Proceedings of the First International Conference on Information Technology and Knowledge Management pp. 159–163
DOI: 10.15439/2018KM41
ISSN 2300-5963 ACSIS, Vol. 14

Application of ASIP in Embedded Design with Optimized Clock Management

Mood Venkanna[1], Rameshwar Rao[2], P. Chandra Sekhar[3]

Electronics and Communication Engineering,
University College of Engineering, Osmania University, India.
[1]venkatmood03@gmail.com, [2]rameshwar_rao@hotmail.com, [3]sekharp@osmania.ac.in

***Abstract*—As the demand for high performance computing increases, new approaches have to be found to automate the design of embedded processors. Simultaneously, new tools have to be developed to short the execution time consumption, and simpler design resulting in time to market. These are to be applied for the system architecture to achieve rapid exploration in on power consumption, chip area, and performance constraints. This enables interest in Application Specific Instruction Processors (ASIPs) design and application considerably. It has higher flexibility as compared to dedicated hardware. The current case study focuses on an ASIP design methodology considering the classical parameters computational performance and area as well as energy consumption simultaneously. In this paper, the clock gating is analyzed and designed. Further it is optimized using Fast genetic algorithm (FastGA). The optimization result is shown for ICORE (ISS-core) ASIP for DVB-T acquisition and tracking algorithms. Observation shows a potential of about one order of magnitude in savings of energy for optimization.**

***Index Terms*—Embedded Processor, ASIP, Clock, Optimization, FastGA**

I. Introduction

APPLICATION-specific processing elements need modern optimized embedded systems. ASIPs architecture for mixed control/data-flow oriented taskshas been effective for medium to low data rate. A number of methodologies have been proposed in the last two decades in this regard. ASIPs are appropriate to implement embedded systems because these offer high energy performance with high programmability. To design high performance embedded systems, ASIPs with very Long Instruction Word found suitable [1-3].In this case, the Design Space Exploration (DSE) helps to determine the optimal parameters of the architecture. For small embedded systems, small scalar ASIPs with specific instructions need to be designed based on the characteristics of the target systems [4-5].

An embedded system is a computer system which performs a specific function according to our given application requirements with specific hardware environment. Some critical applications such as automotive design, controls designs (robotic machine), railways, aircraft, aerospace, DNA Sequencing, neural network, Eye lens design and fingerprinting currently working on embedded technology. Efficient co-design technology is required to reduce the operational complexity and challenges of application designing and effective memory design is required to reduce the operational complexity of the given application [6-7]. An Embedded processor evolution mechanism is required for an increasing number of features at lower power and integrated into a single chip. The Embedded system challenge is implemented with reduction of power consumption and integration of heterogeneous systems into the single chip to reduce area, power and delay [8]. An Embedded system consists of ASICs, ASIP and field programming gate array as well as the programming unit such as the DSP and these processor designs are used in various situations or time to market [9-10].

The software environment implements application developments and compilation process and hardware units implement user logic or behavior synthesis [11]. The Hardware side of design most likely consists of interconnection components such as processors, memories and communication units (buses, output/input device I/O interfaces, sensor, RTOS devices etc.) [12-13]. Embedded systems with specific constraints need to take care of Cost, Size, power and high Performances for real time design applications. An Embedded system has few basic needs for high performance as explained below.

Cost reduction for real-time design implementation Short time span for application execution and Complexity reductive architectures Runtime-aware architectures and Deploying time-analyzable Effective resource management schemes and runtime aware environments. Effective simulation tools that allow us to make design space explorations and are used for comparisons between different hardware/software designs [14-15].

Embedded design requires a temperature- aware OS solution for real-time and high-performance systems.Effective compiler technology Applicable for high-performance computing (HPC) and real-time embedded computing (EC) world [16].

ASIPs found to be performing better in specific application that includesservo motor control, digital signal processing (DSP), automatic control systems, cellular phones avionics, etc [19]. It maintains a balance between two extremes such as general programmable processors and ASICs by Liem et al[20].They offer custom section availability for time critical tasks such as thereal time multiply-adder or DSP and also provides the desired flexibility viatheinstruction-set. Complex applications require moreflexibilityto withstand design errors that includes specification changes at later stages. However, an ASIC is normally designed for specificbehavior;hencemake the design difficult to change afterwards. Inthissituation, the ASIPs can offer the intended flexibility as compared to the conventional programmable processors at lower cost. Among several issues pertaining to the ASIP design, this work intends to classify the approaches involved for different steps. It surveys the work done so far in the field of ASIP design and highlights the important contributions [21].

II. System Model and Problem Formulation

For similar task, theASIP implementations consume more power as compared to the dedicated hardware due to interconnection structure overhead and to the processor control activity. However, the processors are enough flexible and can take any software-programmable task. It creates a trade-off between low-power consumption and the flexibility. There have been several optimization options to reduce theASIP power consumption such as the clock-gating, ISA optimization,logic netlist restructuring, instruction memory power reduction etc. In some cases dedicated coprocessor are also used.

The processor unit contains pipeline unit which is controlled by DMA circuits. There are two kinds of pipeline commonly used in processor arithmetic pipeline and instruction pipeline. An instruction pipeline uses the instruction cycles' overlapping fetch, decode, and execute phases for its operation. Currently, long instruction memory plays a dominant role in the pipeline mechanism (Fig. 3). Various Pipeline mechanisms are used by various processor developer companies such as ARM, Intel and Motorola etc. according to their performance. Instruction set architecture plays a dominant role in memory storage due to code optimization.

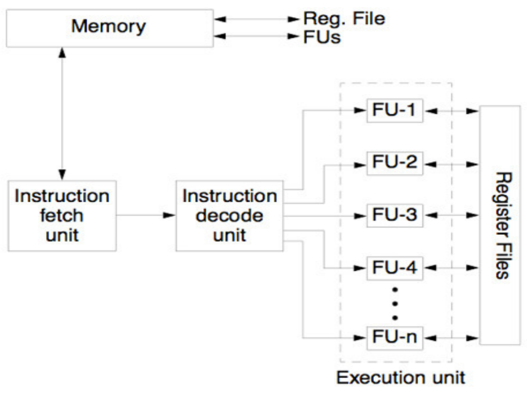

Fig. 1: Basic units for embedded system

Fig.1 shows the Basic units for embedded system. Embedded system designer used various mechanisms for processors developments have different design metrics and analyzed various processor architectures which is used in our real-time environments or System on chip such as GPP, DSP, ASIP and ASICs.

The clock gating reduces the power using the gating signals of registers in which the signals are obtaineda register's execution conditions.To pipeline register power consumption in Very Long Instruction Word (VLIW) ASIP large-scale data path can be reduced by extracting the minimum execution conditions automatically during ASIP generation procedures. Results reveal a drastic reduction in VLIW ASIPs power consumption and with small clock gating overheads.

Clock gating reduces the power consumption because of following. First, it shuts off the supply to flip flops by the redundant clock when not required in calculation stages. Second, it reduces the power of clock trees in case the clock gate is placed on higher level of the tree. But, the placed gates in clock gates results in difficult clock tree synthesis due an increase in clock skew. Similarly, the operand isolation can block unwarranted signal switching of combinational logics and thus reduces power consumption although it is associated with many circuit overheads.

For automatic ASIPs generation, scalar ASIP and VLIW ASIP methods have been proposed [18]. With an ADL also known as the Micro Operation Description (MOD), these methods help design and development of ASIPs.

A. Gating Methods

Currently there are a number of automatic clock gating insertion techniques and tools are available. Among these, the Power Compiler has been very popular and commercially used tool which automatically inserts the designated gates into the registers clock lines [22]. Since, this tool is unable to extract the registers gating signals,the clock efficiency depends on the designers. The toolcompels the designers to derive the gating signals from complex RTL manually for additional power reduction which is time consuming, as VLIW ASIP has hundreds of pipeline registers. This makes it unsuitable to explore the design space. Thus, it is essential to extract the gating signals automatically.

Finite state machine based clock gating method need feedback-free pipelines to extracts the registers gating signals. Since the method is not suitable for pipeline processors hence are not applied for generation of VLIW ASIP.

B. Clock Design Mechanism for Memory Implementation

The system performance is also strongly affected by various factors besides its instruction set, the time required to move instruction & data between the CPU and memory components. The clock system is designed for implementing memory operation execution. The average cycle is designed for implementing the clock cycle required per machine instruction is a measure of computer performances. The clock signal has various characteristics such as clock period, Clock pulses, leading and trailing edge. Clock behavior depends on upon the behavior of clock elements with memory architecture with its scheduling approaches. Clock effect can be analyzed by following parameters such as Set up time, Hold time and Propagation delay time.

TheASIP design with ADL has four basic functions such as the (a) architecture exploration (b) architecture implementation (c) application software design (d) system integration and verification.

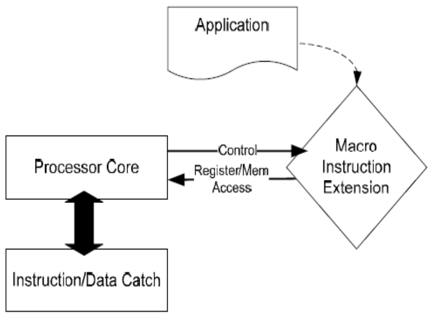

Fig. 2: ASIP *Processing Technique*

The throughput of a system depends on thedelay time of the slowest sub-circuit which is decided by the storage register.In a larger synchronous circuit, the combinational logic unit Fi is connected to two dynamic D-FFs. The input D-FF supplies sequential, clock-synchronous data to the sub-circuit. The results of the sub-circuit are accepted by the out-put D-FF sequentially and clock-synchronously, as with the input data, and are then passed on. A dynamic D-FF is a simplification of the quasi-static D-FF of Fig.2. This simplification is an appropriate solution for continuously clocked MOS circuits with a clock frequency in the MHZ range. A non-overlapping, two-phase clock is assumed. Such clocking is particularly safe and is often used within integrated circuits [14], [17]. The clock period must fulfil the following requirement:

$$T_{CLK}{}^{3}T_{D,\Phi_2} + T_{D,F_i} + T_{D,\Phi_1} + T_{\Phi_1,\Phi_2} \qquad (1)$$

Here, T_{D,Φ_2} is the delay time of a piece of data between the transmission gate clocked by Φ_2 and the input of the sub-circuit F_i. T_{D,F_i} is the delay time of the sub-circuit F_i T_{D,Φ_1} is the delay time between the output of F_i and the input of the inverter behind the transmission gate clocked by, Φ_1 . $T_{\Phi_1\Phi2}$ is the clock pause between Φ_1 and Φ_2 . It is to be noted that the delay times T_{D,Φ_2} and T_{D,Φ_1} depend on the characteristics of the sub-circuit. The input capacitance of F$_i$ influences T_{D,Φ_2} and the output resistance of F$_i$ influences T_{D,Φ_1} .The total delay resulting from the D-FFs is given by

$$T_{D,FF} = T_{D,\Phi_2} + T_{D,\Phi_1} + T_{\Phi_1,\Phi_2} \qquad (2)$$

A corresponding delay can also be given for other clock systems. In single phase clock systems with edge triggered FFs, for example, the sum of the hold and set-up time must be substituted into the equation.

In the following, a simplified representation of synchronously clocked functional units will be used. Here, the D-FFs are symbolized by a simple dot in the wiring. The delay between the input and the output of the D-FF can be described by a delay operator with delay D. In case of word oriented processing the delay operator represents a register of D-FFs.

The achievable throughput RT of a system in bits per unit time is proportional to the clock rate, i.e.

$$R_T \square \frac{1}{T_{CLK}} \qquad (3)$$

On the other hand, the clock period that determines the maximal throughout is specified by the least favorable sub-circuit.

$$T_{CLK} = \max_i \left(T_{D,F_i} + T_{D,FF_{i,}} \right) \qquad (4)$$

For high throughout, small delays are essential. Modest delays can be achieved through technological measures. By shrinking the geometric structures (scaling), the capacitances can be reduced, thus achieving a reduction in the delay. The effects of such scaling are discussed in the literature [15].

Besides technological measures, circuit techniques for increasing the throughput are also possible. Various circuit structures for the implementation of elementary operations were presented in [16]. The alternatives shown demonstrate diverse delay characteristics. According to eq. 4, the maximal delay of the sub-circuits is to be minimized. This means that only the slowest module must be improved. For example, in a signal processing task using multiple additions and multiplications, only the multiplier would have to be optimized in its propagation delay. This would be pointless for the adder. Thus, architectural measures for increasing the throughput are sought with which the dominance of the slowest module can be defeated.

Power gating may be used effectively to minimize the static power consumption or leakage. It helps to

□ Cut-off power supply to inactive units/components

 Reduces leakage

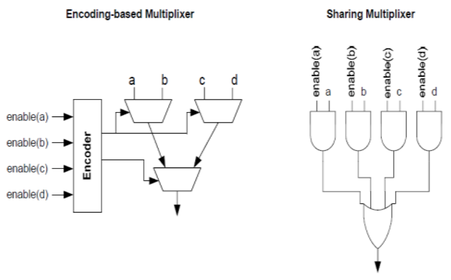

Fig. 3: Modified Clock Gating Method

III. Results and Discussion

We evaluated this work on a reconfigurable processor presented in Henkel et al. [23] that we extended for execution with hard real-time guarantees. In this the optimization of power in ICORE has been incrementally achieved. This allows us to evaluate each optimization step quantitatively. Power compiler of Synopsys with back-annotated toggle activity from gate level simulationshas been used for measurement.Table 1provides both the area and the timing delay as the optimized,unoptimized, and the original description of ICORE from Infineon and ISS68HC11 from Motorola.

Table 1. Performance Comparison

Architecture Type	Area in terms of Gates	Clock Period in nsec
ISS68HC11 (unopt.)	24.52	5.82
ISS68HC11 (opt.)	19.55	5.57
Original M68HC11	15.00	5.00
ICORE (unopt.)	50.85	6.07
ICORE (opt.)	41.40	6.80
ICORE handwritten	42.00	8.00

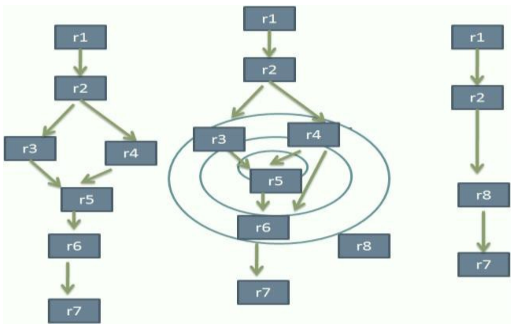

Fig. 4: Memory area implemented with resources Reducible flow mechanism and operational optimal condition

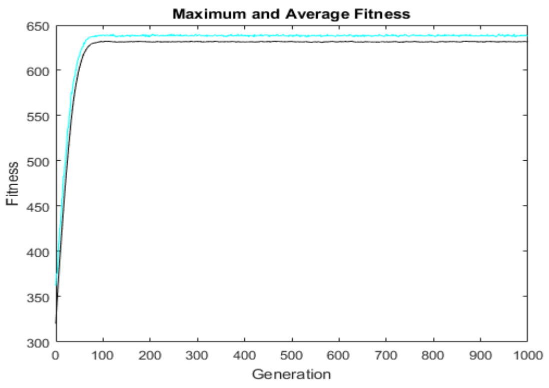

Fig. 5:Graph of fitness function using FastGA

IV. Conclusion and Future Work

In this work a novel attempt is made to estimate the WCET of a program. In this process, five key steps are identified for the ASIP design. We have surveyed the research done and performed the classification of the approaches in every step during the synthesis process. The estimation of the performance is based on either the scheduler based or simulation based method. Instruction set is generated correspondingly either using the synthesisor the selection process. The code has been synthesized either using the retargetable code generator or using the custom generatedcompiler. However, even in case of a number of approaches used in formulation of every key steps, these methods has the limit in exploring the target space architecture. Use of integration may support chip memory hierarchies and these are not explored in an integrated way. Similarly issues related to pipelined ASIP design and pertaining to low power ASIP design has not been matured yet. It has been concluded that the processor synthesis problems and retargetable code generation have been used for isolation process.

References

[1] R. White, F. Muller, C. Healy, D. Whalley, and M. Harmon,“Timing Analysis for Data Caches and Set-Associative Caches,” in *Proc. 3rd IEEE Real-Time Technology andApplications Symposium (RTAS’97)*, Jun 1997, pp. 192–202.

[2] J. Engblom and A. Ermedahl, “Pipeline Timing Analysis Using a Trace-Driven Simulator,” in *Proc. 6th International Conference on Real-Time Computing Systems and Applications(RTCSA’99)*. IEEE Computer Society Press, Dec1999.

[3] M. Arnold and H. Corporaal, “Designing domain-specific processors,” In Proc. Codesign Symposium 2001.

[4] A. Alomary*et al.*, “PEAS-I: A hardware/software co-design system for ASIPs,” In Proc. EURO-DAC 1993.

[5] J. Van Praet*et al.*, "Instruction set definition and instruction selection for ASIPs," In Proc. HLS Symposium 1994.

[6] N. Clark, H. Zhong and S. Mahlke, “*Processor Acceleration Through Automated Instruction Set Customization*”. In Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture (MICRO 36), 2003.

[7] R. Klemm, J. P. Sabugo, H. Ahlendorf and G. Fettweis, “*Using LISATek for the Design of an ASIP core includingFloating Point Operations*”, Technical report, 2008.

[8] R. R. Hoare et al., “*Rapid VLIW Processor Customization for Signal Processing Applications Using Combinational Hardware Functions*”. EURASIP Journal on Applied Signal Processing, vol. 2006 ID 46472, 2010.

[9] F. Tlili and A. Ghorbel, "*ASIP Solution for Implementation of H.264 Multi Resolution Motion Estimation*". In theInternational Journal of Communications, Network and System Sciences, Vol.3 No.5, May 2010.

[10] G. Kappen, L. Kurz, O. Priebe and T. G. Noll, "*Design Space Exploration for an ASIP/Co-Processor Architecture used in GNSS Receivers*". Journal of Signal Processing Systems, vol. 58 (1), pp. 41-51, 2010.

[11] ChristophSteiger, Herbert Walder, Marco Platzner, and Lothar Thiele. 2003. Online scheduling and placement of real-time tasks to partially reconfigurable devices. In Proc. of Real-Time Syst. Symp. IEEE,224–225.

[12] Pan Yu and TulikaMitra. 2004. Scalable custom instructions identification for instruction-set extensible processors. In Proc. of Int. Conf. on Compilers, Architecture and Synthesis for Embed. Syst. ACM, 69–78.

[13] Pan Yu and TulikaMitra. 2005. Satisfying real-time constraints with custom instructions. In Proc. Int. Conf.on Hardware/Software Codesign and System Synthesis. IEEE, 166–171.

[14] C. Mead, L. Conway*: Introduction to VLSI Systems*, Addison-Wesley, 1980.

[15] N. Weste, K. Eshraghain: *Principles of CMOS VLSI Design*, Addison-Wesley, 1985.

[16] L. A. Glasser D. W. Dobberpuhl: *The Design and Analysis of VLSI Circuits*, Addison- Wesley, 1985.

[17] Sato, J.; Imai, M.; Hakata, T.; Alomary, A.Y.; Hikichi, N. : "An integrated design environment for application specific integrated processor.", Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers andProcessors 1991, ICCD '91, 14-16 Oct. 1991, Pages: 414-417.

[18] R. R. Hoare et al., "*Rapid VLIW Processor Customization for Signal Processing Applications Using Combinational Hardware Functions*". EURASIP Journal on Applied Signal Processing, vol. 2006 ID 46472, 2010.

[19] Sato, J.; Imai, M.; Hakata, T.; Alomary, A.Y.; Hikichi, N. : "An integrated design environment for application specific integrated processor.", Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers andProcessors 1991, ICCD '91, 14-16 Oct. 1991, Pages: 414-417.

[20] Liem, C.; May, T.; Paulin, P. : "Instruction-set matching and selection for DSP and ASIP code generation.", Proceedings ofthe European Design and Test Conference, 1994. EDAC, The European Conference on Design Automation. ETC EuropeanTest Conference. EUROASIC, 28 Feb.-3 March 1994, Pages: 31-37.

[21] Praet J. V.; Goossens, G.; Lanneer, D.; De Man, H.: "Instruction set definition and instruction selection for ASIPs.", Proceedingsof the Seventh International Symposium on High-Level Synthesis 1994, 18-20 May 1995, Pages: 11-16.

[22] L. Benini, G.D. Micheli, E. Macii, M. Poncino, and R. Scarsi, "Symbolic synthesis ofclock-gating logic for power optimization of synchronous controllers," ACM Trans. Des.Autom. Electron. Syst., vol.4, no.4, pp.351–375, 1999.

[23] Jorg Henkel, Lars Bauer, Michael Hubner and Artjom Grudnitsky, 2011. I-core: A run-time adaptive processor for embedded multi-core systems. In Proc. Int. Conf.on Engineering of Reconfig. Syst. And Algorithms.