Mission Overview

Your mission is to help develop a "Load Before Launch Sequencer" (LBLS) for the USS Harry S. Truman (CVN-75). The purpose of the LBLS is to alert the "Yellow Shirts" (the people who flag the planes on and off of the ship) as to how many planes are in the launch queue. The LBLS system monitors the number of planes in the queue by counting them as they roll over a sensor in the deck. Your job is to build a system that does the following:

- Count the number of jets in the launch queue (which can range from 0 to 7). Use *momentary* switch #0 as the deck sensor. Each time this switch is thrown, a new jet is either added (brought up from the hangar deck) or subtracted (launched from a catapult) from the launch queue, depending in the current mode. This switch should be LOW when it is not in use (i.e., in a column labeled 0).
- SPDT switch #0 will be the mode control. It is used to indicate whether the next jet should be added or subtracted. When this switch is low, the next jet should be added. When it is high, the next jet should be subtracted. The queue should not change when this switch changes; the queue only changes when a momentary switch is used.
- The number of jets in the queue will be indicated using a 7-segment decimal display.
- Three LEDs will be used to indicate the queue status:
  - Status light L1 is on only when the queue is empty (Q = 0).
  - Status light L2 is on only when the queue has 1, 2, 3, 4, 5, or 6 planes in it (Q=1, 2, 3, 4, 5, or 6).
  - Status light L3 is on only when the queue is full (Q=7).
- If Q = 7, the system should remain at Q = 7 if the user attempts to add another plane.
- If Q = 0, the system should remain at Q = 0 if the user attempts to subtract a plane.
- Momentary switch #1 should be used to reset the queue (to Q = 0) in case all the planes fall off the ship due to a big wave.

I/O Summary: The final system should have:

3 LEDs	Queue status indicators
one 7-segment display	Queue quantity indicator
one SPDT switch	Add/subtract mode setting
two momentary switches	One to add or subtract, and one to clear

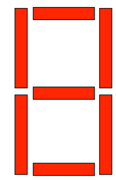
Equipment: Two 74112 chips (JK flipflops)
One 7-Segment display/decoder combination package
One 7400 NAND chip, one 7402 NOR chip, two 7408 AND chips, one 7432 OR chip, one 74151 MUX chip, and one 7404 inverter chip.

Discussion of New Equipment

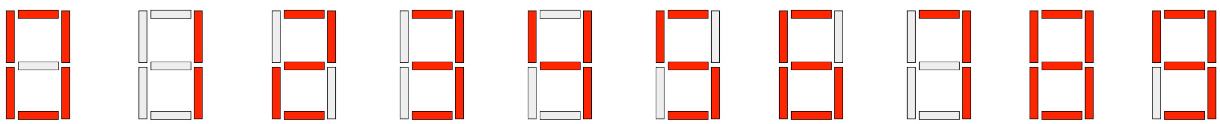
7-Segment Displays

So far, our only output device has been the single element LED (Light Emitting Diodes). In our pre-made black project boxes, the LEDs are already fixed up with other elements for us, so that they light if you give them 5 volts, and don't light otherwise.

This week, we are using a more sophisticated output device: the 7-segment display. These are really a combination of 7 simple LEDs, each in the shape of a long bar. The 7 are grouped in a funny way; they look like this:



We've all seen these used before. Depending on which of the 7 segments are turned on or off, they can *look like* a digital number.



As you can see, some of the 7 segments are used more often than others. For these 10 digits, the top LED is used in 8 of them, but the lower left one is used in only 4 of them. Figuring out a logic circuit to attach to each LED is a little tedious. Fortunately, many 7-segment displays (including the ones we'll use today) are provided with a special chip, called a *decoder*, already attached. This decoder takes in four inputs that represent a binary number, and figures out which of the 7 outputs need to active to get the LEDs to light up correctly.

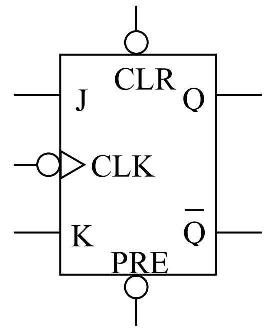
This group of 7 LEDs can visually represent a digital number from 0_{10} to 9_{10}; in binary, that's 0000_2 to 1001_2. As you can see from the "1001_2", we'll need 4 bits to represent the input to the decoder. The decoder chip we're using today is the 7447 chip. For your convenience, we have already connected the 7447 decoders to the 7-segment LEDs, along with some resistors and such. So, you'll learn how to use the display and its decoder as a single unit (see next page).

For this week's project, we will always keep the LT, RBI, and BI/RBO inputs *high*. The 4 bit input is represented by wires DCBA, with D being the MSB, and A being the LSB. However, we only want to count from 0 to 7 today. Since the BCD ranges from 000 to 111, we only need to control 3 of the 4 wires to our 7-segment display; D will always be held low.

Flip Flops and Simple Memory

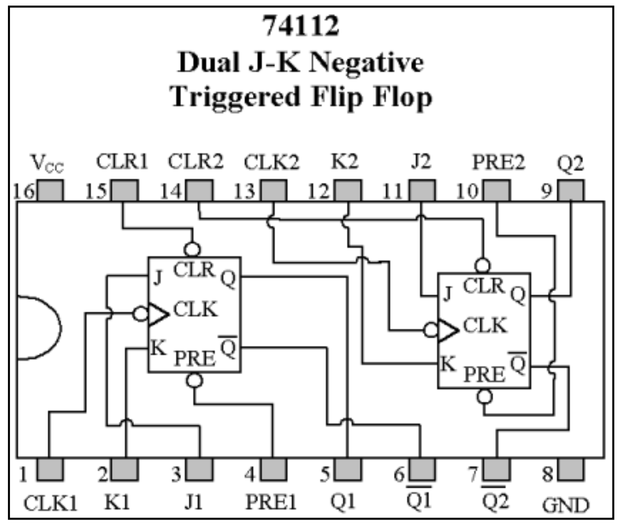
If we are going to count airplanes, then we need some way to *remember* things: we need memory. The simplest memory unit available is called a "Flip-Flop". A single Flip-Flop can have a variety of uses; one such use is as a one-bit memory. There are many kinds of flip-flops; today, we will be using JK Flip Flops. This particular chip, the 74112, actually has two JKFF units on board (just as many of our chips have 4 logic gates on board).

A single JKFF has 5 inputs, and 2 outputs. Here is the schematic, and the detailed chip layout:



Here is how it works:

1. If "Preset" PRE = 0, then Q is set to 1.
2. If "Clear" CLR = 0, then Q is set to 0.
3. If PRE = CLR = 0, it's a disaster. The chip tries to do both rule 1 and rule 2 at the same time, and neither one wins. Don't let this happen.
4. During normal operation, PRE = CLR = 1. If this is true, then nothing happens to Q unless CLK *changes* from 1 to 0. Symbolically, this is written as CLK = ↓. When this change happens, then Q *might* change, depending on the values of J and K at the time CLK changes:
   a. If J = K = 0, then Q is unchanged.
   b. If J = 1, and K =0, then Q is set to 1.
   c. If J = 0, and K =1, then Q is set to 0.
   d. If J = K = 1, then Q toggles to the opposite of whatever it was before. If Q was high before CLK changed, then Q becomes low. If Q was low, then it becomes high. This mode is the one we'll always use today.



Since we need a 3-bit number to count our jets, we'll need three JKFF's today (one for each bit). Also, we'll always keep the PRE set to "high" for this entire project.

Note: The text mentions counters on p. 146, but unfortunately that circuit is much more complicated than it needs to be, due to using D flip flops instead of JK flip flops.

Procedure for This Week:

You will be graded in stages. You may not proceed to the next stage until the instructor has checked your work on an earlier stage.

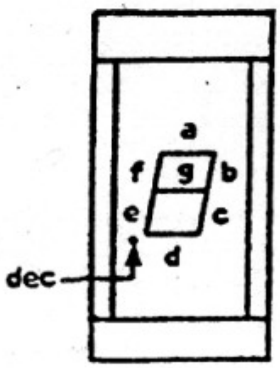
Part 1 [10 points]: Using only the 74112 chips and the 7 segment display, build a *ripple counter* circuit that can add one jet at a time, or clear them all (but not subtract any). Ripple counters are not mentioned in our textbook. The "Add" switch should be LOW when it is not being used.

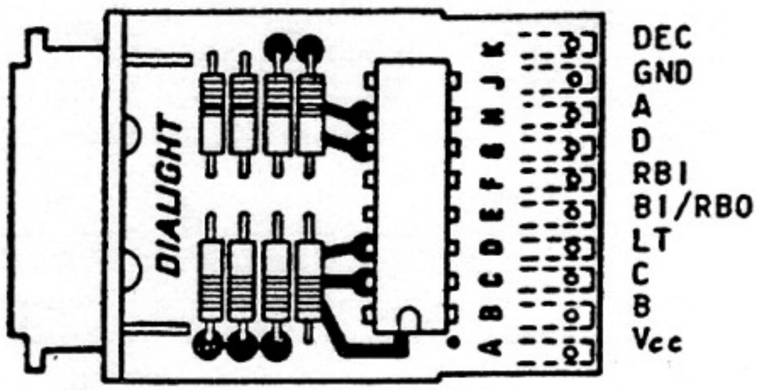
Part 2 [5 points]: Build the 3-LED status panel display using the NAND and NOR chips.

Part 3 [3 points]: Modify the circuit to allow subtraction of jets, using the SPDT switch as your "mode" control, plus two OR gates, 6 AND gates, and an inverter. **This requires converting to a *synchronous counter*.** There will be class discussion about this.

Part 4 [2 points]: Use the MUX and one more AND gate to prevent the circuit from adding when $Q = 7$ or from subtracting when $Q = 0$. The three select inputs to the MUX will be L1, L3, and the "mode" switch.

7 Segment Display Packages





DECIMAL OR FUNCTION	INPUTS			*	RESULTANT DISPLAY
	LT	RBI	D C B A	BI/RBO	
0	1	1	0 0 0 0	1	0
1	1	X	0 0 0 1	1	1
2	1	X	0 0 1 0	1	2
3	1	X	0 0 1 1	1	3
4	1	X	0 1 0 0	1	4
5	1	X	0 1 0 1	1	5
6	1	X	0 1 1 0	1	6
7	1	X	0 1 1 1	1	7
8	1	X	1 0 0 0	1	8
9	1	X	1 0 0 1	1	9
BI	X	X	X X X X	0	
RBI	1	0	0 0 0 0	0	
LT	0	X	X X X X	1	8

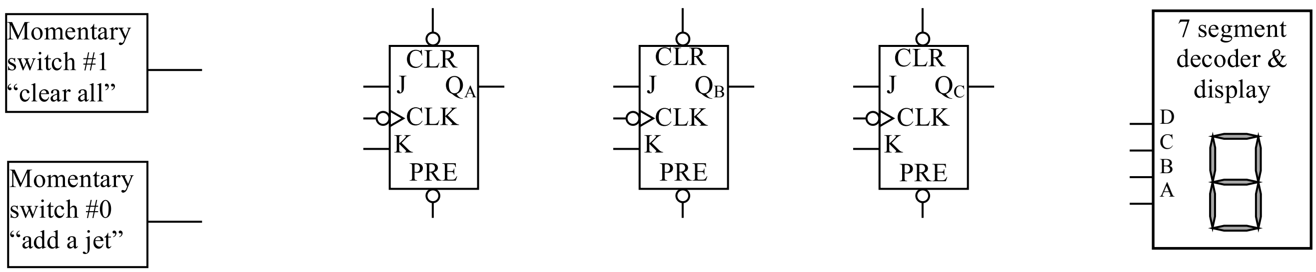
'1'=HIGH LEVEL, '0'=LOW LEVEL, 'X'=ANY LEVEL

* BI/RBO IS AN INPUT AND/OR OUTPUT

Name:_________________________________

You MAY NOT write on this sheet in pen!

Part 1: Your count-up circuit (requires only wires, grounds, and Vcc's):



Part 2: Your status panel (uses 4 NAND gates and 4 NOR gates):

Part 3 & 4: Your final circuit (do not redraw the above status panel):