Internet of Things Laboratory

Assignment 5

Prajakta Deokule

C22019221332

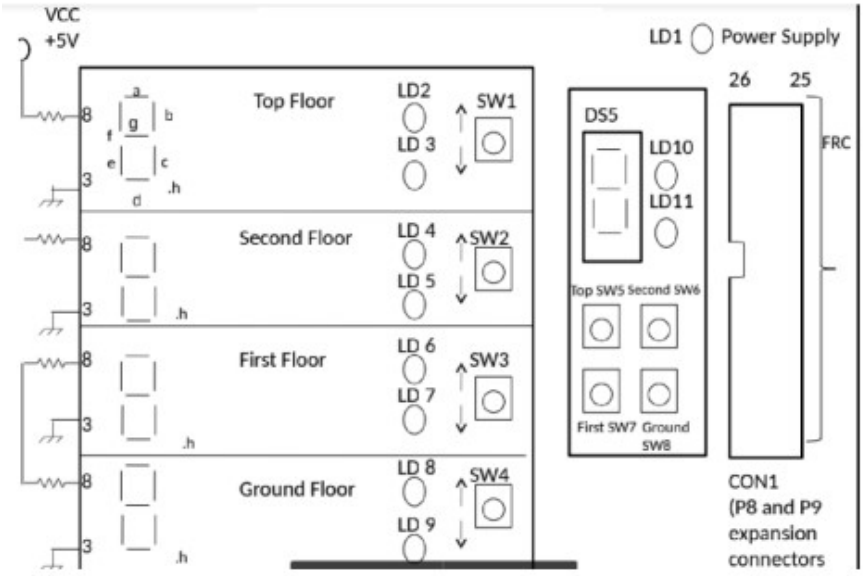
4329

Problem Statement:

Write an application using Beagle Board to control the operation of a hardware simulated lift elevator.

1. **Draw layout of the elevator kit.**

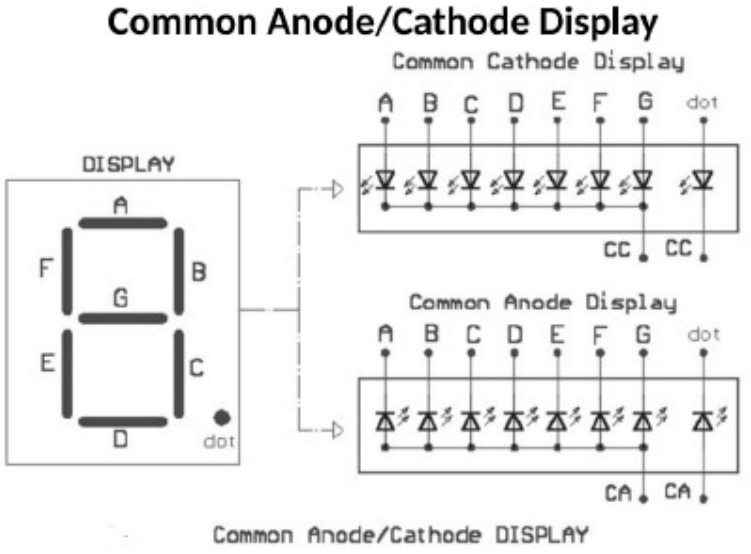
Elevator Study Card



2. **Explain the LED types with diagram (Common anode and Common cathode)**
   A 7-segment is a packaged set of 8 LEDs (7 number-segments & 1 decimal point).
   a) Common Anode: Common anode means that the anode (positive) side of all the LEDs are electrically connected at one pin and each LED

cathode has its own pin. So, turning on any segment will involve running the current from the common anode (positive) pin to the cathode (negative) pin of the desired segment. The common anode pin is connected to 5V supply. Such types of LEDs glow when we set their output to high.

b) Common Cathode: Common cathode means that the cathodes of all the LEDs are common and connected to a single pin. The anode for each LED has its own pin. Driving one of these means running the current from the anode (positive) pin of the desired segment to the common cathode pin. The common anode pin is connected to ground pin. Such type of LEDs glows when we set their output to low.



3. **Give details about the P8 and P9 connectors used for the assignment with differentiation for switches, LEDs and 7 segment display.**

● **Switches:**

→ Outside the lift:

- (Top floor) SW1 - P8_7

- (Second floor) SW2 - P8_9

- (First floor) SW3 - P8_8 -(Ground floor) SW4 - P8_10

→ **Inside the lift:**

- (Top floor) SW5 - P8_7
- (Second floor) SW6 - P8_8
- (First floor) SW7 - P8_8
- (Ground floor) SW8 - P8_10

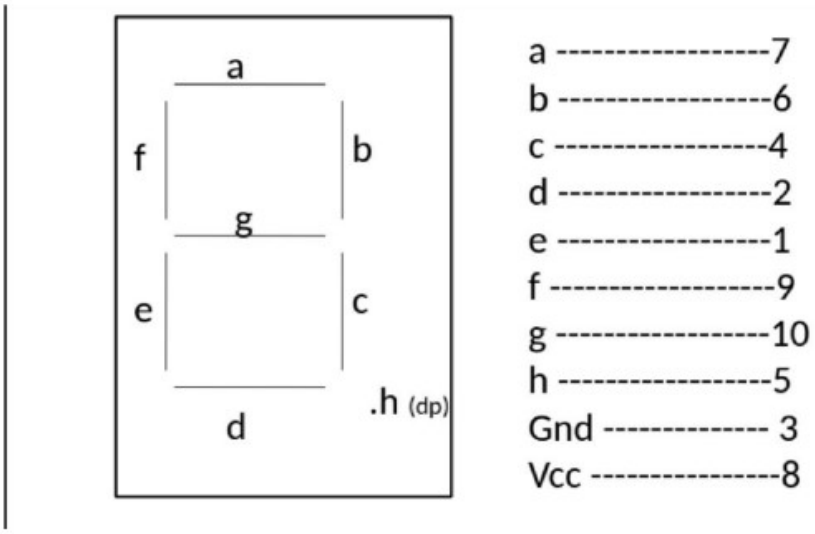
● **LEDs**

- LED1 - P9_11, LED2 - P9_12, LED3 - P9_13, LED4 - P9_14, LED5 - P9_15, LED6 - P9_16, LED7 - P9_24, LED8 - P9_23,

● **7-Segment**

- a - P8_11
- b - P8_12
- c - P8_12
- d - P8_14
- e - P8_15
- f - P8_16
- g - P8_17
- h - P8_18

4. Draw a neat and clean diagram of a 7-segment display with pin numbers.



Code

```
import Adafruit_BBIO.GPIO as GPIO
import time

led_pins = ['P9_23', 'P9_24', 'P9_11', 'P9_12', 'P9_13', 'P9_14','P9_15','P9_16']
seg = ['P8_11', 'P8_12', 'P8_13', 'P8_14', 'P8_15', 'P8_16','P8_17','P8_18']
switch = ['P8_7', 'P8_8', 'P8_9','P8_10']

zero = ['P8_11', 'P8_12', 'P8_13', 'P8_14', 'P8_15', 'P8_16']
one = ['P8_12', 'P8_13']
two = ['P8_11', 'P8_12', 'P8_14','P8_15', 'P8_17']
three = ['P8_11', 'P8_12', 'P8_13','P8_14', 'P8_17']

for i in range(len(led_pins)):
        GPIO.setup(led_pins[i], GPIO.OUT)
        GPIO.setup(seg[i], GPIO.OUT)

        for j in range(len(switch)):
                GPIO.setup(switch[j], GPIO.IN)

def led_clear():
        for i in range(len(led_pins)):
                GPIO.output(led_pins[i], GPIO.LOW)

def seg_clear():
        for i in range(len(seg)):
                GPIO.output(seg[i], GPIO.HIGH)

def seg_disp(b):
        if b==0:
                seg_clear()
                for i in range(len(zero)):
                        GPIO.output(zero[i], GPIO.LOW)

        if b==1:
                seg_clear()
                for j inrange(len(one)):
                        GPIO.output(one[j], GPIO.LOW)

        if b==2:
                seg_clear()
                for k in range(len(two)):
                        GPIO.output(two[k], GPIO.LOW)

        if b==3:
                seg_clear()
                for l in range(len(three)):
                        GPIO.output(three[l], GPIO.LOW)
```

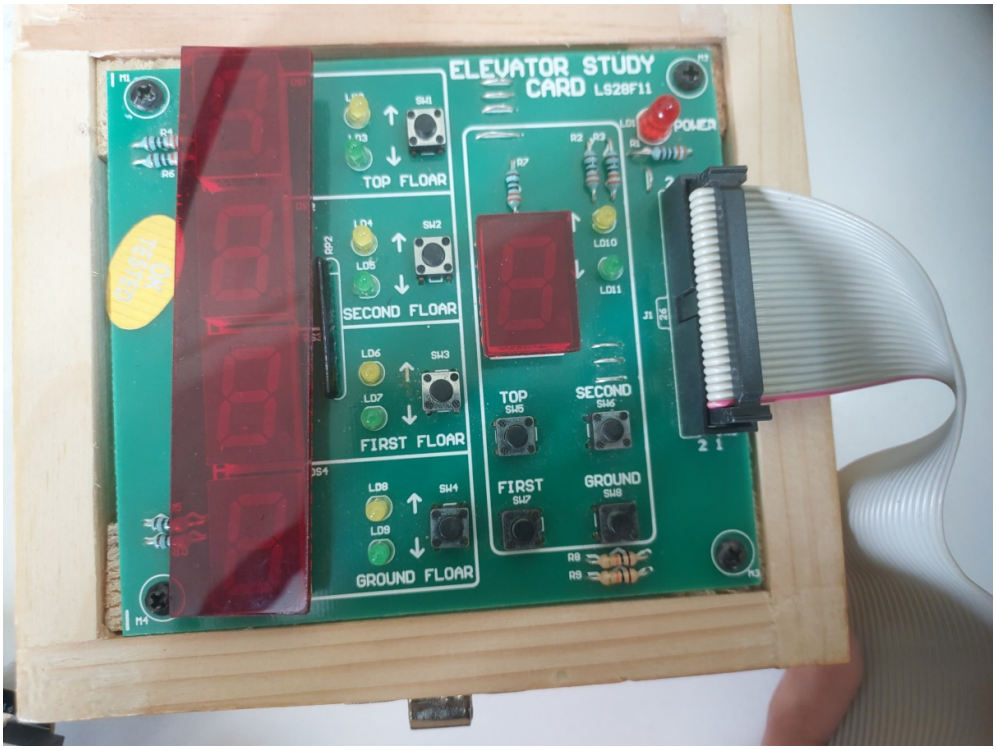
```
old_state = 0
new_state = 0

while True:
        while True:
                if(GPIO.input("P8_10")==0):
                        print("0 pressed")
                        new_state = 0
                        break

                if(GPIO.input("P8_8")==0):
                        print("1pressed")
                        new_state = 1
                        break

                if(GPIO.input("P8_9")==0):
                        print("2 pressed")
                        new_state = 2
                        break

                if(GPIO.input("P8_7")==0):
                        print("3 pressed")
                        new_state = 3
                        break

                if(old_state == 0 and new_state ==0):
                        led_clear()
                        seg_disp(0)
                        GPIO.output("P9_24",GPIO.LOW)
                        GPIO.output("P9_23",GPIO.HIGH)
                        time.sleep(1)
                        old_state = 0
                        #(0-0)
                if(old_state == 0 and new_state ==1):
                         led_clear()
                        seg_disp(0)
                        GPIO.output("P9_24",GPIO.HIGH)
                        GPIO.output("P9_24",GPIO.LOW)
                        time.sleep(1)
                        seg_disp(1)
                        GPIO.output("P9_15",GPIO.HIGH)
                        GPIO.output("P9_16",GPIO.HIGH)
                        old_state = 1
                        #(0-1)

                if(old_state == 0 and new_state == 2):
                        led_clear()
                        seg_disp(0)
                        GPIO.output("P9_24",GPIO.HIGH)
```

```
            time.sleep(1)
            #(0-2)
            GPIO.output("P9_24",GPIO.LOW)
            seg_disp(1)
            GPIO.output("P9_15",GPIO.HIGH)
            time.sleep(1)
            GPIO.output("P9_15",GPIO.LOW)
            seg_disp(2)
            GPIO.output("P9_13",GPIO.HIGH)
            GPIO.output("P9_14",GPIO.HIGH)
            old_state = 2

    if(old_state == 0 and new_state ==3):
             led_clear()
             seg_disp(0)
            GPIO.output("P9_24",GPIO.HIGH)
            time.sleep(1)
            GPIO.output("P9_24",GPIO.LOW)
            seg_disp(1)
            GPIO.output("P9_15",GPIO.HIGH)
            time.sleep(1)
            GPIO.output("P9_15",GPIO.LOW)
            seg_disp(2)
            GPIO.output("P9_13",GPIO.HIGH)
            time.sleep(1)
            GPIO.output("P9_13",GPIO.LOW)
            seg_disp(3)
            GPIO.output("P9_11",GPIO.HIGH)
            GPIO.output("P9_12",GPIO.HIGH)
            old_state = 3
            #(0-3)

    if(old_state == 1 and new_state ==1):
             led_clear()
            seg_disp(1)
            GPIO.output("P9_15",GPIO.HIGH)
            GPIO.output("P9_16",GPIO.HIGH)
            time.sleep(1)
            old_state = 1
            #(1-1)

    if(old_state == 1 and new_state == 2):
            led_clear()
            seg_disp(1)
            GPIO.output("P9_15",GPIO.HIGH)
            GPIO.output("P9_15",GPIO.LOW)
            time.sleep(1)
            seg_disp(2)
            GPIO.output("P9_13",GPIO.HIGH)
            GPIO.output("P9_14",GPIO.HIGH)
```

```
            old_state = 2
            #(1-2)

    if(old_state == 1 and new_state == 0):
             led_clear()
            seg_disp(1)
            GPIO.output("P9_16",GPIO.HIGH)
            GPIO.output("P9_16",GPIO.LOW)
            time.sleep(1)
            seg_disp(0)
            GPIO.output("P9_23",GPIO.HIGH)
            GPIO.output("P9_24",GPIO.HIGH)
            old_state = 0
            #(1-0)

    if(old_state == 1 and new_state == 3):
             led_clear()
            seg_disp(1)
            GPIO.output("P9_15",GPIO.HIGH)
            time.sleep(1)
            GPIO.output("P9_15",GPIO.LOW)
            seg_disp(2)
            GPIO.output("P9_13",GPIO.HIGH)
            time.sleep(1)
            GPIO.output("P9_13",GPIO.LOW)
            seg_disp(3)
            GPIO.output("P9_11",GPIO.HIGH)
            GPIO.output("P9_12",GPIO.HIGH)
            old_state = 3
            #(1-3)

    if(old_state == 2 and new_state ==2):
             led_clear()
            seg_disp(2)
            GPIO.output("P9_13",GPIO.HIGH)
            GPIO.output("P9_14",GPIO.HIGH)
            time.sleep(1)
            old_state = 2
            #(2-2)

    if(old_state == 2 and new_state == 1):
             led_clear()
            seg_disp(2)
            GPIO.output("P9_14",GPIO.HIGH)
            GPIO.output("P9_14",GPIO.LOW)
            time.sleep(1)
            seg_disp(1)
            GPIO.output("P9_16",GPIO.HIGH)
            GPIO.output("P9_15",GPIO.HIGH)
            old_state = 1
```

```
        #(2-1)

if(old_state == 2 and new_state == 0):
        led_clear()
        seg_disp(2)
        GPIO.output("P9_14",GPIO.HIGH)
        #(2-0)
        time.sleep(1)
        GPIO.output("P9_14",GPIO.LOW)
        seg_disp(1)
        GPIO.output("P9_16",GPIO.HIGH)
        time.sleep(1)
        GPIO.output("P9_16",GPIO.LOW)
        seg_disp(0)
        GPIO.output("P9_23",GPIO.HIGH)
        GPIO.output("P9_24",GPIO.HIGH)
        old_state = 0

if(old_state == 2 and new_state == 3):
        led_clear()
         seg_disp(2)
        GPIO.output("P9_13",GPIO.HIGH)
        GPIO.output("P9_13",GPIO.LOW)
        time.sleep(1)
        seg_disp(3)
        GPIO.output("P9_11",GPIO.HIGH)
        GPIO.output("P9_12",GPIO.HIGH)
        old_state = 3
        #(2-3)

if(old_state == 3 and new_state == 3):
        led_clear()
        seg_disp(3)
        GPIO.output("P9_11",GPIO.HIGH)
        GPIO.output("P9_12",GPIO.LOW)
        time.sleep(1)
        old_state = 3
        #(3-3)

if(old_state == 3 and new_state == 2):
        led_clear()
        seg_disp(3)
        GPIO.output("P9_12",GPIO.HIGH)
        GPIO.output("P9_12",GPIO.LOW)
        time.sleep(1)
        seg_disp(2)
        GPIO.output("P9_13",GPIO.HIGH)
        GPIO.output("P9_14",GPIO.HIGH)
        old_state = 2
        #(3-2)
```

```
if(old_state == 3 and new_state == 1):
        led_clear()
        seg_disp(3)
        GPIO.output("P9_12",GPIO.HIGH)
         time.sleep(1)
        GPIO.output("P9_12",GPIO.LOW)
        seg_disp(2)
        #(3-1)
        GPIO.output("P9_14",GPIO.HIGH)
         time.sleep(1)
        GPIO.output("P9_14",GPIO.LOW)
        seg_disp(1)
        GPIO.output("P9_15",GPIO.HIGH)
        GPIO.output("P9_16",GPIO.HIGH)
        old_state = 1

if(old_state == 3 and new_state == 0):
        led_clear()
        seg_disp(3)
        GPIO.output("P9_12",GPIO.HIGH)
        time.sleep(1)
        GPIO.output("P9_12",GPIO.LOW)
        seg_disp(2)
        GPIO.output("P9_14",GPIO.HIGH)
        time.sleep(1)
        GPIO.output("P9_14",GPIO.LOW)
        seg_disp(1)
        GPIO.output("P9_16",GPIO.HIGH)
        time.sleep(1)
        GPIO.output("P9_16",GPIO.LOW)
        seg_disp(0)
        GPIO.output("P9_23",GPIO.HIGH)
        GPIO.output("P9_24",GPIO.HIGH)
        old_state = 0
```


ELEVATOR STUDY
CARD LS28F11
POWER
TOP FLOAR
SECOND FLOAR
FIRST FLOAR
GROUND FLOAR
TOP
SECOND
FIRST
GROUND
TESTED