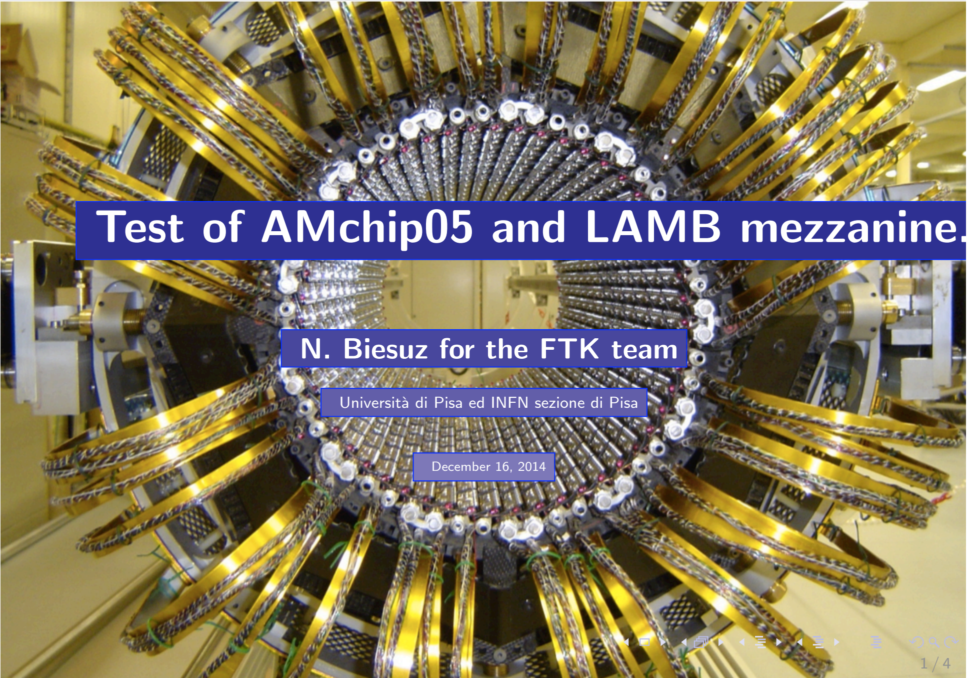


Test of AMchip05 and LAMB mezzanine.

N. Biesuz for the FTK team

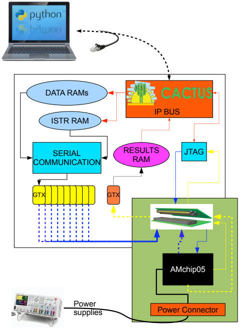
Università di Pisa ed INFN sezione di Pisa

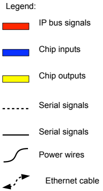
December 16, 2014

TEST on the AMchip05

- The latest prototype of AM chip named **AMchip05 has been delivered in July**.
  - → 3k pattern memory bank;
  - → low power consumption;
  - → ~~Full compatibility with AMchip06~~
    → pinout might change;
- The test **checked complete functionality of the chip**.
  - → Access all JTAG options;
  - → Verify the quality of links and jitter introduced in the daisy chain by the chip itself;
  - → Check patten matching performances (all AM cells are functional);
  - → Check functionality of Signal dedicated to daisy chain operation;

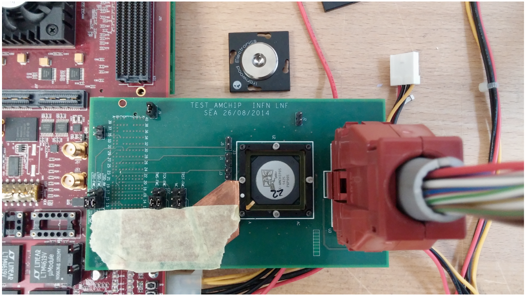
Test Setup





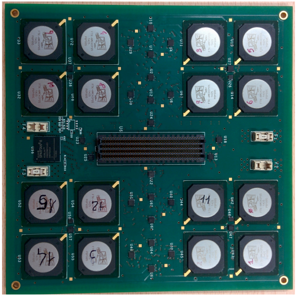
- The test are performed using Python scripts that "talk" to a FPGA Evaluation Board that sends signals to the chip.
- **The chip must be connected the Evaluation Board trough a mezzanine card**. (Printed Circuit Board with a ZIF socket)

Test Setup



- The test are performed using Python scripts that "talk" to a FPGA Evaluation Board that sends signals to the chip.
- **The chip must be connected the Evaluation Board trough a mezzanine card**. (Printed Circuit Board with a ZIF socket)
- We got few problems with the mezzanine so the test were delayed.

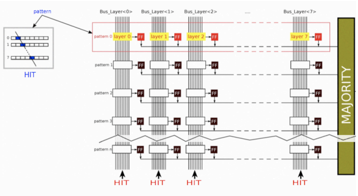
Test Setup



- The test are performed using Python scripts that "talk" to a FPGA Evaluation Board that sends signals to the chip.
- **The chip must be connected the Evaluation Board trough a mezzanine card**. (Printed Circuit Board with a ZIF socket)
- We got few problems with the mezzanine so the test were delayed.
- **We decided to mount chips directly on the LAMB**

Results

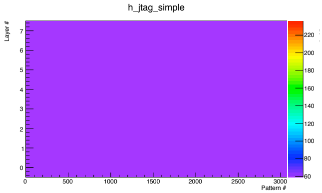
Got few issues that have already been solved!



- LinkError flag is stuck to 1 $\rightarrow$ The register containing the correct information was deleted after 1 clock cycle.
- Two pattern are over written when trying to access other addresses $\rightarrow$ Glitch on line decoder.
- Minimum supply voltage higher than nominal $\rightarrow$ Due to an IR drop along power lines inside the chip, now we can simulate it;

Results

Got few issues that have already been solved!

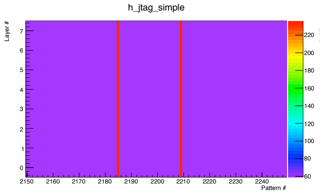


- LinkError flag is stuck to 1 $\rightarrow$ The register containing the correct information was deleted after 1 clock cycle.
- Two pattern are over written when trying to access other addresses $\rightarrow$ Glitch on line decoder.
- Minimum supply voltage higher than nominal $\rightarrow$ Due to an IR drop along power lines inside the chip, now we can simulate it;

We are able to put up FTK even with AMchip05.

Results

Got few issues that have already been solved!

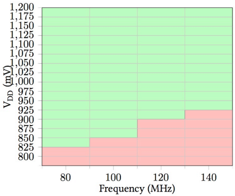


- LinkError flag is stuck to 1 $\rightarrow$ The register containing the correct information was deleted after 1 clock cycle.
- Two pattern are over written when trying to access other addresses $\rightarrow$ Glitch on line decoder.
- Minimum supply voltage higher than nominal $\rightarrow$ Due to an IR drop along power lines inside the chip, now we can simulate it;

We are able to put up FTK even with AMchip05.

Results

Got few issues that have already been solved!

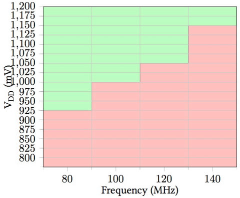


- LinkError flag is stuck to 1 → The register containing the correct information was deleted after 1 clock cycle.
- Two pattern are over written when trying to access other addresses → Glitch on line decoder.
- Minimum supply voltage higher than nominal → Due to an IR drop along power lines inside the chip, now we can simulate it;

We are able to put up FTK even with AMchip05.

Results

Got few issues that have already been solved!

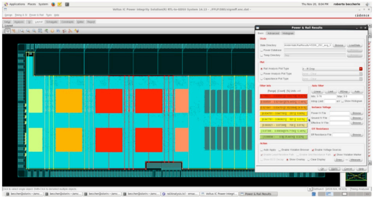


- LinkError flag is stuck to 1 $\rightarrow$ The register containing the correct information was deleted after 1 clock cycle.
- Two pattern are over written when trying to access other addresses $\rightarrow$ Glitch on line decoder.
- Minimum supply voltage higher than nominal $\rightarrow$ Due to an IR drop along power lines inside the chip, now we can simulate it;

We are able to put up FTK even with AMchip05.

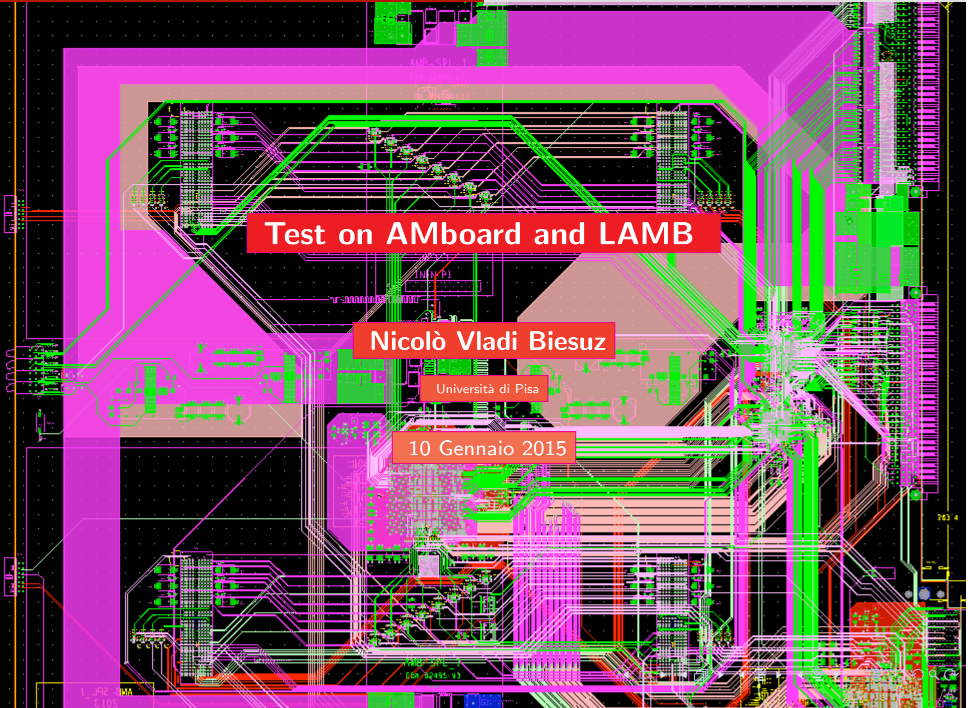
Results

Got few issues that have already been solved!



- LinkError flag is stuck to 1 $\rightarrow$ The register containing the correct information was deleted after 1 clock cycle.
- Two pattern are over written when trying to access other addresses $\rightarrow$ Glitch on line decoder.
- Minimum supply voltage higher than nominal $\rightarrow$ Due to an IR drop along power lines inside the chip, now we can simulate it;

We are able to put up FTK even with AMchip05.



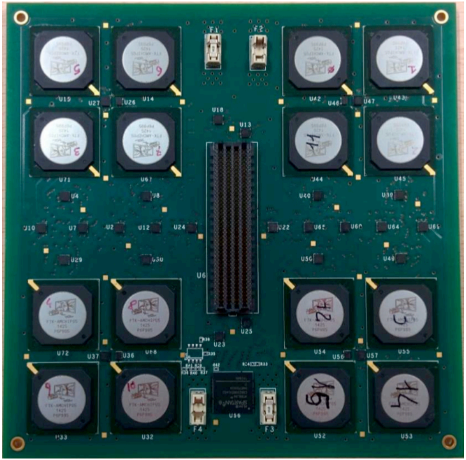
Test on AMboard and LAMB

Nicolò Vladi Biesuz

Università di Pisa

10 Gennaio 2015

LAMB: standalone test



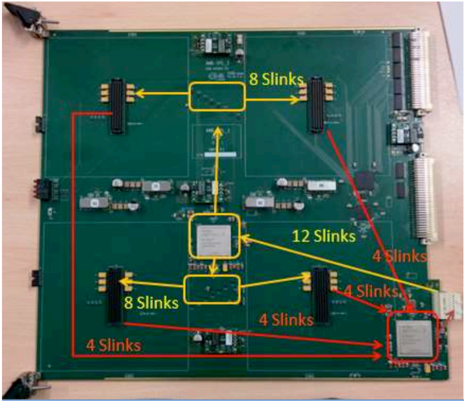
- We have an independent test structure for the validation of the LAMB design.
- The test structure is the same as the one used for the chip test (see previous talk).
  - ⇒ FPGA Evaluation Board (used for AMchip validation);
  - ⇒ Custom mezzanine (power and mechanical connection);
  - ⇒ Python scripts (used for chip validation);
- This structure has been used:
  - ⇒ to validate the FPGA firmware (minilamb)
  - ⇒ to perform the initial test on the chip itself (*LAMB prototype produced before the chip test mezzanine*).

AMBoard + LAMB: test 0

- To perform any test on the Boards we must be sure about what we are sending to the board.
- The macros implementing VME communication were written during CDF era: little repairs were needed!

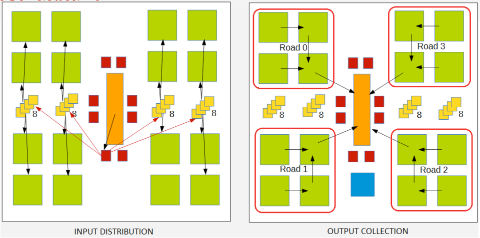
(= We can talk to our boards! =)

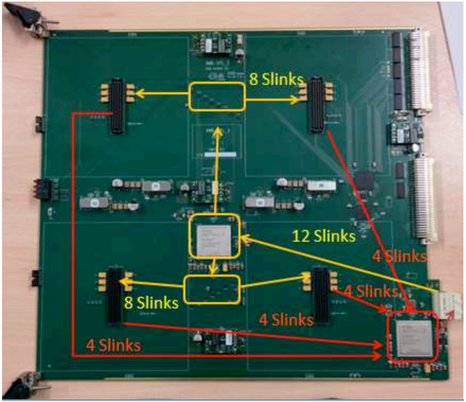
AMBoard + LAMB: test 1



- First goal was to **validate links functionality @ 2 Gbit/s**;
- We can send a Pseudo-Random Bit Sequence from/ to the AMchip and check the received sequence.
- We have **no access to points along the transmission line**: we know what we are transmitting and what we are receiving!
- **It is the fastest way to understand if all links are working**!
- **All links are working properly except one**: first soldering problem discovered! Tks to LeCroy!

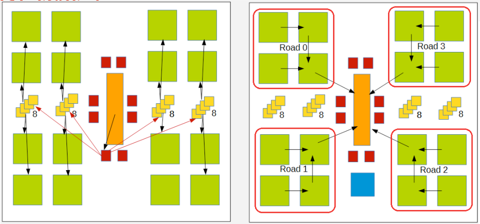
AMBoard + LAMB: test 1



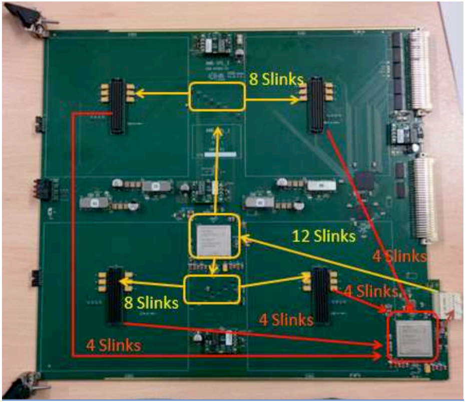


- First goal was to **validate links functionality @ 2 Gbit/s**;
- We can send a Pseudo-Random Bit Sequence from/ to the AMchip and check the received sequence.
- We have **no access to points along the transmission line**: we know what we are transmitting and what we are receiving!
- **It is the fastest way to understand if all links are working**!
- **All links are working properly except one**: first soldering problem discovered! Tks to LeCroy!

AMBoard + LAMB: test 1

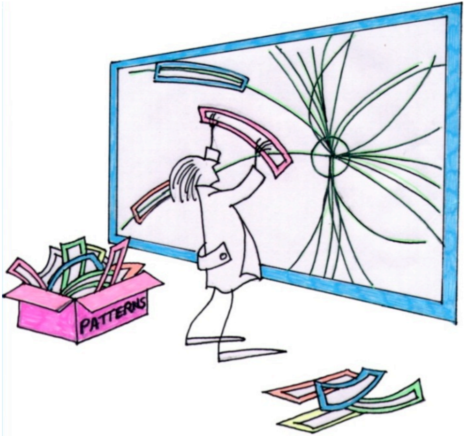


INPUT DISTRIBUTION OUTPUT COLLECTION



- First goal was to **validate links functionality @ 2 Gbit/s**;
- We can send a Pseudo-Random Bit Sequence from/ to the AMchip and check the received sequence.
- We have **no access to points along the transmission line**: we know what we are transmitting and what we are receiving!
- **It is the fastest way to understand if all links are working**!
- **All links are working properly except one**: first soldering problem discovered! Tks to LeCroy!

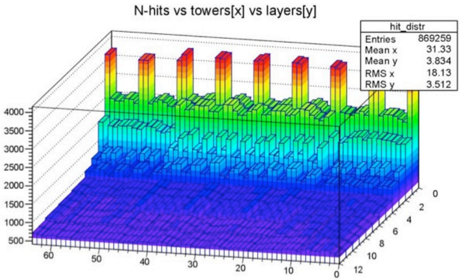
AMBoard + LAMB: test 2 *moving towards pattern recognition*



- Power-up SERDES ≈ 100 JTAG registers.
- Write a pattern: pick an easy one, e.g. [0x5,0x5,0x5,0x5,0x5,0x5,0x5,0x5];
  - Select the chip you want to address;
  - Put chip in "write" mode;
  - Set the threshold;
  - Select number of DC bits, e.g. 2;
  - Select the memory block (the pattern number);
  - Expand the DC bits, e.g. [0x5,0x5,0x5,0x5,0x5,0x5,0x5,0x5] → [0x16,0x16,0x16,0x16,0x16,0x16,0x16,0x16];
  - Load the expanded pattern;
  - Write the pattern.
- Load hits in the hits FPGA;
- Send the Hits plus an end of event word;
- Read the fired patterns and check them.

DONE with simple patterns (deterministic)

AMBoard + LAMB: test 2-bis *moving towards pattern recognition*



- We want **realistic patterns**;
- Waiting for real ones we can have a good approximation using random numbers as superstrip;
- DC bits must be encoded properly: we can **define a probability of encoding a bit as Don't Care**;
- The number of patterns must be enough to fill the chips;
- Hist must satisfy basic requirements:
  - must be **divided in events**:
  - each event **must contain a realistic number of hits**:
  - The number of hits **must reflect detector occupancy**:
  - The number of matches **must reflect the expected output rate**.