ESIstream

The **E**fficient **S**erial **I**nterface

PACKAGE ESISTREAM XILINX XCKU060-FFVA1517-1-C

Getting started with ADA-SDEV-KIT2

01/02/2022

Purpose of this document

Explain how to use the Python API of the ESItream package developped for ADA-SDEV-KIT2 FPGA carrier board.

Materials:

- Alpha-data ADA-SDEV-KIT2
- EV12AQ600-FMC-EVM or a loopback board (XM107 or equivalent)
- XM105 Debug Board.
- USB to UART FTDI cable : FTDI TTL-232RG-VREG1V8-WE

UART, write

- The design embeds a UART slave which uses the following configuration:
  - Baud rate: 115200
  - Data Bits: 8
  - No parity

The UART frames layer protocol defined here allows to perform read and write operations on the registers listed in the register map (see user guide or vhdl sources).

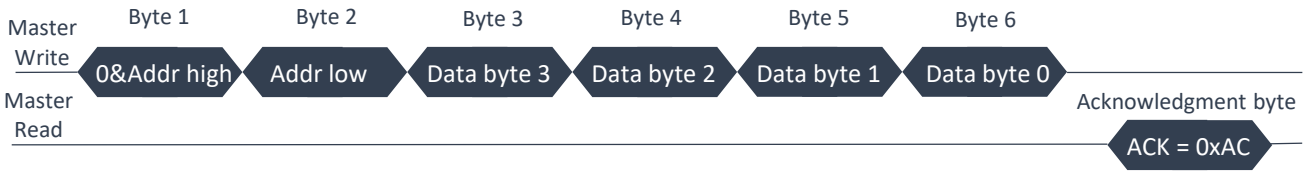
- Register Write operation:

The UART master must send the data in the order described on the figure below to be able to write a register.

Firstly, the master send the 15-bit register address and then the 32-bit data word.

- The most significant bit of the first transmitted byte (bit 7) must be set to 0 for write operation.
- The bits 6 down to 0 of the first transmitted byte contain the bit 14 down to 8 of the register address.
- The second byte contains the bit 7 down to 0 of the register address.
- The third byte contains the bit 31 down to 24 of the register data.
- The fourth byte contains the bit 23 down to 16 of the register data.
- The fifth byte contains the bit 15 down to 8 of the register data.
- The sixth byte contains the bit 7 down to 0 of the register data.

Finally, the master read the acknowledgment word to check that the communication has been done correctly. The acknowledgment word is a single byte of value 0xAC (172 is the decimal value).



UART, read

- The design embeds a UART slave which uses the following configuration:
  - Baud rate: 115200
  - Data Bits: 8
  - No parity

The UART frames layer protocol defined here allows to perform read and write operations on the registers listed in the register map (see user guide or vhdl sources).

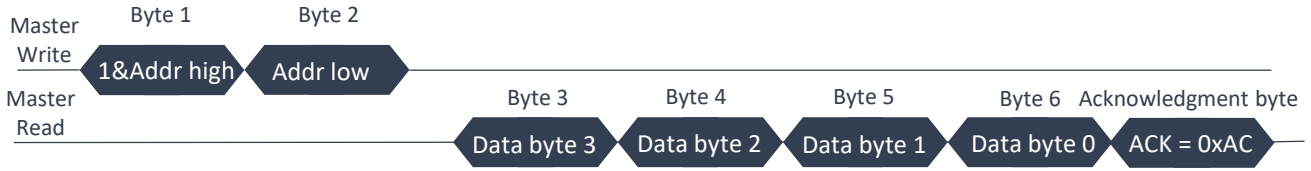
- **Register Read Operation**

The UART master must send the data in the order described on the figure below to be able to read a register value. Firstly, the master send the 15-bit register address.

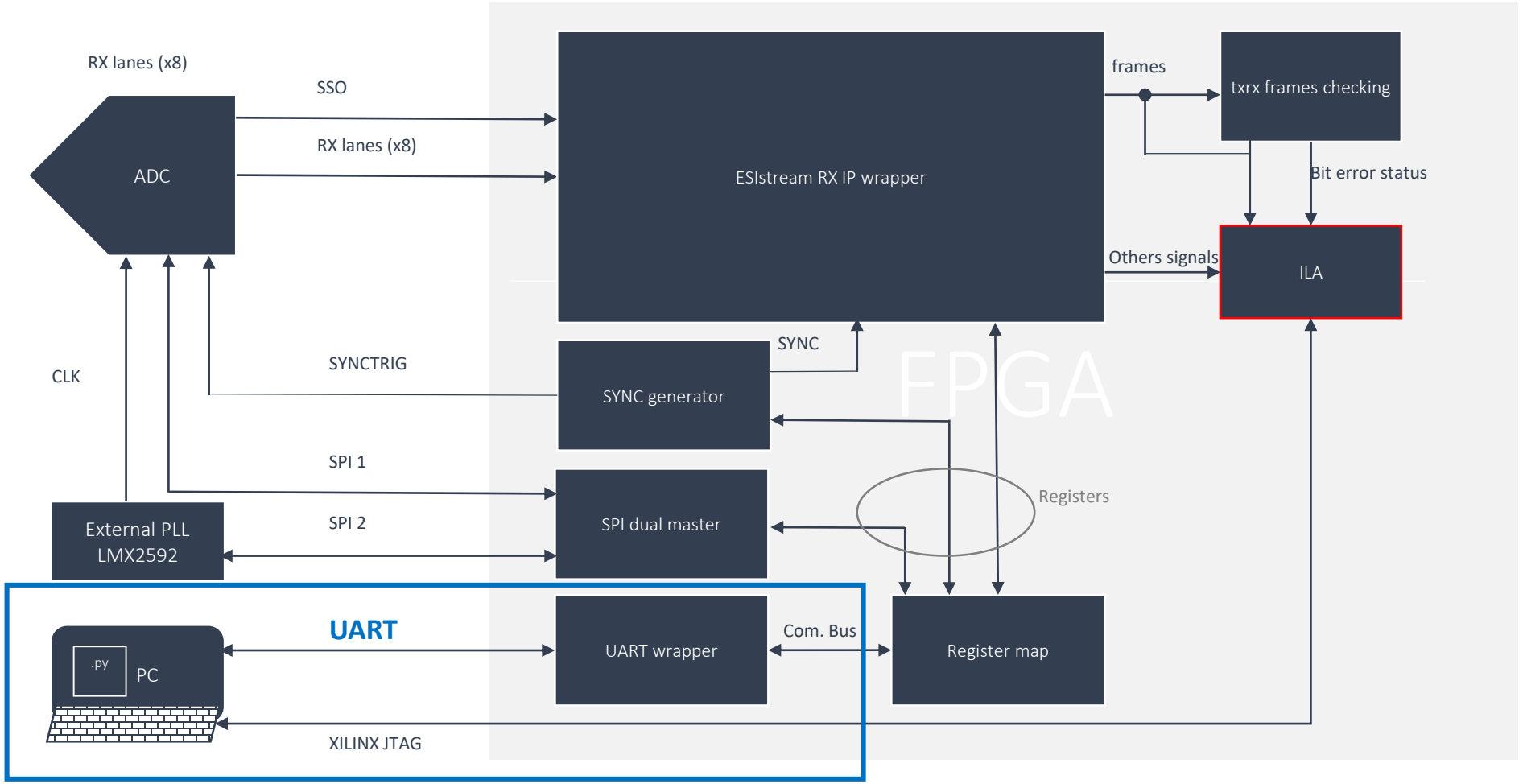
- The **most significant bit of the first transmitted byte (bit 7) must be set to 1 for read operation.**
- The bits 6 down to 0 of the first transmitted byte contain the bit 14 down to 8 of the register address.
- The second byte contains the bit 7 down to 0 of the register address.

Then, the master read the data and the acknowledgment word to check that the communication has been done correctly. The acknowledgment word is a single byte of value 0xAC (172 is the decimal value).

- The third byte contains the bit 31 down to 24 of the register data.
- The fourth byte contains the bit 23 down to 16 of the register data.
- The fifth byte contains the bit 15 down to 8 of the register data.
- The sixth byte contains the bit 7 down to 0 of the register data.

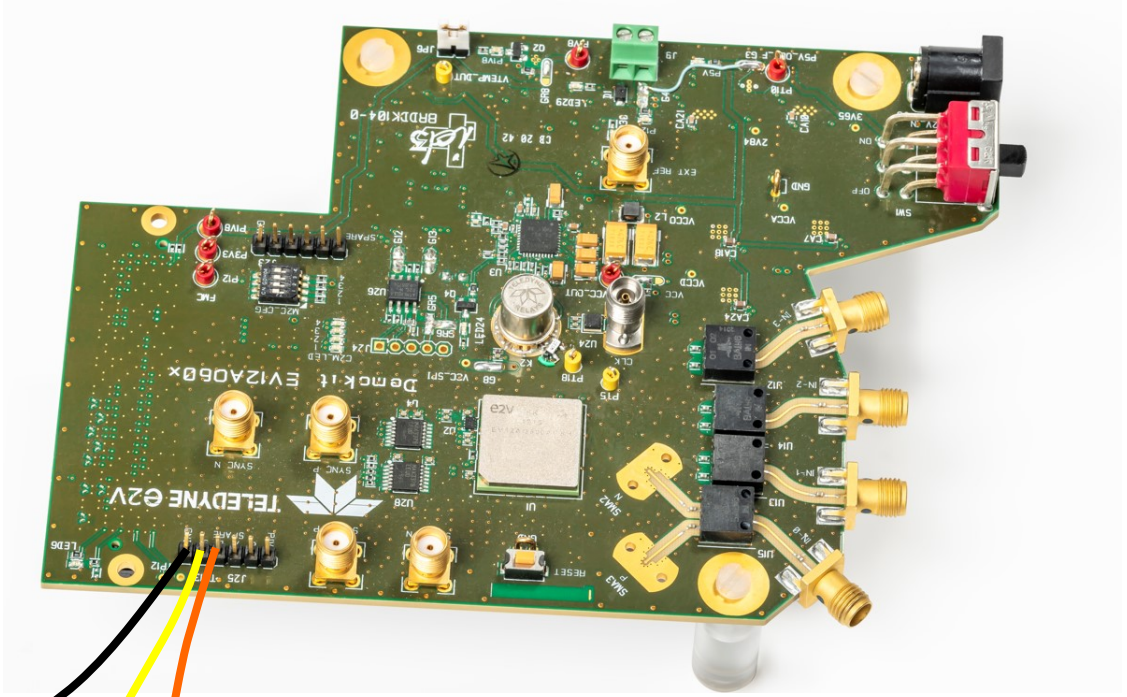


UART, FPGA PL architecture block diagram



Package project: *vivado rx ev12aq60x* UART

- Orange wire: Spare 8 UART TX
- Yellow wire: Spare 9 UART RX
- Black wire: GND

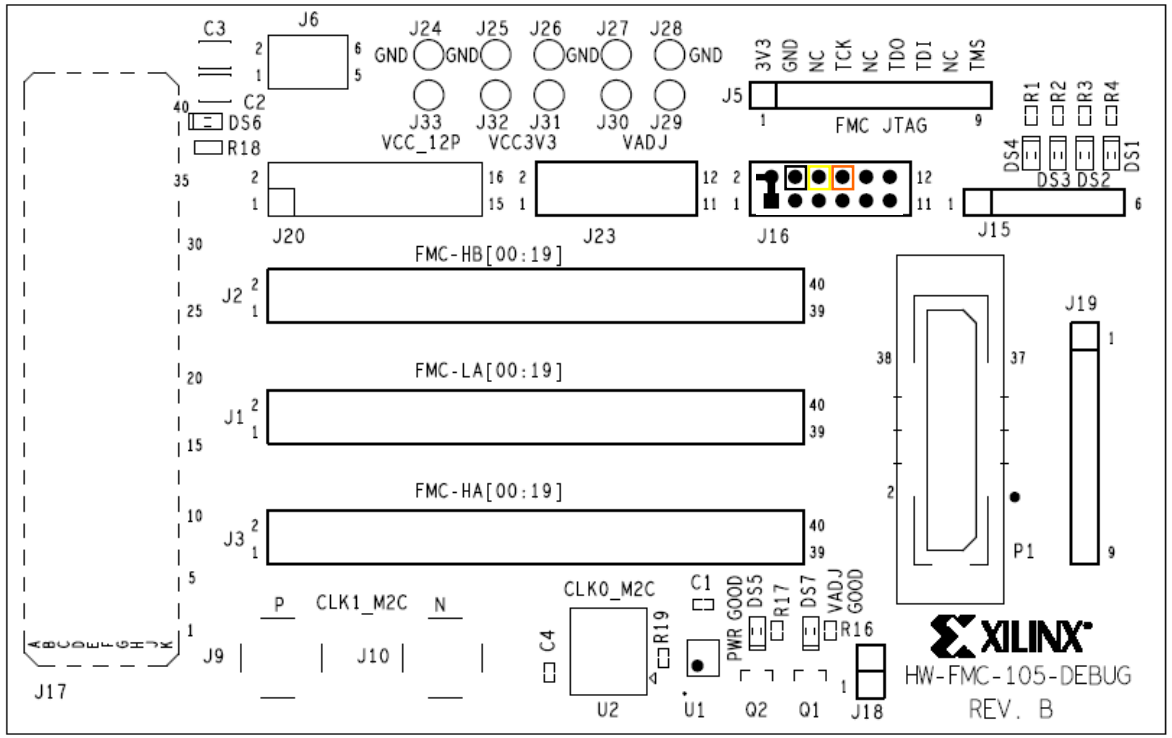


Colour	Name	Type	Description
Black	GND	GND	Device ground supply pin.
Brown	CTS#	Input	Clear to Send Control input / Handshake signal.
Red	VCC	Output or input	Power Supply Output except for the TTL-232RG-VIP-WE where this is an input and power is supplied by the application interface logic.
Orange	TXD	Output	Transmit Asynchronous Data output.
Yellow	RXD	Input	Receive Asynchronous Data input.
Green	RTS#	Output	Request To Send Control Output / Handshake signal.

FTDI Chip

- FTDI TTL-232RG-VREG1V8-WE

Package project: vivado_rx_xm107
UART using XM105 debug board



FTDI Chip

- FTDI TTL-232RG-VREG1V8-WE

Colour	Name	Type	Description
Black	GND	GND	Device ground supply pin.
Brown	CTS#	Input	Clear to Send Control input / Handshake signal.
Red	VCC	Output or input	Power Supply Output except for the TTL-232RG-VIP-WE where this is an input and power is supplied by the application interface logic.
Orange	TXD	Output	Transmit Asynchronous Data output.
Yellow	RXD	Input	Receive Asynchronous Data input.
Green	RTS#	Output	Request To Send Control Output / Handshake signal.

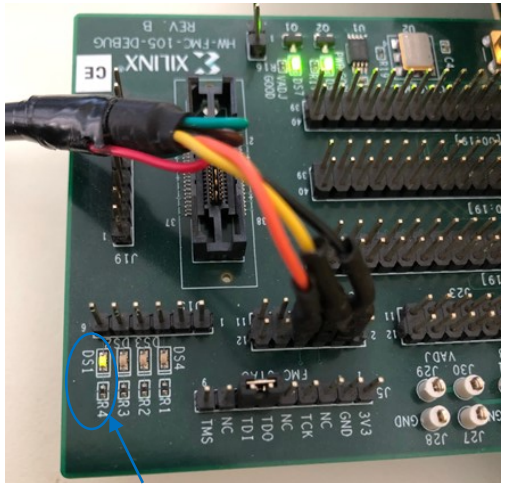
Package project: vivado_rx_xm107
UART using XM105 debug board

USB cable - SYSMON
to force VADJ to 1.8V
on FMC1 (J1)

JTAG to program FPGA

USB / UART FTDI cable

Jumper for
JTAG chain loopback of
TDI to TDO through FMC1 (J1)



After FPGA bitstream is loaded

DS1 LED should be ON indicating UART is ready

```
C:\Windows\System32\cmd.exe
C:\Users\sbreysse\Desktop\Package_ESIstream_Xilinx
_xcku060-ffva1517-1-c-v3.0-main\python\api>python
get_status.py


------------------------------------------------
------
-- Serial communication opened... True
-- Status: 0x20152018
-- HDL version: 0x300
-- Get Chip ID (2324):
addr = 17
0
-- (e) Chip ID error
-- Serial communication closed... True
------------------------------------------------
```

Alpha-data ADA-SDEV-KIT2 Hardware setup procedure

- ❑ Connect alpha-data config FMC board on alpha-data base board FMC J2
  - ❑ Connect micro USB cable to sysmon connector J3
  - ❑ Connect Xilinx platform cable USB II JTAG cable to J2
- ❑ Connect EV12AQ600-FMC-EVM board or XM107 board on alpha-data base board FMC+ J3
  - ❑ Connect UART FTDI cable on EV12.
  - ❑ Check power switch is OFF
  - ❑ Connect power cable +12V/5A.
- ❑ Check alpha-data base board switch SW3 is OFF
- ❑ Connect CX650M power supply connector on alpha-data base board connector J5
  - ❑ Switch ON CX650M rear panel power button.
  - ❑ Check alpha-data base board D2 and D1 leds are respectively RED and GREEN.
    - ❑ If yes, switch ON SW3.
    - ❑ Else if, D2 and D1 are OFF there is probably a short somewhere. **Do not switch ON SW3, contact support.**

Alpha-data ADA-SDEV-KIT2 Hardware setup procedure

- Open Vivado hardware manager and program the FPGA.
- Switch ON EV12AQ600-FMC-EVM board else if XM107 board, nothing to do.

Python setup

- Go to package directory /python/api/
- Open a cmd prompt:
  - Launch python scripts:

    > python get_status.py

    > python hw_config.py

    > python ev12aq600_external_pll_6250.py

    > python link_config.py

- Refresh FPGA in Vivado hardware manager to use Vivado Integrated Logic Analyzer.