AD-A223 754

(2)

Semiannual Report: January 1990 - June 1990

Asynchronous Design for Parallel Processing Architectures

Contract No.: N00014-89-J-3036
Principal Investigator: Teresa H.-Y. Meng
CIS 132, Stanford University, Stanford, CA. 94305
Phone Number: (415) 725-3636
E-Mail Address: meng@tilden.stanford.edu

Goal: The objective of this research is to provide an interconnect design synthesis methodology which facilitates a modular design approach without compromising the global performance. The main tasks of this effort will be the development of the theory for optimal interconnect synthesis from a high-level specification, with emphasis on testability and fault-tolerance asynchronous interface among concurrent computing hardware objects, and the application of this design methodology to physical implementations of parallel processing systems.

Progress:

1. Self-Timed Circuits Synthesis with Timing Constraints

   Timing constraints can be used to reduce dependencies of circuit behavior on signal transitions. The circuits synthesized using timing constraints are not speed-independent (by definition), but hazard-free if the timing constraints hold. Since timing constraints usually make some states unreachable, this information can be used to derive a more efficient implementation by introducing more "don't cares" into the synthesis procedure.

   We have described some results in incorporate timing constraints into circuit synthesis in a previous report (Dec. 1989), and a more elaborate algorithm to reduce transition dependencies was recently derived [1]. Since this problem can be shown to be at least NP-hard, we proposed the following heuristic algorithm whose complexity is polynomial in the number of signals involved. Synthesis experiments showed that this algorithm will produce optimal results with a high probability.

   We first represent the behavior of an asynchronous circuit by a signal transition graph which describes the dependencies among signal transitions. By incorporating timing constraints into the graph, we hope to delete some arcs (dependencies) in the graph such that more don't-care states can be generated and fed into the synthesis procedure. The timing algorithm is based on two parts, an aggregation algorithm and a graph-search algorithm. To delete possible transition dependencies, both algorithms are used in tandem; alternating the two algorithms one after the other better reduces the dependency graph than either alone.

   The aggregation algorithm groups nodes that are directly related to each other into a "super-node", which has input and output arcs that maintain timing equivalence with its internal components. Reducing the graph into super-nodes simplifies the graph problem with no expense in generality. This aggregation examines the reduction of single parent nodes, though other reductions are possible.

   The graph-search algorithm analyzes the minimum time and maximum time for token flow between signal transitions. We consider a single arc at a time. Based on the set of values obtained by graph search, we decide whether an edge is useless and analyze if we can remove it; that is, when the sibling arcs always control whether the transition fires independent of the timing of the arc currently being considered. If a particular arc in the graph can be deleted, this information is fed into the synthesis procedure and an implementation with less hardware complexity can be generated. As will be discussed in the next subsection, this techniques has been combined with conditionals to synthesized various asynchronous circuits correctly.

2. Conditionals in Asynchronous Circuits

The synthesis of asynchronous control circuits with conditionals has not been given proper attention in the literature. Examples of control circuits with conditionals are multiplexers, demultiplexers, or any circuit whose behavior is changed as a function of time. The behavior of a circuit with conditionals is deterministic, as the circuit behavior can be predicted exactly when the the values of conditionals are known. If defined reasonably, conditionals do not require arbitration: the behavior of the implementation depends only on the value of the conditionals, not on their timing.

The criterion for our synthesis procedure is to allow as much concurrency as possible in the circuit operation without generating circuit hazards or adding arbitrary state variables. Our treatment of conditionals allows substantial sharing among the components handling the different cases of the condition, and allows concurrency between the end of a cycle handling one case and the beginning of the cycle handling the next.

The main problem with synthesizing conditionals is to prevent hazards. Since the circuit is basing decisions on the value of the conditional input, a change at an awkward time can cause incorrect behavior. A necessary condition for a hazard-free implementation of a circuit with a conditional is that the conditional input have some recognizable window of stability. If the window of stability is sufficiently large, the conditional input can be used as a signal to control gates and elements inside the circuit. If the window is small, an internal conditional signal (a stored version of the external conditional input) will have to be generated to guarantee a hazard-free implementation.

Based on our study of asynchronous circuit behavior, we have provided a procedure for synthesizing asynchronous control circuits with conditionals [2] and an algorithm for incorporating timing assumptions into the synthesis procedure [1]. Combining conditionals with timing constraints, we have demonstrated on a variety of examples (e.g. multiplexers with external timing constraints and an MC68000 VME bus interface) that real timing delays can be used to optimize an implementation during synthesis, and that conditionals can be handled effectively by synthesizing an internal conditional signal. These implementations have been checked against the Petri net specification using an automatic speed-independent circuit verifier and confirmed to be hazard-free at the gate level (each logic gate has an arbitrary delay).

3. Testability of Asynchronous Control Circuits

Testability has become a major design consideration in IC industry and the question that whether asynchronous circuits will simplify the testing tasks is to be answered. Asynchronous combinational circuits have been shown to be fully testable with single-stuck-at-faults (SSAF) if precharged circuits were used for implementation, but the testability of asynchronous control circuits (sequential circuits with environmental constraints) has not been addressed. Level-sensitive scan-path design can be used to aid testing for asynchronous sequential circuits. However, we are mostly interested in synthesis for testability as scan-path testing usually requires too long a testing delay.

We began our research by constructing a general model for asynchronous control circuits, decomposing them into a next-state block, a C-element array, and an output-logic block. A comprehensive study of the testability issues in terms of hazard-free asynchronous control circuits has been conducted and the sufficient conditions under which a given circuit can be ensured to be fully testable with the SSAF assumption have been proved. Our conditions do not require direct access to the C-element array and requires no added circuitry [3]. This result is not yet complete in the sense that we need necessary conditions to guide the synthesis procedure so that fully testable asynchronous control circuits can be synthesized directly from a high-level description.

Future work in the next 2 quarters:

To improve the heuristics in the timing algorithm, future research will center on improving the aggregation techniques: best-fit and probabilistic aggregations may be used to break down very large asynchronous circuits into small components. Also, better search techniques can use the cycles in a graph to analyze possible flows and the exact initial condition can guide and improve the accuracy of the graph-search algorithms. Finally, analysis of transient states of a system-- states that are never returned to because of timing-- is another open problem.

On the logic synthesis side, it is not obvious how we can address the property of hazard-freedom in a circuit synthesized with timing constraints. We can only state at the moment that if the specified timing constraints are met, the synthesized control circuits can be said to be hazard-free at the *functional* level (where a block of logic is modeled by a single Boolean function rather than implemented with distinct logic gates). Whether an algorithm can be used to construct a hazard-free gate-level implementation of an asynchronous control circuit with timing constraints remains to be seen in future research.

With the understanding of how to derive a gate-level hazard-free implementation from a circuit's functional description, we hope to incorporate the conditions for full testability and ultimately provide the theories and algorithms for synthesizing hazard-free fully testable asynchronous control circuits with a minimum number of gates.

References

[1]. Andy C. Hung and Teresa H.-Y. Meng, "Asynchronous Self-Timed Circuits Synthesis with Timing Constraints", *Proc. of IEEE ISCAS 90*, May 1990.

[2]. Teresa H.-Y. Meng and Steve M. Nowick, "Conditionals in Asynchronous Circuit Synthesis", to be submitted to *IEEE Trans. on CAD*, September 1990.

[3]. Peter Beerel and Teresa H.-Y. Meng, "Sufficient Conditions for Testability of Asynchronous Control Circuits", to be submitted to *IEEE Trans. on CAD*, August 1990.