Introduction to Computing (ECE 306H) – Fundamental Ideas of Computing

Dawson Zhang | Instructor: Dr. Yerraballi

Fall 2025

1 "Computing" in a Nutshell

1.1 Physics in Computing

- Voltages
- Bits

1.2 Math in Computing

- Logic - NOT, AND, OR
- Numbers - int, char

1.3 Higher Level Concepts

- Circuits - combinatoric logic, sequential logic
- Computers
  - ISA Architechture - x86, x64, ARM, RISC-V

    > **Remark 1.1.** ISA's have syntaxical differences between their assembly syntaxes because of the separate choices that were taken to develop the architechture.

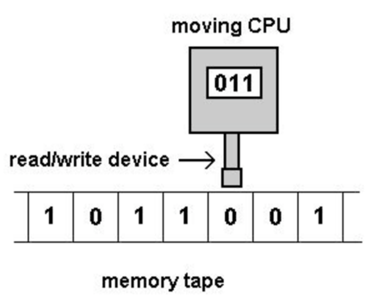
  - **LC3** is what we will be working on

    > **Remark 1.2.** Dr. Y believes in a "META" view of ISA Architechture so instead of learning one specific architechture, his goal for us is to be able to transfer knowledge between separate ISA's and gain a big picture idea to ask the right questions when it comes to different architechture.

- Assembly Tools - make, gcc

2 The Modern Model of Computation

we have Alan Turing to thank for this :)



Definition 2.1. An **algorithm** is a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.

2.1 Turing Machine & the Turing Thesis

"A Turing machine is a mathematical model of computation describing an abstract machine that manipulates symbols on a strip of tape according to a table of rules. Despite the model's simplicity, it is capable of implementing any computer algorithm."

> **Remark 2.2.** Dr. Y compares the 1970s Cray 1 supercomputer of its time to the first generation iPhone and makes a point about the evolution of computing although the basic capabilities of computation have remained the same.

3 What is Data?

3.1 Data Representation

- Text
- Numbers
- Images
- Audio
- Video

> **Remark 3.1.** Dr. Y asks the class what we think "data" is, the above are the responses that the students give. He explains that for the most part, computers understand of "data" as text and numbers.

3.2 Computer Binary

How do computers understand binary? Based on voltages. For the mspm0 machine,

$2.31 - 3.6$ V : 1

$0 - 0.99$ V : 0

Conjecture 3.2. For the most park, computer logic of binary is performed at a maximum of 3.3 V

3.3 Text Characters

ASCII

dec	oct	hex	ch	dec	oct	hex	ch	dec	oct	hex	ch	dec	oct	hex	ch
0	0	00	**NUL** (null)	32	40	20	(space)	64	100	40	**@**	96	140	60	**`**
1	1	01	**SOH** (start of header)	33	41	21	**!**	65	101	41	**A**	97	141	61	**a**
2	2	02	**STX** (start of text)	34	42	22	**"**	66	102	42	**B**	98	142	62	**b**
3	3	03	**ETX** (end of text)	35	43	23	**#**	67	103	43	**C**	99	143	63	**c**
4	4	04	**EOT** (end of transmission)	36	44	24	**$**	68	104	44	**D**	100	144	64	**d**
5	5	05	**ENQ** (enquiry)	37	45	25	**%**	69	105	45	**E**	101	145	65	**e**
6	6	06	**ACK** (acknowledge)	38	46	26	**&**	70	106	46	**F**	102	146	66	**f**
7	7	07	**BEL** (bell)	39	47	27	**'**	71	107	47	**G**	103	147	67	**g**
8	10	08	**BS** (backspace)	40	50	28	**(**	72	110	48	**H**	104	150	68	**h**
9	11	09	**HT** (horizontal tab)	41	51	29	**)**	73	111	49	**I**	105	151	69	**i**
10	12	0a	**LF** (line feed - new line)	42	52	2a	*****	74	112	4a	**J**	106	152	6a	**j**
11	13	0b	**VT** (vertical tab)	43	53	2b	**+**	75	113	4b	**K**	107	153	6b	**k**
12	14	0c	**FF** (form feed - new page)	44	54	2c	**,**	76	114	4c	**L**	108	154	6c	**l**
13	15	0d	**CR** (carriage return)	45	55	2d	**-**	77	115	4d	**M**	109	155	6d	**m**
14	16	0e	**SO** (shift out)	46	56	2e	**.**	78	116	4e	**N**	110	156	6e	**n**
15	17	0f	**SI** (shift in)	47	57	2f	**/**	79	117	4f	**O**	111	157	6f	**o**
16	20	10	**DLE** (data link escape)	48	60	30	**0**	80	120	50	**P**	112	160	70	**p**
17	21	11	**DC1** (device control 1)	49	61	31	**1**	81	121	51	**Q**	113	161	71	**q**
18	22	12	**DC2** (device control 2)	50	62	32	**2**	82	122	52	**R**	114	162	72	**r**
19	23	13	**DC3** (device control 3)	51	63	33	**3**	83	123	53	**S**	115	163	73	**s**
20	24	14	**DC4** (device control 4)	52	64	34	**4**	84	124	54	**T**	116	164	74	**t**
21	25	15	**NAK** (negative acknowledge)	53	65	35	**5**	85	125	55	**U**	117	165	75	**u**
22	26	16	**SYN** (synchronous idle)	54	66	36	**6**	86	126	56	**V**	118	166	76	**v**
23	27	17	**ETB** (end of transmission block)	55	67	37	**7**	87	127	57	**W**	119	167	77	**w**
24	30	18	**CAN** (cancel)	56	70	38	**8**	88	130	58	**X**	120	170	78	**x**
25	31	19	**EM** (end of medium)	57	71	39	**9**	89	131	59	**Y**	121	171	79	**y**
26	32	1a	**SUB** (substitute)	58	72	3a	**:**	90	132	5a	**Z**	122	172	7a	**z**
27	33	1b	**ESC** (escape)	59	73	3b	**;**	91	133	5b	**[**	123	173	7b	**{**
28	34	1c	**FS** (file separator)	60	74	3c	**<**	92	134	5c	****	124	174	7c	**|**
29	35	1d	**GS** (group separator)	61	75	3d	**=**	93	135	5d	**]**	125	175	7d	**}**
30	36	1e	**RS** (record separator)	62	76	3e	**>**	94	136	5e	**^**	126	176	7e	**~**
31	37	1f	**US** (unit separator)	63	77	3f	**?**	95	137	5f	**_**	127	177	7f	**DEL** (delete)

4 Numbers

- Integers
  - Unsigned (only positive)
  - Signed (both positive and negative)
- Real
- Floating-point

4.1 Positional Number System

Example 4.1

$$(4705)_{\text{base10}}$$

$$4 \times 1000 + 7 \times 100 + 0 \times 10 + 5 \times 1 = 4705$$

Example 4.2

$$(10011)_{\text{base2}}$$

$$1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 19$$

Example 4.3

$$(23)_{\text{base8}}$$

$$2 \times 8^1 + 3 \times 8^0 = 19$$

Example 4.4

$$(88)_{\text{base8}}$$

Remark 4.5. Notice that the coefficients of given base n can only go up to $n - 1$

base-10	base-2
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Remark 4.6. This is probably the most important point that Dr. Y reinforces in the class: "Computers don't know anything." What he means when he says this is that computers only see binary in the form of n-bit packages. If you tell it that this package is an integer, it will process it as an integer. If you tell it that this package is a character, it will process it as a character.

What about signed numbers? Numbers that can be both positive and negative?

4.2 Signed Magnitude

Same as binary except you stick a signed bit 0 for positive and 1 for negative in front of the binary.

4.3 1s Complement

1. Convert magntiude ($|xyz|$) to binary in $n-1$ bits given that xyz is a signed number $(xyz)_{10}$
2. Append a 0 to make room for sign bit
3. If xyz is positive $\Rightarrow$ done.
   If xyz is negative $\Rightarrow$ flip all bits.

Example 4.7

$(45)_{10}$ where $n = 7$ bits.

Solution. □

1. Convert base-10 to binary
   $(45)_{10} = (101101)_2$
2. Append a 0.
   $\boxed{(0101101)_{\text{1s comp}}}$

Example 4.8

$(-25)_{10}$ where $n = 7$ bits.

Solution. □

1. Convert base-10 to binary
   $(-25)_{10} = (011001)_2$
2. Append a 0.
   (0011001)
3. Flip bits
   $\boxed{(1100110)_{\text{1s comp}}}$

Remark 4.9. The issue with the 1s complement system is that there is both a 0 and a -0 represented as $(0000)_{\text{1s comp}}$ and $(1111)_{\text{1s comp}}$ respectively.

4.4 2s Complement

1. Convert magntiude ($|xyz|$) to binary in $n-1$ bits given that xyz is a signed number $(xyz)_{10}$
2. Append a 0 to make room for sign bit
3. If xyz is positive $\Rightarrow$ done.
   If xyz is negative $\Rightarrow$ flip all bits and add 1.

> **Remark 4.10.** Dr. Y says here that 2s complement is a Positional Number System.

Example 4.11

$(-25)_{10}$ where $n = 7$ bits.

Solution. □

$$\begin{aligned}(-25)_{10} &= (011001)_2 \\ &= (1100110)_{\text{1s comp}} \\ &= (1100110)_{\text{1s comp}} + (00000001)_{\text{2s comp}} \\ &= \boxed{(1100111)_{\text{2s comp}}}\end{aligned}$$

4.5 Floating Point

$(125.32)_{10} = 1 \times 10^2 + 2 \times 10^1 + 5 \times 10^0 + 3 \times 10^{-1} + 2 \times 10^{-2}$
$(1.011)_2 = 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2_{-2} + 1 \times 2_{-3}$

Scientific notation tells us where the period (.) is.

$$\boxed{-}1.\boxed{101011} \times 2^{\boxed{3}}$$

1. Convert iii to binary by division with 2
2. Convert fff to binary by multiplication with 2
3. Write in scientific notation $1.xxxx \times 2^n$
4. Write in fp standard

Example 4.12

Convert $(6.875)_{10}$ to floating point with $n = 12, e = 4$

Solution. □

1. $(6)_{10} \Rightarrow (110)_2$
2. $(.875)_{10} \Rightarrow (.111)_2$
   $(110.111)_2$
3. 1.10111×2^2
   $(0|1001|1011100)$

5 Logic

5.1 Boolean Logic

Booleans only consist of two states: (0, 1), (yes, no), (T, F), etc. What operations can we perform on booleans?

X	NOT(X)
0	1
1	0

X	Y	AND(X, Y)
0	0	0
0	1	0
1	0	0
1	1	1

X	Y	OR(X, Y)
0	0	0
0	1	1
1	0	1
1	1	1

X	Y	XOR(X, Y)
0	0	0
0	1	1
1	0	1
1	1	0

Example 5.1

NOT(001101)

Solution. 110010 □

Example 5.2

Suppose you are a car security system and the following table represents the state of the doors (D), hatch (H), and windows (W)

LD	RD	H	LW	RW
1	1	0	0	1

Lets say you are tasked with determining the state of just the right door (RD). Given the current state how would you isolate the bit of RD?

Solution. 11001 AND 01000 □