Hindawi Publishing Corporation
VLSI Design
Volume 2012, Article ID 450302, 17 pages
doi:10.1155/2012/450302

Research Article

Homogeneous and Heterogeneous MPSoC Architectures with Network-On-Chip Connectivity for Low-Power and Real-Time Multimedia Signal Processing

Sergio Saponara and Luca Fanucci

Department of Information Engineering, University of Pisa, Via G. Caruso 16, 56122 Pisa, Italy

Correspondence should be addressed to Sergio Saponara, sergio.saponara@iet.unipi.it

Received 28 March 2012; Accepted 20 June 2012

Academic Editor: Marcelo Lubaszewski

Two multiprocessor system-on-chip (MPSoC) architectures are proposed and compared in the paper with reference to audio and video processing applications. One architecture exploits a homogeneous topology; it consists of 8 identical tiles, each made of a 32-bit RISC core enhanced by a 64-bit DSP coprocessor with local memory. The other MPSoC architecture exploits a heterogeneous-tile topology with on-chip distributed memory resources; the tiles act as application specific processors supporting a different class of algorithms. In both architectures, the multiple tiles are interconnected by a network-on-chip (NoC) infrastructure, through network interfaces and routers, which allows parallel operations of the multiple tiles. The functional performances and the implementation complexity of the NoC-based MPSoC architectures are assessed by synthesis results in submicron CMOS technology. Among the large set of supported algorithms, two case studies are considered: the real-time implementation of an H.264/MPEG AVC video codec and of a low-distortion digital audio amplifier. The heterogeneous architecture ensures a higher power efficiency and a smaller area occupation and is more suited for low-power multimedia processing, such as in mobile devices. The homogeneous scheme allows for a higher flexibility and easier system scalability and is more suited for general-purpose DSP tasks in power-supplied devices.

1. Introduction

Telecommunications, consumer, and infotainment applications are characterized by a growing interest in the real-time and low-power implementation of DSP techniques, for images, videos, and audio, to improve system performances in terms of coding efficiency, visualization quality, and audio reproduction fidelity. At the state-of-the-art, several algorithms have been proposed [1–13] which represent mature solutions for video and audio coding, noise and artifact suppression, high-dynamic-range signal management, interpolation, video motion estimation, and low-distortion digital audio amplification. The implementation of such algorithms in real-time and with low-power consumption is a challenging open issue. To face the high computational power required by real-time multimedia processing applications multicore architectures are needed [14, 15]. At the state-of-the-art, in most cases, the proposed algorithms still refer to off-line software implementation on a programmable platform. Even in case real-time processing is obtained by using graphic-specific processing units (GPUs) [16–20], the relevant power consumption is unsuitable for embedded or mobile systems. A GPU has a power consumption ranging from tens to hundreds Watts, depending on the workload [21]. Dedicated integrated circuits (ICs) have been proposed in literature [12, 22–30] whose power consumption is limited to hundreds mW; however, they are dedicated to a specific algorithm, for example, motion estimation for interframe video coding in [28] or dynamic range compression for display of mobile devices in [27] or audio oversampling and noise shaping in [12, 30]. Instead, a programmable solution covering multiple tasks is needed. The availability of low-power and integrated solutions, offering enough programmability to support different classes of audio and image/video processing algorithms, is a strategic target to be achieved to foster the adoption of such techniques in new

application fields. The problem is not only the design of the computing architecture but also of the communication infrastructure due to the nature of multimedia algorithms, dominated by data transfer and storage costs [31].

To address the above issues, this work presents 2 programmable and scalable MPSoC architectures for multimedia signal processing:

(i) One is based on the use of an array of multiple heterogeneous tiles, each acting as an application-specific instruction set processor (ASIP) covering a DSP algorithmic class for 1D (audio) and/or 2D (image and video frames) signals;

(ii) The other is based on an array of homogeneous tiles, each composed of a 32-bit RISC core plus a 64-bit VLIW computing unit.

In both cases, a network-on-chip (NoC) with Spidergon topology is adopted as communication infrastructure to overcome the limits of classic circuit-switched bus. The paper is structured as follows. Section 2 reviews state-of-the-art computing architectures for real-time multimedia processing. Sections 3 and 4 describe the homogeneous-tile and the heterogeneous-tile MPSoCs, respectively, for audio, image, and video processing. Section 5 discusses the design of the NoC infrastructure for both architectures. Functional characterization of the MPSoC architectures with reference to two case studies is reported in Section 6: an H.264/MPEG4 AVC video coder and a low-distortion reproduction audio system. Implementation results in submicron CMOS technology for the MPSoC architectures are discussed in Section 7. Conclusions are drawn in Section 8.

2. Computing Architectures for Multimedia Processing

The computing architectures proposed in literature to achieve multimedia real-time processing can be clustered into 3 main categories.

(1) Hardware platforms achieving real time for handheld/mobile devices at power costs lower than 1 Watt, but limited to a specific algorithm, for example, [22–25, 27–29, 32, 33].

(2) Software implementation on general purpose processors with clock frequencies in the GHz domain and with a single-core, for example, Atom in [26], or multicores, for example, Core2 in [26, 34], depending on the computational load. The power consumption of such solutions is up to tens of Watts. The real-time processing of multimedia algorithms is typically limited to simple tasks or to small image/video formats.

(3) Software implementation on massively parallel processors, such as GPUs [16–20] with computational throughput of billions of floating point operations per second (GFLOPS) ensuring real-time multimedia processing but with power cost up to hundreds of Watts.

From a technology point of view, although the performances achievable with FPGA (field programmable gate array) are continuously increasing, the realization of dedicated or programmable platforms on submicron CMOS technology (with standard cells or full-custom or hybrid design flows) is still the approach ensuring the best trade-off between power consumption and computational/storage capabilities. As example the Virtex 6 or the StratixIV FPGA families include devices with several Mbits of RAM on chip, several hundreds of DSP blocks with multiply and accumulate (MAC) capabilities at hundreds of MHz, I/O transceivers running up to several Gbits/s. Both configurable hardware logic and hard or soft microprocessor cores are available in today FPGAs. However, the overall complexity of such FPGAs in 45 nm silicon technology is in the order of billions of transistors, occupying an area of several hundreds of mm^2, and the power consumption is up to tens of Watts. As example, from the analysis carried out in [35], the Virtex6 LX760 FPGA device offers a DSP computational capability of roughly 380 GFLOPS when implementing a fast Fourier transform (FFT) algorithm but for a power cost higher than 50 W. From [35] it emerges how the LX760 FPGA has worse performances, measured as the ratio between computational capability and power consumption (GFLOPS/W) or area (GFLOPS/mm^2), by one order of magnitude versus a dedicated ASIC design and similar performances to a GPU (GTX480 from NVidia) realized in a 45 nm CMOS technology. Moreover, the market of multimedia and consumer platforms is a large volume one and hence is suited for CMOS designs characterized by higher nonrecurrent costs but lower recurrent costs versus FPGAs.

Although a comparison of hardware/software platforms with different architectures, instruction sets, and computing performances is difficult to implement, Figure 1 provides a visual representation of the trade-off between die size (due to circuit complexity spent for increased parallelism and hence increased performance), power consumption, and computational capabilities in state-of-art single-core and multicore systems. Reported data refer to platforms realized in 45 nm silicon technology nodes and to operations on 32-bit data (integer or single-precision floating-point).

Figure 1 highlights that considering as performance metric the ratio GOPS/mm^2 there are two different trend-lines: one for platforms optimized for general purpose computing applications (e.g., Atom, Core2) and one for platforms optimized for DSP applications. When the level of parallelism increases (e.g., from 1 in the SP_CELL to many cores in the CELL-BE or Tile64 or GTX480), the difference between the two trend-lines is reduced since at high parallelization levels the performance bottleneck is represented by communication and the computational power increases slowly versus the number of cores.

The target of our work is designing programmable multi core architectures with power consumption limited to few Watts, for example, <3Watts, a die size around 50 mm^2, and enough computing performances to ensure real-time processing of CD and DVD-quality audio (up to 96 kHz, 16 bit/sample) and high definition (HD) images and videos with frame resolutions up to 1080×720 or 1024×768.

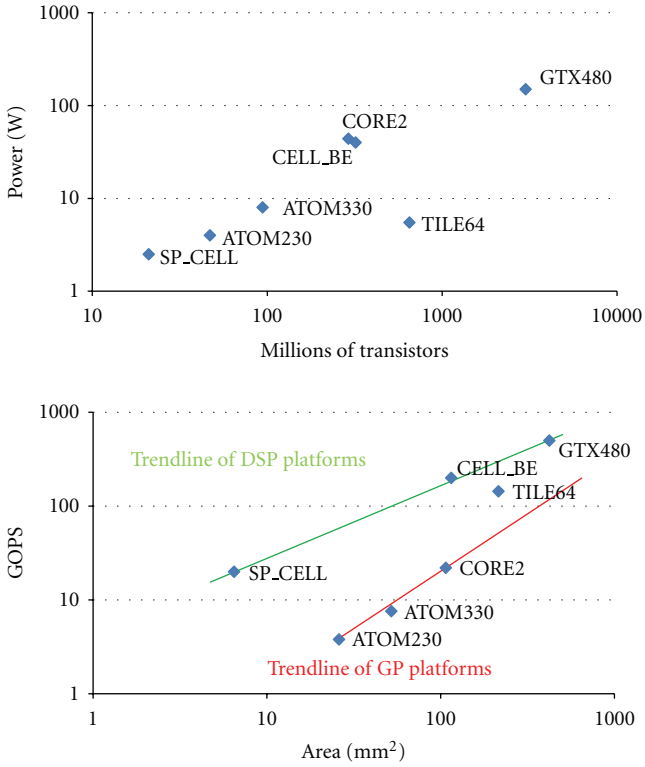

Figure 1: Die size, number of transistors, computational power (GOPS), and power consumption of state-of-art single-core, ATOM 230 and 1 synergistic processor (SP) of the Cell BE, and multi core systems, ATOM330, Fermi GPU GTX480, Core2, Cell BE, and Tile64.

The display and camera resolution of smart phones and handheld multimedia devices is typically lower, for example, 30 Hz 352 × 288 CIF format or 640 × 480 VGA format [36] and hence the functional assessment of the MPSoC architecture in Section 6 has been carried out mainly with CIF and VGA test input videos. Similarly, the most common audio sampling rates in handheld devices are 44.1 and 48 kHz [36], used for the tests in Section 6.

As example of pure software implementations of a scale invariance feature transform (SIFT) algorithm for image feature extraction [26] and of a 3D DCT/IDCT video codec have been realized on the Intel Core 2 processor. In [26], an Intel Core 2 6600@2.4 GHz implements the SIFT algorithm with a processing-rate of 16 Hz for VGA format. In [34], an Intel Core 2 6300@1.86 GHz achieves real-time processing of 3D DCT/IDCT codec for 24 Hz VGA format. Therefore, software implementations of multimedia processing tasks are possible although limited to suboptimal techniques. since a 3D DCT/IDCT codec has poor coding performance, in terms of compression efficiency, when compared to the H.26x/MPEGx motion-compensated hybrid scheme considered in Section 6. A Core 2 processor, realized in 65 nm CMOS technology, integrates 2 64-bit CPU cores and up to 4 Mbytes of L2 cache memory. It has a die size of roughly 143 mm^2 and a transistor count of about 300 millions; the power consumption is up to 65 Watts, unsuited for battery powered, mobile or handheld terminals. Realizations of the same Core 2 processor in 45 nm CMOS technology, with a clock frequency still of 2.4 GHz and a L2 cache of 6 Mbytes, occupy an area of 107 mm^2 with a power consumption up to 44 W.

A lower power consumption, in the order of several Watts, can be achieved by targeting single-core CPUs such as the Intel ATOM 230@1.6 GHz used in [26] to implement the already-mentioned SIFT algorithm. The Intel ATOM processor, realized in 45 nm CMOS technology, integrates a single 32-bit CPU core with 512 kbytes of L2 cache. It has a die size of 26 mm^2, a transistor count of 47 millions and a power consumption of few Watts, less than 4 Watts. The lower size and power cost of ATOM versus Core2 is paid in terms of frame-rate performances. Indeed, in [26], only 5 images per second can be processed with the single-core ATOM processor which is not enough for real-time video representations.

As a matter of fact the computational capability of the Atom core at 1.6 GHz, measured with Dhrystone benchmarks, is around 3,8 billions of operations per second (GOPS) while that of a dual-core Core2 processor at 2.4 GHz is roughly 5.5 times higher.

Dual-core versions of the 32-bit ATOM processor have been realized in 45 nm CMOS technology, such as the Intel ATOM 330@1.6 GHz, with an overall L2 cache of 1 Mbyte: the computational capabilities are doubled versus the single-core ATOM230 processor but at the expenses of increased area, transistor count and power consumption by a factor of 2: 52 mm^2, 94 millions of transistor and less than 8 Watts, respectively.

Multi core platforms for embedded multimedia applications, based on ARM processors, have been presented in literature as MALI family: as example the MALI-200 has 2 ARM cores with AMBA AXI 3 interface; realized in 65 nm CMOS technology it occupies 4 mm^2 and has a computational capability of 275 Millions of pixels/s at 275 MHz. Such computational capability is not enough to support complex multimedia algorithms, such as a complete H.264 encoder at high definition; indeed for such applications dedicated coprocessors, such as the MALI VE6 video engine, has to be added [37].

Real-time performance with complex image/video processing algorithms, such as a complete H.264/MPEG4 AVC video coder, can be achieved with GPUs, for example, the cell broadband engine (BE) from IBM [16, 17, 19, 20, 38] or the FERMI architecture from NVIDIA [18, 35]. The cell BE has a capability of more than 200 GFLOPS with a 3.2 GHz clock when realized in 90 nm CMOS silicon-on-insulator (SOI) technology. In 65 nm and in 45 nm CMOS SOI, the clock frequency can rise up to 6 GHz and the peak computational throughput is roughly 380 GFLOPS [20, 38]. The cell BE integrates on-chip one 64-bit power processor element (PPE) core in charge of operation scheduling and data flow control, and 8 synergistic processors (SPs) dedicated to DSP computing and working according to an SIMD (single instruction multiple data) scheme. The cell BE has 256 kbytes of local memory for each SP, 32 kbytes of L1 data and instruction caches for the PPE, and 512 kbytes L2

shared cache. Each SP core has a transistor count of 21 millions, an area of 14.77 mm^2 and a power cost of 5 Watts in 90 nm CMOS technology [19]. Each SP has an area of 11.08 $mm^2$and 6.47 mm^2 in 65 nm and 45 nm technology respectively. The multi core cell BE has an area of 235 mm^2 in 90 nm CMOS technology, 174 mm^2 in 65 nm CMOS, and 115 mm^2 in 45 nm CMOS SOI. Therefore, in submicron CMOS technology by scaling the same architecture from 90 nm to 65 nm and then to 45 nm technology node, the overall processor area scales with a factor roughly $\lambda 2/\lambda 1$ being $\lambda 2$ and $\lambda 1$ the channel length of the MOS devices in the two technologies; in the past, see [39, 40], as example, with larger technology nodes above 100 nm such ratio was $(\lambda 2/\lambda 1)^{\theta}$ being θ a fitting parameter between 1 and 2. The power cost of the Cell BE at 3.2 GHz is up to 80 W in 90 nm CMOS technology while at the same clock frequency in 45 nm CMOS technology is up to 40 W.

A high parallel multi core processor, called AsAP, has been proposed in [41]: it consists of 36 basic processors (32-bit RISC cores with 9 stage pipeline, 40-bit multiply and accumulate unit and 8Kbit of local memory) connected through a nearest-neighbor topology and running at 500 MHz. The area of the whole chip is 32 mm^2 (5.65 mm $\times$ 5.68 mm) and the power consumption is 1.15 W in typical conditions. Although the area and power consumption of AsAP meet the target requirements of our application, the offered computational power is enough only for real-time JPEG coder, that is, encoding of still images not H.264/AVC video coding. This is mainly due to the limited DSP capabilities of the basic AsAP tile. In [42], an evolution of the AsAP architecture, nicknamed AsAP2, has been proposed in 65 nm CMOS technology composed of 164 basic processors running at 1.2 GHz with a computational capability of 196.8 billions of multiply and accumulate operations per second (GMAC/s). AsAP2 also integrates 3 coprocessors dedicated to computing intensive DSP tasks such as FFT, video motion estimation, and viterbi decoder, and 3 SRAM blocks each of 16 kbytes. The overall complexity of AsAP2 rises to 55 millions of transistors for an area of roughly 40 mm^2. The computational power is enough to support 1080 HD multimedia applications at 30 frames/s with a power consumption of roughly 10.5 W.

A tentative realization of an MPSoC based on heterogeneous processors has been done in [43], which however is limited to a specific application domain (only the enhancement of image and video signals) while the aim of this work is more general: targeting real-time multimedia (image, video and also audio) tasks. GPUs are now evolving as massively parallel platforms for graphics but also for computing-intensive general purpose algorithms with high degree of parallelism. The Fermi architecture is composed of 512 CUDA (compute unified device architecture) processing cores hierarchically organized in 16 streaming multiprocessors each with 64 kbyte L1 cache and 128 kbyte local register file, and sharing a common 768 kbyte L2 cache. The total amount of on-chip memory is roughly 4 Mbytes. The Fermi GPU architecture has also PCIe host interface and six 64-bit DDR (double data rate) DRAM interfaces. Each CUDA core is capable of both integer and floating point operations with 64-bit results. Each of the 16 streaming processors has also 4 40-bit special function units for fast approximation of nonlinear operators (square root, sin, cos, exp, log functions) and 16 load/store units. The Fermi NVIDIA GPU leads to high computational power, up to 1500 GFLOPS in single-precision, orders of magnitude higher than Core 2 or Atom general-purpose processors. Although real-time processing is not an issue for such GPUs, their area and power consumption are suited only for desktop applications and workstations, not for handheld or mobile devices. As an example from the analysis in [35], the GTX480 Fermi NVIDIA, realized in 45 nm CMOS technology, has a transistor count of 3 billions, an area occupation for the core of 422 mm^2, and the power consumption higher than 150 W when implementing algorithms with a computational workload higher than 500 GFLOPs. Scaled versions of CUDA-based GPU exist: for example, the NVIDIA Quadro 4000 GPU has 256 CUDA cores, 4 64-bit DDR DRAM interface, and a maximum power consumption of 100 W. The massively parallel approach has been followed also by TILERA: its Tile 64 processor proposed in [44] realizes in CMOS 90 nm an array of 64 basic processors with 64-bit instruction set and 32-bit integer ALU communicating through a 2D Mesh on-chip interconnect. The Tile 64 reaches a computational throughput of 144 integer GOPS (32-bit operations) and 384 integer GOPS (8-bit operations) with 750 MHz clock. Its complexity amounts to 615 millions of transistors; the area is 433 mm^2, 215 mm^2 in 45 nm, and the power consumption at full load is roughly 11 W.

To reduce the power consumption of general purpose CPUs or high-parallel GPUs, several configurable hardware designs have been proposed in literature. Such designs have a power consumption below 1 Watt but achieve real-time processing only for a specific task and for low/mid-size image and video formats. Most of the proposed solutions use an RISC-like processing core with local instruction/data cache, for operation and data control flow, enhanced by external DRAM controller and hardware accelerators for the most demanding tasks. However, these hardware designs are customized for a specific algorithm: for example, dynamic image compression in [27], feature extraction in [26], 3D rendering in [23], video or audio coding in [12, 25]. If a mobile device needs a number of these functions several ICs should be used and mounted on a PCB board. Moreover, the communication between the RISC core and the coprocessors is based on classic bus architectures, for example, AMBA AHB bus. This limits the scalability of the solution and represents a bottleneck in case of multiple algorithms to be executed in parallel.

To overcome the limits of the state-of-the-art the following sections present two MPSoC architectures to achieve real-time processing for multiple algorithmic classes, running in parallel, at a reasonable power cost. The target is a power consumption of few Watts when implementing complex systems such as a complete H.264/MPEG AVC encoder in real time. To this aim, the following ideas will be exploited

(i) NoC as on-chip communication infrastructure, easing architecture scalability and management of

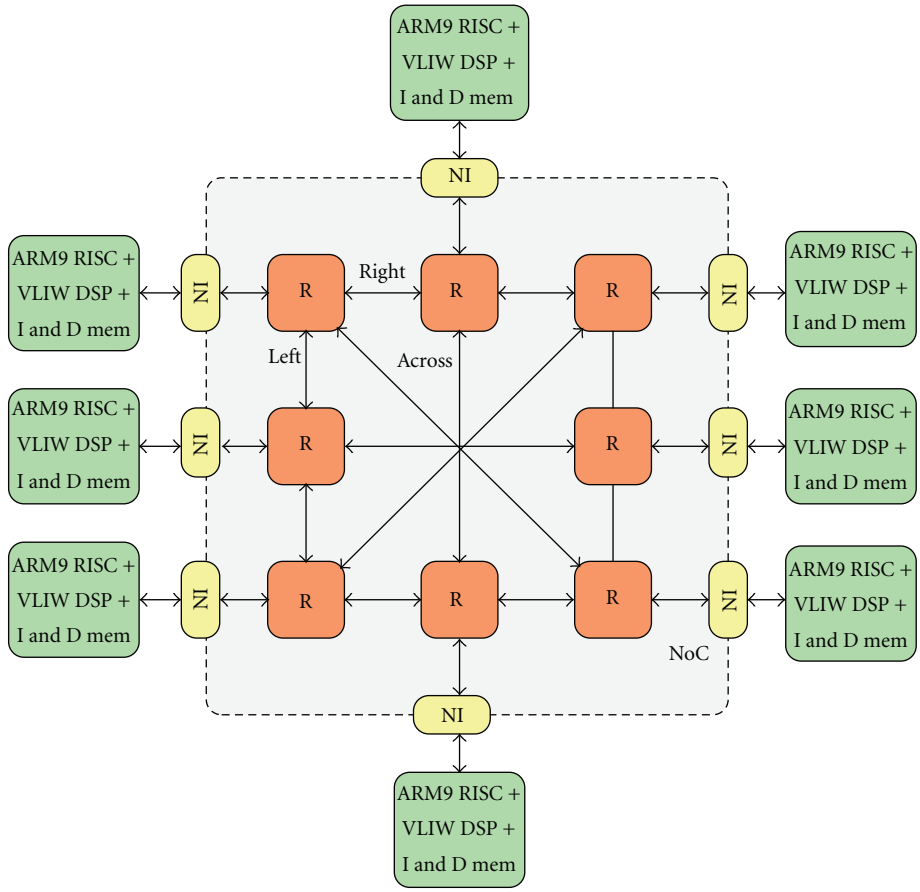

Figure 2: Block diagram of the Homogeneous NoC-based MPSoC architecture.

computing tasks with irregular or massively parallel data flow thanks to the packet-switched communication scheme.

(ii) Use of multiple tiles, with heterogeneous and homogeneous approaches, working in parallel to provide the computational power required by real-time multimedia tasks.

3. Homogeneous NoC-Based MPSoC

The proposed homogeneous MPSoC architecture is the one developed in the SHAPES (scalable software hardware architecture platform for embedded systems) European project involving among the partners University of Pisa, STMicroelectronics and ATMEL. In SHAPES, eight identical tiles building a scalable MPSoC are interconnected by means of a packet-switched on-chip network, see Figure 2. The NoC uses eight 4-port Routers (R in Figure 2): three router ports are dedicated to *across*, *right*, and *left* router-to-router connections and one port is dedicated to the connections between the NoC and the tiles. Conversion of protocol, data size, and clock frequency between the NoC and each tile is managed by eight Network Interfaces (NIs). A SHAPES tile, see Figure 3, contains the following:

(i) A 32-bit RISC processor based on the ARM926 core with dedicated instruction and data caches, 64 kbits overall, and a computational power of 1.1 MIPS/MHz evaluated with Dhrystone 2.1 (i.e., 275 MIPS when the tile is clocked at 250 MHz, see CMOS implementation results in Section 7);

(ii) A VLIW floating-point DSP based on the 64-bit mAgicV architecture by ATMEL featuring in submicron CMOS technology a complexity of 915 kgates and roughly 2 Mbits of program and data memory;

(iii) A distributed network processor (DNP) interface for extra tile communication which is connected to the NoC through a network interface; the DNP has also resources to be interconnected with off-chip host processors; there are also low-speed peripherals (UARTS, timers, interrupt controller) plus an interface towards external RAM.

The mAgicV core [45, 46] is capable of 10 arithmetic operations (multiply, add, subtract) per cycle and operates on IEEE 754 40-bit extended precision floating-point and 32-bit integer numeric format for numerical computations. Internal memory accesses are supported by a 16-bit MAGU (multiple address generation unit) with 4 different addresses that can be generated at each cycle. Hence the VLIW DSP

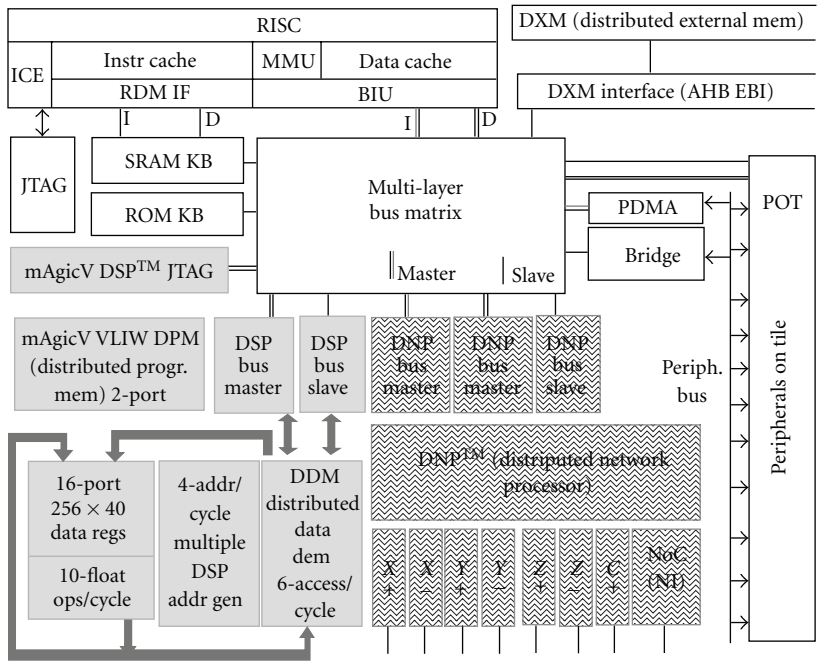

Figure 3: Content of a typical SHAPES tile [45].

is suited to process both 1D (audio) and 2D (images, video frames) data. All the above blocks are intratile interconnected through a 32-bit AXI bus. As detailed in Section 7, the 8-tile homogeneous MPSoC processor can be clocked at 250 MHz in 45 nm and 65 nm CMOS technologies. At such frequency it is capable of a computational power of 2.5 GFLOPS for each core, that is, up to 20 GFLOPS for the whole 8-tile MPSoC. The platform has a total on-chip memory of about 16 Mbits equally distributed among the tiles.

4. Heterogeneous NoC-Based MPSoC

The heterogeneous MPSoC architecture is sketched in Figure 4. It is composed of one general purpose core acting as control unit, one tile integrating external memory (DRAM, FLASH) controllers plus seven tiles each acting as an ASIP for a different class of algorithms: some units (in red) are dedicated to image/video processing such as the motion estimation (ME), some implements configurable 1D and 2D functions (in gray) such as frequency transforms or filters, one is dedicated to audio processing for low distortion digital input audio amplification (DIAA unit in blu in Figure 4). There are also seven shared SRAM memories, and in addition each computing tile has local memory resources. All tiles and memories are interconnected through an NoC which uses different routers from the architecture in Figure 2. In the heterogeneous MPSoC, new 5-port routers are used: three router ports are dedicated to *across*, *right*, and *left* router-to-router connections and two ports are dedicated to the connections between the NoC and the computing or memory tiles. The tile labeled *CPU* in Figure 4 is a 32-bit SPARC V8 core and it is in charge of instruction and data flow management. This tile has a 5-stage integer pipeline plus hardware support of MAC operations and 8 kbytes of instruction cache and 8 kbytes of data cache. The *CPU* tile has also the following low-speed local peripherals: timers, interrupt controller, memory controller for connection to external E2PROM, UART, GPIO, and JTAG interface. When compared to other SPARC V8 cores available in literature such as LEON2 and LEON3 the proposed CPU integrates also an IEEE-754 compliant floating point coprocessor, missing in the LEON2, while versus the LEON3 [47] it is characterized by a reduced complexity since a 5-stage pipeline is adopted instead of a 7-stage ones. The reason is that our CPU is part of an heterogeneous core with dedicated DSP coprocessors and hence its role will be more devoted to control tasks rather than on computing intensive applications. The computational power of the designed CPU amounts to 1 Dhrystone MIPS/MHz, that is, about 250 MIPS when the tile is clocked at 250 MHz (see CMOS implementation results in Section 7). The complexity of the *CPU* tile is 70 k logic gates. The tile named *Ext Mem Ctrl* works as an external memory controller (DDR-DRAM or ROM/EEPROM). The tile has DMA (direct frame memory access) functionalities and 1 Mbit local buffer. After describing the general purpose tiles, hereafter, the architectural block diagram of the main tiles for signal processing tasks is discussed. By exploiting a design reuse approach these tiles are implemented starting from the RTL descriptions of IP cells, we proposed as stand-alone processors in past works [28, 32, 39, 48].

Two tiles named *ME* are dedicated to motion estimation processing, and they work on 2D image blocks. Each ME application-specific tile, see Figure 5, consists of a 2D hardware search engine plus local buffer memories, a 32-bit AHB interface towards the NoC and an ME controller to

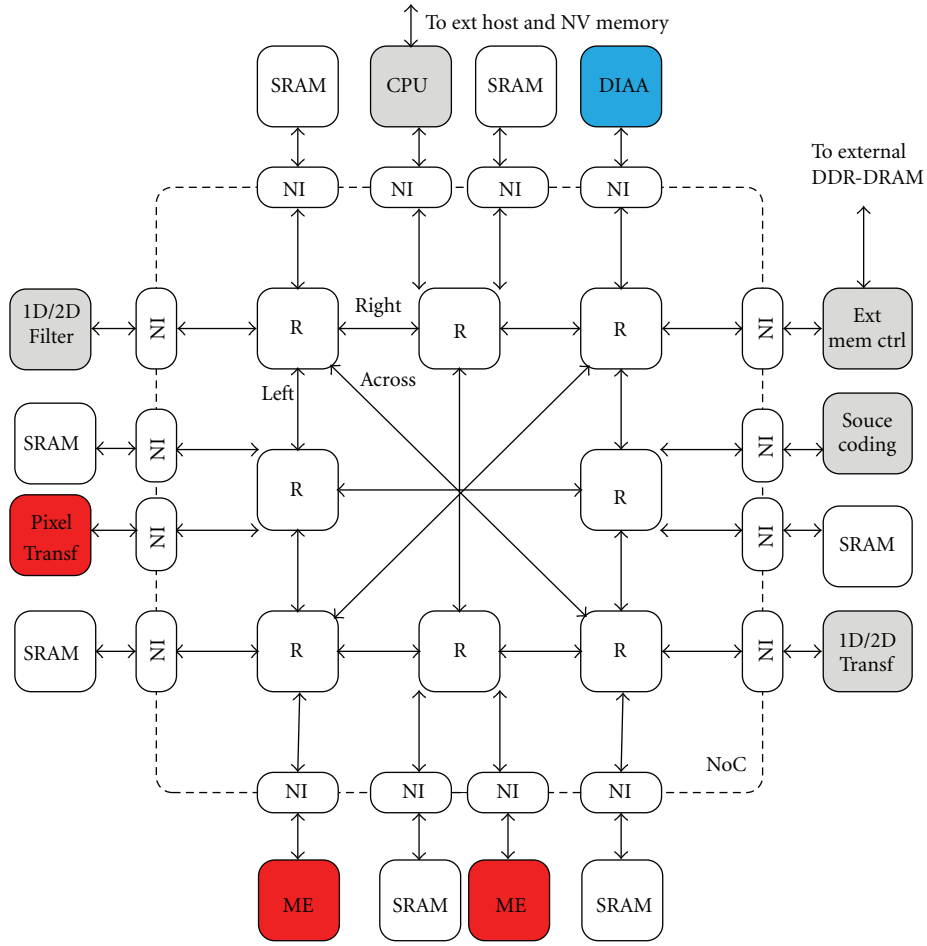

Figure 4: NoC-based Heterogeneous MPSoC: tiles for audio and image/video tasks are in gray; tiles dedicated to audio tasks are in blue, tiles dedicated to image/video tasks are in red. Memories and NoC units are in white.

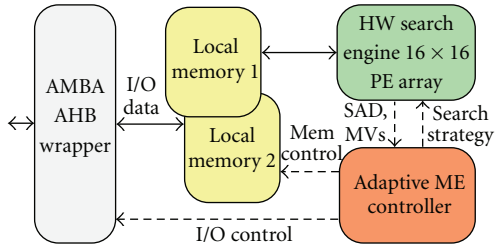

Figure 5: Architectural block diagram of the *ME* tile.

support fast search strategies. The search engine is a regular array of 256 processing elements each implementing at pixel-level absolute difference (AD) operations plus an adder tree, a unit for minimum detection calculation, and a unit used to compare block matching results with programmable thresholds (in case of fast ME search with early stop criteria). The search engine supports programmable search size and implements sum of absolute difference (SAD) and motion vector (MV) field calculation; it is derived from a parametric 2D search engine we previously proposed in [39]. In this work, the ME core has been configured for a 256 PE array capable of 256 AD operations per clock cycle, that is, 64 billions of AD operations per second (GAD/s) when the tile is clocked at 250 MHz. The use of both tiles allow for a computational capability of 128 GAD/s. The local memory resources for each ME tile have a size of 40 kbits, enough to (i) store a 16×16-pixel block and its search area with ± 16 pixel displacement in horizontal and vertical directions and (ii) ensuring the prefetch of the next image block and its search area. The logic complexity of the two ME tile amounts to 230 k logic gates.

The architecture in Figure 5 implements a Full Search, but thanks to a programmable search area and to the support of early termination criteria it also realizes fast ME algorithms, such as the predictive ME in [28, 49]. In case of fast ME algorithms, the context-aware control strategies are elaborated by the configurable ME controller reported in Figure 5, which adapts the search strategy analyzing SAD and MV results.

The 1D/2D *Transf* tile supports frequency transforms, such as DCT/IDCT and FFT/IFFT, applicable to array or matrix data structures and hence to 1D audio signals or 2D image and video frames. Twiddle coefficients are stored in ROM memories and hence the type of transform can

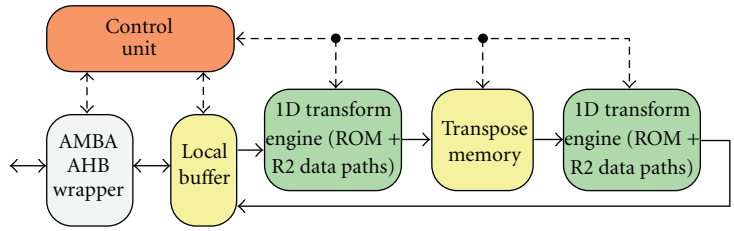

Figure 6: Architectural block diagram of the 1D/2D *Transf* tile.

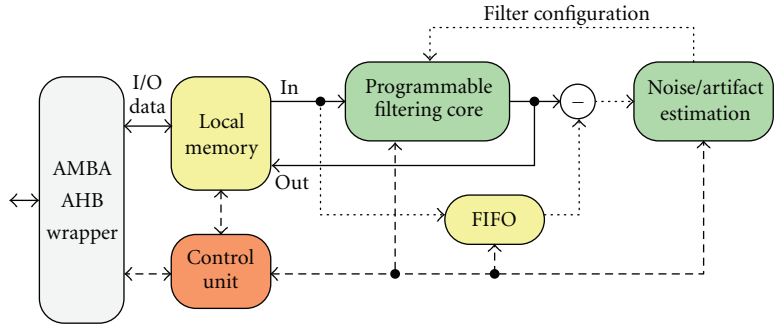

Figure 7: Architectural block diagram of the *Filt* tile.

be changed by selecting the proper coefficient memory. By exploiting the separability technique of 2D transforms, the *Transf* tile is composed of the cascade of 1D transform engines, see Figure 6. The second stage is bypassed in case of audio signal. The elementary data structure is an array of 8 samples. Image and video frames are decomposed in blocks of 8×8 pixels, elaborated row-wise in the 1st stage while the second engine works on the transposed results. Each engine is composed by 4 radix-2 (R2) butterflies, implementing MAC operations between input samples and twiddle coefficients with block floating point arithmetic. A processing capability up to 250 illions of transform operations per second is possible by clocking the tile at 250 MHz. The circuit complexity is less than 50 k logic gates. This tile also adopts a local memory buffer of 32 kbits: 16 kbits to prefetch up to 1024 samples (corresponding to sixteen 8×8 blocks image blocks), and 16 kbits to store the results for 1024 samples. The memory occupancy of each sample is 16 bits.

The 1D/2D *Filter* tile, see Figure 7, is an ASIP dedicated to filtering of 1D and 2D data structures; it supports linear (e.g., FIR, IIR) or nonlinear (e.g., rational filters) operators working at array level for audio signals and image block level for image/video frames. The filter tile has a 32-bit AHB interface towards the NoC and a throughput of 1 sample/clock cycle, that is, 250 Msample/s can be processed in real-time with a 250 MHz clock. The total local memory plus FIFO amounts to 35 kbits while the circuit complexity is around 45k logic gates. The 1D/2D *Filter* is made up of the following building blocks:

(1) a filtering core configurable to implement linear or rational nonlinear filters;

(2) a unit for noise/artifact estimation and filter tuning;

(3) memory resources for data flow management;

(4) a control unit providing all relevant control signals.

The tile named *Pixel Transf* is dedicated to special functions widely used in image and video processing: pixel transformations such as gamma correction and contrast enhancement, color domain conversion (supporting RGB, YUV, YCrCb), frame size conversion, log-linear and linear-log domain conversion, and clipping. The tile named *SourceCoding* is in charge of source coding techniques such as variable-length coding or context-adaptive binary arithmetic coding (CABAC). It works at bit stream level, it can be applied to both 1D and 2D data structures. It has a complexity of 30 k logic gates plus 30 kbits of local memory. The core of this tile is the coding engine we proposed in [29].

The DIAA tile is an ASIP implementing the audio DSP tasks needed for a high-quality reproduction of digital input audio streams (e.g., 16-bit PCM digital tracks stored in a CD or DVD or memory card) through a switching power output stage. With reference to scheme of a complete digital input power amplifier in Figure 8, the tasks supported by the DIAA ASIP are the mainly oversampling, noise shaping, multi level PWM and dead time insertion. Other supported audio function such as SPDIF interfacing or volume control or signal decimation are not shown in Figure 8. The oversampling unit implements zero padding and subsequent digital filter interpolation, needed to increase the incoming input frequency (F_{IN} ranging from 44.1 kHz to 96 kHz) by a factor M (typically $M = 8$ or 16), thus easing the reject of distortions introduced by the quantization and sampling process from the base-band audio signal (which is typically

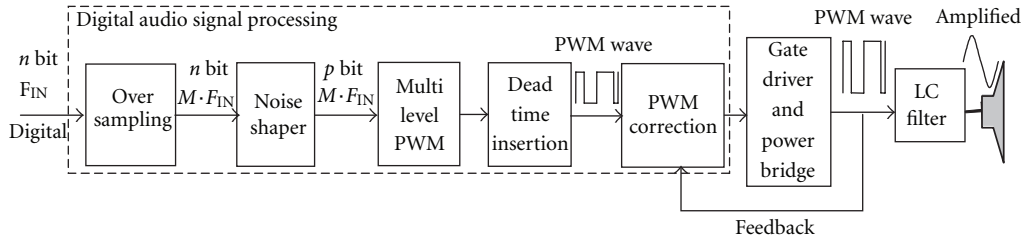

Figure 8: Architectural block diagram of a low distortion audio amplification processing chain.

below 15-16 kHz). The digital filter interpolation is implemented through a polyphase FIR filter with programmable coefficients. Converting the n-bit oversampled PCM signal to PWM leads to a minimum impulse time $T_{\min} = 1/(M \cdot F_{\text{IN}} \cdot 2^n)$ s, for example, roughly 0.25 ns considering $M = 16$ and the 44.1 kHz 16-bit PCM signal of audio CDs. Such values are too low for commercial power transistors with rise and fall times, T_r and T_f, of tens of ns. To reduce such requirement while keeping unaltered source audio quality, a noise shaper is used. It reduces the used bits from n to p, while the added quantization noise can be spread outside the audio band using a K-th order FIR shaping filter. For stability reasons, the maximum supported order is $K = 5$ while p is at maximum 6 so that $T_{\min} = 1/(M \cdot F_{\text{IN}} \cdot 2^p)$, that is, *22* ns for $M = 16$, $F_{\text{IN}} = 44.1$ kHz,compliant with rise and fall times of fast power MOS available in the market [50].

The multi level PWM and dead time insertion units convert the oversampled and noise-shaped PCM audio track to a PWM signal that can be binary-coded $(-1, +1)$ or ternary-coded $(-1, 0, +1)$. The latter technique achieves higher efficiency: when the audio signal is null there is no switching activity while in binary-coded PWM when the audio signal is null, the PWM signal is switching with a duty cycle of 50% between -1 and $+1$ symbols. Configurable guard time intervals can be digitally inserted (dead times) to compensate the turn-on and turn-off delays of real off-chip Power MOS and hence avoiding short-circuit current flowing in the power H bridge between supply and ground (e.g., if a low-side MOS already turned on while the high side MOS is still not turned off or vice versa).

The complexity of the DIAA tile amounts to roughly 30 Kgates. To be noted that the DIAA supports the DSP audio task needed for high-fidelity digital audio reproduction. Other audio DSP functionality, such as 1D filtering or FFT for noise removal, signal enhancement or equalization, insertion of polyphonic effects, among others, are supported by the other ASIP tiles, particularly the *1D/2D Filter* and *1D/2D Transf* ones.

All computing and memory tiles in Figure 4 have a 32-bit AHB interface towards the corresponding NI. The size of each memory unit is a parameter of the HDL description configurable at synthesis time. In this work, a size of 3 Mbits is considered for each of the 7 SRAM blocks in Figure 4 which is enough to store a whole VGA frame or half of a HD frame in a single memory block. In case of audio processing, a single memory can store up to 200 K samples at 16 bits, that is, 4.5 s for a single CD-quality channel at 44.1 kHz.

5. NoC Communication Infrastructure Design

5.1. NoC Architecture. The high computational power and memory access rate needed for real-time multimedia processing can be satisfied by exploiting parallel computation [51–56]. To this aim, a communication backbone allowing for parallel communication among the tiles is needed. For both the heterogeneous and homogeneous MPSoC architectures, the communication infrastructure is based on the design of a NoC (and its building macrocells, NI, and routers) with a Spidergon topology. The Spidergon approach includes the classic ring topology where each router is connected to its *Left* and *Right* neighbours, see Figures 2 and 4, but as enhancement an additional diagonal connections is introduced (each router is linked to the *Across* neighbour; see Figures 2 and 4) to minimize the number of hops to reach the destination. In the NoC in Figures 2 and 4, any packet goes across a maximum of three routers to reach its destination, since no more than two hops separate any two routers in the network. Each 4-port router in Figure 2 presents a port to the local NI, while in Figure 4 the 5-port routers are connected to two NIs: one NI for interfacing to the computing tile and one NI for interfacing to a memory block. With respect to a classic on-chip communication infrastructure based on bus hierarchy and circuit-switched connections, the NoC has two advantages: it is easier scaling the MPSoC architecture adding other computing or memory tiles; the packet-switched approach allows multiple tiles working in parallel and hence the capabilities (computational throughput and/or supported set of algorithms) for the whole homogeneous and heterogeneous MPSoCs are higher than the single tile.

The proposed NoC routers implement packet-switched wormhole routing with credit-based flow control, consequently they require small buffers and allow for a deep pipelined packet communication. Since the routing path is assigned by the NI at packet injection, the router does not need slow complex logic or look-up tables, but it simply extracts the forward information from the packet header (that has a fixed size due to the symmetry of the topology). Also the QoS (quality-of-service) information is encoded in the packet header at the injection point, and it is not explicitly linked to the path of a data flow through the network. The routers then perform a simple 2-step arbitration based on the priorities extracted from the headers, hence the QoS support in the router does not require any complex computation logic. Figure 9 illustrates the main blocks of the router, with reference to a 5-port configuration:

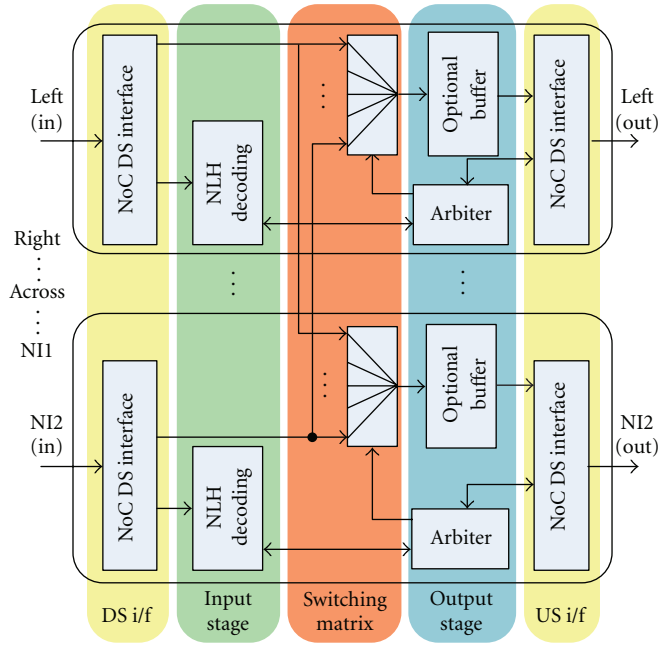

Figure 9: NoC router architecture.

the NoC DownStream (DS) interface, which is the input from the network and contains the small buffers for the incoming traffic; the Input stage, where the routing and QoS information are extracted from the packet header (network layer header, NLH in Figure 9); the switching matrix, which connects any router input port to any output port; the output stage, where multiple requests to access the same output are arbitrated, and a single input request is granted access (extra optional buffering can also be performed, if required by the application); and the NoC UpStream (US) interface, which is the output to the network. The architecture is modular, and the logic depicted schematically in Figure 9 for ports Left and NI2 is replicated on each port.

The router has a configurable number of pipeline stages in the data path, where registers can be removed and buffers are optional or by passable, thus resulting in a minimum crossing latency variable from 0 up to 2 clock cycles. Further details can be found in our recent patent [57]. The reduced crossing latency of the NI and of the router and their configurability are the main features of the proposed NoC design versus the state-of-the-art. Minimization of communication and processing latency is important at application level for interactive multimedia systems such as videoconferencing, videotelephony. The actual latency value may be increased by other factors, for example, if the packet does not win immediately an arbitration, or if the output port is temporarily stalled due to full buffers at the other side of the link. The NoC NIs in Figures 2 and 4 constitute the IP cores entry point to the communication backbone. Figure 10 shows the basic architecture of an NI. The upper part of the figure, with traffic from the IP core to the NoC, is the request path, while the lower part, from NoC to the IP bus, is the response path. In the NI, two main components can be identified: Shell and Kernel, from left to right in Figure 10. The Shell is responsible for the handshake with the IP bus and for encoding/decoding the NoC packet header (network layer and transport layer headers, NLH and TLH, respectively, in Figure 10) thus translating IP transactions into NoC packets and vice versa. The Kernel, IP-protocol independent may contain the header and payload bisynchronous FIFOs where size and frequency conversion between the IP and the NoC domains are performed. The FIFOs are also exploited in case store & forward transmission is required, where a whole packet is stored before being sent: this feature can change an intermittent transmission into a bubble-free traffic, hence optimizing the use of the network. The Kernel also contains the Upstream and Downstream interfaces, where the credit-based flow control is managed.

The NI data pipeline stages are configurable, from 0 to 3. When all input/output registers and FIFOs are removed (zero pipeline stages), size or frequency conversion are not supported, nor the store & forward mechanism. The minimum NI crossing latency equals the number of pipeline stages instantiated, but its actual value may be increased by other factors (e.g., the synchronization delay if the frequency conversion support is enabled). The proposed NI has also optional support of advanced features such as handling of out-of-order transactions, detection of error transactions, secure memory access control. However, such features are not necessary in our application, and the NoC building blocks are configured to implement just the basic functionalities: conversion of protocols, data size, and frequency. To guarantee efficient parallel communication between cores and memory spaces, the store & forward mechanism is enabled.

5.2. NoC Packet Format, Latency, Bandwidth. The NoC packet format, carrying header and payload data, is

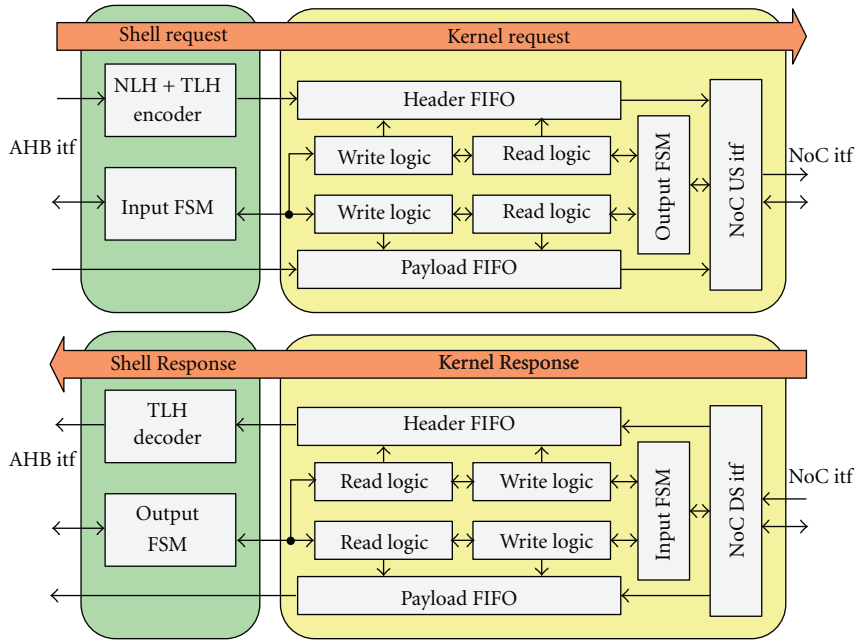

Figure 10: Main blocks in the NI architecture.

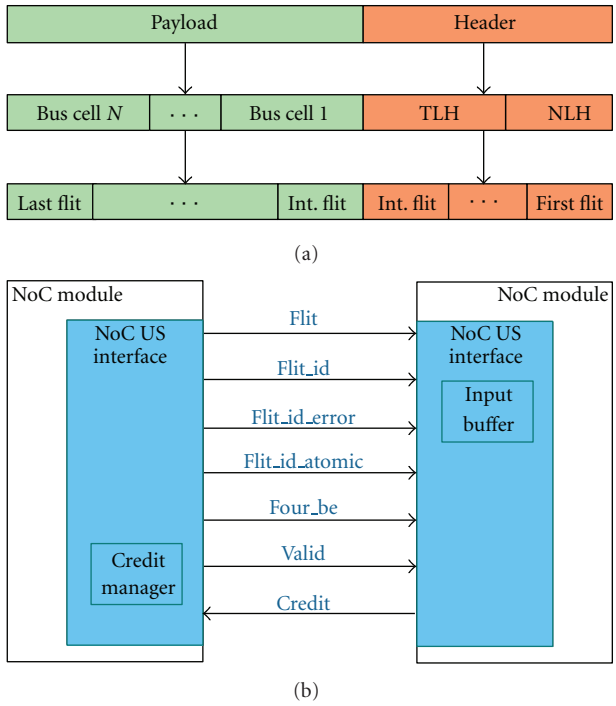

Figure 11: (a) Internal organization of a NoC packet, (b) physical link at the NoC side.

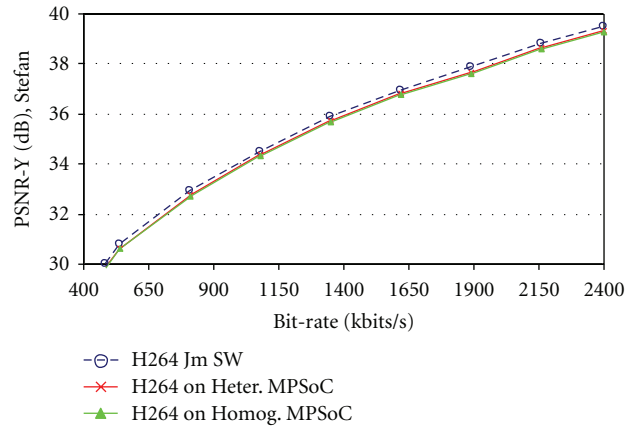

Figure 12: PSNR versus bit-rate for Stefan, a fast ME is used for homogeneous and heterogeneous MPSoC.

illustrated in Figure 11(a). The header field is composed of a network layer header (NLH) and a transport layer header (TLH). In the NLH, there are the packet routing and QoS information that are used by the NoC routers; the TLH, instead, contains IP protocol information and is necessary at the destination node, to translate the received NoC packet into a bus transaction. The payload of the NoC packet transports the payload data cells of the bus transaction. The NoC packet travels over the network in subunits called flits. Header and payload information need to travel in separate flits, and all flits of a packet are routed through the same path across the network. As far as the physical link is concerned, at the NoC interface side, there are the hardwired lines showed in Figure 12.

N-bit flits are used to transfer NoC packets, with N configurable at synthesis time. The flit_id field identifies first, intermediate, and last flits of a packet or if a flit is a single one. The optional K-bit four_be signal ($K = N/32$) marks meaningful 32-bit pieces of data within a flit and is used in end-to-end size conversion. The optional 2-bit flit_id error is used for signaling a slave side error or an interconnect error.

The optional flit_id atomic signal enables support of atomic operations: an NI can lock paths towards a Slave IP so that a Master IP can perform a generic number of consecutive operations without any interference from other masters. The credit and valid signals allows for credit-based flow control. A flit is sent only when there is room enough to receive it: neither retransmission nor flit dropping are allowed. This is done automatically by setting an initial number of credits in the U.S. interface (in its credit manager), equal to the size of the input buffer in the DS interface it communicates with. Since the U.S. interface sends flits only if the connected DS interface can accept them, there are no pending flits on the link wires.

In both homogenous and heterogeneous MPSoC designs, the AMBA AHB IP cores have a 32-bit data bus, while the NoC uses 128-bit flit size. Moreover, the different tiles might be running at different speeds because of different operating modes (see results in Section 7). For these reasons, the NIs are configured to support both size and frequency conversion, and thus they instantiate bisynchronous FIFOs in the Kernel. No extra pipeline stages are needed, as the maximum supported frequency is only 250 MHz.

The NoC routers are also configured to have a single pipeline stage (so, a minimum latency of one cycle), since it is enough to support the target NoC frequency. The global network latency can, therefore, be estimated as the sum of the routers crossing latency (i.e., a maximum of three cycles, since a maximum of three 1-cycle-latency routers are crossed to reach any destination in the network) plus the NIs latency, resulting from the size conversion operation and the synchronization delay required by frequency conversion. Any contention to access the link adds up to the global latency of the system. However, in the considered application, the network latency is critical only if the local frame memories become empty during operation. In both the heterogeneous and homogeneous MPSoC platforms, the NoC interconnect has been properly sized (128 bit NoC versus 32 bit of each connected IP) to support a data flow capable to feed the local memory resources and avoid them to go empty, thus resulting in a system which is not latency-sensitive.

The 128-bits NoC data size together with a 500 MHz NoC clock frequency (supported by the selected NoC building blocks configurations) can guarantee a nominal throughput of 128 bits $\times$ 500 MHz = 64 Gbps per link. This throughput is enough to support all target image/video and audio processing functionalities, and allows sustaining the maximum throughput theoretically generated by 8 cores working concurrently in Figures 2 or 4 through the 32 bit AMBA interface running at 250 MHz. Moreover, thanks to the NI store & forward support, the path in the network is engaged only by a bubble-free data flow, thus making full use of the available bandwidth. To exploit the high bandwidth available in the NoC interconnect it is necessary to execute long AMBA AHB transactions, since NoC packet length depends on the amount of data to be transferred. To clarify this point, suppose to execute "Store 1 byte" operations: this generates NoC packets composed of a header (whose overhead is approximately 80 bits per packet, and is transmitted in a separate flit) plus a 128-bit payload flit where only a single byte is significant, thus resulting in a real throughput of only 2 Gbps. Executing "Store 16 bytes" operations, instead, produces NoC packets composed of a header flit and a payload flit containing 128 data bits, thus achieving a 32 Gbps throughput. It is, therefore, clear that for an efficient use of the high NoC bandwidth it is important to properly select the size of the operation executed in the AMBA AHB domain. In our application, the NoC QoS mechanism is not exploited, since the NoC bandwidth capability is much higher than the AHB one and thus there is no real need to give higher priority to critical data flows. All data flow injected in the NoC by the NIs have the same bandwidth reservation (the same priority), and the arbitration performed in the routers is a simple Least Recently Used. The proposed NoC sizing is done to avoid saturation in case of the maximum bandwidth request by the target applications, that is, when running video applications like H264 encoding of one multimedia stream with 30 frames/s HD videos.

6. MPSoC Functional Assessment

The heterogeneous architecture in Figure 4 is tailored for power and area efficient real-time implementation of 1D (audio) and 2D (image/video) processing functionalities: 250 MIPS of general-purpose computing capabilities are provided by the CPU tile, while the other heterogeneous tiles provide high-throughput DSP capabilities for several algorithms: 128 billions of AD operations (GAD/s) for ME plus 250 millions of operations per second (MOPS) for each of the 5 ASIPs for filtering, transform, source coding, audio processing, pixel-level image processing. Large memory resources are integrated on-chip since multimedia applications are data-dominated. Memory resources are hierarchically organized in a first level of local memories in the computing tiles, 1.5 Mbits, plus a second level of shared frame memories (21 Mbits SRAM) accessible via NoC, and a third level of off-chip DRAMs accessible via the *Ext Mem Ctrl* tile.

The homogeneous architecture in Figure 2 offers a higher degree of flexibility since the 8 tiles provide general-purpose capabilities (for each tile, 275 MIPS of a standard ARM CPU core plus 2.5 GOPS of a DSP core) useful not only for audio or image/video processing, but also for DSP applications in general. The memory hierarchy organization of the homogeneous MPSoC is different from the heterogeneous one, since it is based on 2 levels. At Level 1, large memory resources are integrated on-chip, 16 Mbits, distributed 2 Mbits for each tile. A dedicated external DDR-DRAM can be added for each tile in Figure 2. Table 1 summarizes the computation, memory, and on-chip communication capabilities of the two MPSoCs.

In both heterogeneous and homogeneous MPSoC, thanks to the NoC paradigm, the architectures can be scaled by modifying the configuration discussed in this work in terms of: number of used computing tiles, number and size of the memories, as well as size of the tile data bus and of the NoC links. This way, the desired trade-off between complexity and performance can be set. Being composed by identical tiles, the homogeneous architecture requires less

Table 1: Performance of the MPSoC architectures.

MPSoC	Memory	DRAM interface	Computation capabilities	NoC	Scalability
Homogen.	16 Mbits L1	1 for each tile	2200 MIPS-ARM + 20 GFLOPS DSP	8 NIs + 8 4-port Routers 32 bit IP bus at 250 MHz, 128 bit flit at 500 MHz	Easy for on-chip communication and tiles
Heterogen.	22.5 Mbits (L1 + L2)	1 shared	250 MIPS-SPARC + 1.25 GOPS + 128 G AD/s	16 NIs + 8 5-port Routers 32 bit IP bus at 250 MHz, 128 bit flit at 500 MHz	Easy for on-chip communication, not for tiles

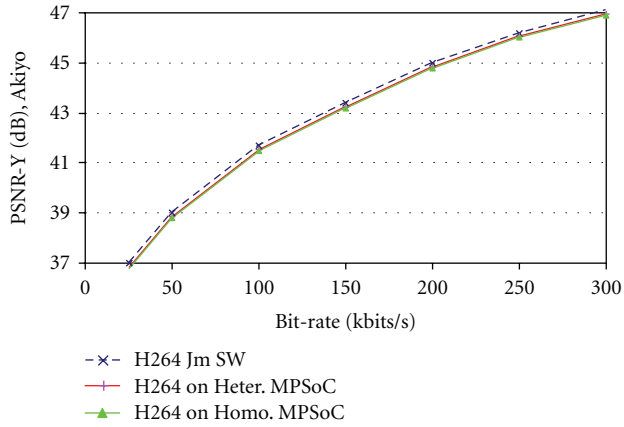

Figure 13: PSNR versus bit-rate for Akiyo, a fast ME is used for homogeneous and heterogeneous MPSoC.

design and verification time and is easier to scale. In case of the heterogeneous architecture, all tiles are different and have to be designed and tested independently before the integration and test in the whole MPSoC.

To assess the functional performances of the proposed MPSoCs, we show the results achieved when exploiting their instruction set and computing/memory capabilities for two applications (selected as case studies among the large set of algorithms that are supported). The first selected case study is the real-time compression of a video with an H.264 AVC compliant hybrid coder scheme configured as follows: fast search adaptive ME with ±16 pixel search displacement, maximum 5 previous reference frames for best block matching, SAD error cost function, CABAC entropy coding, 2D Integer DCT transform as well as, in-loop de-blocking filter. This is a very challenging computing intensive task. With reference to scenes with different degrees of dynamism, such as the sport scene Stefan at 30 Hz VGA video format, and a videoconferencing scene such as Akiyo 30 Hz CIF, a low-dynamic scene, Figures 12 and 13 present the obtained rate-distortion curves (PSNR in dB versus bit-rate). The achieved performances are compared to the ones of a reference MPEG AVC encoder, using JM software implementation, configured using CABAC and FS ME with similar parameter set as described above. From Figures 12 and 13, it is clear that both MPSoCs offer optimal performances for video processing with a PSNR degradation for a fixed bit-rate of 0.1-0.2 dB versus the reference JM software implementation. The same coding efficiency with respect to the JM reference software has been obtained with other test scenes and 30 Hz video formats including HD frames.

As proved in [11], the original JM software does not allow for real-time processing, even considering small image formats. On the contrary, the proposed MPSoCs have a computational capability enough to ensure real-time processing of 30 Hz HD videos.

Comparing in terms of maximum computational power the two platforms is not easy since the heterogeneous platform is dedicated to specific tasks while the homogenous one has more general purpose DSP capabilities. As example considering the specific H.264/AVC encoder the heterogeneous platform, offering 128 G AD/s, allows for a frame rate up to 200 frames/s while the homogenous platform is 5 times slower. Considering a more general purpose algorithm, such as the FFT, the heterogeneous platform allows for a throughput of 250 Msamples/s while the homogenous one can sustain a throughput 10 times faster.

The proposed MPSoCs can sustain in real-time lots of image, video, or audio processing algorithms. As example of audio processing both MPSoCs have been used to reproduce audio tracks from a CD-quality source. The incoming audio signal is stored in the on-chip memories, processed following the chain in Figure 8, and then sent to an off-chip output power stage accepting PWM signals and driving 8 Ω loudspeaker. Table 2 reports the achieved performance measured in terms of THD (total harmonic distortion) at 1 kHz, and 10 KHz, respectively. The results are compliant with high-fidelity audio applications.

Table 2: THD results for the reproduction of 1 kHz and 10 kHz audio signals (16-bit 44.1 KHz CD quality).

MPSoC	% THD at 1 kHz	% THD at 10 kHz
Homogeneous	0.09	0.12
Heterogeneous	0.09	0.13

7. CMOS Implementation Results

The proposed MPSoC architectures have been designed using VHDL language and then synthesized in STMicroelectronics 45 nm CMOS technology with 1.1 V supply. The RTL to gate-level synthesis has been accomplished within Synopsys CAD environment while the back-end phases have been conducted within Cadence environment. The back-end of both the homogeneous MPSoC (within the SHAPES

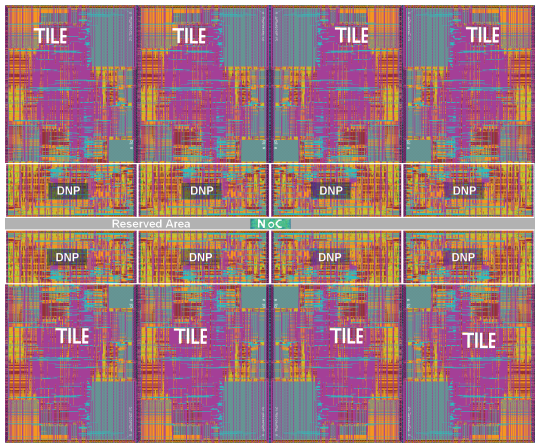

Figure 14: Layout of the SHAPES homogeneous MPSoC architecture with NoC.

project) and the heterogeneous MPSoC architectures have been successfully realized in 45 nm STMicroelectronics CMOS technology. As far as speed performance is concerned, the maximum achievable frequency depends on the used standard-cell library version. Indeed, the used technology provides 3 types of standard-cells: beside the SVT (standard voltage threshold) version there are also an HVT (high voltage threshold) version optimized in terms of leakage power consumption, and a LVT (Low voltage threshold) optimized in terms of speed but with a much higher power cost. By using a mix of HVT and SVT cells (HVT for all paths with noncritical time performances, while SVT only for time-critical paths limiting maximum. clock speed), we were able to run the NoC at 500 MHz while keeping the cost figure power consumption/(MHz·gate) near the minimum value permitted by the technology library. A faster clock, about 800 MHz, could be achieved using LVT cells; however, the power cost with LVT would be much higher and the performances achieved at 500 MHz with the HVT/SVT cells are enough for the target mobile and/or handheld applications.

To reduce the power consumption by a factor of 2 or 4 for applications not requiring the full-bandwidth capability the NoC clock frequency can be reduced at 250 MHz or 125 MHz. The clock frequency of the IP tiles can be reduced from 250 MHz to 125 MHz.

7.1. Homogeneous MPSoC Results. With reference to the scheme in Figure 2, Figure 14 reports the layout of the 8-tile NoC-based MPSoC configuration of SHAPES. The area of each tile is $2 \times 3.5\,\text{mm}^2$ and the number of logic gates is 4.46 Millions. The density of the tile chip, that is, the ratio between the area occupied by devices (logic gates, memory, I/O,...) and the total occupied silicon area after place and route, is roughly 70%. Considering the whole platform in Figures 2 and 14, the area is $8 \times 7.1\,\text{mm}^2$, that is, about $56\,\text{mm}^2$, and the number of gates is about 36 Millions. The number of transistors is roughly 150 millions. The density of the whole MPSoC is 68%. The typical value for chip density is about 75%. In the 8-tile SHAPES chip, density has been kept a little bit lower in order to minimize crosstalk effects and improve timing performance. The power consumption depends on the workload. Considering for the homogeneous MPSoC platform a target frequency of 250 MHz and a computing intensive application such as the H.264/AVC video encoder applied to the Stefan video sequence in Section 6, the total dynamic core power consumption amounts to roughly 2.8 Watts (350 mW per each tile). The leakage power consumption is 65 mW.

As far as the NoC communication infrastructure is concerned, from Figure 14, it is clear that its overhead in terms of area is minimal. The occupied area of the synthesized NoC interconnect, after place and route, is $0.123\,\text{mm}^2$ in 45 nm CMOS. The area occupancy is approximately due for 2/3 to the 8 4-port routers and for the remaining 1/3 to the 8 NIs. Power consumption analysis has been performed for the NoC at a target frequency of 250 MHz and using a 1.1 V supply voltage. The results show that the overall power consumption (dynamic power plus leakage power) is less than 4 mW, 60% due to the 8 4-port routers and 40% due to the 8 NIs. These data confirm that the overhead due to the NoC infrastructure, $0.123\,\text{mm}^2$ area and few mW of power, is negligible versus the power (2.8 W) and area ($56\,\text{mm}^2$) costs of the interconnected tiles in the MPSoC. The above NoC results refer to the following configuration: NI with data bus size and flit size of 32 bits; tiles and NoC running at 250 MHz and 500 MHz, respectively; header and payload FIFOs in the Kernel have 2 locations of 32 bits; advanced NI services (security, order handling, error management, frequency, and data size conversion,etc.) disabled.

7.2. Heterogeneous MPSoC Results. A similar characterization has been done for the heterogeneous MPSoC architecture of Figure 4 in the same CMOS technology. The computational and memory complexity of all tiles in Figure 4 amounts to 550 k logic gates plus 22.5 Mbits of on-chip SRAM, hierarchically organized in 21 Mbits level 2 frame memories (*memory tiles* in Figure 4) accessible via NoC, and around 1.5 Mbits level 1 local memories distributed in the tiles. The overall circuit complexity of the heterogeneous MPSoC is determined by both the tiles discussed in Section 4 and the resources required by the NoC infrastructure presented in Section 5. To meet the functional requirements of the proposed MPSoC each NI is configured to support 32-bit IP AHB bus, 128-bit flits on the NoC side, conversion of protocol, data size and frequency up to a maximum of 500 MHz, FIFOs sized to store 2 locations (data or headers). The complexity of this NI configuration is 14 k logic gates. As far as the router is concerned, it has 5 ports, 128-bit flits data size, an input buffer for each port of 2 locations, and supports the features described in Section 5 with a clock frequency of 500 MHz and a complexity of 32 k logic gates. The FIFOs in the NoC building blocks are not memory-based, since their small size makes more convenient a register-based implementation. The overall NoC complexity (16 NIs plus 8 5-port routers) for the heterogeneous MPSoC amounts to 480 k logic gates. Hence, the total heterogeneous MPSoC complexity, due to computing and memory tiles plus NoC infrastructure, amounts to 1030 k logic gates and

Table 3: Comparison of the proposed MPSoCs versus the state of the art (all in 45 nm technology node except Tile64 in 90 nm CNOS and AsAP2 in 65 nm CMOS).

Processor	Area	Tr. count	Power	CMOS Tech.	Computational Power	Encoder supported/format
Het. MPSoC	49 mm^2	120 M	1 W	45 nm	1.5 GOPS + 128 G AD/s	H264/AVC video codec, HD
Hom. MPSoC	56 mm^2	150 M	2.8 W	45 nm	2.2 GOPS + 20 GFLOPS (extended-precision)	H264/AVC video codec, HD
CELL-BE [20]	115 mm^2	320 M	40 W	45 nm SOI	200 GFLOPS (single-precision)	H264/AVC video codec, HD
Core2 [34]	107 mm^2	290 M	44 W	45 nm	22 GOPS at 2.4 GHz	3D DCT codec, VGA
AsAP2 [42]	40 mm^2	55 M	10.5 W	65 nm	196.8 GMAC/s + 15 G AD/s + 680 M FFT-sample/s	H264/AVC video codec, HD
ATOM 330 [58]	52 mm^2	94 M	<8 W	45 nm	3.8 GOPS at 1.6 GHz	H264/AVC video codec, CIF
ATOM 230	26 mm^2	47 M	<4 W	45 nm	7.6 GOPS at 1.6 GHz	H264/AVC video codec, QCIF
Tile64 [44]	430 mm^2	615 M	11 W	90 nm	144 GOPS at 750 MHz	H264/AVC video codec, HD

22.5 Mbits of on-chip SRAM. The number of transistors of the whole platform is about 130 millions while the area occupied by the heterogeneous MPSoC is about 49 mm^2. The power consumption depends on the workload and on the configuration status of the different tiles. For the applications described in Section 6, the estimated power consumption amounts to 1 W in case of the H.264/AVC encoding task of the Stefan sequence, the same test used for power analysis of the homogeneous MPSoC. The main contribution to the above power consumption figures is due to the computing and memory tiles.

7.3. Comparison. Table 3 summarizes the area, power consumption results, and computational capabilities of the proposed MPSoCs and some state-of-art multimedia processors. Also, the target encoder that can be supported in real-time by the platform is reported.

To be noted that computational power is expressed, depending on the target platform, as sum of operations on integer data (GOPS), on floating-point operations (GFLOPS, 32-bit single or 40-bit extended precision) or custom instructions such as multiply and accumulate (MAC) or absolute difference (AD). From Tables 1 and 3, it emerges that the heterogeneous MPSoC, made up of application specific tiles, has a smaller area and power cost versus the homogeneous MPSoC, which instead ensures a higher flexibility, being composed by general purpose tiles with DSP capabilities, and easier scalability. The heterogeneous MPSoC is particularly suited for multimedia processing on mobile and handheld devices.

Compared to state-of-the-art solutions reviewed in Section 2, the proposed MPSoCs offer the possibility of achieving real-time processing for complex multimedia tasks with a power consumption much lower than GPU. The power cost is even lower than low-power general purpose CPUs, such as the Intel ATOM 330, or application-specific multicore architecture as AsAP2 [42]. With respect to such low-power solutions the proposed MPSoCs offer a higher computational capability: for example, real-time H264/AVC encoding up to 30 Hz HD frames is supported by our MPSoCs but not by AsAP, limited to JPEG encoding of still images [41], or Intel Atom 330, limited to H264/AVC video encoding of CIF frame in surveillance applications [58]. With respect to ASICs achieving real-time and very low-power performance but for a specific algorithm, the proposed MPSoCs offer a higher flexibility and the support, even in case of the heterogeneous approach, of a large number of audio and image/video algorithmic classes.

8. Conclusions

The paper has presented the design and characterization in submicron CMOS technology of homogeneous and heterogeneous MPSoC architectures for real-time multimedia processing. In both architectures, the multiple tiles are interconnected by a network-on-chip infrastructure with Spidergon topologies and minimum crossing-latency NIs and Routers. The proposed packet-switched data transfer scheme avoids communication bottlenecks when more tiles are working concurrently. The heterogeneous architecture ensures a higher power efficiency and a smaller area occupation while the homogeneous scheme allows for higher flexibility since the basic tile offers more general purpose computation resources. With respect to the state-of-the-art, a better trade-off between cost (area, power) and performance (throughput, latency, flexibility) is achieved. The functional performances of the NoC-based MPSoC architectures are assessed with reference to H264/MPEG AVC video coding and digital input audio reproduction case studies. The heterogeneous MPSoC is particularly suited for multimedia processing in mobile and handheld terminals.

Acknowledgments

This work has been partially supported by the FP6 EU project SHAPES in collaboration with STMicroelectronics (M. Coppola, R. Locatelli, G. Maruccia, V. Catalano, AST, Grenoble), University of Cagliari (L. Raffo), ATMEL Roma (P. S. Paolucci).

References

[1] J. U. Garbas, B. Pesquet-Popescu, and A. Kaup, "Methods and tools for wavelet-based scalable multiview video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 2, pp. 113–126, 2011.

[2] G. D. Hines, Z. U. Rahman, D. J. Jobson, G. A. Woodell, and S. D. Harrah, "Real-time enhanced vision system," in *Enhanced and Synthetic Vision*, vol. 5802 of *Proceedings of SPIE*, pp. 127–134, March 2005.

[3] D. J. Jobson, Z. U. Rahman, and G. A. Woodell, "Properties and performance of a center/surround retinex," *IEEE Transactions on Image Processing*, vol. 6, no. 3, pp. 451–462, 1997.

[4] L. Shao, H. Hu, and G. De Haan, "Coding artifacts robust resolution up-conversion," in *Proceedings of the 14th IEEE International Conference on Image Processing (ICIP '07)*, pp. V409–V412, September 2007.

[5] N. E. L'insalata, S. Saponara, L. Fanucci, and P. Terreni, "Automatic synthesis of cost effective FFT/FFT cores for VLSI OFDM systems," *IEICE Transactions on Electronics*, vol. E91-C, no. 4, pp. 487–496, 2008.

[6] F. Luisier, T. Blu, and M. Unser, "Image denoising in mixed poissongaussian noise," *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 696–708, 2011.

[7] S. Marshall and G. L. Sicuranza, *Advances in Nonlinear Signal and Image Processing*, Hindawi Publishing Corporation, New York, NY, USA, 2006.

[8] S. Marsi, G. Impoco, A. Ukovich, G. Ramponi, and S. Carrato, "Using a recursive rational filter to enhance color images," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. 6, pp. 1230–1236, 2008.

[9] J. Ostermann, J. Bormans, P. List et al., "Video coding with H.264/AVC: tools, performance, and complexity," *IEEE Circuits and Systems Magazine*, vol. 4, no. 1, pp. 7–28, 2004.

[10] C. Pascual, Z. Song, P. T. Krein, D. V. Sarwate, P. Midya, and W. B. J. Roeckner, "High-fidelity PWM inverter for digital audio amplification: spectral analysis, real-time DSP implementation, and results," *IEEE Transactions on Power Electronics*, vol. 18, no. 1, pp. 473–485, 2003.

[11] S. Saponara, K. Denolf, G. Lafruit, C. Blanch, and J. Bormans, "Performance and complexity co-evaluation of the Advanced Video Coding standard for cost-effective multimedia communications," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 2, pp. 220–235, 2004.

[12] S. Saponara, P. Nuzzo, C. Nani, G. Van Der Plas, and L. Fanucci, "Architectural exploration and design of Time-interleaved SAR arrays for low-power and high speed A/D converters," *IEICE Transactions on Electronics*, vol. E92-C, no. 6, pp. 843–851, 2009.

[13] A. Yoneya, "Pulse width and position modulation for fully digital audio amplifier," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '08)*, pp. 1692–1695, May 2008.

[14] A. Chonka, W. Zhou, L. Ngo, and Y. Xiang, "Ubiquitous multi-core (UM) methodology for multimedia," in *Proceedings of the International Symposium on Computer Science and Its Applications (CSA '08)*, pp. 131–136, October 2008.

[15] K. Popovici, X. Guerin, F. Rousseau, P. S. Paolucci, and A. A. Jerraya, "Platform-based software design flow for heterogeneous MPSoC," *Transactions on Embedded Computing Systems*, vol. 7, no. 4, article 39, 2008.

[16] J. Park and S. Ha, "Performance analysis of parallel execution of H.264 encoder on the cell processor," in *Proceedings of the 5th Workshop on Embedded Systems for Real-Time Multimedia (ESTIMedia '07)*, pp. 27–32, October 2007.

[17] K. Kim, J. Lee, H. W. Park, and S. Ha, "Automatic H.264 encoder synthesis for the cell processor from a target independent specification," in *Proceedings of the IEEE/ACM/IFIP Workshop on Embedded Systems for Real-Time Multimedia (ESTIMedia '08)*, pp. 95–100, October 2008.

[18] J. Nickolls and W. J. Dally, "The GPU computing era," *IEEE Micro*, vol. 30, no. 2, pp. 56–69, 2010.

[19] D. C. Pham, T. Aipperspach, D. Boerstler et al., "Overview of the architecture, circuit design, and physical implementation of a first-generation cell processor," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, pp. 179–196, 2006.

[20] J. Pille, C. Adams, T. Christensen et al., "Implementation of the CELL broadband engine in a 65nm SOI technology featuring dual-supply SRAM arrays supporting 6GHz at 1.3V," *IEEE Journal of Solid State Circuits*, vol. 43, no. 1, pp. 163–171, 2008.

[21] X. Ma, M. Dong, L. Zhong, and Z. Deng, "Statistical power consumption analysis and modeling for GPU-based computing," in *Proceedings of the Workshop on Power Aware Computing and Systems*, Big Sky, Mont, USA, October 2009.

[22] B. G. Nam, J. Lee, K. Kim, S. Lee, and H. J. Yoo, "Cost-effective low-power graphics processing unit for handheld devices," *IEEE Communications Magazine*, vol. 46, no. 4, pp. 152–159, 2008.

[23] B. G. Nam and H. J. Yoo, "An embedded stream processor core based on logarithmic arithmetic for a low-power 3-D graphics SoC," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 5, pp. 1554–1570, 2009.

[24] C. M. Chang, S. Y. Chien, Y. M. Tsao, C. H. Sun, K. H. Lok, and Y. J. Cheng, "Energy-saving techniques for low-power graphics procrssing unit," in *Proceedings of the International SoC Design Conference (ISOCC '08)*, pp. I242–I245, November 2008.

[25] S. Y. Chien, Y. W. Huang, C. Y. Chen, H. H. Chen, and L. G. Chen, "Hardware architecture design of video compression for multimedia communication systems," *IEEE Communications Magazine*, vol. 43, no. 8, pp. 123–131, 2005.

[26] M. Murphy, K. Keutzer, and H. Wang, "Image feature extraction for mobile processors," in *Proceedings of the IEEE International Symposium on Workload Characterization (IISWC '09)*, pp. 138–147, October 2009.

[27] Quick Logic's Visual Enhancement Engine (VEE) Brings iridix to Mobile Devices, 2010.

[28] S. Saponara, M. Martina, M. Casula, L. Fanucci, and G. Masera, "Motion estimation and CABAC VLSI co-processors for real-time high-quality H.264/AVC video coding," *Microprocessors and Microsystems*, vol. 34, no. 7-8, pp. 316–328, 2010.

[29] S. Saponara, L. Fanucci, and P. Terreni, "Design of a low-power VLSI macrocell for nonlinear adaptive video noise reduction," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 12, pp. 1921–1930, 2004.

[30] NXP, UDA1355H: audio stereo codec with SPIDIF interface, 2003.

[31] K. Masselos, F. Catthoor, C. E. Goutis, and H. Deman, "A systematic methodology for the application of data transfer and storage optimizing code transformations for power consumption and execution time reduction in realizations of multimedia algorithms on programmable processors," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 10, no. 4, pp. 515–518, 2002.

[32] L. Fanucci, R. Saletti, and S. Saponara, "Parametrized and reusable VLSI macro cells for the low-power realization of 2-D discrete-cosine-transform," *Microelectronics Journal*, vol. 32, no. 12, pp. 1035–1045, 2001.

[33] R. Hameed, W. Qadeer, M. Wachs et al., "Understanding sources of inefficiency in general-purpose chips," in *Proceedings of the 37th International Symposium on Computer Architecture (ISCA '10)*, pp. 37–47, June 2010.

[34] T. Fryza, "Introduction to implementation of real time video compression method," in *Proceedings of the 15th International*

Conference on Systems, Signals and Image Processing (IWSSIP '08), pp. 217–219, June 2008.

[35] E. S. Chung, P. A. Milder, J. C. Hoe, and K. Mai, "Single-chip heterogeneous computing: does the future include custom logic, FPGAs, and GPGPUs?" in *Proceedings of the 43rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '10)*, pp. 225–236, December 2010.

[36] Apple, 2011, http://www.apple.com/iphone/specs.html.

[37] Mali Graphics Hardware, http://www.arm.com/products/multimedia/mali-graphics-hardware/index.php.

[38] O. Takahashi, C. Adams, D. Ault et al., "Migration of Cell Broadband Engine from 65nm SOI to 45nm SOI," in *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC '08)*, pp. 81–87, February 2008.

[39] L. Fanucci, S. Saponara, and L. Bertini, "A parametric VLSI architecture for video motion estimation," *Integration, the VLSI Journal*, vol. 31, no. 1, pp. 79–100, 2001.

[40] P. Pirsch, N. Demassieux, and W. Gehrke, "VLSI architectures for video compression—a survey," *Proceedings of the IEEE*, vol. 83, no. 2, pp. 220–246, 1995.

[41] Z. Yu, M. J. Meeuwsen, R. W. Apperson et al., "AsAP: an asynchronous array of simple processors," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 3, pp. 695–705, 2008.

[42] D. N. Truong, W. H. Cheng, T. Mohsenin et al., "A 167-processor computational platform in 65 nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 4, pp. 1130–1144, 2009.

[43] S. Saponara, L. Fanucci, and E. Petri, "A multi-processor NoC-based architecture forreal-time image/video enhancement," *Journal of Real-Time Image processing*. In press.

[44] S. Bell, B. Edwards, J. Amann et al., "TILE64 processor: a 64-core SoC with mesh interconnect," in *Proceedings of the IEEE International Solid State Circuits Conference (ISSCC '08)*, vol. 51, pp. 88–89, 2008.

[45] P. S. Paolucci, A. A. Jerraya, R. Leupers, L. Thiele, and P. Vicini, "SHAPES: a tiled scalable software hardware architecture platform for embedded systems," in *Proceedings of the 4th International Conference on Hardware Software Codesign and System Synthesis (CODES+ISSS '06)*, pp. 167–172, October 2006.

[46] P. S. Paolucci, "Four levels of parallelism to be managed in the DIOPSIS based SHAPES multi-tiled architecture," in *Proceedings of the 8th International Forum on Application-Specific Multi-Processor SoC (MPSOC '08)*, pp. 23–27, Aachen, Germany, 2006.

[47] Z. Zhou, W. Wu, M. He, and L. Hou, "A SoPC design based on LEON3 SoC platform," in *Proceedings of the 1st Asia Pacific Conference on Postgraduate Research in Microelectronics and Electronics (PrimeAsia '09)*, pp. 400–403, November 2009.

[48] L. Fanucci, S. Saponara, and A. Morello, "Power optimization of an 8051-compliant IP microcontroller," *IEICE Transactions on Electronics*, vol. E88-C, no. 4, pp. 597–600, 2005.

[49] A. Chimienti, C. Ferraris, and D. Pau, "A complexity-bounded motion estimation algorithm," *IEEE Transactions on Image Processing*, vol. 11, no. 4, pp. 387–392, 2002.

[50] Infineon, "Data Sheet BSO200N03S: Optimos 2 Power-Transistor. rev 1. 6," 2008.

[51] L. Benini and G. De Micheli, "Networks on chips: a new SoC paradigm," *Computer*, vol. 35, no. 1, pp. 70–78, 2002.

[52] M. Coppola, M. D. Grammatikakis, R. Locatelli, G. Maruccia, and L. Pieralisi, *Design of Cost-Efficient Interconnect Processing Units: Spidergon STNoC*, CRC Press, Boca Raton, Fla, USA, 2008.

[53] H. G. Lee, N. Chang, U. Y. Ogras, and R. Marculescu, "On-chip communication architecture exploration: a quantitative evaluation of point-to-point, bus, and network-on-chip approaches," *ACM Transactions on Design Automation of Electronic Systems*, vol. 12, no. 3, Article ID 1255460, 2007.

[54] F. Vitullo, N. E. L'Insalata, E. Petri et al., "Low-complexity link microarchitecture for mesochronous communication in networks-on-chip," *IEEE Transactions on Computers*, vol. 57, no. 9, pp. 1196–1201, 2008.

[55] S. Saponara, T. Bacchillone, E. Petri, and L. Fanucci, "Design of a NoC interface Macrocell with hardware support of advanced networking functionalities," *IEEE Transactions on Computers*. In press.

[56] M. Palesi, G. Ascia, F. Fazzino, and V. Catania, "Data encoding schemes in networks on chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 5, pp. 774–786, 2011.

[57] M. Coppola, R. Locatelli, S. Saponara, E. Petri, T. Bacchillone, and L. Fanucci, "Network on Chip Router," Patent Number 11-GR2CO-0356EP01, France, 2011.

[58] Intel, "Optimizing H. 264 software codec on Intel Atom and Intel Core2 processors, targeting Intel digital security surveillance applications," Tech. Rep. 323064, 2010.