Version AHG/2

EGT3
ENGINEERING TRIPOS PART IIB

Monday 11 May 2026 9.30 to 11.10

Module 4F14

COMPUTER SYSTEMS

*Answer not more than **two** questions.*

All questions carry the same number of marks.

*The **approximate** percentage of marks allocated to each part of a question is indicated in the right margin.*

*Write your candidate number **not** your name on the cover sheet.*

STATIONERY REQUIREMENTS
Single-sided script paper

SPECIAL REQUIREMENTS TO BE SUPPLIED FOR THIS EXAM
CUED approved calculator allowed
Engineering Data Book

10 minutes reading time is allowed for this paper at the start of the exam.

You may not start to read the questions printed on the subsequent pages of this question paper until instructed to do so.

You may not remove any stationery from the Examination Room.

1 (a) Explain why the latency of the arithmetic logic unit (ALU) is of paramount importance in computer hardware design. If the ALU were half as fast, estimate the impact on performance for a typical MIPS datapath (i) without and (ii) with pipelining. State any assumptions you make. [15%]

(b) Figure 1 shows the layout of a three-level, 8-bit carry-lookahead adder. The numbers to be added are the inputs $a_7 \ldots a_0$ and $b_7 \ldots b_0$, while the outputs $s_7 \ldots s_0$ are the resulting sum. Blocks A and B contain combinatorial logic circuits.

(i) Explain the purpose of the block A. What are the unlabelled outputs? [10%]

(ii) What are the inputs and outputs of the block B? Give sum-of-product logic expressions to show how the outputs are derived from the inputs. [20%]

(iii) The circuit in Fig. 1 does not compute the carry-out c_8. Which of the blocks should be modified, and how should this block be modified, to produce this signal? [10%]

(iv) For a generalised n-bit adder laid out as in Fig. 1, using carry-lookahead blocks that provide m carry-in signals ($m = 2$ in Fig. 1), how does the rectangular space requirement (i.e. width times height) scale with m and n? [10%]

(v) Distinct from the asymptotic space requirement is the asymptotic gate count. Explain why the total number of B blocks is given by

$$\sum_{i=1}^{k} \frac{n}{m^i} \text{ , where } k = \log_m n$$

and hence show that the asymptotic gate count scales linearly with n. [25%]

(vi) Discuss briefly the relative importance of asymptotic space requirements and gate counts. [10%]

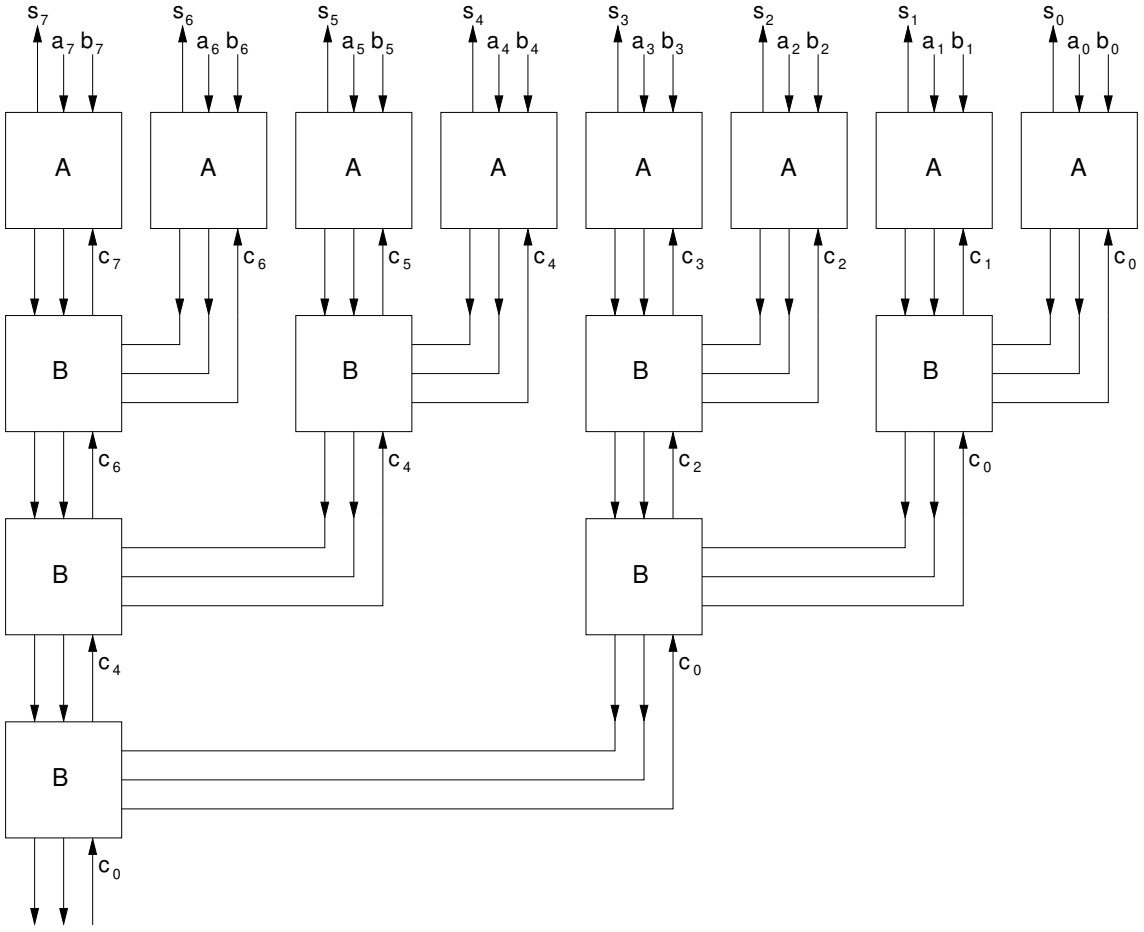
s7 a7 b7
s6 a6 b6
s5 a5 b5
s4 a4 b4
s3 a3 b3
s2 a2 b2
s1 a1 b1
s0 a0 b0
A
B
c7
c6
c5
c4
c3
c2
c1
c0

Fig. 1

2 (a) Explain what is meant by a *pipelined* datapath. What are pipeline *hazards*? [20%]

(b) The following series of MIPS instructions is executed on the pipelined datapath illustrated schematically in Fig. 2.

```
lw $8,0($9)          # $8 loaded with data at address $9+0
add $9,$9,$10        # $9 loaded with $9+$10
add $9,$9,$8         # $9 loaded with $9+$8
sw $9,0($10)         # $9 stored at address $10+0
```

If hazards are resolved by stalling the pipeline, how many clock cycles are required to execute the instructions (i) without data forwarding, and (ii) with data forwarding? [20%]

(c) Figure 3 shows an extension of the MIPS pipeline to support floating point arithmetic. Note how the floating point addition unit (A) is pipelined in two stages, the multiplication unit (M) is pipelined in four stages, while the division unit (DIV) is not pipelined but requires six clock cycles to execute. The IF, ID, MEM and WB stages are largely unchanged and can accommodate only one instruction at a time. Hazards are resolved preferentially by data forwarding and by stalling the pipeline if necessary. In the following series of instructions, the suffix `.f` denotes a floating point operation.

```
add.f $9,$9,$10      # $9 loaded with $9+$10
lw.f $8,0($11)       # $8 loaded with data at address $11+0
mult.f $12,$9,$8     # $12 loaded with $9x$8
lw.f $12,0($10)      # $12 loaded with data at address $10+0
```

(i) How many clock cycles are required to execute the instructions? [30%]

(ii) Note how the final two instructions both write `$12` with no intervening read. How likely is this situation to arise in practice? Explain how the hazard resolution would change were some other instruction to read `$12` between the two writes. [20%]

(iii) Beyond this code segment, what are the broader implications of the division unit not being pipelined? [10%]

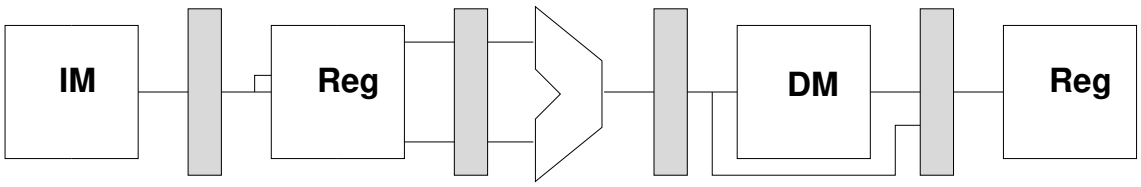
IM
Reg
DM
Reg

Fig. 2

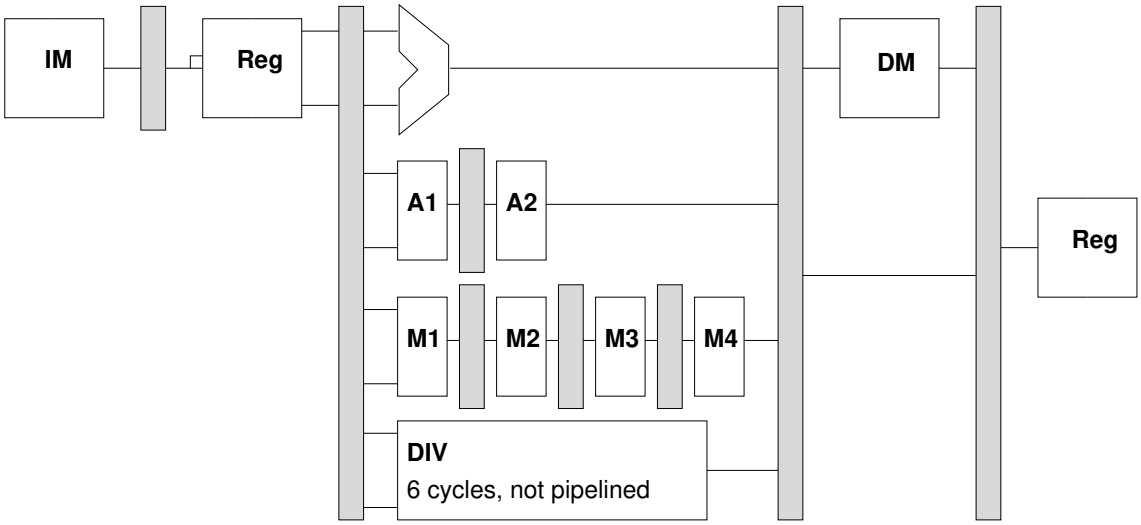
IM
Reg
DM
A1
A2
Reg
M1
M2
M3
M4
DIV
6 cycles, not pipelined

Fig. 3

3 (a) Explain why the inclusion of a cache, between the CPU and main memory, generally improves a computer's performance. Discuss briefly the relative advantages and disadvantages of direct mapped and set-associative caches. [20%]

(b) What requirements of a modern computer system motivate the adoption of a virtual memory system? [10%]

(c) Figure 4 shows the translation lookaside buffer (TLB) and one of the caches in a computer system. What is the page size? Is the TLB direct-mapped, set associative or fully associative? What about the cache? What is the cache block size? [20%]

(d) The cache in Fig. 4 is physically indexed, physically tagged (PIPT). An alternative is the virtually indexed, virtually tagged (VIVT) cache, which uses the virtual address for both the index and the tag. VIVT caches suffer from *aliasing*, whereby the same virtual address might map to different physical addresses (*homonyms*) or the same physical address might map to different virtual addresses (*synonyms*).

(i) Aliasing aside, what would be the main advantage of a VIVT cache over a PIPT cache? [10%]

(ii) What common use cases might give rise to the homonym and synonym problems? Suggest how the homonym problem might be mitigated in practice. [20%]

(iii) A third possibility is the virtually indexed, physically tagged (VIPT) cache. Explain why this might be a good compromise, especially if the index does not extend beyond address bit 11 for a virtual memory system with 4 KiB pages. For such an index, what is the maximum size of the cache? [20%]

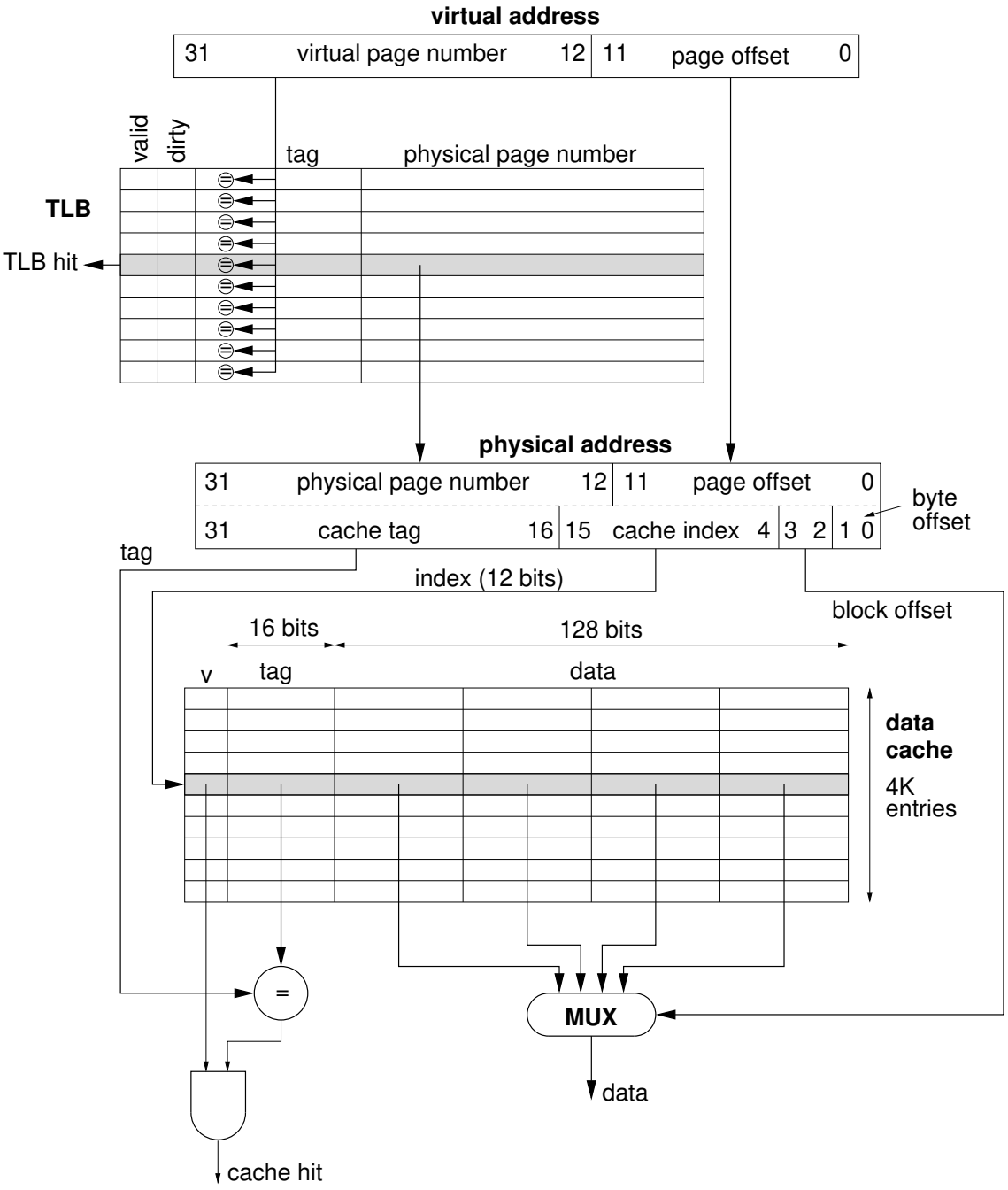

Fig. 4

END OF PAPER

THIS PAGE IS BLANK