

UC Berkeley Computer Science

Subject: Four Views of FLEET
Date: November 2, 2005
From: Ivan Sutherland
UCIES #2005-is12

References:

UCIES# 2005-is02: FLEET – A One-Instruction Computer, Ivan Sutherland, 24 August 2005
UCIES# 2005-is03: Defining Some SHIPs, Ivan Sutherland, 24 August 2005
UCIES# 2005-is04: A Dozen Problems, Ivan Sutherland, 6 September 2005
UCIES# 2005-is05: Notation Questions, Ivan Sutherland, 12 September 2005
UCIES# 2005-is06: Some Ideas About Notation, Ivan Sutherland, 13 September 2005
UCIES# 2005-is07: Literals for FLEET, Ivan Sutherland, 20 September 2005
UCIES# 2005-is08: Function or Addressing, Ivan Sutherland, 20 September 2005
UCIES# 2005-is10: More About Literals, Ivan Sutherland, 28 October 2005
UCIES# 2005-is11: Indirection for Memory Read and Write, Ivan Sutherland, 28 October 2005

PURPOSE

This memo offers four views of FLEET. We discussed these views in class on 31 October 2005. Think of the figures here as your notes from class. The memo also offers a brief explanation of each view.

AN ABSTRACT VIEW – Figure A

Figure A is an abstract representation of four SHIPs connected by a switch fabric. A blue rectangle with a black outline represents each SHIP. A curved line cuts each SHIP in half so that SHIP outputs can appear at the left right of the drawing and SHIP inputs can appear at the right of the drawing; a better drawing might be cylindrical. Green arrows represent the switch fabric with sources and destinations labeled.

The red structure, the “source decoder” transports instructions to the physical location of their data sources. A simpler view of FLEET might treat the red structure as part of the switch fabric, but I prefer to separate the red and green structures. Instructions in the red section are still in the instruction pool; they remain concurrent with other instructions that have not yet got data to move.

This abstract view avoids showing how instructions get into the instruction pool and how instructions control the switch fabric. This view also avoids details of the switch fabric, suggesting only that the switch fabric can carry data values from SHIP outputs to SHIP inputs.

This document is a product of a collaboration between Sun Microsystems and the University of California at Berkeley.. The ideas contained herein are freely available for any academic purpose.

A PROCESS FLOW VIEW – Figure B

This view suggests a little more detail. Although this figure suggests a tree structure for the source decoder and the switch fabric, any form will serve. Trapezoids represent multiplexers and decoders. The red parts of the figure handle instructions only, the blue parts handle data only, and the green parts handle both instructions and data.

When an instruction enters the pool it moves to a location near the outputs of its source SHIP. There it waits for data from the SHIP. When instruction and data get together, the instruction will direct the switch fabric to deliver the data to one or more destinations. Instructions wait for data at the tall multi-color rectangles before entering the switch fabric.

The three colors of the tall rectangles remind us that they serve multiple functions. The blue part receives data from the SHIP. The green part is the first stage of the switch fabric. The red part is an extension of the instruction pool that is distributed geographically throughout the space occupied by sources. Standing move instructions, clearly in the pool, remain here to send each new data value to its destination.

A SIMULATION VIEW – Figure C

Within the source decoding mechanism and the switch fabric communication is asynchronous. Thus each stage in these structures must acknowledge receipt of each data element sent to it. The simulator models these two-way handshake signals as separate signals. Data and appropriate parts of the instruction move to the right, and acknowledgement signals move to the left.

Figure C illustrates this two-way communication explicitly. This view assumes a “horn and funnel” implementation of the switch fabric, but any implementation form will do. The brighter arrows pointing toward the right carry data and instruction bits. The fainter arrows pointing towards the left carry acknowledge signals only.

There can be several instructions partly in process. For example, eight numbered circles represent eight instructions as they might back up in the switch fabric and the source decode mechanism. Such a back-up would occur only if a sequence of instructions delivers data from the same source to a blocked destination. When the destination finally takes the data, a series of acknowledge signals will pass through the system from right to left to advance each instruction one position.

The acknowledge signals shown in Figure C are quite different from the tokens that flow through the switch fabric. The tokens serve to provide end-to-end acknowledgement for communications set up by the programmer. The acknowledge signals in Figure C are at a finer grain than the programmer can see.

A GasP WIRE VIEW – Figure D

The GasP circuit family uses a single state wire not only to indicate validity but also to acknowledge receipt of data. In addition several data wires carry data from one stage to another. This view shows the data wires bold and with arrows and the state wires as narrower lines without arrows.

Notice that the same data wires from the decoders go to both successor stages. Each decoder captures the data sent to it and retransmits it to both successors. It does the decoding by telling only one of the successors that the data are valid. Of course one might save some power by sending separate wires to the two successors and driving only one such wire.

Notice that where the simulation view, Figure C, indicated only demand join combination of inputs at the source end of the switch fabric, Figure D specifies arbitration. Figure D attempts to represent real hardware rather than mere concepts.

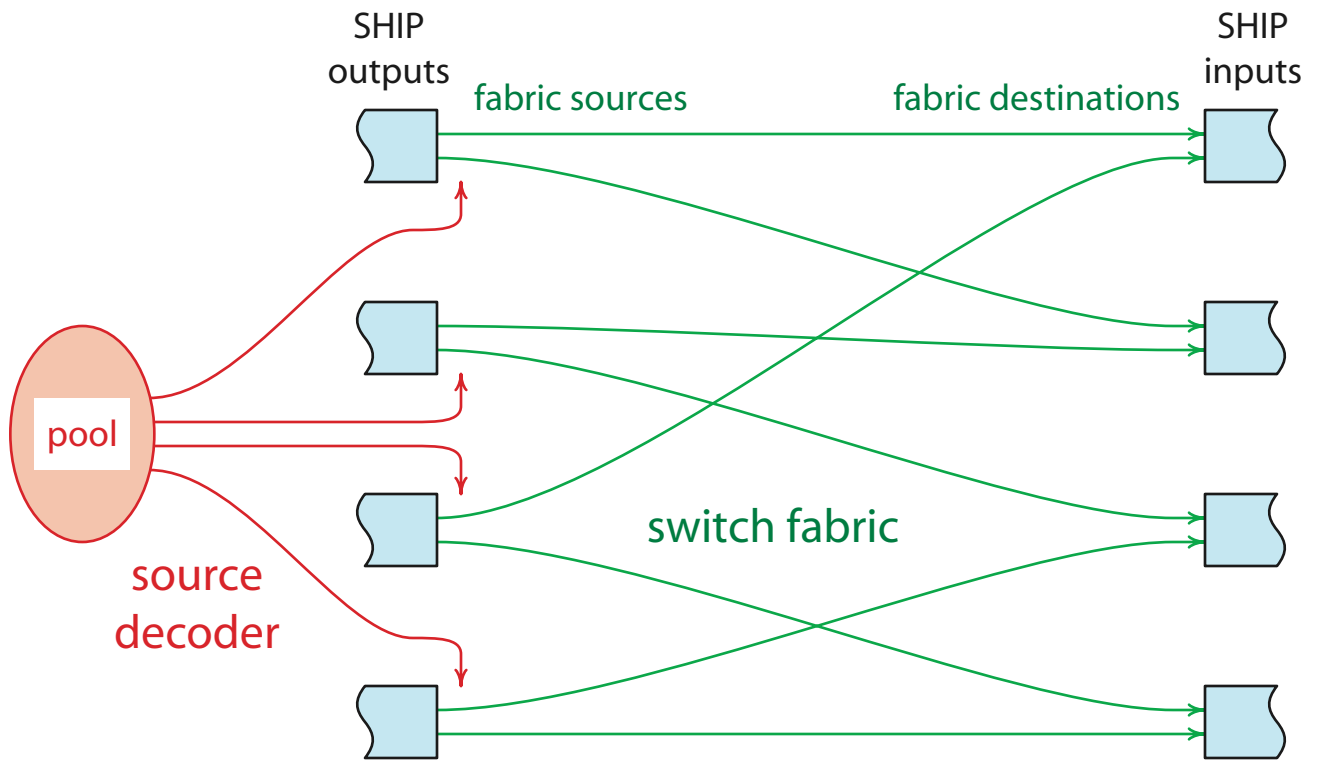


Figure A: An abstract view

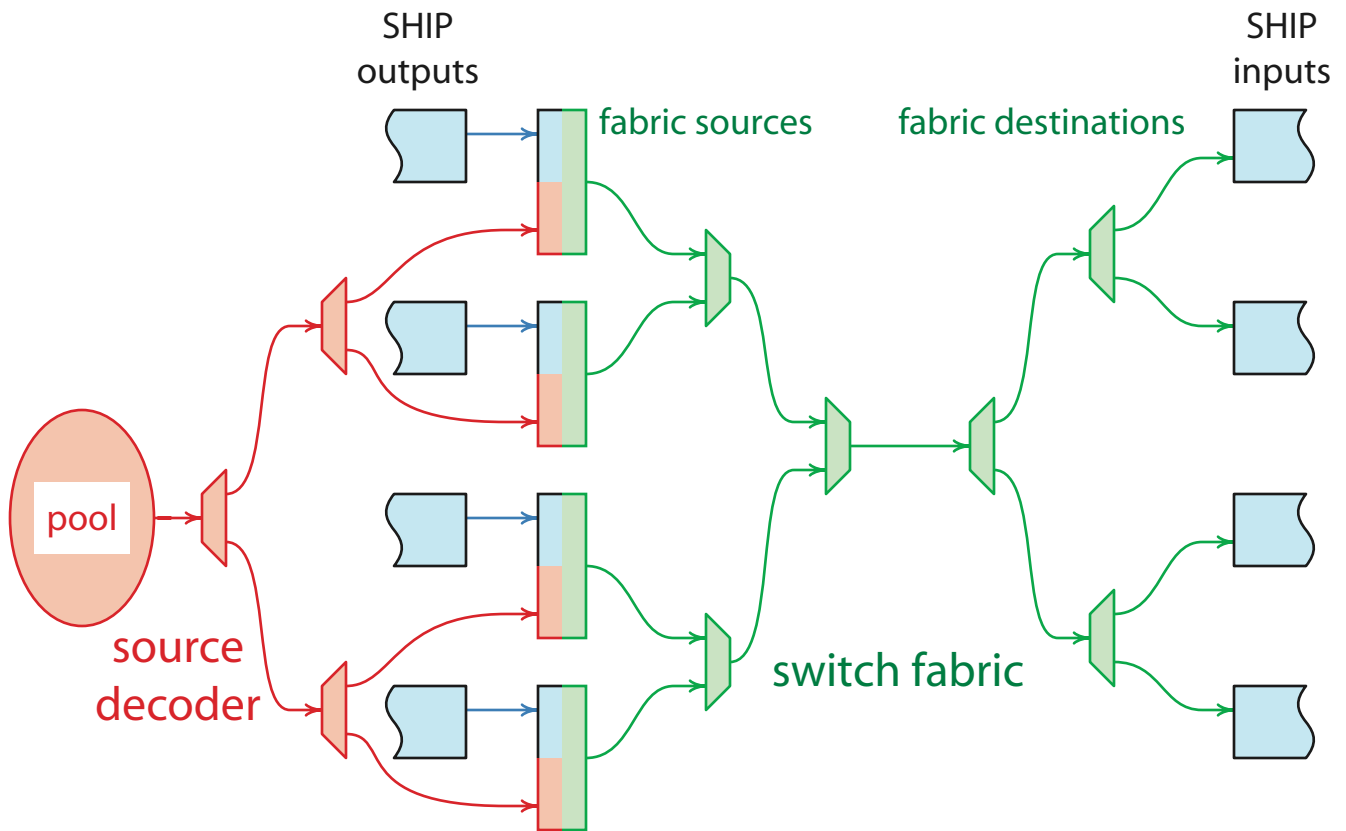


Figure B: A process flow view

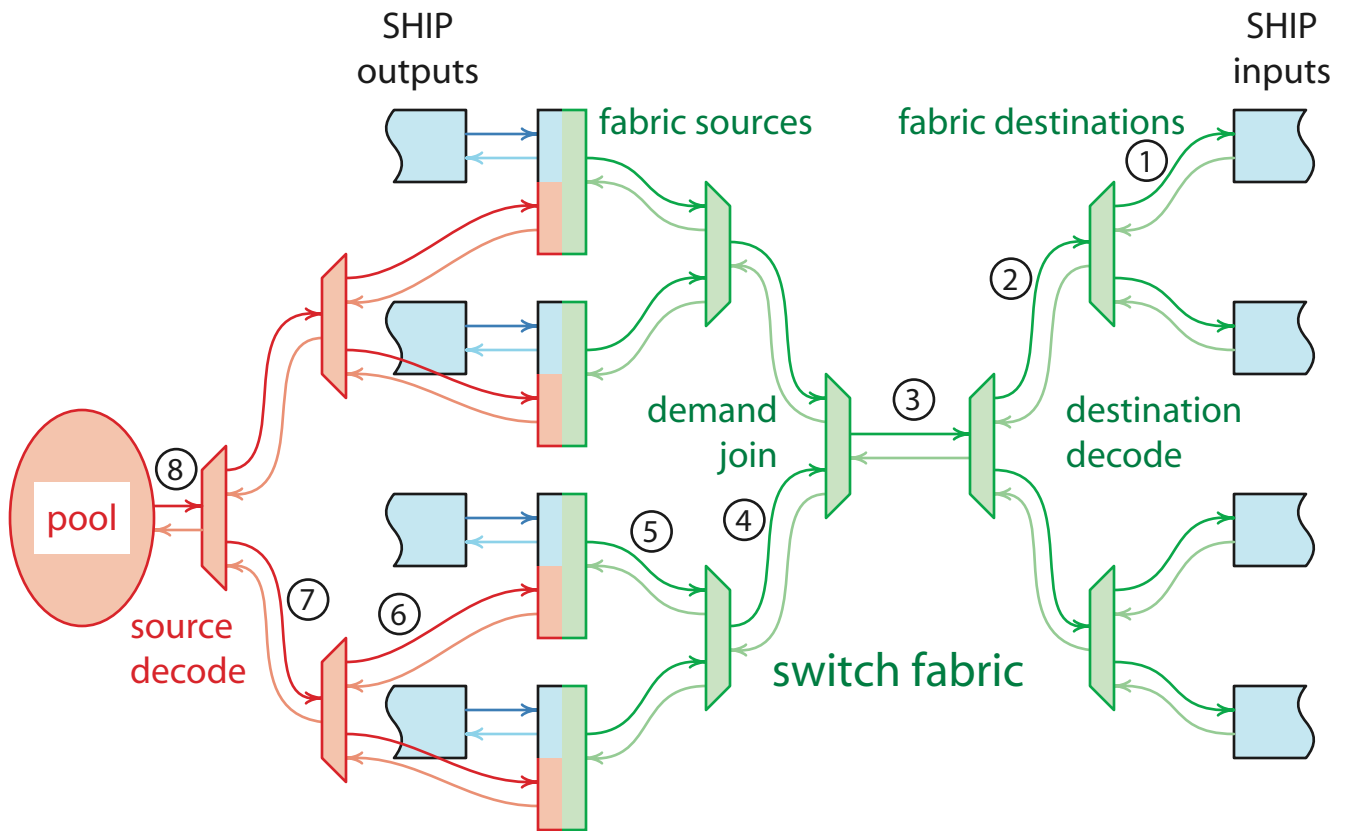


Figure C: A simulation view

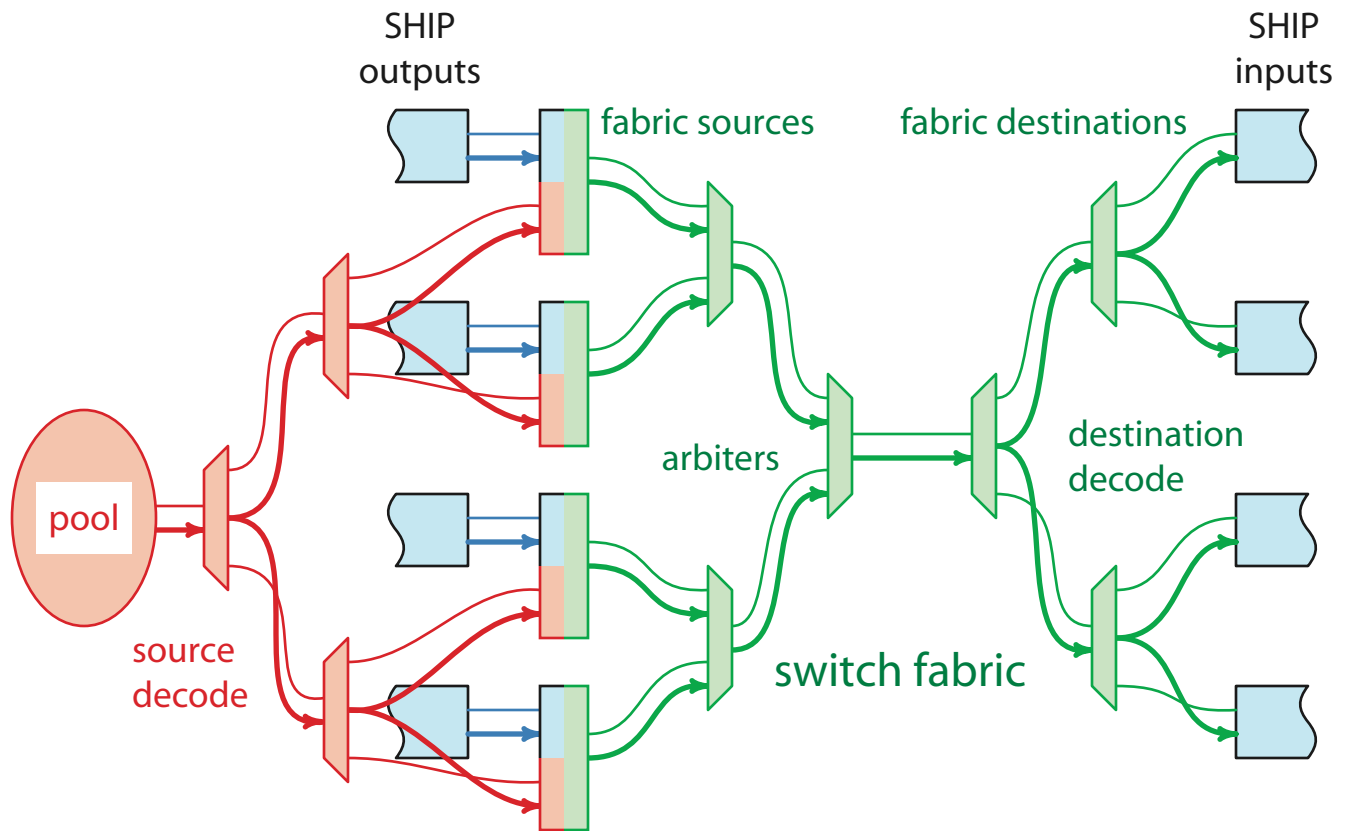


Figure D: A GasP wire view