California Institute of Technology
Department of Computer Science
Computer Architecture

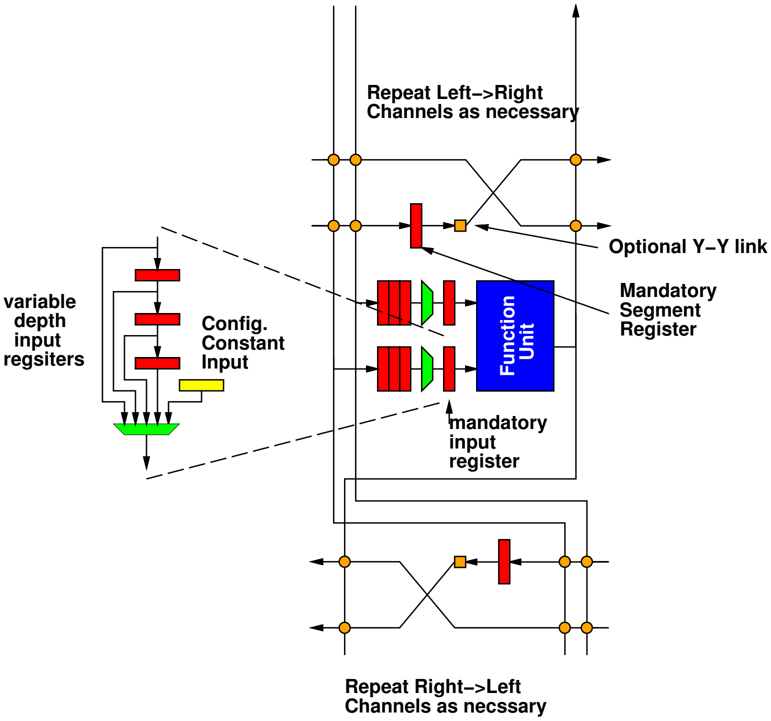
CS184a, Winter 2003

Assignment 7: Retiming

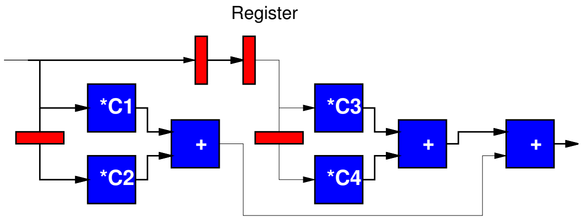
Wednesday, February 19

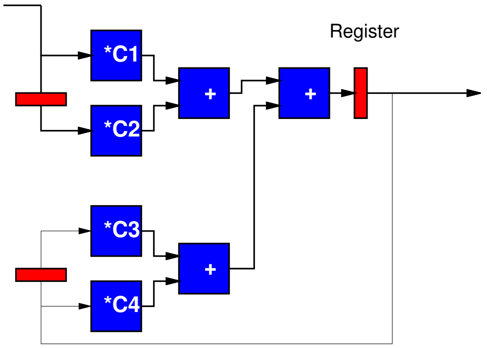
Due: Wednesday, February 26, 9:00AM

Shown below is a single slice of a one-dimensional datapath architecture. A 6 slice instance of the architecture is shown on the last page. Datapaths are multiple bits wide. The functional units are ALUs which include a multiplier. Every functional unit has a mandatory register on its input which is preceded by a variable delay input register. The network consists of length-2 lines and there is a mandatory register at the programmable switch between segments (the funniness with crossover at the bottom maintains the invariant that outputs are driven after the segment register and inputs are consumed before the segment register so that we maintain the length-2 property suggested above).



Consider the two computational graphs shown here:





For each graph:

1. Place the graph onto the 6 cell instance of the architecture. Show the routing. What is the minimum left→right and right→left channel width?

2. Pipeline and retime the placed graph so that it produces a new result on every cycle. Make the design C-slow if necessary. Attempt to minimize C. Report the C.

3. Identify the programming of each cell:
   - Compute function
   - Retiming depth of each input (or configured constant)
   - Input sources
   - Output destination
   - Y-Y link programming

4. Repeat for a time-multiplexed, 3-cell design. Each cell has two instructions which execute in round-robin fashion. Target producing a single result every second cycle.

   (a) give placement of functions

   (b) show routing (may share wires in time)

   (c) give minimum channel widths

   (d) give necessary maximum depths for input retiming registers

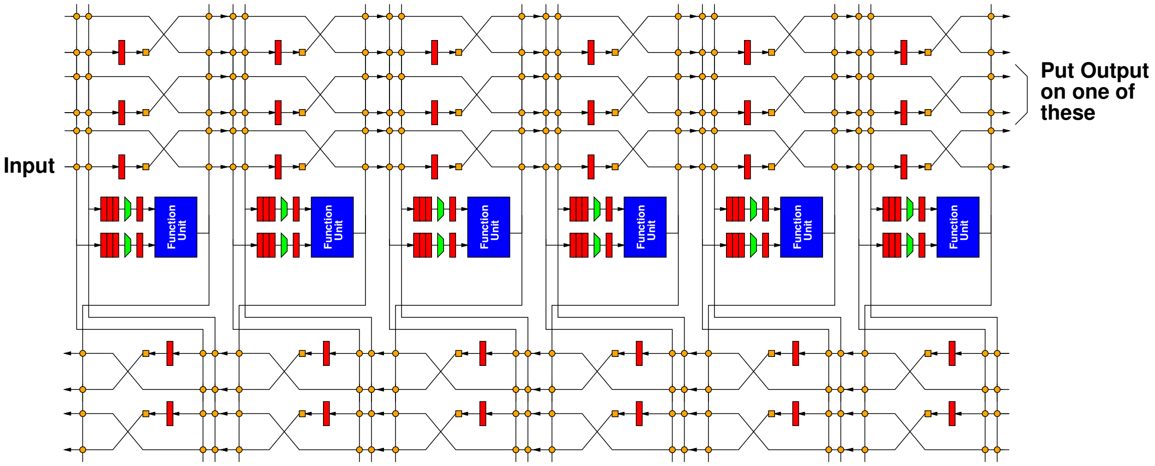
   (e) identify minimum C for each design

   (f) give programming for each cell (all timesteps)

5. Repeat for a time-multiplexed, 1-cell design. Each cell has six instructions which execute in round-robin fashion. Target producing a single result every sixth cycle. Give same information as above

I expect you might approach the retiming:

- Generate a modified graph which adds delay blocks to model the mandatory retiming in the placed design.
- Determine what it will take to retime the resulting graph so that it is fully pipelined (pipeline, C-slow?).
- Slide registers to match the existing, mandatory registers and place the balance on the inputs of functional units.
- Read off/summarize the registers per input from your retimed design.


Input
Put Output on one of these
Function Unit
Function Unit
Function Unit
Function Unit
Function Unit
Function Unit