VHDL-Based Design in Ptolemy

Michael C. Williamson
Edward A. Lee

UC Berkeley
Dept. of EECS

Ptolemy Miniconference
UC Berkeley
March 14, 1997

Extending High-Level Synthesis

- HLS: Demonstrated benefits for certain application domains, e.g. DSP
- Realized shortening of design time and improved quality of result
- Suggests opportunities for improving the design process at a still higher level
  - Make critical decisions while design is still mostly abstract
  - Not locked into a lower-level partitioning where HLS can only be applied to relatively small sub-partition groupings
  - Use coarse-grain information to advantage in resource-sharing decisions
- Synthesis of hardware from SDF graph information

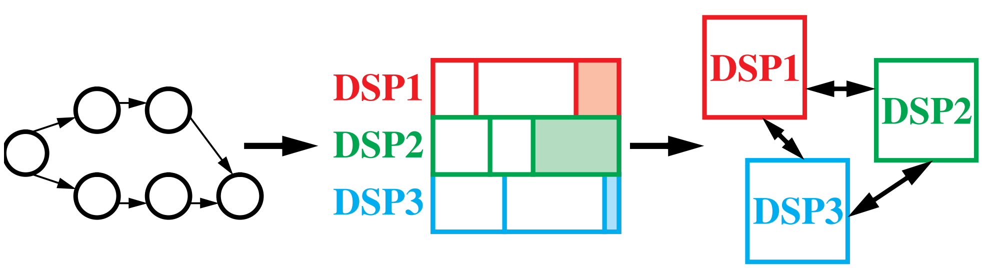
Problems and Opportunities

- Operating on the firing precedence directed acyclic graph (DAG), not the SDF dataflow graph
- Number of precedence graph nodes can grow exponentially in the number of dataflow graph nodes

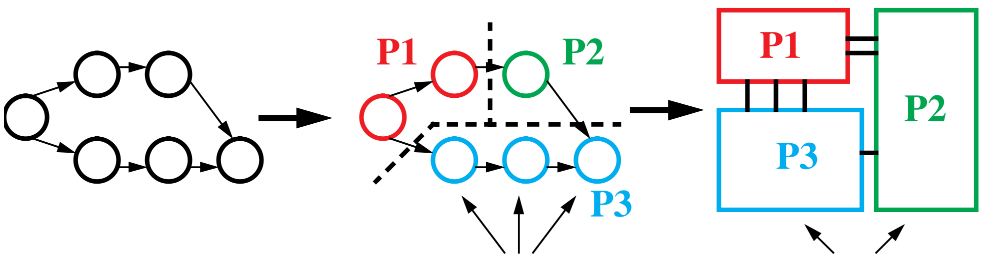
- Opportunity, through hierarchy, to simplify algorithms for HLS - coarser granularity and fewer graph nodes
- Keeping the structure grouped by functions preserves structural similarity information - crucial to guiding resource sharing decisions

- No universal algorithmic solution - we will still need and want to have the designer's interactive input

Synthesizing Hardware from SDF Graphs



firings multiprocessor schedule multiprocessor architecture



firings execution units hardware units

Problem Statement

- Scheduling N firing tasks onto hardware which is to be synthesized from the dataflow firing DAG

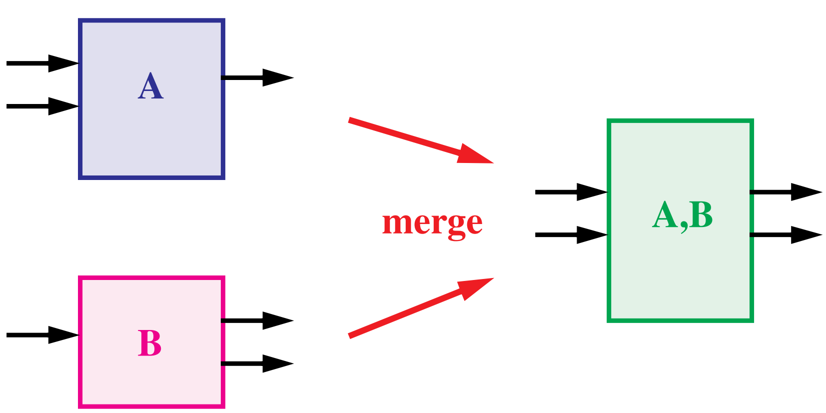
Problem: **Find the minimum-cost hardware to meet the deadline for the makespan (time to execute all firings in the DAG)**

- Firing tasks to perform F_j, $j = 1,\ldots,N$
- Individual firing dedicated HW costs C_j
- Individual dedicated HW execution times T_j
- Note that each C_j, T_j can vary significantly with particular implementations, so we base them on estimates from straightforward RTL synthesis costs, latencies

Merging Firings: Resource Sharing

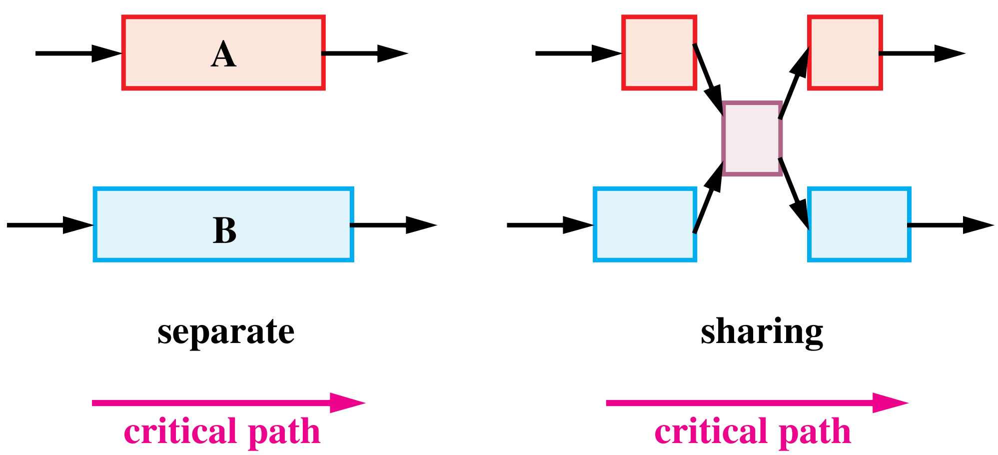
- Hardware Cost Bounds:

$$max(C_A, C_B) \le C_{A,B} \le C_A + C_B$$



Merging Firings: Execution Times

- **Pessimistic:** $T_{A|(A,B)} = T_A + T_B$
- **Optimistic:** $T_{A|(A,B)} \cong T_A$
- **We make the optimistic choice as a simplifying assumption**



Classifying Degrees of Resource Sharing

0. No resource sharing. The only precedences are the inherent data precedences.

- Greedy: Maximum hardware, minimum makespan
- Do I meet deadline? If not, then infeasible. If so, then reduce HW cost.

1. Only share among firings of the same actor

- Guaranteed to have the same parameters
- If state dependencies, makes sense to share, but if not, opportunity missed

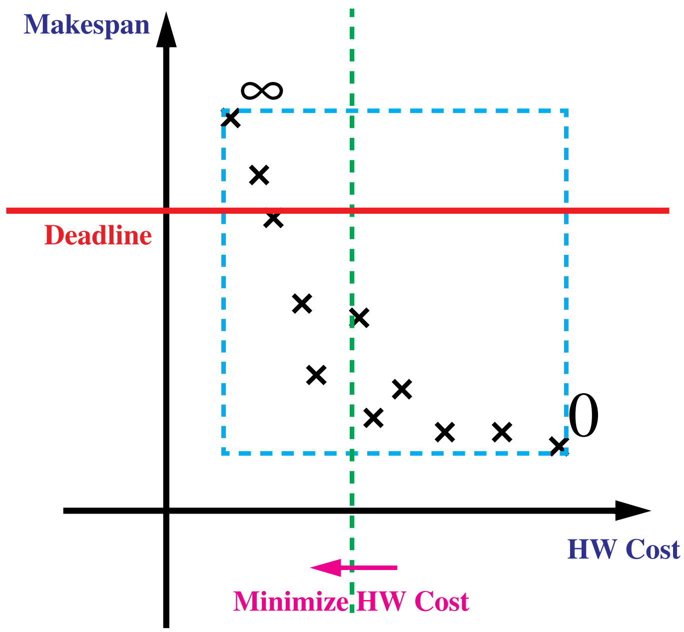
2. Share among firings of identical actors

3. Share among firings of similar actors (different params)

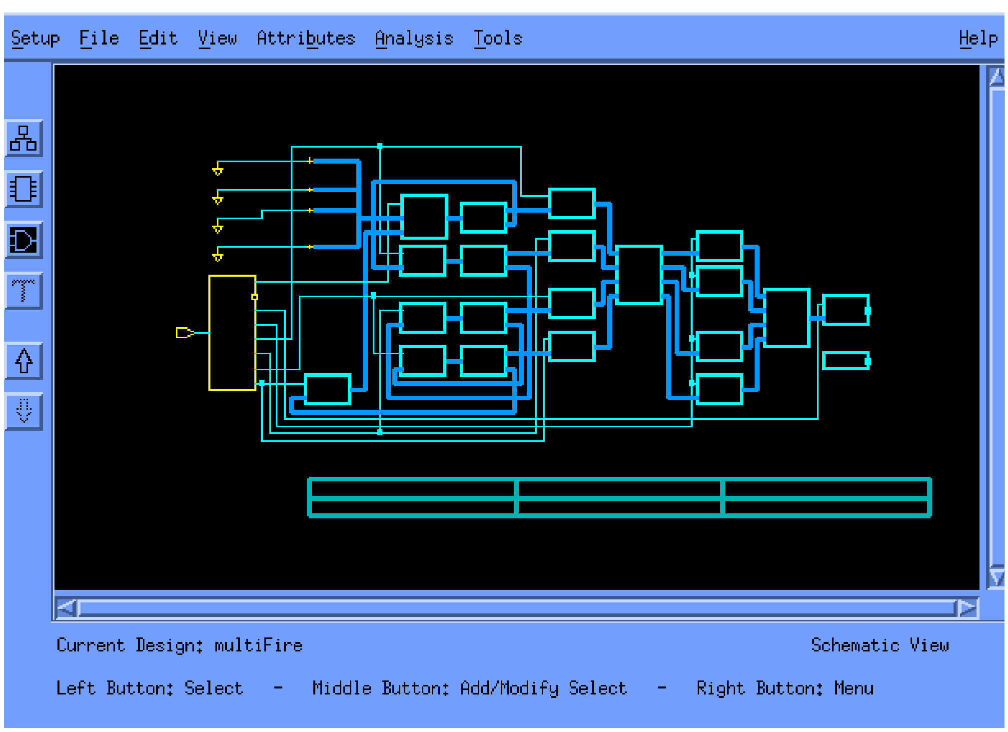
∞. Share a single sequential HW unit among all firings

- Do I meet deadline? If so, achieved cheapest HW - if you can synthesize
- If not, must de-cluster until deadline is met

The Design Space



VHDL SynthTarget



Two Heuristics

I. Cluster from maximum HW solution

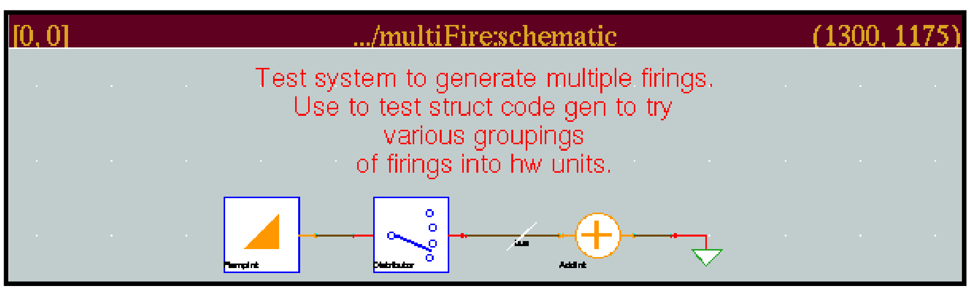
1. Start with maximum HW, minimum makespan solution - parallel
2. Cluster firings onto shared HW units until no more moves can be made without exceeding the deadline
3. Choose to cluster firings that can be executed at complementary times, giving priority to tasks with the highest degree of similarity

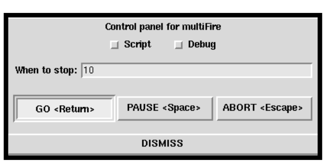
II. Decluster from minimum HW solution

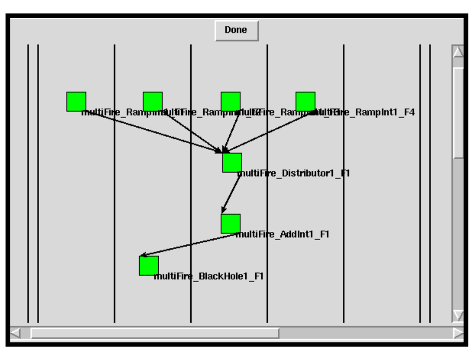
1. Start with minimum HW, maximum makespan solution - uniprocessor
2. Decluster groups of firings onto new HW units until the deadline is met
3. Choose to decluster longest paths of firings that aren't in the DAG critical path - largest opportunities to meet deadline while only increasing number of hardware units by one

Refinement to I. and II.: Move firings to procs with similar firings, as long as deadline is not violated

Automatic/Interactive Architecture Design



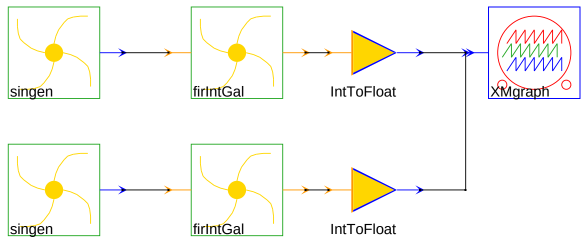




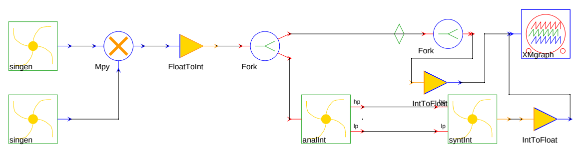
UNIVERSITY OF CALIFORNIA AT BERKELEY

Example Applications

- **Ex 1: Two FIR filters operating on parallel data streams**



- **Ex 2: 2-stage perfect reconstruction filterbank where stages have two FIRs each with up/downsample and addition**



Impact of Partitioning on Synthesis

- **Ex 1: parallel FIR**
  - **1 HW unit: 125 sec, 1,733 gates**
  - **2 HW unit: 138 sec, 1,788 gates (total)**
  - **Slightly worse synthesis time and gate area, but no significant cost to automatically generate the interprocessor communication correctly**
- **Ex 2: perfect reconstruction filterbank**
  - **1 HW unit: 520 sec, 6,249 gates**
  - **2 HW unit: 452 sec, 5,834 gates (total)**
  - **Significant savings in synthesis time and in gate area, and also performed the automatic synthesis of correct interprocessor communication in a complex multirate system**

- **With interactivity plus automation, can explore many more candidates in the design space**

Conclusions

- Showed a method for generating parallel hardware from dataflow graphs, with bounds on the design space
- Ptolemy VHDL targets provide automatic communication and interconnect without any manual design time penalty
- Demonstrated savings in overall synthesis time and gate count, allowing exploration of more design choices

Contact Information:

cameron@eecs.Berkeley.edu

http://ptolemy.eecs.berkeley.edu/~cameron