San Jose State University

Department of Electrical Engineering

EE104, Fall 2022, Pham

Laboratory Assignment #4

# Objectives

For this lab, you will perform the circuit analysis using Python’s Matrix Method & Ahkab Method similar to what we normally do with SPICE, and compare the Python methods with the SPICE analysis. You will also learn how to use Python to control hardware on the PYNQ-Z2 board.

# Grading

Refer to the section **Python Programming** for grading criteria.

# Bibliography

Refer to the lecture notes for Matrix Method & Ahkab Method.

# Download, Installation, and Licensing

You will need to PIP INSTALL the followings: numpy, ahkab.

# Sample Python Program

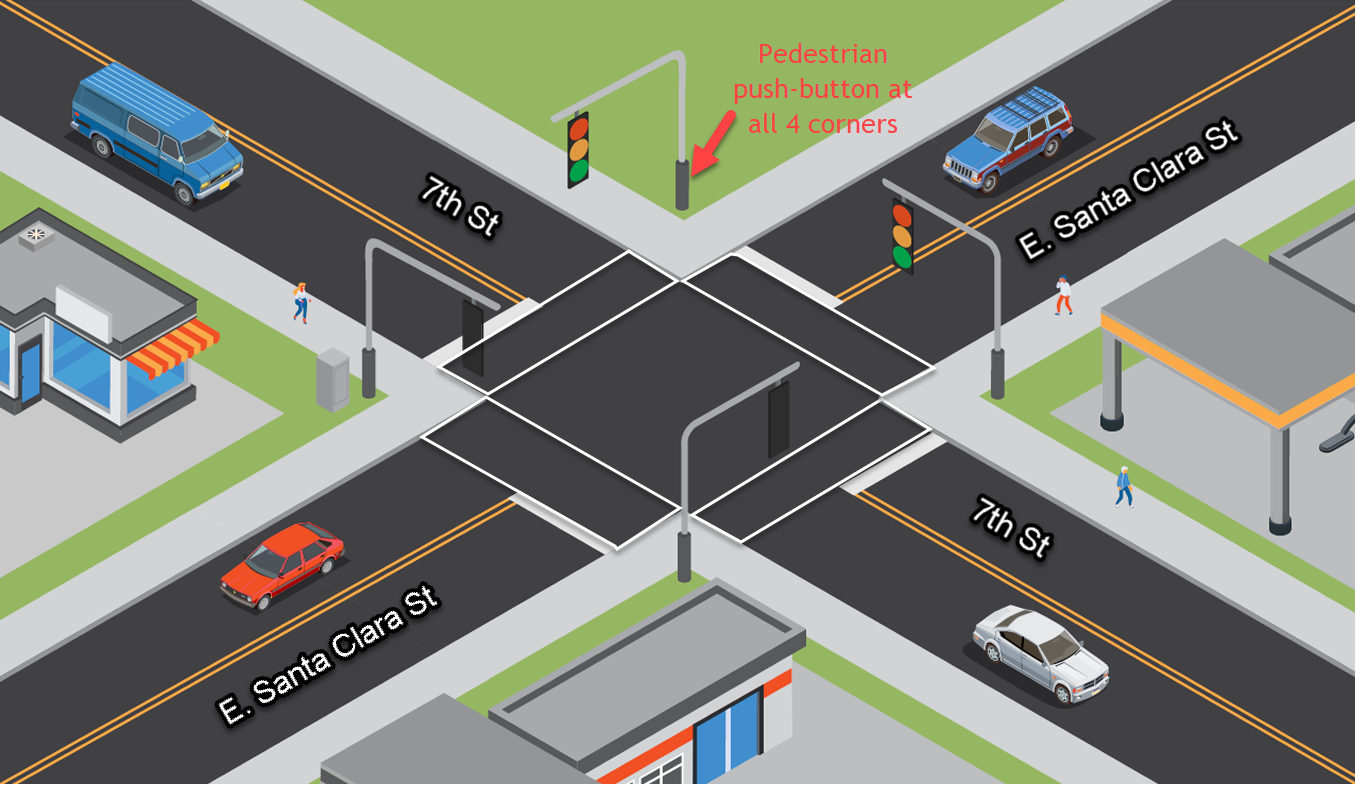
1. AHKAB: Refer to the lecture notes for sample Python codes to perform circuit analysis using Matrix Method & Ahkab Method.
2. Hardware Programming: You will use the board PYNQ-Z2 board.

# Python Programming & Hardware Design

#### Lab Submission

|  |  |  |
| --- | --- | --- |
| **Program or Requirement** | **Use Case** | **Earned Score / Max Score** |
| README file and Developer Note | README: This is a brief user guide so that the user can install the proper python packages and knows how to execute your program. The README file can contain sample screenshots with explanation.  Developer Note: Add to your documentation the discussion about the results similarity or differences from the two methods and why. | \_\_\_\_\_ / 10 |
| Simple OPS analysis | Create an arbitrary IRV circuit with at least 3 loops, then perform and plot a simple OPS analysis using Ahkab package. | \_\_\_\_\_ / 10 |
| Simple AC and TRAN analysis | Using the same (or different) RCL circuit with at least 3 loops, perform and plot a simple AC and TRAN analysis using Ahkab package. | \_\_\_\_\_ / 10 |
| Simple PZ analysis | Using the same (or different) RCL circuit with at least 3 loops, perform and plot a simple PZ analysis using Ahkab package. | \_\_\_\_\_ / 10 |
| Symbolic Analysis – Transfer Function | Use Ahkab to obtain the transfer function | \_\_\_\_\_ / 10 |
| SPICE Simulation | Pick one analysis above and run the same on SPICE. Document the results and compare and provide your explanation WHY they are the same or different. | \_\_\_\_\_ / 10 |
| Simple Traffic Controller:  Part 1, Hardware | Simple Traffic Controller: Part 1, Hardware  See notes below | \_\_\_\_\_ / 20 |
| Simple Traffic Controller:  Part 2, Software Integration | Part 2 – See notes below | \_\_\_\_\_ / 20 |
|  | **TOTAL** | **80% (+20 for Python integration in a future lab)** |

# Simple Traffic Controller: Part 1, Hardware



East Santa Clara Street has the priority, i.e. it always has green light and pedestrian walkways along E. Santa Clara St. is always allowed, except for when there is a request for pedestrian crossing from the 7th Street or a car coming from 7th Street.

You will be given the following hardware to implement your hardware controller portion:

LED For traffic signals.

7-Segment For a count-down pedestrian signal along the 7th Street

Push-button To accept pedestrian’s input request

74LS47 BCD to 7-Segment Decoder

74LS193 Binary Up/Down Counter with Clear

74LS90 Decade Counter

555 Timer For use as clock

Resistors For connecting LEDs, 7-Segments, and other open-collector signals

Capacitors

You will supply your own:

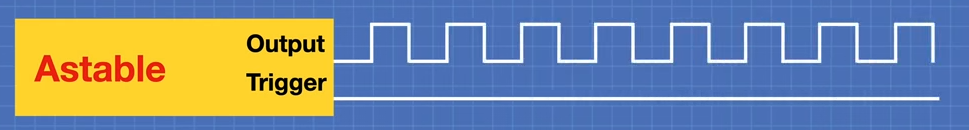
Breadboard You will reuse your own breadboard from your previous courses

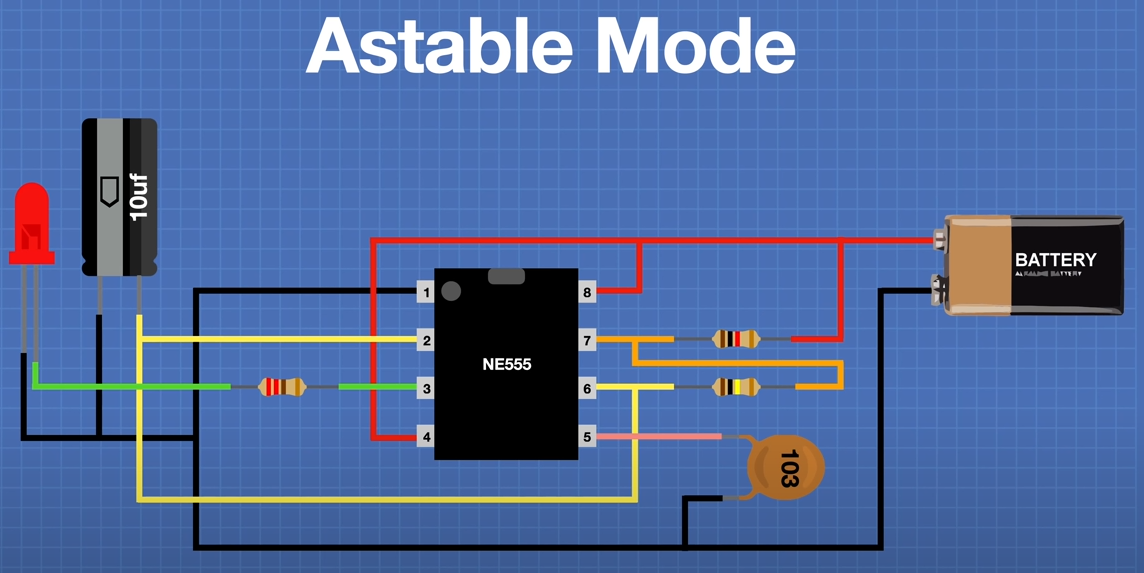
Power/Ground Power/Ground from a battery or a power source. Use 3 x 1.5V = 4.5V

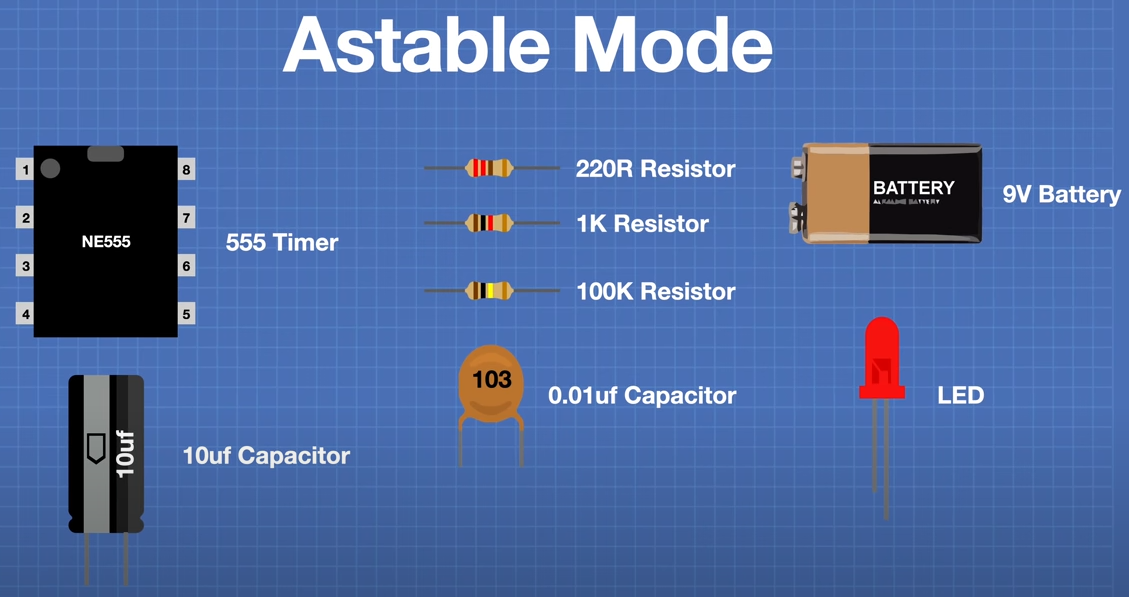
**74-Series datasheet**: <https://www.futurlec.com/IC74LS00Series.shtml>

**Clock generation using 555 chipset using Astable Mode:**

Ref: <https://www.youtube.com/watch?v=ABWU7FlM1T0&ab_channel=DroneBotWorkshop> (time=10:20 )



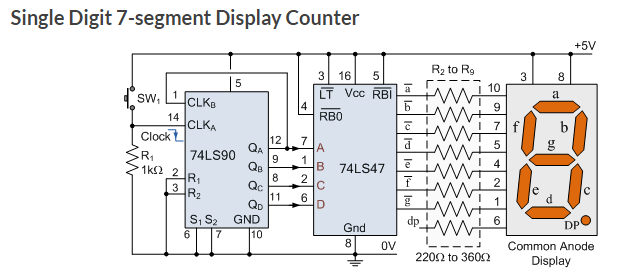
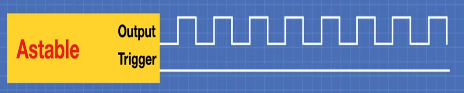


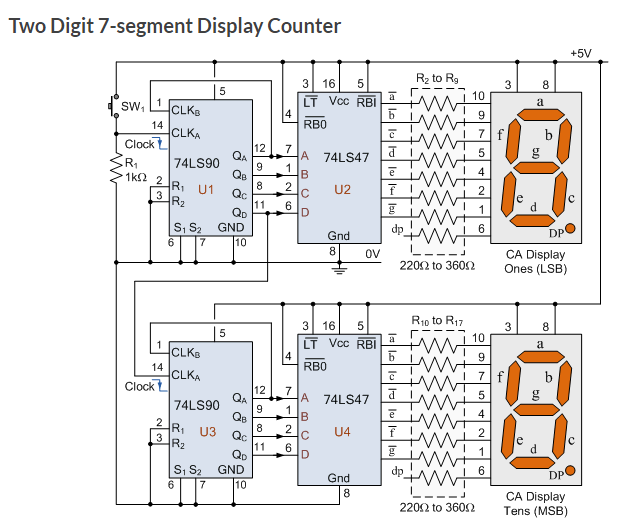
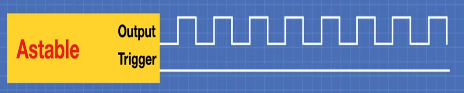


**7-Segment circuit using clock, decade counter, BCD-to-7segment Decoder, and the 7-Segment Display**:

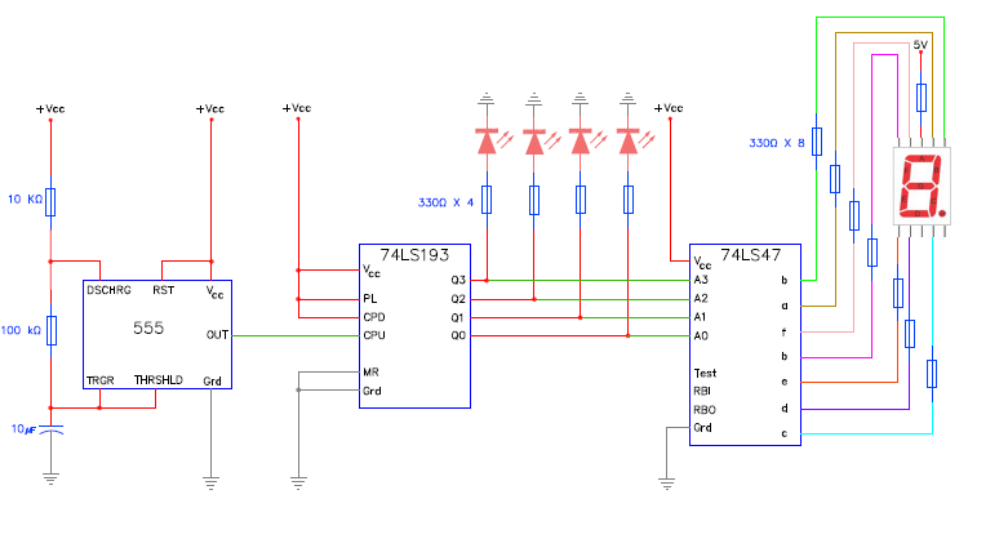
One-digit Ref: <https://www.youtube.com/watch?v=XCJqoae4hgY&ab_channel=element14presents>

One and Two-digit Ref: <https://www.electronics-tutorials.ws/counter/7-segment-display.html>





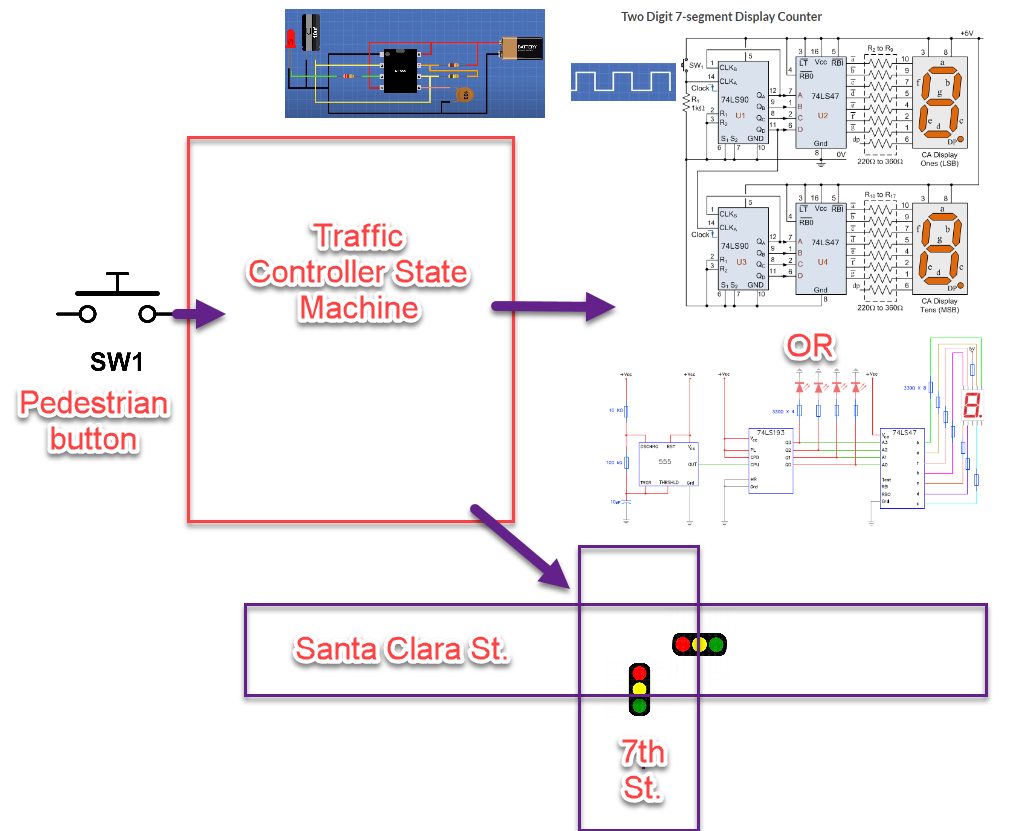
Alternatively, using 74LS193 up/down counter: <https://www.fwdskillzone.com/digital_logic_design.html>



**State Machine Design:**

You can refresh your state machine circuit design using your old notes from previous classes or from this site showing a 4-state Mealy FSM implementation: <https://www.cpsc.ucalgary.ca/custom/321_challenge/04%20Finite%20State%20Machines.html> for from this video clip: <https://www.youtube.com/watch?v=Z4Zz7n-Lj0g>

Note: Your design may be a combination of Mealy & Moore FSM.



When you have the basic hardware working, in the next lab we will write the Python driver to drive the hardware.

That’s all for this lab. Hopefully you found it useful and increase your interest in the Python world! See you in the next lab.

# Simple Traffic Controller: Part 2, Software Integration

Replace any of the following major block of the hardware that you designed and implemented in Part 1 – Hardware with your new Python code. The Python code will interface with your hardware via the KRIA KV260 PMOD port. Below are some scenarios.

SCENARIO ONE: Replace the entire Finite State Machine by software

Graphical user interface

Description automatically generated

SCENARIO Two: Replace the 555-clock circuitry and the Counter by software

You will use the KRIA KV260 to drive the 7-segment display(s).

A picture containing graphical user interface

Description automatically generated

# Laboratory Hand-In Requirements

Once you have completed a working design, prepare for the submission process. You are required to demonstrate a working design. You are also required to submit an archive of your project in the form of a ZIP file. Use 7-Zip option to create the ZIP file. Name the archive lab#\_yourlastname\_yourfirstname.zip. Refer to Lab 1 for detail instructions.

You will submit your zip file to the instructor through Canvas by the due date and time. If the class will be on campus, then you will expect to demonstrate in the classroom. If we ever have to go back to an online mode, turn in your archive to Canvas along with a narrated video capturing the screen of your computer running your program demonstration. If your program is not completely functional by the due date, you should demonstrate and turn in what you have accomplished to receive partial credit. See the syllabus for the late penalty guideline