NETWORK OF EXCELLENCE

COMPUTER SCIENCE TEACHING

@CSMTJPatterson

Jeanette Patterson

CAS Regional Coordinator - Master Teacher

CTL Technology – Kings Priory School

Jeanette.Patterson@computingatschool.co.uk

Funded by:

Department for Education

The Chartered Institute for IT

Google

Microsoft Research

CPHC
The Council of Professors and Heads of Computing

OCR
RECOGNISING ACHIEVEMENT

AQA

Outcomes

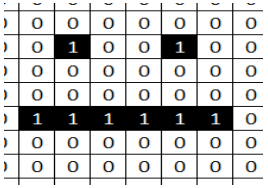
- I am sure you have thought about your expected outcomes for this workshop

Aims

- To give you confidence when dealing with binary numbers, and related topics.
- To provide you with a selection of resources you can utilise and build upon.
- To show you the wealth of resources available from the CAS community and beyond

Representation of Data Workshop

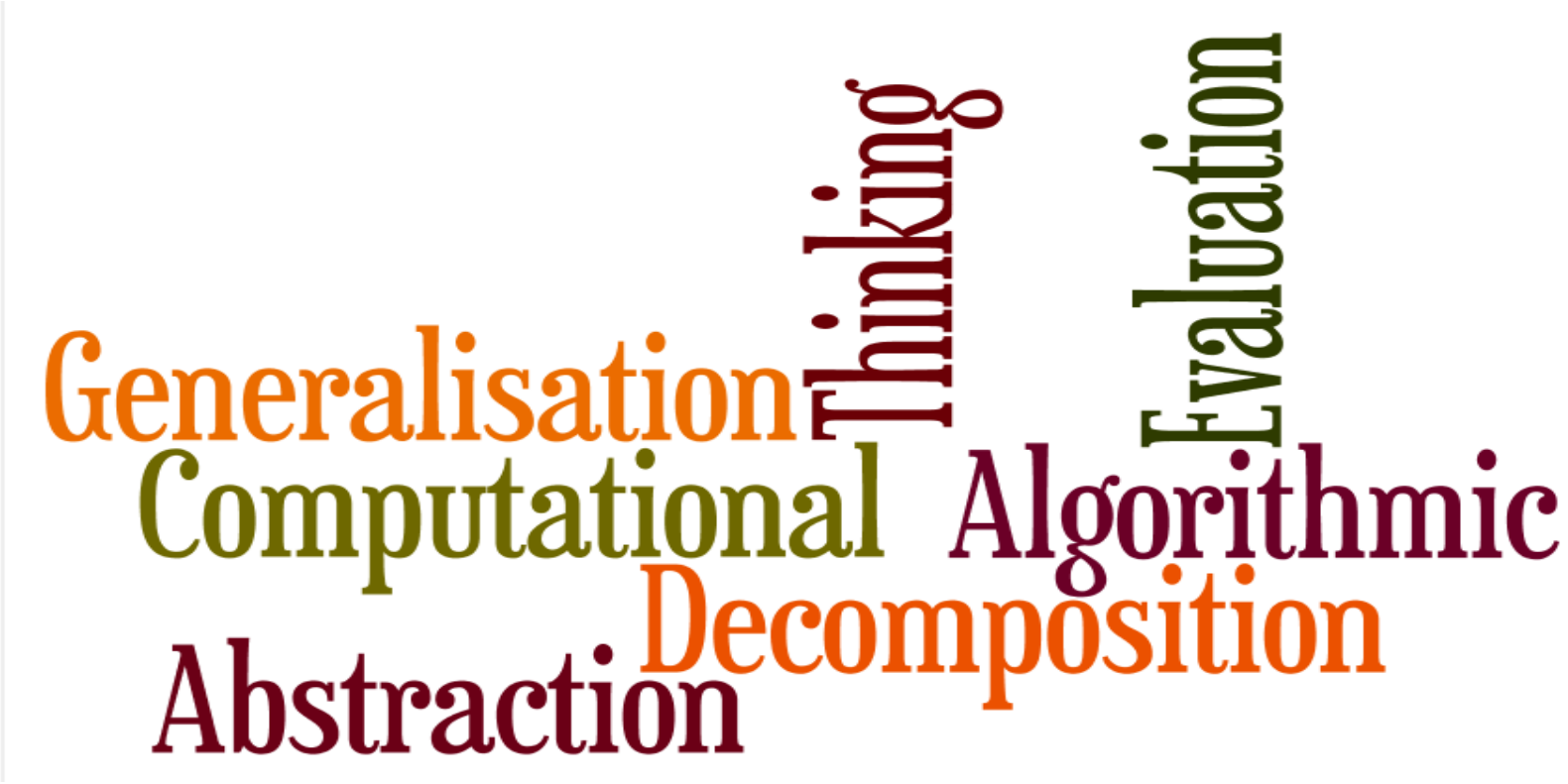
Schedule



- Introduction
- From base 10(denary) to base 2(binary)
- Binary Integers
- Binary Addition
- Twos Complement
- Fixed Point Binary
- Hexadecimal
- *Characters*
- *Graphics and Sound*

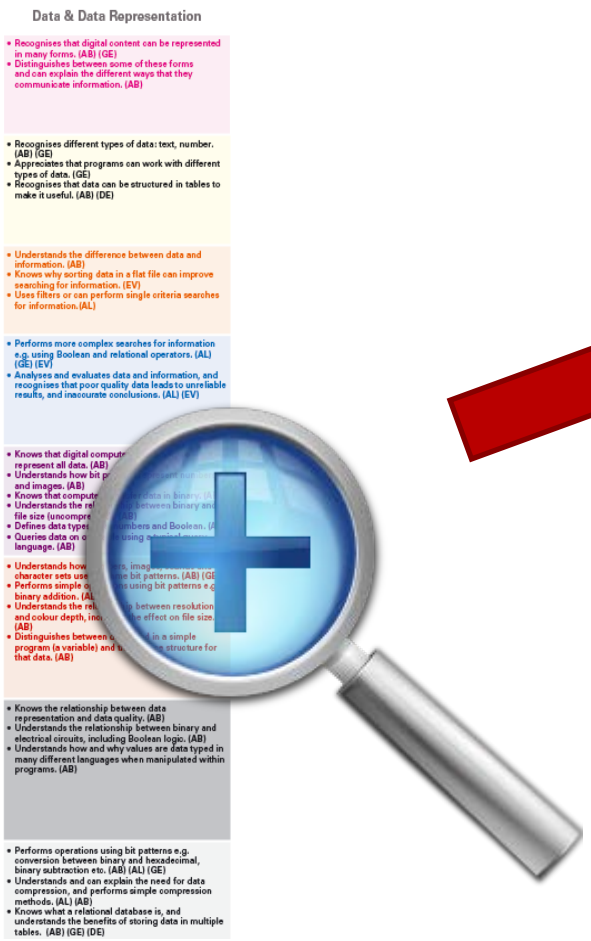
Computational Thinking

- How does binary fit in with computational thinking?



Computational Thinking and Data Representation

- Let us check the progression pathways



• Knows that digital computers use binary to represent all data. (AB)
• Understands how bit patterns represent numbers and images. (AB)
• Knows that computers transfer data in binary. (AB)
• Understands the relationship between binary and file size (uncompressed). (AB)
• Defines data types: real numbers and Boolean. (AB)
• Queries data on one table using a typical query language. (AB)
• Understands how numbers, images, sounds and character sets use the same bit patterns. (AB) (GE)
• Performs simple operations using bit patterns e.g. binary addition. (AB) (AL)
• Understands the relationship between resolution and colour depth, including the effect on file size. (AB)
• Distinguishes between data used in a simple program (a variable) and the storage structure for that data. (AB)
• Knows the relationship between data representation and data quality. (AB)
• Understands the relationship between binary and electrical circuits, including Boolean logic. (AB)
• Understands how and why values are data typed in many different languages when manipulated within programs. (AB)

Data Representation

- Data comes in many forms
- There are several different interpretations of bit patterns including :

Numbers	Integers and Real Numbers
Characters	ASCII and Unicode
Graphics	Images and Videos
Sound	Music and Speech

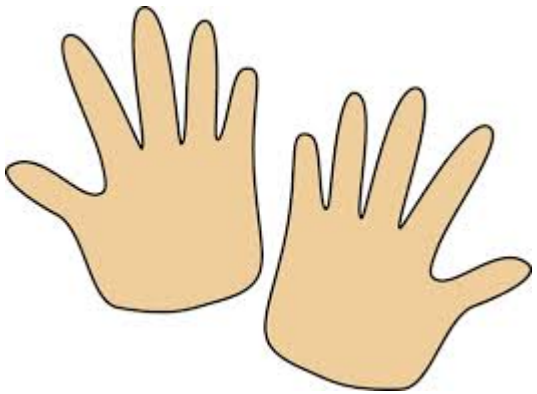
Data Representation – Binary and Denary

Why do computers use **bi**nary (base 2)?

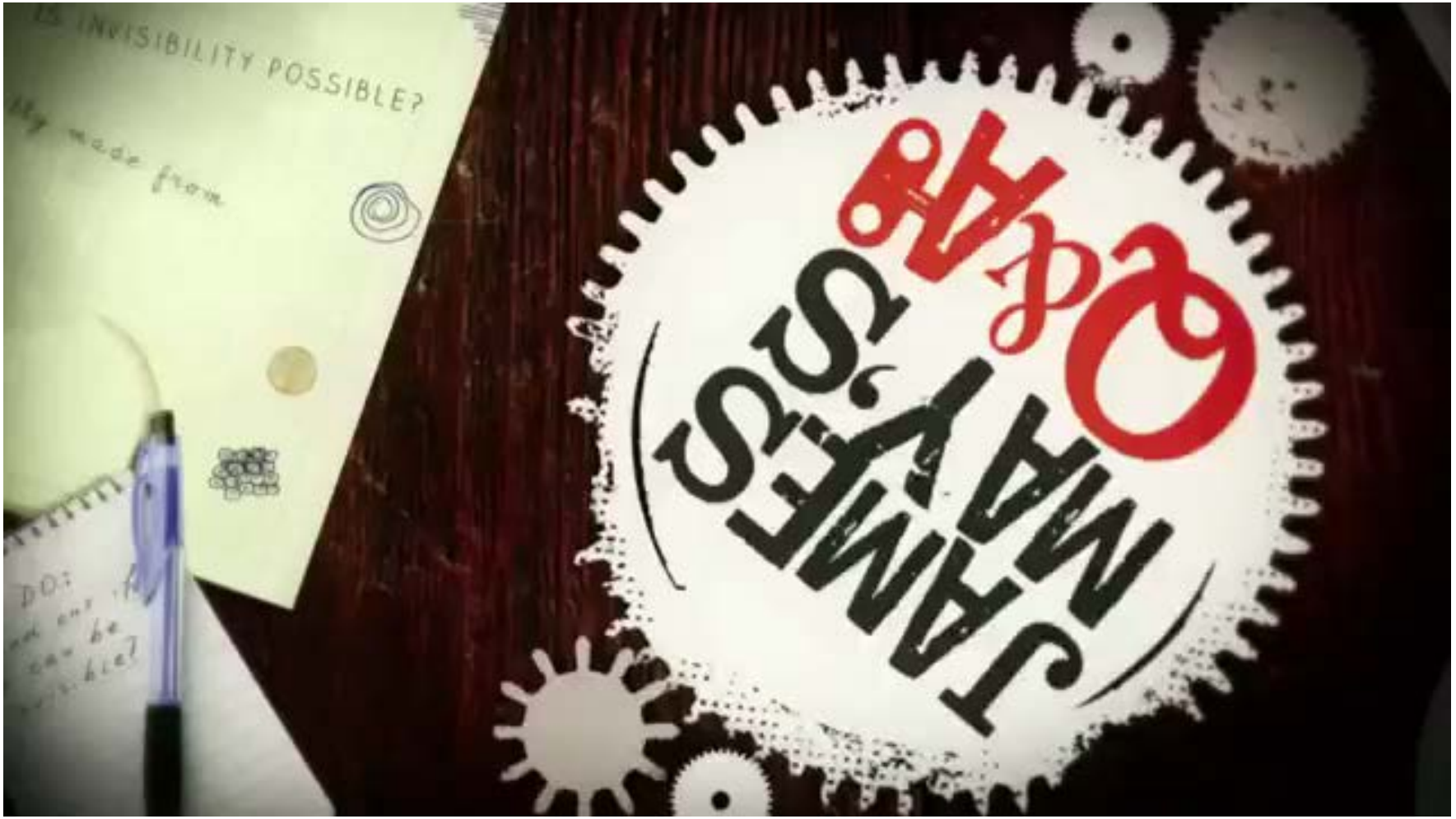
Electronic devices transmitting current around circuits

Why do we use denary (base 10) ?

Probably because we have ten digits



Binary and Denary – Example Starter



Data Representation – Binary and Denary

- Let's go back to primary school

H	T	U
3	2	5

Base ten or denary

Going up by a power of ten each time

- And for Binary or Base two

128	64	32	16	8	4	2	1
0	0	1	0	0	0	1	0

Going up by a power of 2 each time

Data Representation – Binary and Denary

- So denary (base 10) to binary (base 2)

128	64	32	16	8	4	2	1
0	0	1	0	0	0	1	0

(1 X 32) + (1 X 2) = 34

- Binary (base 2) to denary (base 10)

- 57 (base 10) =

128	64	32	16	8	4	2	1
0	0	1	1	1	0	0	1

(base 2)

Now try some for yourself !

Team Game – Transmitting Binary

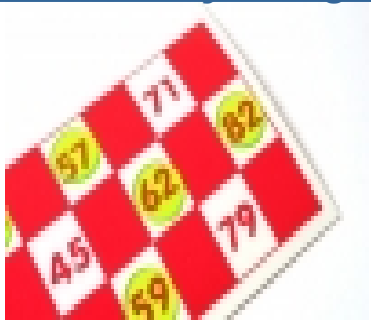
- Get teams of up to 8
- Organise Opposing teams

- They need to transmit the numbers 10, 23, 8, 49
- One team transmits in binary using chairs (sit down for zero, stand up for one)
- The other team transmits using their bodies in the shapes of the numbers

Binary Bingo

- Another great tool to ensure your learners are having fun with binary AND so you can check their progress is BINGO

Let's Play Bingo



I use a great on-line tool called **Triptico-** this has lots of teaching and learning tools. (***Triptico.com)***

Data Representation – Binary Addition

- Adding binary numbers is easy as the rules are quite simple (if you can count to 3)

- 0 + 0 = 0
- 1 + 0 = 1
- 1 + 1 = 10 *(2 in denary)*
- 1 + 1 + = 11 *(3 in denary)*

- Let's try some exercises

Negative Numbers and Two's Complement

- To make a positive binary number negative we can use a method known as **two's complement**

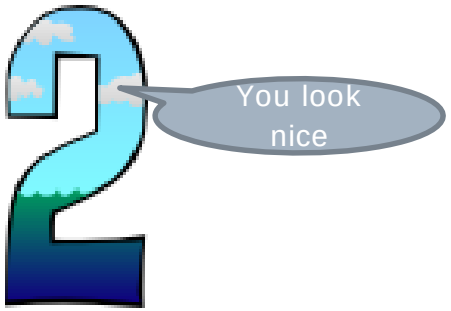
- One method states that :
  - Find the binary value for the equivalent positive number
  - Change all the 0's to 1's and 1's to 0's
  - Add 1 to the result

Negative Numbers and Two's Complement

- Another method :
  - Starting from the right, leave all the digits the same up to and including the first 1
  - Change all the other digits from 0 to 1 or 1 to 0

- I prefer the second method but you may opt for the first if you find that easier

- Let's try some examples



Negative Numbers and Two's Complement

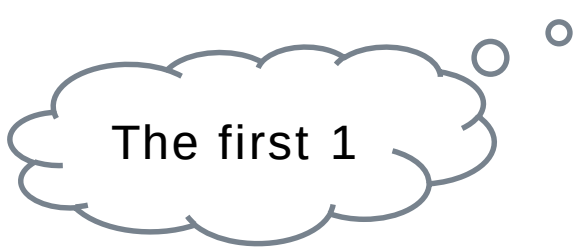
- Let's say we want to represent -14

- Step 1 ..find +14

	128	64	32	16	8	4	2	1
+14	0	0	0	0	1	1	1	0

- Starting from the right, leave all the digits the same up to and including the first 1
- Change all the other digits from 0 to 1 or 1 to 0

	128	64	32	16	8	4	2	1
- 14	1	1	1	1	0	0	1	0



Decimal (Real) Numbers – Fixed Point

- So far we have only been looking at whole numbers (integers)
- In the real world we use decimal, or <u>real numbers</u> all the time:

  - Your bank balance
  - Your mortgage payment
  - Anything you measure (nothing measured can be exact)

Fixed Point Numbers

- We are back to primary school again

In base 10 325.75 can be shown as

100	10	1	.	1/10	1/100
3	2	5	.	7	5

In base 2 36.75 can be shown as

128	64	32	16	8	4	2	1	.	1/2	1/4	1/8	1/16
0	0	1	0	0	1	0	0	.	1	1	0	0

Binary Fractions

- Fixed point notation assumes there is a **binary point** in a set position
- Notice that in decimal the first digit after the point is worth one tenth but in binary it is worth a half
- This means that with the same number of digits after the point binary is less precise, after all we are trying represent base 10 using base 2

Binary Fraction	Fraction	Decimal
0.1	1/2	0.5
0.01	1/4	0.25
0.001	1/8	0.125
0.0001	1/16	0.0625

Hexadecimal

- Humans find large binary numbers difficult to work with and programmers like to work with numbers which are easier to remember

- Hexadecimal uses base 16

- A byte has 8 bits and can be split into 2 nibbles ie. 4 bits each which can then be used to represent a hex digit

The Hex System

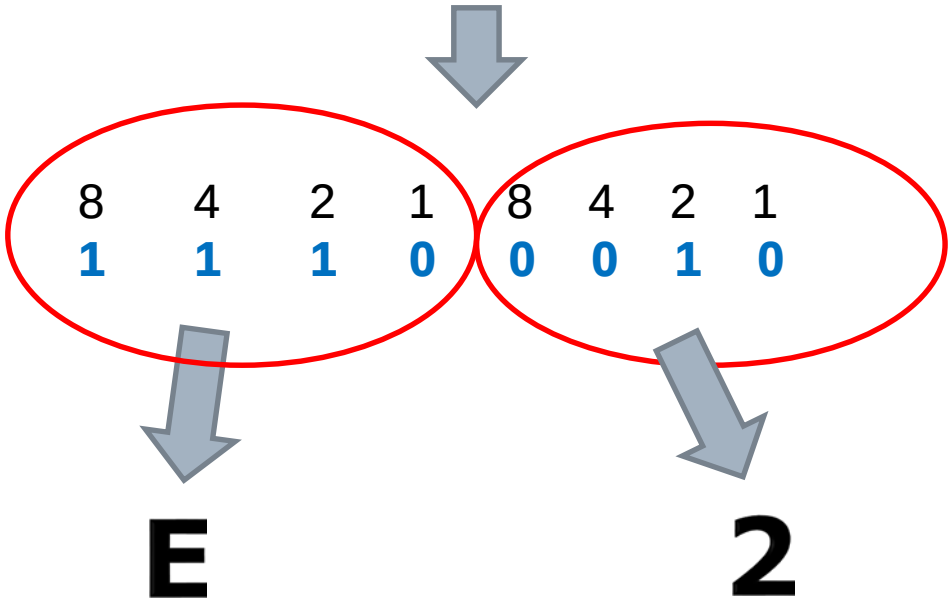
Denary	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Data Representation- Characters

- To Convert from Binary to hex, we split the binary into groups of 4 bits (a nibble)

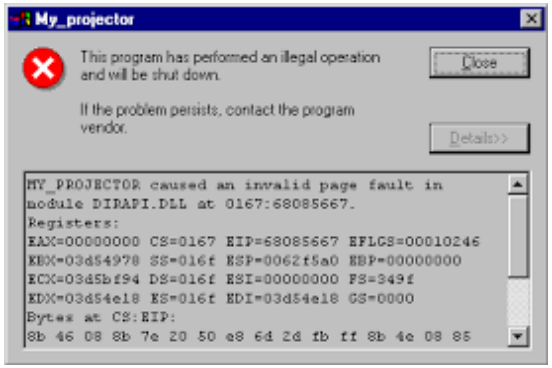
For example

128	64	32	16	8	4	2	1
1	**1**	**1**	**0**	**0**	**0**	**1**	**0**



Uses of Hex

- Used for error messages



- Used to represent colours in images and graphics

- Used in assembly language

LDA #F6D8

Flipped or Blended Learning

- Many pupils need practice
- There are lots of videos and tutorials out there
- You could even make your own

Useful Tools

YouTube
CS Unplugged

CAS Community
Resources / Discussions / Events

Click School

blendspace

OCR MOOC