

Digital electronics recap

prepared for [Programmable Logic Lessons / 1.1](#) by ~skmp

Kindly hosted by hackerspace.gr

Binary system

Simplest counting system

Only two symbols:

0, 1

(Also known as
False, True
bit)

Bool algebra

- Logic using the binary system
- Basic operations
 - And
 - If x and y. Common symbols &, &&, *
 - Or
 - X or Y. Common symbols |, ||, + (in bool algebra)
 - Xor
 - X or Y, but not X and Y. Common symbols: ^ (C/C++), \oplus
 - Not
 - Inverse. Common symbols: ~, -

Truth tables

(An easy easy to remember)

x	y	X and Y	X Or y	X Xor y	Not X
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Doing useful work !

- Numbers with many digits
 - 0110101
 - Each digit represents an increasing power of 2 (RtL)
 - $1 * 2^0 + 0 * 2^1 + 1 * 2^2 + 0 * 2^3 + 1 * 2^4 + 1 * 2^5 + 0 * 2^6$
 - $1*1 + 0*2 + 1*4 + 0*8 + 1*16 + 1*32 + 0*64$
 - = 53 (base 10)
 - base 16 (Hex) is also handy for large numbers
 - 0-9, A, B, C, D, E, F
 - 0xF0E3
 - $3*16^0 + E*16^1 + 0*16^2 + F*16^3$
 - $3 + 14*16 + 0*256 + 15*4096$
 - = 61667 (base 10)
 - Base 16 ↔ 2 conversions are easy ([Google it up](#))

Doing useful work !

- Representing Negative numbers
 - 2's complement
 - $-x$ is defined as $\sim x + 1$
 - For 8 bits, -128 to 127
 - $-01010 \rightarrow 10110$
 - 1's complement (not used anymore)
 - $-x$ is defined as $\sim x$
 - Sign bit
 - An extra bit to store positive or negative
 - For 8 bits, -127 to 127, with +0 and -0
 - 7 bits range and sign bit

Doing useful work !

- Bit-wise operators
 - And, Or, Xor, Not
 - Each bit is processed in “parallel”
 - $z = x \text{ and } y \rightarrow z[0] = x[0] \text{ and } y[0], z[1] = x[1] \text{ and } y[1], \text{ and so on}$
- Math primitives
 - Addition, Subtraction, Multiplication (sometimes)
 - [http://en.wikipedia.org/wiki/Adder_\(electronics\)](http://en.wikipedia.org/wiki/Adder_(electronics))
 - http://en.wikipedia.org/wiki/Binary_multiplier
- Memory & Routing
 - Flip flop (Memory)
 - [http://en.wikipedia.org/wiki/Flip-flop_\(electronics\)](http://en.wikipedia.org/wiki/Flip-flop_(electronics))
 - Mux/Demux (Selection/Routing)
 - <http://en.wikipedia.org/wiki/Multiplexer>

Flip flops

- memory elements
 - S/R (Set/Reset)
 - Basic building block
 - Other Types
 - J/K is like S/R, but clocked
 - D (Data)
 - T (Toggle)

Mux/Demux

- Use an input signal to select from two (or more) inputs
 - Basic logic that allows if/conditions/selection
- Demux is the inverse operation

Combinational logic

- Output only depends on input
 - Typical example is an Adder
- Summarized in truth tables
 - http://en.wikipedia.org/wiki/Truth_table
- http://en.wikipedia.org/wiki/Combinational_logic

Sequential Logic

- Output depends on past values
 - The system has “memory”
 - Typical example is a counter
 - The output depends on the internal state
- Is usually synchronized via a global clock
- Typically summarized in Transition/Excitation Tables
 - http://en.wikipedia.org/wiki/State_transition_table
 - http://en.wikipedia.org/wiki/Excitation_table
- http://en.wikipedia.org/wiki/Sequential_logic

Other useful links

(for wikipedia addicts)

- http://en.wikipedia.org/wiki/Theory_of_computation
 - http://en.wikipedia.org/wiki/Automata_theory
 - http://en.wikipedia.org/wiki/Finite_state_machine
- http://en.wikipedia.org/wiki/Asynchronous_circuit
- http://en.wikipedia.org/wiki/Synchronous_system
- ... :)

Thanks !

Next week we'll do some hands-on
“experiments” on an actual fpga !

Feel free to drop by #hsgr @ freenode

... or the [hsgr mailing list](#)

... and use the [wiki](#) !

(or, send direct feedback – skmp@emudev.org)