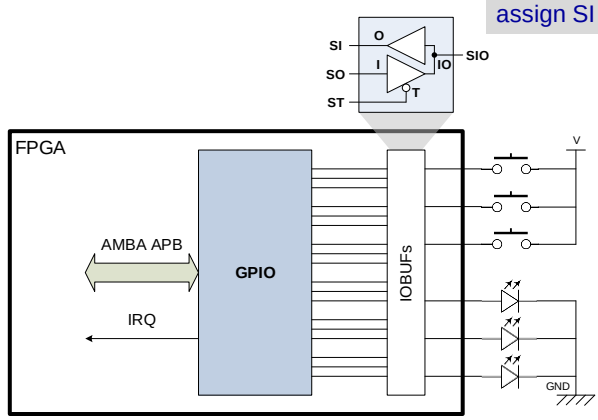
GPIO with AMBA APB interface

2020

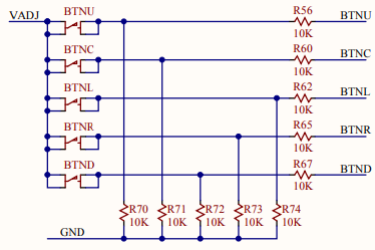
Ando Ki
(adki@future-ds.com)

GPIO testing environment

```
// Xilinx Tristate buffer:
(active-low output enable)
assign SIO = (ST) ? 1'bZ : SO;
assign SI = SIO;
```

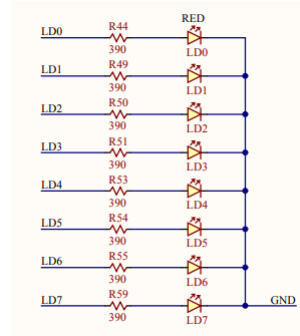


Pushbutton switch and LED




```
set_property PACKAGE_PIN P16 [get_ports {BTNC}];  # "BTNC"
set_property PACKAGE_PIN R16 [get_ports {BTND}];  # "BTND"
set_property PACKAGE_PIN N15 [get_ports {BTNL}];  # "BTNL"
set_property PACKAGE_PIN R18 [get_ports {BTNR}];  # "BTNR"
set_property PACKAGE_PIN T18 [get_ports {BTNU}];  # "BTNU"
set_property IOSTANDARD LVCMOS25 [get_ports BTN?]
```

VADJ selecting jump




```
set_property PACKAGE_PIN T22 [get_ports {LD0}];  # "LD0"
set_property PACKAGE_PIN T21 [get_ports {LD1}];  # "LD1"
set_property PACKAGE_PIN U22 [get_ports {LD2}];  # "LD2"
set_property PACKAGE_PIN U21 [get_ports {LD3}];  # "LD3"
set_property PACKAGE_PIN V22 [get_ports {LD4}];  # "LD4"
set_property PACKAGE_PIN W22 [get_ports {LD5}];  # "LD5"
set_property PACKAGE_PIN U19 [get_ports {LD6}];  # "LD6"
set_property PACKAGE_PIN U14 [get_ports {LD7}];  # "LD7"
set_property IOSTANDARD LVCMOS33 [get_ports LD?]
```


GPIO

- General Purpose Input/Output
  - an uncommitted digital signal pin on an integrated circuit or electronic circuit board whose behavior—including whether it acts as input or output—is controllable by the user at run time.
- Capabilities of GPIO
  - GPIO pins can be configured to be input or output
  - GPIO pins can be enabled/disabled
  - Input values are readable (typically high or low)
  - Output values are writable/readable
  - Input values can often be used as IRQs

Requirements

- It support AMBA APB.
- The number of GPIO ports or pins can be defined when it is instantiated.
  - Up to 32
  - Need parameterized design
- Each pin can be input or output by control register
  - Output port can be read as well
- Input pins can be used as interrupt
  - It can be edge sensitive for both rising and falling.
  - It can be level sensitive for both high and low.

GPIO module

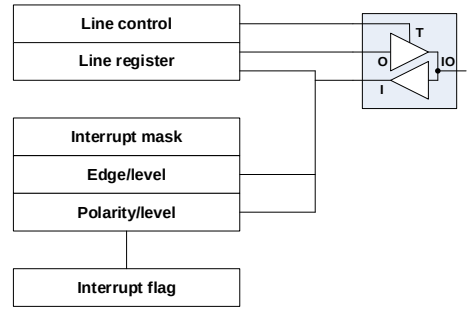
- Module

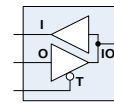
```
module gpio_apb #(parameter GPIO_WIDTH=32)
(
       input   wire                  PRESETn
     , input   wire                  PCLK
     , input   wire                  PSEL
     , input   wire                  PENABLE
     , input   wire [31:0]           PADDR
     , input   wire                  PWRITE
     , output  reg  [31:0]           PRDATA
     , input   wire [31:0]           PWDATA
     , input   wire [GPIO_WIDTH-1:0] GPIO_I
     , output  wire [GPIO_WIDTH-1:0] GPIO_O
     , output  wire [GPIO_WIDTH-1:0] GPIO_T
     , output  wire                  IRQ
);
........
endmodule
```

GPIO registers: CSR (Control Status Register)

- [REGISTERS]
  - 0x00: Line Control Register
    - '0' = input mode (default)
    - '1' = output mode
  - 0x04: Line Register
    - Current GPIO pin level
  - 0x08: Interrupt Mask Register
    - '0' = enabled
    - '1' = masked (disabled) (default)
  - 0x0C: Interrupt Flag
    - '0' = no interrupt
    - '1' = interrupt
  - 0x10: Interrupt Edge/Level Sensitivity Mode Register
    - '0' = Level sensitivity mode (default)
    - '1' = Edge sensitivity mode
  - 0x14: Interrupt Pol Sensitivity Mode Register
    - '0' = active-low for level mode, falling for edge mode (default)
    - '1' = active-high for level mode, rising for edge mode

Active-high output enable





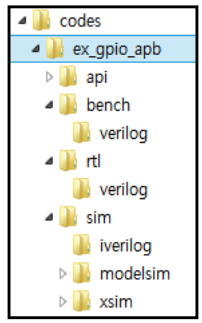

```
// Xilinx Tristate buffer:
assign SIO = (ST) ? 1'bZ : SO;
assign SI = SIO;
```


How GPIO works

- Use a pin as an input:
  - Program the corresponding bit in the Control Register to 'input mode' ('0'). Then, the pin's state (input level) can be checked by reading the Line Register.
    - Note that writing to the GPIO pin's Line Register bit while in input mode has no effect.
- Use a pin as an output:
  - Program the corresponding bit in the Control Register to 'output mode' ('1'). Then, program the GPIO pin's output level by writing to the corresponding bit in the Line Register.
    - Note reading the GPIO pin's Line Register bit while in output mode returns the current input pin level so that it may not reflect the value written.
- Use a pin as an interrupt source:
  - Program the corresponding bit in the Edge Register to the desired sensitivity mode (level or edge). Program the corresponding bit in the Pol Register to the desired sensitivity mode (low/falling or high/rising). Program the corresponding bit in the Mask Register to 'un-masked mode' ('0').

What to do for IP

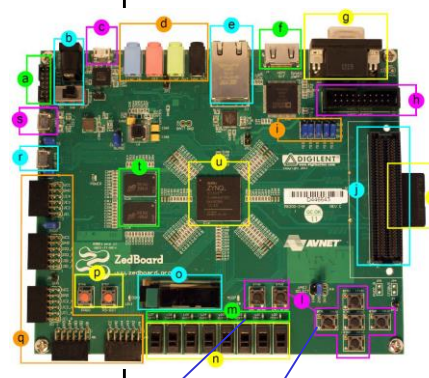
- Make your own project to build GPIO IP itself.
  - ex_gpio_apb



Top level design for ZedBoard

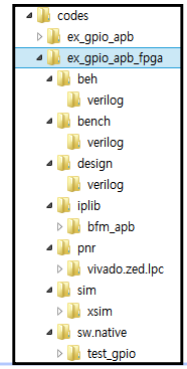
- fpga.v (i.e., fpga_zed.v)

```
module fpga #(parameter SL_PCLK_FREQ   =80_000_000
                      , USR_CLK_FREQ   =80_000_000 )
(
       input   wire           BOARD_RST_SW
     , input   wire           BOARD_CLK_IN
     , input   wire           SL_RST_N
     , output  wire           SL_CS_N
     , output  wire           SL_PCLK
     , input   wire           SL_FLAGA
     , input   wire           SL_FLAGB
     , input   wire           SL_FLAGC
     , input   wire           SL_FLAGD
     , output  wire           SL_RD_N
     , output  wire           SL_WR_N
     , output  wire           SL_OE_N
     , output  wire           SL_PKTEND_N
     , output  wire  [ 1:0]   SL_AD
     , inout   wire  [31:0]   SL_DT
     , input   wire  [ 1:0]   SL_MODE
     , inout   wire           LD0
     , inout   wire           LD1
     , inout   wire           LD2
     , inout   wire           LD3
     , inout   wire           BTNU
     , inout   wire           BTNL
     , inout   wire           BTNR
     , inout   wire           BTND
);
```



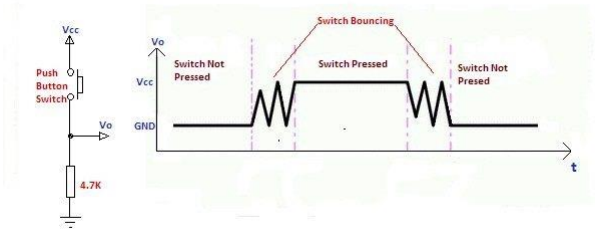
What to do for FPGA board

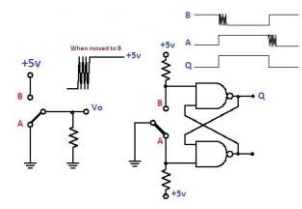
- After preparing your own ‘gpio_apb.v’, make FPGA project which uses push-button switches and LED’s.
  - Pushbutton X corresponds to LED X.
  - Pushing a pushbutton switch alternately toggles its corresponding LED light on OR off.
    - It may not work correctly due to bouncing problem.
    - Do not try to push the BTN-C, which is system reset.



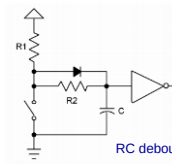
How to deal with bounce

- Bouncing is the tendency of any two metal contacts in an electronic device to generate multiple signals as the contacts close or open
- De-bouncing is any kind of hardware device or software that ensures that only a single signal will be acted upon for a single opening or closing of a contact.





SR debouncer



RC debouncer