MAPPING FUTURE GENERATION MOBILE TELECOMMUNICATION APPLICATIONS ON A DYNAMICALLY RECONFIGURABLE ARCHITECTURE

Raphael David [†], *Daniel Chillet* [†], *Sebastien Pillement* [†]

Olivier Sentieys [†,‡]

[†] LASTI - University of Rennes I
6, rue de kerampont
F-22300 Lannion, France
name@enssat

[‡] INRIA / IRISA
Campus de Beaulieu
F-35042 Rennes cedex, France
sentieys@irisa.fr

ABSTRACT

In addition to the high performance requirements inherent to multimedia processings or to W-CDMA, future generation mobile telecommunications bring new constraints to the semiconductor design world. In fact, the traditional solutions based on the use of hardware devices (ASIC) or software ones (DSP) are unable to associate the flexibility and the high level of performance to the low energy consumption required by this application domain. In this paper, we study the efficiency of an architecture based on the use of the functional reconfiguration for such systems. Thanks to the implementation of key applications of UMTS, we will show that reconfigurable architectures can offer new compromises to associate high performances and low energy consumption in a flexible architecture and so, can be the solution to the set of problems associated with the future generation mobiles telecommunications systems.

1. INTRODUCTION

Traditionally, signal applications are implemented in hardware or in software. Hardware implementations are based on the use of dedicated piece of silicon which are optimized for one application (ASIC) or eventually one set of applications (ASIP). These solutions allow to support the processings in a very efficient way and the designer may optimize its design in area, time, power or even a combination of these three parameters. On the other hand, software implementations are not limited to few applications since they use a programmable processor which may support every kinds of processing. This flexibility is however obtained at the price of a large amount of energy and time waste since, to be generic, a programmable processor has to integrate a lot of resources that will, most of the time, not be used during the execution.

Between this two kinds of implementations, a third way has been proposed by FPGAs. This kind of architecture allows to optimize the circuit for one type of application but if the processing to implement has to be changed, the circuit may be reconfigured, in some milliseconds, to be fitted to these new computation patterns. This solution allows new compromises between performance, power consumption and flexibility of the system. However, even if it can be attractive for numerous applications, its flexibility (limited by the time of the reconfiguration) and its performances (limited by the extra-cost introduced for the genericity of the architecture) prohibits its used for high complexity tasks or for those that successively have to execute different calculation patterns (figure 1). Hence, the concept of reconfigurable architectures has evolved in order to allow some far more quick reconfigurations while offering a very high-level of performance and consuming very few energy [1].

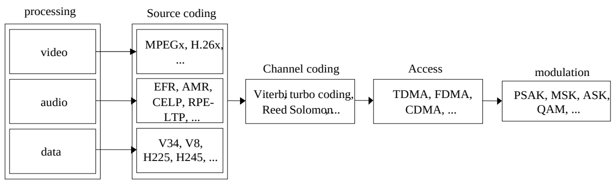

Fig. 1. Block diagram of a third-generation transmission system

The aim of this paper is to envisage the architecture reconfiguration at the functional level as a new possible solution for systems that need to associate flexibility, high performance and low energy consumption. In particular, we will describe the use of a dynamically reconfigurable architecture (DART) developed at the laboratory within the scope of next generation mobile telecommunications. This paper is organized as follow: In the next section we will present the third generation telecommunications and the architectural constraints associated with this application domain. The architecture DART will then be presented in the third section of the paper and in the fourth section we will study the implementation of some key applications of the UMTS on this architecture. We will finally evoke the development flow associated with DART before to conclude on the adequacy between this architecture and the application domain targeted.

2. THIRD GENERATION TELECOMMUNICATIONS : AN ARCHITECTURAL POINT OF VIEW

In addition to the high performance requirements inherent to multimedia processings or to W-CDMA and to the low-power constraints associated with their portability notion, a third generation mobile telecommunication system (figure 1) bring new constraints to the semiconductor design world [2]. In particular, the success of the UMTS will be linked to a greater flexibility of the standard than that of the GSM or the IS-95.

Thus, different types of processing will have to be supported by the multimedia terminals or base stations. For example, for speech coding, the signal has to be coded according to the CELP standard or to the GSM norm with an EFR coding. At the same time, a multimedia terminal will have to support the evolution of

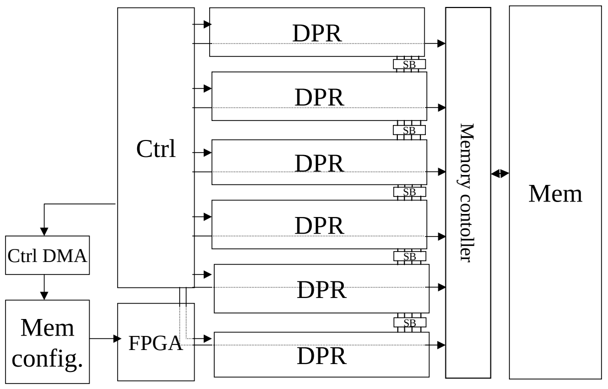

Fig. 2. Architecture of a DART's cluster

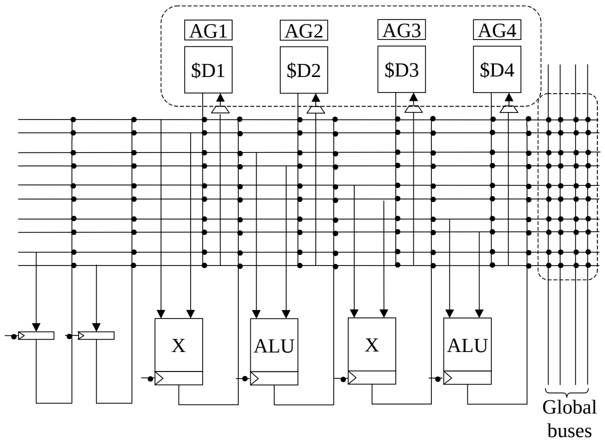

Fig. 3. Architecture of a DPR

the different standards and services, mentioned in figure 1, until the UMTS services are completely defined. This kind of flexibility is usually referred to as software, because of the way that these modifications will affect the architecture. In fact, these modifications will, in this case, adapt the program to be implemented.

The third-generation systems must have however another kind of flexibility which is far more problematic in such portable devices : That of the Hardware. In fact, multimedia terminals will have to ensure successively the execution of very different applications in terms of calculation patterns. Moreover, these applications will have to handle different data types. For example, an MPEG coder working on 8-bit data could be followed by a Viterbi coder working at the bit level. These processing modifications are then much more problematic since it will be necessary, in order to be efficient, to dynamically adapt the architecture to these changes.

Because of the lack of flexibility in ASICs and the low level of performance associated with the high power consumption of the DSPs, the reconfigurable architectures are more and more taken into consideration to answer the problems associated with the third generation mobile (3G) telecommunications. A reconfigurable architecture can be defined as an architecture whose resources (calculation, storage, ...) can be modified in order to be adapted to the processing to be implemented. The large design space of the reconfigurable architecture allows the designer to solve many of the problems previously mentioned [3]. In particular, the variety of processing granularities can be mapped onto the architecture which integrate different kinds of operator. Moreover, the problem of hardware flexibility could be solved with dynamic reconfiguration which allow the adaptation of the architecture at each calculation pattern change.

3. THE DART ARCHITECTURE

The constraints listed above led us to the definition of an architecture associating dynamic reconfiguration and low energy consumption [4]. In order to simplify the programming model and hence to allow an efficient exploitation of the numerous processing primitives of DART, this architecture has been organized into a hierarchy which concerns the processing as well as the interconnection and the control resources.

At the highest abstraction level of DART, a task controller has to distribute the different configurations to clusters (figure 2) under resource availability and urgency constraints. Those configurations are translated, at the cluster level, into a modification of the program sequenced by the cluster controller and by the loading of the data handled during the execution of the task. This program is composed of a succession of instructions to be sequenced to punctually reconfigure the two kinds of processing primitives integrated in each clusters: the FPGA core and the DPRs. By integrating two kinds of operator, DART is efficient for a large amount of applications which may work at the arithmetic level as well as the logic level. In fact, the FPGA core could be used for the fine grain applications such as the channel coding or the PN code generation. Thanks to its bit-level reconfiguration it would be optimized for every kind of processings that handle bit or small data. For arithmetic processings, at least one of the six DPRs would be used in each cluster. They are reconfigurable at the functional level and it is thus possible to reconfigure the interconnections between arithmetic operators. Hence it allows the optimization of the datapath for the calculation pattern.

In practical terms, these 6 DPRs are interconnected within the clusters via a segmented mesh network. They are constituted of 4 arithmetic operators, namely two 16-bits multipliers and two 40-bits ALUs, supporting SWP (Sub-Word Parallelism) processings and handling data stored in four local memories of 256 16-bits words. Moreover these memories, two registers are also integrated in each DPR. They are mainly dedicated to data-flow oriented applications where the different functional units are working on a same data-flow but on samples delayed from one iteration to the following. All these resources are connected via an entirely connected multibus network. Thanks to this network, every resource may communicate with every others in the DPR. Hence, the datapath may be optimized for every kind of calculation pattern.

The originality of this architecture comes from the two types of reconfiguration that are supported by the DPRs. They allow to be efficient for a large amount of applications by distinguishing regular processings where the configurations are used for long periods of time and irregular ones where they often have to be changed. Thus, the DPRs may be reconfigured in two ways:

- For regular processings such as loop kernels, the datapath mapped on the DPRs is optimized for the calculation pattern. In that case, we talk about hardware reconfiguration and a total flexibility of the DPRs is assured. This kind of reconfiguration requires between 4 and 9 cycles.

- In order to allow the reconfiguration of the DPRs in 1 cycle and so, to efficiently execute every kind of processings, the flexibility of the DPRs has to be limited to decrease the configuration data volume. For these processings, a software reconfiguration of the DPRs has been defined. In that case, the configuration only concerns the functionality of the operators and the memories. It has been decided to adopt a calculation pattern of *Read-Modify-Write* type, such as in conventional DSPs.

These two reconfiguration modes allow an appreciable decreasing of the energy consumption since the traditional energy waste associated to the instruction readings is drastically reduced. The instruction memory accesses will indeed be realized only when a reconfiguration will be necessary. Moreover, data memories accesses are also reduced thanks to the registers which allow to create some delay chains. The use of more classic technics such as the guarded clocks allows a final energy savings.

4. MAPPING AN APPLICATION ON DART

DART, rapidly described in the previous section, is able to deliver up to ten 16-bits GOPS/cluster or eighteen 8-bits GOPS/cluster, for an energy efficiency of 15 MOPS/mW, and this potential may be fully exploited thanks to its flexibility. The aim of this section is to present some implementations that have been done on this architecture. The applications presented below have been chosen as representative samples of the application domain.

Complex Despreading. Because of the complexity of W-CDMA, its implementation in a multimedia terminal has to be very efficient [5, 6]. In order to test DART for such application, a subset of the norm has been implemented: a finger of a rake receiver which realizes the complex despreading, represented by equations 1 and 2, with a spreading factor of 256. The data handled are issued from the receiver filter and are coded on 8bits [5].

$$I_{NB} = \sum_{j=0}^{SF-1} I_{WB}(j)*C_P(j+\tau_i)+Q_{WB}(j)*C_Q(j+\tau_i) \quad (1)$$

$$Q_{NB} = \sum_{j=0}^{SF-1} I_{WB}(j)*C_Q(j+\tau_i)+Q_{WB}(j)*C_P(j+\tau_i) \quad (2)$$

DCT 2-D. To illustrate video processings we also have implemented a Discrete Cosine Transform on 8x8 pixel macroblocs, since this kind of algorithm is nearly systematic in video compression standards like MPEGx or H.26x[7]. The 2 loop kernels of this algorithm are based on a *Multiplication-ACcumulation*. They are given by equations 3 and 4, respectively for the raws and the columns.

$$value[y] = \sum_{j=0}^{7} coe[y][x]*bloc[j+x*8] \quad (3)$$

$$value[y] = \sum_{j=0}^{7} coe[y][x]*bloc[j+x] \quad (4)$$

Autocorrelation. The multimedia terminals of next generation will still be used for the transmission of speech between two distant persons and so need very effective speech coder. To illustrate this kind of processing, we have implemented an autocorrelation on a signal of 240 samples preamble of Levinson-Durbin algorithm which permits the adaptation of the prediction filter coefficients in speech coders such as the EFR or the AMR[8]. The mathematical description of this processing is given by equation 5.

$$r(p) = \sum_{n=0}^{239} x(n)x(n-p) \qquad \text{for p} \in [0..239] \quad (5)$$

The table1 resumes the implementation results. The first implementation of the complex despreading uses two DPRs, one to calculate I_{NB} and the other for Q_{NB}. For this implementation, we suppose that the synchronization has been done before the despreading [5]. Since we are handling 8-bit data, we are able to treat simultaneously two data thanks to the SWP capabilities of the operators. The operating frequency for this implementation is half that of the chip (3.84 MHz/2= 1.92 MHz). Given that the clusters can run at 130 MHz, we are far more over the required level of performance. Moreover, the clusters of DART are autonomous on the operating frequency and the operating voltage. It will thus be possible to reduce this two parameters in order to minimize the energy consumption. It has to be noticed that even if the main critical processings of W-CDMA is a FIR filter, we do not show the results since the regularity of this algorithm will be very well exploited on DART.

The other implementation of the complex despreading exploits only one DPR and the I_{NB} and Q_{NB} coefficients are calculated by using one multiplication to evaluate $I_{WB}(j)*C_P(j+\tau_i)$ (respectively $I_{WB}(j)*C_Q(j+\tau_i)$) then (at the next cycle) $Q_{WB}(j)*C_Q(j+\tau_i)$ (respectively $Q_{WB}(j)*C_P(j+\tau_i)$) and one addition to sum this two results. The cluster can then run at f_c (3.84 MHz) since two data are red every two cycles (2*$\frac{f_c}{2}$), which is still allowing an operating voltage decreasing. Moreover, since all the data reads come from the same DPR, a better data sharing can be done thanks to the registers. This implementation allows to implement 6 fingers of a Rake receiver in each cluster and in a such configuration, the coefficients I_{WB} and Q_{WB} can be shared between all the DPRs thanks to the segmented mesh network.

The table 1 shows the interest of SWP for processings like the DCT (issued from MPEG-4 core profile @ level 2: CIF, 60Hz, 2Mbps) which is working on 8-bit data since a 8x8 pixels bloc can be transformed in 72 cycles by using 4 DPRs. The latest results presented in table 1 concern the autocorellation and show the efficiency of DART in two other ways. The first one is the small number of instructions that are needed to conclude this processing. Indeed, only 5 instructions are needed to specify the behaviour of the cluster during the time of the task. Once these 5 configuration instructions have been red, the only control that is necessary to conclude the execution is the sequencing of the data addresses which are manipulated on the DPRs. Comparatively, on a conventional DSP processor more than 57,000 readings from the instruction memory have to be done. Given the cost in energy of a memory access, the gain in power consumption is therefore very important. The second point that can be put forward is the large amount of data sharing allowed by DART. In fact, DART enables to divide by 12 the number of accesses to the data memory and so the energy waste due to these accesses (which is typically very high). This data sharing is made easier by the high degree of flexibility of the interconnection network and the use of the registers in data-flow oriented applications.

Application	DPR	operations	cycles	frequency	inst. read	data read
Complex Despreading_v1 (SF=256)	2	2048	258	1.92 MHz	4	1024
Complex Despreading_v2 (SF=256)	1	2048	515	3.84 MHz	4	1024
DCT 2-D	4	2048	72	250kHz	5	256
Autocorrelation (240 samples)	6	57600	1520	8 kHz	5	2560

Table 1. Implementation results on DART

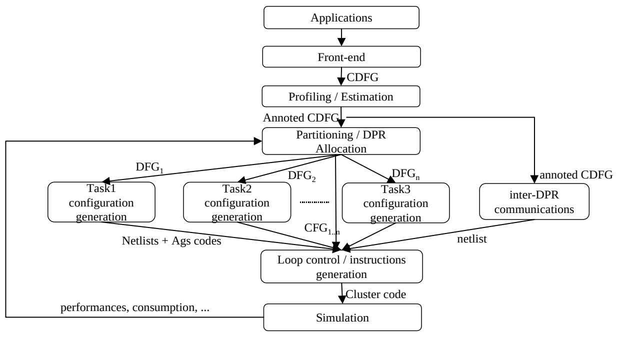
Applications
Front-end
CDFG
Profiling / Estimation
Annoted CDFG
Partitioning / DPR Allocation
DFG$_1$
DFG$_2$
DFG$_n$
annoted CDFG
Task1 configuration generation
Task2 configuration generation
Task3 configuration generation
inter-DPR communications
CFG$_{1..n}$
Netlists + Ags codes
netlist
Loop control / instructions generation
Cluster code
performances, consumption, ...
Simulation

Fig. 4. Development flow of an application for DART

5. FROM A HIGH LEVEL DESCRIPTION TO THE BINARY EXECUTABLE CODE

Even if the efficiency of DART can be practically proved, a tool support have to be provided in order to quickly deliver a code optimized under performances and power consumption constraints. Hence we develop a methodology, illustrated on the figure 4, which is built around the modules described below.

Based on the Control-Data-Flow-Graph (CDFG) issued from the semantic and syntactic analysis of the application description, a first module is in charge of estimating the complexity of the different parts of the algorithm. The annotated CDFG resulting from this profiling step is used in a partition module which extracts some sub-tasks liable to run concurrently. A number of DPRs are then allocated to the sub-tasks according to the available resources on the cluster and their parallelism degree. These sub-tasks must then be transformed into a succession of configurations. These configurations may be of Software or Hardware type and they are distinguished by detecting the loop in the CDFGs. In fact, when a loop is detected, its kernel is synthesized on the DPRs under resource constraints. All the code between two loops is then transformed into software configurations after a compilation step, similar to those that are done for DSPs.

In the same time, the addresses of the data to be handled by the functional units in the DPRs are generated. The module in charge of this work extracts in the CDFG the data accesses in order to generate the programs that will be sequenced by the address generators associated to the DPR memories. Moreover the configuration of the DPRs, those of the interconnection between these DPRs have also to be generated. All these configurations are then synchronized thanks in order to generate the program to be run on the cluster. This code will finally be simulated in order to verify its functionality and to extract some informations about the performances and the energy consumption of the implementation.

6. CONCLUSION

In this paper we have shown the interest of the functional reconfiguration of DART for next generation mobile telecommunication applications. Thanks to its flexibility, this architecture is able to implement compute intensive applications as well as irregular ones in an optimized way and thanks to the integration of both arithmetic and logic operators it will support a large variety of computation granularity. Moreover this flexibility, DART is able to deliver up to ten 16-bits GOPS/cluster or eighteen 8-bits GOPS/cluster while consuming less than 0.07 mW/MOPS. Hence, its use in a mobile telecommunication system is only depending on the ability of the development tools to quickly provide an executable code optimized under performances, energy consumption or even a combination of both constraints. It has to be noticed that the results presented in this paper do not exploit the FPGA since the applications have only been implemented on the DPRs. Hence, to improve the performances of DART, the next step in our study is to develop an FPGA architecture which will allow the association of high performances and low energy consumption.

Acknowledgments

This work is part of a project, associating the University of Brest and ST Microelectronics, funded by the industry and research French ministry. We would like to thanks B. Pottier (University of Brest), T. Ben-Ismal and O. Colavin (ST Microelectronics) for their contributions to this project.

7. REFERENCES

[1] J. Rabaey, “Reconfigurable Processing : The Solution To Low-Power Programmable DSP,” in *ICASSP*, Apr. 1997.

[2] P. Pirsch, “Architectures For Multimedia Signal Processing,” in *SIPS*, 1999.

[3] R. Hartenstein, “A Decade of Reconfigurable Computing : A Visionary retrospective,” in *DATE*, 2001.

[4] R. David, D. Chillet, S. Pillement, and O. Sentieys, “A Dynamically Reconfigurable Architecture for Low-Power Multimedia Terminals,” in *VLSI-SOC*, Dec. 2001.

[5] T. Ojanperä and R. Prasard, *Wideband CDMA For Third Generation Mobile Communication*, Hartek Publishers, 1998.

[6] Erik Dahlman, Björn Gudmundson, Mats Nilson, Joan Sköld, “UMTS/IMT-2000 Based on WideBand CDMA,” *IEEE Communications Magazine*, pp. 70–80, September 1998.

[7] L. Hanzo and al., “Interactive cellular and cordless video telephony : State-of-the-art system design principles and expected performance,” *Proccedings of the IEEE*, 2000.

[8] T. Amada et al., “CELP speech coding based on an adaptative pulse position codebook,” in *ICASSP*, 1999.