High Level Synthesis and Vitis

Lucy Lewitt

April 11, 2025

Agenda

Common Tools:

- Summary of the ecosystem
- Using streams

Hls Tutorial:

- Target audience:
  - Firware developers (Hls and Hdl)
  - Software developers (Cpu and Gpu)
- TBC: Hosted by RAL/Third week of august
- One day (afternoon then morning)

Hls Development

Getting started with Hls can be complex:

- Build system (makefiles, v++, vitis)
- Development environments (vitis, vitis_hls, vitis –classic, vivado)
- Host code (OpenCL, Xrt)
- Installing all the software

Even more so once you start optimising:

- Hls pragmas
- Task parallelism (pipelining, dataflows)
- Data parallelism (unrolling, array partitioning)
- Floating point vs fixed point
- Hls low level libraries (vector, ap_int, stream, ...)

Resources and Tools

Widely available resources and tools:

- https://github.com/Xilinx/Vitis-Tutorials
- https://github.com/Xilinx/Vitis-HLS-Introductory-Examples
- Vitis libraries
  - Matrix solvers
  - Fourier transforms
  - Lots more not directly applicable to HEP
- hls4ml

Things I'm working on:

- https://gitlab.cern.ch/llewitt/HlsFramework
- Vitis in CMake/CTest
- Struct (de-)serialisation

Streams

- Hls designs must be modular to benefit from shared tools
- This is where streams become important
- Shared register with handshake (Valid + Ready)
- FIFO, asynchronus communication

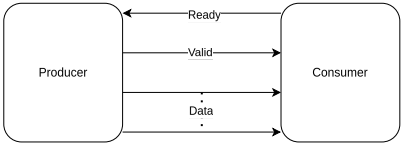
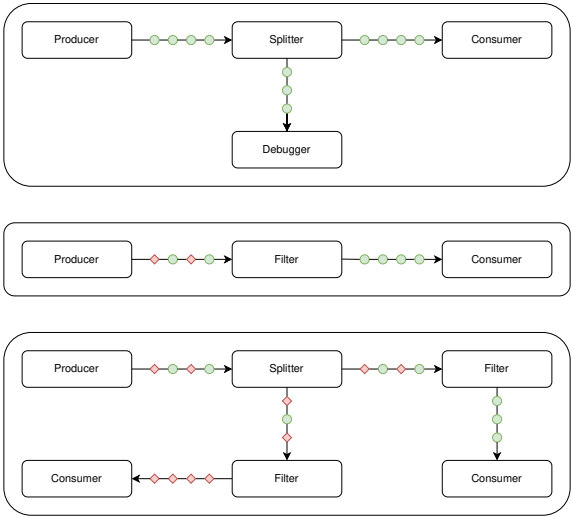


Figure: Simplified Axi-Stream pin out.

```
void square(
  hls::stream<int>& input,
  hls::stream<int>& output
) {
  while (true) {
    const int number = input.read();

    output.write(number * number);
  }
}
```

Streams


Producer
Splitter
Consumer
Debugger
Producer
Filter
Consumer
Producer
Splitter
Filter
Consumer
Filter
Consumer

Hls Tutorial: Elevator pitch

Topics:

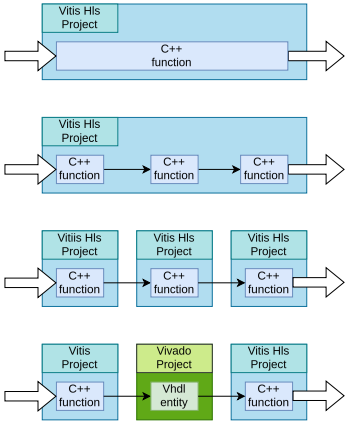
- The focus:
  - Using streams to create modular firmware
  - Task parallelism
- Provided but not emphasised:
  - Build system (based on CMake)
  - Host opencl code (reusable for all tutorial examples)
- Not included:
  - Data parallelism
  - Vectors and loop unrolling
  - Very well covered in other tutorials
  - https://github.com/Xilinx/Vitis-Tutorials
  - https://github.com/Xilinx/Vitis-HLS-Introductory-Examples

Goals:

- Persistent access to software post-tutorial (accounts provided by RAL)
- Provide resources that can be used in the real world
- Avoid duplicating material
- Be fun!

Hls Tutorial: Rough plan

- Start with a very simple kernel
- Build/run and demonstrate all the tools
- Rework the kernel into three separate functions
- Rework the three functions into separate kernels
- Show a VHDL drop in replacement for the middle kernel
- Hands on with practice questions and solutions



Hls Development: Revisited

Getting started with Hls can be complex:

- Build system (makefiles, v++, vitis) ✓
- Development environments (vitis, vitis_hls, vitis –classic, vivado) ✓
- Host code (OpenCL, Xrt) ✓
- Installing all the software ✓

Even more so once you start optimising:

- Hls pragmas
- Task parallelism (pipelining, dataflows) ✓
- Data parallelism (unrolling, array partitioning)
- Floating point vs fixed point
- Hls low level libraries (vector, ap_int, (stream ✓), ...)

Hls Tutorial: Preflight Checks

- Resources
  - Tutorial content on gitlab
  - https://gitlab.cern.ch/llewitt/HlsFramework
- RAL account creation
  - Form: https://forms.office.com/e/0muGhDFfyr
  - PPD contact: alison.elliot@stfc.ac.uk
- Xilinx/Amd account creation
  - Shouldn't be needed for the tutorial but...
  - Is required if you want to install the tools yourself
  - Reccomend creating one just in case
  - https://www.amd.com/en/registration/create-account.html
- Get in touch
  - Email: llewitt@cern.ch
  - Mattermost: https://mattermost.web.cern.ch/signup_user_complete/?id=mxgki6uhfbby3rc99j1msmai5h&md=link&sbr=su
  - @cern.ch email required
  - Expression of interest:
  - https://docs.google.com/spreadsheets/d/1rYPqCTaOvM-_F9BIrBF_djm-XrxsG0pBLHkbxcS7vTI