**CSCI 540 Homework Assignment 2**

**Poojitha Chennuru - 50307728**

1. **Briefly describe memory hierarchy. (5%)**

Computer hierarchy is divided based on the speed of access by the CPU the main divisions are CPU registers, Cache, Main memory or primary memory and Secondary Memory.

CPU registers being the fastest of all of them, but they are expensive and volatile in nature.

**Cache** is the second fastest memory. It is volatile too and expensive. It is used to store most frequently accessed data. There are different levels of cache.

**Main memory**: RAM comes under this category. The Main memory is volatile, but it is less expensive than cache or registers and more expensive than secondary storage.

**Secondary Memory:** Hard disks, SSD etc. comes under Secondary memory.

It is nonvolatile memory. Cheap and large.

The memory flows from **Secondary -> Primary -> Cache -> Registers**.

1. **What are the 3 types of CPU pins? (5%)**

The CPU has mainly 3 different types of pins:

1.DATA PINS

2.ADDRESS PINS

3.CONTROL PINS (BUS CONTROL, INTERRUPTS)

1. **What is the difference between a synchronous bus and an asynchronous bus? (5%)**

**Synchronous:**

The synchronousbus has a master clock driven by crystal oscillator.

All the bus tasks are completed in integral multiples of master clock frequency. These cycles are called Bus cycles.

All the parts communicating with synchronous bus must be in sync master clock.

The disadvantage comes system parts are heterogenous. The clock has to be geared down or adjusted to meet the different clocked device.

**Asynchronous:**

In asynchronous bus does not have any master clock.

Fractional cycles are allowed.

The asynchronous bus has main advantage when the system parts are heterogenousI.e., when system parts have different clock.

It uses a method call handshake to communicate.

1. **In an asynchronous bus, what are the steps that a master device let a slave device do a job? (5%)**

* Without an expert clock, it's anything but an exceptional sign MSYN (master synchronization).
* At the point when the slave sees MSYN stated, it plays out the work as quick as possible.
* When the slave finishes the work, it asserts SSYN (slave synchronization).
* At the point when the expert sees SSYN affirmed, it realizes the work is done, so it nullifies MSYN.
* At the point when the slave sees MSYN refuted, it invalidates SS.

1. **What is a multiplexed bus? (5%)**

A type of bus structure in which the number of signal lines comprising the bus is less than the number of bits of data, address, or control information being transferred between elements of the system.

For example, a multiplexed address bus might use 8 signal lines to transmit 16 bits of address information. The information is transferred sequentially, i.e. time-domain multiplexed, with additional control lines being used for sequencing the transfer.

1. **What is bus arbitration? Briefly describe centralized bus arbitration and decentralized bus arbitration. (10%)**

Bus arbitration:

Generally, if there are multiple I/O devices which are trying to become the BUS master there should be a mechanism to who gets to be a BUS master this phenomenon is called bus arbitration.

There are two types of bus arbitration:

1. Centralized arbitration
2. De-centralized arbitration

**Centralized arbitration**:

There will be a single bus arbiter which decides who gets to be next BUS master.

The centralized arbiter has a wired- or request line. All the devices will make a request on the same line. So, when a request is made the arbiter doesn't know from which device the request has been made it only knows whether there is a request or not.

when the arbiter senses the request on the request line it will issues a grant buy a setting the bus grant line.

the input our output device which is near to the arbiter, the arbiter will check the input or output device made an request if the request is made the input output device will be thebusmaster if not it will pass the grant to next device

this scheme is called Daisy chaining.

**De-centralized arbitration:**

In decentralized Arbitration there will be multiple priority request lines. SO, the input and output devices will make a request on the priority request lines each request line has different priority so, the device which made request on the high priority line will be given control over the bus. All devices monitor all the request lines by the end of each bus cycle every device knows whether the request his high priority or not and whether it can use the bus in the next bus cycle. decentralized arbitration uses this more lines but it reduces the cost of arbiter when compared to centralized arbitration.

1. **Briefly describe the topology of PCI, PCIe, and USB. (10%)**

**PCI (Peripheral Component Interconnect): -**

As the need for high quality videos and graphics are increases the high bandwidth buses are needed. Then intel came up with the PCI which uses parallel bus architecture so there will be many masters and slaves. But at beginning the PCI are not backward compatible with ISA.

So, the intel came up with new architecture with two bridges in 1995 with Pentium chips.

1. PCI bridge connects CPU, memory, and PCI bus.

2.ISA bridge connects ISA bus and PCI bus.

Architecture of early Pentium chip.:

Diagram

Description automatically generated

**PCIe (Peripheral Component Interconnect express): -**

Theobjectiveof PCIe is to get rid of the multiple masters and slaves and go to high-**speed serial p2p** connection.

PCIe uses high speed switches.

It is inspired from local area networking concept.

It uses three key features or concepts.

**1.Packet**

**2.Header (to reduce the control signals)**

**3. Payload which contains data.**

To use these concepts, they developed new protocol **PCIe express Protocol stack**.

Totally they created a miniature **packet switching network**.

PCIe architecture:

Diagram

Description automatically generated

**USB(Universal serial bus):**

PCI and PCIe are too expensive for the low-speed devices like keyboard, mouse etc.

So, they had to come up with low-cost effect for these types of devices.

They began with some goals and started project.

Goals are:

1. Users must not have to set switches or jumpers on boards or devices.

2. Users must not have to open the case to install new I/O devices.

3. There should be only one kind of cable to connect all devices.

4. I/O devices should get their power from the cable.

5. Up to 127 devices should be attachable to a single computer.

6. The system should support real-time devices (e.g., sound, telephone).

7. Devices should be installable while the computer is running.

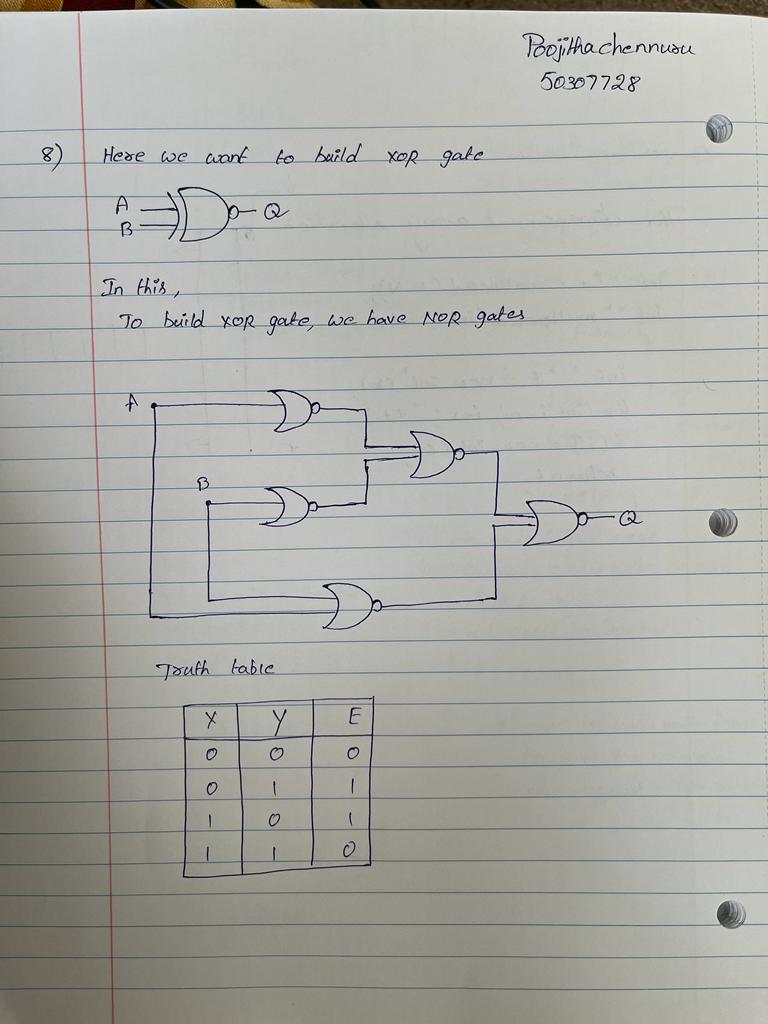
8. No reboot should be needed after installing a new device.

9. The new bus and its I/O devices should be inexpensive to manufacture

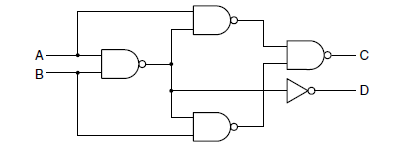
A USB system consists of a **root hub** that plugs into the main bus.This hub has sockets for cables that can connect to I/O devices or toexpansion hubs, to provide more sockets, so the topology of a USB system is a treewith its root at the root hub, inside the computer. The cables have different connectorson the hub end and on the device end, to prevent people from accidentallyconnecting two hub sockets together.

The cable consists of four wires: two for data, one for power (+5 volts), andone for ground. The signaling system transmits a 0 as a voltage transition and a 1as the absenceof a voltage transition, so long runs of 0s generate a regular pulsestream.

1. **We want to build an XOR gate, but unfortunately, we only have a bunch of NOR ⊕gates available. How to construct an XOR gate? Draw your digital circuit. (Do not search online to waste such a chance of practice.) (10%)**



**9. What does this circuit do? (10%)**



Analyze the circuit for different inputs of A and B:

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C(OUT) | D(OUT) |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

From truth table and using sum of products.

C= A’B+B’A = A XOR B

D=AB

These equations look exactly same as two-bit half adder where C is Sum and D is carry.

1. **Construct a digital circuit to implement the following Boolean logic. You may use any gates. (10%)**

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | X (result) |
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

Using sum of products X=A’BC’+AB’C+ABC

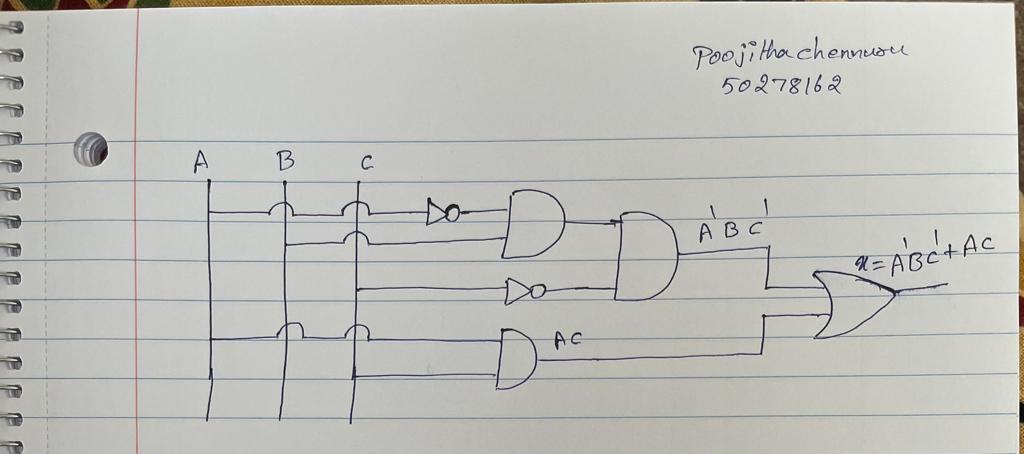
Simplify X

A’BC’+AC(B+B’)

A’BC’+AC SINCE B+B’=1

SO X= A’BC’+AC

**CIRCUIT DIAGRAM**:

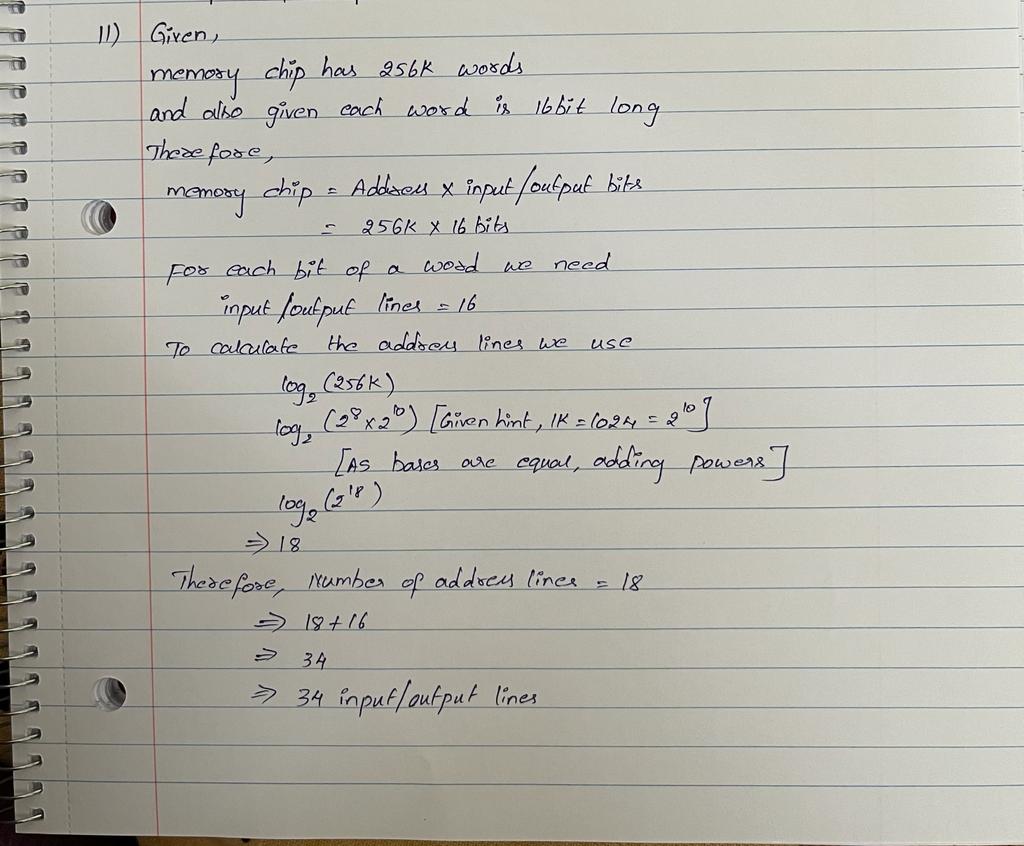


1. **Using the design of Figure 3-28, if the memory chip has 256K words, and each word is 16 bit long, then.**

**How many data input/output lines do we need? (5%)**

**How many address lines do we need? (5%)**

**Hint: 1K = 1024 = 210.**



1. **[*Open question*] Which part of Intel Core i7 design most impresses you? (5%)**

In intel core i7 processor design the ability to put the CPU to sleep to reduce the power draw from the battery. It is amazing that only one part of CPU is awake and other some part of CPU is asleep.

In addition to the there are levels of sleep modes based on how long the system will be away from being used.

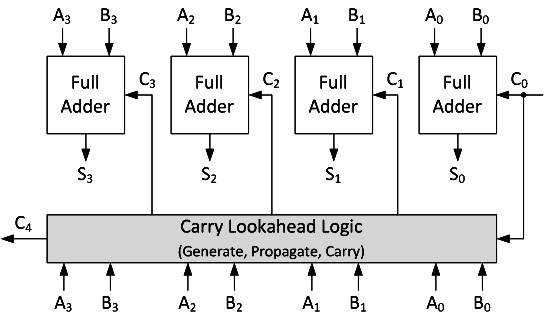
Sleep modes help systems to wake up fast than compared to cold start.

The other feature impressed me is hardware level hyperthreading as i7 is a multicore CPU the hyper threading enables to perform multitasking and very low latencies when compared to software threading.

1. **[*Carry-lookahead adder*] An *n*-bit adder can be constructed by cascading *n* full adders in series, with the carry into stage *i*, *Ci*, coming from the output stage *i* – 1. The carry into stage 0, *C*0, is 0.**

**However, gate has propagation delay. If each stage takes *T* nsec to produce its sum and carry, the carry into stage *i* will not be valid until *iT* nsec after the start of addition. For large *n* the time required to carry to ripple through to the high-order stage may be unacceptable long.**

**Design a 4-bit adder that works faster. I.e. Figure out the digital circuit of the gray part of the diagram below. (10%)**

****

**Hint:**

**For a 1-bit full adder, if A, B are both 1, then the carry-out is certainly 1.**

**If A+B (arithmetic addition) is 1, and the carry-in is 1, then the carry-out is 1.**

**I.e. if A⊕B (logical XOR) is 1, and the carry-in is 1, then the carry-out is 1.**

**So the carry-out can be written as**

**Cout = AB + (A⊕B)Cin**

**where + is logical OR.**

**Based on this, in a multi-bit adder, each *Ci* can be expressed in terms of the operand bits *Ai* – 1 and *Bi* – 1 as well as the carry *Ci* – 1. Using this relation, it is possible to express *Ci* as a function of the inputs to stages 0 to *i* – 1, so all the carries can be generated simultaneously. You will need more gates to implement it.)**

From hint we can write Ci+1= AiBi + (Ai⊕Bi) Ci

Let AiBi= Gi and Ai⊕Bi= Pi

From above

C1=G0 + P0 C0

C2=G1 + P1 C1

=G1 + P1(G0 + P0 C0)

=G1 + P1G0 +P1 P0 C0

C3= G2 + P2C2

=G2 + P2(G1 + P1G0 +P1 P0 C0)

=G2 + P2G1 + P2P1G0 +P2P1 P0 C0

C4=G3 + P3C3

=G3+P3(G2 + P2G1 + P2P1G0 +P2P1 P0 C0)

=G3+P3G2 + P3P2G1 + P3P2P1G0 +P3P2P1 P0 C0

So, looking at the above equations to calculate carry separately we need a lot more gates, so all the carries are calculated at the same time, but the hardware complexity increases.

**Circuit diagram for the grey block**:

