MKSSS's
CUMMINS COLLEGE OF ENGINEERING FOR WOMEN
DEPARTMENT OF COMPUTER ENGINEERING

Shreya Pawaskar
Cno - C22018881961
Roll no - 4947
Division - C

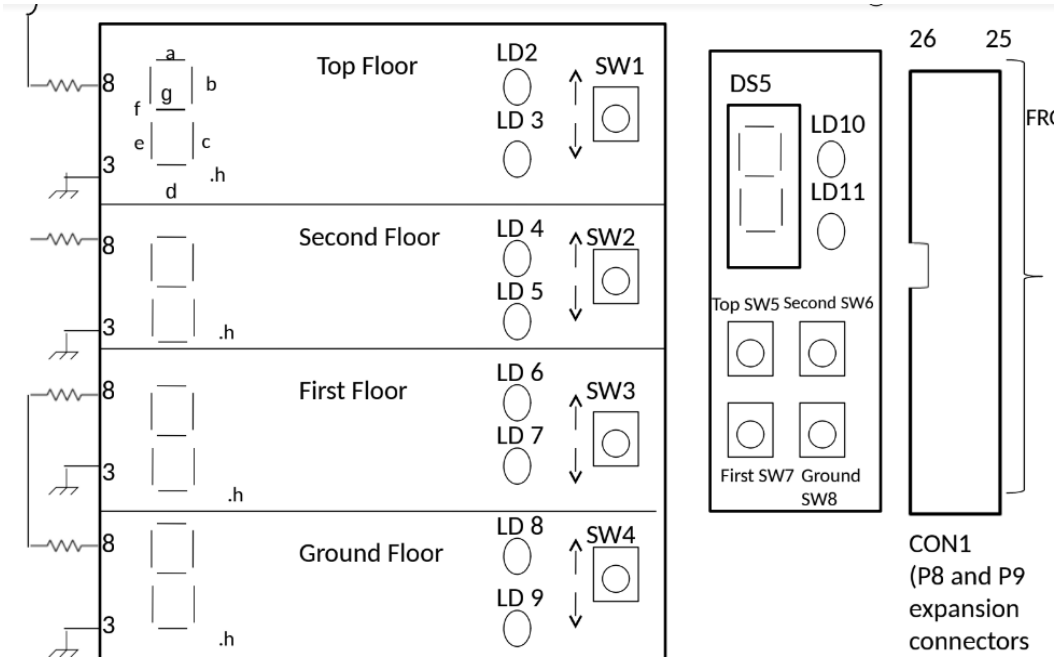
Internet of Things Laboratory

Assignment 5

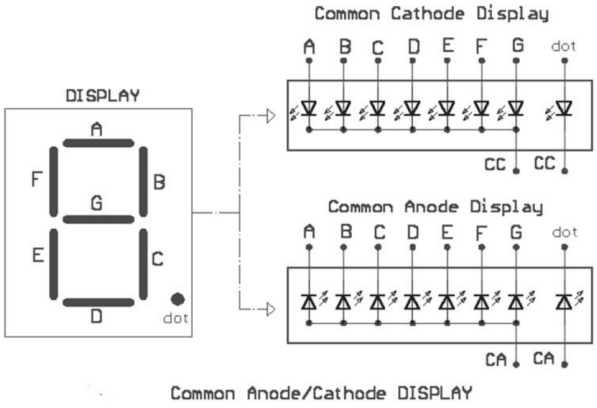
Problem Statement: Write an application using a Beagle bone board to control the operation of a hardware simulated Lift.

Theory:

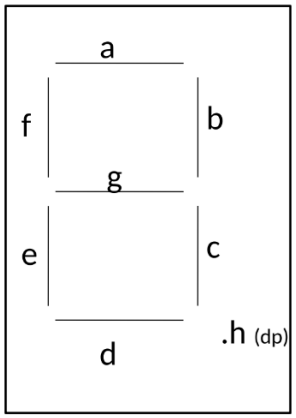
1. Elevator Study Card



Common Anode/Cathode Display



LED Display

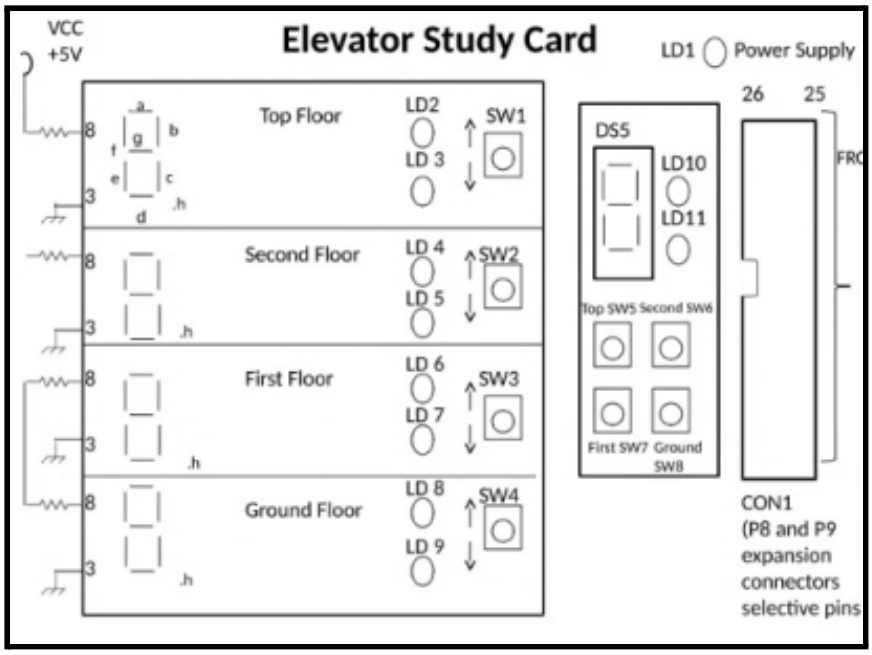


a ----------7
b ----------6
c ----------4
d ----------2
e ----------1
f ----------9
g ----------10
h ----------5
Gnd ---------- 3
Vcc ----------8

Mapping of P8-P9 pins

Top floor	SW1 - P8_7	LED 1 – P9_11	a – P8_11
Second floor	SW2 - P8_9	LED 2 – P9_12	b – P8_12
First floor	SW3 - P8_8	LED 3 – P9_13	c – P8_13
Ground floor	SW4 - P8_10	LED 4 – P9_14	d – P8_14
		LED 5 – P9_15	e – P8_15
Top Floor	SW5 – P8_7	LED 6 – P9_16	f – P8_16
Second Floor	SW6 – P8_9	LED 7 – P9_24	g – P8_17
First Floor	SW7 – P8_8	LED 8 – P9_23	h – P8_18
Ground Floor	SW8- P8_10		

1. **The layout of the Kit.**



---

2. **Explain the LED type with a diagram (Common cathode or common Anode).**

Common Anode Display:

In a common anode display, all the anode connections of the LED segments are joined together to logic “1”. The individual segments are illuminated by applying a ground logic “0” or low. Common anodes are more generally used because the logic circuits that are used to switch the elements on or off are usually better at sinking current.

Common Cathode Display:

In a common cathode display, all the cathode connections of the LED segments are joined together to logic “0” or ground. The individual segments are illuminated by the application of the logic “1” or high via a current limiting resistor to forward bias the individual Anode terminals.

---

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

Switches are used as lift buttons.

Here 2 sets of 4 switches are used and they are as follows:

Set 1:

Top floor switch_1: P8_7

Top floor switch_2: P8_9

Top floor switch_3: P8_8

Top floor switch_4: P8_10

Set 2:

Top floor switch_5: P8_7

Top floor switch_6: P8_9

Top floor switch_7: P8_8

Top floor switch_8: P8_10

LEDs are used as the output for the elevator.

Each LED is connected to a register of appropriate resistance.

Total 8 LEDs are utilized for each floor.

LED_1 : P9_11

LED_2 : P9_12

LED_3 : P9_13

LED_4 : P9_14

LED_5 : P9_15

LED_6 : P9_16

LED_7 : P9_24

LED_8 : P9_23

7 segment display.

a: P8_11

b: P8_12

c: P8_13

d: P8_14

e: P8_15

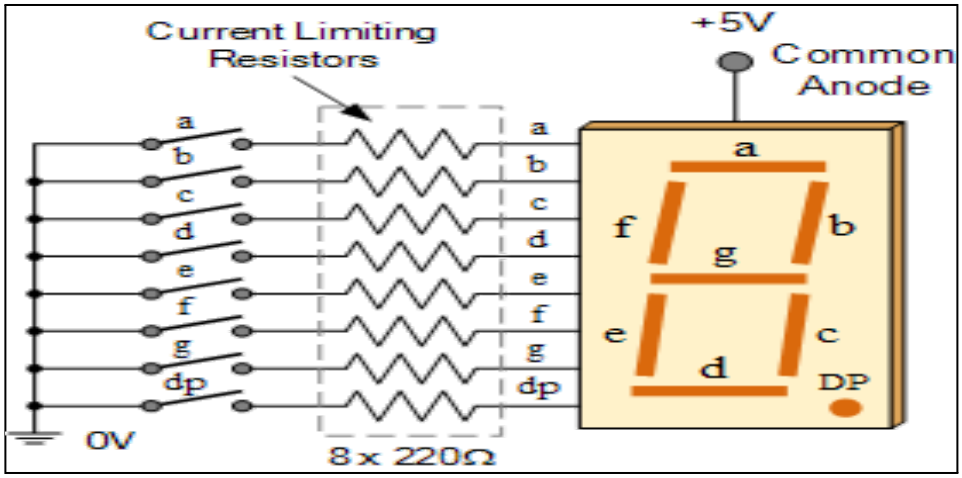
f: P8_16

g: P8_17

h: P8_18

---

4. **Draw a neat and clean diagram for a 7 segment display with PINs.**



---

5. **Program should be handwritten in write up.**

```
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):
                  new_state = 1 print("1
                  pressed")
```

```
            break
        if(GPIO.input("P8_9")==0):
            new_state = 2 print("2
            pressed")
            break
        if(GPIO.input("P8_7")==0):
            new_state = 3 print("3
            pressed")
            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
```