UNIVERSITY COLLEGE TATI

UNIVERSITY COLLEGE TATI (UCTATI)

FINAL EXAMINATION QUESTION BOOKLET	
COURSE CODE	: BMT 4033
COURSE TITLE	: EMBEDDED SYSTEM DESIGN
SEMESTER/SESSION	: 1-2024/2025
DURATION	: 3 HOURS

Instructions:

1. This booklet contains **4** questions. Answer **all questions**.
2. All theory and calculation answers should be written in the answer booklet.
3. Write legibly and draw sketches wherever required.
4. If in doubt, raise your hands and ask the invigilator.

DO NOT OPEN THIS BOOKLET UNTIL YOU ARE TOLD TO DO SO

THIS BOOKLET CONTAINS 14 PRINTED PAGES INCLUDING COVER PAGE

QUESTION 1

a) **Describe** the difference between python language with c language in developing embedded applications by giving **five (5)** comparisons.

(5 marks)

b) **Figure out** a flow chart based on the program below.

```
from machine import ADC,Pin
from utime import sleep

pot = ADC(Pin(25))      # create ADC object on ADC pin
led1 = Pin(0, Pin.OUT)  #create led output at pin 0
led2 = Pin(1, Pin.OUT)  #create led output at pin 1
while 1:
      value=pot.read_u16()   # read adc put in value
      print("raw value:",value)
      volt=value/65535*3.300;
      print("voltage:",volt)
      if volt<0.5:
            led1.value(0) # led is off
            led2.value(0)
            print("led all off")
      elif volt<2.5:
            led1.value(1) # led is on
            led2.value(0)
            print("led 2 off, led 1 on")
      else:
            led1.value(1) # led is on
            led2.value(1)
            print("led all on")
      sleep(0.5)
```

(15 marks)

QUESTION 2

a) Figure 1 shows the pin out of the 16x2 LCD. **Describe** the function of VSS, VDD, VEE, RS, RW, E and data bus (D0-D7) at LCD.

(7 marks)

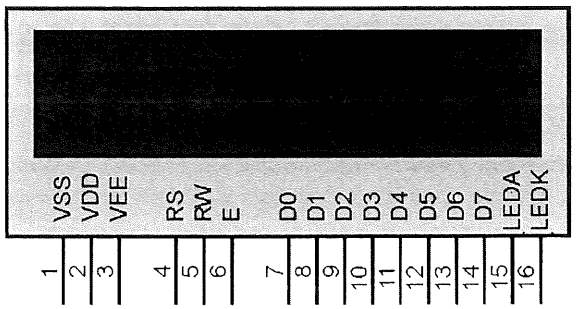


Figure 1: LCD pin out

b) The Raspberry Pi Pico has 3 analog input pins, which convert an analog input (between 0-3.3V by default) into a digital value with 16 bits of resolution. Create an application which **flashes LED change** based on the **setting of the potentiometer,** which minimum time is **0.5 sec** and maximum time is **2.5 sec**. Use a bank of **8 LEDs** and **one potentiometer** reading as ADC input. Connect each LEDs to an output of the Raspberry Pi Pico. Make your code to make all the LEDs flash together.

i. **Produce** schematic diagram for LED connection to raspberry pi Pico by completing the attachment 2 sheet. You may refer to Pi Pico pinout in the attachment section.

(4 marks)

ii. **Identify** a program for this application.

(6 marks)

QUESTION 3

a) **Draw** the internal construction of the 4x3 matrix keypad.

(5 marks)

b) Create an application that will give output at LCD 16x2 using **4-bit mode** connection based on the **4x3 keypad input** value as follows:

- When the any value is pressed, the value is saved as a password key in queue and '*' is displayed at **line 2** of lcd for every number pressed.
- When the '*' is pressed, the password key is set back to clear and lcd also clear.
- When '#' is pressed, it will compare the current password key with preset password key. If the password is correct it will display "correct password" otherwise it will display "wrong password". All display is done at **line 1** of lcd.

i. Referring to the keypad in Figure 2, **produce** the wiring diagram between these devices and the Pi Pico microcontroller. Please use attachment 3 to draw the diagram. You may refer to Pi Pico pinout in the attachment section.

(10 marks)

ii. **Identify** a program for this application. Refer attachment section for the LCD and keypad basic program as reference and make necessary modification as needed.

(15 marks)

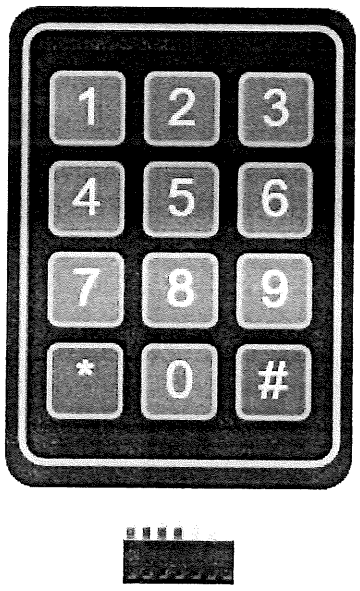


Figure 2: 4x3 Keypad

QUESTION 4

a) **State** the function of a 7-segment display unit, then **identify types** of 7 segment display unit and **describe the difference** of these display types.
(5 marks)

b) Create an application that working as follows:

- Read distance data from SR04 ultrasonic distance sensor in centimeters.
- Read temperature value using NTC sensor in Celsius.
- In LCD display using 4 bit-mode, it will display distance value and state either far or near. Also, it will display current temperature reading and temperature state hot or cold.
- If distance reading is more than 100 cm, it will display far and if not, it will display near.
- If the temperature reading is more than 30 degrees Celsius, it will display hot and if not, it will display cold.
- Using 2 units of TM1637 7 segment display, display temperature at first unit of TM1637, then display distance at second unit of TM1637. Only integer values are displayed.

i) **Produce** schematic diagram for the connection between Pi Pico, SR04 ultrasonic distance sensor, NTC temperature sensor, TM1637 module and LCD. Refer Figure 3 for Tm1637 module, Figure 4 for SR04 ultrasonic distance sensor, Figure 5 for NTC temperature sensor and Figure 1 for LCD. Please use attachment 4 to complete the diagram. You may refer to Pi Pico pinout in the attachment section.
(15 marks)

ii) **Identify** a program for this application.
(13 marks)

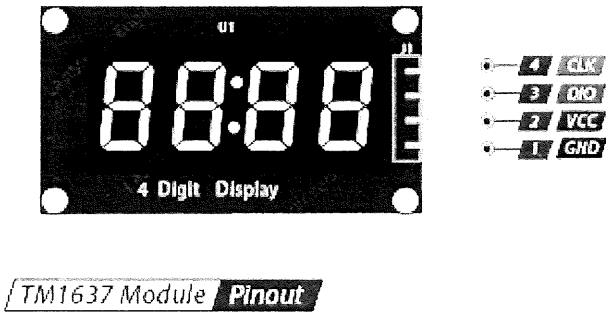


Figure 3: TM1637 module

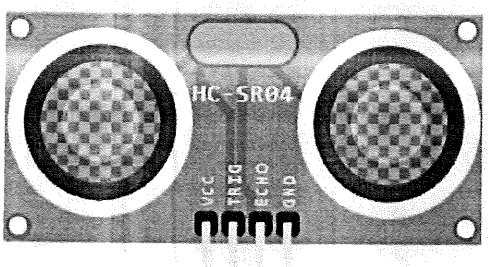


Figure 4: SR04 ultrasonic distance sensor

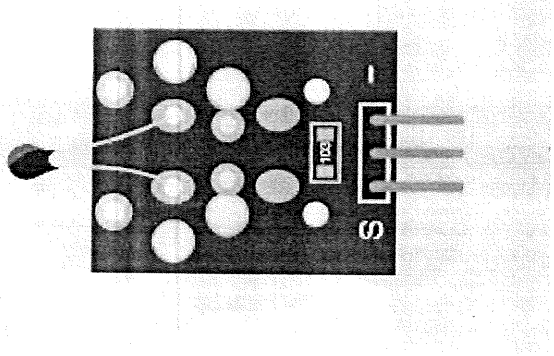


Figure 5: NTC temperature sensor

----------End of question----------

Attachment 1:

LCD.py 4-bit mode basic program routine

```
from machine import Pin
from utime import sleep
rs=Pin(0,Pin.OUT)
e=Pin(1,Pin.OUT)
dataline=(Pin(6,Pin.OUT),Pin(7,Pin.OUT),
          Pin(8,Pin.OUT),Pin(9,Pin.OUT))

def write(val):
    for i in range(4):
        dataline[i].value(((val>>4)&(0x01<<i))>>i)
    e.value(0)
    sleep(0.001)
    e.value(1)
    sleep(0.001)
    for i in range(4):
        dataline[i].value(((val)&(0x01<<i))>>i)
    e.value(0)
    sleep(0.001)
    e.value(1)
    sleep(0.001)

def datawrite(val):
    rs.value(1)
    write(val)

def cmdwrite(val):
    rs.value(0)
    write(val)
    sleep(0.01)

def startlcd():
    cmdwrite(0x02)
    sleep(0.01)
    cmdwrite(0x28)
    sleep(0.01)
    cmdwrite(0x0c)
    sleep(0.01)
    cmdwrite(0x01)
    sleep(0.01)
    cmdwrite(0x80)
    sleep(0.01)
```

```
def strwrite(word):
    for i in word:
        datawrite(ord(i))
def zfl(s, width):
    return '{:0>{w}}'.format(s, w=width)
def zfl_s(s, width):
    return '{:' '>{w}}'.format(s, w=width)
startlcd()
```

Code (Hex)	Command to LCD Instruction Register
1	Clear display screen
2	Return home
4	Decrement cursor (shift cursor to left)
6	Increment cursor (shift cursor to right)
5	Shift display right
7	Shift display left
8	Display off, cursor off
A	Display off, cursor on
C	Display on, cursor off
E	Display on, cursor blinking
F	Display on, cursor blinking
10	Shift cursor position to left
14	Shift cursor position to right
18	Shift the entire display to the left
1C	Shift the entire display to the right
80	Force cursor to beginning to 1st line
C0	Force cursor to beginning to 2nd line
38	2 lines and 5x7 matrix

LCD command code

Keypad.py 4x4 keypad basic program routine

```
from machine import Pin
from utime import sleep

row=(Pin(15,Pin.OUT),Pin(14,Pin.OUT),Pin(13,Pin.OUT),
     Pin(12,Pin.OUT))
col=(Pin(11,Pin.IN,Pin.PULL_UP),Pin(10,Pin.IN,Pin.PULL_UP),
     Pin(9,Pin.IN,Pin.PULL_UP), Pin(8,Pin.IN,Pin.PULL_UP))

label="123A456B789C*0#D"

def getkey():
    val=0
    for j in range (4):
        for i in row:
            i.value(1)
        row[j].value(0)
        for i in range(4):
            if col[i].value()==0:
                val=label[i+j*4]
                sleep(0.1)
                while col[i].value()==0:
                    pass
    return val
```

Tm1637 example program

```
MicroPython TM1637 quad 7-segment LED display driver get at
https://github.com/mcauser/micropython-tm1637

import tm1637
from machine import Pin
from utime import sleep
disp = tm1637.TM1637(clk=Pin(0), dio=Pin(1))

while 1:
    for i in range(10000):
        disp.number(i)
        sleep(0.3)
```

ADC program example

```
from machine import ADC, Pin
adc = ADC(Pin(26))      # create ADC object on ADC pin
adc.read_u16()          # read value, 0-65535 across voltage range 0.0v -
3.3v
```

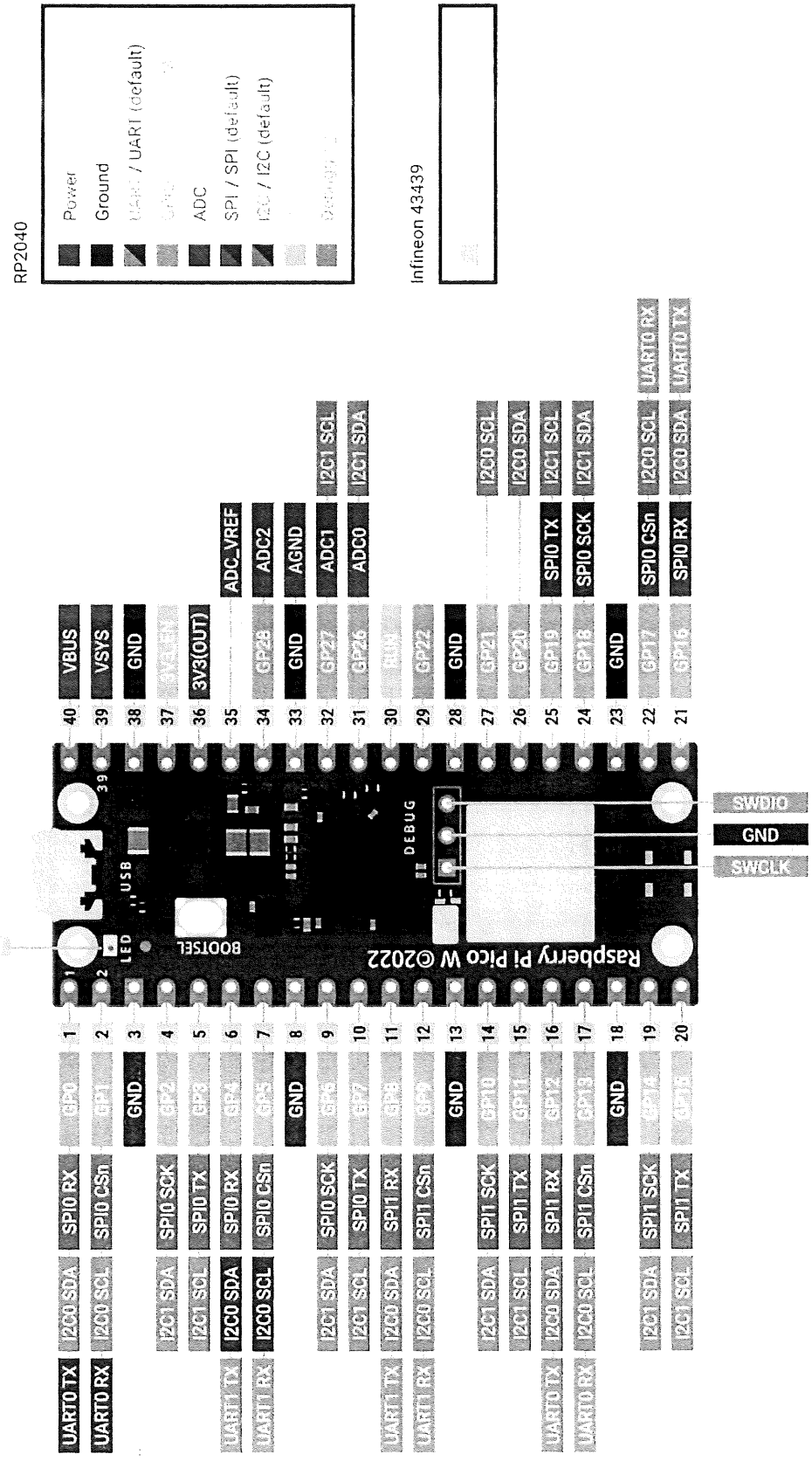
SR04.py ultrasonic routine

```
from machine import Pin,time_pulse_us
from time import sleep,sleep_us
echo = Pin(6, Pin.IN)
trigger = Pin(7, Pin.OUT)
def get_distance():
    trigger.value(1)
    sleep_us(10)
    trigger.value(0)
    timing=time_pulse_us(echo,1,1000000)
    if timing<0:
        print("error measurement")
    else:
        distance=timing/58
    return distance
```

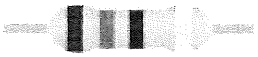
NTC.py temperature routine

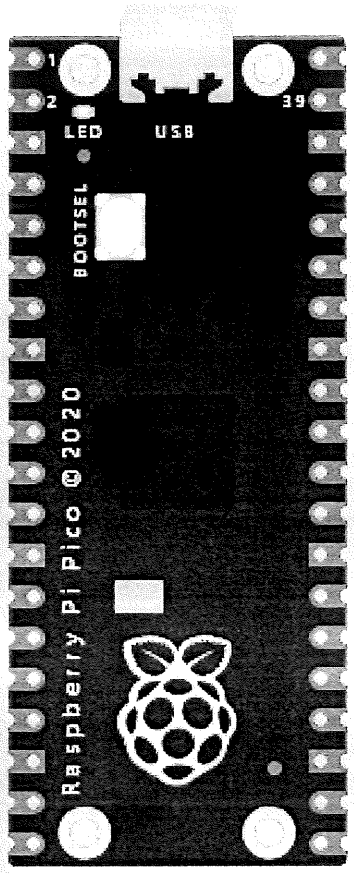
```
from machine import ADC, Pin
from utime import sleep
from math import log
ntc = ADC(Pin(27))      # create pot object on ADC pin
BETA=3950
def get_temp():
  raw=ntc.read_u16() #read raw value
  volt=raw*3.3/65535 # get voltage value
  temp= 1 / (log(1 / (65535. / raw - 1)) / BETA + 1.0 / 298.15) - 273.15
  return temp
```

Raspberry Pi Pico detail Pinout

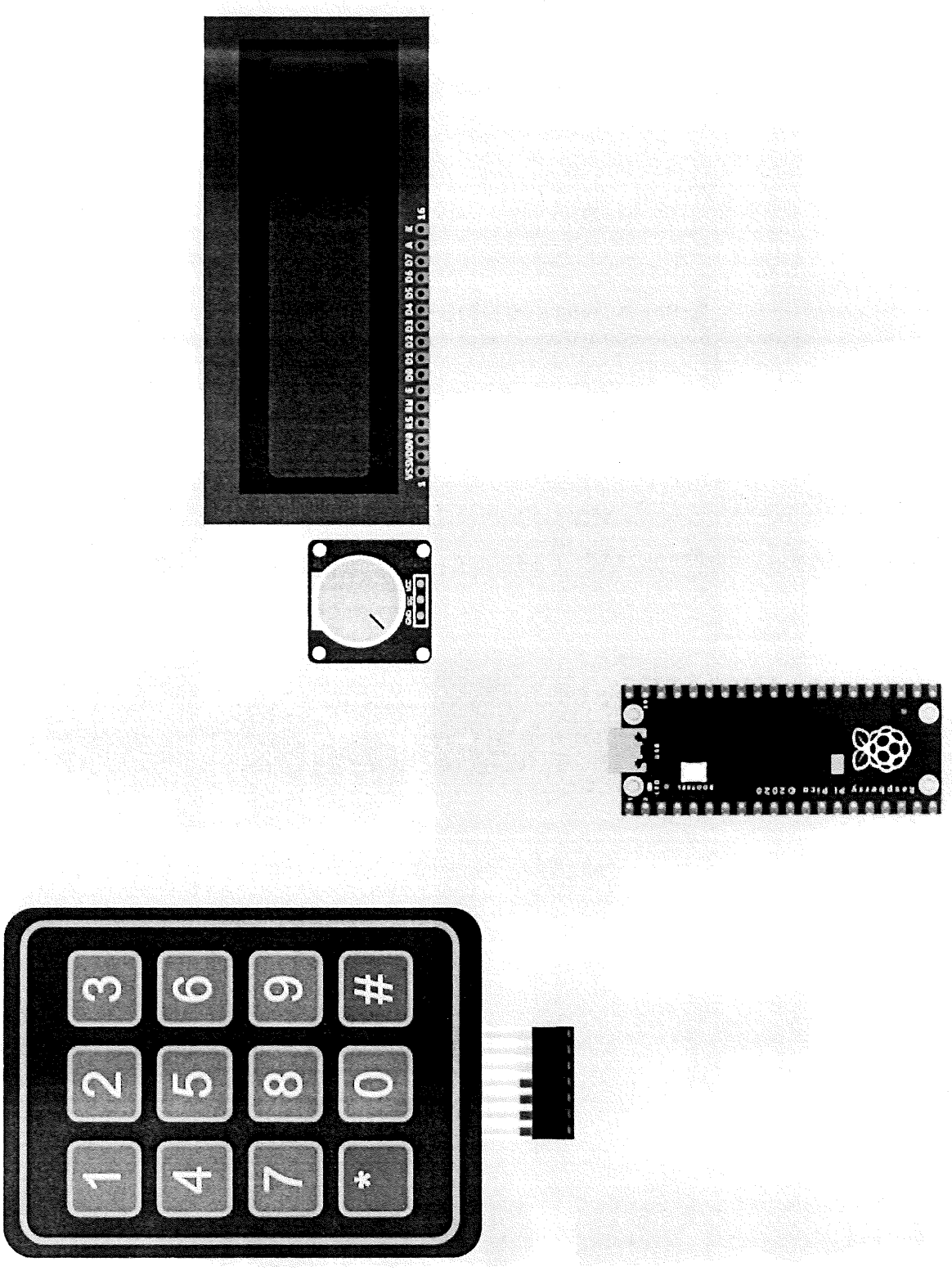


Attachment 2: answer for question 2(b)(i)





Attachment 3: answer for question 3(b)(i)



Attachment 4: answer for question 4(b)(i)