gpio

January 12, 2026

1 Interacting with GPIO from MicroBlaze

[164]:
```
from pynq.overlays.base import BaseOverlay
import time
from datetime import datetime
base = BaseOverlay("base.bit")
```

[165]:
```
%%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);
}
```

[166]:
```
write_gpio(0, 2)
read_gpio(1)
```

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

[166]:
```
1
```

2 Multi-tasking with MicroBlaze

[167]:
```
base = BaseOverlay("base.bit")
```

[168]:
```
%%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;
}
```

[169]:
```
val = 1
write_gpio(0, val)
print(read_gpio(4))
```

```
1
```

[170]:
```
add(2, 30)
```

[170]:
```
32
```

3 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.

[ ]:
```python
# New test PMODA
from pynq.overlays.base import BaseOverlay
import time
from datetime import datetime
base = BaseOverlay("base.bit")
```

[171]:
```c
%%microblaze base.PMODA

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

// Channel 1 PMODA D0-D7
gpio parent = gpio_open_device(0);
gpio tx = gpio_configure(parent, 0, 1, 1);  // D0, D1
gpio rx = gpio_configure(parent, 2, 3, 1);  // D2, D3

//Function to turn on/off a selected pin of PMODA
void write_gpio(unsigned int pin, unsigned int val){
    if(pin>7){
        pyprintf("D0-D7 only");
        return;
    }
    if (val > 1){
        pyprintf("pin value must be 0 or 1");
        return;
    }
    gpio pin_out = gpio_open(pin);
    gpio_set_direction(pin_out, GPIO_OUT);
    gpio_write(pin_out, val);
    pyprintf("set write pin: %d \r\n",pin);
}

//Function to read the value of a selected pin of PMODA
unsigned int read_gpio(unsigned int pin){

    if(pin>7){
        pyprintf("D0-D7 only");
        return 0;
    }

    gpio pin_in = gpio_open(pin);
    gpio_set_direction(pin_in, GPIO_IN);

    pyprintf("set read pin: %d\r\n", pin);
    return gpio_read(pin_in);
}
```

```
int read_state_gpio(unsigned int pin){

    if(pin>7){
        pyprintf("D0-D7 only");
        return -1;
    }

    gpio pin_in = gpio_open(pin);
    return gpio_read(pin_in);
}

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

// init
void init(){
    pyprintf("init \r\n");
    for(int i=0; i<8; ++i){
        read_gpio(i); // read only
    }
}

unsigned int read_rx_handle(){
    gpio_set_direction(rx, GPIO_IN); // By deafult GPIO_IN is high(1), so D1D0␣
↪=> 0b11 => 3 (decimal)
    unsigned int rval = gpio_read(rx) & 0b11; // masking gives D3D2
    pyprintf("RX read %d\r\n", rval);
    return rval;
}

void send_tx_handle(unsigned int val){
    if(val > 3){
        pyprintf("val must be 0..3\r\n"); // since 2 bits (0b11) can only be␣
↪max 3 in decimal
        return;
    }
    gpio_set_direction(tx, GPIO_OUT);
    gpio_write(tx, val); // LSB is D0, so val is D1D0
    pyprintf("TX wrote %d\r\n", val);
}
```

[172]:
```
init() # will set all Data pins D0-D7 to 1.
# NO WIRES CONNECTED to test read values
read_rx_handle() # expecting 0b11 value 3
```

```
init
set read pin: 0
set read pin: 1
set read pin: 2
set read pin: 3
set read pin: 4
set read pin: 5
set read pin: 6
set read pin: 7
RX read 3
```

[172]: 3

[173]:
```
# current wire setup:
# D0 is wired to D2 (D0->D2)
# D1 is wired to D3 (D1->D3)
send_tx_handle(0) # 0b00 => D1D0
```

[174]:
```
read_rx_handle(); # expect 0b00 => D3D2
```

```
TX wrote 0
RX read 0
```

[175]:
```
send_tx_handle(2) # expect 0b10 => D1D0
```

[176]:
```
read_rx_handle() # expect 0b10 => D3D2
```

```
TX wrote 2
RX read 2
```

[176]: 2

[178]:
```
# intentional switching of wires to confirm bits are swapped.
# Originally, D0->D2 and D1->D3
# Swapped wires test: D0->D3 and D1->D2
send_tx_handle(2) # 0b10 => D1D0
read_rx_handle() # expect 0b01 => D0D1
```

```
TX wrote 2
RX read 1
```

[178]: 1

[183]:
```
# New test PMODB
from pynq.overlays.base import BaseOverlay
import time
from datetime import datetime
base = BaseOverlay("base.bit")
```

[184]:
```
%%microblaze base.PMODB

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

// Channel 1 PMODB D0-D7
gpio parent = gpio_open_device(0);
gpio tx = gpio_configure(parent, 0, 1, 1);  // D0, D1
gpio rx = gpio_configure(parent, 2, 3, 1);  // D2, D3

//Function to turn on/off a selected pin of PMODA
void write_gpio(unsigned int pin, unsigned int val){
    if(pin>7){
        pyprintf("D0-D7 only");
        return;
    }
    if (val > 1){
        pyprintf("pin value must be 0 or 1");
        return;
    }
    gpio pin_out = gpio_open(pin);
    gpio_set_direction(pin_out, GPIO_OUT);
    gpio_write(pin_out, val);
    pyprintf("set write pin: %d \r\n",pin);
}

//Function to read the value of a selected pin of PMODA
unsigned int read_gpio(unsigned int pin){

    if(pin>7){
        pyprintf("D0-D7 only");
        return 0;
    }

    gpio pin_in = gpio_open(pin);
    gpio_set_direction(pin_in, GPIO_IN);

    pyprintf("set read pin: %d\r\n", pin);
    return gpio_read(pin_in);
}

int read_state_gpio(unsigned int pin){

    if(pin>7){
        pyprintf("D0-D7 only");
        return -1;
    }
    gpio pin_in = gpio_open(pin);
    return gpio_read(pin_in);
}

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

// init
void init(){
    pyprintf("init \r\n");
    for(int i=0; i<8; ++i){
        read_gpio(i); // read only
    }
}

unsigned int read_rx_handle(){
    gpio_set_direction(rx, GPIO_IN); // By deafult GPIO_IN is high(1), so D1D0␣
  ↪=> 0b11 => 3 (decimal)
    unsigned int rval = gpio_read(rx) & 0b11; // masking gives D3D2
    pyprintf("RX read %d\r\n", rval);
    return rval;
}

void send_tx_handle(unsigned int val){
    if(val > 3){
        pyprintf("val must be 0..3\r\n"); // since 2 bits (0b11) can only be␣
  ↪max 3 in decimal
        return;
    }
    gpio_set_direction(tx, GPIO_OUT);
    gpio_write(tx, val); // LSB is D0, so val is D1D0
    pyprintf("TX wrote %d\r\n", val);
}
```

[185]:
```
init() # will set all Data pins D0-D7 to 1.
# NO WIRES CONNECTED to test read values
read_rx_handle() # expecting 0b11 value 3
```

```
init
set read pin: 0
set read pin: 1
set read pin: 2
set read pin: 3
set read pin: 4
set read pin: 5
set read pin: 6
set read pin: 7
RX read 3
```

[185]:
```
3
```

[186]:
```
# current wire setup:
# D0 is wired to D2 (D0->D2)
# D1 is wired to D3 (D1->D3)
send_tx_handle(0) # 0b00 => D1D0
```

[187]:
```
read_rx_handle(); # expect 0b00 => D3D2
```

```
TX wrote 0
RX read 0
```

[188]:
```
send_tx_handle(2) # expect 0b10 => D1D0
```

[189]:
```
read_rx_handle() # expect 0b10 => D3D2
```

```
TX wrote 2
RX read 2
```

[189]:
```
2
```

[190]:
```
# intentional switching of wires to confirm bits are swapped.
# Originally, D0->D2 and D1->D3
# Swapped wires test: D0->D3 and D1->D2
send_tx_handle(2) # 0b10 => D1D0
read_rx_handle() # expect 0b01 => D0D1
```

```
TX wrote 2
RX read 1
```

[190]:
```
1
```

[ ]:
```

```