Install or invoke?: The optimal trade-off between performance and cost in the design of multi-standard reconfigurable radios

Virgilio Rodriguez†, Christophe Moy and Jacques Palicot∗†
SCEE Research Team, IETR, Supélec, Cesson-Sevigné, France

Summary
We approach the design of multi-standard reconfigurable radio as a choice between two extremes: (a) installing complex, self-contained dedicated components that provide high performance at a high cost (‘Velcro approach’), versus (b) supporting all targeted standards through primitive lower level components, which reduces cost but maximises latency. To find the optimal trade-off between performance and cost, we build a mathematical model, in which we represent the design choices through a graph of progressively simpler functional modules. We illustrate our approach through a simple but realistic design example. Additionally, we show that this optimisation can be reformulated as a version of the well-known network-design problem, which opens the door to a wealth of results, algorithms and experience, already available in the scientific literature. We identify some specific algorithms that can help us find the optimal design and discuss limitations and future directions of our research program. Copyright © 2007 John Wiley & Sons, Ltd.

KEY WORDS: reconfigurability; software-defined radio; design; graph theory; network design problem; optimisation; multi-standard; SDR; parametrisation

1. Introduction
We view the design of a multi-standard reconfigurable radio as choosing the optimal point between two extremes. At one extreme is the ‘Velcro’ solution: to support several communication standards through self-contained complex components, each exclusively dedicated to a given standard. At the other extreme, we can attempt to support all desired standards through very simple, primitive operators (adders, multipliers, MAC, etc.) by invoking them repeatedly, in order to perform the various communication tasks necessitated by the supported standards. The Velcro approach will generally provide the best performance, but at the highest monetary cost. In a Velcro design, much functionality is wastefully replicated. Conversely, by going to the other extreme, we can minimise the monetary cost of the radio, but at a performance level that may be unacceptable for many practical applications. Thus, we need to find the right level of complexity at which to use self-contained dedicated components. This is the level that gives the optimal trade-off between performance and cost.

*Correspondence to: Jacques Palicot, Supélec; Avenue de la Bouliaie, BP 81127, 3551 Cesson-Sevigné, France.
†E-mail: Jacques.Palicot@supelec.fr
‡V. Rodriguez is now with the Institut für Nachrichtentechnik, at the Universität Karlsruhe, in Germany.
Copyright © 2007 John Wiley & Sons, Ltd.
To find the optimal performance/cost trade-off, we model the equipment as a hypergraph of progressively simpler functional modules. The functionality of one such module can be provided either by installing a self-contained component, or by the invoking (possibly many times) simpler (lower level) modules. Thus, for us, ‘module’ refers to a functionality, while a self-contained component is an optimised hardware/software combination, dedicated to provide a functionality in the most efficient manner. One such component could be an equaliser, for example or an implementation of the fast Fourier transform (FFT). But rather than installing such component, we could achieve the given functionality by invoking lower level modules, instead. For example some forms of equalisation can be achieved through the FFT [1]; and the functionality of the FFT operator can be achieved via a butterfly module, which itself could be implemented via CORDIC [2] (and each of the butterfly and CORDIC modules could be implemented through an installed component, or by invoking yet simpler modules, and so on). Simpler, lower level components are generally less expensive to build and can be reused by several upper level modules inside and across standards. Thus, the use of lower level components reduces the monetary cost of the equipment. Unfortunately, such use generally increases the execution time of the concerned task. As a consequence, the total execution time of a given operation may exceed practical limitations. Thus, to achieve the optimal trade-off between performance and cost, we need to answer repetitively the question: should we install or invoke?, that is should we install now, at this level of complexity, a self-contained component, or invoke yet lower level modules to achieve the desired functionality?

At the foundation of our study is the ‘common operators’ approach to the design of reconfigurable equipment. Its main principle is the identification and (re)use throughout the design of common components that can each match several processing contexts, via a simple parameter adjustment. This approach can greatly increase the efficiency of a multi-standard software-defined radio, both in terms of its cost, and of the speed of reconfiguration during operation. The common operators approach is discussed more extensively in Reference [3].

The present work builds upon some recent publications of ours. In Reference [4], we introduce our formal approach to the design of multi-standard radios. To explore the cost/performance trade-off in Reference [4], we combine economic and latency considerations into a single cost function. The combined cost function (a weighted sum) is a reasonable first step, as it simplifies the exposition and the solution procedure. But we acknowledge that this combination has some drawbacks: it adds the monetary cost (paid once by the designer) with the ‘delay costs’ incurred by the user during each use, throughout the entire useful life of the radio, it fails to account for the hard time constraints, often arising from communication applications, and ultimately makes the chosen design highly dependent on the weights (which are themselves arbitrary, even if judiciously chosen with attention to real-life considerations). To address these concerns, in Reference [5], we minimise the (monetary) cost only, while considering latency through appropriate execution time constraints that cannot be exceeded. Neither of References [4,5] provides the ideal algorithm to achieve the solution. Both References [4,5] utilise exhaustive search (which may be impractical for large design problems), and Reference [4] briefly explores simulated annealing (a general algorithms not specifically oriented to our kind of problem [6]). In Reference [7], we recast our problem as a ‘network-design problem’. The network-design problem is well known and counts with a very extensive literature, which offers algorithms, computational complexity results and many practical applications in a variety of contexts such as telecommunication, transportation, water resources and even social interactions [8]. Indeed, we have already identified specific algorithms which seem particularly well suited to our purposes, and which we are examining [9–11]. In the present work, we consolidate, clarify and where appropriate correct and extend the work reported in References [4,5,7].

Neither our previous nor the present work addresses the identification of new common operators, useful to the design of multi-standard reconfigurable radios. Such identification is, in its own right, an active area of research. Researchers are proceeding along several directions. For instance, Reference [1] shows that many important tasks of a communication receiver can be implemented through the FFT. In turn, the FFT can be implemented via the butterfly operator, and, as argued more recently [2], via CORDIC. With this in mind, some researchers are seeking frequency-domain implementations for different families of algorithms. For example Reference [12] studies the frequency-domain implementation of Reed-Solomon channel decoding.

Other relevant works describing interesting approaches to the design of reconfigurable radios include: References [13,14], which argue for parametrised
design from a 'common functionality' perspective, as originally advocated in Reference [15]; Reference [16] whose approach is inspired by object-oriented programming; Reference [17] which attempts to cover hardware and software design under a common methodology; and Reference [18] which proposes designs that integrate the entire system on a chip (SoC).

The rest of the paper proceeds as follows. First, we build the mathematical model. This step includes the drawing of a graph that represents design alternatives, the consideration of possible performance metrics and the specification of an objective function and appropriate constraints. Subsequently, we discuss the main approaches we have taken to consider cost and latency in the same optimisation problem. We then undertake the optimisation procedure under two different cost functions (i) weighted sum of monetary and 'delay' costs, as in Reference [4], and (ii) monetary cost only, with 'delay' as a constraint, as in Reference [5]. Subsequently, we discuss and interpret our results. Then, we show that our architecture optimisation enquiry can be reformulated as a 'network-design problem', and comment on specific algorithms available in the literature. We conclude the paper, with a discussion addressing further interpretations, as well as limitations and future directions of our research program.

2. Mathematical Model

2.1. Graph-Theoretic Representation

As illustrated by Figure 1, we represent a multi-standard radio, as a hypergraph of progressively simpler functional modules. Each node (module) represents a functionality that can either be implemented via a dedicated hardware/software component (e.g. an ASIC, or an intellectual property core), or can be achieved by invoking lower level modules. The hyperarcs leaving a node (parent) specify the simpler modules (descendants) that could provide the required functionality through multiple calls. Descendant nodes may not all be at the same level.

An OR arc (direct arrow) means that only one of the descendant nodes is necessary to implement the functionality of the parent node. For example, \( S_1 \) can be implemented through any one of \( A_1 \), \( A_2 \) or \( B_1 \). An AND arc (inverted Y connection as pointing from \( S_3 \) to \( A_4 \) and \( A_5 \)) means that all descendant nodes are needed to implement the functionality of the parent node. In some cases, a parent node may have both AND and OR dependencies with its descendants. For example there are three alternatives for providing the functionality of \( S_2 \): (1) installing a self-contained complex component dedicated to \( S_2 \), (2) making available both \( A_2 \) and \( A_3 \), (3) making available \( A_4 \).

The roots of this graph, at the highest level, represent the standards to be supported by the radio. The fact that the graph is acyclic helps the process of finding the solution to our design problem.

2.2. Optimisation Parameters

The decision to provide a functionality via a self-contained, dedicated component or by invoking multiple-times simpler modules is determined by two key considerations: (monetary) cost and (execution) time.

The monetary cost of a component (which is paid only once during the useful life of the radio) represents the total cost of including the component in the design. In some architectures for reconfigurable radios, the monetary cost can be represented by (is proportional to) the number of logic units necessitated by an FPGA implementation. It is best to take the view point of a system integrator that ‘outsources’ the components (software or hardware). Then, the monetary cost represents the fair market value, at design time, of acquiring the finished component in the open market.

Execution time (incurred every time a component is employed throughout the life of the system) is also a critical consideration, because the user has limited patience, and in fact, communication standards impose hard time constraints (deadlines) for the completion of certain operations.
2.3. Considering Both Cost and Latency: Two Approaches

There are at least two basic approaches to considering cost and latency in the design of a reconfigurable equipment: (i) the ‘combined’ cost function and (ii) the deadline approaches.

As we have done in Reference [4], we can combine economic and latency considerations into a single cost function (a weighted sum of both ‘costs’). The combined cost function has at least three serious problems: (i) it adds the monetary cost (paid once by the designer) with the ‘delay costs’ incurred by the user, each time it executes a standard throughout the useful life of the equipment, (ii) it fails to account for the hard time constraints, often arising from communication applications, and (iii) it makes the design highly dependent on the weights, which are themselves arbitrary. Nevertheless, the combined cost function is a reasonable first step, as it simplifies the exposition and the solution algorithm. It may also be helpful in pre-design work, and could allow the designer to gear a design to the ‘luxury’ (high performance) or ‘value’ market sectors.

As an alternative, as in Reference [5], we can minimise the economic cost only and take latency into account through execution time constraints (‘deadlines’) that cannot be exceeded. However, determining the ‘deadlines’ to be observed while designing the radio to support a given standard is non-trivial and should be done with great care. If deadlines are set too high, they may lead to a design that fails to perform at acceptable levels in practical situations. But if deadlines are set lower than necessary, the resulting design will be more expensive than necessary. The key to determining the right delay tolerances is to examine the ‘transmission chain’, applicable to each supported standard (the block diagram depicting the various steps necessary to establish end-to-end communication under a considered standard, as shown by Figure 2), and the numerical specifications provided by the standardisation bodies. A chosen architecture must yield a performance able to support end-to-end communication under each of the supported standards. We do not further address here the determination of these deadlines and assume that the pertinent information has been obtained elsewhere and given to us.

2.4. Formalising the Optimisation Problem

Conceptually, the ‘deadline approach’ includes as a special case, the ‘combined cost’ approach. This is so, because when we use the combined cost, we can pretend that this cost is the original monetary cost, and that we ignore the deadlines because they are ‘very large’. Thus, we simply choose the least expensive design. Presumably, the designs that are ‘too slow’ are also (under the combined cost function) very expensive and we would not choose them for that reason. Thus, below we formalise the deadline approach only, and think that when we use the combined cost function, all deadlines are set to infinity.

Let:

\[ S_i \text{ with } i = 1, \ldots, I \text{ be the standards to be supported,} \]
\[ \delta_i \text{ the ‘design deadline’ of standard } S_i. \]
\[ F_k \text{ with } k = 1, \ldots, K \text{ be 0-1 variables such that,} \]
\[ F_k = 1 \text{ if component } k \text{ is chosen to be installed in self-contained (dedicated) form.} \]
\[ C_k \text{ and } T_k \text{ be respectively the (monetary) cost and} \]
\[ \tau_i(F_1, F_2, \ldots, F_K) \text{ be the time of executing the tasks} \]
\[ \text{used to calculate the ‘deadline’ of standard } i \text{ for a} \]
\[ \text{particular choice of components.} \]

Notice that not all the possible designs are acceptable. For example if there are three root nodes (standards), and the self-contained components that can each execute an entire standard (for the ‘Velcro’ design) are labelled 1, 2 and 3, then, the point (1, 1, 0, 0, 0, 0, 0, 0) is in principle acceptable (but not necessarily optimal, of course!... this is the ‘Velcro’ architecture). But some other choices would not support the desired standards (e.g.
THE OPTIMAL DESIGN OF MULTI-STANDARD RECONFIGURABLE RADIOS

(0, 0, . . . , 0), which means building nothing at all, is clearly unacceptable.

Let Ω denote the set of all K-tuples of binary numbers that correspond to acceptable choices of the set of components (this is information taken directly from the hypergraph).

Then we seek to solve:

\[
\min_{(F_1, F_2, \ldots, F_K) \in \Omega} \sum F_k C_k \\
\text{subject to} \\
\tau_1(F_1, F_2, \ldots, F_K) \leq \delta_1 \\
\vdots \\
\tau_I(F_1, F_2, \ldots, F_K) \leq \delta_I
\]

3. Solving the Optimisation

In principle, the preceding problem can be solved directly by computing the cost of every feasible design (choice of components that supports the desired standards and obeys the deadlines, if applicable), and choosing the design of least cost. For larger problems, a 'smarter' algorithm is needed. Below, we provide an illustration which can be solved with minimal computing effort.

3.1. A Realistic Illustration

Figure 3 (representing an evolution of a figure from Reference [4]) shows a sub-graph, corresponding to the decomposition of several processing elements (equalisation, multi-channel, OFDM) that could be part of a multi-standard radio system. The sub-graph is not intended to show all the possible alternative implementations for each of the considered processing elements. Root nodes (standards) are not shown.

The equalisation block compensates for the multi-path impairment typical of wireless channels. It can be either implemented through a finite-impulse response filter (FIR) or through the FFT operator. The implementation of equalisation in the frequency domain is particularly attractive for channels that exhibit long-impulse responses, which leads to FIR filters with a very high number of taps [19]. Notice that the inverse FFT is computationally equivalent to the FFT; thus, we attach a multiplicative factor of 2 to the arc pointing from the equaliser to the FFT. Multi-channel refers to the channelisation function of a cellular base station. This can be accomplished via the 'classical' channel per channel procedure, or by proceeding in parallel, through a filter-bank channeliser (which can be implemented via FFT). Other lower level modules correspond to well-known signal processing constructs.

The graphical values, that are near the bottom right of a block represent cost/time associated with the component that could provide that functionality. The units of measurement are immaterial for our purposes. We only need relative figures to be able to compare one design alternative to another. But the time figures must be consistent with those in which the deadlines are expressed. At the top left, there is a numerical identifier for the corresponding module.

Below, we assume that the channeliser handles 25 channels, that the FFT is of order 64, and that to implement this FFT through a butterfly, 192 calls are needed.

3.2. Results and Interpretations

(1) Deadline approach: The sub-graph of Figure 3 is small enough to be solved by exhaustive search, although the number of design alternatives quickly explodes. The thick broken blue line in Figure 3 shows an FFT-only design, which is a conceivable outcome of the optimisation, for a specific set of deadlines.

In Figure 4, we integrate the information in Table I and Figure 3 into the widely available spreadsheet model, which may be sufficient for smaller designs. Figure 4 illustrates the solution procedure and shows some interesting design alternatives. For this illustration, we have ignored the designs that use the simplest (lowest level) components (adder, multiplier, etc). Thus, the simplest considered component is the CORDIC.
A 1 in the column corresponding to a module means that, in that particular design, the module is implemented via a dedicated component. $T_1$, $T_2$ and $T_3$ are the execution times corresponding to OFDM, equalisation and the channeliser for a given design. Nevertheless, for the sub-design illustration under consideration, we pretend that the tasks at the top of the graph have associated deadlines (determined by the supported communication standards). We have considered that an order-64 FFT operator satisfies our requirements and that the multi-channeliser handles 25 channels.

The 'Cost' column shows that the least expensive design consists of a CORDIC component only, but it performs slowly. This would be the chosen design if the deadlines are, for example 6,200, 12,300 and 320,000, respectively (recall that no time unit has been specified so far). If the deadlines are tighter, the CORDIC-only design falls outside the feasibility region. Then other candidates may be chosen. For example the butterfly-only design performs a good deal better.
THE OPTIMAL DESIGN OF MULTI-STANDARD RECONFIGURABLE RADIOS

Fig. 4. The figure illustrates how to integrate the design data and procedure into the widely available spreadsheet model, which may be sufficient for smaller designs (the data displayed vertically may be easier to read in Table I). Only certain interesting design alternatives are shown. A row represents a design. A 1 in the column corresponding to a module means that this module is implemented via a self-contained component.

\(T_1, T_2\) and \(T_3\) are the execution times corresponding to, respectively, OFDM, equalisation and the multi-channel channeliser for a given design. The ‘Cost’ column shows that the least expensive design consists of a CORDIC component exclusively, but performs poorly. The ‘Velcro’ solution (bottom) performs fastest but costs most, as it implements the top modules with self-contained components.

in all three tasks, and is only marginally more expensive. The FFT-only design costs about 70 times more than the butterfly-only alternative, but performs seven to eight times better, and could very well be the optimal choice under tighter time constraints. The ‘Velcro’ solution performs best and costs most, as it implements the top modules with dedicated components. The FFT-only design is far behind the Velcro approach, only with respect to \(T_3\) (multi-channeliser). This performance gap narrows considerably if one complements the FFT component with a filter-bank component. The Filter-bank/FFT combination provides performance near Velcro’s, for about one quarter of the cost.

(2) Combined cost function: We can use the data of Figure 4 to build Table II and illustrate the combined cost function approach, which we first applied in Reference [4]. In Table II, the columns labelled \(CC(w)\) contain for the corresponding design the ‘combined cost’, which is a weighted sum of the form \(w \times \text{cost} + T_1 + T_2 + T_3\), where \(w\) is the weight. A weight of zero simply means that ‘money is no object’, and we want the best performing design at any monetary cost. In that case, as column \(CC(0)\) shows, the ‘Velcro’ design is optimal. But as soon as we consider cost at all, even with a fairly small weight of only 1 (column \(CC(1)\)), the FFT-only design is non-optimal. With a weight of 13 the FFT-only design is the winner, which is consistent with the results presented in Reference [4]. And it takes a much larger weight (1000) for the butterfly-only design to become optimal. With a huge weight on

Copyright © 2007 John Wiley & Sons, Ltd.
DOI: 10.1002/wcm
Table I. The table shows for each module in Figure 3, its numeric identifier, verbal description, as well as its monetary cost and execution time, if implemented through a self-contained component. Money and time units are not specified.

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Money</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>73</td>
<td>OFDM</td>
<td>1200</td>
<td>300</td>
</tr>
<tr>
<td>72</td>
<td>Equaliser</td>
<td>1500</td>
<td>200</td>
</tr>
<tr>
<td>71</td>
<td>Multi-channeliser</td>
<td>10000</td>
<td>10</td>
</tr>
<tr>
<td>62</td>
<td>Filter bank</td>
<td>2000</td>
<td>100</td>
</tr>
<tr>
<td>61</td>
<td>Channel-per-channel</td>
<td>1500</td>
<td>200</td>
</tr>
<tr>
<td>51</td>
<td>Multi-rate filter</td>
<td>700</td>
<td>700</td>
</tr>
<tr>
<td>42</td>
<td>FIR</td>
<td>500</td>
<td>1000</td>
</tr>
<tr>
<td>41</td>
<td>Down sampler</td>
<td>1000</td>
<td>200</td>
</tr>
<tr>
<td>33</td>
<td>FFT</td>
<td>1500</td>
<td>500</td>
</tr>
<tr>
<td>23</td>
<td>Butterfly</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>22</td>
<td>Down converter</td>
<td>30</td>
<td>10</td>
</tr>
<tr>
<td>21</td>
<td>CIC filter</td>
<td>300</td>
<td>50</td>
</tr>
<tr>
<td>13</td>
<td>MAC</td>
<td>12</td>
<td>8</td>
</tr>
<tr>
<td>12</td>
<td>CORDIC</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>11</td>
<td>CIC cell</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Look-up table (LUT)</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Adder</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Multiplier</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>Delay</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

monetary cost (50 000), which essentially means we seek the least expensive design without concern for performance, the CORDIC-only design is chosen.

4. Toward an Efficient Algorithm: The Network Theory Reformulation

4.1. Motivation

Above, we have described a formulation that allows us to find for a multi-standard reconfigurable radio, an architecture which is optimal in view of both cost (money) and performance (computational execution time). We can find the exact optimum through exhaustive search (‘brute force’). In Reference [4], we also explored finding an approximate solution through simulated annealing. The ‘brute force’ approach is computationally impractical when there is a very large number of design alternatives. The second approach utilises a ‘generic’ algorithm not specifically oriented to the problem at hand, and may or may not yield a solution near the ‘true’ optimum. A third alternative is to seek a ‘smart’ algorithm which fully utilises the special structure of the graph to efficiently search the solution space. In the present section, we move in that direction, by recasting our problem as a single-source (or single terminal (sink)) ‘network-design problem’.

The network-design problem is well known and counts with a very extensive literature, which offers algorithms, computational complexity results and many practical applications in a variety of contexts such as telecommunication, transportation, water resources and even social interactions [8]. Below, we show how to perform the problem reformulation and give a specific example grounded on a practical design. In support of our approach, we identify and discuss several specific algorithms which seem particularly well suited to our purposes [9–11].

4.2. ‘Network-Design’ Problem

Figure 5(a) provides a simple illustration of the single-source network-design problem. Suppose there is an initial point called the origin, o, and three ‘terminals’ (destinations), t₁, t₂ and t₃. We wish to connect o to each tᵢ in a way that satisfies certain optimality criteria. There may also be some intermediate nodes, which themselves may be (partially) interconnected. There are many conceivable solutions. The most obvious one is simply to build three direct ‘roads’, one from o to t₁, a different one from o to t₂ and a third one from o to t₃. This solution is certainly plausible and probably provides the shortest ‘travel times’, but it is unlikely to be ideal. By building dedicated roads, we are possibly missing the savings resulting from having ‘common segments’ which may be used by all traffic,

<table>
<thead>
<tr>
<th>Design</th>
<th>Cost</th>
<th>T₁</th>
<th>T₂</th>
<th>T₃</th>
<th>CC(0)</th>
<th>CC(1)</th>
<th>CC(13)</th>
<th>CC(1000)</th>
<th>CC(50 000)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORDIC</td>
<td>11</td>
<td>6144</td>
<td>12 288</td>
<td>313 344</td>
<td>331 787</td>
<td>331 919</td>
<td>342 776</td>
<td>881 776</td>
<td></td>
</tr>
<tr>
<td>CORDIC+DDSamp</td>
<td>12</td>
<td>6144</td>
<td>12 288</td>
<td>307 925</td>
<td>326 377</td>
<td>326 356</td>
<td>338 357</td>
<td>326 357</td>
<td></td>
</tr>
<tr>
<td>Butterfly</td>
<td>15</td>
<td>3840</td>
<td>7680</td>
<td>195 840</td>
<td>207 360</td>
<td>207 385</td>
<td>207 575</td>
<td>222 360</td>
<td></td>
</tr>
<tr>
<td>FIR+CORDIC</td>
<td>51</td>
<td>6144</td>
<td>1000</td>
<td>31 144</td>
<td>38 288</td>
<td>38 799</td>
<td>44 951</td>
<td>549 288</td>
<td></td>
</tr>
<tr>
<td>FFT+butterfly</td>
<td>315</td>
<td>3840</td>
<td>1000</td>
<td>28 840</td>
<td>33 660</td>
<td>34 195</td>
<td>40 375</td>
<td>548 660</td>
<td></td>
</tr>
<tr>
<td>FIT+FFT</td>
<td>1000</td>
<td>500</td>
<td>1000</td>
<td>25 500</td>
<td>27 100</td>
<td>28 000</td>
<td>40 000</td>
<td>1027 000</td>
<td></td>
</tr>
<tr>
<td>‘Velcro’</td>
<td>1200</td>
<td>300</td>
<td>200</td>
<td>10</td>
<td>510</td>
<td>13 210</td>
<td>165 610</td>
<td>12700 510</td>
<td></td>
</tr>
</tbody>
</table>

Copyright © 2007 John Wiley & Sons, Ltd.


DOI: 10.1002/wcm
Fig. 5. The network-design problem and the design of multi-standard reconfigurable radios. (a) A simple network-design problem; (b) The ‘network-design’ version of a section of the design graph. Solid arrows indicate ‘free’ pre-built ‘algorithmic roads’ that take time to travel, but require no monetary expenditure.

regardless of the destination. For example we could build a road from o to some intermediate point, A, plus dedicated segments from A to each t_i. Because the segment o → A is used by all traffic, certain savings may result (although this may lengthen the travel time to all traffic).

By the same reasoning, building dedicated segments from A to each t_i may not be optimal. Perhaps we can build a dedicated road A → t_3, but also a segment from A to some intermediate point B, followed by segments B → t_1 and B → t_2. In the latter proposal, all traffic would travel over the o → A segment, the t_3 traffic would go directly over A → t_3 while both the traffic to t_1 and to t_2 would continue over A → B and from B over B → t_1 and B → t_2, respectively. There are many other possibilities.
4.3. From a Hypergraph to a Network-Design Problem

Our problem can be recast as a network-design problem by proceeding as follows. The communication standards correspond to terminals we wish to reach. Building a direct road from the origin to each terminal corresponds to the ‘Velcro’ solution: choosing a self-contained dedicated complex component to support each standard. This will provide the ‘fastest routes’ to be sure, but probably not the ideal solution. Installing a self-contained component of ‘medium’ complexity is the equivalent of building a road from the origin to the intermediate point A in the example above. For instance, Reference [1] shows that many important tasks of a communication receiver can be implemented through the FFT. In turn, the FFT functionality can be implemented with a butterfly operator. Thus, we can ‘build a road’ from the origin to the FFT (i.e. install a self-contained FFT component) to be shared by several tasks. This will likely reduce the overall cost of the design, although it will generally ‘increase the travel time’. But we could also ‘reach’ the FFT node (i.e. provide the functionality of the FFT operator), by first ‘building a road’ from the origin to the intermediate junction ‘butterfly’ and from there utilise an ‘existing road’ to the FFT (i.e. install a self-contained butterfly component, and invoke it repeatedly to provide the FFT functionality). Further explanation of the reformulation is given below through examples.

4.4. Illustration of the Graph-to-Network Conversion

Figure 5(b) shows the network-design problem, corresponding to a section of the realistic hypergraph of design choices given by Figure 3. The pairs of the graph that concern lowest level components and the channeliser have been ignored for pedagogical reasons. Recall that now the objective is to build a ‘road network’ that provides a path from the origin to each ‘terminal’. The solid arrows mark ‘free’ (pre-built) roads that take time to travel, but require no monetary expenditure. These represent known algorithms which can be used to provide a higher level functionality. For example from the FFT node there is a pre-built (free) ‘road’ to OFDM, another to equalisation and yet another to the FIR node, because there are known algorithms that allow the implementation of equalisation, OFDM and FIR through the FFT operator [1]. The dashed arrows indicate ‘possible roads’ that definitely cost money to build (but possibly negligible time to travel). Building a ‘road’ means installing a self-contained component to provide the functionality pointed by the arrow. For example there is a dashed arrow from the origin to FFT (meaning we can provide the FFT functionality by installing a self-contained FFT component). There is also a ‘possible road’ from the origin to butterfly. If we build this ‘road’ (i.e. if we install a self-contained butterfly component) we can ‘reach’ the FFT node via a free (algorithmic) path (shown by a solid arrow from butterfly to FFT), because one can provide the FFT functionality by repeatedly invoking a butterfly operator.

Figure 6 shows in greater detail the ‘network-design problem’, corresponding to Figure 3. The fact that the digital down conversion module can be done through CORDIC, and the lowest level modules are not considered, for clarity. And some obvious dashed arrows from the origin (e.g. from the origin to equaliser, meaning installing an equaliser component) are omitted to reduce clutter. The main purpose of Figure 6 is to show that representing the AND dependencies is challenging in some cases, and may require some ingenuity and additional research. The FIR ‘triple node’ illustrates some of the main issues.

A ‘triple’ FIR node is used in Figure 6 to show that there are three different ‘routes’ to get the FIR functionality, and the route has an impact on what can be done from there. The ‘sub-node’ FIR1 concerns the case where the FFT is used to implement the FIR. The ‘sub-node’ FIR2 refers to the case when an FIR component is installed, even though the FFT functionality is present (this may be done to reduce execution time). FIR3 denotes a case in which an FIR component is installed and the FFT functionality is not available (perhaps OFDM and equaliser dedicated components have been installed, and the FFT module is unnecessary). In the cases FIR1 and FIR2, the functionality of the filter-bank module can be achieved algorithmically, because both the FFT and the FIR are available. However, under FIR1, there is no FFT module, thus, the filter-bank functionality would require an installed component. For this reason, there is no arrow pointing to the filter-bank module from the FIR1 sub-node. On the other hand, the solid arrow pointing to the equaliser from the FIR is drawn from the oval, around the various FIR sub-nodes. This indicates that we can achieve equalisation through the FIR, irrespective of how we achieve the FIR functionality (contrary to the filter-bank case).

Shown also on Figure 6 is a simpler case of AND dependency, which involves achieving multi-rate filtering (MRateFilt) through both FIR and down sampling (DSamp). The dash line from the FIR oval
4.5. Possible Algorithms

The reformulation of Section 4 provides us with several well-studied algorithms. Of these algorithms, we shall now identify and discuss some that seem particularly promising.

Reference [9] considers a network-design problem with two metrics: cost and ‘distance’. This fits well with our formulation, with execution time as ‘distance’. It follows the first approach discussed in Section 2.3, that is it combines both cost and ‘distance’ into a single objective. But this algorithm is probabilistic. Reference [10] modifies it to make it deterministic, utilising linear programming.

The basic idea of Reference [9] is as follows. The algorithm has several stages. At each stage, it forms a matching (‘pairing’) of terminal nodes (not previously eliminated). Then, of each pair of nodes, it chooses a ‘centre’. At this point, the non-centre nodes are removed from further consideration and a new stage of the algorithm starts (with a new matching among the ‘centres’ only). Because for us, the terminals correspond to standards to be supported by the radio, they should be relatively few. Thus, this algorithm will give an answer in a relatively short number of stages.

For reasons given in Section 2.3, it may be preferable to utilise an algorithm which minimises costs, while keeping ‘length’ (time/latency) under a specified budget. Reference [11] provides an algorithm that does exactly that. However, this algorithm appears more complex than the preceding ones.

Currently, we are evaluating these algorithms, while continuing our exploration of the rich network-design literature.

5. Discussion and Outlook

We have built a mathematical model that enables us to identify an architecture for a multi-standard reconfigurable radio, which is optimal, in view of both performance and cost. The chosen architecture represents an optimal point between two extremes: (i) highly complex communication components, each exclusively dedicated to a given standard (‘Velcro’ approach), and (ii) very simple components to be invoked by ‘higher layers’, in support of the various standards. We represented the radio as a hypergraph of progressively simpler functional modules.

We have illustrated our approach through a simplified ‘sub-design’. We do not claim that it corresponds to a system available today, but it is sufficiently close to reality to provide useful insights. Our results agree with intuition. When the execution time constraints, imposed by the supported standards
on the considered high-level modules are very tight (or when the weight given to monetary cost in a combined cost function is relatively low), the more complex dedicated components are chosen. On the other hand, if the time constraints are lenient (or if the weight given to money in a combined cost function is relatively high), money is saved by supporting the standards through simpler (lower level) components, such as the FFT, butterfly or even the CORDIC operator. The cost and time figures we have used are, in our judgement, reasonable and consistent. But we acknowledge that obtaining better estimates of these values for realistic designs is an important future task.

Our analysis also sheds some light on some of the economic issues that may arise when a single architecture targets several communication standards. An optimal architecture that exclusively supports standards oriented to ‘slow’ applications, such as voice and text messaging, will favour lower cost, simpler, reusable components. Thus, a consumer who is primarily interested in such ‘traditional’ applications will be better off by purchasing equipment that supports only a few simple communication standards. In other words, certain care must be taken in choosing the set of standards to be supported by a single multi-standard reconfigurable radio.

The algorithm chosen to find the optimal architecture requires additional attention. In solving our ‘sub-design’ illustration, we adopted a ‘manual’ (spread-sheet) approach, while in previous work, we explored briefly simulated annealing [6]. Both of these choices have problems already discussed. In search of a better algorithm, we have reformulated our design problem as a ‘network-design problem’, which opens the door to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extensive literature, offering many algorithms and results. Indeed, we have already identified and to an extent. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced’. The object-oriented programming sense, as advocated in Reference [16]. The cost of one object would be its ‘fair market value’ if ‘outsourced'. The basic trade-off between performance and cost would remain. Presumably, primitive objects should be very inexpensive to source or develop. But supporting all communication tasks by invoking very simple objects.

Several important relevant issues have been neglected so far, or require additional attention. For example, we have assumed that for every functional module, there is exactly one component which can provide the desired functionality at a given performance level, and for a given cost. In practise, for a given functionality (say the FFT operation), the market may provide many alternatives, each offering a different price/performance trade-off. Our formulation can be modified to consider various components providing the same functionality. Building the hypergraph of design choices is itself an object of research. We have already pointed out that 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 that is typically provided by time-domain procedures. The discovery of any such algorithm will imply a modification of the graph with new arcs, pointing to the FFT operator. Also, as communication standards evolve, the graph may need to be modified.

Other important issues requiring (additional) attention include consideration of (i) the time needed to re-configure the radio, while switching from a standard to another, (ii) the ‘travel time’ of signals from a component to another inside the radio, and (iii) the possible contention among high-level modules for the service of the same lower level module. This contention may be particularly important if the radio needs to support simultaneous communication over several standards at the same time, such as, for example, downloading a file through a wireless local area network, while simultaneously video conferencing through UMTS.

Our framework is easier to visualise if one thinks of a ‘component’ as a physical entity (such as a chip) that may or may not be installed. However, our approach is more general than that. For instance, it can accommodate a central processor-based architecture. In a design based on a digital signal processor (DSP), the components could be ‘objects’ (in the object-oriented programming sense, as advocated in Reference [16]). The cost of one object would be its ‘fair market value’ if ‘outsourced’. The basic trade-off between performance and cost would remain. Presumably, primitive objects should be very inexpensive to source or develop. But supporting all communication tasks by invoking very simple objects.
should likely lead to unacceptably slow designs. At the other extreme, a highly optimised object that is dedicated to support a standard should provide the best performance, but require the highest monetary expenditure, whether this object is outsourced or developed ‘in house’. On the other hand, the trade-off between the processing power and the cost of the DSP should also be considered in the optimisation, which we have not directly addressed in the present report. Besides cost and computing performance, other factors such as the DSP’s energy consumption and heat production may have to be considered as well. Ultimately, we are well aware that we have raised more questions than we have answered. Nevertheless, finding for a multi-standard reconfigurable radio, an architecture that is optimal in view of both performance and cost is a very worthy cause. We believe that this report establishes the general validity and usefulness of our approach. Indeed, its greatest contribution might be to attract the attention of other researchers whose work may address some of the important questions we have left unanswered.

Acknowledgement

Financial support by the ‘Région Bretagne’ of France is gratefully acknowledged.

References


Authors’ Biographies

Virgilio Rodríguez is a Research Associate with the Institut für Nachrichtentechnik, at the Universität Karlsruhe, in Germany. He is involved with ultra wide band (UWB) technology, as part of the European project PULSERS. From the Fall of 2005 through the summer of 2006, he was at SapCèle (Campus de Rennes), in France, where he applied mathematical formalisms to achieve designs of multi-standard reconfigurable radios that are optimal in view of both cost and performance. Before this, he spent a year as a Research Fellow at the University of Surrey, England, where he applied market principles to the dynamic management of the radio spectrum, as part of the End-to-End Reconfigurability (E2R) European project. He received his PhD degree in May 2004, from the Polytechnic University (Brooklyn, New York), where he focused on analytical studies for radio resource management in the context of third-generation cellular communication networks. Previously, he received Master’s degrees from both Purdue University and
the Virginia Polytechnic Institute, for work, respectively, in electrical and systems engineering.

Christophe Moy is an engineer of the INSA (National Institute of Applied Sciences), Rennes, France, 1995. He received his M.Sc. and Ph.D. degrees in Electronics in 1995 and 1999 from the INSA. He worked from 1995 to 1999 on spread spectrum and RAKE receivers for the IETR. He then worked 6 years at Mitsubishi Electric ITE-TCL research lab where he was focusing on Software Radio systems and concepts, including digital signal processing, HW and SW architecture, co-design methodology and reconfiguration. He also addressed the design of SDR impulse UWB systems. He represented Mitsubishi Electric at the SDR Forum and worked on the French research program A3S, as well as the IST project E2R phase 1. He is now an Associate Professor at Supélec, campus of Rennes, France. His research is done in the SCEE laboratory, specialized in software and cognitive radio, which is part of the IETR. He addresses heterogeneous architectures design and reconfiguration management for SDR and cognitive radio. He participates to the IST Network of Excellence Newcom, the IST project E2R phase 2, as well as the French research program IDROMel.

Jacques Palicot received the Ph.D. degree in signal processing from the University of Rennes, France, in 1983. Since 1988, he has been involved in studies about equalization techniques, applied to digital transmissions and new analog TV systems. Since 1991, he has been involved mainly in studies concerning the digital communications area and automatic measurement techniques. He has taken an active part in various international bodies (EBU, CCIR), and in European projects. He has published various scientific articles, notably on equalization techniques, echo cancellation, neural techniques and hierarchical modulations. He is currently involved in adaptive signal processing and in new techniques as software radio and Cognitive Radio techniques. Since October 2003, he has been with Supelec, Campus of Rennes, where he is a Professor and head of the Signal, Communication & Embedded Electronics research team.