Assignment 1

Aim:
a)Nachos Implementation of Fork,Exec and SysStats System Calls.
b)Use the above system calls to implement a Multiprogrammed Execution Environment.

Changes in Nachos:
Userprog/syscall.h:
Declared 3 New Syscall

- #define SC_Fork
- #define SC_Exec
- #define SC_SysStats

and their corresponding function Declaration.

- int Fork();
- SpaceId Exec(char* exec_name);
- int SysStats();

test/Start.s

Added corresponding Assembly code.

Memory Management:
Default Nachos executes user programs only using a simple memory model and only one program can run at a time. We altered address space creation in such way so that multiprogramming is supported, and several user programs can reside in the provided memory.
So we divided whole Memory in 4 Slots of 32 pages each. When even Ever a new process comes,if free memory slot is available then that slot is given to it.
userprog/adrrspace.h: An extra data entry "slot" is added to Addrspace Class.
Macros for NumSlot and NumSlotPages is added

```
#define NumSlot        4
#define SlotPages      32
```

- Addspace.cc: A global array int SlotArray[NumSlot] representing current memory allocation is declared.
- In AddrSpace class constructor code for "finding empty slot and Assign or throw error" is added.
- A pageTable having 32 virtual pages is assigned to this AddrSpace. (The default pagetable of nachos was 128 virtual pages)
- A mapping form virtual page number to physical page number is added.
- In destructor code we deleted pageTable and marked the flag of that memory slot as free.

Frok Implementation.
exception.cc

We wrote syscall handler for new system call created in expection.cc

- A new AddrSpace and a Thread is created for new process to be forked.
- Newly created AddrSpace is allocated to this Thread.

- A new Function Copy_Parent_Addrspace(Thread* ParentThread,Thread* Child) is added to copy Addrspace of parent.
- In this function content of memory slot assigned to parent is copied to memory slot assigned to child.
- userRegisters is saved to update the register.
- A new Function void Copy_Register(Thread* ParentThread, Thread* Child) is added to copy registers of parent to child.
- userRegister and machineState is copied.
- To allocate Stack and execute this newly thread the thread::Fork function is invoked.
- To return parent 1 on success 1 is written in register 2.
- Incremented previous current and next Program Counter.

Exec Implementation:

- As the pointer of the process to be executed is in kernel memory,but SC_Exec the pointer is inside user memory so we have to use machine->ReadMemory(...) to read the location the location that is saved in register in 4
- using while loop filename is read in “filename array” which can take string of maximum 100 length character.
- Cleaned out AddrSpace memory(Zeroed out).
- Code of file name is Loaded in this memory using AddrSpace::Load()

Note: In load while loading code into addrspace virtual addrspace is translated into Physcial Addrspace using a new function AddrSpace::VirtualToPhy(unsigned int virtAddr).

- Thread is executed is executed using AddrSpace::Execute()

Implementation Of SysStates:

- .New Structure SysInfo and ProcInfo is defined.
- New Function Info is defined to get the info and print it.
- Information such as Total nachos running time (totalTicks),Time Spent Idle(idleTicks),Time spending in executing system code (systemTicks) is founded using “stats” object defined in stats.h
- Total Number of process is total number of process in ready list + Running process.
- From linked list temp individual process information is founded.

Implementation of Exit:

- A new syscall is added which find new thread to run using scheduler::FindNextToRun()
- If there is no thread in ready list then calls SysHalt.

Assignment1.c

In the program a list of job is made and while list is not empty we takes out each file name and for each file we forks a new child process and the execute the file using that child.