Implementing SHA1 on a Xilinx Zynq SoC FPGA

ComS/CprE 583 – Project Final Presentation

Group 2: Jay Guild, Nathan Miller, Matt Weber

Background

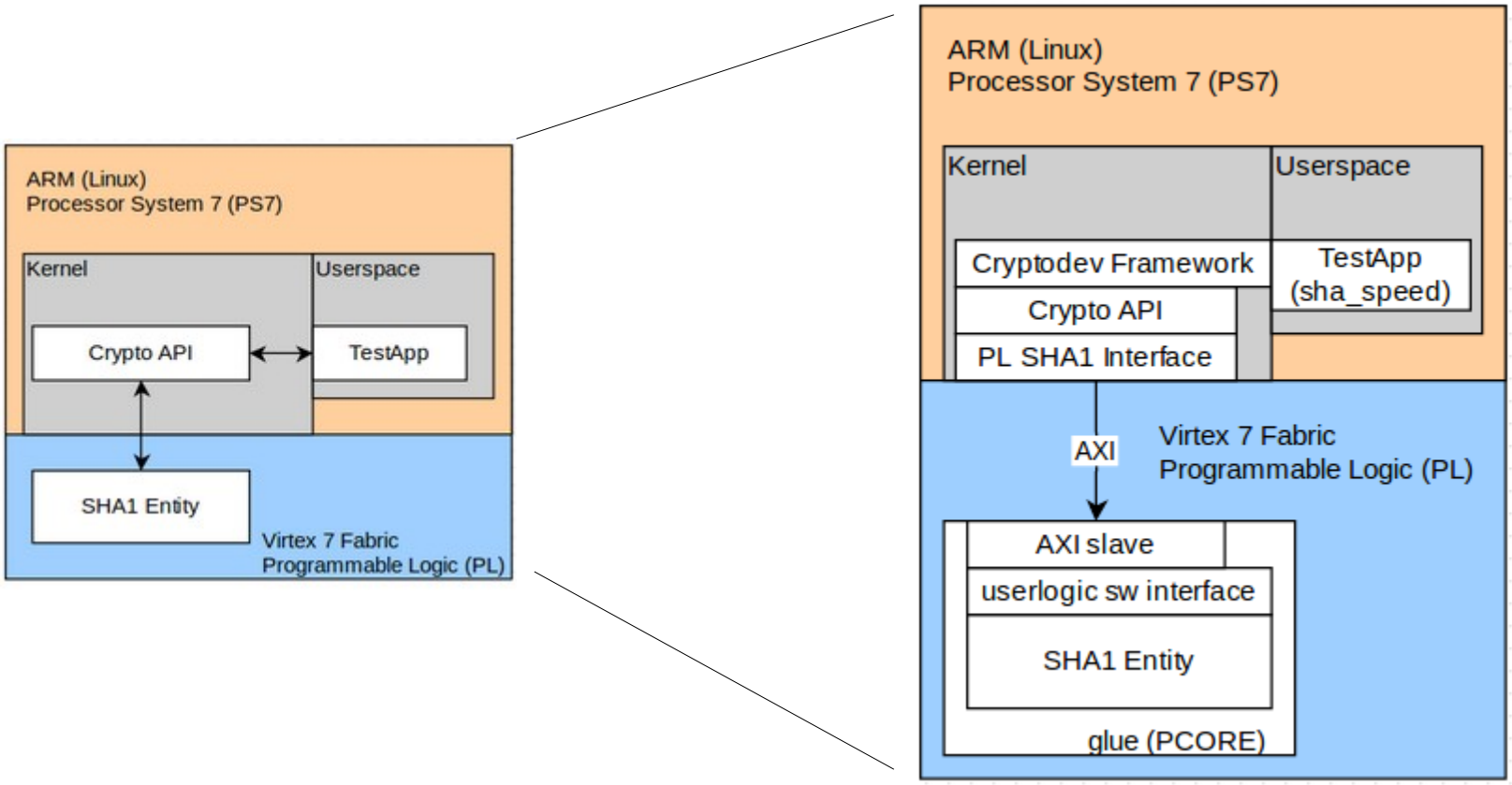
- SoC FPGAs:
  - CPUs + FPGA + Interfacing = SoC FPGAs
  - Recently, FPGA vendors have begun offering “System On a Chip” products.
    - Example: Xilinx Zynq SoC
    - Example: Altera Cyclone5 SoC
  - Potential Advantages:
    - Less power
    - Better performance
    - Ease of programming.

Project Overview

- Implement SHA1 hash on Xilinx Zynq SoC.
- Compare three implementations on Zynq:
  - Pure software (C-based) (ACHIEVED)
  - Hand-coded VHDL (ACHIEVED)
  - Synthesized VHDL from C-source (stretch goal).

- Comparisons:
  - Performance (MB/sec) (ACHIEVED)
  - Resource utilization (logic gates)
    (NA as only one VHDL approach was evaluated)

System Diagram



Software Components

- Linux 3.12
  - Cryptodev Framework (Kernel Module)
    - Gives user-space applications access to crypto functions via a device file API.
  - Crypto API (Linux Kernel)
    - Standard interface for all kernel crypto modules.
  - PL SHA1 Interface Module (Kernel Module)
    - Custom kernel crypto module that interfaced to our PL (aka FPGA).

Software Components Cont.

- Test-App
  - Based on the sha1_speed application that is part of the cryptodev-linux package.
  - Modified to support:
    - Arbitrarily sized messages.
    - Known message sequences.
      - Important to validate the correctness of our hash results.

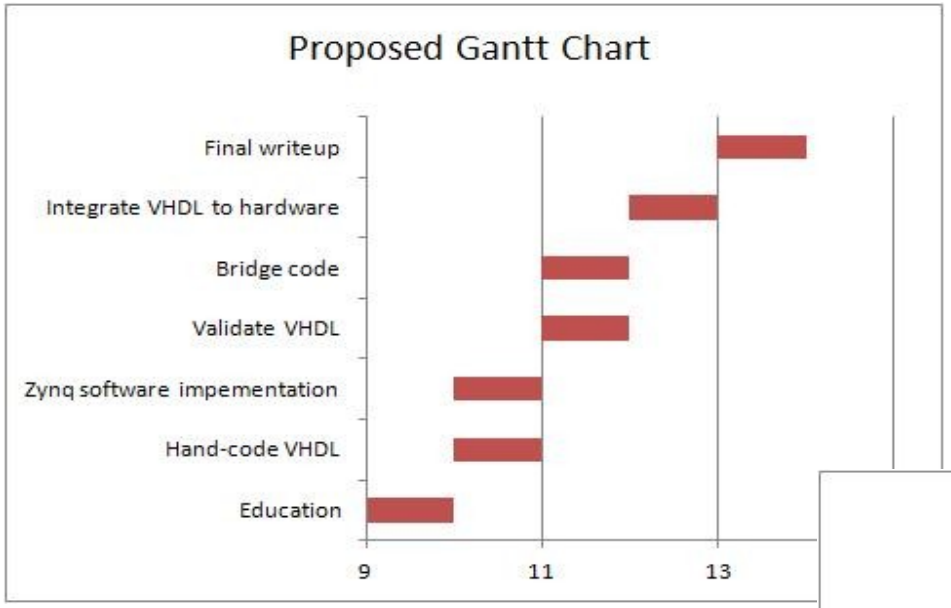
Programmable Logic (PL) Components

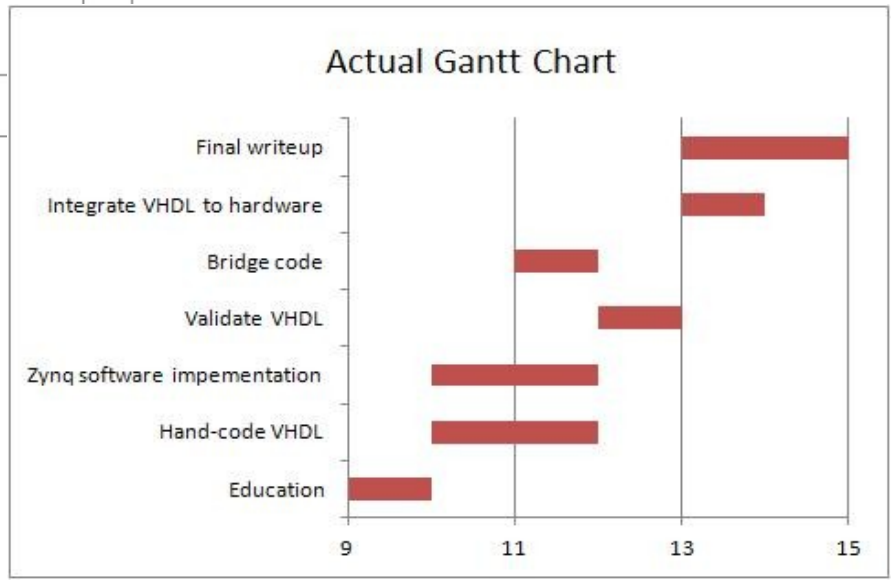
- SHA-1 Core
  - Open-core. Created by Arif Nughoro.
  - Modified to:
    - Stall until a new message data was provided.
    - Use IEEE std_logic and std_logic_vector types.
- Glue
  - AXI bus slave. Interfaced to the CPU.
  - Generated by Xilinx CORE Generator System.
  - Unmodified.

Programmable Logic (PL) Components

- User Logic
  - Generated by Xilinx CORE Generator System.
  - Provided 32 general purpose 32-bit PL registers accessible by software.
  - Modified:
    - To define roles for each of the registers.
      - Status (1), Command (1), Message Data (16), Hash Results (5).
    - To handle stale commands and slow users.
    - To interface with the SHA1 core.
      - New FSM: user_FSM.

Project Schedule





Simulation

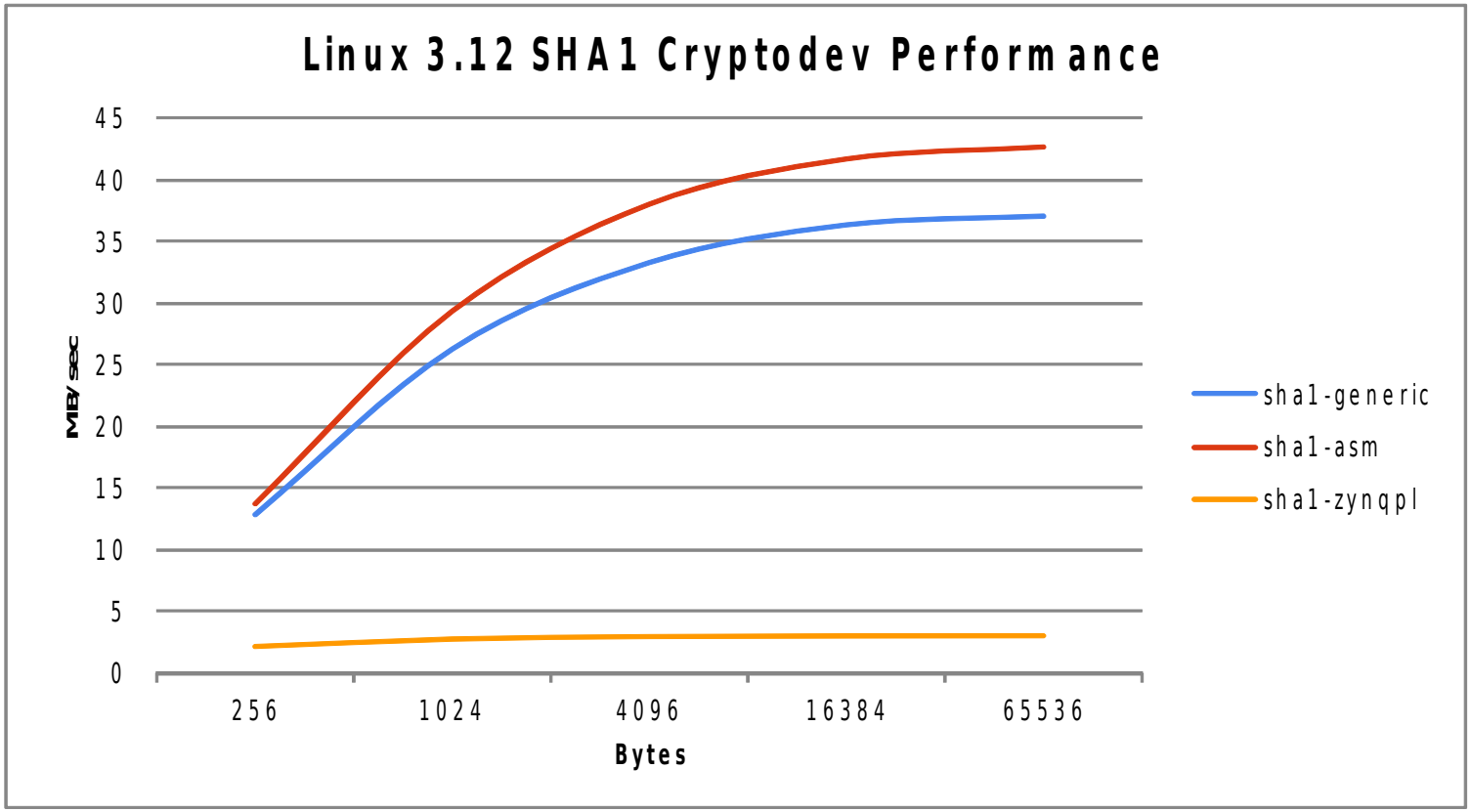
- ModelSim
  - Simulated two components:
    - SHA1 core.
    - User Logic.
  - Used to verify correctness.
  - Used to determine relative duration of FSM states.
  - Test harness + Tests
  - Validated with independent hash comparisons.

Hardware Implementation

- Xilinx Zynq Dev Kit
  - ARM core (667Mhz).
  - Programmable Logic (FPGA) and AXI bus (50Mhz).

Results

- Our FPGA implementation is significantly slower than a pure software based approach!



Analysis

- The SHA-1 algorithm is inherently sequential:
  - Each round of hashing depends upon the immediate result of the previous round.
  - Limited opportunities for parallelism.
  - Mitigation:
    - Pick a different algorithm.
    - Pipeline multiple hash streams through the FPGA.

Analysis Continued

- The SHA-1 algorithm works well on a CPU:
  - Common ALU operations:
    - add, and, or, xor, rotate, mod
  - Common data size: 32-bits.
  - Our CPU runs an order-of-magnitude faster than our FPGA.
  - Mitigation:
    - Pick a different algorithm.
    - Increase the FPGA's clock frequency.

Analysis Continued

- PL got its data from the CPU.
  - Similar to MP3.
  - Mitigation:
    - DMAs
    - Wider data paths.

Grading Rubric

	Highest Level	Middle Level	Lowest Level
Milestones			
SHA1 Implementations	Synthesized VHDL: 30	Hand-Coded VHDL: 20	Software Based: 15
Zynq Development	Modifications to Run Synthesized VHDL: 30	Modifications to Run Hand-Coded VHDL: 20	Stock Linux Kernel Boot (Hello World): 15
Zynq Execution	Synthesized Execution: 30	Hand-Coded VHDL Execution: 20	Software Based Execution: 15
Results			
Project Presentation	Fantastic Presentation: 15	Good Presentation: 10	Poor Presentation: 5
Project Report	Fantastic Report: 40	Good Report: 30	Poor Report: 20
Totals	**145**	**100**	**70**

(110 pts)