flexSD

Reconfigurable Hardware Platform for Control System Development and DSP Using ΣΔ Techniques

Alec Dibble, Dan Kouba
Mentor: Prof. Forrest Brewer

ABSTRACT

Control Systems are becoming ubiquitous. Common examples include thermostats, ABS brakes, manufacturing processes, robots, and autopilot. Unfortunately, the tools to develop these systems are slow, have high power requirements, and are cost-prohibitive for educational and research use. Currently, it is common practice for dynamical system control to be taught using the Simulink® simulation and design suite running on a PC, and an external hardware interface is used to communicate with the devices under control.

The goal of FlexSD is to provide a high speed, low power, and low cost platform for control and signal processing experiments. The control algorithms are performed on a field programmable gate array (FPGA) that is interfaced to an ARM processor. By using an FPGA as a controller, high speeds are possible because the FPGA is clocked significantly faster than the incoming data stream. The ARM processor can reprogram the FPGA's memory mapped filters and log data off of running experiments without modifying the loaded bitstream. This framework is advantageous as it can be configured without requiring time-intensive logic synthesis for every experiment revision. A MATLAB®/Simulink® interface will be provided that would allow a user to write control algorithms in Simulink® and port them to the device.

One of the novel aspects of this project is that the FPGA hosts a custom made filter framework that processes a 1 bit, ΣΔ encoded stream from the analog to digital converter (ADC). This allows a large number of filters to be synthesized on the FPGA and also decreases the routing cost between the filters in the design.

The project will be completely open source. All the code and printed circuit board files will be posted online to make it easy for other universities, companies, and hobbyists to experiment using the platform. The ARM processor and FPGA reside on a single board computer (SBC) that can be inexpensively obtained from Technologic Systems®.

PROBLEMS WE ARE SOLVING

Digital Signal Processing algorithms are typically implemented using multiplication operations. In VLSI, hardware multipliers require large amounts of logic and limit the overall timing of the device. Furthermore, increasing the resolution of the signal significantly increases the size of the hardware multipliers and signal routing structures.

- Using 1-bit ΣΔ modulation and relevant VLSI control structures, hardware multipliers are eliminated and replaced with adders. This implementation greatly reduces hardware size, timing constraints, and provides for a much lower power VLSI implementation.
- In an FPGA implementation, the ratio between the hardware clock and the ADC sampling rate is large. Therefore, using memory based structure design, a single hardware structure can be used several times "virtually" before a new sample is received. This allows high order filters to be implemented using a low order piece of hardware.

Many control system laboratories and teaching facilities use CPU-based experiment implementation. Modern operating systems require a large processor overhead, meaning that the experiment loop rate is greatly affected when using a desktop CPU for this purpose. Simulink® provides a package for synthesizing control systems on an FPGA. Since the algorithms are implemented in digital logic, their loop speed can be much higher than the CPU-based alternative. However, synthesis times can be on the scale of 5 minutes or longer. This greatly reduces the effectiveness of experimentation workflow.

- Our FPGA implementation eliminates the operating system overhead and our memory mapped filter structure allows for algorithms to be updated with new values without resynthesizing the FPGA bitstream. Using a networked interface, experiments can be modified in little time. Additionally, experiments can be datalogged in order to view controller performance and analyze operation.
- Our hardware system is much less expensive than the cost of a dedicated PC and the I/O hardware required for control system experimentation. This will make the control system experimentation on our hardware platform cheaper and easier to integrate with existing PCs.

SYSTEM SPECIFICATIONS

Hardware:
- 4 channel ΣΔ ADC
  - 70kHz bandwidth
  - 16 ENOB
  - Single ended 0-5V input
  - Oversampled, 20MHz data rate
- 4 channel DAC
  - 16 bit
  - Bipolar 5V output
- RCA jacks for I/O
- Status LEDs
- 4 rotary encoder interfaces

FPGA Bitstream and Client Software:
- 200MHz internal clock speed
- 2 custom slices, each virtualized 10x
  - Each slice (Figure 1):
    - Can implement 1, 10th order filter, or more lesser-order filters
    - Up to 1024 signed 18 bit coefficients
    - Up to 16 state variables
    - 1, 10x virtualized 2nd order ΣΔ modulator (Figure 2)
- Coefficients and initial state loaded over Ethernet interface from Simulink parser on PC

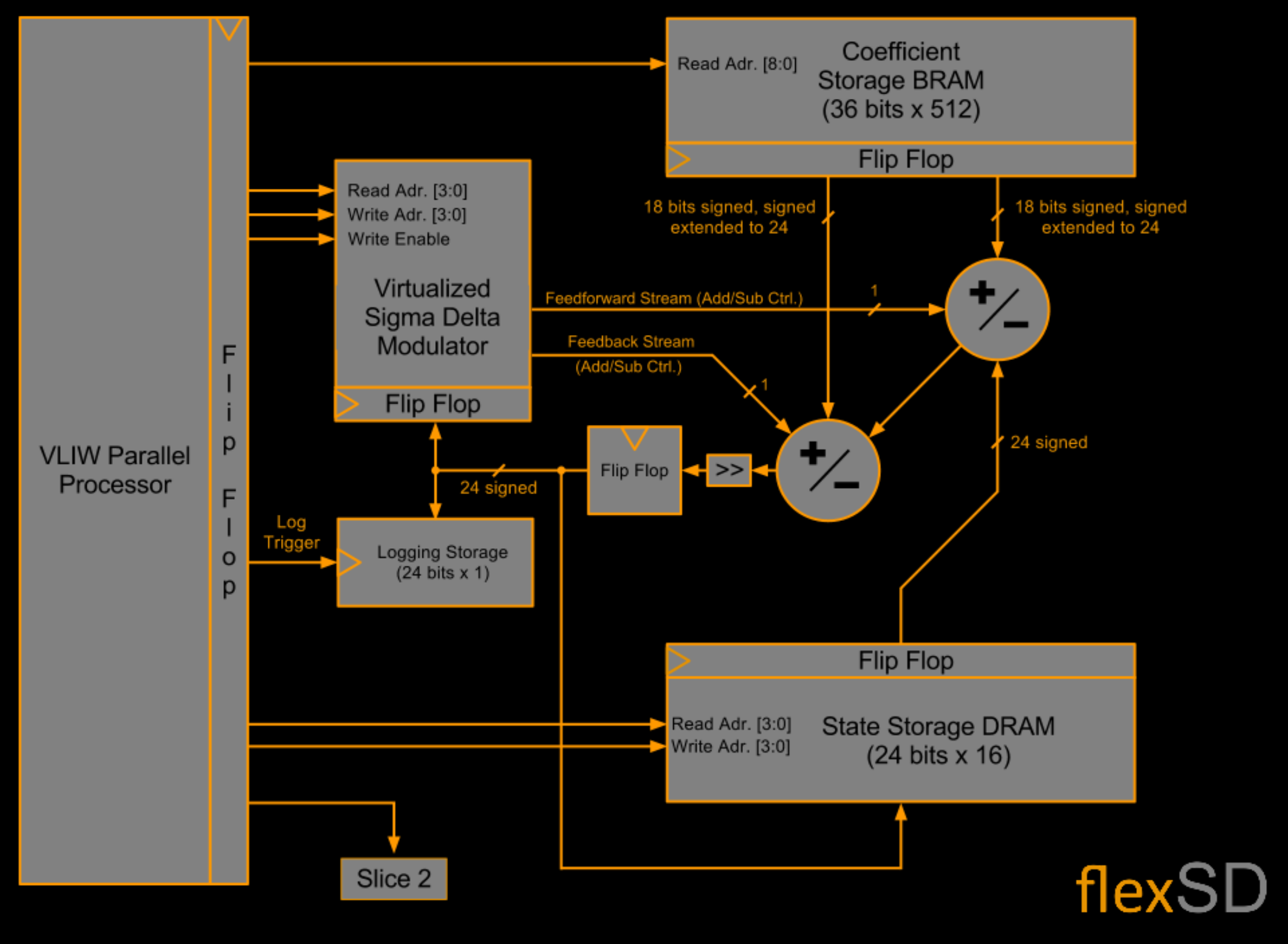
VLIW Parallel Processor
Flip Flop
Read Adr. [8:0]
Coefficient Storage BRAM (36 bits x 512)
Flip Flop
Read Adr. [3:0]
Write Adr. [3:0]
Write Enable
Virtualized Sigma Delta Modulator
Flip Flop
18 bits signed, signed extended to 24
18 bits signed, signed extended to 24
Feedforward Stream (Add/Sub Ctrl.)
Feedback Stream (Add/Sub Ctrl.)
24 signed
24 signed
Flip Flop
Log Trigger
Logging Storage (24 bits x 1)
Flip Flop
Read Adr. [3:0]
Write Adr. [3:0]
State Storage DRAM (24 bits x 16)
Slice 2
flexSD

FIGURE 1: SLICE ARCHITECTURE

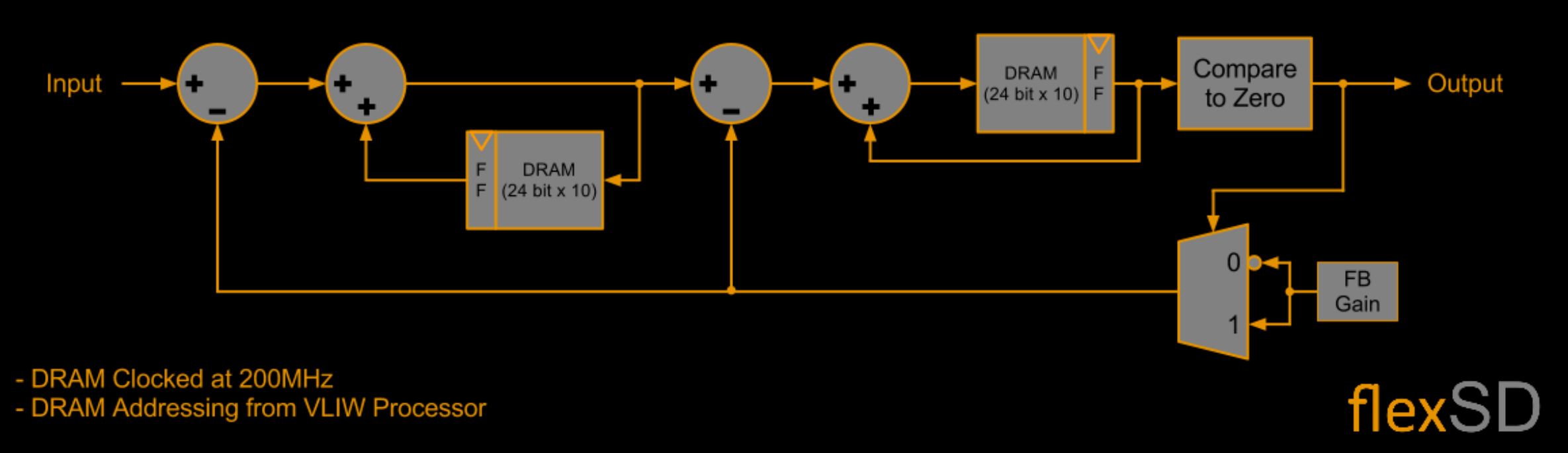

FIGURE 2: VIRTUALIZED 2ND ORDER ΣΔ MODULATOR ARCHITECTURE

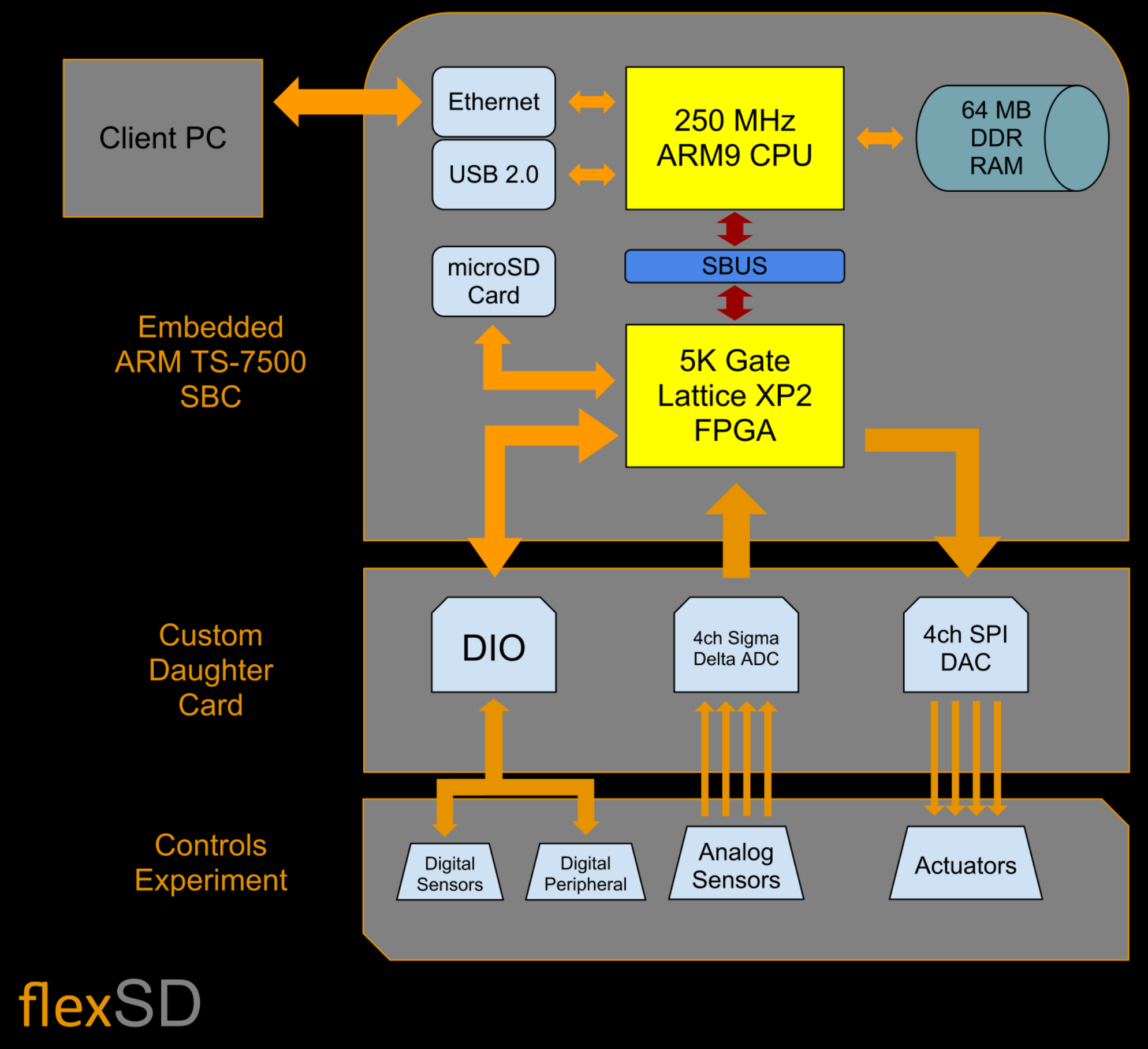

FIGURE 3: HARDWARE FLOWCHART

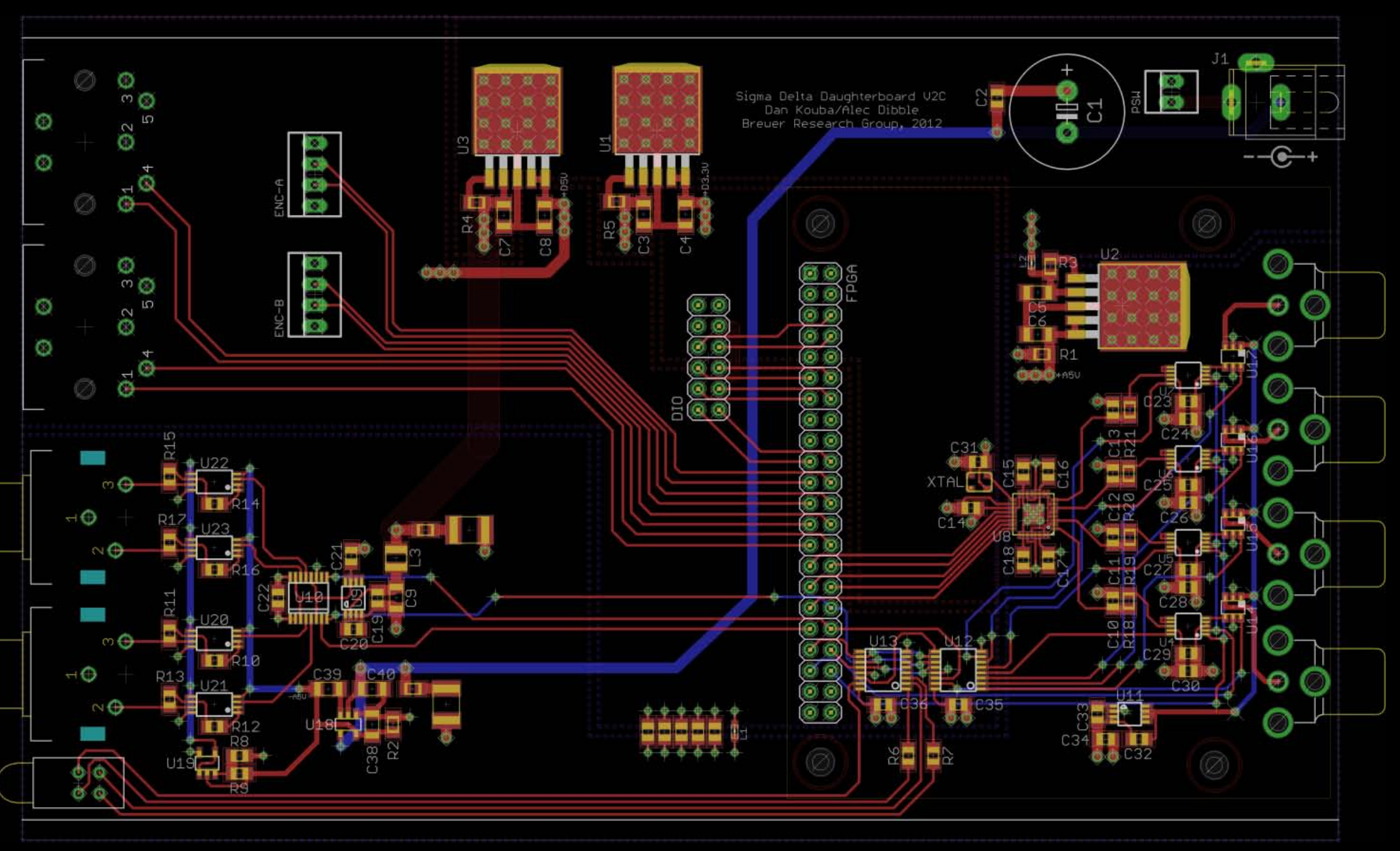

FIGURE 4: BOARD LAYOUT

DESIGN CHALLENGES

An FPGA-Processor interface is required to satisfy the specifications of the design. The first design iteration involved co-habitating a FPGA with a soft processor and a control system. For reconfiguration, the FPGA would have to partially reconfigure the control hardware portion without affecting the soft processor states. Advanced partial reconfiguration proved infeasible using budget FPGAs and so this design was discarded. Additionally, most available FPGA development boards require proprietary connectors that would have significantly raised the price of the design and hassle of connecting a daughter card.

A development board created by Technologic Systems®, the TS-7500, was chosen for the design platform. The TS-7500 is low cost, included the necessary features such as a FPGA and CPU, and the manufacturer guarantees it to have a long life cycle. This also helped avoid a custom board containing the FPGA and CPU and the challenges associated with design and debugging. However, using the TS-7500 introduced many hard constraints that had to be accounted for in our design:

- The FPGA the TS-7500 provides is relatively small (5000 LUTs), requiring us to optimize the control structure and glue logic to a point that would fit within the logic constraints and allow room for additional functionality.
- In order to provide the functionality of multiple high-order filters for all 4 inputs, the filters had to be "virtualized" to save hardware space. By using the FPGA block rams and memory mapping those to the filter structure, a single hardware filter structure can act as multiple virtual structures before having to handle the next signal sample on the input. This is made possible by the fact that there is a 10:1 ratio between the FPGA clock speed and the sampling rate. Instead of having the structure being idle for 9 out of 10 cycles, it can do operations on every FPGA clock cycle.
- The SBUS interface that provides communication between the ARM processor and the FPGA is not ideal for high speed, datalogging purposes. Therefore, event driven software-side code is needed to make sure that the processor can receive all incoming corrections without losing any data. This could be amended by implementing a buffer in the FPGA hardware. However, because of the space constraints, a large buffer is not practical.
- The TS-7500 only provides 500 bytes of ROM space for user programs on the onboard flash. This presented difficult cross-compiling constraints due to shared library and linker difficulties in getting the program size down. The constraint was worked around using the built-in microSD card on the TS-7500. The microSD card is interfaced to the FPGA, meaning once the custom bitstream is loaded onto the FPGA, microSD communication is not possible. This constraint was solved by create a bootup script that stored the program files in the microSD card, loaded them into the RAM, and then uploaded the custom bitstream.

OPERTATION

Once the device is built and configured, operation is simple. FlexSD must be networked with a host PC. Once the network connection is established, FlexSD can be controlled completely over the network. Experiments can be configured, started, stopped, and datalogged. Furthermore, special Simulink diagram blocks can be used to configure the filter structures, assign them to the various inputs and outputs on FlexSD, and setup signal monitoring.

FUTURE IMPROVEMENTS

While FlexSD is very powerful, there are still many additional features that can be added to improve the overall functionality. These are described below:

- State recognition functionality to allow development of hybrid systems
- Hardware function generators
- Additional hardware structures that will provide greater integration with Simulink®
- Client side program to compile arbitrary Simulink diagrams into the ΣΔ structure
- Additional daughter card built for high performance applications
- Improved web interface for experiment control and viewing

ACKNOWLEDGEMENTS

Professor Forrest Brewer
Dr. Ilan Ben-Yaacov
National Science Foundation

Kunal Arya
Merritt Miller
Joseph Poverelli

Mongoose Embedded Server Project©
Technologic Systems®
Lattice Semiconductor®
MyroPCB©
OpenCores©