Binary addition, overflow and shifts

Computer systems (Paper 1) · revision notes

Adding two binary numbers of up to 8 bits, explaining overflow errors, and carrying out left and right binary shifts and describing their effect on the value. Expect short calculations (1 to 3 marks) plus 'explain' marks about overflow and lost bits.

Key facts

- 0 + 0 = 0; 0 + 1 = 1; 1 + 1 = 10; 1 + 1 + 1 = 11
- Overflow: the result is too large for the number of bits available
- Right shift n places = ÷ 2^n (any remainder is lost, so the result rounds down)
- Bits pushed off either end are lost
- Largest 8-bit value: 11111111 = 255
- Left shift n places = × 2^n (× 2, × 4, × 8 …)
- Empty places are filled with 0s

Notes

Binary addition

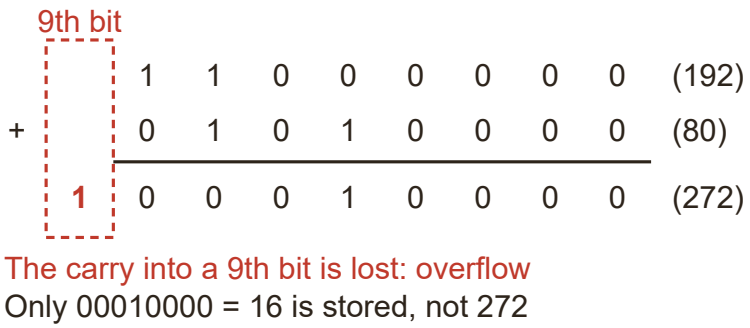
- Rules: 0 + 0 = 0; 0 + 1 = 1; 1 + 1 = 10 (write 0, carry 1); 1 + 1 + 1 = 11 (write 1, carry 1).
- Line the numbers up on the right, add each column from right to left, and write each carry under the next column to the left.
- Example: 93 + 54 = 147

```
  01011101    (93)
+ 00110110    (54)
  11111       carries
  --------
  10010011   (147)
```

- Check your answer by converting all three numbers to denary.

Overflow

- 8 bits can hold a maximum value of 255. A result bigger than 255 needs a 9th bit.
- An **overflow error** happens when the result of a calculation is too large to be stored in the number of bits available.
- The carry out of the most significant bit is lost, so the stored result is wrong. E.g. `11000000` (192) + `01010000` (80) = `100010000` (272), but only `00010000` (16) is stored.

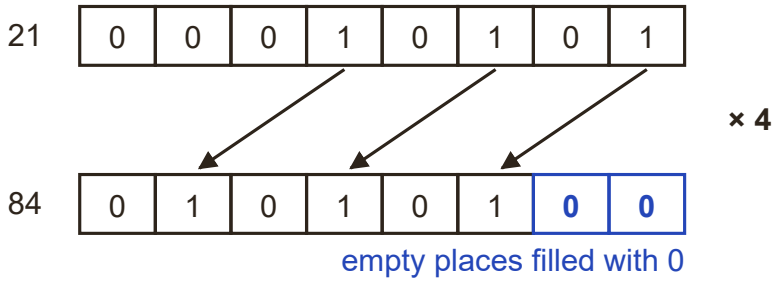


Overflow: the answer needs 9 bits, so the carry out of the most significant bit is lost

- Overflow can cause a program to give wrong results or crash. It can be prevented by using more bits (e.g. 16) or by checking whether the result will fit before storing it.

Binary shifts

- **Left shift**: move every bit left by the given number of places and fill the empty places on the right with 0s. Each place **multiplies** the value by 2: 1 place × 2, 2 places × 4, 3 places × 8.
- E.g. `00010101` (21) shifted left 2 places → `01010100` (84) = 21 × 4.



Left shift 2 places: every bit moves 2 places left, 0s fill the right, value × 4

- **Right shift**: move every bit right and fill the empty places on the left with 0s. Each place **divides** the value by 2.
- E.g. `01101000` (104) shifted right 3 places → `00001101` (13) = 104 ÷ 8.
- Bits pushed off the end are lost. If a 1 is lost on a right shift, the result is rounded down (loss of precision): `00001011` (11) shifted right 1 place → `00000101` (5), not 5.5. Grade 7+ only
- If a 1 is lost off the left on a left shift, the true answer is too big for 8 bits, so the result is wrong: `10100000` (160) shifted left 1 place → `01000000` (64), not 320. Grade 7+ only

How to answer each type of question

Add two 8-bit binary numbers

2 marks | Grade 5

1. Write the numbers one above the other, lined up on the right.
2. Add each column from right to left using the four rules, writing carries clearly.
3. Give the 8-bit answer, and mention overflow if there is a carry out of the left-hand column.
4. Check by converting to denary.

Example. Add the binary numbers 00111001 and 01010110. Give your answer as an 8-bit binary number. Show your working. [2]

Model answer. 10001111: left-hand nibble 1000 (1), right-hand nibble 1111 (1). Check: 57 + 86 = 143.

Explain why an overflow error occurs

2 marks | Grade 6

1. Work out the true result (in denary or as a 9-bit number).
2. Say that it is more than 255 / needs more than 8 bits.
3. Say that the extra bit is lost, so the stored result is wrong.

Example. The 8-bit numbers 10110100 and 01100000 are added together. Explain why an overflow error occurs. [2]

Model answer. The result is 276 (100010100), which is greater than 255 / needs 9 bits (1). Only 8 bits are available, so the carry out of the most significant bit is lost and the stored result (00010100 = 20) is wrong (1).

Carry out a binary shift and state its effect

2 marks | Grade 6

1. Move every bit the stated number of places in the stated direction.
2. Fill the empty places with 0s and keep the answer to 8 bits.
3. State the effect: multiply (left) or divide (right) by 2, 4, 8 …

Example. (a) Perform a left binary shift of 3 places on 00001011. [1]
(b) State the effect of this shift on the value of the number. [1]

Model answer.

(a) 01011000 (1)

(b) The number is multiplied by 8 (11 becomes 88) (1)

Explain the effect of a shift that loses bits 2 marks **Grade 7**

1. Carry out the shift.
2. Identify which 1s have been shifted off the end.
3. Explain the effect: rounded down (right shift) or wrong because of overflow (left shift), using denary values.

Example. The binary number 00100111 is shifted 2 places to the right. Give the result and explain why it is not exactly a quarter of the original value. [2]

Model answer. 00001001 (1). The two right-hand 1s are shifted off and lost, so the result is 9 instead of 39 ÷ 4 = 9.75: it has been rounded down (loss of precision) (1).

Don't lose marks

- Forgetting a carry, especially in a column with 1 + 1 plus a carry.
- Writing a 2 in a column: binary digits can only be 0 or 1.
- Filling empty places with 1s after a shift, or wrapping the lost bits round to the other end.
- Saying overflow means 'the number is too big' without saying 'for the number of bits available'.
- Giving a 9-bit answer when an 8-bit answer is asked for, without mentioning overflow.

More tips

Memory tricks

- Count the 1s in a column, including any carry: 0 → write 0; 1 → write 1; 2 → write 0, carry 1; 3 → write 1, carry 1.
- Left = larger (multiply); right = reduced (divide).
- Check any binary addition in denary: a total over 255 means overflow in 8 bits.
- Shift by n places means × or ÷ by 2^n: 1 place → 2, 2 places → 4, 3 places → 8, 4 places → 16.

Exam technique

- Show the carries: if your final answer has a slip, clear working can still earn a mark.
- Use numbers in overflow explanations: 'the total is 276, but 8 bits can only store up to 255'.
- For shift questions, give the new number with all 8 bits, then the effect (× or ÷ by 2, 4, 8 …).

What each grade needs

What you need to be able to do, from the first marks up to the top grade.

Grade 4 **Apply the four binary addition rules**
0 + 0 = 0, 0 + 1 = 1, 1 + 1 = 10 (write 0, carry 1), 1 + 1 + 1 = 11 (write 1, carry 1).

Grade 5 **Add two 8-bit binary numbers**
Work from right to left, writing each carry clearly under the next column.

Grade 5 **Carry out a left or right shift**

Move every bit the given number of places and fill the empty places with 0s.

Grade 6

Explain what an overflow error is

The result of a calculation is too large to be stored in the number of bits available, e.g. more than 255 in 8 bits.

Grade 6

State the effect of a shift

Each place of left shift multiplies by 2; each place of right shift divides by 2.

Grade 7

Explain the effect of bits being lost

1s shifted off the right make the result round down; 1s shifted off the left make the result wrong (overflow).

Grade 8

Solve problems combining shifts and addition

E.g. multiply by 10 by adding the results of a left shift of 3 places (× 8) and a left shift of 1 place (× 2), and spot when overflow will happen.

Free notes, Full packs, practice papers and mock exams at chhetriacademy.com. Written for this site; not taken from past papers. Stuck? 1-to-1 tutoring with a free 30-minute trial lesson.