



01 · Exam map

COURSE.JSON · BRIEF

ASSESSMENT	WEIGHT	WHAT MATTERS HERE
Final Exam	50%	Modules 1-4, 6-11; Inspira IIB Mode D; 36 questions / 50 marks
Mid-Semester Test	20%	Earlier modules; Module 5 belongs here, not the final
Assignments	20%	Practical grouping
Essays	8%	
Tutorials	2%	

"An A4-cheat sheet (typed or handwritten) can be brought into the exam hall." (COMPSCI 110 exam revision slides, Semester Two 2026)

SCOPE Final weighting: OS 8 marks; Compilers 7; Security 6; Turing 6; Networks 5; AI 5; Binary 4; Logic, Organisation and Assembly 3 each. Spend time in that order, but bank every mechanical mark.

02 · Binary toolkit

M1 · CH 1

234₁₀ = 11101010₂

1110 = E

1010 = A

4-bit groups → EA₁₆

3-bit groups → 11|10|010 = 352₈

Source redraw: group from the right; 4 bits per hex digit, 3 per octal digit.

POSITIONAL VALUE / SIZING
(d_n...d₀)_b = ∑ d_i · bⁱ
k distinct patterns need ⌊log₂k⌋ bits
N address bits select 2^N cells

TASK	PROCEDURE	CHECK
Decimal → binary	Repeatedly ÷2; record remainder; read bottom-up	Odd ends 1; even ends 0
Fraction → binary	Repeatedly ×2; record integer parts top-down	0.1875 ₁₀ = 0.0011 ₂
Shift	Left one place ×2; right one place ÷2	Bits falling off are lost
Fixed-width sum	Add right-to-left; 1+1 writes 0, carries 1	Overflow when result cannot fit

SIGNED REPRESENTATIONS

FORM	NEGATIVE	N-BIT RANGE / ZERO
Sign-magnitude	Flip sign bit	±(2 ⁿ⁻¹ -1); two zeros
One's complement	Invert all bits	±(2 ⁿ⁻¹ -1); two zeros
Two's complement	Invert+1	-2 ⁿ⁻¹ ...2 ⁿ⁻¹ -1; one zero

TWO'S-COMPLEMENT SUBTRACTION
a - b = a + (-b) = a + ~b + 1
Fix width → invert b + carry-in 1 → add → discard carry out.

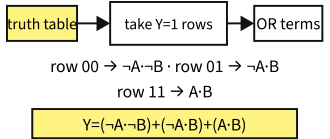
03 · Logic → circuit

M2 · CH 2

A	B	AND	OR	NAND	NOR
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	1	0	0

NOT flips 0↔1. NAND and NOR are *functionally complete*; tie their inputs together to make NOT. An n-input table has 2ⁿ rows.

DE MORGAN
~(A + B) = ~A · ~B
~(A · B) = ~A + ~B



Source redraw: SOP means one product for every true row; bar each input that is 0.

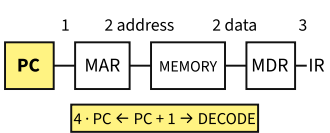
HARDWARE RULES

ELEMENT	INPUTS → OUTPUTS / RULE
Full adder	a _i ,b _i ,c _i → sum s _i and carry c _{i+1} ; chain carry to next bit
Subtract	Invert b and set initial carry-in 1
Multiplexor	2 ⁿ inputs, N selectors, 1 output
Decoder	N inputs, 2 ^N outputs
SR latch	00 hold; 10 set; 01 reset; 11 forbidden

EXAM MOVE From a supplied truth table: circle every output-1 row; write all variables in each AND term; bar only the 0 inputs; OR every term. Simplify only if asked.

04 · Stored-program machine

M3 · CH 3



Source redraw: fetch order is PC→MAR; cell→MDR; MDR→IR; increment PC; then decode and execute.

REGISTER	ROLE
PC	Address of next instruction
IR	Instruction now executing
MAR	Address accessed: the "where"
MDR	Data fetched/stored: the two-way "what"

Von Neumann: one memory holds instructions and data; subsystems are Memory, I/O, ALU and Control Unit; CPU = ALU + CU. Shared sequential traffic causes the bottleneck. Fetch preserves; store overwrites.

05 · Assembly / machine code

M4 · CH 4

one 16-bit instruction word

1101 0000 0000 1001

opcode · IN address 9 · D009₁₆
Source redraw: high 4 bits = opcode; low 12 = address. One instruction occupies one cell.

INSTRUCTION	OPCODE	EFFECT
LOAD X	0000	CON(X) → R
STORE X	0001	R → CON(X)
ADD X	0011	R + CON(X) → R
SUBTRACT X	0101	R - CON(X) → R
COMPARE X	0111	Set GT/EQ/LT flags
HALT	1111	Stop; address 000...0

Use supplied Figure 6.5: **lookup → 12-bit address → concatenate → hex nibbles**. Arithmetic uses one register R and 16-bit two's complement.

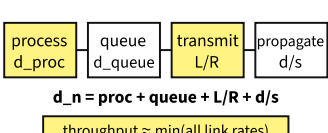
DESK-CHECK / ASSEMBLE

- After every line record R, changed memory and GT/EQ/LT.
- .BEGIN/.END/.DATA are directives; LABEL : .DATA θ binds an address.
- Pass 1 builds label→address; pass 2 inserts opcodes/addresses and emits words. This resolves forward labels.

TRAP Do not "remember" an opcode when Figure 6.5 is supplied. Read it; then keep opcode in the top nibble and pad the operand to exactly 12 bits.

06 · Network calculations

M6 · CH 6



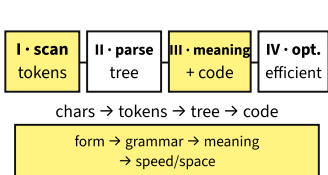
Source redraw: transmission depends on bits/rate; propagation depends on distance/speed.

USE WHEN THE QUESTION GIVES
packet length L and rate R: d_{trans} = L/R
distance d and signal speed s: d_{prop} = d/s
s = 2×10⁸ m/s; convert Mbit-bits and km-m first.

CONTRAST	A	B
Switching	Packet: dynamic; queue/drop	Circuit: fixed reserved rate
Path action	Forwarding: local table	Routing: network path
Delay test	Bigger packet ↑ d _{trans}	Longer link ↑ d _{prop}

07 · Compiler decision chain

M7 · CH 7 · 7 MARKS



Source redraw: the phase tells you which error or artefact is being tested.

PHASE	INPUT → OUTPUT	DOES / DOES NOT
I Lexical	characters → tokens	Longest valid token; discards whitespace/comments; does not check syntax
II Parsing	tokens + grammar → parse tree	Checks syntactic form using BNF productions
III Semantic + generation	tree + records → instructions	Types, names, addresses and meaning; emits code
IV Optimisation	code → improved code	Improves time or space without changing meaning

TOKENISE UNDER PRESSURE

- Scan left→right. Take the **longest valid token**.
- Classify: keyword; identifier/symbol; operator; bracket; number; literal; terminator/symbol.
- Discard spaces and comments. = is one operator token and differs from assignment =.
- Count the list. An illegal arrangement still tokenises; the parser rejects its grammar later. A misplaced eLse remains a keyword.

SOURCE	CLASS
if	keyword
()	brackets
a, b, answer	symbols / identifiers
==, =	operators; each is one token
42	number
;	symbol / terminator

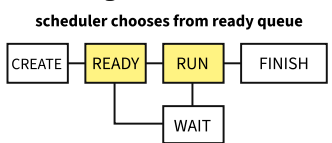
BNF: productions expand non-terminals into terminals/non-terminals. A parse tree is valid only if its leaves match the token stream and its root reaches the grammar's goal symbol. Syntax asks whether the form is legal; semantics asks whether that legal form makes sense.

ASSEMBLER / COMPILER SPLIT
assembly – one machine instruction per source instruction; table lookup + labels
high-level – many machine instructions; Lexical + syntax + meaning

MARK CHECK If the prompt says "scanner", do not repair syntax. If it says "parser", use the supplied grammar. If it says "semantic", discuss meaning/type/name records, not punctuation.

09 · OS scheduling

M8 · CH 8 · 8 MARKS



Source redraw: a process is a program in execution; waiting means blocked, not deadlocked.

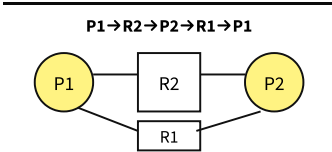
POLICY	RULE	CONSEQUENCE
FCFS	Longest-waiting process runs to completion	Simple; long first job causes convoy effect
Round-Robin	Run ≤ quantum; unfinished job is preempted to queue tail	Responsive/fair; more switches

ROUND-ROBIN METHOD
1 build exact timeline by queue order
2 read each completion time
3 if all arrive at 0: waiting = completion - burst
4 average = ∑ waiting / process count
CONTEXT-SWITCH COST
CPU efficiency = slice / (slice + switch time) × 100%
larger slice ↑ efficiency but ↓ responsiveness; tiny slice wastes time switching.

A context switch saves PC, registers and resource handles, then loads the next process. Useful work pauses. If arrival times are not all zero, do not use the shortened waiting formula without accounting for arrival.

OS managers: process, memory, device, file and network managers plus the command interpreter. The ready queue holds runnable processes; the waiting queue holds processes blocked on an event or I/O.

DEADLOCK DECISION



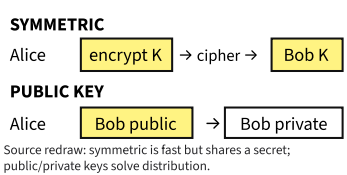
Source redraw: process→resource means waiting; resource→process means allocated.

RAG TEST	CONCLUSION
One unit per resource + cycle	Deadlock
Multiple units + cycle	Necessary, not sufficient; check whether any process can still finish
No cycle	No deadlock

TRAP A process merely waiting for I/O is not deadlocked. Deadlock is permanent circular waiting: each holds something the next needs and none can proceed.

10 · Cryptography chooser

M9 · CH 9 · 6 MARKS



Source redraw: symmetric is fast but shares a secret; public/private keys solve distribution.

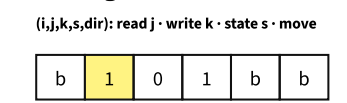
AXIS	CHOOSE / REMEMBER
Processing	Block; fixed block/diffusion. Stream: elementwise.
Keys	Symmetric: one shared secret. Asymmetric: public/private pair.
Systems	DES 56/64; 3DES ≈168; AES keys 128/192/256, block 128. RSA/Diffie-Hellman are public-key.
Certificate	Trusted binding: identity ↔ public key.

MATRIX BLOCK CIPHER
A=0...Z=25; row vector V
encrypt C = V·M mod 26
decrypt V = C·M' mod 26, where M·M' = I mod 26

Caesar: 25 shifts + preserved frequencies → brute force/frequency attack. Matrix blocks add diffusion.

11 · Turing trace

M10 · CH 10 · 6 MARKS



head starts on leftmost non-blank · state 1
Source redraw: tape is infinite both ways; it is input, working memory and output.

- Record state & symbol; choose the unique matching rule.
- Write field 3; go to state 4; move field 5.
- No match ⇒ **HALT**; repeating a configuration indicates a loop.

Valid: no duplicate (state,symbol) left sides. Church-Turing: any effectively computable function is Turing-computable; thesis, not theorem.

12 · AI concept distinctions

M11 · CH 11 · 5 MARKS

CONCEPT	DISCRIMINATOR
Turing test	Text imitation; judge cannot reliably distinguish person/machine.
Winograd	Pronoun needs commonsense; wording can flip referent.
Neural net	Layered units + weighted links.
Back-propagation	Error moves backward to adjust weights; no arithmetic here.

TRAP Turing tests behaviour, not inner mechanism; Winograd tests meaning, not grammar alone.

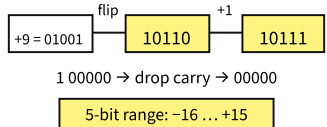


Full guide + ask Sia about COMPSCI 110

13 · Worked: -9 in 5-bit two's complement

M1 · WORKED EXAMPLE

Question: represent -9, verify with +9, and state the signed range.



Every value is width-sensitive. The leading 1 is a sign bit only because the width is fixed.

- +9: 01001₂.
- Invert: 10110₂.
- Add 1: 10111₂, so this encodes -9.
- Verify: 01001 + 10111 = 1 00000; discard carry → 0. Range = -2⁴...2⁴-1 = -16...+15.

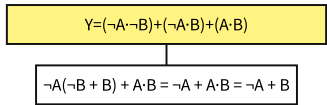
ANSWER
-9₁₀ = 10111₂ in 5-bit two's complement.

TRAP 10110 is one's complement, off by one. The range is asymmetric: -16 exists but +16 does not.

14 · Worked: SOP from a table

M2 · WORKED EXAMPLE

A	B	Y	PRODUCT WHEN Y=1
0	0	1	¬A¬B
0	1	1	¬AB
1	0	0	—
1	1	1	AB



The direct SOP earns the construction marks; simplification is a cross-check unless requested.

- Select rows 00, 01, 11 because Y=1.
- Write one AND term per row, barring each 0 input.
- OR all terms: (¬A¬B) + (¬AB) + (AB).
- Optional check: first two terms = ¬A; result simplifies to ¬A + B and reproduces 1,1,0,1.

TRAP Do not bar the 1s; bar the 0s. Do not omit an output-1 row. A combinational circuit depends only on current inputs; an SR latch stores state.

15 · Binary micro-checks

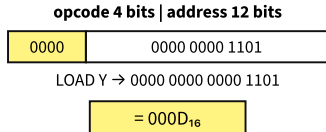
M1 · BODY TEXT

PROMPT	CORRECT RESULT / REASON
0101 ₂ + 0011 ₂	1000 ₂ ; carry right-to-left
0.1001 ₂	½ + ⅙ = ⅘
1101 ₂ < 3	1101000 ₂ ; >8
4-bit +7 +1	0111 → 1000 = -8 in two's complement

16 · Worked: assembly → machine

M4 · WORKED EXAMPLE

Given: Y at address 13; THREE at 14; X at 12. Translate LOAD Y · ADD THREE · STORE X · HALT.



Source redraw: pad the address, concatenate, then read four-bit hex nibbles.

SOURCE	16-BIT BINARY	HEX
LOAD Y	0000 0000 0000 1101	000D
ADD THREE	0011 0000 0000 1110	300E
STORE X	0001 0000 0000 1100	100C
HALT	1111 0000 0000 0000	F000

- Lookup opcodes: LOAD 0000; ADD 0011; STORE 0001; HALT 1111.
- Write 13, 14, 12 as 12-bit addresses; no-operand HALT uses zeros.
- Join 4+12 bits. Group into nibbles only after the 16-bit word is complete.

TRAP Addresses are values 13, 14, 12, not the values stored in Y, THREE and X.

17 · Worked: field sizing

M3 · WORKED EXAMPLE

Machine: 16-bit IR, 16 instructions, 16 registers. Can OPCODE REG1 REG2 REG3 fit?

opcode 4 bits	R1 4	R2 4	R3 4
---------------	------	------	------

4 + 4 + 4 + 4 = 16 · exact fit

Different questions seek different fields: instruction count → opcode; register count → register field.

- 16 operations: 2⁴=16 → opcode = 4 bits.
- 16-bit word - 4 = 12 operand bits.
- 16 registers: 2⁴=16 → each register field = 4 bits.
- 4-bit opcode + three 4-bit registers = 16 bits: fits exactly.

ANSWER
opcode 4 bits; operands 12 bits total; one register 4 bits; three-register format fits.

18 · Fetch trace scaffold

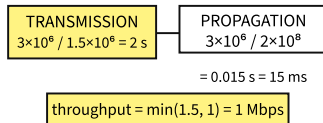
M3 · BODY TEXT

START	MICRO-STEP	AFTER
PC=120	PC → MAR	MAR=120
cell 120 = LOAD 45	FETCH	MDR=LOAD 45
MDR set	MDR → IR	IR=LOAD 45
PC=120	PC ← PC+1	PC=121
cell 45=7	execute LOAD	R=7

19 · Worked: link delay + bottleneck

M6 · WORKED EXAMPLE

Given: L=3 Mbit; R=1.5 Mbps; distance=3000 km; s=2×10⁸ m/s; later link=1 Mbps.



Write units into the fraction: bits cancel bits/s; metres cancel m/s.

- d_{trans} = L/R = 3×10⁸ bits / 1.5×10⁶ bits/s = 2 s.
- 3000 km = 3×10⁶ m. d_{prop} = d/s = 3×10⁶ / 2×10⁸ = 0.015 s = 15 ms.
- Nodal delay also includes processing and queueing.
- Path throughput = minimum link rate = 1 Mbps.

IF DOUBLED	D_TRANS	D_PROP
Packet length L	×2	unchanged
Rate R	÷2	unchanged
Distance d	unchanged	×2

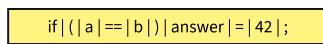
Queueing varies with congestion; processing checks the packet and selects its output link.

TRAP Throughput is never the sum of link rates. Improving a non-bottleneck link changes nothing.

20 · Worked: lexical analysis

M7 · WORKED EXAMPLE

if (a = b) answer = 42;



keyword · bracket · symbol · operator · symbol
bracket · symbol · op · number · ;

TOTAL = 10 tokens

Longest-match rule keeps == together; spaces do not become tokens.

TOKEN(S)	CLASS
if	keyword
(and)	bracket
a, b, answer	symbol / identifier
= and =	operators, each one token
42	number
;	symbol / terminator

- Ignore whitespace/comments.
- Scan left-to-right, longest token first.
- Classify each token; count = 10.
- Do not claim lexical analysis proves valid syntax; only parsing does.

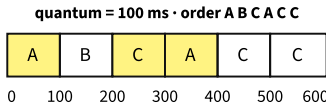
COMPILER TRAP TABLE

CLAIM	VERDICT
Comments are tokens	No; scanner discards them

21 · Worked: Round-Robin

M8 · WORKED EXAMPLE

Given: all arrive at 0; A=200 ms, B=100 ms, C=300 ms; quantum=100 ms; queue A,B,C; ignore switching.



Source redraw: B completes at 200; A at 400; C at 600.

PROCESS	FINISH	BURST	WAIT = FINISH-BURST
A	400	200	200 ms
B	200	100	100 ms
C	600	300	300 ms

ANSWER
average waiting = (200+100+300)/3 = 600/3 = 200 ms.

- Take at most one quantum from queue head.
- If unfinished, append to queue tail; if finished, remove.
- Read completion times only after the timeline is correct.
- For simultaneous arrival, subtract burst from completion.

TRAP B finishes after its first slice; do not return it to the queue. C takes consecutive final slices only because nobody else remains.

22 · Scheduling + deadlock checks

M8 · TOPIC MAP

choose the time quantum



efficiency = slice/(slice+switch) × 100%

The quantum is a responsiveness/overhead trade-off, not a universally maximised value.

QUESTION CUE	METHOD / EXACT CONDITION
"Runs to completion in arrival order"	FCFS
"Fixed time slice, preempt"	Round-Robin
Useful-work fraction	slice/(slice+switch)
RAG single-unit cycle	Deadlock
RAG multi-unit cycle	May deadlock; test whether anyone can finish

RESOURCE-ARROW GRAMMAR

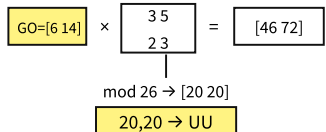
process → resource	process requests / waits
resource → process	resource unit allocated / held

State order: CREATE → READY → RUN → FINISH, with RUN ↔ WAIT when I/O blocks/unblocks. Ready processes can run; waiting processes cannot until their event completes.

23 · Worked: matrix cipher GO → UU

M9 · WORKED EXAMPLE

Given: A=0...Z=25; V=[6 14] for GO; M=[[3,5],[2,3]]; arithmetic mod 26.



Multiply first, reduce each entry modulo 26, then map numbers back to letters.

- G=6 and O=14, so V=[6 14].
- V·M = [6·3+14·2, 6·5+14·3] = [46,72].
- 46 mod 26=20; 72 mod 26=20 → [20,20].
- 20=U, so ciphertext=UU. Decrypt with M': [20 20]·M' mod 26=[6 14]=GO.

ANSWER
C = UU; inverse condition M·M' = I mod 26.

TRAP Keep row-vector orientation V·M. Both outputs use both input letters: that diffusion is the point.

24 · Worked: Turing complement

M10 · WORKED EXAMPLE

Rules: (1,0,1,1,R) and (1,1,0,1,R). Start state 1 on leftmost symbol of ...b 1 0 1 b...

read 1/write 0/R · read 0/write 1/R



next symbol is blank b · no (1,b) rule

HALT · output 010

Source redraw: one transition per bit; the missing blank rule is what stops the machine.

STEP	READ/ACTION	TAPE FRAGMENT AFTER
1	1→0, R	0[0]1
2	0→1, R	01[1]
3	1→0, R	010[b]
4	No (1,b) rule	halt; 010

ANSWER
101 → 010, then halt on blank.

TRAP Tuple field 3 is the symbol written; field 4 is next state. The head move occurs after the write/state change.

25 · Security fast recall

M9 · TOPIC MAP

PROMPT	ANSWER DISCRIMINATOR
Bulk data, speed	Symmetric, e.g. AES
Secure key distribution	Public-key / Diffie-Hellman
Authenticate public key	Digital certificate
Preserved letter frequency	Caesar/substitution weakness

26 · Worked: Winograd reasoning

M11 · WORKED EXAMPLE

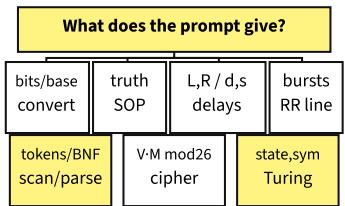
"The trophy doesn't fit in the suitcase because it is too big."

- Grammar permits trophy or suitcase.
- Commonsense says the too-big object fails to fit: "it" = trophy.
- Change "big" to "small" and the referent flips to suitcase: meaning, not syntax, resolves it.

Turing test: a text judge cannot reliably distinguish machine from person; this tests behaviour, not mechanism. **Neural net:** weighted, layered units; forward pass predicts; back-propagation sends error backward to adjust weights. No numeric calculation here.

27 · Method chooser

ALL FINAL MODULES



One cue selects one mechanical procedure. Write the setup before doing arithmetic or tracing.

WRITE THIS BEFORE WORKING
binary: fix width · circuit: mark Y=1 rows · machine code: 4/12 fields · network: units · scheduler: queue · cipher: V·M · Turing: state/head.

28 · Final error scan

FAQ + STUDY STRATEGIES

BEFORE SUBMITTING	COMMON WRONG TURN
State bit width and range	Symmetric two's-complement range
Drop carry beyond fixed width	Report a fifth bit
Bar 0s in SOP	Bar the 1s
Fetch before decode	Scramble PC/MAR/MDR/IR
Keep units visible	Mix Mbit, bit, km, m
Tokenise longest match	Split == or count comments
Timeline before waiting	Guess completion times
Cycle rule states resource units	Call any cycle a deadlock
Matrix orientation + mod	Reverse M·V or forget mapping
Tuple fields 3/4 distinct	Swap write and next state

LAST PASS Mechanical questions reward visible method. Preserve the fixed width, operand order, units, queue order, matrix orientation and head position so a small slip cannot propagate unseen.