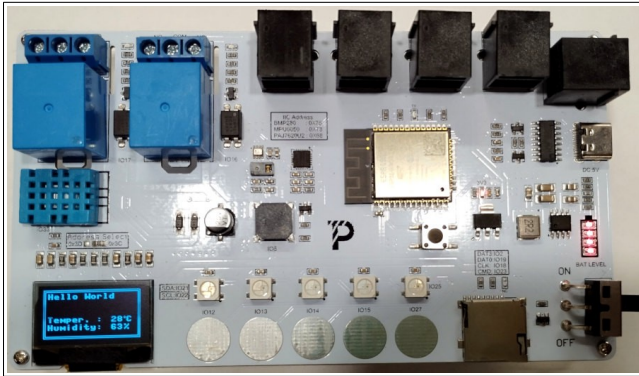


## Άσκηση 1

Στην πλακέτα ARD:icon II (IOT) του κιτ ρομποτικής S4T της Polytech, να γίνει πρόγραμμα σε Micropython το οποίο εμφανίζει την θερμοκρασία και την σχετική υγρασία από τον αισθητήρα DHT 11 στην ενσωματωμένη οθόνη OLED του συστήματος. Οι τιμές θα ανανεώνονται κάθε 2 δευτερόλεπτα.



Μέσα στο σύστημα αρχείων πρέπει να αντιγραφτεί η βιβλιοθήκη `ssd1306.py`

```
1 #Πρόγραμμα το οποίο παρουσιάζει μετρήσεις Θερμοκρασίας και υγρασίας από DHT11 στην οθόνη OLED
2 import machine, time, dht
3 i2c = machine.I2C(0, sda = machine.Pin(21), scl = machine.Pin(22), freq = 400000)
4 #Η εναλλακτικά
5 #i2c = machine.SoftI2C(sda=machine.Pin(21), scl=machine.Pin(22))
6 from ssd1306 import SSD1306_I2C
7 sensor = dht.DHT11(machine.Pin(33))
8 sensor.measure()
9
10 #display = SSD1306_I2C(128, 64, i2c) #Δίνω μπορώ να βάλω διεύθυνση π.χ. SSD1306_I2C(128, 64, i2c, 0x3c)
11 display = SSD1306_I2C( width=128, height=64, i2c=i2c, addr=0x3c, external_vcc=False )
12 display.fill(0) #Καθαρίζει οθόνη
13 #Δημιουργώ ετικέτες
14 display.text('Hello World', 5, 5, 1) #Θέση x, y, color 1/0
15 display.text("Temper. : ", 5, 35, 1)
16 display.ellipse(106, 36, 1, 1, 1)
17 display.text("C", 108, 35, 1)
18 display.text("Humidity: ", 5, 45, 1)
19 display.text("%", 105, 45, 1)
20 #Δημιουργώ διπλό περίγραμμα
21 display.rect(0,0,127,63,1,0) #x, y, width, height, color=0/1, fill=0/1
22 display.rect(2,2,123,59,1,0) #x, y, width, height, color=0/1, fill=0/1
23 display.show() #Παρουσιάζει στην οθόνη
24
25 #Για πάντα
26 while True:
27     sensor.measure()
28     display.rect(80, 34, 25, 9, 0, True) #Σβήσε παλιά τιμή
29     display.text("{: 3}".format(sensor.temperature()), 80, 35, 1) #Εμφάνισε νέα τιμή
30     display.rect(80, 44, 25, 9, 0, True) #Σβήσε παλιά τιμή
31     display.text("{: 3}".format(sensor.humidity()), 80, 45, 1) #Εμφάνισε νέα τιμή
32     print("Θερμοκρασία:", sensor.temperature()) #Εμφάνισε στο τερματικό
33     print("Σχετ. Υγρασία:", sensor.humidity())
34     display.show() #Παρουσιάζει στην οθόνη
35     time.sleep(2) #Περίμενε 2 sec
```

## Άσκηση 2

Να τροποποιηθεί το προηγούμενο ώστε να εμφανίζει και μετρήσεις του αισθητήρα BMP280. Ο αισθητήρας BMP280 μπορεί να μετρήσει ατμοσφαιρική πίεση και θερμοκρασία.

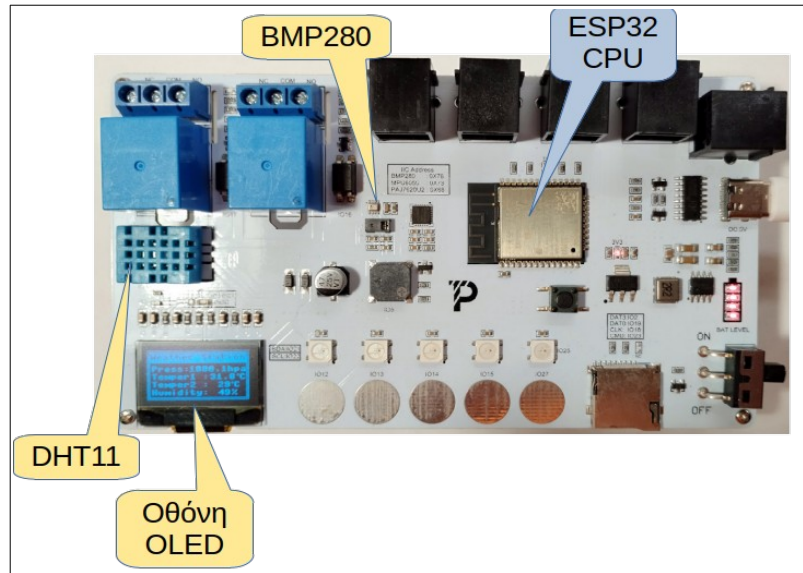
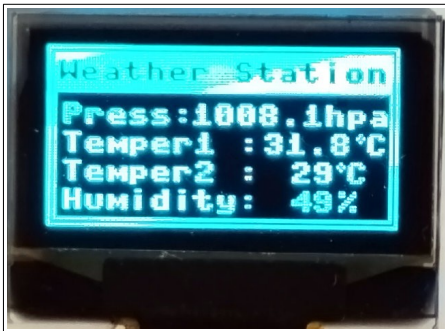
Μέσα στο σύστημα αρχείων πρέπει να αντιγραφτεί η βιβλιοθήκη `ssd1306.py` και η βιβλιοθήκη `bmp280.py`.

```
1 # Πρόγραμμα το οποίο παρουσιάζει μετρήσεις Θερμοκρασίας και υγρασίας από DHT11
2 # και ατμ. πίεσης και θερμοκρασίας από το BMP280 στην οθόνη OLED
3 import machine, time, dht, math
4 i2c = machine.I2C(0, sda = machine.Pin(21), scl = machine.Pin(22), freq = 400000)
5 #Η εναλλακτικά
6 #i2c = machine.SoftI2C(sda=machine.Pin(21), scl=machine.Pin(22))
7 from ssd1306 import SSD1306_I2C #0 ελεγκτής της οθόνης OLED
8 sensor = dht.DHT11(machine.Pin(33)) #0 αισθητήρας DHT11
9 from bmp280 import *
10 bmp = BMP280(i2c) #Στιγμιότυπο του BMP280
11 #Στιγμιότυπο της οθόνης
12 display = SSD1306_I2C(width=128, height=64, i2c=i2c, addr=0x3c, external_vcc=False)
13 display.fill(0) #Καθαρίζει οθόνη
14
15 #Δημιουργώ διπλό περίγραμμα
16 display.rect(0, 0, 127, 63, 1, 0) #x, y, width, height, color=0/1, fill=0/1
17 display.rect(2, 2, 123, 59, 1, 0) #x, y, width, height, color=0/1, fill=0/1
18
```

```

19 #Δημιουργίες ετικέτες
20 display.rect(3, 2, 121, 14, 1, 1) #x, y, width, height, color=0/1, fill=0/1
21 display.text('Weather Station', 3, 5, 0) #θέση x, y, color 1/0
22 display.text("Press: ", 5, 20, 1)
23 display.text("hpa", 100, 20, 1)
24 display.text("Temper1 : ", 5, 30, 1)
25 display.ellipse(113, 31, 1, 1, 1)
26 display.text("C", 115, 30, 1)
27 display.text("Temper2 : ", 5, 40, 1)
28 display.ellipse(106, 41, 1, 1, 1)
29 display.text("C", 108, 40, 1)
30 display.text("Humidity: ", 5, 50, 1)
31 display.text("%", 105, 50, 1)
32 display.show() #Παρουσίασε στην οθόνη
33
34 #Για πάντα
35 while True:
36     bmp.force_measure() #Διάβασε τιμές από BMP280
37     display.rect(52, 19, 48, 9, 0, True) #Εβήσε παλιά τιμή ατμ. πίεσης
38     display.text(str(round(bmp.pressure / 100), 1)), 52, 20, 1) #Εμφάνισε νέα τιμή ατμ. πίεσης
39     display.rect(77, 29, 34, 9, 0, True) #Εβήσε παλιά τιμή θερμοκρασίας BMP
40     display.text(str(round(bmp.temperature, 1)), 78, 30, 1) #Εμφάνισε νέα τιμή θερμοκρασίας BMP
41     print("Θερμοκρασία BMP: " + str(round(bmp.temperature, 1))) #Εμφάνισε σε τερματικό
42     print("Πίεση: " + str(round(bmp.pressure / 100), 1)) #Εμφάνισε σε τερματικό
43
44     sensor.measure() #Διάβασε τιμές από DHT11
45     display.rect(80, 39, 25, 9, 0, True) #Εβήσε παλιά τιμή Θερμοκρασίας DHT
46     display.text("{: 3}".format(sensor.temperature()), 80, 40, 1) #Εμφάνισε νέα τιμή θερμοκρ. DHT
47     display.rect(80, 49, 25, 9, 0, True) #Εβήσε παλιά τιμή σχετ. υγρασίας
48     display.text("{: 3}".format(sensor.humidity()), 80, 50, 1) #Εμφάνισε νέα τιμή υγρασίας
49     print("Θερμοκρασία DHT:", sensor.temperature()) #Εμφάνισε στο τερματικό
50     print("Σχετ. Υγρασία:", sensor.humidity()) #Εμφάνισε στο τερματικό
51     print() #Άσε μια γραμμή κενή
52
53     display.show() #Παρουσίασε στην οθόνη
54     time.sleep(2) #Περίμενε 2 sec

```



### Ασκηση 3

Να γίνει πρόγραμμα σε Micropython το οποίο θα ανιχνεύει το άγγιγμα των κουμπιών IO12 και IO13. Όταν πατιέται το IO12 θα ανάβει το RGB Led πάνω απ' αυτό σε λευκό χρώμα και όταν πατιέται το IO13 θα σβήνει. Κατά το πάτημα των κουμπιών θα ακούγεται και χαρακτηριστικός ήχος από τον βομβητή.

Για την υλοποίηση δεν χρειάζονται επιπλέον βιβλιοθήκες.

```

1 import machine, time, neopixel
2
3 #Ορίσμοι
4 threshold = 200 # Threshold to be adjusted
5 Debounce = 50 #msec
6
7 #Στιγμιότυπο RGB Leds
8 rgb = neopixel.NeoPixel(machine.Pin(25), 5) #IO25, 5 x RGB Leds
9
10 #Ορίσμος ακροδεκτών κουμπιών αφής
11 t_pin1 = machine.TouchPad(machine.Pin(12)) #1ο κουμπί αφής
12 t_pin2 = machine.TouchPad(machine.Pin(13)) #2ο κουμπί αφής

```

```

13
14 #Ορισμός ακροδέκτη Beeper
15 beeper = machine.Pin(5, machine.Pin.OUT) #Ο βομβητής συνδέεται στο pin IO5
16 #Συνάρτηση παραγωγής τόνου συχνότητας freq σε Hz και διάρκειας dur σε msecs
17 def beep(freq = 500, dur = 500): #Προκαθορισμένο 500Hz, 500msec
18     #Υπολογισμοί
19     period = 1.0 / freq #Περίοδος
20     half_per = int((period / 2) * 1000000) #Ημιπερίοδος σε msecs
21     times = int(dur / 1000 / period) #Αριθμός κύκλων
22     for i in range(times):
23         beeper.value(1)
24         time.sleep_us(half_per)
25         beeper.value(0)
26         time.sleep_us(half_per)
27
28 #----- Ορισμός κλάσης ελέγχου κουμπιών αφής (Μόνο απλό κλικ) -----
29 class Touch:
30     #Κατασκευαστής
31     def __init__(self, pin):
32         self.pin = pin #Το GPIO του κουμπιού αφής
33         self.validclick = False #Έγκυρο πάτημα
34         self.timer = 0 #Ο timer που κρατάει τα ticks σε msec
35         self.downtime = 0 #Πόσο χρόνο είναι πατημένο (Δάχτυλο πάνω στο κουμπί αφής)
36
37     #Συμβάν απλό κλικ
38     def onClick(self):
39         self()
40
41     #Καλείται περιοδικά και ελέγχει το κάθε κουμπί
42     def checkButton(self):
43         self.timer = time.ticks_ms() #Κράτησε τον χρόνο συστήματος
44         cVal = self.pin.read() #Διάβασε τιμή του αισθητήρα touch
45         state = False #Δεν πατήθηκε
46         if cVal < threshold: #Αν είναι κάτω από το κατώφλι (Δάχτυλο πάνω στην νησίδα)
47             state = True #Πατήθηκε το κουμπί
48         #----- Πατημένο -----
49         #Πατήθηκε τώρα για πρώτη φορά και ο χρόνος δεν μετράει
50         if state and self.downtime == 0:
51             self.downtime = self.timer #Αρχισε να μετράς τον χρόνο που είναι πατημένο - downtime
52         #Παραμένει πατημένο και έλεγξε αν πέρασε ο χρόνος debounce και δεν έχει ενεργοποιηθεί valid click
53         elif state and not self.validclick and (self.timer - self.downtime) > Debounce:
54             self.validclick = True #Να μην ξαναμπείς εδώ
55             self.onClick() #Κάλεσε συνάρτηση εξυπηρέτησης
56         #----- Ελεύθερο -----
57         #Αλλιώς αν ήταν πατημένο και τώρα το άφησε
58         elif not state:
59             self.downtime = 0 #Επαναφορά
60             self.validclick = False #Επαναφορά για την επόμενη φορά
61         #----- Τέλος ορισμού κλάσης -----
62
63 #----- Συναρτήσεις εξυπηρέτησης συμβάντων -----
64 def key1_click(): #Αναψε
65     print("Led is ON")
66     #----- R, G, B -----
67     rgb[4] = (255, 255, 255) #Τιμές από 0 (σβηστό) έως 255 (μέγιστη ένταση)
68     rgb.write()
69     beep(2000, 50)
70
71 def key2_click(): #Σβήσε
72     print("Led is OFF")
73     rgb[4] = (0, 0, 0)
74     rgb.write()
75     beep(400, 50)
76
77 #===== Κυρίως πρόγραμμα =====
78 #----- Δημιουργία στιγμιοτύπων και ορισμός συναρτήσεων εξυπηρέτησης -----
79 ts1 = Touch(t_pin1)
80 ts1.onClick = key1_click
81
82 ts2 = Touch(t_pin2)
83 ts2.onClick = key2_click
84
85 print("\nESP32 Touch Buttons & RGB Leds Demo")
86
87 #----- Για πάντα -----
88 while True:
89     ts1.checkButton() #Έλεγχος 1ου κουμπιού
90     ts2.checkButton() #Έλεγχος 2ου κουμπιού
91     time.sleep_ms(5) #Περίμενε 5 - 10msec

```

## Άσκηση 4

Να κατασκευαστεί κλάση για τα κουμπιά αφής της πλακέτας η οποία ανιχνεύει μονό κλικ, διπλό κλικ και παρατεταμένο κλικ. Για κάθε συμβάν καλείται ξεχωριστή συνάρτηση εξυπηρέτησης. Επίσης να γραφεί κώδικας ο οποίος δημιουργεί στιγμιότυπα για δύο κουμπιά και εξυπηρετεί και τις τρεις λειτουργίες για το καθένα.

```

1 import machine, time
2 #----- Ορισμοί -----
3 threshold = 200 # Threshold to be adjusted
4 Debounce = 50 #msec
5 DbClickDelay = 300 #msec
6 LongPressDelay = 1000 #msec
7 t_pin1 = machine.TouchPad(machine.Pin(12)) #1ο κουμπί αφής
8 t_pin2 = machine.TouchPad(machine.Pin(13)) #2ο κουμπί αφής
9
10 #----- Ορισμός κλάσης -----
11 class Touch:
12     #Κατασκευαστής
13     def __init__(self, pin):
14         self.pin = pin
15         self.laststate = True
16         self.timer = 0
17         self.downtime = -1

```

```

18     self.uptime = -1
19     self.ignoreUP = False
20     self.singleClickOK = False
21     self.dblClickWaiting = False
22     self.dblClickOnNextUp = False
23     self.longPressHappened = False
24
25     #Συμβάντα
26     #Απλό κλικ
27     def onClick(self):
28         self()
29
30     #Διπλό κλικ
31     def onDblClick(self):
32         self()
33
34     #Παρατεταμένο κλικ
35     def onLongClick(self):
36         self()
37
38     #Καλείται περιοδικά και ελέγχει το κάθε κουμπί
39     def checkButton(self):
40         resultEvent = 0
41         self.timer = time.ticks_ms() #Κράτησε τον χρόνο συστήματος
42         cVal = self.pin.read() #Ανάβασε τιμή του αισθητήρα touch
43         state = False #Δεν πατήθηκε
44         if cVal < threshold: #Αν είναι κάτω από το κατώφλι
45             state = True #Πατήθηκε το κουμπί
46             #----- Πατημένο -----
47             #Πατήθηκε τώρα και πριν δεν είχε πατηθεί και ο uptime έχει ξεπεράσει τον χρόνο debounce.
48             #Άραικά θα μπει εδώ αμέσως όταν πατηθεί γιατί δεν έχει κρατηθεί ο uptime
49             if state == True and self.laststate == False and (self.timer - self.uptime) > Debounce:
50                 self.downtime = self.timer #Άρχισε να μετράς τον χρόνο που είναι πατημένο - downtime
51                 self.ignoreUP = False
52                 self.singleClickOK = True
53                 self.longPressHappened = False
54                 #Αν δεν έχει περάσει ο χρόνος uptime τον χρόνο double click και δεν έχει ενεργοποιηθεί το double click στην επόμενη επαναφορά
55                 #και περιμένει για double click. Εδώ θα μπει κατά το 2ο πάτημα
56                 if (self.timer - self.uptime) < DblClickDelay and self.dblClickOnNextUp == False and self.dblClickWaiting == True:
57                     self.dblClickOnNextUp = True #Στο άφημα να το θεωρήσει διπλό κλικ
58                 else: #Εδώ θα μπει στο 1ο κλικ του διπλού κλικ
59                     self.dblClickOnNextUp = False
60                     self.dblClickWaiting = False #Επανάφορά για την επόμενη φορά
61             #----- Ελεύθερο -----
62             #Αλλιώς αν ήταν πατημένο και τώρα το άφησε και πέρασε ο downtime έχει ξεπεράσει τον χρόνο debounce
63             #Εδώ θα μπει αν κρατήθηκε πατημένο για χρόνο > debounce
64             elif state == False and self.laststate == True and (self.timer - self.downtime) > Debounce:
65                 #Αν δεν θέλει να αγνοήσει το Up. Εδώ δεν μπαίνει κατά το άφημα του long click
66                 if self.ignoreUP == False:
67                     self.uptime = self.timer #Άρχισε να μετράς τον χρόνο που είναι πάνω - uptime χρήσιμος για double click
68                     if self.dblClickOnNextUp == False: #Έχει γίνει το 1ο κλικ και τώρα το άφησε
69                         self.dblClickWaiting = True #Θα περιμένει για δεύτερο πάτημα
70                     #Εδώ έχει συμβεί διπλό κλικ. Είναι πάνω μετά από διπλό κλικ
71                 else:
72                     resultEvent = 2 #Αποτέλεσμα διπλό κλικ
73                     self.dblClickOnNextUp = False #Επανάφορά
74                     self.dblClickWaiting = False
75                     self.singleClickOK = False
76
77             #Ελέγχος για μονό κλικ. Ο χρόνος του διπλού κλικ έχει λήξει
78             if state == False and (self.timer - self.uptime) >= DblClickDelay and self.dblClickWaiting == True and self.dblClickOnNextUp == False and \
79                 self.singleClickOK == True and resultEvent != 2:
80                 resultEvent = 1
81                 self.dblClickWaiting = False
82
83             #Ελέγχος για παρατεταμένο κλικ
84             if state == True and (self.timer - self.downtime) >= LongPressDelay:
85                 #Πυροδότησε το παρατεταμένο πάτημα
86                 #Αν δεν πατήθηκε πριν παρατεταμένα
87                 if self.longPressHappened == False:
88                     resultEvent = 3
89                     self.ignoreUP = True #Αγνόησε το άφημα
90                     self.dblClickOnNextUp = False
91                     self.dblClickWaiting = False
92                     self.longPressHappened = True
93
94             self.laststate = state #Κράτα την προηγούμενη κατάσταση
95             #Κλήσεις ανάλογα με την ενέργεια
96             if resultEvent == 1:
97                 self.onClick()
98             if resultEvent == 2:
99                 self.onDblClick()
100             if resultEvent == 3:
101                 self.onLongClick()
102             #----- Τέλος ορισμού κλάσης -----
103
104             #----- Συναρτήσεις εξυπηρέτησης συμβάντων -----
105             def key1_click():
106                 print("key1 Click")
107
108             def key2_click():
109                 print("key2 Click")
110
111             def key1_dblclick():
112                 print("key1 Double Click")
113
114             def key2_dblclick():
115                 print("key2 Double Click")
116
117             def key1_longclick():
118                 print("key1 Long Click")
119
120             def key2_longclick():
121                 print("key2 Long Click")
122             #----- Τέλος συναρτήσεων εξυπηρέτησης συμβάντων -----
123
124             #===== Κυρίως πρόγραμμα =====
125             #----- Δημιουργία σιγμιγνότητων και ορισμός συναρτήσεων εξυπηρέτησης -----
126             ts1 = Touch(t_pin1)
127             ts1.onClick = key1_click
128             ts1.onDblClick = key1_dblclick
129             ts1.onLongClick = key1_longclick
130
131             ts2 = Touch(t_pin2)
132             ts2.onClick = key2_click
133             ts2.onDblClick = key2_dblclick
134             ts2.onLongClick = key2_longclick
135
136             print("\nESP32 Touch Demo")
137
138             #----- Για πάντα -----
139             while True:
140                 ts1.checkButton() #Ελέγχος 1ου κουμπιού
141                 ts2.checkButton() #Ελέγχος 2ου κουμπιού
142                 time.sleep_ms(10) #Περίμενε 5 - 10msec

```