University of Pennsylvania
Department of Electrical and System Engineering
System-on-a-Chip Architecture

ESE532, Fall 2017 — Analysis Milestone — Wednesday, October 25

Due: Friday, Nov. 3, 5:00PM

Group: Forming your team, assessing the requirements and parallelism are group tasks for this milestone. You should discuss and refine your understanding of the task as a project team. You may share pseudocode and diagrams and collaborate on performance models.

Individual: Writeup is an individual task.

1. Report the partner you have agreed to work with for the project duration.
2. Working from the high-level description of the computations involved, assess the computational and memory requirements for each of the coarse-grained operations (Content-Defined Chunking, SHA-256, chunk matching for deduplication, LZW encoding).
   (a) Summarize the computation of each coarse-grained operation in pseudocode. (write in your own words; credit sources)
   (b) What memory is needed to support each task?
   (c) What computational work is required per byte of input (or per chunk, where appropriate)?
   (d) Based on this analysis and a simple model of processor execution (define as needed), what throughput can a single ARM processor achieve on this task?
3. Identify and characterize parallelism available.
   (a) What parallelism exists among the operations in the coarse-grained task flow?
   (b) Within each operation, characterize opportunities for or inhibitions to thread-level data parallelism (i.e., what can be processed independently and what must be processed in a sequence).
   (c) Within an operation thread, what opportunity is there for data-level parallelism?
   (d) Within an operation thread, what opportunities exist for pipelined computations? What II is achievable? What dependencies determine the II? What is the depth of the pipelines?

Questions 2 and 3 should give you enough information to begin reasoning about how you can achieve the throughput goals (1Gb/s, 10Gb/s).

We are deliberately asking you to consider these computations from the high-level description rather than a particular piece of sequential code so that you can reason about the fundamental requirements and opportunities apart from the specific artifacts of a particular, sequential implementation.