LiteDIP

Bridging the gap between open hardware, and open OSes

2020/09

Martin Peres
Agenda

- Why did I start a new project?
- FPGAs: Why people should care now?
- What’s missing?
- LiteDIP
Why did I start a new project?

- After Nouveau and Intel GFX CI, I needed a new hobby!
  - Nouveau taught me reverse engineering and elegant HW designs
  - Intel GFX CI taught me about quality, and classes of bugs
  - Work on the Intel’s display driver motivated me to make my own

- Ben Widawsky called for an open GPU last year
  - I dismissed it too quickly, why would it now work?
FPGAs: Why care now?

- Open drivers are fun, but what’s next after production-readiness?

- What changed in the FPGA world?
  - HW capabilities: We can implement 10 years old hardware for cheap!
  - Languages / Workflows:
    - Open source flows now provide full support for multiple FPGAs
    - Better programming languages are now available
    - Plenty of powerful IP blocks are available
FPGAs: What can we expect?

~ $70

~ $3000 used
FPGAs: What can we expect?

- From thousands of LUTs to over a million!

- Fixed functions blocks
  - Integrated ARM CPU
  - Serializer/Deserializers: Enabling up to 100s of GBit/s links
  - DSP: Multiply and Add blocks, providing up to multiple TOPs
  - Memory controllers: DDR3/4, HBM2
  - PCIe controller
  - And more...
Symbiflow: The GCC of FPGAs

<table>
<thead>
<tr>
<th>Basic Tiles:</th>
<th>Lattice ice40</th>
<th>Lattice ecp5</th>
<th>Xilinx 7 Series</th>
<th>Vendor supported!</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Block RAM</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Advanced Tiles:</th>
<th>Lattice ice40</th>
<th>Lattice ecp5</th>
<th>Xilinx 7 Series</th>
<th>Vendor supported!</th>
</tr>
</thead>
<tbody>
<tr>
<td>DSP</td>
<td>✔️</td>
<td>✔️</td>
<td>×</td>
<td>✔️</td>
</tr>
<tr>
<td>Hard Blocks</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Clock Tiles</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>IO Tiles</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Routing:</th>
<th>Lattice ice40</th>
<th>Lattice ecp5</th>
<th>Xilinx 7 Series</th>
<th>Vendor supported!</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
<tr>
<td>Clock</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>
Migen / LiteX: Bringing OOP for HW design

- Python is really well-suited to templatize IP blocks
- Migen is a python-based Hardware-Description Language (HDL)
- LiteX is a library of HW blocks which can mix-and-match
  - Blocks are exposed through a wishbone bus, accessible through PCIe/ETH/UART
Overview of the LiteX capabilities
LiteX + Symbiflow = ❤

- Quad core RISC-V running Linux on a Series 7 FPGAs @100 MHz?
Open source boards - ULX3S ($115-$155)
Open source boards - NeTV2 ($215+)
What’s missing?

- Upstream Linux drivers!
- Requires a way to know where blocks are on the bus
  - LiteX is going with the device tree
  - LiteDIP is going with discoverability straight on the bus
LiteDIP: Current state

- Core: Iterating on more effective ways of getting discoverability

- Testing:
  - HW unit tests for all blocks
  - Driver unit tests with CMocka
  - TODO: Integration testing with QEmu and verilator?

- Current blocks supported
  - Identification of the device
  - Sensors: Expose any kind of sensor, along with the calibration function
  - Fan controller with tachometer / HWMON

- See for yourself: https://gitlab.freedesktop.org/mupuf/litedip/-/tree/fan_wip/
LiteDIP: When to expect a full GPU?

- Let’s focus first on exposing the current LiteX blocks!
- My next objective is writing a USB/PCIe display controller
  - Demonstrate generating SoCs on two different boards and manufacturers
  - ETA: First demo at XDC!
  - NOOOPE, life got in the way!
Questions?

- Thanks for your attention!