Propagation Delay, Circuit Timing & Adder Design

ECE 152A – Fall 2006

Reading Assignment

- Brown and Vranesic
  - 2 Introduction to Logic Circuits
    - 2.9 Introduction to CAD Tools
      - 2.9.1 Design Entry
      - 2.9.2 Synthesis
      - 2.9.3 Functional Simulation
      - 2.9.4 Summary
  - 3 Implementation Technology
    - 3.3.1 Speed of Logic Circuits
    - 3.5 Standard Chips
      - 3.5.1 7400-Series Standard Chips

Reading Assignment

- Brown and Vranesic (cont)
  - 5 Number Representation and Arithmetic Circuits
    - 5.1 Positional Number Representation
      - 5.1.1 Unsigned Numbers
      - 5.1.2 Conversion Between Decimal and Binary Systems
      - 5.1.3 Octal and Hexadecimal Representations
    - 5.2 Addition of Unsigned Numbers
      - 5.2.1 Decomposed Full-Adder
      - 5.2.2 Ripple-Carry Adder
      - 5.2.3 Design Example

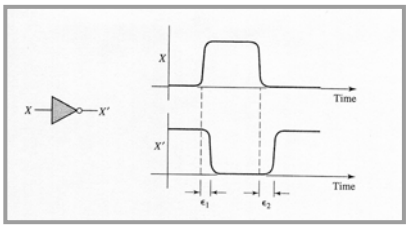
Reading Assignment

- Roth
  - 1 Introduction Number Systems and Conversion
    - 1.2 Number Systems and Conversion
    - 1.3 Binary Arithmetic
  - 8 Combinational Circuit Design and Simulation Using Gates
    - 8.3 Gate Delays and Timing Diagrams

Propagation Delay

- When gate inputs change, outputs don't change instantaneously
  - This delay is known as "gate" or "propagation" delay

$$\varepsilon_1 = t_{PHL}$$
$$\varepsilon_2 = t_{PLH}$$

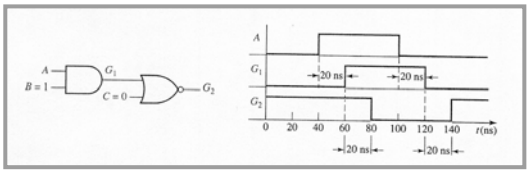


Propagation Delay

- ε_1 is the propagation delay from input going high to output going low (inverting logic)
  - t_{PHL}
- ε_2 is the propagation delay from input going low to output going high (inverting logic)
  - t_{PLH}
- Terminology (t_{PHL} and t_{PLH}) always refers to the transition on the output (whether circuit is inverting or not)

Propagation Delay

- Multiple Gate Delays
  - Example assumes that t_{PLH} and t_{PHL} equal 20 ns for both AND and NOR gate
    - Not always the case for different transitions or different gate types



Propagation Delay

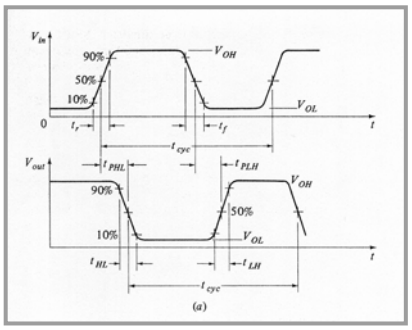
- Maximum propagation delay is the longest delay between an input changing value and the output changing value
- The path that causes this delay is called the *critical path*
  - The critical path imposes a limit on the maximum speed of the circuit
    - Max frequency = f (clk to q + critical path + setup time) … much more on this later

Propagation Delay

- For example circuit, critical path is from any change in the A input resulting in a change in G_2
  - Circuit is inverting (from A to G_2)
    - With B = 1 and C = 0, A↑ causes G_2↓ (t_{PHL} = 20 ns) and A↓ causes G_2↑ (t_{PLH} = 20 ns)
  - Maximum propagation delay
    - 20 ns + 20 ns = 40 ns
      - Same for either A↑ or A↓
      - Not always the case

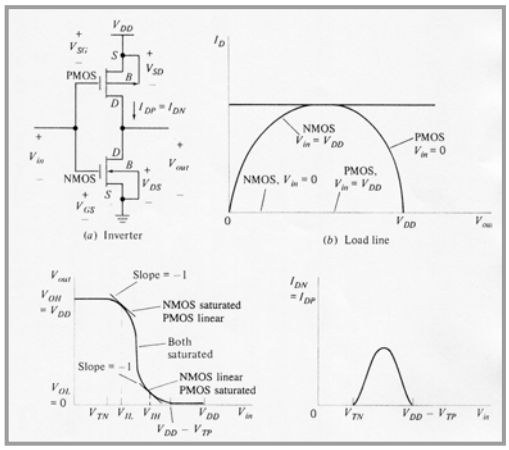
Propagation Delay

- Definitions of transitions and delay times for (inverting) digital circuits



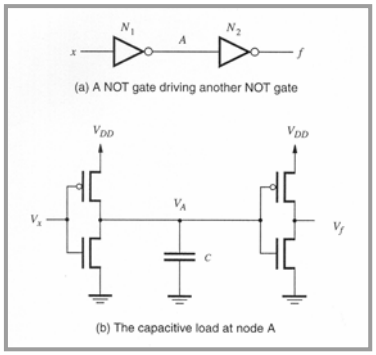
The CMOS Inverter

- Alternate symbol and more details
  - Current flows only when output switching
  - Power is frequency dependent



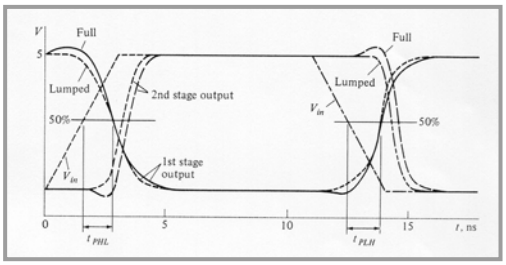
The CMOS Inverter

- Output switching requires charging (or discharging) parasitic and gate capacitance through a resistor(s)
  - Transistor "on resistance"
  - Wire capacitance and resistance
  - Gate capacitance



The CMOS Inverter

- SPICE Simulation of CMOS inverter pair
  - First inverter driven by ideal source
  - Full (distributed) and lumped RC loads



Transistor-Transistor Logic (TTL)

- Bipolar Junction Transistor (BJT) based technology and logic family
- Both input and output stages implemented with transistors (hence, TTL)
  - Earlier logic families used resistors (RTL) or diodes (DTL) in the input stage
- TTL first commercialized in mid 1960's
  - Driven by many issues, not the least of which was the need for an on-board computer for the Lunar Excursion Module (LEM) in NASA's Apollo program

Transistor-Transistor Logic (TTL)

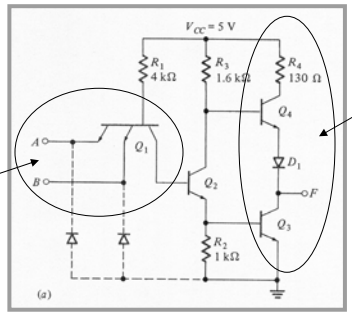
- First "complete" family of digital integrated circuits
  - Small and medium scale integration (SSI and MSI)
    - SSI < 10 gates per device
    - MSI > 10 and < 100 gates per device
    - LSI and VLSI followed
- Commercial and military temperature ranges
  - 74XX – Commercial temperature range
    - 0 – 70° C
  - 54XX – Military temperature range
    - -55 – 125° C

Transistor-Transistor Logic (TTL)

- "Significant" evolution of Texas Instruments' TTL technology
  - Standard TTL (1965) 54/74XX
  - Schottky-Clampled TTL (1970) 54/74SXX
  - Low Power, Schottky-Clamped TTL (1975) 54/74LSXX
  - Advanced, Low Power, Schottky-Clamped TTL (1980) 54/74ALSXX
  - TTL compatible CMOS (1985) 54/74ACTXX
- Compatible TTL families from other vendors
  - Fairchild, Intel, Motorola, National and others

Transistor-Transistor Logic (TTL)

- Standard TTL, 2-input NAND Gate



"*totem-pole*" output stage

multiple emitter input stage

Binary Numbers

- Unsigned and Signed Integers
  - Unsigned integers represent all positive values in the range 0 to 2^n -1
  - Signed integers in several flavors
    - Sign magnitude
    - One's complement
    - Two's complement
- We will be concerned with unsigned binary integers for this discussion of adders

Conversion Between Binary and Decimal

- Binary to Decimal

$$1011.11_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2}$$
$$= 8 + 0 + 2 + 1 + \tfrac{1}{2} + \tfrac{1}{4} = 11\tfrac{3}{4} = 11.75_{10}$$

- Decimal to Binary

Convert 53_{10} to binary.

$2 \overline{)53}$		
$2 \overline{)26}$	rem. $= 1 = a_0$	
$2 \overline{)13}$	rem. $= 0 = a_1$	
$2 \overline{)6}$	rem. $= 1 = a_2$	$53_{10} = 110101_2$
$2 \overline{)3}$	rem. $= 0 = a_3$	
$2 \overline{)1}$	rem. $= 1 = a_4$	
0	rem. $= 1 = a_5$	

Convert $.625_{10}$ to binary.

$F = .625$	$F_1 = .250$	$F_2 = .500$	
$\times \ 2$	$\times \ 2$	$\times \ 2$	$.625_{10} = .101_2$
1.250	0.500	1.000	
$(a_{-1} = 1)$	$(a_{-2} = 0)$	$(a_{-3} = 1)$	

Octal and Hexadecimal Representation

Octal (2^3)

↕

Binary

↕

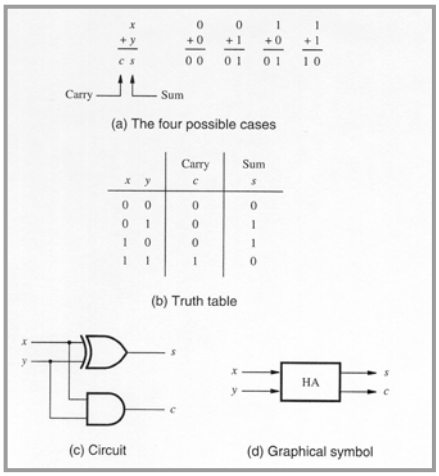
Hexadecimal (2^4)

Table 5.1 Numbers in different systems.

Decimal	Binary	Octal	Hexadecimal
00	00000	00	00
01	00001	01	01
02	00010	02	02
03	00011	03	03
04	00100	04	04
05	00101	05	05
06	00110	06	06
07	00111	07	07
08	01000	10	08
09	01001	11	09
10	01010	12	0A
11	01011	13	0B
12	01100	14	0C
13	01101	15	0D
14	01110	16	0E
15	01111	17	0F
16	10000	20	10
17	10001	21	11
18	10010	22	12

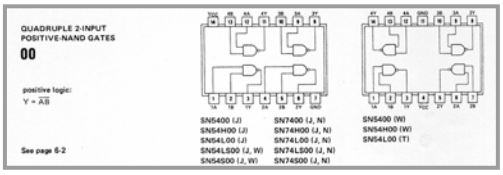
Addition of Unsigned Numbers

- Half Adder
  - 2 input bits
    - x
    - y
  - 2 output bits
    - s (sum)
    - c (carry)



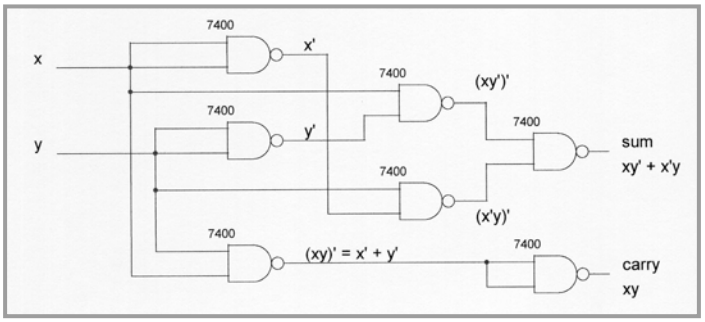
TTL Implementation

- SN7400 : Quad, 2-input, positive NAND gates with totem pole outputs
  - SN indicates Texas Instruments
  - Pin assignments (top view) for dual-in-line package (DIP)



TTL Implementation

- Schematic with SN7400's
  - 2 IC's, 1 spare NAND gate



TTL Implementation

- SN7400
  - Switching characteristics (propagation delays)
    - t_{PLH} (max) = 22 ns
    - t_{PHL} (max) = 15 ns

switching characteristics at V_{CC} = 5 V, T_A = 25°C

TYPE	TEST CONDITIONS#	t_{PLH} (ns) Propagation delay time, low-to-high-level output			t_{PHL} (ns) Propagation delay time, high-to-low-level output		
		MIN	TYP	MAX	MIN	TYP	MAX
'00, '10	C_L = 15 pF, R_L = 400 Ω		11	22		7	15
'04, '20			12	22		8	15
'30			13	22		8	15

TTL Implementation

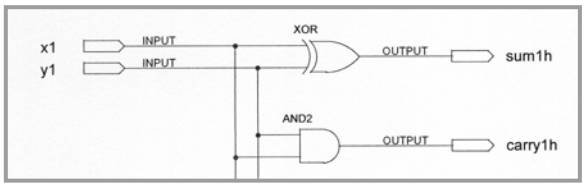
- Worst case propagation delay
  - Critical path is x (or y) to sum
  - Three levels of gate delay and three levels of inversion
    - Two possibilities
      - $t_{PLH} + t_{PHL} + t_{PLH}$
      - $t_{PHL} + t_{PLH} + t_{PHL}$
  - Max delay is $t_{PLH} + t_{PHL} + t_{PLH}$
    - 22 ns + 15 ns + 22 ns = 59 ns
      - Max frequency = 1 / (clk to q + 59 ns + setup time)

Programmable Logic Devices

- A Programmable Logic Device (PLD) is a single, programmable device capable or replacing multiple, discrete TTL chips
  - PLD is comprised of “uncommited” gates and programmable switches to interconnect the gates
  - Simple PLD’s can realize 2 to 10 functions of 4 to 16 input variables
  - Complex PLD’s can implement circuits requiring 100’s of thousands of gates

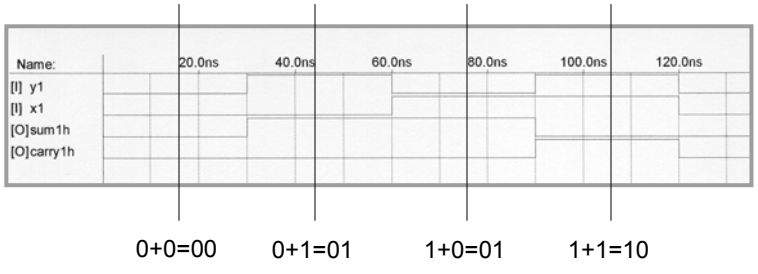
Half Adder Implementation with a Programmable Logic Device (PLD)

- Schematic Capture (Design Entry)
  - Using "Primitive" library of logic elements
    - Specify logic function using generic logic gates rather than selecting physical devices (e.g., 7400 TTL)
    - CAD tool will determine actual implementation



PLD Implementation of Half Adder

- Functional Simulation
  - All propagation delays set to zero



PLD Implementation of Half Adder

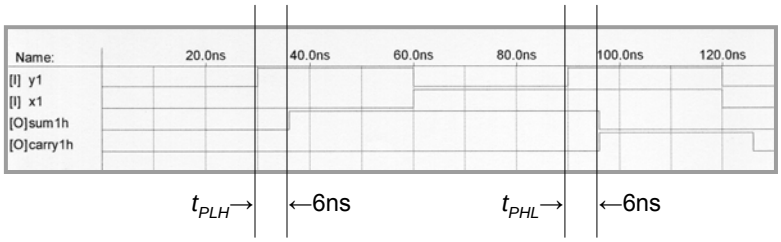
- Map logical design onto a target architecture and physical device using CAD tool
  - *Logical function* is specified via the primitive library and implemented using *logical structures* incorporated into the target architecture
  - The physical device is a single chip hardware implementation of the design incorporating the structures of the target architecture
    - Altera MAX 7000 Complex Programmable Logic Device (CPLD) family for this example

PLD Implementation of Half Adder

- Timing Simulation
  - Must know specific device and package combination in PLD environment
    - Both contribute to performance
  - Simulation of physical implementation of design
    - Logical (gate) delays
    - Physical (interconnect) delays
    - I/O (package input/output) delay

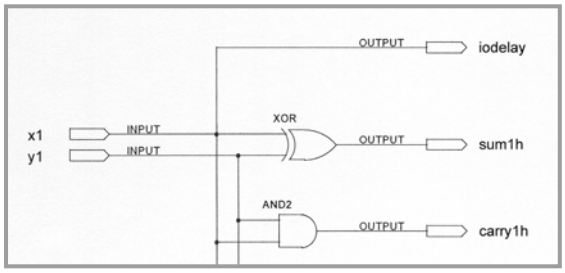
PLD Implementation of Half Adder

- Approximately 6ns delay from input to output
  - t_{PLH} and t_{PHL}



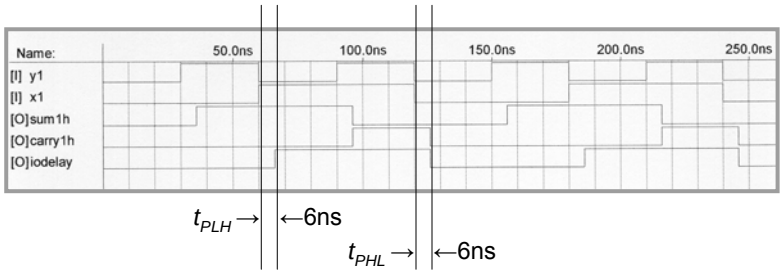
I/O Delays

- Circuit to measure I/O delay
  - X1 to iodelay path through input receiver and output driver
    - Allows I/O delay to be separated from internal (core) delays



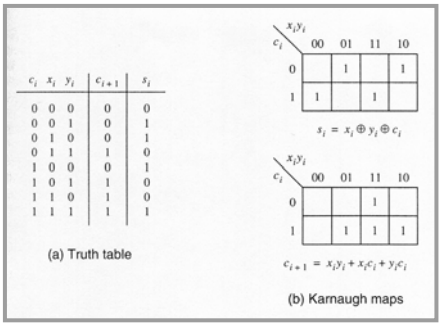
I/O Delays

- Timing Simulation
  - Simulation indicates I/O delay dominates logic circuit delays for this (very small) design



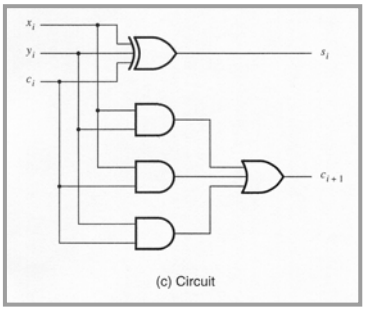
Full Adder

- Full Adder
  - By adding a *carry in* input, multiple-bit numbers can be added by cascading full adder stages
    - The sum and carry out become functions of three variables x, y and cin



Full Adder

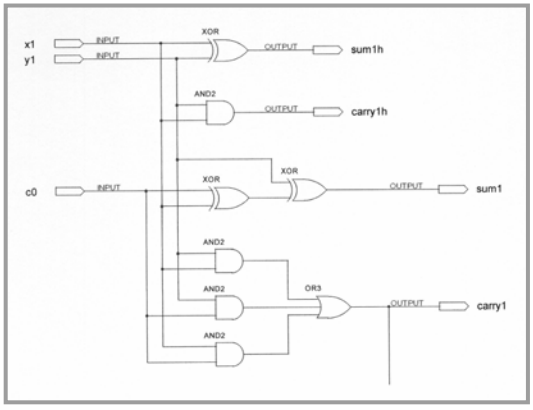
- Generic Circuit Implementation



(c) Circuit

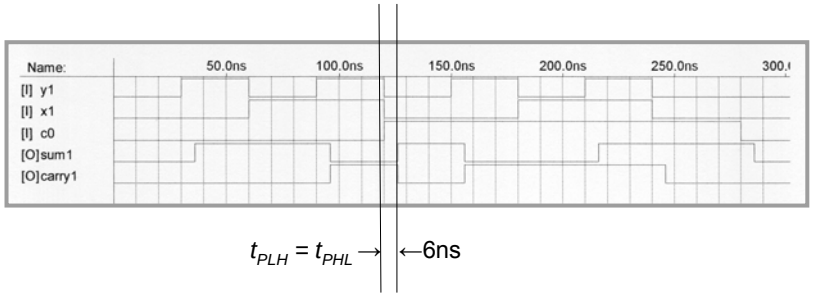
Full Adder Implementation

- Schematic Capture



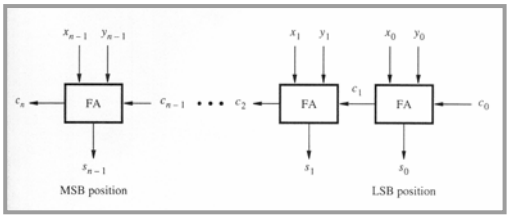
Full Adder Implementation

- Timing Simulation
  - As with the half adder, I/O delays dominate



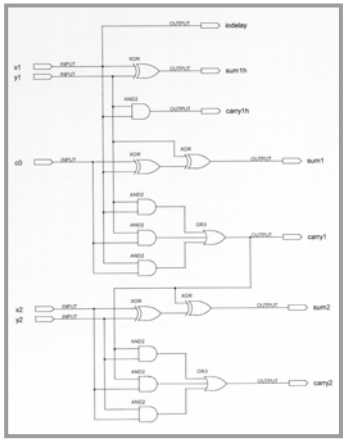
Ripple Carry Adder

- n-bit, Ripple Carry Adder
  - By cascading full adders, carry "ripples" from least to significant bit toward most significant bit
    - Critical path becomes input to full adder 0 to output of full adder n



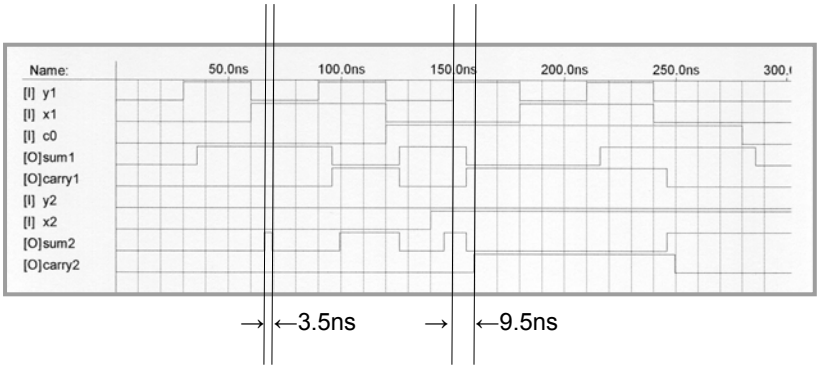
Two-Bit Ripple Carry Adder

- Schematic with I/O test circuit, half-adder, full adder and two-bit ripple carry adder



CPLD Implementation

- Timing Simulation



CPLD Implementation

- Timing Simulation
  - Note propagation delay from y1 to carry2 is measured at 9.5 ns
    - Greater than simulated I/O delay of 6ns
    - Internal delays now visible (and measurable) at device pins
  - Note also 3.5 ns “glitch” at 66ns
    - Resolution of simulation implied to be 3.5ns