

CSCI 540 Homework Assignment 2

Name: Rama Krishna Kamma
CWID: 50321021

1. Briefly describe memory hierarchy. (5%)

Answer:

The basic divisions of the computer hierarchy are the CPU registers, the cache, the main memory, also known as primary memory, and the secondary memory.

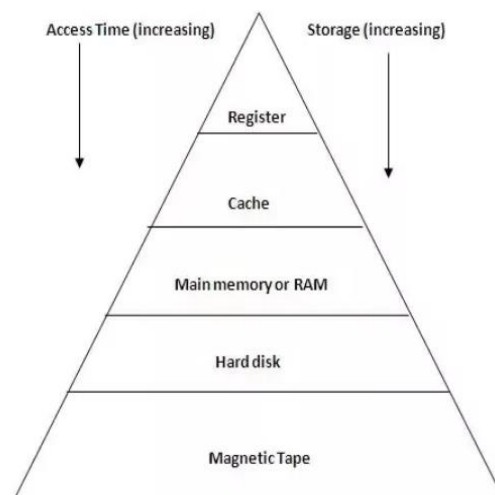
Registers: The smallest and fastest type of memory found inside the CPU is called a register. They hold information that the CPU is currently processing or executable instructions. Register access times are measured in a few clock cycles, which is incredibly quick. Although CPU registers are the quickest of all of them, they are also expensive and unstable.

Cache: The second-fastest memory is cache. Additionally, it is pricey and fickle. The most commonly accessible data is kept there. There are various cache levels.

Main Memory: RAM falls under the category of main memory. Although the main memory is volatile, it is more expensive than secondary storage and less expensive than cache or registers.

Secondary Memory: Hard drives, SSDs, and other storage devices are considered secondary memory. This memory is nonvolatile. big and affordable.

The memory is distributed as follows: Secondary, Primary, Cache, and Registers.



2. What is the difference between a synchronous bus and an asynchronous bus? (10%)

Answer:

Synchronous:

- An oscillator in a crystal powers the master clock on the synchronous bus.
- In integral multiples of the master clock frequency, all bus tasks are accomplished. Bus cycles are the name for these cycles.
- The master clock must be in sync with all the components connecting with the synchronous bus.
- The fact that system components vary is a drawback. To accommodate the various clocking devices, the clock must be geared down or altered.

Asynchronous:

- There is no master clock in an asynchronous bus.
- Cycles with fractions are accepted.
- The main benefit of the asynchronous bus is when the system components are heterogeneous, or when the clocks of the components disagree.
- It communicates using a technique known as the handshake.

Difference:

Synchronous Bus	Asynchronous Bus
Both the transmitter and the receiver are in sync with the clock.	Both the transmitter and the receiver are not in sync with the clock.
Data bits are sent when the clocks are synchronized.	Data bits are transmitted with the constant rate.
Used to transmit data at a rapid pace.	Used to transmit data at a slow pace.
Longer physical distances cannot be handled by synchronized buses.	Longer physical distances can be handled by asynchronized buses.
To prevent issues with clock-skewing, synchronous bus lengths could be restricted.	The asynchronous bus length could not be restricted.

3. What is a multiplexed bus? (10%)

Answer:

A multiplexed bus, usually referred to as a multiplex bus or just a multiplex, is a kind of data bus used in computer systems to transmit many signals or streams of data across a single physical channel. It increases efficiency and simplifies the system by allowing several devices or components to share a single bus.

A bus structure in which the number of signal lines is less than the amount of data, address, or control bits being exchanged between system components.

For instance, a multiplexed address bus might send 16 bits of address data over 8 signal lines. Additional control lines are utilized to sequence the transfer of the information, which is done time-domain multiplexed.

The data from several sources is combined or multiplexed into a single stream before being broadcast over the bus in a multiplexed bus. Data is demultiplexed at the receiving end to separate the various signals or data streams. A device known as a multiplexer at the transmitting end and a demultiplexer at the receiving end normally controls this multiplexing and demultiplexing process.

In many computer systems, including input/output (I/O) interfaces, memory subsystems, and microprocessors, multiplexed buses are employed. The number of physical lines needed for communication can be decreased by multiplexing many signals onto a single bus, which simplifies the system architecture and lowers costs.

The Address/Data bus of a microprocessor is a well-known example of a multiplexed bus. In this scenario, data and address information are both transported across a single set of bus lines between the microprocessor and the memory or other peripheral devices. The memory or other devices employ demultiplexing algorithms to separate the address and data information from the multiplexed signals sent on the same set of lines by the microprocessor.

Overall, a multiplexed bus makes it possible for computer systems to share communication channels effectively, maximizing resources and enhancing system performance.

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

Answer:

Bus arbitration:

In general, if there are several I/O devices competing to become the BUS master, there should be a mechanism to determine who is selected as the BUS master.

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 the next BUS master. A wired- or request line is available to the central arbiter. On the same line, a request will be made by each device. As a result, when a request is made, the arbiter is unaware of the device from which it originated; instead, they just know if a request has been made or not.

The arbitrator will issue a grant by setting the bus grant line when it detects the request on the request line.

the input or output device that is close to the arbiter will be checked by the arbiter to see if a request was made; if so, the input or output device will become the bus master; otherwise, the grant will be passed to the next device the phenomena is called Daisy chaining.

Decentralized arbitration:

There will be several priority request channels in decentralized arbitration. Therefore, a request will be made on the priority request lines by the input and output devices. The device that submitted a request on the line with the highest priority will be given control of the bus since each request line has a distinct priority. Every device understands whether or not the request is a high priority towards after each bus cycle ends and whether it can be used the bus in the subsequent bus cycle. Decentralized arbitration, as opposed to centralized arbitration, uses more lines but has cheaper arbitrator costs.

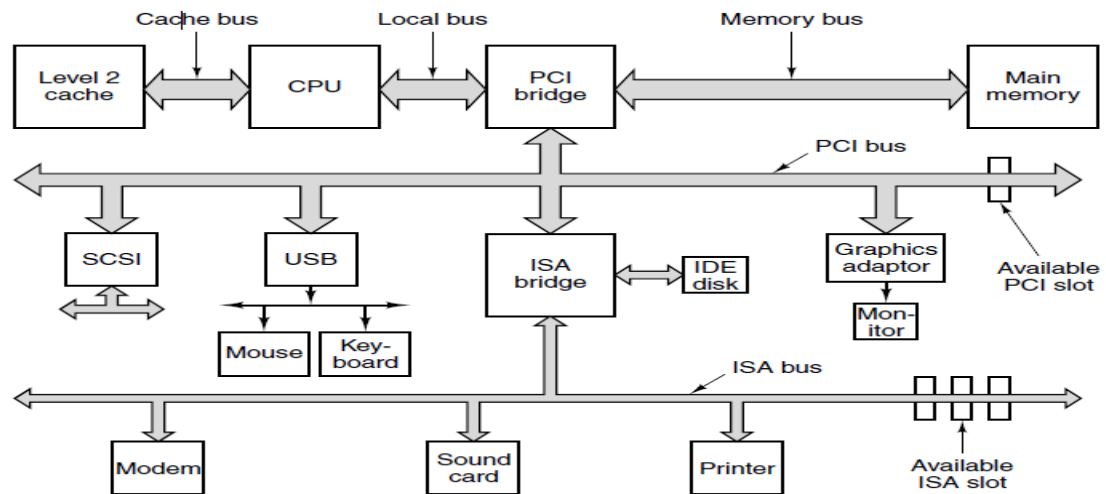
5. Briefly describe the topology of PCI, PCIe, and USB. (10%)**Answer:****PCI (Peripheral Component Interconnect):**

High bandwidth buses are required as demand for films and graphics of the highest quality grows. Then Intel developed PCI, a parallel bus architecture that allows for numerous masters and slaves. However, ISA and PCI are not initially backward compatible.

So, in 1995, using Pentium CPUs, Intel developed a new architecture with two bridges.

1. A PCI bridge links the PCI bus, memory, and CPU.
2. The ISA bus and the PCI bus are connected by an ISA bridge.

Architecture of early Pentium chip.:



PCIe (Peripheral Component Interconnect Express):

PCIe's goal is to replace several masters and slaves with a high-speed serial peer-to-peer connection.

High-speed switches are used by PCIe.

It draws inspiration from the idea of local area networking.

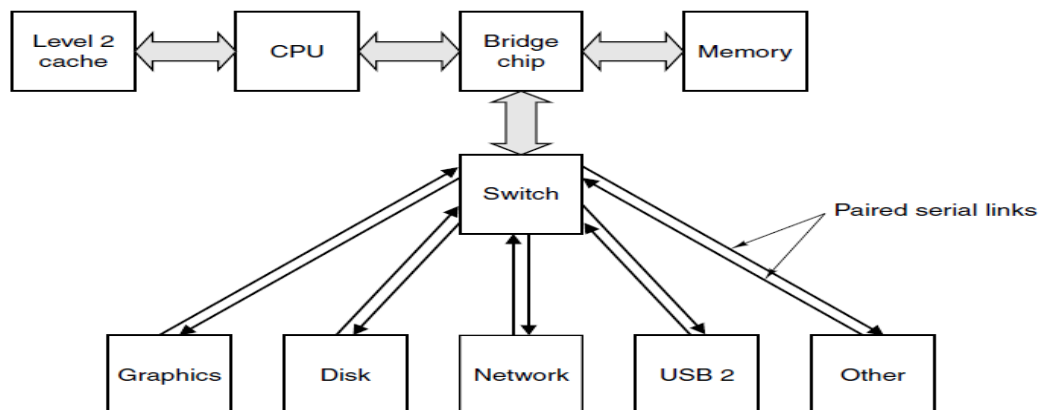
It uses three key features or concepts.

- Packet
- Header (to reduce the control signals)
- Payload which contains data.

They created a new protocol, the PCIe express Protocol stack, using these ideas.

All in all, they built a tiny packet switching network.

PCIe Architecture:



USB (Universal Serial Bus):

Low-speed devices like keyboards, mouse, and other similar items are too expensive for PCI and PCIe.

They have to develop low-cost effects for these kinds of gadgets as a result.

They started the endeavor with a few objectives.

The following are the objectives:

1. On boards or devices, users shouldn't have to modify switches or jumpers.
2. The installation of new I/O devices shouldn't need users to open the case.
3. Only one type of cable should be used to connect all devices.
4. The wire should supply power to I/O devices.
5. A single computer should support 127 device attachments.
6. Real-time devices (such as sound and telephone) should be supported by the system.
7. Hardware should be able to be installed even when the computer is running.
8. After installing a new device, there shouldn't be a requirement for a reboot.
9. The new bus and its I/O components should be manufactured at a reasonable price.

A root hub that connects to the main bus makes up a USB system. In a computer, the root hub serves as the top node of a tree representing the topology of a USB system. In order to provide additional connections, this hub includes ports designed for connecting cables to extension hubs or I/O devices. The cables utilized here have distinct connectors on the device end and the hub end, ensuring that users cannot accidentally connect together by two hub sockets.

The cable contains of four wires: a ground wire, a power wire, and two data lines. The way the signaling system operates is by considering the voltage transition 0 when it is present and in the absence of a voltage transition is considered as 1. As a result, when there are long sequences of 0s, it generates a continuous pulse stream.


6. 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%)

Answer:

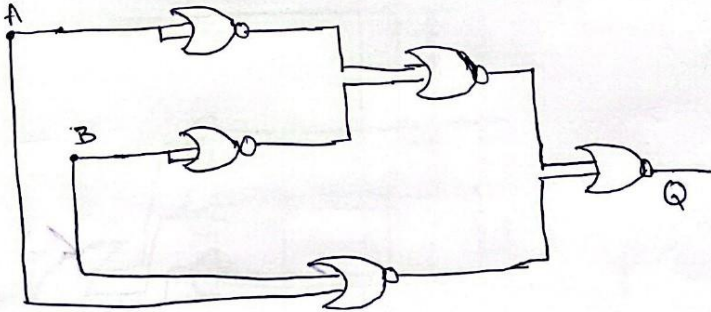
Name : Rama Krishna kamma
CUID : 50391021

Question-6:

Here we want to build XOR Gate



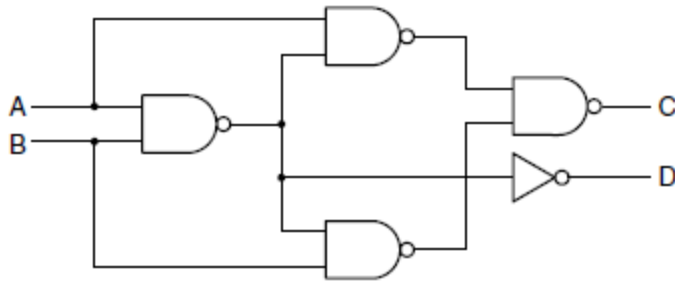
In this scenario,
we have to use NOR gates.



Truth table:

A	B	E
0	0	0
0	1	1
1	0	1
1	1	0

7. What does this circuit do? (10%)



Answer:

Analyze the circuit for different inputs of A and B:

A(Input)	B(Input)	C(Output)	D(Output)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

From truth table and using sum of products.

$$C = A'B + B'A = A \text{ XOR } B$$

$$D = AB$$

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

8. 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

Answer:

Using sum of products

$$X = A'BC' + AB'C + ABC$$

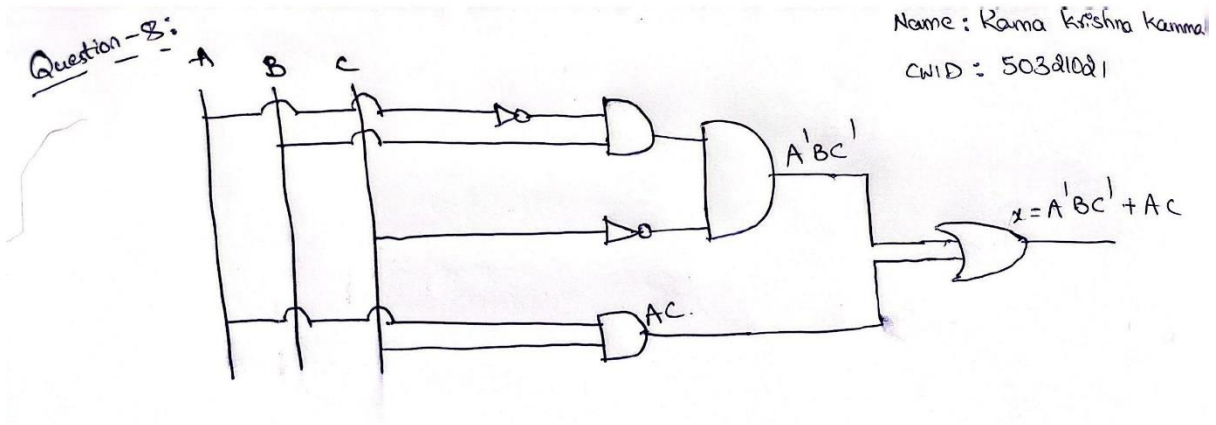
By Simplifying the X

$$= A'BC' + AC(B+B')$$

$$= A'BC' + AC \quad (\text{SINCE } B+B'=1)$$

So finally

$$X = A'BC' + AC$$



9. 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: $1K = 1024 = 2^{10}$.

Answer:

As per the Given data

Each Work is of length 16 bit.

Memory chip has 256k words.

Memory Chip = Address x Input/Output bits

$$= 256k \times 16\text{bits}$$

For Each bit of a word we need at least

Input/output lines = 16

In order to calculate address lines, we use.

$$= \log_2(256k)$$

$$= \log_2(2^8 \times 2^{10}) [1k = 1024 = 2^{10}]$$

$$= \log_2(2^{18})$$

$$= 18$$

Therefore, the number of address lines = 18

Total number of Input/Output lines is
= 18 + 16
= 34 input/output lines

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

Answer:

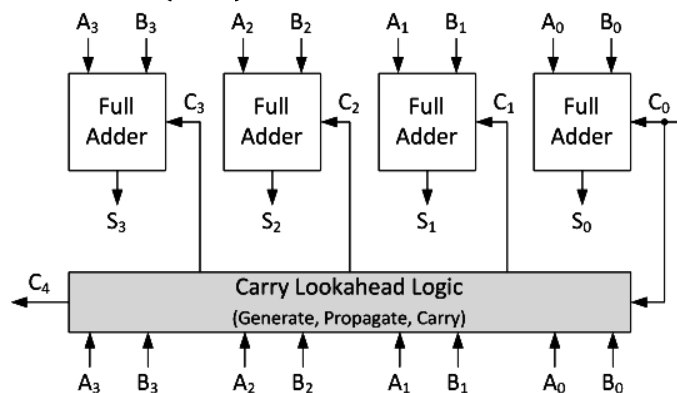
The option to put the CPU to sleep was incorporated into the design of the Intel Core i7 chip to minimize battery consumption. It's incredible that only one portion of the CPU is active while the others are dozing off. There are also many levels of sleep modes depending on how long the system will be idle. Systems that are in sleep mode wake up more quickly than those that start cold.

The other feature that I found impressive was hardware level hyper threading. Because the i7 is a multicore CPU, this function allows for multitasking and has far lower latencies than software threading.

11. [Carry-lookahead adder] An n -bit adder can be constructed by cascading n full adders in series, with the carry into stage i , C_i , 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 $A \oplus B$ (logical XOR) is 1, and the carry-in is 1, then the carry-out is 1.

So the carry-out can be written as

$$C_{out} = AB + (A \oplus B)C_{in}$$

where + is logical OR.

Based on this, in a multi-bit adder, each C_i can be expressed in terms of the operand bits A_{i-1} and B_{i-1} as well as the carry C_{i-1} . Using this relation, it is possible to express C_i 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.)

Answer:

Name: Rama Krishna Kamra
CUID: 50321021

Question-11:

From the question hint we can understand

$$C_i + 1 = A_i B_i + (A_i \oplus B_i) C_i$$

Let us consider

$$A_i B_i = G_i$$

$$A_i \oplus B_i = P_i$$

From above equations

$$C_1 = G_0 + P_0 C_0$$

$$\begin{aligned} C_2 &= G_1 + P_1 C_1 \\ &= G_1 + P_1 (G_0 + P_0 C_0) \\ &= G_1 + P_1 G_0 + P_1 P_0 C_0 \end{aligned}$$

$$\begin{aligned} C_3 &= G_2 + P_2 C_2 \\ &= G_2 + P_2 (G_1 + P_1 G_0 + P_1 P_0 C_0) \\ &= G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0 \end{aligned}$$

$$\begin{aligned} C_4 &= G_3 + P_3 C_3 \\ &= G_3 + P_3 (G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0) \\ &= G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0 \end{aligned}$$

From the above observation we can see the previously discussed formulae to calculate each carry independently would require a lot more gates, increasing the complexity of the hardware

Name : Rama Krishna Kanna

CWID : 50321021

Question - 11:

