

ESIstream

The Efficient Serial Interface

ESISTREAM 62B64B

Get started with KCU105 loopback package

Software setup

- ❑ Download and install Python Environment:
 - ❑ Python 3.x.x: [Download Python | Python.org](#)
 - ❑ Download and install Pyserial library: [pyserial · PyPI](#)
- ❑ Download and install [Vivado 2019.1](#) (Design suite) or latest, with Kintex Ultrascale FPGA family libraries. If latest version, you must upgrade Vivado IPs.
- ❑ [KCU105](#) Evaluation kit USB to UART:
 - ❑ Board [user guide](#).
 - ❑ Download and install Silicon Labs USB to UART [CP210x Virtual COM port \(VCP\) drivers](#).

Hardware setup

Ordering information

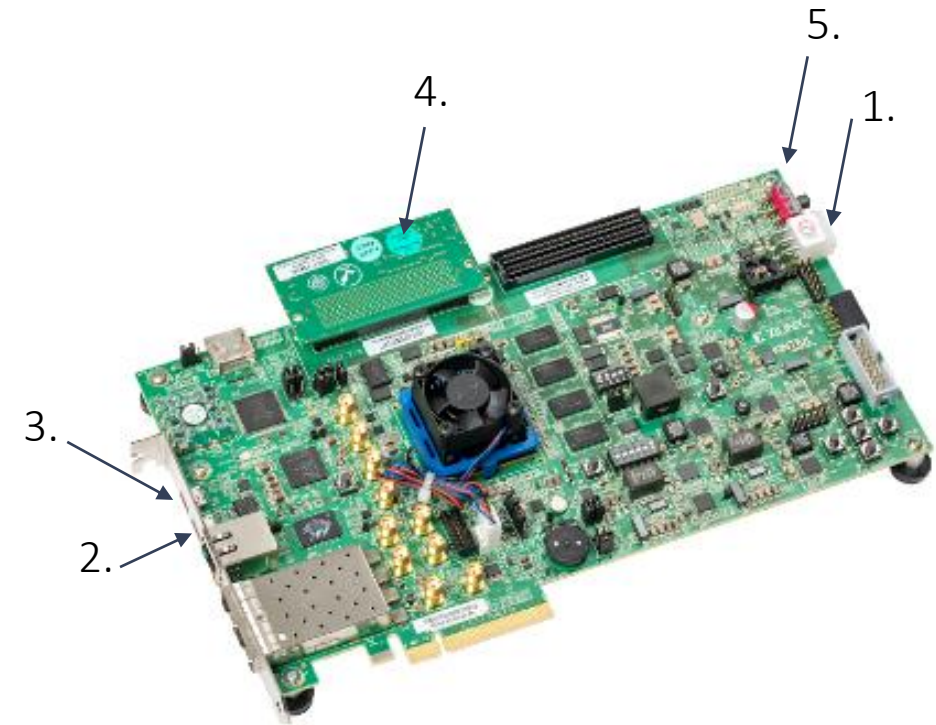
- ❑ [KCU105](#) Evaluation kit: [EK-U1-KCU105-G by AMD Xilinx Evaluation & Development Kits | Avnet](#)
- ❑ FMC loopback board provided with KCU105 Evaluation kit.



Hardware setup

Procedure

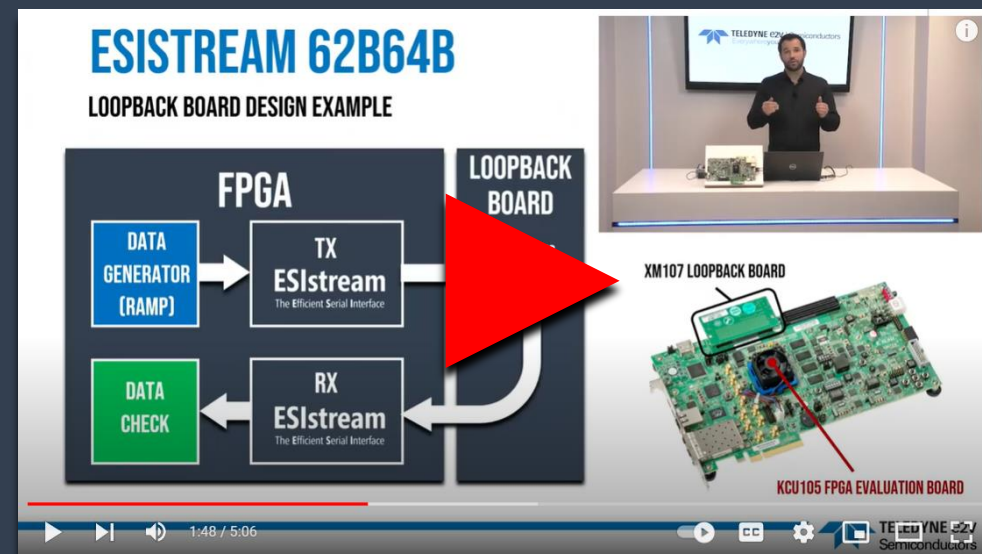
1. Connect power cable to KCU105
2. Connect USB-JTAG cable between KCU105 and PC
3. Connect USB-UART cable between KCU105 and PC
4. Connect FMC loopback board on KCU105 J22 FMC connector
5. Power on KCU105 switching SW1



Get started with KCU105 loopback board package

Tutorial

- ☐ To perform the following tasks...:
 - ☐ Download the package
 - ☐ Create vivado project
 - ☐ Generate bitstream
 - ☐ Load bistream in the FPGA using JTAG
 - ☐ Synchronize ESistream 62B64B High-speed serial link interface using cmd window and python script.
- ☐ ... check the youtube tutorial video starting at 1:35.



UART - FPGA interface protocol

Write command

- 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.

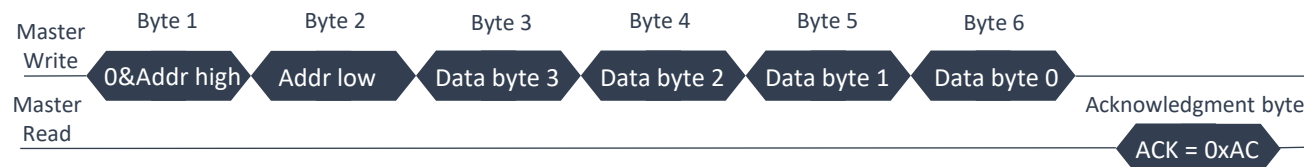
Register Write operation:

The UART master must send the data in the order described on the figure below to be able to write a FPGA 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 - FPGA interface protocol

Read command

- 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.

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.

