Low-power system-on-chip architecture for wireless LANs

L. Bisdounis, C. Dre, S. Blionas, D. Metafas, A. Tatsaki, F. Ieromnimon, E. Macii, P. Rouzet, R. Zafalon and L. Benini

Abstract: The authors present the architecture of a low-power system-on-chip (SoC) that implements baseband processing as well as the medium access control and data link control functionalities of a 5 GHz wireless system. The design is based on the HIPERLAN/2 wireless LAN standard, but it also covers critical processing requirements of the IEEE 802.11a standard. The options, constraints and motivations for the taken design decisions are presented, and the followed design steps, starting from the system specifications up to the architecture definition and the system implementation, are explained. The system's functionality covers both mobile terminal and access point devices. A critical design task in such systems is the assignment of the target system's tasks on the different types of processing elements available. Processor cores, dedicated hardware as well as memory elements and advanced bus architectures are used in order to achieve the target implementation. The architecture is targeted for a low-power SoC platform, due to the fact that power consumption is a critical parameter in electronic portable system design where excess power dissipation can lead to expensive and less reliable systems. A system prototype has been developed on a FPGA-based platform (including microprocessor modules). This FPGA-based prototype is currently being migrated to a SoC, which requires that the treatment of important issues such as clock handling, synthesis, testability and debugging is addressed.

1 Introduction

A significant increase in market growth rates and a rapidly evolving technology for the wireless office and home networking have created a significant opportunity for chip design houses and manufacturers to develop new products [1]. For many years, the use of wireless LANs has been limited to a very few specialised vertical applications. However, since the introduction of standards [2] such as the ETSI BRAN HIPERLAN/2 (high performance radio local area network type 2) [3, 4] and the IEEE 802.11a [5] for wireless LANs, the market is moving in a new direction, gaining even greater momentum. The number of chips sold worldwide in 2002 in order to satisfy this demand exceeded 14 million units, an almost 75% increase from the previous year [1]. By 2006 it has been estimated that over 60 million wireless LAN semiconductor components will be shipped [1]. Although, both aforementioned standards operate in the 5 GHz band and are based on orthogonal frequency division multiplexing (OFDM) technology, their system characteristics differ significantly, especially in synchronisation and upper protocol layers' processing [6].

Wireless communication systems require the optimisation of different factors including real-time performance, area, power, flexibility and time-to-market [7]. In order to optimise the combination of the above factors, instruction-set processors, custom hardware blocks as well as low-power memory and bus interface synthesis and mapping techniques are applied, offering a good balance between flexibility and implementation efficiency [8, 9]. The evolving scenario has serious consequences for any system-on-chip (SoC) development to be used in wireless systems. The protocol processor (that is able to run both the HIPERLAN/2 and IEEE 802.11a medium access control/data link control (MAC/DLC) processes) is included in the proposed SoC in contrast with previous 5 GHz WLAN ASIC implementations [10, 11] in which an external protocol processor must be used. Note, that this external protocol processor cannot be the host processor (e.g. in a notebook or a base station), but is an additional dedicated processor able to run such applications. The lower MAC and DLC processing requirements of both WLAN standards are such that a dedicated processor in close collaboration with the baseband modem hardware (full access, on-chip buses for high speed) is needed for the real-time operation of the system. Additional advantages of the proposed architecture over the previous ones include: (i) the adopted low-power design methodology; (ii) the flexibility created by using an embedded CPU core for controlling the baseband modem and implementing the lower MAC processing of the

IEE Proceedings online no. 20030978

doi: 10.1049/ip-cdt:20030978

Paper first received 12th June and in revised form 3rd September 2003

L. Bisdounis, C. Dre, S. Blionas, D. Metafas, A. Tatsaki and F. Ieromnimon are with INTRACOM S.A., Technical Division, 19.5 Km Markopoulo Ave., P.O. Box 68, GR-19002 Peania, Athens, Greece

E. Macii is with the Dipartimento di Automatica e Informatica, Politecnico di Torino, Corso Duca degli Abruzzi 24, I-10129 Torino, Italy

P. Rouzet is with STMicroelectronics, Advanced System Technology, Broadband Wireless LAN Group, Chemin du Champ des Filles 39, CH-1228 Plan-les-Quates-Geneve, Switzerland

R. Zafalon is with STMicroelectronics, Advanced System Technology, Low-Power System Design Group, Via C. Olivetti 2, I-20041 Agrate Brianza, Milano, Italy

L. Benini is with the Dipartimento di Elettronica Informatica e Sistemistica, Universitá di Bologna, Viale Risorgimento 2, I-40136 Bologna, Italy

HIPERLAN/2 standard; and (iii) a custom processor (MAC hardware accelerator) for implementing the lower MAC processing of the IEEE 802.11a standard. In [10], the baseband processing of both 5 GHz WLAN standards is implemented by using a combination of a custom digital signal processor (DSP) and a custom streaming co-processor, while in [11] ASIC implementations of the OFDM baseband modem are obtained with some performance-improving add-ons. A major advantage of the proposed design is that a power optimisation methodology is applied at all levels of the system starting from the embedded software and the hardware-software mapping and ending with the memory and bus interface synthesis. This compares favourably with the existing solutions that only use traditional techniques such as clock-gating implementation in order to reduce the consumed power.

To define the architecture and verify the functionality of the dual-mode HIPERLAN/2 – IEEE 802.11a SoC, the following steps were followed:

- Definition of the requirements and parameters for a wireless system that hosts the developed SoC.
- Development of the system's high-level model.
- Application of a power-conscious hardware-software mapping procedure that also takes into account the results of the above steps.
- Architecture exploration and architectural template definition.
- Architecture refinement that incorporates the results of the low-power methodological tasks.
- Final architecture definition.
- FPGA-based prototyping.

2 The requirements and parameters of the wireless system

In order to define the SoC architecture, it is very important to start with the system parameters and requirements definition that will host the final chip. The developed SoC will be suitable for use in a 5 GHz WLAN system that includes an access point and mobile terminals. In a 5 GHz WLAN system, the access point (AP) and mobile terminal (MT) exchange Ethernet frames through RF connection [3]. Figure 1 illustrates the design of an AP – MT system.

As previously mentioned, the developed SoC hosts the modem functionality of the HIPERLAN/2 and IEEE 802.11a standards. Orthogonal frequency division multiplexing (OFDM) [12] has been selected as the modulation scheme for HIPERLAN/2 [4] and IEEE 802.11a [5] due to its low-cost implementation for frequency selective channels. The sampling frequency is chosen to be equal to 20 MHz. The obtained subcarrier spacing is 0.3125 MHz. In order to facilitate implementation of filters and to achieve sufficient adjacent channel suppression, 52 subcarriers are used per channel. 48 subcarriers carry actual data and 4 subcarriers are pilots which facilitate phase tracking for coherent demodulation. The duration of the cyclic prefix is equal to 800 ns, which is sufficient to enable good performance on channels with a root-mean-square (rms) delay spread of up to 250 ns (at least). A list of the timing OFDM parameters [4, 5] is given in Table 1.

To correct for subcarriers in deep fades, channel estimation and forward-error correction across the subcarriers are used with variable coding rates, giving coded data rates from 6 to 54 Mbps. A key feature of the physical layer is to provide several physical layer modes with different coding rates and modulation schemes, which are selected by link adaptation. BPSK, QPSK and 16-QAM are the supported subcarrier modulation schemes. Furthermore, 64-QAM can be used in an optional mode. Forward error control is performed by a convolutional code of rate 1/2 and constraint length seven. It appends six tail bits at the end of the PDU (protocol data unit) train in order to set the convolutional encoder at a zero state. In the case of the HIPERLAN/2 standard, the first puncturing unit is defined by a pattern that specifies which bits should be punctured (deleted), and it is applied to the first 156 bits of the PDU train. A second puncturing unit follows to provide code rates of 1/2, 3/4 and 9/16 and it is performed equally to all PDU train types. The modes should be chosen such that the

Table 1: Timing OFDM parameters

Parameter	Value
Sampling rate	$f_S = 20\,\text{MHz}$
Useful symbol part duration (IFFT/FFT period)	$T_U = 64 \times T = 3.2\,\mu s$
Cyclic prefix (CP) duration	$T_{CP} = 16 \times T = 0.8\,\mu s$
Symbol interval	$T_S = 80 \times T = 4\,\mu s$
Subcarrier spacing	$\Delta f = 1/T_U = 0.3125\,\text{MHz}$
Spacing between the two outmost subcarriers	$\Delta f \times$ number of Subcarriers $= 16.25\,\text{MHz}$
Broadcast burst preamble duration	$16\,\mu s$
Downlink burst preamble duration	$8\,\mu s$
Uplink burst short preamble duration	$12\,\mu s$
Uplink burst long preamble duration	$16\,\mu s$

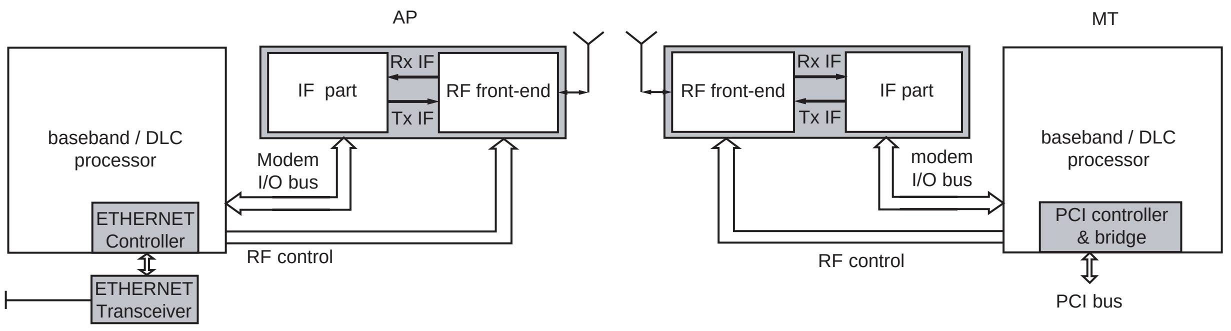

Fig. 1 *AP – MT system design*

number of encoder output bits of a PDU fits to an integer number of OFDM symbols. IEEE 802.11a includes only one (rate-dependent) puncturing stage. The remaining differences [6] between the physical layers of HIPERLAN/2 and IEEE 802.11a concern the support of different coding rates (see Table 2), the different contents of preamble sequences, and the different scrambling initialisation sequences. The mode-dependent parameters for both standards [4, 5] are listed in Table 2.

Apart from the physical layers, the developed SoC should be capable of supporting the DLC layer of the HIPERLAN/2 standard [13, 14] as well as the lower MAC layer's critical functionality of the IEEE 802.11a standard [15]. The HIPERLAN/2 basic protocol stack and its functions are shown in Fig. 2 [13]. The convergence layer (CL) offers services to the higher protocol layers. These layers are beyond the scope of this discussion. The DLC layer consists of the error control function (EC), the medium access control function (MAC) and the radio link control function (RLC). It is divided into the data transport functions, located mainly on the right-hand side of Fig. 2 (user plane), and the control functions located on the left-hand side (control plane). The user data transport function on the right-hand side is fed with user data packets from the higher layers via the user service access point (U-SAP). This part contains the EC, which performs an ARQ (automatic repeat request) protocol. The DLC protocol operates connection-oriented, which is shown by multiple connection end points in the U-SAP. One EC instance is created for each DLC connection. In the case where the higher layer is connection-oriented, DLC connections can be created and released dynamically. In the case where the higher layer is connectionless, at least one DLC connection must be set up which handles all user data, since HIPERLAN/2 is purely connection-oriented. The left part of the protocol stack contains the RLC sublayer, which delivers a transport service to the DLC connection control (DCC), the radio resource control (RRC) and the association control function (ACF). Only the RLC is standardised which defines implicitly the behaviour of the DCC, ACF and RRC. One RLC instance needs to be created per MT. The CL on top is also separated into data transport and control parts. The data transport part provides the adaptation of the user data format to the message format of the DLC layer (DLC SDU). In the case of higher layer networks other than ATM, it contains a segmentation and reassembly function. The control part can

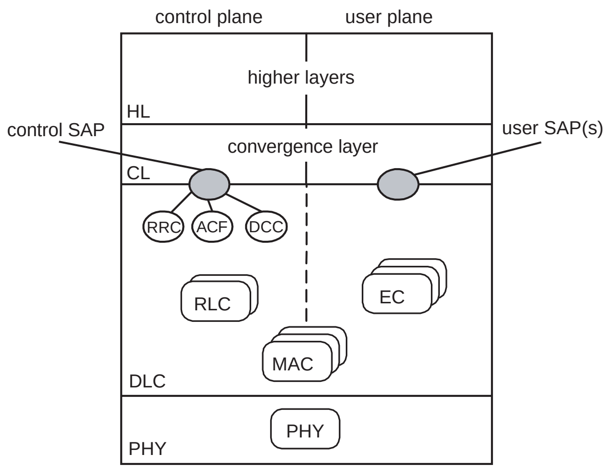

Fig. 2 *HIPERLAN/2 protocol stack and functions*

make use of the control functions in the DLC, e.g. when negotiating CL parameters at association time.

The HIPERLAN/2 DLC functions are divided in data transport [13] and DLC control functions [14]. The DLC functions include the following main operations:

- (Des)Association.
- DLC user (de)connection.
- Encryption, decryption (56 bit key DES, TripleDES optional).
- (De)Framing.
- Contention management mechanism.
- BCCH (broadcast control channel) and FCCH (frame control channel) analysis and synthesis.
- DLC-CL buffering.
- ARQ (automatic repeat request) mechanism for asynchronous transactions.
- Power saving.
- Dynamic frequency selection (DFS).
- Transmission power control (TPC).

The architecture of the IEEE 802.11 MAC layer is given in Fig. 3. The basic characteristics of the IEEE 802.11 MAC layer are given below [15]:

- Support for both *ad-hoc* and infrastructure wireless LAN.
- On the fly encryption/decryption WEP (wired equivalent privacy) this is a (48 bit RCA PRNG algorithm).

Table 2: Mode-dependent parameters for both standards

Modulation scheme	Coding rate	Data rate, Mbps	Coded bits per subcarrier	Coded bits per OFDM symbol	Data bits per OFDM symbol
Both Standards					
BPSK	1/2	6	1	48	24
BPSK	3/4	9	1	48	36
QPSK	1/2	12	2	96	48
QPSK	3/4	18	2	96	72
16-QAM	3/4	36	4	192	144
HIPERLAN/2 standard only					
16-QAM	9/16	27	4	192	108
IEEE 802.11a standard only					
16-QAM	1/2	24	4	192	96
64-QAM	2/3	48	6	288	192
Both standards					
64-QAM	3/4	54	6	288	216

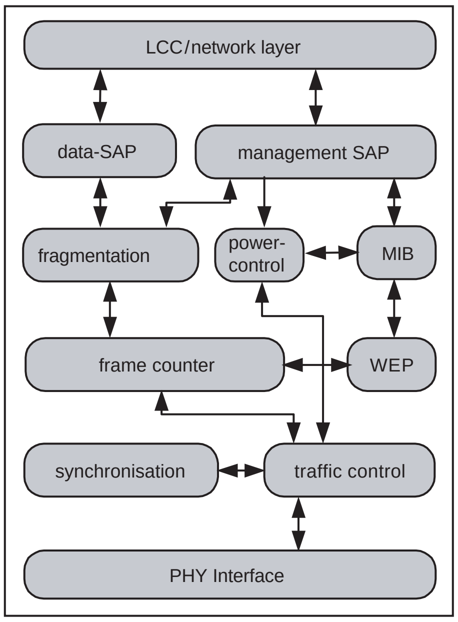

Fig. 3 *IEEE 802.11a MAC architecture*

- MAC-level fragmentation and de-fragmentation.
- Allows support of QoS (quality of service) (802.11e), Security (802.11i) and DFS/TPC standard extensions (802.11 h)
- DCF (Distributed coordination function), EDCF (enhanced DCF), PCF (point coordination function) and HCF (hybrid coordination function) support: bandwidth reservation, contention free medium access, traffic category management (priority and queue based).
- Support of advanced QoS-oriented schemes such as burst acknowledge mode.

3 High-level model

A high-level model of a HIPERLAN/2-based 5 GHz WLAN system was developed and tested. The developed model fulfils two major targets: (i) it assists in the software development and the testing of the 5 GHz system; and (ii) it is used as input for the low-power methodology tasks that run in parallel with the SoC implementation. The high-level model is used as the specification input for the profiling within the power-conscious hardware-software mapping and memory/bus interface synthesis procedures. Also, the high-level model is used as the specification input for the data transfer and storage optimisations that are achieved through preprocessing/pruning and code transformations. The model was developed using a UML-based software development process inside the rational rose real-time framework [16]. The UML model without any user intervention automatically generates the executable software. The main advantage of this UML-based process is that the same model can be tested on the development environment as executed on the actual hardware. The UML-based tool can produce executable C++ code that can be ported to different combinations of RTOS (real-time operating system) and processor. The developed model consists of three major building blocks, (i) the AP model; (ii) the MT model; and (iii) the testbench core module, the tester. The AP model includes all the required functional blocks of an AP, and the required models that support its operation. The MT model is the proportional model for the MT (Fig. 4). Finally, the testbench module contains all the

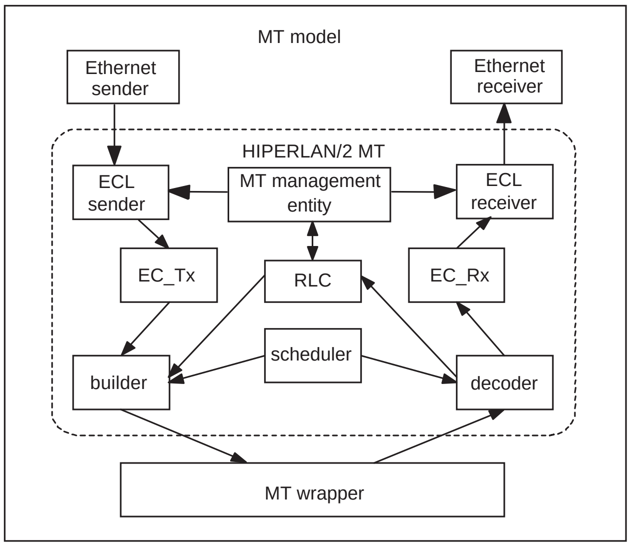

Fig. 4 *MT model structure*

testing software for proving the correctness of the AP and MT's implementation.

The described high-level model has been derived after an exhaustive analysis of the process view of the HIPERLAN/2 standard. An analysis of the processes that are performed within the HIPERLAN/2 devices and that must be supported by the developed SoC was performed. A schematic representation of a simplified process view for a HIPERLAN/2 MT is shown in Fig. 5. In this representation the interactions between the processes are also shown. A process view for the AP is implemented in a similar way. An IEEE 802.11 high-level model can be derived similarly, by analysing the process view of the IEEE 802.11a standard [15].

The baseband modem's process is a heavy DSP data path, thus for the modelling of this process a Matlab representation was selected. This Matlab modem is valuable not only for the algorithmic verification of the modem, but also for the subsequent design phase as it can provide checkpoints for the intermediate processing stages of its hardware implementation.

4 Architecture exploration and definition

The target SoC consists of instruction-set processor cores, the modem datapath, internal memory modules and DMA engine for fast data transfers plus a number of peripheral components for I/O and auxiliary tasks. All these components are organised as master and slave peripherals of a central advanced microcontroller bus architecture (AMBA) AHB bus [17]. Two alternative architectures have been studied for the target SoC. Both architectures have an ARM7 TDMI RISC core [18]. The two alternatives concern the way in which the lower MAC controller as well as the baseband modem's controller will be implemented. The first choice was to realise it in software running on an instruction-set processor (e.g. ARM) and the second was to design a dedicated hardware unit. The first solution has been selected mainly due to the fact that it is easier to modify. The second alternative will also be investigated for future use. The decision for a dual-processor architecture was based on technical criteria related to specific system requirements and parameters as well as on criteria related to the reusability and flexibility of the architecture in order to not only cover both existing standards, but also upcoming ones.

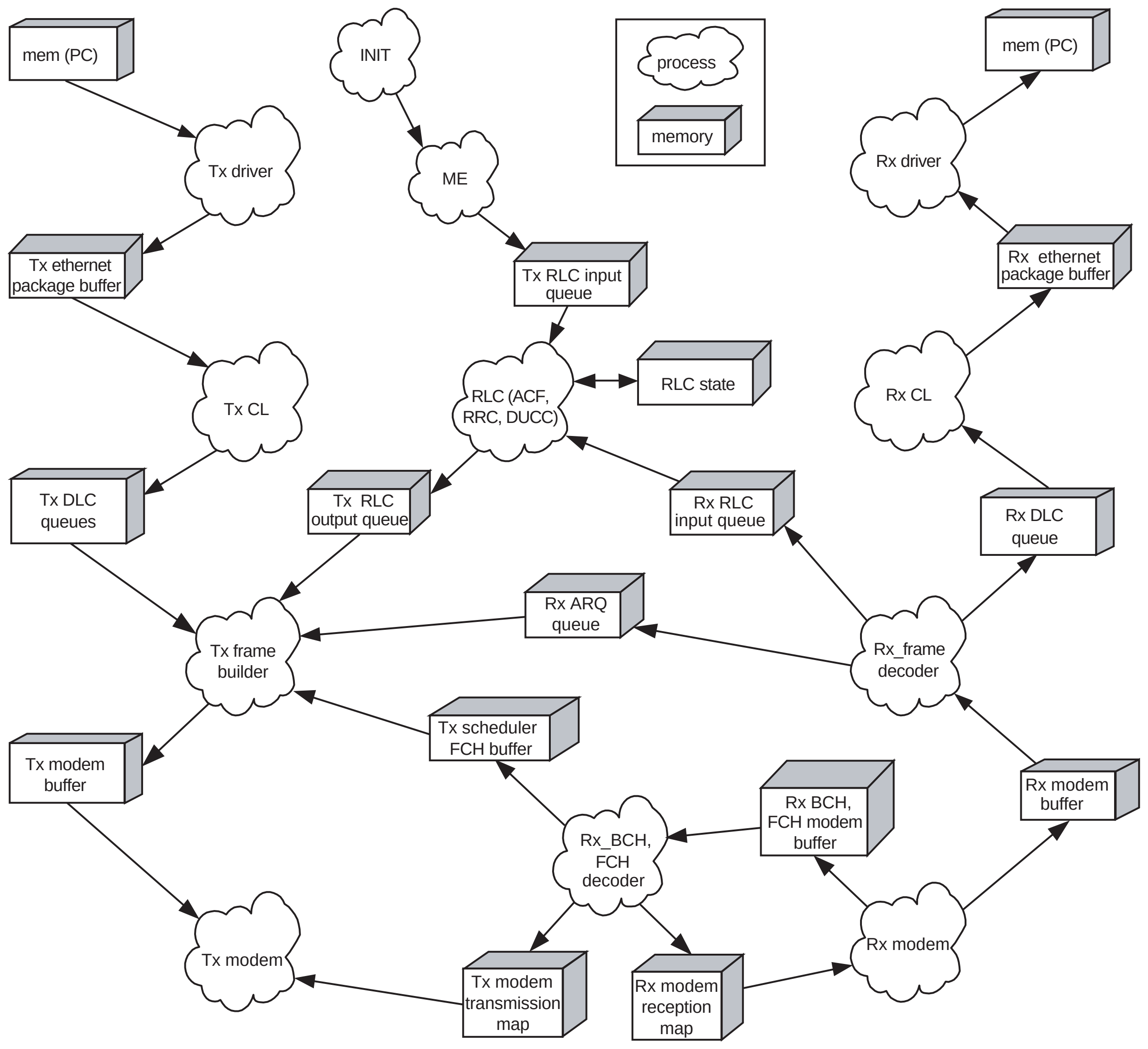

Fig. 5 *MT process view*

The most common alternative solutions for choosing an embedded microprocessor are ARM, ARC, IBM PowerPC, MIPS, 8051, 6502. For the developed design ARM, ARC and MIPS cores were investigated. Choosing the right core is a tricky procedure that should take into account target performance, availability to the system designer, cost (NRE, chip production). The final choice was the ARM7 TDMI core, since it meets our requirements in a multi-processor architecture and with its small size (smaller than 0.6 mm^2 on 0.18 μm process) and 3600 MIPS/W power efficiency, offers the smallest footprint and longest battery life. The ARM7 family's instruction set reduces system memory requirements and costs to a minimum, and provides 32-bit performance for a 16-bit budget. Also, it benefits from a wide range of application software, operating systems and development tools support.

The support of the AMBA [17] by the ARM cores, as well as its technical characteristics were the reasons for its adoption to the developed SoC architecture. AMBA specification defines an on-chip communications standard for designing high-performance embedded microcontrollers. A test methodology is included with the AMBA specification, which provides an infrastructure for modular macrocell test and diagnostic access. The AMBA AHB which is used for high-performance, high clock frequency system modules acts as the high-performance system backbone bus and supports the efficient connection of processors, on-chip memories and off-chip external memory interfaces with low-power peripheral macrocell functions. AHB is also specified to ensure ease of use in an efficient design flow using synthesis and automated test techniques.

After the processing and balancing of the pros and cons of the architecture alternatives derived in terms of technical and marketing issues, the final architectural template shown in Fig. 6 was defined. The presented template is based on a memory-mapped architecture, based on the AMBA AHB bus. A typical AMBA AHB system design contains the following components [17]:

- AHB master: A bus master is able to initiate read and write operations by providing an address and control information.
- AHB slave: A bus slave responds to a read or write operation within a given address-space range. The bus slave signals back to the active master the success, failure or waiting of the data transfer.
- AHB arbiter: The bus arbiter ensures that only one bus master at a time is allowed to initiate data transfers. Even though the arbitration protocol is fixed, any arbitration algorithm, such as highest priority or fair access can be implemented depending on the application requirements.

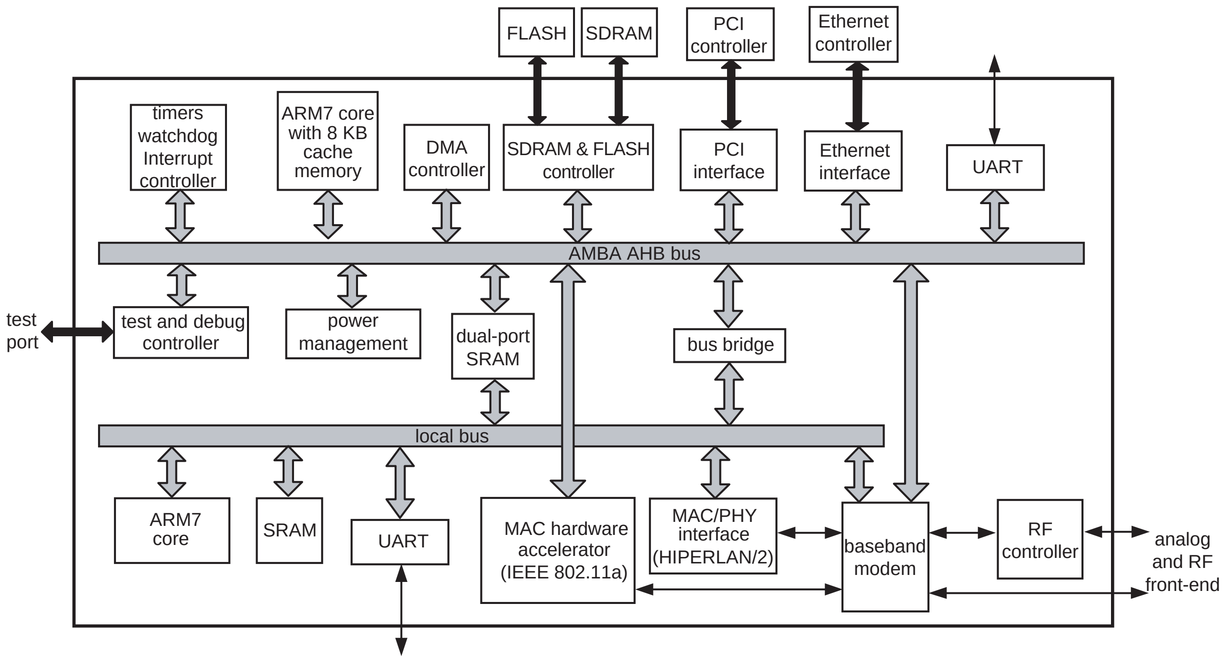

Fig. 6 *Architectural template of the SoC*

• AHB decoder: The AHB decoder is used to decode the address of each transfer and provide a select signal for the slave that is involved in the transfer.

There are two ARM7 processors, one serves as the protocol processor, while the other serves as the lower MAC and baseband modem's controller supporting these modules for timing critical tasks. The protocol processor includes a cache memory (8 KB) where the protocol stack program and the received control messages are stored (DLC HIPERLAN/2 or IEEE 802.11a). The timers, the interrupt controller and the watchdog circuit are used for the determination of the timing resolution and the handling of the communication between the protocol processor and the internal/external memories. The PCI controller is used for the communication of the HIPERLAN/2 device with the host processor in the case of MT operation, while the Ethernet controller is used in the case of AP operation. Although, the integration of the PCI and Ethernet controllers inside the chip is feasible and would further reduce the power consumption, it was decided mainly for time-to-market and cost purposes to have these components outside the chip. For quick block-memory transfers, a DMA controller is attached to the AHB, configured as a third bus-master of the highest priority. The DMA controller must be able to assign priorities, perform static address translation in order to ease system partitioning and also dynamically assign bus bandwidth according to current requests being serviced.

Having the local bus showed in Fig. 6, eases chip resource requirements (which would be large for the case of two complete AHB systems, one per processor). It is used for the interconnection of the second ARM core with the baseband/lower MAC processing modules, allowing enough bandwidth and latency for the application. The dual-port SRAM as well as the SRAM at the local bus will be used not only for program but also local data processing. The SDRAM and Flash controller will ensure the communication between the protocol processor and the external memory components. The Flash memory is used to avoid loss of the code during the power down of the system. At the beginning of the system's operation the protocol code is transferred from the Flash to the SDRAM memory through the DMA.

The power management unit will contain special circuitry in order to save power dissipated by the system. This special circuitry will be based on techniques such as: clock gating, supply shutdown, data encoding for low energy etc. The test controller will allow external access to the internal AHB bus. That is mainly for manufacturing testing of internal modules without using the functional interfaces. Optionally some interfaces to the internal bus for debugging purposes can be foreseen. Additional test functionality will be implemented to enable chip testing without expensive production testers. The test controller can be extended to support other test and debug functionalities like memory test and logic BIST.

Other basic macrocells included to the architectural template are:

• A MAC hardware accelerator (MHA) which supports the implementation of the IEEE 802.11a standard's lower MAC processing [15], and includes dedicated hardware modules for encryption/decryption, fragmentation, timing control (generation of ACK (acknowledgment), RTS (request to send) and CTS (clear to send) signals), protocol medium access (backoff process, real carrier sense using CCA (clear channel assessment) and virtual carrier sense using NAV (network allocation vector)), CRC (cyclic redundancy code) etc. This block also includes the appropriate MAC/PHY interface circuitry for the case of the IEEE 802.11a standard.
• A dedicated block in order to interface the MAC and PHY processing elements for the case of the HIPERLAN/2 standard.
• OFDM baseband modem supporting both standards.
• RF front-end controller.
• UART I/O serial interfaces.

The MHA is connected to the PHY layer with dedicated signals for control and a data bus for data transfer. The MHA, in transmit mode, receives the MSDU or packet to transfer, from the host to the wireless medium, through the AMBA bus. Before transfer, the time critical data is stored on an internal dual-port SRAM where it is eventually encrypted or fragmented on-the-fly by the MHA. The data transfer, from the host main memory to the local memory, does not use the local ARM processor resources, instead the upper-level DMA controller or directly the host CPU performs the task. The basic functionality of the MHA in transmit mode is to avoid collisions on medium access using real and virtual carrier sense, and optimise the data rate transfer using protocol features such as fragmentation, RTS/CTS, PHY layer transmit rate selection. The MHA, in receive mode, checks and transfers the MPDU or frames received from the medium to the local memory. The MHA is not mastering the data transfer to the host memory; this task is left to the upper-level DMA controller. The basic functionality of the MHA in receive mode is to validate incoming frames, select correctly addressed frames, decrypt and rebuild on entire MAC service data unit (MSDU) to be forwarded to the host. Figure 7 describes the internal structure of the MAC hardware accelerator. A summary of the functionality of the modules shown in Fig. 7 is given below (see also in [15]):

- The tx_llc module receives the frames and builds the header register.
- The tx_prepare module performs the fragmentation and encryption if required; the data are then forwarded to tx_data.
- The tx_ctrl implements DCF, RTS, Beacon generation, also supports the EDCF protocol and QoS functionalities such as HCF.
- The tx_data formats the frames (at the octet level) for the PHY layer: CRC calculation, timestamp insert, and sends the octets to the PHY layer.
- The chan_state generates slot time reference for the tx_bkoff module, detects busy channel from the CCA signal from the PHY layer, grants access to the PHY with correct timing such as DIFS (distributed interframe space), EIFS (extended interframe space) and random slot count.
- The tx_bkoff module counts a random number of slots, or defers the count when the medium is busy, and gives the transmit time slot to tx_ctrl when counts reaches zero.
- The rx_defrag rebuilds the MSDU from fragmented received frames and decrypts.
- The rx_filter decodes the incoming frame, checks the destination address and duplicated frames, sends ACK and CTS requests to rx_ctrl, maintains NAV.
- The rx_ctrl generates control frames (ACK and CTS) to be sent directly to tx_data.
- The rx_data validates the incoming frame: CRC and length check, forwards header frame to rx_filter and data to rx_defrag through the rx_fifo.
- The Bus interface, manages the protocol access to the system bus, and contains configuration and MIB (management information base) registers.
- The MIB and CSR (command status registers) contain the configuration and management information base registers.

The block diagrams of both the transmit and receive paths of the baseband modem [4, 5, 19] supporting both the HIPERLAN/2 and the IEEE 802.11a standards are illustrated in Fig. 8. In the transmit path, the binary input data are scrambled and then encoded by a standard rate 1/2 convolutional encoder. Puncturing the coded output bits may increase the rate. After interleaving, the binary values are modulated by using PSK or QAM. The input bits are divided into groups of 1, 2, 4 or 6 bits and converted into complex numbers representing BPSK, QPSK, 16QAM or 64QAM values [19]. To facilitate coherent reception, four pilot values are added to each 48 constellation points, so a total of 52 values is reached per OFDM symbol, which are modulated onto 52 subcarriers, and then by applying the IFFT the output is transformed to the time domain. To make the system robust to multipath propagation, a cyclic prefix is added, and the PHY burst formation is completed through the insertion of the proper preambles. The next step is the digital IF encoding. After that, the digital output signals can be converted to analog signals, which are then up-converted to the 5 GHz band, amplified and transmitted through an antenna.

In the receive path, the first step is the digital IF decoding, which is followed by the time, the frequency and the data domains of the receiver. Before the receiver can demodulate the subcarriers, it has to perform two synchronisation tasks: symbol synchronisation and frequency offset estimation and correction [20]. The time domain of the receive path is continued with the cyclic prefix extractor that performs the removal of the 16 samples of the cyclic prefix, in order that 64 of the total 80 samples of each symbol can be inserted into the FFT block. The transition of the received OFDM signal from the time domain to the frequency domain is achieved by the application of the 64-point FFT.

The frequency domain of the receive path contains the channel estimator that is used for the comparison of the received signal (channel estimation part of the frame) with

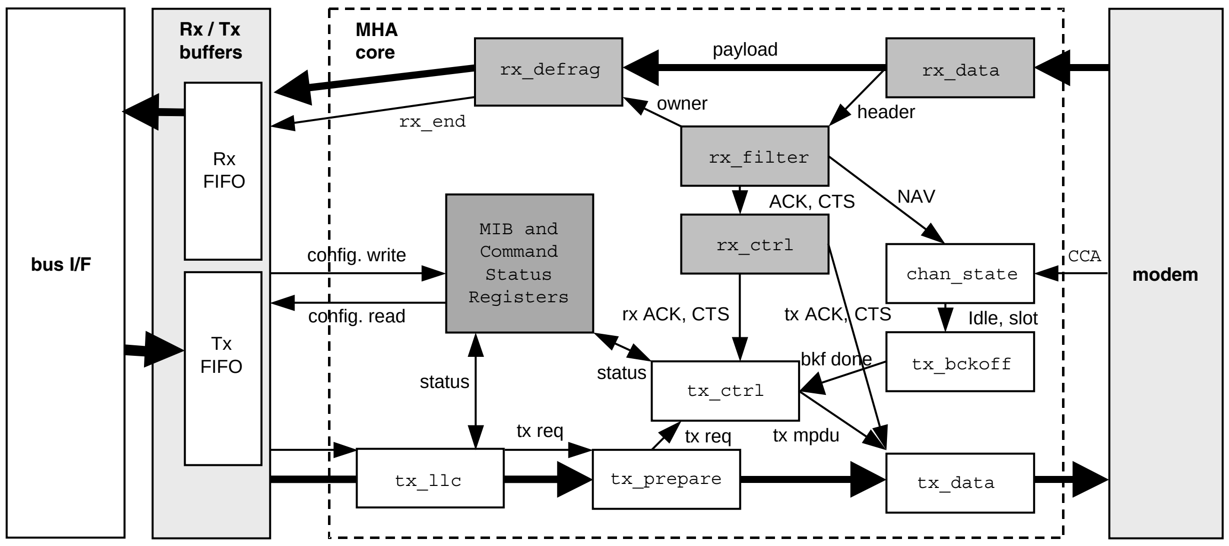

Fig. 7 *Architecture of the MAC hardware accelerator block*

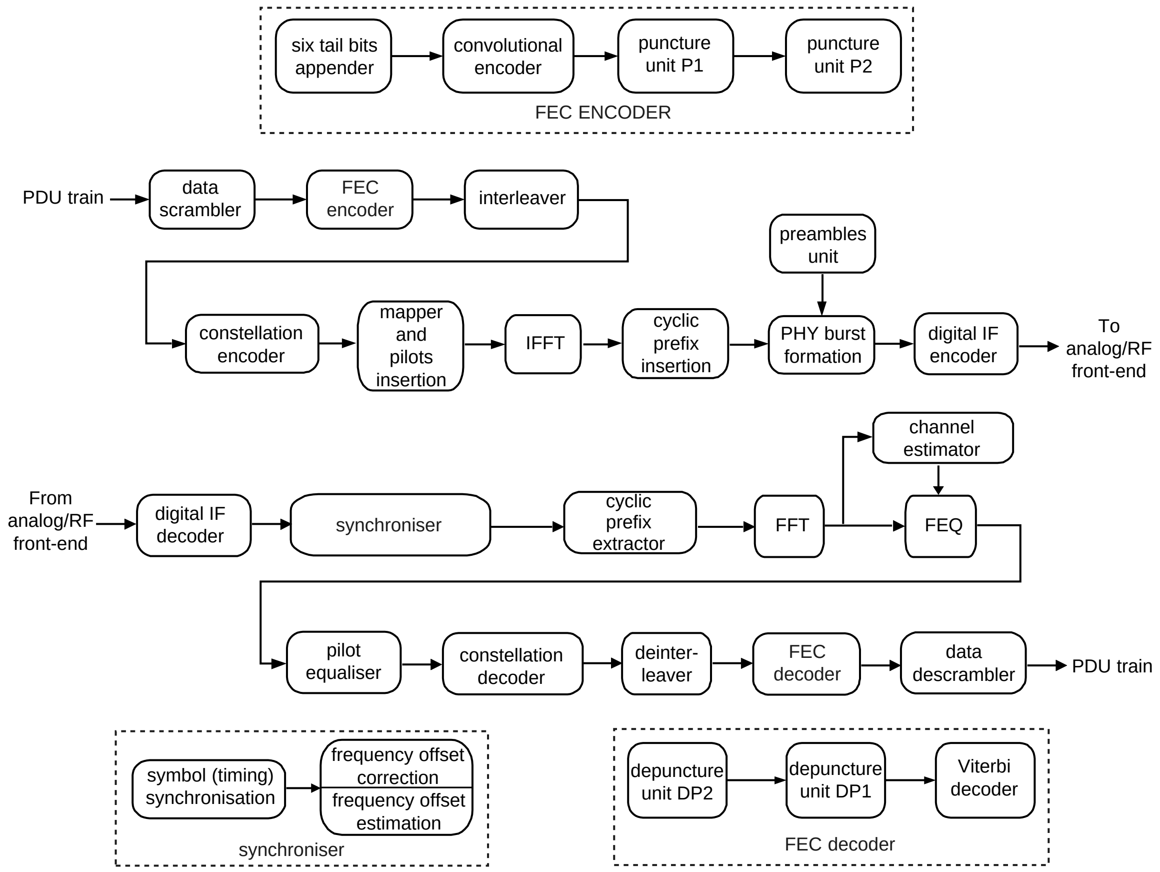

Fig. 8 *Baseband modem's block diagram (transmit and receive paths)*

the expected sequence in order to estimate the phase and amplitude attenuation produced by the channel. The FEQ (frequency equaliser) performs the equalisation of the received signal by using the channel coefficients (complex multiplication in the frequency domain), while the pilot equaliser corrects the phase of the received data carriers by using the pilot symbols as a reference. In the data domain, initially the decoding methods for the four modulation schemes (BPSK, QPSK, 16-QAM and 64-QAM) are applied by the constellation decoder. Then, the data are deinterleaved, and inserted to the FEC (forward error correction) decoder that consists of two depuncturing units followed by a convolutional decoder (Viterbi decoder). Finally, the descrambler performs the descrambling of the decoded data bits.

5 Refinement of the system architecture to reduce the power consumption

In order to provide competitive networking devices with advanced features, the implementation of the SoC is based on a design methodology, which contains energy optimisation techniques [9] that affect both the embedded software and the architecture of the system. The techniques that affect the embedded software concern data transfer and storage optimisations through preprocessing/pruning and code transformations [21], as well as embedded software exploration in terms of the estimation of the energy consumed by the software through instruction-level power models [22]. The techniques that lead to a refinement of the architecture for low-power consumption include power conscious hardware-software mapping of the system's functionality [23, 24], a low-power memory [25] and bus interface synthesis [26] (Fig. 9).

The developed optimisation techniques explicitly target applications from the specific application domain (wireless protocols). For this reason the special features of the applications in the target domain, mainly related to control flow, data organisation and type of processing, will be evaluated and taken into consideration during the development of the optimisation techniques. It is expected that these application specific optimisation techniques will significantly reduce the optimisation time while leading to good quality results as compared to general optimisation techniques.

The hardware-software mapping that takes as its input the high-level model of the system, which is in fact a combination of process views and timing diagrams that meet the specifications of both standards, is the main task that leads to the SoC architecture definition. Apart from that, a systematic approach was developed for the mapping of the optimised system specification to the available processors (instruction set and custom hardware) that is mainly driven by a power-oriented cost function that is also developed. In order to do this, a profiling and energy estimation tool was developed. The tool is based on a commercial instruction set simulator, augmented with functional, timing and power models for peripherals and memory blocks. The tool takes, as its input, the source code of the target application and the power and timing models for various system components, as well as instruction-level power models. It produces detailed profiling information, with aggregate performance and energy estimation for all functions in the source code. The profiling data are then used to select the majority of the energy and performance critical kernels in the application. The results of the above procedure lead to a refinement of the SoC architecture in terms of power and performance.

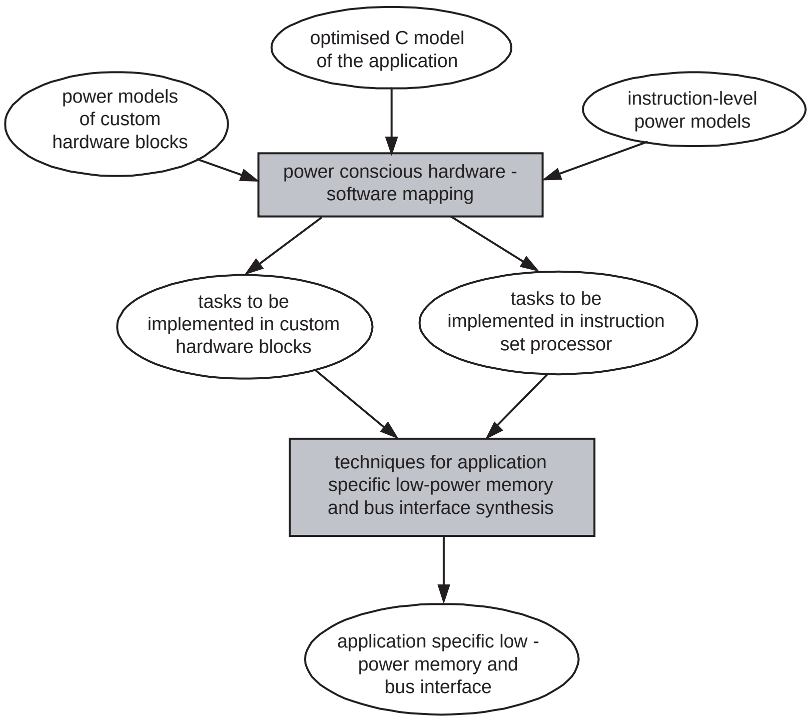
optimised C model of the application
power models of custom hardware blocks
instruction-level power models
power conscious hardware - software mapping
tasks to be implemented in custom hardware blocks
tasks to be implemented in instruction set processor
techniques for application specific low-power memory and bus interface synthesis
application specific low - power memory and bus interface

Fig. 9 *Architecture refinement techniques for low-power consumption*

Based on existing hardware-oriented memory partitioning techniques [27], an automatic optimisation methodology for on-chip memories to be used in embedded SoCs was also developed, and it will be applied to the on-chip memories of the SoC. First, the dynamic execution profile of the embedded application running on a given processor core is analysed, and then a multi-banked memory architecture is synthesised which is optimally fitted to such a profile. The rationale behind the approach is to partition the memory into multiple banks that can be independently accessed. The power per access is reduced as the size of a memory bank is decreased. On the other hand, as the number of banks increases, there is an unavoidable hardware overhead caused by: (i) duplication of addressing and control logic; and (ii) increased communication resources required to transfer information. Such an overhead manifests itself in increases in the power, access time and area that prevents arbitrarily fine partitioning. Hence, we need to find an optimal partition with a tight constraint on the maximum number of memory banks.

In a traditional approach, all addresses in the range are mapped to a single memory array, generally the smallest array in the library, that is large enough to contain the specified range, as shown in Fig. 10*a*. This solution is not optimal from the power dissipation viewpoint. Assume, for the sake of illustration, that the dynamic access profile is that shown in Fig. 10*b*. This profile is obtained by standard instruction-level simulators that are available for all processor cores (in our case the 32-bit ARM7 processor is considered). As shown, a small subset of the addresses in the range is very 'hot'. A power-optimal partitioned memory organisation is shown in Fig. 10*c*. It consists of three memories and a memory selection block. Two relatively large cuts contain the top and bottom parts of the range, while the hot addresses are stored into a small memory. The average power in accessing the memory hierarchy is decreased, because a large fraction of access addresses are concentrated in a small, power-efficient memory.

According to the defined architecture, there are in principle two internal memory structures on which memory partitioning could be applied with a significant chance of success: a single-port 16 KB SRAM and a dual-port 120 KB SRAM. In Table 3, the energy costs and savings before and after the memory partitioning are shown. The address traces provided, as input, to the memory-partitioning tool refer to an ARM7 TDMI processor. Memory addresses are thus expressed as 32-bit patterns. The processor is assumed to operate at a frequency of 150 MHz, and the bus frequency is assumed to be 60 MHz. The address traces used for determining the best memory architectures showed a total of 4096 accessed words in the 16 KB SRAM and 30 720 accessed words in the 120 KB SRAM. The chosen technology for the SRAM is the 0.18 μm process by STM. Each SRAM cell is made of six transistors. Energy savings achieved for the partitioned memory subsystem have been determined using the PowerChecker tool [28] for what concerns the memory components, and the Synopsys PowerCompiler [29] for what concerns the cost of the memory controller.

Another way to reduce power consumption in the context of the SoC architecture is to apply data encoding techniques [30] to the information transmitted on the SoC buses. These techniques consist of modifying the way the binary words are represented. For instance, a word could be represented by adding one parity bit. Bus encoding has been shown to be a very effective technique for power reduction, because even on-chip bus lines have a relatively large parasitic capacitance, compared to the cell capacitance. Buses are in fact a significant source of power in today's systems. For example, it has been measured that they may account for up to 30% of the total power in a modern microprocessor. Therefore, the relative impact of a significant energy reduction on a bus is non-negligible. Most bus encoding techniques are based on the idea of reducing the switching activity of a bus. The switching activity reduction can be achieved by inserting proper encoders and decoders (codec) at the sender and receiver's end of the bus. This scheme implies point-to-point buses. When comparing the energy efficiency of various encoding schemes, it is essential to also consider the impact of the encoder/decoder in the resulting

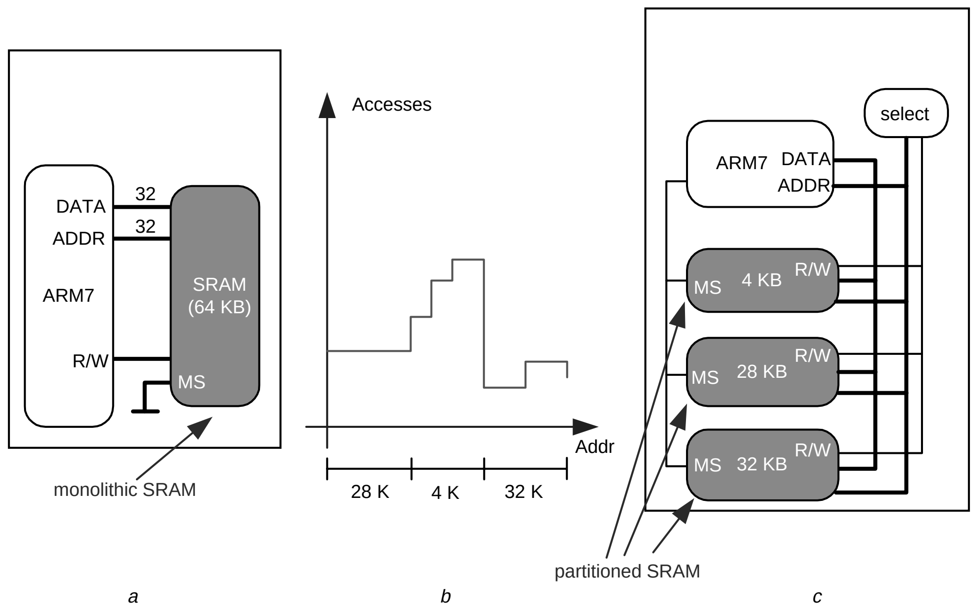

Fig. 10 *Example of memory partitioning*
a Traditional architecture
b Dynamic access profile
c Energy-optimal architecture

Table 3: Memory energy costs and saving

Memory components, KB	Energy cost, mJ	Components after partition	Energy cost, mJ	Overhead cost, mJ	Energy savings, %
16	0.634 444	block 1 (3.4 KB)	0.439 755	0.014 529	27.36
		block 2 (12.6 KB)	0.0067 767		
120	0.721 850	block 1 (1 KB)	0.133 081	0.016 516	70.23
		block 2 (119 KB)	0.0651 217		

power budget (in addition to timing and latency overheads). Encoders and decoders will obviously consume energy, which should not offset the power reduction achieved by reducing the total number of transitions.

Bus encoding techniques can be categorised according to three parameters: (i) redundancy; (ii) knowledge of statistics; and (iii) type of activity. Some of the encoding techniques that can be used to reduce the switching activity on the bus are: bus-invert encoding [31], Gray encoding [32], T0 encoding [33], Beach encoding [34], transition-based encoding [35], zone encoding [36], working zone encoding [37], adaptive encoding [30], frequent-value encoding [38].

Concerning the analysis of the potential sources of applicable low-power bus-encoding techniques within the SoC architectural template, we can identify two categories of buses: (i) the local bus; and (ii) the AMBA AHB bus. The main difference between them lies in the type of bus transactions supported. A local bus transmits plain words and, once the arbitration of the bus has been resolved, it can be considered as a point-to-point bus, where exactly one sender and one receiver exist. The AMBA bus falls in the more general category of multi-point buses. Here, the bus protocol distinguishes between control and data packets, allowing complex types of transaction types such as burst transfers. Furthermore, the possibility of arbitrating the bus through explicit commands introduces the concept of bus master(s) and slave(s), with the possibility of multiple masters. When considering energy optimisation, local, point-to-point buses typically offer more opportunities for power reduction than multi-point buses, thanks to their essential functional specification. As an intuitive justification to this fact, consider that most low-energy bus encoding schemes exploit the presence of a correlation between bus patterns (e.g. in address buses). Clearly, correlation does exist between 'information' patterns (data or addresses) but not between control patterns. In this sense, a multi-point bus that supports complex protocols will interleave data/addresses with control patterns, thereby reducing the amount of correlation on the travelling onto the bus.

In order to apply bus-encoding techniques on the local bus, an exploration tool has been developed, in which several of the aforementioned encoding methods have been implemented in software. Data and address traces that have been obtained by profiling of the embedded application are used as input to the exploration tool. The output of the exploration tool is the savings in terms of number of power consuming transitions. Synthesis and optimisation of the bus codec at the gate-level has been performed using the Synopsy's DesignCompiler [39], while placement and routing of buses and codec has been made using Cadence's SiliconEnsemble [40]. Energy consumption of the buses and of the codec has been estimated using Synopsy's Power-Compiler [29]. The STM (0.18 μm) standard cell library was used for the implementation of the bus interface logic (i.e. the encoder/decoder).

Given a set of input bus traces, regarding addresses and data, the first step we performed was to identify the most convenient encoding scheme using our exploration tool. Address traces did show a significant amount of sequentiality between consecutive patterns; as a consequence, encoding schemes such as Gray [32] and T0 [33] did provide the best results in terms of energy savings and codec complexity. In particular, also taking into account the available bus latency, the final decision was to go for the Gray scheme. If we look at the number of address bus transitions, we observe a reduction with respect to the unencoded bus in the order of 66%. This is clearly a purely theoretical result. In fact, when comparing the power efficiency of various encoding schemes, it is essential to also consider the impact of the codec in the resulting power budget. Concerning data traces, the situation is a bit more complicated. In fact, the bus profiles we have analysed do not seem to show any particular type of correlation, neither spatial, nor temporal. The expected reduction in transition count is thus smaller. As a consequence, it seems appropriate to focus on encoding schemes for which the cost of the codec is minimal, thus allowing some overall power reductions in spite of the small power reduction that may be achieved on the bus lines. Using the exploration tool with appropriate bus latency constraints leads to two possible options for data bus encoding. The bus-invert code [31] and the adaptive code [30]. More specifically, as the overhead introduced by the adaptive codec is significantly higher than that of the bus-invert scheme [31], we have opted for the latter. The achieved reduction in the number of bus transitions is smaller than in the case of the address bus, and it is around 57%.

After an exploration regarding the address bus, significant power savings (41.5%) are obtained when the bus length goes beyond 1000 μm as the cost of the encoder/decoder is only 64 μW. Regarding the data bus, the activity reduction provided by the bus-invert code [31] is smaller than for the case of Gray code [32] on the address bus. In addition to that, the implementation of the encoder/decoder is more expensive (around 143 μW). Power savings up to 39% are obtained when the bus length goes beyond 1000 μm.

6 FPGA-based prototype implementation

In order to continue with the prototyping of the SoC design, three advanced FPGA-based platforms were explored: NEC FPGA platform (SocLite, formely Socrates), ARM Integrator, and SISDA FPGA platform (CARMEN). Taking into consideration the technical, market and cost issues, we concluded to an FPGA platform using the ARM Integrator solution [41], for prototyping the SoC design. This solution is flexible enough for the defined architecture (multi-processor support and large user-defined gate count, AHB system bus) and there is no development effort for the FPGA board. The prototyping procedure will help to verify the SoC architecture (ARM cores, buses, bandwidth etc.), and to perform early software development and debugging on a hardware platform, early test of the baseband modem in a real environment (e.g. real signals with noise for the receiver) and early validation of the whole system (RF and analog parts included).

The block diagram of the FPGA-based architecture is illustrated in Fig. 11. In this, two core modules including the protocol and the lower MAC/modem control processors are used, along with two logic modules for the baseband modem and its interfaces. Concerning the partitioning of the required logic to the logic modules, the top logic module implements the modem's transmit path, the time domain of the modem's receive path and the MAC/PHY interface, while the bottom logic module implements the data and the frequency domains of the modem's receive path. Each logic module hosts a XILINX Virtex E 2000 FPGA [42] (0.18 μm) with 500 K usable gates, 640 KB of additional RAM (BlockRAM), and built-in clock management circuitry (eight DLLs). Figure 12, shows a more detailed view of the logic modules in the ARM Integrator platform. The total utilisation of the bottom logic module (FPGA) is 85%. The total utilisation of the top logic module is 89%. The utilisation per resource type for the bottom and the top logic modules is presented in Table 4. In order to have a full system design of the AP or MT wireless LAN 5 GHz components, the baseband modem's functionality is followed by an IF (20 to 880 MHz) and an RF (880 MHz to 5 GHz) stage. The analog-to-digital and digital-to-analog conversion, for communicating with the IF analog front ends of the receiver and the transmitter respectively, is implemented on a separate board which seats on a dedicated connector for external communications on the 'top' of the stack of logic modules. Also the communication with the PCI or Ethernet interface is done through that port.

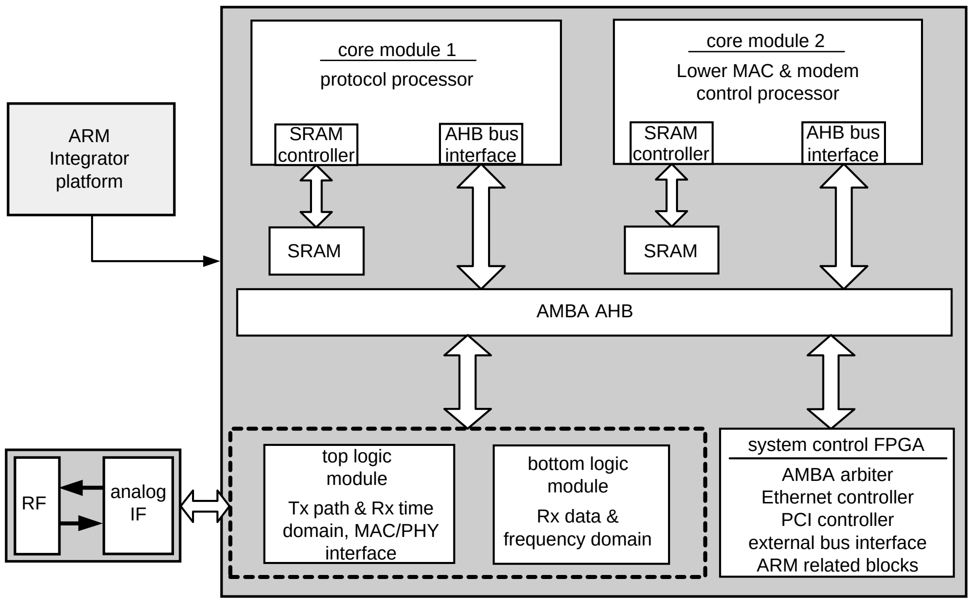

Fig. 11 *Block diagram of the FPGA-based architecture*

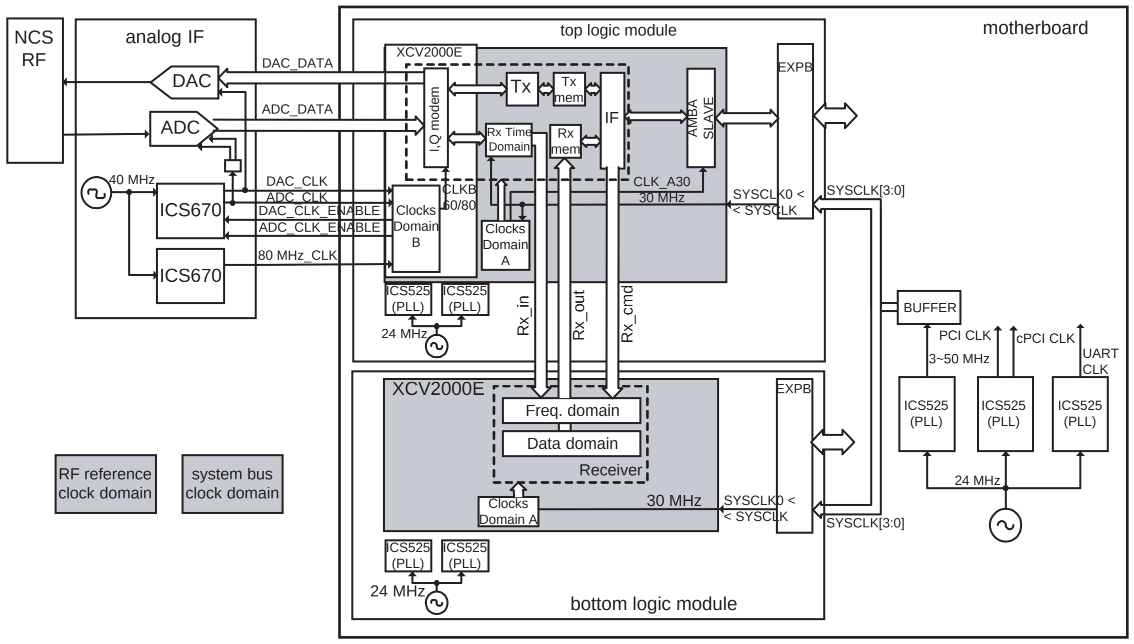

Fig. 12 *Detailed view of the logic modules*

Table 4: Utilisation of the top and the bottom logic modules (FPGAs)

	Used		Utilisation, %	
Resource	Bottom FPGA	Top FPGA	Bottom FPGA	Top FPGA
I/Os	93	312	18.16	60.93
Function generators	14 923	16 527	38.86	43.04
CLB slices	12 164	11 252	63.35	58.60
DFFs or Latches	6368	8544	15.60	20.94

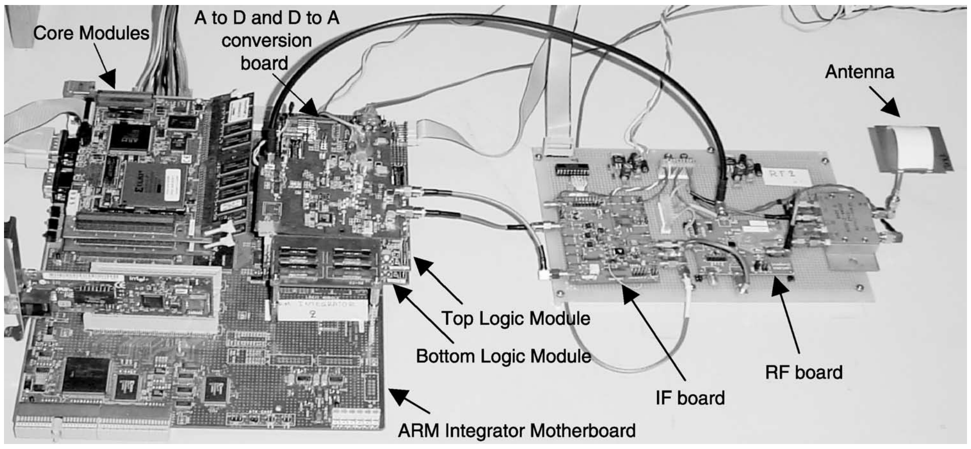

Fig. 13 *Photograph of the ARM Integrator platform along with the IF, RF boards and the antenna*

Measurements showed an excellent performance of the modem i.e. a 23 dB signal is needed for the 64-QAM when for QPSK 10 dB is sufficient. Finally, Fig. 13 is a photograph that illustrates the ARM Integrator platform along with the IF, RF boards and the antenna, which are needed for the implementation of the MT or AP wireless LAN 5 GHz devices. By using the illustrated system twice (one operating as an AP and a second operating as a MT), a 5 GHz wireless system is demonstrated. In Fig. 13, the location of each component is indicated.

The functional evaluation/verification of the whole system on the FPGA-based platform gives valuable results and proves that we can have a real-time SoC architecture capable of implementing the baseband, MAC and DLC processing of both standards. This verification can lead to new refinement of the SoC architecture to meet real-time processing requirements. This step can change e.g. the clock frequency of specific hardware blocks or can lead to a decision that more functions of the MAC/DLC processing of both standards need hardware support. Other important features of the final SoC (which will be implemented in the STM 0.18 μm process) that were not implemented in this first phase (ARM Integrator) are, the clocking strategy, design for testability, and enhanced debugging circuitry and strategy.

7 Conclusions

The design of a low-power SoC for wireless LANs has been described. The system will realise the HIPERLAN/2 and IEEE 802.11a standards and will cover the operation of both MT and AP devices. Emphasis was given on the options, constraints and motivations for the taken design decisions, and on the followed design steps, starting from the system specifications until the architecture definition and the system implementation. Also, the application of an energy-aware methodology was presented. The prototype implementation of the system on the ARM Integrator platform with an architecture that is very close to the one that is followed in the ASIC phase proves that the proposed flexible SoC architecture can meet the functional and timing requirements of both of the WLAN standards. This architecture in terms of functionality and timing with the refinements for low-power consumption will lead to a low-power WLAN processor.

The FPGA-based implementation of the WLAN processor cannot so far prove the savings in power dissipation. The purpose of the FPGA-based implementation on the ARM Integrator platform was to prove that with a FPGA-based architecture that is close to the final SoC architecture the WLAN processor is fully functional for both standards and can meet the real-time constraints. This aim was successfully achieved. The architecture refinements for low-power described in Section 5 will lead to a low-power flexible WLAN processor in the ASIC design phase. As, at this point in time, the chip placement is not yet available, in order to assess the quality of our results we have derived the achievable power savings, i.e. quantified results for the power gain by analysing some significant design corners. Energy savings achieved for the partitioned memory subsystem have been determined using the PowerChecker tool for what concerns the memory components and the PowerCompiler tool for what concerns the cost of the memory controller. Concerning the main memory components of the SoC, power gains of 27.36 and 70.23% can be achieved in the single-port and dual-port on-chip SRAMs, respectively. In order to determine the power gain for the bus encoding techniques, routed buses were extracted at the layout-level using the SiliconEnsemble tool. After an exploration using several bus-encoding techniques, power gains of 41.5 and 39% can be achieved at the address and data bus respectively, when the local bus (shown in Fig. 6) length goes beyond 1000 μm. In addition, after the collection of profiling data obtained from simulations, some power-consuming system functions have been identified and would benefit from implementation in dedicated hardware units, leading to additional reductions in the consumed power.

8 Acknowledgments

This work was partially supported by the project IST-2000-30093 EASY (Energy-aware System-on-Chip design of the HIPERLAN/2 standard), funded by the EU. The authors also thank the reviewers for their valuable suggestions and comments to improve the paper's structure and research results.

9 References

1 In Stat/MDR, 'WLAN chipset market – The incredible journey is just beginning', Report No. IN020271WT, March 2002
2 van Nee, R., Awater, G., Morikura, M., Takanashi, H., Webster, M., and Halford, K.W.: 'New high-rate wireless LAN standards', *IEEE Commun. Mag.*, 1999, **37**, (12), pp. 82–88
3 European Telecommunications Institute (ETSI), Broadband Radio Access Network (BRAN), 'HIPERLAN Type 2: System Overview', TR 101 683, February 2000
4 European Telecommunications Institute (ETSI), Broadband Radio Access Network (BRAN), 'HIPERLAN Type 2: Physical (PHY) Layer', TS 101 475, April 2000
5 The Institute of Electrical and Electronics Engineers, 'Supplement to IEEE standard for information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer in the 5 GHz band', IEEE Std. 802.11a, 1999
6 Doufexi, A., Armour, S., Butler, M., Nix, A., Bull, D., and McGeehan, T.: 'A comparison of the HIPE-RLAN/2 and IEEE 802.11a wireless LAN standards', *IEEE Commun. Mag.*, 2002, **40**, (5), pp. 172–180
7 Rashinkar, P., Paterson, P., and Singh, L.: 'System-on-Chip verification' (Kluwer Academic Publishers, Boston, MA, 2001)
8 Berger, A.S.: 'Embedded system design: An introduction to processes, tools and techniques' (CMP Books, Lawrence, KS, 2002)
9 Rabaey, J.M., and Pedram, M.: 'Low-power design methodologies' (Kluwer Academic Publishers, Boston, MA, 1996)
10 Kneip, J., Weiss, M., Drescher, W., Aue, V., Strobel, J., Oberthur, T., Bole, M., and Fettweis, G.: 'Single chip programmable baseband ASSP for 5 GHz wireless LAN applications', *IEICE Trans. Electron.*, 2002, **85**, (2), pp. 359–367
11 Eberle, W., Derudder, V., Vanwijnsberghe, G., Vergara, M., Deneire, L., Van der Perre, L., Engels, M.G.E., Bolsens, I., and De Man, H.: '80-Mb/s QPSK and 72-Mb/s 64-QAM flexible and scalable digital OFDM transceiver ASICs for wireless local area networks in the 5-GHz band', *IEEE J. Solid-State Circuits*, 2001, **36**, (11), pp. 1829–1838
12 van Nee, R., and Prasad, R.: 'OFDM for mobile multimedia communications' (Artech House, Boston, MA, 1999)
13 European Telecommunications Institute (ETSI), Broadband Radio Access Network (BRAN), 'HIPERLAN Type 2: Data Link Control (DLC) layer – Part 1: Basic data transport functions', TS 101 761-1, Nov. 2000
14 European Telecommunications Institute (ETSI), Broadband Radio Access Network (BRAN), 'HIPERLAN Type 2: Data Link Control (DLC) layer – Part 2: Radio Link Control (RLC) sublayer', TS 101 761-2, April 2000
15 The Institute of Electrical and Electronics Engineers, 'Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications', ANSI/IEEE Std. 802.11, 1999
16 Rational Software Corp., 'Rational Rose RealTime: Modeling language guide', May 2002
17 ARM Limited, 'AMBA Specification', Rev. 2.0, May 1999
18 ARM Limited, 'ARM7TDMI technical reference manual', Rev. 4.0, 2001
19 Khun-Jush, J., Schramm, P., Wachsmann, U., and Wegner, F.: 'Structure and performance of the HIPERLAN/2 physical layer'. Proc. IEEE Vehicular Technology Conf., Houston, TX, 16–20 May 1999, vol. 5, pp. 2667–2671
20 Schmidl, T.M., and Cox, D.C.: 'Robust frequency and timing synchronization for OFDM', *IEEE Trans. Commun.*, 1997, **45**, (12), pp. 1613–1621
21 Catthoor, F., Wuytack, S., De Greef, E., Balasa, F., Nachtergaele, L., and Vandecappelle, A.: 'Custom memory management methodology' (Kluwer Academic Publishers, Dordrecht, The Netherlands, 1998)
22 Tiwari, V., Malik, S., Wolfe, A., and Lee, M.T-C.: 'Instruction-level power analysis and optimizations of software', *Journal VLSI Signal Process.*, 1996, **13**, (2-3), pp. 223–238
23 Benini, L., De Micheli, G., Lioy, A., Macii, E., Odasso, G., and Poncino, M.: 'Computational kernels and their application to sequential power optimization'. Proc. ACM/IEEE Design automation Conf., San Francisco, CA, 15–19 June 1998, pp. 764–769
24 Henkel, J.: 'A low-power hardware/software partitioning approach for core-based embedded systems'. Proc. ACM/IEEE Design automation Conf., New Orleans, LA, 21–25 June 1999, pp. 122–127
25 Benini, L., Macii, A., Macii, E., and Poncino, M.: 'Increasing energy efficiency of embedded systems by application-specific memory hierarchy generation', *IEEE Des. Test Comput.*, 2000, **17**, (2), pp. 74–85
26 Benini, L., Macii, A., Macii, E., Poncino, M., and Scarsi, R.: 'Architectures and synthesis algorithms for power-efficient bus interfaces', *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2000, **19**, (9), pp. 969–980
27 Pedram, M.: 'Power aware design methodologies' (Kluwer Academic Publishers, Dordrecht, The Netherlands, 2002)
28 BullDAST s.r.l., 'PowerChecker User Manual', Version 3.1, 2003
29 Synopsys Inc., 'Power Compiler data sheet', http://www.synopsys.com/products/power/power_ds.html, September 2001
30 Ramrasad, S., Shanbhag, N., and Hajj, I.: 'Signal coding for low-power: Fundamental limits and practical realizations'. Proc. IEEE Int. Symp. on Circuits and Systems, Monterey, CA, 31 May–3 June 1998, vol. 2, pp. 1–4
31 Stan, M.R., and Burleson, W.P.: 'Bus-invert coding for low-power I/O', *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 1995, **3**, (1), pp. 49–58

32 Mehta, H., Owens, R.M., and Irwin, M.J.: 'Some issues in Gray code addressing'. Proc. ACM/IEEE Great Lakes Symp. on VLSI, Arnes, IA, 22–23 March 1996, pp. 178–180

33 Benini, L., De Micheli, G., Macii, E., Sciuto, D., and Silvano, C.: 'Asymptotic zero-transition activity encoding for address buses in low-power microprocessor-based systems'. Proc. ACM/IEEE Great Lakes Symp. on VLSI, Urbana, IL, 13–15 March 1997, pp. 77–82

34 Benini, L., De Micheli, G., Macii, E., Poncino, M., and Quer, S.: 'Power optimization of core-based systems by address bus encoding', *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 1998, **6**, (4), pp. 554–562

35 Aghaghiri, V., Fallah, F., and Pedram, M.: 'Irredundant address bus encoding for low-power'. Proc. ACM/IEEE Int. Symp. Low-Power Electronics and Design, Huntington Beach, CA, 6–7 August 2001, pp. 182–187

36 Aghaghiri, V., Fallah, F., and Pedram, M.: 'EZ encoding: A class of irredundant low-power codes for data, address and multiplexed address buses'. Proc. Design Automation and Test in Europe Conf., Paris, France, 4–8 March 2002, pp. 1102–1105

37 Mussol, E., Lang, T., and Cortadella, J.: 'Working-zone encoding for reducing the energy in micro-processor address buses', *IEEE Trans. Very Large Scale Integ. (VLSI) Syst.*, 1998, **6**, (4), pp. 568–572

38 Yang, J., and Gupta, R.: 'FV encoding for low-power data I/O'. Proc. ACM/IEEE Int. Symp. on Low-power electronics and design, Huntington Beach, CA, 6–7 August 2001, pp. 84–87

39 Synopsys Inc., 'Design Compiler data sheet', http://www.synopsys.com/products/logic/design_compiler.html, September 2003

40 Cadence Design Systems Inc., 'Silicon Ensemble PKS datasheet', http://cadence.com/datasheets/silicon_ens_pks.html, April 2002

41 ARM Limited, 'Integrator/AP user guide', 1999-2001

42 Xilinx Inc., 'Virtex-E 1.8V Field Programmable Gate Arrays, Preliminary Product Specification', Version 2.0, April 2001