VLSI Architecture Design Course (048853)
Final Exam
July 10th, 2005

Electrical Engineering Department

Student name:_____ Student number:_____

This exam contains TWO questions.
The exam duration is 2.5 hours.
Please fill the answers ON THE EXAM forms.

Please explain or provide a formula for each computation.

TAKE YOUR TIME, READ THE QUESTIONS THOUROUGHLY, UNDERSTAND THE CONTENT AND, *ONLY THEN,* START TO ANSWER

Good luck!

Q1	
Q2	
Total	

Question 1: Power & Architecture (50%)

Power and performance can be traded in several ways. In this question we want to understand the power/performance inter-relations among the following various microarchitectural options.

- **SC - Single Core micro-architecture enhancements.**
- **SMT - Simultaneous Multi-Threading**
- **CMP - Chip MultiProcessing**
- **Dynamic Voltage/Frequency Scaling**

In this question, assume that performance is strictly linear with frequency.
That is: Perf = c*Freq

A. DVS in single core (SC).

The first section deals with Dynamic Voltage Scaling (DVS).
DVS works as follows:

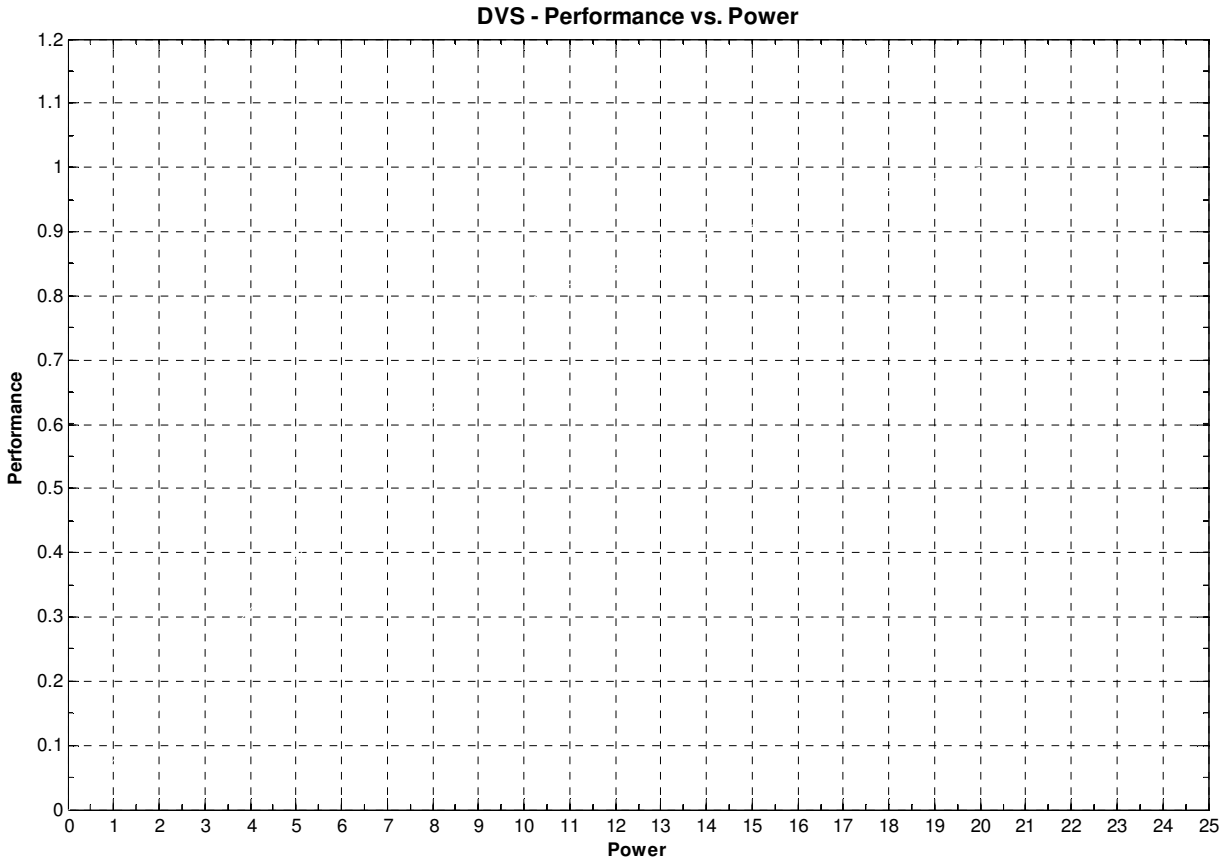
- **Voltage ranges between V_{MIN} and V_{MAX}. The processor cannot operate below V_{MIN}!**
  In our case: V_{MIN} = 1V, V_{MAX} = 1.25V. No operation below 1V.
- **Within this voltage range, (Max) frequency is linear with voltage (f=kV)**
  In our case: 2GHz at 1V, 2.5GHz at 1.25V
- **For a certain voltage, processor can operate at frequency below or equal the maximum frequency allowed for that voltage.**
  e.g. at V_{MIN}=1V, the processor can run at any frequency between 0GHz to 2GHz.
- **Power is always αCV^2f, where**
  α = activity factor, C = capacitance, V = Voltage, f = Frequency.
  For a certain processor and workload, $\alpha * C$ is constant.

A1. What are the performance/voltage/power values in the following cases?

Frequency (GHz)	Performance	Min Voltage (V)	Power (W)
2.5	1X	1.25	20
2.25			
2.0		1.0	
1.5			
1.0			
0.5			
0			

Please explain the formulas you use to fill the table:

A2. Plot the graph of Performance vs. Power for the entire range. You can use the data points from the table above.



A3. Which distinct cases are viewed in this graph? Explain.

B. Architecture trade-offs.

Each of the three options – SC/SMT/CMP – has typical performance/power/area characteristics. Rough estimates for operating in the same voltage are:

- **SC: Adding/removing microarchitecture features can yield: ½X to 2X performance gain, relative to the original performance (X).**
  Performance gain/decrease of Z requires Z^2 investment of both area and power
  e.g., 30% performance gain costs 69% additional area and power ($1.3^2 = 1.69$);
  30% performance decrease can save 51% in area and power ($0.7^2 = 0.49$).
- **SMT: can yield from 10% to 30% additional performance.**
  Each 2% performance costs 1% in area and power
  In our case, assume SMT is taken aggressively:
  30% performance for 15% area and power.
- **CMP: each additional core results in additional 80% performance gain.**
  Each additional core costs same area and power as the original core.

Notes:

1. **SC enhancements improve both Single Thread (ST) and Multi-Threaded (MT) performance. SMT & CMP improve only MT performance.**
2. **For this question: Assume power and area goes linearly, that is, a certain % increase in area results in the same % increase in power. Leakage power is ignored.**

B1. You are the chief architect of a new generation processor.

Your baseline is: A single core design (1X performance):

- **Area 25mm^2**
- **Power 20Watts at V_{MAX}**

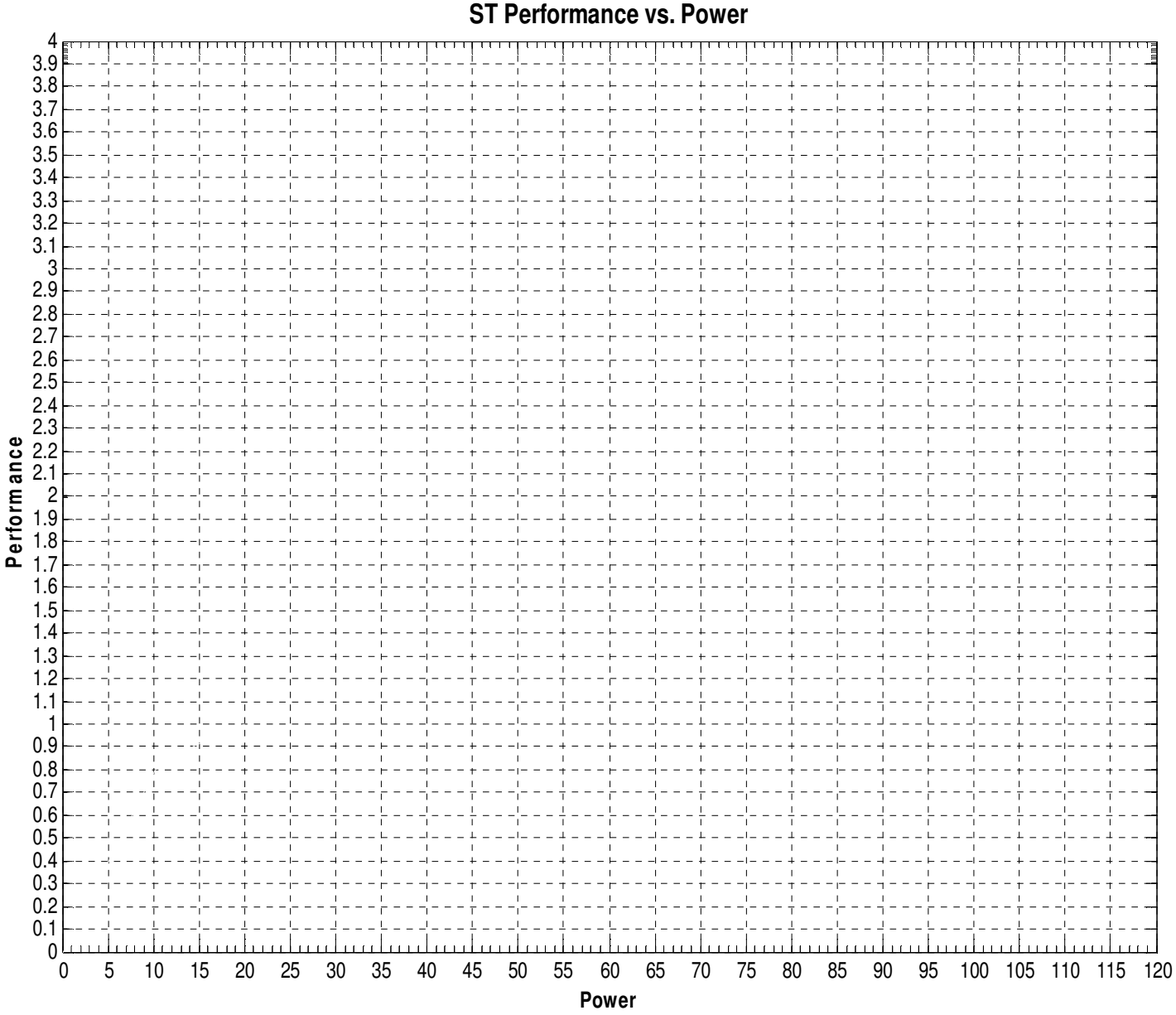
Your silicon budget for the new processor is about 100-115mm^2 (~4X area of original).
You want to consider the following options to use this silicon area.
(Note: Caches are included in the cores. Ignore them for this question)

Fill the following table.
For ST performance, assume only one core is active. The rest are inactive and do not consume power.
In an SMT processor, SMT power is consumed even if only 1 thread is running.
For MT performance, assume all cores are active and consume power.
Note that V_{MIN}, V_{MAX} and DVS ratios/assumptions are the same as in section A above.

				Single Thread					*Multithreading*				
				Performance		*Power*			*Performance*		*Power*		
Option Name	Cores	SMT	SC Area	SC-ST Max Perf.	SC-ST Perf. at V_{MIN}	SC-ST V_{MAX} Power	SC-ST V_{MIN} Power	SC-ST 1GHz Power	MT Max Perf.	MT Perf. at V_{MIN}	MT V_{MAX} Power	MT V_{MIN} Power	MT 1GHz Power
Base	Base1	N	25	1		20			1		20		
N1	1	N	100										
N2	2	N	50										
N4	4	N	25	1									
Y1	1	Y	115										
Y2	2	Y	58										
Y4	4	Y	29										

Please explain the formulas you use to fill the table:

B2. Plot the full curve of <u>ST Performance</u> vs. Power for the entire range of DVS for each of the above 6 options.
You can use the data points from the table above.



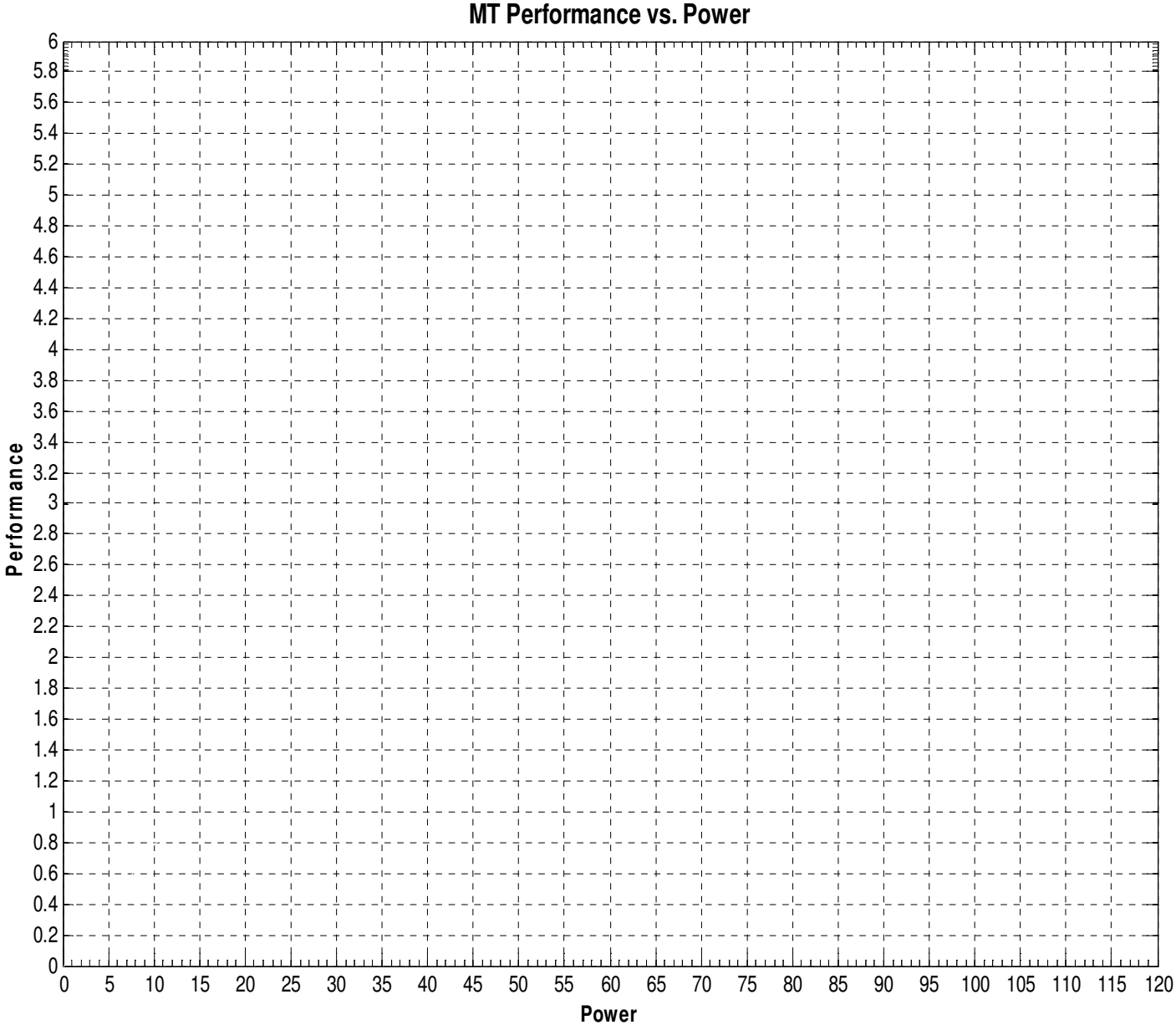
B3. Which are the best ST performers for the following power budget?
What is the relative performance in that case?

80W _____

20W _____

5W _____

B4. Plot the full curve of <u>MT Performance</u> vs. Power for the entire range of DVS for each of the above 6 options.
You can use the data points from the table above.



B5. Which are the best MT performers for the following power budget?
What is the relative performance in that case?

80W _____

20W _____

5W _____

C1. (BONUS QUESTIONS)
You were asked to use the 100mm^2 budget to achieve "the best of all worlds"

1. ST performance that is 10% better than the existing baseline
2. The best MT performance you can get under the above ST restriction.
3. Stay below or equal to 100mm^2

Propose a <u>new</u> configuration that achieves the above goals. Explain.

Give rough estimate of the ST and MT performance and power attainable in your suggested configuration.

Question 2 : Performance

This question will use the following benchmark to investigate the performance of several processors' microarchitectures:

```
int *index_arr[SIZE];
int values_arr[X];
int *end_addr = &index_arr[SIZE];
…
for(; i != end_addr; ++i) {
     int addr = index_arr[i];
     values_arr[addr]= 2* values_arr[addr];
}
```

The above C-like code is compiled into the following assembly:

	<Op>	<dst,src1,src2>	
Loop:			
A:	**LD**	**R1, 0(R2)**	**// R1 = Mem[R2], Unpredictable address**
B:	**LD**	**R3, 0(R1)**	**// R3 = Mem[R1]**
C:	**ADD**	**R3, R3, R3**	**// R3 = R3*2**
D:	**SW**	**0(R1), R3**	**// Mem[R1] = R3**
E:	**ADDI**	**R2, R2, 4**	**// increment pointer**
F:	**BNEQ**	**R2, R4, LOOP**	

Remarks:

1. **Array size is very big, i.e. the number of iterations is practically infinite.**
2. **All pointers in index_arr are unique and point to different addresses in value_arr (i.e., every iteration handles a different element in value_arr)**
3. **There is no overlap between index_arr and value_arr**

In Order multithreaded machine

A. Consider a simple In Order processor with 4 stage pipeline:
- Fetch, Decode, Execute, Write-back.
- The machine is simple scalar (one way)
- Memory is accessed in the Execution stage.
- The processor has a fully pipelined non blocking cache with miss penalty of *k* cycles.
- The cache can access only one load transaction on every cycle.
To compensate for cache misses, the machine is enhanced with fine-grain multithreading and the benchmark is divided into enough threads.
Switching thread has no performance overhead.
What is the minimal number of threads needed to achieve the maximal performance in the worst case?

Out of order machine

Consider an 8-way superscalar Out Of Order processor with 4 stages:
Fetch & Decode, Rename & Issue, Execute, Retire.
The cache can handle up to 8 accesses per cycle.
A data cache is being used with cache miss penalty of 100 cycles.
Cache hit time is 1 cycle, i.e., after 1 cycle in Execution stage the data is loaded from memory (no bubbles on cache hit).

B. Assuming that the ROB and issue buffer are of infinite size, an oracle branch predictor (perfect Branch Predicator) and no cache misses.
(1) Fill the attached table
(2) What is the IPC of the benchmark? (consider the dependencies within an iteration and among iterations)

No.		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	LD R1, 0(R2)																									
2	LD R3, 0(R1)																									
3	ADD R3, R3, R3																									
4	SW 0(R1), R3																									
5	ADDI R2, R2, 1																									
6	BNEQR2, R4, LOOP																									
7	LD R1, 0(R2)																									
8	LD R3, 0(R1)																									
9	ADD R3, R3, R3																									
10	SW 0(R1), R3																									
11	ADDI R2, R2, 1																									
12	BNEQR2, R4, LOOP																									
13	LD R1, 0(R2)																									
14	LD R3, 0(R1)																									
15	ADD R3, R3, R3																									
16	SW 0(R1), R3																									
17	ADDI R2, R2, 1																									
18	BNEQR2, R4, LOOP																									
19	LD R1, 0(R2)																									
20	LD R3, 0(R1)																									
21	ADD R3, R3, R3																									
22	SW 0(R1), R3																									
23	ADDI R2, R2, 1																									
24	BNEQR2, R4, LOOP																									
25	LD R1, 0(R2)																									
26	LD R3, 0(R1)																									
27	ADD R3, R3, R3																									
28	SW 0(R1), R3																									
29	ADDI R2, R2, 1																									
30	BNEQR2, R4, LOOP																									
31																										

(3) May performance be improved by using a wider machine? May it be enough to use just a narrower machine? Explain.

(4) May performance be improved by using a data (value) prediction scheme? If so, explain which instruction(s) should be predicted and how the performance is improved, and by how much.

C. Assume that 1 out of 500 accesses to memory by instruction A results in a cache miss, Cache miss penalty is 100 cycles (instruction B never misses).

(1) Will a reorder buffer of 128 instructions be able to hide the misses? What about 1000? Explain.

(2) Repeat section C. (1) when instruction B can miss the cache too. Consider the worst case.

(3) **Assume the ISA supports a prefetch instruction. Prefetch instruction is similar to a load, but has no destination and is used to bring the addressed memory into the cache. How can you re-write the program using prefetch to cover almost all cache misses? (Concrete code is not required here, just explain the main principles)**

(4) May loop unrolling be used to cover for the cache misses? Explain

D. **Discuss the potential of SMT to enhance the benchmark performance over a base system of OOO machine with ROB of 128 instructions. Cache misses may occur for both of the load instructions. Assume that processor resources (ROB, etc.) are smartly distributed among the threads**

(1) Refer to a configuration of two threaded SMT machine

(2) Let *n* be the number of supported threads in the SMT machine. How will benchmark performance be affected from the increment of *n*. (Assume that the ROB is proportionally big enough to support the threads)