Metrics for Reconfigurable Architectures Characterization: Remanence and Scalability

Pascal Benoit, Gilles Sassatelli, Lionel Torres, Didier Demigny, Michel Robert, Gaston Cambon

Metrics for Reconfigurable Architectures Characterization: Remanence and Scalability

Pascal Benoit[1], Gilles Sassatelli[1], Lionel Torres[1],Didier Demigny[2], Michel Robert[1], Gaston Cambon[1]

LIRMM, UMR UM2-CNRS C5506,
161 rue Ada, 34392 Montpellier Cedex 5, France
(33)(0)4-67-41-85-69
{first_name.last_name}@lirmm.fr

ETIS, UMR-CNRS 8051,
6 av. du Ponceau, 95014 Cergy Pontoise Cedex, France
(33)(0)1-30-73-66-10
{name}@ensea.fr

ABSTRACT

Target applications for mobile devices such as PDAs and cell phones require increasingly powerful architectures. This challenge has spawned different hardware acceleration styles like configurable instruction set processors, coprocessors, and ASICs. Despite acceptable, these solutions show today a lack of flexibility considering rapidly changing standards. Structurally programmable architectures can today provide a trade-off between performance of hardwired logic and flexibility of processors. More and more reconfigurable architecture are today available as IP cores for SoC designers. These ones often differ according to several parameters (granularity, reconfiguration mode, topology…). Therefore, it is not straightforward to compare different architectures and choose the right one considering both actual and future requirements. This paper proposes a general model for reconfigurable architectures and give a set of metrics which prove useful for architecture characterization. The methodology will be illustrated on a dynamically reconfigurable architecture: The Systolic Ring.

1. INTRODUCTION

Thanks to process geometries dropping, nowadays silicon technologies allow the integration of complete systems on the same silicon die (SoC), merging different IP (Intellectual Property) cores. New communication products like cellular phones and pocket PCs are more and more based on a SoC approach, allowing to significantly decrease cost and power consumption for leading edge communication devices. All digital-level functions are managed by both processors cores (one or several, either general-purpose or specific) and hardwired accelerators. The resulting lack of flexibility has motivated the integration of reconfigurable cores. These architectures provide hardware-like acceleration style (e.g. ability to process concurrently multiple data), while retaining most of the software flexibility : a simple bitstream define the functionality. Among the last couple of years lots of new approaches appeared [2][3]. Real innovations like coarse grain reconfigurable fabrics [3] or dynamical reconfiguration have brought numerous improvements, solving several weaknesses of traditional FPGA architectures. Besides this point, several recurrent issues remain, and the proliferation of architectures lays to an additional problem for SoC designer: choose the right IP core for a given set of specifications. Despite some works have already proposed some useful tools, like the *Dehon criterion*[11], allowing to compare the computing density for different architectures in different silicon technologies, the need of additional metrics is now obvious. For instance, no previously defined metric take into account that a given application may be implemented using different algorithms, according to different execution models (processor or hardwired). The goal of this paper is to address this characterization problem by the way of defining two metrics: *remanence* and *scalability* allow to compare more efficiently different architectures dedicated to digital signal processing. This paper is organized as follows:

- The first section presents briefly existing architecture families dedicated to digital signal processing. A general model for these architectures is then introduced and illustrated by several examples.

- The second section presents the *remanence,* a metric allowing to quantify the dynamical character of the reconfiguration. It gives some interesting information for under constraints power consumption reduction. *Scalability* issues are then introduced, and the developed characterization methodology is described.

- The third section presents both the Systolic Ring, a coarse-grain dynamically reconfigurable architecture dedicated to data-flow dominated applications acceleration.

- The fourth section is an extended analysis of the Systolic Ring according to the previously described characterization methodology; i.e. *remanence* and *scalability*.

2. DIGITAL SIGNAL PROCESSING SOLUTIONS

Each architecture dedicated to digital signal processing exhibits benefits as well as limitations, extensively listed in [8] for Von Neumann architecture and [3] for reconfigurable architectures (RA).

Figure 1 depicts a general model for both processors and RAs. Depending on the architecture, each constituting element differs.

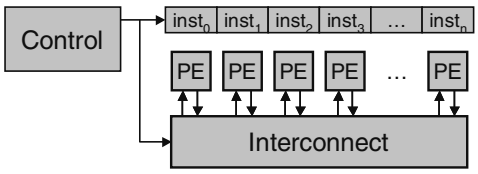

Figure 1. General model

The constituting elements are:

- Interconnect subsystem
- Array of processing elements (PE), PE structure
- Control unit
- Instruction / Configuration memory

2.1 "*Von Neumann*" architectures

Processors are based on the Von Neumann paradigm [8][9]. Their architectures feature two distinct components: the controller and the data path. The first is in charge of reading an instruction from the memory each machine cycle, then it applies the corresponding micro-instruction sequence to the data path, and so forth. Operation execution is thus carried out in the data path in a sequential way.

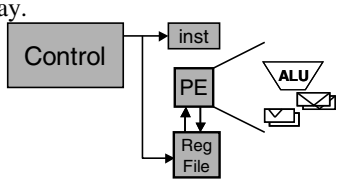

Figure 2. RISC and CISC processor model

2.1.1 CISC and RISC

The first processors were called CISC (Complex Instruction Set Computer) [8]. Due to the presence of hundreds of instructions, multiple addressing modes, multiple instruction formats, the corresponding sequencer occupied an imposing silicon area, usually between 70% and 90% of the processor area. In addition, the average number of micro instructions needed for the execution of an assembler instruction was significant (increased machine cycle time, decreased performance). Afterwards, RISC processors (Reduced Instruction Set Computer) appeared featuring a largely reduced instruction set. The usual presence of one to two addressing modes and simple instructions led to a largely decreased sequencer silicon area, and also increased performance: the finer instruction granularity allows the execution of any instruction in one to two clock cycles. Compilers exploit plentifully the limited instruction set and usually prove efficient (in comparison to hand-coded assembly programs). This execution model does not allow any form of parallelism, performances are for that reason limited. However, RISC processors are usually highly pipelined (form of parallelism), and the simplicity of the data path induced the emergence of *single-controller*, *multiple-datapath* architectures, later defined as VLIW (Very Long Instruction Word) processors.

2.1.2 Superscalar Processors

The challenge posed by the computational bandwidth requirements induced the emergence of architectures allowing Instruction Level Parallelism (ILP) [8]. Thus, in opposition to scalar processors (performing a single instruction by machine cycle), superscalar processors are able to carry out several instructions, by dynamic deduction of the data dependencies. This task is performed by the control unit (and not by the compiler). Several processing elements are thus available (Figure 3). The performances achieved by these processors are generally higher than their scalar counterparts. However performances are at the price of a greatly increased silicon area of the controller (deduction of dependencies, hazard resolution, etc.), these architectures are for that reason poorly scalable.

2.1.3 VLIW Processors

VLIW architectures are more and more used in current DSPs [10]: they also carry out parallelism at the instruction level. The essential difference relies in the fact that the data flow graph is built during compilation and not at run-time: parallelism at the instruction level is thus not performed by the hardware but at the software level, by the compiler. A VLIW instruction consists of several RISC instructions (Figure 1), each one being carried out in a dedicated unit. Due to the fact that each PE must be configured each cycle, scalability is limited: instruction bandwidth / memory becomes too high. Moreover, compilers usually don't benefit entirely from the ILP capabilities, compelling programmers to hand-code critical part at the assembly level. Performance gaps of one order of magnitude are sometimes stated [7].

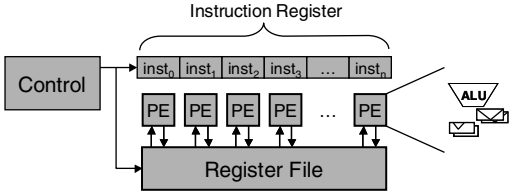

Figure 3. Superscalar and VLIW processors model

2.2 Reconfigurable architectures

2.2.1 Fine grain reconfigurable architectures

Well known fine grain RAs like FPGAs [1] allow in some cases interesting acceleration over processor-only solutions. A major interest with these components is the ability to synthesize pipelined datapath: the PE array is two-dimensional. This operation is not possible in Von Neumann-like architectures: multiple processing units are in some cases available (superscalar, VLIW) but only in a one-dimensional array topology. These components feature bit-level reconfigurable logic, often Look-Up-Table based.

We can distinguish two main families of FPGAs:

Statically reconfigurable FPGAs.

The configuration is fixed during the whole computing phase, i.e. processing must be stopped to reconfigure the FPGA. No control is in this case needed (figure 4) as the configuration is preloaded in the configuration layer.

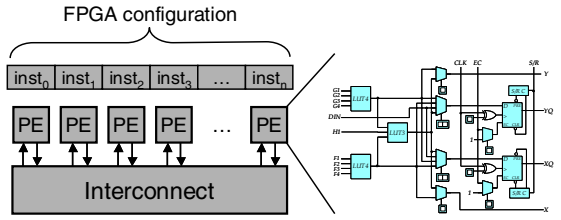

Figure 4. Statically reconfigurable FPGA

Dynamically reconfigurable FPGAs.

The configuration can be modified during processing. It allows to make partial reconfiguration: a set of the reconfigurable logic is reconfigured while the remaining resources keep on processing. In this case a control unit is needed to manage the partial reconfiguration (figure 5).

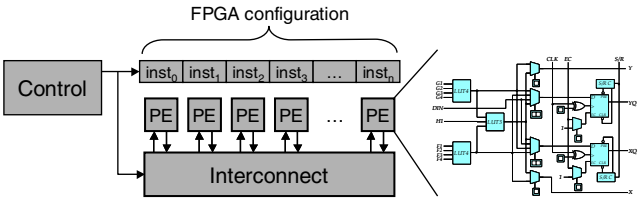

Figure 5. Dynamically reconfigurable FPGA

2.2.2 Coarse grain reconfigurable architectures

Coarse grain RAs propose an interesting compromise: while providing higher clock frequencies they lower the cost overhead for dataflow dominated applications thanks to the presence of hardwired arithmetic operators (coarse grain) instead of bit-level reconfigurable logic. Almost all existing architectures are dynamically reconfigurable [5]: the unavailability of bit-level reconfigurable logic in the operating layer prevents the realization of control structure (like Finite States Machines). Hence, all control operation must be managed externally, by a dedicated control unit. Existing architecture are for these reasons often close to a VLIW processor (figure 6).

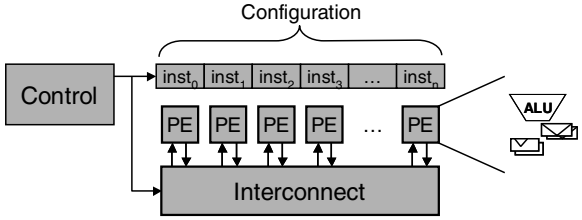

Figure 6. Coarse grain reconfigurable architecture

The main differences between a VLIW processor and a coarse grain RA are:

- Reconfiguration rate, later defined as remanence. A VLIW processor modifies the instruction (i.e. configuration) of all its PEs each machine cycle while only a subset of the PEs are reconfigured in a coarse grain RA. Hence, the presence of numerous PEs would imply an too important instruction bandwidth / control unit.

- Operating layer topology. The PE array of a VLIW processor is one-dimensional which implies load/store data from/to a register file each cycle. Most RAs are two-dimensional which allow them to implement pipelined datapath.

3. Remanence and Scalability

3.1 Remanence

A RA is constituted by a set of operators N_a running at the clock frequency F_e. Each architecture is able to reconfigure N_c operators each configuration cycle of frequency F_c. F_c may be different from F_e, depending on the considered architecture. The remanence is simply defined by the following expression:

$$R = \frac{N_a.F_e}{N_c.F_c}$$

The remanence subsequently characterizes the dynamical character of the RA by reporting the number of cycles needed to reconfigure the PE array. This criterion provides an information on the minimal amount of data to be processed between two configuration cycles.

- If the configuration phase is shadowed, a new configuration is loaded during processing. The

configurations are then switched within the next clock cycle. The architecture is efficient if during this cycle most of the operators are processing data.

- If the configuration phase is not shadowed, the number of processing cycles must be greater than R for a limited overhead: usually in the range of 10 to 20 times R.

Moreover, a data parallelism of β (β data processed concurrently) increase according to a factor β the minimal number of data to be processed between two configuration cycles. Therefore, the ratio between the amount of data to be computed and R figure out an important information which helps to choose between data or instruction parallelism. Besides this point, one can notice that 1/R is a metric assessing the dynamical character of an architecture. The less R, the more dynamically reconfigurable the architecture is. The system reconfiguration frequency is lower to F_e/R.

This metric has three main advantages:

- It reports the dynamical character of an architecture independently from its granularity: The operators can either be fine grain (CLBs) or coarse grain (multipliers, ALUs). This is done thanks to the use of the concept of operators instead of any lower-level consideration.

- Although some architecture provide only inter-operators path routing, this implies to stop processing while configuring. Hence, it is functionally equivalent to reconfigure the operators. It can nevertheless be more efficient to directly reconfigure the operators. For a given processing power, N_c can be greater or/and require less configuration bits. This it implicitly taken into account by the remanence thanks, again, to the concept of operators.

- No matter how the reconfiguration takes place. It can be done in a single pass, after the processing related to the current configuration is done, or continuously, a few operators being reconfigured each cycle, while processing keeps on.

Remanence and power consumption

In a processor, up to 50% of the power is consumed in the control unit. Besides technological considerations, reconfiguration frequency and volume (i.e. number of bits) might consequently impact on the power consumed. Some architectures which provide a fixed mode (configuration fixed for the processing of an important amount of data) can consequently achieve interesting power savings.

The processing power P_{proc} of a given architecture can be expressed as the product between the number of operators N_a and the clock frequency F_e ($P_{proc} \sim N_a.F_e$).The power consumed can then be expressed as:

$$P_{cons} \sim N_a.F_e.U^2$$

with U being the voltage supply. According to this formula, equivalent power saving might be achieved by either optimising N_a or F_e. However, decreasing the clock frequency allow to decrease proportionally the voltage supply. Let assume such a solution, the power consumed can be expressed as:

$$P_{cons} \sim N_a.F_e^3$$

Then the ratio P_{cons}/P_{proc} grows according to a factor F_e^2. For a given processing power, it is then worthwhile to increase the number of operator and reduce accordingly the clock frequency. Nevertheless, applying such an approach might increase consequently the control unit complexity and then its power consumption. This observation figures out clearly the significance of the remanence. The power consumed is proportional to the bit switching activity (each second). Hence, it is possible to define a cost in power consumption per MIPS by the way of considering both processing-related cost and configuration-switching cost.

3.2 Scalability

Due to the continuous technology scaling, scalability is today becoming a key issue ; the problem can be stated as follows: given a customisable architecture model (in terms of number of PEs), how does the N_a/A ratio grow, N_a being the number of PEs and A the core area. We define the operating density OD as the ratio N_a/A. Hence, for an architecture fully scalable $OD(N_a)$ will be constant.

Accordingly to our general model (figure 1), and assuming the core area as the sum of the constituting elements' area, architecture scalability analysis sum up to each component scalability analysis:

- Processing elements: This is probably the only element which is almost always scalable, independently of the architecture. The silicon area increase is directly proportional to the number of PEs.

- Configuration memory: Applying a factor *x* on the size of the considered structure usually multiply by *x* the memory requirements. Nevertheless, some architectures like the Systolic Ring (described later) feature a complex routing system [6] allowing full interlayer connectivity and implying a non-linear law.

- Control unit: This part is the 'kernel' of the architecture. It manages the dynamical reconfiguration. Depending on the chosen remanence and also on deeper architecture details, the corresponding scalability may vary a lot.

- Interconnect: Fully systolic architectures with only neighbour-to-neighbour connections are the more scalable (linear increase). More flexible architectures usually feature a decreased scalability, due a non-constant maximal routing radius (longest path between two PEs, expressed in number of PEs).

Hence, the global operating density OD_{tot} is then:

$$OD \sim \frac{N_a}{A_{PEs} + A_{control} + A_{config\_mem} + A_{interconnect}}$$

4. THE SYSTOLIC RING

The Systolic Ring architecture features a highly optimized, DSP-like coarse grain reconfigurable block; following an original concept (figure 2). This component is configured by a microinstruction code. The configuration can either come from the configuration layer (FPGA-like mode, *global mode*) or from a local sequencer (*local mode*) also depicted in figure 7.

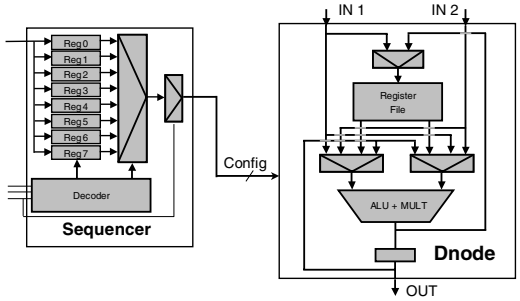

Figure 7. The Dnode architecture

A custom instruction set RISC processor (*configuration sequencer*) is also used in order to upload the microprograms into the local sequencers of the Dnodes set to *local mode*. It is also used to write the configuration into the configuration layer (*global mode*). In both modes the structure is dynamically reconfigurable, either using the global configuration sequencer or the Dnodes local sequencers.

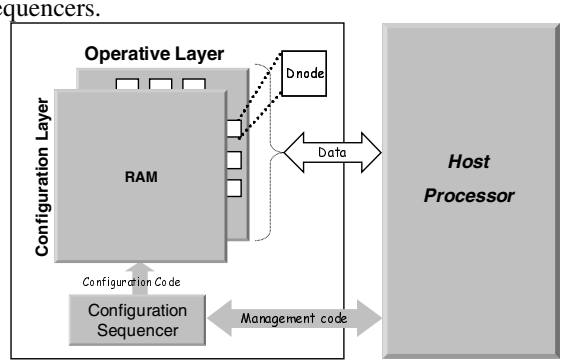

Figure 8. System overview

The specific structure of the operating layer is depicted on figure 9. The Ring topology allows an efficient implementation of pipelined datapath. The switch components establish a full connectivity between two layers, refer to [6] for complete description. The Systolic Ring also provides a feedback network which proves useful for recursive operations. It allows to feedback data to previous layers by the way of using feedback pipelines implemented from *each* switch in the structure. Each other switch in the architecture has a read access on *each* other switch's pipeline.

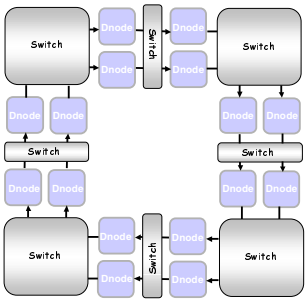

Figure 9. The operating layer

Figure 10 depicts the east switch's feedback pipeline. In addition a bus connecting all switches in the architecture and the global sequencer is available, mainly for conditional configuration: a data computed in the operating layer can be retrieved in the configuration sequencer for further analysis and thus different configuration evolution.

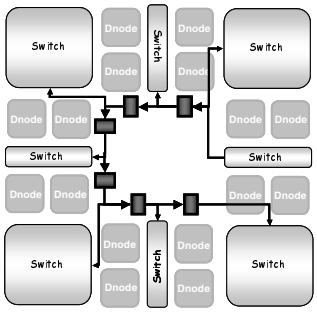

Figure 10. A switch's feedback pipeline

The program running on the global sequencer is able to modify the configuration of an entire Dnode layer (2 Dnodes on the 16 Dnodes Systolic Ring depicted on figure 9) each cycle. Up to 12.5% of the Dnodes can be reconfigured each cycle in the actual version, but this can be tailored, especially when C/N vary, C being the number of Dnodes per layer and N the number of layers.

An assembler/simulator environment has been developed. This environment also generates the object code running on the global sequencer and managing dynamically the configuration [6].

5. Remanence and scalability analysis

5.1 Remanence analysis

Considering the *global mode*, the remanence of the Systolic Ring (Figure 9) is R_{sring_static}=8, 8 cycles are indeed needed to reconfigure the whole structure, F_e being equal to F_c. As previously said, the Systolic Ring is customisable, thus the remanence can be tailored. This of course impacts the instruction size, and other parameters like memory bandwidth. This will be pointed out in the scalability section.

The *local mode* allows to change the configuration of each Dnode of the structure each cycle (assuming that all Dnodes are in local mode). However, 8 configuration cycles are needed to store a maximum length microprogram (one local sequencer register loaded per cycle, Figure 7), this microprogram being considered as a single Dnode configuration. In this case, a maximum of 64 cycles are needed, thus $R_{sring\_dynamic}$=64.

It must be pointed out that:

- A microprogram being considered as a single instruction, 8 instructions are needed to carry out a single data. Therefore, the amount of data is only characterized by R_{sring_static}.

- Despite in local mode all Dnodes can modify their configuration each cycle, from a system point of view, only $R_{sring\_dynamic}$ should to be taken into account. This mode is worthwhile only when the number of cycles of the considered process is at least 10 times greater than $R_{sring\_dynamic}$. The global mode is of great interest for data parallelism while the local mode features intermediate granularity data parallelism and potential data parallelism.

Table 1 gives remanence values for three different architecture described below:

- Texas Instruments TMS320C62: this one is a powerful VLIW processor featuring 8 processing units. It reaches 1600 MIPS (max power) when running at 300MHz. The remanence R_{C62} is equal to 1: it is able to reconfigure all its processing units each cycle.

- Xilinx Virtex XC2V2000 FPGA [4]: this one is partially reconfigurable, and requires 14.29 ms to be totally reconfigured at F_c=66 MHz. While F_e is application-dependant, the ration F_e/F_c is non constant. Results depicted in table 1 are given for F_e=100 MHz.

- Systolic Ring: a 16 Dnodes realisation, described above in section 4.

Table 1: Remanence comparisons

	TMS320 C62	Xilinx XC2V2000	Ring-8	
			Dynamic	Static
Number of op.(Na)	8 PEs	2688 CLBs	8 Dnodes	
Reconfigured op. / cycle	8	$2.8.10^{-3}$	0.25	2
Fe/Fc	1	1 (F_e=66MHz)	1	1
Remanence (R)	1	936540	64	8

As shown in table 1, the remanence of the Systolic Ring in full global mode (i.e. static) is 8, as to say, 8 cycles are required to fully reconfigure the structure. The Systolic Ring also provides a *hybrid mode*, allowing to set independently each Dnode in the structure in global or local mode. In this last case, the effective remanence is ranging from R_{sring_static} to $R_{sring\_dynamic}$. The most 'dynamically reconfigurable' architecture is however the VLIW processor. Hence, it use should be recommended for highly irregular applications implying instruction-level parallelism. The remanence however does not give the number of PEs that one can expect to have for a given silicon area : the scalability analysis address this problem.

5.2 Scalability analysis

As assumed in 3.2, the total area is approximated by the sum of the 4 constituting elements of our model.

Two different scaling techniques are to be considered:

Scaling technique 1: N/C tradeoff

N_a can be tailored between N (number of Dnodes per layer) and C (number of layers) according to the formula:

$$N_a = N.C$$

Increasing N will encourage parallelism level (either instruction or data) while increasing C will improve pipeline depth (i.e. computation parallelism).

Scaling technique 2: MIMD approach

It is also possible to increase N_a by the way of using multiple Systolic Rings witch will lead to a MIMD (Multiple Instructions Multiple Data) like solution. This technique provides a maximal scalability as the resulting silicon area will be proportional to the number N_a of PEs.

$$\left(\frac{N_a}{A}\right)_{MIMD} = \alpha = cte$$

In the following, only scalability issues related to technique 1 will be considered.

5.2.1 *Processing elements (i.e. Dnodes)*

Given a PE of core area A, the instantiation of N_a PEs occupies αA area units on the die, as to say this part is fully scalable, independently from the N/C ratio:

$$\left(\frac{N_a}{A}\right)_{PEs} = \alpha = cte$$

5.2.2 *Control unit*

The global sequencer (i.e. control unit) is a simple RISC processor featuring a specific instruction set. The 16 lower bits of the instruction format are dedicated to internal RISC management, whereas the upper ones are directly addressing a given Systolic Ring layer (configuration of N Dnodes and the corresponding switch). No additional logic is needed for decoding purposes as the Dnodes configuration and program (local or global mode selection) codes are directly aggregated with the switch configuration and layer selection in the upper bits of the instruction format. Figure 11 depicts the format of the instruction register used in the configuration sequencer.

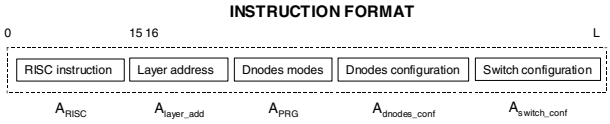

Figure 11. RISC instruction format

In the following, the area A_{part} corresponding to a given *part* of the instruction register will be considered proportional to the number of bits required for its coding, M_{part}.

- A_{RISC}. The size of the sequencer-related instruction is constant, thus, fully scalable.

$$A_{RISC} \sim M_{RISC} = 16$$

- $A_{layer\_address}$. $M_{layer\_address}$ bits being required for a C-layer addressing (2^M=C), and taking into account that C may not be a power of two, it comes quickly:

$$A_{layer\_address} \sim M_{layer\_address} = \left|1 + \log 2\,(C\text{-}1)\right|$$

-A_{PRG}. 2 bits are required to code the 4 run-modes. Hence, for N Dnodes, the required number of bits given above exhibit a maximal scalability:

$$A_{PRG} \sim M_{PRG} = 4.N$$

- A_{dnodes_conf}. Again, considering that 17 configuration bits are required for each Dnode, the resulting area is:

$$A_{Dnodes\_conf} \sim M_{Dnodes\_conf} = 17.N$$

- A_{switch_conf}. In order to provide a full inter-layers connectivity, let n be the number of inputs of the MUX and p the number of outputs: C(n,p) addresses combinations must be supported. The presence of a bus implies to be able to write the result of any Dnode output, plus an additional bit putting the bus driver in high impedance. The resulting number of bits required is:

$$A_{switch\_conf} \sim M_{switch\_conf} = \left|1+\mathrm{Log}_2(C_n^p\text{-}1)\right| + \left|\mathrm{Log}_2(N)+1\right|$$

The number of inputs is determined by the expression:

$$n = 2.N + (C\text{-}1).N + 1$$

The first term is related to number of Dnodes of the upper layer, while the second is related to the feedback network: C-1 feedback network are implemented, each one constituted by the aggregation of N Dnodes outputs. The number of outputs p is equal to N (number of Dnodes per layer).

5.2.3 *Configuration memory*

The use of a coarse grain technology drastically decreases the size of the configuration memory. In addition, the size of the PE-only configuration memory grow linearly with the number of PEs. Only the routing-relative configuration size grows non-linearly with respects to the number of processing elements, due to the facts that the Systolic Ring provides full interlayer connectivity. However, this can be tailored for big realizations by the way of defining reduced interlayer communication. Only the 'worst' case will be considered, as to say, providing full inter-layer interconnections. The size required for the storage of a (N,C) version of the Systolic Ring is:

$$A_{config} \sim M_{config} = C.(M_{PRG} + M_{Dnodes\_conf} + M_{switch\_conf})$$

5.2.4 *Synthesis*

Figure 12 et 13 show respectively the relative area for each part of the architecture and the evaluated core areas.

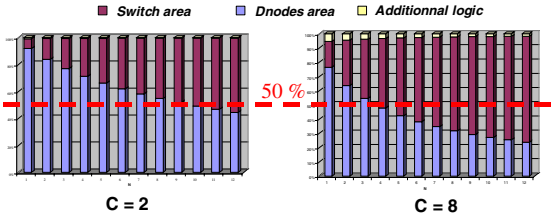

Figure 12. Relative area comparisons

It proves area costly to promote C (pipeline depth) with respects to N (i.e. intra-layer parallelism level). Moreover,

and independently from the C/N tradeoff, starting from a given number of operators the cost becomes prohibitive (Figure 12 and 13), due to the only poorly scalable part of the Systolic Ring: the interconnect. The exposed core areas have been calibrated thanks to a few place and routed implementations on a 0.35µm CMOS process (3 metal layers). Routing impact is estimated around 10 % in this case.

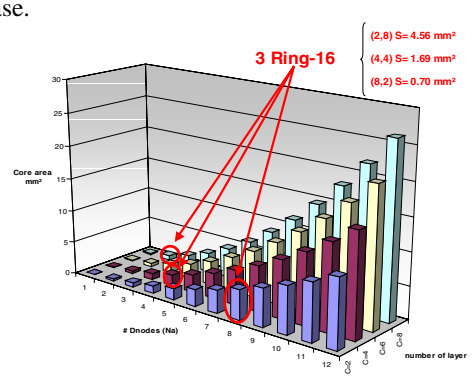

Figure 13. Relative area comparisons

The figure 14 depicts both operating density and remanence of the Systolic Ring architecture model. Three different C/N tradeoffs are plotted. As shown, this can help the designer choosing a version which matches the requirements: for instance, below a given number of operators, the processing power might be considered not sufficient. Increasing the number of operators might however lead in an unaffordable implementation cost (operating density too low) and/or an unacceptably high remanence. This helps to define an acceptable implementation region where the designer can then finely tune his architecture.

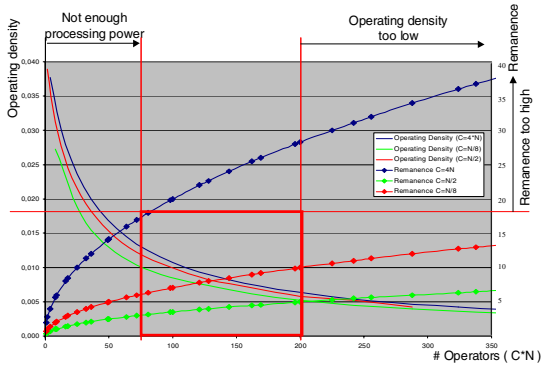

Figure 14. Operating Density (OD) and Remanence

6. CONCLUSION

We have in this paper presented a general model for digital signal processing dedicated architectures, either reconfigurable or Von Neumann-based. Two major issues are tackled: architecture characterization / classification and scalability. The former was addressed by the way of defining an additional metric: the remanence. This criterion helps the designer to choose between different architectures providing different instruction / data parallelism tradeoffs. The latter issue was addressed thanks to the definition of the operating density (OD) which characterizes the scalability of a given architecture. A dynamically reconfigurable architecture, The Systolic Ring was then used as a case study for both remanence and scalability analysis. These considerations helped to determine architecture features and also contributed to establish the limitations of an architecture considering a set of application-relative constraints (level of parallelism, area, etc.). While some basic analysis on power consumption was given in this article, future works takes place in a deeper analysis on crucial factors in a SoC design context (e.g. clock frequency, communication, memory requirements, etc.).

7. REFERENCES

[1] Stephen Brown and J. Rose, "Architecture of FPGAs and CPLDs: A Tutorial," IEEE Design and Test of Computers, Vol. 13, No. 2, pp. 42-57, 1996

[2] W. H. Mangione-Smith et al, "Seeking Solutions in Configurable Computing," IEEE Computer, pp. 38-43, December 1997

[3] R. Hartenstein, H. Grünbacher: The Roadmap to Reconfigurable computing Proc.FPL2000, Aug.27-30, 2000; LNCS,Springer-Verlag

[4] Xilinx, the Programmable Logic Data Book, 2000

[5] Why reconfigurable computing, Computer Structures Group http://xputers.informatik.uni-kl.de/

[6] G. Sassatelli, "Architectures reconfigurables dynamiquement pour les systèmes sur puce", Ph.D. thesis, Université Montpellier II, France, April 2002.

[7] D. Rizzo, O. Colavin, "A Video Compression Case Study on a Reconfigurable VLIW Architecture", Proc. DATE'02, Paris, 5-7 mars 2002, pp540-546.

[8] Sajjan G. Shiva, "Computer Design and Architecture", Marcel Dekker editor, ISBN 0 8247 0368 5, 2000.

[9] Arvind Robert, A. Iannucci, "A critique of multiprocessing von Neumann style", 10th International Symposium on Computer Architecture, pages 426-436, 1983.

[10] Didier Demigny, "Méthodes et architectures pour le TSI en temps reel" Traité IC2 – Série Traitement du signal et de l'image, ISBN : 2-7462-0327-8.

[11] André DeHon, "Comparing Computing Machines", Configurable Computing: Technology and Applications, Proc. SPIE 3526, 2-3 November 1998.