DOWNLOAD PDF PROCEEDINGS, EUROMICRO SYMPOSIUM ON DIGITAL SYSTEM DESIGN

Chapter 1 : Journals and Proceedings - UniversitÃ¤t Siegen

Systems for coding style analysis - so called HDL code checkers - can accomplish an important contribution for the IP entrance check, that means selection, compliance test and quality estimation of reusable components for system design.

Comparison of software implementations and reconfigurable hardware implementations will need fast and easy-to-use estimation techniques. In this paper, we present an estimation approach for the resource utilization of the embedded FPGA co-processor. Our approach is based on the principles of high-level synthesis, such as force-directed scheduling, resource allocation, operation assignment and interconnection binding. Introduction The silicon capacity will breakthrough the billion- transistor mark in the near future according to the latest technology forecasts [1]. The available capacity will introduce the possibility to implement very complex systems in a single chip. These systems may consist of general-purpose processors and software, fixed hardware accelerators, and embedded reconfigurable logic. The essential issue with these designs based on System-on- Chip SoC technology is the trade-off between cost and flexibility. Flexibility is typically achieved using software. The general-purpose processors execute a set of instructions to perform a certain function. By re-writing the software codes, the function can be easily changed without any hardware modification. The cost of this flexibility is the degradation of the performance. By taking advantage of the customer design, fixed hardware accelerators can run functions much faster but with lower power consumption. Reconfigurable logic can achieve much higher performance than software, while maintaining a higher level of flexibility than hardware [2]. Including reconfigurable logic in System-on-Chip SoC gives designers more options toward an optimized solution. There are several ways to couple the reconfigurable logic with processors. Firstly, it can be put into the processors as a reconfigurable unit, which implements custom instructions. Secondly, it can work as co- processors: Thirdly, it can behave as additional processors and work more independently. Fourth alternative is the most loosely coupled connection. Reconfigurable logic performs as an external stand-alone processing unit and rarely exchanges information with the host processors [2]. In addition to coupling, the granularity of the computation elements, the form of routing resources and the reconfiguration methods are important issues in design as well. When designing such reconfigurable systems, designers need to have the knowledge of how to estimate the cost of a potential implementation, how to do the partitioning decisions, how to define the reconfiguration requirements, etc. During the architecture design, designers need to have estimated information that indicates how to size the reconfigurable logic. The ultimate goal of our research is to study an estimation-based system-level partitioning method for the reconfigurable systems, where reconfigurable logic is used as a co-processing functional unit. Estimating the performance, e. The CDFG and abstract processor models are used to estimate the software performance and the mappability of algorithm - architecture pairs [3]. A high-level synthesis- based hardware estimator gives the estimates in terms of the speed-up and the hardware resource utilization. This paper focuses on the high-level synthesis-based resource utilization estimation, which consists of functional unit estimation and interconnection estimation. The functional unit estimation is performed using Force- Directed Scheduling based technique, in which the total force is defined as the number of Look Up Tables LUT required in a temporary schedule [5]. We assume a multiplexer-based interconnection. After using weighted- bipartite matching algorithm to solve the register allocation and function allocation problems, we collect the estimated HW resources in terms of the number of LUTs required for the multiplexers dedicated to the interconnection. The structure of the paper is as following. In section 2, known high-level synthesis algorithms and previous work for estimating lower bounds on area cost and control steps are reviewed. Our proposed solution to the system-level partitioning and detailed illustration to the HW estimation are given in section 3. Section 4 gives the experimental results when comparing the HW estimation with synthesis results. Section 5 gives the conclusion. Previous Work High-level synthesis has been studied extensively since s. The work has been formally divided into scheduling

and allocation. In [4], an integer linear programming ILP method is used to find an optimized schedule using a branch-and-bound search algorithm. The well-known FDS [5] intends to minimize the functional units required with a given time constraint. A technique using simulated annealing algorithm to schedule a dataflow graph is presented in [6]. The work in allocation can be formally divided into register allocation, operation assignment and interconnection binding. The allocation work is mostly resolved using graph theory algorithms. In [7], allocation is mapped to the clique-partitioning problem, which is a classic NP-complete problem. A heuristic procedure is usually used in implementation. In [10], the scheduling, allocation and binding are integrated into a single algorithm. Several lower bounds estimation methods have been studied. A mathematical model for predicting the area- delay curve is given in [11]. In [12], interconnection estimation is taken into account but with the assumption that all data is transferred via bus. In [13], a multiplexer- based interconnection is assumed, and the additional area in terms of the multiplexers is estimated in a similar way to the interconnection-binding technique used in high- level synthesis. In our approach, we use the principles of high-level synthesis techniques only for the estimation of the HW resource utilization and the speed-up for design space exploration purposes. A scheduling-based approach is used to estimate the resource utilization for functional units and an allocation-based approach is used to estimate the resource utilization for interconnection. The objective is to find out the feasibility of the partition with an adequate confidence level. Design flow for estimation based partitioning 3. Estimation-based partitioning approach The proposed estimation-based partitioning approach is illustrated in Figure 1. The starting point is the functional description given as a C-language algorithm, because C-language is extensively used in various systems and most researchers are familiar with it. The designer decides the granularity of partitioning by decomposing the algorithm down to function blocks. A single function block may then be assigned to either SW, reconfigurable logic or even implemented as a fixed functional unit. Combining with profiling information that is collected by running the executable algorithms with certain application data, the C codes are transformed into CDFG using a SUIF compiler [14]. In addition to the mappability results, the SW estimator produces the estimated execution time. The HW estimator produces the estimated HW execution time and the estimated HW resource utilization for each individual function block. The estimates can be used to narrow the design space, which leads designers to a satisfactory solution with few iterations. In case no feasible solution is found, designers may start over from the beginning by reducing the complexity or the functionality of the algorithm. Architecture template Figure 2 gives a simplified version of the target architecture, in which we have an embedded Field Programmable Logic Array eFPGA as a co-processing unit. We assume the function block is the basic element to be implemented on the eFPGA co-processor. The result is a co-processor containing a big class of algorithms, which might be inefficient in terms of resource utilization. Using a massively parallel architecture with generalized, reused data-paths common to function blocks will result in better overall performance. However, this type of architecture is left for future research. The processor in our architecture is assumed to be a simple RISC processor. The basic parameters, such as instruction set, the number of pipeline stage and the number of parallel units of the processor core are taken into account in SW estimation. Architecture template In the current approach the data traffic and bus capacity between computation components and storage components are not taken into account 3. Application analysis The C codes representing the application are executed with real input data, and the profiling tool, gcov, is used to gather the information of the execution count and the branch probability. The application is decomposed into several function blocks, which are the basic partitioning elements. Each function block is transformed into a CDFG and will be studied individually. The CDFG with the embedded profiling information captures the data dependence, the control flow of the application, and the intrinsic characteristic of the application regarding the input data. The mappability estimation is carried out by analyzing the correlation of function blocks and the characteristics of processor core architecture. The software execution time is estimated using a profile-directed operation-counting based technique as follows. The execution count of CN is CNw. The execution time for a single pipeline stage is Ex. The proportion of the total number of branch operations to the total number of

operations in function block n is nB. The effect of branch penalty is modeled with coefficient. Speed- up is the ratio of the estimated SW execution time to the estimated HW execution time. The timing and LUT requirement of each basic operation is created using the commercial synthesis tool, Synplify [15] and coded into a text file to be used as input to the HW estimator. Naturally it introduces some amount of error to the final results. Our HW estimation approach is based on high-level synthesis, in which scheduling is used to determine the number of control steps and the number of LUTs required for operations; allocation is used to determine the number of LUTs required for multiplexer-based interconnection. We assume that the implementation uses only multiplexers to connect the functional units and registers. The scheduling is based on Force-Directed Scheduling FDS technique, which intends to reduce the number of required functional units by balancing the concurrency of operations assigned to them without lengthening the total execution time [5]. Chained operators and operators crossing several control steps are taken into account in the scheduling process. The valid range is a time frame in which an operation may be legally scheduled without increasing the total control steps. Operations are equally distributed within its time frame, and scheduled one at a time. As in FDS, operations, whose valid ranges are larger than one control step, are uniformly distributed within its valid range, but one of them is temporarily scheduled in a control step within its valid range. The distribution of its predecessors and successors are correspondingly modified, because fixing this node in a certain control step affects the whole distribution graph. The temporary schedule with the largest reduction of total force is selected. Instead of calculating self-force, parent force and child force separately for each temporary schedule, we use the following cost function to calculate the total force in terms of the HW resources required for this control node. We define jiRsc , as the number of HW resource of type i required for operation of type j , iCst as the number of available resource of type i in a given type of eFPGA, kjDG , as the distribution value of operation of type j in control step k , and CS as the set of control steps. Allocation is formally divided into three parts: Register allocation and operation assignment are closely related to each other. There is no standardized solution to which one should be carried out first. By taking the other into consideration while performing one, we can obtain equally satisfactory design qualities [9].

Chapter 2 : dblp: Euromicro Symposium on Digital Systems Design

Proceedings Euromicro Symposium on Digital Systems Design September 4 - 6, Warsaw, Poland Sponsored by Euromicro IEEE @ COMPUTER SOCIETY Los Ala~tos, California.

Chapter 3 : CiteSeerX â€” A Formal Verification Methodology for IP-based Designs

Euromicro Symposium on Digital System Design Abstract: The following topics are dealt with: processor and memory architectures; synthesis (HL, LS, PS); special architectures; system-on-a-chip; system-on-a-chip and validation/verification; applications of (embedded) digital systems; and specification and modeling.

Chapter 4 : Euromicro homepage - Events - Past

Abstract: The adoption of hardware description languages as a design specification formalism, in the electronic design automation industry, has reached acceptance during the last years.

Chapter 5 : dblp: Euromicro Symposium on Digital Systems Design

Proceedings Euromicro Symposium on Digital System Design Architectures, Methods and Tools September 4 - 6, Dortmund, Germany Edited by: Martyn Edwards.

DOWNLOAD PDF PROCEEDINGS, EUROMICRO SYMPOSIUM ON DIGITAL SYSTEM DESIGN

Chapter 6 : DSD : 18th Euromicro Conference on Digital Systems Design

Proceedings, Euromicro Symposium on Digital System Design: Belek-Antalya, Turkey, September 1st to 6th, Paperback Be the first to review this item See all formats and editions Hide other formats and editions.