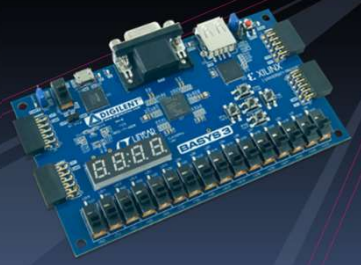
LAB 3

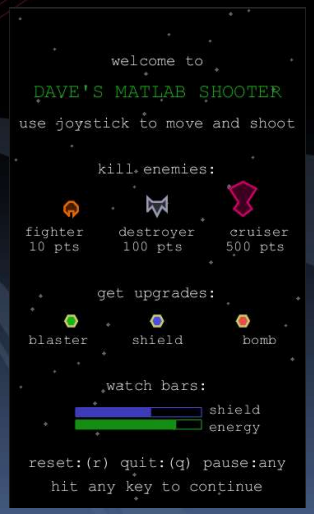
Exercise Goal

Using the AXI4-Stream RS232 module, and Pmod_JSTK2 build a communication between joystick and MATLAB "Shooter" game.

SPI



RS232



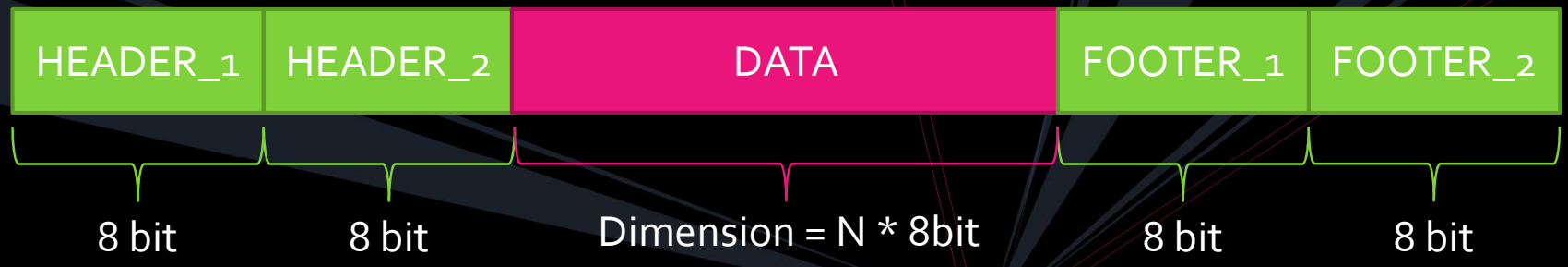
Exercise Goal

You have to manage data to and from MATLAB:

- MATLAB -> FPGA: RGB LED control, the game will send a value proportional to the remain energy of the gun ship
- FPGA -> MATLAB: You have to send the position of joystick axis (used to move the ship) and the trigger button value (used to shoot with the gun)

Packet specification

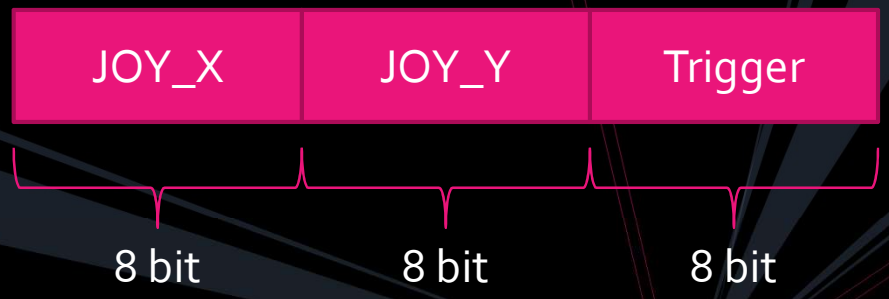
All packet, in both directions, must have this stucture:



Remember to check the correctness of received packet (check header and footer) before updating the led status

Ship Control

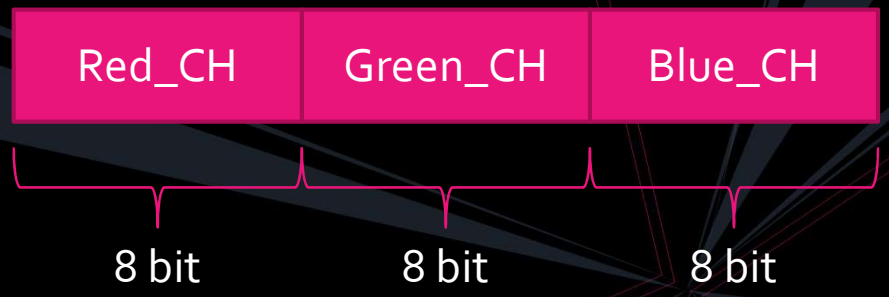
Send to MATLAB a Packet with this DATA:



- JOY_X and JOY_Y: [0, 255] take the most significant bit from Pmod_JSTK2 module
- Trigger: [0,1] 1 if button pressed else 0

LED Control

Received DATA from MATLAB and drive LED according to this specifications:



Top Prototipe

Use this prototipe (also avaiable on piazza) for your top file:

```
library IEEE;
   use IEEE.STD_LOGIC_1164.ALL;
   use IEEE.NUMERIC_STD.ALL;

entity joyToSerial is
   generic(
      SEND_RATE     : INTEGER := 20; -- Send Data x times per second
      HEADER_1      : STD_LOGIC_VECTOR := x"FE";
      HEADER_2      : STD_LOGIC_VECTOR := x"B0";
      FOOTER_1      : STD_LOGIC_VECTOR := x"0B";
      FOOTER_2      : STD_LOGIC_VECTOR := x"EF"
   );
   Port (
      reset  : in std_logic;
      clk    : in std_logic;  --100MHz

      RS232_TX : OUT STD_LOGIC;
      RS232_RX : IN STD_LOGIC;

      SPI_io0 : OUT STD_LOGIC;
      SPI_io1 : IN STD_LOGIC;
      SPI_sck : OUT STD_LOGIC;
      SPI_ss : OUT STD_LOGIC
   );
end joyToSerial;
```

Pmod_JSTK2 Instantiation

The Pmod_JSTK2 module is build for both master and slave interface, you must use the master one. Here you can find an example:

```
PmodJSTK2_0_inst : Pmod_JSTK2_0
        PORT MAP (
            SPI_io0_i  => '0',
            SPI_io0_o  => SPI_io0,    -- MOSI (Output)
            SPI_io0_t  => open,
            SPI_io1_i  => SPI_io1,    -- MISO (Input)
            SPI_io1_o  => open,
            SPI_io1_t  => open,
            SPI_sck_i  => '0',
            SPI_sck_o  => SPI_sck,    -- SCLK (Output)
            SPI_sck_t  => open,
            SPI_ss_i   => "0",
            SPI_ss_o   => SPI_ss_vect,    -- SS (Output)
            SPI_ss_t   => open,
            clk        => clk,
            jstk_btn   => jstk_btn,
            jstk_x     => jstk_x,
            jstk_y     => jstk_y,
            led_b      => led_b,
            led_g      => led_g,
            led_r      => led_r,
            out_valid  => out_valid,
            reset      => reset,
            trigger_btn => trigger_btn
        );
```

Modularity

Write and build your project using the modularity: write two modules, one for building ship movment data packet and another for elaborating the incoming packets for LED driving.

How to connect the Joystick

Connect the Joystick with the provided cable paying attention to the VCC and GND position