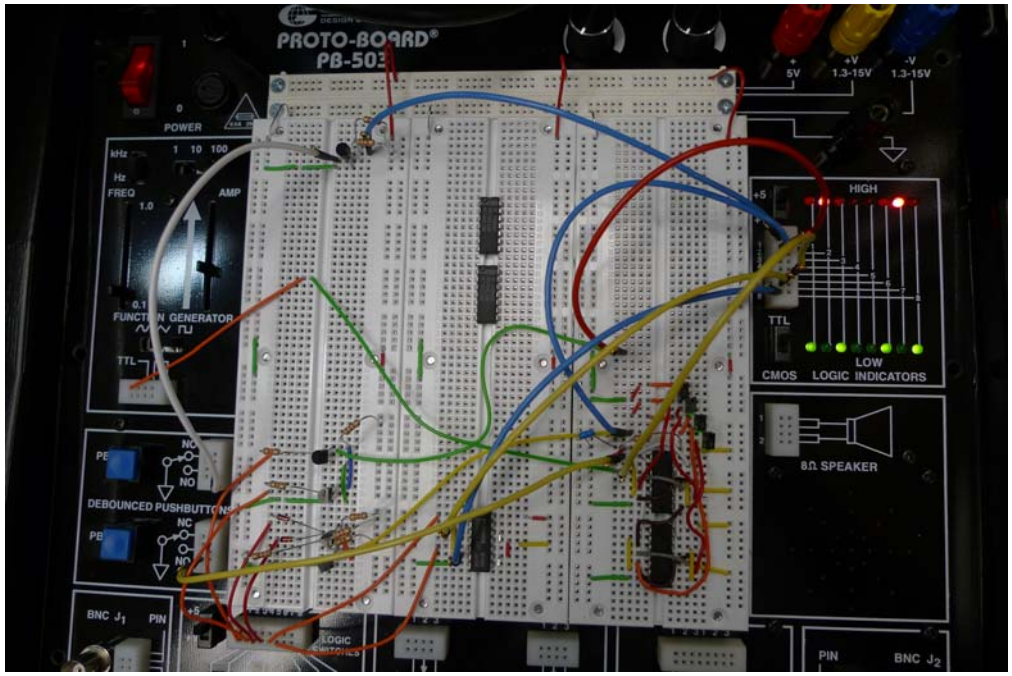
Neutron Monitor Workshop 2(A): Principles of Digital Logic



Mahidol University

June 25, 2009

Paul Evenson
University of Delaware
Bartol Research Institute

Workshop Series Idea

- Introduce students to technical aspects of neutron monitor operation
- Rotating workshop series that will repeat every two years at a two per year rate
- Independent enough so students can join at any point
- Accommodate wide skill range with an emphasis on “hands on” experience and individual discussion

Workshop Series Plan

1. Detector operation
   A. Detector Physics and Analog Electronics
   B. Art and Science of Soldering
2. Digital Circuits
   A. Principles of Digital Logic
   B. Neutron Monitor Digital Electronics
3. Microcontrollers (including data transfer methods within the electronics system)
4. Real time data acquisition
   A. Principles of Telemetry and Data Acquisition
   B. Data Conversion and Manipulation with Visual Basic

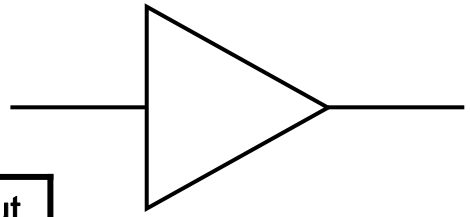
Plan For Today

- Lecture with Demonstrations
  - Questions and answers
- Describe Design Problems
  - Questions and answers
- Individual work on design problems
  - Only one prototype system here
  - Limited number of parts
  - Testing your design will be left for later

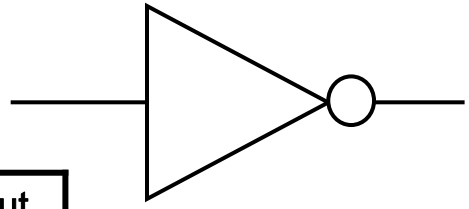
Digital Circuits

- Ultimately composed of the same type of electronic components as analog electronics
- Digital signals are expressed as analog levels
- Typically there are only two logic levels
  - Called “0” and “1”; “Low” and “High”
  - Low is usually near ground
  - High is usually near supply voltage (often +5 volts)
  - (Bad practice to use supply itself for “High”)
- One could define a range of discrete levels but typically this is not done

Simplest Logic Elements



In	Out
0	0
1	1

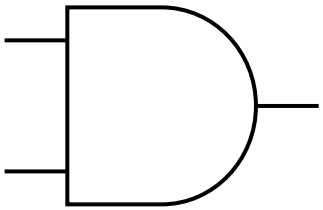


In	Out
0	1
1	0

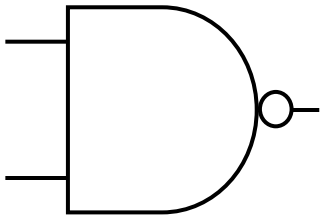
- Buffer: Really doesn't do anything except isolate the input signal and condition the output
- Inverter: The circle on an input or output indicates inversion

AND and NAND

A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1



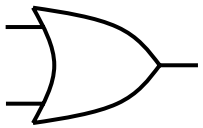
A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0



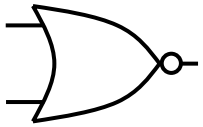
- Note that you can make *any* logic function out of NAND gates

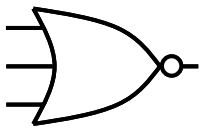
OR and NOR

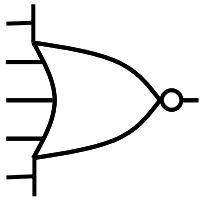
A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1



A	B	Out
0	0	1
0	1	0
1	0	0
1	1	0



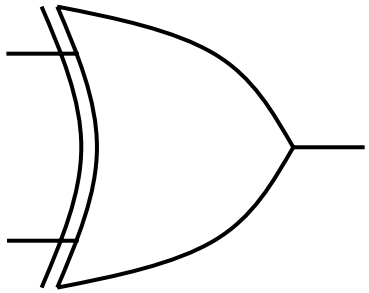




- You can also make *any* logic function out of NOR gates
- OR, NOR, AND, NAND all have natural multi-input extensions

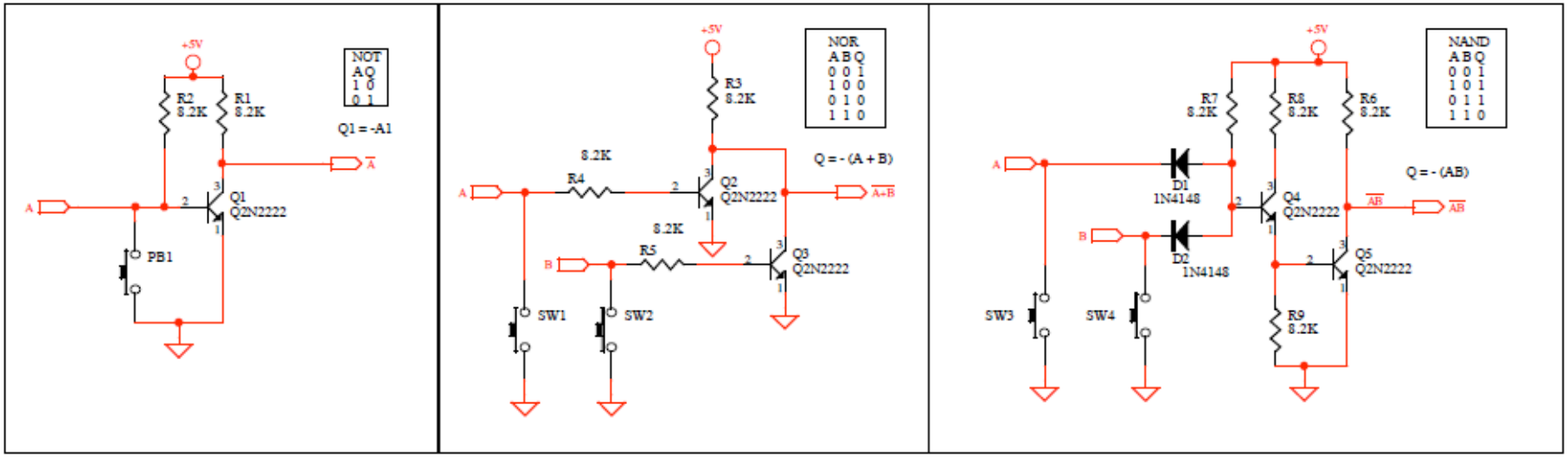
XOR (Exclusive OR)

A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0



- This is the “even parity” operation
- Multi input operation is not well defined
  - One and only one?
  - Odd number?
  - All but one?

Simple Implementations



- Note that these work, but they are poorly buffered on inputs and outputs, and cannot be combined easily into more complex circuits

Device Properties

74HC00; 74HCT00
Quad 2-input NAND gate

PINNING

PIN	SYMBOL	DESCRIPTION
1	1A	data input
2	1B	data input
3	1Y	data output
4	2A	data input
5	2B	data input
6	2Y	data output
7	GND	ground (0 V)
8	3Y	data output
9	3A	data input
10	3B	data input
11	4Y	data output
12	4A	data input
13	4B	data input
14	V_{CC}	supply voltage

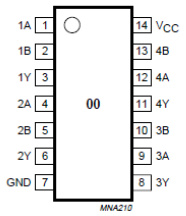


Fig.1 Pin configuration DIP14, SO14 and (T)SSOP14.

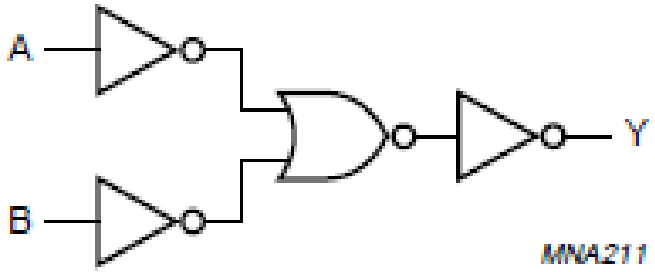


Fig.3 Logic diagram (one gate).

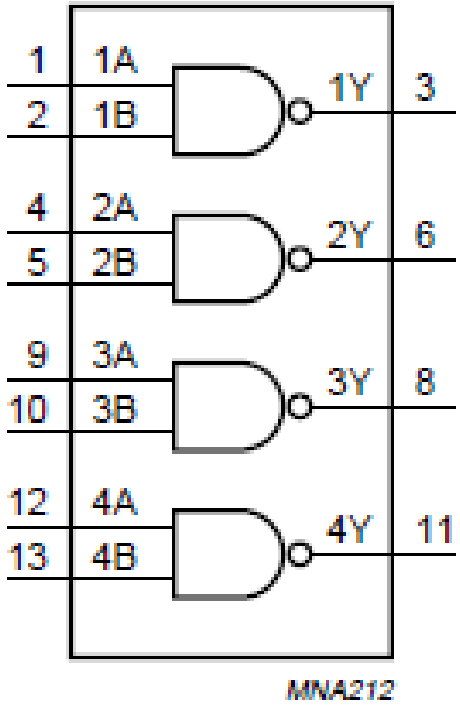


Fig.4 Function diagram.

Key DC Design Parameters

SYMBOL	PARAMETER	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
		OTHER	V_{CC} (V)				
T_{amb} = −40 to +125 °C							
V_{IH}	HIGH-level input voltage		2.0	1.5	–	–	V
			4.5	3.15	–	–	V
			6.0	4.2	–	–	V
V_{IL}	LOW-level input voltage		2.0	–	–	0.5	V
			4.5	–	–	1.35	V
			6.0	–	–	1.8	V
V_{OH}	HIGH-level output voltage	V_I = V_{IH} or V_{IL}					
		I_O = −20 μA	2.0	1.9	–	–	V
		I_O = −20 μA	4.5	4.4	–	–	V
		I_O = −20 μA	6.0	5.9	–	–	V
		I_O = −4.0 mA	4.5	3.7	–	–	V
		I_O = −5.2 mA	6.0	5.2	–	–	V
V_{OL}	LOW-level output voltage	V_I = V_{IH} or V_{IL}					
		I_O = 20 μA	2.0	–	–	0.1	V
		I_O = 20 μA	4.5	–	–	0.1	V
		I_O = 20 μA	6.0	–	–	0.1	V
		I_O = 4.0 mA	4.5	–	–	0.4	V
		I_O = 5.2 mA	6.0	–	–	0.4	V
I_{LI}	input leakage current	V_I = V_{CC} or GND	6.0	–	–	±1.0	μA
I_{OZ}	3-state output OFF current	V_I = V_{IH} or V_{IL}; V_O = V_{CC} or GND	6.0	–	–	±10.0	μA
I_{CC}	quiescent supply current	V_I = V_{CC} or GND; I_O = 0	6.0	–	–	40	μA

Note

1. All typical values are measured at T_{amb} = 25 °C.

- These tell you:
  - Voltage ranges for valid 1 and 0 on input
  - Guaranteed output voltages as a function of load
  - How much load the gate input puts on the output of anything it is connected to

Key AC Characteristics

AC CHARACTERISTICS

Type 74HC00

GND = 0 V; $t_r = t_f$ = 6 ns; C_L = 50 pF.

SYMBOL	PARAMETER	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
		WAVEFORMS	V_{CC} (V)				
T_{amb} = −40 to +85 °C; note 1							
t_{PHL}/t_{PLH}	propagation delay nA, nB to nY	see Fig.6	2.0	–	25	115	ns
		see Fig.6	4.5	–	9	23	ns
		see Fig.6	6.0	–	7	20	ns
t_{THL}/t_{TLH}	output transition time		2.0	–	19	95	ns
			4.5	–	7	19	ns
			6.0	–	6	16	ns

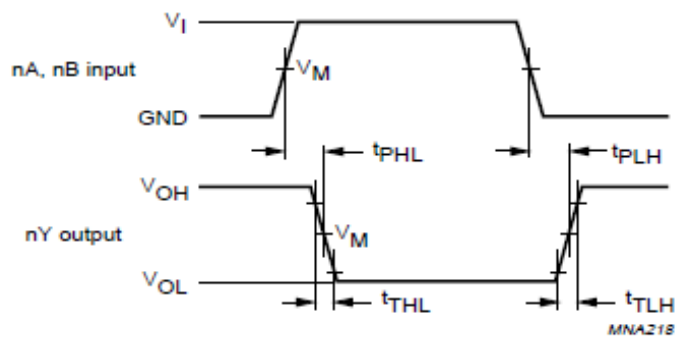


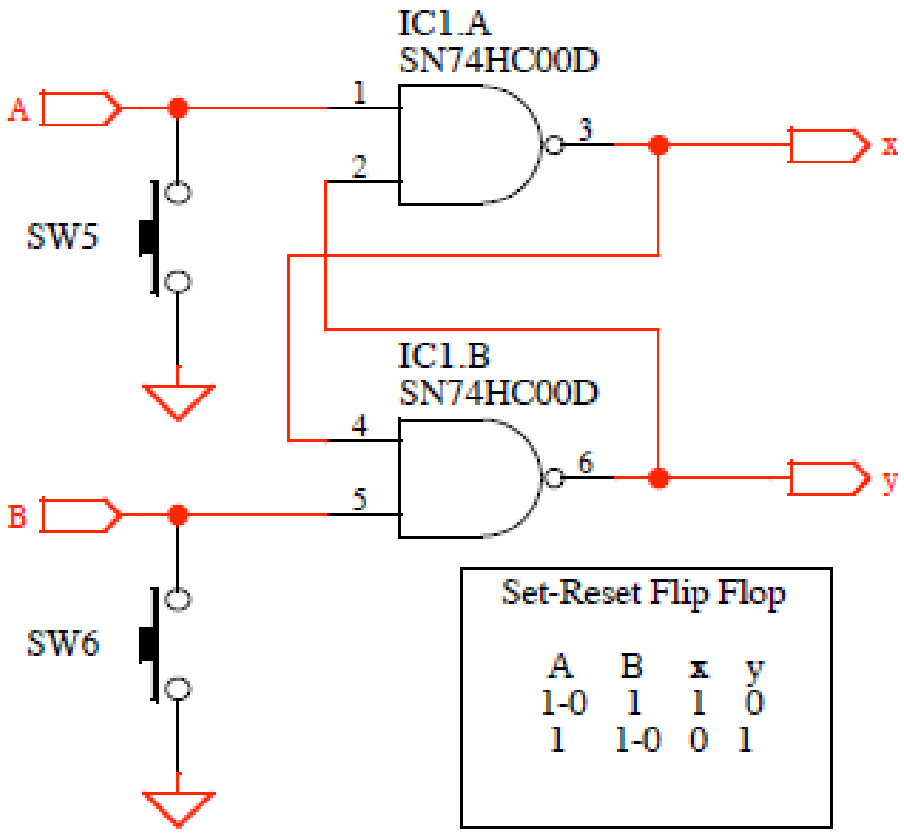
Fig.6 Waveforms showing the input (nA, nB) to output (nY) propagation delays.

- Since this is ultimately an analog device it is essential to consider propagation delays in any array of gates
- The delays are compounded
- Parallel tracks in the logic equation give rise to races and glitches if you are not careful
- Often a clock function is used to allow a “settling time” and then to record (or “register” the output before proceeding to the next step in the device operation

Digital Signal Requirements

- Signals must never remain in “never-never land”. At a minimum this increases power draw and may damage the chip
- Transitions must be monotonic and “fast”
- Slow or sloppy transitions must be conditioned by triggers or discriminators
- If the signals are only slow, most logic families include devices with Schmidt trigger inputs
- Mechanical contacts must be de-bounced

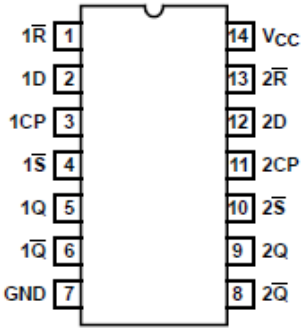
Simple Memory – Set/Reset Flip-Flop



- Pushing a button sets (A) or resets (B) the device
- This state remains until the other button is pushed
- "Set" and "Reset" describe the device state – one output is always the complement of each other

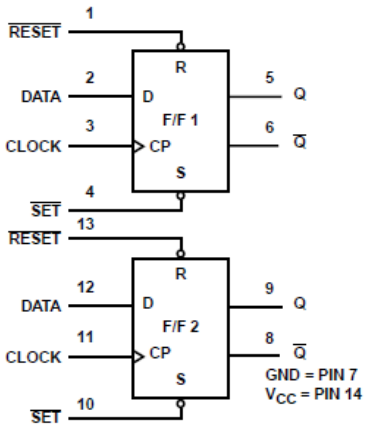
The Clocked "D" Flip-Flop

CD54HC74, CD54HCT74
(CERDIP)
CD74HC74, CD74HCT74
(PDIP, SOIC)
TOP VIEW



Pinout

Functional Diagram



TRUTH TABLE

INPUTS				OUTPUTS	
$\overline{SET}$	$\overline{RESET}$	CP	D	Q	$\overline{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H (Note 1)	H (Note 1)
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Q0	$\overline{Q0}$

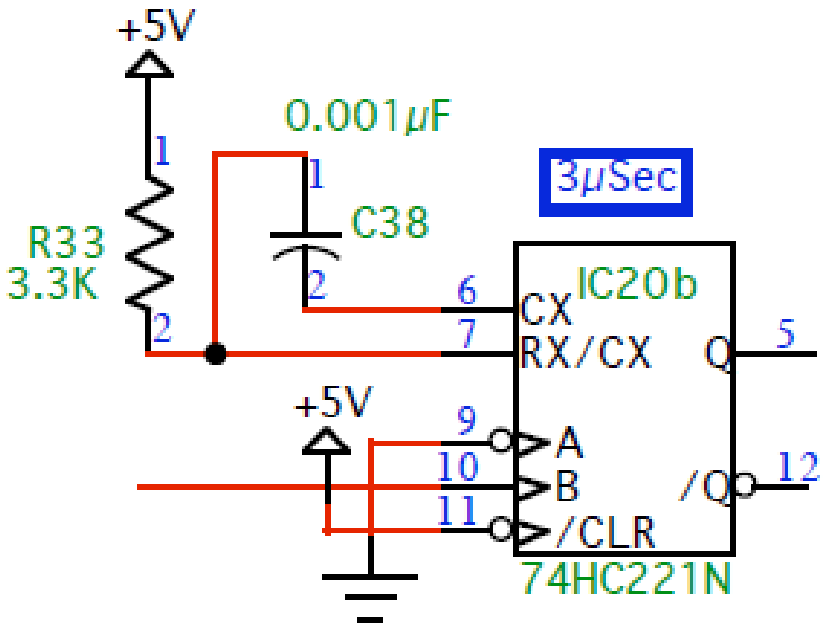
H= High Level (Steady State)
L= Low Level (Steady State)
X= Don't Care
↑= Low-to-High Transition
Q0 = the level of Q before the indicated input conditions were established.
NOTE:

1. This configuration is nonstable, that is, it will not persist when set and reset inputs return to their inactive (high) level.

- On the leading edge of the clock, the state of D is remembered in the outputs
- The state is remembered before the outputs change

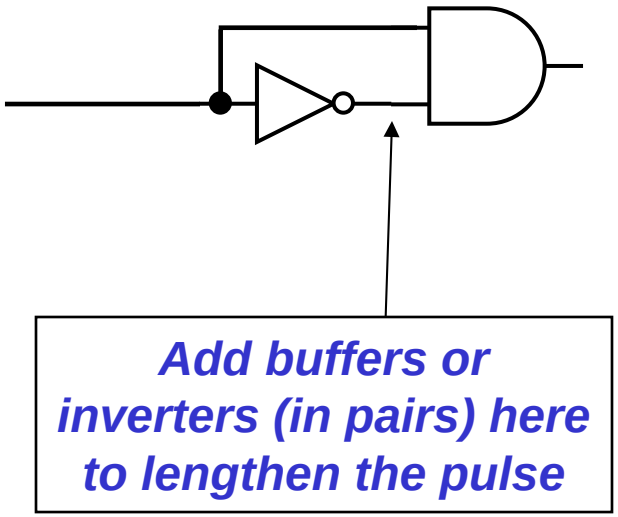
Therefore the outputs may be fed back into the logic that determines the input without ambiguity

One-Shots



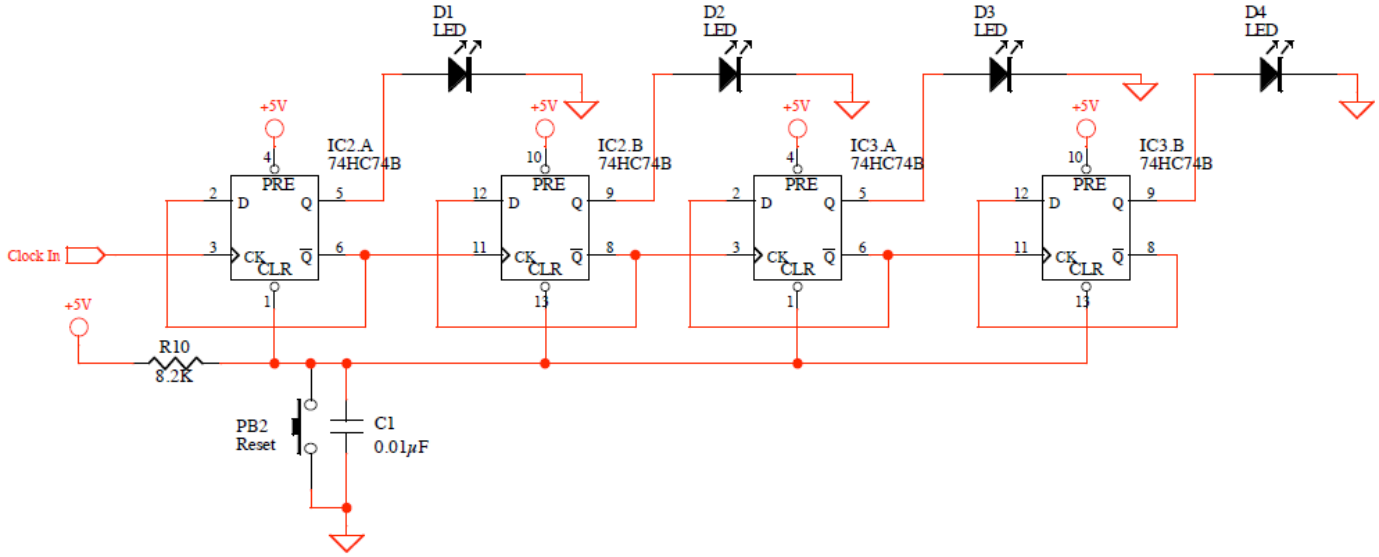
- A “one-shot” produces a pulse in response to a “leading edge” on the input
- The opposite transition produces no response
- If a precise (or very long) pulse duration is needed, use a device where this is controlled by analog components

Simple One-Shot for Edge Triggering



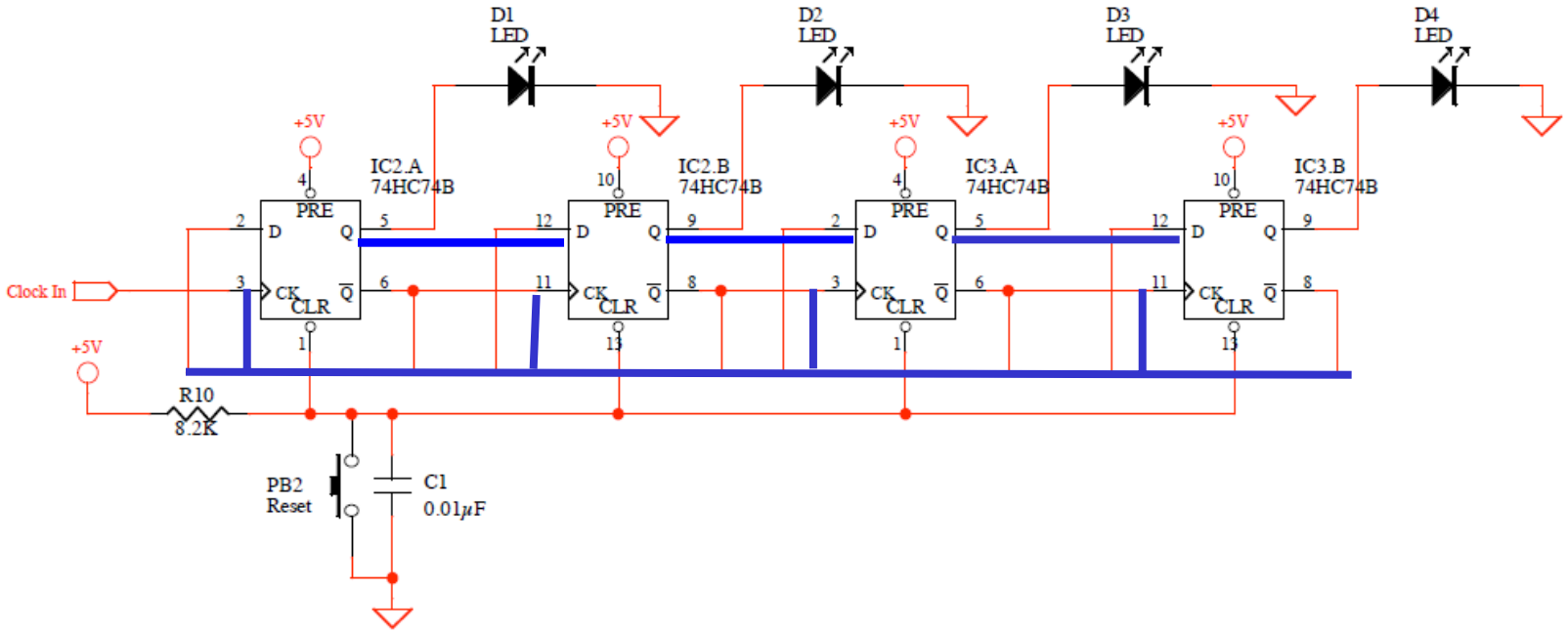
- Produces a short pulse on positive edge
- No reaction to the following negative edge
- Duration is a function of temperature, supply voltage, and device variation
- If implemented with devices in the same logic family the pulse is usually long enough with one inverter
- More buffers or inverters will lengthen it (to a point)

Ripple Counter from D Flip-Flops



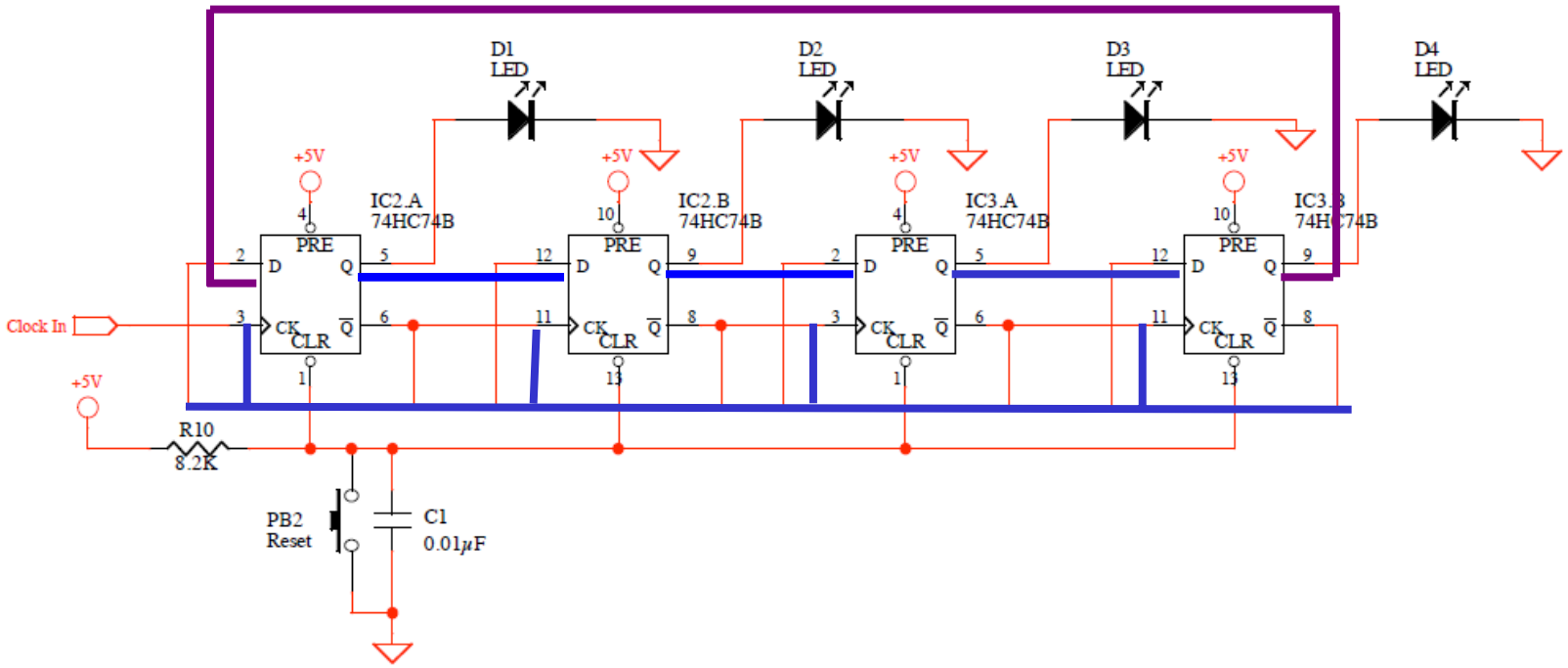
- When clocked, each D changes state because Qbar is fed back into D
- When and only when Qbar rises the next flipflop is clocked
- Each thus operates at half the rate of the previous stage forming a counter
- Because of the ripple, not all stages change state at the same time so the count is actually only well defined after a settling time
- Synchronous counters also exist, where all outputs change simultaneously
- Even there, the finite time of the transition can cause glitches

Shift Registers



- The ripple counter becomes a “shift register” if Q of one stage is connected to D of the following, and all the clocks are connected together

Johnson Counter – Multiphase Clock

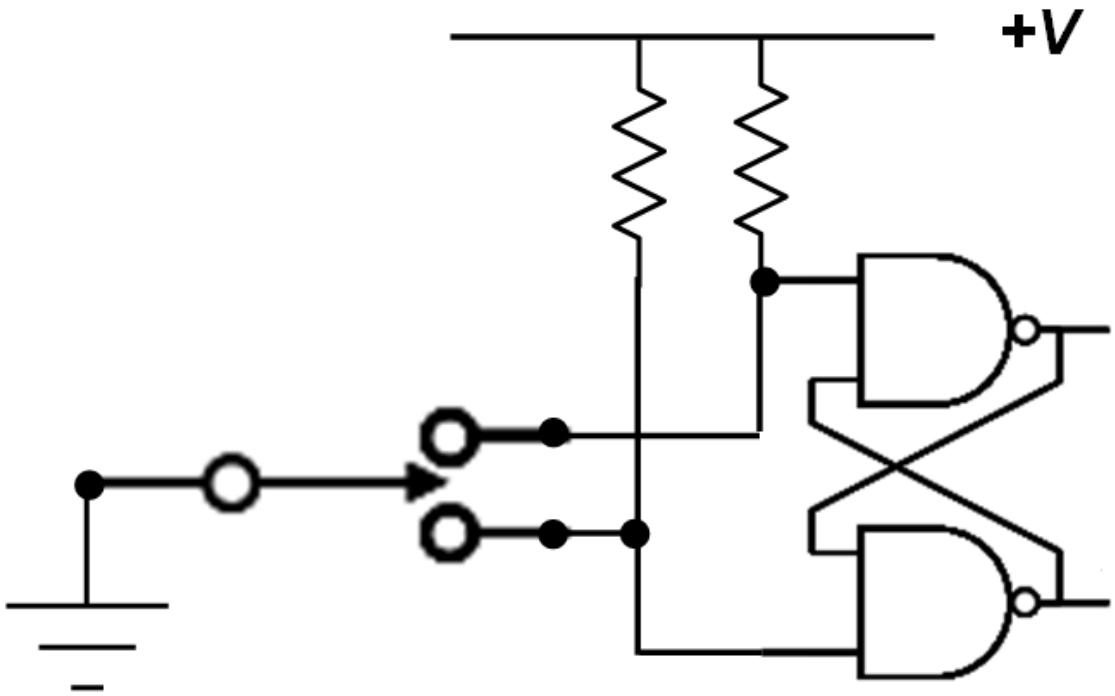


- A shift register loaded with a single “1” becomes what is known as a Johnson Counter
- The “1” moves from stage to stage, and is often the simplest way to sequence logic and allow for setup time
- Fed back on itself this becomes a “multiphase clock”
- Typically only the “even” phases are used to sequence logic, ensuring that there is no overlap and good settling time

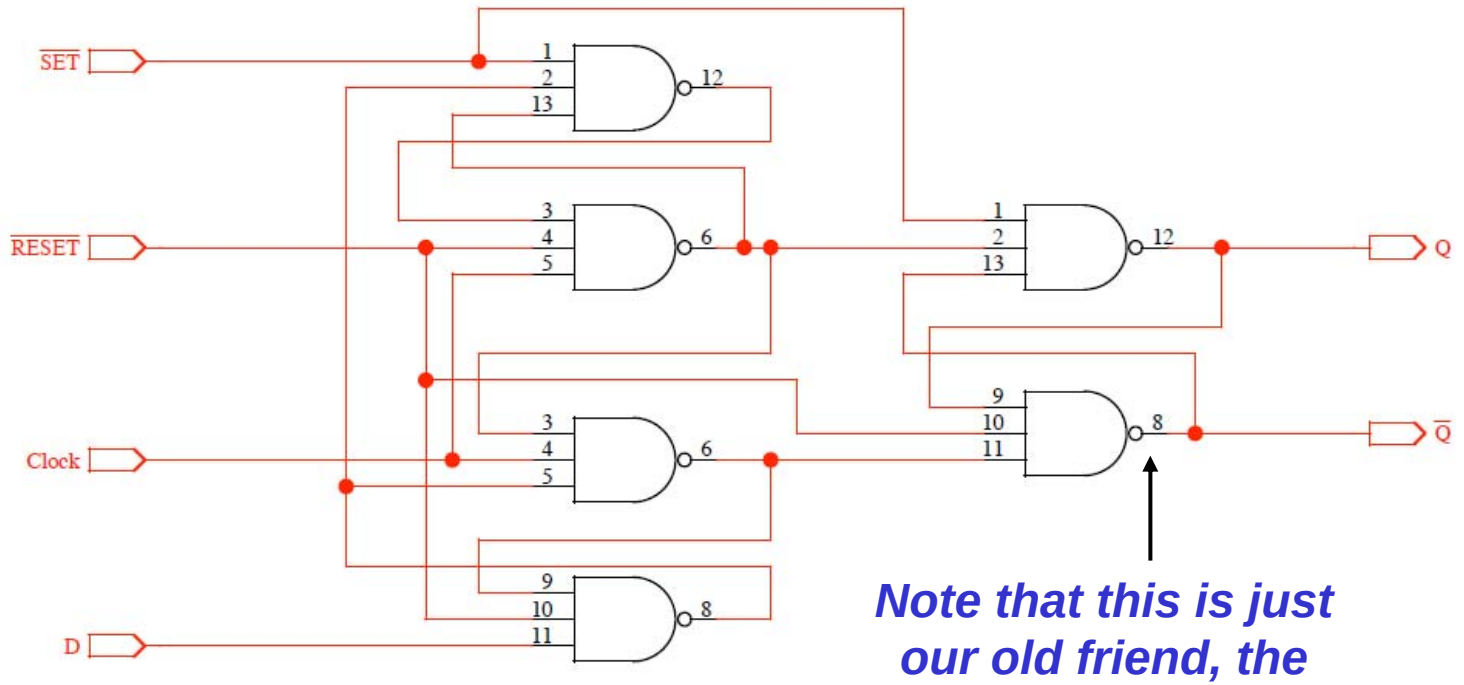
Problem 1: Implement the following using only NAND gates

- Buffer
- Inverter
- AND
- OR
- XOR (Exclusive Or)

Problem 2: Explain How This Circuit "Debounces" the Contacts



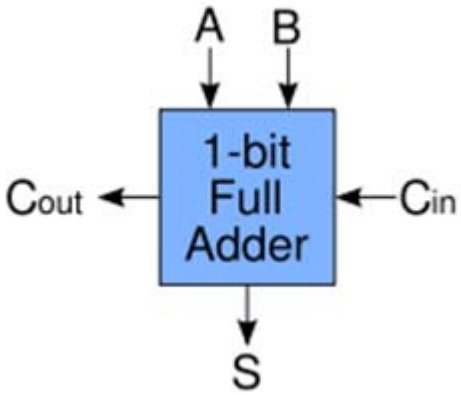
Problem 3: How a D Flipflop Works



Note that this is just our old friend, the set-reset flipflop

- Figure out how gate delays implement the edge triggered feature of the D flipflop
- Using the chips on the prototype board build this circuit and use it to add one more stage to the counter

Problem 4: Design a Full Adder



Input			Output	
A	B	C_i	C_o	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S = (A \oplus B) \oplus C_{in}$$

$$C_{out} = (A \cdot B) + (C_{in} \cdot (A \oplus B)) = (A \cdot B) + (C_{in} \cdot B) + (C_{in} \cdot A)$$

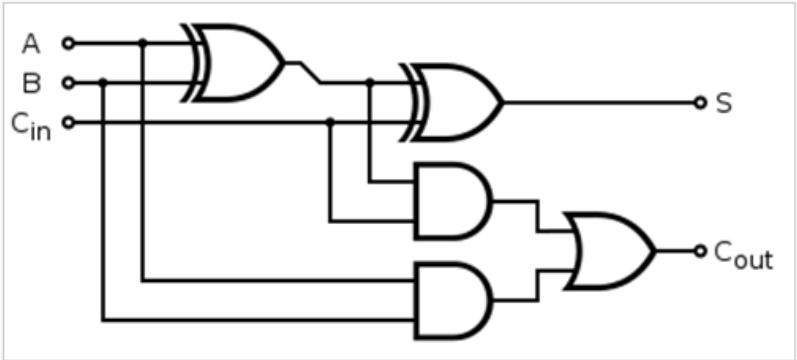
Problem 5: Design an 8 Bit "Accumulator" to Add 4 Bit Numbers

- An accumulator is an array of memory cells to which successive numbers can be added
- Use four of the switches for input
- Use one of the push buttons as a clock
- Display the results in the lights

Problem 6: Design a Circuit to Multiply Two 4 Bit Numbers

- Multiplying two, four bit numbers yields an eight bit product
- This actually could be implemented with a lot of gates, since there is an 8 bit → 8 bit truth table that uniquely specifies the answer, but here you should do it with clocked logic
- Use two groups of four switches for the inputs
- Display the product in the lights
- Use a push button to initiate the clock sequence

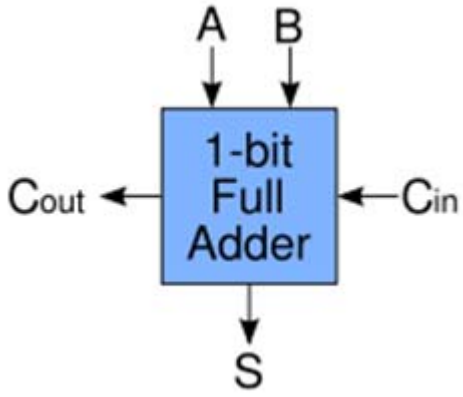
Problem 4: Answer



Full adder circuit diagram

Inputs: {A, B, CarryIn} → Outputs: {Sum, CarryOut}

Input			Output	
A	B	C_i	C_o	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1



$$S = (A \oplus B) \oplus C_{in}$$

$$C_{out} = (A \cdot B) + (C_{in} \cdot (A \oplus B)) = (A \cdot B) + (C_{in} \cdot B) + (C_{in} \cdot A)$$