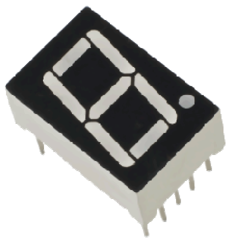
Project 10 - Using the Raspberry Pi to write digits on a 7-segment LED display

Outline

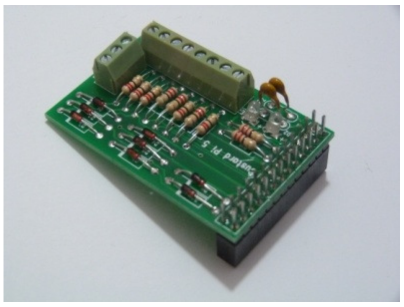
This application note shows the hardware and Python code required to display single-digit numbers on a 7-segment LED display.

Hardware

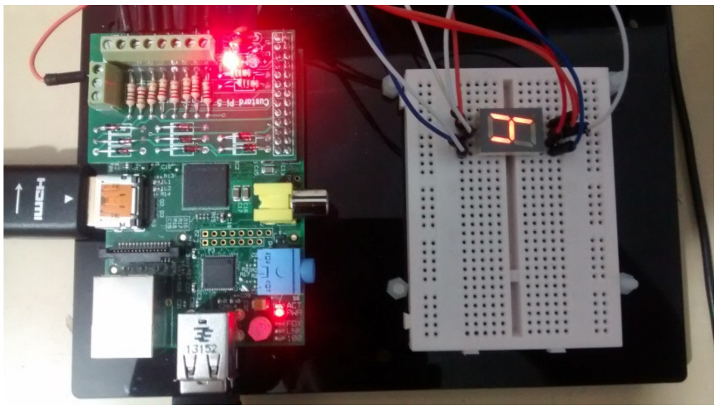
7-segment display: The 7 segments of the display are used to show digits 0 to 9 by lighting up the different LEDs that make it up. The decimal point is the 8th segment and is not used here.



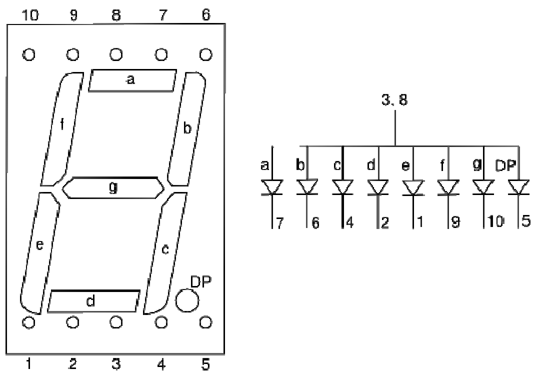
Custard Pi 5: The Raspberry Pi GPIO is not easy to connect to and can be damaged if the wrong voltage is connected to certain pins. We are going to use the Custard Pi 5 break out board as this has some protection built in to prevent accidental damage. It also has current limiting resistors built in, making it easy to assemble this project.



For ease of assembly, this project uses a Custard Pi base and a prototyping board. The assembled hardware is shown below.



Connections on the 7-segment display

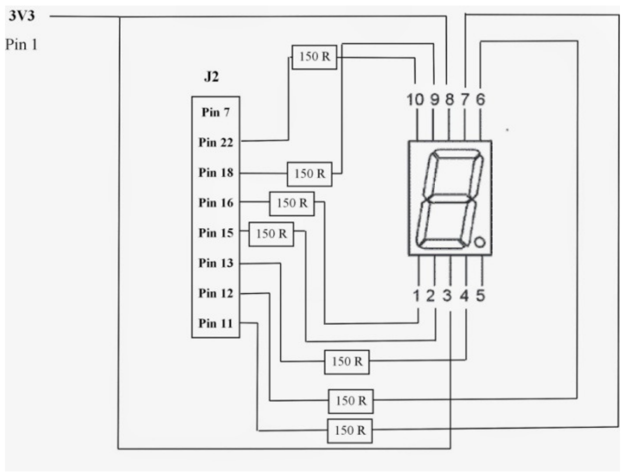


Pins 3 and 8 are connected to a voltage supply. In this application we are going to use the 3.3V from the Raspberry Pi. When any of the other pins are taken to 0V (using a current limiting resistor) the appropriate LED segment will light up. For example, when pins 4 and 6 are taken to 0V segments b and c light up to show digit ‘1’. To display digit ‘3’ segments a, b, g, c and d need to light up. This is done by taking pins 7, 6, 10, 4 and 2 down to 0V.

The connections from the Custard Pi 5 board to the 7-segment display are shown below.

GPIO ref number	Pin on Custard Pi	Pin on 7 segment display
GPIO 25	J2 pin 22	10
GPIO 24	J2 pin 18	9
GPIO 23	J2 pin 16	1
GPIO 22	J2 pin 15	2
GPIO 27	J2 pin 13	4
GPIO 18	J2 pin 12	6
GPIO 17	J2 pin 11	7
3.3V	J3 pin 3.3V	3 and 8

The drawing below shows how the 7 segment display is connected up. Note that there is no need to use the 150 ohm resistors as the Custard Pi 5 board already has 220 ohm current limiting resistors built in.



Software

The Python code presented here writes digits 0 to 9 to the 7 segment display with a pause of 1 second between digits.

```
#!/usr/bin/env python
#Python code to drive a 7-segment display
#www.sf-innovations.co.uk
import RPi.GPIO as GPIO
import time
import os, sys

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)

#setup output pins
GPIO.setup(11, GPIO.OUT)
GPIO.setup(12, GPIO.OUT)
GPIO.setup(13, GPIO.OUT)
GPIO.setup(15, GPIO.OUT)
GPIO.setup(16, GPIO.OUT)
GPIO.setup(18, GPIO.OUT)
GPIO.setup(22, GPIO.OUT)

#define 7 segment digits
digitclr=[1,1,1,1,1,1,1]
digit0=[0,0,0,0,0,0,1]
digit1=[1,0,0,1,1,1,1]
digit2=[0,0,1,0,0,1,0]
digit3=[0,0,0,0,1,1,0]
digit4=[1,0,0,1,1,0,0]
digit5=[0,1,0,0,1,0,0]
digit6=[0,1,0,0,0,0,0]
digit7=[0,0,0,1,1,1,1]
digit8=[0,0,0,0,0,0,0]
digit9=[0,0,0,1,1,0,0]

gpin=[11,12,13,15,16,18,22]

#routine to clear and then write to display
def digdisp(digit):
    for x in range (0,7):
        GPIO.output(gpin[x], digitclr[x])
    for x in range (0,7):
        GPIO.output(gpin[x], digit[x])

#routine to display digit from 0 to 9
digdisp(digit0)
time.sleep(1)
digdisp(digit1)
time.sleep(1)
digdisp(digit2)
time.sleep(1)
digdisp(digit3)
time.sleep(1)
digdisp(digit4)
time.sleep(1)
digdisp(digit5)
time.sleep(1)
digdisp(digit6)
time.sleep(1)
digdisp(digit7)
time.sleep(1)
digdisp(digit8)
time.sleep(1)
digdisp(digit9)
time.sleep(1)

#tidy up
GPIO.cleanup()
import sys
sys.exit()
```

(Download this code from here: http://www.sf-innovations.co.uk/downloads)

Notes:
The Custard Pi 5 and Custard Base are available from amazon.co.uk and directly from SF Innovations. The code presented here can be downloaded from the SF Innovations website.

Ideas for Advanced projects

See how many letters of the alphabet can be displayed on a 7-segment display either as a capital or in lower case.

Use a Custard Pi 1 (which has more digital outputs) to interface to a ‘star burst’ display which allows letters and numbers to be displayed. The starburst display shown below has 15 LED segments plus a decimal point.