

---

## Energy-aware compilation and hardware design for VLIW embedded systems

---

José L. Ayala\* and Marisa López-Vallejo

Departamento de Ingeniería Electrónica,  
Universidad Politécnica de Madrid, Spain  
E-mail: jayala@die.upm.es      E-mail: marisa@die.upm.es  
\*Corresponding author

David Atienza

DACYA, Universidad Complutense de Madrid, Spain  
E-mail: datienza@dacya.ucm.es

Praveen Raghavan and Francky Catthoor

IMEC vzw, Leuven, Belgium  
E-mail: ragha@imec.be      E-mail: catthoor@imec.be

ESAT, Katholieke Univ. Leuven, Belgium

Diederik Verkest

IMEC vzw, Leuven, Belgium  
E-mail: dverkest@vub.ac.be

ESAT, Katholieke Univ. Leuven, Belgium

Vrije Univ. Brussel, Belgium

**Abstract:** Tomorrow's embedded devices need to run multimedia applications demanding high computational power with low energy consumption constraints. In this context, the register file is a key source of power consumption and its inappropriate design and management severely affects system power. In this paper, we present a new approach to reduce the energy of shared register files in forthcoming embedded VLIW processors running real-life applications up to 60% without performance penalty. This approach relies on limited hardware extensions and a compiler-based energy-aware register assignment algorithm to deactivate at run-time parts of the register file (i.e., sub-banks) in an independent way.

**Keywords:** register file; low-power optimisations; compiler; VLIW processors; register assignment.

**Reference** to this paper should be made as follows: Ayala, J.L., López-Vallejo, M., Atienza, D., Raghavan, P., Catthoor, F. and Verkest, D. (2007) 'Energy-aware compilation and hardware design for VLIW embedded systems', *Int. J. Embedded Systems*, Vol. 3, Nos. 1/2, pp.73–82.

**Biographical notes:** José L. Ayala is an Assistant Professor at the Department of Electronic Engineering at the Politecnica University of Madrid, Spain. He completed his PhD Degree from the Politecnica University of Madrid in 2005. Since 2001, he has been researching in power estimation and power optimisation techniques for processor-based systems.

Marisa López-Vallejo received the MS and PhD Degrees in Telecommunication (Electrical) Engineering from the Technical University of Madrid in 1992 and 1999, respectively. She is currently an Associate Professor in the Department of Electronic Engineering. Her research interests include VLSI design for communication systems and CAD tools for hardware-software co-design, with particular emphasis on power optimisation for embedded systems.

David Atienza received the PhD Degree in Computer Science from Complutense University of Madrid, Spain in 2005. He is currently an Assistant Professor at the Computer Architecture and Automation Department (DACYA) of UCM. His research interests include several aspects of design technologies for integrated circuits and systems, with particular emphasis on flexible Networks-on-Chip (NoC) interconnection paradigms for Multi-Processors System-on-Chip, design automation and low-power design.

Praveen Raghavan is currently pursuing a PhD Degree in the Department of Electrical Engineering at ESAT, Katholieke Universiteit Leuven and also works as a Researcher at IMEC vzw, Leuven. His research areas include design of ultra low-power processors, embedded system design and processor architecture.

Francky Catthoor received a PhD in Electrical Engineering from the K.U. Leuven, Belgium in 1987. Since then, he has headed several research domains in the area of architectural methodologies and system synthesis for embedded multimedia and telecom applications. His current research activities mainly belong to the field of system-level exploration, with emphasis on data storage/transfer and concurrency exploitation.

Diederik Verkest received the PhD Degree in Applied Sciences from the Katholieke Universiteit Leuven (Belgium) in 1987 and 1994, respectively. Currently, he is a Professor at the University of Brussels (VUB) and at the University of Leuven (KU-Leuven). His research work is related to design technology for embedded systems, hardware/software co-design and re-configurable systems.

## 1 Introduction

Current trends in communications, multimedia, networking, and other areas encourage the development of high-performance platforms that include structures to hold complex algorithms that cannot be supported by simple hardware. In this sense VLIW-like processors are the only solution that can provide a suitable performance-power trade-off for this kind of applications. Moreover, product complexity continues to increase, making scalability a must for these complex architectures. Additionally, the time-to-market pressure provokes a situation where a design group can no longer start from scratch. These assumptions make the use of heterogeneous multi-processor architectures the only solution to meet design goals in the required time to market (Wolf, 2004). In fact, several platforms have recently appeared from important semiconductor companies that confirm this tendency to multi-processor systems, such as ST Nomadik (ST Microelectronics, 2004), Philips Nexperia (Philips Electronics, 2004) or TI OMAP (Texas Instruments, 2004).

On the other hand, most current systems require battery operation, which puts intense pressure on energy consumption. Actually, given the impact that static power causes in current sub-micron technologies, as indicated by Kim et al. (2003), energy consumption becomes a critical design metric. New embedded applications require an enormous computational performance (2–30GOPS) that need to be executed with low energy consumption demands (0.3–2 W) for battery duration constraints (Viredaz and Wallacha, 2003). Although new processors with multiple processing units can fulfil these performance figures, they consume too much power (10–100 W) as outlined by Bose et al. (2002). Therefore, while keeping the performance results, the power consumption needs to be at least two or three orders of magnitude lower to be used in embedded environments. Within this context, methods to reduce the power consumption of the new multi-processor embedded platforms are in great demand.

One of the main factors that affects performance and power consumption of the new embedded platforms is the memory hierarchy. As a matter of fact, an inappropriate

design of the memory subsystem and the management of the data transfers between its levels can use up to 70% of the total power consumed in the system and up to two orders of magnitude in performance for dynamic multimedia systems (Viredaz and Wallacha, 2003). Moreover, very recently it has been found that in new proposed embedded platforms with several processing elements, the shared register file plays a very important role as part of the memory subsystem and it can heavily affect the cycle time and consumes a very significant portion of the aforementioned percentage of the total energy consumed by the memory hierarchy (Texas Instruments, 2004). Furthermore, it has become one of the critical processor hotspots, especially for VLIW architectures, as Lambrechts et al. (2005) have shown. The main reasons are similar to those found in other layers of the memory subsystem. The register file has to support concurrent access of the several present processors and continuous exchanges of information with the L1 memory caches, therefore it is large and multi-ported. These characteristics, as they occur with memory layers on the top, lead to a large increase in the power dissipation (and indirectly temperature as well) of the whole system (Abella and Gonzalez, 2003). Hence, it is crucial to reduce the energy spent on it.

In this paper we introduce a new hardware approach to reduce the energy of the shared register file in upcoming embedded architectures with several VLIW processors. This work relies on a set of special hardware extensions that are controlled by the compilers of these new embedded platforms. This complete hardware/software approach enables reduction of the energy consumed in the register file of forthcoming embedded architectures with multiple processing elements without incurring in performance penalties. The experimental setup is built over the Coarse-grained Reconfigurable Instruction Set Processor (CRISP) framework (Op de Beeck et al., 2001), which provides the compilation and simulation capabilities.

The remainder of the paper is organised as follows. In Section 2, we describe some related work. In Section 3, we present the baseline architecture used in our case studies. In Section 4, we explain the proposed architectural

extensions for these new embedded platforms, while the compiler modifications are explained in Section 5. Then, in Section 6, we briefly introduce the case studies and present the experimental results obtained with our proposed hardware/software approach. Finally, in Section 7 we draw our conclusions.

## 2 Related work

Nowadays, two major types of processing architectures have been proposed to achieve low power processing of multimedia and consumer applications. First of all, most forthcoming low power embedded architectures (e.g., ST Nomadik by ST Microelectronics (2004), Philips Nexperia by Philips Electronics (2004) and TI OMAP by Texas Instruments (2004)) fall under the Digital Signal Processing (DSP) paradigm. These types of platforms are typically customised to handle signal processing operations efficiently. Interesting domain-specific commercial DSP processors are the TI *C64x* series by Texas Instruments (2001) and Coolflux Philips-PDSL (Philips PDSL, 2005). Second, Application-Specific Instruction set Processors (ASIPs) have been presented as an alternative for low power embedded processing (Glokler and Meyr, 2002). Good examples of commercially available ASIPs are Altera's NIOS (Altera, 2001), Tensilica's Xtensa (Gonzalez, 2002) and STLx (Faraboschi et al., 2000). The academic research performed by Biswas et al. (2004) and Yu and Mitra (2004a, 2004b) in the design of ASIPs has focused on the problem of identification and implementation of an efficient set of instruction set extensions. Although most of the work has focused on improving performance, not much work has been done specifically in the area of reducing energy consumption. Atasu et al. (2003) present a way to extend the instruction set based on the energy-efficiency figures of the new instructions. Even though these two types of architectures provide reduced power consumption compared to general-purpose processors, since these are both largely based on the Very Long Instruction Word (VLIW) paradigm (Benini et al., 2001), the register files are usually quite large and have several ports (although they tend to be distributed). This leads to an energy/power wastage that is the aim of our research in this paper.

All these options of multi-processor platforms for multimedia embedded processing, one key element that heavily affects their deployed processing efficiency and consumed energy, apart from the register file, is the rest of the memory subsystem. Therefore, many researchers have studied different ways to optimise the memory bandwidth of the different levels of the on-chip memory hierarchy, taking into account power consumption apart from performance (see Benini and de Micheli, 2000; Panda et al., 2001 for good overviews). Also, Saghir et al. (1996) and Grun et al. (2001) propose to exploit the off-chip memory bandwidth of new dynamic memories in embedded systems, such as DRAM and SDRAM memories, by using the compiler/linker. Additional work in this field by Chang et al. (2000) and Luz et al. (2002) has suggested how to

maximise performance by distributing the data across the different banks such that as many accesses as possible can be done in parallel.

However, even though the memory hierarchy has been already largely studied, work related to the register file has started only recently. In high performance processors research work can be found that is devoted to defining mechanisms that decrease the energy of multiported register files. Regarding the hardware approaches to the problem, Zyuban and Kogge (1998) have studied the complexity of shared register files and Seznec et al. (2002) and Zyuban and Kogge (2001) have proposed distributed schemes and techniques to split the global microarchitecture into distributed clusters with subsets of the register file and functional units. Similarly, trying to reduce the complexity of the register files, Cruz et al. (2000) have studied the benefits of multilevel register file organisations. Conversely, Park et al. (2002) present other techniques that retain the idea of a centralised architecture, but the register file is split into interleaved banks, which reduces the total number of ports in each bank. In a more general context, Koen et al. (2003) have proposed efficient voltage scaling techniques according to the application's behaviour, which can efficiently reduce the overall power consumption of the system. However, in all these previous approaches it has not been studied how to apply similar techniques to multi-processor systems with shared register files by including a set of hardware extensions and power-aware compilation to control them, as we propose in this paper.

In addition, from the software viewpoint, several approaches have been proposed to alleviate the problem of the register file. In the last years, several software pipelining strategies to distribute the use of the register file, targeted at reducing memory pressure in VLIW systems, have been outlined (Akturan and Jacome, 2000, 2001). Also, Ayala et al. (2003) and Ayala and López-Vallejo (2004) have recently presented different compiler techniques, including complex register renaming, to reduce the energy spent in the register file of in-order processors. Nevertheless, such techniques were not aimed at enabling the use of voltage scaling mechanisms in multi-processor environments as we introduce in the present approach.

Finally, loop unrolling mechanisms have been extensively studied in the literature related to multi-processor systems for many years. All the proposed techniques perform better for in-order architectures and were originally designed for mono-processor systems (Huang and Leng, 1999). Therefore, current widely-available compilers are not able to exploit the dynamic scheduling facilities found in out-of-order processors, and the instruction level parallelism achieved is not so spectacular. On the other hand, the unrolling of outer loops (or the unrolling of inner loops by large factors) exploits the register requirements and increases the energy consumption of the register file (Vijaykrishnan et al., 2000). Moreover, recent research in modern architectures performed by Seng and Tullsen (2003) has shown how loop unrolling proved to have little effect in terms of program execution

time. As a result, we are not aware of any previous work in this field that considers the increment on energy consumption due to the existence of unused registers during normal operations in the original code or after loop unrolling transformations, and studies convenient mechanisms to power them off using a combination of hardware and software techniques as we propose.

### 3 Baseline hardware architecture

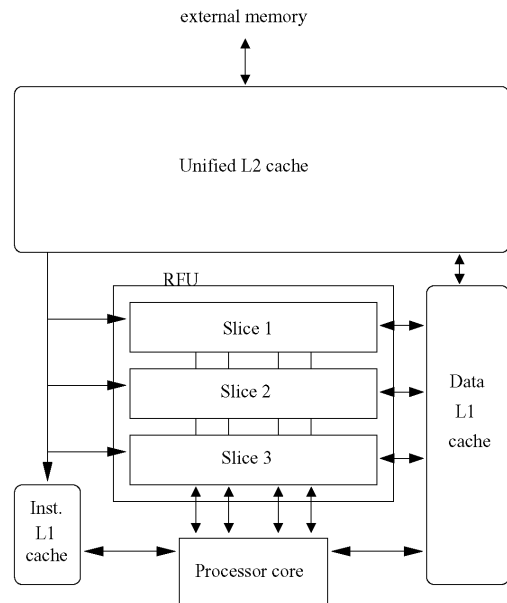
In this work, we have used the compilation and simulation capabilities provided by the CRISP framework (Op de Beeck et al., 2001). It is a re-targetable compiler and simulator framework based on Trimaran (Trimedia Technologies Inc., 1999), which is cycle accurate. The baseline architecture described by CRISP consists of a selectable number of processing elements as in VLIW processors, with a coarse-grained reconfigurable logic that can be adapted to each desired DSP instruction set to be simulated. Furthermore, this framework also enables data and instruction clustering. The overall architecture of the type of VLIW processor that we use in this paper is shown in Figure 1.

As Figure 1 indicates, it includes a main processor core and coarse-grained reconfigurable logic, which is divided into slices. The main processor core can be any type of processor and it is included in the CRISP template architecture to be able to schedule instructions that in real-life VLIW systems cannot be executed efficiently (e.g., control and non-parallel operations). In our simulated architecture the main processor core is a simple RISC (Reduced Instruction Set Computer) core, but it is not relevant in our simulations since our case studies only include a very insignificant proportion of operations compared to DSP-like or loop operations, as we indicate in our experimental results (Section 6). Then, the slices for the configurable VLIW processing part are mapped onto the CRISP's Reconfigurable Functional Unit (RFU) part (Op de Beeck et al., 2001). The RFU allows extensive customisation of the VLIW processing units for the desired instruction set to be used in the simulated architecture within CRISP. In the RFU an operation can be issued after every clock cycle. Regarding interconnections, the RFU part reads/writes data from/to the main shared register file. Then, the additional RISC processor core and the RFU read their instructions from the level 1 instruction cache and move the data from the level 1 data cache to the shared register file. Finally, both types of caches are connected to a unified level 2 cache, which is, in turn, connected to an external memory.

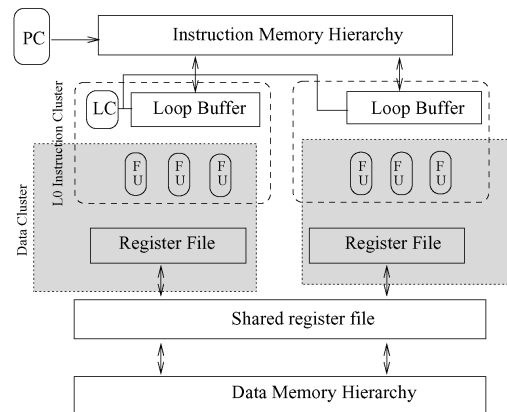
As we have previously mentioned, the RFU part of the emulated architecture is divided in reconfigurable slices (e.g., in Figure 1 three slices are depicted). Then, Figure 2 details the internal structure of each slice. This figure shows that each slice contains several coarse-grained Processing Elements (PEs), which can be an ALU, shifter, multiplier or memory unit. As a matter of fact, such complex processing elements are better suited than the traditional

logic blocks based on Look Up Tables (LUTs) for the execution of the operations typically found in multimedia applications, which are word-oriented and not bit-oriented. These complex PEs allow the reconfigurable logic to operate at higher frequencies with lower power consumption when compared to other possible embedded multi-processor architectures using traditional FPGAs.

**Figure 1** CRISP: overall emulated architecture



**Figure 2** Internal architecture of one slice of RFU in CRISP



In addition, each slice typically includes two instructions and data clusters. A data cluster includes a distributed register file, which can be used across multiple slots of functional units. Next, an instruction cluster is formed by using a distributed instruction buffer (or also called loop buffer (Op de Beeck et al., 2001)) across multiple issue slots of the VLIW processor. In fact, all the units of one data cluster can access the data register file in that cluster. However, accessing data from another data cluster (i.e., another local register file) is relatively costly since it requires an explicit inter-cluster copy operation to transfer the data from the source data cluster to the destination data cluster. In contrast, one data cluster with all the functional units can access any data present in that register file.

Therefore, in order to provide enough bandwidth for all these potential concurrent accesses, our baseline architecture includes two read and one write port of the register file, which are allocated to each slot of the VLIW processor. All functional units in one slot are connected to these three ports via a full crossbar.

#### 4 Proposed architectural extensions

The baseline architecture described by CRISP has been extended and modified in several ways to support the power reduction mechanisms proposed in this paper. They are described in the following paragraphs of this section. Moreover, these architectural modifications enable the use of voltage scaling techniques to reduce power consumption in the register file by turning down the voltage power supply of the unused areas of this device.

First, the register file shared among all processing elements has been split into several banks, which can be independently accessed by the processors. Then, a Dynamic Voltage Scaling (DVS) technique is applied to turn the unused banks into a low power state and thus save as much energy as possible in the system.

Turning memory devices into a low-power state has been proposed before in the literature for different objectives. In fact, previous research has focused on turning off unused memory banks (or other resources) by gating the power source. However, when the objective is a memory device, the cost of recovering the lost information could hide any power saving or, at least, represent a very significant time penalty. For example, there is an extra cost to load the data from the main memory. Moreover, when working with the register file, there is no way to recover data from memory without extra accesses to the cache, and something has to be done in the unused registers to prevent the information from being lost. In fact, various authors have suggested that by turning these unused registers into a low-power state (*drowsy* state), the power consumption can be reduced to a minimum without data loss (Hanson et al., 2001).

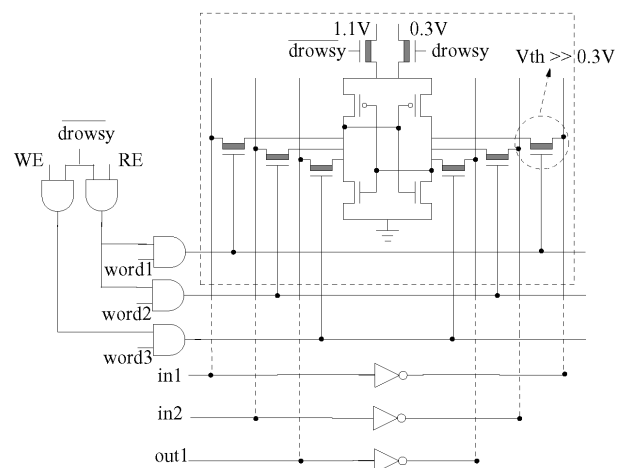
As a result, the information in a memory cell is preserved while it is in the *drowsy* state. However, the data line must be restored to a high-energy mode before its contents can be accessed. One circuit technique for implementing *drowsy* memory devices is Adaptive Body-Biasing with Multi-Threshold CMOS (ABB-MTCMOS), where the threshold voltage is dynamically increased to yield reduction in leakage energy. This leakage reduction technique requires that the voltages of the N-well and of the power and ground supply rails are changed whenever the circuit enters or exits the *drowsy* mode. Since the N-well capacitance of the PMOS devices is quite significant, this increases the energy required to switch the memory cell to high-power mode and can also significantly increase the time needed to transition to/from *drowsy* mode. A more efficient approach to achieve the *drowsy* state is proposed by Flautner et al. (2002), where a DVS technique is exploited to reduce static power

consumption. In fact, due to short-channel effects in deep-submicron processes, leakage current is significantly reduced with voltage scaling. Thus, the combined effect of reduced leakage current and voltage yields a dramatic reduction in leakage power. This is the solution used in our approach to reduce energy consumption.

The method proposed by Flautner et al. utilises DVS to reduce the leakage power of cache cells. By scaling the voltage of the cell to approximately 1.5 times  $V_{th}$ , the state of the memory cell can be maintained. Due to the short-channel effects in high-performance processes, the leakage current will dramatically reduce with voltage scaling. Since both voltage and current are reduced in DVS, a dramatic reduction in leakage power can be obtained. Since the capacitance of the power rail is significantly less than the capacitance of the N-wells, the transition between the two power states occurs more quickly in the DVS scheme than the ABB-MTCMOS approach. Possible disadvantages of the *drowsy* circuit are increased susceptibility to noise and the variation of  $V_{th}$  across process corners. The first problem may be corrected with careful layout because the capacitive coupling of the lines is small. The second problem, variation of  $V_{th}$ , may be handled by choosing a conservative  $V_{DD}$  value.

Figure 3 shows the modified register file cell used to support the *drowsy* state. As can be observed, the dual power supply is switched to low  $V_{DD}$  when the cell is in *drowsy* state. It is necessary to use *high* -  $V_{th}$  devices as pass transistors because the voltage on bit lines could destroy the cell contents. Before a register cell can be accessed, the power supply has to be switched to high  $V_{DD}$  to restore the contents and allow the access. An extra *read\_enable*/*write\_enable* gating circuit assures that the memory cell is not accessed (i.e., read or written) while being in *drowsy* state.

Figure 3 Register file cell



The register file architecture is split into independent banks and each sub-bank counts with the additional logic required to implement the DVS state. Since the low power consumption state is selected for the whole bank instead of a specific register, the overhead of the control logic is greatly minimised.

The hardware support described above is exploited by the compiler to power up banks of registers in the shared register file when they are needed by the several processing elements. In normal execution of the system, most banks of the shared register file are kept in a low power state thanks to our modified register assignment implemented in the compiler. In fact, only when needed, the register file banks are powered up to fulfil the register demands of the code, without performance degradation.

## 5 Compiler optimisations

The complete hardware architecture depicted in the previous section has also been extended with the design of a specific compiler. This compiler, Trimaran, is included in the CRISP framework that we have used to perform the experimental analysis. The compiler includes a modified version of the register assignment algorithm which takes advantage of the underlying architecture.

The register assignment is the phase of any compiler that determines which register(s) to use for each program value selected during register allocation, while the register allocation is the phase that determines which values will be placed in registers. As a matter of fact, computer architectures (*out-of-order* processors) can destroy this first assignment by means of a hardware mechanism designed to avoid hazards, namely using *register renaming*, as was studied by Ayala et al. (2003).

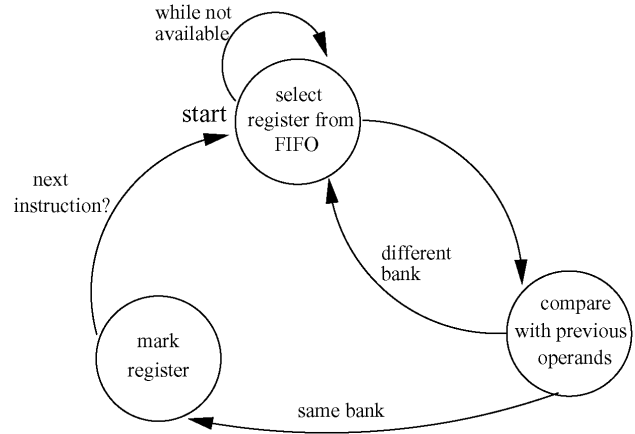
Traditionally, the register assignment algorithm has been designed to choose registers from the whole pool of free registers without any other constraint. In the case of Trimaran, as in many other compilers, when it tries to assign an architectural register to the instruction operands, it retrieves the first available register from a First Input First Output (FIFO) list of free registers. In fact, the order of the registers inside the list is not representative and depends on the specific hardware architecture. Moreover, since Trimaran does not consider any restriction on assigning the registers, they are selected from the FIFO without a particular order. Therefore, the assigned registers can easily come from different register file banks if no modification to the register assignment algorithm is accomplished.

The register assignment policy we have implemented in the compiler modifies the aforementioned traditional assignment of Trimaran by promoting every operand in the instruction to the same register file bank. With this modification, most of the registers are selected from the first bank in the register file and the other banks can be kept in the low power state by turning down the voltage power supply.

The structure of the algorithm, followed by the compiler to assign the architectural registers, is shown in Figure 4. First, the first available register in the list of free registers is selected. This register is double-checked to be free and not system-reserved. Then, it is compared to the registers assigned to the other operands of the instruction. If the register file bank for the operand under assignment does not

match any of the other operands of the instruction, this register is discarded and the procedure is repeated until a register belonging to the same bank is found. When the register is selected, the liveness of the register is calculated and the annotation is generated. It is important to remark that the compiler has been designed with enough flexibility to perform the register assignment algorithm with varying sizes of the register file bank, as the experimental results in Section 6 show.

Figure 4 Register assignment algorithm



Finally, when the register assignment is performed with our proposed algorithm, we can observe that in most of our studied multimedia and encryption real-life applications, the registers can belong to the same bank and just a few of them need to be selected from another register file bank (see Section 6 for more details).

## 6 Case studies and experimental results

We have applied the proposed method to several case studies that represent different modern multimedia, wireless and encryption application domains. These benchmarks come from the MediaBench benchmark suite (Lee et al., 1997) and real-life applications. They are the following ones:

- *adpcm\_decode*. This benchmark performs adaptive differential pulse code demodulation, which is a very simple audio decoding algorithm particularly suited for embedded systems.
- *g721decode*. It is a reference implementation of the CCITT (International Telegraph and Telephone Consultative Committee) for the G.721 voice decompression standard.
- *mesatexgen*. It is an implementation of the Mesa 3D-graphics library clone of OpenGL. In this demo programme it is used to generate a texture mapped version of the Utah teapot. All display output functions were removed from the library due to the lack of support in CRISP for this type of I/O operation.

- *aes*. This benchmark is an implementation of the Advanced Encryption Standard (AES), also known as *Rijndael*. It is a block cipher that uses a fixed-block size of 128 bits and a key size of 128, 192 or 256 bits. It has been adopted since 2001 as an encryption standard and is used worldwide today.
- *blowfishencode*. It is an implementation of the popular royalty-free block encryption algorithm, designed by the highly respected cryptographer Bruce Schneier. The blowfish encoder encrypts and decrypts in 64-bit blocks, and can use a key length of up to 448 bits.
- *epic*. This benchmark is an image compression utility that uses a bi-orthogonal critically sampled dyadic wavelet decomposition and a combined run-length/Huffman entropy coder. It has been designed specifically for embedded systems to enable fast decoding without floating-point hardware.
- *sha*. It is an implementation of the Secure Hash Algorithm (SHA-1 Hash) for use in the Secure Hash Standard (SHS), which can be used to generate a condensed representation of a message called a message digest. It produces a 160-bit hash and there are no known attacks against it, making it very convenient for inter-communication between wireless embedded terminals.
- *mpeg2decode*. It is an implementation of the decoder of the MPEG2 standard for digital video transmission. In this case, the most important kernel inside the application is the inverse cosine transform, which is a highly parallel operation.

All these benchmarks are largely loop-execution dominated, like most embedded applications. For this reason, the algorithm and our shown results focus on the registers assigned inside loops in the VLIW processing part of the system (slices of RFU, see Section 3 for more details), which account for the largest part of the energy consumed (both dynamic and leakage), and do not consider the scalar registers used in the additional RISC processing core.

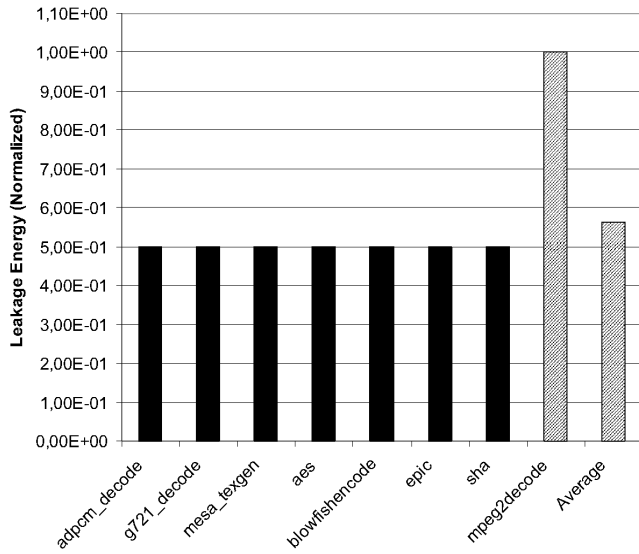
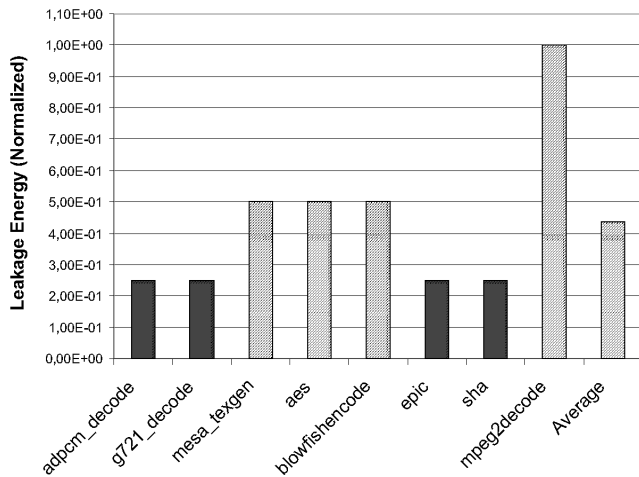
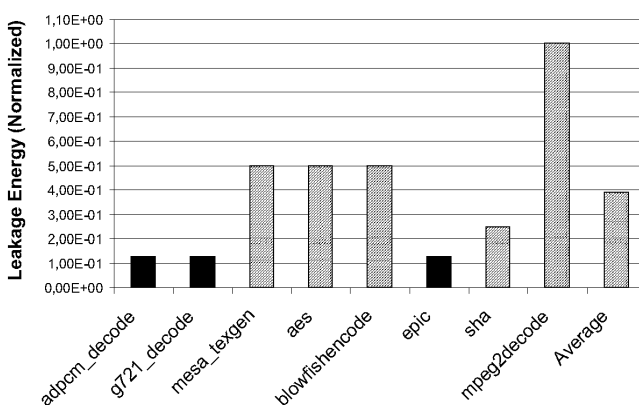
The CRISP framework detailed in Section 3 was used to simulate a VLIW processor. The processor chosen for our simulations was a 32-bit, 4 issue VLIW processor, trying to represent the forthcoming tendencies in commercial VLIW platforms for embedded multimedia applications. The register file was considered to have 12 ports (8 read and 4 write), 128 entries deep. Three cases were chosen for banking strategies: 8, 4 and 2 banks. The 90 nm leakage model proposed by Raghavan et al. (2005) was used for the different register file architectures considered.

When the benchmark compilation process has finished in the CRISP framework, the percentage of utilisation of the

register file can be obtained. Therefore, using this information obtained during the power-aware register assignment the resulting energy savings can be calculated for each of the banking strategies.

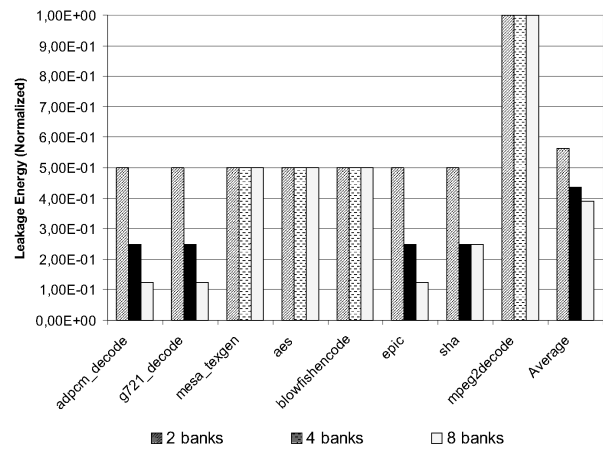
During our first set of experiments we considered the simplest possible partition with respect to extra hardware overhead and its complexity for the management of the register file, namely, it was divided only into 2 banks. The results obtained regarding leakage energy are depicted in Figure 5. The results shown for each benchmark using our proposed hardware/software approach have been normalised to the baseline architecture considered where all the banks of the register file are switched on during the complete execution of the programme. First of all, we can observe that even with such a simple two-banked register file, we can actually reduce the leakage energy in all benchmarks (except *mpeg2decode*) by 50% as one bank can be turned to low power mode roughly during the whole execution. In the case of *mpeg2decode*, after a careful analysis of the compilation and simulation results, we could verify that this particular application puts an extremely high pressure in the register file. Hence, all the registers are indeed used to the fullest extent during the whole execution and no bank could be turned off.

Then, during the second set of experiments we explored the effect of using a more complex partitioning of the register file by considering the options of 4-banked and 8-banked register files. The results obtained for our case studies are shown in Figures 6 and 7 respectively. It can be seen that, as the number of banks defined for the register file increases, the gains improve as well in most of the studied embedded applications, since they do not use a large number of simultaneous registers. Thus, more banks can be turned to low power mode. In addition, after a careful study of the results, we can observe that the type of benchmarks that benefit more from the 8-banked partitioning of the register file are the embedded multimedia applications (e.g., *mesa\_texgen*, *g721\_decode*, etc.). Moreover, *adpcm\_decode* shows that even though it requires a significant amount of main memory to process the incoming audio (in our experiments up to 1 MB), it puts very little register pressure. Hence, it enormously benefits from the increase in the number of banks considered for the register file. Conversely, the benchmarks from the cryptography domain (e.g., *blowfishencode*, *sha*, etc.) utilise more registers concurrently and do not show any improvement beyond partitioning the register file in 4 banks (see Figure 6). In the case of the *mpeg2decode* benchmark, the pressure it generates in the register file does not enable any benefit of partitioning the register file in any number of banks. Thus, there is no gain in leakage energy for any of the studied configurations.

**Figure 5** Leakage energy on MediaBench benchmarks with 2 banks**Figure 6** Leakage energy on MediaBench benchmarks with 4 banks**Figure 7** Leakage energy on MediaBench benchmarks with 8 banks

Finally, the previously described trends of benefits achieved by bank-partitioning in the total energy consumed of the register file for both types of application domains are

illustrated in Figure 8. It shows, as has been already mentioned, that the 8-banked configuration saves more energy than the 2-banked and 4-banked options since it keeps a larger area of the register file in the low-power state. On the other hand, this configuration presents more overhead of extra logic needed to turn the banks into the low power state. However, this extra logic has been found to be quite simple. As a result, the gains achieved due to the low-power state at run-time of a very significant part of the register file overcome the energy overheads of the extra logic. Therefore, most of the studied benchmarks still benefit from partitioning the register file in eight banks.

**Figure 8** Leakage energy savings for the three configurations

## 7 Conclusions

New consumer applications have recently increased in complexity and demand a very high level of performance in the next generation of low-power embedded devices. Therefore, new techniques and mechanisms that can provide solutions for an efficient mapping of these complex applications in such platforms are in great need. One of the most important factors of power consumption and performance penalty is the shared register file between all processing units. In this paper we have presented and shown with realistic examples the applicability of a new set of architectural extensions to enable the use of sub-banks in the register file, which achieves important reductions in the energy of the shared register file in upcoming embedded architectures with several VLIW processors. Our results indicate that this new integral approach enables on average a 60% reduction of the energy consumed in the register file of such forthcoming embedded architectures when they run real-life embedded multimedia, wireless network and cryptography applications without introducing performance penalties.

## Acknowledgements

This work is partially supported by the Spanish Government Research Grant TEC2006-00739 and TIN2005-05619.



## References

- Abella, J. and Gonzalez, A. (2003) 'On reducing register file pressure and energy in multiple-banked register files', *Proceedings of ICCD*, San Jose, USA, pp.14–20.
- Akturan, C. and Jacome, M.F. (2000) 'FDRA: a software-pipelining algorithm for embedded VLIW processors', *Proceedings of ISSS*, Madrid, Spain, pp.34–40.
- Akturan, C. and Jacome, M.F. (2001) 'Caliber: a software pipelining algorithm for clustered embedded VLIW processors', *Proceedings of ICCAD*, San Jose, USA, pp.112–118.
- Altera (2001) *Nios Embedded Processor System Development*, <http://www.altera.com>.
- Atasu, K., Pozzi, L. and Ienne, P. (2003) 'Automatic application-specific instruction-set extensions under microarchitectural constraints', *Proceedings of Design Automation Conference (DAC)*, Anaheim, USA, pp.256–261.
- Ayala, J.L. and López-Vallejo, M. (2004) 'Improving register file banking with a power-aware unroller', *Proceedings of PARC*, Pisa, Italy, pp.15–20.
- Ayala, J.L., López-Vallejo, M. and Veidenbaum, A. (2003) 'Energy-efficient register renaming in high-performance processors', *Proceedings of WASP*, San Diego, USA, pp.56–61.
- Benini, L. and de Micheli, G. (2000) 'System level power optimization techniques and tools', *ACM Transactions on Design Automation for Embedded Systems (TODAES)*, Vol. 5, pp.115–192.
- Benini, L., Bruni, D., Chinosi, M., Silvano, C., Zaccaria, V. and Zafalon, R. (2001) 'A power modeling and estimation framework for VLIW-based embedded systems', *Proceedings of PATMOS*, Yverdon Les Bains, Switzerland, pp.2.1.1–2.1.10.
- Biswas, P., Choudhary, V., Atasu, K., Pozzi, L., Ienne, P. and Dutt, N. (2004) 'Introduction of local memory elements in instruction set extensions', *Proceedings of DAC*, San Diego, USA, pp.729–734.
- Bose, P., Brooks, D., Uktosuno, A., Cook, G., Das, K., Emma, P., Gschwind, M., Jacobson, H., Karkhanis, T., Kudva, P., Schuster, S., Smith, J., Srinivasan, V., Zyuban, V., Albonesi, D. and Dwarkadas, S. (2002) 'Early-stage definition of LPX: a low power issue-execute processor', *Proc. PACS'02*, held in conjunction with HPCA, Cambridge, MA, pp.257–268.
- Chang, N., Kim, K. and Lee, H.G. (2000) 'Cycle-accurate energy consumption measurement and analysis: case study of arm7tdmi', *Proceedings of the 2000 International Symposium on Low Power Electronics and Design*, Rapallo, Italy, ACM Press, NY, USA, pp.185–190.
- Cruz, J.L., Gonzalez, A. and Valero, M. (2000) 'Multiple-banked register file architectures', *Proceedings of ISCA*, Vancouver, Canada, pp.316–325.
- Faraboschi, P., Brown, G., Fisher, J.A., Desoli, G. and Homewood, F. (2000) 'Lx: a technology platform for customizable VLIW embedded processing', *Proceedings of ISCA*, Vancouver, Canada, pp.203–213.
- Flautner, K., Kim, N.S., Martin, S., Blaauw, D. and Mudge, T.N. (2002) 'Drowsy caches: simple techniques for reducing leakage power', *Proceedings of ISCA*, Alaska, USA, pp.148–157.
- Glokler, T. and Meyr, H. (2002) *Design of Energy-efficient Application-specific Instruction Set Processors*, Kluwer Academic Publishers, P.O. Box 322, 3300 AH Dordrecht, The Netherlands.
- Gonzalez, R. (2002) 'Xtensa: a configurable and extensible processor', *IEEE Micro*, Vol. 20, No. 2, pp.60–70.
- Grun, P., Dutt, N. and Nicolau, A. (2001) 'Access pattern based local memory customization for low power embedded systems', *Proceedings of DATE 2001*, Piscataway, NJ, USA, pp.778–784.
- Hanson, H., Hrishikesh, M., Agarwal, V., Keckler, S.W. and Burger, D. (2001) 'Static energy reduction techniques for microprocessor caches', *International Conference on Computer Design*, Austin, USA, pp.303–313.
- Huang, J. and Leng, T. (1999) 'Generalized loop unrolling: a method for program speed-up', *Proceedings of the ASSET*, Richardson, USA, pp.244–248.
- Kim, N.S., Austin, T., Blaauw, D., Mudge, T., Flautner, K., Hu, J., Irwin, M., Kandemir, M. and Vijaykrishnan, N. (2003) 'Leakage current: Moore's law meets static power', *Computer*, Vol. 36, No. 12, pp.65–77.
- Koen, J.P., Langendoen, K. and Sips, H.J. (2003) 'Application-directed voltage scaling', *IEEE Transactions on Very Large Scale Integration (TVLSI)*, Vol. 11, No. 5, pp.812–826.
- Lambrechts, A., Raghavan, P., Leroy, A., Talavera, G., Vander, T., Jayapala, M., Catthoor, F., Verkest, D., Deconinck, G., Corporaal, H., Robert, F. and Carrabina, J. (2005) 'Power breakdown analysis for a heterogeneous NoC platform running a video application', *Proceedings of ASAP*, Samos, Greece, pp.179–184.
- Lee, C., Potkonjak, M. and Mangione-Smith, W.H. (1997) 'Mediabench: a tool for evaluating and synthesizing multimedia and communications systems', *Proceedings of International Symposium on Microarchitecture (MICRO)*, San Francisco, USA, pp.330–335.
- Luz, V.D.L., Kandemir, M. and Kolcu, I. (2002) 'Automatic data migration for reducing energy consumption in multi-bank memory systems', *Proceedings of DAC*, New York, USA, pp.213–218.
- Op de Beeck, P., Barat, F., Jayapala, M. and Lauwereins, R. (2001) 'Crisp: a template for reconfigurable instruction set processors', *FPL*, Belfast, Ireland, pp.296–305.
- Panda, P.R., Catthoor, F., Dutt, N.D., Danckaert, K., Brockmeyer, E. and Kulkarni, C. (2001) 'Data and memory optimizations for embedded systems', *ACM Transactions on Design Automation for Embedded Systems (TODAES)*, Vol. 6, No. 2, pp.142–206.
- Park, I., Powell, M.D. and Vijaykumar, T.N. (2002) 'Reducing register ports for higher speed and lower energy', *Proceedings of MICRO*, Istanbul, Turkey, pp.171–182.
- Philips Electronics (2004) *Philips Nexperia – Highly Integrated Programmable System-on-chip (MPSoC)*, <http://www.semiconductors.philips.com>.
- Philips PDSL (2005) *CoolFlux DSP*, <http://www.coolfluxdsp.com>.
- Raghavan, P., Lambrechts, A., Jayapala, M., Catthoor, F. and Verkest, D. (2005) 'Empirical power model for register files', *Workshop on Media and Streaming Processors (with MICRO-38)*, Barcelona, Spain.

- Saghir, M.A.R., Chow, P. and Lee, C.G. (1996) 'Exploiting dual data-memory banks in digital signal processors', *Proceedings of ASPLOS-VII*, ACM Press, New York, NY, USA, pp.234–243.
- Seng, J.S. and Tullsen, D.M. (2003) 'The effect of compiler optimizations on Pentium 4 power consumption', *Proceedings of INTERACT*, Anaheim, USA, pp.51–56.
- Seznec, A., Toullec, E. and Rochecouste, O. (2002) 'Reducing register ports for higher speed and lower energy', *Proceedings of MICRO*, Istanbul, Turkey, pp.383–394.
- ST Microelectronics (2004) *ST Nomadik Multimedia Processor*, <http://www.st.com>.
- Texas Instruments (2001) *TMS320C64x Programmer's Guide*, Texas Instruments, Canada.
- Texas Instruments (2004) *TI's OMAP Platform*, <http://focus.ti.com/omap/docs/>.
- Trimedia Technologies Inc. (1999) *Trimaran: An Infrastructure for Research in Instruction-level Parallelism*, <http://www.trimaran.org>.
- Vijaykrishnan, N., Kandemir, M., Irwin, M., Kim, H. and Ye, W. (2000) 'Energy-driven integrated hardware-software optimizations using SimplePower', *Proceedings of ISCA*, Vancouver, Canada, pp.95–106.
- Viredaz, M. and Wallacha, D. (2003) 'Power evaluation of a handheld computer', *IEEE Micro*, Vol. 23, No. 1, pp.66–74.
- Wolf, W. (2004) 'The future of multiprocessor systems-on-chips', *Proceedings DAC*, San Diego, USA, pp.681–685.
- Yu, P. and Mitra, T. (2004a) 'Characterizing embedded applications for instruction set extensible processors', *Proceedings of DAC*, San Diego, USA, pp.723–728.
- Yu, P. and Mitra, T. (2004b) 'Scalable custom instructions identification for instruction-set extensible processors', *Proceedings of CASES*, New York, NY, USA, pp.69–78.
- Zyuban, V.V. and Kogge, P.M. (1998) 'The energy complexity of register files', *Proceedings of ISLPED*, Monterey, USA, pp.305–310.
- Zyuban, V.V. and Kogge, P.M. (2001) 'Inherently lower-power high-performance superscalar architectures', *IEEE Transactions on Computers*, Vol. 50, No. 3, pp.268–285.