Energy-Efficiency of the MONTIUM Reconfigurable Tile Processor

Paul M. Heysters, Gerard J.M. Smit, Egbert Molenkamp
University of Twente, department EEMCS
P.O. Box 217, Enschede, The Netherlands
p.m.heysters@utwente.nl

Abstract – *Primary requirements of wireless multimedia handheld computers are high-performance, flexibility, energy-efficiency and low costs. A compromise for these contradicting requirements can be found in a heterogeneous SoC. Besides conventional architectures such a SoC contains domain specific coarse grain reconfigurable processors and fine-grain reconfigurable entities. The MONTIUM is a prototype of a novel coarse-grain reconfigurable processor. The SoC template offers a balance between flexibility, efficiency and performance. In this paper the energy and performance characteristics of the MONTIUM are compared with the characteristics of other state-of-the-art (reconfigurable) architectures.*

Keywords *– energy-efficiency, coarse-grain reconfigurable, mobile computing, system-on-chip, MONTIUM*

I. INTRODUCTION

In the CHAMELEON project the subject of research is a heterogeneous reconfigurable SoC architecture in combination with a QoS driven operating system for portable multimedia devices [7]. In this tiled SoC organization conventional architectures are complemented by domain specific coarse-grain reconfigurable architectures. The MONTIUM architecture was devised as a vehicle to investigate the advantages and disadvantages of coarse-grain architectures [3].

The key issue in the design of portable multimedia systems is to find a good balance between flexibility and high-processing power on one side, and area and energy-efficiency of the implementation on the other side. In this paper an implementation of the MONTIUM architecture is compared with other architectures. First, an overview of the MONTIUM architecture is given (section II), followed by realization details (section III) and, finally, these results are compared with current state-of-the-art architectures (section IV).

II. MONTIUM RECONFIGURABLE ARCHITECTURE

The MONTIUM processing tile [3] targets the 16-bit digital signal processing (DSP) algorithm domain. A single MONTIUM processing tile is depicted in Figure 1. The lower part of Figure 1 shows the Communication and Configuration Unit (CCU) and the upper part shows the reconfigurable Tile Processor (TP). The CCU implements the interface for off-tile communication. The definition of the off-tile interface depends on the interconnect technology that is used in the SoC.

The TP is the computing part that can be configured to implement a particular algorithm. At first glance the MONTIUM TP bears a resemblance to a VLIW processor. However, the control structure of the MONTIUM TP is very different. For (energy-) efficiency it is imperative to minimize the control overhead. This can be accomplished by statically scheduling instructions as much as possible at compile time.

Figure 1 reveals that the hardware organization of the tile processor is very regular. The five identical ALUs (ALU1…ALU5) in a tile can exploit spatial concurrency to enhance performance. This parallelism demands a very high memory bandwidth, which is obtained by having 10 local memories (M01…M10) in parallel. The small local memories are also motivated by the locality of reference principle. The datapath has a width of 16-bits and the ALUs support both signed integer and signed fixed-point arithmetic. The ALU input registers provide an even more local level of storage. Locality of reference is one of the guiding principles applied to obtain energy-efficiency in the MONTIUM. A vertical segment that contains one ALU together with its associated input register files, a part of the interconnect and two local memories is called a Processing Part (PP). The five Processing Parts together are called the Processing Part Array (PPA). A relatively

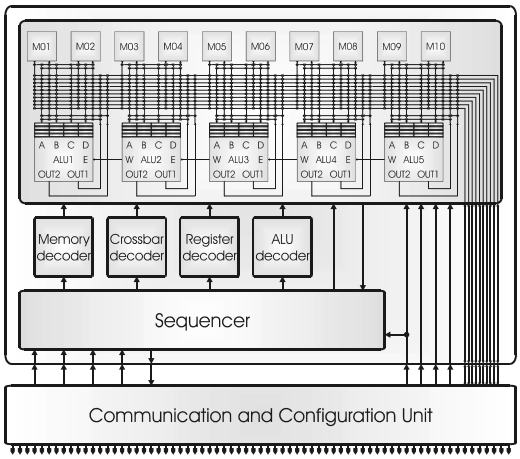

Figure 1: *The MONTIUM processing tile*

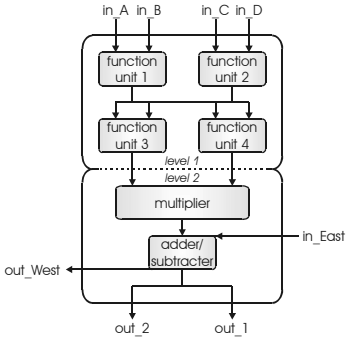

Figure 2: *The MONTIUM TP ALU*

MONTIUM Tile Processor		
Subcomponent	**Area [mm^2]**	**Percentage**
ALUs	0.32	17
Interconnect	0.14	7
Register files	0.09	5
Memories	0.51	29
AGUs	0.09	5
Configuration registers	0.28	15
Decoders	0.33	18
Sequencer	0.07	4
Total area	1.83	100

Table 1: *Area of the MONTIUM TP*

simple sequencer controls the entire PPA. The sequencer selects configurable PPA instructions that are stored in the decoders of Figure 1.

Each local memory is a single-ported SRAM with a capacity of 512 16-bit words (i.e., 8 Kbit per local memory). A reconfigurable Address Generation Unit (AGU) accompanies each memory. It is also possible to use the memory as a lookup table for complicated functions that cannot be calculated using an ALU, such as sine or division (with one constant). A memory can be used for both integer and fixed-point lookups.

Figure 2 depicts a single MONTIUM TP ALU. It has four 16-bit inputs and each input has a private input register file that can store up to four operands. The input register file cannot be bypassed, i.e., an operand is always read from an input register. Input registers can be written by various sources via a flexible interconnect. An ALU has two 16-bit outputs, which are connected to the interconnect. The ALU is entirely combinational and consequentially there are no pipeline registers within the ALU. Neighbouring ALUs can also communicate directly on level 2. The west-output of an ALU connects to the east-input of the ALU neighbouring on the left. The east-west connection does not introduce a delay or pipeline, as it is not registered.

III. ASIC REALIZATION

The ASIC synthesis of the MONTIUM TP was performed using the Philips CMOS12 process technology [5]. This process has a (drawn) gate length of 0.13 µm and a density of 200 kgates/mm^2. For the local data memories and sequencer instruction memory of the MONTIUM TP embedded SRAMs are used. For ASIC synthesis military specification (MILSPEC) requirements (i.e., worst case conditions) are assumed. In particular, the supply voltage is 1.1 V and the temperature is 125°C.

Area

The area of the MONTIUM TP according to the synthesis tool is about 1.8 mm^2. This figure is the area before place and route and hence does not include the area occupied by the routing. If the additional area needed for the wiring is assumed to be 10% of the total area (a realistic figure according to Philips experts) then the size of one MONTIUM TP becomes about 2 mm^2. Table 1 lists the area for the top level MONTIUM TP components. It can be seen that the area utilization is reasonable balanced: roughly 1/3rd is used for the actual datapath (ALUs, registers and interconnect), 1/3rd for storage (memories and AGUs) and 1/3rd for control. The storage capacity of the decoders and the Configuration Registers (CRs) together is about the same as the storage capacity of the sequencer instruction memory. Yet, the decoders and CRs together comprise 33% of the area, whereas the entire sequencer comprises only 4%. This is partially due to the logic that implements the complex configuration address decoding. The larger part of the size difference is due to the difference in implementation. The decoders and CRs are described in VHDL, whereas the SRAMs used in the sequencer are optimized library components. The configuration registers and the decoders are therefore likely candidates for custom optimization.

Timing

The maximum clock frequency for the MONTIUM TP is, according to the synthesis tool, about 40 MHz. This seems a bit disappointing at first, but is better understandable when considered in perspective. The synthesis was performed assuming worst case conditions (i.e., MILSPEC requirements). Under typical conditions, the clock frequency will be higher. An experiment with an FFT processor ASIC (see section IV) in the same technology confirmed this. Typical conditions (1.2 V, 25°C) resulted in a 44% higher clock frequency than worst case conditions (1.1 V, 125°C). Except for the usage of SRAMs, the MONTIUM TP design has not been optimized. A design optimized for the target technology can enhance the performance substantially. There is a third reason why the 40 MHz is a lower bound for the maximum clock frequency. The timing analysis of the synthesis tool is based on the critical path in the MONTIUM TP design. This path, however, is the worst case critical path. For most algorithms the critical path that is actually used is (much) shorter. This situation is comparable to FPGAs in which the worst case critical path would be obtained by chaining all the CLBs (without

Location	Arrival		Departure	
	Signal	Time [ns]	Signal	Time [ns]
PP 5 register file	clk	0.000	rf_dout	0.233
PP 5 FU 1	a	0.233	wf1	3.351
PP 5 FU 4	wf1	3.351	z1b	6.379
PP 5 multiplier	wmy	6.514	zm	9.521
PP 5 adder	wmw	9.705	z2	11.237
PP 5 west			west	11.719
PP 4 east	east	11.719		
PP 4 adder	wme	11.999	z2	13.763
PP 4 west			west	14.331
PP 3 east	east	14.331		
PP 3 adder	wme	14.568	z2	16.260
PP 3 west			wes t	16.812
PP 2 east	east	16.812		
PP 2 adder	wme	17.043	z2	18.754
PP 2 west			west	19.365
PP 1 east	east	19.365		
PP 1 adder	wme	19.628	z2	21.311
PP 1 subtracter	wtnrbf	21.953	bfoa	23.001
PP 1 output latch			out2	23.801

Table 2: *Timing of the MONTIUM TP*

using registers). Obviously, this would result in a very low clock frequency.

Table 2 shows the timing of the worst case critical path in the MONTIUM TP. As can be expected, this path follows all the east-west connections between the ALUs (see Figure 1 and Figure 2). Table 2 can also be used to estimate the delay of the critical path for actual algorithms. This is done by subtracting the delay of unused elements from the total delay. In this way, it can be estimated that the MONTIUM TP ASIC realization can implement an FIR filter at about 140 MHz or an FFT at about 100 MHz. Determining the maximum clock frequency attribute of an algorithm for a particular MONTIUM TP realization can easily be automated. After all, the route of the critical path of an algorithm can be determined at design or compile time. The maximum delays encountered on this route are static properties of the MONTIUM TP hardware. The maximum clock frequency attribute of an algorithm is an important parameter for the QoS driven CHAMELEON SoC template that has an individual configurable clock for every tile.

Energy consumption

The energy consumption of the MONTIUM TP ASIC was estimated for several algorithms using a proprietary Philips Research tool called Diesel, which is an acronym for dissipation estimation software extension for logic simulation. Diesel provides existing logic simulators with additional functionality. It keeps track of the instantaneous signal transitions that occur during a simulation. By combining this transition information with a one-time library characterization, it determines the instantaneous supply current and derivatives thereof. These derivatives are the total instantaneous and average power and energy per net and per hierarchical block and the activity figures per hierarchical block.

For 0.13 µm technology the dynamic power consumption is the major contributor to the overall power behaviour of a cell (typically 70 to 90%) [8]. This contribution scales linearly with increasing load capacitance according to:

$$E = \frac{1}{2} C V_{dd}^{\ 2}$$

where V_{dd} is the supply voltage and C is the sum of the wire capacitance and the input capacitance of the ports connected to the net. The input capacitances are available from known attributes of each cell in the technology library. They do not depend on the final lay-out of the circuit. The wire capacitance does depend on the final lay-out. To obtain realistic power figures the wire capacitance should be specified for each net. Wire capacitances can either be obtained by extraction or by estimation. The first method assumes that the lay-out is available for the circuit. The second method uses hierarchical wire models to estimate the capacitance of each wire in the circuit. For each wire the size of the block containing this wire is used to select the appropriate wire model. The number of loads of this wire is a parameter for the wire model to calculate the wire capacitance.

Since a lay-out is not yet available for the MONTIUM TP, wire models are used for the power simulations. Furthermore, typical operation conditions (1.2 V, 25°C) are used for the power simulations. This is in contrast to the synthesis, which was performed for worst case military conditions (1.1 V, 125°C). However, worst case military conditions are not worst case for power simulations as they assume a lower supply voltage.

Initial power simulations showed that in an early version of the MONTIUM TP more than 50% of the power was dissipated by the clock net during execution of an algorithm. This bad result was a consequence of the technology independent design methodology. In order to improve the power dissipation, clock gating was added to the design. Clock gating is a technique that dynamically switches off parts of the clock net that are not used at a particular time. Clock gating is difficult to implement in a technology independent way. For example, FPGAs do not (really) support clock gating. Instead, it is advised to use clock enables. A clock enable is a separate signal that indicates whether a clock is to be used or not. However, the associated clock wire is always active. Nevertheless, a rudimentary form of clock gating was added to the MONTIUM TP design. New power simulations, with clock gating, showed that the power consumption during execution of an algorithm was reduced by 46%.

The energy and power consumption of the MONTIUM TP ASIC were estimated for a 64 and a 1024-point FFT and for a 5 and a 20-tap FIR filter. Implementation details of the FFTs and FIR filters can be found in [4]. The test algorithms all operate in block processing mode (in contrast to streaming processing mode). Therefore, four stages can be discriminated for each algorithm:

configuration, loading of input data, executing the algorithm and retrieving the results. Configuring the MONTIUM TP is a one time overhead necessary to load an algorithm. After configuration the first block of input data and the coefficients are loaded into the local memories of the MONTIUM TP. Next, the algorithm executes. After it has finished, the results are retrieved from the local memories and the next block of input data can be loaded. The energy estimations for the 64-point and the 1024-point FFT were obtained using one data block of respectively 64 and 1024 complex-number samples. For the 5-tap FIR filter an input block of 512 samples was used and for the 20-tap FIR filter two input blocks of 512 samples each. The estimated power figures by Diesel heavily depend on the blocks of test samples that are simulated. It was endeavored to choose the blocks of test samples in such a way that realistic power figures are obtained.

The energy consumption for the mentioned algorithms is summarized in Table 3 and the average power consumption is summarized in Table 4. The latter is normalized to a clock frequency of 1 MHz. By multiplying the normalized average power with the Montium TP clock frequency the average power consumption at that clock frequency is obtained.

The SRAMs used in the sequencer instruction memory are (3.2 times) too big. During synthesis the right memory size was not available and therefore larger SRAMs were used. The power consumption figures of the sequencer are therefore more pessimistic than they need to be.

Table 5 shows the distribution of the energy dissipation for the execution stage of the 1024-point FFT algorithm. In this figure can be seen that the energy usage of the MONTIUM TP is reasonably balanced: about 1/3rd is used for computation, 1/3rd for storage and 1/3rd for overhead.

Algorithm	Energy [nJ]			
	Configure	Load	Execute	Retrieve
FFT64	183	7	111	7
FFT1024	276	116	2964	104
FIR5	47	76	193	77
FIR20	105	77	864	78

Table 3: *Energy consumption of the MONTIUM TP*

Algorithm	Average power [µW/MHz]			
	Configure	Load	Execute	Retrieve
FFT64	386	229	541	208
FFT1024	386	228	577	203
FIR5	382	148	374	151
FIR20	389	150	420	152

Table 4: *Average power consumption of the MONTIUM TP*

Function	Energy[nJ]	Percentage
Computation	996	34
Storage	839	28
Interconnect	339	11
Control	319	11
Clock	468	16
Total	2964	100

Table 5: *MONTIUM TP energy distribution for a 1024-point FFT*

IV. ENERGY-EFFICIENCY

The MONTIUM TP was not only developed in order to assess its area, performance and energy figures, but above all to estimate the energy-efficiency of the architecture. Intuitively, the notion of energy-efficiency is very clear: the less energy it takes to perform a certain amount of work, the more energy-efficient it is. It is difficult to formalize this intuitive definition into a measure that can be used to determine the energy-efficiency of a hardware architecture. Often the term *computational efficiency* is used in MIPS/Watt [2]. However, this ratio is difficult to use in practice because instructions of various architectures are difficult to compare. For example, the computational precision (e.g., word length, floating- or fixed-point) can greatly differ between various architectures. Just as the amount of parallelism in an instruction (e.g., EPIC, VLIW). Some architectures have control instructions that are in essence pure overhead. Etc. Therefore, the amount of actual work per instruction may vary greatly between the different architectures. It is, therefore, better to compare the energy consumption for a well defined (and realistic) task.

The energy consumption of the same task should be determined for different architectures. In this way, the relative energy-efficiency of the various architectures is obtained. Needless to say, this relative energy-efficiency is only valid for the specific task. For other tasks the comparison may reveal a different outcome. In fact this is the main motivation of a heterogeneous architecture. Some algorithms perform best on fine-grain reconfigurable architectures while others perform better on coarse-grain reconfigurable or general purpose processing (GPP) tiles. When comparing the energy-efficiency of different architectures in this way, care should be taken to choose a set of representative tasks for the application domain at hand.

The MONTIUM TP was designed to be energy-efficiency within the domain of 16-bit DSP algorithms. In this section this design goal is evaluated for the FFT algorithm. The energy consumption of a MONTIUM TP, an ASIC, an FPGA and a GPP are compared. Even for a well defined algorithm like the FFT it remains difficult to compare the energy consumptions of different architectures in a fair way. For example, an algorithm implemented inefficiently on a programmable processor (e.g., due to a poor compiler) will have a negative effect

on the energy consumption. Furthermore, the hardware under comparison may be manufactured using different process technologies or may operate on different minimal supply voltages. Also, the methods (e.g., actual measurements or estimation using a model) used to obtain energy figures may differ. Even when the method is the same, different conditions and assumptions may apply. Comparing energy figures provided by literature or industry is often easier said than done, because not all the details are given. All in all, it is not easy to make a fair energy consumption comparison between different architectures.

FFT algorithm specific reference architecture

An FFT algorithm specific reference architecture (FASRA) was developed in order to compare the MONTIUM TP with both an ASIC and an FPGA. The FASRA design was used to implement a dedicated FFT processor for the latter two architectures. Of course, already many FFT processors exist that can be used to compare with the MONTIUM TP. However, implementing our own gave the control necessary to make a (more) fair energy comparison between the different architectures.

The FASRA design was specified using VHDL. The FFT size and the precision (i.e., the datapath bus width) are generic. The FFT size can be any power of two. Two's complement fixed-point arithmetic is used, with the point positioned directly after the most significant bit. The datapath can compute one radix-2 FFT butterfly per clock cycle. The FFT butterfly component in the FASRA datapath does not contain any internal pipeline registers. The datapath contains ten single ported memories: eight are used for the data and two for the FFT twiddle factors.

FASRA ASIC

A 1024-point FASRA ASIC with a 16-bit datapath was synthesized using the same Philips CMOS12 process technology [5]. The operating conditions were once more set to the worst case (1.1 V, 125°C). Embedded SRAM components from an optimized cell library are used for the memories in the FASRA datapath. The area (excluding wires) of the resulting ASIC is 0.63 mm^2. The area of the datapath is 0.62 mm^2. The remaining 0.01 mm^2 is used for the controller of the datapath. The controller uses less than 2% of the total area. The maximal clock frequency for the FASRA ASIC is 120 MHz.

Power figures are obtained using Philips' Diesel tool. The power simulations assume typical operating conditions (1.2 V, 25°C). Wire models are used for the nets of the 1024-point FASRA ASIC, because a lay-out of the chip has not been made. Only the power dissipation for the execution stage of the FFT computation is estimated.

This means that only the energy consumption for the actual FFT computation is considered. The loading of initial input samples into and the retrieving of the final results from the memories is not part of the power simulation. The same test data was used that was also used for the energy estimation of the 1024-point FFT on the MONTIUM TP ASIC. The total energy consumption of the FASRA ASIC for a 1024-point FFT is 1,938 nJ. The total average power consumption is 377 µW/MHz.

FASRA FPGA

The FASRA design was also implemented on a Xilinx Virtex-II Pro FPGA. The Virtex-II Pro was chosen because it is realized using a 0.13 µm CMOS process, which was also used for the MONTIUM TP ASIC. Unfortunately, the area of a Virtex-II Pro is not known to us. The Virtex-II Pro is a hybrid architecture that contains both fine and coarse-grain elements. The fine grain elements are configurable logic blocks (CLBs).

Programmable input/output blocks (IOBs) and block memories are available in the Virtex-II Pro. The block memories of the Virtex-II Pro are dual port RAMs with a capacity of 18 Kb each. In the Virtex-II Pro a coarse-grain block multiplier is associated with each memory block. The block multiplier is a dedicated 18×18-bit two's complement signed multiplier. It is optimized for operations based on the block memory content on one port, but can be used independently of the block memory. In addition, the Virtex-II Pro also has processor blocks. A processor block contains a PowerPC 405 RISC CPU.

A XC2VP2 device was used for synthesis. The XC2VP2 is the smallest device in the Virtex-II Pro family. It contains 1,408 slices, 12 multiplier blocks and 12 block memories. It does not contain a PowerPC. Both a 64-point and a 1024-point FASRA processor with a 16-bit datapath were implemented on a XC2VP2. The resource utilization is listed in Table 6. Of both processors two versions were made: one that makes use of the dedicated block multipliers (i.e., design A) and one that does not make use of them (i.e., design B). All implementations make use of the block memories.

Power consumption estimations for the XC2VP2 were done using Xilinx' XPower tool [9]. XPower uses (XC2VP2) device knowledge, design information and simulation data to calculate an estimate of power within a 10% margin of error. The design information consists of a circuit description (netlist) of the design that has been placed and routed. The simulation information is obtained

Design		Multiplier blocks [#]	Block memories [#]	Slices [#]	Max clock frequence [MHz]
FASRA 64	A	4 (33%)	10 (83%)	502 (35%)	62
	B	0 (0%)	10 (83%)	1110 (78%)	48
FASRA 1024	A	4 (33%)	10 (83%)	583 (41%)	63
	B	0 (0%)	10 (83%)	1190 (84%)	47

Table 6: *XC2VP2 resource utilization*

XC2VP2 power consumption [μW/MHz]				
Design	**FASRA64**		**FASRA1024**	
Multipliers used	**yes**	**no**	**yes**	**no**
Clocks	132	151	134	158
Quiescent	6200	6200	6200	6200
Signal	2847	6075	3127	6897
Logic block	2952	4940	3684	5561
Inputs/outputs	24	24	9	8
Total	12155	17390	13154	18824
Total dynamic	5955	11190	6954	12624

Table 7: *XC2VP2 power consumption*

from a timing simulation of the (same) netlist. The simulated activity of all the nets in the netlist is logged in a value change dump (VCD) file. XPower uses the VCD simulation data to determine the activity rates of all the nets in the design.

For the energy estimations typical operation conditions were used. The (minimal) core supply voltage for the XC2VP2 is 1.5 V. The ambient temperature is set to 25°C. Again, only the energy consumption for the actual FFT computation is considered. The loading of input data into and the retrieving of the results from the memories is not part of the VCD simulation data. The same test data is used that was also used for the energy estimations of the 64 and 1024-point FFTs on the MONTIUM TP ASIC. The results of the power estimations are presented in Table 7. The quiescent (or static) power is the power consumed with no signals switching. The static power consumption has been subtracted from the total power consumption in order to obtain the (total) dynamic power consumption.

Avispa

The Avispa block accelerator from Silicon Hive [6] has been developed for efficient and reconfigurable acceleration of DSP algorithms such as FFT, OFDM, and error correction. The Avispa contains a highly parallel ultra large instruction word (ULIW) datapath composed of five interconnected processing and storage elements. The Avispa has in total 75 functional units, which include four 16×16 multipliers, five ALUs and five local storage memories with a total capacity of 12 KB.

The Avispa is developed using the same technology and tools that have been used for the MONTIUM TP ASIC. The automatically generated VHDL description of the Avispa was synthesized using the Philips CMOS12 process technology. The synthesis conditions were set according to MILSPEC requirements (1.1 V, 125°C). The SRAMs used in the Avispa are similar to those in the MONTIUM TP. The area of the Avispa is 6.5 mm^2 and the maximum clock frequency is 150 MHz.

Power simulations for typical operating conditions (1.2 V, 25°C) were performed using Philips' Diesel tool. Wire models are used for the nets of the Avispa. The Avispa power simulations included a 64-point FFT and a 2048-point FFT. Only the energy consumption for the actual FFT computation is considered. The loading of input data into and the retrieving of the results from the memories is not part of the power simulation. The average power consumption for the Avispa is 852 μW/MHz.

ARM920T

To put the above in perspective we will give a quick estimation for a general purpose processor, i.e., the ARM920T hard macro-cell [1]. The ARM920T is a 32-bit low-power processor for mobile embedded systems. It features a 16KB instruction cache and 16KB data cache and a memory management unit (MMU) enabling support for all major operating systems. In 0.13 μm technology the ARM920T has an area of 4.7 mm^2. It has a maximal clock frequency of 250 MHz, when worst case operating conditions (1.1 V, 125°C) are assumed. According to ARM Ltd. [1], the typical power consumption is 0.25 mW/MHz. According to [10], the execution part of a FFT butterfly takes 21 clock cycles. If we assume that all instructions are already in the data cache and all the data is already in the data cache, then the power consumption of one FFT butterfly is about 5250 μW/MHz. The FFT butterfly frequency of an ARM running at 250 MHz is 12 MHz.

Comparison

Figure 3 and Figure 4 depict a comparison between various architectures normalized to one FFT butterfly per clock cycle. Figure 3 shows the average power consumption when FFT butterflies are computed at a rate of 1 MHz. The average power consumption of an particular architecture is obtained by multiplying the normalized average power with the clock frequency. This figure clearly shows the energy advantage of coarse-grain reconfigurable architectures. Note that this can also be observed from the Xilinx Virtex-II Pro implementations. The usage of the coarse-grain multiplier blocks (design A in Figure 3) improves the energy consumption of the FFT computation considerably.

Figure 4 shows the maximum frequency at which an architecture can compute FFT butterflies. The Avispa outperforms the ASIC because it is has a highly pipelined architecture. The implementations based on the FASRA design do not support pipelining. The MONTIUM TP ALUs are also not pipelined. Even though performance was not the main goal for the MONTIUM TP architecture Figure 4 shows that an un-optimized version of the MONTIUM TP hardware can achieve a considerable performance.

Of all FFT implementations, only the FFT implementation on the ARM was highly optimized. Although we assumed that the ARM runs a clock frequency of 250 MHz, the number of butterflies that the ARM can compute is rather low. This is because the ARM has to perform a number of instructions for one butterfly whereas the other architectures perform a butterfly in one clock cycle.

V. CONCLUSION

In this paper the coarse-grain reconfigurable MONTIUM tile processor is compared with other (reconfigurable) architectures. It is shown that the MONTIUM TP balances both energy distribution and area (about 1/3rd processing, 1/3rd storage, 1/3rd overhead). It is important to minimize the energy consumption of control overhead and data storage, as they do not contribute to the actual computation. In the Montium TP the share of the energy consumption that is used for the actual computation is much bigger than in general purpose processors.

The FFT algorithm is used to compare the energy consumption and performance of a MONTIUM TP with an ASIC, a Xilinx Virtex-II Pro FPGA, a Silicon Hive Avispa and an ARM920T processor. The results presented in this paper confirm that coarse-grain reconfigurable architectures provide a useful alternative for mobile devices for which energy-efficiency is crucial.

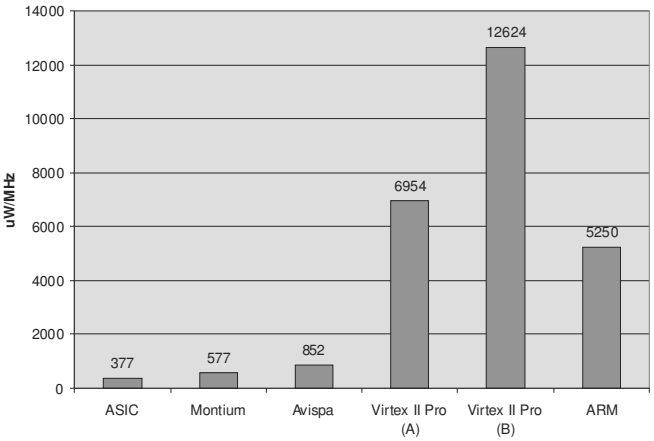

Figure 3: *Average dynamic power consumption when FFT butterflies are computed at a rate of 1 MHz*

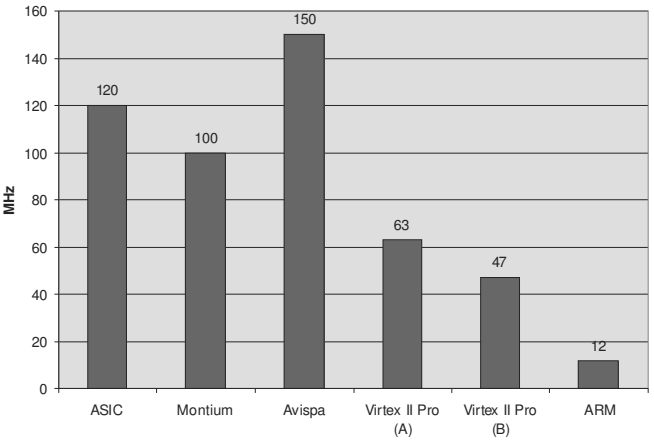

Figure 4: *Maximum frequency at which FFT butterflies can be computed*

ACKNOWLEDGEMENTS

This research is supported by the PROGram for Research on Embedded Systems & Software (PROGRESS) of the Dutch organization for Scientific Research NWO, the Dutch Ministry of Economic Affairs and the technology foundation STW.

Furthermore we would like to thank Jos Huisken from Philips Research / Silicon Hive for his assistance during the synthesis and power estimation sessions.

REFERENCES

[1] http://www.arm.com/.

[2] Th. Claasen: "Is High-Speed the Only Solution to Exploit the Intrinsic Computational Power of Silicon?", *keynote at the ISSCC 99*, February, 1999.

[3] P.M. Heysters, G.J.M. Smit & E. Molenkamp: "Montium – Balancing between Energy-Efficiency, Flexibility and Performance", *Proceedings of Engineering of Reconfigurable Systems and Algorithms (ERSA) 2003*, pp 235-242, Las Vegas, Nevada, 2003.

[4] P.M. Heysters & G.J.M. Smit: "Mapping of DSP Algorithms on the Montium Architecture", *Proceedings of the 17th International Parallel & Distributed Processing Symposium Reconfigurable Architectures Workshop (RAW) 2003*, Nice, France, April 2003, ISBN 0-7695-1926-1.

[5] http://www.semiconductors.philips.com/.

[6] http://www.silicon-hive.com/.

[7] G.J.M. Smit, P.J.M. Havinga, L.T. Smit, P.M. Heysters & M.A.J. Rosien, "Dynamic Reconfiguration in Mobile Systems", *Proceedings FPL 2002,* Montpellier France, pp 171-181, September 2002.

[8] H. Veendrick: "Deep-Submicron CMOS ICs", *2nd edition, Kluwer academic publishers*, 2000, ISBN 90-440-01116.

[9] http://www.xilinx.com/.

[10] K. Yarlagadda, "ARM Refocuses DSP Effort", *Micro-design resources*, Microprocessor report, June 1999.