7 Pipelining [70 points]

The following piece of code runs on an in-order pipelined processor as shown in the table (F: Fetch, D: Decode, E: Execute, M: Memory, W: Write back). Instructions are in the form "Instruction Destination,Source1,Source2/Immediate." For example, "ADD A, B, C" means A ← B + C.

	Cycles	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	MUL R5, R6, R7	F	D1	D2	E1	E2	E3	M	W								
2	ADDI R4, R6, 5		F	-	D1	E1	-	-	M	W							
3	MUL R4, R7, R8				F	D1	D2	E1	E2	E3	M	W					
4	ADD R5, R5, R6					F	-	D1	E1	-	-	M	W				
5	ADD R6, R7, R5							F	D1	-	-	-	E1	M	W		
6	ADD R7, R1, R4								F	-	-	-	D1	D2	E1	M	W

Use this information to reverse engineer the microarchitecture of this processor to answer the following questions. Answer the questions as precisely as possible with the provided information. If the provided information is not sufficient to answer a question, answer "Unknown" and explain your reasoning clearly.

(a) [10 points] What is the ALU's latency for an addition and for a multiplication, respectively?

Addition: 1 cycle for an addition (E1).

Multiplication: 3 cycles for a multiplication (E1, E2, E3).

(b) [10 points] Does this processor implement data forwarding? If so, between which pipeline stages? Explain your reasoning.

The processor implements data forwarding from *W* to *E1*.

(c) [10 points] The number of cycles in the decode stage dynamically varies between instructions. Explain why this might be the case. **Hint:** Register values are read from the register file in the decode stage.

All listed instructions require two operands for the ALU, which require up to two cycles to read from the register file. If one of the inputs is an immediate (e.g., instruction 2) or is forwarded from an earlier instruction, the register file has to be queried for only one input. Then, a shorter decode stage is sufficient.

(d) [10 points] What is the minimum number of register file read ports and write ports that this processor implements? Explain.

The register file has one read port and one write port.

Acording to the timeline in cycles 2 and 3, the decode stage operates in two cycles for an instruction that has two register operands. Also, acording to cycle 4, the decode stage takes one cycle for an instruction with one register operand. We conclude that the decode stage needs one cycle to decode and read each register operand which means the register file has one read port.

The register file has at least one dedicated write port since acording to cycle 12, the decode stage and the write back stage are both using the register file, and we are sure that both have been serviced by the register file within that cycle since the next cycle is not a stall for any of them.

(e) [15 points] Can we reduce the execution time of this code by enabling more read or write ports in the register file? Explain. If yes, what is the speedup compared to the baseline microprocessor assuming the changes do not impact clock frequency? Show your work.

Yes. Adding a new read port to the register file will enable the register file to service the decode unit in one cycle for any instruction with one or two register operands. The speedup is 16/13. Here is the new timeline:

	Cycles	1	2	3	4	5	6	7	8	9	10	11	12	13
1	MUL R5, R6, R7	F	D	E1	E2	E3	M	W						
2	ADDI R4, R6, 5		F	D	E1	-	-	M	W					
3	MUL R4, R7, R8			F	D	E1	E2	E3	M	W				
4	ADD R5, R5, R6				F	D	-	E1	-	M	W			
5	ADD R6, R7, R5					F	-	D	-	-	E1	M	W	
6	ADD R7, R1, R4							F	-	-	D	E1	M	W

(f) [15 points] Is it possible to run this code faster by adding more data forwarding paths to the original pipeline? If it is, explain how and calculate the speedup with respect to the original pipeline assuming the changes do not impact clock frequency. Otherwise, explain why it is not possible.

Yes, it is possible. Adding a forwarding path from *M* to *E1* can improve the performance. The speedup is 16/15. Here is a new timeline:

	Cycles	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	MUL R5, R6, R7	F	D1	D2	E1	E2	E3	M	W							
2	ADDI R4, R6, 5		F	-	D1	E1	-	-	M	W						
3	MUL R4, R7, R8				F	D1	D2	E1	E2	E3	M	W				
4	ADD R5, R5, R6					F	-	D1	E1	-	-	M	W			
5	ADD R6, R7, R5							F	D1	-	-	E1	M	W		
6	ADD R7, R1, R4								F	-	-	D1	D2	E1	M	W