[15 points; 5 points for each for result] Does the timing of the 555 clock chip output match the predictions of the equations for frequency, t\_high, and t\_low that are given below the schematic? Show a calculation of the expected value for each of the three parameters.

Yes, it does.

f = 1/( ln(2)\*C2\*(R1 + 2\*R2) ) = 1/ ( ln(2)\* 1000\*(470 + 2\*470) )= 0.00000102318 1/uF\*ohm = 1.02318 1/F\*ohm = 1.02318 s^-1

T\_high = ln(2)\*(R1 + R2)\*C2 = ln(2)\*(470+470)\*1000 = 651558.349726 ohm\*uF = 0.651558 ohm\*F = 0.651558s

T\_low = ln(2)\*R2\*C2 = ln(2)\*(470)\*1000 = 325779.174863 ohm\*uF = 0.325779 ohm\*F = 0.325779s

[15 points] Counter outputs are made in the weighted positional unsigned integer format. The 74HC163 is a 4-bit counter, but the decoder requires a 3-bit counter input to sequence through its inputs modulo 8.

How did you obtain outputs equivalent to a 3-bit counter from the outputs of the 4-bit counter?

We can simply assert the last 3 outputs and ignore the most significant number. Because A number that is greater than 8 is just 8+(number-8), which behaves the same on last 3 outputs as number-8. For example, 9 is 1001 and 1 is 001, if we assert the last 3 outputs, they are the same. Same for all other cases.

What are all the possible combinations of three 4-bit counter outputs that provide the same sequence as a 3-bit counter?

000

001

010

011

100

101

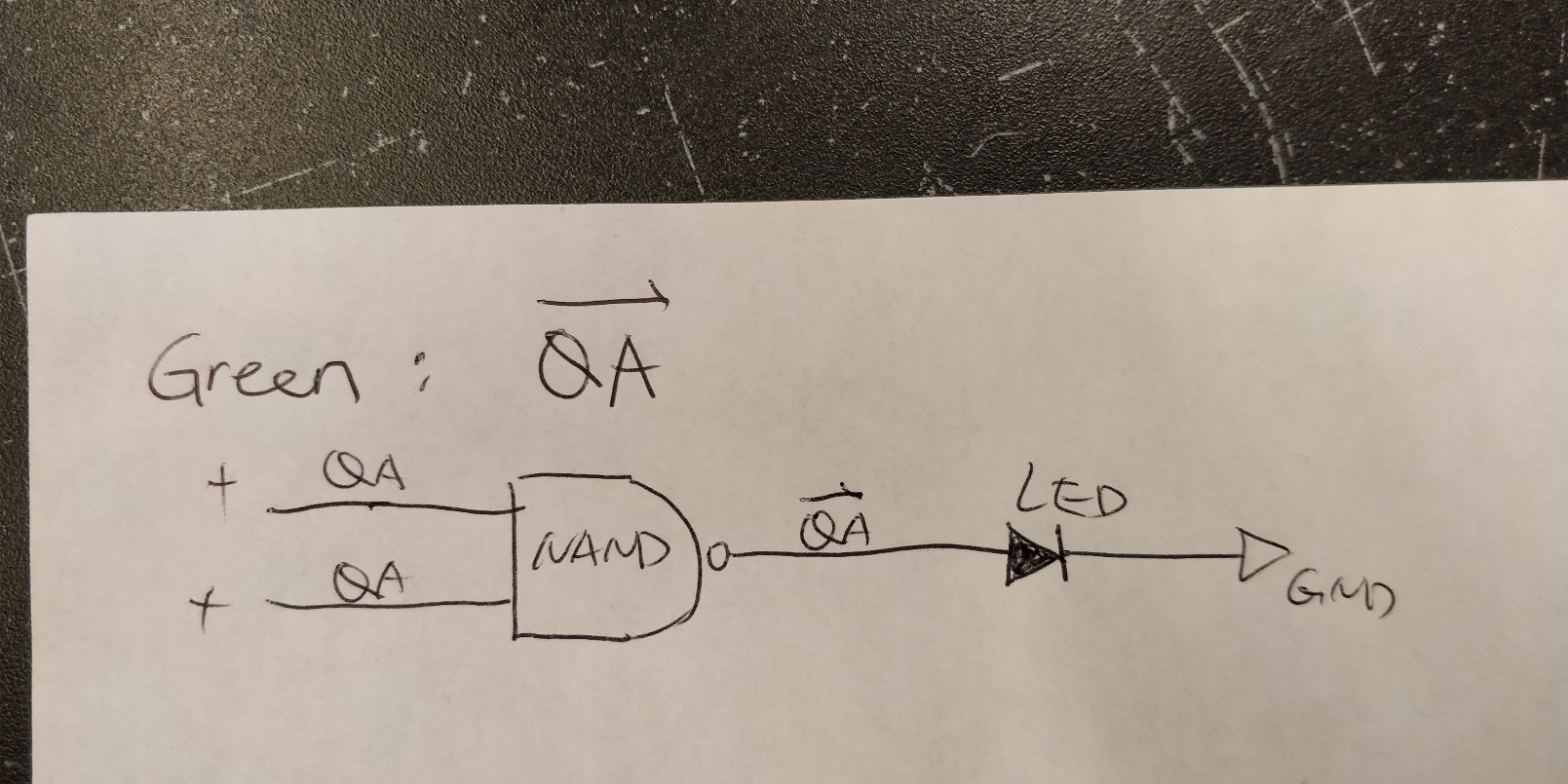
110

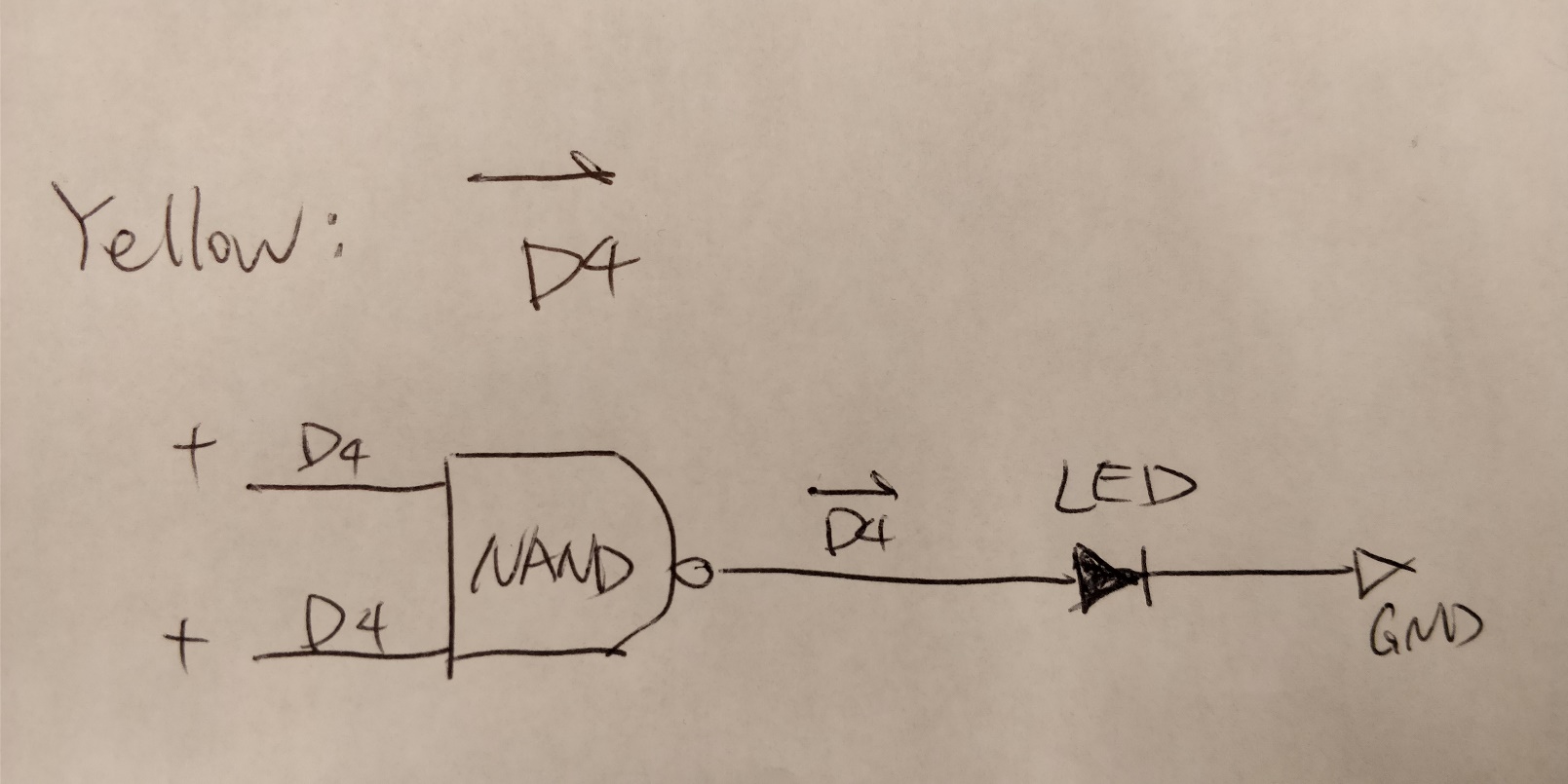
111

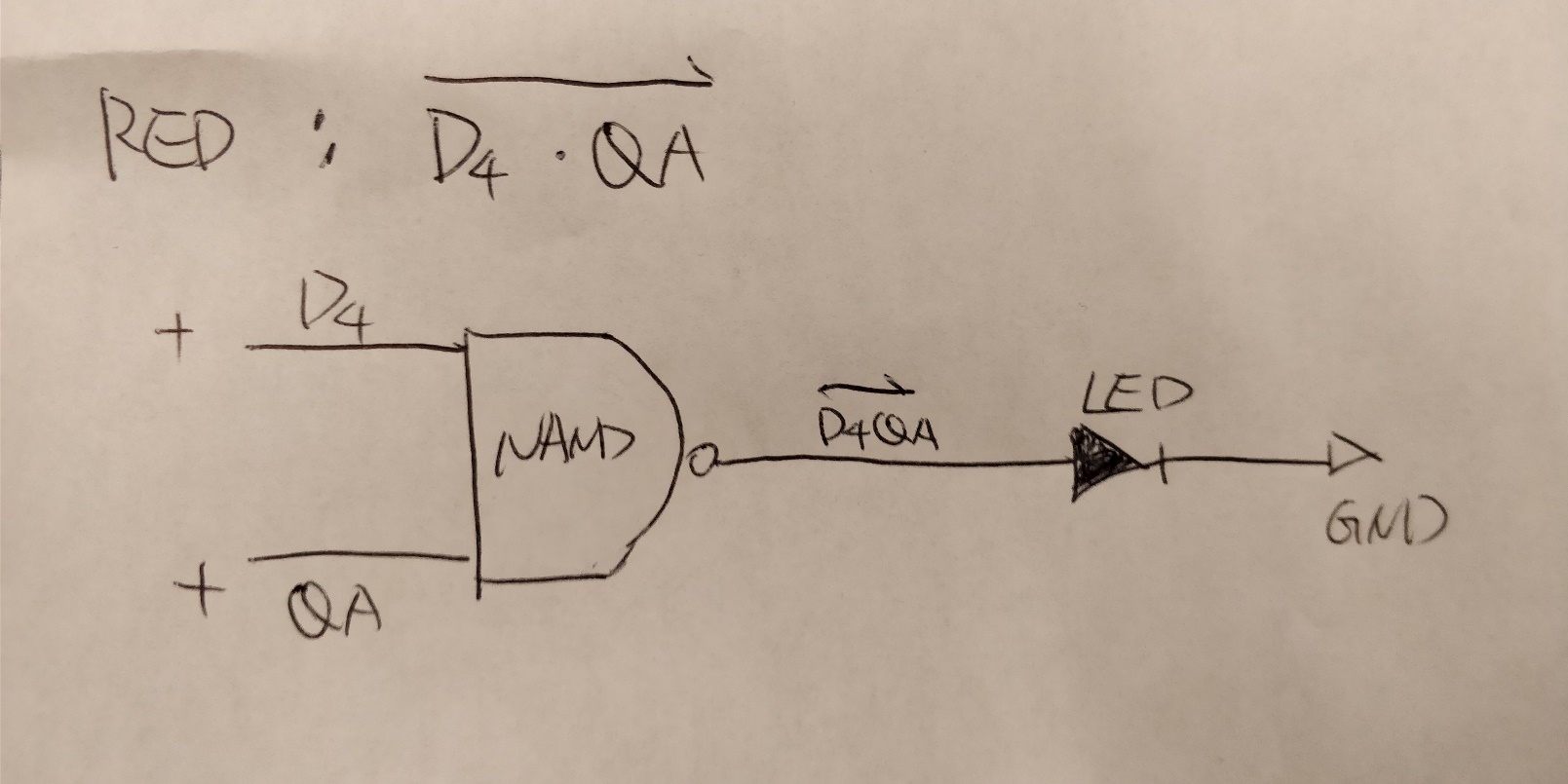
Why did you not choose the other ways to obtain a 3-bit counter from the 4-bit counter outputs for your automatic sequencer circuit?

Because in this project, we need a consecutive 8 numbers. This is the best way and probably the only way that can get outputs as a sequence from 0 to 7.

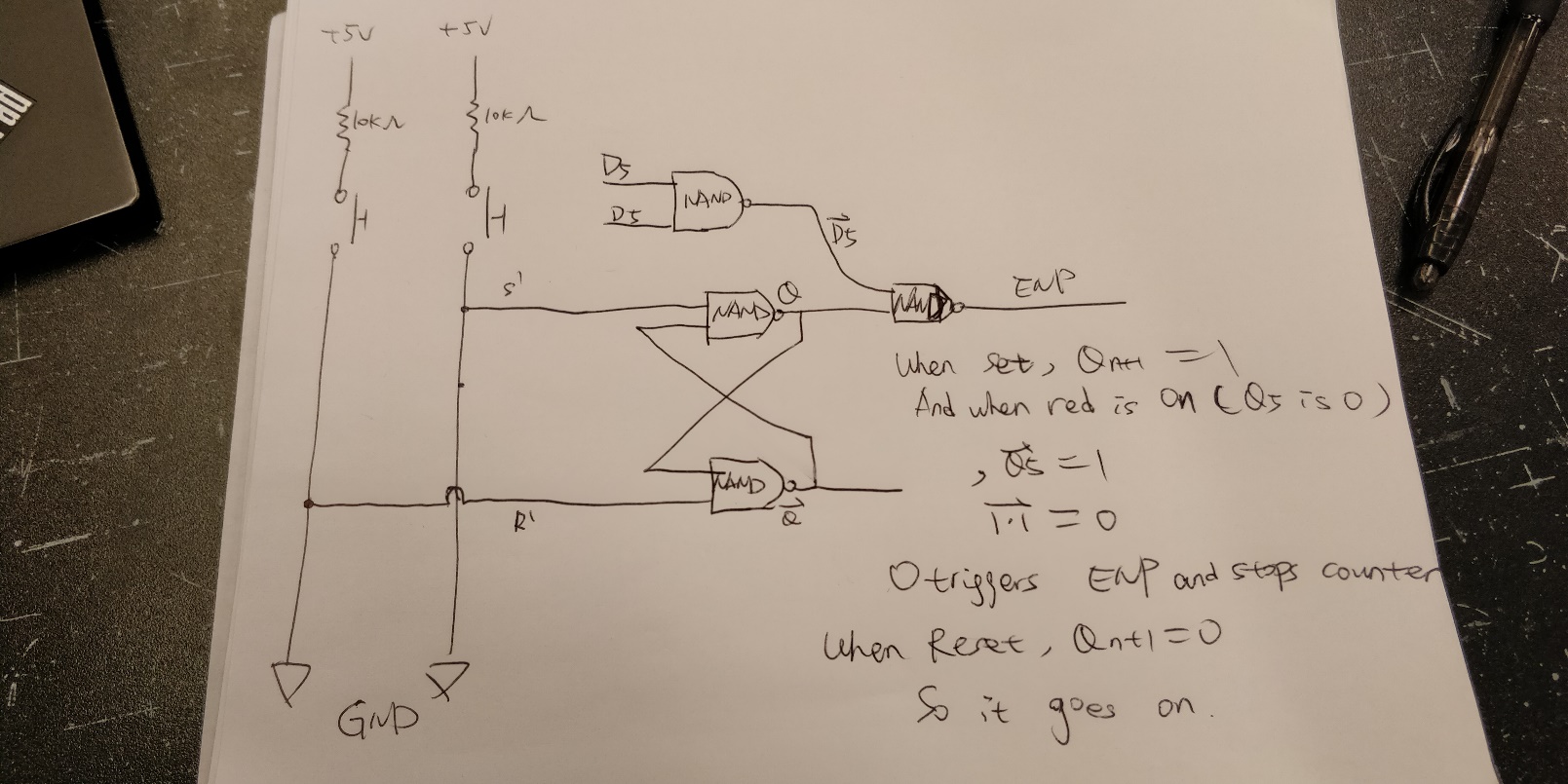
[30 points, as follows] Derive the Boolean logic expressions for each color of the traffic signal (Function 1) and simplify in terms of 2-input NAND and NOR gates. Show your work to earn credit. Your Boolean expression must be in terms of QA, QB, QC, and QD, for the counter outputs, and/or D0 – D7 for the eight decoder/demux outputs. Draw the final schematic diagram using NAND and NOR gates for each color.

[10 points] Show green light Boolean expression and schematic drawing of gates only here.  


[10 points] Show yellow light Boolean expression and schematic gates only here.  


[10 points] Show red light Boolean expression and schematic drawing of gates only here.  


[10 points] Draw the gate-level schematic of your circuit for Function 2. Clearly label all inputs with their source. Clearly show how this circuit interfaces with the automatic sequencer.



[20 points] Demonstrate your circuit to your TA during your lab session.

Function 1 [15 points, as follows] [5 points] Green light turns on for 4 seconds, followed by [5 points] Yellow light turns on for 1 second, followed by [5 points] Red light turns on for 3 seconds. (Order of light color sequence must be correct, not reversed. There must be a single, distinct logic signal for each one of the three light colors.)

Function 2 [5 points] Light sequence stops during red and restarts as described.

[10 points] Live questions from your TA during lab.

## Build an automatic sequencer (a computational platform)

An automatic sequencer (see our textbook, Figures 2.21 and 2.22) is a basic building block of computers. It combines three key functions: Clock, Counter, and Decoder. A clock is a circuit that emits a steady stream of alternating logic levels and is the key to controlling behavior using time. A counter generates a sequence of numeric values modulo some base. With counter output sent to the address inputs of a decoder, a sequence of assertions of decoder outputs becomes available to control a collection of devices for any number of purposes.

The lab kit contains three integrated circuits packaging each of these key functions: Clock (555 Timer), Binary counter (74HC163), and Decoder/Demultiplexor (74HC138). Use these chips to build the automatic sequencer step by step as follows.

## Step 1. 555 timer

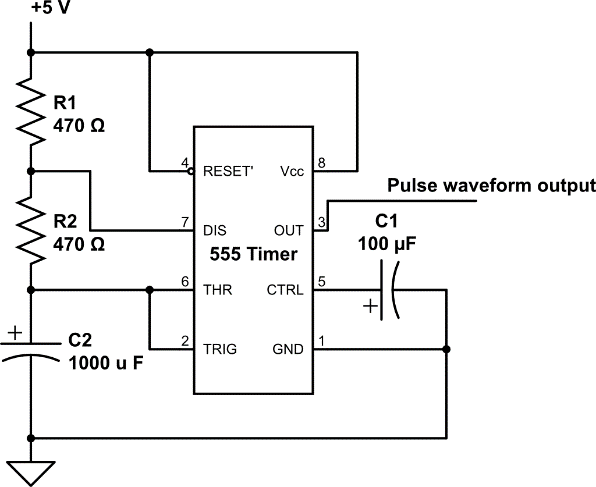
Many digital circuits need a centralized pulse source, called a clock, to synchronize signals and events. The 555 timer is a widely used, general purpose circuit for timer, pulse generation, and oscillator applications. Read more at

<http://www.ehow.com/facts_5977976_555-timer-used-for_.html> and <http://en.wikipedia.org/wiki/555_timer_IC> . See an animation of 555 operation at <http://www.falstad.com/circuit/e-555int.html> . The functions of the 8 pins of the 555 chip are as follows.

|  |  |
| --- | --- |
| Pin | Name and purpose |
| 1 | Ground connection |
| 2 | Trigger. When this input falls below 1/2 of the voltage at CTRL (Pin 5) then OUT (Pin 3) emits a rising edge and a timing interval starts |
| 3 | Output. This pin provides the clock signal output. |
| 4 | Reset. Active low. Timer is reset by driving this input to GND. Timing begins when RESET rises above approximately 0.7V. |
| 5 | Control. Provides control access to the internal voltage divider (by default, 2/3 Vcc). |
| 6 | Threshold setting. The timing interval ends when the voltage at THR is greater than CTRL. |
| 7 | Discharge. | |
| 8 | Vcc. Connection to positive voltage supply. |

When configured for clocking operation (called astable) the schematic for the 555 looks like this. Note carefully that the pin positions on the 555 symbol in the schematic are not in the same configuration as they are on the physical chip. This is to make the schematic have a simple, untangled arrangement of wires. The pins on the physical 555 chip are numbered 1 to 8 with 1 at the lower left corner with the chip upright, 4 at the lower right corner, 5 at the upper right, and 8 and the upper left.

**Use caution in connecting capacitors C1 and C2. They are polarized. Be sure the match the polarity orientation shown in the schematic.**



The values of the external components R1 and R2 in Ohms and C2 in units of Farads control the waveform of the pulses emitted by the 555. The frequency f of the pulse stream generated by the 555 is given by the equation f = 1/( ln(2)\*C2\*(R1 + 2\*R2) ). The time that the signal is at a high voltage t\_high within each pulse is t\_high = ln(2)\*(R1 + R2)\*C2. And the low time from each pulse is t\_low = ln(2)\*R2\*C2.

The power capability (heat dissipation rating) of R1 must be greater than Vcc2 / R1 Watts. The smaller the physical size of a resistor, the smaller its heat dissipation capability.

>>> Double check that capacitors C1 and C2 are placed into the circuit with correct polarity.

Once building the 555 circuit is done, use an LED to probe the output of the 555 to check for correct operation

## Step 2. Binary Counter

Next add the <http://www.futurlec.com/74HC/74HC163.shtml> 74163 binary counter to the microcontroller circuit. A pin diagram is available through the link.

|  |  |
| --- | --- |
| Name | Purpose |
| Power & Ground | As is common, Pin 16 is supply voltage (Vcc) and Pin 8 is Ground (zero reference, or GND). |
| Clock input | Pin 2 is the clock input pin.| |
| Data input | Pins 3, 4, 5, and 6 are data input pins corresponding to D0 to D3. Unused. Safe to leave unconnected. |
| Count enable | This pin provides the amount to increment the count by. If it is zero the count does not advance. If it is 1, then the count advances by one with each rising edge of the clock signal. This design also allows connecting one counter to the next. Connect Pin 7 to Vcc to enable the counting function. |
| Clear | An active low (active with a 0 value) input. Connect Pin 1 to +5 V (Vcc) to not clear the counter. |
| Flip-flop output | Pins 14, 13, 12, and 11 are the output pins C3, C2, C1, and C0, respectively. |
| Count enable carry input | Connect Pin 10 to Vcc to enable the carry input function. This is used to connect multiple counter chips into one larger counter. |
| Parallel enable input | Allows loading a count value from the data input pins. Connect Pin 9 to Vcc to disable this function. |

Connect the 555 output to the clock input of the 74HC163. Then wire the rest of the 74163 to enable counting. Probe the 74163 output signals using LEDs to confirm correct operation.

## Step 3. Decoder/Demultiplexer

Finally, add the <http://pdf1.alldatasheet.com/datasheet-pdf/view/51038/FAIRCHILD/74138.html> 74138 decoder/demultiplexer to the circuit. A pin diagram is available through the link.

|  |  |
| --- | --- |
| Name | Purpose |
| Power pin | As usual, Pin 16 and Pin 8 correspond to the voltage supply (Vcc) and ground (GND). |
| Output pin | Pins 15 through 9 correspond to outputs “Y0” through “Y6”. Pin 7 outputs “Y7”. |
| Enable pin | Pins 4, 5, and 6 are the enable pins an allow daisy-chaining this chip with one or three copies to make 4x16 and 5x32 decoders, respectively. To enable only 3x8 decode, set pins 4, 5, and 6 to LOW, LOW, and HIGH respectively. |
| Input pin | Pins 1 through 3 are the address pins to be decoded. Connect them to the counter output. |

The 74HC163 is a 4-bit counter, but the decoder input is a 3-bit address. For your circuit design, you will need to decide which three of the 74HC163 output signals to send to the decoder so that each output of the decoder is selected in turn, or in sequence, modulo 8.

Use 8 red and green LEDs as probes of the output of the 74HC138 and verify correct circuit operation – each output goes low one at a time in turn.

## Circuit to design that uses and controls the automatic sequencer

Using the automatic sequencer circuit as a computational platform, design and build an additional computational logic circuits to compute three signals for a traffic signal as follows.

**Function 1:** Traffic lights continuously cycle the illumination of three different colored lights: green, yellow, and red. The traffic signal for this lab must show green for 4 seconds, followed by yellow for 1 second, followed by red for 3 seconds, and repeat in this order as long as power is supplied. Design a circuit to implement this function.

**Function 2:** Design the additional capability to have the traffic signal stop automatically upon reaching display of the red signal with a continuous display of red. This functionality will be enabled (triggered to occur when the time is correct (red light comes on) and not sooner) at any time within the cycle of lights by pushing a button switch briefly. This functionality will be disabled, and the traffic light sequence will immediately resume, by briefly pushing a second pushbutton switch. [Actual traffic signals may have the capability for an approaching emergency vehicles to suspend normal light sequencing so that cross-street traffic has a steady red light and the emergency vehicle receives a green light for its direction of travel, until the emergency vehicle clears the intersection, when normal light sequencing resumes.]

Design three combinatorial circuits, one to compute the signal for each traffic light color. Design the feedback circuit necessary to carry out Function 2. The automatic sequencer circuit provides a variety of pre-computed outputs: eight from the demultiplexer and three from the counter. Any of these outputs may be useful operands in support of computation of each traffic light color signal and for the feedback signal. Use pushbutton voltage divider input circuits, NAND gates, and NOR gates to implement your design, as you see fit.

Use the steps you have learned in class to design your traffic light circuit. Create a truth table for each color of the traffic light. Use outputs from either the counter or decoder as inputs for the truth tables. Think about the operation of a traffic light and think about which of the counter or decoder outputs are similar in timing and logic level to the signal you need for a given traffic light. Using an existing signal that is similar to the signal needed for a given color light means a simpler computation to perform to create the actual signal and, thus, a simpler combinatorial circuit to build.

Use green, RGB, and red LEDs to create green, yellow, and red output. Use the RGB LED to display yellow light (approximately) by turning on both the red and green emitters.

**Breadboard reminders:**

**When a circuit required 5 or more chips** it is wise to position all the chips and any buttons first before starting to place hookup wire. Because each chip and switch pin has 4 tie points connected to it, there is (almost) always a way to make the needed connections even with chips tightly spaced.

Double check that LEDs are placed with correct polarity. **Note, for active low gate outputs, the long LED lead may be connected to power and the short lead used to show a visual inversion of an active low gate output.**

Use one or more 470 ohm resistors in series with each color of the RGB LED dim and balance the red and green emitter intensities to generate a somewhat better yellow color. In fact, dimming the green LED enough to be a good match for the red LED brightness may require using 10,000 ohms in series with the green LED.

Placing a translucent cup over the RGB LED will help mix the red and green light sources together, making for a slightly better yellow experience. Your TA has translucent plastic cups; please return it after use for students in later lab sessions to use.