Marble v1.4.1 Bring-up Procedure
Feb 2024

Getting Started

This guide is broken up into a couple of sections. The First being a one time setup required for the testing hardware and environment. The second section will outline how to run the automated testing of Marble boards.

This guide assumes you are using a raspberry pi 5 board with a working wifi connection. All appropriate accessories for the raspberry pi5 are assumed; power supply, keyboard, mouse, monitor, monitor cable.

One-Time Machine & Environment Setup

- New Raspberry Pi 5 (RBPi) with default 64-bit OS install using the Official Raspberry Pi Imager [link]
- Insert the newly created SD card and power up the RBPi
- Install Dependencies
    - open terminal

    $ sudo apt install build-essential git gcc-arm-none-eabi
      python3-serial iverilog ftdi-eeprom openocd

        - git, build-essential, python3-serial are currently included in base image
    - close terminal

- Clone two repositories from github
    - open terminal

    $ cd ~
    $ mkdir Repositories
    $ cd Repositories
    $ git clone https://github.com/osprey-dcs/Marble-MMC.git
    $ git clone https://github.com/osprey-dcs/Bedrock.git

- Make necessary binaries

    $ cd ~/Repositories/Bedrock/badger/tests && make udprtx
    $ cd ~ && mkdir bin && cd ~/Repositories/Bedrock/badger/tests && \
      cp udprtx ~/bin/

- Logout and Back in

This is to have user bin folder added to path

- Add the include bitfile with this doc to the ~/Repositories folder
    - Alternatively this can be found here (requires lbnl login): https://gitlab.lbl.gov/hdl-libraries/bedrock/-/jobs/105791/artifacts/file/projects/test_marble_family/marble2.b513a799.bit
    - Alternatively this can be found on a google drive here: https://drive.google.com/file/d/19y7nvDCZjkqpGflG6Oi0jeYxVZ_3inEb
- Pick IP & MAC address range of cards to be tested
- Modify scripts to use you preferred ip and mac ranges

*Note: these can be changed after the initial set-up, but it may be nice to get it set properly now.

```
$ nano ~/Marble-MMC/scripts/config_ip_mac.sh
```

Change lines 42&43 from:

```
mac=$(printf "12:55:55:0:1:%x" "$snum")
ip=$(printf "192.168.19.%s" "$snum")
```

To [note using this change as an example]:

```
mac=$(printf "12:55:55:0:1f:%x" "$snum")
ip=$(printf "192.168.79.%s" "$snum")
```

*Note the last part of the IP and MAC will be set as a serial number assigned to the board

- Setup ethernet interface for local connections to the IP range you chose in the last step. An example is shown below
    - edit wired connection 1 settings
    - go to ipv4 settings
    - change method from Automatic(DHCP) to Manual
    - 192.168.79.100 /24
    - This can be done many ways including desktop gui, nmtui, edit /etc/dhcpcd.conf, etc.

Start of testing

Hardware required:
Power supply +12 V, 2 A
Cables: RJ45 Cable, USB A to Micro B (Qty:2)
Multimeter
J-Link EDU Mini - https://www.segger.com/products/debug-probes/j-link/models/j-link-edu-mini/

Optional hardware for additional tests:
IAM FMC loopback modules (Qty:2) - https://www.tindie.com/products/iam_electronic/fpga-mezzanine-card-fmc-loopback-module/
QSFPs (Qty:2) with loopback fiber cables
Digilent 8 LED PMOD board https://digilent.com/shop/pmod-8ld-eight-high-brightness-leds/

Steps 1 will need to be repeated for each test batch, or if the raspberry pi is power cycled.

1) Export environment variables used by scripts.

   A file 'Marble_Env.txt" has been provided on the RBPi Desktop where these commands can be copied from.

   $ export BEDROCK_PATH=~/Repositories/Bedrock

   $ export MMC_PATH=~/Repositories/Marble-MMC

   $ export LTM_SCRIPT=$MMC_PATH/LTM4673_reglist.txt

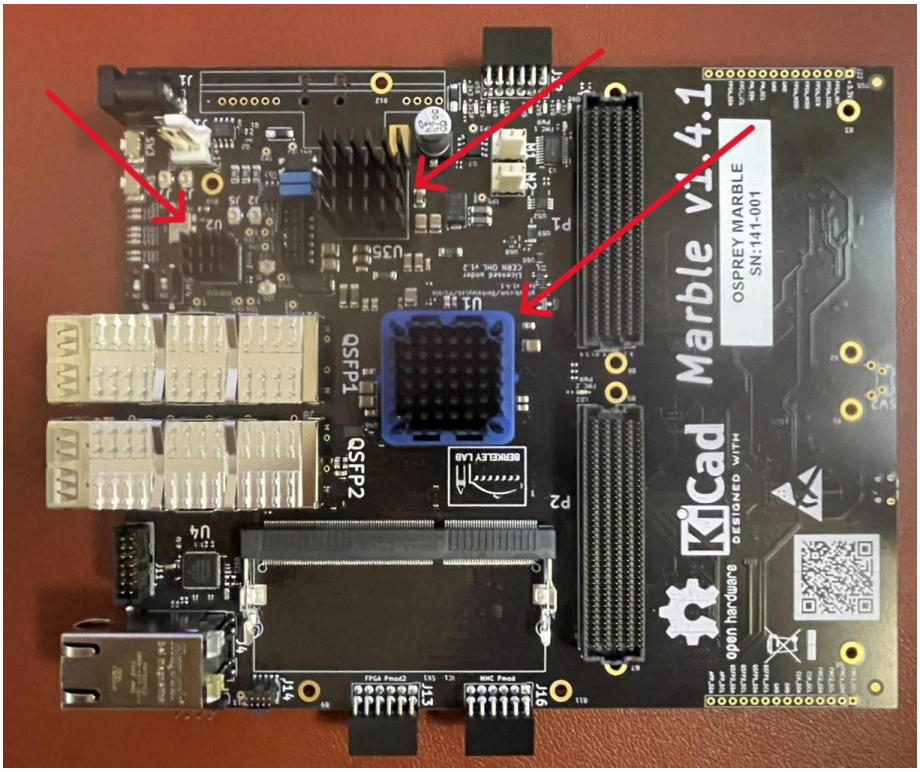
   $ export BITFILE=~/Repositories/marble2.b513a799.bit

   [this is an example .bit file used above. Use tab auto complete if entering this manually to ensure correct hex code in the bitfile's name]

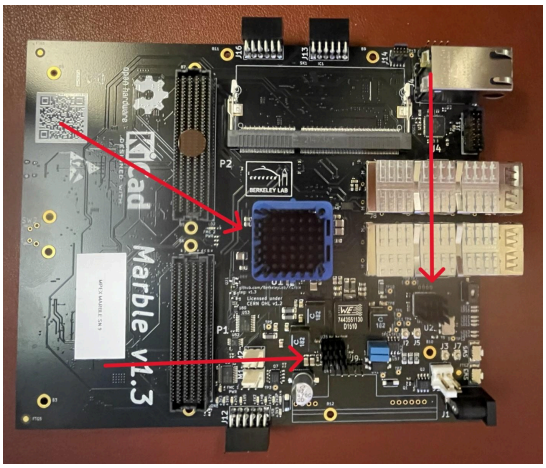
Step 2 starts the beginning of individual board testing. The following steps apply to a single Marble Board and will be repeated with each Marble Board to be tested.

2) Print out and add serial number labels Marble boards to be tested.
3) Apply any heatsinks to the Marble Boards to be tested [optional for now]
   a) 1 x big, on FPGA (U1) [PN:Cool Innovations 4-101011P]
      i) Old Heatsink was[PN:Wakefield-Vette 904-27-2-23-2-B-0] with a thermal pad [PN:t-Global Technology TG-T1000-28-28-0.25-5PT]
   b) 1 x medium, on power management chip (U35) [PN:Advanced Thermal Solutions ATS-55170R-C1-R0]
   c) 1 x small, on clock mux (U2) [PN:Watterott Electronic 20188]
   d) 5 x feet on back

   Version 1.4 heatsink locations (note heatsink orientation)

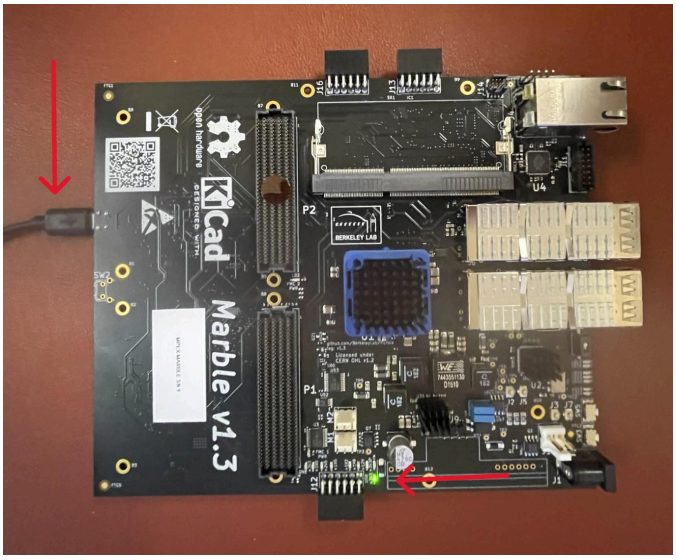


PREVIOUS VERSION v1.3 heatsink locations

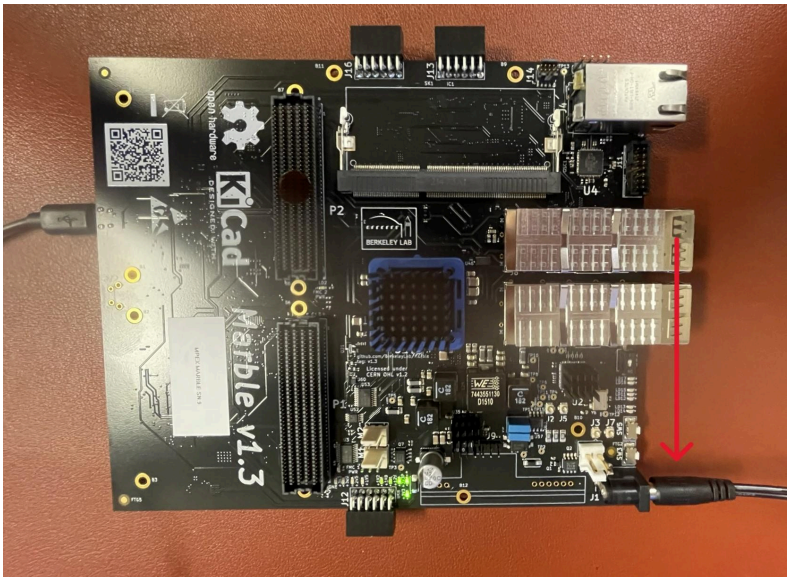


4) Measure resistance in ohms using a multimeter at power input J19
    a) This should read greater than 1.0Mohm. Stop testing of this board if shorted or low measure resistance.

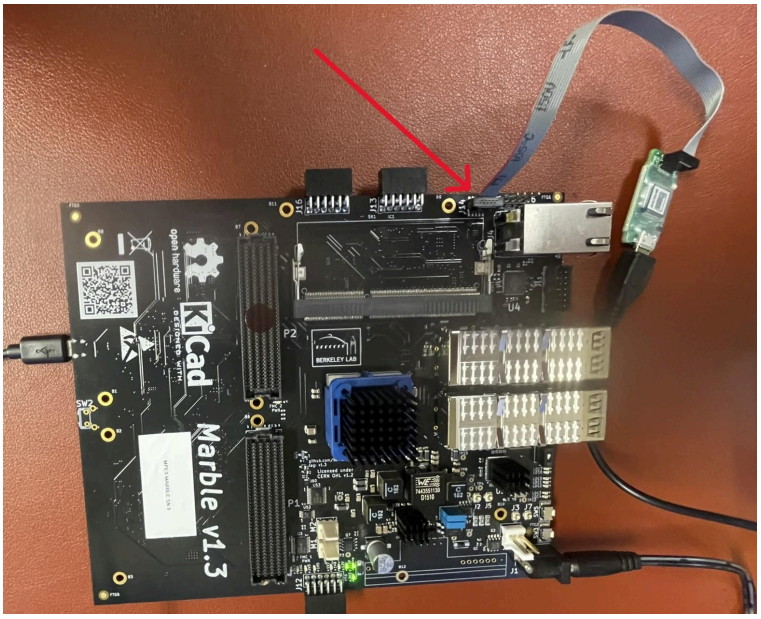
5) Connect USB micro-B cable to J10 - *observe LED USB +3V3 ON*



6) Using a 12V | 2A+ bench supply or power brick plug into J1 (barrel) - *observe some LEDs may come on and be flickering*



7) Attach J-Link Segger to J14



8) Run automatic marble bring-up script to program MMC and LTM4673(version 1.4+):
-open a terminal

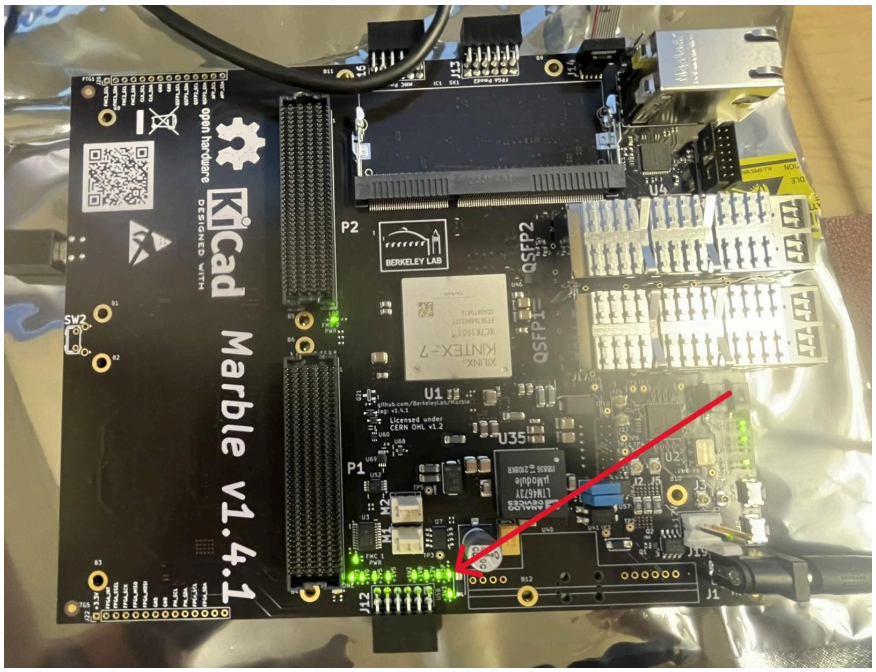
Version 1.4+

```
$ cd $MMC_PATH
$ ./bringup_mmc.sh
```

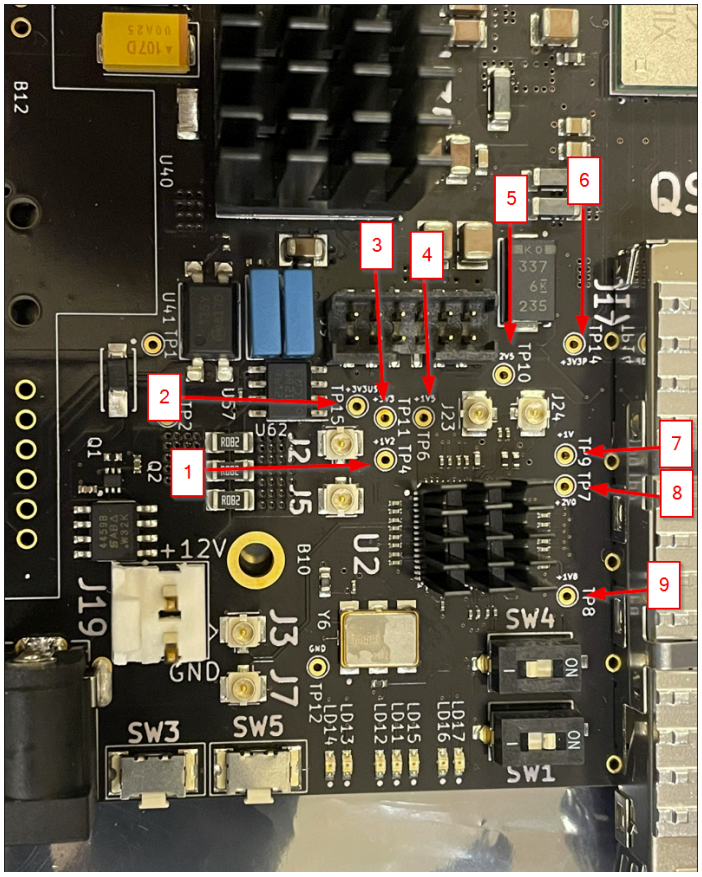
Previous Versions (ex: v1.3)

```
$ cd $MMC_PATH
$ ./bringup_mmc_noPMIC.sh
```

-Note (for V1.4+):10 LEDs (8 indicate power rails) near PMOD J12 should be ON



9) Measure voltage at locations below



Number 1 - (1.2V).
Number 2 - +3V3_USB (3.3V).
Number 3 - +3V3 (3.3V).
Number 4 - +1V5 (1.5V).
Number 5 - +2V5 (2.5V).
Number 6 - +3.3P always on (3.3V).
Number 7 - VCCBRAM (+1.0V).
Number 8 - VCCAUXIO2V0 (+2.0V)
Number 9 - VCCAUX (+1.8V).

10) Connect Ethernet (J4) to the configured RBPi's ethernet adapter- *observe amber link light should come ON*

11) Run automatic marble bring-up script:

In the following example the serial number is 48

```
$ export SN=48
$ export IP=192.168.79.$SN
$ cd $MMC_PATH
$ ./bringup.sh $SN
```

Script Includes:

Program FTDI EEPROM with the given serial number
Write IP and MAC addresses based on serial number
*In this example IP = 192.168.79.**48** | MAC = 12:55:55:0:1f:**48***
Load golden bitfile to FPGA through openocd
Read frequency counter outputs
Run stress test on Ethernet using udprtx
Record various device readouts and save it to a file using first_readout.sh
(three INA219 Voltage + current, SI570 output frequency, MAX6639 temperature in deg C, speed in rpm, Device DNA, and XADC)

This create a file within the folder title bringup_logfile_$SERIAL_NUMBER

12) Verify logfile has non-zero size

```
$ ls -l bringup_logfile_$SN
```

13) Set FPGA boot flash One Time Programmable bits

More instructions in bedrock/badger/flash.md

```
$ cd $BEDROCK_PATH/badger/tests
$ python3 -m spi_test --ip $IP --id
```

Check if TBPARM and TBPORT bits are set skip to step 14.

If not, run the following command…

```
$ python3 -m spi_test --ip $IP --config_init
```

Check if TBPARM and TBPORT bits are now set.

```
$ python3 -m spi_test --ip $IP --id
```

14) Burn projects/test_marble_family bitfile into address 0

make sure write protect switch (SW1) is off (see step 15 for location)

```
$ python3 -m spi_test --ip $IP --program $BITFILE \
  --force_write_enable
```

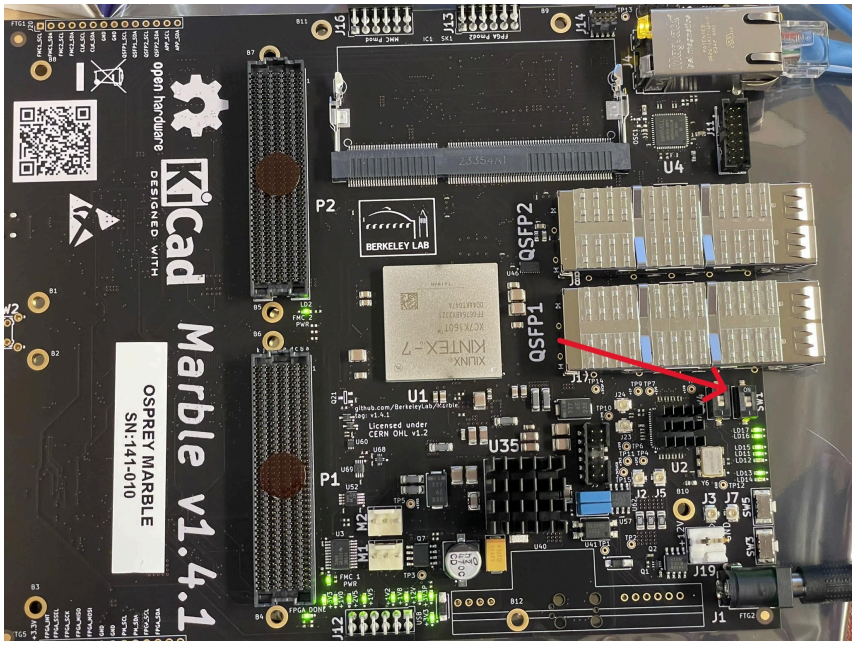
takes about 146 seconds

Now verify if the bitfile was successfully burnt using,

```
$ python3 -m spi_test --ip $IP --program $BITFILE --verify
```

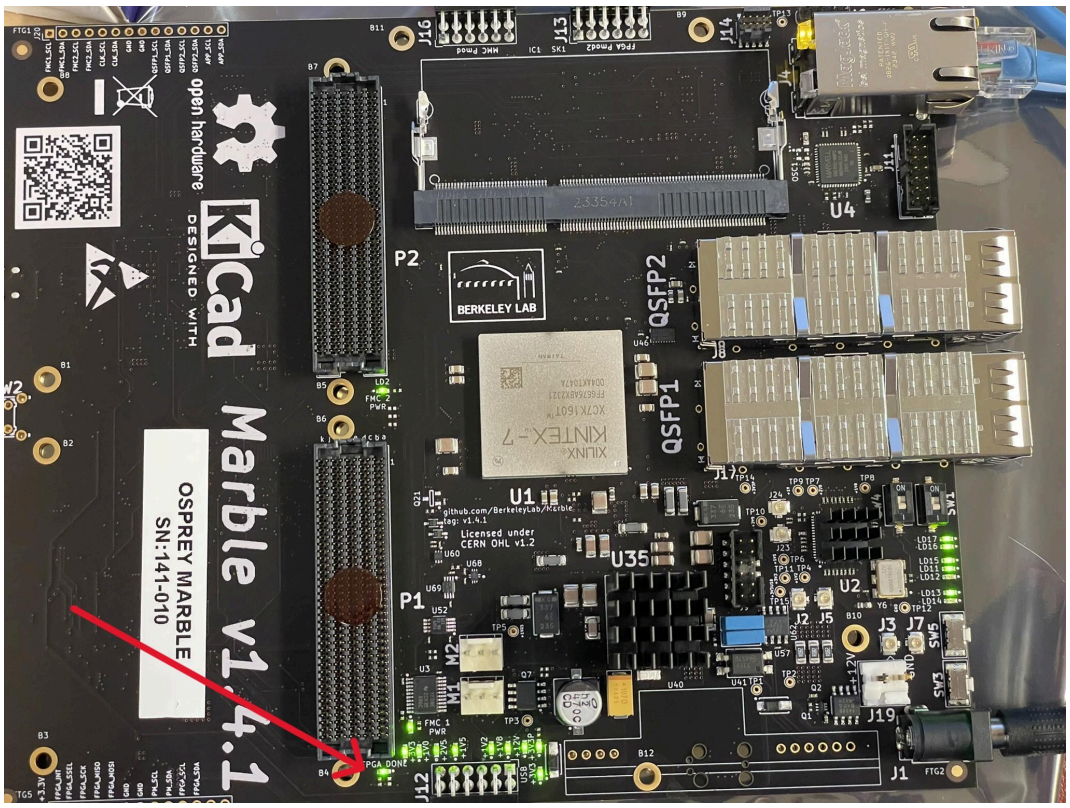
takes about 40 seconds

15) Turn on Write protect switch (SW1)



ADD PHOTO

16) Power cycle the board so it defaults to bedrock golden image. The FPGA_Done light should come on after ~2 sec.



Additional Functionality testing.

At this point you can return to step 4 with a new board, or continue with the following additional tests.

17) FMC I/O test

Attach two IAM FMC testers to P1 And P2 and run the following:

$ cd $BEDROCK_PATH/projects/test_marble_family

$ python3 -m fmc_test_iam -a $IP --plugged=12

Check if it passed (good output shown below),

leep://192.168.19.19:803

test_iam_fmc 240fb041573404326acf3d25618888

f4fb1b0c28 plugged=2

P1L .....

P2L .....

P2H .....

PASS

18) QSFP loopback test

Plug QSFP loopback module to QSFP1

Connect the FPGA JTAG programmer to J11

- Run command in terminal: **vivado**
- Go to Flow --> Open Hardware Manager --> Tools --> Auto Connect
- Click Tools --> Program Device --> xc7k160t_0 to open the programming window.
- Choose the ***marble_ibert.bit (included with these instructions)*** file and click Program

After successful programming, the Dashboard should start automatically.

Detect links by clicking Serial I/O Links --> Auto-detect links

Correctly detected and working links should report 10 Gbps status with no errors.

Close and open vivado again

Now connect the QSFP loopback module to QSFP2 connector and repeat the steps above.

19) DDR3 test

Power off the board and plug in the DDR3 memory to the slot

In a new terminal open up the FPGA UART…

$ miniterm /dev/ttyUSB2 115200

or

$ python3 -m serial.tools.miniterm /dev/ttyUSB2 115200

In a new terminal…….

$ cd $BEDROCK_PATH/projects/test_marble_family/

$ BITFILE=berkeleylab_marble.bit ./mutil usb

Look at the FPGA UART console, and check if the following is present

Switching SDRAM to hardware control.
Memtest at 0x40000000 (2.0MiB)...
Write: 0x40000000-0x40200000 2.0MiB
Read: 0x40000000-0x40200000 2.0MiB
Memtest OK
Memspeed at 0x40000000 (Sequential, 2.0MiB)...
Write speed: 112.6MiB/s
Read speed: 93.6MiB/s

Run the following command, to write and read 1GiB of memory

$ mem_test 0x40000000 0x40000000

check if it is successful and output should look like…

Memtest at 0x40000000 (1.0GiB)...
Write: 0x40000000-0x80000000 1.0GiB
Read: 0x40000000-0x80000000 1.0GiB
Memtest OK

20) FMC loopback transceiver test

Power off the board and plug in two IAM electronics FMC tester card to P1 and P2

Power on the marble board and *check if all the LEDs on the FMC testers are ON*

 # Change the GTX routing to have 2 channels on FMC P2 and 2 channels on FMC P1 using the following command…..

$ python3 -m load -d /dev/ttyUSB3 “r off”

$ mgtmux.sh -d /dev/ttyUSB3 M1

Follow the same steps as QSFP loopback testing and verify all the four channels are at 10 Gbps

 # Change the GTX routing to have 4 channels on FMC P2 using the following command…..

$ mgtmux.sh -d /dev/ttyUSB3 M2

Follow the same steps as QSFP loopback testing and verify all the four channels are at 10 Gbps.

Remove the JTAG programmer from J11 before continuing to the next step.

21) Enable mailbox

$ python3 -m load -d /dev/ttyUSB3 “r on”

22) Unplug all connectors and power from the marble. Box it up.

Useful commands:
Ctrl+r → reverse-i-search for recently used bash commands
ls → list current directory contents
ls -lt →list directory contents by modification time
cat → reads the file and prints its contents in the terminal. Eg: cat bringup_logfile_48
ifconfig → displays current network interface
ip addr → displays the addresses for every link configured on the system
sudo ip route add 192.168.19.0/24 dev enp4s0