lem cannot, in general, be posed as a single well-defined optimization problem. Typically, the designer needs to explore the possible options, tools, and architectures. We believe that managing the design process plays an equally important role in system-level design, as do the tools used for different aspects of the design. To this end, we present a framework that supports design methodology management.

We embody all the above concepts in a tool called the Design Assistant. The Design Assistant contains tools for partitioning, synthesis, and simulation and operates in the design methodology management framework.

Edward A. Lee, Thesis Committee Chairman

tioning problem involves the joint determination of the mapping (hardware or software), schedule, as well as the implementation option for each node, so that the overall area allocated to nodes in hardware in minimum. This problem is considerably harder (and richer) than the traditional binary partitioning that determines just the best mapping and schedule.

Both extended and binary partitioning problems are constrained optimization problems and are shown to be NP-hard. We first present an efficient heuristic, called GCLP, to solve the binary partitioning problem. The heuristic reduces the greediness associated with serial traversal-based algorithms by formulating a global criticality (*GC*) measure. The *GC* measure also permits an adaptive selection of the objective (optimizing either area or time). We then present an efficient heuristic for extended partitioning, called MIBS, that alternately uses GCLP and an implementation-bin selection procedure. The implementation-bin selection procedure chooses the bin that maximizes the area-reduction gradient. Solutions generated by both heuristics are shown to be reasonably close to optimal. Extended partitioning generates considerably smaller overall hardware area as compared to binary partitioning.

Our approach to cosynthesis is to generate synthesizeable descriptions of the hardware and software components and to use pre-existing synthesis tools such as Ptolemy and Hyper to generate the actual software and hardware implementations. Techniques for generating these synthesizeable descriptions, starting with a partitioned graph, are discussed. Cosimulation involves simulating the hardware, the processor, and the software that the processor executes, within a unified framework. The requirements of a cosimulation framework are discussed and the use of Ptolemy for hardware-software cosimulation is illustrated.

Due to the large system-level design space, the system-level design prob-

Abstract

SYSTEM-LEVEL CODESIGN OF MIXED HARDWARE-SOFTWARE SYSTEMS

by

Asawaree Prabhakar Kalavade

Doctor of Philosophy in Electrical Engineering

University of California, Berkeley

Professor Edward A. Lee, Chair

This thesis provides a systematic approach to the system-level design of embedded signal processing applications. Such applications tend to have mixed hardware-software components and are often subject to severe cost, performance, and design-time constraints. Our approach is to codesign these systems. The codesign approach allows the hardware and software designs to be tightly coupled throughout the design process. We focus on the four key problems of partitioning, cosynthesis, cosimulation, and design methodology management. Applications are assumed to be specified using synchronous dataflow semantics.

A key contribution of the thesis is to formulate the *extended partitioning problem*. In system-level design, applications are represented as task graphs where tasks (called nodes) have moderate to large granularity and each node has several implementation options differing in area and execution time. The extended parti-