The MU0 is a 16 bit architecture, what make us these 16 of the instruction?

1

In the MU0, what are the two programmer visible registers?

2

How many bits long is the program counter and what does it store?

3

How many bits is the accumulator and what does it store?

4

What happens when 'LDA s' is run?

5

What happens when 'STA s' is run?

6

What happens when 'ADD s' is run?

7

What happens when 'SUB s' is run?

8

4 bits for an operation and 12 for the address.

1

The program counter and the accumulator.

2

12 bits, stores the address in memory of the next instruction to be executed.

3

16 bits, stores the result of the last arithmetic operation.

4

$ACC = [s]$

5

$[s] = ACC$

6

$ACC \mathrel{+}= [s]$

7

$ACC \mathrel{-}= [s]$

8

What happens when 'JMP s' is run?

9

What happens when 'JGE s' is run?

10

What happens when 'JNE s' is run?

11

Briefly explain what the fetch-execute cycle is.

12

What three steps occur during the fetch phase?

13

In order for every execution to run smoothly what needs to be worked out?

14

What control signals do all registers need?

15

What control signal does a multiplexer need?

16

PC = s

9

if ACC ¿= 0 then PC = s

10

if ACC != 0 then PC = s

11

The next instruction is fetched from memory (at the address pointed to by the PC), then the instruction is executed.

12

1. *Read the data stored in the address that the PC is pointing to.*
2. *Save result of read in the instruction register*
3. *Increment the PC.*

13

1. *The critical path for each operation.*
2. *The time allowed for signals to propagate through even the longest critical path.*

14

An enable signal

15

A signal to select an input

16

What control signals does the memory need?

17

Which 3 signals control the ALU?

18

What two jobs do operating systems do in general?

19

What is a process?

20

What is the address space?

21

What is a thread?

22

What is multi-threading?

23

How do we make programs think they have sole use of memory?

24

add, sub & byp

18

Ren (read enable) and Wen (write enable)

17

A program in execution, the thread + address space.

20

1. *Manage the resources of the system.*
2. *Abstract the implementation of the system from the running programs.*

19

A sequence of instructions that are obeyed.

22

All memory locations the process can use.

21

*Use **relocation**, where we swap a program out of memory and later swap it back in.*

24

This is where we have multiple threads within the same process

23

What are the two most common modes of operation?

25

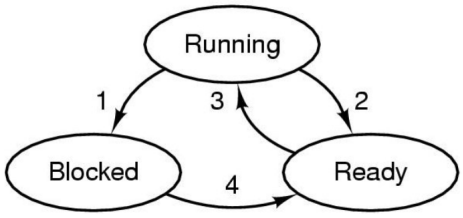
What are the three different approaches to engineering an OS?

26

What are the three process states?

27

In the diagram, what is happening at each stage?



28

What is a PCB table?

29

In scheduling, what do the following mean?

1. *CPU burst*
2. *I/O burst*
3. *CPU bound*
4. *I/O bound*

30

What is a processes turnaround time?

31

What is a processes waiting time?

32

Monolithic, layered and micro-kernels.

26

User and system

25

1. *Process needs to wait for I/O or event.*
2. *Process forcibly preempted -* ***interrupt / relinquish CPU / time-slice expired****.*
3. *Scheduler selects process to run.*
4. *I/O or event occurs.*

28

Running, ready, blocked

27

1. *Process executing on CPU*
2. *Process blocked, waiting for I/O*
3. *Long CPU bursts*
4. *Short CPU bursts*

30

Process control block, it contains all of the information needed about processes.

29

The time that the process waits to run.

32

The time from a process being submitted to it getting completed.

31

Briefly explain the first come first served scheduling algorithm.

33

Briefly explain the shortest remaining time first scheduling algorithm.

34

What is meant by pre-emptive scheduling?

35

What is meant by non-pre-emptive scheduling?

36

What is the fixed time amount called in non-pre-emptive processing?

37

What is process starvation?

38

In scheduling, what are static priorities?

39

In scheduling, what are dynamic priorities?

40

For each newly ready process, if CPU-burst is less than the time to complete the running process then context-switch and run the new process.

34

The first process in the ready state gets CPU time first. Once it is blocked or complete, the next process in the queue is run. Processes that require CPU time are added to the back of the queue.

33

Scheduling where a process can run for some fixed maximum time, once it has reached its maximum time, it is interrupted and set 'ready' and the scheduler runs the next process.

36

Scheduling where processes run until they are terminated or blocked.

35

When the scheduling algorithm leaves a process out for a long time, causing the process to not receive any CPU time.

38

The 'time-slice' or 'time-quantum'.

37

Priorities that are assigned by the system to achieve certain goals.

40

Priorities that are predetermined for each process.

39

What is a race condition?

41

What do the following terms mean?

1. *Data inconsistency*
2. *Synchronisation*
3. *Critical section*
4. *Mutual exclusion*

42

What is deadlock?

43

What is the base register of a program?

44

What is the limit register of a program?

45

What is the base register usage sequence?

46

What is the limit register usage sequence?

47

What is the virtual address?

48

When one or more processes/threads execute in parallel, but the outcome depends on which finishes first.

41

1. *Disagreement about data values*
2. *Using appropriate policies and mechanisms to ensure the correct operation of cooperating processes*
3. *Section of code in which shared data is used*
4. *At most 1 process can be in its critical section at once*

42

Where there are a set of waiting processes where each process is waiting for something that can only be provided by another of the processes.

43

A register that is loaded with the physical address where the program begins in memory.

44

A register that is loaded with the length of the program.

45

When the processor references memory, either fetch an instruction or read or write a data word, the CPU hardware automatically adds the base value to the address generated by the processor before sending the address out on the memory bus.

46

When the base register usage sequence happens, the OS checks if the address offered is greater than the value in the limits register, in which case a fault is generated and access aborted.

47

An address that is generated by a program. It is converted to the actual 'physical address' which is used in memory.

48

What performs the virtual to physical address conversion?

49

What is the difference between a partition and a program?

50

What are the two main reasons for virtual memory in a computer system?

51

What is a page table?

52

What are the two techniques used to implement virtual memory?

53

Segmented memory is split into chunks of a _____ *size.*

54

What is the formula to work out the number of pages used in a system?

55

What is the format of an address to paged memory?

56

A partition is: division of the storage area of a memory.
A program is: supplies a computer with a set of pre-written instructions.

50

The memory management unit (MMU)

49

A table used by the MMU to translate from a virtual to a physical address.

52

1. *To allow a processor to address a much larger address space than is implemented by the physical memory*
2. *To support the OS in the management of processes*

51

Segmented memory is split into chunks of a variable size.

54

Paged virtual memory and segmented virtual memory.

53

Page number + offset

56

$$\#pages = \frac{addressSpace}{PageSize}$$

Where addressSpace and pageSize are measured in bits.

55

What steps does the MMU take when it receives a paged address?

57

What columns are present in the page table?

58

In a page table, the resident flag checks whether the page is

59

In a page table the used flag check whether the page

60

In a page table, what does the dirty flag do?

61

LRU is a page replacement algorithm, what does LRU stand for?

62

FIFO is a page replacement algorithm, what does FIFO stand for?

63

Briefly explain what the FIFO page replacement algorithm does.

64

1. *Look up the page number in the page table and see if it's in physical memory.*
2. *If it is in memory, generate a physical address ((pageSize × pageNumber) + offset), and request that from RAM.*
3. *If it's not, abort the memory access, OS will load the page into memory (this is a page fault).*

57

- *Resident flag*
- *Used flag*
- *Dirty flag*
- *Physical address*
- *Disk address*

58

In a page table, the resident flag checks whether the page is currently in virtual memory

59

In a page table the used flag check whether the page has been used

60

Checks whether the page has been written to while it's been in physical memory (so it needs to be copied back in full when it's put back onto the disk).

61

Last recently used

62

First in first out.

63

Identifies the oldest page in memory and gets rid of it.

64

Briefly explain what the second chance page replacement algorithm does.

65

Briefly explain the last recently used page replacement algorithm.

66

What is a write-back?

67

What are two attributes that segments have that define how they are used?

68

When does a segment fault occur?

69

Explain external fragmentation.

70

What can we do to prevent fragmentation?

71

This picks the page that has been used the least amount of time. Implemented by using a timestamp.

66

This picks the oldest page with the fewest number of accesses since the last pass of the algorithm.

65

Usage rights and access rights.

68

When we write a page back to the disk when we swap it out.

67

When segments of different sizes are moved between main memory and the backing store.

70

When the OS tries to access a segment that is not in memory.

69

Use algorithms such as best fit and first fit when the OS is placing segments in memory.

71