Study Topics

Historical Perspective

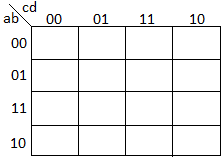
* Characteristics of computer generations 0-4
  + Gen0 = Mechanical Calculating Machines
  + Gen1 = Vacuum Tube Computers
  + Gen2 = Transistorized Computers
  + Gen3 = Integrated Circuit Computers
  + Gen4 = VLSI Computers
* Moore's Law
  + "The density of transistors in an integrated circuit will double every year."

Computer Organization and Architecture Chapter 3 ---------------------------------------------------

Boolean Algebra

* Behavior of basic identities (Identity, Null, Idempotent, Inverse, Commutative, Associative, Distributive).

|  |  |  |
| --- | --- | --- |
| Identity Name | AND Form | OR Form |
| Identity Law | 1x = x | 0 + x = x |
| Null Law | 0x = x | 1 + x = 1 |
| Idempotent Law | xx = x | x + x = x |
| Inverse Law | x!x = 0 | x + !x = 1 |
| Commutative Law | xy = yx | x + y = y + x |
| Associative Law | (xy)z = x(yz) | (x + y) + z = x + (y + z) |
| Distributive Law | x + yz = (x + y) (x + z) | x(y + z) = xy + xz |
| Absorption Law | x(x + y) = x | x + xy = x |
| DeMorgan’s Law | !(xy) = !x + !y | !(x + y) = !x!y |
| Double Complement Law | !(!x) = x | |

* Truth tables.
* Karnaugh Maps of 2, 3, and 4 variables.

* Sum of Products / Minterms.

Circuits and Gates

* Gate symbols and behavior.
* AND, OR, NOT, XOR Gates.
* Universal NAND and NOR gates.
* Logic diagrams.
* Block diagrams.

Combinational Circuits

* Multiplexers and Decoders
  + Decoder 2-to-4 decoder makes every combination of x input, y input
* Full adder
  + Cout(X, Y, Cin) = XY + XCin + YCin
  + Sum(X, Y, Cin) = (X!Y!Cin) + (!XY!Cin) + (!X!YCin) + (XYCin)
* Designing circuits with multiple output bits

Sequential Circuits

* Characteristics of a sequential circuit.
  + Sequential circuits compute their output based on input and state, the state being changed based on a clock.
* S-R Flip Flops as the basis of other flip flops
* S-R, J-K, and D flip flop behavior
* Counters and RAM circuits.

Bitwise Operators in C/C++/Java

* 01001000 &
* 10111000 = AND operator
* --------
* 00001000
* 72 & 184 = 8
* 01001000 |
* 10111000 = OR operator
* --------
* 11111000
* 72 | 184 = 248
* ^ XOR (exclusive-or)
* ~ complement
* << The leftshift operator is the equivalent of moving all the bits of a number a specified number of places to the left.
* Equivalent to multiplication by 2^shift\_arg.

int mult\_by\_pow\_2(int number, int power)

{

return number<<power;

}

* >> division by 2
* Setting, clearing, toggling arbitrary bits
* Packing and extracting groups of bits
* Testing individual bits

Basic C/C++ programming

* Main function
  + int \_tmain(int argc, TCHAR\* argv[])
* Basic I/O
* Conditionals/Loops
  + #include <iostream>
  + using namespace std;
  + int main()
  + {
    - for (x=0; x<10;x++){
      * cout << x << endl;
    - }

Cin.get();

* + }