

NTS in a FPGA



Christer Weinigel <wingel@netnod.se>

June 22, 2022

Who am I?

- Christer Weinigel <christer@weinigel.se>
 - Freelance contractor working for Netnod
 - Software developer dabbling in hardware
 - Knows enough about FPGAs to be dangerous
- Actual FPGA development done by Assured AB

Netnod Time & Frequency

- Clock nodes
 - 6 nodes (gbg, lul, mmo, 2x sth, svl)
 - Gothenburg, Luleå, Malmö, 2x Stockholm, Sundsvall
- Redundancy within a node
 - 2x caesium clock
 - 2x measurement, microstepper, distribution amplifiers
 - 2x time scales realized as 10MHz and PPS
 - 2x NTP servers
 - and some more



Netnod Time & Frequency (2)

- GNSS for synchronisation of time scales
 - Within 20 ns of UTC(SP)
- NTP and NTS
 - Free service over the internet
 - Accurate to about +/-10 ms depending on distance
 - Can be much better if there are only a few hops to the server
- PTP grandmasters
 - For customers needing more accurate time
 - For best accuracy use a dedicated connection

What is NTP?

- NTP - Network Time Protocol
 - *The* protocol for time exchange over the internet
 - One of the oldest application protocols on the internet
 - In use since 1985
 - No security (most of the time)
 - Authentication
 - MD5 and SHA hash algorithms
 - Shared secrets
 - Does not scale to internet sizes
 - No other practical security (autokey is not good enough)

What is NTS?

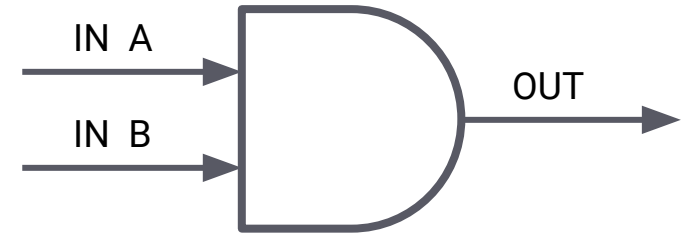
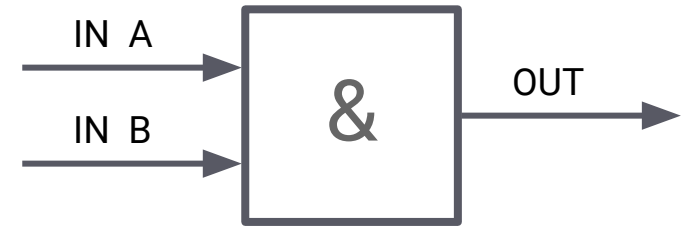
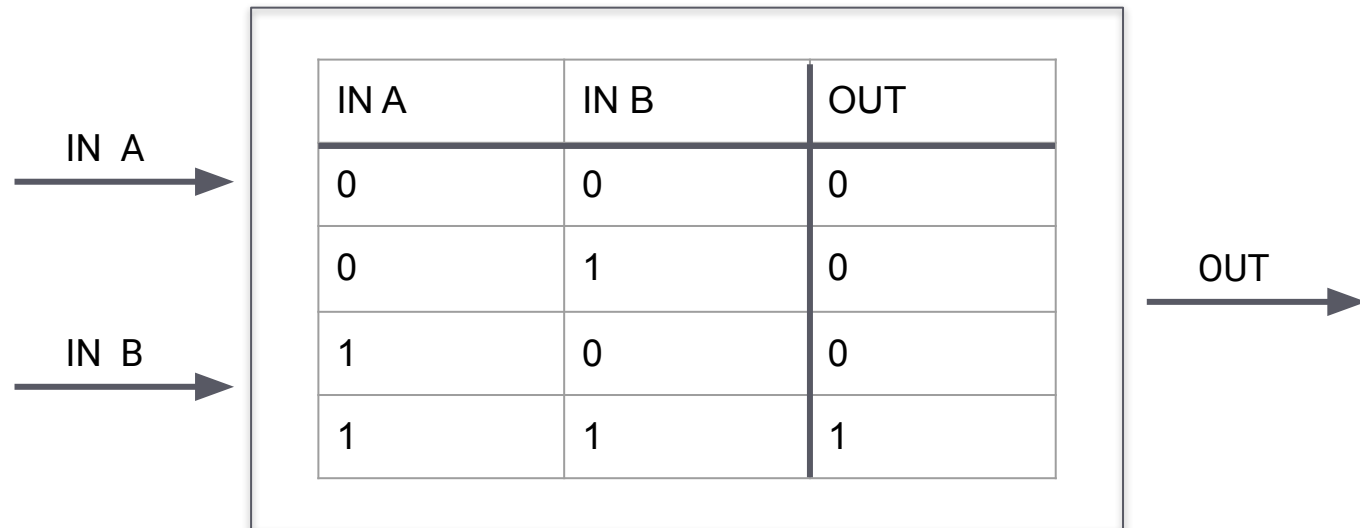
- NTS - NTP with Security
 - Adds authentication and encryption
 - NTP with scalable security
 - Similar to how HTTPS adds security to HTTP
 - Netnod have been participating in development of NTS
 - IETF draft and RFC process (2018 and onwards)
 - IETF hackathon (proof of concept implementation 2019)
 - NTS accepted as RFC8915 (September 2020)

What is an FPGA?

- Programmable hardware
 - Not as efficient as a real ASIC, but often good enough
- Lookup tables, lots of them
 - A few thousands to millions
 - Can implement logic gates: AND, OR, NOT
 - Registers in a soft CPU
- Can do many things in parallel

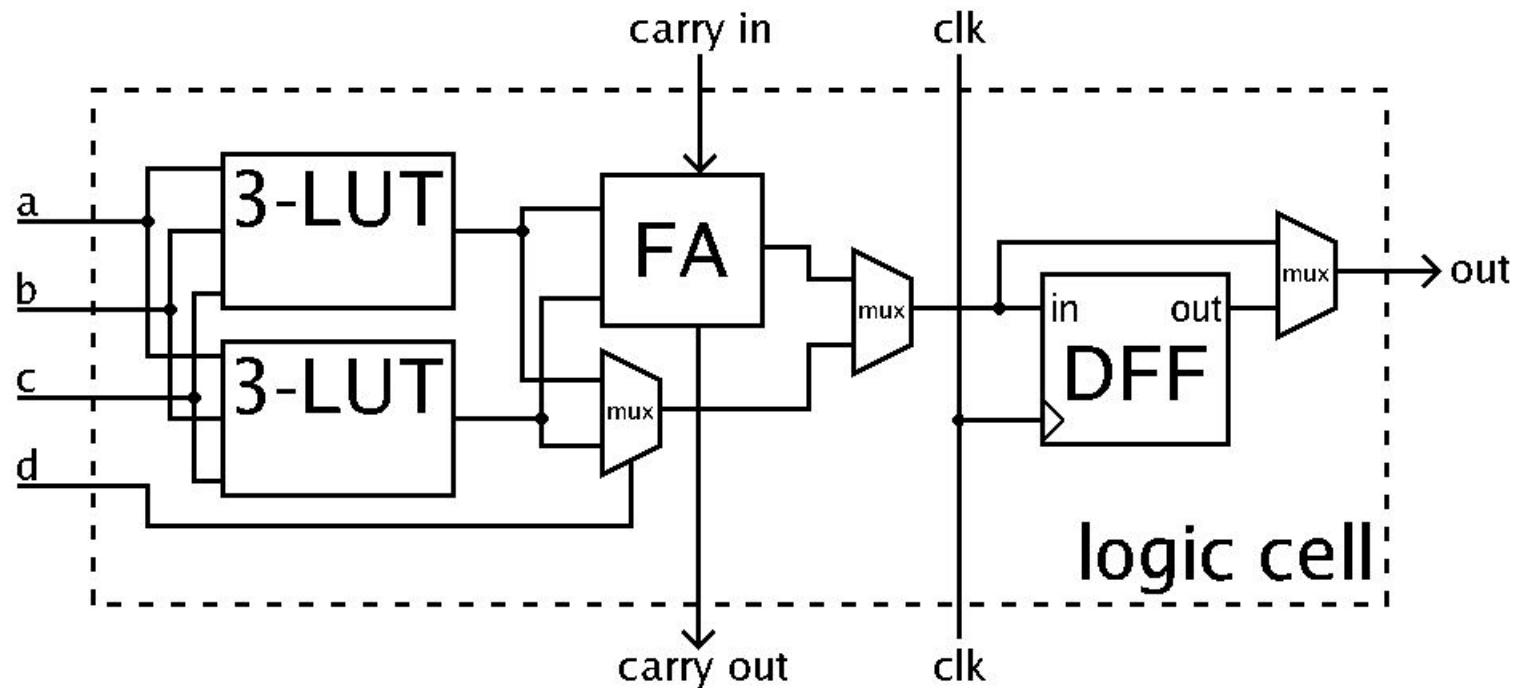
What is an FPGA (2)

- Lookup table (LUT) for AND gate
 - All equivalent



What is an FPGA (3)

- Combine LUTs with registers for more complex logic



Source: Wikimedia commons https://en.wikipedia.org/wiki/File:FPGA_cell_example.png

NTP packet

- This is plain old NTP without any security
- Stateless UDP
 - Only 48 bytes of payload
 - Ethernet frame, IP frame, normal checksums
 - Fixed position fields in payload



NTP authentication

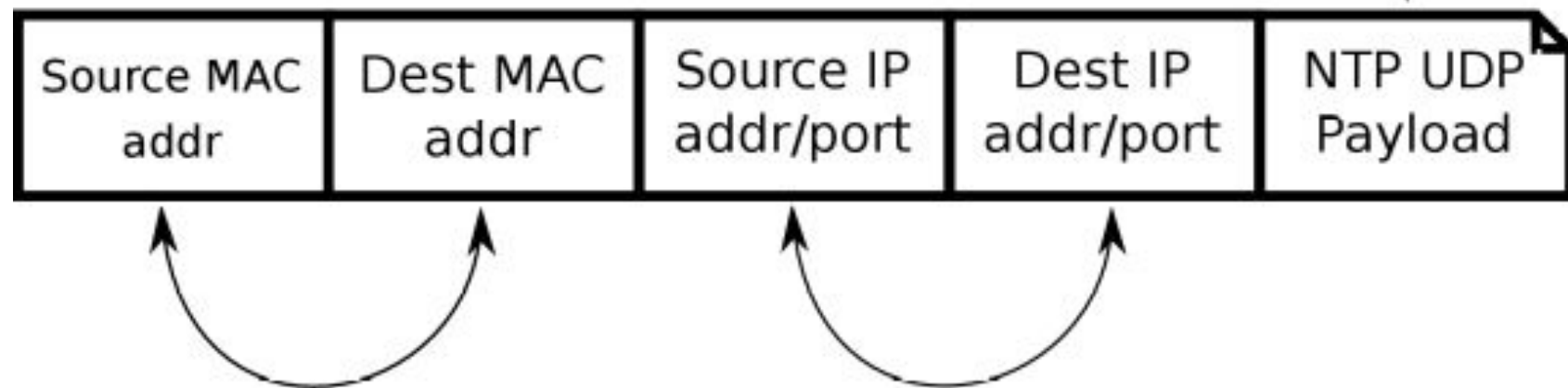
- Optional signature with MD5, SHA
 - Adds about 20 bytes to payload
 - Requires a shared secret for each client
 - Limited number of secrets
 - Does not scale to internet sizes



NTP FPGA implementation

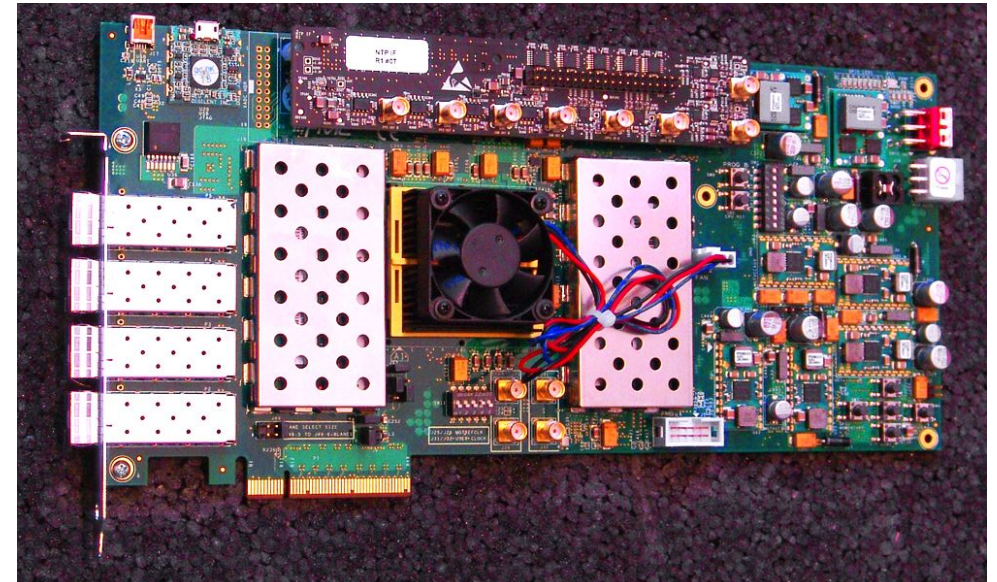
- FPGA implementation is fairly simple
 - Verify checksum and request
 - Fill in metadata and timestamps
 - Swap source/destination
 - Calculate checksum

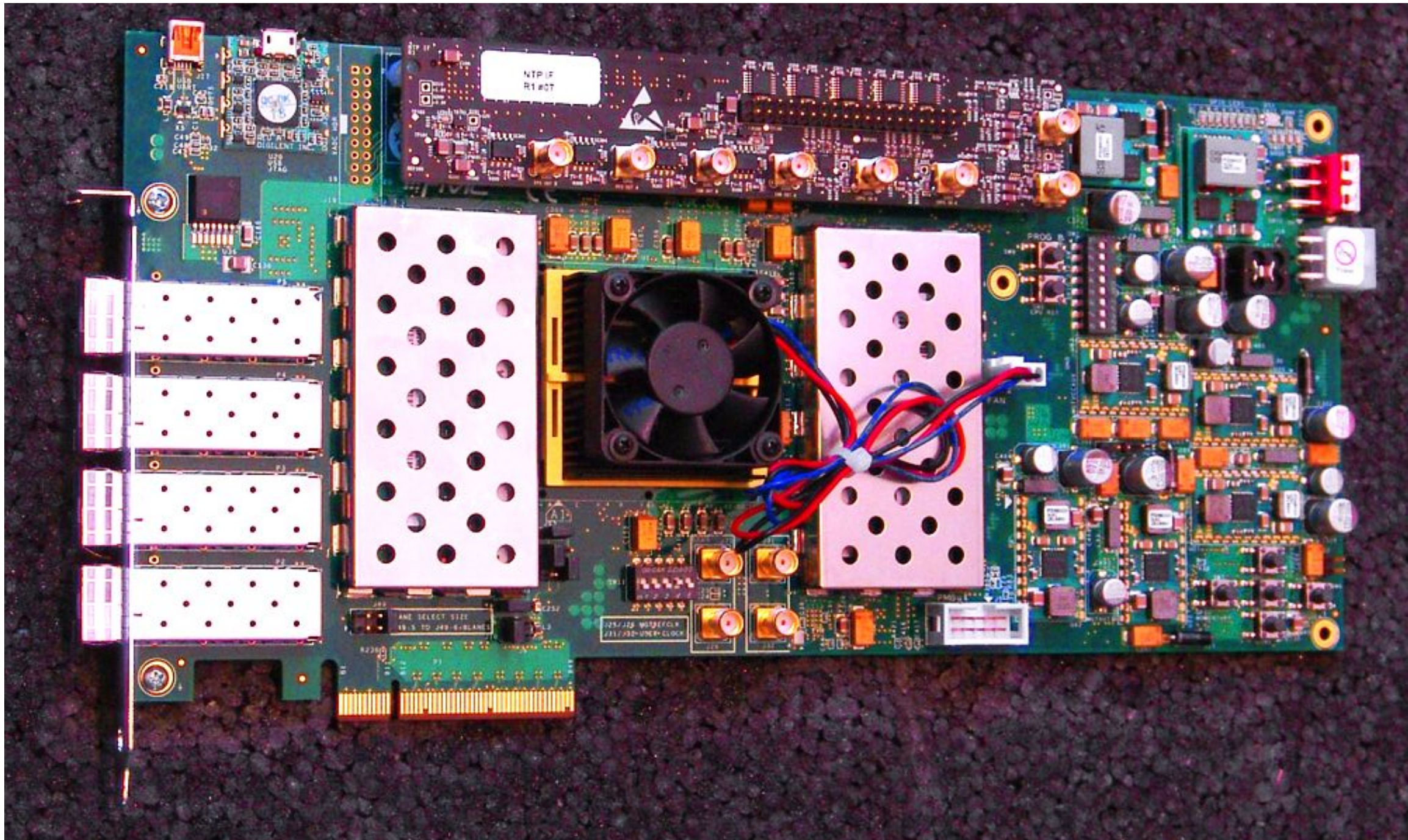
Payload as displayed in previous slides:
Mode client/server, Leap second,
Estimated precision, Received
timestamp, Transceive timestamp etc



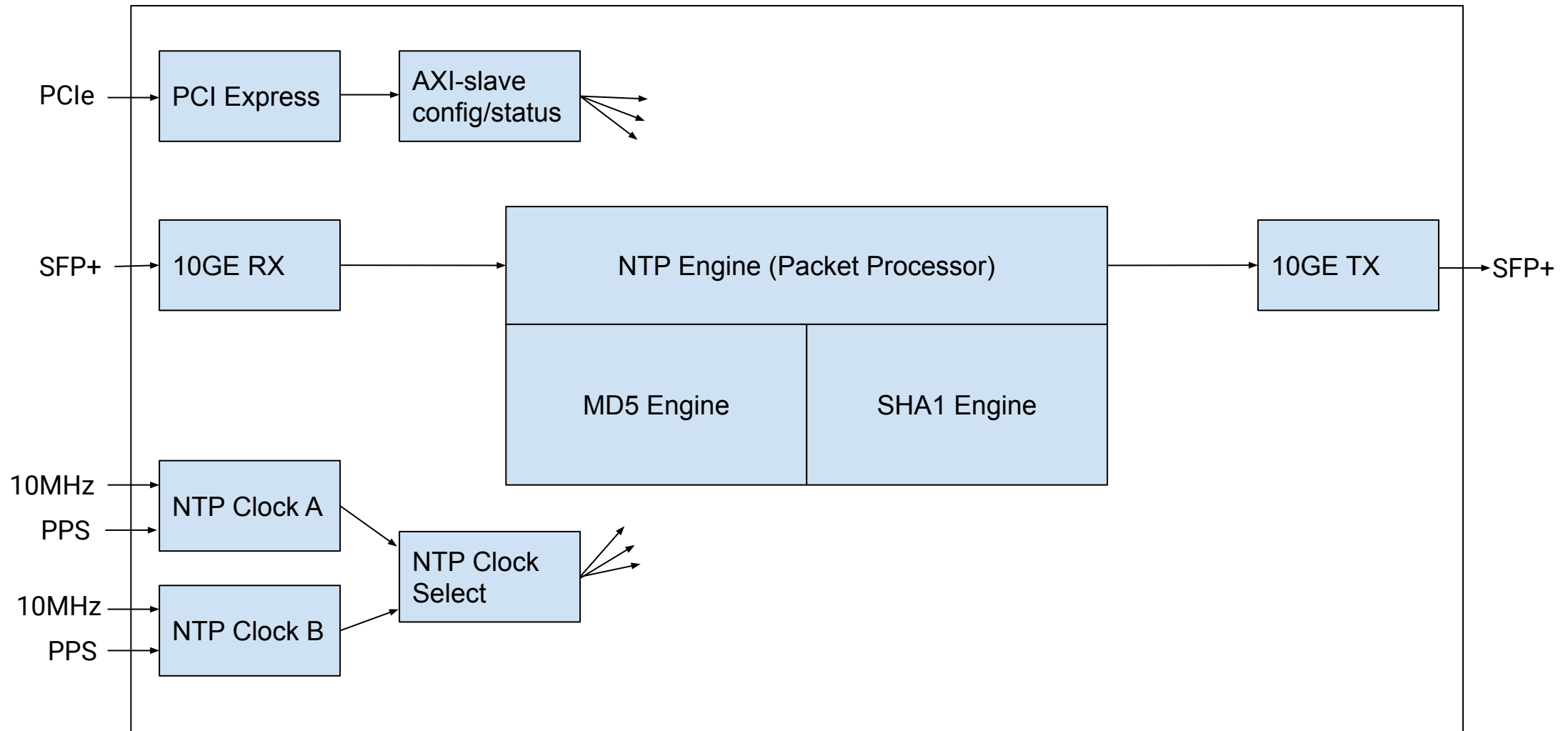
NTP FPGA at Netnod

- In production at Netnod since 2016
- Handles 4x 10Gbit NTP traffic
 - On a Xilinx VC709 reference board with about 690 000 LCs
- Reference board inside a rack mounted PC
 - Power and control





NTP FPGA architecture



Key points for NTP FPGA

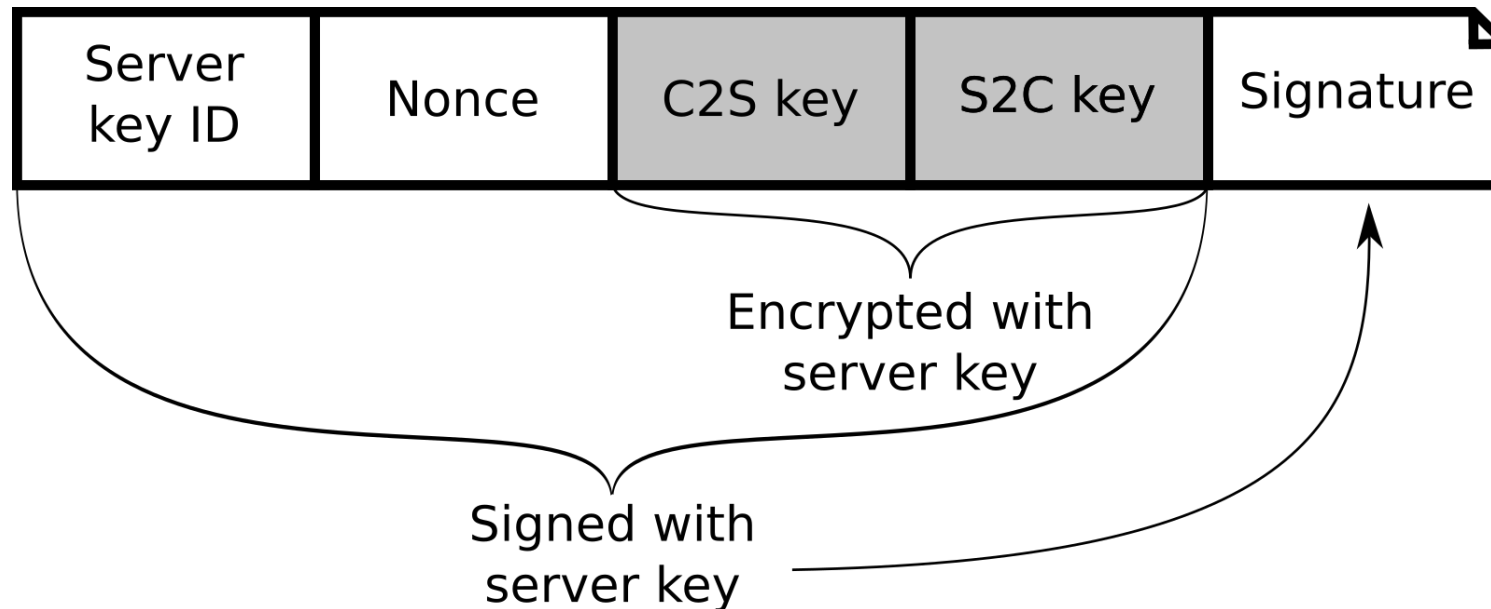
- Streaming architecture
 - Runs at wire speed (4x 10Gbit)
 - No buffering
 - No reordering
 - Predictable latency
 - +/-10ns
 - Large MD5 and SHA1 implementations to be able to keep up
- The bad
 - Does not support NTS
 - No scalable security

NTS architecture

- Adds authentication and encryption
- Two stages
 - NTS-KE Key Establishment
 - Using TLS infrastructure
 - Same certificates as HTTPS
 - Handled by a PC
 - NTS-TS Timestamping
 - NTP with extensions
 - Stateless
- NTS cookies

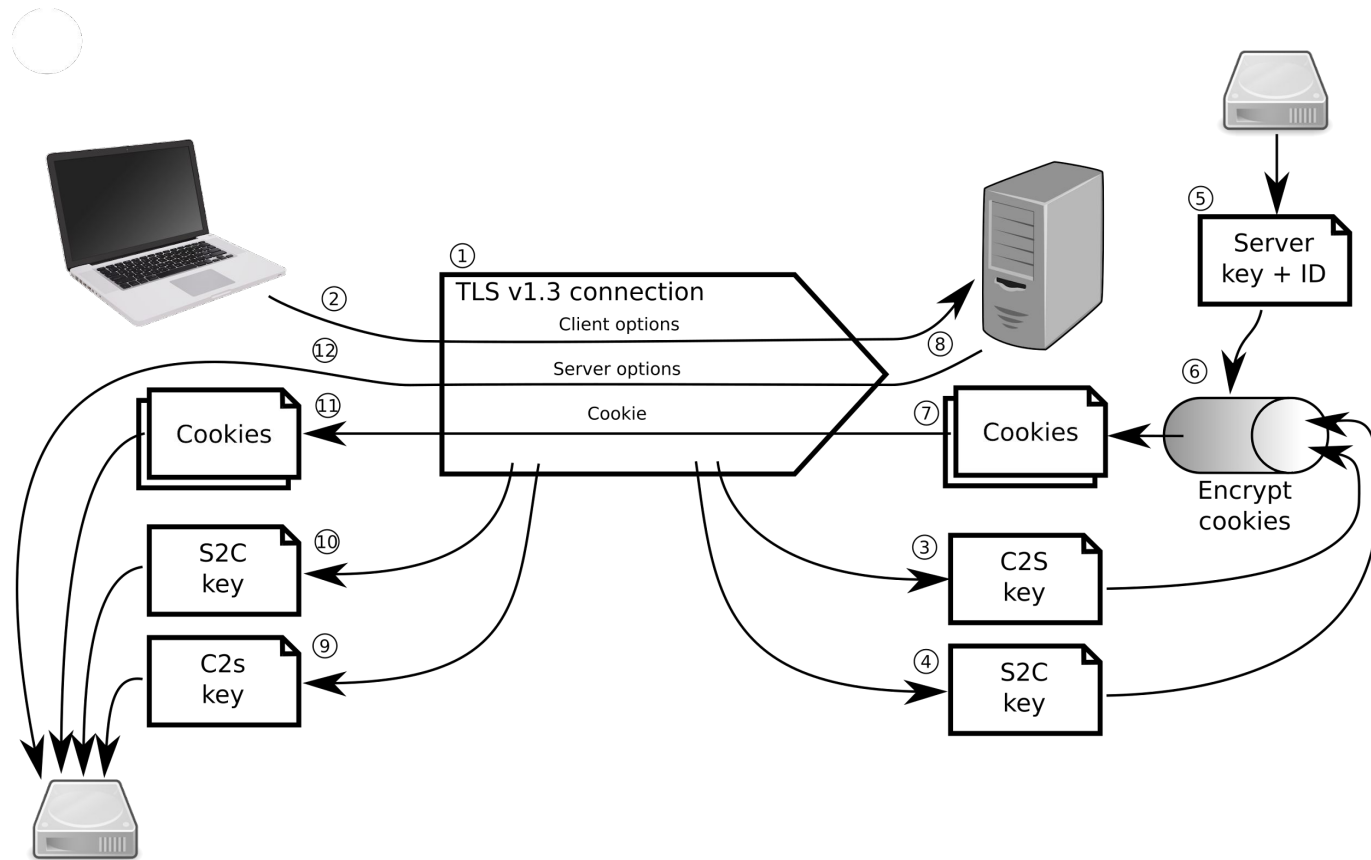
NTS cookie

- During NTS-KE the server and client agree on C2S and S2C keys
 - Server places keys in 8 encrypted cookies
 - Hands all cookies to the client
 - Client stores keys and cookies



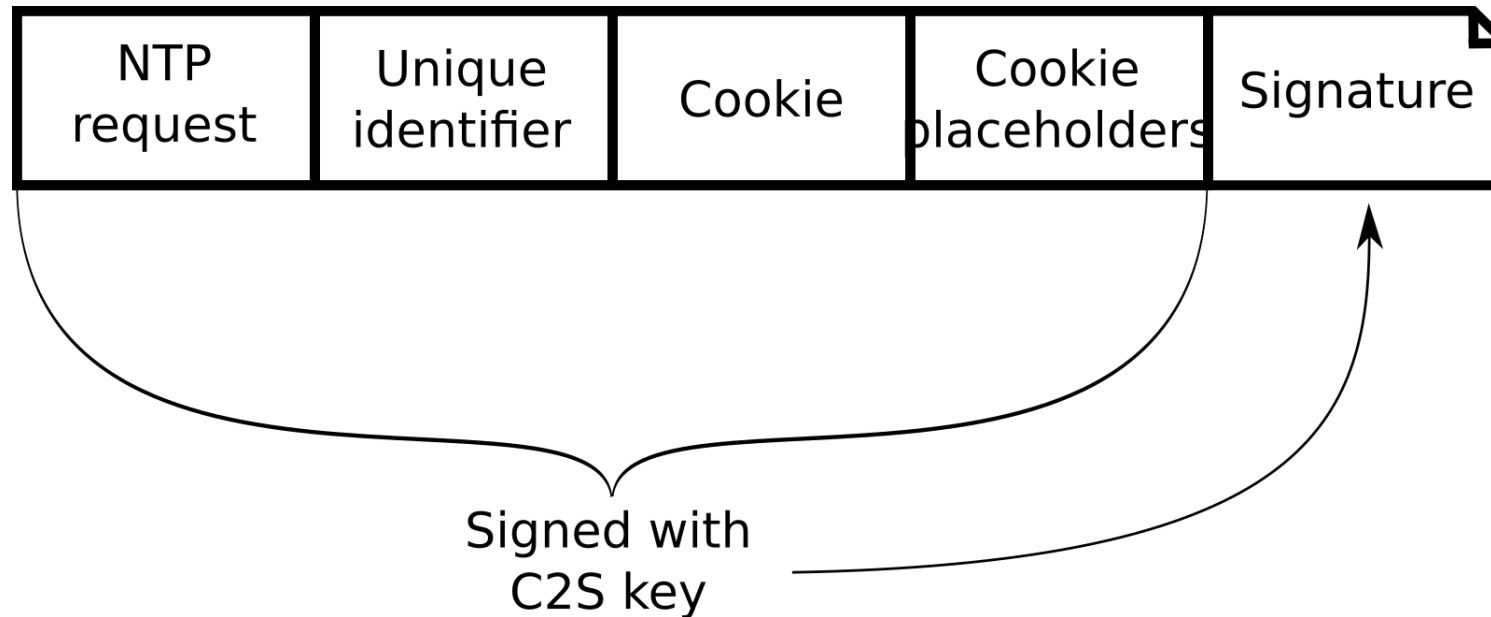
NTS-KE Key Establishment

- TLS with normal HTTPS certificates



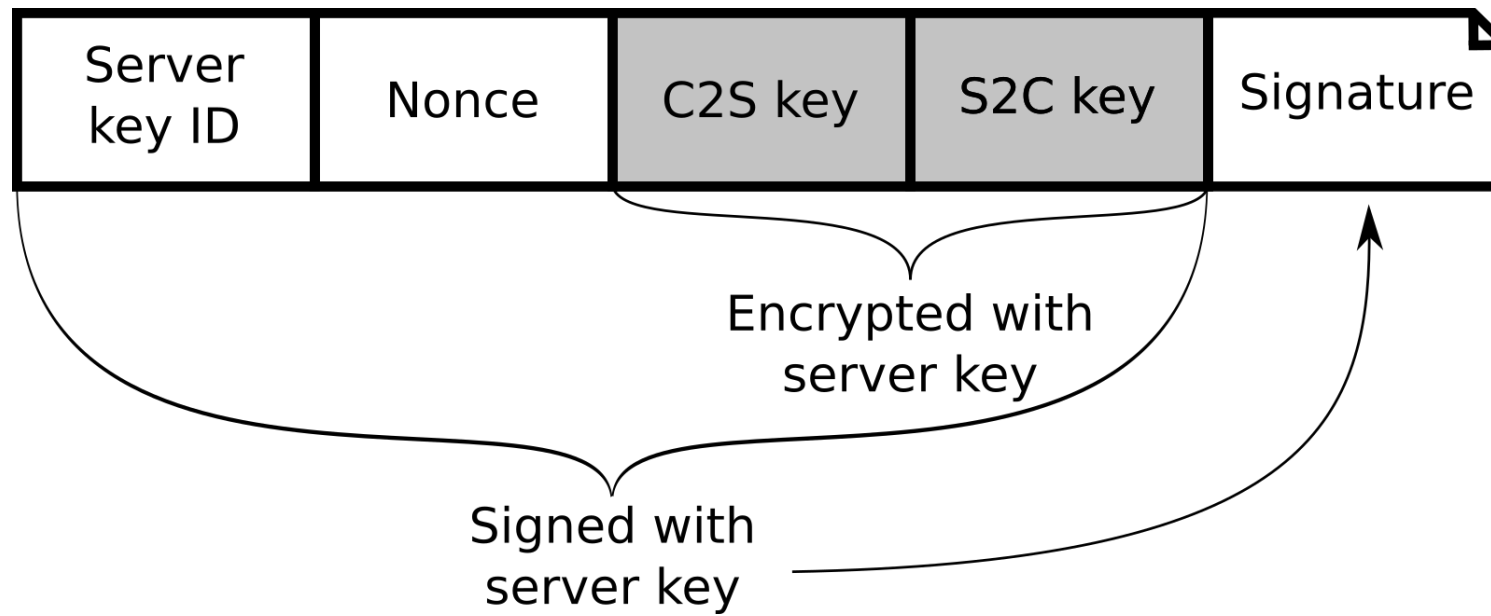
NTS-TS Timestamping request

- NTP request is the same as for normal NTP
- Extensions for NTS
 - variable size, order, and count



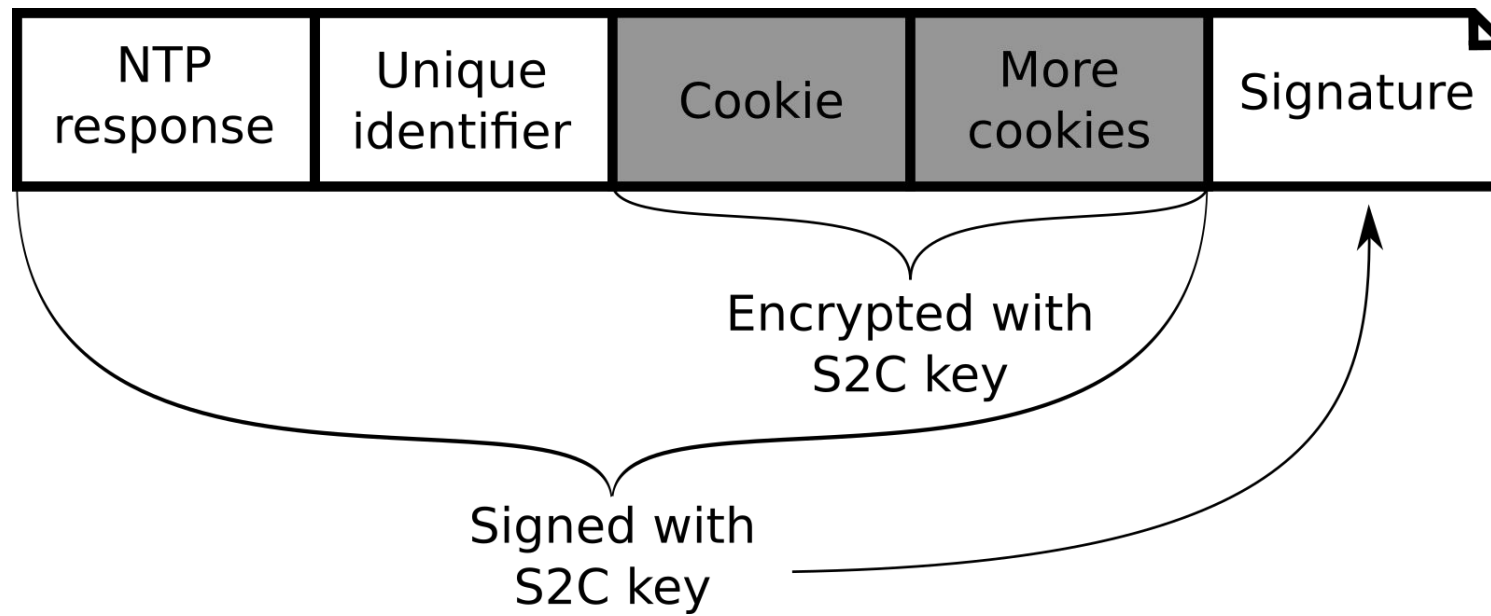
Server decrypts NTS cookie

- Server can extract C2S and S2C keys from cookie
 - Use C2S key to validate request
 - Use S2C key to encrypt and sign response

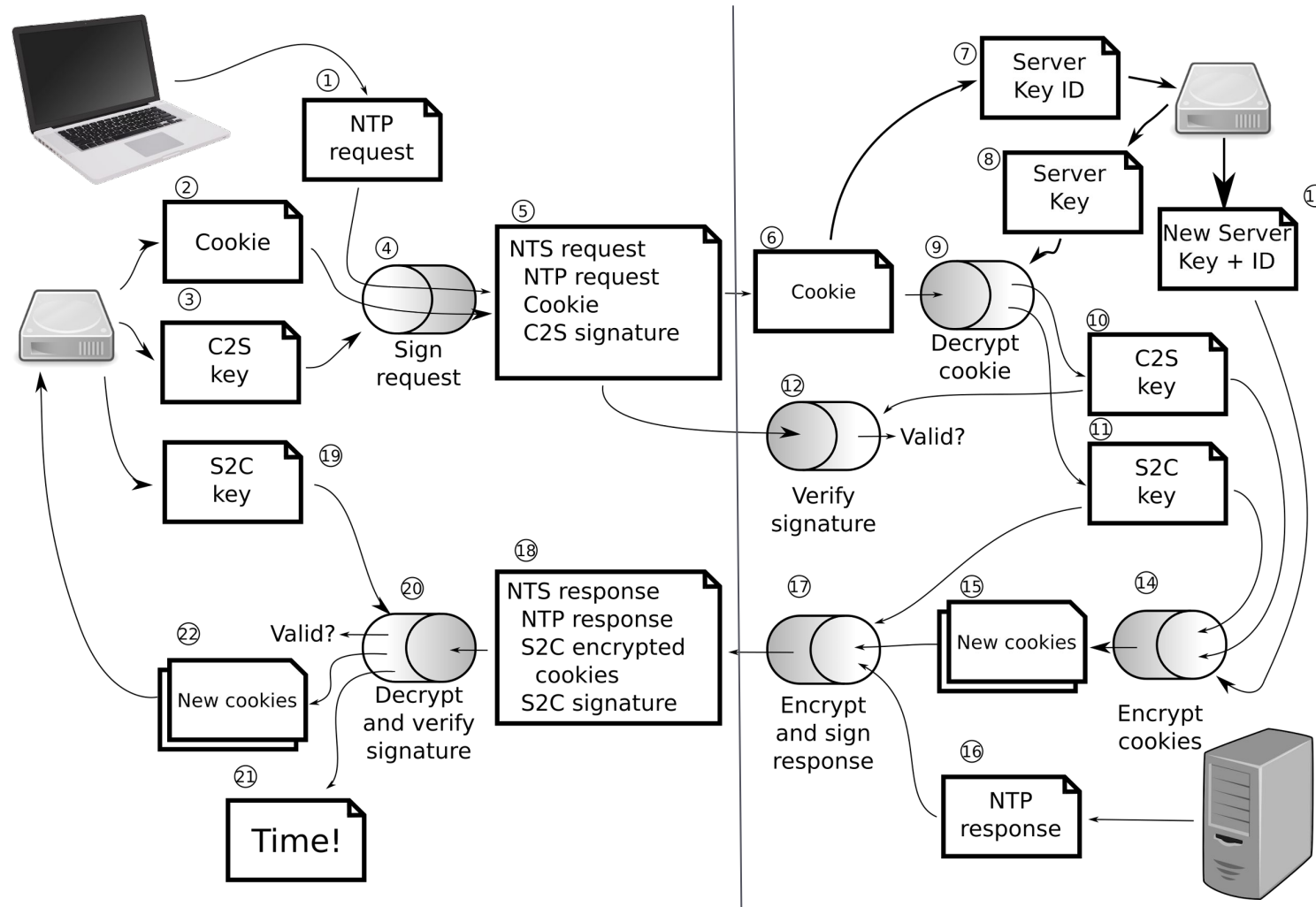


NTS-TS Timestamping response

- NTP response is the same as for normal NTP
- Extensions for NTS
 - Variable size, order, and count



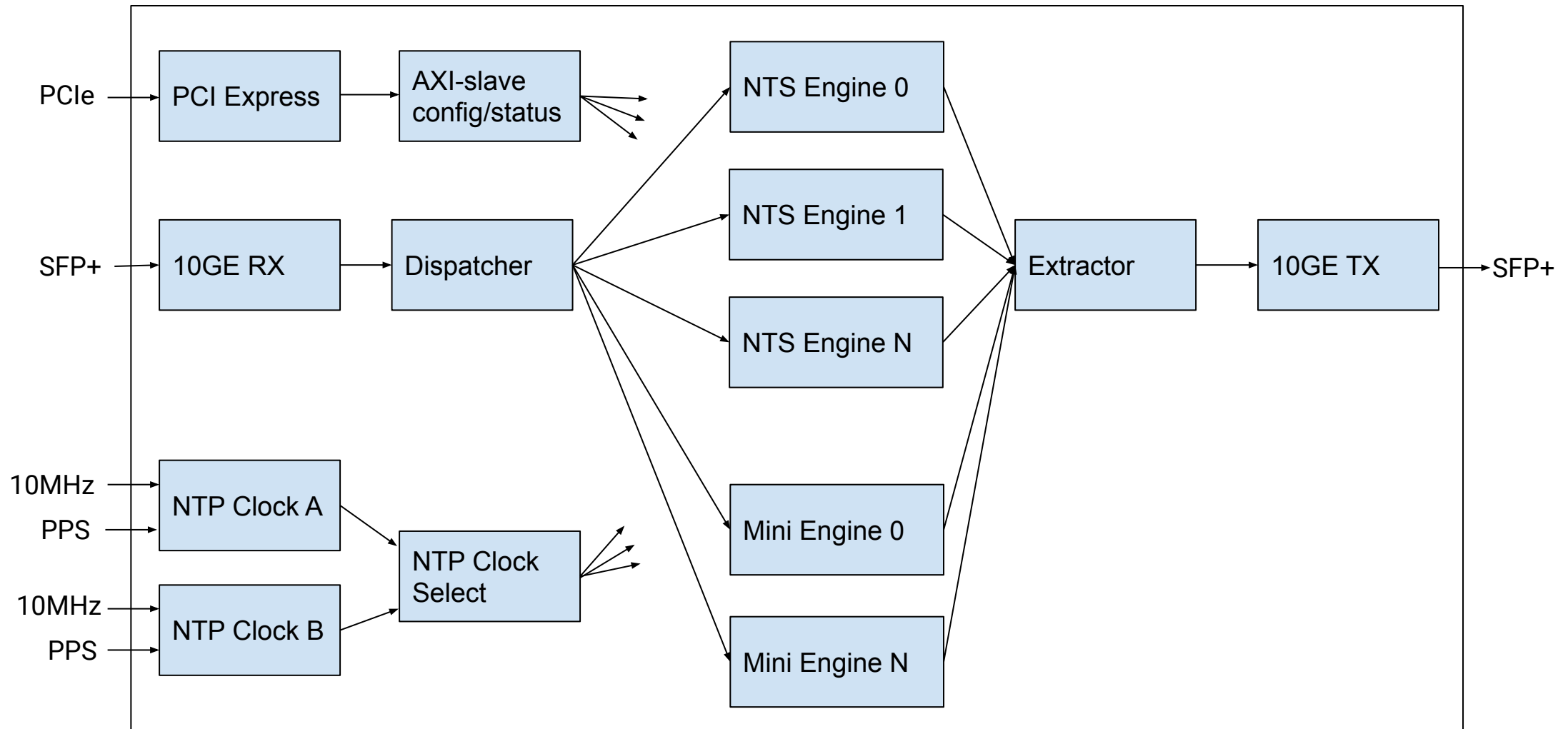
NTS-TS Timestamping



NTS FPGA implementation

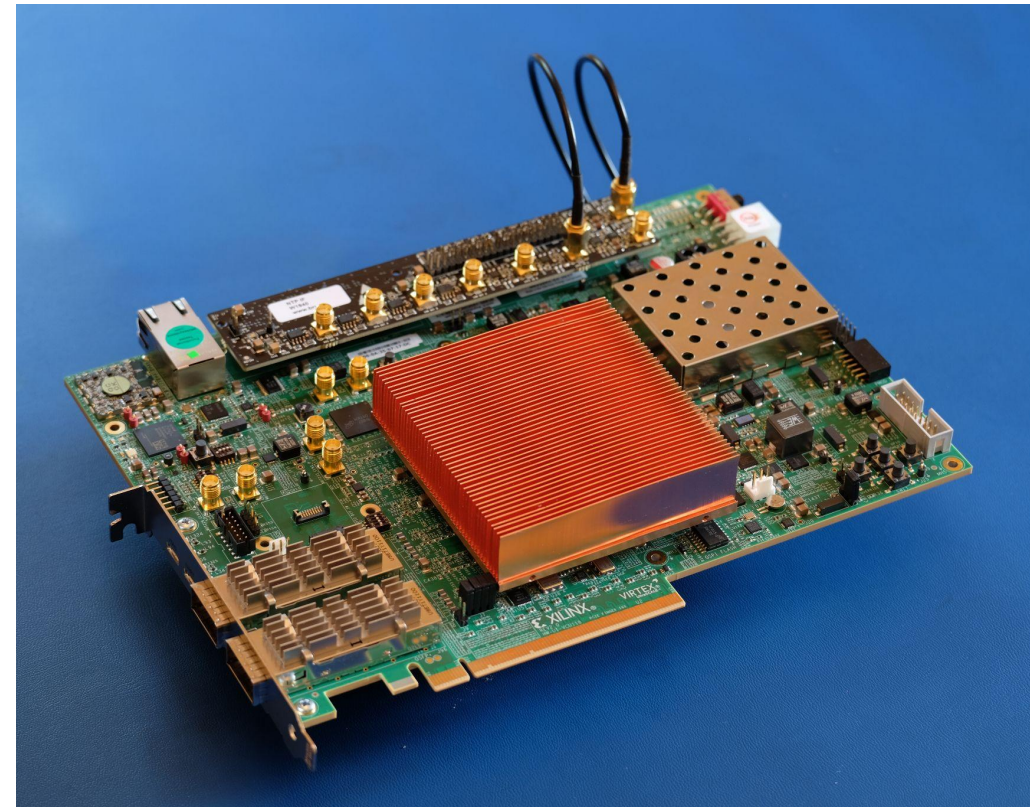
- Netnod have run NTS servers since 2019
 - On a PC
- NTS-TS timestamping can be done in a FPGA
 - NTS is more complex than NTP though
- AES-SIV (AES Synthetic Initialisation Vector)
 - Authenticated Encryption with Associated Data (AEAD)
 - Requires multiple passes over data
 - A streaming implementation is hard
 - Buffering is needed
- Many small NTS engines working in parallel

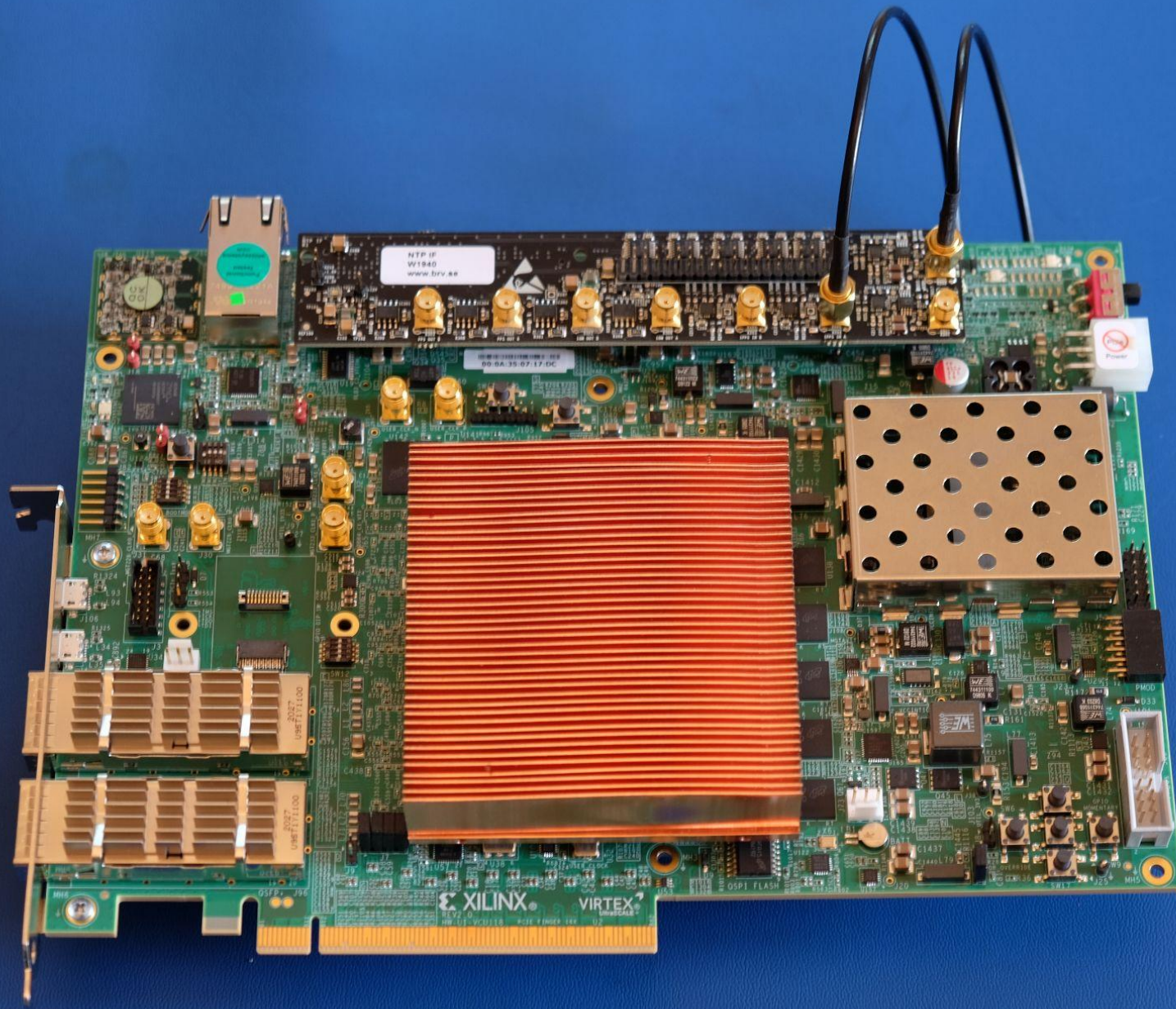
NTS FPGA architecture



Key points for new NTS FPGA

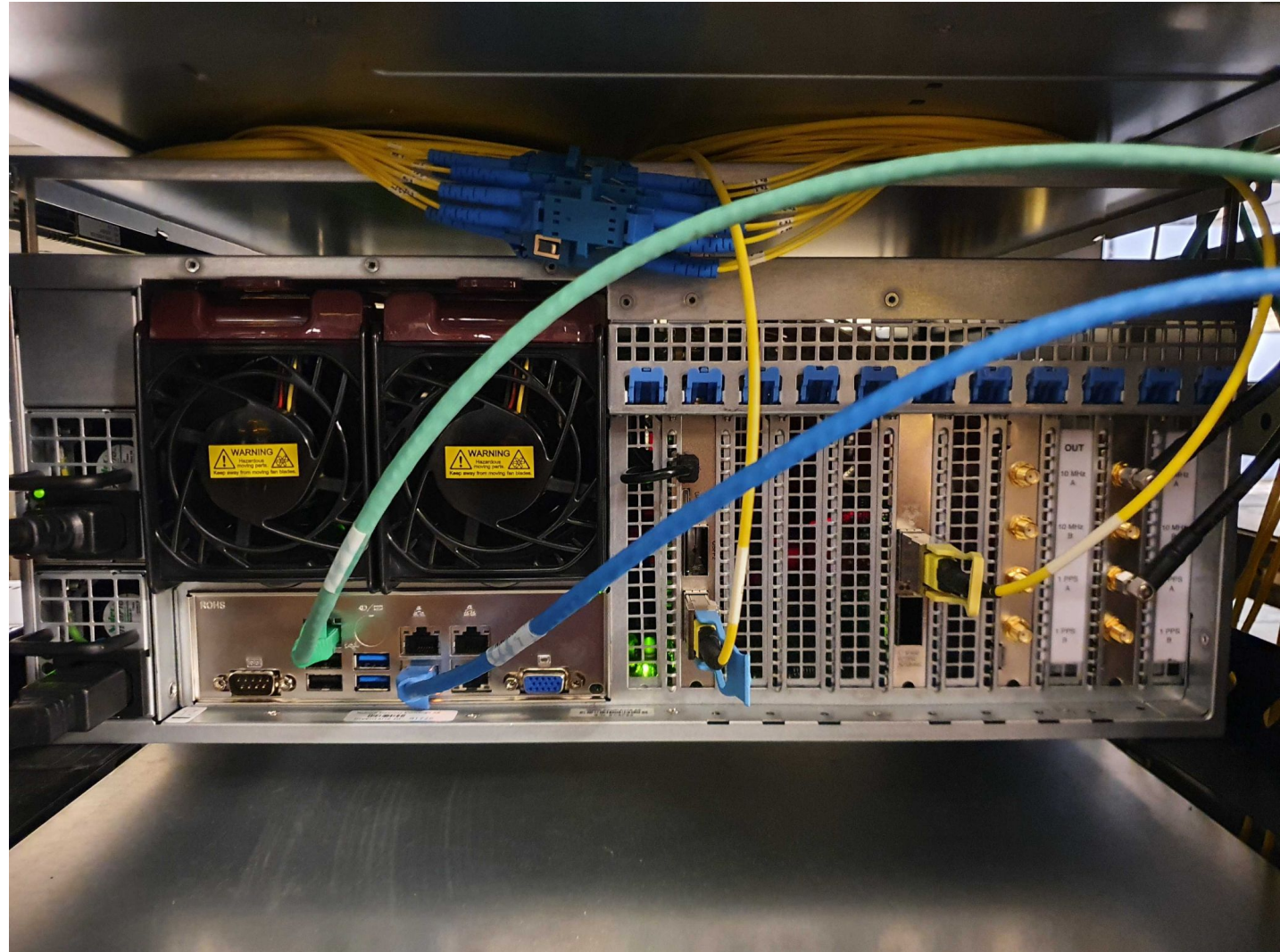
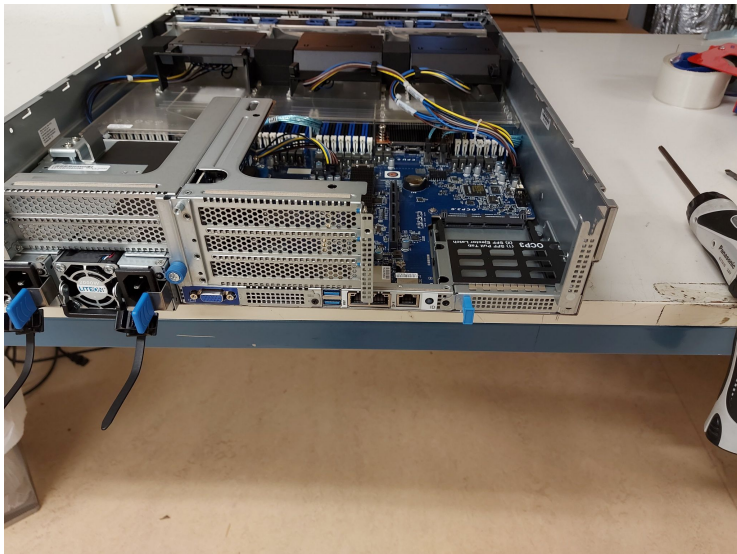
- Developed on a same VC709 reference board as NTP
 - Fits 16 NTS engines
 - Can handle 3Gbit of traffic
- Switched to a larger FPGA
 - Xilinx VCU118 reference board
 - about 2.4 million LCs
 - Fits 40+ NTS engines
 - Handles 10Gbit of traffic





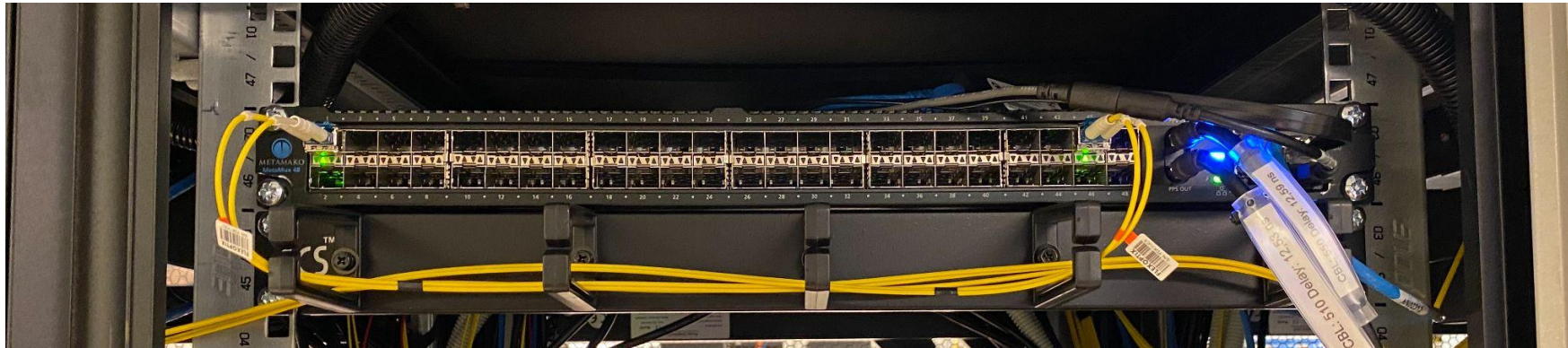
VCU118 issues

- VCU118 won't fit in a 4U PC
- Creative solutions



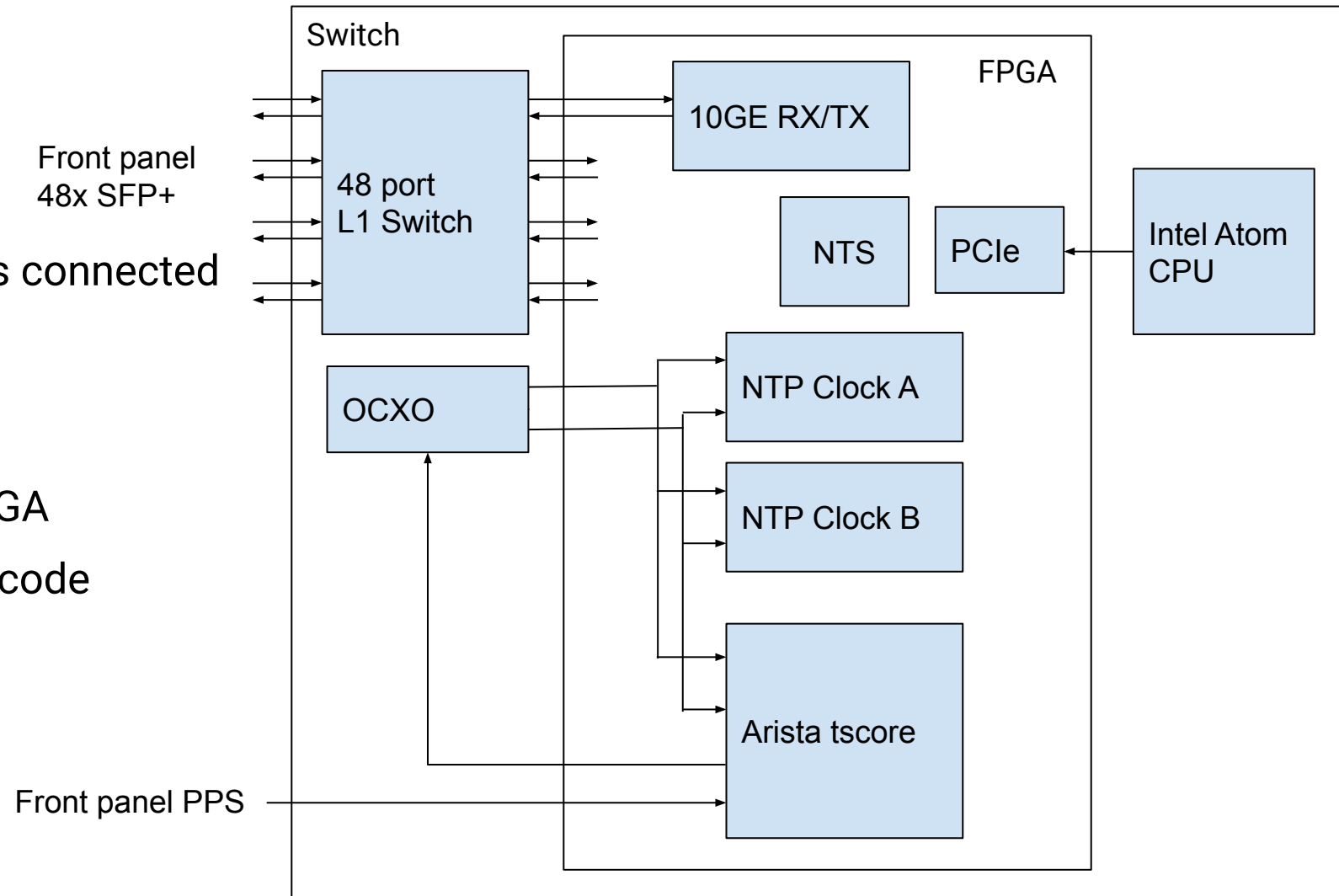
FPGA in Arista 7130L

- Production code runs on an Arista 7130L
 - Similar FPGA to VCU118
 - Standalone, 1U, redundant power supplies



Arista 7130L implementation

- Intel Atom CPU
- Front panel 10GE ports connected to L1 switch
- Front panel PPS input
- Internal OCXO delivers 10MHz and PPS to FPGA
- Same core NTS FPGA code
- Arista "tscore"
- Arista CLI

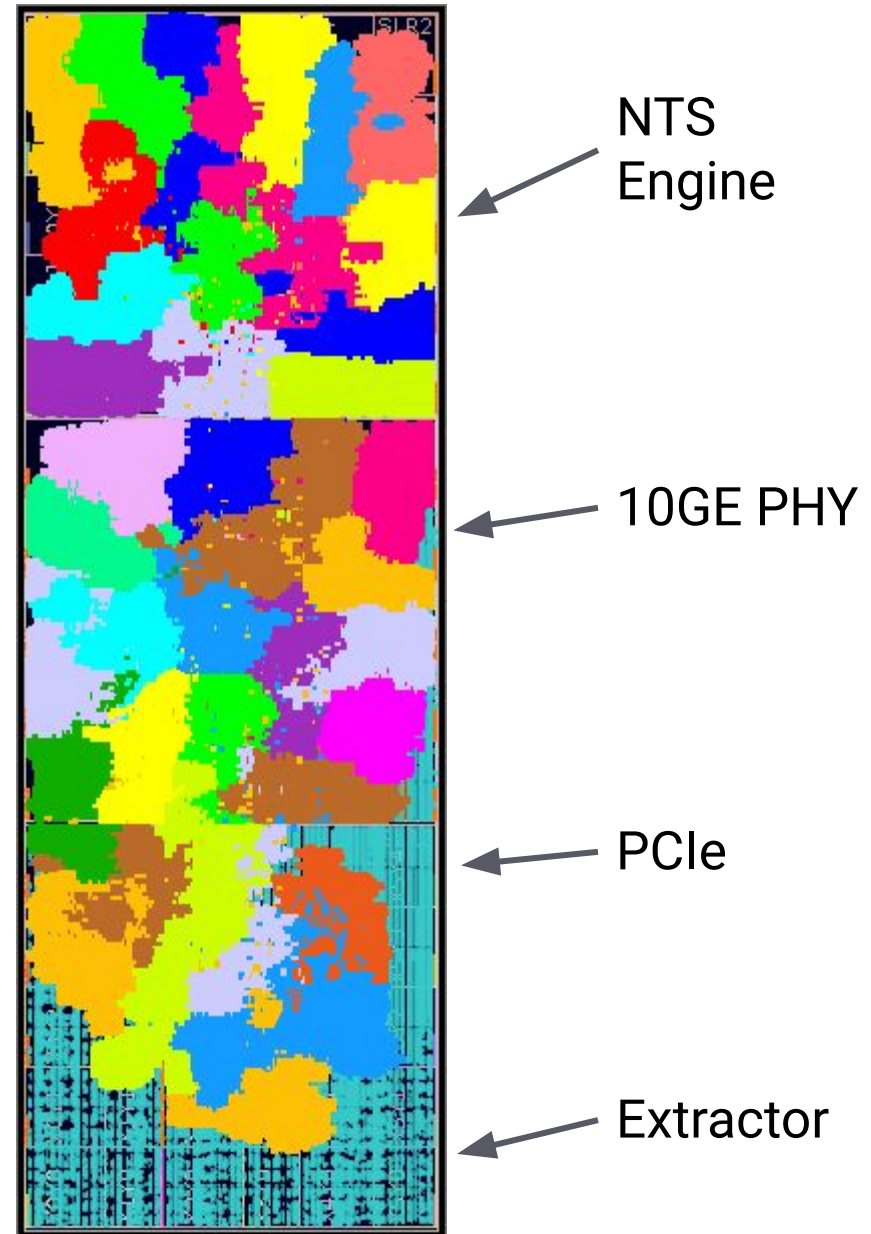


Key points for Arista implementation

- Same same but different
- L1 switch must be configured
- Uses internal OCXO instead of external 10MHz clock
 - Arista "tscore" handles synchronisation of OCXO to front panel PPS input
- FPGA configuration using Arista MOS CLI
 - Extensions written in Python

FPGA utilisation

- 40 engines (colored)
 - 2.2% each - 88% of total FPGA usage
- 10GE PHY
 - 0.7% of total FPGA usage
- PCIe
 - 3.3% of total FPGA usage
- Dispatcher
 - 3.1% of CLBs, only 1.1% of CLB LUTs or 0.3% of CLB Regs
 - Spread out all over FPGA
- Extractor
 - 15% of CLBs, only 11% of CLB Regs or 3% of CLB LUTs. Sparse.



NTS latency / jitter

- Latency not as predictable as for NTP
 - Engines buffer data
 - Reordering of packets
 - Adds a few microseconds of jitter
- Doesn't matter
 - The first router will add tens of microseconds of jitter
 - A few internet hops adds hundreds of microseconds of jitter
 - Same is true for NTP so the extra jitter for NTS is insignificant

NTS FPGA improvements

- Many engines cause some issues
 - Complex dispatcher and extractor
- Possible optimisations?
 - Each engine is fairly small and slow
 - Better to have fewer and slightly larger but faster engines?
 - Can simplify dispatcher and extractor

NTS FPGA summary

- NTS FPGA works fine
 - Handles 10Gbit and in production on Arista 7130L
 - Follows standards
 - Works with NTPsec and Chrony (available in Debian 11 bullseye)
 - Some room for improvement

Questions?

- Netnod Time & Frequency

<https://www.netnod.se/time-and-frequency>

- Netnod NTS White papers

<https://www.netnod.se/time-and-frequency/white-paper-how-does-nts-work-and-why-is-it-important>

<https://www.netnod.se/netnod-white-paper-on-the-worlds-first-nts-hardware-implementation>

- VC709/VCU118/Arista code published on github (BSD license)

https://github.com/Netnod/FPGA_NTP_SERVER

- Production NTS servers

sth1.nts.netnod.se, sth2.nts.netnod.se

- I'm Christer Weinigel <wingel@netnod.se>

Feel free to mail me if you come up with any questions later

netnod.se



Bonus slides