Electronics for IoT

General Purpose Input-Output GPIO

Bernhard E. Boser
University of California, Berkeley
boser@eecs.berkeley.edu

Topics

- Digital Input
  - Monitoring button presses
- Interrupts
- Timers
- PWM

Digital Input

Example: Switch

Internal Pull-Up/Down Resistors

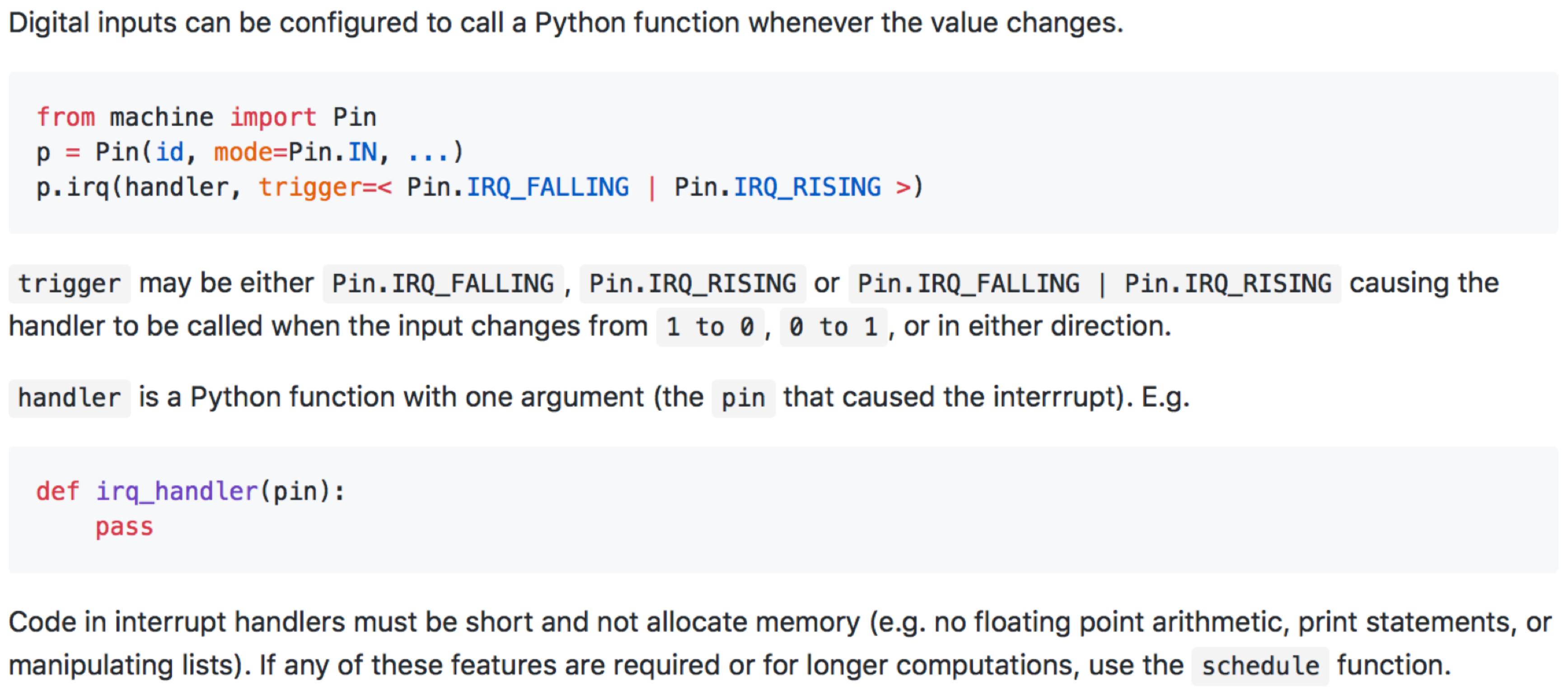
```
from machine import Pin
p = Pin(id, mode=Pin.IN, pull=<None|Pin.PULL_UP|Pin.PULL_DOWN>)
```

Switch ”Bounce”

Switch “debouncing”

Polling versus Interrupts

Interrupts with MicroPython



Digital inputs can be configured to call a Python function whenever the value changes.

```
from machine import Pin
p = Pin(id, mode=Pin.IN, ...)
p.irq(handler, trigger=< Pin.IRQ_FALLING | Pin.IRQ_RISING >)
```

`trigger` may be either `Pin.IRQ_FALLING`, `Pin.IRQ_RISING` or `Pin.IRQ_FALLING | Pin.IRQ_RISING` causing the handler to be called when the input changes from `1 to 0`, `0 to 1`, or in either direction.

`handler` is a Python function with one argument (the `pin` that caused the interrrupt). E.g.

```
def irq_handler(pin):
    pass
```

Code in interrupt handlers must be short and not allocate memory (e.g. no floating point arithmetic, print statements, or manipulating lists). If any of these features are required or for longer computations, use the `schedule` function.

Rising / Falling Edge

Interrupt Handler

Example: Count button presses

Timers

Example: Periodic Timer

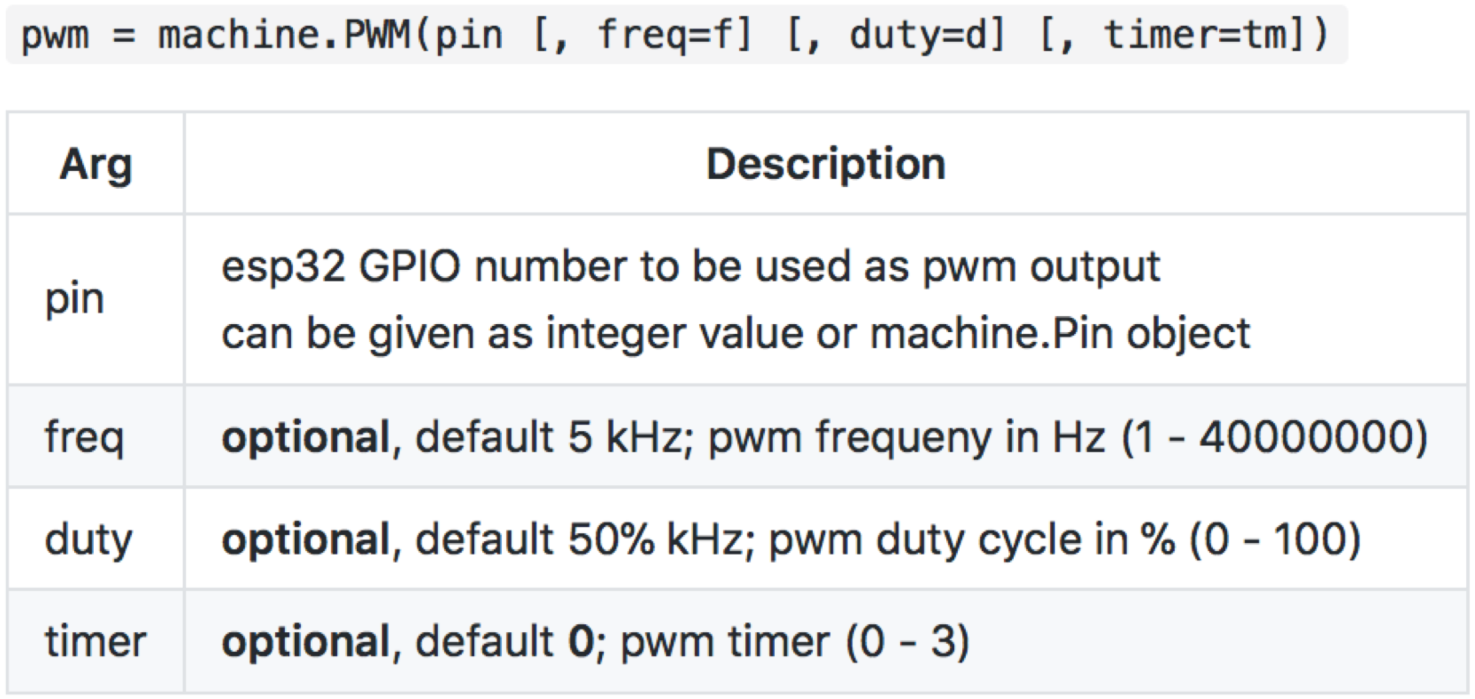
```
import machine

tcounter = 0

p1 = machine.Pin(27)
p1.init(p1.OUT)
p1.value(1)

def tcb(timer):
    global tcounter
    if tcounter & 1:
        p1.value(0)
    else:
        p1.value(1)
    tcounter += 1
    if (tcounter % 10000) == 0:
        print("[tcb] timer: {} counter: {}".format(timer.timernum(), tcounter))

t1 = machine.Timer(2)
t1.init(period=20, mode=t1.PERIODIC, callback=tcb)
```

PWM

- Example: Dim an LED
- Options:

Example: Dimming LED

Hardware PWM



```
pwm = machine.PWM(pin [, freq=f] [, duty=d] [, timer=tm])
```

Arg	Description
pin	esp32 GPIO number to be used as pwm output can be given as integer value or machine.Pin object
freq	**optional**, default 5 kHz; pwm frequeny in Hz (1 - 40000000)
duty	**optional**, default 50% kHz; pwm duty cycle in % (0 - 100)
timer	**optional**, default **0**; pwm timer (0 - 3)

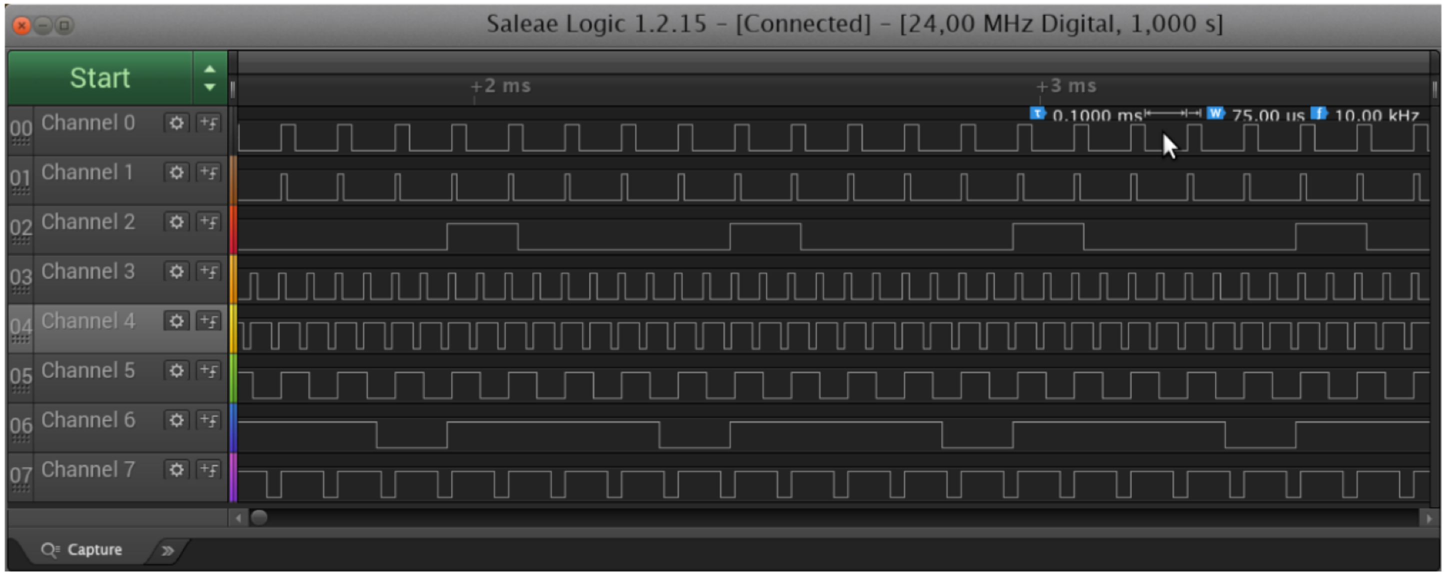
Frequency and Duty Cycle

Implementation Details – Hardware Limitations

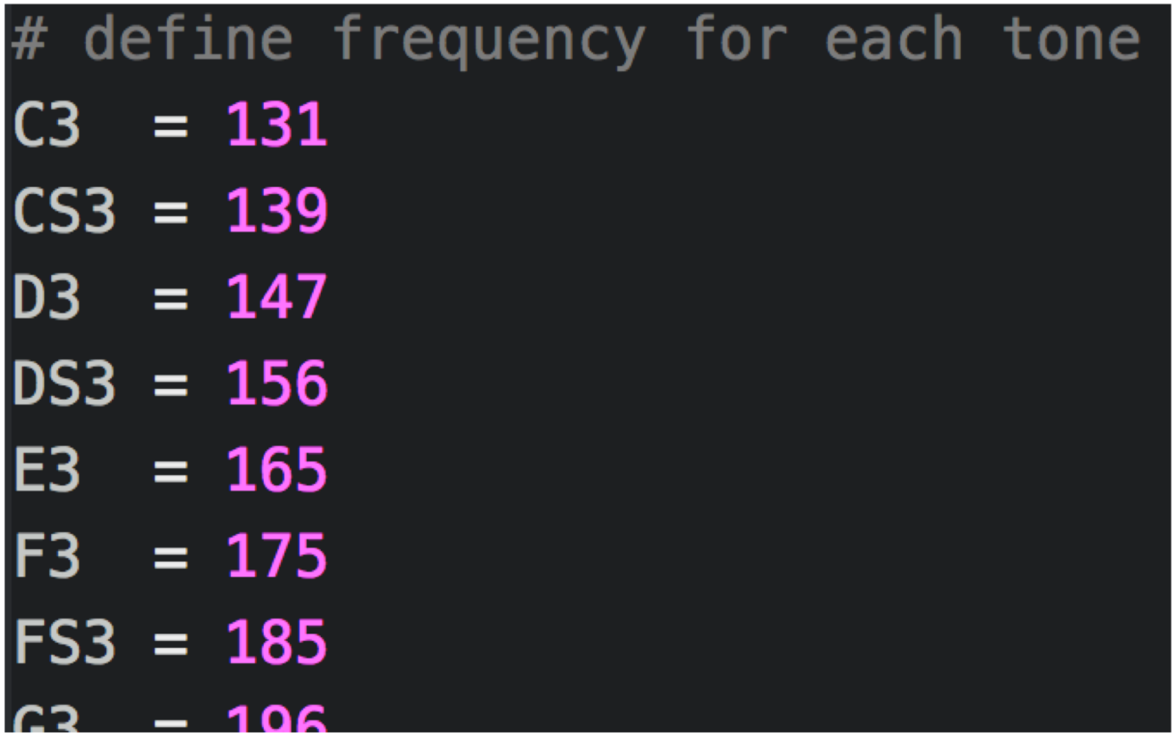
- Max 8 PWM channels
  - Independent duty cycles
- Max 4 “timers”
  - Each set to specific frequency
  - → max 4 different frequencies

PWM Example

```
>>> PWM.list()
PWM(pin: 21, freq=10000 Hz, duty=25% [1024], duty resolution=12 bits, channel=0,
PWM(pin: 23, freq=10000 Hz, duty=9% [409], duty resolution=12 bits, channel=1, t
PWM(pin: 25, freq=2003 Hz, duty=25% [8192], duty resolution=15 bits, channel=2,
PWM(pin: 27, freq=20000 Hz, duty=25% [512], duty resolution=11 bits, channel=3,
PWM(pin: 4, freq=20000 Hz, duty=75% [1536], duty resolution=11 bits, channel=4,
PWM(pin: 19, freq=10000 Hz, duty=50% [2048], duty resolution=12 bits, channel=5,
PWM(pin: 22, freq=2003 Hz, duty=25% [8192], duty resolution=15 bits, channel=6,
PWM(pin: 26, freq=10000 Hz, duty=75% [3072], duty resolution=12 bits, channel=7,
```

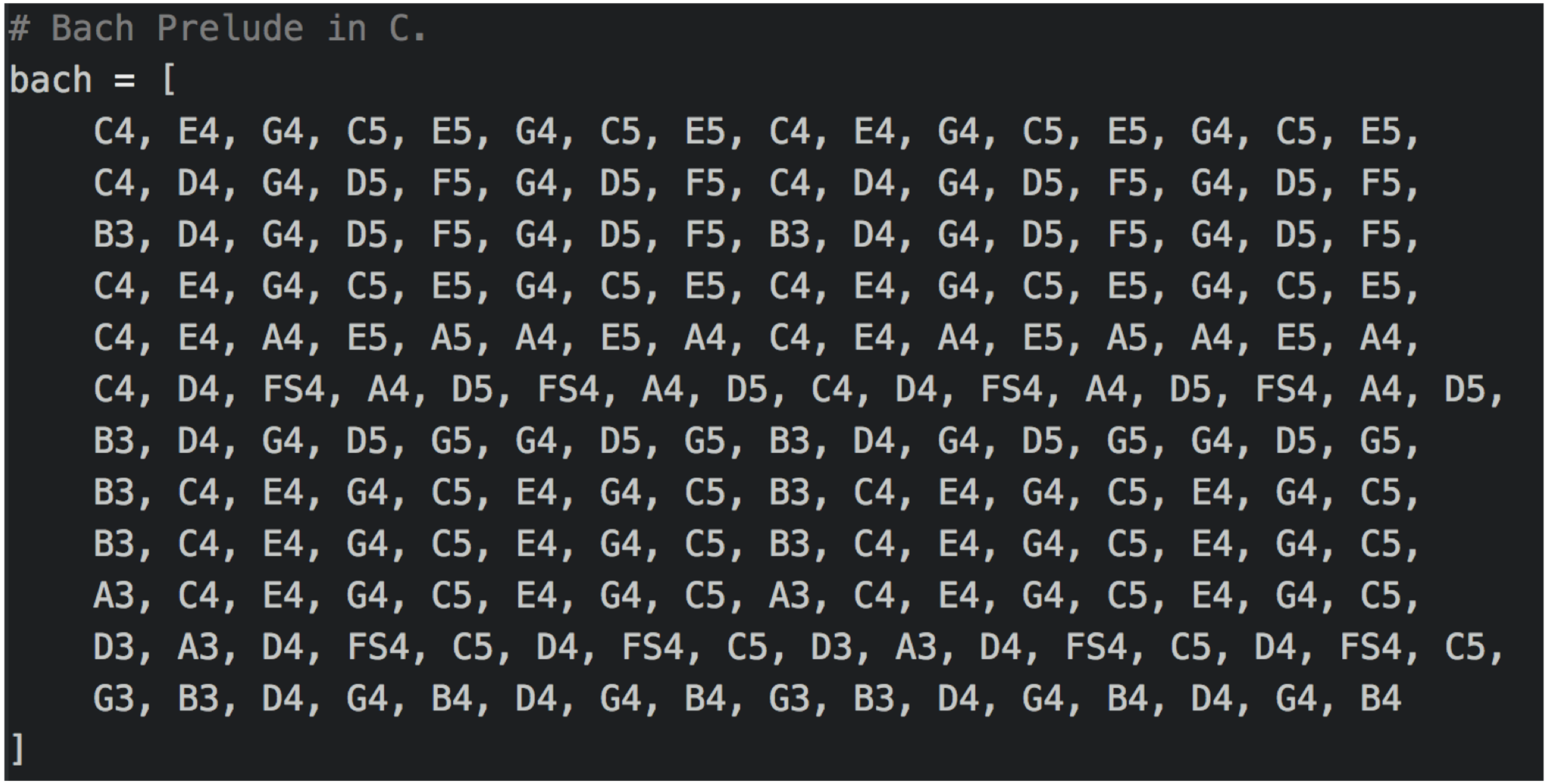


Application: Music!



```
# define frequency for each tone
C3  = 131
CS3 = 139
D3  = 147
DS3 = 156
E3  = 165
F3  = 175
FS3 = 185
G3  = 196
```

Define a "Tune"



```
# Bach Prelude in C.
bach = [
    C4, E4, G4, C5, E5, G4, C5, E5, C4, E4, G4, C5, E5, G4, C5, E5,
    C4, D4, G4, D5, F5, G4, D5, F5, C4, D4, G4, D5, F5, G4, D5, F5,
    B3, D4, G4, D5, F5, G4, D5, F5, B3, D4, G4, D5, F5, G4, D5, F5,
    C4, E4, G4, C5, E5, G4, C5, E5, C4, E4, G4, C5, E5, G4, C5, E5,
    C4, E4, A4, E5, A5, A4, E5, A4, C4, E4, A4, E5, A5, A4, E5, A4,
    C4, D4, FS4, A4, D5, FS4, A4, D5, C4, D4, FS4, A4, D5, FS4, A4, D5,
    B3, D4, G4, D5, G5, G4, D5, G5, B3, D4, G4, D5, G5, G4, D5, G5,
    B3, C4, E4, G4, C5, E4, G4, C5, B3, C4, E4, G4, C5, E4, G4, C5,
    B3, C4, E4, G4, C5, E4, G4, C5, B3, C4, E4, G4, C5, E4, G4, C5,
    A3, C4, E4, G4, C5, E4, G4, C5, A3, C4, E4, G4, C5, E4, G4, C5,
    D3, A3, D4, FS4, C5, D4, FS4, C5, D3, A3, D4, FS4, C5, D4, FS4, C5,
    G3, B3, D4, G4, B4, D4, G4, B4, G3, B3, D4, G4, B4, D4, G4, B4
]
```

Play

Summary

- Digital input
  - Pull-ups
  - Buttons – debouncing
- Interrupts versus polling
- Timers
  - (periodic) interrupts
- PWM
  - Dim LED
  - Play tunes …
  - Later: control motor speed