ACA Assignment # 3

Virtualization is a technique to make things virtual i.e software dependent or virtual of the hardware components. This feature helps in deploying the same hardware in various platforms without the need to alter the pre-existing software. The virtual machine monitors(VMM) were proved to be advantages during the time when the hardware was expensive and also in the applications where the available hardware was scarce.

In 1980 and 1990’s when the hardware had flourished and there was a significant drop in the prices of the hardware systems saw a declining trend in the VMM. Again, in the 2005 the VMM were revived when the researches thought they could be used as the solution to the various hardware problems. The VMM were again deployed to address the various issues such as security and reliability. The most significant feature that VMM incorporated was, the systems have become crash proof. The features that VMM were to incorporate are, they should be able to offer complete encapsulation of the VM hardware, compatible, simple, reliable and were be able to revert back to the previous state if the hardware was proving to be catastrophic.

CPU virtualization was one of the issue that had raised due to the implementation of the VMM. The CPU architecture in general should be able to support virtualization by using direct extension, this required the VMM to be run on the kernel mode and this meant that CPU had to be controlled by the VMM. The processors that were designed did not support virtualization. The technique which was incorporated to overcome this backdrop was “Paravirtualization” along with direct execution with fast binary translation. The paravirtualization used the technique of replacing the non-virtualized part by a virtualizable part with more efficient equivalents. DISCO was a nonvirtualizable MIPs architecture which had installed this feature. The setback in the paravirtualization was incompatibility. Research has been going on by the INTEL and the AMD to overcome this drawback. The main idea behind the research work is instead of making the existing systems virtualizable they planned to make add a mode in the VMM that safely and transparently uses direct execution that supported running of the virtual machines.

Memory virtualization was implemented by make the VMM have a copy of the virtual machine memory management data structure. The data structure had the information of which parts of the memory were virtualizable. The main challenge to be faced was the lack of the memory information. To address this issue they used a balloon process inside the virtual machine OS which could communicate with the operating system of that machine. Storing excess copies of the VMM for multiple instances was another issue with memory management. This issue was resolved by storing a content-based page i.e VMM kept a record of the content pages by comparing if they were identical. The future support for this issue was to address the issue caused by the overhead due to the frequent changes in the page-tables.

Input- Output virtualization another issue caused by the virtualization of the mainframes, because the communication here was through a separate channel processor. This resulted in a low virtualization overhead of the I/O. Current computing environments with their deeper I/O made virtualization more difficult. Instead of communicating with the device using the traps in VMM, the VMM could directly communicate. Another issue with the I/O virtualization was compatibility This issue was addressed by using a host architecture which used the virtualization layer that had device drivers for the host operating system The virtualization can be used in the future to point towards the hardware support for high performance input output device virtualization.

In the future the VMM can be used on the server side, i.e in the data centres, the administrators would be able to manage and monitor a lot of data from a single device. VMM could also change the approach of people have towards computers. The VMM could be used for the security improvements by providing greater security and for software distribution to distribute entire virtual machines containing complex software environments.

----------------------------------------------------------------------------------------------------------------

B)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameters | DDR4 SDRAM | ZRAM | GDDR5 SDRAM | MRAM |
| Speed | 2133MHz | 120MHz | 1683MHz | 250Mhz |
| Size | 8GB | 150MB | 6GB | 512MB |
| Access Time | 10ns | 3ns | 2.5ns | 2ns |
| Availability | High Availability  (RAM) | Available (Linux Kernel) | Available (Graphics Processor RAM) | Hard Disk |
| Cost | $90 | $2 | $500-$900 | $64 |

----------------------------------------------------------------------------------------------------------------

Exercises

1a) The number of bits required to store the tag in the cache is 24 bits

1 b)

16 bytes per block

Offset bits= 4 bit

No. of bits =16 2 way cache = 4 bits

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Address | Tag | Index | Offset | Hit/Miss? | Type of Miss |
| 0x100 | 0001 | 0000 | 0000 | Miss | Compulsory |
| 0x104 | 0001 | 0000 | 0100 | Miss | Compulsory |
| 0x108 | 0001 | 0000 | 1000 | Hit | - |
| 0x200 | 0010 | 0000 | 0000 | Miss | Conflict |
| 0x204 | 0010 | 0000 | 0100 | Miss | Conflict |
| 0x410 | 0100 | 0001 | 0000 | Miss | Compulsory |
| 0x100 | 0001 | 0000 | 0000 | Miss | Conflict |
| 0x108 | 0001 | 0000 | 1000 | Miss | Conflict |
| 0x40C | 0100 | 0000 | 1100 | Miss | Conflict |
| 0x408 | 0100 | 0000 | 1000 | Miss | Conflict |
| 0x300 | 0011 | 0000 | 0000 | Miss | Conflict |
| 0x284 | 0010 | 1000 | 0100 | Miss | Compulsory |
| 0x280 | 0010 | 1000 | 0000 | Miss | Compulsory |
| 0x304 | 0011 | 0000 | 0100 | Hit | - |

1 c) miss rate (12/14) \*100 = 85.7 %s

1 d) the number of blocks occupied are 5

1 e) If the same cache is direct mapped the tag field is 23 bits

----------------------------------------------------------------------------------------------------------------

2 a)

Physical memory is 1KB and 64 B pages,

Therefore, there are 256 physical pages

Physical Frame Number = 4 bits

Each entry has a PFN and 4 protection bits, which is 12 bits which is 2 bytes.

A single – level page table has one entry per virtual page

With a 16- bit virtual address space, there is a 64KB of memory. Since each page is 64 bytes, that is there are 512 pages. At 2 bytes per page, the page table is 512bytes in size.

2 b)

----------------------------------------------------------------------------------------------------------------

3 a) Page table = 54-14 = 40 bits

Number of entries = (2^40)

PTE = 4 bytes

* Total size = (2^42) = 4TB

b) PTE size = 4 bytes (32 bits)

page size = 16K bytes

therefore the number of bits to hold the page offset is = (log(16\*(2^10)\*(2^3)) i.e 17 bits

number of bits for protection = 32-8 = 24 bits

the largest physical memory size is ((2^41)/(2^3)) = 2^38

c) application page table has an entry for every physical page

therefore (2^24)\*4 bytes

* Remaining physical space to process is = (2^38) – (2^26) bytes

----------------------------------------------------------------------------------------------------------------