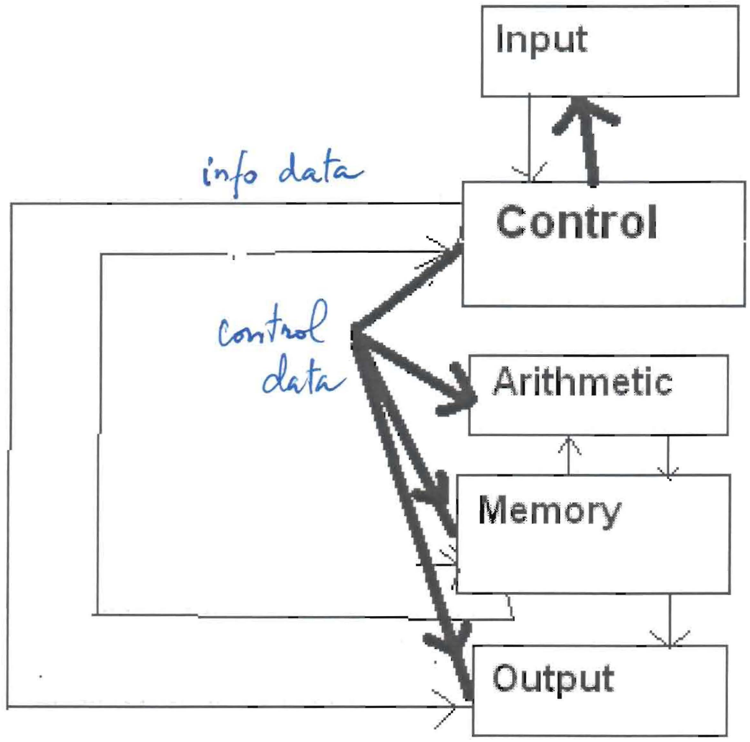
Engin 241
Digital Systems
Spring '10

Basic organization of a digital computer:



Analog & Digital Representations

Analog → Decimal $\xleftrightarrow[\text{Number System}]{\text{Positional}}$ Digital → Binary

$872.64 =$

$8\times10^2 + 7\times10^1 + 2\times10^0 + 6\times10^{-1} + 4\times10^{-2}$

$1001.1 =$

$1\times2^3 + 0\times2^2 + 0\times2^1 + 1\times2^0 . 1\times2^{-1} =$

$9 . 5$

$8.25 =$

$8\times10^0 + 2\times10^{-1} + 5\times10^{-2}$

⟷

$1000.01 =$

8.25

Addition

$$
\begin{array}{r}
872.64 \\
+\quad 8.25 \\
\hline
880.89
\end{array}
\qquad\qquad
\begin{array}{r}
1001.1 \\
+\ 1000.01 \\
\hline
10001.11
\end{array}
$$

Subsfraction

$$
\begin{array}{r}
872.64 \\
-\quad 8.25 \\
\hline
864.39
\end{array}
\qquad\qquad
\begin{array}{r}
1001.1 \\
-\ 1000.01 \\
\hline
0001.01_{(2)}
\end{array}
$$

→ $1.25_{(10)}$ ✓

Multiplication

$$
\begin{array}{r}
872.64 \\
\times \quad 8.25 \\
\hline
436320 \\
174528 \\
698112 \\
\hline
7199.2800
\end{array}
$$

$$
\begin{array}{r}
1000.01_{(2)} \\
\times \quad 1001.1_{(2)} \\
\hline
100001 \\
100001 \\
100001 \\
\hline
1001110.011_{(2)}
\end{array}
\qquad
\begin{array}{r}
8.25 \\
9.5 \\
\hline
4125 \\
7425 \\
\hline
78.375_{(10)}
\end{array}
$$

$64 + 8 + 4 + 2 + 0.25 + 0.125 = 78.375$ ✓

Division:

$$
\begin{array}{r}
109.08 \\
8 \;\big)\; 872.64 \\
072 \\
-\;72 \\
\hline
0064 \\
-\;64 \\
\hline
00
\end{array}
$$

$$\frac{1001.1}{1000}$$

$$
\begin{array}{r}
1.0011_{(2)} \\
1000 \;\big)\; 1001.1 \\
-\,1000 \\
\hline
0001100 \\
-\,1000 \\
\hline
01000 \\
-\,1000 \\
\hline
0000
\end{array}
$$

$$1.0011_{(2)} = 1\times 2^{0} + 1\times 2^{-3} + 1\times 2^{-4}$$
$$= 1 + 0.125 + 0.0625$$
$$= 1.1875_{(10)}$$

$\frac{9.5}{8} = 1.1875$ ✓

Digital representations:

Base 2	Binary	0, 1
Base 3	Ternary	0, 1, 2
Base 4	Quaternary	0, 1, 2, 3
Base 5	Quinary	0, 1, 2, 3, 4
Base 8	Octal	0, 1, 2, 3, 4, 5, 6, 7
Base 10	Decimal	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Base 12	Duodecimal	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B
Base 16	Hexadecimal	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

Decimal	Binary	Ternary	Octal	Hexadecimal
0	0	0	0	0
1	1	1	1	1
2	10	2	2	2
3	11	10	3	3
8	1000	22	10	8
17	10001	122	21	11
⋮				

(5)

$8_{(10)}$

Binary (0,1)

2^3	2^2	2^1	2^0
1	0	0	$0_{(2)}$

Ternary (0,1,2)

3^3	3^2	3^1	3^0
0	0	2	$2_{(3)}$

$17_{(10)}$

Binary (0,1)

2^4	2^3	2^2	2^1	2^0
1	0	0	0	$1_{(2)}$

Ternary (0,1,2)

3^3	3^2	3^1	3^0
0	1	2	$2_{(3)}$

Octal (0,1,2,3,4,5,6,7)

8^2	8^1	8^0
0	2	$1_{(8)}$

Hexadecimal (0,1,2,3,4,5,6,7,8,9, A,B,C,D,E,F)

16^2	16^1	16^0
0	1	$1_{(16)}$

Decimal → Binary, Ternary, Octal, Hexadecimal

We have used polynomial method of conversion (using powers of the base)

241 Labs → Mondays 3-5 pm in S-3-126.
1st Lab on Feb 1st, 2010.
TA: Lanting Cheng (lanting.cheng001@umb.edu)
Office hours: Mondays 12-3 pm & Wednesdays 9-11AM in S-3-126

Number Conversions:

Convert $43_{(10)}$ to its binary equivalent:

Method #1 : looking positions in binary representation

$$\begin{array}{cccccc} 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\ 1 & 0 & 1 & 0 & 1 & 1_{(2)} \end{array}$$

Method #2: iterative:

$\frac{43}{2} = 21 \; R\underline{1} \rightarrow$ Binary digit of order 0

$\frac{21}{2} = 10 \; R\underline{1} \rightarrow$ " " " " 1

$\frac{10}{2} = 5 \; R\underline{0} \rightarrow$ " " " " 2

$\frac{5}{2} = 2 \; R\underline{1} \rightarrow$ " " " " 3

$\frac{2}{2} = 1 \; R\underline{0} \rightarrow$ " " " " 4

$\frac{1}{2} = 0 \; R\underline{1} \rightarrow$ " " " " 5

$\Rightarrow \underline{1}01011_{(2)}$

Convert $201_{(3)}$ into its binary equivalent:

Method #1: $201_{(3)} = 2\times 3^2 + 0\times 3^1 + 1\times 3^0 = 19_{(10)}$

$19_{(10)}$:	2^4	2^3	2^2	2^1	2^0
	1	0	0	1	$1_{(2)}$

Method #2: iterative:

$\frac{201_{(3)}}{2} = 100_{(3)}\ R\,\underline{\underline{1}} \rightarrow$ binary digit order 0

$\frac{100_{(3)}}{2} = 11_{(3)}\ R\,\underline{\underline{1}} \rightarrow$ " " " 1

$\frac{11_{(3)}}{2} = 2_{(3)}\ R\,\underline{\underline{0}} \rightarrow$ " " " 2

$\frac{2_{(3)}}{2} = 1_{(3)}\ R\,\underline{\underline{0}} \rightarrow$ " " " 3

$\frac{1_{(3)}}{2} = 0_{(3)}\ R\,\underline{\underline{1}} \rightarrow$ " " " 4

$201_{(3)} \Rightarrow 10011_{(2)}$

Special Conversion Methods:

(2) $\leftrightarrow$ (8)   (2) $\leftrightarrow$ (16)   etc..
Binary   Octal   Binary   Hexadecimal

$011\ 111\ 101.\ 001\ 100_{(2)}$

Grouping by 3

$3\ 7\ 5\ .\ 1\ 4_{(8)} = 2^{3}$ → Group every 3 binary digits

$173.24_{(8)}$

Splitting into 3

$001\ 111\ 011.\ 010\ 100_{(2)}$

$173.24_{(8)} = 001111011.010100_{(2)}$

$0010\ 1011\ 0110.\ 1110_{(2)}$

Grouping by 4

$2\ B\ 6\ .\ E_{(16)} = 2^{4}$ → Grouping every 4 binary digits

$3\ A\ B\ .\ 2_{(16)}$

Splitting into 4

$0011\ 1010\ 1011\ .\ 0010$

$3AB.2_{(16)} = 001110101011.0010_{(2)}$

Signed Numbers using Complements:

Decimal number 123.45

10's complement: $10^3 - 123.45 = 1000 - 123.45$
(r's complement) $= 876.55$

$\Rightarrow$ Positive signed : $0_s\,123.45$
Negative signed : $1_s\,876.55$

9's complement: $10^3 - 10^{-2} - 123.45 = 1000 - 0.01 - 123.45$
(r-1)'s complement $= 876.54$

$\Rightarrow$ Positive signed: $0_s\,123.45$
Negative signed: $1_s\,876.54$

Binary number: $1101.011_{(2)}$

2's complement: $10^4_{(2)} - 1101.011_{(2)}$
(r's complement)

$10000_{(2)} - 1101.011_{(2)} = 0010.101_{(2)}$

$$
\begin{array}{r}
10000 \\
-\ 1101.011 \\
\hline
0010.101
\end{array}
$$

Positive signed: $0_s\,1101.011_{(2)}$
Negative signed: $1_s\,0010.101_{(2)}$

1's complement : $10^{4}_{(2)} - 10^{-3}_{(2)} - 1101.011_{(2)}$

(r-1)'s complement.

$10000_{(2)} - 0.001_{(2)} - 1101.011_{(2)}$

$= 0010.100_{(2)}$

$$\begin{array}{r} 0010.101 \\ -\quad 0.001 \\ \hline 0010.100 \end{array}$$

Positive signed: $0_s\,1101.011_{(2)}$

Negative signed: $1_s\,0010.100_{(2)}$

Substraction with r's complements:

Decimal: $N_1 = 532$; $N_2 = 146$

10's complement of N_2 is $\overline{\overline{N_2}} = 10^3 - 146 = 854$

Conventional Substraction

$$\begin{array}{r} 532 \\ -\ 146 \\ \hline 386 \end{array}$$

$N_1 - N_2$

Substraction by adding the 10's complement

$$\begin{array}{r} 532 \\ +\ 854 \\ \hline 1\,|\,386 \end{array}$$

$N_1 + \overline{\overline{N_2}}$

end carry is ignored!

Binary: $N_1 = 11101.11$; $N_2 = 01011.10$

2's complement of N_2 is $\overline{\overline{N}}_2 = 10^5_{(2)} - 01011.10$

$= 10100.10$

Conventional substraction	Substraction using the 2's complement
$\begin{array}{r} 11101.11 \\ -\ 01011.10 \\ \hline 10010.01 \end{array}$	$\begin{array}{r} 11101.11 \\ +\ 10100.10 \\ \hline \boxed{1}\,10010.01 \end{array}$
	end carry is ignored!
$N_1 - N_2$	$N_1 + \overline{\overline{N}}_2$

Codes:

BCD or Binary Coded Decimal scheme: coding the 1st ten decimal numbers using a 4-bit binary representation

Decimal	BCD or $\boxed{8421}$ code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

$2^3\ 2^2\ 2^1\ 2^0$

Just to establish a code, we could also use 2421 code, 5421 code, 7536 code, or Biquinary code. 50 43210

Decimal	8421	2421	5421	+ + +(−) 7536	Biquinary
0	0000	0000	0000	0000	0100001
1	0001	0001	0001	1001	0100010
2	0010	0010	0010	0111	0100100
3	0011	0011	0011	0010	0101000
4	0100	0100	0100	1011	0110000
5	0101	0101	1000	0100	1000001
6	0110	1100	1001	1101	1000010
7	0111	1101	1010	1000	1000100
8	1000	1110	1011	0110	1001000
9	1001	1111	1100	1111	1010000

(2 1's in each number) one 1 in each group.

Weighted code:

4th	3rd	2nd	1st
8	4	2	1
2	4	2	1
5	4	2	1
7	5	3	$\overset{(-)}{6}$

Biquinary. → Built in error detection

Non-weighted code:
- Excess-three (XS-3) : 8421 + 3 code
- 2-out-of-5 (2 ones out of 5 digit binary code)
  → Builtin error detection.

Decimal	XS-3	2-out-of-5
0	0011	11000
1	0100	00011
2	0101	00101
3	0110	00110
4	0111	01001
5	1000	01010
6	1001	01100
7	1010	10001
8	1011	10010
9	1100	10100

Draw USPS bar code for 02125-3393:

Frame bar 0 2 1 5 1 3 3 9 3 Frame bar

Unit-distance Codes : very useful in analog-digital conversion.

Decimal	8421	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101

Gray Code : only one bit change between one number to the next.

- One flip from a number to the next.
- Non-weighted.
- This helps reduce A-D conversion errors.

Table 2.10 The 7-bit American Standard Code for Information Interchange (ASCII)

	$b_6 b_5 b_4$							
b_3 b_2 b_1 b_0	000	001	010	011	100	101	110	111
0 0 0 0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	STX	DC2	"	2	B	R	b	r
0 0 1 1	ETX	DC3	#	3	C	S	c	s
0 1 0 0	EOT	DC4	$	4	D	T	d	t
0 1 0 1	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	ACK	SYN	&	6	F	V	f	v
0 1 1 1	BEL	ETB	'	7	G	W	g	w
1 0 0 0	BS	CAN	(	8	H	X	h	x
1 0 0 1	HT	EM	)	9	I	Y	i	y
1 0 1 0	LF	SUB	*	:	J	Z	j	z
1 0 1 1	VT	ESC	+	;	K	[	k	{
1 1 0 0	FF	FS	,	<	L	\	l	|
1 1 0 1	CR	GS	-	=	M	]	m	}
1 1 1 0	SO	RS	.	>	N	^	n	~
1 1 1 1	SI	US	/	?	O	—	o	DEL

Control Characters

NUL	Null	DC1	Device Control 1
SOH	Start of Heading	DC2	Device Control 2
STX	Start of Text	DC3	Device Control 3
ETX	End of Text	DC4	Device Control 4
EOT	End of Transmission	NAK	Negative Acknowledge
ENQ	Enquiry	SYN	Synchronous Idle
ACK	Acknowledge	ETB	End of Transmission Block
BEL	Bell	CAN	Cancel
BS	Backspace	EM	End of Medium
HT	Horizontal Tab	SUB	Substitute
LF	Line Feed	ESC	Escape
VT	Vertical Tab	FS	File Separator
FF	Form Feed	GS	Group Separator
CR	Carriage Return	RS	Record Separator
SO	Shift Out	US	Unit Separator
SI	Shift In	SP	Space
DLE	Data Link Escape	DEL	Delete