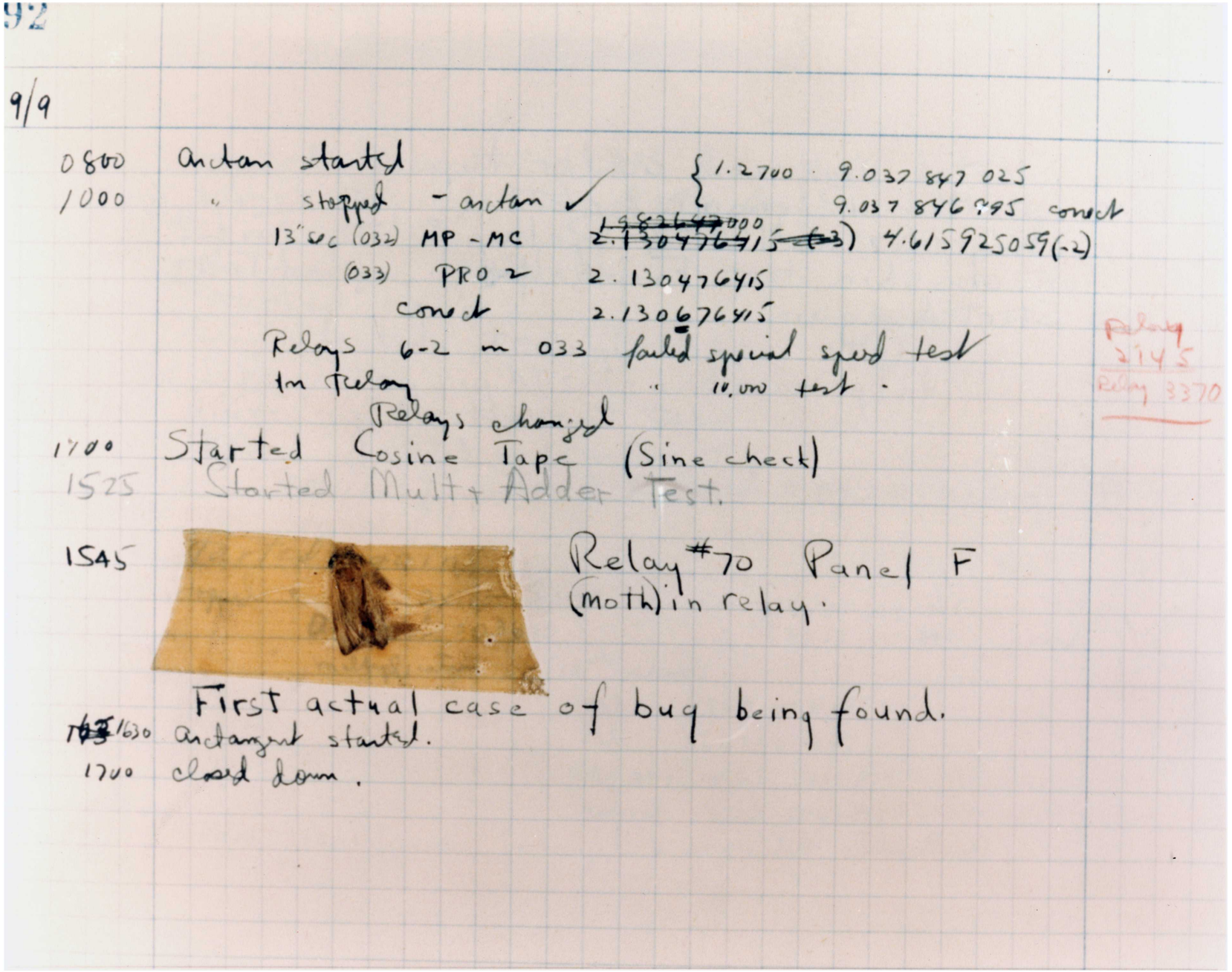
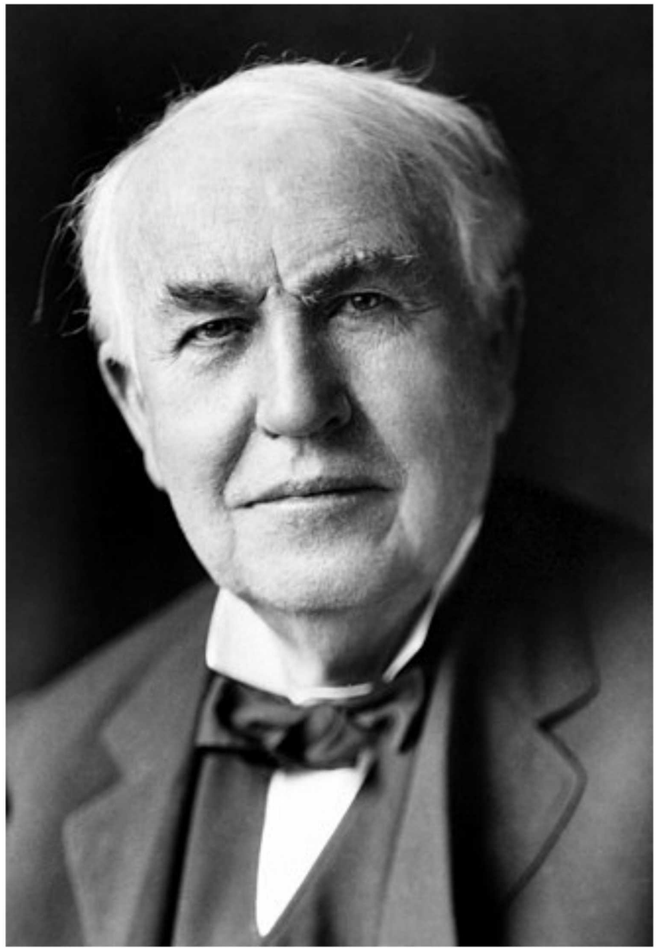
FPGA design debugging...
How hard can it be?

FPGA Conference 2025

Kamil Rudnicki, PhD

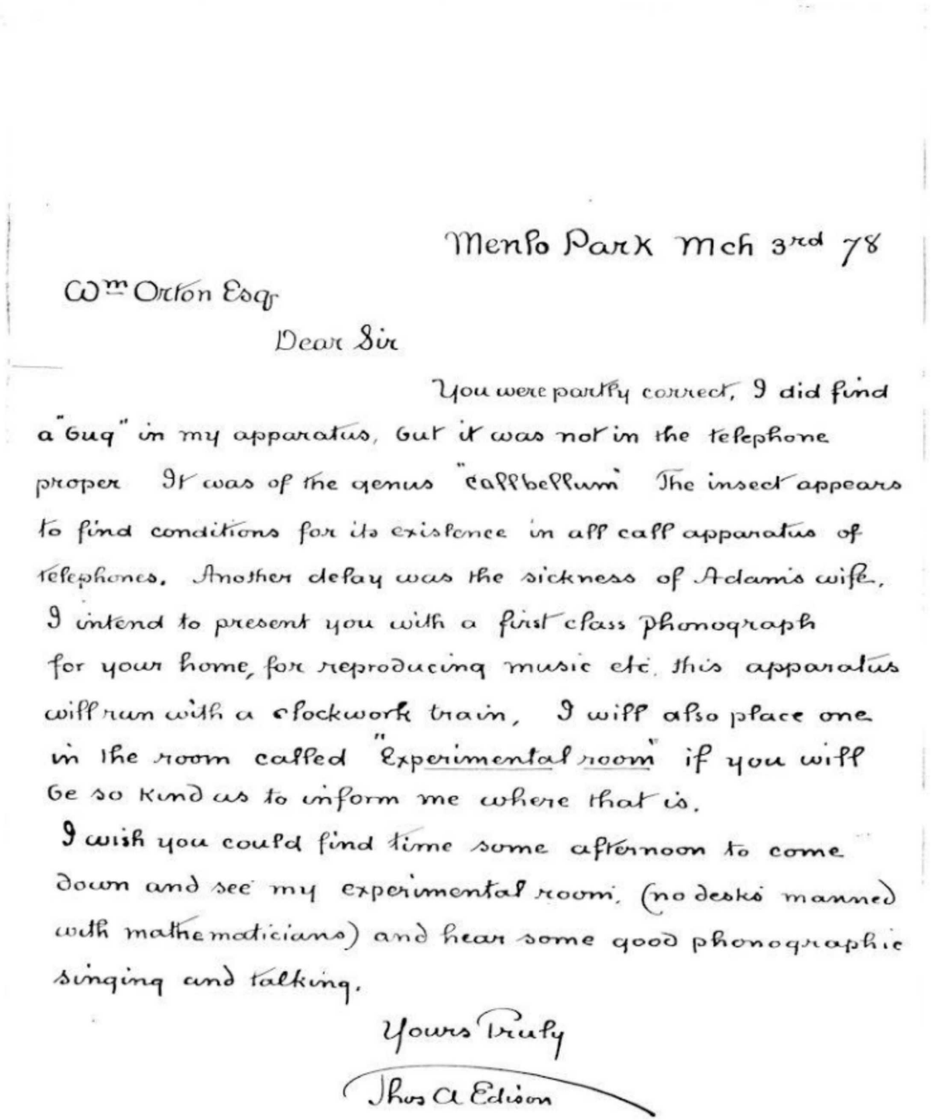
1940s - Admiral Grace Hopper
a Mark II computer at Harvard University





Thomas Alva Edison
(1847-1931)

- over 1000 patents
- established journal
  Science (1880)
- lightbulb inventor
- improved Bell's telephone
  (inductor coil)
- phonograph (predecessor
  of gramophone)
- invented nickel–iron
  battery (1904)
- ...



Menlo Park Mch 3rd 78

Wm Orton Esqr

Dear Sir

You were partly correct, I did find a "bug" in my apparatus, but it was not in the telephone proper It was of the genus "callbellum" The insect appears to find conditions for its existence in all call apparatus of telephones. Another delay was the sickness of Adams wife. I intend to present you with a first class phonograph for your home, for reproducing music etc. this apparatus will run with a clockwork train, I will also place one in the room called "Experimental room" if you will be so kind as to inform me where that is.

I wish you could find time some afternoon to come down and see my experimental room, (no desks manned with mathematicians) and hear some good phonographic singing and talking.

Yours Truly

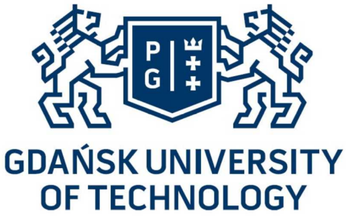
Thos A Edison


19 years of FPGA experience:

commercial (Ericsson, GE, Arrow, Brightelligence)

scientific

- ✔ Lodz University of Technology, Poland
- ✔ Gdansk University of Technology, Poland
- ✔ Military University of Land Forces, Poland
- ✔ Nicolaus Copernicus Astronomical Center, Poland
- ✔ Universitat Politècnica de Catalunya, Spain
- ✔ University of Glasgow, Scotland
- ✔ University of Trento, Italy



Are quality-oriented techniques part of debugging? NO, but today **YES.**

Vendor specific? **NO.**

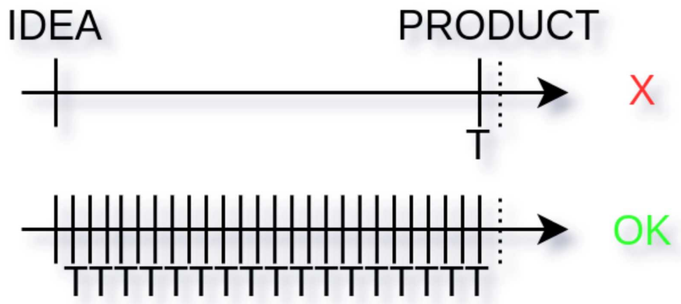
Scientific ideas approachable? **YES.**

Where is the bug?

- ✔ hardware
- ✔ software
- ✔ firmware

At which stage?

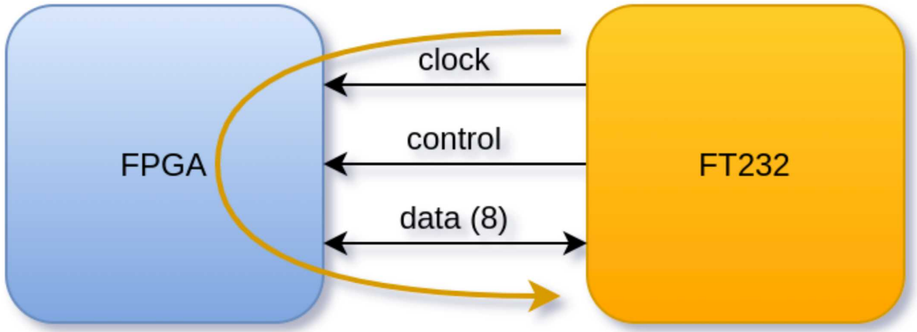
- ✔ idea
- ✔ schematics
- ✔ bring-up
- ✔ development
- ✔ ready product
- ✔ released product

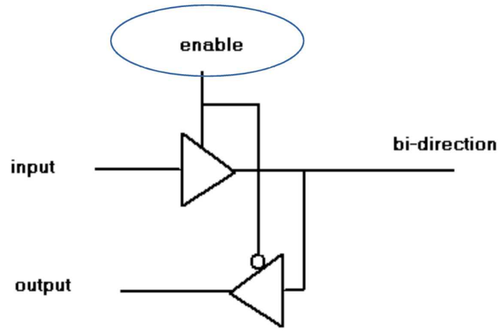


General practices

- ✔ read messages / warnings, reports
- ✔ constraints exist
- ✔ constraints are correct

  set min/max input/output delays
- ✔ timing constraints are met
- ✔ using IP cores (proper synchronizer)





Name	Min	Nom	Max	Units	Comments
t1		16.67		ns	CLKOUT period
t2		8.33		ns	CLKOUT high period
t3		8.33		ns	CLKOUT low period
t4	1.4		9	ns	CLKOUT to RXF#
t5	2.4		9	ns	CLKOUT to read DATA valid
t6	2.4		9	ns	OE# to read DATA valid
t7	8		16.67	ns	OE# setup time
t8	0			ns	OE# hold time
t9	7.5		16.67	ns	RD# setup time to CLKOUT (RD# low after OE# low)
t10	0			ns	RD# hold time
t11	1		9	ns	CLKOUT TO TXE#
t12	8		16.67	ns	Write DATA setup time
t13	0			ns	Write DATA hold time
t14	8		16.67	ns	WR# setup time to CLKOUT (WR# low after TXE# low)
t15	0				WR# hold time

Table 17 - FT245 Synchronous FIFO Interface Signal Timings

1 register (EN) driving 8 bidir buffer – caused timing issues

SOLUTION: for EN set fanout = 1

87% FPGA projects went with bugs into production (2024).[1]

little / no verification AND not enough testing

"If we were going to do it wrong, we wouldn't do it at all."

Wojciech Slupski in • 1.
FPGA, Electronics and other

... it never fails until you test it.
Therefore the problem is with the testing.
Stop testing.

[1] Siemens – 2024 Wilson Research Group – FPGA functional verification trend report

Functional verification

- ✔ independent people
- ✔ proper comprehensive test plan
  - methodology UVM / OSVVM,
  - methods (analysis, inspection, testing, demonstration),
  - metrics, coverage, reports, ...
- ✔ specific test cases
- ✔ equivalent model testing

Schematics design

Bring-up

- ✔ boundary scan
- ✔ interface test design

Document everything:

1) the process in the company
2) the design verification

[1]

[1] Brightelligence® whitepaper - Push FPGA boundaries

Hardware debugging – in a moment

Built-in tools

- ✔ link analyzer (prbs, eye diagram)

Quality assurance

- ✔ CI/CD
- ✔ system scaling – 2015 Kielbik[1], 2021 Biookaghazadeh[2]

BIST

- ✔ configuration readback
- ✔ power-on interface testing

[1] Methodology of Firmware Development for ARUZ–An FPGA-Based HPC System
[2] Toward Multi-FPGA Acceleration of the Neural Networks

High complexity, repeatable design

Solution: generate the code

- ✔ 2012 Asaad[1]
- ✔ 2020 Kielbik[2]

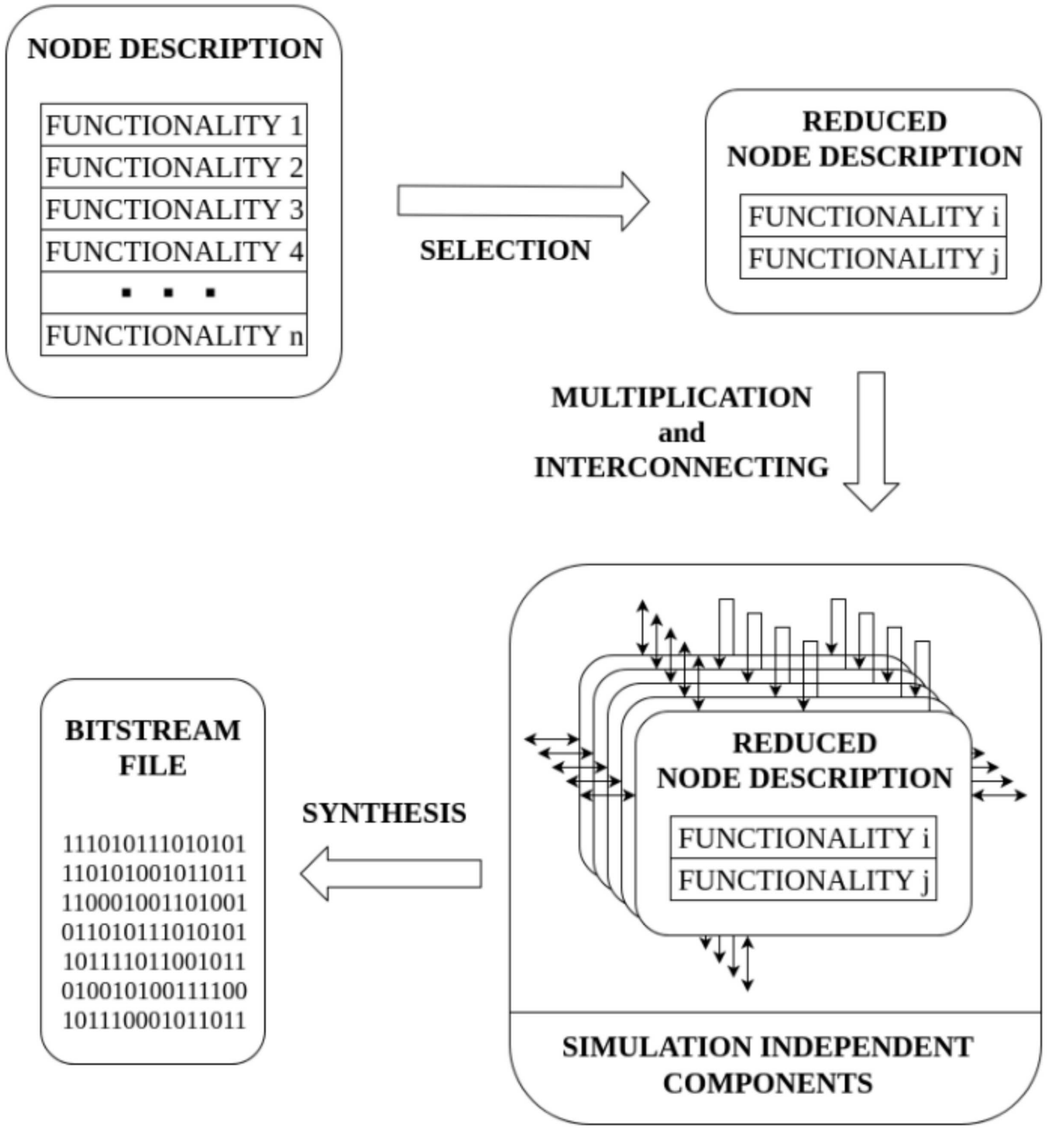


Figure 3. Preparation of the DSlave configuration.

[1] A cycle-accurate, cycle-reproducible multi-FPGA system for accelerating multi-core processor simulation
[2] Methodology of Firmware Development for ARUZ–An FPGA-Based HPC System

Limited memory

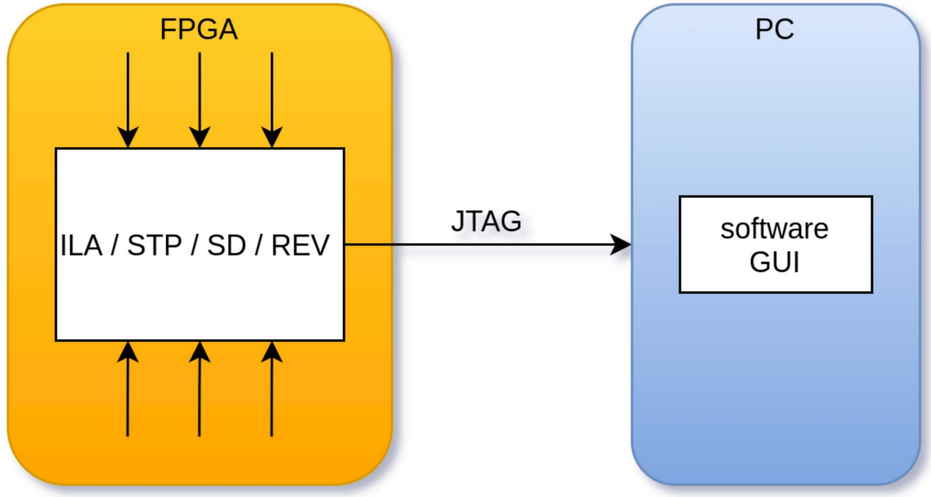
- add external memory

  (2008 Camera[1], 2016 Azeem[2])

- cut the design (scale down)

- low bandwidth: LEDs, TP,

  flags, timer




```
entity bright_timer is
    generic (
        g_clk_freq          : natural := 10_000_000
    );
    port (
        pi_clk              : in    std_logic;
        pi_rst              : in    std_logic;

        po_time_min         : out   std_logic_vector(5 downto 0);
        po_time_sec         : out   std_logic_vector(5 downto 0);
        po_time_mili        : out   std_logic_vector(9 downto 0);
        po_time_micro       : out   std_logic_vector(9 downto 0)
    );
end bright_timer;
```

[1] An integrated debugging environment for FPGA computing platforms
[2] Multiple FPGAs based prototyping and debugging with complete design flow

Limited memory

- high bandwidth: flags, off-load
  - if SoC – share DDR memory
  - existing interface:

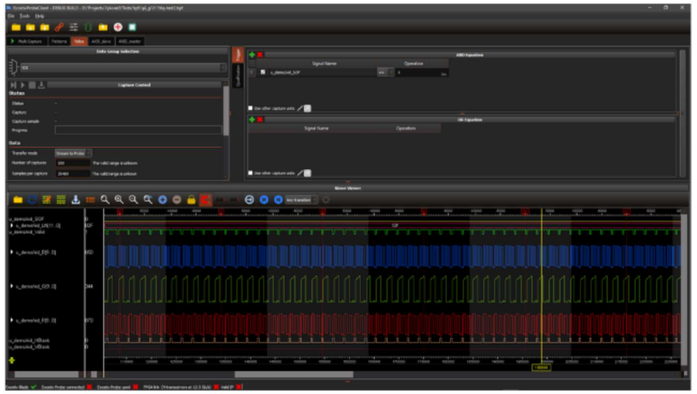
    PCIe (2014 Putnam[1])

    dedicated protocol (2018 Kielbik[2])
- Agilent Trace Core (ATC)

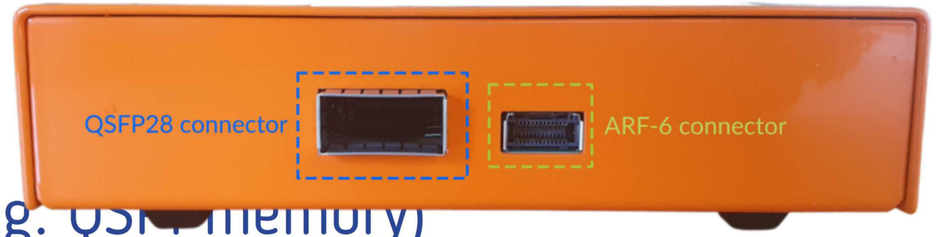
  external logic analyzers (2011 DS650)
- Exostiv Labs – over MGT
- step mode (2005 Aguirre[3], 2016 Tzimpragos[4]) – VIO (eg. QSPI memory)

Exostiv Probe EP16000

EP16000 Probe



Exostiv software environment v2 (DARK)



1 A reconfigurable fabric for accelerating large-scale datacenter services
2 ARUZ – Large-scale, massively parallel FPGA-based analyzer of real complex systems
3 Microprocessor and FPGA interfaces for in-system co-debugging in field programmable hybrid systems
4 Application Debug in FPGAs in the Presence of Multiple Asynchronous Clocks

Limited programmability

- ✔ bug prevention
- ✔ prototype on different FPGA
- ✔ live debug (2 probes, no re-synthesis)
- ✔ generate internal flags (MUX)
- ✔ log to BRAM or external on-board memory
- ✔ ATC-like approach
- ✔ step mode

[1] Algebraic Machine Learning

Predictive debugging

- ✔ In-Spec (A) - speculative signal selection - 2011 Hung[1]

Bitstream modification

- ✔ (A) static MUX + optional shift - 2012 Poulos[2]

Statistical corelation

- ✔ software signal-bug corelation - 2007 Jones[3]

1 Speculative Debug Insertion for FPGAs
2 Leveraging Reconfigurability to Raise Productivity in FPGA Functional Debug
3 Debugging in parallel

Reset

- 100G Ethernet – Xilinx PG203 (2022.2) – 272 pages long – Reset (p. 48)

  “The 100G Ethernet IP core provides sys_reset input
  to ~~reset GTs and~~ integrated CMAC block and
  gtwiz_reset_tx_datapath and gtwiz_reset_rx_datapath
  to reset GT and CMAC RX and TX datapaths individually.”

  In hardware debug (p. 263):
  ”The GT requires a GTRXRESET after the serial data becomes valid
  to insure correct CDR lock to the data. This is required after a cable
  pull and re-plug, or after powering on or resetting the link partner.”

- a ninja bug - asynchronous reset
- time dependent deadlock

Recommendations

- a small haystack is better than a large one

  divide and conquer
- fight ninjas - tripple check reset strategy
- run power-cycle tests as standard procedure
- always simplify - use known, predictable patterns
- write own IPs
- don't trust the documentation 100%
- prevention is the best policy - verification is your friend
- test, test, test, test...

Thank you