

NoC System for MPEG-4 SP using heterogeneous tiles

Antoni Portero, Guillermo Talavera, Marius Montón, Borja Martínez, Jordi Carabina
 Dep. MiSE ETSE 08193 Bellaterra Spain
[\[name.surname\]@uab.es](mailto:{name.surname}@uab.es)

Abstract.- In this paper we present a high level methodology to get optimal work points for a MPEG-4 Single Profile implementation. The flow goes from a C++ description of a MPEG-4 encoder to a SystemC implementation. Afterwards, code is migrated to an architecture processor research framework (trimaran) and a real implementation on an Altera FPGA and on a DSP. These different implementations are possible tiles in a mesh NoC that is modelled in TLM level and afterward synthesized with a high level synthesis tool.

1. INTRODUCTION

Currently, one of the most challenging issues in the telecom market segment is to manage the growing complexity of embedded multimedia systems and to reduce their design productivity gap. These elements stimulate a continuous development of industrial methodologies and flows design exploiting reuse, extended verification, and high level-synthesis.

SystemC [1] is a system-level design language that allows modelling and verification of complex HW/SW components. Starting from some tool-dependent HW coding styles (e.g. Forte Design Systems Cynthesizer), it is possible to reach the silicon platforms out of an industrial design flow.

Furthermore, the emergence of high capacity reconfigurable devices is starting a revolution in its use for general-purpose computing applications. Many coarse-grain reconfigurable architectures appeared as reconfigurable coprocessors, structures ASICs, etc. considerably relieving the burden from the main processor in many multimedia applications due to their very high degree of parallelism. In addition, they generally have wider flexibility than an application specific circuit.

TLM (Transaction Level Modelling) SystemC description permits a rapid system development. Parts of the systems are locally synchronized by a clock but globally there is a handshaking or a shared-bus communication. When we try to synthesize we do not know exactly the number of resources and clock timing of each tile. Hence, GALS philosophy (Globally Asynchronous

Locally Synchronous) make independent computation and communication. Any tile of the system can be connected to another if they have the same protocol. The communication to connect a tile and the NoC is based on a burst transfer to an AMBA bus [2].

Following SoC evolution driven by the increase in integration density, NoCs (Networks on a Chip) [2] are being used as a communication infrastructure between tiles containing IPs (Intellectual Property blocks). OCP-IP organization is leading a community trying to standardize the communication protocols and promote the development of tools to automate the NoC strategy for multiprocessor System-on-Chip.

This paper shows an efficient use of NoCs to interconnect several replicated tiles of a MPEG-4 Single Profile implementation with different computation requirements. The NoC employs packet switching, a communication mechanism in which packets are individually routed between cores, with no previously established path [4]. The wormhole packet switching mode is used to avoid the need for large buffer spaces. The routing algorithm defines the path taken by a packet between the source and the destination. The deterministic algorithm is employed.

The system will have a manager that will decide the tile assignment according to performance requirements, basically execution time versus power consumption.

Paper is divided in seven parts; first section is a description of our IPs MPEG-4SP and NoC, second part explains Systemc development. Third part, describes the tiles that are connected to the network. Part fourth briefly the energy model that we have used. Fifth part gives details about the different possible target platforms for the Network tile. Part six presents some results. Finally, part seven explains conclusions and future work.

1. MPEG-4 SP TILE DESCRIPTION

The starting point of the flow was a C++ specification of the MPEG4 standard. From this specification we built a MPEG-4 single profile code without dynamic memory allocation and without pointers needed to offer a posterior SystemC synthesis [5].

Once we had the modified C++ implementation, we carried out a series of data structures transformations to obtain an optimization to fetch data from external memory (Level 2) to the internal cache (L1). These optimizations are called DTSE transformations [6] and produce code that minimizes memory transfers between L2 and L1. The ME algorithm depends on input data whose model process is a pixel (combination of picture & element) is the smallest element of a display which can be assigned to a color. The new data structures also will perfectly fit in the memories of a FPGA which will be one of the target platforms of our implementations.

Later on we modeled a SystemC description of the resulting C++ code and we realized a study of the performance we can achieve with different hardware architectures and the gains we can reach applying Dynamic Voltage and Frequency Scaling (DVFS) [7] for multiple frequencies (clock-domains). Our SystemC and energy model is fully explained in section 4.

With the different work points for performance and energy efficiency, we targeted the MPEG4 description in different state of the art microprocessors as we show in section 5.

1.2 NoC TILE DESCRIPTION

The router is based on the one proposed in SoCIN[8] and RaSoc[9]. It is a router composed by five channels: L (Local), N (North), S (South), E (East) and W (West). Each one of these channels is composed by two opposite unidirectional channels, one as an input and the other as an output. Besides, each channel has two control signals (*val* and *ack*) that let to do a communication with *handshake*.

There is a process for each channel. Then, with five independent processes in the same router, we can address simultaneously different information incoming from different channels that goes to different destinations. For example, one router may address data that incomes from E and goes to N while data that incomes from S goes to L.

However, we have to control the order priority if more than one process wants to write in the same port at the same time. We have implemented a round-robin priority policy that guarantees the port usage according to the order in which the blocks arrives to the router. Each input port has a counter and each output port has an internal busy signal to indicate if the port is been used. If this is the case, the counter of the input port goes increasing until the output port is free, and then the input port with the higher counter value will be the next to write to the output.

The five channels router architecture lets to implement easily 2D orthogonal topologies as mesh or torus. These kinds of topologies have an

easy routing, in our case routing XY, and simple connection that lets a direct replication of blocks through the system. The system is established as a system of coordinates where each router in the network is identified by a pair of coordinates XY. To send data to a resource means to send data to the router which contains it.

The resource connected to the router determines which will be the next resource to process the information. The coordinates of each resource are shared in a common file for all the routers. Every router has its own wrapper which generates its own routing table, because this depends on the resource type joined to the L channel.

In cases N, S, E and W, the wrapper checks the routing information of the header of the incoming block and, if it is not the destination, it updates the routing information (decreasing the *Xmod* or *Ymod* fields) and addresses it to the output channel. If the router which reads the block is the destination, the wrapper will pass the NxM block data information, but not the header, to the resource connected to the L port.

The communication between the resource and the router is *store-and-forward*: all the packets that compound the block are stored in the router and it is not sent until all NxM packets have been received. However, the communication between routers is a *wormhole*: the packets are sent as soon as they arrive to the router. In this way, once the wrapper has created the packet with the header, its routing information determines a wormhole routing with XY routing algorithm. The XY routing algorithm always route first in X dimension and then in the Y dimension. This is a deterministic routing because packets always follow the same route for the same source-destination pair.

The router has five SC_THREAD processes (clocked thread process), one for each input channel. SC_THREAD is the only process of the three that has SystemC(SC_METHOD, SC_THREAD and SC_THREAD) which let to use the wait_until() call. This instruction let us to implement the handshaking communication and behavioural synthesis.

At the starting point, with the wait_until() instruction, each thread of each channel is waiting for a *val* event, which means that there is some data in the channel to be read.

As it has been mentioned before, the router has five input channels and five output channels. Channels L, which connects the router with the resource, have been implemented with the primitive *sc_signal*. On the other hand, N, S, E and W channels have been implemented with *sc_fifo* channels of SystemC. Currently, the *sc_fifo* channel is a TLM synthesizable primitive with Cynthesizer 3.1 tool. Furthermore, it let us to implement a fast simulation of the system and determine the fifo sizes for synthesis of the system. *Sc_fifo* is a SystemC

channel that may contain any data type and it is used to manage data flow. Consequently, it can manage the structure packet build by the wrapper (NxM data block and a header). For the simulation we indicate the number of clock cycles with the sentence wait(t) (where $t = NxM + \text{header}$).

The packet header size is of 16 bits, where 8 bits are routing bits and the other 8 bits are reserved for future information. From the header's routing bits (see table 1) Xdir and Ydir have size 1 bit, and Xmod and Ymod have size 3 bits.

The 3 bits of Xmod and Ymod fields let to implement NoCs of 8x8 routers that means the possibility to connect until 64 resources. However, the tests we have done were on a network of only 3x3 elements. More accurate information about this NoC implementation is explained in [10][11]

TABLE I
ROUTING INFORMATION

Field	Meaning
Xdir	0/1, packet must be routed in the East/West direction
Xmod	Number of links remaining to be traversed on X
Ydir	0/1, packet must be routed in the North/South direction
Ymod	Number of links remaining to be traversed on Y

2. TILES CONNECTED TO THE NETWORK

Connected to a tile there is a DMA that is the responsible to manage information from L2 external memory to L1 internal memory. This DMA fetch data pixel that have to be compressed from different images depending on the GOP type that is going to be produced. DMA just fetch datapixels to L1 memory of macro-block size. The NIC (Network Interface Controller) is AMBA bus compliant and connect L1 memory with the Local connector. See fig. 1 below:

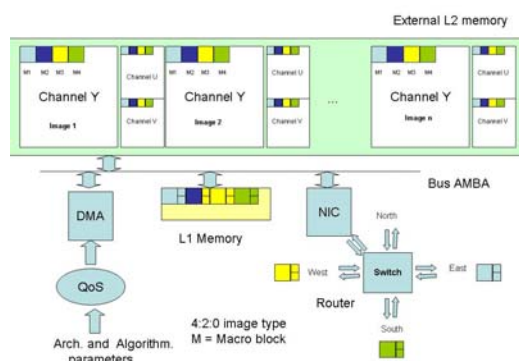


Fig. 1. Tile that connect external memory with NoC NIC.

These macro-blocks are encapsulated in a packet and sent to the tile where are connected several MPEG-4 SP MPEG modules. Macro block are

sized of 6 blocks of 8x8 pixels due that we are compressing images type 4:2:0.

MPEG 4 SP tile

MPEG-4 [11] SP SystemC model was 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. In our case, we selected the Motion Estimation (ME), Motion Compensation (MC), Discrete Cosinus Transformation (DCT), Quantization (Q) and Zig-Zag (ZZ) algorithms. All next sections, this part will be named model *kernel* of the tile; they are computation intensive and a good target to exploit the available Instruction Level Parallelism.

The control flow dominated part is related to the creation of the transport stream and Arithmetic Coding (AC) which could be easily implemented in a micro-processor and does not consume many resources. In the rest of the paper we will call that part of the MPEG model as *COD*. The implementation of the control flow part in software also provides another advantage: the reuse of the algorithms between MPEG-2 and MPEG-4 SP because of the similarity of the kernel algorithms.

The Module Quality of Service (QoS) is the part of the model that decides if data is going to be compressed creating macro-blocks B, P or I depending on channel bandwidth and image quality requirements. A B macro-block compresses more the data than a P one and a P one more than an I macro-block [11]. The compression ratio is directly related with the time the process need to create and assemble the data for the different macro-blocks then the creation of a B macro-block takes longer than a P one and this more than an I.

These differences in algorithm complexity, processing computation and configurations to form a macro-block are managed by the QoS module to obtain the user requirement for a real time implementation

The ME and MC algorithms (computing the inter-frame compression) need data structures to produce P and B macro blocks. All macro blocks are Intra-frame compressed with the DCT algorithm. These spatial components are transformed to frequencies ones thanks to the Discrete Cosinus Transform (DCT) and resulting high frequencies are quantized since human vision is less sensitive to them.

It is the way that we lose non-visual high frequencies and finally we recollect these frequencies with a zigzagged scan. These frequencies values are translated to codes with an arithmetic encoder algorithm. Finally, these codes are loaded in a transport stream that produces the MPEG-4 SP stream.

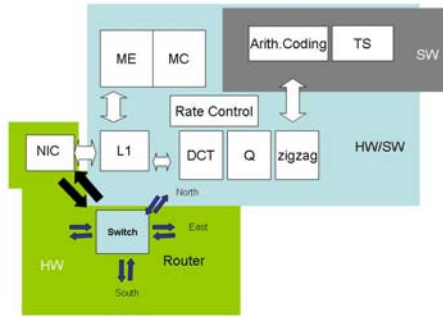


Fig. 2. MPEG-4 SP block diagram.

3. METHODOLOGY

When one or more processors execute a set of concurrent tasks, a predefined scheduling algorithm must be applied to decide the task execution order. For a multi functional unit system, an assignment procedure must determine now many functional units FUs will execute each task.

We use the terminology defined in [13] and we define:

Sub-Task: the atomic scheduling unit of our design-time scheduler; consists of control dataflow graph (CDFG) nodes and arcs. See figure Fig. 3.

Task: a group of thread nodes. The design-scheduler works inside each task, whereas the run-time scheduler treats a subtask as an atomic scheduling unit.

The purpose of task concurrency management [14] is to determine a cost-optimal, constraints-driven scheduling, allocation, and assignment of various tasks to a processor with a set of Functional Units (FUs). Different FUs execute a GOP (Group of Pictures) as different tasks. Each task can be divided in the minimal image size (a macro block) and each macro block is divided in diverse sub tasks (T1 to T8), the algorithms to obtain the compressed data are allocated in the different FUs.

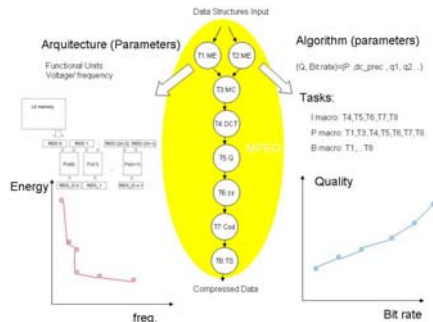


Fig. 3. Task Graph: One Task with several sub-task Nodes

These differences make possible to explore the cost-performance trade-off at system level. Each of these tasks consists of many sub-task nodes (ST) that can be seen as an almost independent section of the code.

The second stage consists of a three phase scheduling approach:

- a) First the design-time scheduling step is applied to each one of the Tasks identified in the system. Different from traditional design-time scheduling, it does not generate a single solution but a set of possible cost-performance trade-off points.
- b) Second, if the application is executed, a specific run-time scheduler dynamically selects one of these trade-off points for each running task to find a global energy efficient solution.
- c) Finally, once the energy-performance work point is selected, we can still tune our application and obtain different types of compressed macro blocks (I, P or B).

4. ENERGY MODELLING

From SystemC implementation of the encoder (Fig. 2.), we evaluated different versions of the encoder with different numbers of Functional Units (1, 2, 4 and 8 FU). In our model we considered the energy of the computation of data and the transmission between the different levels of memory.

The energy and power figures were estimated from different technologies and scaled to 90 nm technology with different values of Vdd: 1V, 1.2V, 1.5V and 1.95V which are the technological parameters used as the basis for our case study.

5. TARGET PLATFORMS

In this work, we used a methodology that goes from a C++ description of the target application to a SystemC implementation. From that implementation, we can obtain different Pareto 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. As base for all our experiments, we took an ARM7TDMI [15] and the results in section all the experiments are referred to the results obtain with the ARM processor.

5.1. Embedded processor on a FPGA

The Nios 0 embedded processor is a general-purpose RISC CPU implemented as a soft core in Altera FPGAs. Our experimental system was based on a Nios Development Kit Cyclone Edition running a Nios2 system with CPU set to economical configuration. The system clock was the board

default clock of 50MHz. Also, we set up all RAM in the board (1 MByte) available to the processor.

5.2. TI DSP

The TMS320C64x [16] 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 processor.

5.3. CRISP-Trimaran

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 the CRISP framework [17], which is a retargetable compiler and simulator framework based on Trimaran [18] (Trimedia Technologies Inc 1999). The architecture described by CRISP consists on a number of Functional Units (FUs) with coarse-grained reconfigurable logic.

6. RESULTS

6.1. Energy-optimal work points

Derived from our SystemC model, we can see in Fig. 4. some possible work points for different power and time constraints. This work points were obtained for different hardware architectures, voltages and running at different frequencies (150, 200, 450 and 600 Mhz).

For a given frequency, the most power hungry points in the graph correspond to the bigger number of FUs with higher voltage. We can derive that for compressing a macro block, we can use eight FU for the MPEG kernel. Then, increasing voltage and frequency (that increases power consumption) we can reduce the time needed to compress the macro block (point up on the left). In the other hand, if we do not have timing constraints, we can decrease power using just one FU (down point on the right). Several intermediate points were also obtained with diverse power-timing trade-offs.

Fixing the architectural coordinates in a multi-dimensional objective plane, we prune our research space but we still have some freedom tuning algorithm parameters. In figure 4. we show the image quality versus bit rate for a fixed OWP. In fact, these points should be a range of values

where users can perceive reasonable image quality in real time.

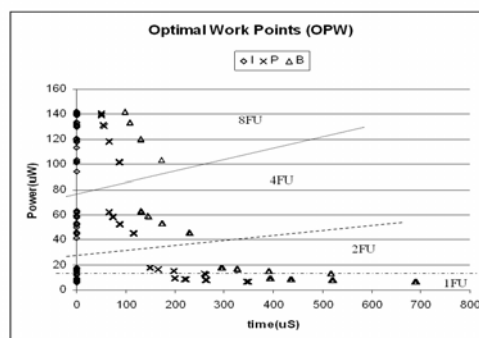


Fig. 4. OWP (Optimal Work Points)

a. Target Processors Work points.

We decided to employ diverse processors to get different performances and decide what code part is more suitable to be implemented in each one according with different

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. For our experiments, 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 is VLIW based and, as all VLIW processors, the processor executes instructions that are composed of parallel operations. These Operations are executed in the Functional Units (FUs) with coarse-grained reconfigurable logic.

Compiler research optimizations doable are high-level machine independent code transformations and “back-end” machine dependent optimizations such as instruction scheduling, register allocation, etc.

Cycles of execution of the MPEG kernel for different hardware architectures on the CRISP framework.

The DMA send macro block data to each MPEG-4 in round robin way. Currently, for system verification all MPEG modules are deterministic in future work each MPEG behavioral that depends on image type Q/bit_rate requirements will be completely independent. The used work point is 8FU, 600MHz. More statistics have to be done and other IP will be connected to the NoC (p.e: Disktra algorithm, and algorithms to find characteristics of images such as color, objects, or edges). We are working in a MPEG decoder and a Disktra algorithm that is an algorithm that work for finding characteristics in a sea of informations. We want to use this algorithm to find characteristics of the image for example color , edges or objects to create metadata compliant with MPEG-7.

6.2. Synthesis model verification MPEG with ForteDS

We used benchmarks for the verification of our models. The images we used are the foreman (176x144 sizes) benchmark. This image is verified with tool for MPEG-4 streaming. To evaluate the size of the different implementations, we used ForteDS Cynthesizer [5], a powerful SystemC synthesis tool. We synthesized the kernel of the MPEG encoder onto a FPGA with one FU and eight FU for 0.9um 4 metal layers from AustrianMicrosystem technology. The results obtained are shown in Fig. 5. At this moment, we are able to simulate produced rtl code produced by forteDS. The NoC is synthesized at TLM level with forteDS cynthesizer 3.1.

MPEG Synth	area (square mm)		Total	register bits
	combinational	register		
1 FU	0.67	0.31	0.98	1156
8 FU	0.75	0.34	1.09	1219
NoC Tile	0.68	0.56	1,24	1986

Fig. 5. Area of the synthesis of the MPEG and tile of the NoC kernel.

7. 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 implementation we derive some optimal work points. Also, the same application has been mapped into different platforms which show diverse hardware configurations and the time to accomplish the encoding of I, P and B macro blocks. The previous results show that the OWP are not always easy to find. Different considerations as frequency and voltage (hence consumption) and the number of functional units can not be neglected if we want to

maximize battery life. Run time dynamic voltage and frequency scaling will allow us to save lot of power.

References

- [1] www.Systemc.org
- [2] <http://www.arm.com/products/solutions/AMBAHomePage.html>
- [3] David Bertozzi et al, "NoC Synthesis Flow for Customized Domain Specific Multiprocessor Systems-on-Chip ", IEEE Transactions on Parallel and Distributed Systems, vol 16, No.2, Feb.2005.
- [4] Guerrier. P, Greiner. A. "A generic architecture for on-chip packet-switched interconnection". In design Automation and Test in Europe (DATE), 2000, pp.250-256.
- [5] <http://www.fortedds.com/products/cynthesizer.asp> Cynthesizer 3.1 Tool
- [6] F. Catthoor et al, "Data access and storage management for embedded programmable processors", Kluwer AP. 2002
- [7] A. Azevedo et al, "Profile-based dynamic voltage scheduling using program checkpoints," in *Proc. IEEE Design, Automation and Test in Europe Conference.*, March 2002, Paris, France.
- [8] C. A. Zeferino and A. A. Susin, "SoCIN: A Parametric and Scalable Network-on-Chip", *SBCCI'2003*, IEEE CS Press, 2003. pp.169-174.
- [9] C. A. Zeferino, M. E. Kreutz and A. A. Susin, "RASoC: A Router Soft-Core for Networks-on-Chip", *DATE'2004 - Designer's Forum*, IEEE CS Press, 2004.
- [10] Antoni Portero, Ramon Pla, Jordi Carrabina, "SystemC Implementation of NoC", International Conference. on. Industrial Technology, IEEE-ICIT 2004
- [11] Antoni Portero, Ramon Pla, Aitor Rodriguez, Jordi Carrabina "NoC Design of a Video Encoder in a Multiprocessor System on Chip Solution", The 17th International Conference on Micro Electronics, IEEE-ICM 2005
- [12] Ian E.G. Richardson, "H.264 and MPEG-4 Video Compression. Video Coding for Next-generation Multimedia", John Wiley & Sons Inc.
- [13] Peng Yang et al; "Energy-aware runtime scheduling for embedded-multiprocessor SOCs" *Design & Test of Computers*, IEEE ,Volume: 18 , Issue: 5 , Sept.-Oct. 2001
- [14] F. Cathoor et al., "Custom Memory Management Methodology", Kluwer AP. 1998.
- [15] <http://www.arm.com/products/CPUs/ARM7TDMI.html>
- [16] TMS 320C6000 CPU and Instruction Set Reference Guide, SPRU189F, Texas Instruments. October 2000.
- [17] P. Op de Beeck, et al, "Crisp: A template for reconfigurable instruction set processors". In *FPL*, pages 296-305, 2001, Belfast,Ireland.
- [18] The Trimaran Compiler Infrastructure, <http://www.trimaran.org>