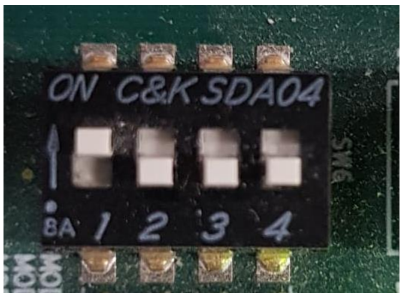
RFSoC BP Change User Guide

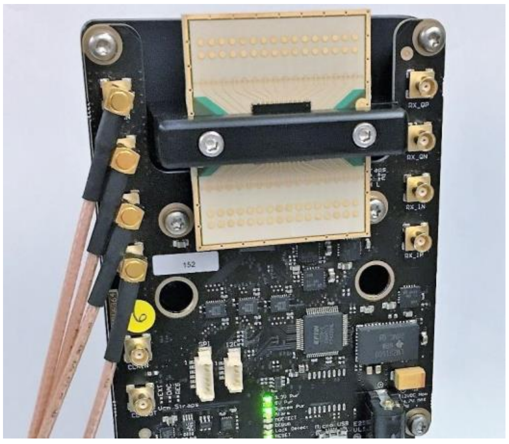
RFSOC PREPARATIONS

1. Download the BOOT.bin file and save it at the SD card.
2. Set Switch SW6 pins to 1110(OFF, OFF, OFF, ON) to configure the RFSoC in SD boot mode.

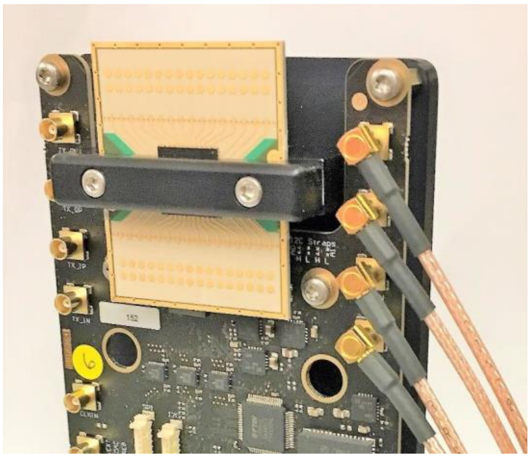


SW6 Configuration

3. Assign a static IP address in the host machine.
   IP address: 192.168.1.3
   Subnet mask: 255.255.255.0
   Default gateway: 192.168.1.1
4. Plug an ethernet cable from the FPGA to the PC Host.
5. Plug the 4 MMCX to SMA-F cables into I_N, I_P, Q_N and Q_P for the receiver and transmitter antenna respectively.

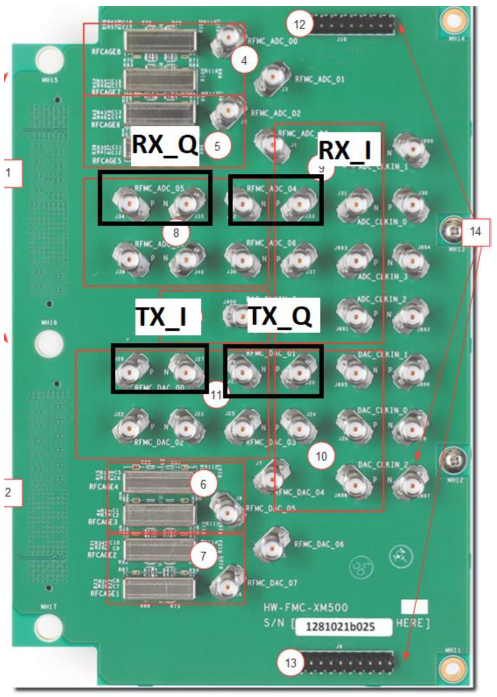


Connection for TX antenna



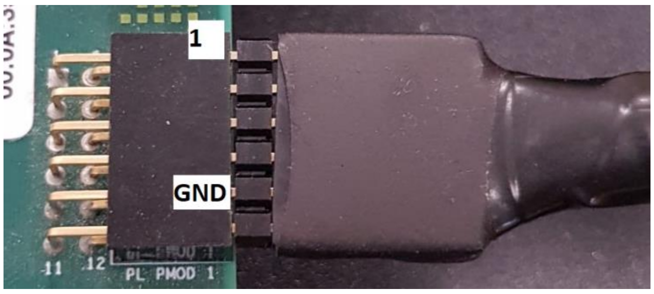
Connection for RX antenna

6. Use the SMA-M/SMA-M cables to connect between the FPGA converters to the antennas. See the figure to match the connections.



7. XMC500 Daughterboard

8. Plug the BP change cable (check section about how to build the cable) into the upper row of the PMOD_1 connector (see figure). Make sure you connect it in the right direction.



PMOD to FPGA

9. Turn on the FPGA

SIVERS SOFTWARE ENVIROMENT

This section was directly extracted from the *User Manual EVK06002/0.* You can find this document within the installation files from the stfp repository.

System requirements

• A computer with USB interface and Linux Ubuntu 16.04 or later. Equivalent Linux distribution might also work.
• Python 2.7 or later

Installation instructions

1. Connect the USB to Micro USB cable to the computer

2. Connect the Power adapter to EVK06002/00 and a Wall Wart.

3. Download software package from sftp.

    IP Address: 193.104.100.110
    Use an stfp-ssh client:
    - Host: 193.104.100.110
    - Port: 22
    - Protocol: Choose "SFTP – SSH File Transfer Protocol"
    - User: Will be sent after NDA is signed
    - Password: Will be sent after NDA is signed

4. Open Linux command window.

5. From root create a new directory.

```
> mkdir ~/evk06002/
```

6. Copy ederenv_<date>_<time stamp>.tar.gz to the newly created evk06002 directory.

```
> cp ~/media/your-usb/ ederenv_<date>_<time stamp>.tar.gz ~/evk06002/
```

7. Unpack the Eder software environment.

```
> tar -zxvf ederenv_<date>_<time stamp>.tar.gz
```

8. Run Eder install script.

```
> ./install_mb1.sh
```

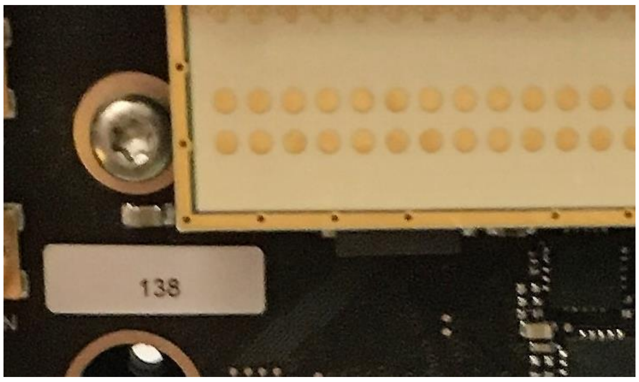
9. Enter password as administrator if prompted.

10. Install the newest ederftdi library to manage the source of SPI signals. To do that follow the information from the file *how_to_install_EderFtdi.txt*

11. At this point it's possible to choose to run either Eder command line shell or Eder GUI

12. For Eder command line shell run start script mb1. Script shall contain serial number located at the motherboard PCB Sivers SN motherboard ID sticker,

```
> ./start_mb1.sh SN<device serial number>
```



EVK Serial Number

13. Copy 'CONFIG_TX.PY' and 'CONFIG_RX.PY' from the IMDEA repository to the folder *./Eder_B* inside the framework. These two scripts will prepare the RX and TX antennas respectively to send/receive data and change BPs.

```
> execfile('CONFIG_TX.PY')

> execfile('CONFIG_RX.PY')
```

14. In order to enable/disable the antennas, use this functions. Take into account that there are different commands for RX and for TX.

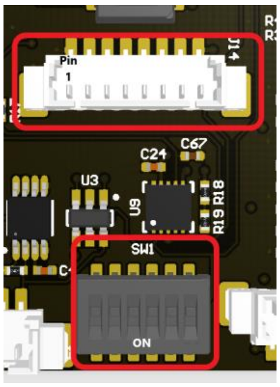
```
> eder.rx.enable()

> eder.rx.disable()

> eder.tx.enable()

> eder.tx.disable()
```

14. Once you configured the antenna, send this command to disable commands from USB and to manage the antennas via external SPI signals.

```
> eder.ederftdi.spioff ()
```

Connect BF control cable ---- GPIO

Move pin 3,4 and 5 switches to ON (low) to allow external BP changing. Plug the BF control cable to J14.



BF GPIO Switch and connector

MATLAB CONTROL FUNCTIONS

- **<ts> = system_init (ip)**: configures the TCP/IP stack in the host PC and opens a connection with platform.
  - **ip** -> platform IP address.
  - **ts** -> tcp session identificator.

- **<pl> = transmit_1CH (ts, path , idle_time, SD):** sends the set of I/Q samples the user wants to transmit using the number of macro channels X, loads them and starts the transmission synchronously in all enabled channels.
  - **ts** -> Tcp session identificator.
  - **path** -> location of the samples to transmit**.**
  - **idle_time** -> samples between packets.
  - **SD** -> (*true*: reads data from SD; *false*: reads data from PC Host via TCP/IP).
  - **pl** -> packet length.

- **capture_samples_1CH (ts,rx,nSamples,path,filename,SD):** enables the continuous capture of I/Q samples. The captured samples can be sent via TCP/IP or saved on an SD card (if present).
  - **ts** -> Tcp session identificator.
  - **rx ->** receiver bitmask. For SISO designs use '0001'
    - '0001' – stream 1
    - '0010' – stream 2
    - '0100' – stream 3
    - '1000' – stream 4
  - **nSamples ->** number of samples to store.
  - **path** -> location of the samples to store.
  - **filename** -> name of the file.
  - **SD** -> (*true*: reads data from SD; *false*: stores data at PC Host via TCP/IP).

- **capture_pkt_1CH (ts,tx,nPackets, pl,path,filename,SD)**: it captures a certain number of packets of a specific length. The captured packets are again sent to the host PC or saved on an SD card.
  - **ts** -> Tcp session identificator.
  - **path** -> location of the samples to store.
  - **nPackets ->** number of packets stored by packed detector.
  - **pl ->** packet length.
  - **filename** -> name of the file.

        - **SD** -> (*true*: reads data to SD; *false*: stores data at PC Host via TCP/IP).

- **configure_PD (ts, highTime)**: configures the settings for packet detector. Set this function before starting to capture packets.
    - **ts** -> Tcp session identificator.
    - **pl** -> length of the packet to be detected.

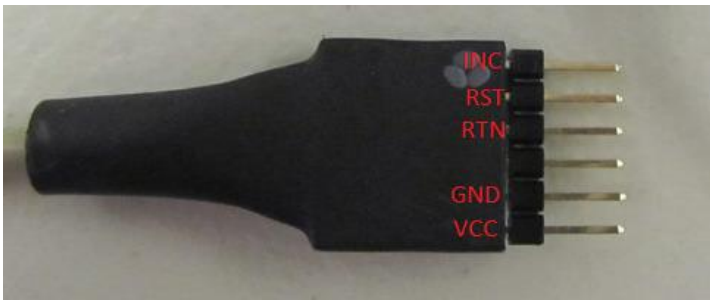
- **configure_AWV_control(ts, type, P, M, N, L, T_INIT, T_HIGH, enable):** manages the configuration of the GPIO pulses. Enable and disables the block.
    - **ts** -> Tcp session identificator.
    - **P**-> Number of cycles or the first TRN field (without BP change.
    - **M** -> Number of BP changes per P beam patterns.
    - **N** -> Number of clock cycles per each of the M BPs.
    - **L** -> Number of repetitions of P+M BPs.
    - **T_INIT** -> Number of delay cycles after the trigger before start with the BP P.
    - **T_HIGH** -> Number of clock cycles the INC, RTN and RST pulses stay high at the output.
    - **enable** -> (1: enable pulses; 0: disable pulses).

- **configure_BP_array (ts, enable, ptr, BP)**: configures the corresponding pointer of the set of arrays to change through SPI
    - **ts** -> Tcp session identificator.
    - **enable** -> enables or disables the change of BPs while capturing.
    - **ptr** -> pointer of the array.
    - **BP** -> Beam pattern assigned to the corresponding position of the array.
- **write_SPI (ts, addr, val, interface, ant)**: writes a value to a SPI register
    - **ts** -> Tcp session identificator.
    - **addr** -> address of the SPI slave registers.
    - **val** -> byte to write.
    - **interface** -> select between SPI drivers. For SISO designs use '0'.
    - **ant** -> select between SPI slaves. For SISO designs use '0'.

BUILD BP CHANGE CABLE ---- GPIO

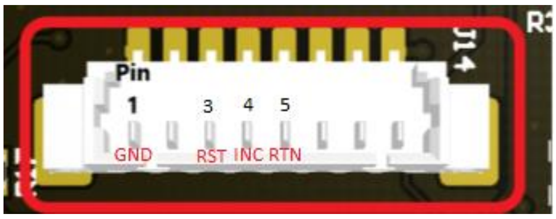
You need 4 pins for doing BP changes at the SIVERS kits. Increment BP index (INC), reset BP index (RST), return to default BP (RTN) and ground (GND).

1. Split the PMOD cable in two so that you have access to individual pins.



PMOD Connector

1. Identify which wire (colour) correspond to each pin. The easiest way is to use a polimeter to measure continuity.
2. Take 8-10 inches of 6 wire cable and strip both sides. You will only need 4 wires to cover your 4 signals.
3. Split the Molex cable in two as you did with the PMOD cable. Strip wires from pins 1, 3, 4 and 5 (see Molex figure).



Molex Connector

4. In order to connect everything, you can either solder each wire individual or plug it into an electrical terminal with screws. If you do not have experience with soldering it is suggested to follow the last approach.

Electrical Terminal

5. Follow the table to match each pin for both connectors.

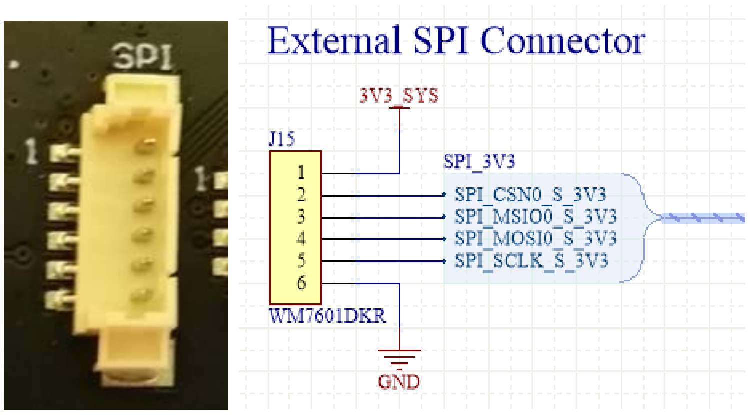
	PMOD Connector Pin	Molex Connector Pin
INC	1	4
RST	2	3
RTN	3	5
GND	5	1

BUILD BP CHANGE CABLE ---- SPI

There are 3 signals to manage the writing at the SPI registers of the antenna: CLOCK, MOSI (Master Output Slave Input) and SS (Slave Select).

1. Split the PMOD cable in two so that you have access to individual pins.
2. Identify which wire (colour) correspond to each pin. The easiest way is to use a polimeter to measure continuity.
3. Take 8-10 inches of 6 wire cable and strip both sides. You will only need 4 wires to cover your 4 signals.

4. Split the Molex cable in two as you did with the PMOD cable. Strip wires from pins 2,4, 5 and 6 (see Molex figure).



Molex Connector

5. In order to connect everything, you can either solder each wire individual or plug it into an electrical terminal with screws. If you do not have experience with soldering it is suggested to follow the last approach.

Electrical Terminal

6. Follow the table to match each pin for both connectors.

	PMOD Connector Pin	Molex Connector Pin
SS	1	2
CLOCK	2	4
MOSI	3	5
GND	5	6