A-LEVEL COMPUTER SCIENCE • EXERCISE SHEET

4.3 Bit manipulation

Name: _____ Class: _____ Date: _____ **Total: 19 marks**

Objective

Build the skills to answer exam questions on **bit manipulation** 位操作—binary shifts and using bit masks to monitor and control a device.

You must be able to:

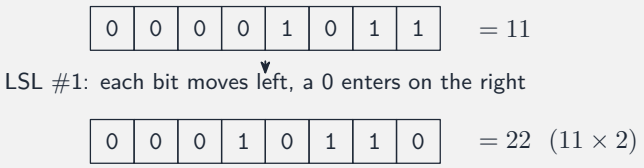
- perform **logical binary shifts** (left and right) and state their effect on the value
- use a **mask** 掩码 to **set**, **clear**, **toggle** and **test** a single bit

1 Worked examples

Study these first. Each one shows the method for a question type used later —follow the steps and you can do the Practice and Exam-style questions yourself.

■ Binary shifts

A **logical shift** moves every bit left or right, filling the new positions with 0.



set bit 2		clear bit 6		toggle bit 3	
	01001000		01001000		01001000
OR	00000100	AND	10111111	XOR	00001000
=	01001100	=	00001000	=	01000000

Bit masking with AND/OR isolates or sets chosen bits

- **left shift** by $n \rightarrow \times\ 2^n$ (for an unsigned number).
- **right shift** by $n \rightarrow$ **integer** $\div\ 2^n$.
- An **arithmetic** right shift keeps the **sign bit**, so a negative signed number stays negative.

- A **cyclic shift** 循环移位 (rotate) feeds the bit that falls off one end back in at the other end, so no bits are lost.

■ **Bit masks (monitor and control)**

A device often uses **one bit per signal** (bit n = LED n). Combine the register with a **mask** using a logic operation:

Goal	Operation	Mask (for bit n)
set bit n to 1	`R OR mask`	bit $n = 1$, rest 0
clear bit n to 0	`R AND mask`	bit $n = 0$, rest 1
toggle bit n	`R XOR mask`	bit $n = 1$, rest 0
test bit n	`R AND mask`, then check 0	bit $n = 1$, rest 0

Example —turn on bit 0 of `00000100` without changing the rest: `00000100 OR 00000001 = 00000101`.

2 Practice

Now apply the methods above.

2.1 Perform a **logical shift left by 1** on `00000101`, and give the denary value before and after. [2]

2.2 State the arithmetic effect of a **logical shift right by 1** on an unsigned number. [1]

2.3 Give the mask and the operation needed to **set bit 2** of a register to 1, leaving the other bits unchanged. [2]

2.4 A logical shift **left by 3** multiplies an unsigned number by what? [1]

3 Exam-style questions

3.1 (a) Perform a logical shift **left by 2** on `00000110`, and give the denary result. [2]

(b) State the effect this shift has on the value. [1]

3.2 An 8-bit register controls 8 LEDs (bit n = LED n).

(a) Give the mask and operation to turn **on only LED 0**, leaving the others unchanged. [2]

(b) Give the mask and operation to **test whether LED 3 is on**. [2]

3.3 Explain why an **arithmetic** right shift, not a logical one, is used when halving a **negative** signed number. [2]

3.4 State the difference between a **logical** shift and a **cyclic** shift. [2]

4 Go further

You are now ready for the real exam questions on this subtopic. Open the **4.3 Bit manipulation** past-paper sheet in the Library, or try these in **Practice mode**:

- 9618/11 J24 —Q4 (binary shifts and bit operations)
- 9618/12 J25 —Q7 (bit manipulation)
- 9618/13 J25 —Q7 (logical shifts)
- 9618/13 N24 —Q8 (shifts and masks)

Solutions

2.1 `00000101` (5) → `00001010`; denary before **5**, after **10**.

2.2 It **halves** the number (integer ÷ 2).

2.3 Mask `00000100`, operation **OR**: `R OR 00000100`.

2.4 By **8** (that is 2^3).

3.1 (a) `00000110` → `00011000`; denary **24**.
(b) It **multiplies the value by 4**.

3.2 (a) Mask `00000001`, operation **OR** (`R OR 00000001`).
(b) Mask `00001000`, operation **AND** (`R AND 00001000`); if the result is non-zero, LED 3 is on.

3.3 A logical shift puts a **0** into the top bit, which would make a negative number look positive; an arithmetic shift **copies the sign bit**, so the number stays negative.

3.4 A logical shift fills the vacated end with **0** and the bit shifted off the other end is **lost**; a cyclic shift feeds that bit **back in** at the other end, so no bits are lost.