Working with MIO LEDs and Pushbuttons

Vivado Environment:

1. Create a Vivado project with the same device settings as for the first tutorial.
2. Create a new block design and include only PS in it.
3. Right click and make “FIXED_IO” as external connection.

Note: You can perform “Run automation” to make DDR and FIXED_IO connections external. Connections to DDR are avoided because our C code is relatively small in size and can be accomodated in OCM.

We do not require:

1. AXI GPIO
2. Reset system
3. XDC file.
**PS customization:** Enable both UART1 and GPIO MIO under MIO configuration of PS.

Rest of the operations:

1. Validate your design
2. Create the HDL wrapper
3. Generate block design
4. Synthesize, implement and generate the bit stream
5. Export the hardware and launch SDK

**SDK Environment:**

1. Create an application project
2. Add the C source file “led_btns_ps.c” in your “src” directory.
3. Build the program and generate the binary
4. Program FPGA
5. Do RUN configurations and RUN your project
Responses:

1. When button is not pressed

   Indication: MIO LED is ON when button is not pressed

2. When button is pressed

   Indication: MIO LED is **OFF** when button is pressed
Source code for “led_btns_ps.c”:

```c
// header files
#include "xparameters.h"
#include "xgpiops.h"
#include "xstatus.h"
#include <xil_printf.h>

// defines
#define GPIO_DEVICE_ID XPAR_XGPIOPS_0_DEVICE_ID
#define OUTPUT_PIN 7 /* MIO7, pin connected to LED */
#define INPUT_PIN 50 /* MIO50, pin connected to button */
#define printf xil_printf /* Smaller foot-print printf */

// global variables
XGpioPs GpioPs;
/* The driver instance for GPIO Device. */
XGpioPs_Config *ConfigPtr;

// main function
int main(void)
{
    int Status;

    // initialize the GpioPs driver.
    ConfigPtr = XGpioPs_LookupConfig(GPIO_DEVICE_ID);
    Status = XGpioPs_CfgInitialize(&GpioPs, ConfigPtr, ConfigPtr->BaseAddr);
    if (Status != XST_SUCCESS) { return XST_FAILURE; }

    // configure the LED pin as output
    XGpioPs_SetDirectionPin(&GpioPs, OUTPUT_PIN, 1);
    XGpioPs_SetOutputEnablePin(&GpioPs, OUTPUT_PIN, 1);

    // Set the direction for the specified pin to be input.
    XGpioPs_SetDirectionPin(&GpioPs, INPUT_PIN, 0);

    while(1)
    {
        // if input is read as HIGH, button is not pressed
        u8 data = XGpioPs_ReadPin(&GpioPs, INPUT_PIN);
        XGpioPs_WritePin(&GpioPs, OUTPUT_PIN, !data);
        // LED MIO7 ON when data = 0, OFF when data = 1
    }

    return XST_SUCCESS;
}
```