PrepWithTee

NOTES

Data Representation

Cambridge IGCSE Computer Science (0478)

The full notes for this chapter: the topic taught from the basics, then how it is examined - question types, marking points, keywords and real past-paper questions solved - with practice to finish.

Contents

1	Number systems: binary, denary and hexadecimal	19 min
2	Hexadecimal uses, binary addition and overflow	15 min
3	Logical shifts and two's complement	13 min
4	Text, sound and images	14 min
5	Data storage units, file sizes and compression	15 min

Each note has two parts. **Part 1** teaches the topic from the basics. **Part 2** shows how it is examined: the types of question, what the examiner gives marks for, the keywords to use, and real past-paper questions solved with the official mark scheme. The answers to every practice question are at the end of each note.

NOTE 1

Number systems: binary, denary and hexadecimal

1 In this note

SYLLABUS 1.1 NUMBER SYSTEMS

- 1.1.1 Explain **how and why computers use binary** to represent all data.
- 1.1.2 Understand the **denary**, **binary** and **hexadecimal** number systems.
- 1.1.2 Convert positive whole numbers between denary, binary and hexadecimal, in every direction (up to 16 bits).
- 1.1.2 Work out the largest number a register can hold and the fewest bits a number needs.

Next: Hexadecimal uses and binary addition.

PART 1

Learn the topic

The notes: every idea from the start, with our own examples.

2 What you should already know

Place value in denary

In 365 the 3 is worth 300, the 6 is worth 60 and the 5 is worth 5. Each column is worth **ten times** the column to its right: 1, 10, 100, 1000.

Quick check: what is the 4 worth in 4 072?

4000 (the thousands column).

Powers of 2

Keep doubling from 1: 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 … You will use these in every question on this page, so learn them up to 128 (and 2048, 4096 … for 12 and 16 bits).

Quick check: what comes after 128?

256.

3 Why computers use binary

Inside a computer, data is processed by millions of tiny electronic **switches** called transistors, wired together into **logic gates**. A switch can only be in one of **two states**: on or off. So a computer can only hold two values, which we write as **1 and 0**.

That is why **every** kind of data has to be turned into binary before a computer can process or store it: numbers, text, images, sound, and the program instructions themselves. Binary values are held in **registers** (small stores inside the processor) and in memory.

> **Binary** is a **base 2** number system: it uses only the two digits **0 and 1**. One binary digit is called a **bit**.

> Why binary, in one sentence: **computers are made of switches / logic gates that have only two states, so they can only process 1s and 0s.**

4 Denary, binary and hexadecimal

A number system's **base** is how many different digits it uses. Every system works with place value: each column is worth the base times the column to its right.

System	Base	Digits	Column values (right to left)	Who uses it
Denary	10	0 to 9	1, 10, 100, 1000 …	people, every day
Binary	2	0 and 1	1, 2, 4, 8, 16, 32, 64, 128 …	the computer itself
Hexadecimal (hex)	16	0 to 9, then A to F	1, 16, 256, 4096	programmers: a short way to write binary

In hex the letters stand for the numbers 10 to 15, because one column must hold a single digit:

Denary	0	1	2	3	4	5	6	7	8	9	10
Hex	0	1	2	3	4	5	6	7	8	9	A
Binary	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010

EXAM TIP

You do not need to memorise the binary row: in the exam, count up in four columns 8 4 2 1. Two worth knowing by heart:

ANSWER

A = 10 = 1010 and F = 15 = 1111.

5 Binary to denary

An 8-bit binary number has these column (place) values. A 1 means "include this column", a 0 means "leave it out":



1001 1100 = 128 + 16 + 8 + 4 = 156

GENERAL METHOD **Binary to denary**

1. Write the place values over the bits: 1 under the **rightmost** bit, then 2, 4, 8 … going left.
2. Add the place values that have a 1 under them.
3. Check: the answer is odd only if the last bit is 1, and 8 bits can never give more than 255.

EXAMPLE **Convert 0010 1101 to denary.**

Add the place values of the 1s.

32 + 8 + 4 + 1

ANSWER **45**

TRY IT YOURSELF

1. Convert 0101 0101 to denary.
2. Convert 1111 1111 to denary.

3 Convert 1100 1000 to denary.

6 Denary to binary

Two methods give the same answer. Use whichever you find quicker; the mark scheme accepts both.

Method A: subtract place values

GENERAL METHOD **Take away the biggest place value that fits**

1. Write the place values 128 64 32 16 8 4 2 1.
2. Start at 128. Does it fit into what is left? Yes: write 1 and subtract it. No: write 0.
3. Move to the next place value and repeat until you reach 1.
4. Read the bits from left to right.

EXAMPLE **Convert 156 to 8-bit binary.**

Take away the biggest place value that fits, each time.

128 fits: 1, and 156 − 128 = 28 left

64? no: 0. 32? no: 0. 16 fits: 1, and 28 − 16 = 12 left

8 fits: 1, 4 left. 4 fits: 1, 0 left. 2? no: 0. 1? no: 0

ANSWER **1001 1100**

Method B: divide by 2

GENERAL METHOD **Keep dividing by 2 and read the remainders upwards**

1. Divide the number by 2. Write down the answer and the remainder (0 or 1).
2. Divide the answer by 2 again. Keep going until the answer is 0.
3. The **first** remainder is the **rightmost** bit: read the remainders from the bottom up.
4. Add 0s on the left to make 8 bits (or the number of bits asked for).

Divide	Answer	Remainder
156 ÷ 2	78	0
78 ÷ 2	39	0
39 ÷ 2	19	1
19 ÷ 2	9	1
9 ÷ 2	4	1
4 ÷ 2	2	0
2 ÷ 2	1	0
1 ÷ 2	0	1

Reading upwards: 1001 1100.

✗ COMMON MISTAKE

If the question says **8-bit**, the answer must have exactly 8 bits: 14 is **0000 1110**, not 1110. Leading zeros never change the value, but they show the size of the register.

TRY IT YOURSELF

1. Convert 50 to 8-bit binary.
2. Convert 102 to 8-bit binary.

3 Convert 221 to 8-bit binary.

7 How big a number can a register hold?

A **register** has a fixed number of bits. With all bits set to 1 you get the largest number it can hold.

Bits	Smallest	Largest	How many different values
4 (a nibble)	0	15 (1111)	16
8 (a byte)	0	255 (1111 1111)	256
12	0	4095 (FFF in hex)	4096
16	0	65 535 (FFFF in hex)	65 536

Largest value

n bits hold up to $2^n - 1$: 8 bits → $2^8 - 1 = 255$.

Fewest bits for a number

Keep doubling until the place value is **bigger** than the number, then count the place values you used. 301: 256 is not bigger, 512 is → 1, 2, 4 ... 256 is **9 bits**.

Fewest bits for a hex number

Each hex digit is exactly 4 bits, so count the digits × 4: A4D has 3 digits → **12 bits**.

TRY IT YOURSELF

1 What is the largest denary number an 8-bit register can store?

2 What is the least number of bits needed to store the denary number 200?

3 What is the least number of bits needed to store the hexadecimal number 3F2?

8 Converting with hexadecimal

The key fact: **one hex digit is exactly 4 bits** (a nibble), because $16 = 2^4$. So binary and hex convert by **grouping in fours**; no adding up of big numbers is needed.

Binary to hex

GENERAL METHOD **Binary to hexadecimal**

1. Split the bits into groups of 4 **starting from the right**. If the left group is short, add 0s in front.
2. Work out each group with the place values 8 4 2 1 (10 to 15 become A to F).
3. Write the hex digits in the same order.

	8	4	2	1	8	4	2	1	8	4	2	1
Binary	1	0	1	1	0	0	1	1	1	0	1	0
Hex				B				3				A

1011 0011 1010 → B 3 A. Each group of four is worked out on its own.

Hex to binary

GENERAL METHOD **Hexadecimal to binary**

1. Turn **every** hex digit into exactly 4 bits, zeros included.
2. Write the groups side by side.

EXAMPLE **Convert A04 to binary.**

Four bits for every digit, even the 0.

A = 1010

0 = 0000

4 = 0100

ANSWER **1010 0000 0100**

✗ COMMON MISTAKE

Dropping a 0000 group is the classic mistake: A04 is 1010 **0000** 0100 (12 bits), not 1010 0100.

Hex to denary

Method A: hex place values

GENERAL METHOD **Multiply by 1, 16, 256**

1. Hex place values from the right are 1, 16, 256 (and 4096).
2. Multiply each digit (A = 10 … F = 15) by its place value.
3. Add them up.

EXAMPLE **Convert C0F to denary.**

Multiply each digit by its place value.

C = 12 in the 256s: 12 × 256 = 3072

0 in the 16s: 0

F = 15 in the 1s: 15

ANSWER **3072 + 0 + 15 = 3087**

Method B: through binary

GENERAL METHOD **Go through binary**

1. Turn each hex digit into 4 bits.
2. Convert the binary to denary by adding place values.

EXAMPLE **Convert F0 to denary.**

Through binary.

F0 = 1111 0000

128 + 64 + 32 + 16

ANSWER **240**

✗ COMMON MISTAKE

F0 is **not** "150" (writing 15 and 0 side by side). Every digit must be multiplied by its place value: 15 × 16 + 0 = 240.

Denary to hex

Method A: through binary (safest)

GENERAL METHOD **Go through binary**

1. Convert the denary number to binary.
2. Group the bits in fours from the right.
3. Turn each group into one hex digit.

EXAMPLE **Convert 251 to hex.**

Binary first, then group in fours.

251 = 1111 1011

1111 = F, 1011 = B

ANSWER **FB**

Method B: divide by 16

GENERAL METHOD **Divide by 16 and use the remainders**

1. Divide by 16. The remainder (0 to 15) is the **rightmost** hex digit.
2. Divide the answer by 16 again; the remainder is the next digit to the left.
3. Stop when the answer is 0. Read the remainders from the bottom up.

EXAMPLE **Convert 301 to hex.**

Divide by 16, keep the remainders.

301 ÷ 16 = 18 remainder 13 → D

18 ÷ 16 = 1 remainder 2 → 2

1 ÷ 16 = 0 remainder 1 → 1

ANSWER **12D**

TRY IT YOURSELF

1. Convert 1001 0011 to hex.
2. Convert 2D to binary.
3. Convert 236 to hex.
4. Convert 1A to denary.

9 Summary

- Computers use binary because they are made of switches / logic gates with only two states (1 and 0). All data is converted to binary.
- Denary is base 10, binary is base 2 (0, 1), hexadecimal is base 16 (0-9, A-F).
- 8-bit place values: 128 64 32 16 8 4 2 1. Binary to denary: add the place values of the 1s.
- Denary to binary: subtract the biggest place value that fits (or divide by 2, read remainders upwards). Give the number of bits asked for.
- One hex digit = 4 bits. Binary ↔ hex: group in fours from the right.
- Hex to denary: digit × 16 + digit (× 256 for a third digit). Denary to hex: go through binary, or divide by 16.
- n bits hold 0 to $2^n - 1$ (8 bits: 0-255). Fewest bits: first power of 2 bigger than the number; hex: digits × 4.

IF YOU SEE …

"Convert … to 8-bit binary"	→ exactly 8 bits, with leading zeros
"least number of bits"	→ first power of 2 bigger than the number (hex: 4 per digit)
a hex digit 0 in the middle	→ write it as 0000

PART 2

How it is examined

Question types, what the examiner gives marks for, and real past-paper questions solved.

10 Ways they test it

TYPE 1 Convert three numbers

> *"Convert these three denary numbers to 8-bit binary numbers: 50, 102, 221" [3]. Also binary → hex, hex → binary, denary → hex, often as a table to fill in.*

Spot it a list of numbers and a working space.

1. One mark per number, all or nothing, so check each one.
2. Binary answers: the number of bits asked for.
3. Hex answers: "one mark per correct character in the correct order" for longer conversions.

TYPE 2 Explain the system

> *"Explain what is meant by a binary number system" [2]*

Spot it "what is meant by" / "explain".

1. It is base 2.
2. It uses only two values, 1 and 0.

TYPE 3 Why binary

> *"Explain why data input into the machine needs to be converted to binary" [2]*

Spot it "why" + "binary".

1. The computer contains logic gates / switches ...
2. ... that only process two states / 1 and 0.

TYPE 4 **Size of a register**

> *"State the highest denary number in an 8-bit register" [1], "Calculate the least number of bits to store 301" [1]*

Spot it "highest", "least number of bits".

1. $2^n - 1$ for the highest.
2. First power of 2 bigger than the number for the bits.

TYPE 5 **Similarities and differences**

> *"Give one similarity and two differences between hexadecimal and binary" [3]*

Spot it two systems compared.

1. Similarity: both are number systems (both can represent the same values).
2. Differences: base 2 vs base 16; binary uses only 0 and 1, hex uses 0-9 and A-F; hex is shorter.

11 What the examiner looks for

"Explain why computers use binary" [2]

Marking points (one mark each, up to the marks shown)

- Computers contain logic gates / switches ...
- ... that only have two states // only process the values 1 and 0.

KEYWORDS **logic gates** **switches** **two states** **1 and 0**

TOO VAGUE "computers understand binary"; "binary is easier for computers"

"Explain what is meant by a binary number system" [2]

Marking points (one mark each, up to the marks shown)

- It has a base of 2.
- It only uses two values ...
- ... that are 1 and 0.

KEYWORDS **base 2** **1 and 0**

Wording trap · Writing an 8-bit binary answer

THE QUESTION **0478/13/O/N/23 Q2** ↗ Paper 1 · Computer systems **3 marks**

Convert the denary number 14 to binary.

✓ CREDITED

- 0000 1110
- 1110 (the scheme wrote (0000)1110: leading zeros optional when no number of bits is set)

✗ NOT CREDITED

- 111 (a bit missing)
- 0111 0000 (written backwards)

WHY

When the question says 8-bit, write all 8 bits; then a missing leading 0 can never cost you the mark.

From: 0478/13/O/N/23 Q2(b) mark scheme

12 Past-paper questions, solved

0478/13/O/N/23 Q2 ↗ Paper 1 · Computer systems **8 marks** ⤢ Full size

2 Humans use a denary number system and computers use a binary number system.

(a) Explain what is meant by a binary number system.

..... [2]

(b) Convert the denary numbers 14, 59 and 234 to binary.

234 .....
[3]

Working space

(c) Convert the denary numbers 9, 26 and 65 to hexadecimal.

[3]

Working space

Swipe sideways to read the whole question

How to start Part (a) is two facts about binary (base, two values). For (b) and (c) use the place-value table; go through binary for the hex.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

(a) A binary number system has a **base of 2**. It uses only **two values, 1 and 0**.

(b) 14 = 0000 1110 · 59 = 0011 1011 · 234 = 1110 1010

(c) 9 = 9 · 26 = 0001 1010 = 1A · 65 = 0100 0001 = 41

PART	ANSWER / WORKING	MARKS	GUIDANCE
2(a)	Any two from: it has a base of 2 · it only uses two values · … that are 1 and 0	B2	
2(b)	(0000)1110 · (00)111011 · 11101010	B3	one mark each
2(c)	9 · 1A · 41	B3	one mark each

Total: 8 marks

2 A car park has a payment machine that allows a customer to pay for their parking.

The cost of parking is displayed as a denary number on a screen on the payment machine.

The cost of parking is stored in two 8-bit binary registers.

For the parking cost of $10.50:
- register 1 stores the denary value 10 as binary
- register 2 stores the denary value 50 as binary.

(a) Give the parking cost that would be displayed on the payment machine when the registers store:
- register 1: 00010001
- register 2: 01000110

Parking cost displayed $ ..... [2]

Working space

(b) The parking cost of $14.98 is displayed on the payment machine.

Give the 8-bit binary numbers that are stored in the registers to display the parking cost.

Register 1 .....

Register 2 .....
[2]

Working space

(c) The payment machine gives the customer a ticket when they have paid their parking cost. Each ticket has a 4-digit hexadecimal ticket number that is stored as binary.

The binary number 1010000000111101 is stored for a customer's ticket number.

Give the hexadecimal ticket number that would be displayed on this customer's ticket.

Hexadecimal ticket number ..... [4]

Working space

Swipe sideways to read the whole question

How to start Each register is a separate 8-bit number: convert register 1 for the dollars and register 2 for the cents. For (c) group the 16 bits in fours.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

(a) 0001 0001 = 16 + 1 = 17 and 0100 0110 = 64 + 4 + 2 = 70, so **$17.70**.

(b) 14 = 0000 1110 and 98 = 64 + 32 + 2 = 0110 0010.

(c) 1010 | 0000 | 0011 | 1101 = A 0 3 D, so **A03D**.

PART	ANSWER / WORKING	MARKS	GUIDANCE
2(a)	17 · 70 (correct fee $17.70)	B2	one mark for each part, in the correct order
2(b)	Register 1: 00001110 · Register 2: 01100010	B2	
2(c)	A · 0 · 3 · D	B4	one mark per correct character in the correct order

Total: 8 marks

EXAMINER TIP A whole 16-bit number given at once? Split it into nibbles first and write the four groups down before converting.

0478/13/M/J/23 Q1 ↗ Paper 1 · Computer systems **5 marks** ⤢ Full size

1 Computers store data as binary. The binary number 10101110 is stored.

(a) Convert the binary number to denary.

..... [1]

Working space

(b) Convert the binary number to hexadecimal.

..... [2]

Working space

(e) The denary number 301 needs to be stored.

Calculate the least number of bits that can be used to store the denary number 301.

..... [1]

Working space

(f) The hexadecimal number A4D needs to be stored.

Calculate the least number of bits that can be used to store the hexadecimal number A4D.

..... [1]

Working space

Swipe sideways to read the whole question

How to start Same number twice: add the place values for denary, group in fours for hex. For the least number of bits, find the first power of 2 above the number.

▼ **Show a full-mark answer and the mark scheme** · official marking

WHAT TO WRITE

(a) 128 + 32 + 8 + 4 + 2 = **174**

(b) 1010 = A, 1110 = E → **AE**

(e) 256 < 301 < 512 = 2^9 → **9** bits

(f) 3 hex digits × 4 bits = **12** bits

PART	ANSWER / WORKING	MARKS	GUIDANCE
1(a)	174	B1	
1(b)	A · E	B2	one mark per character in the correct order
1(e)	9	B1	
1(f)	12	B1	
			Total: 5 marks

13 Common mistakes

✗ 14 → 1110 when 8 bits are asked for ✓ 0000 1110

Give the number of bits the question asks for.

✗ A04 → 1010 0100 ✓ 1010 0000 0100

Every hex digit is 4 bits, the 0 too.

✗ F0 → 150 ✓ F0 = 15 × 16 = 240

Multiply each hex digit by its place value.

✗ Grouping bits from the left: 101101 → 1011 01

✓ 0010 1101 → 2D

Always group from the right and pad the left group.

✗ "Computers use binary because it is easy"

✓ "made of switches / logic gates with two states, 1 and 0"

Name the hardware reason.

14 Practice

Practice: easy, medium, hard

1. Convert 0110 1001 to denary.
2. Convert 77 to 8-bit binary.
3. Convert 1100 0111 to hex.
4. What is the base of hexadecimal?
1. Convert B9 to denary.
2. Convert 175 to hex.
3. Convert 3C to 8-bit binary.
4. What is the least number of bits needed to store 1000?
1. Convert 5E3 to binary.
2. Convert 1000 0000 1111 to denary.
3. Convert 700 to hex.
4. A register is 12 bits. Give the largest number it can store in denary and in hex.

15 Can I do this?

0 / 6 ticked

- [ ] I can explain why computers use binary (switches / logic gates, two states).
- [ ] I know the base and the digits of denary, binary and hexadecimal.
- [ ] I can convert binary to denary and denary to 8-bit binary.
- [ ] I can convert between binary and hex by grouping in fours, keeping every 0000.
- [ ] I can convert between hex and denary both ways.
- [ ] I can give the largest value of a register and the fewest bits for a number.

Anything unticked: revise that section first (each line links to it). All ticked: take the quiz.

Answers

Answers to the "Try it yourself" and practice questions on this page. Try every question first.

Try it yourself 1

1.

1. 64 + 16 + 4 + 1 = 85

ANSWER **85**

2.

1. 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 255 (the largest 8-bit number)

ANSWER **255**

3.

1. 128 + 64 + 8 = 200

ANSWER **200**

Try it yourself 2

1.

1. 32 + 16 + 2 = 50

ANSWER **0011 0010**

2.

1. 64 + 32 + 4 + 2 = 102

ANSWER **0110 0110**

3.

1. 128 + 64 + 16 + 8 + 4 + 1 = 221

ANSWER **1101 1101**

Try it yourself 3

1.

ANSWER **255**

2.

1. 128 ≤ 200 < 256, and 256 = 2^8, so 8 bits

ANSWER **8**

3.

1. 3 hex digits × 4 bits = 12

ANSWER **12**

Try it yourself 4

1.

1. 1001 = 9, 0011 = 3

ANSWER **93**

2.

1. 2 = 0010, D = 1101

ANSWER **0010 1101**

3.

1. 236 = 1110 1100
2. 1110 = E, 1100 = C

ANSWER **EC**

4.

1. 1 × 16 + 10 = 26

ANSWER **26**

Practice: easy, medium, hard

Easy 1.

1. 64 + 32 + 8 + 1

ANSWER **105**

Easy 2.

1. 64 + 8 + 4 + 1

ANSWER **0100 1101**

Easy 3.

ANSWER **C7**

Easy 4.

ANSWER **16**

Medium 1.

1. 11 × 16 + 9 = 176 + 9

ANSWER **185**

Medium 2.

1. 175 = 1010 1111
2. A F

ANSWER **AF**

Medium 3.

ANSWER **0011 1100**

Medium 4.

1. $512 < 1000 < 1024 = 2^{10}$

ANSWER **10**

Hard 1.

ANSWER **0101 1110 0011**

Hard 2.

1. 2048 + 8 + 4 + 2 + 1

ANSWER **2063**

Hard 3.

1. 700 ÷ 16 = 43 r 12 (C)
2. 43 ÷ 16 = 2 r 11 (B)
3. 2 ÷ 16 = 0 r 2

ANSWER **2BC**

PART	ANSWER / WORKING	MARKS	GUIDANCE
	2BC	B2	one mark for two correct characters, two marks for all three in order

Total: 2 marks

Hard 4.

1. 12 bits all 1 = 2^{12} − 1 = 4095
2. three groups of 1111 = FFF

ANSWER **4095 · FFF**

NOTE 2

Hexadecimal uses, binary addition and overflow

1 In this note

SYLLABUS 1.1 NUMBER SYSTEMS

- 1.1.3 Explain **how and why hexadecimal is used** as a beneficial way to represent data, and name where it is used.
- 1.1.4 **Add two positive 8-bit binary integers**, showing the carries.
- 1.1.4 Explain **overflow** and why it happens in binary addition.

Before this: Number systems (conversions). Next: Logical shifts and two's complement.

PART 1

Learn the topic

The notes: every idea from the start, with our own examples.

2 What you should already know

Binary ↔ hex

One hex digit is exactly four bits: group binary in fours from the right.

Quick check: what is 1101 0010 in hex?

D2.

Adding in columns

In denary, 7 + 5 = 12: write 2 and carry 1 into the next column. Binary works the same way, but you carry as soon as a column reaches 2.

Quick check: in denary, what is carried in 8 + 6?

1 (8 + 6 = 14: write 4, carry 1).

3 Why hexadecimal is used

A computer never processes hexadecimal: inside, everything stays binary. Hex is used by **people** (programmers, technicians) as a short way of **writing** binary. Compare the same 16-bit value:

Binary	Hexadecimal
1010 0000 0011 1101	A03D

The hex is four characters instead of sixteen. That is the whole idea, and the benefits all come from it:

Shorter

A **shorter representation** of binary: one hex digit replaces four bits, so it **takes up less space on screen** or paper.

Easier for humans

Easier and quicker to read, write, understand and remember than a long string of 1s and 0s.

Fewer mistakes

Less likely to make a mistake copying or typing it, and **easier to spot errors / debug**.

Converts easily

Each hex digit is **exactly 4 bits**, so it converts straight to and from binary (no big sums, unlike denary).

✗ COMMON MISTAKE

Never say hex "uses less memory" or "is faster for the computer". The computer still stores the binary. Every benefit is **for humans**.

4 Where hexadecimal is used

Use	What it looks like	What it is
HTML colour codes	#FF8000	a colour on a web page: two hex digits each for the amount of red, green and blue (00 to FF = 0 to 255)
MAC addresses	00:1A:2B:3C:4D:5E	the unique address of a network interface card (12 hex digits = 48 bits)
IPv6 addresses	2001:0DB8:85A3::8A2E:0370:7334	the address of a device on the internet (128 bits, written in groups of 4 hex digits)
Memory addresses and **memory dumps**	A2F, 0x7FF3A2	locations in memory shown to a programmer when **debugging**
Error codes	0x80070057	codes shown when a program or device fails, so a technician can look them up
Assembly language / machine code	LDA &3F	instructions and values written in hex by low-level programmers
URLs	%20 for a space	characters that can't appear in a web address are written as hex codes
Character codes	ASCII 41 = 'A'	the ASCII / Unicode value of a character

TRY IT YOURSELF

1. An HTML colour code is #00FF00. What is the denary value of the green part (the middle two digits)?
2. How many bits does a MAC address of 12 hex digits use?

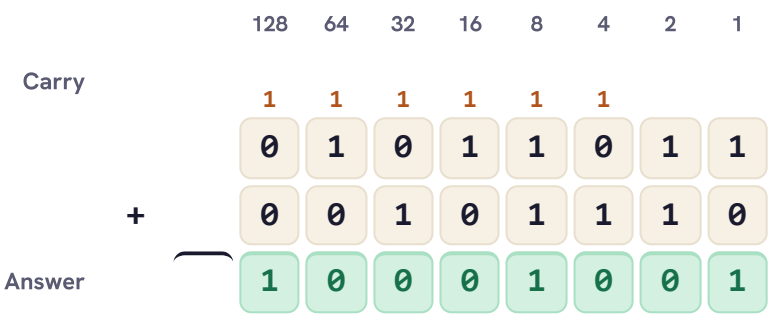
5 Adding two binary numbers

Binary addition works column by column from the right, exactly like denary, except that there are only two digits. There are only four sums to know:

Column adds up to	Write	Carry to the next column
0 + 0 = 0	0	0
0 + 1 = 1	1	0
1 + 1 = 2 (binary 10)	0	1
1 + 1 + 1 = 3 (binary 11), with a carry	1	1

GENERAL METHOD **Adding two 8-bit binary numbers**

1. Write the two numbers one under the other, lined up on the right. Leave a row above for the carries.
2. Start at the rightmost column. Add the two bits and any carry.
3. Write the result bit (0 or 1) underneath; write any carry above the next column to the left.
4. Repeat for every column, moving left.
5. If there is still a carry out of the leftmost (8th) column, it needs a 9th bit: that is **overflow**.
6. Check: convert both numbers and the answer to denary; the sums must match.



0101 1011 + 0010 1110 = 1000 1001. Check: 91 + 46 = 137.

EXAM TIP

Show the carries. Mark schemes give a mark for "correct working, e.g. showing carries" on top of one mark per correct nibble.

ANSWER

Write every carry above its column, even when you can do it in your head.

TRY IT YOURSELF

1. Add 0011 0011 and 0111 1000.
2. Add 1001 1011 and 0001 0011.

6 Overflow

A register has a **fixed number of bits**. An 8-bit register can hold 0 to **255**; a 16-bit register 0 to 65 535. Every computer and device has a predefined limit like this.

If the answer to a sum is **too large to fit in the number of bits available**, it would need an extra bit that the register does not have. This is an **overflow error**. The extra bit is lost and the value left in the register is wrong.

	OF	128	64	32	16	8	4	2	1
Carry	1	1							
		1	1	0	0	1	0	1	0
+		0	1	1	1	0	1	0	1
Answer	1	0	0	1	1	1	1	1	1

1100 1010 + 0111 0101: 202 + 117 = 319, more than 255. The 9th bit is the overflow; the register keeps only 0011 1111 = 63, which is wrong.

Overflow in two marks: **the result is greater than 255 (too large for the register) · it would need more than 8 bits / more bits than the register has.**

TRY IT YOURSELF

1. Add 1100 0011 and 0101 0000. Is there overflow?

2 What is the largest number a 16-bit register can hold?

7 Summary

- Hex is for humans: a shorter representation of binary, easier to read, write, remember and debug, fewer mistakes, less screen space. It converts easily because one hex digit = 4 bits.
- Uses of hex: HTML colour codes, MAC addresses, IPv6 addresses, memory addresses / dumps, error codes, assembly language, URLs, character codes.
- Binary addition: 1 + 1 = 0 carry 1; 1 + 1 + 1 = 1 carry 1. Work right to left and show every carry.
- Overflow: the result is too large for the number of bits in the register (over 255 for 8 bits), so an extra bit would be needed and the stored value is wrong.

IF YOU SEE …

"Give a reason hexadecimal is used"	→ a benefit FOR HUMANS: shorter / easier to read / fewer mistakes
"Show all your working" in an addition	→ write the carry row
a carry out of the 8th column	→ show the 9th bit and say overflow

PART 2

How it is examined

Question types, what the examiner gives marks for, and real past-paper questions solved.

8 Ways they test it

TYPE 1 Benefit of hex

> *"Give one reason why the addresses are displayed in hexadecimal instead of binary" [1-2]*

Spot it "instead of binary", "benefit", "reason a programmer may use hexadecimal".

1. Pick one benefit per mark: shorter / easier to read / fewer mistakes / easier to debug / less screen space.
2. Write it as a comparison: "it is shorter than binary".

TYPE 2 Uses of hex

> *"Identify two other ways hexadecimal is used in computer science" [2]*

Spot it "identify", "other ways".

1. Name two uses from the table, not the one already in the question.
2. A name is enough for "identify".

TYPE 3 Binary addition

> *"Add the two 8-bit binary numbers using binary addition. Show all of your working" [3-4]*

Spot it two 8-bit numbers one above the other.

1. Carry row, then the answer.
2. Marks: one per correct nibble, one for the carries, one more for showing the overflow bit when there is one.

TYPE 4 **Explain overflow**

> *"Explain why the overflow error occurred" [2]*

Spot it "overflow".

1. The result is greater than 255 / too large for the register.
2. It needs more than 8 bits / the register has a fixed number of bits.

TYPE 5 **Definition (multiple choice)**

> *"Tick one box to show the correct definition of overflow" [1]*

Spot it four statements.

1. Choose "too large to represent in the number of bits available"; not "too small", not "negative".

9 What the examiner looks for

"Give two reasons why a programmer may use hexadecimal to represent binary" [2]

Marking points (one mark each, up to the marks shown)

- Easier / quicker to understand, read or write
- Easier / quicker to debug
- Less likely to make a mistake
- Shorter representation // takes up less screen space

KEYWORDS **shorter** **easier to read** **fewer errors** **debug**

TOO VAGUE "takes less memory"; "the computer processes it faster"

"Explain why the overflow error occurred" [2]

Marking points (one mark each, up to the marks shown)

- The result of the calculation is greater than 255 // the value is larger than can be stored in the register
- The result would need more than 8 bits // a register has a predetermined number of bits and there are too many bits for it

KEYWORDS **greater than 255** **more than 8 bits** **fixed number of bits**

TOO VAGUE "the number is too big" with no mention of the bits or 255

Wording trap · A reason for using hexadecimal

THE QUESTION **0478/13/M/J/24 Q3** ↗ Paper 1 · Computer systems **1 mark**

Give one reason why the addresses are displayed in hexadecimal instead of binary.

✓ CREDITED

- It is easier / quicker to read, understand or debug
- It is a shorter representation of binary // takes up less screen space

✗ NOT CREDITED

- It uses less storage / memory
- The computer can process it faster

WHY

The computer stores binary whatever you display. Hex only helps the person reading it.

From: 0478/13/M/J/24 Q3(c) mark scheme

10 Past-paper questions, solved

0478/13/M/J/24 Q3 ↗ Paper 1 · Computer systems **8 marks** ⤢ Full size

3 A programmer is creating a computer game. One character is **not** moving correctly.

The programmer needs to debug the program. To do this they need to look at addresses that are locations in memory.

The addresses are displayed as hexadecimal numbers.

(a) One address is A2F.

(i) Convert the address to binary.

..... [3]

(ii) Convert the address to denary.

..... [1]

Working space

(b) The binary number stored for another address is 000110011011.

(i) Convert the binary number to hexadecimal.

..... [3]

(ii) Convert the binary number to denary.

..... [1]

Working space

(c) Give **one** reason why the addresses are displayed in hexadecimal instead of binary.

..... [1]

(d) Identify **two** other ways that hexadecimal is used in computer science.

[2]

Swipe sideways to read the whole question

How to start Hex to binary is four bits per digit (keep the 0010). For binary to hex group in fours from the right. Parts (c) and (d) are recall.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

(a)(i) A = 1010, 2 = 0010, F = 1111 → **1010 0010 1111** (ii) 10 × 256 + 2 × 16 + 15 = **2607**

(b)(i) 0001 | 1001 | 1011 → **19B** (ii) 256 + 128 + 16 + 8 + 2 + 1 = **411**

(c) It is shorter than binary, so it is easier to read and debug.

(d) HTML colour codes; MAC addresses.

PART	ANSWER / WORKING	MARKS	GUIDANCE
3(a)(i)	1010 0010 1111	B3	one mark for each correct nibble, in the correct order
3(a)(ii)	2607	B1	
3(b)(i)	1 · 9 · B	B3	one mark for each correct character, in order
3(b)(ii)	411	B1	
3(c)	Any one: easier / quicker to read / understand / debug · shorter representation of binary // less screen space	B1	
3(d)	Any two: HTML colour codes · URL · memory dump · IP address · MAC address · assembly language · error codes · ASCII / Unicode	B2	

Total: 11 marks

0478/12/M/J/25 Q1 ↗ Paper 1 · Computer systems **5 marks** ⤢ Full size

1 A computer stores data temporarily in random access memory (RAM).

(e) The 8-bit binary numbers 01100101 and 01110000 are stored in RAM.

The binary numbers are added together.

Add the **two** binary numbers using binary addition.

Give your answer in binary. You must show all your working.

```
  0 1 1 0 0 1 0 1
+ 0 1 1 1 0 0 0 0
```

[3]

(f) An overflow occurs when two other 8-bit binary numbers are added together.

Explain why the overflow error occurs.

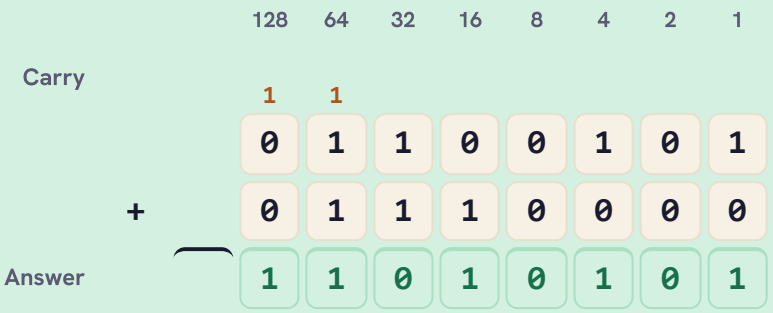
..... [2]

Swipe sideways to read the whole question

How to start Set out the carry row first. For (f), give two separate facts: the size of the result and the size of the register.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE



(e) **1101 0101** (101 + 112 = 213, so no overflow here).

(f) The result is greater than 255, so it is too large for the register. It would need more than 8 bits, but the register only has 8.

PART	ANSWER / WORKING	MARKS	GUIDANCE
1(e)	1101 0101	B2	one mark for each correct nibble
1(e)	correct method of working, e.g. showing carries	M1	
1(f)	Any two: the result is greater than 255 // too large · cannot be stored in 8 bits // in the bits available for the register	B2	
			Total: 5 marks

0478/13/O/N/24 Q1 ↗ Paper 1 · Computer systems **4 marks** ⤢ Full size

1 Data can be measured using different units of storage.

(d) Binary integers can be added together.

Add the **two** binary integers using binary addition. Show all your working. Give your answer in binary.

```
  1 1 1 0 0 0 1 1
+ 1 1 0 0 1 1 0 0
```

[4]

Swipe sideways to read the whole question

How to start Two big numbers (both start with 11): expect a carry out of the last column. Show it as a 9th bit.

▼ **Show a full-mark answer and the mark scheme** · official marking

WHAT TO WRITE

	OF	128	64	32	16	8	4	2	1
Carry	1	1							
		1	1	1	0	0	0	1	1
+		1	1	0	0	1	1	0	0
Answer	1	1	1	0	1	0	1	1	1

1 1010 1111: the 9th bit is an overflow (227 + 204 = 431, more than 255).

PART	ANSWER / WORKING	MARKS	GUIDANCE
1(d)	1010 1111	**B2**	one mark for each correct nibble
1(d)	method of working, e.g. carries	**M1**	
1(d)	identification of overflow (the extra 1)	**B1**	

Total: 4 marks

EXAMINER TIP The overflow bit earns its own mark: write it, and label it.

11 Common mistakes

✗ "Hex takes up less memory"

The computer stores binary either way.

✓ "Hex is shorter / easier for humans to read"

✗ Leaving out the carries

The working mark is for the carries.

✓ A carry row above the sum

✗ Dropping the 9th bit silently

✓ Write the extra 1 and label it overflow

There is a mark for identifying the overflow.

✗ "Overflow is when the number is too small / negative"

✓ "Too large for the number of bits available"

12 Practice

Practice: easy, medium, hard

1. Add 0001 0101 and 0010 0110.
2. Give one use of hexadecimal.
3. What is the largest denary value an 8-bit register can store?
1. Add 0111 0110 and 0011 0000.
2. Give two reasons why hex is used instead of binary to display error codes.
3. An HTML colour is #8000FF. Give the denary amount of red (the first two digits).
1. Add 1011 1101 and 0110 0110. Explain what happens in an 8-bit register.
2. A memory dump shows the address 0FC8. Give it in binary and in denary.

13 Can I do this?

0 / 4 ticked

- [ ] I can give benefits of hexadecimal, all of them for humans.
- [ ] I can name at least four uses of hexadecimal.
- [ ] I can add two 8-bit binary numbers with a carry row.
- [ ] I can explain overflow in two marking points and show the overflow bit.

Anything unticked: revise that section first (each line links to it). All ticked: take the quiz.

Answers

Answers to the "Try it yourself" and practice questions on this page. Try every question first.

Try it yourself 1

1.

1. FF = 15 × 16 + 15 = 255

ANSWER **255**

2.

1. 12 × 4 = 48

ANSWER **48**

Try it yourself 2

1.

1. 51 + 120 = 171 = 1010 1011

ANSWER **1010 1011**

2.

1. 155 + 19 = 174 = 1010 1110

ANSWER **1010 1110**

Try it yourself 3

1.

1. 195 + 80 = 275
2. 275 needs 9 bits: 1 0001 0011

ANSWER **1 0001 0011: yes, overflow (195 + 80 = 275 > 255)**

2.

ANSWER **65 535**

Practice: easy, medium, hard

Easy 1.

1. 21 + 38 = 59

ANSWER **0011 1011**

Easy 2.

ANSWER **e.g. HTML colour codes / MAC addresses / error codes**

Easy 3.

ANSWER **255**

Medium 1.

1. 118 + 48 = 166

ANSWER **1010 0110**

Medium 2.

ANSWER

Shorter / takes less screen space; easier to read and remember; fewer mistakes when copying

Medium 3.

1. 80 = 8 × 16 = 128

ANSWER **128**

Hard 1.

1. 189 + 102 = 291

2. 9 bits needed

ANSWER **1 0010 0011: overflow, because 189 + 102 = 291 > 255 and the 9th bit is lost**

PART	ANSWER / WORKING	MARKS	GUIDANCE
	1 0010 0011 (each nibble)	B2	
	carries shown	M1	
	overflow identified	B1	
			Total: 4 marks

Hard 2.

1. 0 = 0000, F = 1111, C = 1100, 8 = 1000

2. 15 × 256 + 12 × 16 + 8 = 3840 + 192 + 8 = 4040

ANSWER **0000 1111 1100 1000 · 4040**

NOTE 3

Logical shifts and two's complement

1 In this note

SYLLABUS 1.1 NUMBER SYSTEMS

- 1.1.5 Perform **logical left and right shifts** (one or several places) on a positive 8-bit binary integer.
- 1.1.5 Explain the **effect of a shift**: multiply or divide by 2, 4, 8 ..., and what happens to bits that fall off.
- 1.1.6 Use **two's complement** to represent positive and negative 8-bit integers, in both directions.

Before this: binary addition and overflow. Next: Text, sound and images.

PART 1

Learn the topic

The notes: every idea from the start, with our own examples.

2 What you should already know

8-bit place values

128 64 32 16 8 4 2 1. The leftmost bit is the **most significant bit** (MSB, worth the most); the rightmost is the **least significant bit** (LSB, worth 1).

Quick check: what is the MSB of 0100 0001 worth if it is 1?

128.

Multiplying by 10 in denary

Moving every digit one place left multiplies a denary number by 10: 36 → 360. In binary, one place left multiplies by **2**.

Quick check: what does moving 2 places left do in denary?

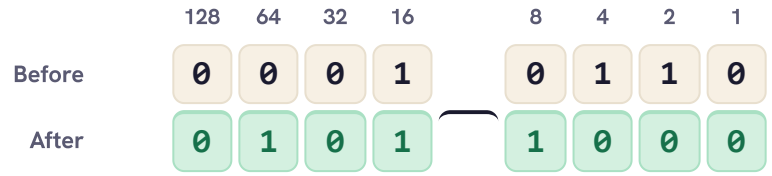
Multiplies by 100.

3 Logical shifts

A **logical shift** moves every bit in a register a set number of places to the left or to the right. The register stays 8 bits wide, so:

- bits that move off the end of the register are **lost**;
- the empty places at the other end are filled with **0s**.

Logical left shift

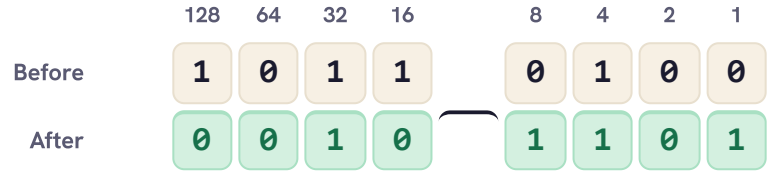


Logical left shift of 2 places: 0001 0110 (22) becomes 0101 1000 (88). Two 0s come in on the right.

GENERAL METHOD **Logical left shift by n places**

1. Move every bit n places to the left.
2. The n bits at the left end (the most significant bits) are lost.
3. Fill the n empty places on the right with 0s.
4. Effect: the number is **multiplied by 2^n** (1 place × 2, 2 places × 4, 3 places × 8), as long as no 1s were lost.

Logical right shift



Logical right shift of 2 places: 1011 0100 (180) becomes 0010 1101 (45). Two 0s come in on the left.

GENERAL METHOD **Logical right shift by n places**

1. Move every bit n places to the right.
2. The n bits at the right end (the least significant bits) are lost.
3. Fill the n empty places on the left with 0s.
4. Effect: the number is **divided by 2^n** (1 place ÷ 2, 2 places ÷ 4, 4 places ÷ 16).

When bits that are 1 fall off

If a **1** is pushed off the end, the answer is no longer exactly × or ÷ 2^n: the value becomes **incorrect / inaccurate**.

Shift	Before	After	What happened
right 1	0000 1101 (13)	0000 0110 (6)	13 ÷ 2 = 6.5, but the 1 on the right was lost: only the whole number 6 is left
left 1	1100 0011 (195)	1000 0110 (134)	195 × 2 = 390 needs 9 bits; the MSB 1 was lost, so the result is wrong

TRY IT YOURSELF

1. Perform a logical left shift of 3 places on 0000 1101.
2. Perform a logical right shift of 3 places on 0101 1000.
3. What is the effect of a logical right shift of 4 places on a positive binary integer?

4 Two's complement: negative numbers

Ordinary binary can only hold positive numbers. **Two's complement** lets one 8-bit register hold positive **and** negative numbers. The trick: the most significant bit is worth **−128** instead of +128.



1101 0011 = −128 + 64 + 16 + 2 + 1 = −45

The sign bit

MSB **0** → the number is positive (or 0). MSB **1** → the number is negative.

Positive numbers

Exactly the same as ordinary binary, with a 0 at the front: 45 = 0010 1101.

Range

8-bit two's complement holds **−128 to +127**: 1000 0000 = −128 and 0111 1111 = +127.

−1

1111 1111 = −128 + 127 = −1. All 1s is −1, not −255.

Denary to two's complement

Method A: flip and add 1

GENERAL METHOD **Flip and add 1**

1. Write the positive number in 8-bit binary.
2. If the number is positive, stop: that is the answer.
3. If negative, **flip** every bit (0 → 1, 1 → 0).
4. **Add 1** to the result.

EXAMPLE **Convert −45 to 8-bit two's complement.**

Write 45, flip every bit, add 1.

45 = 0010 1101

flip: 1101 0010

add 1: 1101 0011

ANSWER **1101 0011**

Method B: start from −128

GENERAL METHOD **Start from −128**

1. Put a 1 in the −128 column.
2. Work out what must be added to −128 to reach the number: −45 − (−128) = 83.
3. Write 83 in the other seven columns: 64 + 16 + 2 + 1.

EXAMPLE **Convert −45 to 8-bit two's complement.**

−128 + 83 = −45.

83 = 101 0011 in the 7 columns

ANSWER **1101 0011**

Two's complement to denary

GENERAL METHOD **Two's complement to denary**

1. Look at the MSB. If it is 0, convert as ordinary binary.
2. If it is 1, the number is negative: add the place values with the MSB worth −128.
3. (Or: flip every bit, add 1, convert, and put a minus sign in front.)

EXAMPLE **Convert the two's complement integer 1110 1100 to denary.**

MSB is 1, so the first column is −128.

−128 + 64 + 32 + 8 + 4

ANSWER **−20**

TRY IT YOURSELF

1. Convert −78 to an 8-bit two's complement integer.
2. Convert the two's complement integer 1011 0111 to denary.

3 Convert 45 to an 8-bit two's complement integer.

5 Summary

- Logical shift: every bit moves left or right; bits that fall off are lost; 0s fill the empty places.
- Left shift n places = × 2^n; right shift n places = ÷ 2^n. If 1s are lost, the result is inaccurate / wrong.
- Two's complement: the MSB is worth −128. MSB 1 = negative. Range −128 to +127.
- Negative denary → two's complement: write the positive in binary, flip every bit, add 1.
- Two's complement → denary: −128 + the other place values (when the MSB is 1).

IF YOU SEE …

"state the effect" of a shift	→ × or ÷ by 2^n, and bits lost → may be inaccurate
"describe the process" of a shift	→ bits move, MSB / LSB lost, 0s added at the other end
a negative number to store	→ two's complement: flip and add 1, and show it

PART 2

How it is examined

Question types, what the examiner gives marks for, and real past-paper questions solved.

6 Ways they test it

TYPE 1 **Do the shift**

> *"Give the 8-bit binary number stored after a logical left shift of three places" [1]*

Spot it a register and a number of places.

1 Copy the 8 bits, move them, fill with 0s, count that you still have 8 bits.

TYPE 2

Effect of the shift

> *"State one effect this logical shift has on the binary number" [1] / "State the mathematical effect of a right shift of four places" [2]*

Spot it "effect".

1. Multiplied / divided by 2^n (give the number: by 8, by 16).
2. Bits are lost, so the value may become incorrect.

TYPE 3

Describe the process

> *"Describe the process of the logical left shift" [2]*

Spot it "describe" + "process".

1. Every bit is moved left (by the number of places).
2. The leftmost / most significant bit is lost.
3. A 0 is added as the rightmost / least significant bit.

TYPE 4

Convert to or from two's complement

> *"Complete the register for −78. Show all your working" [2]*

Spot it a negative denary number or a binary integer starting with 1.

1. 1 mark for a working method (flip and add 1, or −128 + ...).
2. 1 mark for the correct answer.

TYPE 5

Name it

> *"Give the name of the number system used to represent negative binary integers" [1]*

Spot it "negative".

1. Two's complement.

7 What the examiner looks for

"Describe the process of a logical left shift" [2] + "State the effect" [1]

Marking points (one mark each, up to the marks shown)

- Each / every bit in the binary number is shifted / moved to the left
- The leftmost / most significant bit is lost
- A 0 is added as the rightmost / least significant bit
- Effect: the binary integer is multiplied by 2 (per place)

KEYWORDS **moved left** **most significant bit lost** **0 added** **multiplied by 2**

Two's complement conversion [2]

Marking points (one mark each, up to the marks shown)

- One mark for a full method of working, e.g. conversion to binary then flipping and adding 1
- One mark for the correct 8-bit answer

KEYWORDS **flip** **add 1** **−128**

TOO VAGUE an answer with no working (loses the method mark); fewer than 8 bits

Wording trap · The effect of a right shift

THE QUESTION **0478/12/M/J/23 Q2** ↗ Paper 1 · Computer systems — **1 mark**

State one effect this logical right shift (of three places) has on the binary number.

✓ CREDITED

- It is divided by 8
- The value becomes incorrect / inaccurate because the rightmost bits are lost

✗ NOT CREDITED

- It gets smaller
- It is divided by 3

WHY

Give the exact operation: 3 places = ÷ 2^3 = ÷ 8. "Smaller" is not specific enough.

From: 0478/12/M/J/23 Q2(b)(ii) mark scheme

8 Past-paper questions, solved

0478/12/O/N/23 Q2 ↗ Paper 1 · Computer systems **3 marks** ⤢ Full size

2 A register stores the binary number:

1	1	1	0	0	0	1	1

(c) A logical left shift of **two** places is performed on the binary number stored in the register.

Complete the binary register to show its contents after this logical left shift.

[1]

(d) The negative denary number −99 needs to be stored in the register.

Complete the register to show the binary number that would be stored, using two's complement. Show all your working.

Working space .....

Register:

[2]

Swipe sideways to read the whole question

How to start For (c) move every bit two places left and add two 0s. For (d) write 99 in binary first, then flip and add 1.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

(c) 1110 0011 → **1000 1100** (the two 1s on the left are lost).

(d) 99 = 0110 0011 → flip: 1001 1100 → add 1: **1001 1101**. Check: −128 + 16 + 8 + 4 + 1 = −99.

PART	ANSWER / WORKING	MARKS	GUIDANCE
2(c)	1 0 0 0 1 1 0 0	B1	
2(d)	suitable working method, e.g. flip and add 1	M1	
2(d)	1001 1101	A1	

Total: 3 marks

0478/12/F/M/24 Q8 ↗ Paper 1 · Computer systems **5 marks** ⤢ Full size

(b) Two's complement can be used to represent positive and negative binary integers.

(i) Convert the two's complement integer 10110111 into denary.

Show all your working.

..... [2]

(ii) Convert the denary number 45 into an 8-bit two's complement integer.

..... [1]

Working space

(c) Logical shifts are performed on binary integers.

State the mathematical effect of a right shift of **four** places on a positive binary integer.

..... [2]

Swipe sideways to read the whole question

How to start The MSB of 10110111 is 1, so it is negative: start from −128. 45 is positive, so it is plain binary. Four places right = ÷ 2^4.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

(b)(i) −128 + 32 + 16 + 4 + 2 + 1 = **−73** (or flip 1011 0111 → 0100 1000, add 1 → 0100 1001 = 73, so −73).

(b)(ii) **0010 1101**

(c) The integer is **divided by 16**.

PART	ANSWER / WORKING	MARKS	GUIDANCE
8(b)(i)	working, e.g. flip and add 1: 10110111 = 01001001	M1	
8(b)(i)	−73	A1	
8(b)(ii)	00101101	B1	
8(c)	divide · … by 16 // 2^4	B2	one mark each

Total: 5 marks

0478/12/O/N/24 Q3 ↗ Paper 1 · Computer systems — **3 marks** — ⤢ Full size

3 A user enters data that is hexadecimal into a computer system. The data is converted to binary to be processed by the computer.

(d) A binary integer that is stored in a register in the computer has a logical left shift performed on it.

(i) Describe the process of the logical left shift that is performed on the binary integer.

..... [2]

(ii) State what effect this will have on the binary integer.

..... [1]

Swipe sideways to read the whole question

How to start "Describe the process" wants the three things that physically happen to the bits; "effect" wants the maths.

▼ **Show a full-mark answer and the mark scheme** · official marking

WHAT TO WRITE

(d)(i) Every bit in the integer is moved one place to the left. The leftmost (most significant) bit is lost and a 0 is added as the rightmost (least significant) bit.

(d)(ii) The integer is multiplied by 2.

PART	ANSWER / WORKING	MARKS	GUIDANCE
3(d)(i)	Any two: every bit is shifted left · the leftmost / MSB is lost · a 0 is added as the rightmost / LSB	**B2**	
3(d)(ii)	The binary integer is multiplied by 2	**B1**	
			Total: 3 marks

9 Common mistakes

✗ Shifting and ending up with 9 or 7 bits ✓ Always exactly 8 bits after the shift

Lost bits are gone; new places are 0s.

✗ "Right shift of 3 divides by 3" ✓ Divides by $2^3 = 8$

Each place is a factor of 2.

✗ Flipping the bits and forgetting to add 1 ✓ Flip, THEN add 1

Flip alone gives one's complement, which is not on the syllabus.

✗ Writing –73 as 1100 1001 (73 with a 1 in front) ✓ 1011 0111

Two's complement is not "sign and size".

✗ No working for two's complement

✓ Show the flip and the +1

One of the two marks is for the method.

10 Practice

Practice: easy, medium, hard

1. Logical left shift of 1 place on 0010 0101.
2. Logical right shift of 1 place on 0110 0100.
3. Is the two's complement integer 1000 0101 positive or negative?
1. Convert −25 to an 8-bit two's complement integer.
2. Convert the two's complement integer 1000 1110 to denary.
3. A logical right shift of 2 places is done on 1100 1011. Give the result and say why it is not exactly ÷ 4.
1. Convert −22 to an 8-bit two's complement integer. Show your working.
2. What are the smallest and largest numbers in 8-bit two's complement? Give each in binary.
3. A value 0001 0110 is shifted left 3 places. Give the result and its denary value. Has any data been lost?

11 Can I do this?

0 / 4 ticked

- [ ] I can do left and right logical shifts of any number of places.
- [ ] I can state the effect of a shift (× or ÷ 2^n, bits lost).
- [ ] I can convert a negative denary number to 8-bit two's complement, with working.
- [ ] I can convert a two's complement integer to denary.

Anything unticked: revise that section first (each line links to it). All ticked: take the quiz.

Answers

Answers to the "Try it yourself" and practice questions on this page. Try every question first.

Try it yourself 1

1.

1. 13 × 8 = 104 = 0110 1000

ANSWER **0110 1000**

2.

1. 88 ÷ 8 = 11

ANSWER **0000 1011**

3.

ANSWER **It is divided by 16 (2^4)**

Try it yourself 2

1.

1. 78 = 0100 1110
2. flip: 1011 0001
3. +1: 1011 0010

ANSWER **1011 0010**

2.

1. −128 + 32 + 16 + 4 + 2 + 1 = −73

ANSWER **−73**

3.

1. positive: ordinary binary with 0 at the front

ANSWER **0010 1101**

Practice: easy, medium, hard

Easy 1.

ANSWER **0100 1010**

Easy 2.

ANSWER **0011 0010**

Easy 3.

ANSWER **negative (MSB is 1)**

Medium 1.

1. 25 = 0001 1001
2. flip 1110 0110
3. +1 1110 0111

ANSWER **1110 0111**

Medium 2.

1. −128 + 8 + 4 + 2

ANSWER **−114**

Medium 3.

1. 203 ÷ 4 = 50.75, the register holds 50

ANSWER **0011 0010: the two 1s on the right were lost**

Hard 1.

1. 22 = 0001 0110
2. flip 1110 1001
3. +1 1110 1010

ANSWER **1110 1010**

PART	ANSWER / WORKING	MARKS	GUIDANCE
	flip and add 1 shown	M1	
	1110 1010	A1	

Total: 2 marks

Hard 2.

ANSWER **−128 = 1000 0000; +127 = 0111 1111**

Hard 3.

ANSWER **1011 0000 = 176; no 1s were lost (22 × 8 = 176)**

NOTE 4

Text, sound and images

1 In this note

SYLLABUS 1.2 TEXT, SOUND AND IMAGES

- 1.2.1 Explain how and why a computer represents **text**, using a **character set** such as **ASCII** or **Unicode**.
- 1.2.2 Explain how **sound** is sampled and stored, and the effects of the **sample rate** and **sample resolution**.
- 1.2.3 Explain how an **image** is stored as pixels, and the effects of the **resolution** and **colour depth**.

Before this: Number systems. Next: Data storage and compression (file-size calculations use these ideas).

PART 1

Learn the topic

The notes: every idea from the start, with our own examples.

2 What you should already know

Everything becomes binary

A computer is made of switches with two states, so text, sound and pictures must all be turned into 1s and 0s before it can process or store them.

Quick check: why can't a computer store the letter A directly?

It can only store binary (1s and 0s), so A must be given a binary code.

n bits give 2^n values

1 bit: 2 values; 2 bits: 4; 8 bits: 256; 16 bits: 65 536.

Quick check: how many values do 3 bits give?

8.

3 Text and character sets

When you press a key, the computer has to turn the character into binary. It does this with a **character set**: a list of all the characters the computer can use, with a **unique binary code** (number) for each one.

Each character typed is looked up in the character set and its code is stored. A word is stored as the codes of its characters, one after another, **in sequence**. Upper and lower case letters have different codes.

Character	Denary code	Binary (ASCII)
A	65	0100 0001
B	66	0100 0010
C	67	0100 0011
a	97	0110 0001
space	32	0010 0000

Codes follow the alphabet: if A is 65, then C is 67. So "CAB" is stored as 67, 65, 66.

ASCII and Unicode

	ASCII	Unicode
Bits per character	**7 bits** (8 bits in extended ASCII)	**up to 16 or 32 bits**
Number of characters	**128** (256 extended)	**over a million possible** (about 65 000 with 16 bits; about 4 billion with 32)
Languages	English only (letters, digits, punctuation, control codes)	**almost every language** in the world
Emojis and symbols	no	**yes**
Storage	less: 1 byte per character	**more storage per character**, so files are bigger

Unicode allows a **greater range of characters and symbols** (different languages, emojis), but it needs **more bits per character**, so text takes more storage space.

TRY IT YOURSELF

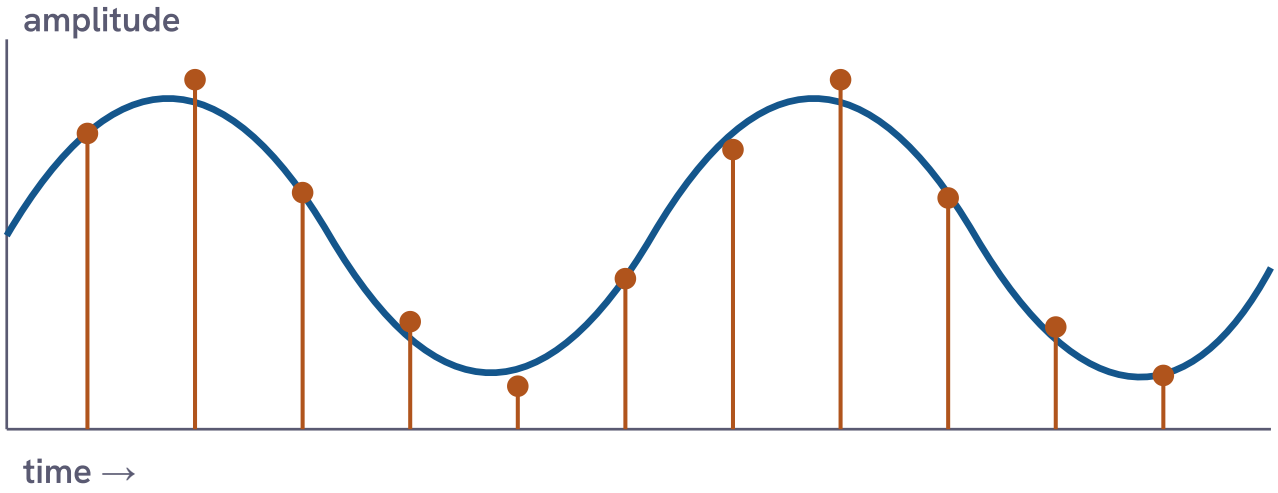
1. In ASCII, D is 68. What is the denary code for G?
2. A text message uses Arabic letters and emojis. Which character set must be used?

4 Sound

Sound is an **analogue** wave: it changes smoothly all the time. A computer can only store separate binary numbers, so the wave must be turned into **digital** data. A microphone picks up the sound, then the wave is **sampled**:

GENERAL METHOD **How sound is recorded and converted to binary**

1. The sound is recorded by a **microphone**, which turns it into an electrical (analogue) signal.
2. The sound wave is **sampled** at regular time intervals: the **amplitude** (height) of the wave is measured.
3. Each amplitude measured is given a **binary value**.
4. The **sample rate** sets how many samples are taken **per second**.
5. The **sample resolution** sets how many **bits are used for each sample**.
6. The binary values of all the samples are stored, in order.



Each orange line is one sample: the height of the wave at that moment, stored as a binary number.

Change	Effect on the recording	Effect on the file
Higher **sample rate** (more samples per second)	**more accurate**, closer to the original sound	**larger file size**
Higher **sample resolution** (more bits per sample)	more amplitudes can be recorded, so **more accurate** / better quality	**larger file size**
Lower rate or resolution	less accurate, poorer quality	smaller file

Sample rate: the number of samples taken per second (measured in hertz, Hz). **Sample resolution**: the number of bits used to store each sample (e.g. 16-bit).

TRY IT YOURSELF

1. A song is recorded with a sample resolution of 8 bits. How many different amplitudes can be stored for each sample?
2. Give one drawback of increasing the sample rate.

5 Images

A digital (bitmap) image is a grid of tiny squares called **pixels**. Each pixel is one colour, and that colour is stored as a binary number. The whole image is stored as the binary codes of all its pixels, plus some extra data about the image.

Term	Meaning
Pixel	the smallest component of an image: one square of one colour
Resolution	the number of pixels in the image: pixels wide × pixels high, e.g. 1920 × 1080
Colour depth (bit depth)	the number of bits used to represent each colour (each pixel)
Metadata	extra data stored with the image, e.g. its width and height, colour depth, date and time taken

Colour depth	Number of colours	Example
1 bit	2	black and white
8 bits	256	simple graphics
24 bits	about 16.7 million	photographs (8 bits each for red, green, blue)

Change	Effect on the image	Effect on the file
Higher **resolution** (more pixels)	more detail, better quality	**larger file size** (more pixels to store)
Higher **colour depth** (more bits per pixel)	**a greater range of colours**, closer to real life	**larger file size** (more bits per pixel)
Lower resolution or colour depth	less detail / fewer colours	smaller file

TRY IT YOURSELF

1. An image has a colour depth of 4 bits. How many colours can it use?
2. An image is 600 pixels wide and 400 pixels high. How many pixels does it have?

6 Summary

- Text: a character set gives every character a unique binary code; the codes are stored in sequence.
- ASCII: 7 (or 8) bits, 128 (256) characters, English. Unicode: up to 16/32 bits, many languages and emojis, more storage per character.
- Sound: microphone → wave sampled → amplitude measured → each sample given a binary value.
- Sample rate = samples per second; sample resolution = bits per sample. Higher → more accurate, bigger file.
- Image: grid of pixels, each colour a binary code. Resolution = number of pixels; colour depth = bits per colour. Higher → better quality, bigger file.

IF YOU SEE …

"describe how text is converted to binary"	→ character set · unique binary value per character · e.g. ASCII / Unicode
"explain how sound is converted"	→ sampled · amplitude · binary value · sample rate · resolution
"what is meant by colour depth"	→ number of bits used for each colour

PART 2

How it is examined

Question types, what the examiner gives marks for, and real past-paper questions solved.

7 Ways they test it

TYPE 1 Text to binary

> *"Describe how the text is converted to binary" [2-3]*

Spot it keyboard / text / characters.

1. A character set is used (name one: ASCII or Unicode).
2. Each character has a unique binary value.
3. The codes are stored in sequence.

TYPE 2 ASCII vs Unicode

> *"Explain the differences between ASCII and Unicode" [4]*

Spot it both names in the question.

1. Write each point as a comparison: "ASCII has fewer characters, whereas Unicode has more".
2. Max 3 marks if you only describe one of them: always cover both.

TYPE 3 Sound conversion

> *"Explain how the analogue sound is recorded and converted to digital" [5]*

Spot it "analogue", "recorded", "converted".

1. Follow the six steps of the method: microphone, sampled, amplitude, binary value, sample rate (defined), resolution (defined).

TYPE 4 **Effects**

> *"State two ways the accuracy of the recording could be increased" [2] / "Explain the effect of changing the colour depth" [2]*

Spot it "effect", "increase", "accuracy", "quality".

1. Say what happens to quality AND to file size.
2. Give the reason: more bits per sample / per colour.

TYPE 5 **Define a term**

> *"State what is meant by the image resolution" [1]; a table of terms to complete [4]*

Spot it "what is meant by".

1. Use the exact definitions from the table above.

8 What the examiner looks for

"Explain the differences between the ASCII and Unicode character sets" [4] (MAX 3 for one set only)

Marking points (one mark each, up to the marks shown)

- ASCII has fewer / a limited number of characters // Unicode has more characters
- ASCII covers a limited set of / fewer languages
- Unicode includes many more languages and emojis
- ASCII uses 7 / 8 bits per character; Unicode uses up to 16 / 32 bits per character
- ASCII has 128 / 256 characters; Unicode about 65 536 / over 4 billion

KEYWORDS **character set** **bits per character** **languages** **emojis**

"Explain how the analogue sound is recorded and converted to digital" [5]

Marking points (one mark each, up to the marks shown)

- The sound is recorded using a microphone
- The sound wave is sampled ...
- ... measuring the height / amplitude
- Each amplitude has a unique binary value
- The sample rate is set ... that is the number of samples taken per second
- The sample resolution is set ... that is the number of bits used for each sample
- Each sample taken is converted to binary

KEYWORDS **sampled** **amplitude** **sample rate** **sample resolution** **binary value**

Wording trap · Colour depth

THE QUESTION **0478/13/M/J/24 Q2** ↗ Paper 1 · Computer systems **1 mark**

State what is meant by a 16-bit colour depth.

✓ CREDITED

- 16 bits are used to represent each colour in the image
- the number of bits used for each colour / pixel (with the 16)

✗ NOT CREDITED

- the number of colours in the image
- how bright the colours are
- 16 colours

WHY

Colour depth is a number of BITS. 16 bits give 2^{16} = 65 536 colours, not 16.

From: 0478/13/M/J/24 Q2(a) mark scheme

9 Past-paper questions, solved

0478/11/O/N/23 Q5 ↗ Paper 1 · Computer systems **11 marks** ⤢ Full size

5 A musician is recording herself playing the music for a song on the piano.

(a) Explain how the analogue sound is recorded and converted to digital.

..... [5]

(b) State **two** ways that the accuracy of the recording could be increased.

[2]

(d) The musician types the words for the song into a document.

Two character sets that can be used when converting text to digital are the American standard code for information interchange (ASCII) and Unicode.

Explain the differences between the ASCII character set and the Unicode character set.

..... [4]

Swipe sideways to read the whole question

How to start (a) is the six-step sound method. (b) names the two settings. (d) needs points about BOTH character sets, written as comparisons.

▼ **Show a full-mark answer and the mark scheme** · official marking

WHAT TO WRITE

(a) The piano is recorded with a microphone. The sound wave is sampled at regular intervals and the amplitude (height) of the wave is measured. Each amplitude is given a binary value. The sample rate is the number of samples taken per second and the sample resolution is the number of bits used for each sample.

(b) Increase the sample rate. Increase the sample resolution.

(d) ASCII has fewer characters (128, or 256 extended) whereas Unicode has many more. ASCII covers mainly English, whereas Unicode covers many languages and emojis. ASCII uses 7 or 8 bits per character, whereas Unicode uses up to 16 or 32 bits per character.

PART	ANSWER / WORKING	MARKS	GUIDANCE
5(a)	Any five: microphone · wave is sampled · … measuring the amplitude · each amplitude has a unique binary value · sample rate set · … number of samples per second · sample resolution set · … bits per sample · each sample converted to binary	B5	
5(b)	Increase the sample rate · increase the sample resolution	B2	
5(d)	Any four (MAX 3 for ASCII or Unicode alone): fewer / more characters · fewer / more languages, emojis · 7/8 vs 16/32 bits per character · 128/256 vs 65 536 / 4 billion characters	B4	

Total: 11 marks

0478/11/M/J/25 Q3 ↗ Paper 1 · Computer systems **4 marks** ⤢ Full size

3 An image is converted to binary to be represented by a computer.

Complete the table with the missing terms and definition about representing an image.

Term	Definition
	This is the smallest component of an image.
resolution	
	This is the number of bits used to create each colour in an image.
	This is additional data that is stored with an image that can provide information such as the time and date the image was taken.

[4]

Swipe sideways to read the whole question

How to start Two rows give the term, two give the definition. Match each to the four key image words: pixel, resolution, colour depth, metadata.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

pixel: the smallest component of an image.

resolution: the number of pixels high by the number of pixels wide.

colour depth: the number of bits used to create each colour.

metadata: additional data stored with the image, such as the date and time it was taken.

PART	ANSWER / WORKING	MARKS	GUIDANCE
3	pixel / resolution / colour depth (bit depth) / metadata (header), one mark for each correct term or definition	B4	

Total: 4 marks

0478/12/M/J/25 Q2 ↗ Paper 1 · Computer systems **3 marks** ⤢ Full size

2 A student has an image file stored on their computer.

(c) The image file has a colour depth of 32 bits.

(i) State what is meant by a colour depth of 32 bits.

..... [1]

(ii) The colour depth of the image file is changed to 64 bits.

Explain what effect this change has on the image file.

[2]

Swipe sideways to read the whole question

How to start (i) is the definition with the number in it. (ii) asks for the effect on the FILE, with a reason.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

(c)(i) 32 bits are used to represent each colour in the image.

(c)(ii) The file size increases, because the number of bits used to store each colour has doubled.

PART	ANSWER / WORKING	MARKS	GUIDANCE
2(c)(i)	32 bits are used to represent each colour // 2^{32} different colours available	B1	
2(c)(ii)	the size of the image file increases · ... as the number of bits per colour has increased	B2	

Total: 3 marks

10 Common mistakes

✗	✓
"Colour depth is the number of colours"	"The number of bits used for each colour"
"Resolution is how clear the image is"	"The number of pixels (wide × high)"
"Sample rate is how fast the sound is played"	"Number of samples taken per second"
Describing only Unicode in a compare question Max 3 marks for one side only.	A point about each, as a comparison
"Higher sample rate makes the sound louder"	"More accurate / closer to the original; bigger file"

11 Practice

Practice: easy, medium, hard

1. State what is meant by a pixel.
2. How many characters can 7-bit ASCII represent?
3. Define sample resolution.
1. In ASCII, 'm' is 109. Give its 8-bit binary code and its hex.
2. Explain one benefit and one drawback of increasing the colour depth of an image.
3. Describe how the word 'Hi' is stored using a character set.
1. A music app lowers the sample rate and resolution of songs. Explain the effect on the songs and on the files. [4]
2. Give two reasons why Unicode files are larger than ASCII files of the same text.

12 Can I do this?

0 / 5 ticked

- [ ] I can describe how text is converted to binary using a character set.
- [ ] I can compare ASCII and Unicode in four points.
- [ ] I can explain how sound is sampled and define sample rate and resolution.
- [ ] I can define pixel, resolution, colour depth and metadata.
- [ ] I can give the effects of changing resolution, colour depth, sample rate and resolution.

Anything unticked: revise that section first (each line links to it). All ticked: take the quiz.

Answers

Answers to the "Try it yourself" and practice questions on this page. Try every question first.

Try it yourself 1

1.

1. D, E, F, G: 68 + 3

ANSWER **71**

2.

ANSWER **Unicode**

Try it yourself 2

1.

1. $2^8 = 256$

ANSWER **256**

2.

ANSWER **The file size increases / it needs more storage space**

Try it yourself 3

1.

ANSWER **16**

2.

ANSWER **240 000**

Practice: easy, medium, hard

Easy 1.

ANSWER **The smallest component of an image: one square of one colour**

Easy 2.

ANSWER **128**

Easy 3.

ANSWER **The number of bits used for each sample**

Medium 1.

1. 64 + 32 + 8 + 4 + 1
2. 0110 = 6, 1101 = D

ANSWER **0110 1101 · 6D**

Medium 2.

ANSWER **Benefit: a greater range of colours, closer to real life. Drawback: larger file size**

Medium 3.

ANSWER

Each character is looked up in the character set (e.g. ASCII); its unique binary code is stored; H then i in sequence

Hard 1.

ANSWER

Fewer samples per second and fewer bits per sample: less accurate / lower quality, further from the original; smaller files, less storage, quicker to download

PART	ANSWER / WORKING	MARKS	GUIDANCE
	fewer samples per second / bits per sample	B1	
	less accurate / lower quality	B1	
	smaller file size	B1	
	less storage / quicker transmission	B1	
			Total: 4 marks

Hard 2.

ANSWER

Unicode uses more bits per character (16/32 vs 7/8), so each character needs more storage

NOTE 5

Data storage units, file sizes and compression

1 In this note

SYLLABUS 1.3 DATA STORAGE AND COMPRESSION

- 1.3.1 Know the units of data storage, from **bit** to **exbibyte (EiB)**, and convert between them (always **1024**).
- 1.3.2 **Calculate the file size** of an image file and of a sound file, in the unit asked for.
- 1.3.3 Explain the **purpose of and need for compression**.
- 1.3.4 Explain **lossy** and **lossless** compression, including **run-length encoding (RLE)**, and choose one for a situation.

Before this: Text, sound and images (resolution, colour depth, sample rate and resolution). This is the last page of the chapter.

PART 1

Learn the topic

The notes: every idea from the start, with our own examples.

2 What you should already know

Image and sound settings

Resolution = pixels wide × high; colour depth = bits per pixel; sample rate = samples per second; sample resolution = bits per sample.

Quick check: a sound has 8-bit sample resolution: how many bits does one sample use?

8.

Dividing by 1024

1024 = 2^{10}. Halving ten times divides by 1024: 2048 ÷ 1024 = 2, 512 ÷ 1024 = 0.5.

Quick check: what is 3072 ÷ 1024?

3.

3 Units of data storage

Unit	Equal to	Notes
bit (b)	one binary digit, 0 or 1	the smallest unit
nibble	4 bits	one hex digit
byte (B)	8 bits (2 nibbles)	one ASCII character
kibibyte (KiB)	1024 bytes	
mebibyte (MiB)	1024 KiB	
gibibyte (GiB)	1024 MiB	
tebibyte (TiB)	1024 GiB	
pebibyte (PiB)	1024 TiB	
exbibyte (EiB)	1024 PiB	the largest unit on the syllabus

> From byte upwards, every step is **× 1024** going up and **÷ 1024** going down. Never use 1000 in these questions.

EXAM TIP

A way to remember the order: **K**ind **M**onkeys **G**et **T**en **P**eanuts **E**ach.

ANSWER

KiB, MiB, GiB, TiB, PiB, EiB

GENERAL METHOD **Converting between units**

1. Going to a **bigger** unit: divide (÷ 1024 per step; ÷ 8 from bits to bytes).
2. Going to a **smaller** unit: multiply (× 1024 per step; × 8 from bytes to bits).
3. Count the steps between the two units and do one × or ÷ per step.

EXAMPLE How many bits are there in 1 KiB?

KiB → bytes → bits: two steps down, so multiply.

1 × 1024 = 1024 bytes

1024 × 8 = 8192 bits

ANSWER **8192 bits**

TRY IT YOURSELF

1. How many nibbles are in 8 bytes?
2. Convert 512 KiB to MiB.
3. Convert 4 GiB to MiB.
4. How many tebibytes are in 2 PiB?

4 Calculating file sizes

Image file size (bits)

LEARN IT

width × height × colour depth

resolution in pixels, colour depth in bits

Use it when an image with a resolution and a colour depth

Sound file size (bits)

LEARN IT

sample rate × sample resolution × length in seconds

rate in Hz (samples per second), resolution in bits

Use it when a recording with a rate, resolution and length

Bits → bytes → KiB → MiB

LEARN IT

÷ 8, then ÷ 1024, then ÷ 1024

Use it when an answer asked in KiB or MiB

GENERAL METHOD **Any file-size question**

1. Multiply the values given to get the size in **bits** (watch out: a colour depth given in bytes is already bytes).
2. Divide by 8 to get **bytes**.
3. Divide by 1024 for each step up to the unit the question asks for (KiB, then MiB …).
4. Write the answer **in the unit asked**, and show each stage: there are marks for the working.

EXAMPLE **An image is 1024 × 768 pixels with a colour depth of 16 bits. Calculate its size in MiB.**

Bits first, then ÷ 8, then ÷ 1024 twice.

1024 × 768 × 16 = 12 582 912 bits

÷ 8 = 1 572 864 bytes

÷ 1024 = 1536 KiB

÷ 1024 = 1.5 MiB

ANSWER **1.5 MiB**

EXAMPLE **A sound is recorded for 60 seconds at a sample rate of 16 384 Hz with a sample resolution of 16 bits. Calculate the size in KiB.**

Rate × resolution × seconds gives bits.

16 384 × 16 × 60 = 15 728 640 bits

÷ 8 = 1 966 080 bytes

÷ 1024 = 1920 KiB

ANSWER **1920 KiB**

TRY IT YOURSELF

1. An image is 640 × 480 pixels with a colour depth of 8 bits. Calculate the file size in KiB.

2. An image is 1000 × 1000 pixels with a colour depth of 2 bytes. Calculate the file size in bytes.

3. A 5-second sound has a sample rate of 8000 Hz and a sample resolution of 8 bits. Give the size in KiB.

5 Why files are compressed

Compression reduces the **size of a file**. A smaller file:

- needs **less storage space**;
- is **quicker to transmit** (upload, download, send by email);
- needs **less bandwidth** to transmit;
- uses less of a mobile **data allowance**;
- is more likely to fit **file size limits**, such as email attachment limits.

6 Lossy and lossless compression

	Lossy	Lossless
What it does	**permanently removes** some data	**no data is permanently removed**: the **original file can be restored exactly**
File size	much smaller	smaller, but usually less reduction than lossy
Quality	reduced (often not noticed by people)	unchanged
How (examples)	reduce the resolution or colour depth of an image; reduce the sample rate or sample resolution of a sound; remove sounds people cannot hear; JPEG, MP3	**run-length encoding (RLE)**; indexing repeated words or patterns
Use it for	photos, music and video for sharing or streaming	**text** and documents, program files, images or sound that will be edited or printed at full quality

✗ COMMON MISTAKE

Lossy is **never** suitable for text or program files: removing data would change the words or corrupt the program.

7 Run-length encoding (RLE)

Run-length encoding is a lossless method. It finds **runs of repeated data** (characters, or pixels of the same colour) and stores each run **once, with the number of times it repeats**, instead of storing every item.

Original data	RLE	Items stored
AAAAABBBCC (10 characters)	5A 3B 2C	6 instead of 10
a row of pixels: W W W W W W B B W W W W	6W 2B 4W	6 instead of 12

For text, a lossless algorithm can also find **repeated words or phrases**, store each one once in an index, and record the **positions** where it appears.

✗ COMMON MISTAKE

RLE only helps when the data has long runs (simple graphics, large areas of one colour). A photo with every pixel different could even get bigger.

TRY IT YOURSELF

1. Write GGGGGGRRRRB using RLE (number then letter).
2. A file must be emailed and the original must be restored exactly. Which type of compression should be used?

8 Summary

- bit → nibble (4 bits) → byte (8 bits) → KiB → MiB → GiB → TiB → PiB → EiB, × 1024 each step from byte up.
- Image size (bits) = width × height × colour depth. Sound size (bits) = sample rate × resolution × seconds.
- Bits ÷ 8 = bytes; ÷ 1024 = KiB; ÷ 1024 = MiB. Answer in the unit asked, show every stage.
- Compression reduces file size: less storage, quicker transmission, less bandwidth, fits email limits.
- Lossy permanently removes data (smaller, lower quality). Lossless removes nothing permanently (original restored).
- RLE (lossless): runs of repeated characters / pixels stored once with the number of repeats.

IF YOU SEE …

"calculate the file size … in KiB"	→ multiply → ÷ 8 → ÷ 1024, show each stage
text, documents, programs, editing, printing	→ lossless
"describe how lossless compression works"	→ RLE: repeated patterns identified, stored once with the count

PART 2

How it is examined

Question types, what the examiner gives marks for, and real past-paper questions solved.

9 Ways they test it

TYPE 1 Units

> *"State how many kibibytes equal a mebibyte" [1]; "Complete each conversion" [4]; "Tick the largest unit" [1]*

Spot it unit names.

1. Know the ladder and 1024.
2. Show working for conversions with more than one step.

TYPE 2 File size

> *"Calculate the file size of the image in kibibytes. Show all your working" [2-3]*

Spot it resolution + colour depth, or rate + resolution + length.

1. Marks: one for each correct stage of working, one for the answer.
2. Answer in the unit asked.

TYPE 3 Why compress

> *"Give two reasons why the file may have been compressed" [2-3]*

Spot it "compressed", "reason", "benefit".

1. Pick from: less storage · quicker to transmit · less bandwidth · email size limit · less data allowance.

TYPE 4 **Choose and justify**

> *"Explain why the musician would choose lossless compression instead of lossy" [3]*

Spot it both types named.

1. Lossless keeps all the data / original quality.
2. Lossy would permanently remove data and reduce quality.
3. Link to the scenario: editing, printing, text.

TYPE 5 **Describe a method**

> *"Describe how lossless compression compresses the text file" [3-4]*

Spot it "describe how" + lossless.

1. Name RLE.
2. Repeating characters / pixels / patterns are identified ...
3. ... stored once with the number of repeats (and the colour / position).

10 What the examiner looks for

"Explain how the image file is compressed using lossless compression" [3]

Marking points (one mark each, up to the marks shown)

- The size of the file is reduced without permanently removing any data
- A compression algorithm is used ... such as run-length encoding / RLE
- Repeating pixels are identified / grouped // patterns are identified ...
- ... and stored with the number of times they are repeated // indexed

KEYWORDS **RLE** **repeating** **number of times** **no data permanently removed**

"Explain why lossless was chosen instead of lossy" [3]

Marking points (one mark each, up to the marks shown)

- Lossless will keep all the data / the image is not changed
- Lossy would permanently remove data ...
- ... and reduce the quality (of the print / recording)
- Link to the scenario: they want to edit it / print it in detail / it is text

KEYWORDS **permanently** **original** **quality**

Wording trap · Working for a file size

THE QUESTION **0478/12/M/J/24 Q1** ↗ Paper 1 · Computer systems **3 marks**

A 16-bit colour image has a resolution of 512 × 512 pixels. Calculate the file size in KiB. Show all your working.

✓ CREDITED

- 512 × 512 × 16 = 4 194 304 bits, ÷ 8 = 524 288 bytes, ÷ 1024 = 512 KiB
- 512 × 512 = 262 144; × 2 bytes = 524 288 bytes; ÷ 1024 = 512 KiB

✗ NOT CREDITED

- dividing by 1000 (524.288)
- only the final answer, with a slip in it

WHY

Each stage of working earns a mark even if the last number goes wrong. The syllabus always uses 1024.

From: 0478/12/M/J/24 Q1(d) mark scheme

11 Past-paper questions, solved

0478/12/F/M/23 Q3 ↗ Paper 1 · Computer systems **4 marks** ⤢ Full size

3 A new computer comes with primary and secondary storage.

(a) Data storage is measured using binary denominations.

Complete each conversion.

8 bytes = ... nibbles

512 kibibytes (KiB) = ... mebibytes (MiB)

4 gibibytes (GiB) = ... mebibytes (MiB)

1 exbibyte (EiB) = ... pebibytes (PiB)

[4]

Working space

Swipe sideways to read the whole question

How to start For each, count the steps on the unit ladder: going up divide, going down multiply.

▼ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

8 bytes = **16** nibbles · 512 KiB = **0.5** MiB · 4 GiB = **4096** MiB · 1 EiB = **1024** PiB

PART	ANSWER / WORKING	MARKS	GUIDANCE
3(a)	16 · 0.5 · 4096 · 1024	B4	one mark each
			Total: 4 marks

0478/12/M/J/24 Q1 ↗ Paper 1 · Computer systems **3 marks** ⤢ Full size

1 Data can be measured in bits.

(d) A 16-bit colour image has a resolution of 512 pixels wide by 512 pixels high.

Calculate the file size of the image in kibibytes (KiB). Show all your working.

Answer ..... KiB

[3]

Swipe sideways to read the whole question

How to start Pixels × colour depth gives bits; then ÷ 8 and ÷ 1024.

▾ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

512 × 512 = 262 144 pixels · × 16 = 4 194 304 bits · ÷ 8 = 524 288 bytes · ÷ 1024 = **512 KiB**

PART	ANSWER / WORKING	MARKS	GUIDANCE
1(d)	512 × 512 (= 262 144)	M1	
1(d)	× 16 ÷ 8 // × 2 bytes, then 524 288 / 1024	M1	max 2 for working
1(d)	512 (KiB)	A1	

Total: 3 marks

0478/12/O/N/24 Q2 ↗ Paper 1 · Computer systems **5 marks** ⤢ Full size

2 Data storage can be measured using different units of measurement.

(c) A 10 second sound effect is recorded for a movie.

It is recorded with a sample rate of 22,016 Hz and a sample resolution of 8 bits.

Calculate the file size of the sound effect in kibibytes (KiB). Show all your working.

Answer ..... KiB

[3]

(d) The sound effect file is compressed for storage.

(i) State what is meant by file compression.

..... [1]

(ii) Give **one** benefit of compressing the file for storage.

..... [1]

Swipe sideways to read the whole question

How to start Rate × resolution × seconds gives bits; then ÷ 8 and ÷ 1024. (d) is recall.

▾ Show a full-mark answer and the mark scheme · official marking

WHAT TO WRITE

(c) 22 016 × 8 × 10 = 1 761 280 bits · ÷ 8 = 220 160 bytes · ÷ 1024 = **215 KiB**

(d)(i) Reducing the size of a file. (ii) It takes up less storage space.

PART	ANSWER / WORKING	MARKS	GUIDANCE
2(c)	Any two stages: 22 016 × 8 then ÷ 8 · × 10 · 220 160 / 1024	M2	
2(c)	215 KiB	A1	
2(d)(i)	reducing the size of a file	B1	
2(d)(ii)	it will take up / use less storage space	B1	

Total: 5 marks

2 An artist creates an image on a computer.

(c) The artist compresses the image file before uploading it to their website. Users can download the image file from the website to print as a poster.

The image file is compressed using lossless compression.

(i) Explain the reasons why the artist chose lossless compression instead of lossy compression to compress this image file.

..... [3]

(ii) Identify **one** lossless method of compressing an image.

Describe your chosen method.

Method .....

Description .....

[3]

Swipe sideways to read the whole question

How to start (i) contrast the two types and link to the scenario (people print the image as a poster). (ii) name RLE, then describe it.

▼ **Show a full-mark answer and the mark scheme** · official marking

WHAT TO WRITE

(c)(i) Lossless keeps all of the image data, so the downloaded image is exactly the original. Lossy would permanently remove data and reduce the quality, and the artist wants users to print a detailed poster.

(c)(ii) Run-length encoding. Runs of repeating pixels of one colour are identified and each run is stored once, as the number of repetitions and the colour code.

PART	ANSWER / WORKING	MARKS	GUIDANCE
2(c)(i)	Any three: lossless retains all parts of the image // does not delete data · lossy reduces quality · artist wants a detailed image · image may be small anyway / server has space / download time OK	B3	
2(c)(ii)	Run-length encoding (name)	B1	
2(c)(ii)	repeating pixels of one colour / patterns identified · … indexed / put into a table · … as the number of repetitions and the colour code	B2	max 2 for describing
			Total: 6 marks

12 Common mistakes

✗ Dividing by 1000

✓ Divide by 1024

The syllabus says calculations must use 1024.

✗ Forgetting ÷ 8

✓ Bits ÷ 8 = bytes, then ÷ 1024

Image and sound formulas give BITS.

✗ Answer in the wrong unit

✓ Give exactly the unit in the question

✗ "Lossless removes some data"

That is lossy.

✓ Lossless removes no data permanently

✗ "Compression makes the file faster"

Be specific.

✓ "Quicker to transmit / less storage"

13 Practice

Practice: easy, medium, hard

1. How many bits are in a byte?
2. How many MiB are in 3 GiB?
3. Give two benefits of compressing a file.
1. A file is 3072 bytes. Give its size in KiB.
2. An image is 800 × 600 pixels with 24-bit colour depth. Calculate the size in bytes.
3. Explain why lossy compression should not be used for a text document.
1. A 2-minute song is recorded at 32 768 Hz with 16-bit resolution. Calculate the file size in MiB. Show your working.
2. A photographer needs to send 40 photos by email but they are too big. Explain which compression they should use and how it reduces the size. [4]

14 Can I do this?

0 / 5 ticked

- [ ] I know the units from bit to EiB in order and convert with 1024.
- [ ] I can calculate an image file size and a sound file size in the unit asked.
- [ ] I can give reasons for compressing a file.
- [ ] I can compare lossy and lossless and choose one for a situation.
- [ ] I can describe run-length encoding and apply it.

Anything unticked: revise that section first (each line links to it). All ticked: take the quiz.

Answers

Answers to the "Try it yourself" and practice questions on this page. Try every question first.

Try it yourself 1

1.

1. 2 nibbles per byte

ANSWER **16**

2.

ANSWER **0.5**

3.

ANSWER **4096**

4.

ANSWER **2048**

Try it yourself 2

1.

1. 640 × 480 × 8 = 2 457 600 bits
2. ÷ 8 = 307 200 bytes
3. ÷ 1024 = 300 KiB

ANSWER **300**

2.

1. 1000 × 1000 = 1 000 000 pixels
2. × 2 bytes = 2 000 000 bytes (already bytes: no ÷ 8)

ANSWER **2 000 000**

3.

1. 8000 × 8 × 5 = 320 000 bits
2. ÷ 8 = 40 000 bytes
3. ÷ 1024 = 39.0625 KiB

ANSWER **39.0625**

Try it yourself 3

1.

ANSWER **6G 4R 1B**

2.

ANSWER **Lossless**

Practice: easy, medium, hard

Easy 1.

ANSWER **8**

Easy 2.

ANSWER **3072**

Easy 3.

ANSWER **Less storage space; quicker to transmit (or less bandwidth / fits email limits)**

Medium 1.

ANSWER **3**

Medium 2.

1. 800 × 600 × 24 = 11 520 000 bits
2. ÷ 8 = 1 440 000 bytes

ANSWER **1 440 000**

Medium 3.

ANSWER

It permanently removes data, so words / characters would be lost and the document corrupted

Hard 1.

1. 120 s × 32 768 × 16 = 62 914 560 bits
2. ÷ 8 = 7 864 320 bytes
3. ÷ 1024 = 7680 KiB
4. ÷ 1024 = 7.5 MiB

ANSWER **7.5 MiB**

PART	ANSWER / WORKING	MARKS	GUIDANCE
	32 768 × 16 × 120	M1	
	÷ 8 and ÷ 1024 (twice)	M1	
	7.5 MiB	A1	
			Total: 3 marks

Hard 2.

ANSWER

Lossy: permanently removes data, e.g. reduces resolution / colour depth / removes detail people won't notice; much smaller files, quicker to send; quality loss acceptable for viewing