Importing some libraries

In [1]:
```
from pynq.overlays.base import BaseOverlay
import pynq.lib.rgbled as rgbled
import time
```

Programming the PL

In [2]:
```
base = BaseOverlay("base.bit")
```

Defining buttons and LEDs

In [4]:
```
btns = base.btns_gpio
led4 = rgbled.RGBLED(4)
led5 = rgbled.RGBLED(5)
```

Using a loop to blink the LEDS and read from buttons

In [5]:
```
while True:
    led4.write(0x1)
    led5.write(0x7)
    if btns.read() != 0:
        break
    time.sleep(0.1)
    led4.write(0x0)
    led5.write(0x0)
    if btns.read() != 0:
        break
    time.sleep(0.05)
    led4.write(0x1)
    led5.write(0x7)
    if btns.read() != 0:
        break
    time.sleep(0.1)
    led4.write(0x0)
    led5.write(0x0)
    if btns.read() != 0:
        break
    time.sleep(0.05)

    led4.write(0x7)
    led5.write(0x4)
    if btns.read() != 0:
        break
    time.sleep(0.1)
    led4.write(0x0)
    led5.write(0x0)
    if btns.read() != 0:
        break
    time.sleep(0.05)
    led4.write(0x7)
    led5.write(0x4)
    if btns.read() != 0:
        break
    time.sleep(0.1)
    led4.write(0x0)
    led5.write(0x0)
    if btns.read() != 0:
        break
    time.sleep(0.05)

led4.write(0x0)
led5.write(0x0)
```

Using asyncio to blink the LEDS and read from buttons

In [6]:

```
import asyncio
cond = True

async def flash_leds():
    global cond, start
    while cond:
        led4.write(0x1)
        led5.write(0x7)
        await asyncio.sleep(0.1)
        led4.write(0x0)
        led5.write(0x0)
        await asyncio.sleep(0.05)
        led4.write(0x1)
        led5.write(0x7)
        await asyncio.sleep(0.1)
        led4.write(0x0)
        led5.write(0x0)
        await asyncio.sleep(0.05)

        led4.write(0x7)
        led5.write(0x4)
        await asyncio.sleep(0.1)
        led4.write(0x0)
        led5.write(0x0)
        await asyncio.sleep(0.05)
        led4.write(0x7)
        led5.write(0x4)
        await asyncio.sleep(0.1)
        led4.write(0x0)
        led5.write(0x0)
        await asyncio.sleep(0.05)

async def get_btns(_loop):
    global cond, start
    while cond:
        await asyncio.sleep(0.01)
        if btns.read() != 0:
            _loop.stop()
            cond = False

loop = asyncio.new_event_loop()
loop.create_task(flash_leds())
loop.create_task(get_btns(loop))
loop.run_forever()
loop.close()
led4.write(0x0)
led5.write(0x0)
print("Done.")
```

```
Done.
```

Lab work

Using the code from previous cell as a template, write a code to start the blinking when button 0 is pushed and stop the blinking when button 1 is pushed.

In [ ]:

```
import asyncio

blinking = False   # controls whether LEDs blink
running = True     # controls main loop lifetime

async def flash_leds():
    global blinking, running
    while running:
        if blinking:
            # Pattern A
            led4.write(0x1)
            led5.write(0x7)
            await asyncio.sleep(0.1)
            led4.write(0x0)
            led5.write(0x0)
            await asyncio.sleep(0.05)

            # Pattern B
            led4.write(0x7)
            led5.write(0x4)
            await asyncio.sleep(0.1)
            led4.write(0x0)
            led5.write(0x0)
            await asyncio.sleep(0.05)
        else:
            # Idle briefly to avoid a tight CPU loop
            await asyncio.sleep(0.01)

async def get_btns(loop):
    global blinking, running
    while running:
        await asyncio.sleep(0.01)
        state = btns.read()

        if state & 0x1:           # Button 0 → start blinking
            blinking = True
        if state & 0x2:          # Button 1 → stop blinking and exit program
            blinking = False
            running = False
            loop.stop()

loop = asyncio.new_event_loop()
asyncio.set_event_loop(loop)

loop.create_task(flash_leds())
loop.create_task(get_btns(loop))

try:
    loop.run_forever()
finally:
    running = False
    led4.write(0x0)
    led5.write(0x0)
    loop.close()
    print("Done.")
```

```
In [ ]:
```