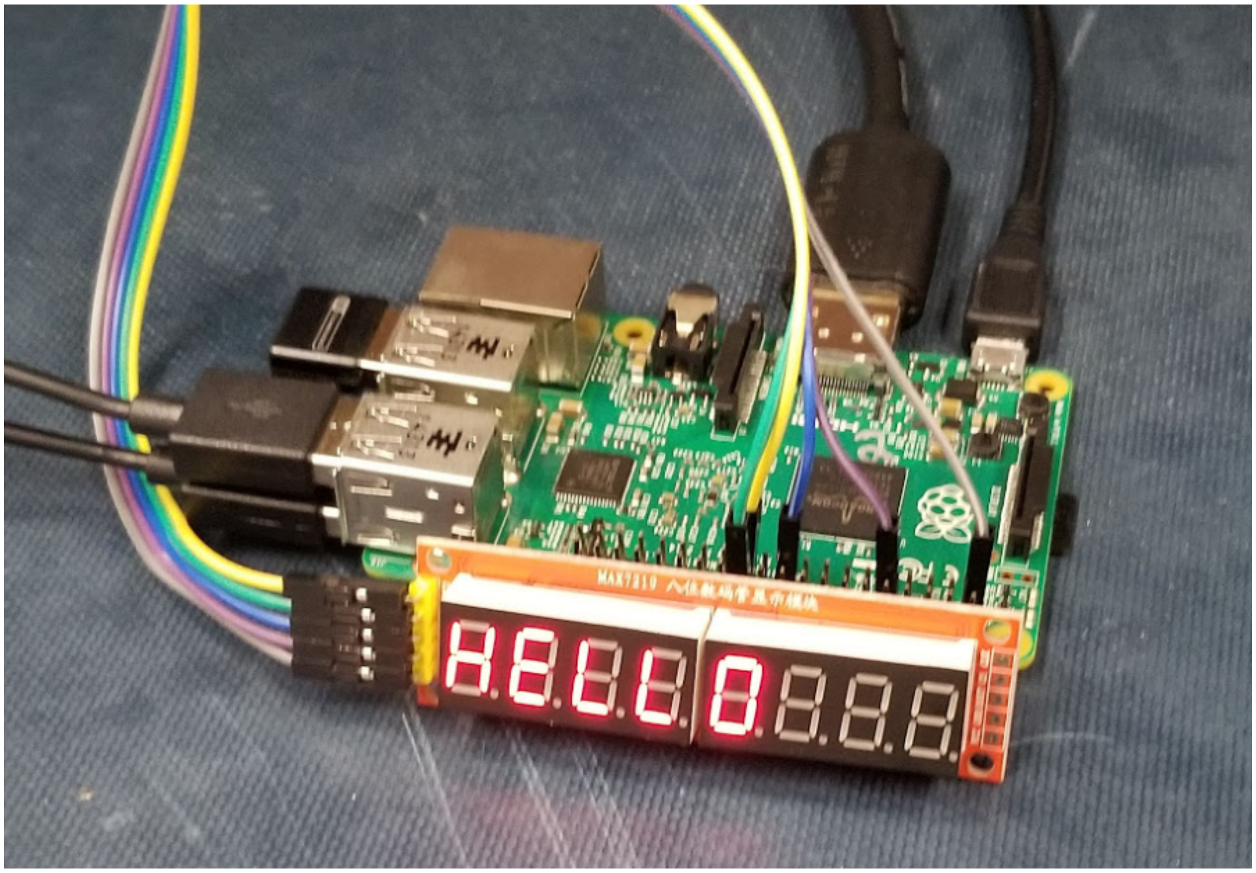
CP320 - Exploration Project

By: Kevin Davis and Ceejay Quach

Introduction



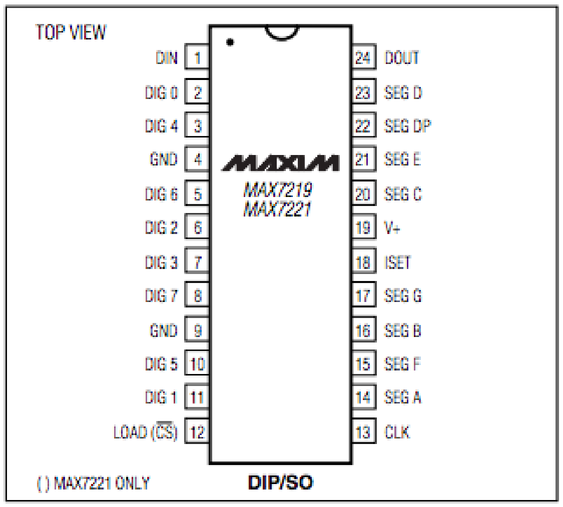
This project explored using the MAX7219 with the Raspberry Pi to output to 2, 4-digit 7 segment displays adjacent to each other. A python library exists online that allows for easy use of the MAX7219. The circuit for the MAX7219 and 7 segment displays can be purchased as a complete package online. The data communication protocol used to control the MAX7219 for this project is SPI.

Documentation

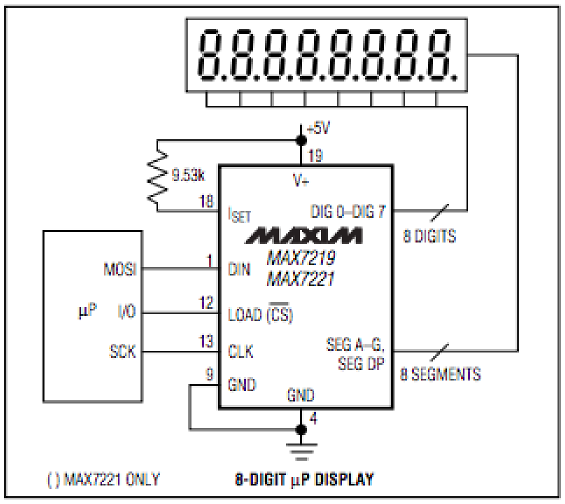
Overview

- The MAX7219 are compact, serial input/output common-cathode display drivers that interface microprocessors (µPs) to 7-segment numeric LED displays of up to 8 digits, bar-graph displays, or 64 individual LEDs. The MAX7219 is compatible with SPI™, and has slew rate-limited segment drivers to reduce EMI. For this project the Raspberry Pi is the microprocessor.

Pin Configuration



Typical Application Circuit



-
- The package library for the MAX7219 includes many useful examples of how to control the device. However, it is time consuming as it simply runs through all the different examples in series taking several minutes to run through the full program. We updated this library to allow for individual testing of each library function. This made it much faster to see the desired example code running.
- We explored further on the MAX7219's ability to manually scroll through text that is displayed. The test function for manual scrolling that came with the library was not clear and did not display the expected results. Through our own experimentation and analysis, we wrote our function to clearly show the manual scrolling function. An important discovery we made is that this function throws away all data that scrolls off of the display. Our code demonstrates this occurring.

Package/Libraries Needed

- https://github.com/rm-hull/max7219/tarball/0.2.3
  - MAX7219-0.2.3.tar.gz

Challenges

- Challenge: Having difficulty installing the library correctly onto the Raspberry Pi
- Solution:
    - Attempted downloading and installing three different libraries
    - Consulted with other teams who were also having trouble installing the MAX7219
    - We initially tried installing the latest version of the library directly from PyPi, but this failed multiple times for us during installation
    - Instead, we manually downloaded the max7219-0.2.3.tar.gz file from https://github.com/rm-hull/max7219/tarball/0.2.3 and copied it onto the Raspberry Pi
    - Then, we unzipped the file, and within that directory, we ran the installation manually
    - To run the installation manually, go into the root directory of the unzipped library and type:
        - *sudo python setup.py install*
    - Note: You can install using Python 3 as well using the following:
        - *sudo python3 setup.py install*
    - However, if you install using Python 3, Python 2.7 will not work and vice versa.

3 Useful URLs

- https://pypi.org/project/max7219/
- Usefulness:
    - Used this website as a step-by-step guide in installing the MAX7219 0.2.3 library

- https://max7219.readthedocs.io/en/0.2.3/
- Usefulness:
    - Used the GPIO pinout diagram to properly connect the MAX7219 to the Raspberry Pi 3

- http://denethor.wlu.ca/pc320/datasheets/MAX7219-MAX7221.pdf
- Usefulness:
    - Used the MAX7219 Datasheet

Youtube Video

- Youtube Link: https://youtu.be/3bL_PyZEmMo

Software Code

```
import time
import RPi.GPIO as GPIO
import spidev
import random
import max7219.led as led
from datetime import datetime

#define constants
SEG_CE = 0 #CE0 is used for 7seg
HEX_START = 0xfa909
HEX_END = 0xfa920

#stepup GPI and SPI
GPIO.setmode(GPIO.BOARD)
GPIO.setup(12, GPIO.OUT)

spi=spidev.SpiDev()
spi.open(0,1) #use CE1 for ADC

#create seven segment device
device = led.sevensegment(cascaded=1)

def date(device, deviceId):
    now   = datetime.now()
    day   = now.day
    month = now.month
    year  = now.year - 2000

    # Set day
    device.letter(deviceId, 8, int(day / 10))     # Tens
    device.letter(deviceId, 7, day % 10)          # Ones
    device.letter(deviceId, 6, '-')               # dash
    # Set day
    device.letter(deviceId, 5, int(month / 10))   # Tens
    device.letter(deviceId, 4, month % 10)        # Ones
    device.letter(deviceId, 3, '-')               # dash
    # Set day
    device.letter(deviceId, 2, int(year / 10))    # Tens
    device.letter(deviceId, 1, year % 10)         # Ones
```

```
def clock(device, deviceId, seconds):
    """
    Display current time on device.
    """
    for _ in range(seconds):
        now = datetime.now()
        hour = now.hour
        minute = now.minute
        second = now.second
        dot = second % 2 == 0                # calculate blinking dot
        # Set hours
        device.letter(deviceId, 4, int(hour / 10))     # Tens
        device.letter(deviceId, 3, hour % 10, dot)     # Ones
        # Set minutes
        device.letter(deviceId, 2, int(minute / 10))   # Tens
        device.letter(deviceId, 1, minute % 10)        # Ones
        time.sleep(1)


#main
while True:
      print("Select function")
      print("1 - Write test")
      print("2 - Clear display test")
      print("3 - Simple text... test")
      print("4 - Scrolling Alphabet Text... test")
      print("5 - Display time test")
      print("6 - Brightness test")
      print("7 - Scrolling function test")
      print("8 - Negative numbers test")
      print("9 - Hex numbers test")
      print("10 - Random numbers test")
      print("Else - quit")

      strvalue=input("choose a test (0...10, else=quit):")

      if strvalue==1:
            #write dist to 7seg display
            device.write_text(SEG_CE, "HELLO")
            time.sleep(1)

      elif strvalue==2:
            device.clear()

      elif strvalue==3:
            print('Simple text...')
            for _ in range(3):
                  device.write_text(SEG_CE, "HELLO")
                  time.sleep(0.6)
                  device.write_text(SEG_CE, " GOODBYE")
                  time.sleep(0.6)

      elif strvalue==4:
            # Scrolling Alphabet Text
            print('Scrolling alphabet text...')
            device.show_message("abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ")

      elif strvalue==5:
            # display time
            date(device, 0)
            time.sleep(5)
            clock(device, 0, seconds=5)

      elif strvalue==6:
            # Brightness
            print('Brightness...')
            device.write_text(SEG_CE, "BRIGHT")
            for x in range(5):
                  for intensity in range(16):
                        device.brightness(intensity)
                        time.sleep(0.1)
                        device.brightness(7)

      elif strvalue==7:
            # Scrolling
            print('Scrolling...left, left, right')
            device.show_message("SCROLL LEFT LEFT RIGHT")
            device.write_text(0, "12345678")
            for _ in range(8):
                  device.scroll_left()
                  time.sleep(.25)
                  device.scroll_left()
                  time.sleep(.25)
                  device.scroll_right()
                  time.sleep(1)

      elif strvalue==8:
            # Negative numbers
            print('Negative numbers...')
            for x in range(-30, 30):
                  device.write_number(SEG_CE, value=x)
                  time.sleep(0.05)

      elif strvalue==9:
            # Hex numbers
            print('Hex numbers...')
            for x in range(HEX_START, HEX_END):
                  device.write_number(SEG_CE, value=x, base=16,
leftJustify=True)
                  time.sleep(0.025)

      elif strvalue==10:
            print('Random numbers...')
            a = random.randint(-999, 999)
            # b = random.randint(-3223, 999)

            device.brightness(3)
            for x in range(5):
                  a += random.random() * 10
                  device.write_number(SEG_CE, value=a, decimalPlaces=1)
                  time.sleep(0.5)

      else:
            device.write_text(SEG_CE, "goodbye")
            break


#cleanup
GPIO.cleanup()
```