Optimal Determination of Common Operators for Multi-Standards Software-Defined Radio

Christophe Moy, Jacques Palicot, Virgilio Rodriguez, Denis Giri

Optimal Determination of Common Operators for Multi-Standard Software-Defined Radio

Christophe Moy, Jacques Palicot, Virgilio Rodriguez, Denis Giri
Supelec/SCEE research team – IETR/AC group CNRS 6164
Campus de Rennes, Av. de la Boulais, BP 81127
35511 Cesson-Sévigné, France
Tel. : +33.2.99.84.45.34
Fax : +33.2.99.84.45.99

christophe.moy@supelec.fr, jacques.palicot@supelec.fr, virgilio.rodriguez@supelec.fr, giri_den@rennes.supelec.fr

***Abstract*—This study proposes a new methodology for selecting the *common operators* for Software Defined Radios. It aims at optimizing resources use in the design of multi-standard SDR systems. A multi-standard SDR or reconfigurable radio system consists of software components that execute specific processing steps in a continuous flow. Their behavior can be changed by a reconfiguration procedure. In a *common operator* design, this change only requires an adjustment of certain parameters, which minimizes the reconfiguration overhead at run-time. In order to optimize the choice of those operators, the interrelationships among the various components of the system are represented in an oriented graph. The associated metrics and the cost function investigated to solve the optimization problem are presented. An illustration is given on an example , which even if not fully realistic, is sufficiently representative in its structure and orders of magnitude to provide useful insights. The optimization problem is solved by an heuristic based on simulated annealing and validated by comparison with an exhaustive search.**

***Index Terms*—Common operators, optimization, Software-Defined Radio, graph theory, heuristic**

I. Introduction

The design of software-defined radio (SDR) systems is very challenging [1]. The fields of research that aim at solving or at least improving the methods and the technologies implicated in the SDR area are numerous. Most of them are not specific to the design of radio systems. Challenges are the same for most of the embedded real-time electronics applications. Both software (SW) [2] and hardware (HW) [3] sides are concerned. Nevertheless, the SDR community is particularly driving the activities focusing on reconfiguration and reconfigurability [4] in an heterogeneous context [5]. Far from defining new high-level tools and methodologies, a very promising approach consists in designing radio systems entities in a way that permits to take advantage of the programmable or at least reconfigurable capabilities of the underlying HW of SDR systems. The parameterization approach [6] in particular aims at designing multi-standard systems made of entities (typically functions) whose operation can be modified by a simple parameter adjustment. Reference [7] proposes to extend this approach to lower level entities called *common operators*. We tackle in this paper the issue of the selection of these operators to build a multi-standard SDR system.

We opt here for a structural description and exploration of an SDR system intended to support several standards. This exploration is intended to reveal the designer the best ways of implementing the processing elements of a communication standard, while taking into account the constraints introduced by the other communication standards that must also be supported by the system [8]. Our approach, and other techniques mentioned below, can help the designer identify the processing elements that should be implemented to build an optimized multi-standard SDR system.

The paper is organized as follows. After the present section, we discuss the *common operators* approach to SDR design. Afterwards, we build the mathematical model. This includes the drawing of a graph that represents design alternatives, the consideration of possible performance "metrics", and the specification of a cost function. In the subsequent section we discuss the optimization procedure, whose results are discussed in the following section. A conclusion/discussion section ends this paper.

II. Common Operators Approach

This study's main purposes are to present a mathematical model for the design of multi-standard Software Defined Radios (SDR) and to solve the optimization problem associated with it. It is based on a *common operators* approach. It involves identifying an optimal level of granularity for operations (simpler than a self-contained module implementing a major communication task, but more complex than primitive operators such as AND, OR, adder, etc.), in order to support several communication standards with a single architecture.

The main principle relies on the use of *common operators* that can each match several processing contexts by a simple parameter adjustment. The *common operators* approach can

greatly increase the efficiency of Software Defined Radio, both in terms of manufacturing costs, and of the speed of reconfiguration during operation. This approach can reduce manufacturing cost, through the re-use throughout the system of common structures, which can be either reconfigurable hardware or reprogrammable software. It can also reduce reconfiguration time, because each operator can change from one processing mode to another, via a simple change of parameters.

The advantages of the *common operators* approach have already been discussed in [7]. But reference [7] does not demonstrate such advantages. We therefore describe in this paper a method that allows us to identify the best selection of known *common operators* for the design of reconfigurable radio system.

The identification of new *common operators* useful to SDR design is, in its own right, an active area of research. Researchers are proceeding along several directions. For instance, [9] shows that many important tasks of a communication receiver can be implemented through the fast Fourier transform (FFT). In turn, the FFT can be implemented with butterflies. Thus, the butterfly can serve as a common operator for several FFT implementations of different orders. With this in mind, some researchers are seeking frequency-domain implementations for different families of algorithms. For example, [10] studies the frequency domain implementation of Reed-Solomon channel decoding.

III. A MATHEMATICAL MODEL FOR THE DESIGN OF MULTI-STANDARD SDR

The map of the SW components of a reconfigurable radio can be represented in a graph with several layers depending on the granularity of the considered processing elements. Our methodology aims at selecting the most appropriate level of granularity. A great advantage is that at certain levels of granularity, the operators can be several times re-used inside and between standards, for an optimized design of the resulting SDR system.

A. Graph Model

From here on, we will view a multi-standard reconfigurable system as a hypergraph of elementary functional modules. Each of these modules can either be directly implemented in the system, or can represent a functionality obtained by invoking lower-level modules. We need to use a hypergraph instead of a simple graph in order to introduce two different types of dependencies between the nodes, as illustrated in Fig. 1.

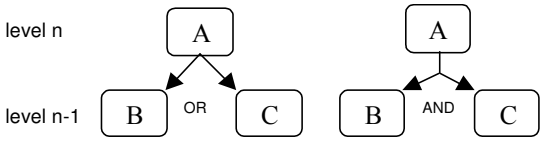


Fig. 1 OR and AND hyperarcs

Nodes dependencies occur between nodes of different levels. A node of a higher level, called a *parent node*, may have dependencies with nodes of underlying levels, called *descendant nodes*. Descendant nodes may not all be at the same level. Nevertheless, they all are at a lower level than their parent(s). An OR arc (direct arrow) means that only one of the descendant nodes is necessary to implement the parent node. An AND hyperarc ("inverted Y" connection) means that all descendant nodes are needed to implement the parent node. Note that in some case, a parent node may have both AND and OR dependencies with its descendants. All the destinations of a AND hyperarc coming from the node must be selected, or at least one in the case of a OR. Extending this process, a complete SDR communication system can be implemented by selecting the adequate nodes. Note that they concretely can be either hardwired (ASIC), or programmed in a processor (DSP, GPP), or a FPGA.

The roots of this graph, at the coarsest grain (highest level), will be called "standards"; since they will most likely represent communication standards that the reconfigurable system needs to support. As shown on Fig. 2, standards S1, S2 and S3 do not have parent nodes.

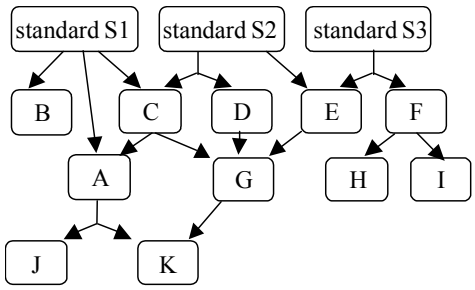


Fig. 2 Global structure of the graph for a tri-standards SDR system

By construction, the resulting graph is "acyclic", which makes the problem easier to solve. Specifically, it allows us to decide upon a topological sorting, without any ambiguity. In other words, a list of nodes in which the descendants are always after their parents can be found.

B. Cost parameters

As our optimization problem balances between economy and computing efficiency, two key parameters enter our cost evaluation:

- the cost of the module capable of computing a function, called *monetary cost,* which is paid only once during the useful life of the radio,
- the computing time required to perform a particular function, called *computational cost,* incurred every time a component is employed.

In some SDR architectures, the *monetary cost* can be represented by (is proportional to) the number of logic units necessitated by an FPGA implementation. This allows several standards to use the same components and share their cost, which is exactly what the concept of *common operator* is about.

The *computational cost* parameter is a bit tricky. Some modules (nodes) will be implemented by invoking several descendant nodes, each a certain number of times (where we may use an average number if this is not a constant). A multiplicative factor is needed to represent the number of

times a descendant module is called. Some communications between components may be also needed, including an additive *communication cost*. Thus, Below, we do not consider and neglect the *communication cost*.

C. Cost function

The cost function must account for both of the cost parameters we just discussed: *monetary cost* and *computational cost*. We settle for the following cost function:

$$\widetilde{I}.\sum_i c_i.S_i + \sum_k I_k.V_k((S_n)_{n\in N}) \quad (1)$$

- $S_i \in \{0,1\}$ tells if the node i is present in the system.
- c_i is the *monetary cost* of the component.
- $\sum_i c_i.S_i$ is the cost of all the components present in the SDR communication system.
- $V_k((S_n)_{n\in N})$ represents the *computational cost* of the standard k, $(S_n)_{n\in N}$.
- $\widetilde{I}$ and I_k are respectively the weights given to the total *monetary cost* of the system (sum of the *monetary costs* of each node implemented in the design) and to the *computational cost* of executing standard k once.
- I_k weights the cost of a communication standard in function of its rate of activation compared to the other standards supported by the SDR system.

Unfortunately, while this cost function seems very simple, the problem is still very complex, and can be shown to be NP-hard.

IV. Optimization

A. Validation methods

A designer wishing to build a multi-standard SDR system will specify the set of standards to be supported. The solution for an optimal design consists in minimizing:

$$C_{SDR} = \min_{bool((S_n)_{n\in N})}\left(\widetilde{I}.\sum_i c_i.S_i + \sum_k I_k.V_k((S_n)_{n\in N})\right) \quad (2)$$

The graph can be translated in a boolean formula with respect to AND and OR arcs. The constraint *bool* $((S_n)_{n\in N})$ therefore checks that all the standards are implemented in the SDR system.

Having considered the computing time and the accuracy of the results given by several optimization methods, we will discuss in detail two methods: the one that yields the exact solution and another that gives an approximate solution in less computing time. The *exhaustive search* method, which computes the values of all the feasible solutions and compares them all, gives the true optimal solution every time but it is very slow. Since this problem is NP complex, the *exhaustive search* becomes impractical when the number of nodes exceeds certain value (which depends on the capabilities of the computing resources available to perform the optimization). It will however be used on simple cases, to test the accuracy of the results given by the second procedure.

To obtain an approximate solution we chose the *simulated annealing* method [11], which provides the optimal solution with a good probability at relatively low computing time cost.

B. Greedy heuristics

A greedy algorithm is used to obtain a good starting point for the *simulated annealing* method; greedy meaning an algorithm that only takes into account the next step, and never goes back on its decision. Several sorting functions, all using a *factorization criterion* and the cost of the components of each node, have been considered as greedy algorithms. The selected sorting function evaluates the importance of a specific node and grades it. The aim is to order the nodes in relation to each other. For each sort function, graphs were found for which the greedy algorithm gave a solution as far from the optimum as we wanted. For this reason, we use the greedy heuristic to find a starting point for another algorithm, not a solution to the problem.

The *factorization criterion* mentioned above, noted *F(n)*, can be calculated using the recursive formula:

$$F(n) = \sum_{k\in parents(n)} \frac{F(k)}{N(k)}, \quad (3)$$

where *N(k)* is the number of arcs coming out of the node k and $k \in parents(n)$ if there exists an arc going from *k* to *n*. The resulting function *F(n)* is proportional to the number of occurrences of node *k* in the boolean equation of the graph mentioned in IV.A.

In the developed algorithm, the two greedy heuristics involved can be completely specified by giving their sorting functions:

$$F(n) \quad (4)$$

$$\frac{F(n)}{C(n)} \quad (5)$$

C(n) here refers to the *monetary cost* of the node n. *F(n)* aims at finding the nodes with the more occurrences. Equation (5) takes also into consideration the *monetary cost* of the most occurring nodes, so that the cheapest will be taken first.

V. Results

A. Graph

To test the simulated annealing algorithm, we use the graph from [7], now reproduced in Fig. 3, to which we added the necessary cost parameters and specified the hyperarcs. It is not in the purpose of this graph to give all the possible implementation alternatives for each of the considered processing elements. The graph is small enough (less than 10 useful nodes) that *exhaustive search* can be applied to test the accuracy of the approximated method. Dotted lines of Fig. 3 show the decision made by the algorithm: select the *FFT* operator.

Fig. 3 shows a sub-graph corresponding to the decomposition of several processing elements (*equalization*, *multi-channel*, *OFDM*) that could be part of a SDR multi-standard system. Root nodes at the top (standards) are

consequently not represented, but would be at a higher level not shown here. The *equalization* operation is a signal processing entity that compensates for the multipath impairment induced by the propagation channel. It can be either implemented through a finite-impulse response filter (*FIR*) or through the *FFT* operator. The implementation of the *equalization* in the frequency domain is indeed of particular interest for channels having long impulse responses. The number of taps of the FIR filter (time-domain) becomes so high that the cost of the *FFT* and IFFT (inverse FFT) is compensated by the simplification of the frequency-domain operation [12]. Note that IFFT is equivalent to *FFT* in terms of computation complexity and butterfly structure, so that we here can set at '2' the multiplication factor between *equalization* and *FFT*, as well as between *FIR* and *FFT*. *Multi-channel* stands for a demodulation technique that aims at considering a plurality of frequency channels in the demodulation process, instead of demodulating channels one by one. This can be accomplished by repeating the same process on each channel (*channel per channel*), or by proceeding in parallel, working on all the channels at once (*filter bank*). *OFDM* here shows that at least an *FFT* is needed to execute the demodulation. We point out that both *equalization* and *OFDM* could employ the *FFT* operator, even if it is required for *OFDM*, and optional for FIR-based *equalization*.

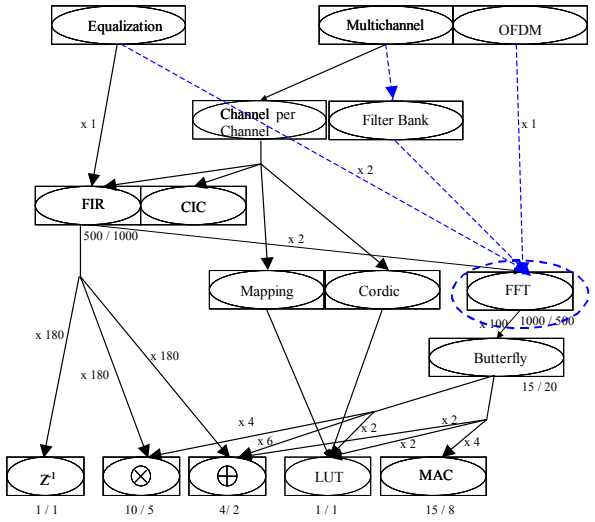


Fig. 3 Hypergraph example showing the different ways of enabling a SDR system to implement *equalization*, *multi-channel* and *OFDM* processing.

The numerical values under each node represent the *monetary cost* (MC) / *computational cost* (CC). The arcs are tagged with a number of calls (NoC) figure. These are not absolute values representing a precise physical meaning in this example. The values have to be considered relatively, giving some kind of order of magnitude. For clarity and simplification purposes, un-necessary costs and NoC have not been represented.

B. Metrics

The metrics are given with the following hypothesis. Data are supposed to be processed by packets, whose size we do not need to consider because it would not affect the cost of a node relative to the cost of the others. When a node is needed several times by a higher level module, it is called several times, and not physically duplicated. Accordingly, the multiple calls affect the *computational cost*, but not the *monetary cost*. As an example, let us consider the implementation of an *OFDM* processing element using different layers of the graph, as illustrated in Tab. 1.

	nodes	**MC**	***NoC***	**CC**	***cost of the OFDM design***
#1	*FFT*	1000	1	500	
			cost of alternative #1		1500
#2	butterfly	15	100	20	
			cost of alternative #2		2015
#3	LUT	1	2	1	
	x	10	4	5	
	+	4	6	2	
			cost of alternative #3		3415
#4	LUT	1	2	1	
	MAC	15	4	8	
	+	4	2	2	
			cost of alternative #4		3820

Tab. 1 – *OFDM* cost using different implementation alternatives

The detail explanation of the cost alternative #3 for the design of the *OFDM* is the following. The selected operators are *LUT*, *x* (multiplier) and + (adder). Two *LUT*, four *x* and six + are necessary to build a *butterfly*. A *butterfly* is called 100 times to make the *FFT*, which is the *OFDM* first operation at the receiver side. Consequently, the *monetary cost* of the operators is involved once for each of them. Their *computational cost* is multiplied by the number of calls done at the operator to make run the global processing of the system, here 100 for the *FFT* with one occurrence to run the *OFDM*. This gives the result:

$$(2x1+4x5+6x2)x100+(1+10+4) = 3415 \quad (6)$$

$$\text{cost\#3} = \left(\sum_{i=0}^{2} CC_i \times NoC_i \right) \times NoC_{i+1} + \sum_{i=0}^{2} MC_i \quad (7)$$

If a *OFDM* design is directly made with a *FFT* whose design is dedicated (and optimized) to the *OFDM* context under study, the resulting cost is of course lower (1500 for alternative #1). By construction of the graph and cost assignment, it is conformed in Tab. 1 that it is less expensive to use an integrated design of an algorithm than an exploded version.

C. Results

We just focus in this paragraph on the choice between the several design alternatives for the *equalizer* in the context of a multi-standard system already including a *OFDM* communication mode. If we consider independently the implementation of the *equalizer*, it is less expensive to implement it through the *FIR* (1500 = 1000x1 + 500) than through the *FFT* (2000 = 500x2 + 1000) as explained by equation (7). But in a combined design involving both *OFDM* and *equalizer* modules as shown in Tab. 2, the overall cost of the design is only 2500 if the *equalizer* uses

the *FFT*, versus 3000 if the equalizer uses the FIR. This is because the *monetary cost* of the *FFT* counts only once in the calculation of the overall cost of *OFDM* plus *equalizer*, saving 1000.

equalizer design alternative	equalizer MC	equalizer CC	*OFMD = FFT CC+MC*	overall cost
FIR	500	1000	1500	3000
FFT	0 (~~1000~~)	1000	1500	2500

Tab. 2 – Combined *OFDM* and *equalizer* costs depending on the *equalizer* design alternative *FFT* or *FIR*.

We can see that it in this example that the common design is 17% less expensive than the two separate designs, each independently optimized on its side.

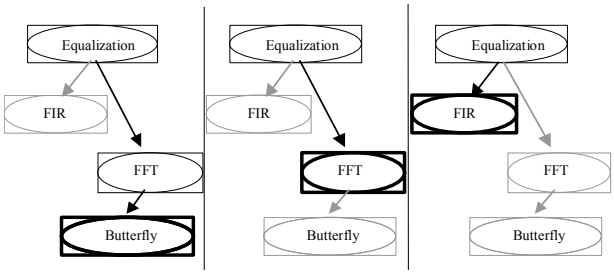
The simulated annealing algorithm developed for this study is capable of generalizing this approach to the global design of a multi-standard SDR. It will be the scope of a future paper to describe further this algorithm and its performance.

D. Interpretation

Numerical values can lead to many kinds of interpretation. As previously stated we do not consider that the figures we express here are definitive and exactly correspond to a justifiable concrete reality. These are only chosen for illustration purposes and help to understand which should be the best metrics to take into consideration. Nevertheless, orders of magnitude and comparative weights have been studied and respected. The simulated annealing algorithm developed for the optimization of the graph concludes that the *FFT* can indeed be a *common operator*.

In fact, from this algorithm point of view, the only thing that differentiates the cost of the two designs is the importance given to *computational cost* over *monetary cost*. Generalizing this to a global graph with standards at the highest level, the following can be concluded. If $\tilde{I}$ is kept high (or I_k low), the heuristic favors the *monetary cost* in equation (2). The choice consequently privileges low cost nodes and the bottom of the graph. This has the drawback to speed-down the system computation but to factorize processing resources at its maximum. This is a processor-centric approach. The theoretical limit on this side is to use the most basic operator that can be found. Even lower operators than the arithmetic and logical unit (ALU) elements of processors, this could go down to the transistors. Decreasing $\tilde{I}$ (or increasing I_k) makes the heuristic climb in the graph. When $\tilde{I}$ is very low, the nodes of the highest level are selected. The theoretical limit is the use of the root nodes themselves, which would consist in designing the standards themselves as a unique element. This is the "Velcro" approach of SDR. It would optimize the execution speed of each standard, but would completely avoid any re-usability between standards because of a lack of modularity.

Focusing in Fig. 4 on the choice between *FFT* and *butterfly* in the context of a multi-standard system already including *OFDM*, the three situations (a), (b) and (c) maybe met depending on the relative value assigned to the *monetary cost* and the *computational cost*.



(a) - $\tilde{I}$ high (b) - $\tilde{I}$ medium (c) - $\tilde{I}$ low

Fig. 4 Heuristic exploration alternatives in function of $\tilde{I}$, which sets the relative weights of *monetary cost* and *computational cost*.

Case (a) with $\tilde{I}$ high, selects the lowest level of node in this graph (*Butterfly*) as its *monetary cost* is much lower than any other alternative. In (b), some trade-off between the *monetary cost* and the *computational cost* makes the heuristic chose the *FFT* node, at a higher step than before. If $\tilde{I}$ gets even lower, as in (c), the *monetary cost* becomes so low compared to the *computational cost* that it brings no significant savings to re-use the *FFT* (already needed for the *OFDM*), and consequently other functions, as *FIR,* are selected.

This is the intermediate level we are looking for, since it permits a factorization of operators of medium level, combining both modularity and computation efficiency. This requires a trade-off between the *computational cost* and the *monetary cost*. This is the essence of the *common operator* approach to find a granularity at a higher level than the basic operators but at a lower level than a function.

VI. CONCLUSION

We have presented an approach and a procedure that can lead a designer of a multi-standard SDR system to an optimal architecture that balances cost and performance. The key is the drawing of a hypergraph which represents alternatives available to the designer. Building the graph is itself a working subject. For example, researchers are trying to identify new operators that may be common to several communication "blocks" within a given standard, or across several standards. The identification of any such operator will lead to a modification of the graph. Likewise, some researchers seek frequency-domain algorithms that provide functionality typically provided by time-domain procedures. With such algorithms, the number of communication functional blocks that can be implemented via the FFT operator grows larger, and new "arcs" pointing to the FFT operator can be added to the graph. Also, the graph may need to be modified as a result of the evolution of the communication standards.

Relevant issues have been ignored in the present work. The list includes (i) hard latency constraints imposed by the communication applications, (ii) the time needed to re-configure the SDR while switching from a standard to another, and (iii) contention among high level modules for

the service of the same lower-level module at the same time, in particular if the SDR system needs to support communication over several standards at the same time. These and other important issues will be addressed in the future.

ACKNOWLEDGMENT

This work has been partially funded by the "Région Bretagne".

REFERENCES

[1] W. H.W. Tuttlebee, "Software-Defined Radio: Facets of a Developing Technology", *IEEE Personal Communications Magazine*, Vol. 6, No. 2, Apr. 1999.

[2] D. L. Tennenhouse, V. G. Bose, "The SpectrumWare Approach to Wireless Signal Processing", *Wireless Networks*, vol 2, pp. 1-12, 1996.

[3] Jean Belzile, Steve Bernier, Manuel Uhm, "Software Radio Modems Enter the SoC Era", COTS Journal, 2005.

[4] Apostolos Kountouris, Christophe Moy, L. Rambaud, "Re-configurability: a Key Property in Software Radio Systems", *1st Karlsruhe Workshop on Software Radios*, Mar. 2000.

[5] Apostolos Kountouris, Christophe Moy, "Reconfiguration in Software Radio Systems", *2nd Karlsruhe Workshop on Software Radios*, Mar. 2002.

[6] Arnd-Ragnar Rhiemeier, "Benefits and Limits of Parameterized Channel Coding for Software Radio", *2nd Karlsruhe Workshop on Software Radios, Germany*, March 2002.

[7] J. Palicot, C. Roland, Y. Louet, A. Al Ghouwayel, "A New Parameterization Technique for Reconfigurable Radio : the Common Operator Approach", submitted to Annales des Télécommunications.

[8] Denis Giri, "Optimisation du graphe représentant le fonctionnement d'un terminal radio logicielle", training period report of Polytechnique made at Supélec, in French, July 2005.

[9] J. Palicot, C. Roland, "FFT: A Basic Function for a Reconfigurable Receiver", *ICT'03 Conference*, Thaiti, France, 2003.

[10] Ali Al Ghouwayel, Yves Louët and Jacques Palicot, "A Reconfigurable Butterfly Architecture for Fourier and Fermat Transforms", *4th Karlsruhe Workshop on Software Radios*, Karlsruhe Germany, March 2006.

[11] Kirkpatrick, S., C. D. Gelatt Jr., M. P. Vecchi, "Optimization by Simulated Annealing",*Science*, 220, 4598, 671-680, 1983

[12] B. Bailly, E. Bidet, JM. Cardin, M. Djoko Kouam, C. Joanblanq, J. Palicot, "FDF: A 512 FIR filter using a Mixed Temporal-Frequential Approach", *CICC'95 Custom Integrated Circuit Conference*.