High Data Rate Fully Flexible SDR Modem

Advanced configurable architecture & development methodology

KASPERSKI F., PIERRELEE O., DOTTO F., SARLOTTE M.
THALES Communication
160 bd de Valmy, 92704 Colombes, FRANCE
francois.kasperski@fr.thalesgroup.com
olivier.pierrelee@fr.thalesgroup.com
frederic.dotto@fr.thalesgroup.com
michel.sarlotte@fr.thalesgroup.com

***Abstract*—With the multiplication of mobile and wireless communication networks and standards, the physical layer of communication systems (i.e. the modem part of the system) has to be completely flexible. This assumption leads to the well known Software Defined Radio concept which enables the implementation and the deployment of different waveforms on the same platform. This concept has been widely investigated since the early 2000's mainly for processors and Sw approach but less for reconfigurable Hw or DSP implementation. This paper deals with a specific architecture and an innovative design methodology which were designed within the framework of a fully flexible high data rate Software Defined Radio wireless modem. This approach is focused on the waveform part of the system and its goal is to reach a fully flexible physical layer. In case of modem evolutions or upgrades, it enables to avoid significant rework and extra cost in term of waveform development. Moreover the association of the right architecture and the right methodology allows to master and to manage the complexity of the modem (which presents several hundred configurations available with different kind of parameters) and permits to provide the needed flexibility. The development methodology is based on a C/C++ approach which allows to manage all the parameters at a system level. The architecture coupled to this development methodology offers a high level of modularity which enables to easily modify the waveform only in replacing blocks by other blocks. The efficiency and the flexibility of the modem is then obtained by designing not a single waveform but a waveforms family.**

Key words — Physical layer, high data rate modem, mobile and wireless communication, Software Defined radio (SDR), flexibility, modularity, architecture, high level synthesis, FPGA.

I. Introduction And Application Overview

A. Introduction

The wireless communications revolution started with the mobile phone at the beginning of the 90's and all the improvements, generation after generation, lead to the multiplication of mobile and wireless communications networks and standards. The physical layer has then to support a large variety of complex algorithms which depends on a large variety of parameters. Some of these algorithms are not yet finalized or standardized and can evolve or change. Year after year, new algorithms are also introduced to optimize the communications efficiency and QoS. In the current approach, a waveform redevelopment is needed when a single algorithm parameter or the whole algorithm gets changed. Sometimes the architecture is no more compatible with this evolution and the rework may become significant. To prevent technical issues and then to master cost and delay in the development phases, a fully flexible physical layer is required. It can easily evolve depending on the algorithm types and/or parameters and ensures to reach new productivity gain in the realization of physical layer designs as much as an increased reactivity to time to market. The two fundamental keys to achieve this goal of a fully flexible physical layer are:

- A modular architecture
- A specific development methodology based on C/C++ synthesis methodology.

These themes are detailed in the following chapters.

B. Application overview

The considered professional application is a fully flexible high data rate Software Defined Radio wireless modem. The focus of this paper is put on the waveform part of the communication system's physical layer (refer to the crosshatched part on Figure 1).

The physical layer represents the duplex processing achieved on the signal from (to) the Intermediate Frequency IF to (from) the data bits transmitted to (received from) the protocol layers. In a SDR approach, the platform is unique. It combines the hardware and the Common Operating Environment (COE) deployed on hardware components [1]. The COE represents the drivers, the low level layers, the Sw devices and the services associated to the platform. The transceiver, which converts the signal from (to) the IF to (from) base band, is a part of the COE. On this unique platform, it is possible to implement different kind of waveforms which process the base band signal. In the following pages, the waveform term represents the waveform part of the physical layer.

978-3-9810801-5-5/DATE09 © 2009 EDAA

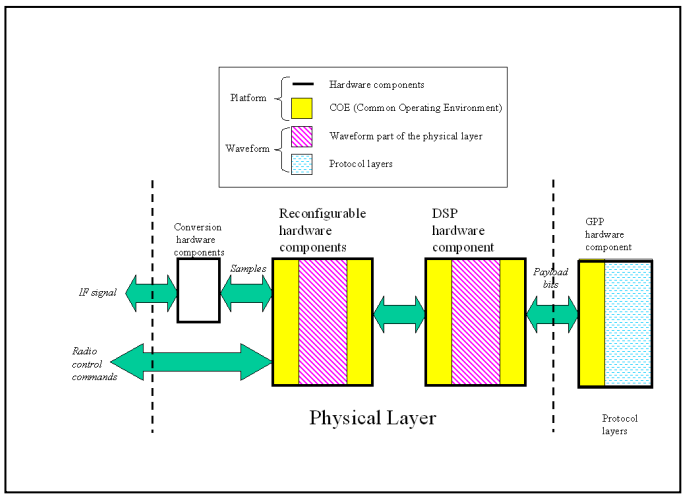

Figure 1. SDR base band overview

The functional breakdown of the considered waveform is described in the following figure:

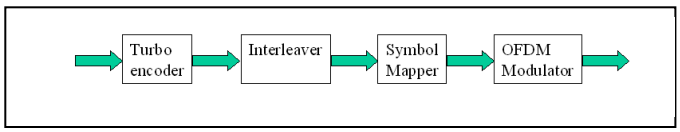

Figure 2. Waveform functional breakdown

The waveform is fully flexible with the possibility to tune a large set of parameters or to change the processing achieved in each block:

- For mobility purposes and to satisfy multi paths requirements, the Orthogonal Frequency Division Multiplexing (OFDM) technique has been chosen. In the framework of others requirements, other approaches like single carrier modulation for example could be considered without changing the architecture. This is possible thanks to the methodology described in the following chapters.
- To adapt the signal bandwidth to different kind of environments, the number of useful sub carriers in the OFDM modulation is programmable till 256 which corresponds to a 5 MHz canalization.
- The supported symbol mapping could be differential and programmable from DBPSK to D16PSK, or coherent like m-QAM mapping.
- The OFDM guard time is also programmable to adapt the trade off between the robustness of the modem in front of the multi paths and the efficiency from the point of view of the data rate.
- Different interleaving schemes are available.
- The fully flexible channel coding is achieved thanks to the use of a Turbo coder for which the block size, the coding rate and the latency (only for the decoder) are programmable.
- All these parameters enable to tune the trade off between the robustness of the communication and the data rate (up to 10 Mbit/s in mobile conditions)

The result of this approach is to obtain a waveform with a full panel of parameters combinations (up to several hundred). This enables the protocol layers to adapt the physical layer and its performances (particularly the trade off between robustness and data rate) to any environment.

II. GLOBAL DEVELOPMENT METHODOLOGY

The challenge in the development process of such type of waveform family is to master its complexity. Full generic system model (as the waveform described in the previous chapter) is often too complex to decline to an implementation model (based on a RTL language) with the same level of flexibility as the initial model. Generally, in the final implementation model, some parameters are fixed to a constant value to simplify the implementation and to reduce the cost / delay of the design task. In this case, when a parameter has to be changed, a rework (often not negligible) is necessary on the implementation model and in some cases the architecture has to be changed too. The issue is the same when an algorithm has to be upgraded or completely changed according to the operational need of the modem use.

The following figure shows a global synopsis of the methodology used to master the complexity of this flexible waveform in the development phase [2].

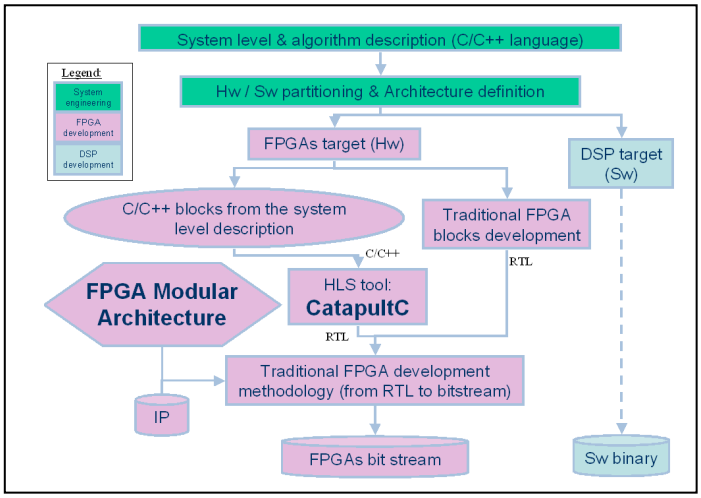

Figure 3. Development methodology overview

The first step is classical and consists in the system engineering to define the system requirements and the algorithms. The algorithms are described in C/C++ language for simulation performances reasons. This is completed by the Hw / Sw partitioning. A first result of this task allows to identify and to size the processing resources. For this waveform implementation, reconfigurable Hw (i.e. FPGAs) and Sw processors (i.e. DSP ideally or GPP) are required. The partitioning result enables to identify the functional blocks which will be mapped on FPGAs or on DSP. With the consideration of needed performances, all signal processing is

mapped on FPGAs and waveform management is implemented in DSP.

The second step concerns the Sw development which enables to obtain the DSP (or GPP) binary file. This is not the focus of this paper, so this part will not be detailed. Note that the Sw development is achieved with now classical the C/C++ system level model [3].

The third step consists in the FPGA development process which includes two powerful lever:

- The FPGA modular architecture that we have developped. It allows to standardize the interface between signal processing blocks
- The High Level Synthesis (HLS) methodology thanks to the CatapultC tool from Mentor Graphics which enables to develop the functional signal processing blocks

The rest of the FPGA development process is classical:

- by including internal or external Intellectual Properties (IPs) if needed
- by developing some RTL blocks with a traditional method if needed
- and with a design flow to convert a RTL model to a downloadable FPGA bit stream file.

III. FPGA MODULAR ARCHITECTURE

To avoid sequencing and/or interface problems which can occur when a modem functionality is modified, added or removed during a modem upgrade, the architecture of the Modem FPGA shall be modular, fully flexible, highly parameterizable and shall support the following constraints:

- the modules implementing modem functionalities are commutable and deal with in-the-fly parameters.
- Every elementary modem functionality corresponds to one basic component,
- The data flow interface of every basic signal processing module is 'standardized', i.e. all the basic signal processing modules which compose the modem have the same data flow transfer protocol,
- System sequencing is distributed inside all the basic modules. There is no global sequencing.

Three specific areas of the system architecture will be detailed: the Modem FPGA architecture, the transmitter and receiver architecture and FPGA distributed sequencing.

A. Modem FPGA architecture description

The Modem FPGA architecture is described in Figure 4.

As it is described in the Figure 1, COE belongs to the platform part and the waveform (WF) component implements waveform capabilities of the modem. The WF component is composed of four basic blocks: the WF interface, the Transmitter, the Receiver and the Time Manager.

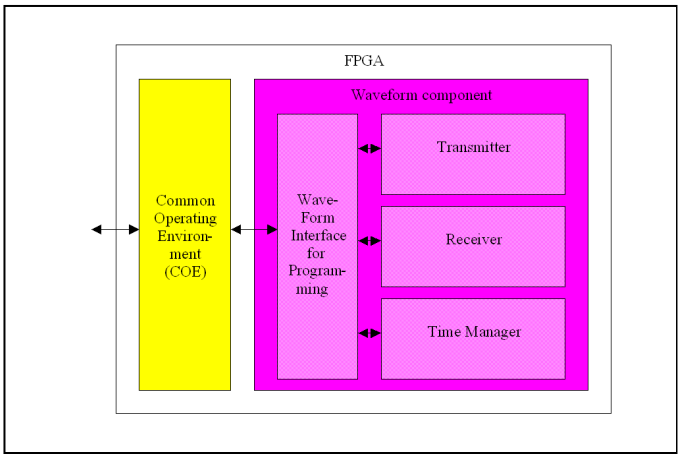

Figure 4. General architecture of the Modem FPGA

The WF interface is the interface between the COE and the modem physical layer functions (Transmitter, Receiver and Time Manager). This module is used every time the others blocks communicate between them. It's the only block where cross-clock domains data transfers occur.

The three last modules deal with their own clock domain and implement the following modem features:

- Time management for the protocol access to the physical layer (Time Manager).
- Transmit treatments of the modem physical layer (Transmitter).
- Receive treatments of the modem physical layer (Receiver).

The main architecture feature of the Transmitter and the Receiver is that they are made up of a set of several basic signal processing modules.

B. Transmitter and Receiver architecture

Thanks to the breakdown of the waveform, the architecture of the Transmitter and Receiver is divided into a set of basic modules which treat bits, bytes, symbols or samples packet by packet (a packet is a set of several data) and which implement elementary signal processing function.

Thus, the Transmitter and the Receiver are composed of a chain of basic modules which exchange data between them. The Transmitter architecture (the Receiver one is the same) is outlined in Figure 5:

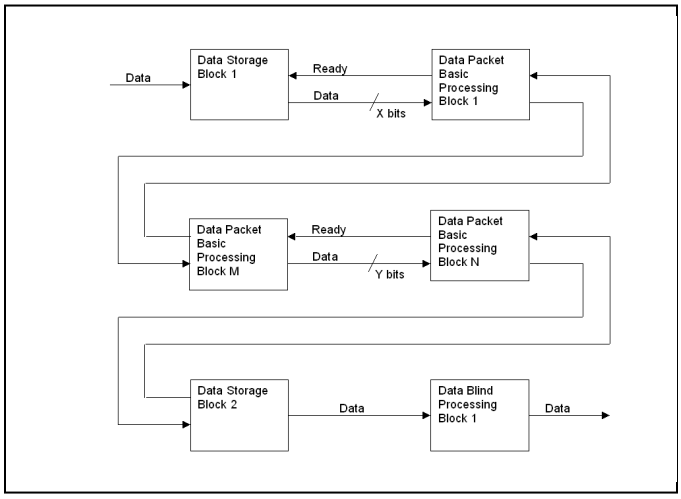

Figure 5. Architecture of the modem Transmitter

The Transmitter and the Receiver are composed of a set of three kinds of blocks:

- Data Storage Block: this module stores data at the incoming rate into a memory without data loss. The memory shall be well-proportioned by considering the input and output data rates of the buffer.
- Data Blind Processing Block: this module performs blind processing on incoming data. The input and output rates of the block can not be controlled by itself, i.e. the component shall accept incoming data whose rate is set by the upstream module.
- Data Packet Basic Processing Block: this module implements elementary data processing and its data flow interface is 'standardized' (all these blocks have the same data transfer protocol). Furthermore, this module can control the rate of incoming data by indicating its availability to the upstream block. At last, the component is made up of two sub-components: a local sequencer and a basic processing core.

C. Transmitter and Receiver sequencing

There is no global sequencing inside the Transmitter and Receiver. Sequencing is distributed inside all the basic modules. Furthermore, sequencing of the Transmitter and Receiver is naturally performed by cascading Data Packet Basic Processing Blocks.

Thus, transfer of data packet between blocks occur every time that a data packet is available inside the upstream block and that the downstream block is ready to perform its own data treatment.

At last, the Transmitter blocks receive all their parameters in parallel of the data interface. Module Parameters propagation from blocks to blocks is performed by sampling them every time a new data packet is received at the interface of the component.

IV. THE HIGH LEVEL SYNTHESIS (HLS) METHODOLOGY

The previous chapters have described how the waveform was split into elementary blocks and how these blocks were interfaced and sequenced together. This chapter explains now the efficient and flexible methodology to implement functional algorithm in reconfigurable Hw target.

The High Level Synthesis (HLS) methodology is based on the CatapultC tool from Mentor Graphics. It enables from algorithmic un-timed ANSI C/C++ to generate automatically synthesizable RTL to FPGA targets (ASIC targets are available too). The tool offers the following characteristics:

- an automatic RTL generation (to speed up FPGA implementation especially for signal processing blocks),
- an automatic SystemC test bench generation (to speed up FPGA functional validation)
- an help to the micro architecture choice. This is also called Design Space Exploration and refers to an automatic process which enables to reach for the implementation of a function the best trade off between the complexity (in term of logic elements or equivalent) and the performance (in term of maximum frequency).

HLS methodology and CatapultC are used to develop functional signal processing blocks in the modular architecture of the waveform previously described. They provide a significant gain in term of productivity, flexibility, parameterization and reusability because developing signal processing blocks combines two types of competences: algorithm skill and implementation skill. The tool provides an homogeneous environment which associates closely the system level model description in C/C++ and the implementation process in a co-design framework. So the iteration loop between algorithm and implementation tasks is sensitively reduced especially when the algorithm has to evolve or when a parameter has to change.

The fact is that in this type of flexible, highly configurable and complex waveform, RTL traditional development is reaching its limits because of the number of parameters to manage at the implementation model level. So evolution of the waveform are often really complex and costly. With HLS methodology, it is possible to manage all parameters directly from the C/C++ model (i.e. at the system level) which is simpler and more efficient. Moreover, the combination of the C++ language and the Algorithmic C types (AC_types from Mentor Graphics) provides a powerful tool to specify a wide kind of algorithmic families. C++ language is used for the algorithmic specification and the power of the templates enables parameterization and flexibility. AC_types are used for two main reasons:

- They enable to achieve optimizations (for example in term of data path length)
- They provide some system level simulation facilities

The C++ model becomes then the reference (i.e. the golden model) even if the RTL model is kept. It is used as an

intermediate model in the process of the bit stream building. The C++ model becomes the unique system engineering model used to create/validate, implement and verify the algorithm. The risk of inconsistency between the algorithm and the implementation model is then significantly reduced.

V. CONCLUSION

To meet the flexibility and the level of parameterization required by new SDR waveforms, it is mandatory to introduce innovative approach based on a modular architecture and a HLS development methodology. The architecture provides a standardized framework which normalizes the interfaces (with in-the-fly parameterization capacities) and the sequencing between the elementary signal processing blocks. This enables block by other block substitution with no architectural impact. The value of the HLS development methodology is to provide a homogeneous link between the algorithm C/C++ description of the elementary signal processing blocks and its implementation in FPGA target. All the parameters are then managed at the algorithm level which is more efficient for complex waveforms and also in case of evolutions. The combination of these two approaches (architecture and HLS development methodology) enables to master the complexity of new type of applications in the initial development but also in future optimizations and modifications phases.

Not only a single waveform development, the global methodology, described in this paper, allows to design a waveforms family. It provides moreover a specific waveform development environment which is useful for all new applications.

REFERENCES

[1] "Mapping Semantics of CORBA IDL and GIOP to Open Core Protocol for Portability and Interoperability of SDR Waveform Components", G.Gaillard and al. Thales and ETIS University, DATE March 2008 Munich.

[2] "Applications and drivers in Aerospace, Defence and Security for Programmable Hardware Systems" B.Candaele Thales, IET Conference on Programmable Hardware Systems, September 2008 London.

[3] "A Design Flow for Architectural exploration and implementation of partially reconfigurable processors" K.Kakuri and al. Aachen University, IEEE Transactions on VLSI, october 2008.