7
8  beq t4, zero, receive_wait
9  lb t2 4(t0)                 # load data
10 transmit_wait:
11 lw t1 8(t0)
12
13 andi t4, t1, 0x00000001     # poll on ready of transmitter
14
15 beq t4, zero, transmit_wait
16
17 sb t2, 12(t0)               # write to transmitter
```