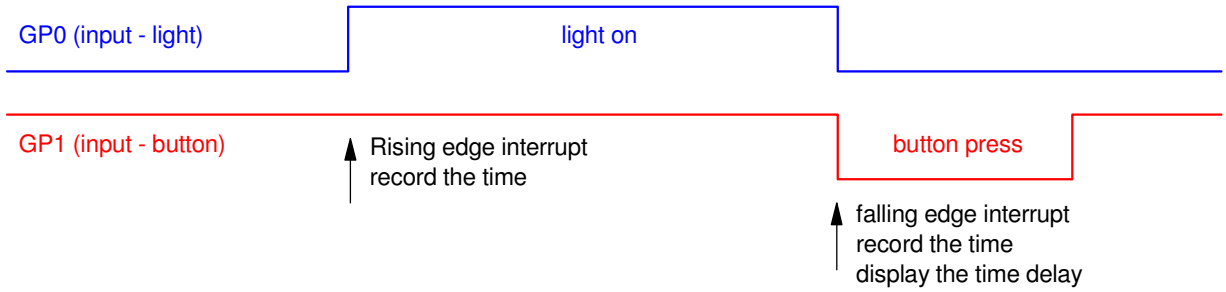
ECE 476/676 - Test #2: Name _____

1) Edge Interrupts: In order to measure your reflex time, a device turns on a light (detected on GP0) and then waits until you press a button (detected on GP1).

Write the interrupt initialization and interrupt service routine which:

- Triggers an edge interrupt on the rising edge of GP0
  - When this interrupt happens, it records the time with a resolution of 1ms
- Triggers an edge interrupt on the falling edge of GP1
  - When this interrupt happens, it records the time with a resolution of 1ms
  - It also sends the time difference to the display with a *print()* statement



Edge Interrupt for GP0 rising edge interrupt - record the time	Edge interrupt for GP1 falling-sdge interrupt - record time & display time delay
Initialization	Initialization
Interrupt Service Routine	Interrupt Service Routine

Problem #1 Code:

Notes:

- Global variables are needed to pass data between routines
- Two separate interrupt service routines are needed (light and button)
- Light interrupt is set up as a rising edge interrupt
- Button interrupt is set up as a falling edge interrupt

```
from machine import Pin
from time import ticks_ms, sleep

Light_Time = Button_Time = dT = 0

Light  = Pin(0, Pin.IN, Pin.PULL_UP)
Button = Pin(1, Pin.IN, Pin.PULL_UP)

def Light_On(Light):
    global Light_Time
    Light_Time = ticks_ms()

def Button_Press(Button):
    global Button_Time, dT
    Button_Time = ticks_ms()
    dT = Button_Time - Light_Time

Light.irq(trigger=Pin.IRQ_RISING, handler=Light_On)
Button.irq(trigger=Pin.IRQ_FALLING, handler=Button_Press)

while (1):
    print(Light_Time, Button_Time, dT)
    sleep(0.2)
```

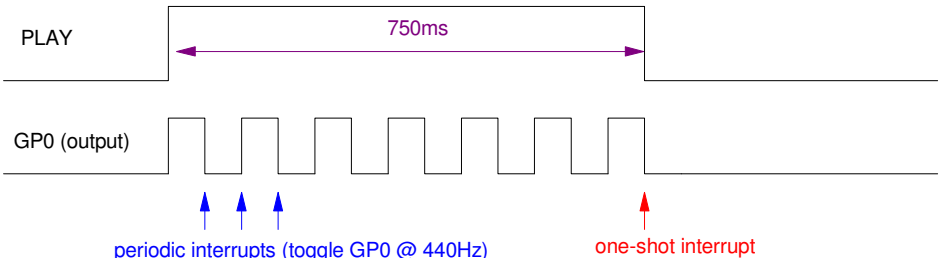
Comments:

- Interrupt #1 is in red
  - rising edge on GP0
  - records the time of the rising edge
- Interrupt #2 is in blue
  - falling edge on GP1
  - records the time and the time difference
- Global variables are needed to pass data
  - main routine
  - interrupt #1
  - interrupt #2

2) Timer Interrupts: Write a python program which uses timer interrupts to

- play a 220Hz note on GP0 (toggle @ 440Hz with a periodic interrupt)
- for 750ms (one-shot interrupt turns off the sound)

Use a global variable, PLAY, to set the duration of the note



Problem 2 Code

```
from machine import Pin, Timer
from time import sleep

t0 = Timer()
t1 = Timer()
PLAY = 0

spkr  = Pin(0, Pin.OUT)

def Play_220Hz(t0):
    global PLAY
    if(PLAY):
        spkr.toggle()
    else:
        spkr.value(0)

def Play_Off(t1):
    global PLAY
    PLAY = 0

t0.init(freq=440, mode=Timer.PERIODIC, callback=Play_220Hz)

while(1):
    PLAY = 1
    t1.init(period=750, mode=Timer.ONE_SHOT, callback=Play_Off)
    sleep(2)
```

Comment

- interrupt t0 is in blue
  - called at 440Hz.
  - If PLAY==1, GP0 is toggled
- interrupt t1 is in red
  - called 750ms after it is turned on (one shot in the main routine)
  - clears PLAY to turn off the sound

3) Analog Sensors: Assume a temperature sensor tells you the temperature in degrees C

- variable degC, type = float

Write a Python subroutine which sets the pulse width on pin GP16 based upon the temperature:

Temperature degC	<20C	20C to 40C	>40C
Duty Cycle	0%	0% to 100% (proportional)	100%

Problem #3 Code

```
from machine import Pin, PWM
from time import sleep

Aout = Pin(16, Pin.OUT)
Aout = PWM(Pin(16))
Aout.freq(1000)

def Analog(degC):
    if(degC < 20):
        duty = 0
    elif(degC < 40):
        duty = 5*(degC - 20)
    else:
        duty = 100
    Aout.duty_u16(int(duty*65535/100))
    return(duty)

for T in range(0,50,5):
    PW = Analog(T)
    print(T, PW)
    sleep(0.1)
```

```
  degC  PW
    0   0
    5   0
    10  0
    15  0
    20  0
    25  25
    30  50
    35  75
    40  100
    45  100
    50  100
```

Comments

- only the part in blue is needed (the subroutine)
  - Without the rest of the code, it won't run
- you could also set the pulse width in nano-seconds
  - 0ns = 0%
  - 1_000_000 ns is 100% (assuming 1kHz)

4) Annoy-A-Tron (Exponential Distribution)

Write a Python program which turns on the beeper (GP13=1) for 100ms every x seconds.

Let x be a random number from 0 to infinity with an exponential distribution which has a mean of 10 seconds

$$cdf(x) = 1 - \exp(-x/10)$$

$$x = -10 \cdot \ln(1 - p)$$ *where p is the probability in the range of (0,1)*

Problem #4 Code

```
from machine import Pin
from random import random
from time import sleep
from math import log

Beeper = Pin(13, Pin.OUT)
Beeper.value(0)

while(1):

    p = random()
    x = -10*log(1-p)

    sleep(x)
    print('Beep', p, x)

    Beeper.value(1)
    sleep(0.1)
    Beeper.value(0)
```

Comments

- you need to set the beeper (GP16) to be an output for it to work
- you need a while(1): loop to keep going and going
- you need to compute a new x every time it beeps
- sleep(0.1) is one option for 100ms
  - timer interrupts could be used
  - that's a different problem on the test

Generally Useful Python Routines

Binary Input (Button Pressed)

```
from machine import Pin

Button = Pin(15, Pin.IN, Pin.PULL_UP)
x = Button.value()
```

Binary Output (Blinking Light)

```
from machine import Pin

LED = Pin(16, Pin.OUT)
LED.toggle()
LED.value(1)
LED.value(0)
```

Analog Input (A2D Read)

```
from machine import ADC

a2d0 = ADC(0)
x = a2d0.real_u16()
```

Analog Output (PWM Output)

```
from machine import Pin, PWM

Aout = Pin(16, Pin.OUT)
Aout = PWM(Pin(16))
Aout.freq(1000)

# 0% duty cycle
Aout.duty_u16(0x0000)

# 100% duty cycle
Aout.duty_u16(0xFFFF)

# 50us pulse
Aout.duty_ns(50_000)
```

Measure a pulse width in milli-seconds

```
from machine import Pin, time_pulse_ms

X = Pin(19, Pin.IN, Pin.PULL_UP)
low = time_pulse_ms(19, 0, 500_000)
high = time_pulse_ms(19, 1, 500_000)
```

Pause 1.23 seconds

```
from time import sleep

sleep(1.23)
```

For Loops

```
for i in range(0,6):
    d1 = i
    for j in range(0,4):
        d2 = j
        y = d1 + d2
```

While Loops

```
t = 0
while(t < 5):
    t = t + 0.01
    print(t)
```

If - else if - else statements

```
if(x < 10):
    a = 1
elif(x < 20):
    a = 2
else:
    a = 3
```

Random Numbers

```
from random import random

p = random()
# x = 0.000 to 0.999
```

Measure time since reset

```
from time import ticks_ms

x0 = ticks_ms()
```

Interrupts

Edge Interrupt: Up Counter

```
from machine import Pin

interrupt_flag=0
N = 0

pin = Pin(15,Pin.IN,Pin.PULL_UP)
def IntServe(pin):
    global interrupt_flag
    global N
    interrupt_flag=1
    N = N + 1

pin.irq(trigger=Pin.IRQ_FALLING,
handler=IntServe)

while(1):
    if(interrupt_flag):
        print("N = ", N)
        interrupt_flag=0
```

Timer Interrupt: periodic @ 1 sec

```
from machine import Pin, Timer
from time import sleep

led = Pin(17, Pin.OUT)
tim = Timer()
N = 0

def tic(timer):
  global N
  N += 1

tim.init(freq=1, mode=Timer.PERIODIC,
callback=tic)

while(1):
  print(N)
  sleep(0.1)
```

Timer Interrupt: (one-shot – 5 sec delay)

```
from machine import Pin, Timer

tim = Timer()

pin1 = Pin(15,Pin.IN,Pin.PULL_UP)
Fan  = Pin(17,Pin.OUT)

def FanOff(pin1):
  Fan.value(0)

while(1):
  while(pin1.value() == 0):
    Fan.value(1)
  tim.init(freq=1/5, mode=Time.ONE_SHOT,
    callback=FanOff)
  while(pin1.value() == 1):
    pass
```