We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

15

Software Defined Radio Platform for Cognitive Radio: Design and Hierarchical Management

Amor Nafkha, Christophe Moy, Pierre Leray,
Renaud Seguier and Jacques Palicot
SUPELEC/IETR, Avenue de la Boulaie,
Cesson Sévigné Cedex,
France

1. Introduction

Cognitive Radio (CR) Mitola (2000) is a promising technology to improve spectrum utilization of wireless communication systems. Current investigations in CR have been focused on the physical layer functionality. The cognitive radio, built on a software-defined radio, assumes that there is an underlying system hardware and software infrastructure that is capable of supporting the flexibility needed by the cognitive algorithms. As already foreseen by Mitola Mitola & Maguire (1999), a Cognitive Radio is the final point of Software Defined Radio (SDR) platform evolution: *a fully reconfigurable radio that changes its communication functions depending on network and/or user demands*. Mitola's definition on reconfigurability is very broad and we only focus here on the reconfigurability of the hardware platform for Cognitive Radio. SDR basically refers to a set of techniques that permit the reconfiguration of a communication system without the need to change any hardware system element. As explained in the schematic of figure 1, this relies on a cognitive circle. Figure 1 (a) is from Mitola (2000) and figure 1 (b) is a simplified view of the cycle summarized in three main steps:

- Observe: gathers all the sensing means of a CR,
- Decide: represents all that implies some intelligence including learning, planning decision taking,
- Adapt: reconfigures the radio, designed with SDR principles, in order to be as flexible as possible.

The figure 2 draw the general approach that can help the radio to better adapt its functionality for a given service in a given environment without restriction on the sensors nature.
Sensors are classified in function of the OSI layers they correspond to, with a rough division in three layers. Corresponding to the lower layers of the OSI model, we find specifically all the sensing information related to the physical layer: propagation, power consumption, coding scheme, etc. At the intermediate level are all the information that participate to vertical handover, or can help to make a standard choice, as a standard detection sensor for instance. The network load of the standards supported by the equipment may also be of interest. It also includes the policies concerning the vicinity, the town or the country. The highest layer is related to the applications and all that concerns the human interaction

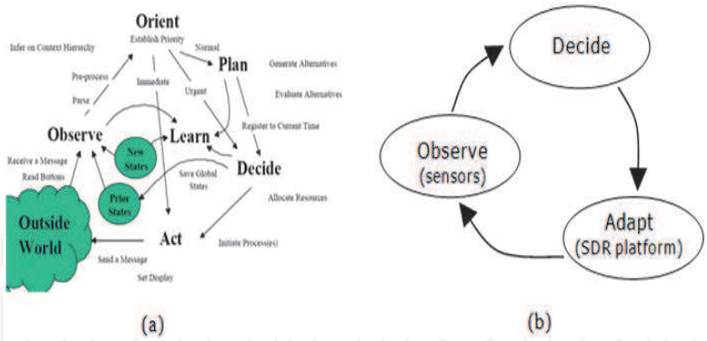


Fig. 1. (a) Mitola's cognitive cycle, (b) simplified version

Sensors	Layer	Literature concepts
User profile (price, personal choices) Localization, sound, video, position, speed, security.	Application	Context Aware
Intra-network, and inter-network vertical handover , standards, load	Transport Network	Interoperability Ambiant networks
Access mode, power modulation, coding, Frequency, handover. Channel Estimation	Data link Physical	Link adaptation

Fig. 2. Simplified OSI model for cognitive radio context

with the communicating device. It is related to everything that concerns the user, his habits, preferences, policies, profile. If a user has the habit to connect to a video on demand service every evening while coming back home from office by metro, a CR terminal should be aware of it to plan all the requirements in terms of battery life, sufficient quantity of credit on his contract, vertical handover succession depending on each area during the trip, etc. The equipment can be aware of its environment with the help of sensors like microphone, video-camera, bio-sensors, etc. As we are at the early beginning of such technology, it is difficult to foresee all the possibilities. We can think, for instance, that user's biometric information and/or facial recognition will ensure equipment security. Video-camera could also be used to indicate if the terminal is outside or inside a building. This may impact propagation features, but also the capability or not to receive GPS signals. Another example could be given in the context of video conferencing, a separation between the face of the speaker and the background could help decreasing the data rate while refreshing slowly the background of the image Nafkha et al. (2007).
Note that this classification is also related to three well-known concepts of the literature:

- Context aware for higher layers Chen & Kotz (2000),
- Interoperability for intermediate layers Aarts et al. (2001),
- Link adaptation for lower layers Qiu & Chuang (1999).

All this may be combined to achieve cross-layer optimizations. This is one of the responsibilities of the cognitive engine in our mind. However, due to the high financial pressure on spectrum issues, CR is often restricted in the research community to spectrum management aspects as in Fan et al. (2008); Ghozzi et al. (2006). Opportunistic spectrum

access approaches are explored to increase the global use of the spectrum resources. FCC has been already opening the door for several years, in the TV broadcasting bands, and permits secondary users (e.g. not licensed) to occupy primary users spectrum when available.
More futuristic CR scenarios may also be considered concerning the spectrum management. We may even imagine in the very long term a fully deregulated spectrum access where all radio connections features would be defined on-the-fly: carrier frequency, modulation, data rate, coding scheme, etc. But this means also to overcome regulatory issues in addition to technological challenges.

2. Background and related work

The objective behind this section is to highlight other cognitive radio platforms and to give our architecture purpose.

2.1 Related work

There are a large number of experimental SDR platforms that have been developed to support individual research projects. The various experimental SDR platforms have made different choices in how they are addressed the issues of flexibility, partitioning and application. To highlight the variety of architectures, five popular platforms will be discussed briefly prior to introducing our platform.

- *NICT SDR Platform*: The Japanese National Institute of Information and Communications Technology (NICT) constructed a software defined radio platform to trial next generation mobile networks. The platform has two embedded processors, four Xilinx Virtex2 FPGA and RF modules that could support *1.9* to *2.4* and *5.0* to *5.3* GHz. The signal processing was partitioned between the CPU and the FPGA, with the CPU taking responsibility for the higher layers. An objective of this platform was to explore selection algorithms to manage handover between existing standards. To this end, a number of commercial standards were implemented, for example *802.11a/b/g*, digital terrestrial broadcasting, *WCDMA* and a general *OFDM* communication scheme.
- *Berkeley Cognitive Radio Platform*: This platform is based around the Berkeley Emulation Engine (BEE2) which is a platform that contains five high-powered Xilinx Virtex2 FPGAs and can connect up to eighteen daughter-boards. In the Cognitive Radio Platform radio, daughter-boards have been designed to support up to *25* MHz of bandwidth in an *85* MHz range in the *2.4* GHZ ISM Band. The RF modules have highly sensitive receivers and to avoid self-generated noise operate either concurrently at different frequencies (FDD) or at the same frequency in a time-division manner. This cognitive radio platform requires only a low-bandwidth connection to a supporting PC as all signal processing is performed on the platform.
- *Kansas University Agile Radio (KUAR)*: The KUAR platform was designed to be a low-cost experimental platform targeted at the frequency range *5.25* to *5.85* GHz and a tunable bandwidth of *30* MHz. The platform includes an embedded *1.4* GHz general purpose processor, Xilinx Virtex2 FPGA and supports gigabit Ethernet and PCI-express connections back to a host computer. This allows for all, or almost all processing to be implemented on the platform.

- *OpenAirInterface*: The mobile communications department at EURECOM proposed an open-source hardware/software development platform and open-forum for innovation in the area of digital radio communications. OpenAirInterface implements in software the Physical and Medium-access layers for wireless communications as well as providing a IPv4/IPv6/MPLS network device interface under Linux. The initiative targets 4^{th} generation wireless systems (UMTS Longterm-evolution (LTE), 802.16e/j) and rapidly-deployable MESH networks using a similar radio interface technologies. The development can be seen as an open-source testbed for advanced algorithmic prototyping and performance evaluation.
- *Universal Software Radio Peripheral (USRP)*: The USRP is one of the most popular SDR platforms currently available and it provides the hardware platform for the GNU Radio project. The first USRP system, released in *2004*, was a USB connected to a computer with a low-performance FPGA. The FPGA was used primarily for routing information but also allowed some limited signal processing. The USRP could realistically support about *3* MHz of bandwidth due primarily to the performance restrictions of the USB interface. The second generation platform was released in September *2008* and utilizes gigabyte Ethernet to allow support for *25* MHz of bandwidth. The system includes a medium range Xilinx Spartan3 device which allows for a local processing. The radio-frequency performance of the USRP was limited and is more directed towards experimentation rather than matching any communications standard.

Our proposed platform has been developed in order to achieve high flexibility and reconfigurability of the wireless baseband processing. For the hardware part, for example, we exploited the ability to reconfigure partial areas of an FPGA anytime after its initial configuration. Our development concerns all processing blocs: from the video treatment to the intermediary frequency signal generation. Our intention is not to develop any commercial platform, but just to test and verify our approach to achieve baseband flexibility using:

- Partial Reconfiguration Nafkha et al. (2007) and Common Operator Alaus et al. (2008).
- Hiearchical Reconfiguration Management Delahaye et al. (2005).
- Hierarchical and Distributed Cognitive Radio Architecture Management Godard. (2009).

2.2 The proposed solution

The proposed solution is a design approach and not a hardware platform itself so that it is not restricted to a specific hardware platform. It intends to answer the design issue of SDR in the following context:

- flexible processing including partial FPGA reconfiguration and Common Operator approach.
- heterogeneous processing, including processors (GPP, DSP), FPGA and ASICs,
- portability from a HW device target to another.

In order to cope with these characteristics, a modular-based approach is privileged. This is the main support of flexibility. It permits indeed to separate the radio application into sub-pieces that can be split in any sub-set depending on the HW devices that compute their processing needs. This also favorites changes in the repartition of the processing modules on the HW devices. As all processing modules are designed independently in a modular-based approach,

this also guarantees the non dependence of processing modules in terms of operating rhythm. One can just not make them run faster than their fastest speed, but anything lower is compatible.
This is very straightforward in a processor environment as the processing modules varies with the processor frequency (or it architecture after compilation). But this is generalized to the reconfigurable HW world while using Globally Asynchronous Locally Synchronous (GALS) principles. It turns HW processing as SW in the sense that the exchanges between processing modules are asynchronous from the data rate they have to process. The consequence is that these processing modules can be ported to several designs at different speeds, with no dependence with the speed of other blocks. Another major effect is that it becomes transparent to replace a SW processing module, e.g. running in a processor, by a HW processing module, e.g. running in a FPGA, and vice versa. Moreover, a HW processing module can be easily moved to a processor instantiated inside a FPGA (such as a NIOS for Altera or a MicroBlaze for Xilinx) without reconsidering the global behavior of the processing modules it is connected to.
This design approach is completely compatible with an Intellectual property (IP) oriented design strategy. Re-usability has several major advantages: gains of time at all development stage, debug and validation stages, and integration stage. It permits also to benefit from third party expertise to speed-up or complete the proprietary designs. To sum-up the proposed solution consists in declaring rules for the design of IPs or processing modules so that they can be easily assembled in the design framework that is detailed below.

3. System structure

The presented real-time platform provides a simple wireless video stream broadcasting system to verify and test our approach. It consists of one transmitter as the base station and one receiver as the terminal. The system architecture is depicted in figure 3. Basically, the transmitter and receiver hosts can communicate and exchange their data through an existing TCP/IP networks or Intermediate Frequency (IF) link. The transmitter host utilizes USB port to communicate with the video camera. At the receiver side, any standard display monitor allows us to display the incoming video stream.

3.1 Hardware architecture

The digital hardware setup of the whole system is based on Sundance modules. The transmitter and receiver side contain a Sundance SMT310Q carrier boards, plugged via Peripheral Component Interconnect (PCI) bus to a standard PC. The hardware architecture is depicted in figure 4.
At the transmitter side shown at figure 5, the Sundance SMT310Q carrier-board is used to carry the processing modules (SMT395, SMT348 and SMT350 ADC/DAC) in the four available TIM-40 slots. The Sundance SMT395 module is placed in the first TIM-slot and controls the operation of other modules. It consists principally a Texas Instruments (TI) 6416T fixed-point Digital Signal Processor (DSP) running at 1 GHz, a Xilinx Virtex II Pro FPGA, and two Sundance High-speed Bus (SHB, up to 400MB/s) for fast data exchange with the other modules. In our platform the DSP is used as a control device for the ADC/DAC and memory modules and to set the parameters for the pre-distortion filter running in real-time on the FPGA at the module SMT350. Based on the Xilinx Virtex4 range, the SMT348 features 16MB

of blistering fast QDRII memory, ensuring ample capacity to develop todays demanding applications. The SMT348 includes SHB and SLB (Sundance LVDS Bus) interfaces. It provides quick and easy connection to rapid ADC and DAC modules for data acquisition or software radio systems. The SMT350 module, is composed of:

- Two DACs DAC5686 from Texas Instrument with 16 bits of resolution and a maximum sampling rate of 500MSPS with interpolation filters
- Two ADCs ADS5500 from Texas Instrument with 14bits of resolution and a maximum sampling frequency of 125MHz
- A CDCM7005 from Texas Instrument which provides individual sample frequency to each converters

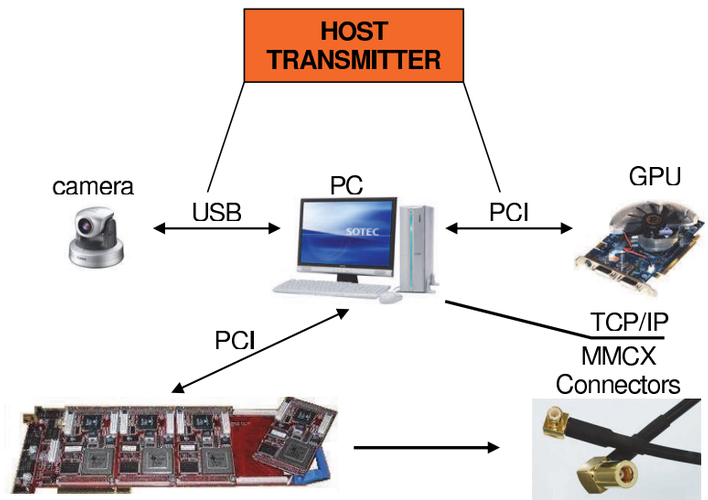


Fig. 3. Hardware Architecture

The stream server program encapsulates the video data into Internet Protocol (IP) stream and saves the IP stream in the buffer allocated in the main memory of the host PC. The DSP module fetches the data in the buffer through the PCI interface provided by the above mentioned carrier board and then executes the partial part of the digital baseband and Intermediate Frequency (IF) signal processing algorithms of the transmitter. The driver of the carrier board offers the DSP module the methods to access the main memory of the host PC through PCI interface by providing C/C++ Application Program Interface (API) functions. The Xilinx FPGA on the DSP module takes care of the Sundance High speed Bus (SHB) interfacing between the DSP module SMT395 and the FPGA module SMT348. The SHB interface is able to transfer 32-bit data at a 100 MHz clock. Via SHB the digital IF signal is forwarded to the SMT350 to generate the analog signal using its integrated Digital to Analog Converter (DAC). The analog IF signal goes through the low-pass filter.

The hardware setup of the receiver is similar to the transmitter, as shown in figure 5. In this case, the SMT350 is configured as an Analog to Digital Converter (ADC) module and the signal experiences the reciprocal of the transmitter. The IF signal coming from the transmitter is sampled synchronously by the ADC on the SMT350 module. The FPGA module SMT348 receives the digital samples from the SHB interface and accomplishes a high parallel part of

digital signal process algorithms of the receiver. The simplest part of the baseband process is sent to the SMT395 module via the SHB.

The final received IP packets are saved to the buffer in the main memory of the host PC through the PCI interface. The network layer program fetches the IP packets from the buffer and emits them to the certain IP port by IP socket programming. The video stream player always listens to the IP port and plays the video back.

In both side, transmitter/receiver, the testbed platform contains a Graphics Processor Unit (GPU). The main reason behind is that the GPU is specialized in compute intensive, highly parallel computation and therefore is designed in such a way that more transistors are devoted to data processing rather than data caching and flow control.

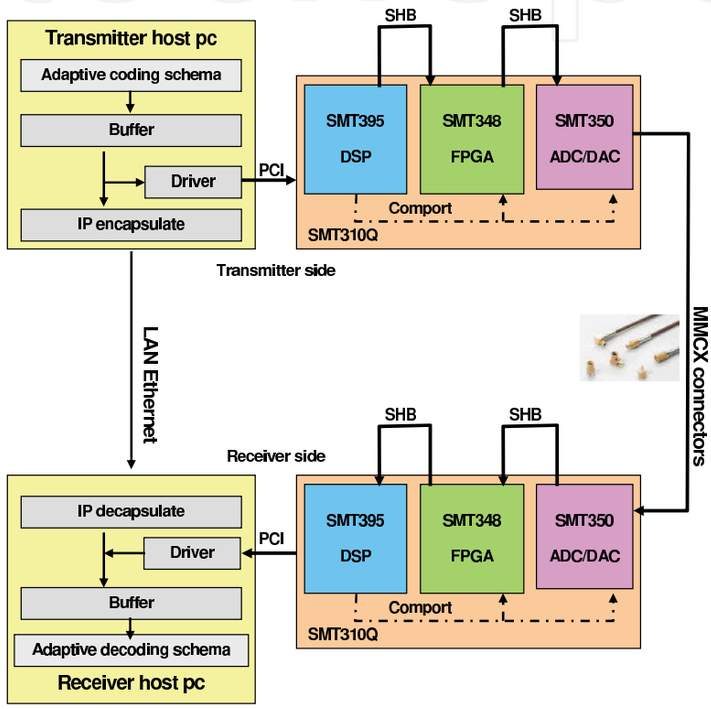


Fig. 4. Transmitter and receiver overview

3.2 Software architecture

The two host stations are a standard personal computers running Microsoft Windows XP and Microsoft Visual C++. Several hardware and software tools, as depicted in figure 6, are necessary for the completion of our testbed. These tools include the physical DSP/FPGA and their associated development board that allowed for continual reprogramming of test systems as well as many features for data storage and output display. Xilinx has also supplied a suite of tools that are used in our platform. These Xilinx software tools are used for developing the hardware and software aspects of the system. Although many of these tools have included documentation from Xilinx, their support of partially reconfigurable systems is currently somewhat lacking. Therefore, the integration of these tools into a working tool flow to achieve the goal of a partial reconfiguration for example required research from numerous sources and some experimentation with the tools. For the partial reconfiguration implementation, we need the following Xilinx tools:

- *Xilinx EDK* provides a framework for design of hardware/software components of the embedded processor systems on programmable logic. Appropriate tools for each stage of the design in addition to IBM PowerPC and Xilinx MicroBlaze processor cores infrastructure and peripheral IP cores facilitate hardware/software partitioning and design reuse.
- *Xilinx ISE* allows for complete FPGA development. It can automatically interpret the HDL syntax, synthesize the description, place and route the logic elements and then provide a software BIT file description to connect these logic elements together to create the circuit described in the HDL. All these tool flow steps require their own respective application program to perform the function.
- *Xilinx PlanAhead* is a floor-planning tool provided by Xilinx to allow developers flexibility on how there synthesis designs should be placed on the FPGA floorplan. This tool is useful in ASIC designs where locations of logic elements are an important factor to the performance of the application. For the scope of our platform, PlanAhead has partial reconfiguration options which make it a required tool in the partial reconfiguration tool flow.

The programming of the TI 6416T fixed-point DSP is achieved using the Code Composer Studio (CCS) Integrated Development Environment (IDE). The CCS IDE allows a user to connect, program in C, and run the DSP through a graphical user interface. furthermore, a user is able to view the memory contents of the DSP and profile the execution time for pieces of their code all in real-time. For high level synthesis of the digital signal process algorithms, we use the SynDEx tool environment Grandpierre et al. (1999) which provides a formal framework based on graphs and system-level computer-aided design (CAD) software. On the one hand, this tool specifies the functions of the applications, the distributed resources in terms of processors and/or specific integrated circuits, and communication media. On the other hand, it assists the designer in implementing the functions onto the resources while satisfying timing requirements and, as far as possible, minimizing the resources. The results is a real-time behavior of the application functions executed on various resources, like processors, integrated circuits or communication media. For the software part of the application, code is automatically generated as a dedicated real-time executive.
As a result of demand for video treatment at the both side (transmitter/receiver), the CUDA programming model is used. It is ANSI C extended by several keywords and constructs. The GPU is treated as a co-processor that executes data-parallel kernel code. The user supplies a single source program encompassing both host (CPU) and kernel (GPU) code. Each CUDA program consists of multiple phases that are executed on either the CPU or the GPU. The phases that exhibit little or no data parallelism are implemented in host (CPU), which is expressed in ANSI C and compiled with the host C compiler. The phases that exhibit rich data parallelism are implemented as kernel functions in the device (GPU) code. A kernel function defines the code to be executed by each of the massive number of threads to be invoked for a data-parallel phase. These kernel functions are compiled by the NVIDIA CUDA C compiler and the kernel GPU object code generator. There are several restrictions on kernel functions: there must be no recursion, no static variable declarations, and a non-variable number of arguments. The host code transfers data to and from the GPU's global memory using API calls. Kernel code is initiated by performing a function call.

SynDEx tools Grandpierre et al. (1999) provides a formal framework based on graphs and system-level software. On the one hand, these specify the functions of the applications, the distributed resources in terms of processors and/or specific integrated circuit and communication media, and the non-functional requirements such as real-time performances. On the other hand, they assist the designer in implementing the functions onto the resources while satisfying timing requirements and, as far as possible, minimizing the resources. This is achieved through a graphical environment (see figure 5), which allows the designer to explore manually and/or automatically the design space solutions using optimization heuristics. Exploration is mainly carried out through timing analysis and simulations. The results of these prediction's is a real-time behavior of the application functions executed on various resources, like processors, integrated circuits or communication media. This approach conforms to the typical hardware/software co-design process. Finally, for the software part of the application, code is automatically generated as a dedicated real-time executive.

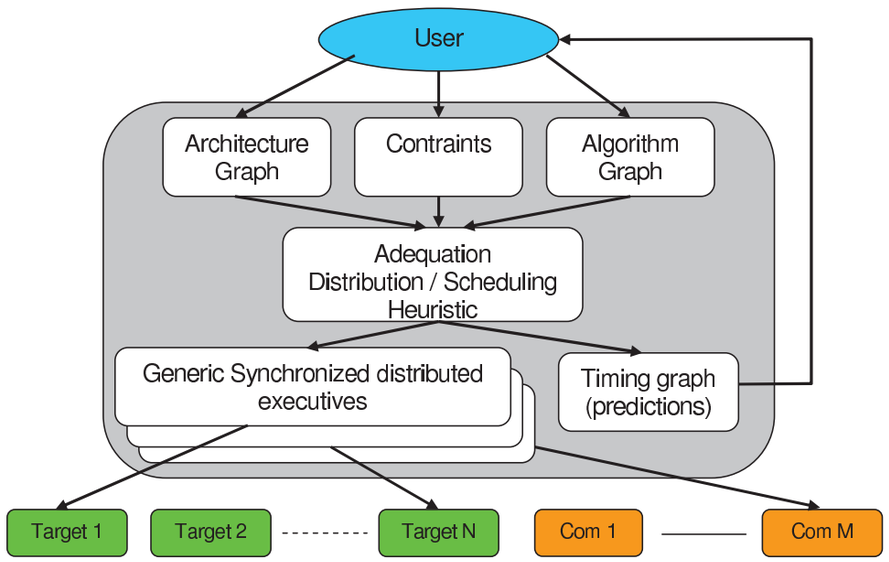


Fig. 5. SynDEx utilization global view

4. Hardware development

Even though the prototyping effort is focused on an FPGA-based design, we are also exploring the architectural benefits of custom integrated circuitry, primarily related to power consumption and the silicon area, which are important performance parameters for hardware designs used in mobile/portable platforms. The approach we have chosen to take involves identifying the hardware architecture appropriate for low-power configurable design based on heterogeneous blocks (i.e. blocks that are highly optimized for a particular function, yet flexible enough to support a variety of configuration parameters) as a compromise for the trade-off between programmability and power consumption/area. In addition to fast prototyping, the additional benefits of using modern FPGAs (e.g. Xilinx Virtex 4) are the availability of highly optimized features implemented as non-standard configurable logic blocks (CLB) like phase-locked loops, low-voltage differential signal, clock data recovery, lots of internal routing resources, hardware multipliers for DSP functions, memory, programmable

I/O, and microprocessor cores. These advantages simplify mapping from hierarchical blocks to FPGA resources.

4.1 Common operator

At the foundation of our study is the 'common operators' technique 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 technique 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 operator technique belongs to the parameterization techniques firstly proposed by Jondral et al. (2002). The common operators' technique is discussed more extensively in Alaus et al. (2008).
A part of the theoretical approach of this technique is presented in Rodriguez et al. (2007). Two different parameterization techniques have been proposed in the literature:

- The Common Function (CF) Technique consists in seeking an optimized generic function (the expected common function) like coding, mapping, among others which can replace the initial task present in a predefined set of standards. This Common Function (CF) technique was historically the first one proposed by several articles from Karlsruhe University (Germany) Jondral et al. (2002), Rhiemeier (2002)
- The Common Operator (CO) technique claims to be independent of the standards by finding the smallest set of highest-level operators like MAC, FFT, etc., which are used by the maximum functions number. It is an open technique. The foundation paper was Palicot & Roland (2003) which identified the FFT as a common operator

The CO technique is implemented in our platform in order to optimize both the area and the reconfiguration time on the FPGA.These operators are very small regards to the needed bit-stream, therefore the time to reconfigure a function using an operator is very small too. These operators are managed by the lowest level of our hierarchical reconfiguration manager(see section 5). This manager level is very close to the operator itself and in most cases embedded in the same resource. Furthermore, thanks to the partial reconfiguration approach, it is very easy to modify either a complete specific operator or parameters of an operator, providing a huge gain in reconfiguration time.

4.2 Partial Dynamic Reconfiguration

Partial Dynamic Reconfiguration is the capability to reconfigure specific areas of the FPGA at run-time after its initial configuration. PDR is carried out to allow the FPGA to adapt to changing hardware algorithms, improve resource utilization, to enhance performance or to reduce power consumption. In March of 2006, Xilinx introduced the early access partial reconfiguration flow along with the introduction of slice based bus macros which are pre-routed intellectual property (IP) cores. The restriction of full column modular PDR was removed allowing reconfigurable modules of any arbitrary rectangular size to be created. The EAPR flow also allows signals from the static regions to cross through the partially reconfigurable regions via the bus macros. Using the principle of glitch-less reconfiguration, no glitches will occur in signal routes as long as they are implemented identically in every reconfigurable module for a region. The only limitation of this approach is that all the partial bit-streams for a module, to be executed on a reconfigurable region, must be predetermined.

The Virtex-II and the Virtex-II Pro are the first Xilinx architectures that support Internal configuration access port (ICAP) Blodget et al. (2003) which is a subset of the Xilinx SelectMAP interface having fewer signals because it only deals with partial configurations and does not have to support different configuration modes. For Virtex-II and Virtex-II Pro series, the ICAP furnishes an 8 bit input data bus and an 8 bit output data bus while with the Virtex-4 Series, the ICAP interface has been updated with 32 bit input and output data buses to increase its bandwidth. The ICAP allows the internal logic of the FPGA to reconfigure and to read-back the configuration memory. With combination of either a hard or a soft microprocessor as a controller, dynamic reconfiguration is carried out through the ICAP interface Blodget et al. (2003)

We consider two possible scenarios for dynamically reconfiguring the partial reconfiguration modules: *exo-configuration* and *endo-configuration* as shown in figure 8. The exo-configuration constitutes the traditional way to configuring an FPGA. A configuration bit-stream is controlled by an external processor like DSP. In this way, new modules, or upgraded versions of them, can be created and used at any moment. This approach exhibits upgrade-ability, but the platform is totally dependent on the processor for modifying its function. The endo-reconfiguration, also known as self-reconfiguration, considers a different scenario. An FPGA reconfigures itself using its own local resources. The platform is thus totally independent as it does not require an external source to provide a bit-stream and to decide whether to self-reconfigure. The main draw-back is that partial bit-streams need to be previously generated by a host computer. This approach benefits, thus, of an autonomous reconfiguration with very limited upgrade-ability.

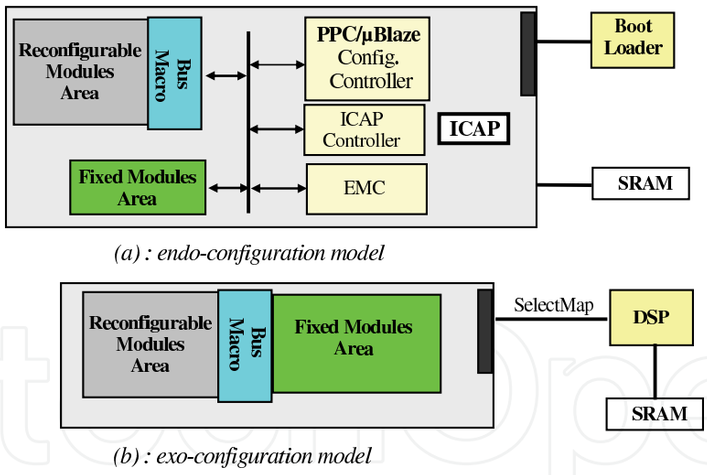


Fig. 6. Exo and Endo FPGA configuration

Our Platform supports both of the previous cited configuration techniques. In fact, at boot time, the initial full configuration bit-stream file is sent to the FPGA. This file includes an internal configuration controller, the internal reconfiguration interface, the initial instantiations of processing bloc units. Depending on the granularity level. One stays internal to the FPGA in case of limited-scale reconfiguration (for co-accelerators configuration) or design parameterization. This implies to interconnect the Micro-blaze to the ICAP internal configuration interface. In this case, small partial bit-streams can be stored inside the FPGA, and the use of endo-configuration lets free the other HW resources of the platform. At a

larger scale, configuration for the HW accelerator is external. This implies to interconnect the DSP to the external SelectMap or internal ICAP reconfiguration interfaces. The bit-stream corresponding to the design of HW accelerators are stored in an external SRAM memory.

5. Software development

The great diversity of processing types in a multi-standard application implies a large number of processing configurations to be managed. The configuration management is complex and we believe that a hierarchical approach of configuration control and management could simplify it Delahaye et al. (2005). We combine two configuration features, as presented in section 5.1 and 5.2, in order to create the complete configuration framework for CR testbed.
The proposal of a hierarchical view enables to manage multi-granularity of configurations, which is of particular interest for heterogeneous architectures. The proposed model is composed of three levels of hierarchy detailed in figure 9. A system architecture compliant with this functional model includes one Configuration Manager Unit at level 1 (L1_CMU), several Configuration Manager Units at level 2 (L2_CMU), each of them being responsible for one or several Configuration Manager Units of level 3 (L3_CMU), which directly manage the processing components.

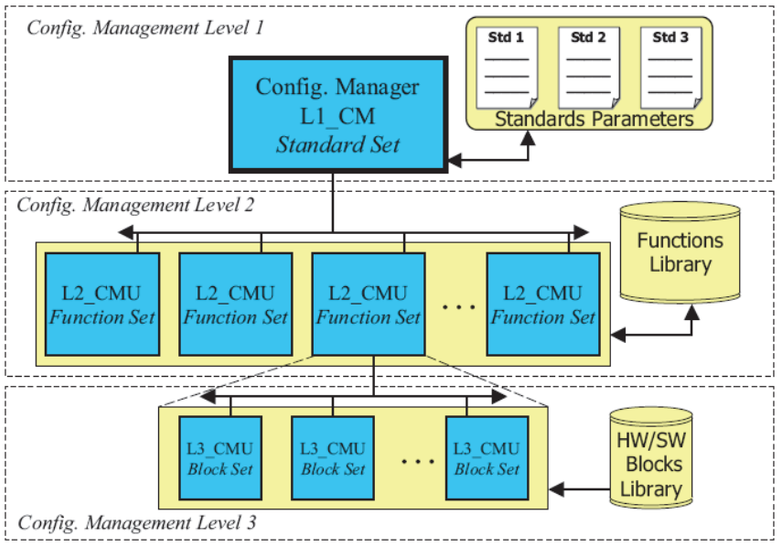


Fig. 7. Hierarchical model of configuration management

5.1 Configuration data-path

As the communication applications are data-flow oriented Delahaye et al. (2005), our approach is based on a data-path model. The functions of the baseband blocks chain are mapped into several Processing Block Units (PBU). Each PBU is optimized using specific reconfigurable hardware resources. In addition, a configuration path, also split into several configuration manager units, controls the reconfigurable processing path. Each CMU, dedicated to a type of PBUs, manages the configuration of a type of baseband function in the chain. The split configuration path offers the possibility to partially reconfigure the baseband chain by an independent reconfiguration of each PBUs.

5.2 Hierarchical management

The hierarchical configuration management model presented in Delahaye et al. (2005) is based on the configuration data path approach. This model is necessary to manage the multi-granularity of configuration required by the different contexts. It is composed of three levels of hierarchy that are detailed below:

- **level 1**: This first high level classification allows a control of category-specific functions to manage parameters at the highest level. The L1_CMU works at the standard level as a host towards the underlying levels of management. This entity is in charge of choosing the functional units which will constitute the entire configuration of the baseband processing chain. At this level, generic functions are handled as generic components. Any hardware implementation is not yet considered.
- **level 2**: The generic functions selected at level 1 are parameterized at the middle level in accordance to standard specifications. The set of attributes of each function is handled by the L2_CMU in order to create each functional context of the entire processing chain.
- **level 3**: The processing data path architecture at this lowest level depends on the reconfigurable computing resources of the hardware architecture. The main task of the L3_CMU in the configuration path is to find the available processing resources and configure them to enable the execution of the functional context created at the middle level.

As presented in figure 8, an hierarchical configuration management is proposed to map processing elements. The CPU and the external storage memories are resources used from a standard PC station. the video coder is implemented in software by a Graphics Processing Unit (GPU). The L1_CMU is a task running on the CPU which controls the configuration of all testbed resources (DSP, PFGA). The DSP works as the master of the (DSP/FPGA) subpart of the platform. The position of master allows the DSP to manage the overall configuration of the functions that run on the SW/HW resources. The FPGA partial reconfigurability is, of course, a mandatory feature to allow reconfiguration of a single component. The L3_CMUs responsible for the configuration of the co-accelerators are implemented as a task into the Micro-blaze soft processor. It allows to perform fine grain reconfiguration of the FPGA without involving any external resource. The hardware and software designs of the processing functions are stored in the external storage memory of the platform where the configuration management can reach them.

6. Application and results

We have designed and implemented a real-time multi-standard application composed of an Active Appearance Models (AAM) schema Cootes et al. (2009) and a digital communication layer (802.11g, UMTS or GSM). The coding application feeds a video coded bit-stream in the transmitter whereas the associated video decoder is connected to the receiver (802.11g, UMTS or GSM). In this section, we give some results of our development approach with the Sundance platform.

6.1 The overall scenario

The platform illustrates the adaptation of the radio link according to the compression of the source in a video-telephony context. A person switches-on his terminal in order to perform an audio-video conversation with another person. At the beginning of the communication,

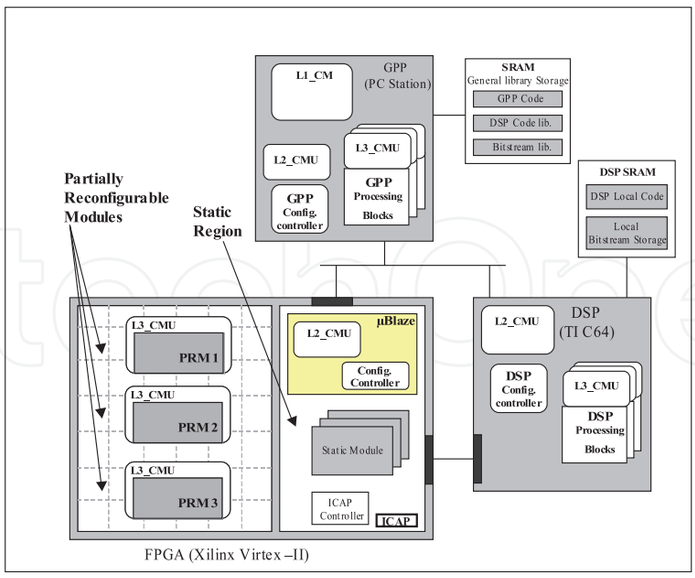


Fig. 8. example of the hierarchical configuration management mapping

the face of the speaker and the background of the image are transmitted using a traditional compression mode. This requires a relatively high data-rate over the time, a model of the person's face is generated at the transmitter's side, and sent to the receiver. Once this model is understood by the receiver, the transmitted parameters of the face's model (orientation, opening of the mouth, of the eyes, direction of the glance) are enough to reproduce the face behavior at the receiver. This permits to save the data amount required to transmit the face of the speaker, by reducing very significantly the data to be transmitted through the air. The data rate variations by step as well as the dynamic reconfigurations of the radio link are illustrated in figure 9.

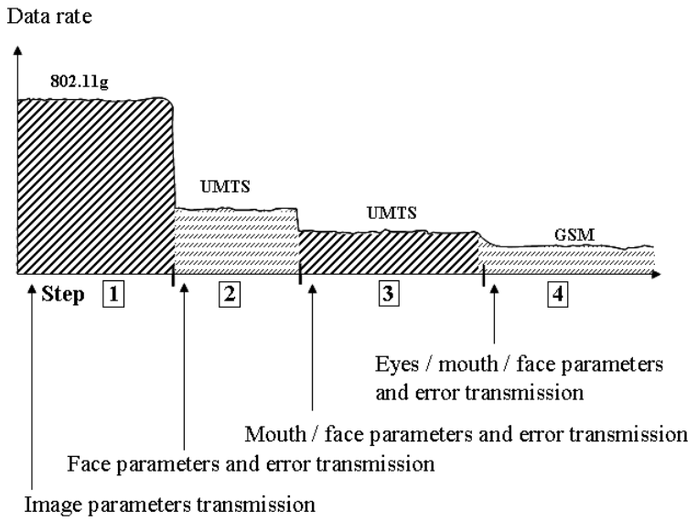


Fig. 9. Standard adaptation as function of the video compression

At the start, the person switches-on his terminal and starts a video-conference service. Video coder starts learning the face model, as well as models for eyes and mouth reconstruction. Then the radio link goes through the following steps.

Step 1: The image is transmitted using a traditional compression mode. The terminal learns the 3D model of the person face and performs a 802.11g modulation with standard error coding.

Step 2: The face model is learned: only high level parameters of the face are transmitted (location, size, orientation) so the receiver can reconstruct the 3D model of the face with its texture on the already sent background. In order to improve the reconstruction at the receiver, errors between the model and the real image are also transmitted by the means of an UMTS modulation with standard error coding.

Step 3: The mouth variations are modeled: The mouth characteristics, as well as high level parameters of the face model are transmitted through UMTS with a very robust error coding on the data for the mouth model.

Step 4: In this last step all face features' models were already learn only high level parameters of all three face, mouth, and eyes models are transmitted, as well as the errors with respect to the real image to help the reconstruction process. GSM modulation with standard error coding can then be used.

The last step is the longer period of the video-call, which permits to reduce very efficiently the global mean throughput necessary for the communication. This justifies the efforts accepted at the beginning of the call in terms of adaptation complexity. Changing from one data rate to another is possible, while permanently reconfiguring the air link characteristics to up a significant degree.

6.2 Experimental results

The matching step of SynDEx consists in performing mapping and scheduling of the algorithm's operations and data transfers onto the architecture processing components and communication media. It is carried out by a heuristic which takes into account durations of computations and inter-component communications to optimize the global application latency.

- Algorithm graph: Application algorithm is represented by a data-flow graph (DFG) to exhibit the potential parallelism between operations. The algorithm model is a direct data dependence graph. An operation is executed as soon as its inputs are available, and this DFG is infinitely repeated. SynDEx includes a hierarchical algorithm representation, conditional statements and iterations of algorithm parts. The application can be described in a hierarchical way by the algorithm graph. The lowest hierarchical level is always composed of indivisible operations. Operations are composed of several input and output ports. Special inputs are used to create conditional statements. Hence an alternative sub-graph is selected for execution according to the conditional entry value. Data dependencies between operations are represented by valued arcs. Each input and output port has to be defined with its length and data type. These lengths are used to express either the total required data amount needed by the operation before starting its computation or the total amount of data generated by the operation on each output port.
- Architecture graph: The architecture is also modeled by a graph, which is a directed graph where the vertices are computation operators (e.g processors, DSP, FPGA) or media (e.g

SHB, SDB, PCI, Ethernet) and the edges are connections between them. So the architecture structure exhibits the actual parallelism between operators. Computation vertices have no internal computation parallelism available. An example is shown in figure 10. In order to perform the graph matching process, computation vertices have to be characterized with algorithm operation execution times. Execution times are determined during the profiling process of the operation. The media are also characterized with the time needed to transmit a given data type.

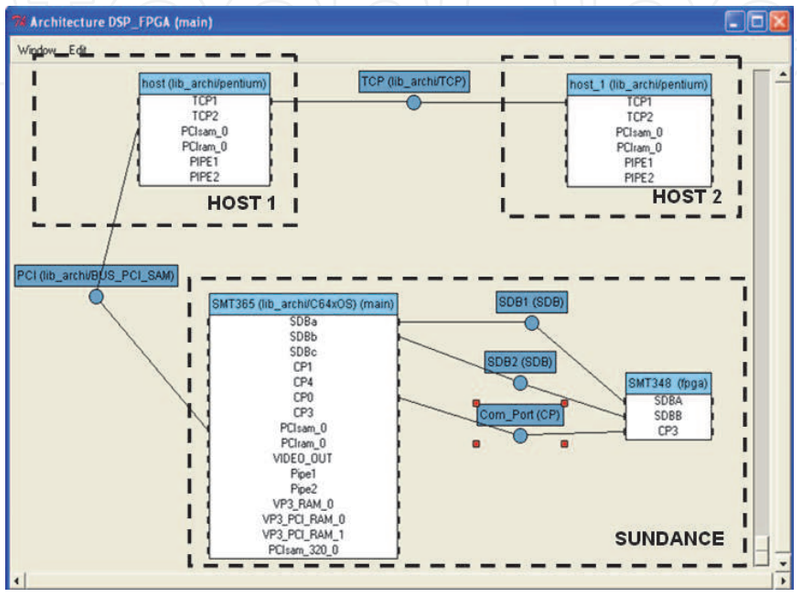


Fig. 10. Syndex model

The output files generated by SynDEx are exploited by our platform to manage correctly the hardware reconfiguration platform. These text files are managed by the L1_CMU of the transmitter (the platform in charge to send video stream) and by L1_CMU of the receiver to be standard compatible with the current or future transmission. In the next section , we present the hardware platform used and its specifications.
This hardware architecture is represented by an architecture graph under SynDEx and in the same manner a DFG is done for the application task (telecommunication chain to be used). Then, the heuristic of SynDEx realizes the adequation between the graph and the hardware and generates constraint files for the DSP and the FPGA that give information for the reconfiguration of the platform. Then the host of the platform sends a new source-code to the DSP and a new order of partial reconfiguration to the FPGA using the architecture described above.

7. Conclusion

In this chapter, we introduce a heterogeneous reconfigurable Sundance platform to support Cognitive Radio in the context of emergency networks. The heterogeneous reconfigurable architecture includes heterogeneous processing elements such as general purpose processors (GPU), DSPs and FPGAs. A key element in this heterogeneous reconfigurable architecture is the run-time partial reconfiguration of the hardware part, which can achieve the

reconfigurability in combination with the energy efficiency. A design methodology is needed to map applications onto a heterogeneous platform which has two new features: transaction level modeling of applications and run-time spatial mapping. In the future, we aim to validate the HDCRAM approach, test our PAPR and spectrum sensing algorithms in the case of MIMO system, and sets of algorithms for cognitive radio. The ultimate goal is to build a heterogeneous reconfigurable radio platform to demonstrate the cognitive radio functionalities.

8. References

J. Mitola, *"Cognitive Radio: An Integrated Agent Architecture for Software Defined Radio,"* Ph.D. dissertation, Royal Inst. of Tech., Sweden, May 2000.

J. Mitola, G. Maguire, *"Cognitive radio: making software radios more personal, Personal Communications,"* IEEE Wireless Communications, Vol. 6, No. 4. (1999), pp. 13-18

A. Nafkha, R. Seguier, J. Palicot, C. Moy, J.P. Delahaye, "A Reconfigurable BaseBand Transmitter for Adaptive Image Coding", *IST Mobile and Wireless Communications Summit*, 1-5 July 2007,

G. Chen, D. Kotz, "*A Survey of Context-Aware Mobile Computing Research*", Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, Nov. 2000

E. Aarts, H. Harwig, and M. Schuurmans, "*Ambient Intelligence: The Invisible Future*", J. Denning, ed., McGraw Hill, New York, 2001.

X. Qiu, J. Chuang, "Link adaptation in wireless data networks for throughput maximization under retransmissions", in Proceedings of *International Conference on Communications (ICC)*, Vancouver, 1999.

W. Fan, M. Krunz, C. Shuguang; "Price-Based Spectrum Management in Cognitive Radio Networks", *IEEE Journal of Selected Topics in Signal Processing*, vol.2, 2008.

M. Ghozzi, M. Dohler, F. Marx, J. Palicot, "Cognitive radio: methods for the detection of free bands, Towards reconfigurable and cognitive communications", Comptes rendus Physique, Paris, vol. 7, no7, 2006.

F. Jondral, "*Software Defined Radio Enabling Technologies*" edited by W.Tuttlebee, Wiley, 2002.

L.Alaus, J. Palicot, C. Roland, Y. Louet, D.Noguet, "Promising technique of parametrization for reconfigurable radio, the Common Operators Technique : fundamentals and examples", *Signal Processing For Software Defined Radio Handsets*, Springer, 2008.

A-R. Rhiemeier, "Benefits and Limits of Parameterized Channel Coding for Software Radio", 2nd Karlsruhe Workshop on Software Radios, Germany, 2002.

J. Palicot, C. Roland, "FFT: a Basic Function for a Reconfigurable Receiver", *International Conference on Telecommunications*, Papeete, Tahiti, 2003.

V. Rodriguez, C. Moy, J. Palicot, "Install or invoke?: The optimal trade-off between performance and cost in the design of multi-standard reconfigurable radios," Wiley Inter-science, *Wireless Communications and Mobile Computing Journal*, Volume 7 Issue 9, Pages 1143 - 1156, 2007.

J.P. Delahaye, J. Palicot, P. Leray, "A Hierarchical Modeling Approach in Software Defined Radio System Design", IEEE *Workshop on Signal Processing Systems*, Athens (Greece), Nov. 2005.

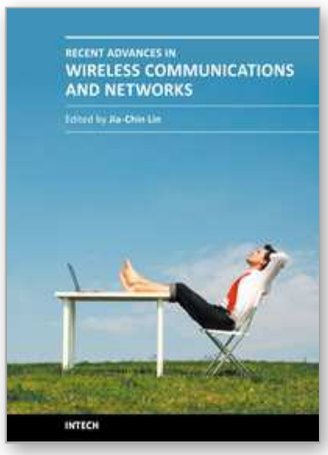
B. Blodget and S. McMillan and P. Lysaght, "A lightweight approach for embedded reconfiguration of FPGAs," in *Design, Automation and Test in Europe Conference and Exhibition*, 2003.

T. Grandpierre, C. Lavarenne, and Y. Sorel, "Optimized Rapid Prototyping for Real-Time Embedded Heterogeneous Multiprocessors",in CODES, Rome, Italy, May 1999.

J.P. Delahaye, C. Moy, P. Leray, J. Palicot, "Managing Dynamic Partial Reconfiguration on Heterogeneous SDR Platforms", Sdr Forum, November, Los Angeles, USA, 2005.

L. GODARD, C. MOY, J. PALICOT, ”An Executable Meta-Model of a Hierarchical and Distributed Architecture Management for the Design of Cognitive Radio Equipments,” Annals of Telecommunications, Special issue on Cognitive Radio, Volume 64, Numbers 7-8, 2009.

T.F.Cootes, G.J.Edwards and C.J.Taylor, ”Active Appearence Models”, Europeen Conference on Computer Vision,1998.



Recent Advances in Wireless Communications and Networks
Edited by Prof. Jia-Chin Lin

ISBN 978-953-307-274-6
Hard cover, 454 pages
Publisher InTech
Published online 23, August, 2011
Published in print edition August, 2011

This book focuses on the current hottest issues from the lowest layers to the upper layers of wireless communication networks and provides â€œreal-timeâ€ research progress on these issues. The authors have made every effort to systematically organize the information on these topics to make it easily accessible to readers of any level. This book also maintains the balance between current research results and their theoretical support. In this book, a variety of novel techniques in wireless communications and networks are investigated. The authors attempt to present these topics in detail. Insightful and reader-friendly descriptions are presented to nourish readers of any level, from practicing and knowledgeable communication engineers to beginning or professional researchers. All interested readers can easily find noteworthy materials in much greater detail than in previous publications and in the references cited in these chapters.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen