Digital Pulse Processor based on SoC-FPGA for Particle Detectors

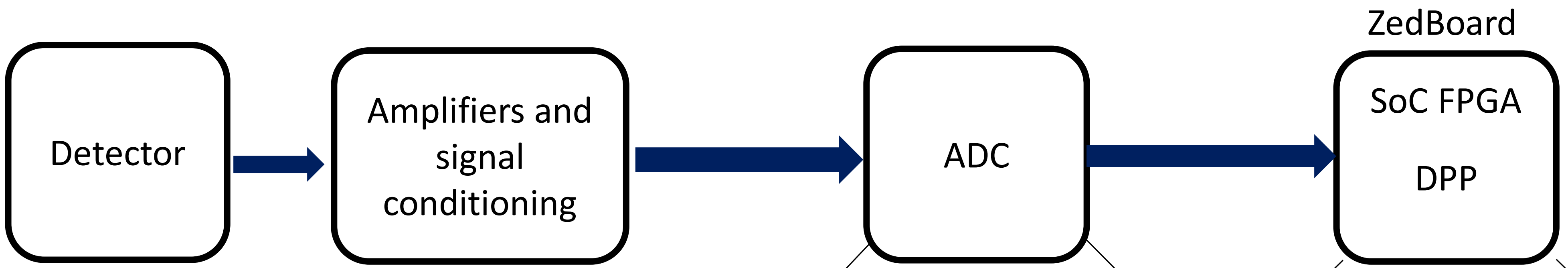
ICTP guided Project

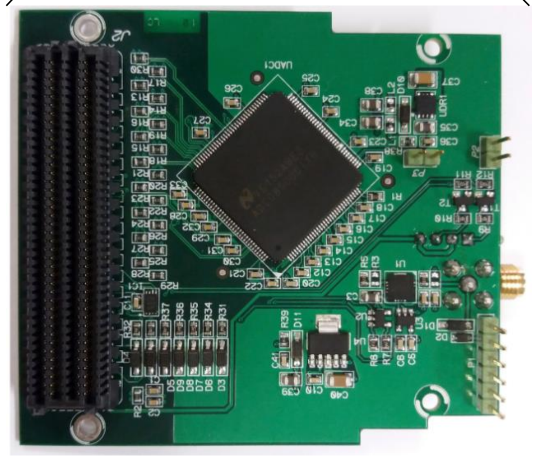
Joint ICTP-IAEA School on FPGA-based SoC and its Applications for Nuclear and Related Instrumentation

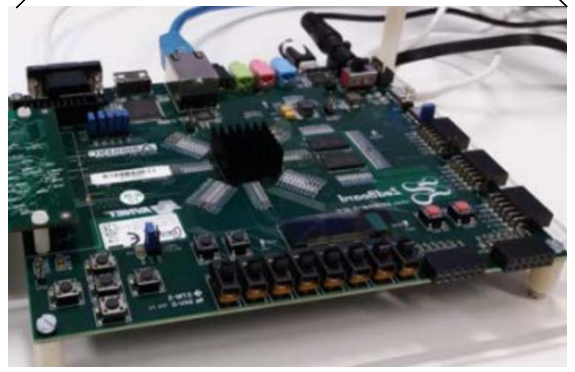
Oscar Olaya

19/02/2021

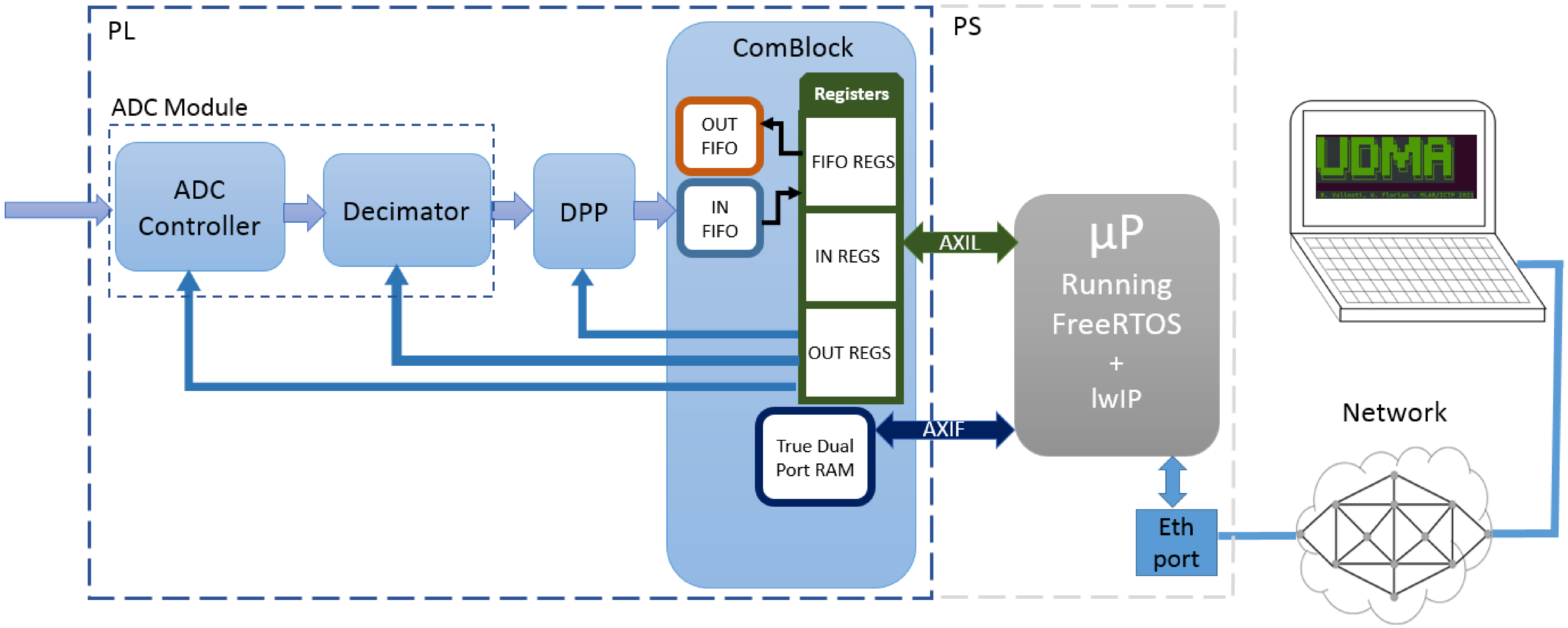
General project structure





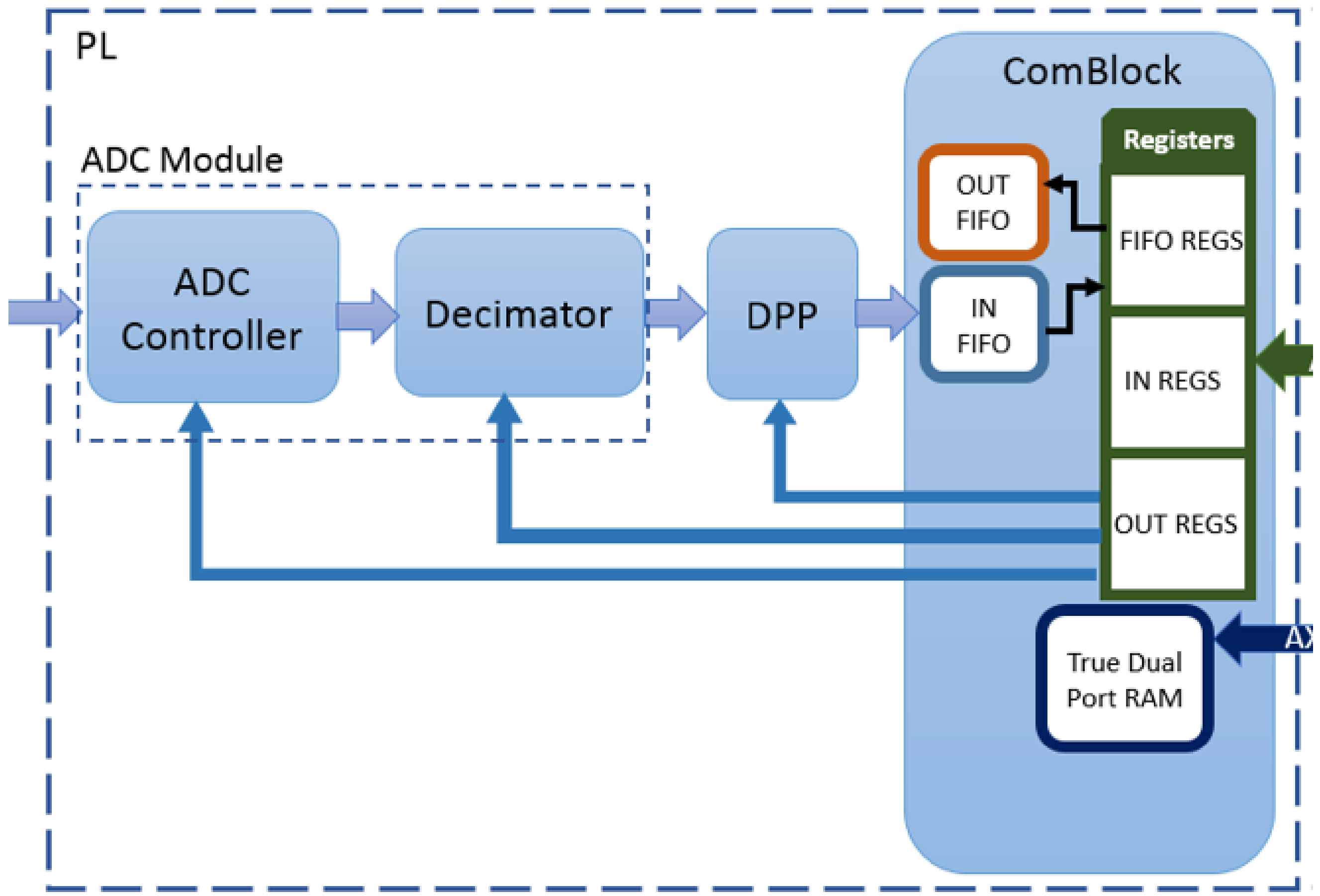


General project structure



Configuration registers

- Configuration registers:
  - ADC Module:
    - ADC control registers
    - ADC decimator
  - DPP Module:
    - FIR coefficients
    - Thresholds
    - Operation mode
    - DPP enable
  - ComBlock FIFO clear



SDK – Application project

Light weight IP – lwIP:

> *“lwIP is a small independent implementation of the TCP/IP protocol suite. The focus of the lwIP TCP/IP implementation is to reduce the RAM usage while still having a full scale TCP. This making lwIP suitable for use in embedded systems with tens of kilobytes of free RAM and room for around 40 kilobytes of code ROM.”*

Taken from:

https://www.nongnu.org/lwip/2_1_x/index.html

ARM µP

+

FreeRTOS

+

lwIP

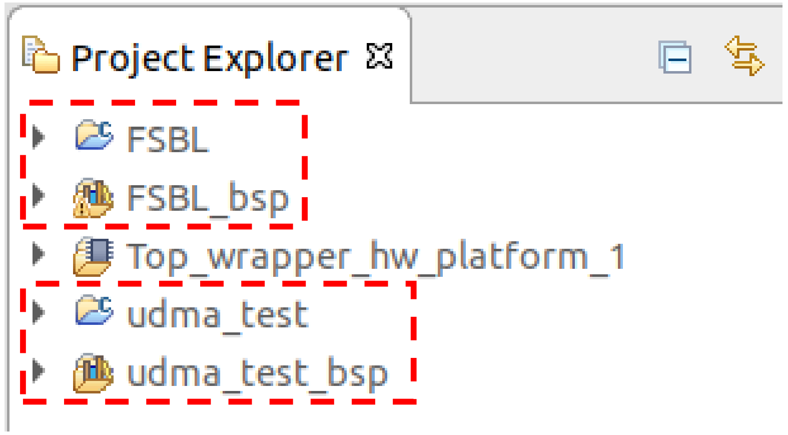
SDK – Application project

"First Stage Bootloader (FSBL) for Zynq. The FSBL configures the FPGA with HW bit stream (if it exists) and loads the Operating System (OS) Image or Standalone (SA) Image…"

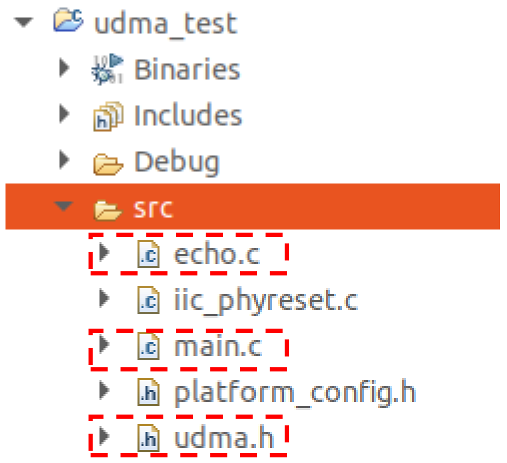
Taken from template description.

"The FreeRTOS lwIP Echo Server application provides a simple demonstration of how to use the light-weight IP stack (lwIP) with FreeRTOS… The server listens for input at port 7 and simply echoes back whatever data is sent to the port"

Taken from template description.



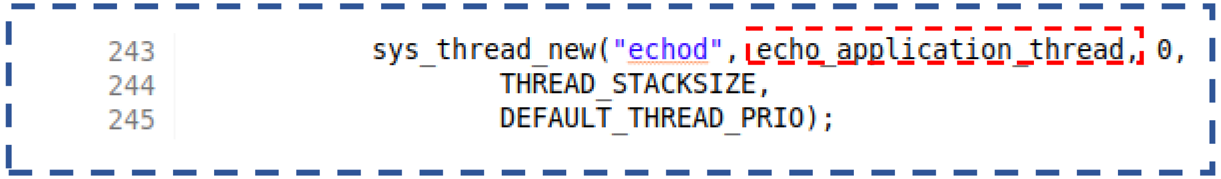
SDK – Application project



```
udma_test
  Binaries
  Includes
  Debug
  src
    echo.c
    iic_phyreset.c
    main.c
    platform_config.h
    udma.h
```

1

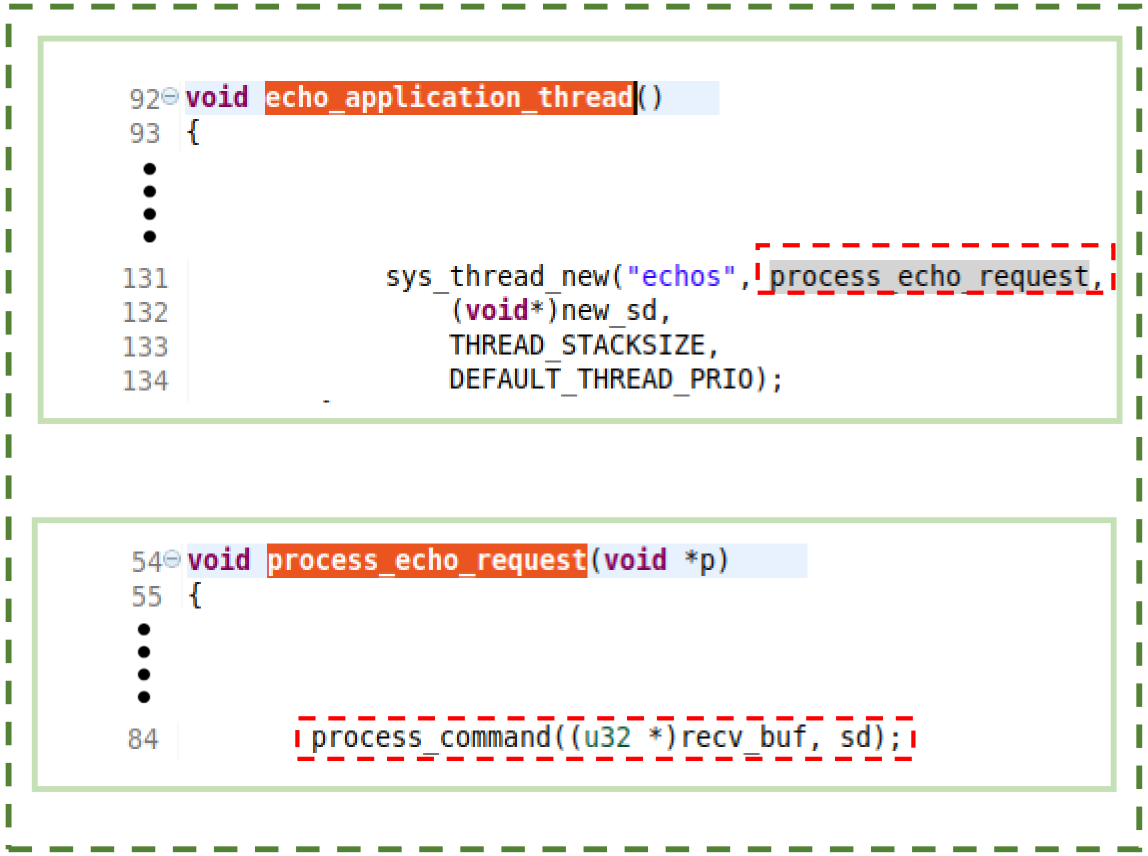
File: main.c



```
            sys_thread_new("echod", echo_application_thread, 0,
                    THREAD_STACKSIZE,
                    DEFAULT_THREAD_PRIO);
```

2

File: echo.c

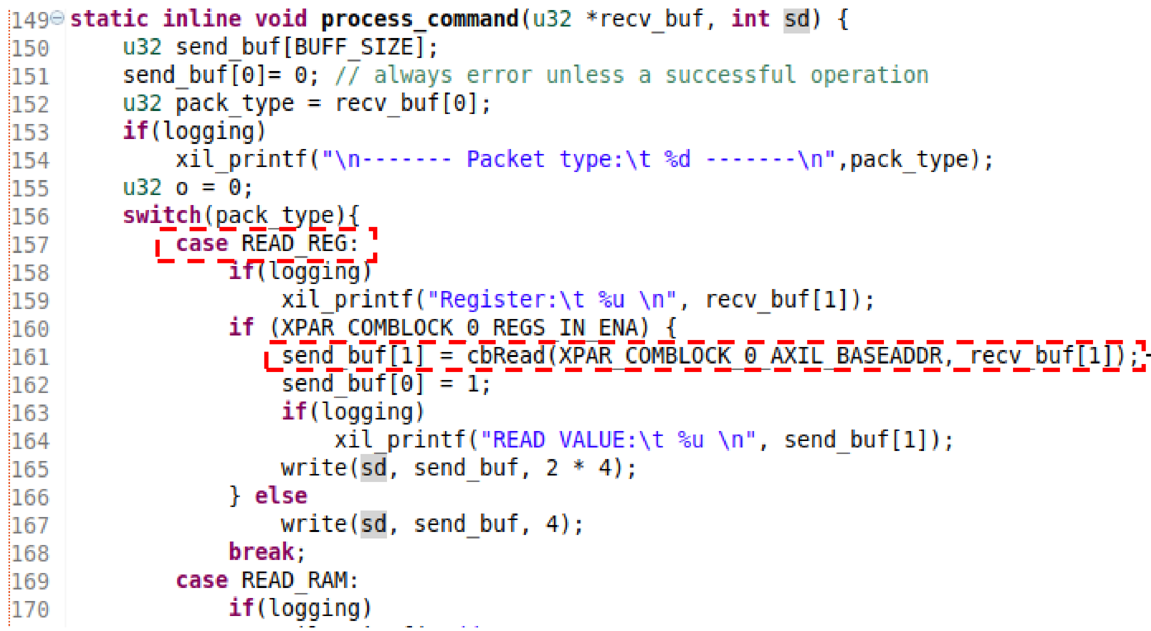


```
void echo_application_thread()
{
•
•
•
•
            sys_thread_new("echos", process_echo_request,
                (void*)new_sd,
                THREAD_STACKSIZE,
                DEFAULT_THREAD_PRIO);
```

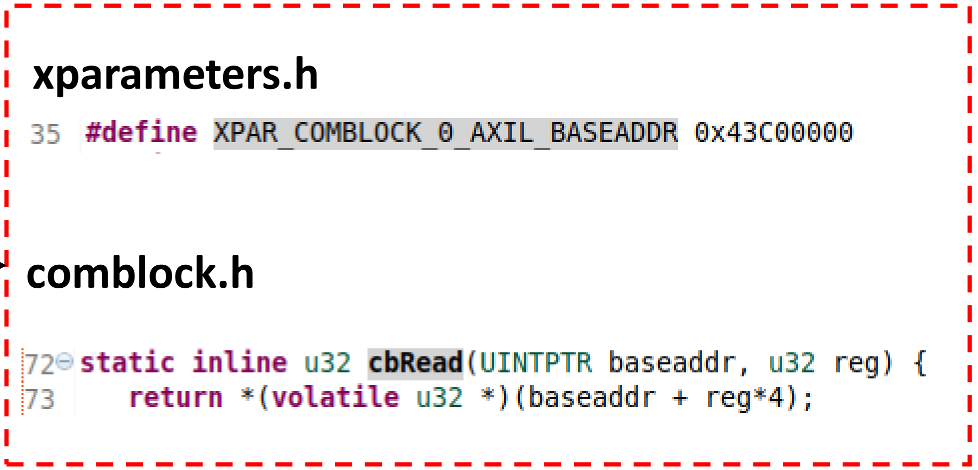
```
void process_echo_request(void *p)
{
•
•
•
•
        process_command((u32 *)recv_buf, sd);
```

SDK – Application project

udma.h



```c
static inline void process_command(u32 *recv_buf, int sd) {
    u32 send_buf[BUFF_SIZE];
    send_buf[0]= 0; // always error unless a successful operation
    u32 pack_type = recv_buf[0];
    if(logging)
        xil_printf("\n------- Packet type:\t %d -------\n",pack_type);
    u32 o = 0;
    switch(pack_type){
        case READ_REG:
            if(logging)
                xil_printf("Register:\t %u \n", recv_buf[1]);
            if (XPAR_COMBLOCK_0_REGS_IN_ENA) {
                send_buf[1] = cbRead(XPAR_COMBLOCK_0_AXIL_BASEADDR, recv_buf[1]);
                send_buf[0] = 1;
                if(logging)
                    xil_printf("READ VALUE:\t %u \n", send_buf[1]);
                write(sd, send_buf, 2 * 4);
            } else
                write(sd, send_buf, 4);
            break;
        case READ_RAM:
            if(logging)
```



xparameters.h

```c
#define XPAR_COMBLOCK_0_AXIL_BASEADDR 0x43C00000
```

comblock.h

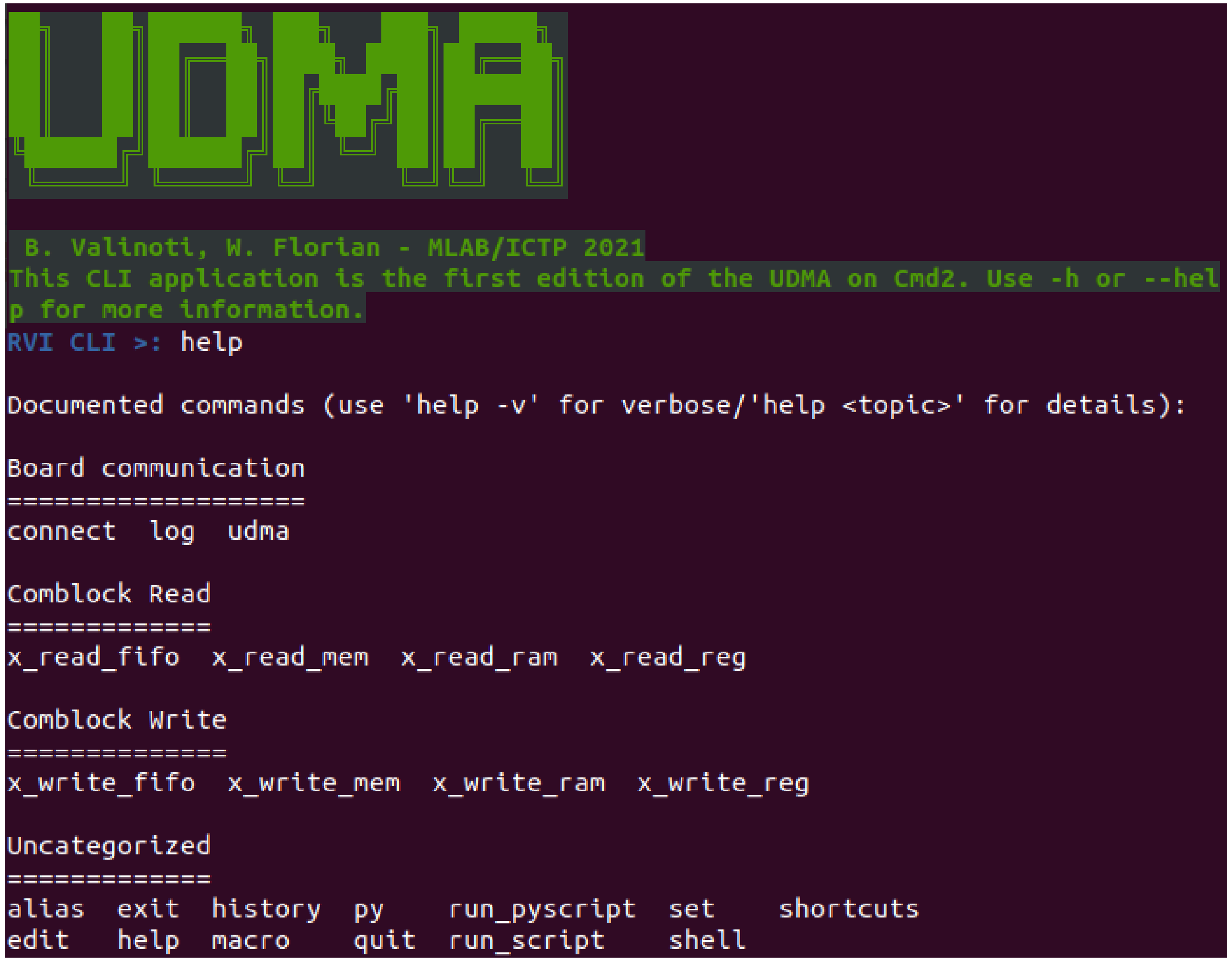
```c
static inline u32 cbRead(UINTPTR baseaddr, u32 reg) {
    return *(volatile u32 *)(baseaddr + reg*4);
```

UDMA

Configure registers:

- ComBlock FIFO clear
- DPP Module:
  - FIR coefficients
  - Thresholds
  - Operation mode
  - DPP enable
- ADC Module:
  - ADC decimator
  - ADC control registers

Possibility of generating scripts that allow creating different configuration templates for the system.



```
 B. Valinoti, W. Florian - MLAB/ICTP 2021
This CLI application is the first edition of the UDMA on Cmd2. Use -h or --hel
p for more information.
RVI CLI >: help

Documented commands (use 'help -v' for verbose/'help <topic>' for details):

Board communication
===================
connect  log  udma

Comblock Read
=============
x_read_fifo  x_read_mem  x_read_ram  x_read_reg

Comblock Write
==============
x_write_fifo  x_write_mem  x_write_ram  x_write_reg

Uncategorized
=============
alias  exit  history  py    run_pyscript  set    shortcuts
edit   help  macro    quit  run_script    shell
```

Experience

- SoC FPGAs design flow
- Working with Xilinx tools
- Working with Git and Vivado
- Nuclear and related instrumentation
- A lot more…

Thank you!