Adv Syst Sci Appl 2018; **1**; 41-58
Published online at http://ijassa.ipu.ru.

Review of recent trends in Coarse Grain Reconfigurable Architectures for signal processing applications

Raghavachari Ramya[1], Sridharan Moorthi [2]

[1)] *VLSI Systems Research Laboratory, Department of Electrical and Electronics Engineering, National Institute of Technology, Tiruchirapalli, INDIA*

E-mail:407114003@nitt.edu

[2)] *VLSI Systems Research Laboratory, Department of Electrical and Electronics Engineering, National Institute of Technology, Tiruchirapalli, INDIA*

E-mail: srimoorthi@nitt.edu

Abstract: Coarse grained reconfigurable architecture got the attention of researchers working in designing computing architectures for processing massive streaming data associated with the multimedia applications in portable entertainment and communication electronics. The algorithms for processing audio, video, and graphics are very complex in nature. These data intensive computation algorithms belong to the domain of signal processing. As the complexity of algorithms increases, a matching improvement in speed performance of the hardware becomes essential to maintain the quality of service. The observed growth of algorithmic complexity is much higher than the growth rate of integration density governed by Moore's law. Also, the constraints on memory bandwidth in the traditional von Neumann architectures along with the slow growth in the battery capacity demands a paradigm shift in computer architecture design. Reconfigurable hardware architecture is proposed as a possible alternative in this regard. The reconfigurable architectures are designed to exploit the regular and repetitive structure of signal processing algorithms and the coarse grained processing elements are designed to match with the word level granularity of these complex algorithms. The research shows that the coarse grain reconfigurable architectures with heterogeneous processing elements are a better option for system design in DSP applications, which exploit granularity matching between the algorithms and the processing hardware, and the inherent parallelism of DSP algorithms for the realization of low power DSP systems.

Keywords: Reconfigurable architectures, CGRA, DSP, FPGA, ASIC

1. INTRODUCTION

The fastest growing segment of the electronic industry is the battery driven products of entertainment electronics and wireless communication systems. This growth is heavily indebted to the recent advancements in Digital Signal Processing (DSP). But the computationally intensive DSP applications limit the battery life in portable devices such as smart phones, MP3 players, hearing aids etc., The hardware platform chosen for the implementation of mobile wireless multimedia applications decides the speed, flexibility and cost. The major attraction of the microprocessor based system development is that its RAM based structure offers large scale flexibility such that products for various applications can be software based and in turn avoids the necessity of costly Application Specific Integrated Circuits (ASICs) for each and every application. But the sequential nature of processing by microprocessors is the major performance limiting factor. The challenging design criteria are extremely low power, high performance, flexibility and low cost. The growing performance gap among application complexity, VLSI technology, and battery technology is the constraint in delivering satisfactory performance at low power and high speed. The flexibility of a system is also an important aspect to accommodate the rapidly changing consumer needs. Limiting the applications to a specific domain is expected to provide better energy efficiency by compromising flexibility. The speed and power performance of applications pertaining to DSP domain can be further improved by suitable architecture designs which exploit the regular and repetitive structure of DSP algorithms. Reconfigurable architecture is a probable architecture suitable for the said purpose.

The idea of restructurable computing put forward by Gerald Estrin in the early sixties [1, 2].The restructurable or reconfigurable architecture is a deviation from the von Neumann architecture with adjustable hardware processing elements. The general purpose processors like microprocessors have fixed structure but Estrin's proposal has provision for adjustable hardware. The adjustable hardware is called as Reconfigurable Fabric (RF) and the process of adjusting the hardware for different applications is called reconfiguration. There are two classes of reconfigurable architectures: fine grain and coarse grain. Field Programmable Gate Array (FPGA) is a fine grain general purpose reconfigurable device that supports a broad range of applications [3, 4]. The processor architecture that uses bit oriented simple processing elements like Look Up Table (LUT) of an FPGA as the fundamental building block of the processor are called fine grained. The architecture that is composed of word oriented complex logical blocks like ALU is termed as coarse grain. The main disadvantage of FPGA like fine grain systems when compared with ASICs are low or medium performance, high configuration overhead, large silicon area, propagation delay, and high power consumption. Coarse grained reconfigurable processors have gained more popularity in the recent past, because they offer a new method for a dynamic and programmable execution similar to FPGA and tend to achieve the performance of application specific hardware. The paradigm shift to meet the computing compulsions of real-time communication and multimedia processing is from instruction stream driven systems to data driven systems and fixed hardware systems to reconfigurable hardware systems. The Coarse Grained Reconfigurable Architecture (CGRA) is a convenient platform for data streaming processing in multimedia DSP applications. These reconfigurable computing architectures have been surveyed extensively in [5-12].

The review discusses the design goals and architectural features of CGRA in section 2. Section 3 enumerates the architectural adaptations of CGRA which makes it very suitable for the domain specific applications of DSP. Section 4 focuses on the development of energy efficient coarse grain architectures for DSP applications. Finally conclusions are presented in the section 5.

2. COARSE GRAIN RECONFIGURABLE ARCHITECTURES

The coarse grain reconfigurable architectures compromise on the flexibility of FPGA to match with the performance of ASIC by limiting themselves to a particular application domain [10, 13-16]. The performance improvement over FPGA is obtained by the inherent word level configurability which is designed to match with the instruction and data granularity of algorithms results in the reduced number of cycles per instruction and hence good performance in the aspects of speed of operation. The common architecture of CGRA is a combination of a control processor and a RF which is either tightly coupled to act as a co-processor or loosely coupled to act as an augmenting independent unit for processing dedicated domain specific instructions. Generally the control processors execute the non-loop sequential code, control the mapping of configurations to the RF grid and supervise the execution activities. The basic architecture of CGRA is shown in Fig. 2.1. The RF is a combination of coarse grain processing elements (PEs) or functional units (FU), word level data paths, and fast interconnects. The presence of coarse grain processing elements reduces the configuration data and this makes the devices to do the reconfiguration faster and reduces the area, delay, and power consumption of circuits.

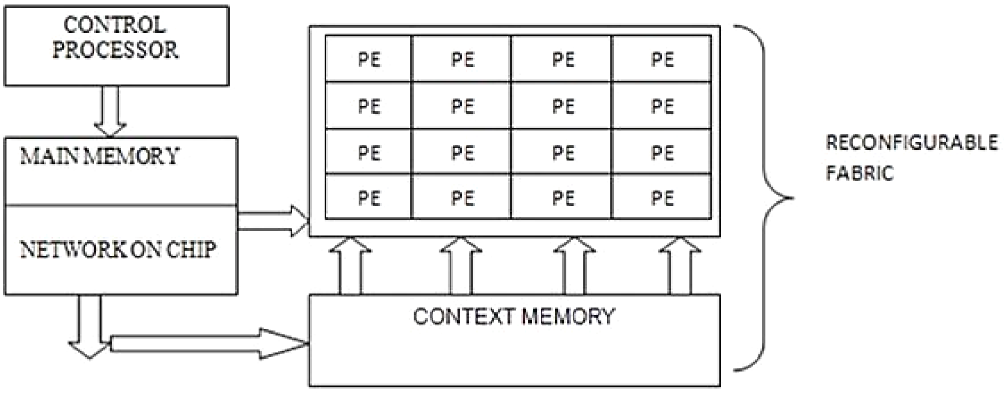

Fig. 2.1. Basic Architecture of CGRA

The data driven processing elements are configured by configuration words stored in a dedicated memory called configuration or context memory. The context words stored in the context memory decides the functionality of each processing element and also the flow of data between PEs. Reconfiguration is effected by choosing another context word in context memory. The different reconfigurable options available to the designer are: Reconfiguration can be designed to take place at the beginning of the execution and retains the context for the rest of the time (static reconfiguration) or it may occur at one or more times during the execution (dynamic reconfiguration). Reconfiguration can affect all elements in the architecture (total reconfiguration) or only some of them (partial reconfiguration). Another flexibility that the CGRA architecture designers can exercise is the option to select either a homogeneous set of PEs or a heterogeneous set of PEs for the RF. The recent approach is to give preference to have reconfigurable System on Chip which includes even General Purpose Processor (GPP), DSP, and FPGA integrated along with CGRA in to the chip [17-19]. Heterogeneous architecture for CGRA is preferred since, some algorithms run more efficiently on bit level configurable architecture and some perform optimal on word level reconfigurable platforms.

An understanding of the algorithmic domain is essential for the design of domain specific applications in order to achieve ASIC like performance. The major application of CGRA is in the domain of DSP. Table 2.1. illustrates the important coarse grain reconfigurable architectures listed in the literature.

Table 2.1. Different coarse grain reconfigurable architectures listed in literature

YEAR OF PUBLICATION	NAME OF THE PROCESSOR	RESEARCH GROUP	PROCESSING ELEMENT(PE)	PROCESSING ELEMENT CONFIGURATION	DATA PATH GRANULARITY	TARGET APPLICATION
1990	PADDI[20]	UNIVERSITY OF CALIFORNIA, BERKELEY	CLUSTER OF 8 EXU; EXU: RFILE,MUX, NANO STORE	CROSSBAR	16-BIT	DSP
1995	KRESS ARRAY[21]	UNIVERSITY OF KAISERSLAUTERN	RDPU	2D-MESH	FAMILY: SELECT PATH WIDTH	TO IMPLEMENT COMPUTATIONAL DATAPATHS
1996	RAPID [22]	UNIVERSITY OF WASHINGTON	MULTIPLIER,3 ALUS, 6 DPR, 3 LOCAL MEMORIES	1D-ARRAY	16-BIT	PIPELINING APPLICATIONS
1996	MATRIX [23]	MIT	ALU, MULTIPLIER, 256 × 8-BIT MEMORY, CONTROL LOGIC	2D-MESH	8-BIT, MULTI-GRANULAR	GENERAL PURPOSE CO-PROCESSOR

1997	PLEIADES [24]	UNIVERSITY OF CALIFORNIA, BERKELEY	SATELLITE PROCESSORS	CROSSBAR	MULTI-GRANULAR	MULTIMEDIA
1997	RAW [25]	MIT	ALU, REGISTERS, INSTRUCTION & DATA MEMORIES, CONFIG. LOGIC,PROG.SWITCH	2D-MESH	8-BIT, MULTI-GRANULAR	GENERAL PURPOSE
1998	PIPERENCH [26]	CARNEGIE MELLON UNIVERSITY	ALU & PASS REGISTER FILE	1D-ARRAY	128-BIT	CO-PROCESSOR FOR STREAMING MULTIMEDIA ACCELERATION
1998	REMARC[27]	STANFORD UNIVERSITY	16-BIT NANO PROCESSOR	2D-MESH	16-BIT	MULTIMEDIA
1998	MORPHOSYS [28]	UNIVERSITY OF CALIFORNIA, IRVINE	ALU- MULTIPLIER, A SHIFT UNIT, TWO INPUT MUX, & 32-BIT CONTEXT REGISTER	2D-MESH	8-BIT/16- BIT	MULTIMEDIA & DSP
1999	CHESS ARRAY[29]	HEWLETT PACKARD LABS	INTERLEAVED ALUs AND SWITCHBOXES	HEXOGONAL ARRAY	MULTI-GRANULAR	MULTIMEDIA
2000	DREAM[30]	UNIVERSITY OF DARMSTADT	8-BIT ALU, 2 SHIFTERS,CU,CM	2D-MESH	8-BIT/16-BIT	NEXT GENERATION WIRELESS
2001	PACT-XPP[31]	PACT INFORMATIONS TECHNOLOGIE AG	32-BIT ALU,ADDER, ROUTER,CM,CU	2D-ARRAY	16-BIT& 32-BIT	MULTIMEDIA, DSP,TELECOMMUNICATION
2002	MONTIUM [32]	UNIVERSITY OF TWENTE	FIVE ALU, TWO RFILES, SEQUENCER	TILE ARRAY	16-BIT	STREAMING DSP APPLICATION
2002	IMAGINE [33]	STANFORD UNIVERSITY	ALU CLUSTERS, MULTI PORT STREAM REGISTER FILE, SCRATCH PAD MEMORY	STREAM MODEL	16-BIT	MEDIA PROCESSING
2002	DART[34]	UNIVERSITY OF RENNES	RECONFIGURABLE DATA PATH	CLUSTER	16-BIT	MOBILE COMMUNICATION
2003	TRIPS[35]	UNIVERSITY OF TEXAS, AUSTIN	32-BIT ALU, ROUTERS,CU,OP-REG, CONFIGURATION CACHE	2D-ARRAY	32-BIT	GENERAL PURPOSE
2003	ADRES[36]	MEC , BELGIUM	32-BIT ALU, REGISTERS,ROUTER, CM,CU	ARRAY	8-BIT	MULTIMEDIA AND MOBILE COMMUNICATION
2004	NEC-DRP[37]	NEC ELECTRONICS	8-BIT ALU,8-BIT DMU,16 8-BIT RFU,8-BIT FFU	TILE	8-BIT	STREAMING MULTIMEDIA APPLICATION
2007	MORA[38]	UNIVERSITY OF CALABRIA	MULTIPLIERS, ADDERS,REGISTERS, MUXES,3:2 COMPRESSORS	LINEAR ARRAY	8-BIT	MULTIMEDIA
2008	RICA[39]	UNIVERSITY OF EDINBURGH	HETEROGENEOUS PE	ARRAY	32-BIT	DSP,VITERBI DECODING
2008	SMARTCELL[40]	WORCESTER POLYTECHNIC INSTITUTE	16-BIT ALU, I/O REGISTERS	TILE ARRAY	8-BIT	DSP, MULTIMEDIA
2009	FLORA[41]	SEOUL NATIONAL UNIVERSITY	ALU,RFILE,MUX, CONFIGURATION MEMORY	2D-ARRAY	24-BIT	DSP, MULTIMEDIA
2009	DRRA[42]	KTH ROYAL INSTITUTE OF TECHNOLOGY, SWEDEN	MORPHABLE DPU-ARITHMETIC PARTITION, LOGIC PARTITION & POST PROCESSING PARTITION	2D-ARRAY	16-BIT	DSP
2011	SYSCORE[43]	UNIVERSITY COLLEGE, DUBLIN	CONFIGURABLE FUNCTIONAL UNIT	SYSTOLIC	24-BIT	BIOMEDICAL SIGNAL PROCESSING
2013	BILRC[44]	BILKENT UNIVERSITY, ANKARA, TURKEY	ALU,MEMORY, MULTIPLIER	2D-ARRAY	16-BIT	MULTICHANNEL FIR FILTERS,VITERBI & TURBO DECODER
2014	FPCA[45]	UNIVERSITY OF CALIFORNIA	CEs, LMUs, ALUs, ON-CHIP BUFFERS, REGISTERS	2D- MESH	32-BIT	DSP,MEDICAL IMAGING, IMAGE PROCESSING
2017	HyCUBE [46]	NATIONAL UNIVERSITY OF SINGAPORE	ALU,MEMORY, SWITCHES	2D-MESH	32-BIT	GENERAL PURPOSE

ALU: Arithmetic Logic Unit; CU: Control unit; CM: Configuration manager; CE: Computation Elements; CONFIG. LOGIC- Configurable logic; DMU: Data management unit; DPU: Data path unit; EXU: Execution unit; FFU: Flip flop unit; LMU: Local Memory Unit; MUX: Multiplexer; OP-REG: Operand registers; PROG.SWITCH- Programmable switch; RFILE-Register file; RDPU: Reconfigurable Data Path Unit; RFU: Register File Unit

3. COARSE GRAIN RECONFIGURABLE ARCHITECTURES FOR DSP

The paradigm shift happened in the IC design space is that it changed from a two dimensional problem of area and speed to a three dimensional problem of speed, power and complexity. Conventional processing architectures such as GPP, DSP, ASIC and even FPGA finds it difficult to satisfy the design constraint of high speed, low power performance to meet the requirements of highly complex algorithms for the current and future applications. But the industry needs processors to fill the gap among algorithmic complexity, VLSI technology and battery technology. Coarse grain computing paradigm which got a boost in the past two decades is one specific answer to the above defined problem. The CGRA addresses these constraints by limiting the design to domain specific applications. The larger chunk of applications addressed by CGRA for high performance is from the application domain of DSP [5, 19, 47-52]. The basic concept used in the design is that hardware adapts to algorithm instead of adapting the algorithm to the hardware.

The DSP algorithms are characterized by the regular and repetitive structures present in them. These regular and repetitive computational parts of DSP algorithms that accounts for large fraction of the execution time and energy consumption are called DSP kernels. These kernels are suitable for spatially distributed computing with word level processing rather than the sequential computing at a bit oriented fashion. For example the DSP kernels of FIR filter contains a multiply-accumulate operation and the kernel of a Fast Fourier Transform (FFT) contains the FFT butterfly which are very much suitable for word oriented processing if suitable functional units are implemented for their processing at word level in a CGRA fabric.

Algorithms belonging to the same algorithmic domain have similar kernels and operate on similar data structures. Therefore the same processing elements can be reconfigured without much control overhead for execution of different algorithms. A thorough understanding of the algorithm domain is important in the design of a power efficient reconfigurable architecture. Considering the application domain of DSP, there is an excessive demand for streaming communication and computation for wireless protocol processing and multimedia processing. Also, the understanding of the underlying VLSI technology plays a vital part in the design of low power systems.

The dominant VLSI technology is the CMOS technology with the major component of energy consumption is dynamic power consumption. A first order approximation to dynamic power consumption in CMOS circuitry is given by the formula

$$\mathrm{P_d} = \propto \mathrm{C_{eff}} \mathrm{V}^2\ \mathrm{f}$$

where P_d is the power in Watts, C_{eff} is the effective switching capacitance in Farads, V is the supply voltage in Volts, α is the switching activity factor and f is the frequency of operation in Hertz. The above equation suggests that the power can be reduced by, reducing the capacitive load C_{eff}, reducing the supply voltage V, reducing the switching frequency f, or reducing the switching activity α. Since the technology already touched the 1V wall [53,54] and increase of frequency of operation also causes undesirable effects, some techniques should be exploited to reduce the switching activity and capacitance. Thus it is explored to see how the concept of locality of reference is exploited to reduce the capacitance and switching activity in order to achieve better power performance in low power systems. Locality of reference [19, 52, 55] is a major concept widely exploited to reduce the power consumption. The switching activity factor α is also reduced by the locality of reference.

Spatial locality of reference refers to the fact that once a specified location is referenced, there is often exists a chance to refer a nearby location in the near future. Accessing a large distant memory is less energy-efficient than accessing small and local memory. Energy efficiency can be substantially improved by exploiting the locality of reference principle. Due to the locality of reference principle communication within a tile dominates communication between tiles. The functional-level reconfiguration offers opportunities to improve energy efficiency of flexible architectures. Minimizing the reconfiguration data volume reduces the energy wastage. The coarse-grain paradigm of computation optimizes storage as well as computation resources from an energy point of view. Another important characteristic for data streaming applications is computing parallelism, which means the computational task has the potential to be distributed across multiple processing components. The temporal and spatial parallelism inherent in DSP algorithms can be exploited to extract better performance.

In temporal parallelism, the pipeline technology is adopted at either the instruction level or at the task level, in which the instruction code or computing task is separated into multiple stages. Several instructions or tasks are overlapped in the same pipeline at different stages to improve system throughput. On the other hand, spatial parallelism distributes the data and tasks onto different computational nodes to process in parallel. In data streaming applications, data-level parallelism (DLP) can often be exploited since there are no data dependencies between different input data blocks. In this case, Single Instruction Multiple Data (SIMD) computational style is widely used to apply the same kernel functions to different data elements.

Similarly, task-level parallelism (TLP) is usually exploited to execute different application threads in data streaming applications. In many cases, the computing task involved in stream processing can be decomposed into multiple stages. These stages can be overlapped into multiple computing resources to concurrently process different data sets through the pipeline. Given plentiful parallelism, it is a key requirement that the computing architecture design for data streaming applications should be able to efficiently exploit and map the parallelism onto available hardware resources.

As mentioned earlier, the configurability of CGRA is due the reconfigurability of the functionality of the PEs according to context words stored in the configuration memory. Recently the CGRA developers focusing on configurability of the communication network in place of the configurability of the PE. In SYSCORE [43] a conceptual interconnect structure named Round About Interconnect (RAI) is introduced. The interconnect configuration registers associated with the RAI supports distant neighbor data transfer without the cost of area and power of commonly used mesh network interconnect or a mesh variation structure. The HyCUBE [46] architecture goes even further with a reconfigurable interconnect to provide single cycle communication between distant neighbors. The reconfigurable interconnect network reduces the density of the interconnect network, hence reduces the power dissipation in the interconnect network.

4. ENERGY EFFICIENT ARCHITECTURES FOR DSP

The DSP algorithms are characterized by the regular and repetitive structures called kernels present in them. Since the algorithms belonging to the same algorithmic domain have similar kernels and operate on similar data structures, the same processing elements can be reconfigured without much control overhead for execution of different algorithms. By executing dominant kernels of a given domain of algorithms on optimized processing elements, significant energy savings with minimum of energy overhead is achieved. This section deals with important energy efficient domain specific low power CGRA processors intended for signal processing and multimedia applications.

4.1 Montium Tile Processor [19, 51, 56]

The Computer Architecture Design and Test for Embedded Systems (CADTES) group at the University of Twente designed the Montium tile processor and the processing core has been further developed by Recore Systems. This coarse grain architecture targets for 16-bit DSP applications. The Montium tile is characterized by its coarse-grained reconfigurability, high performance and low energy consumption. The Montium achieves flexibility through reconfigurability. The block diagram of a single Montium Processing Tile is shown in Fig. 4.1. The lower part contains the communication and configuration unit (CCU) and the upper part shows the reconfigurable tile processor (TP). The tile processor is the computational part which can be configured to implement a specific algorithm. It consists of five ALUs connected to 10 memory banks through a circuit switched network and these five processing parts together called the processing part array. A sequencer controls the operation of the processing part array. An array of configuration memory is provided for memory, register, ALU and interconnects. The sequencer with the help of respective decoders selects appropriate context words during execution of an algorithm. The CCU implements the interface for off-tile communication.

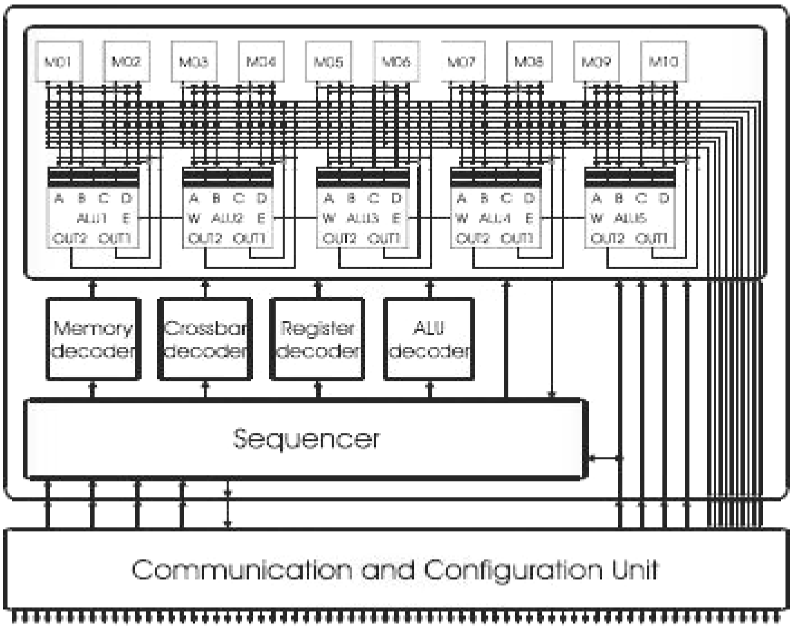

Fig. 4.1. Block Diagram of Montium Processing Tile [56]

High performance is achieved by parallelism, because the Montium has several parallel processing elements. The principle of locality of reference is exploited to reduce the energy wastage. The memory hierarchy of Montium tiles allows different levels of local register, local memory, global memory and global register. The two layer decoding employed in the Montium tile design simplifies the instruction fetch part, which reduces energy consumption and cost. Montium is designed for DSP algorithms found extensively in mobile applications. Such algorithms are usually regular and have high computational density.

It has also be noted that Thread-Level Parallelism is addressed by the multi-core approach as different tiles can run different tasks, Data-Level Parallelism (DLP) is achieved by the Montium processing tiles, which employ parallelism in the data path and Instruction-Level Parallelism (ILP) is addressed by the Montium processing tiles as multiple data path instructions can be executed concurrently. The architecture found to be highly flexible in mapping DSP kernels and energy efficient too.

The main characteristics required for a CGRA to act as a DSP core are low power, high speed and flexibility. The architecture is found to be highly flexible in mapping DSP kernels. The energy performance characteristic of the MONTIUM is compared with the characteristics of other state-of-the-art reconfigurable architectures in [56]. The FFT algorithm is used to compare the energy consumption of a MONTIUM tile processor with an ASIC, a Xilinx Virtex-II Pro FPGA, a Silicon Hive Avispa and an ARM920T processor. The results presented in the paper shows that Montium processor provide an alternative for mobile devices for which energy-efficiency is an important factor. The Montium tile processor satisfies the requirements mentioned above for a DSP core and can be used as a single DSP accelerator core or clustered in a large reconfigurable subsystem.

4.2 Dynamic Reconfigurable Resource Array (DRRA) [57, 58]

DRRA is a heterogeneous coarse grain reconfigurable architecture capable of hosting multiple, complete Radio and multimedia applications. The design template was developed at KTH Royal institute of Technology, Sweden. A fragment of DRRA fabric which consists of DRRA cells arranged in rows and columns fashion is illustrated in Fig. 4.2. The architecture relies on high bandwidth, distributed memory integrated in 3D within the logic tile. Any signal processing applications such as FIR, IIR, FFT etc., can be mapped efficiently in the DRRA fabric. The DRRA fabric is composed of morphable Data Path Units (mDPU) and Register Files (RFile) organized in a 4×N matrix. Bottom and top layers are of RFiles and the inner layers are of mDPUs. A sequencer, switch box, mDPU and RFile form the basic DRRA cell.

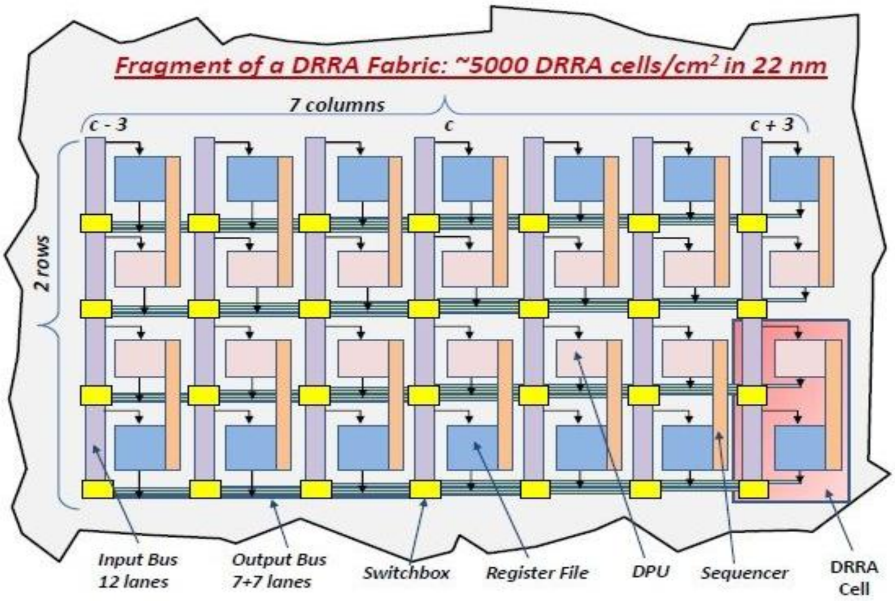

Fig. 4.2. Dynamically Reconfigurable Resource Array Fabric [57]

Reconfigurability of mDPUs and the interconnect that combine the multiple mDPUs and RFiles is central to creating algorithmic building blocks. DRRA cells are connected together through interconnects using a sliding window 3-hop communication scheme. Every resource input is connected to every resource output within three-column range, making the boundaries of the architecture flexible enough to implement most of the algorithmic building blocks. Extended connectivity beyond three columns can be achieved by using intermediate buffers and this seamless connectivity offers flexibility in creating on the fly partition to create new algorithmic building blocks as and when required. The DRRA architecture is characterized by the large pool of resources for computational, storage, and interconnects functionalities. The architecture relies on clustering these resources at run time to serve individual applications and once an application is over, it is reclaimed and reused for next application.

DRRA being a fabric, the computation is distributed across the chip. Multiple threads, algorithms and applications are intended to run in parallel in the fabric. The Distributed Memory Architecture (DiMArch) found in DRRA is partitionable and the architecture is designed to keep the cost of partitioning and re-partitioning low both in terms of cycles and energy. The distributed nature of memory architecture and the concept of private execution environments enable a short distance between storage and computation, which in turn satisfies the locality of reference.

The DRRA reduces the granularity mismatch at three levels. At the building block level, the DPU and the register file have been endowed with custom modes like MACs, butterflies, programmable burst address generation, bit-reverse addressing modes etc., to efficiently implement commonly occurring DSP operations in the signal processing algorithms. This coarse grain mode reduces the silicon and bit-width mismatches. The DRRA cell is the basic unit when composing a DRRA fabric, but its components - DPU, register file and sequencer - can be combined individually to create hierarchical FSMDs (FSMs + Data-paths) of arbitrary complexity; the combining happens in the DRRA interconnect fabric. This is the third level at which DRRA reduces the instruction granularity mismatch. DRRA’s interconnect scheme enables dynamic creation of an arbitrary wide VLIW, where each issue can be an arbitrary instruction.

The power performance of DRRA is computed and compared with ASIC, FPGA in detail and presented in [57]. The performance is tested by simulating FIR, FFT and Sorting algorithms. The static and dynamic power analysis shows that the DRRA achieves performance comparable with that of an ASIC processor. For comparison, the dynamic power dissipated in the clock net is on an average over all the algorithms is 79.47X that of ASIC, but for an FPGA it is 255.57X. Similarly the dynamic power spend for computation is 2.76X compared to 60.87X for FPGA compared to that of an ASIC. The DRRA consumes 24.64X of the static power consumption of ASIC and FPGA consumes 403.19X that of an ASIC. The performance of DRRA is evaluated further by implementing a 1024 FFT. For a 1024-point FFT, in terms of FFT operations per unit energy, DRRA-1 and DRRA-2 outperforms all CGRAs by at least 2X and is worse than ASIC by 3.45X. The number of operations per second achievable by DRRA equals that of the ASIC. Also considering the good flexibility of DRRA, it possesses all the essential features of a DSP core for mobile platforms.

4.3 SmartCell [40, 59,60]

SmartCell is a coarse grain reconfigurable architecture designed for stream based applications. The block diagram of SmartCell architecture is shown in Fig.4.3. The architecture is made up of three main components: cell unit, reconfigurable interconnect fabric and data I/O.

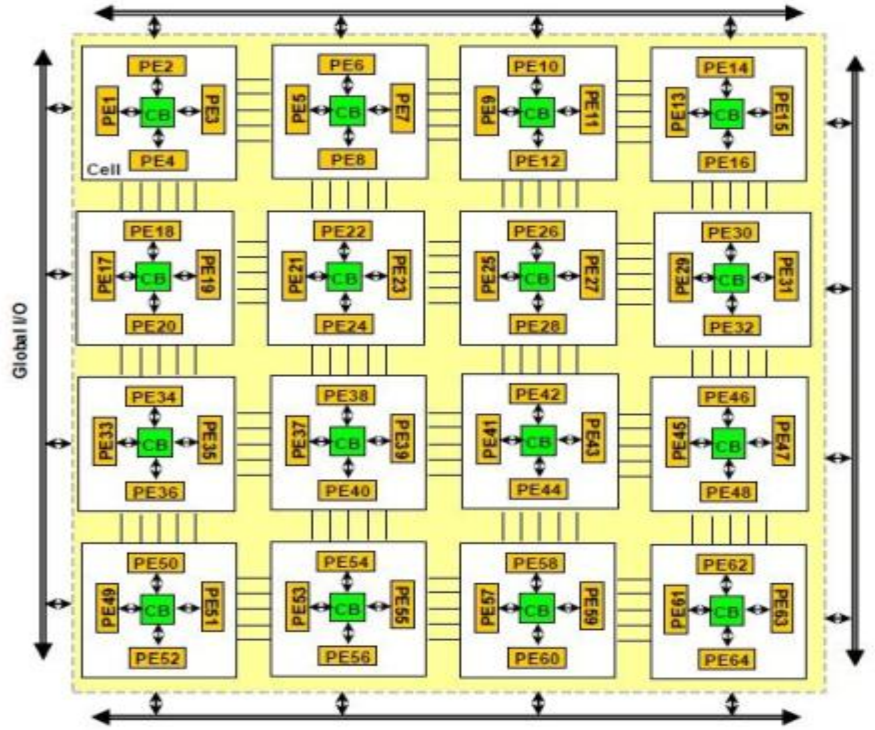

Fig.4.3. Block Diagram of SmartCell Architecture [60]

The reconfigurable cell units are the basic components in SmartCell. They are organized in a tiled structure. There are four identical PEs in each cell block. Multiple PEs can be chained together to implement more complex algorithms. A 4-stage pipeline structure is developed in each processor. The reconfigurable interconnect fabric are used for inter and intra cell communication. The data flow can be dynamically reconfigured for different applications. The deep pipeline and ILP at single PE and the TLP among multiple cells arranged in a tiled mesh are exploited to realize high computing capacity and energy efficiency. An instruction controller is provided to control the data-path and functionality of each PE. The number of PEs involved in an application task, function of each PE and the inner and inter cell connectivity are reconfigurable in real time. Also, the dynamic power consumption is significantly reduced by the use of gated clock technology to turn off the inactive PEs. The features of the smart cell [59] are listed below:

(i) Coarse-grained granularity: SmartCell is designed to generate coarse-grained configurable system targeted for computation intensive applications. The processing elements operate on 16-bit input signals and generate a 36-bit output signal, which avoids high overhead and ensures better performance compared with fine-grained architectures.

(ii) Flexibility: Due to the rich communication resources, versatile computing styles can be easily mapped onto the Smart Cell architecture, including SIMD, MIMD, and 1D or 2D systolic array structures. This also expands the range of applications to be implemented.

(iii) Dynamic reconfiguration: By loading new instruction codes into the configuration memory through the SPI structure, new operations can be executed on the desired PEs without any interruption with others. The number of PEs involved in the application is also adjustable for different system requirements.

(iv) Fault tolerance: In the Smart Cell system, defective cells, caused by manufacturing fault or malfunctioned circuits, can be easily turned off and isolated from the functional ones.

(v) Deep pipeline and parallelism: Two levels of pipeline are achieved—the Instruction Level pipeline in a single processor element and the task level pipeline among multiple cells. The data parallelism can also be explored to concurrently execute multiple data streams, which in combine ensures a high computing capacity.

(vi) Hardware Virtualization: Distributed context memories are used to store the configuration signals for each PE. The cycle-by-cycle instruction execution supports hardware virtualization that is able to map large applications onto limited computing resources.
(vii) SmartCell provides explicit synchronization that eases the exploration of computing parallelisms.
(viii) Unique system topology: The cell units are tiled in a 2D mesh structure with four PEs inside each cell. This topology was designed to meet different computational requirements. With the help of the hierarchical on-chip connections, the SmartCell architecture can be dynamically reconfigured to perform in variant operational styles.

It is experimentally shown that the SmartCell offers significant improvement in energy and speed performance [60] compared to FPGA, DSP and MorphoSys CGRA. The Smart Cell is found to be 1.5 times faster compared to Xilinx's Virtex II Pro XC2VP20 FPGA in executing an FFT block. It is found to be 3.6 times energy efficient than the FPGA. A comparison with DSP for the same experimental setup shows that SmartCell is about 20.8 times faster and about 28.9 time more energy efficient than TMS320C6713. The SmartCell satisfies the basic requirement of low power, high speed, and flexibility required for portable DSP processors and hence a good substitute as a DSP core.

4.4 Fully Pipelined Composable Architecture (FPCA) [45]

The architecture of FPCA is also that of a 2D-mesh with neighbor-to-neighbor(N2N) connectivity. Each tile is a cluster of PEs. The architecture of the PE cluster is shown in Fig.4.4. The cluster is a set of heterogeneous PEs including computation elements(CEs), Local Memory Units(LMUs) and register chain to act as onchip buffers and registers respectively. The architecture may be considered as a two level architecture where processing elements are first connected by a permutation matrix with high connectivity within a cluster and then by a global N2N network for more scalable connectivity.

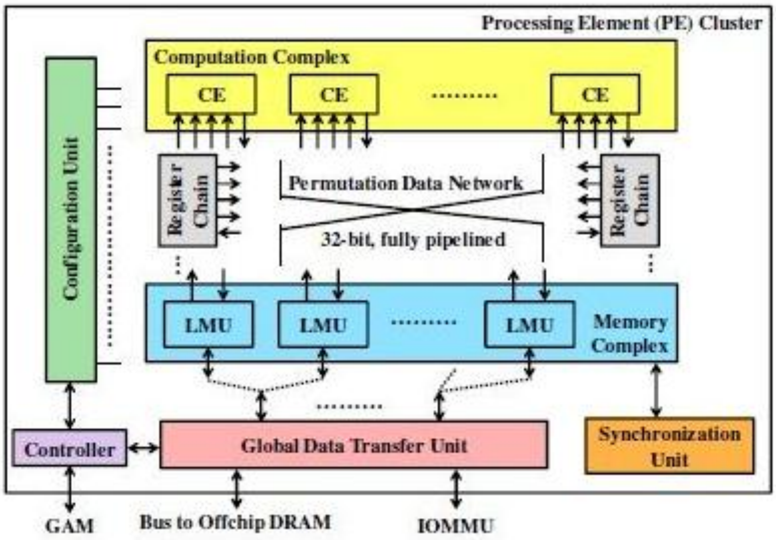

Fig.4.4. Internal structure of a processing element (PE) cluster [45]

Mostly, a single application fit into a single processing element cluster and this alleviates the challenges of routability and dynamic composition. A small part of the resources is used for a single application and hence the unused resources can shared for other applications. Multiple copies of a single application can also be mapped to the unused resources. This process is called dynamic composition which will improve the overall throughput. A runtime scheduler is used to map the idle resources as mentioned to the incoming applications. The challenge of complex routing in dynamic composition is overcome by the programmable interconnects provided with the FPCA architecture. The processing element cluster approach introduces heterogeneity which support memory, arithmetic and logic operations altogether in a tile. This enables one to offload execution control to the PE cluster and hence aggressive pipelining made possible based on a modulo scheduling scheme. The full pipelining and dynamic composition of the architecture improve the energy and speed performance of the device.

It is observed that FPCA prototype achieve 1.5-3.4X speedup compared to the Dual-Core ARM. Also shown that a 2X improvement in speed performance is achieved by duplication based on dynamic composition. Experiments show that FPCA architecture can achieve a >50X energy savings compared to ARM processor.

4.5 HyCUBE [46]

HyCUBE architecture follows the 2D-mesh topology as shown in Fig.4.5. The functional units (FUs) are connected in a 2D-mesh structure. The architecture comprises of two types of tiles. Tiles of the first type are connected to the data memory and capable of performing memory operations. Tiles of the second type compute only tiles and both have an ALU, a configuration memory and cross bar switch. An additional load-store unit is present in the memory tile enable them to interface with data memory.

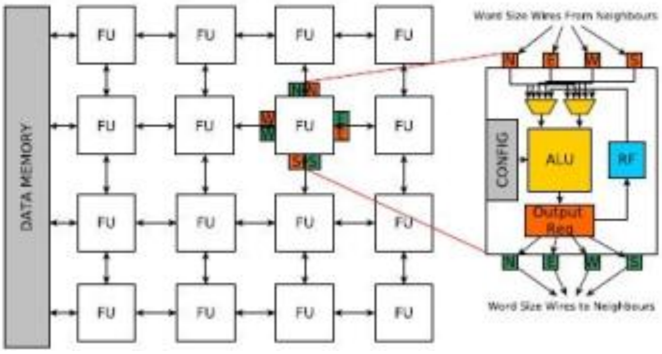
DATA MEMORY
FU
Word Size Wires From Neighbours
N
E
W
S
CONFIG
ALU
RF
Output Reg
Word Size Wires to Neighbours

Fig.4.5. A 4 × 4 HyCUBE CGRA [46]

By introducing a reconfigurable interconnect HyCUBE provides 1-cycle communication between distant tiles. The main circuit element in the programmable interconnect is the crossbar switch that is driven by a clockless repeater. The repeaters can be configured to bypass data to successive hops asynchronously or to receive data. Since data can be passed to distant tiles by bypassing the entry of data into the in between tiles, 1-cycle communication can be realized between tiles that are not nearest neighbors. The HyCUBE has a compiler controlled NOC and hence there is no routing and flow control logic. The input and output ports has only single registers and facilitates multihop multicast path between tiles scheduled completely at compile time.

Given the reconfigurable single-cycle multi-hop interconnect, HyCUBE scales very well for 4×4 CGRA and offers 3X performance and 4X performance-per-watt compared to ARM. The experimental results also show that average power efficiency of HyCUBE is 1.5X and 3X compared to a CGRA with standard NoC and a N2N CGRA, respectively. The experiments gives a demonstration of the improvements made with regard to the interconnect of the HyCUBE enables it to improve the energy performance.

Unlike the other processors discussed HyCube is not a dedicated DSP CGRA architecture. But the idea of reconfigurable interconnect and reduction of total hardware cost due to interconnects are also seems to be welcome features for the design of future CGRAs for improving power performance as demonstrated by the HyCUBE architecture.

5. CONCLUSIONS

In the post Dennard scaling era, the supply voltage scaling is limited by exponential increase in leakage current and it touched the 1V wall. The computer architecture designers are thus exploring alternatives to circumvent the limitations imposed by the VLSI technology. The speed and power performance can be improved by exploiting the inherent word level parallelism of DSP algorithms to design suitable parallel computer architectures matching with the structure and granularity of the algorithms. Another concept well exploited in the design of low power DSP system is the judicious use of locality of reference by providing distributed memory. The flexibility of CGRA were restricted by limiting the applications to a specific domain and suitable control mechanism are provided to turn off unutilized hardware in run-time to get better energy efficiency. Also heterogeneous architectures ensures better granularity matching which can be exploited for realizing low operational frequencies which is directly proportional to the power dissipated. The study reveals that coarse grained reconfigurable heterogeneous architecture with distributed memory is the future solution for the electronic industry to assure speed, power and cost performance. Recent research shows that reconfigurable interconnect networks reduces the density of the interconnect network, and hence reduces the power dissipation in the interconnect network. A clever design choice of reconfigurable processing elements and interconnects may result in delivering ASIC like energy and speed performance by CGRAs especially in the application domain of signal processing.

REFERENCES

[1] Estrin, G. (1960). Organization of computer systems: The fixed-plus-variable structure computer. In *Papers presented at the May 3-5, 1960, western joint IRE-AIEE-ACM computer conference,* New York, USA, 33-40, https://doi.org/10.1145/1460361.1460365

[2] Estrin, G., Bussell, B., Turn, R., & Bibb, J. (1963). Parallel processing in a restructurable computer system. *IEEE Transactions on Electronic Computers*, EC-12(6), 747-755, https://doi.org/10.1109/PGEC.1963.263558

[3] Trimberger, S. M. (2015). Three ages of FPGAs: A retrospective on the first thirty years of FPGA technology. In *Proceedings of the IEEE,* 103(3), 318-331, https://doi.org/10.1109/JPROC.2015.2392104

[4] Tatas, K., Siozios, K., & Soudris, D. (2007). A survey of existing fine-grain reconfigurable architectures and CAD tools. In S. Vassiliades and D. Soudris (Eds), *Fine-and Coarse-Grain Reconfigurable Computing* (pp. 3-87). Springer Netherlands.

[5] Hartenstein, R. (2001). A decade of reconfigurable computing: a visionary retrospective. In *Proceedings of Design, automation and test in Europe Conference and Exhibition 2001*, Munich, Germany, 642-649, https://doi.org/10.1109/DATE.2001.915091

[6] Krishnamurthy, R. B. (2001). A survey of next generation reconfigurable architectures for embedded computing, *Tech.Rep.*, College of Computing, Georgia Institute of Technology.

[7] Compton, K., & Hauck, S. (2002). Reconfigurable computing: a survey of systems and software. *ACM Computing Surveys (CSUR),* 34(2), 171-210, https://doi.org/ 10.1145/508352.508353

[8] George, T., Soudris, D., & Vassiliadis, S. (2007). A survey of coarse-grain reconfigurable architectures and cad tools. In S. Vassiliades and D. Soudris (Eds), *Fine-and Coarse-Grain Reconfigurable Computing* (pp.89-149). Springer Netherlands.

[9] Hauck, S., & DeHon, A. (2010). *Reconfigurable computing: the theory and practice of FPGA-based computation*, Morgan Kaufmann, Amsterdam.

[10] De Sutter, B., Raghavan, P., & Lambrechts, A. (2013). Coarse-grained reconfigurable array architectures. In *Handbook of signal processing systems* (pp. 553-592). Springer New York.

[11] Lyke, J. C., Christodoulou, C. G., Vera, G. A., & Edwards, A. H. (2015). An introduction to reconfigurable systems. In *Proceedings of the IEEE*, 103(3), 291-317, https://doi.org/ 10.1109/JPROC.2015.2397832

[12] Chattopadhyay, A. (2013). Ingredients of Adaptability: A survey of reconfigurable processors. *VLSI Design*, 2013,1-18, http://dx.doi.org/10.1155/2013/683615

[13] Eguro, K., & Hauck, S. (2003). Issues and approaches to coarse-grain reconfigurable architecture development. In *Proceedings of the 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines,(FCCM 2003),* NAPA, CA, USA, 111-120, https://doi.org/10.1109/FPGA.2003.1227247

[14] Eguro, K., & Hauck, S. A. (2005). Resource allocation for coarse-grain FPGA development. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(10), 1572-1581, https://doi.org/10.1109/TCAD.2005.852291

[15] Verbauwhede, I., Schaumont, P., Piguet, C., & Kienhuis, B. (2004). Architectures and Design techniques for energy efficient embedded DSP and multimedia processing. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, 2, 988-993, https://doi.org/10.1109/DATE.2004.1269022

[16] Carroll, A., Friedman, S., Van Essen, B., Wood, A., Ylvisaker, B., Ebeling, C., & Hauck, S. (2007). Designing a coarse-grained reconfigurable architecture for power efficiency. In *Department of Energy NA-22 University Information Technical Interchange Review Meeting.*

[17] Smit, G. J., Kokkeler, A. B., Wolkotte, P. T., Hölzenspies, P. K., van de Burgwal, M. D., & Heysters, P. M. (2007). The chameleon architecture for streaming DSP applications. *EURASIP Journal on Embedded Systems*, 2007:078082 https://doi.org/10.1155/2007/78082

[18] Park, Y., Park, J. J. K., & Mahlke, S. (2012). Efficient performance scaling of future CGRAs for mobile applications. In *2012 International Conference on Field-Programmable Technology(FPT'12),* Seoul, South Korea, 335-342, https://doi.org/10.1109/FPT.2012.6412158

[19] Smit, G. J., Kokkeler, A. B., Wolkotte, P. T., Van De Burgwal, M. D., & Heysters, P. M. (2006). Efficient Architectures for Streaming DSP Applications. *Dynamically Reconfigurable Architectures, Internationales Begegnungs-und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany*. [Online]. Available http://drops.dagstuhl.de/opus/volltexte/2006/743

[20] Chen, D., & Rabaey, J. (1990). PADDI: Programmable arithmetic devices for digital signal processing. In *VLSI Signal Processing IV*, 240-249.

[21] Hartenstein, R. W., & Kress, R. (1995). A datapath synthesis system for the reconfigurable datapath architecture. In *Proceedings of Asia and South Pacific Design Automation Conference 1995,* Makuhari, Chiba, Japan, 479-484, https://doi.org/10.1109/ASPDAC.1995.486359

[22] Ebeling C., Cronquist D.C. & Franklin P. (1996) RaPiD- Reconfigurable pipelined datapth. In: R.W. Hartenstein, M. Glesner (Eds) *Field –Programmable Logic Smart Applications, New Paradigms and Compilers. FPL 1996. Lecture Notes in Computer Science*, (pp. 126-135). Springer, Berlin, Heidelberg, https://doi.org/10.10073/3-540-61730-2_13

[23] Mirsky, E. & DeHon, A. (1996) MATRIX: A reconfigurable computing architecture with configurable instruction distribution and deployable resources. In *Proceedings of IEEE Symposium on FPGAs for custom computing machines (FCCM),* Napa Valley, CA, USA, 157-166, https://doi.org/ 10.1109/FPGA.1996.564808

[24] Rabaey, J. M. (1997). Reconfigurable processing: the solution to low-power programmable DSP. In *1997 IEEE International Conference On Acoustics, Speech, And Signal Processing,* Munich, Germany, 1, 275-278, https://doi.org/10.1109/ICASSP.1997.599622

[25] Waingold, E., Taylor, M., Srikrishna, D., Sarkar, V., Lee, W., Lee, V., & Babb, J. (1997). Baring it all to software: Raw machines. *Computer*, 30(9), 86-93, https://doi.org/10.1109/2.612254

[26] Copen, S., Herman, G., Matthew, S., Mihai, M., Cadambi, B. S., Reed, R., & Laufer,T.R(1999). PipeRench: A coprocessor for streaming multimedia acceleration. In *Proceedings of the 26th International Symposium on Computer Architecture,* Atlanta, GA, USA, 28–39, https://doi.org/10.1109/ISCA 1999.765934

[27] Miyamori, T., & Olukotun, K. (1999). REMARC: Reconfigurable multimedia array coprocessor. *IEICE Transactions on information and systems*, 82(2), 389-397.

[28] Singh, H., Lee, M. H., Lu, G., Kurdahi, F. J., Bagherzadeh, N., Lang, T., & Eliseu Filho, M. C. (1998). MorphoSys: An integrated re-configurable architecture. In *Proceedings of the NATO RTO Symp. on System Concepts and Integration, Monterey, CA, USA* , 1-11.

[29] Marshall, A., Stansfield, T., Kostarnov, I., Vuillemin, J., & Hutchings, B. (1999). A reconfigurable arithmetic array for multimedia applications. In *Proceedings of the 1999 ACM/SIGDA seventh international symposium on Field programmable gate arrays* , Monterey, California, USA, 135-143, https://doi.org/10.1145/296399.296444

[30] Alsolaim, A., Becker, J., Glesner, M., & Starzyk, J. (2000). Architecture and application of a dynamically reconfigurable hardware array for future mobile communication systems. In *Proceedings of 2000 IEEE Symposium on Field-Programmable Custom Computing Machines*, Napa Valley, CA, USA, 205-214, https://doi.org/10.1109/FPGA.2000.903407

[31] Baumgarte, V., Ehlers, G., May, F., Nückel, A., Vorbach, M., & Weinhardt, M. (2003). PACT XPP-A self-reconfigurable data processing architecture. *The Journal of Supercomputing*, 26(2), 167-184, https://doi.org/10.1023/A:1024499601471

[32] Heysters, P., Smit, G., & Molenkamp, E. (2003). A flexible and energy-efficient coarse-grained reconfigurable architecture for mobile systems. *The Journal of supercomputing*, 26(3), 283-308, https://doi.org/10.1023/1025699015398

[33] Kapasi, U. J., Dally, W. J., Rixner, S., Owens, J. D., & Khailany, B. (2002). The Imagine stream processor. In *Proceedings of the 2002 IEEE International Conference on Computer Design: VLSI in Computers and Processors, Freiberg, Germany,* 282-288, https://doi.org/10.1109/ICCD.2002.1106783

[34] David, R., Chillet, D., Pillement, S., & Sentieys, O. (2002). DART: A dynamically reconfigurable architecture dealing with future mobile telecommunications constraints. In *Proceedings of the 16th International Parallel and Distributed Processing Symposium (IPDPS 2002),* Fort Lauderdale, FL,USA, https://doi.org/ 10.1109/IPDPS.2002.1016554

[35] Sankaralingam, K., Nagarajan, R., McDonald, R., Desikan, R., Drolia, S., Govindan, M. S., ... & Liu, H. (2006). Distributed microarchitectural protocols in the TRIPS prototype processor. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture,* ORLANDO,FL,USA, 480-491, https://doi.org/10.1109/MICRO.2006.19

[36] Mei B., Vernalde S., Verkest D., De Man H., Lauwereins R. (2003) ADRES: An Architecture with Tightly Coupled VLIW Processor and Coarse-Grained Reconfigurable Matrix. In: Y. K. Cheung P., Constantinides G.A. (eds) *Field Programmable Logic and Application. FPL 2003. Lecture Notes in Computer Science*, (pp. 61-70). Springer, Berlin, Heidelberg, https://doi.org/10.1007/978-3-540-45234-8_7

[37] Suzuki, M., Hasegawa, Y., Yamada, Y., Kaneko, N., Deguchi, K., Amano, H., ... & Awashima, T. (2004). Stream applications on the dynamically reconfigurable processor. In *Proceedings of the 2004 IEEE International Conference on Field-Programmable Technology,* Brisbane, NSW, Australia, 137-144, https://doi.org/ 10.1109/FPT.2004.1393261

[38] Lanuzza, M., Perri, S., Corsonello, P., & Margala, M. (2007). A new reconfigurable coarse-grain architecture for multimedia applications. In *2007 NASA/ESA Conference on Adaptive Hardware and Systems*, Edinburgh, UK, 119-126, https://doi.org/10.1109/AHS.2007.10

[39] Khawam, S., Nousias, I., Milward, M., Yi, Y., Muir, `M., & Arslan, T. (2008). The reconfigurable instruction cell array. *IEEE Transactions on very large scale integration (VLSI) systems*, 16(1), 75-85, https://doi.org/ 10.1109/TVLSI.2007.912133

[40] Liang, C., & Huang, X. (2008). SmartCell: A power-efficient reconfigurable architecture for data streaming applications. In *IEEE Workshop on Signal Processing Systems, SiPS 2008,* Washington, DC, USA, 257-262, https://doi.org/ 10.1109/SIPS.2008.4671772

[41] Lee, D., Jo, M., Han, K., & Choi, K. (2009). FloRA: Coarse-grained reconfigurable architecture with floating-point operation capability. In *Proceedings of the 2009 International Conference on Field-Programmable Technology(FPT 2009),* Sydney, NSW, Australia, 376-379, https://doi.org/ 10.1109/FPT.2009.5377609

[42] Shami, M. A., & Hemani, A. (2009). Morphable dpu: Smart and efficient data path for signal processing applications. In *Proceedings of IEEE Workshop on Signal*

Processing Systems, Tampere, Finland,167-172 , https://doi.org/10.1109/SIPS.2009.5336246

[43] Patel, K., McGettrick, S., & Bleakley, C. J. (2011). Syscore: A coarse grained reconfigurable array architecture for low energy biosignal processing. In *2011 IEEE 19th Annual International Symposium on Field-Programmable Custom Computing Machines(FCCM),*Salt Lake City, UT, USA, 109-112, https://doi.org/ 10.1109/FCCM.2011.38

[44] Atak, O., & Atalar, A. (2013). BilRC: An execution triggered coarse grained reconfigurable architecture. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 21(7), 1285-1298, https://doi.org/ 10.1109/TVLSI.2012.2207748

[45] Cong, J., Huang, H., Ma, C. , Xiao, B., & Zhou, P. (2014). A fully pipelined and dynamically composable architecture of CGRA. In *2014 IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM),* Boston, MA, USA, 9-16, https://doi.org/ 10.1109/FCCM.2014.12

[46] Karunaratne, M., Mohite, A. K., Mitra, T., & Peh, L. S. (2017). HyCUBE: A CGRA with Reconfigurable Single-cycle Multi-hop Interconnect. In *Proceedings of the 54th Annual Design Automation Conference 2017* , Austin, TX, USA, Article No.45, https:/doi.org/10.1145/3061639.3062262

[47] Tessier, R., & Burleson, W. (2001). Reconfigurable computing for digital signal processing: A survey. *The Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology,* 28(1-2), 7-27, https://doi.org/10.1023/A:1008155020711

[48] Zhang, C., Lenart, T., Svensson, H., & Öwall, V. (2009). Design of coarse-grained dynamically reconfigurable architecture for DSP applications. In 2009 *International Conference on Reconfigurable Computing and FPGAs,ReConFig'09,* Quintana Roo, Mexico 338-343, https://doi.org/ 10.1109/ReConFig.2009.49

[49] Todman, T. J., Constantinides, G. A., Wilton, S. J., Mencer, O., Luk, W., & Cheung, P. Y. (2005). Reconfigurable computing: architectures and design methods. *IEE Proceedings-Computers and Digital Techniques*, 152(2), 193-207, https://doi.org/ 10.1049/ip-cdt:20045086

[50] Abnous, A. (2001). *Low-Power Domain-specific Architectures for Digital Signal Processing* (Doctoral Dissertation), University of California, Berkeley, CA, [online] . Available http://vada.skku.ac.kr/ClassInfo/dsp/sdr/thesis.pdf

[51] Heysters, P. M., & Smit, G. J. (2003). Mapping of DSP algorithms on the MONTIUM architecture. In *Proceedings of International Parallel and Distributed Processing Symposium,* Nice, France, p180.2, https://doi.org/10.1109/IPDPS.2003.1213333

[52] Galanis, M. D., Dimitroulakos, G., & Goutis, C. E. (2006). Mapping DSP applications on processor/coarse-grain reconfigurable array architectures. In *Proceedings of 2006 IEEE International Symposium on Circuits and Systems (ISCAS),* Island of Kos, Greece, 3666-3669, https://doi.org/ 10.1109/ISCAS.2006.1693422

[53] Itoh, K., Yamaoka, M., & Oshima, T. (2010). Adaptive circuits for the 0.5-V nanoscale CMOS era, *IEICE transactions on electronics,* 93(3), 216-233, https://doi.org/10.1587/transele.E93.C.216

[54] Nowak, E. J. (2002). Maintaining the benefits of CMOS scaling when scaling bogs down. *IBM Journal of Research and Development*, 46(2.3),169-180, https://doi.org/ 10.1147/rd.462.0169

[55] Guo, Y. (2006). *Mapping applications to a coarse-grained reconfigurable architecture* (Doctoral Dissertation), University of Twente, Eindhoven, Netherlands.

[56] Heysters, P. M., Smit, G. J., & Molenkamp, E. (2004). Energy-efficiency of the MONTIUM reconfigurable tile processor. In *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA '04)*, Las Vegas, Nev, USA, 38-44.

[57] Shami, M. A. (2012). *Dynamically reconfigurable resource array* (Doctoral Thesis) Stockholm: KTH Royal Institute of Technology, Sweden.

[58] Tajammul, M. A., Shami, M. A., Hemani, A., & Moorthi, S. (2011). NoC based distributed partitionable memory system for a coarse grain reconfigurable architecture. In *Proceedings of the 2011 24th International Conference on VLSI Design (VLSI Design),* Chennai, INDIA, 232-237, https://doi.org/ 10.1109/VLSID.2011.45

[59] Liang, C., & Huang, X. (2009). SmartCell: An energy efficient coarse-grained reconfigurable architecture for stream-based applications. *EURASIP Journal on Embedded Systems*, 2009:518659, https://doi.org/10.1155/2009/518659

[60] Liang, C. (2009). *SmartCell: An energy efficient reconfigurable architecture for stream processing* (Doctoral Dissertation), WPI, Worcester, MA.