Recap

Get the reference materials on 9S12 instructions
TExaS simulates hardware and software

Overview

How numbers are stored on the computer
Precision, basis of numbers
Unsigned and signed numbers
Binary, hexadecimal, decimal
TExaS ViewBox and Help system

2. Information.

Precision is the number of distinct or different values.

The ranges are given for unsigned decimal numbers.

Binary bits	Bytes	Alternatives
8	1	256
10	2	1024
12	2	4096
16	2	65536
20	3	1,048,576
24	3	16,777,216
30	4	1,073,741,824
32	4	4,294,967,296
n	[[n/8]]	2^n

Table 3.1. Relationship between bits, bytes and alternatives as units of precision.

EE319K Lecture 3 in class worksheet
Question 1. How many alternatives is 12 bits?
Question 2. Range is -999 to 999. How many bits are needed?

Decimal digits are used to specify precision of multimeters
0,1,2,3,4,5,6,7,8,9 is a full decimal digit (10 choices)
0,1 is a half decimal digit (2 choices)
0,1,2,3 is three quarters decimal digit (4 choices)
+ or - is a half decimal digit (2 choices)
+ or - 0,1 is three quarters decimal digit (4 choices)

Table 3.2. represents THE DEFINITION of decimal digits. The specification of decimal digits goes 4, 4½, 4¾, 5, with no other possibilities in between. The numbers 4.3 and 4⅛ are not valid representations of decimal digits.

	3½ decimal digits	3¾ decimal digits	4½ decimal digits
	000.0 kΩ	000.0 kΩ	000.00 kΩ
	000.1 kΩ	000.1 kΩ	000.01 kΩ
	. . .	. . .	. . .
	199.9 kΩ	399.9 kΩ	199.99 kΩ
resolution	*0.1kΩ*	*0.1kΩ*	*0.01kΩ*
precision	*2000*	*4000*	*20000 alternatives*
range	*0 to 199.9kΩ*	*0 to 399.9kΩ*	*0 to 199.9kΩ*

decimal digits	exact range	exact alternatives
3	0 to 999	1,000
3½	0 to 1999	2,000
3¾	0 to 3999	4,000
4	0 to 9999	10,000
4½	0 to 19,999	20,000
4¾	0 to 39,999	40,000
5	0 to 99,999	100,000
5½	0 to 199,999	200,000
5¾	0 to 399,999	400,000
N	0 to 10^N-1	10^N
N½	0 to $2*10^N$-1	$2*10^N$
N¾	0 to $4*10^N$-1	$4*10^N$

Table 3.2. Standard definition of decimal digits.

Question 3. How many alternatives is 2¾ decimal digits?

Question 4. Range is -999 to 999. How many decimal digits are needed?

Hexadecimal representation

base 16

convenient to represent binary information

environment	binary	hex	decimal
Freescale	%01111010	$7A	122
Intel and TI	01111010B	7AH	122
C language		0x7A	122
LC-3		x7A	122

Table 2.1. Comparison of various formats.

Easy to convert from binary to hexadecimal:

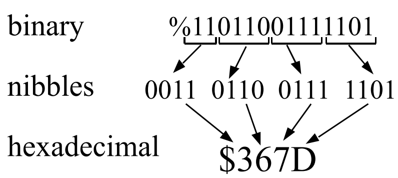

Figure 2.1. Example conversion binary to hex

to convert from hexadecimal to binary we can:

1) convert each hexadecimal digit
into its corresponding 4-bit binary nibble,
2) combine the nibbles into a single binary number.

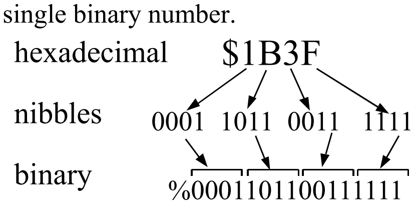

Figure 2.2. Example conversion from hex to binary.

3.3. 8-bit unsigned numbers

b7	b6	b5	b4	b3	b2	b1	b0

Figure 3.3. 8-bit binary format.

the value of the number is
$N = 128 \cdot b_7 + 64 \cdot b_6 + 32 \cdot b_5 + 16 \cdot b_4 + 8 \cdot b_3 + 4 \cdot b_2 + 2 \cdot b_1 + b_0$
Notice that the significance of bit n is 2^n.
There are 256 different unsigned 8-bit numbers.

binary	hex	Calculation	decimal
%00000000	$00		0
%00100100	$24	32+4	36
%01000101	$45	64+4+1	69
%11111111	$FF	128+64+32+16+8+4+2+1	255

Table 3.4. Examples

The *basis* of a number system
a subset from which linear combinations of the basis elements can be used to construct the entire set.
{1, 2, 4, 8, 16, 32, 64, 128}

The values of a binary number system can only be 0 or 1. For example, 69 is (0,1,0,0,0,1,0,1)•(128,64,32,16,8,4,2,1)

What is the 8-bit unsigned binary for 175?

Number	Basis	Need it?	bit	Operation
175	128	yes	bit 7=1	subtract 175-128
47	64	no	bit 6=0	none
47	32	yes	bit 5=1	subtract 47-32
15	16	no	bit 4=0	none
15	8	yes	bit 3=1	subtract 15-8
7	4	yes	bit 2=1	subtract 7-4
3	2	yes	bit 1=1	subtract 3-2
1	1	yes	bit 0=1	subtract 1-1

Table 3.5. Example conversion.

1010,1111 is $AF

Question 5. What are the 8-bit unsigned binary and hex representations for decimal 50?

8-bit signed numbers

b7	b6	b5	b4	b3	b2	b1	b0

Figure 3.3. two's complement number system

$N = -128 \cdot b_7 + 64 \cdot b_6 + 32 \cdot b_5 + 16 \cdot b_4 + 8 \cdot b_3 + 4 \cdot b_2 + 2 \cdot b_1 + b_0$

-128 10000000 +127=01111111

There are 256 different signed 8-bit numbers.

binary	hex	Calculation	dec
%00000000	$00		0
%01000010	$41	64+2	66
%11000110	$C6	-128+64+4+2	-58
%11111111	$FF	-128+64+32+16+8+4+2+1	-1

Table 3.6. Example conversions

For the signed 8-bit number system the basis is

{1, 2, 4, 8, 16, 32, 64, -128}

Observation: *The most significant bit in a two's complement signed number will specify the sign.*

%11111111 could represent either 255 or -1.

You keep track of the number format.

The computer can not determine if signed or unsigned.

signed or unsigned by the assembly instructions you select

e.g., **`lsra`** versus **`asra`**

What is the 8-bit unsigned binary for -90?

Number	Basis	Need it	bit	Operation
-90	-128	yes	bit 7=1	subtract -90 - -128
38	64	no	bit 6=0	none
38	32	yes	bit 5=1	subtract 38-32
6	16	no	bit 4=0	none
6	8	no	bit 3=0	none
4	4	yes	bit 2=1	subtract 6-4
2	2	yes	bit 1=1	subtract 2-2
0	1	no	bit 0=0	none

Table 3.7. Example conversion

1010,0110 is $A6

Observation: *To take the negative of a two's complement signed number we first complement (flip) all the bits, then add 1.*

A second way to convert negative numbers into binary is to first convert them into unsigned binary, then do a two's complement negate.

A third way to convert negative numbers into binary is to first add 256 to the number, then convert the unsigned result to binary using the unsigned method.

Common Error: *An error will occur if you use signed operations on unsigned numbers, or use unsigned operations on signed numbers.*

Maintenance Tip: *To improve the clarity of our software, always specify the format of your data (signed versus unsigned) when defining or accessing the data.*

3.10. Character information

American Standard Code for Information Interchange (ASCII) code.

<table>
<tr><td></td><td></td><td colspan="8">BITS 4 to 6</td></tr>
<tr><td></td><td></td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td></td><td>0</td><td>NUL</td><td>DLE</td><td>SP</td><td>0</td><td>@</td><td>P</td><td>`</td><td>p</td></tr>
<tr><td>B</td><td>1</td><td>SOH</td><td>DC1</td><td>!</td><td>1</td><td>A</td><td>Q</td><td>a</td><td>q</td></tr>
<tr><td>I</td><td>2</td><td>STX</td><td>DC2</td><td>"</td><td>2</td><td>B</td><td>R</td><td>b</td><td>r</td></tr>
<tr><td>T</td><td>3</td><td>ETX</td><td>DC3</td><td>#</td><td>3</td><td>C</td><td>S</td><td>c</td><td>s</td></tr>
<tr><td>S</td><td>4</td><td>EOT</td><td>DC4</td><td>$</td><td>4</td><td>D</td><td>T</td><td>d</td><td>t</td></tr>
<tr><td></td><td>5</td><td>ENQ</td><td>NAK</td><td>%</td><td>5</td><td>E</td><td>U</td><td>e</td><td>u</td></tr>
<tr><td>0</td><td>6</td><td>ACK</td><td>SYN</td><td>&</td><td>6</td><td>F</td><td>V</td><td>f</td><td>v</td></tr>
<tr><td></td><td>7</td><td>BEL</td><td>ETB</td><td>'</td><td>7</td><td>G</td><td>W</td><td>g</td><td>w</td></tr>
<tr><td>T</td><td>8</td><td>BS</td><td>CAN</td><td>(</td><td>8</td><td>H</td><td>X</td><td>h</td><td>x</td></tr>
<tr><td>O</td><td>9</td><td>HT</td><td>EM</td><td>)</td><td>9</td><td>I</td><td>Y</td><td>i</td><td>y</td></tr>
<tr><td></td><td>A</td><td>LF</td><td>SUB</td><td>*</td><td>:</td><td>J</td><td>Z</td><td>j</td><td>z</td></tr>
<tr><td>3</td><td>B</td><td>VT</td><td>ESC</td><td>+</td><td>;</td><td>K</td><td>[</td><td>k</td><td>{</td></tr>
<tr><td></td><td>C</td><td>FF</td><td>FS</td><td>,</td><td><</td><td>L</td><td>\</td><td>l</td><td>|</td></tr>
<tr><td></td><td>D</td><td>CR</td><td>GS</td><td>-</td><td>=</td><td>M</td><td>]</td><td>m</td><td>}</td></tr>
<tr><td></td><td>E</td><td>SO</td><td>RS</td><td>.</td><td>></td><td>N</td><td>^</td><td>n</td><td>~</td></tr>
<tr><td></td><td>F</td><td>S1</td><td>US</td><td>/</td><td>?</td><td>O</td><td>_</td><td>o</td><td>DEL</td></tr>
</table>

Table 3.21. Standard 7-bit ASCII.

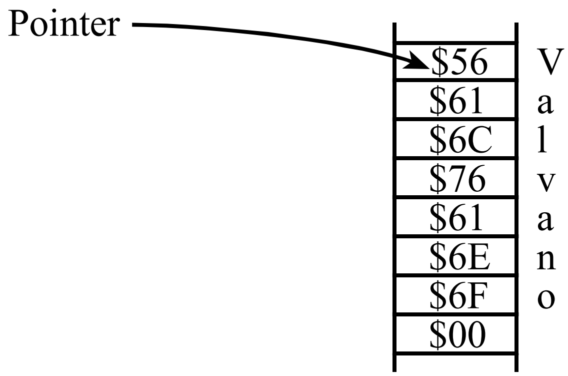

Figure 3.33. Strings are stored as a sequence of ASCII characters, followed by a null.
Unicode Standard, see http://www.unicode.org/.

a unique number for every character,
no matter what the platform,
no matter what the program,
no matter what the language.

3.4. 16-bit unsigned numbers

A *word* or *double byte* contains 16 bits

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0

Figure 3.4. 16-bit binary format.

For the unsigned 16-bit number system the basis is
{1, 2, 4, 8, 16, 32, 64, 128,
256, 512, 1024, 2048, 4096, 8192, 16384, 32768}
For the signed 16-bit number system the basis is
{1, 2, 4, 8, 16, 32, 64, 128,
256, 512, 1024, 2048, 4096, 8192, 16384, -32768}

Common Error: *An error will occur if you use 16-bit operations on 8-bit numbers, or use 8-bit operations on 16-bit numbers.*

Maintenance Tip: *To improve the clarity of your software, always specify the precision of your data when defining or accessing the data.*

4.1.1. Big and little endian

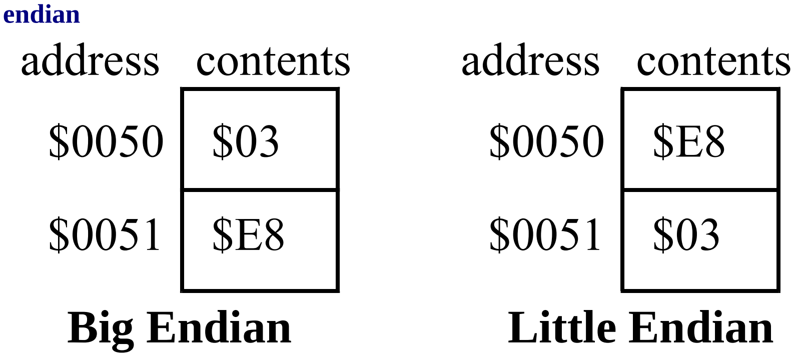

Figure 4.1. Example of big and little endian formats

Big Endian

address	contents
$0050	$12
$0051	$34
$0052	$56
$0053	$78

Little Endian

address	contents
$0050	$78
$0051	$56
$0052	$34
$0053	$12

Figure 4.1. Example of big and little endian formats

Download at
http://users.ece.utexas.edu/~valvano/Starterfiles/Square_DG128asm.zip

Run the Square example, reduce period
show the Windows
show Help system

The bottom line

Everything is binary inside the computer
Programmer must keep track of format
Precision, decimal digits, basis, ASCII
Unsigned, signed (2's complement)
Big and little endian