Discuss different desktop applications that requires the great power of Contemporary microprocessor-based systems?

Different desktop applications that require the great power of Contemporary microprocessor based System are:

Image processing
Three-dimensional rendering
Speech recognition
Video conferencing
Multimedia authoring
Voice and video annotation of files.
Simulation modelling.

) Discuss the techniques used in Comtemporary processors to ~~parcialeeim~~ increase speed?

The techniques used in Comtemporary processors to increase speed are following.

PIPELINING :

Pipelining enables a processor to work simultaneously on multiple instructions at same time.

BRANCH PREDICTIONS :

Branch prediction potentially increases the amount of work available for the processor to execute.

) SUPER SCALE EXECUTION :

This is the ability to issue more than one instruction in every

Processor clock cycle. In effect, multiple parallel pipelines are used.

DATA FLOW ANALYSIS :
The processor analyzes which instructions are dependent on each other's result, or data, to create an optimized schedule of instructions.

SPECULATIVE EXECUTION :
This enables the processor to keep its execution engines as busy as posible by executing instructions that are likely to be needed.

:) Discuss the problems created due to increase in clockk speed and logic density of the processor?
The problems created due to increase in clock speed and logic density of the processor are;

POWER :
As the density of and the clock speed increases, the power density increases too and head

is dissipated.

RC DELAY:

The speed at which electrons can flow on a chip between transistors is limited by the resistance and capacitance of the metal wires connecting them, specifically, delay increases as the RC product increases.

MEMORY LATENCY:

Memory access speed (Latency) and transfer speed (Throughout) log processor speeds.

3.) Discuss the speedup of a program using multiple processors compared to a single processor using Amdahl's Law?

The speedup using a parallel processor with N processor that fully exploits the parallel portion portion of the program is as follows;

Speed up = Time to execute program on a single processor / Time to execute program on N parallel processors.

$$= T(1-f) + Tf/T(1-f) + Tf/N =$$

$$1/(1-f) + f/N$$

1) Discuss the multicore, MIC, and GPCPU in detail?

MULTICORE :

The use of multipl processors on the same chip provides the Potential to increase performance without increasing the clock rate. Strategy is to use simpler processors on the chip rather than one more Complex processor, with two processors larger caches are justified. As caches became larger it made performance sense to create two and then three levels of cache on a chip.

MIC :

Leap in a performance as well as the challangs in developing software to explot such a large number of cores. The multicore and MIC strategy involves a

homogenous Collection of general purpose processors on a Single chip.

GPUs :

Core designed to perform parallel operations on graphic data.
Traditionally found on a plug-in graphics Card, it is used to encode and render 2D and 3D graphics as well as process video used as a vector processor for a variety of applications that requires repetitive Computations.

Question # 8 Part A :

Ans) EFFECTIVE CPI :

$$CPI = (1 \times 46000) + (2 \times 3300) + (2 \times 16000) + (2 \times 9000) / 100$$

$$CPI = 162000 / 100$$

$$CPI = 1620$$

MIPS RATE :

$$MIPS\ rate = 60\,MHz / 1620 \times 10^6$$

$$MIPS\ rate = 60 \times 10^6 / 1620 \times 10^6$$

$$= 60 / 1620$$

$$MIPS\ rate = 0.037$$

EXECUTION TIME :

$$T = Ic / (MIPS \times 10^6)$$

$$T = 104000 / (0.037 \times 10^6)$$

$$= 104000 / 37 \times 10^3$$

$$T = 2811 \times 10^{-3}$$

$$T = 2.811\ sec$$

//

Part B :

Consider two different ---------
---- time for each machine.

Solution :

$$CPI_A = \frac{\sum CPI_i \times I_i}{I_c}$$

$$= \frac{(8\times 1 + 4\times 3 + 2\times 4 + 4\times 3)\times 10^6}{(8+4+2+4)\times 10^6}$$

$$= 2.22$$

$$MIPS_A = \frac{f}{CPI_A \times 10^6} = \frac{200\times 10^6}{2.22\times 10^6} = 90$$

$$CPU_A = \frac{I_c \times CPI_A}{f}$$

$$= \frac{18\times 10^6 \times 2.2}{200\times 10^6}$$

$$= 0.2s$$

$$CPI_B = \frac{\sum CPI_i \times I_i}{I_c}$$

$$= \frac{(10\times 1 + 8\times 2 + 2\times 4 + 4\times 3)\times 10^6}{(10+8+2+4)\times 10^6}$$

$$= 1.92$$

$$MIPS_B = \frac{f}{CPI_B \times 10^6} = \frac{200 \times 10^6}{1.92 \times 10^6}$$

$$= 104$$

$$CPU_B = \frac{I_c \times CPI_B}{f} = \frac{24 \times 10^6 \times 1.92}{200 \times 10^6}$$

$$= 0.23 s$$

Part C:

(a) What is the ---- on the two machines?

The MIPs rate could be computed as the following:

$$[(\text{MIPs rate}) / 106] = I_c / T$$

Thus that

$$I_c = T \times [(\text{MIPS rate}) / 106]$$

Now by computing the ratio of the instruction count of the IBM RS/6000 to the VAX 11/780 which is:

$[X \times 18] / [12x \times 1] = 18x / 12x = 1.5$

(b) What are the CPI values for the two machines?

Regarding to the VAX 11/780, the CPI = (5 MHz) / (1 MIPS) = 5

Regarding to the IBM RS/6000, the CPI = (25 MHz) / (18 MIPS) = 1.4

Question #2 Part D

a) Since we have the same instruction min, that means the additional instructions for each task could be allocated appropriately between the instruction types. Therefor the following table be given.

Instruction Type	CPI	Instruction Mix
Arithmetic and logic	1	60 %
Load/Stor with Cachebit	2	18 %
Branch	4	12 %
Memory reference with Cache miss	12	10 %

The average CPI =

$$(1 \times 0.6) + (2 \times 0.18) + (4 \times 0.12) + (12 \times 0.1)$$

$$= 2.64$$

Therefor, the CPI has been increased since the last time for memory access is also increased.

b) MIPS = 400 / 2.64 $\Rightarrow$ 152

There is a corresponding drop in the MIPS rate.

c) The speed up factor equals to the ratio of the execution times. The execution time is called

as the following.

$$T = Ic / (MIPS \times 10^6)$$

For one processor, T_1

$$T_1 = (2 \times 10^6) / (178 \times 10^6)$$
$$= 11 ms$$

For 8 processors, each processor executes 1/8 of the 2 million instruction plus 25000

$$T_8 = 2 \times 10^6 / 8 + 0.025 \times 10^6 / 152 \times 10^6$$
$$T_8 = 1.8 ms$$

Therefor;

Speed up = Time to execute Program on a single processors / Time to execute program on N parallel processors

$$\text{Speed up} = 11 / 1.8$$
$$= 6.11$$

d) By depending on the information given, it is not obvious how to quantify this effect in Amdahl's equation. Therefor if it is ~~sn~~ supposed that the fraction of code, which is parallelizable, is $f = 1$, then Amdahl's law decrease to Speed up = N = 8. So, the actual speedup is only about 75% of

the theoretical speed up.