History Time Line

Pre-History

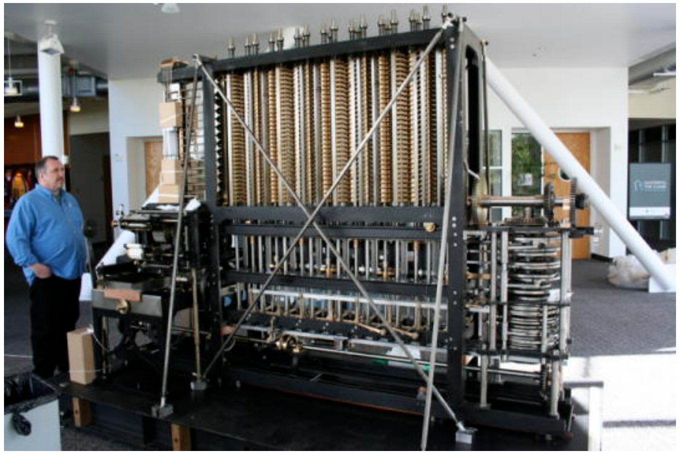
1935

Antikythera mechanism

Analogue computation of planets, moons & festivals

100-150BC

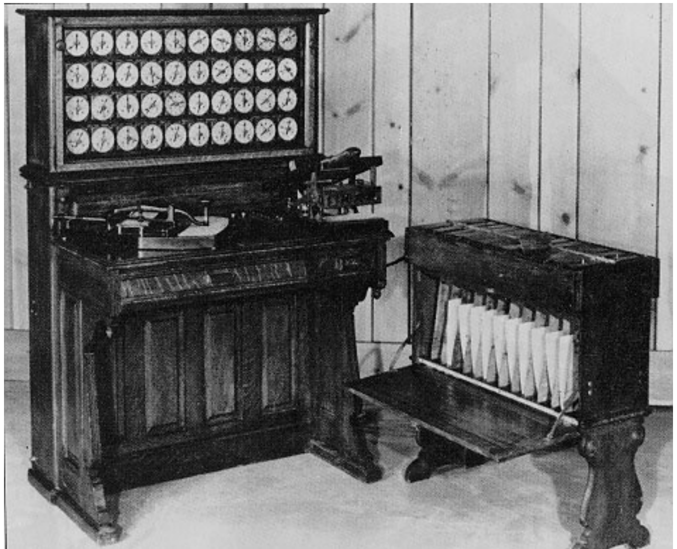
Babbage Difference Engine 1821



Gunnery control units

Analogue

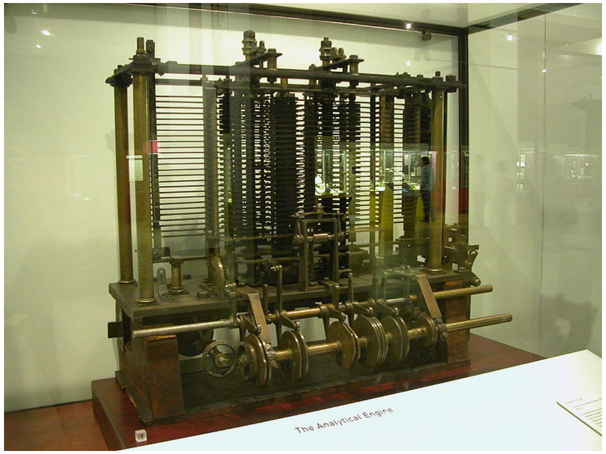
Hollerith. Punch card control. IBM



Jaquard loom

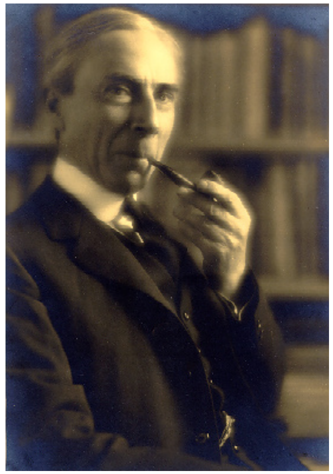
Punch cards

Babbage Analytical Engine 1871



Intellectual Foundations

1830 1945



1925

Principia Mathematica

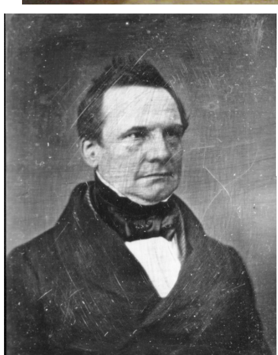
Formalise all of arithmetic

1937

Alan Turing
On computable numbers, with an application to the Entscheidungsproblem

1945

John Von Neumann.
Computer Architecture



Babbage & Lovelace

Stored programs

1931

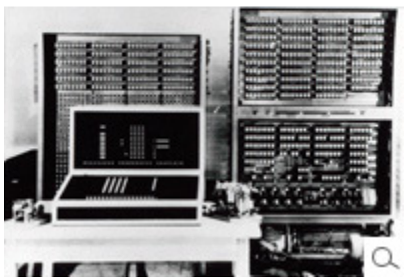
Uber fornal unentsheidbare
Satze der Principia Mathematica
und verwandter Systeme1

Emil Leon Post

History Time Line

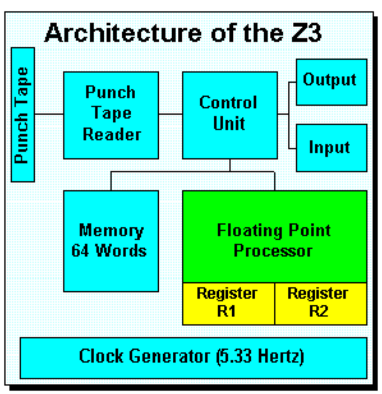
1935 Gestation 1948

1941

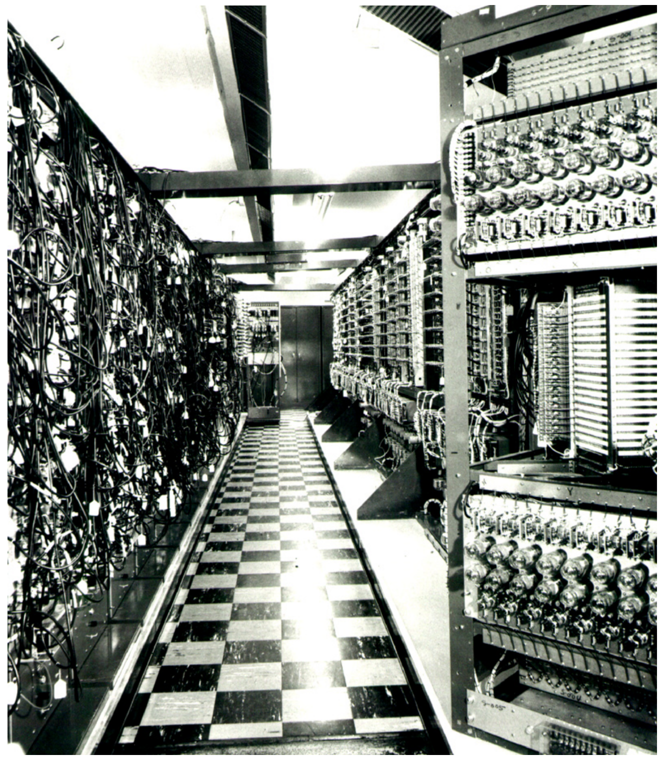


Konrad Zuse Z3

Using 2,300 relays, the Z3 used floating point binary arithmetic and had a 22-bit word length.



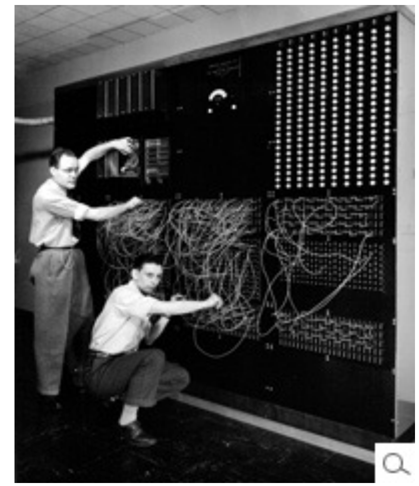
1943



Whirlwind – flight simulator built for US Navy. Completed 1951 – never used

Analogue

1944



Harvard Mark1

Relay-based calculator. The machine had a fifty-foot long camshaft that synchronized the machine's thousands of component parts.

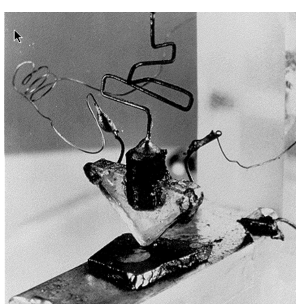
History Time Line

1948 Early Years 1960

1948

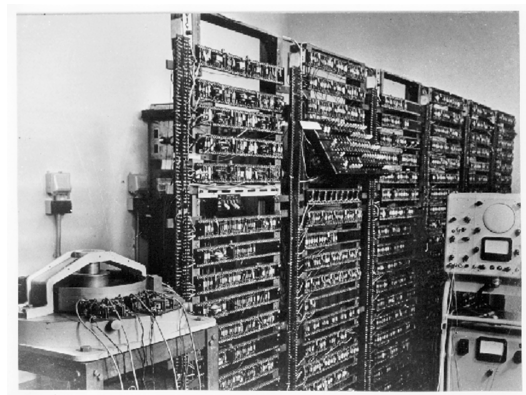
Manchester Baby

First stored program computer

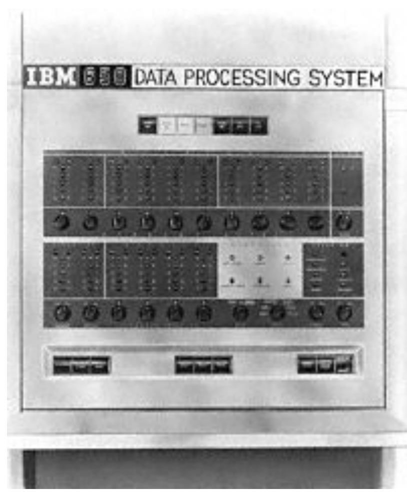


First transistor

First fully transistorised computer. Manchester



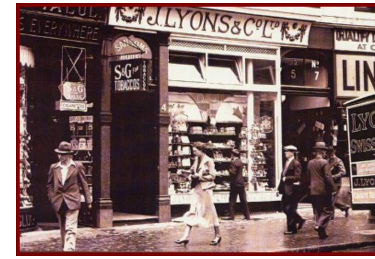
1954



First mass produced computer. Shipped

450 in one year

1951



First commercial application of computers Lyons Electronic Office LEO

Daily scheduling cake production and delivery to Lyons Tea shops

History Time Line

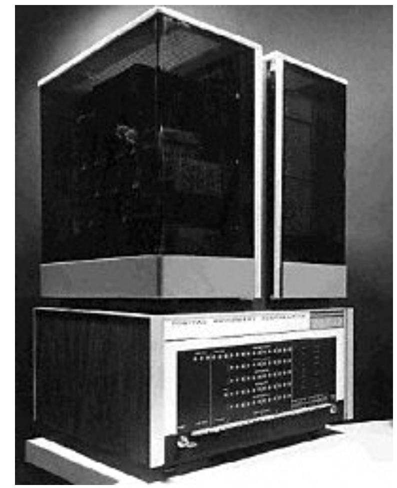
1960 Growing up 1970

1964

IBM 360 family

1964

CDC 6600 first supercomputer 3 MHz



1965

DEC PDP 8 first successful mini-computer

1968

Illiac –iv supercomputer attached to internet

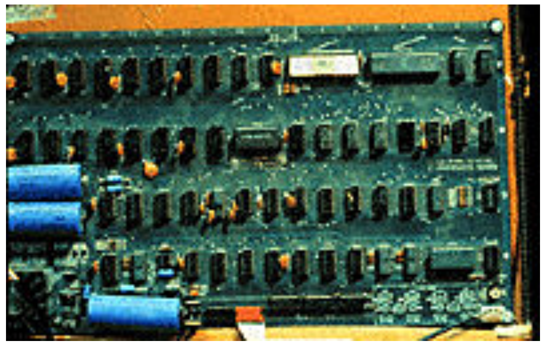
History Time Line

1970 Youth 1980

1971

HP35

1976



Apple 1

1974

Alto

Xerox Parc

Apple II 1977

Dec Vax 11/780

Minicomputer.
Address 4.3 Gbytes memory.

1978 Mandlebrot Set

History Time Line

1980 The personal computer and the world wide web 1990

1981 IBM PC

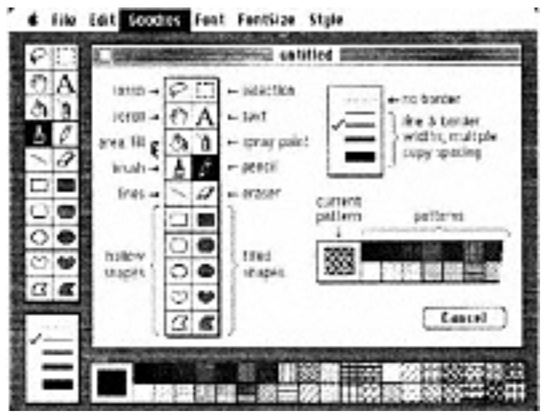
MS-DOS/Intel 8088

1981 Osbourne 1

First portable 10kg

1983 Compaq.

First PC clone



1984 Mac

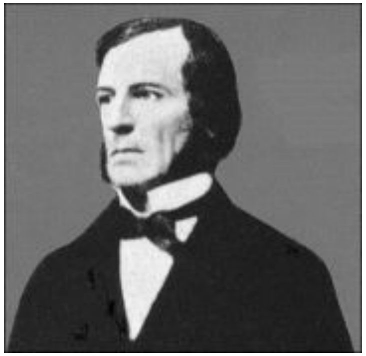
1989 Tim Berners-Lee

WWW

Mike Sendall

1988 Next
Cube

8 Binary

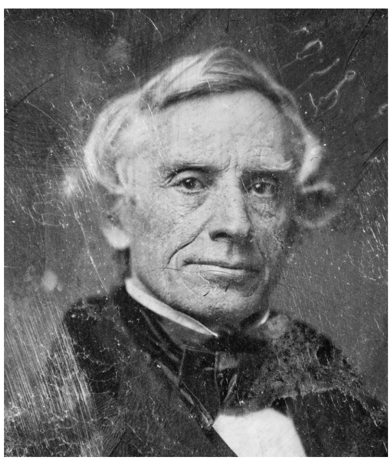


George Boole

History

Often traced back to Boole.

Laws of thought 1854 ....



Samuel Morse

Morse designed his code in 1832.

Is it binary?

Two symbols.

Variable length ... data compression technique

“Gap” is necessary to decode stream.

Computer’s use 1, 0, but also have a word

which is an extra piece of information

Morse code key

Letters		Numbers	
A	•—	1	•————
B	—•••	2	••———
C	—•—•	3	•••——
D	—••	4	••••—
E	•	5	•••••
F	••—•	6	—••••
G	——•	7	——•••
H	••••	8	———••
I	••	9	————•
J	•———	0	—————
K	—•—		
L	•—••		
M	——		
N	—•		
O	———		
P	•——•		
Q	——•—		
R	•—•		
S	•••		
T	—		
U	••—		
V	•••—		
W	•——		
X	—••—		
Y	—•——		
Z	——••		

9 50's-70's

Hardware Costs

Large scale machines

IBM machines only available for rent

Software nearly all 'special purpose'

Hardware costs dominated

Discrete components

Main memory small – and slow. No hierachy.

Local high speed memory restricted – a few **G**eneral **P**urpose **R**egisters (GPR)

Memory access dominated the speed.

As for 'Baby'

Early 'pipeline'

Overlap fetch, decode and execute.

Fetch dominates the number of cycles per instruction.

Reduce the number of instructions required for a programme, by introducing complex instructions.

Complex Instruction Set Computers

Result was multi-cycle instructions to reduce total fetch time.

Hundreds of cycles per instruction.

50's saw the rise of the *High Level Language*

Semantic Gap opened between programmers and architect.

A programmer saw Fortran/Cobol/C statements.

In the 70's advances in hardware allowed some of this gap to be bridged in hardware.

The result was the

Complex Instruction Set Computer (CISC)

(Named only in retrospect with the rise of RISC)

Complexity is unused

IBM discovered in the 70's that a program is dominated by a few instructions.

Load	22%
Conditional Branch	20%
Comparer	16%
Store	12%

Details depend on ISA

Optimise for these ...

- implement them in hardware
- abandon micro-coding of complex instructions
- provide a well engineered optimising compiler

IBM strong in H/W and S/W.

Designed together

Trace back to CDC6600 (1964) a supercomputer

12 MIPS & SPARC

Reduced Instruction Set Computers

Patterson et al at Berkeley looked at VLSI.

Small instruction set meant lots of space on the chips. What could be done with it?

Increased versatility

Introduce large CPU *register file*, global registers accessible to all procedures.

Window registers output from one procedure, input to the next. **SPARC**

Hennessy et al. Stanford **MIPS** concentrated on a pipeline architecture.

Multiple instructions simultaneously

Did not recognise *pipeline hazards* in hardware. Compiler had to identify pipeline hazards and solve them. Compiler writer had to have detailed knowledge of architecture.

Uses single register file with no windows.

Microprocessor without Interlocking Pipeline Stages

Software based approach like Stanley the car that won the DARPA challenge