

ENERGY-AWARE MPEG-4 SINGLE PROFILE IN HW-SW MULTI-PLATFORM IMPLEMENTATION

Antoni Portero⁺, Guillermo Talavera⁺, Marius Montón⁺, Borja Martínez⁺, Marc Moreno⁺, Francky Cathoor⁺, Jordi Carrabina⁺

⁺ Dept. of Microelectronics and Electronics Systems
University Autonomous of Barcelona
08193 Bellaterra Spain
{name.surname}@uab.es

^{*} IMEC vzw, Leuven
BE-3001 Heverlee, Belgium
Francky.Catthoor@imec.be

ABSTRACT

Developers of next generation Multi-Processor Systems-on-a-chip (MPSoC) silicon platforms used in multimedia mobile devices should design efficient systems for diverse execution time vs. energy consumption trade-offs for a given quality of service. By exploiting Dynamic Voltage and Frequency Scaling (DVFS) techniques we can obtain singular computational/power trades offs points and thus design energy efficient platforms.

This paper presents a high level methodology to acquire an optimal set of working points for an MPEG-4 Single Profile (SP) Video encoder implementation. The flow starts from a MPEG-4 encoder described in C++ language which is translated to a SystemC hard/soft description which will be analyzed and further mapped into different platforms. Refined code is migrated to four different processor architectures: a processor research framework (CRISP-Trimaran), a soft core processor with specific functional units implemented on an Altera FPGA, an ASIC and a classic DSP.

I. INTRODUCTION

Heterogeneous platforms seem to be the next integration stage in electronic design. These multiprocessor system-on-a-chip (MPSoC) platforms form the multimedia portable environment will include different processor technologies or flavours: general purpose processors, various digital signal/media/image processors, embedded FPGA, etc. with their own local architectures and a complex interconnection scheme.

In that scenario, design for maximum implantation is centered in design productivity rather than in performance. Therefore, classical engineering design for the worst case scenarios hence lets to a significant energy consumption misuse.

Multimedia applications are usually fully specified in software oriented languages (like, Java, UML, SDL, C++), starting from a reference or golden model, that need in many cases, to be executed in

real-time cost/energy-sensitive heterogeneous SoC platforms.

Present and future SoC platforms will have to satisfy many critical requirements such as energy efficiency for non deterministic applications and should still provide enough computation and communication capacities, etc. at the same time. To satisfy these requirements, most SoCs will contain several types of processor cores and memory units. In most state of the art designs such as: OMAP and daVinci Digital Media SoC from TI or Cell processor from IBM[1]. These units will be connected through a hierarchical shared bus or a NoC. These platforms can handle real-time MPEG-4 SP compression and are based on heterogeneous solutions containing at least one DSP for multimedia data flow acceleration and one processor core for control flow and interactivity.

In this paper, we propose a design methodology oriented to produce efficient implementations of a given algorithm described at system level that can later be mapped into different architectures with intrinsically different power performance trade-offs. In order to illustrate its viability, an energy aware design flow for a MPEG-4 Video Single Profile (SP) specification is used as complex example.

The remainder of the paper is organized as follows. In Section II we describe the design flow followed. Section III details the SystemC energy aware models. Section IV shows the target platforms where the code is mapped. Section V and VI present our results, conclusions and future work.

II. ENCODER FLOW DESCRIPTION

The initial point of the flow was a C++ specification of the MPEG-4 standard. From this specification, we develop a MPEG-4 Single Profile (SP) code without dynamic memory allocation and without pointers. This sort of code is required for final SystemC behavioral synthesis.

Afterwards, on this refined C++ code, we carry out a series of data structure platform independent transformations (DTSE [2][3]) to obtain code that

minimizes memory transfers between Level 2 memory (L2) and the internal cache (L1). Platform dependent transformations optimize the size of the data structures for an efficient fit on the specific target memory architecture.

Later on, we modeled a SystemC description of the resulting C++ code and we analyzed the performance and gain obtained by applying Dynamic Voltage and Frequency Scaling (DVFS) [4] for multiple clock domains (frequencies). As the result of the analysis we can derive dissimilar performance-energy efficient working points. A scheme about the code transformations is shown in Fig 1.

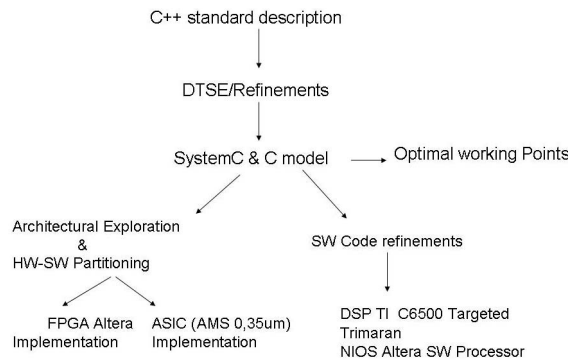


Fig 1. Design flow related to code transformation

Our SystemC model is partitioned in two parts: data flow and control flow dominant part. The data flow dominant part is mostly responsible of the transformation of each pixel in every macro-block and it is responsible for 80% of the computation. In the following of the paper we focus on the data flow part of the encoder that we call the kernel of the application and it is a good target to exploit any available Data-Level (DLP) or Instruction-Level parallelisms (ILP).

For our experiments, we studied the algorithms required to compress a macro block. Macro block can be: I (intra), P (predicted) or B (bidirectional) [5]. I blocks are just spatially compressed with the two dimensional DCT algorithm. P macro-block temporally compresses with a forward ME and also a spatial compression. B macro-block needs two ME (forward and backward vectors) and a spatial compression. Hence, I macro-block is the less computational intensive and B ones have the higher requirements.

III. HIGH LEVEL ENERGY MODEL

The energy model used in our flow takes profit of the fact that multimedia algorithms are based on some computationally intensive loops that are the kernel of the application. In the model, we consider the energy of the data computation and the

transmission between the different memory layers as main sources of power consumption.

In our case, the most computational intensive part of the algorithm is the encoder that requires in its inner loop a Multiply-Accumulate (MAC) operation. The SystemC description of the encoder allows the evaluation of different implementations with one, two, four and eight Functional Units (FU) hence being capable to exploit ILP and DLP.

The energy and power figures were estimated from different technologies and scaled to 90 nm technology with different values of V_{DD} : 1V, 1.2V, 1.5V and 1.95V. The technological parameters used for our case study come from reference 4.

IV. PROCESSOR AND HARDWARE IMPLEMENTATIONS

With the SystemC model, we can obtain different work-points with different energy-computation trade-offs. Afterwards, the application has been targeted in three different real platforms to evaluate the performance achievable with different platforms and hardware architectures [2][3].

A. Hardware prototyping FPGA and ASIC mapping

The QuartusII-FPGA Nios II Development Kit, Stratix II Edition (2S60N) is a HW-SW platform for reconfigurable systems and ASIC prototyping. The Altera architecture consists on fine-grained reconfigured logic and diverse kind of memories resources and diverse memories resources.

Embedded processor on a FPGA

This experimental system is based on an Altera NiosII [6] soft-core processor and uses the Development Kit Cyclone Edition running a NiosII system with CPU set to fast configuration with Dynamic Branch Prediction, hardware Multiplier, hardware Divider and a Barrel Shifter. The system clock is the board default clock of 50MHz. The processor has a Data Memory Cache of 2Kbytes and an Instruction Cache of 2Kbytes. Also, we set-up all RAM memory present in the board (1 MByte) available to the processor.

kernel mapped on an ASIC

Starting from the SystemC description is also possible to reach ASIC technologies, In our case, we dispose of the design flow for Austrian Microsystems AMS 0,35 μ m 4LM process that is good enough as a prove of concept and we mapped the kernel part of our implementation in this technology.

B. Target Processors

We have decided to employ diverse processors to get different performance and decide what code part is more suitable to be implemented in each one.

VLIW Processor

With the cost of silicon area decreasing, Very Long Instruction Word (VLIW) solutions are becoming an interesting and powerful trade-off between design complexity, flexibility and power consumption. In order to deliver high performance, a VLIW must exploit instruction level parallelism (ILP) available in the application. In this work; we used a real processor, a Texas Instruments DSP and a research framework for architectural exploration: CRISP-Trimaran.

TI DSP

The TMS320C64x [7] core is a VLIW processor core specifically designed to maximize channel density in communications infrastructure equipment. It contains two identical clusters with four functional units each and represents a typical clustered DSP.

CRISP-Trimaran

The CRISP framework [8] is a retargetable compiler and simulator framework based on Trimaran [9] (Trimedia Technologies Inc 1999). The architecture described by CRISP consists on a number of Functional Units (FUs) with coarse-grained reconfigurable logic and supports compiler research. Optimizations doable are high-level machine independent code transformations and “back-end” machine dependent optimizations such as instruction scheduling, register allocation, etc.

V. RESULTS

In this section, we show the results of the energy exploration and the performance results in the target processors and platforms.

A. Energy

Derived from our SystemC model Fig.2. shows the work points corresponding to different configurations. These points are obtained with different power and performance constraints. These points are obtained for different hardware architectures (number of FUs), supply voltages and running at different frequencies (150, 200, 450 and 600 Mhz).

For a given frequency, the most power hungry points in Fig.2. correspond to the large number of FUs with higher supply voltage but they need less time to compute any macro block. For example, to compress a macro block, we can use eight FUs for

the MPEG kernel at high frequency and voltage, then processor is quite power hungry. This solution will nevertheless be the fastest one to compute the macro block. On the other hand, if we do not have strict timing constraints, we can reduce voltage and frequency and then decrease power consumption using just one FU. Intermediate points were also obtained with diverse power-timing trade-offs.

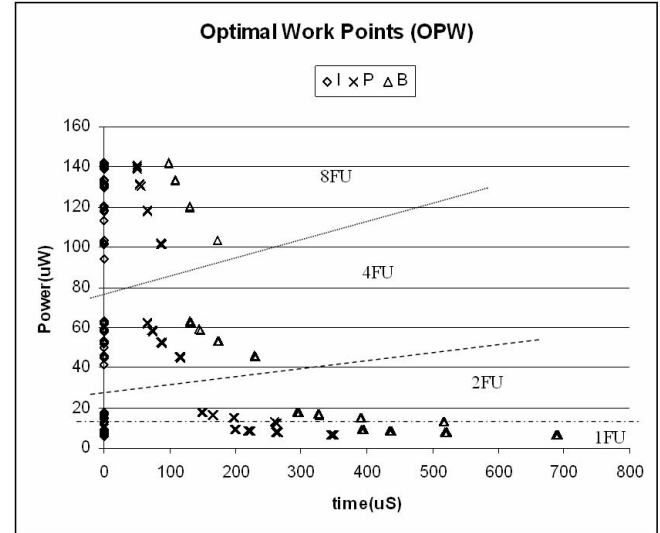


Fig 2. Optimal Work Points for diferent power-consumption vs execution time trade offs using variable number of functional units obtained from SystemC modelling using Dynamic Voltage and Frequency Scaling.

Taking as reference the high level results shown in Fig 2. we can analyzed the real performance in Crisp-Trimaran target platform changing the number of funtional units for a fixed voltage and frequency environment. The Fig 3. illustrates the outcome of this study.

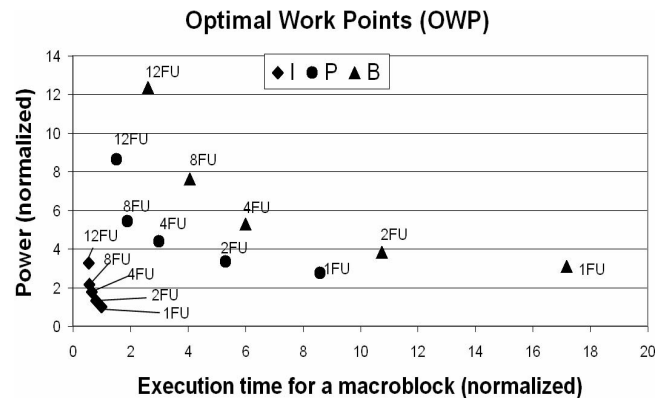


Fig 3. Optimal Work Points for diferent power-consumption vs execution time trade offs using variable number of functional units obtained from the CRISP framework at 300 MHz and 1.8V.

B. Target platform Implementations

The result of the different implementation of the MPEG4-SP encoder on the different target architectures is shown in Fig 4.

All the results in the figure are compared and normalized to the results obtained with ARM920T [10] for the encoding of the three types of macro blocks (I, P and B). For example we can see that the encoding of a macro block I takes 50% more (1,5 times) in the TI DSP than an ARM, but a macro block B is almost 80% faster (0,2 times).

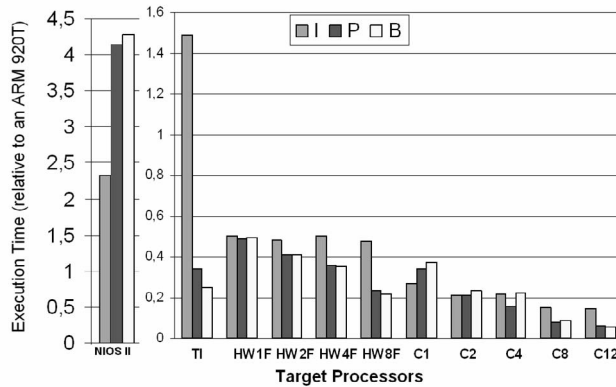


Fig 4. Execution time for the compression of the diferent macroblocks in diferent target architectures: FPGA with a softcore (NIOSII), the TMS320C64x TI DSP (TI), Custom HW with diferent functional units (HWx) and simulations with the CRISP research framework (Cx).

We can see how, as expected, the NIOSII softcore running at 50MHz in a FPGA is the slowest in all the cases. The TI DSP (running at 150Mhz) is a good comercial solution for encoding macroblocks P and B. With dedicated hardware synthesised directly from behavioural SystemC code we can achieve considerable good results which could be improved using a lower level hardware description languages at RTL level. The use of the CRISP (simulated running at 300MHz) framework shows the power of compilation optimizations.

C. MPEG Synthesis

To evaluate the HW size for FPGA and ASIC implementation, we used ForteDS Cynthesizer [11], a behavioral SystemC synthesis tool. We synthesized the MPEG kernel encoder to an Altera FPGA BASIC synthesis and to an ASIC from Austrian Microsystems technology (0,35 μ m four metal layers) and running at 80Mhz. The synthesis where realized for 1, 2, 4 and 8 functional units. We verified (using ModelSim) that the SystemC MPEG behavioral model is equivalent to the RTL model synthesized with ForteDS. Synthesis results are shown in the following table:

		<i>MPEG kernel Synthesis</i>			
		1FU	2FU	4FU	8FU
StratixII	ALUTS	25,5K	25,9K	26K	26,1K
	Registers	1400	1350	1351	1474
	Memory Bits	233K	310K	559K	715K
	DSP blocs 9b	2/36	2/36	2/36	2/36
ASIC	AREA(mm2)				
	Core digital	1,86	1,96	2,13	2,29
	TOTAL ASIC	6,53	6,63	6,80	6,96
	POWER				
	Power (W) leakage (uw)	2,825 8,09	2,833 8,08	2,840 8,35	2,842 8,60

Table 1: MPEG kernel Synthesis results: FPGA and ASIC.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we presented a flow that goes from a C++ description of an application to a SystemC implementation. From the SystemC implementation we can derive some optimal work points. Also, the same application has been mapped into five different platforms which show diverse hardware configurations and the time to accomplish the encoding of I, P and B macro blocks.

Acknowledgement:

Special thanks to David Pursley and ForteDS for helping us to synthesize the MPEG SystemC code to the FPGA platform. Thanks also to Dr. J.L.Merino from IMB-CNM for mapping the code to an ASIC.

VII. REFERENCES

1. Dac C. Pham, et al, "Overview of the Architecture, Circuit Design, and Physical Implementation of First-Generation Cell Processor", IEEE Journal of SSC, Jan2006.
2. F. Catthoor et al., "Custom Memory Management Methodology", Kluwer AP. 1998.
3. F. Catthoor et al, "Data access and storage management for embedded programmable processors", Kluwer AP. 2002.
4. A. Azevedo et al, "Profile-based dynamic voltage scheduling using program checkpoints," in *Proc. IEEE DATE Conference.*, March 2002, Paris, France.
5. Peter Kuhn, "Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation", Kluwer A.P.
6. "NIOSII Proc. Reference Handbook", Altera Corp. May 2006
7. TMS 320C000 CPU and Instruction Set Reference Guide, SPRU189F, Texas Instruments. October 2000.
8. P. Op de Beeck, et al, "Crisp: A template for reconfigurable instruction set processors". In FPL, 2001, Belfast, Ireland.
9. IMPACT Group, "Trimaran: An Infraestructure for Compiler Research in Instruction Level Parallelism", New York University 1998.
10. ARM920T Technical Reference Manual (Rev1). Third Release April 2001 .
11. ForteDS, "Cynthesizer Ref. Guide" ver.2.5.1 Set . 2005.