Modernization of MicroTCA.4 FPGA Firmware Framework.

8th MicroTCA Workshop for Industry and Research

on behalf of MSK Firmware Team

Çağıl Gümüş, Łukasz Butkowski, Burak Dursun
Hamburg, 05/12/2019
Statistics of MSK Firmware Framework

- Version Control: SVN
- Lines of VHDL Code: >460 000
- Lines of tcl: 30 000
- Number of projects: 94
  - Number of active projects: 24
- Number of Board Support Packages: 13
- Number of authors: 45
- Since 2013

Usage examples:
- Multi-Cavity/Single-Cavity LLRF Regulation
  - XFEL, FLASH, REGAE, ELBE, TARLA, NICA, CMTB, AMTF ...
- Laser Synchronization
- Special Diagnostics
- Cavity Tuning (Motor / Piezo)
- Any Light Particle Search (ALPS)
- ...

- Modernization of MicroTCA.4 FPGA Firmware Framework | Cagil Gumus, 05/12/2019
MSK FPGA Firmware Structure for MicroTCA.4 Systems
Framework Structure

- Tree-like structure for the address mapping.
- Proprietary buses between Board ↔ Application:
  - IBUS (Internal Memory Access Bus): Creating registers inside FPGA to be connected to PCIe
  - DAQ (Data Acquisition Bus): Used to write to external memory (for devices older than Virtex6)
  - LLL (Low Latency Link): Uses Ports 12 – 15 to communicate with other FPGAs on the crate (MGT)
Future Framework Changes
Our agenda for the Future

1. Change Version Control to GIT
   Each module will be a separate Git Repo
   Super project combines needed git-submodules under one roof

2. Coding style / Folder Structure change
   • New naming convention that is similar to ‘industry’
   • Strict documentation/comment style → New doc folder

3. Getting rid of properity buses
   • DAQ, LLL and IBUS will be replaced by AXI.4 buses.

4. Expand Verification and Continuous Integration Methods
Version Control Change to **git**

- Switching from centralized to distributed version control system
- Tagging entire trunk on svn is not viable since there is continuous development. We can use submodules to overcome this issue
- Branching is a lot easier on git
- Working offline is possible with git

We are going to open parts our framework on **GitHub**
Automatic documentation extraction to Wiki
Switching to AXI.4

- AXI is the standard for Xilinx IP-Cores
- Packaging our modules into IP
- Sharing is easier.
Verification and Continuous Integration/Deployment (CI/CD)

- **Background:**
  - As complexity increases, verification for algorithm becomes even more critical and harder to implement.

- **Problem:**
  - Our framework is lacking a structured way to verify the components functionality and apply randomization on design parameters.

- **Main goal:**
  - Minimize the FPGA related down-time on deployed systems.
  - Catch bugs faster.
  - ‘Stress test’ the code.

- **Strategy:**
  - Use well known verification methodology (UVVM, OSVVM ..) + DSP Specific in-house verification.
Thank you!

Questions?