Interacting with GPIO from MicroBlaze

In [15]:
```
from pynq.overlays.base import BaseOverlay
import time
from datetime import datetime
base = BaseOverlay("base.bit")
```

In [16]:
```
%%microblaze base.PMODB

#include "gpio.h"
#include "pyprintf.h"

//Function to turn on/off a selected pin of PMODB
void write_gpio(unsigned int pin, unsigned int val){
    if (val > 1){
        pyprintf("pin value must be 0 or 1");
    }
    gpio pin_out = gpio_open(pin);
    gpio_set_direction(pin_out, GPIO_OUT);
    gpio_write(pin_out, val);
}

//Function to read the value of a selected pin of PMODB
unsigned int read_gpio(unsigned int pin){
    gpio pin_in = gpio_open(pin);
    gpio_set_direction(pin_in, GPIO_IN);
    return gpio_read(pin_in);
}
```

In [17]:
```
write_gpio(0, 2)
read_gpio(1)
```

```
pin value must be 0 or 1
```

Out[17]:
```
1
```

Multi-tasking with MicroBlaze

In [18]:
```
base = BaseOverlay("base.bit")
```

In [19]:
```
%%microblaze base.PMODA

#include "gpio.h"
#include "pyprintf.h"

//Function to turn on/off a selected pin of PMODA
void write_gpio(unsigned int pin, unsigned int val){
    if (val > 1){
        pyprintf("pin value must be 0 or 1");
    }
    gpio pin_out = gpio_open(pin);
    gpio_set_direction(pin_out, GPIO_OUT);
    gpio_write(pin_out, val);
}

//Function to read the value of a selected pin of PMODA
unsigned int read_gpio(unsigned int pin){
    gpio pin_in = gpio_open(pin);
    gpio_set_direction(pin_in, GPIO_IN);
    return gpio_read(pin_in);
}

//Multitasking the microblaze for a simple function
int add(int a, int b){
    return a + b;
}
```

In [20]:

```
val = 1
write_gpio(0, val)
read_gpio(1)
```

Out[20]:

```
1
```

In [21]:

```
add(2, 30)
```

Out[21]:

```
32
```

Lab work

Use the code from the second cell as a template and write a code to use two pins (0 and 1) for send and two pins (2 and 3) for receive. You should be able to send 2bits (0~3) over GPIO. You'll need to hardwire from the send pins to the receive pins.

In [25]:

```
from pynq.overlays.base import BaseOverlay
from pynq.lib.pmod import Pmod_IO
import time

base = BaseOverlay("base.bit")

# Configure PMODA pins
tx0 = Pmod_IO(base.PMODA, 0, 'out')
tx1 = Pmod_IO(base.PMODA, 1, 'out')
rx0 = Pmod_IO(base.PMODA, 2, 'in')
rx1 = Pmod_IO(base.PMODA, 3, 'in')

def send_2bit(value):
    if value < 0 or value > 3:
        raise ValueError("Value must be 0..3")
    tx0.write(value & 1)
    tx1.write((value >> 1) & 1)

def receive_2bit():
    b0 = rx0.read() & 1
    b1 = rx1.read() & 1
    return (b1 << 1) | b0

# Loopback test (wire 0→2, 1→3)
for v in range(4):
    send_2bit(v)
    time.sleep(0.01)
    r = receive_2bit()
    print("Sent:", v, "Received:", r)
```

```
Sent: 0 Received: 3
Sent: 1 Received: 3
Sent: 2 Received: 3
Sent: 3 Received: 3
```

In [ ]: