Number Systems

Logic Design

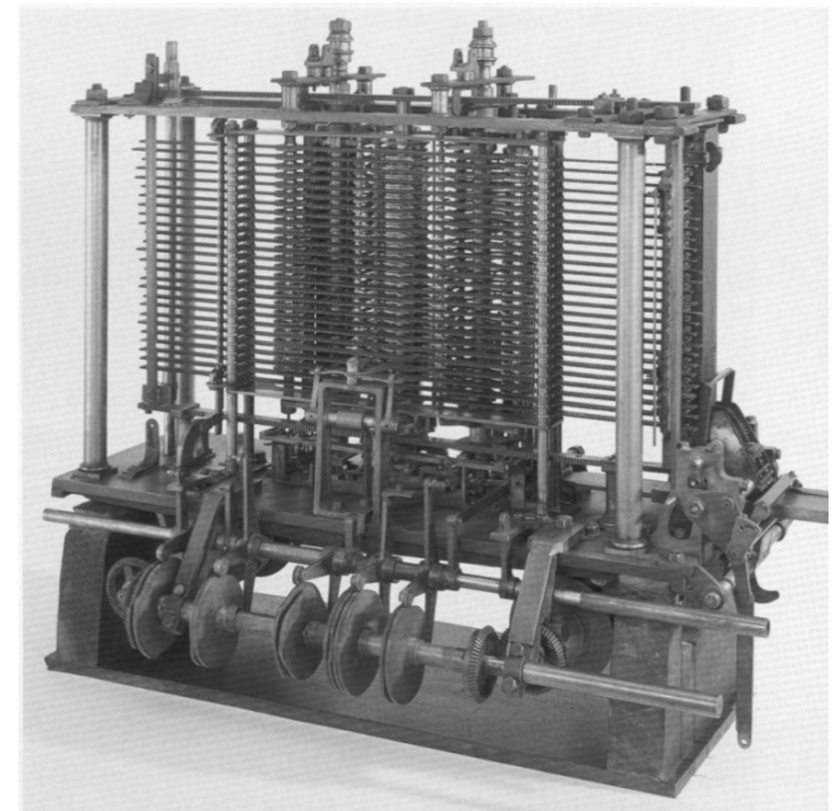
Digital Systems

- Digital discipline
  - **Discrete** voltages instead of **continuous**
  - Simpler to design than analog circuits
    - can build more sophisticated systems
  - Digital systems replacing analog predecessors
    - i.e., digital cameras, digital television, cell phones, CDs

The Digital Abstraction

- Most physical variables are continuous
  - Voltage on a wire
  - Frequency of an oscillation
  - Position of a mass
- Digital abstraction considers **discrete** subset of values

The Analytical Engine



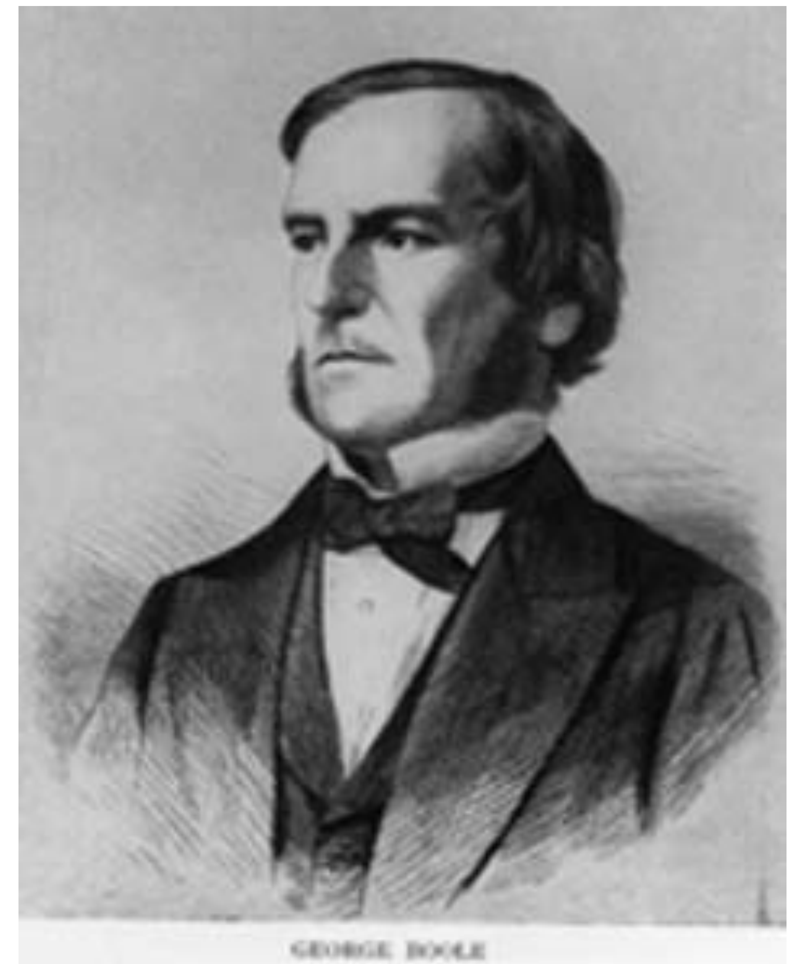
- Designed by Charles Babbage from 1834 – 1871
  - Considered to be the first digital computer
- Built from mechanical gears, where each gear represented a discrete value (0-9)
  - Babbage died before it was finished

Digital Discipline: Binary Values

- **Two discrete values:**
  - 1's and 0's
  - 1, TRUE, HIGH
  - 0, FALSE, LOW
- **1 and 0:** voltage levels, rotating gears, fluid levels, etc.
- Digital circuits use **voltage** levels to represent 1 and 0
- ***Bit*****:** *B*inary dig*it*

George Boole, 1815-1864

- Introduced binary variables
- Introduced the three fundamental logic operations: AND, OR, and NOT



Scanned at the American Institute of Physics

Number Systems

- Decimal numbers

1000's column
100's column
10's column
1's column

$$5374_{10} = 5 \times 10^3 + 3 \times 10^2 + 7 \times 10^1 + 4 \times 10^0$$

five thousands | three hundreds | seven tens | four ones

- Binary numbers

8's column
4's column
2's column
1's column

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10}$$

one eight | one four | no two | one one

Powers of Two

- $2^0 = 1$
- $2^1 = 2$
- $2^2 = 4$
- $2^3 = 8$
- $2^4 = 16$
- $2^5 = 32$
- $2^6 = 64$
- $2^7 = 128$
- $2^8 = 256$
- $2^9 = 512$
- $2^{10} = 1024$
- $2^{11} = 2048$
- $2^{12} = 4096$
- $2^{13} = 8192$
- $2^{14} = 16384$
- $2^{15} = 32768$

Number Conversion

- Binary to decimal conversion:
  - Convert 10011_2 to decimal

- Decimal to binary conversion:
  - Convert 47_{10} to binary

Number Conversion

- Binary to decimal conversion:
  - Convert 10011_2 to decimal
  - $16\times1 + 8\times0 + 4\times0 + 2\times1 + 1\times1 = 19_{10}$

- Decimal to binary conversion:
  - Convert 47_{10} to binary
  - $32\times1 + 16\times0 + 8\times1 + 4\times1 + 2\times1 + 1\times1 = 101111_2$

Decimal to Binary Conversion

- Two methods:
  - **Method 1:** Find the largest power of 2 that fits, subtract and repeat
  - **Method 2:** Repeatedly divide by 2, remainder goes in next most significant bit

Decimal to Binary Conversion

Method 1: Find the largest power of 2 that fits, subtract and repeat

53_{10}

Method 2: Repeatedly divide by 2, remainder goes in next most significant bit

53_{10}

Decimal to Binary Conversion

Method 1: Find the largest power of 2 that fits, subtract and repeat

53_{10}	32×1
53-32 = 21	16×1
21-16 = 5	4×1
5-4 = 1	1×1

= 110101_2

Method 2: Repeatedly divide by 2, remainder goes in next most significant bit

53_{10} =
53/2 = 26 R1
26/2 = 13 R0
13/2 = 6 R1
6/2 = 3 R0
3/2 = 1 R1
1/2 = 0 R1

= 110101_2

Decimal to Binary Conversion

Another example: Convert 75_{10} to binary.

$75_{10} = 64 + 8 + 2 + 1 = 1001011_2$

or

75/2 = 37 R1
37/2 = 18 R1
18/2 = 9 R0
9/2 = 4 R1
4/2 = 2 R0
2/2 = 1 R0
1/2 = 0 R1

Binary Values and Range

- ***N*-digit decimal number**
  - How many values?
  - Range?
  - Example: 3-digit decimal number:

- ***N*-bit binary number**
  - How many values?
  - Range:
  - Example: 3-digit binary number:

Binary Values and Range

- ***N*-digit decimal number**
  - How many values? **10^N**
  - Range? **$[0, 10^N - 1]$**
  - Example: 3-digit decimal number:
    - **10^3 = 1000 possible values**
    - **Range: [0, 999]**

- ***N*-bit binary number**
  - How many values? **2^N**
  - Range: **$[0, 2^N - 1]$**
  - Example: 3-digit binary number:
    - **2^3 = 8 possible values**
    - **Range: [0, 7] = [000_2 to 111_2]**

Hexadecimal Numbers

Shorthand for binary

(Base 16)

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

Hexadecimal to Binary Conversion

- Hexadecimal to binary conversion:
  - Convert $4AF_{16}$ (also written 0x4AF) to binary

- Hexadecimal to decimal conversion:
  - Convert 0x4AF to decimal

Hexadecimal to Binary Conversion

- Hexadecimal to binary conversion:
  - Convert $4AF_{16}$ (also written 0x4AF) to binary
  - $0100\ 1010\ 1111_2$
- Hexadecimal to decimal conversion:
  - Convert $4AF_{16}$ to decimal
  - $16^2 \times 4 + 16^1 \times 10 + 16^0 \times 15 = 1199_{10}$

Bits, Bytes, Nibbles…

- Bits $\underbrace{1}_{\text{most significant bit}}001011\underbrace{0}_{\text{least significant bit}}$

- Bytes & Nibbles $\overbrace{1001\underbrace{0110}_{\text{nibble}}}^{\text{byte}}$

- Bytes $\underbrace{\text{CE}}_{\text{most significant byte}}\text{BF9A}\underbrace{\text{D7}}_{\text{least significant byte}}$

Large Powers of Two

- 2^{10} = 1 kilo $\approx$ 1000 (1024)
- 2^{20} = 1 mega $\approx$ 1 million (1,048,576)
- 2^{30} = 1 giga $\approx$ 1 billion (1,073,741,824)

Estimating Powers of Two

- What is the value of 2^{24}?

- How many values can a 32-bit variable represent?

Estimating Powers of Two

- What is the value of 2^{24}?

  $2^4 \times 2^{20} \approx$ 16 million

- How many values can a 32-bit variable represent?

  $2^2 \times 2^{30} \approx$ 4 billion

Addition

- Decimal

$$\begin{array}{r} 3734 \\ +\ \ 5168 \\ \hline \end{array}$$

- Binary

$$\begin{array}{r} 1011 \\ +\ \ 0011 \\ \hline \end{array}$$

Addition

- Decimal

```
      11   ← carries
     3734
 +   5168
 --------
     8902
```

- Binary

```
      11   ← carries
     1011
 +   0011
 --------
     1110
```

Binary Addition Examples

- Add the following 4-bit binary numbers

$$
\begin{array}{r}
1001 \\
+ \quad 0101 \\
\hline
\end{array}
$$

- Add the following 4-bit binary numbers

$$
\begin{array}{r}
1011 \\
+ \quad 0110 \\
\hline
\end{array}
$$

Binary Addition Examples

- Add the following 4-bit binary numbers

```
    1
   1001
+  0101
-------
   1110
```

- Add the following 4-bit binary numbers

```
  111
   1011
+  0110
-------
  10001
```

Overflow!

Overflow

- Digital systems operate on a **fixed number of bits**
- Overflow: when result is too big to fit in the available number of bits

Signed Binary Numbers

- We have two representation:
  - Sign/Magnitude Numbers
  - Two’s Complement Numbers

Sign/Magnitude Numbers

- 1 sign bit, *N*-1 magnitude bits

$$A: \{a_{N-1}, a_{N-2}, \dots a_2, a_1, a_0\}$$

$$A = (-1)^{a_{N-1}} \sum_{i=0}^{N-2} a_i \, 2^i$$

- Sign bit is the most significant (left-most) bit
  - Positive number: sign bit = 0
  - Negative number: sign bit = 1

Sign/Magnitude Numbers

- Example, 4-bit sign/mag representations of ± 6:

  +6 = **0110**

  \- 6 = **1110**

- Range of an *N*-bit sign/magnitude number:

  $[-(2^{N-1}-1), 2^{N-1}-1]$

Sign/Magnitude Numbers

Problems:

- Addition **doesn't work**, for example -6 + 6:

```
  1110
+ 0110
------
 10100 (wrong!)
```

- Two representations of 0 (± 0):

```
1000
0000
```

Two's Complement Numbers

- Don't have same problems as sign/magnitude numbers:
  - **Addition works**
  - **Single representation for 0**

Two's Complement Numbers

- msb has value of -2^{N-1}

$$A = a_{N-1}(-2^{N-1}) + \sum_{i=0}^{N-2} a_i \, 2^i$$

- Most positive 4-bit number: **0111**
- Most negative 4-bit number: **1000**
- The most significant bit **still indicates the sign** (1 = negative, 0 = positive)
- Range of an *N*-bit two's complement number:

[-(2^{N-1}), 2^{N-1}-1]

“Taking the Two’s Complement”

- “Taking the Two’s complement” **flips the sign** of a two’s complement number
- **Method:**
  1. Invert the bits
  2. Add 1
- **Example:** Flip the sign of $3_{10} = 0011_2$
  1. **1100**
  2. **+ 1**

  $1101 = -3_{10}$

Two's Complement Examples

- Take the two's complement of $6_{10} = 0110_2$

- What is the decimal value of the two's complement number 1001_2?

Two's Complement Examples

- Take the two's complement of $6_{10} = 0110_2$
    1. 1001
    2. $+ \quad 1$

    $1010_2 = -6_{10}$

- What is the decimal value of the two's complement number 1001_2?
    1. 0110
    2. $+ \quad 1$

    $0111_2 = 7_{10}$, so $1001_2 = -7_{10}$

Two’s Complement Addition

- Add 6 + (-6) using two’s complement numbers

$$\begin{array}{r} 0110 \\ +\ \ 1010 \\ \hline \end{array}$$

- Add -2 + 3 using two’s complement numbers

$$\begin{array}{r} 1110 \\ +\ \ 0011 \\ \hline \end{array}$$

Two's Complement Addition

- Add 6 + (-6) using two's complement numbers

```
  111
    0110
+   1010
--------
   10000
```

- Add -2 + 3 using two's complement numbers

```
  111
    1110
+   0011
--------
   10001
```

Increasing Bit Width

Extend number from *N* to *M* bits (*M* > *N*) :

- Sign-extension
- Zero-extension

Sign-Extension

- Sign bit copied to msb's
- Number value is same
- **Example 1:**
  - 4-bit representation of 3 = **0**011
  - 8-bit sign-extended value: **0000**0011
- **Example 2:**
  - 4-bit representation of -5 = **1**011
  - 8-bit sign-extended value: **1111**1011

Zero-Extension

- Zeros copied to msb's
- Value changes for negative numbers
- **Example 1:**
  - 4-bit value = 0011 = 3_{10}
  - 8-bit zero-extended value: **0000**0011 = 3_{10}
- **Example 2:**
  - 4-bit value = 1011 = -5_{10}
  - 8-bit zero-extended value: **0000**1011 = 11_{10}

Number System Comparison

Number System	Range
Unsigned	$[0, 2^N-1]$
Sign/Magnitude	$[-(2^{N-1}-1), 2^{N-1}-1]$
Two's Complement	$[-2^{N-1}, 2^{N-1}-1]$

For example, **4-bit** representation:

	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Unsigned									0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Two's Complement	1000	1001	1010	1011	1100	1101	1110	1111	0000	0001	0010	0011	0100	0101	0110	0111								
Sign/Magnitude		1111	1110	1101	1100	1011	1010	1001	0000 1000	0001	0010	0011	0100	0101	0110	0111								