21st CSEC – Past Year Paper Solution (2017 – 2018 Semester 1)
CE/CZ 3001 – Advance Computer Architecture

1)

a. Speedup = $\frac{1}{1-E+\frac{E}{S}}$

To further improve the speedup, we can parallelize the fraction of program across N cores (by a factor of N) if possible. However, according to Amdahl's law, the speedup via parallelism is limited by that sequential component of application. Thus, the maximum speedup we can achieve is $\frac{1}{1-E}$ if we can only enhance fraction E of the program.

b.

i. M1 frequency = 300 MHz
M2 frequency = 200 MHz

CPI	T1	T2
M1	2	4
M2	3	3

Execution time of M1 = $\frac{(2*10^9)+(4*10^9)}{300*10^6}$ = 20s

Execution time of M2 = $\frac{(3*10^9)+(3*10^9)}{200*10^6}$ = 30s

ii. Speedup of M1 over M2 = $\frac{Execution\ time\ of\ M2}{Execution\ time\ of\ M1} = \frac{30}{20} = 1.5$

c. Execution time = $\frac{800*10^3}{20+60} = 10 * 10^3\ s$
Since $P_{dyn} \propto f$ and $P_{static} \propto I_{leak}$,
New P_{dyn} = 60/2 = 30 Watt
New P_{static} = 0.95 * 20 = 19 Watt
Hence, new energy consumption = (30 + 19) * $10 * 10^3$ = 490 kJ

d. All RISC instructions is simple and can be executed in one clock cycle. Thus, it is easy to split the instructions across multiple cores for execution as there is no need to decompose the instruction into smaller task.

2)

a. Last address = 0x6FFFFFFC
Value of $ra = 0x60000000
For jump instruction, the first 4 bits of the address is fixed. Thus, after the last address, it will go back to the first instruction again.

b.

i.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
I1	IF	ID	EX	M	WB																
I2				IF	ID	EX	M	WB													
I3							IF	ID	EX	M	WB										
I4										IF	ID	EX	M	WB							
I5													IF	ID	EX	M	WB				
I6																IF	ID	EX	M	WB	
I1																				IF	ID

Hence, 1 iteration needs 19 cycles.
Total number of cycles = 19 * $\frac{396}{4}$ = 1881 cycles

ii.
Loop unrolling and reordering

Loop I1: LW $t0, 0($s0)
I4: ADDI $s0, $s0, 0x0004
I5: SUB $t3, $t2, $s0
I6: BNEZ $t3, Loop
I2: ADDI $t0, $t0, 0x0001
I3: SW $t0, -4($s0)

	1	2	3	4	5	6	7	8	9	10	11	12
I1	IF	ID	EX	M	WB							
I4		IF	ID	EX	M	WB						
I5			IF	ID	EX	M	WB					
I6				IF	ID	EX	M	WB				
I2					IF	ID	EX	M	WB			
I3						IF	ID	EX	M	WB		
I1							stall	IF	ID	EX	M	WB

Since 1 iteration needs 7 cycles,
Total number of cycles = 7 * $\frac{396}{4}$ = 693 cycles

3)

a.

i.

Sequence	N	N	T	N	T	N
Predict	T	N	N	N	N	N
Next state	00	00	01	00	01	00

Accuracy = 50%

ii. Accuracy = 4/6 * 100% = 66.67%
In this case, always not taken predictor is better.

b. Loop unrolling allows us to reduce CPI. For example, assuming we have a superscalar processor, one way is exclusively for load and store instructions and the second way can execute all instructions except load and store. For the following instruction, we can reduce CPI to below 1 using loop unrolling. Without loop unrolling, we will not be able to get CPI lower than 1 for this example due to RAW dependencies.

Loop: LW $t0, 0($s0)
ADDI $t2, $t2, -1
ADDI $s0, $s0, 4
XORI $t0, $t0, 0x0F0F
SW $t0, -4($s0)

BNE $t2, $zero, Loop

	Way - 1	Way – 2
Loop	LW $t0, 0($s0)	ADDI $t2, $t2, -2
	LW $t1, 4($s0)	ADDI $s0, $s0, 8
		XORI $t0, $t0, 0x0F0F
		XORI $t1, $t1, 0x0F0F
	SW $t0, -8($s0)	BNE $t2, $zero, Loop
	SW $t1, -4($s0)	

CPI = 8/9 = 0.889

c. Since $P_{dyn} \propto V^2 f$ and $f \propto V$ when threshold voltage V_{th} is small compared to V. To reduce energy, we need to reduce V, not *f*. Assuming we have a task which takes time T and dynamic power P for single core, then the energy consumption resulting from dynamic power is E = P * T. In the case of dual core, we can split the workload equally for each core and reduce the voltage V of each core by half, so the frequency *f* is reduced by half while the execution time remains T. The new energy consumption due to dynamic power is $new\ \mathrm{E} = \left(2 * \frac{Pdyn}{2^2 * 2}\right) * T = E/4$ which is a quarter of the initial energy consumption.

4)

a.

i.

Structure	Tag-bits	Index-bits	Offset-bits	Size of tag array
L1 – I cache	$36 - 7 - 7 = 22\ bits$	$\frac{2^{16}}{\frac{2^7}{4}} = 2^7\ sets \rightarrow 7\ bits$	$log_2 128 = 7\ bits$	$36\ bits$
L1 – D cache	$36 - 8 - 6 = 22\ bits$	$\frac{2^{15}}{\frac{2^6}{2}} = 2^8\ sets \rightarrow 8\ bits$	$log_2 64 = 6\ bits$	$36\ bits$

ii.

The size of a single-entry page table = $no.of\ entries \times entry\ size.$

Having 2^{36} bytes virtual addresses and 2^{12} bytes page size, we may see that the address space needs to be split into 2^{24} pages, hence there are 2^{24} entries.
To find the size of an entry, we must determine how many bits are used to store the information. As there are 2^{32} physical addresses divided into frames of size 2^{12} bytes. There are 2^{20} frames and we require 20-bits to store the frame number.
In addition, the question mentions that we require 5 additional bits to store dirty bit, reference bit, and permission bits.
This gives a total of 25 bits per entry, as the memory is byte-addressable, we can round this to 4 bytes per entry. (1 byte = 8 bits)

Hence, the single-level page table size is $2^{24} * 4 \text{ bytes} = 2^{26} \text{ bytes} = 64\text{MB}$.

b. Homogeneous computing is defined as interconnecting similar processing cores or units to build a high-performance computer (e.g. Ethernet Local Area Network (LAN)). On the other hand, heterogeneous computing refers to systems that use more than one kind of processor or cores (e.g. ARM, FPGA).
Homogeneous computing systems were found mainly on the desktop and in the data centers, whereas heterogeneous systems were found primarily in mobile phones.

Solver: CSEC Academic Committee