Work Package 4 - Practical

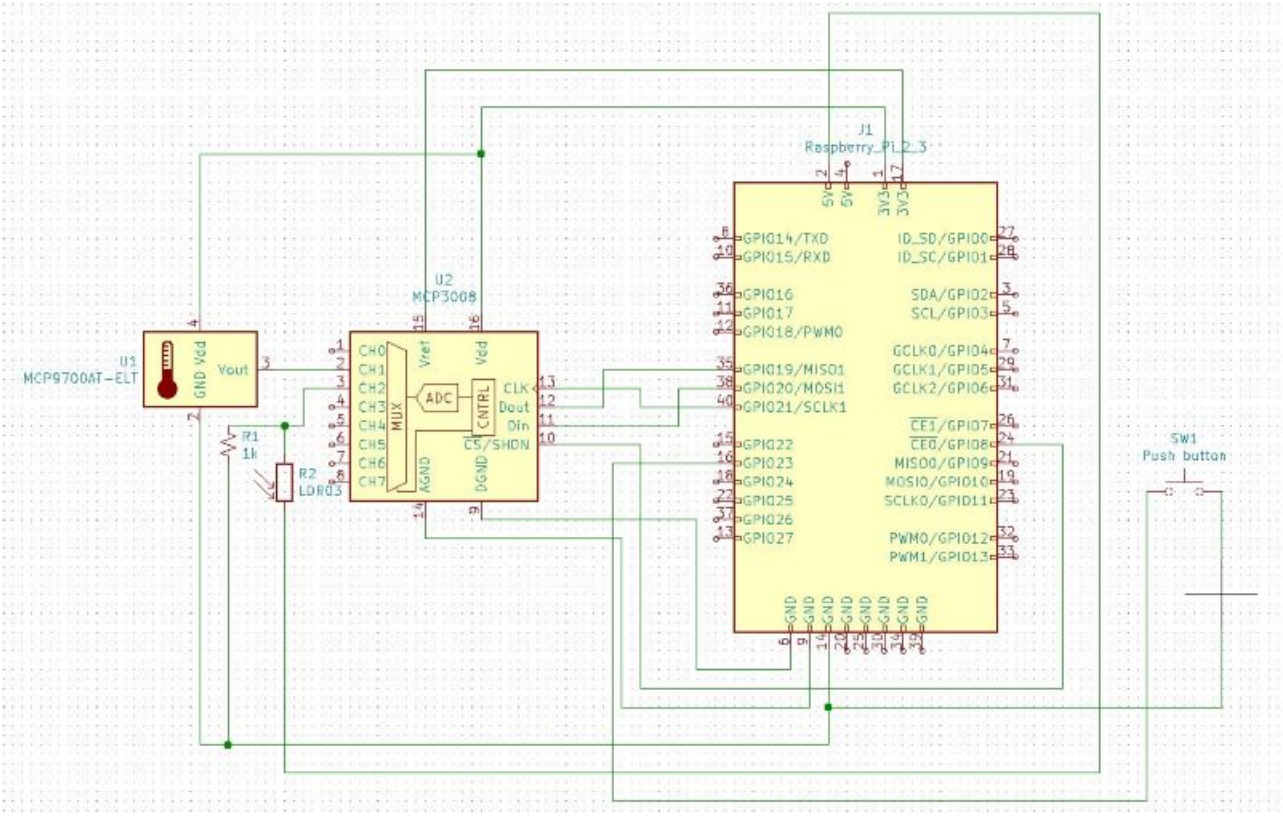
STLTSE004 MVMAMA001
EEE3096S 2021

---

1 Introduction and Aim

In the following practical the aim is to convert an analog signal into a digital input signal using the MCP3008 ADC integrated circuit. A temperature sensor(MCP9700E) is used to capture the analog environmental temperature as a voltage that corresponds to a current sensed temperature which is then converted into degrees Celcius. A light reading is also captured by an LDR; this analogue data is then digitized to represent the amount of light received by the sensor. The MCP3008 is and 8 channel, 10-bit ADC which uses the SPI bus.

2 CIRCUIT DIAGRAM



3 Methodology

The output values from the captured data were tested by allowing various temperature surroundings including normal room temperature, an electric heater temperature, and a refrigerator in order to see the behaviour of the readings. The ldr was also exposed to different lighting conditions to observe the readings of the light intensities. Readings were

taken at intervals of 1 second, 5 seconds, or 10 seconds and the push button was used to switch between these intervals.

4 Validation and Testing

When exposed to a warmer environment, the readings(as seen in the demo video) are seen to rise to higher temperatures. in contrast, the readings decrease when the sensor is exposed to cooler conditions. The ldr also captures less light when covered, resulting in lower values of the "Light Reading". When exposed to more light, the light reading increases.

5 Python Code

Initialization of variables

```
import busio
import digitalio
import board
import adafruit_mcp3xxx.mcp3008 as MCP
from adafruit_mcp3xxx.analog_in import AnalogIn
import threading
from time import time
begin = time()
# creating the spi bus
spi = busio.SPI(clock=board.SCK, MISO=board.MISO, MOSI=board.MOSI)
# creating the counter for changing the interval when pressing the button
counter = 1

cs = digitalio.DigitalInOut(board.D5)
# creating the mcp object
mcp = MCP.MCP3008(spi, cs)
# creating the analog input channel on pin 2
chan_ldr = AnalogIn(mcp, MCP.P2)
chan_mcp = AnalogIn(mcp, MCP.P1)
# creating the button initialising it to connect to gpio23(pin 16)
button = digitalio.DigitalInOut(board.D23)
button.dirrection =digitalio.Direction.INPUT
button.pull = digitalio.Pull.UP

num_press = 0
```

Function used to switch time intervals using the button

```
def button_callback():

    global num_press
    num_press = 1;
    global counter
```

```
    if counter == 10:
        counter = 5
    elif counter == 5:
        counter = 1
    else:
        counter = 10
    print("Interval changed to",str(counter)+"s")
```

Function to display the results at the toggle time configured by the button

```
def Display_Results():
    global num_press
    num_press = 0;
    global start
    global counter
# using the threading timer to time the thread to use a specific interval
    thread = threading.Timer(counter, Display_Results)
    thread.daemon=True
    thread.start()
    len = int(time() - begin)
# converting temperature from volts to degrees C
    temp = round(((chan_mcp.voltage-0.5)/0.01),3)
    print("{:<11}{:<16}{:<10}{:<5}{:}".format(str(len)+"s",chan_mcp.value,temp,"C",
if __name__ == "__main__":
    print("Runtime Temp Reading Temp Light Reading")
    Display_Results()
# The program will run for a maximum of 1 minute 10 seconds with the intervals changing
start = time() +70
while True:
    if time()>start:
        break
    if not button.value:
        if num_press<1:
            button_callback() # calling the button method
```

The main method

```
if __name__ == "__main__":

    main()

    # The program will run for a maximum of 1 minute 10 seconds with the intervals changing

    start = time() +70
    while True:
        if time()>start:
            break
        if not button.value:
```

```
            if num_press<1:
                button_callback()
```

6 Results

The results(as seen in the demo video) show that the temperature readings rise when the sensor is covered(or exposed to heat) and the readings drop when the sensor is uncovered. Moreover, the light readings also change accordingly with the change light exposure to the ldr(as seen in the demo video) The button also performs its respective functions of changing the time intervals

7 Future recommendations

In the future it would be more beneficial to configure this program in such a way that it creates interrupts that will break from certain interval, say 10 second intervals, to 5 second intervals immediately when the button is pressed, without the previously chosen interval overflowing into the current interval in the beginning.

8

The demo video link can be accessed here.