- a. The real-time constraint is one of crucial performance requirements in embedded applications.

 b. CISCs can be also suitable for processors of embedded systems.

 c. Dynamic power consumption is proportional to the product of clock frequency and voltage.

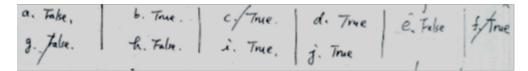
 d. The scheduling of VLIW architectures is performed only by compiler.

 e. Reorder buffer cannot be replaced with register renaming.

 f. The number of entries in a (2,2) predictor of a total of 8K bits is 2K.

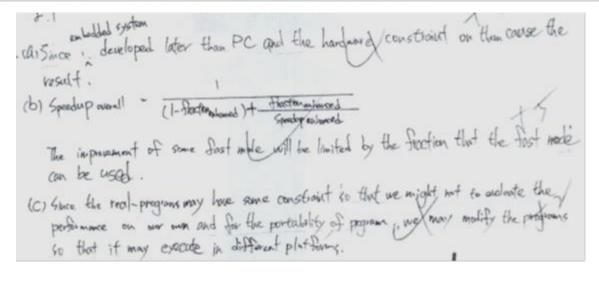
 g. Amdahl's Law is not suitable for muticore processors.

 h. Increasing issue rate per clock cycle can reduce CPL
 - a. Moore's law said that the clock rate of CPU is doubled in 18 months.
 - b. RISC processors can be also suitable for processors of embedded systems due to low power.
 - c. Graphic Processor Units (GPUs) is used to exploit task-level parallelism.
 - d. Dynamic energy is consumed by switching bits from 0 to 1 or vice versa.
 - e. Dynamic Voltage-Frequency Scaling (DVFS) is a common technique to save static power.
 - Compared with SRAM, DRAMs must continue to be refreshed occasionally so as to not lose information with a minimal power.
 - g. DDR and DDR2 are two DRAM standards; each of them has three types. Any type of DDR2 is faster than any type of DDR.
 - h. For virtual machine (VM), the mapping of virtual resources to physical resources is determined by host OS.
 - Disks partitioned by virtual machine monitor (VMM) are to create virtual disks for guest VMs.
 - j. There are four protection levels in paravirtualization of Xen.



(15%) To evaluate the performance, we must find metrics and approaches to measure different computers.

- (a) Why embedded benchmarks are less mature than PC benchmarks?
- (b) Explain what the Amdahl's Law is. Notice please give the formal equation and don't just describe it.
- (c) For synthetic benchmarks, why and what modifications are needed?



(18%) Three enhancements with the following speedups are proposed for a new architecture:

Speedup1 = 30

Speedup2 = 20

Speedup3 = 15

Only one enhancement is usable at a time.

- a. (6%) If enhancements 1 and 2 are each usable for 25% of the time, what fraction of the time must enhancement 3 be used to achieve an overall speedup of 10?
- b.(6%) Assume the enhancements can be used 25%, 35%, and 10% of the time for enhancements 1, 2, and 3, respectively. For what fraction of the reduced execution time is no enhancement in use?
- c.(6%) Assume, for some benchmark, the possible fraction of use is 15% for each of enhancements 1 and 2 and 70% for enhancement 3. We want to maximize performance. If only one enhancement can be implemented, which should it be? If two enhancements can be implemented, which should be chosen?

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{(1-0.15) + \frac{0.15}{10}} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{0.025} = \frac{1}{0.025} = \frac{1}{0.025}$$

$$(C) \text{ (bly the } = \frac{1}{0.025} = \frac{1$$

(20%) Availability is the most important consideration for designing servers, followed closely by scalability and throughput.

- (a) (5%) We have a single processor with a failures in time (FIT) of 200. What is the mean time to failure (MTTF) for this system?
- (b) (5%) If you will build a supercomputer out of inexpensive computers. What is the MTTF for a system with 1000 processors? Assume that if one fails, they all fail.
- (c) (10%) Assume a disk subsystem with the following components and MTTF: 10 disks, each rated at 1,000,000 hour MTTF; 1 SCSI controller, 500,000 hour MTTF; 1 power supply, 200,000 hour MTTF; 1 fan, 200,000 hour MTTF; 1 SCSI cable, 1,000,000 hour MTTF. Using the simplifying assumption that the components lifetimes are exponentially distributed—which means that the age of the component is not important in probability of failure—and that failures are independent, compute the MTTF of the system as a whole.

(a)
$$MTTT = \frac{10^9}{200} = 5.10^6$$

(b) $MTTT = \frac{1 \times 10^6}{1000} = 51 \times 10^3 \times 10^{100}$
(c) I independent. Shirt of the sum of t

(18%) The Whetstone benchmark contains 195,578 basic floatingpoint operations in a single iteration, divided as shown in the following:

Operation	Count
Add	82,014
Subtract	8,229
Multiply	73,220
Divide	21,399

	Convert integer to	FP 6,006	4
	Compare	4,710	MOVE TO
	Total	195,578	200

Whetstone was run on a Sun 3/75 using the F77 compiler with optimization turned on. The Sun 3/75 is based on a Motorola 68020 running at 16.67 MHz, and it includes a floatingpoint coprocessor. The Sun compiler allows the floating point to be calculated with the coprocessor or using software routines, depending on compiler flags. A single iteration of Whetstone took 1.08 seconds using the coprocessor and 13.6 seconds using software. Assume that the CPI using the coprocessor was measured to be 10, while the CPI using software was measured to be 6.

- a. (6%) What is the MIPS rating for both runs?
- b.(6%) What is the total number of instructions executed for both runs?
- c.(6%) On the average, how many integer instructions does it take to perform a floating-point operation in software?

(10%) Assume we have a computer where the clocks per instruction (CPI) is 1.0 when all memory accesses hit in the cache. The only data accesses are loads and stores, and these total 50% of the instructions. If the miss penalty is 25 clock cycles and the miss rate is 2%, how much faster would the computer be if all instructions were cache hits?

(20%) For cache issues, one is that the write policies often distinguish cache designs. There are two basic options when writing to the cache on a hit:

- a. (6%) What are these two options? Please compare them.
- b. (4%) In a, which one is easier to be implemented? Why?
- c. (4%) Give and explain a technique to reduce write stalls.
- d. (6%) Since the data are not needed on a write, there are two are two options on a write miss. Compare these options.

a. write through: 当 cache hit 好. 同时高入 cache 南 (or 7 to cache) write back: 3 cache hit of & 3 x cache \$ 19 in duty, 当 cache block 預室 swap out of 才名入 memory (下降) write back 同为不常等行名入 memory (o 下唐), 1, 效果取好,过度较快 但必須夏披魯衛电的可風險(四尚書名回下層)造改資料不同多。 b. write through 時報為易 implement! write through 為新同時高入下層memory, 造成坑堤。期積個的 田太 wite back 至少領導 dirty bit. reple replace 37 時後 楼查是各家车高回下層, C. write buffer. 当常家 wite 到下層時值高高入 我 做证司 wite buffer, 写有空呼、write buffer 再富入下隐 流動力 wife stell write allecate: 3 wite mis 時. 艺游童料 讚人 cache, 再像 with hit 朝何名> non-write allocate, ywite miss 好, 直接名入 memory (7/4) 不诱入 cache, 著data已经不管再的到的社. non-write/allowite 比对好. 不同语意意意 cacho is of is). 但与 deta 不久约又管被印刷好、wite allocate e 致 村 data 为文入 cache 中。 之经可以降级 wiss nute.

(20%) For the following memory optimizations, explain their purposes and their impact on memory hierarchy.

- a. Way prediction
- b. Pipelined cache access
- c. Critical word first
- d. Hardware prefetching