Hadassah Academic College

Department of Computer Science

Computer Architecture **Exercise 1**

Consider the following code fragment for a contemporary PC (all numbers given in decimal):

	Instruction	Action	Clock Cycles
	`MOV DWORD PTR [EBP-04],0`	`; i = 0`	1
	`MOV DWORD PTR [EBP-12],0`	`; k = 0`	1
	`MOV DWORD PTR [EBP-16],0`	`; l = 0`	1
`L1:`	`CMP DWORD PTR [EBP-04],100`		2
	`JGE L2`	`; break on i ≥ 100`	3
	`MOV EAX,[EBP-04]`	`; EAX ← i`	1
	`SHL EAX,1`	`; EAX ← 2 * EAX`	1
	`MOV [EBP-08],EAX`	`; j ← EAX`	1
	`ADD [EBP-12],EAX`	`; k ← k + EAX`	3
	`SUB [EBP-16],EAX`	`; l ← l - EAX`	3
	`INC DWORD PTR [EBP-04]`	`; i++`	3
	`JMP L1`	`; loop`	3
`L2:`	`RET`		3

Each register holds a 32-bit integer (a **DWORD** in Intel terminology).

1. Find the number of instruction cycles required to execute the program as written.

2. Registers **EBX** , **ECX**, and **EDX** are reserved. Variables can be allocated to registers **ESI** and **EDI** instead of main memory. Operations on registers require 1 CC instead of 2 or 3.

   Carry out the exchange.

   Find the number of instruction cycles required to execute the updated program.

   Find the relative speedup compared to the original program.

3. The CPU in a contemporary PC includes 8 additional registers **R0, R1, ... , R7**.

   Use these registers to improve the program.

   Find the number of instruction cycles required to execute the updated program.

   Find the relative speedup compared to the original program.

4. The program can be split into two threads with identical code except for the variable **i**.

   On a dual core processor, each thread runs in parallel (each thread on a separate core, so that both threads begin at the same time). To good approximation, the run time for each thread will be identical and the program finishes when one thread finishes.

   The thread for **CPU0** runs for **i = 0** and **i < 50**.

   The thread for **CPU1** runs for **i = 50** and **i < 100**.

   Find the number of instruction cycles required to execute the updated program.

   Find the relative speedup compared to the original program.

   The speedup is less than 2, despite the two cores. The clock cycles for the sequential version of question 3 can be split into two groups: those that run fewer times in the threaded version and those that run the same number of times.

   How many clock cycles are in each group?