Libre-SOC Power ISA 180nm ASIC

Concept to completion:
Why we chose nmigen and OpenPOWER
What happened along the way

IIT Roorkee 2021

Sponsored by NLnet's PET Programme

July 16, 2021

Why start a new SoC?

- Intel Management Engine, Apple QA issues, Spectre
- Zero transparency: commodity hardware cannot be audited.
- Endless proprietary drivers, "simplest" solution:
  License proprietary hard macros (with proprietary firmware)
  Adversely affects product development cost
  due to opaque driver bugs (Samsung S3C6410 / S5P100)
- Alternative: Intel and Valve-Steam collaboration
  "Most productive business meeting ever!"
  https://tinyurl.com/valve-steam-intel
- Ultimately it is a strategic *business* objective to develop entirely Libre hardware, firmware and drivers.

Why OpenPOWER?

- Good ecosystem essential
  linux kernel, u-boot, compilers, OSes,
  Reference Implementation(s)
- Supportive Foundation and Members
  need to be able to submit ISA augmentations
  (for proper peer review)
- No NDAs, full transparency must be acceptable
  due to being funded under NLnet's PET Programme
- OpenPOWER: established for decades, excellent Foundation,
  Microwatt as Reference, approachable and friendly.

Choosing an HDL

- 3 months careful evaluation: Chisel3, MyHDL, Verilog, VHDL, migen, nmigen
- Love the OO capabilities of Chisel3, shame about the TIOBE Index (below 20th)
- MyHDL is in python, but limits to a subset of python. Also, community not as large as for nmigen.
- migen extremely inconvenient: errors created and not caught until ASIC synthesis level!
- Verilog is 1980s (like BASIC and FORTRAN) - considered "machine code"
- VHDL is great (has records) but still no OO capability
- Ultimately nmigen chosen due to large community, and because python is 3rd on the TIOBE index (contd)

Why nmigen?

- Uses python to build an AST (Abstract Syntax Tree). Actually hands that over to yosys (to create ILANG file) after which verilog can (if necessary) be created
- Deterministic synthesiseable behaviour (Signals are declared with their reset pattern: no more forgetting "if rst" block).
- python OO programming techniques can be deployed. classes and functions created which pass in parameters which change what HDL is created (IEEE754 FP16 / 32 / 64 for example)
- python-based for-loops can e.g. read CSV files then generate a hierarchical nested suite of HDL Switch / Case statements (this is how the Libre-soc PowerISA decoder is implemented)
- extreme OO abstraction can even be used to create "dynamic partitioned Signals" that have the same operator-overloaded "add", "subtract", "greater-than" operators

Software Engineering Techniques

- Entire project run along standard Libre Software Project Management: public bugtracker, public mailing lists, public IRC, public git repositories, Charter http://libre-soc.org/charter/
- None of our engineers are Hardware trained! They are all software developers who learned HDL!
- The other way round makes life much harder.
- Like Microwatt, we develop unit tests, use git revision control, and bugtrackers (TODO, still - set up Continuous Integration)
- Unlike Microwatt, we also have a wiki for all documentation (backed by a git repository with full revision control)
- Software Engineering techniques are critical for large project management! unit tests, unit tests, unit tests...

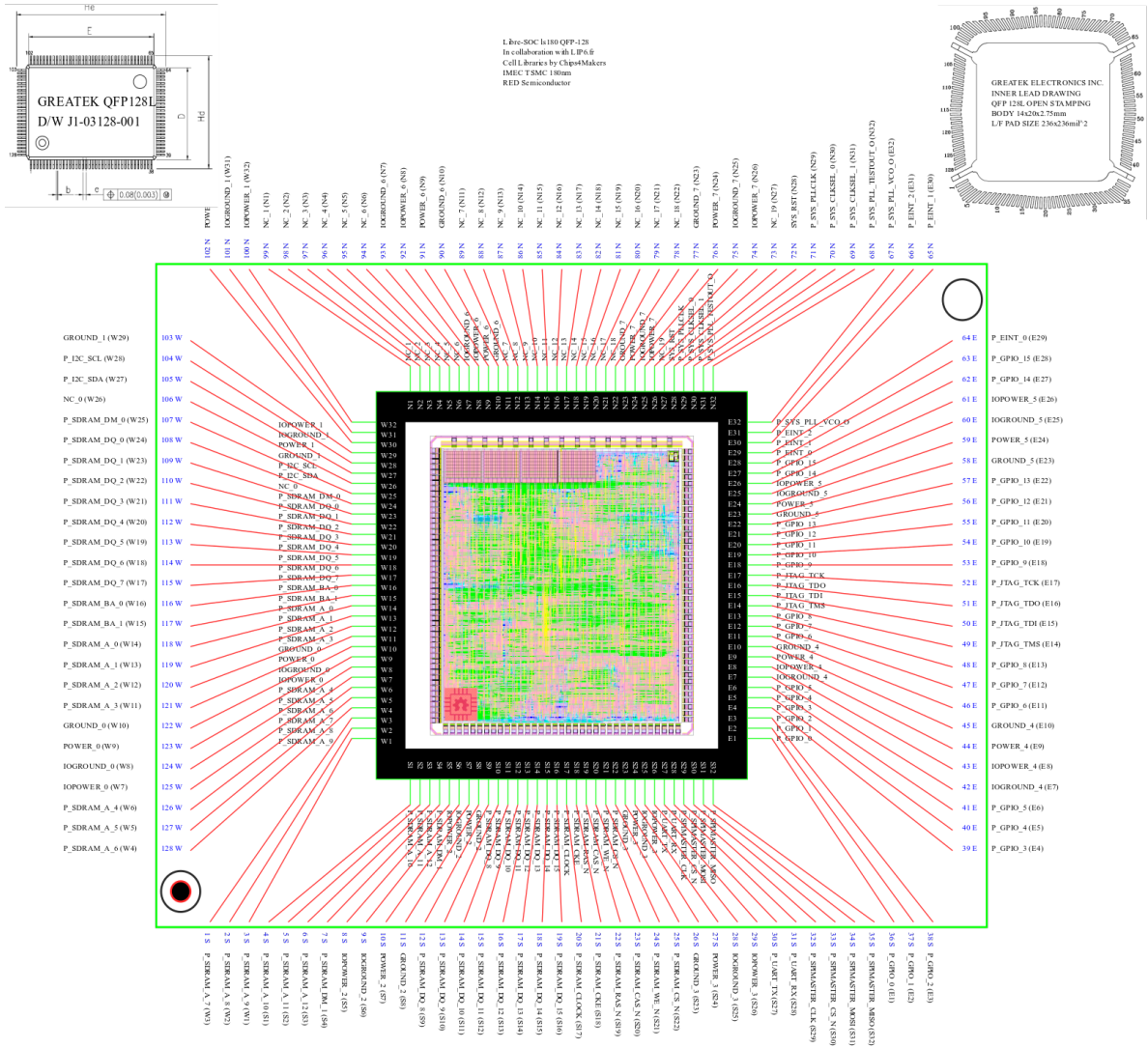
Auditability and Transparency

- Funded by NLnet's "Privacy and Enhanced Trust Programme"
- Matches well with business objectives to provide customers with ability to inspect down to Gate Level (GDS-II files)
- (Impossible for Intel etc to do, due to 3rd party licensing of 50-60 pieces of HDL).
- Sorbonne University LIP6.fr http://coriolis2.lip6.fr is an entirely Libre-licensed VLSI toolchain
- (coriolis2 is "Zero config" from HDL to GDS-II)
- Chips4Makers.io Cell Library - FlexLib - has "Symbolic" and "Real" versions. Symbolic is public and published, uses FreePDK45 "Real" is NDA'd with Foundry.
- Therefore: Libre-SOC team can work side-by-side with LIP6.fr WITHOUT signing a Foundry NDA.

Development Process

- nmigen HDL, in python, converted to verilog using yosys
- developed several thousand unit tests at every level
- Jean-Paul Chaput improved coriolis2 to cope with automated layout of 130,000 Cells. Antenna, buffers, etc.
- Chips4Makers FlexLib Cell Library also needed development, including a custom 4k SRAM block (under NDA, sorry)
- Professor Galayko developed a Voltage-Controlled PLL, capable of between 150 and 900 mhz.
- Marie-Minerve Louerat developed and ran HITAS (Static Timing Analysis tool).
- Tried simulating the laid-out ASIC, required too much resources. Instead, simulated smaller ASICs

ASIC diagram sent to TSMC



The end

Thank you

Questions?

- Discussion: http://lists.libre-soc.org
- Libera IRC #libre-soc
- http://libre-soc.org/
- http://nlnet.nl/PET
- https://libre-soc.org/nlnet/#faq