Debugging a module with bad columns

Francesco Crescioli

Context

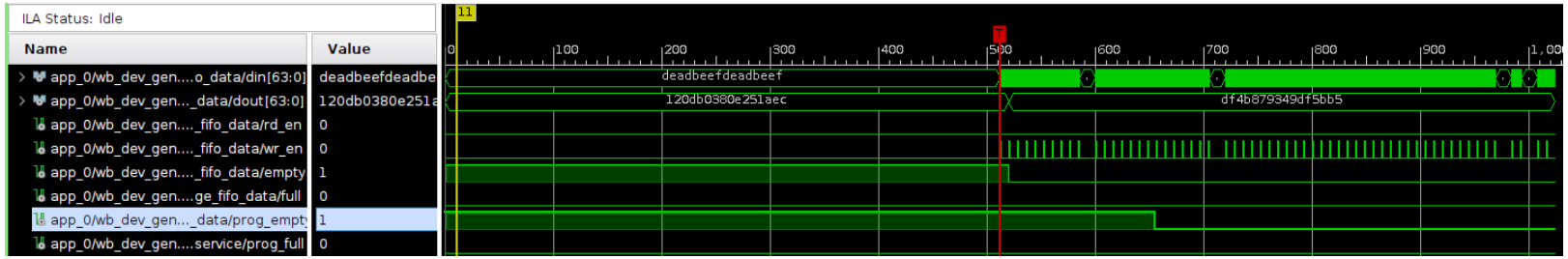
- I have a quad module with a severe case of bad columns
  - It can't even pass the initial communication check without disabling 16 central ccol
  - Once disabled the FE is fine and the module works correctly
    - Tested with standard ITk QC up to PARYLENE_MASKING stage
- Advafab-Micron module **20UPGM22110457**
  - Assembled here in LPNHE so I know the whole history. A rather good assembly except for glue seepage on the HV hole repaired taking HV on the border of the sensor

Clues

- In order to debug the CB issue + another pending issue in YARR about register read I have a modified YARR fw with these two extra features:
  - Ability to count CB and CC frames in the received stream
  - Separate data and service in two different FIFOs
- With or without the bad ccol enabled CC and CB counts are stable
  - It is unlikely that the ccol issue has an effect on the stability of the Aurora link itself
  - FYI with standard YARR config
- With the bad ccol enabled I see a lot of data that should not be there

A closer look at the data

- This is a snapshot of a continous stream of data
- NS = 0 and ChipID = 0 consistenly for all packets
  - The problematic FE is indeed ChipID 0
- Data is super hard to decode but has some patterns




```
NS 0 ChipID 0 Full block 03660e9b6a73dcd9
NS 0 ChipID 0 Full block 107bfcdbc3bc3fd7
NS 0 ChipID 0 Full block 1c40a43fafc00a42
NS 0 ChipID 0 Full block 1f5f801487f7ffe0
NS 0 ChipID 0 Full block 00a644ac2f9f8814
NS 0 ChipID 0 Full block 11be7e025277cfc4
NS 0 ChipID 0 Full block 0148df3f00293be7
NS 0 ChipID 0 Full block 1c40a47f9f881587
NS 0 ChipID 0 Full block 1e7e005217cfc00a
NS 0 ChipID 0 Full block 08df3f10290be7e0
NS 0 ChipID 0 Full block 00a46f9f88148ff3
NS 0 ChipID 0 Full block 1e00561fdfff8014
NS 0 ChipID 0 Full block 100290bebf00291b
NS 0 ChipID 0 Full block 1d7e205217d7e005
NS 0 ChipID 0 Full block 066fafc00a570c97
NS 0 ChipID 0 Full block 1c10003a5f820006
NS 0 ChipID 0 Full block 097e08001d2fc100
NS 0 ChipID 0 Full block 0064bf04002e97e0
NS 0 ChipID 0 Full block 10003a5f8200174b
NS 0 ChipID 0 Full block 1e08001d2fc1000b
NS 0 ChipID 0 Full block 14bf04000e97e080
NS 0 ChipID 0 Full block 00ba5f8200074bf0
NS 0 ChipID 0 Full block 08005d2fc10003a5
NS 0 ChipID 0 Full block 1f04002e97e08001
NS 0 ChipID 0 Full block 1a5f8200064bf040
NS 0 ChipID 0 Full block 001d2fc1000325f8
NS 0 ChipID 0 Full block 04000e97e0801192
NS 0 ChipID 0 Full block 1f8200341e5fddf5
NS 0 ChipID 0 Full block 17eb76bbeb70f277
NS 0 ChipID 0 Full block 1addaefadc3cbfbf
NS 0 ChipID 0 Full block 1d6fd76ed7fd76e1
NS 0 ChipID 0 Full block 1c9ff5dbb7f7fd6f
NS 0 ChipID 0 Full block 1aedc3cbfffebb7e
NS 0 ChipID 0 Full block 176e7e5fddf5bf5b
```

Summary

- At least for this chip the bad core column issue seems to be related to an unexpected behaviour of the internal hit data flow
  - It is NOT a case of link instability
  - The bad columns induce the chip to send out a never ending flow of hit data
  - Everything else in the chip seems functional
- This is NOT the expected behaviour
  - No trigger were issued
    - no trigger command were sent and this is confirmed by a reading of TrigCnt = 0
- This is just one quad, I have others with less severe behaviour I will look into very soon