Hardware/Software Partitioning of Synchronous Dataflow Graphs in ACS domain of Ptolemy

White paper

EE 382 C – 9 – Embedded Software Systems

Gayathri Manikutty

Heather Hanson

I. Introduction and Context of Work

Embedded processors for real-time systems have to meet stringent requirements in terms of performance and power dissipation while keeping the product cost and development cycle low. This can only be brought about by exploiting the synergism of hardware and software through their concurrent design. The difficulty is to choose from the multiplicity of mapping and implementation alternatives available such that the overall design is optimized.

This project will develop a partitioning tool within the Adaptive Computing Systems (ACS) domain of Ptolemy. The ACS domain facilitates the design of adaptive computing systems that can be modeled using synchronous data flow graphs. Applications include signal processing and communication systems, such as modems. These will be implemented in a mixture of reconfigurable hardware, such as FPGA's, and software. The ACS domain enables designers to choose which technology will implement each sub-system.

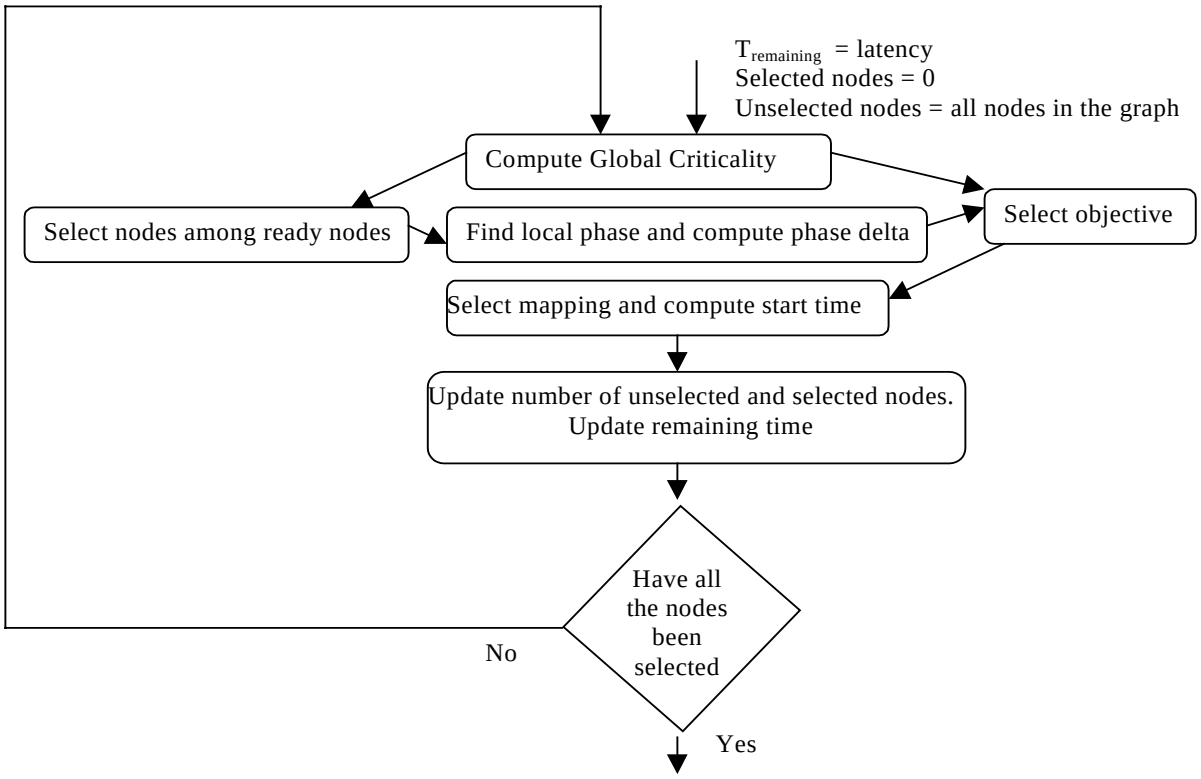
II. Design objective and summary of published work

Our contribution is incorporating a mapping feature within a partitioning tool; the tool will assist designers by performing near-optimal design partitioning into hardware and software modules. We base the mapping on the Global Criticality/Local Phase (GCLP) algorithm [2]. The GCLP algorithm was prototyped within the UC Berkeley Ptolemy software environment to provide design assistance in partitioning but it was never released. To begin with, we provide a brief overview of the partitioning problem and the GCLP algorithm.

The SDF domain in Ptolemy translates the task-level description of an application into a Directed Acyclic Graph (DAG) where each node represents a computation and the arcs represent the data and control precedence between the nodes. A hardware/software partitioning problem is to decide the mapping for each node of the DAG, either into hardware or into software and arrive at a schedule.

This partitioning aims at optimizing the overall design (in terms of area and speed of implementation).

This partitioning problem is computationally intensive. For solving this constrained optimization problem, exact solutions through Integer Linear Programming is intractable. The GCLP algorithm is a heuristic for solving the binary partitioning problem in linear time.



The GCLP Algorithm by Dr. Kalavade

The list-scheduling algorithm forms the underlying scheduling framework for the GCLP algorithm. Unlike list scheduling, which either optimizes for the finish time of the node or for the area of the node, the GCLP algorithm adaptively selects an appropriate mapping at each step based on global criticality and local phase.

III. Plans for design implementation

We will implement the GCLP algorithm developed by Dr. Aswaree Kalavade into the ACS domain within Ptolemy. We are currently working on installing a local version of Ptolemy, and then will

proceed by integrating the GCLP code into a star within the ACS domain and recompiling Ptolemy. With the new features installed, we will use the GCLP algorithm to partition a simple application (as yet unspecified) and compare the results with an optimal partitioning.

IV. References

1. Kalavade, E.A. Lee, “The Extended Partitioning Problem: Hardware/Software Mapping and Implementation-Bin Selection”, Proc. of Sixth Intl/ Workshop on Rapid Prototyping, June, 1995
2. Kalavade, E. A. Lee, “A Global Criticality/Local Phase driven Algorithm for the Constrained Hardware/Software Partitioning Problem”, Proc. of Codes/CASHE’94, Third Intl. Workshop on Hardware/Software Codesign, Sept. 22-24, 1994, pp. 42-48
3. Kalavade, E.A. Lee, “A Hardware/Software Codesign Methodology for DSP applications”, IEEE Design and Test of Computers, Sept. 1993, pp 16-28.
4. Kalavade and E.A. Lee, “The Extended Partitioning Problem: Hardware/Software Mapping, Scheduling, and Implementation-bin Selection.”, Journal of Design Automation for Embedded systems, Vol. 2,no. 2,March 1997,pp. 126 -163
5. J. Buck, S. Ha, E. A. Lee, and D. G. Messerschmitt, "Ptolemy: A Framework for Simulating and Prototyping Heterogeneous Systems," Int. J. Computer Simulation, Vol. 4, April 1994, pp. 155-182.
6. E.A. Lee, et al., University of California at Berkeley, The Almagest, Volumes 1-3, Regents of the University of California, 1995.