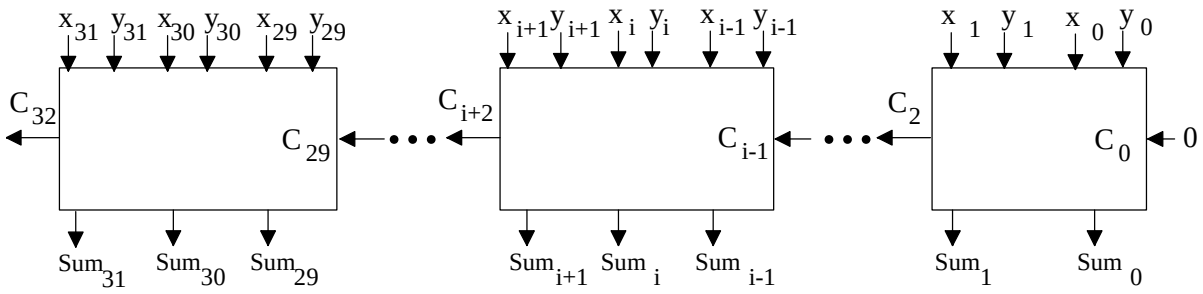
Homework 2 Computer Organization

Due: Friday, 2/3/06 (noon)

Exercise: B.34 on page B-83 of the text

Additional Problems:

1) Recall (lecture 4 on 1-19-06) that a 32-bit adder can be build using 32 1-bit full adders, but the carry bit out of the most-significant bit (MSB) will not be correct until 64 gate delays (i.e., 2 gate delays per 1-bit adder). Using 16 2-bit adders that use carry-lookahead, we reduced the gate delays to 32 (i.e., 2 gate delays per 2-bit adder). You job is to consider using 10 3-bit adders and a 2-bit adder to construct a 32-bit adder as:



a) If c_{i+2} is calculated directly from the inputs: x_{i+1}, y_{i+1}, x_i, y_i, x_{i-1}, y_{i-1}, and c_{i-1}, then how many gate delays would be needed to calculate the c_{i+2} signal **for a single** three-bit adder?
Hints:

- Think about what the SOP, sum-of-products, Boolean expression would look like for c_{i+2}, i.e., how many products, how many inputs to the ANDs and OR gates.
- Remember that a single AND or OR gate must have less than 10 inputs. Any gate requiring more than that must be build using more than one gate.

b) What would be the total number of gate delays in the above 32-bit adder before the c_{32} signal is generated correctly?

2) Draw a 4-bit register that is able to perform the following operations:

- parallel read/output of all bits
- parallel write/input of all bits
- circular shift left one bit position (value shifted out of most-significant bit is shifted into the least-significant bit)
- circular shift right one bit position (value shifted out of least-significant bit is shifted into the most-significant bit)
- arithmetic shift right (sign-extend the most-significant bit)

Hint: For each D-flip flop, use the output of a MUX as the D-input. You can draw block-diagrams for flip-flops and MUXs without showing their internal gate implementations.

3) Using the discussion in the book and the register file handout, draw a complete (not just a one-bit slice) register file that has:

- 2 registers
- 3-bits per register
- one write-port
- two read-ports

You can draw block-diagrams for flip-flops, decoders, and MUXs without showing their gate implementations, but you should show all the flip-flops, decoders, MUXs, and connecting wires.

4) How well does this register-file design scale? Suppose that we are implementing a 64 M x 16 (64M registers, each with 16 bits) register file with one write-port and one read-port.
a) How many and what type of decoder(s) would be needed?

b) How many total gates (assume 9-input limit on AND & OR gates) would be needed to implement this (these) decoder(s)?

c) How many and what type of MUX(s) would be needed?

d) How many total gates (assume 9-input limit on AND & OR gates) would be needed to implement this (these) MUX(s)?

e) Assuming D flip-flops to store each bit (5 gates/flip-flop). What % of the total gates is used to implement the D flip-flops?

5) Redo the previous question using the 64 M x 16 square-memory implementation similar to the "Implementation of Large Memory Chips" class handout.