J Sign Process Syst (2011) 62:373–382
DOI 10.1007/s11265-010-0470-0

Parallel Architecture Core (PAC)—the First Multicore Application Processor SoC in Taiwan Part I: Hardware Architecture & Software Development Tools

David Chih-Wei Chang · Tay-Jyi Lin · Chung-Ju Wu · Jenq-Kuen Lee · Yuan-Hua Chu · An-Yeu Wu

Received: 9 November 2009 / Revised: 3 March 2010 / Accepted: 3 March 2010 / Published online: 20 April 2010

Abstract In order to develop a low-power and high-performance SoC platform for multimedia applications, the Parallel Architecture Core (PAC) project was initiated in Taiwan in 2003. A VLIW digital signal processor (PACDSP) has been developed from a proprietary instruction set with multimedia-rich instructions, a complexity-effective micro-architecture with an innovative distributed & ping-pong register organization and variable-length VLIW encoding, to a highly-configurable soft IP with several successful silicon implementations. A complete toolchain with an optimizing C compiler has also been developed for PACDSP. A dual-core PAC SoC has been designed and fabricated, which consists of a PACDSP core, an ARM9 core, scratchpad memories, and various on-chip peripherals, to demonstrate the outstanding performance and energy efficiency for multimedia processing such as the real-time H.264 codec. The first part of the two introductory papers of PAC describes the hardware architecture of the PACDSP core, its software development tools, and the PAC SoC with dynamic voltage and frequency scaling (DVFS).

Keywords Parallel Architecture Core · PACDSP · VLIW DSP · VLIW compiler · DVFS · Heterogeneous multicore · Application processor

1 Introduction

The design complexity of today's consumer electronics with more integrated functionalities and components is dramatically growing. However, their time to market and time in market (i.e. product lifetime) are getting much shorter. Therefore, the design productivity becomes a vital factor for the product success. Software programmability effectively raises the design productivity, which simplifies portability across platforms and allows late modifications. Moreover, multimedia systems prefer software implementations over hardwired approaches, for they require much more flexibility than ever, even after the product delivery to end customers. For example, today's audio players need to support multiple codecs such as MP3, WMA & AAC, and software-based products can be upgraded as the algorithms evolve or when new standards are introduced.

The PAC project [1–3] was initiated in 2003 and executed by Industrial Technology Research Institute (ITRI) [4] in Taiwan, of which the target is to provide a fully-programmable solution for next-generation media-rich and multi-function portable devices, such as portable media

D. C.-W. Chang
MediaTek Inc.,
Hsinchu, Taiwan

T.-J. Lin (✉)
National Chiao Tung University,
Hsinchu, Taiwan
e-mail: tjlin@twins.ee.nctu.edu.tw

C.-J. Wu · J.-K. Lee
National Tsing Hua University,
Hsinchu, Taiwan

Y.-H. Chu
Industrial Technology Research Institute,
Hsinchu, Taiwan

A.-Y. Wu
National Taiwan University,
Taipei, Taiwan

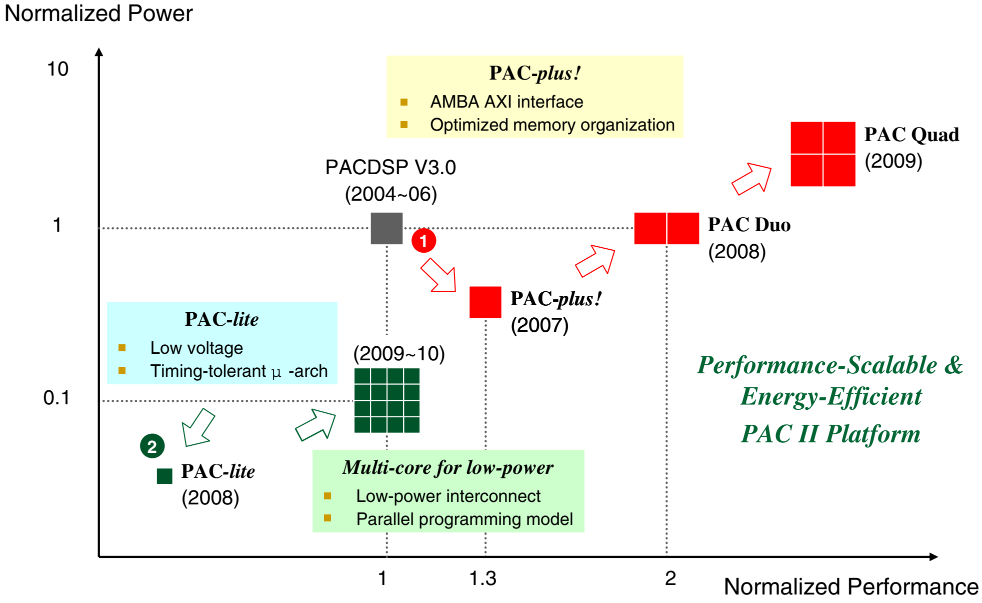
players and smart phones. The speech, audio, image and video processing on these devices demand extremely high computing power under tight power constraints. Digital signal processors (DSP) are programmable processors customized for signal processing, of which the instruction set and the microarchitecture are designed to perform computation-intensive tasks efficiently [5, 6]. Since their inception in the late 1970s, DSPs have been used into various domains, such as multimedia processing, communications, and industrial control. Very long instruction word (VLIW) architectures are the mainstreams of high-performance and energy-efficient designs [7], but they have serious drawbacks both on their register complexity and code density. A VLIW DSP (PACDSP) has been developed in the PAC project [3], of which the innovative distributed & ping-pong register file and variable-length VLIW encoding overcome the VLIW problems. A dual-core PAC SoC, which is composed of a PACDSP core, an ARM9 core and various on-chip peripherals, has been designed with dynamic voltage and frequency scaling (DVFS) capability. The SoC has been fabricated in the TSMC 0.13 μm CMOS technology to demonstrate its outstanding performance and energy efficiency for programmable multimedia processing. Complete software development tools for PACDSP have also been constructed, including an optimizing C compiler for the distributed & ping-pong register organization. Figure 1 shows the our project roadmap, where PACDSP V3 is the latest instruction set architecture and PAC-*plus!* is the latest V3-based core with the AMBA3 AXI interface and enhanced low-power features. The ongoing research and development works include the next-generation PACDSP instruction set & microarchitecture designs for ultra low-power and energy-efficient applications and multi-PACDSP architectures (i.e. PAC Duo & PAC Quad with two & four PACDSP cores respectively).

The rest of this paper is organized as follows. Section 2 reviews the microarchitecture design and the silicon implementation of the PACDSP core. Performance comparison with other licensable DSP cores is available in this section. Section 3 describes the software development tools for PACDSP, with focus on the optimizing compiler for the distributed & ping-pong register file, which is unique in the PACDSP core. Section 4 summarizes the PAC SoC platform and its silicon implementation, PAC Solo, with one PACDSP core and one ARM9 core respectively. The power modes of PAC Solo and an energy-aware H.264 decoder based on DVFS are also addressed. Finally, Section 5 concludes this paper.

2 PACDSP Core

PACDSP is a 32-bit VLIW DSP core, which has been developed from a proprietary instruction set, a low-power and complexity-effective microarchitecture, to a highly-configurable soft IP with several silicon implementations. It combines high-performance and low-power signal processing capability of ASIC and flexibility of microprocessors. PACDSP features scalable datapath for easy adaptation to different applications, an innovative distributed & ping-pong register organization, a rich & optimized instruction set with 8-bit/16-bit SIMD operations, and a high-performance memory subsystem. Figure 2(a) shows the microarchitecture, including a program sequencer, a scalar unit and clustered DSP datapath. Each DSP cluster has a

Figure 1 PAC roadmap.



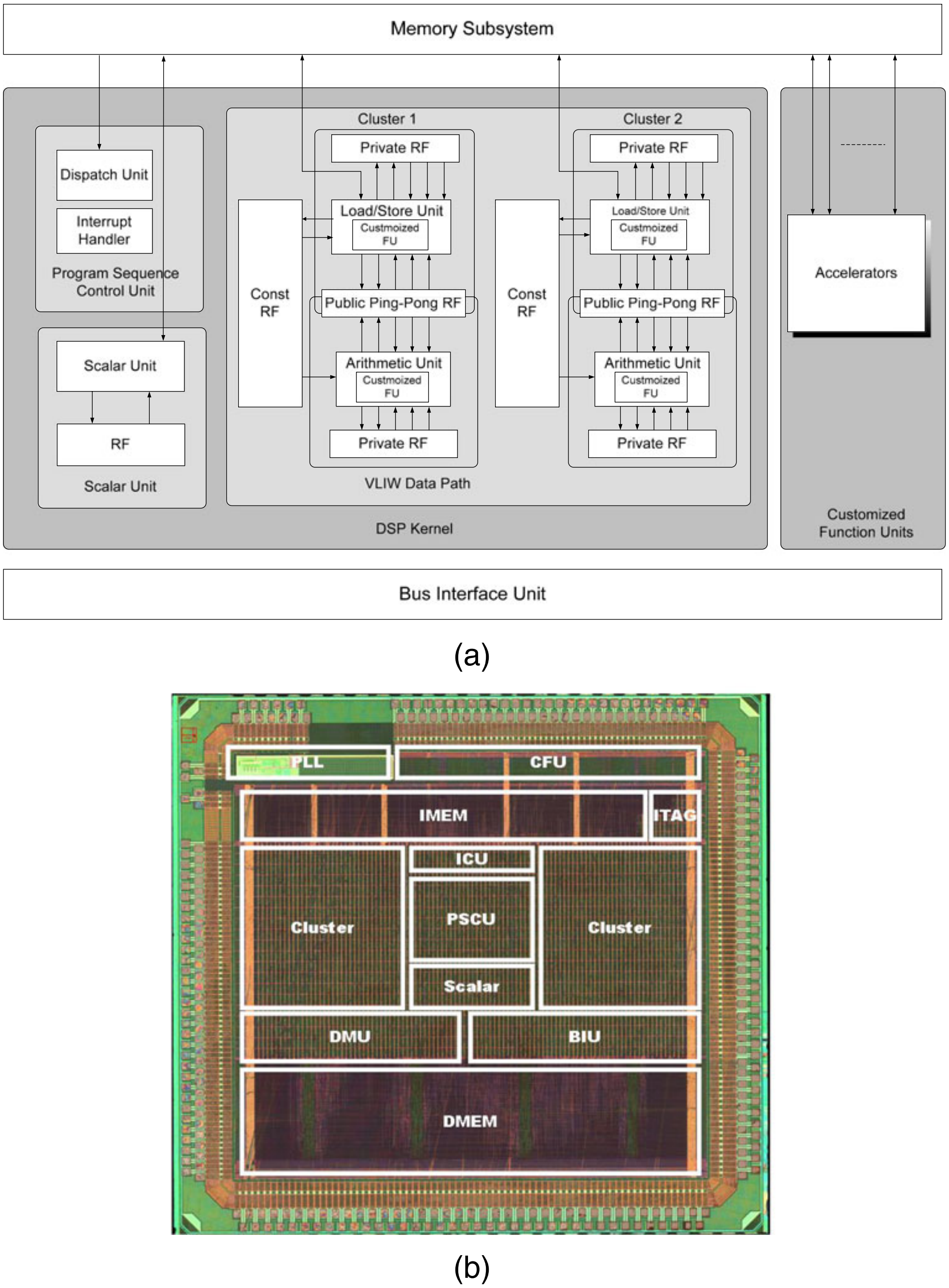


Figure 2 **a** Microarchitecture and **b** die photo of PACDSP.

Table 1 BDTIsimMark2000 scores for licensable cores [10].

Vendor	Processor	Score/MHz
ARM	ARM9E	2.1
Verisilicon	ZSP400	4.7
	ZSP500	7.9
Ceva	TeakLite-III	6.4
	Ceva-X 1620	8.1
ITRI/STC	PACDSP V3	8.8

distributed & ping-pong register file, which supports high-bandwidth data operands to and from the parallel functional units. Compared with an equivalent centralized register file used in state-of-the-art high-performance VLIW processors, the distributed & ping-pong register file reduces 76.8% silicon area and shortens 46.9% access times [3, 8, 9].

Table 1 summarizes the performance comparison of licensable cores for digital signal processing [10]. ARM9E is a multimedia-enhanced single-issue RISC core with rich DSP instructions that exploit subword parallelism. ZSP400 and ZSP500 were developed by LSI Logic and then

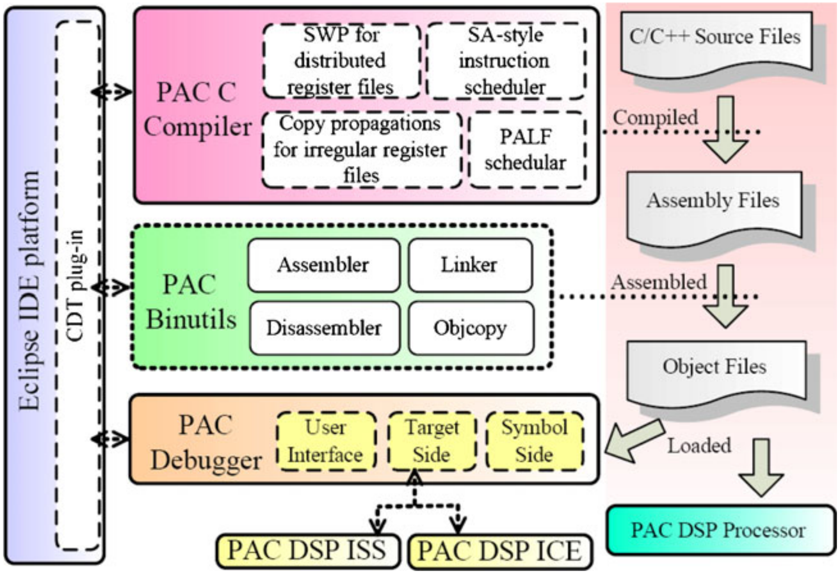


Figure 3 Toolchain for PACDSP [14].

acquired by Verisilicon, which use a 4-issue superscalar architecture that issues and completes up to four RISC-like instructions per cycle. Ceva TeakLite is a conventional MAC-based DSP for audio processing, while Ceva-X is an 8-issue VLIW DSP for high-performance video processing. Among them, ZSP500, Ceva-X 1620, and PACDSP V3 have similar computing resources. However, PACDSP has a low-complexity register file instead of a centralized one in the other two DSP cores, and it still has the outstanding performance for its customized instructions and optimized program flow mechanisms. By the way, PACDSP has very high code density through its variable-length operation encoding, NOP removal, and embedded code replication techniques [3, 11, 12]. The program sequencer dynamically aligns the VLIW packets with different numbers of operations, each of which is itself variable-length encoded.

The PACDSP microarchitecture is modular and highly configurable. For example, PACDSP can be simply integrated into an existing SoC platform with AMBA3 AXI, AMBA2 AHB, or simple FIFO interfaces with a corresponding bus interface unit. PACDSP has an internal power management unit to shut down each of the parallel clusters, the memory blocks and the cache lines individually with an appropriate power mode setting to reduce

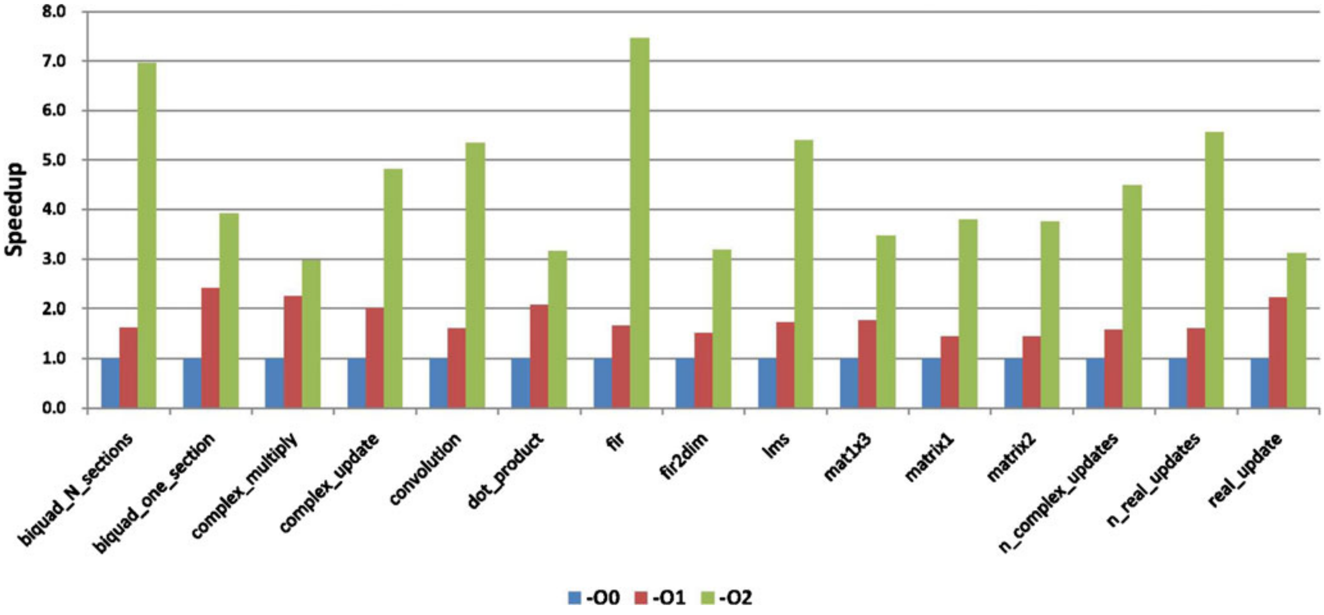
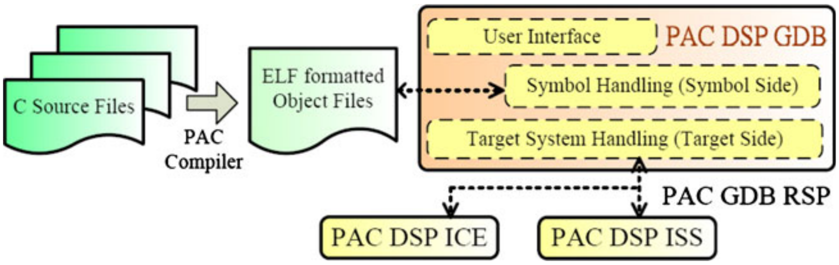


Figure 4 Performance of the PACDSP compiler.

Figure 5 PACDSP GDB.

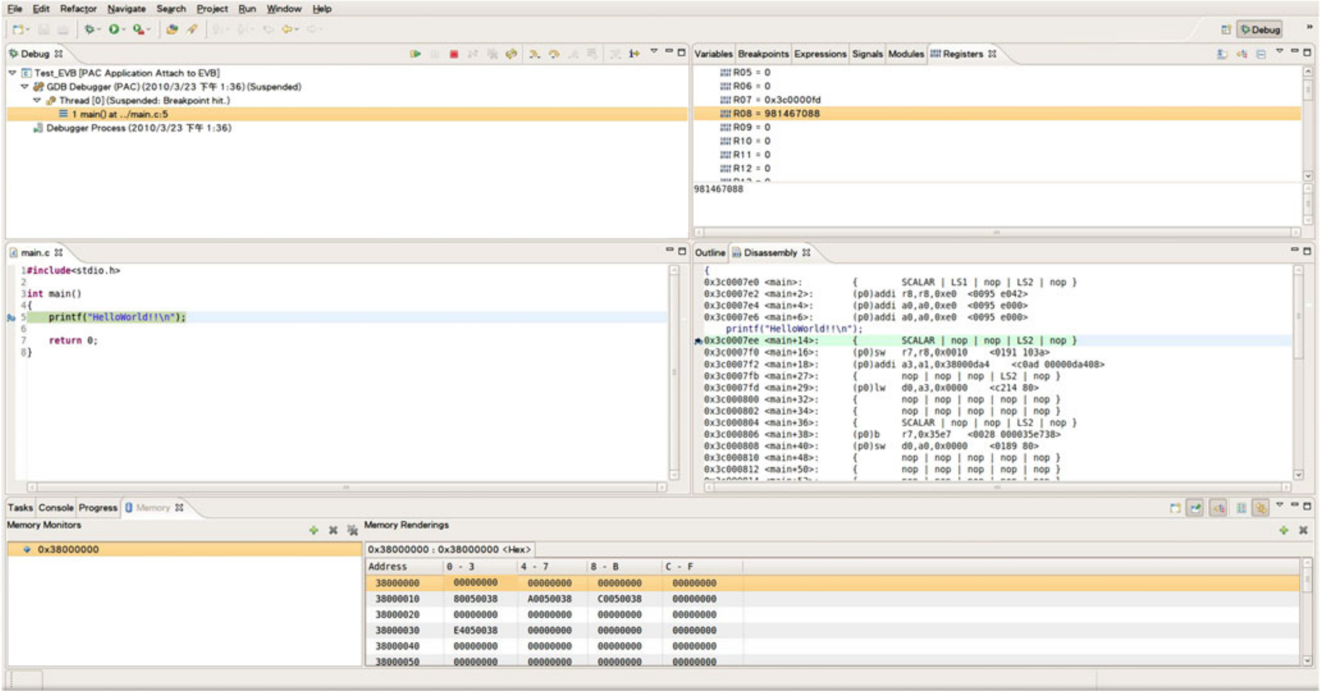

C Source Files
PAC Compiler
ELF formatted Object Files
User Interface
PAC DSP GDB
Symbol Handling (Symbol Side)
Target System Handling (Target Side)
PAC GDB RSP
PAC DSP ICE
PAC DSP ISS

leakage power. A PACDSP testchip has been implemented with a 32 KB direct-mapped instruction cache and a 64 KB data memory and the gate count is 250 K. It has been fabricated in the TSMC 130 nm generic process and the die photo is shown in Fig. 2(b). The maximum operating frequency is 300 MHz and the average power consumption is only 0.08 mW/MIPS.

3 Software Development Tools

As described in the previous section, PACDSP has outstanding performance and energy efficiency owing to its optimized microarchitecture for streaming multimedia applications (i.e. clustered/distributed & ping-pong register file and hierarchical VLIW encoding), but they significantly complicate the application programming. This section summarizes the challenges and our approaches in the toolchain development, including an optimizing compiler, binutils, a debugger, and an Eclipse-based IDE environment as depicted in Fig. 3 [13].

The PACDSP compiler [14, 15] is built based on the open research compiler (ORC) [16], an open source compiler infrastructure that contains various optimizations for high-performance EPIC/VLIW architectures. The ORC frontend helps to generate the intermediate representation, WHIRL, with five representation levels from "very high" to "very low", where various target-independent optimizations are performed, such as control flow optimization, extended basic block (peephole) optimization, integrated global/local scheduling, and loop transformation at the "very low" level. We have developed specific optimization techniques in the back-end for PACDSP, including copy propagation for irregular register files [17], optimal local register file assignment based on simulation annealing (SA-LRFA) [18], ping-pong aware & local favorable register file assignment (PALF-LRFA) [19], and local-conscious &


File Edit Refactor Navigate Search Project Run Window Help
Debug
Debug
Variables Breakpoints Expressions Signals Modules Registers
main.c
Outline Disassembly
Tasks Console Progress Memory
Memory Monitors
Memory Renderings

Test_EVB [PAC Application Attach to EVB]
GDB Debugger (PAC) (2010/3/23 下午 1:36) (Suspended)
Thread [0] (Suspended: Breakpoint hit.)
1 main() at ../main.c:5
Debugger Process (2010/3/23 下午 1:36)

```
R05 = 0
R06 = 0
R07 = 0x3c0000fd
R08 = 981467088
R09 = 0
R10 = 0
R11 = 0
R12 = 0
981467088
```

```c
#include<stdio.h>

int main()
{
    printf("HelloWorld!!\n");

    return 0;
}
```

```
{
0x3c0007e0 <main>:        {         SCALAR | LS1 | nop | LS2 | nop }
0x3c0007e2 <main+2>:      (p0)addi r8,r8,0xe0  <0095 e042>
0x3c0007e4 <main+4>:      (p0)addi a0,a0,0xe0  <0095 e000>
0x3c0007e6 <main+6>:      (p0)addi a0,a0,0xe0  <0095 e000>
    printf("HelloWorld!!\n");
0x3c0007ee <main+14>:     {         SCALAR | nop | nop | LS2 | nop }
0x3c0007f0 <main+16>:     (p0)sw   r7,r8,0x0010    <0191 103a>
0x3c0007f2 <main+18>:     (p0)addi a3,a1,0x38000da4    <c0ad 00000da408>
0x3c0007fb <main+27>:     {         nop | nop | nop | LS2 | nop }
0x3c0007fd <main+29>:     (p0)lw   d0,a3,0x0000    <c214 80>
0x3c000800 <main+32>:     {         nop | nop | nop | nop | nop }
0x3c000802 <main+34>:     {         nop | nop | nop | nop | nop }
0x3c000804 <main+36>:     {         SCALAR | nop | nop | LS2 | nop }
0x3c000806 <main+38>:     (p0)b    r7,0x35e7   <0028 000035e738>
0x3c000808 <main+40>:     (p0)sw   d0,a0,0x0000    <0189 80>
0x3c000810 <main+48>:     {         nop | nop | nop | nop | nop }
0x3c000812 <main+50>:     {         nop | nop | nop | nop | nop }
```

```
0x38000000

0x38000000 : 0x38000000 <Hex>
Address    0 - 3      4 - 7      8 - B      C - F
38000000   00000000   00000000   00000000   00000000
38000010   80050038   A0050038   C0050038   00000000
38000020   00000000   00000000   00000000   00000000
38000030   E4050038   00000000   00000000   00000000
38000040   00000000   00000000   00000000   00000000
38000050   00000000   00000000   00000000   00000000
```

Fig. 6 Eclipse-based IDE environment.

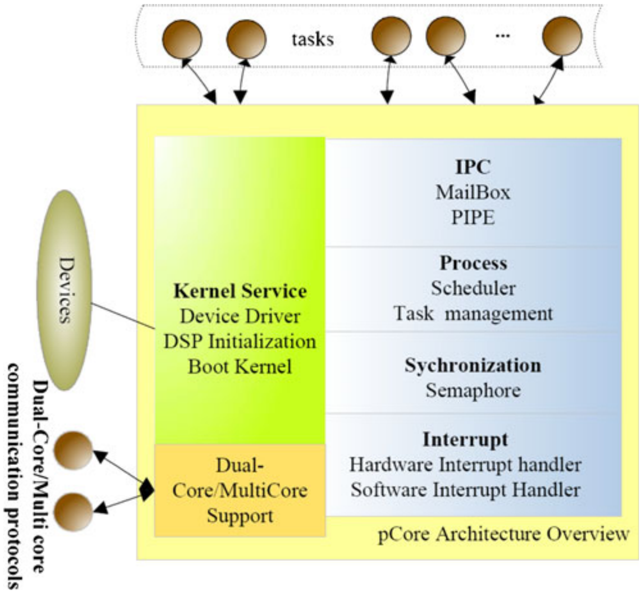

tasks
...
Devices
Dual-Core/Multi core communication protocols
Kernel Service
Device Driver
DSP Initialization
Boot Kernel
Dual-Core/MultiCore Support
IPC
MailBox
PIPE
Process
Scheduler
Task management
Sychronization
Semaphore
Interrupt
Hardware Interrupt handler
Software Interrupt Handler
pCore Architecture Overview

Figure 7 pCore kernel for PACDSP.

global register file assignment (LC-GRFA) [20], etc. LC-GRFA is the most important optimization, which minimizes data communication costs between various registers. It prioritizes the basic blocks by the static and dynamic scores, and assigns physical registers accordingly. Codes for intra-or inter-cluster communications are then inserted to complete the code generation. Figure 4 shows the compiler performance for DSPstone [21], where the execution cycles are normalized by those without optimizations (i.e. O0, shown in blue bars), and maximum 2.2 and 7.5 speedups can be observed by our proposed optimization techniques at level 1 and level 2 respectively (i.e. O1 and O2, shown in red and green bars).

The GNU debugger (GDB) [22] is adopted in the PACDSP toolchain for software debugging, which consists of three components: the target system handler, the symbol handler, and the user interface, as shown in Fig. 5. The PACDSP GDB first retrieves the debug information in its symbol handling component from the object files generated by the PACDSP compiler. It then uses the GDB remote serial protocol (RSP) to communicate with the PACDSP instruction set simulator (ISS) or the PACDSP chip through the in-circuit emulator (ICE) in its target system handling component. In addition to the protocols for accessing memory and register contents and setting breakpoints & watch-points, the PACDSP GDB defines various pseudo registers in RSP to handle system events, such as timers and interrupts. Finally, the PACDSP compiler, the PACDSP GDB, the PACDSP simulator and binutils are integrated in the Eclipse integrated development environment (IDE) [23] to make the application development more convenient as shown in Fig. 6.

Figure 7 shows a microkernel developed for PACDSP [24]—pCore, which provides lightweight OS services such as scheduling, synchronization, resource management, and inter-process communication (IPC), etc, with small memory footprint. The basic unit for resource management in pCore is a task, of which the runtime information is recorded in a task control block (TCB). pCore supports priority-based scheduling and message passing IPC, which facilitate heterogeneous multicore configuration for real-time multimedia processing. All

Figure 8 PAC Solo.

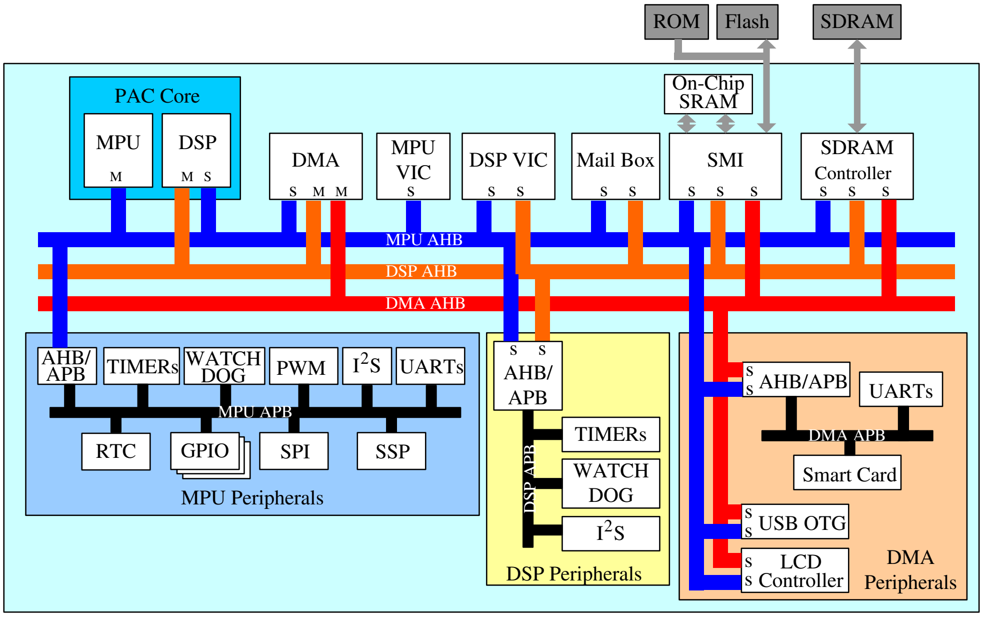


Table 2 Power domains of PAC Solo.

Power domain	Power mode	Condition		Power gating
		Supply (V)	Frequency (MHz)	
MPU	Active-1	1.2	228	N
	Active-2	1.2	152	
	Active-3	1.2	114	
	Inactive	1.2	0	
	Sleep	0	0	
DSP	Active-1	1.2	228	Y
	Active-2	1.0	152	
	Active-3	0.9	114	
	Inactive	0.9	0	
	Pending	0.9	0	
	Sleep	0	0	
AHB	Full speed	1.2	152/114	N
	Low power	1.0	76	
SRAM/LCD	Same as AHB-domain			Y
APB/PMU	Fixed V	1.0	48	N
PLL	Fixed V	1.2	456	N

pCore system services are customizable for specific system integrations.

4 PAC SoC Platform

In addition to PACDSP, a dual-core PAC SoC composed of an ARM9 core and a single PACDSP core (i.e. PAC Solo) has also been designed to demonstrate the outstanding performance and the energy efficiency for multimedia processing, such as real-time H.264 codec. A multi-layered AHB is implemented to connect the two processor cores, a system DMA, an on-chip SDRAM controller, vector interrupt controllers (VIC) and so on, as shown in Fig. 8. System software including the DSP driver and the DSP kernel for dual-core programming has been built with the inter-processor communications (IPC) through mailboxes. PAC Solo adopts various power optimization techniques to reduce both dynamic and leakage power dissipations. Common low-power techniques such as operand isolation and clock gating have been extensively applied in the silicon implementation. Most importantly, the SoC has been divided into six independent power domains for individual control on the power supplies and operating frequencies [25].

Table 2 lists the power domains and the power modes thereof. The MPU domain only has a constant supply voltage due to the lack of control over the ARM9 hard macro. On the other hand, the DSP domain has various operating frequencies and the corresponding minimum supply voltages. Programmers can adjust the performance and thus the power consumption according to the application requirements by setting the control registers in the DVFS controller, which record the current state, the next state and possible states of each power domain. It is of note that the DVFS control is application-dependent and specific control schemes need to be developed to adapt computation dynamically for the optimal power dissipation. For example, H.264/AVC supports various resolutions and profiles for different applications. In video conferencing applications, the bitstreams of low resolutions and low frame rates can be decoded at relatively low operating frequency on handheld devices, where the supply voltage can be lowered accordingly to reduce the power dissipation. On the contrary, the supply voltage should be raised to boost the operating frequency for

Table 3 Power dissipation of H.264 decoding in different power modes.

Power Mode	Condition	Power	Time	Energy (mJ)
Active-1	1.2 V/228 MHz	161.20 mW	10 sec	1612
Active-2	1.0 V/152 MHz	75.99 mW	13 sec	987.87 (**39%**)
Active-3	0.9 V/114 MHz	48.82 mW	17 sec	829.94 (**49%**)

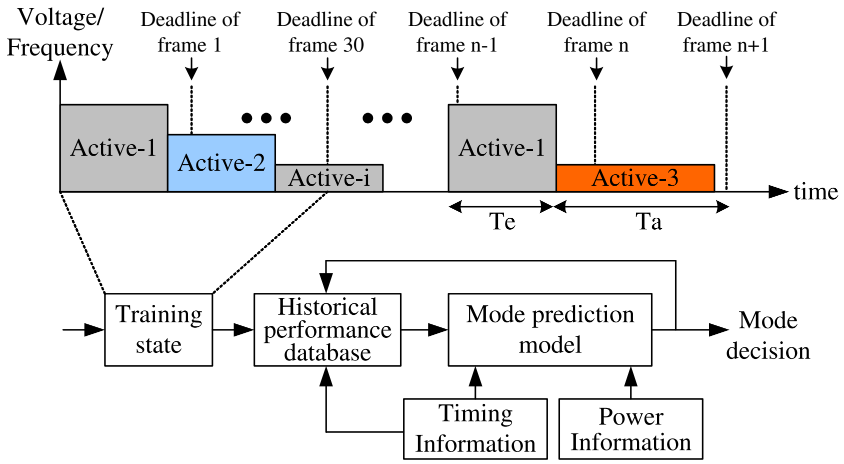


Figure 9 Energy-aware H.264 decoding.

decoding high-quality video on high-definition televisions. Moreover, the decoding complexity and the required computations vary significantly even in a video sequence. In the following, an energy-aware H.264/AVC decoder is briefly described, which utilizes the DVFS on PAC SoC frame by frame. Please refer to Part II of these two introductory papers for the detailed algorithm optimizations for DVFS and VLIW.

The operating frequency and the supply voltage of PACDSP are dynamically adjusted in our energy-aware H.264/AVC decoder, while ARM9 runs at the fixed 114 MHz clock rate. To understand the relationship between power dissipation and frequency/voltage, we first characterize the different power modes of PACDSP. The measured power and energy dissipations are summarized in Table 3, which were obtained by repeatedly decoding the same video sequence on real silicon. Active-2 and Active-3 modes reduce the energy dissipations by 39% and 49% respectively. Figure 9 shows the control scheme, where the DVFS controller dynamically adjusts the frequency/voltage depending on the input sequence. Te and Ta are the execution time of previous frame and the allowed execution time of current frame respectively. At the beginning, 30 frames are used to train the controller by establishing the performance database. The power mode will be determined hereafter with a prediction model based on timing and power information. The results of the same sequence (cars, 1888 frames) with two different frame rates are summarized in Table 4, where the higher frame rate kept PACDSP in the Active-1 mode longer, and 35% and 43% energy savings were observed over the non-DVFS counterpart respectively.

5 Conclusions

This paper introduces the first multicore application processor developed in Taiwan: PAC and describes its hardware architecture including the PACDSP core, the PAC SoC, and the software development tools. Based on the achievements, we are pursuing next-generation embedded computing platforms with even lower power dissipation, higher performance, and improved energy efficiency. We are now studying improved ISA (PAC-lite & PAC-SIMD) for ultra low-power applications and multi-PACDSP architectures (e.g. PAC Duo & PAC Quad with two & four PACDSP cores respectively). In addition to core developments, much more efforts will be put on platform technologies such as low-power and high-bandwidth on-chip network, optimized memory organization & DMA controllers, platform-optimized runtime software, and electronic system-level (ESL) design methodology. Once the overheads on software programming and the data communications are well controlled, these multicore architectures will be much more energy-efficient than pushing a single processor to an even

Table 4 Energy savings.

	Frame rate (fps)	Mode	Occurrence	Saved energy
Case 1	22	Active-1	359 (19%)	35%
		Active-2	874 (46%)	
		Active-3	655 (35%)	
Case 2	20	Active-1	82 (4%)	43%
		Active-2	688 (36%)	
		Active-3	1,118 (60%)	

higher operating frequency. Based on the rich software components already optimized for PACDSP and the expertise of software development on PAC Solo, we are developing a component-based software development methodology and its associated ESL tool. The tool allows drag & drop binding of software tasks developed on the single-core processor for multicore simulation with cycle-accurate transaction modeling. We are now developing an automatic task binder that considers system-level overheads such as cache misses and DMA accesses on the on-chip interconnect.

References

1. Lin, T. J., Liu, C. N., Tseng, S. Y., Chu, Y. H., & Wu, A. Y. (2008). Overview of ITRI PAC project—from VLIW DSP processor to multicore computing platform. In *Proc. VLSI-DAT*, Apr. 2008, pp.188–191.
2. Chang, C. W. et al. (2006). PACDSP core and application processors. In *Proc. ICME*, July 2006, pp.289–292.
3. Lin, T. J., Hsiao, P. C., Chen, S. K., Kuo, Y. T., & Liu, C. W. (2008). Design & implementation of a high-performance & complexity-effective VLIW DSP for multimedia applications. *Journal of Signal Processing Systems, 51*, 209–223.
4. http://www.itri.org.tw/
5. Lapsley, P., Bier, J., & Lee, E. A. (1996). *DSP Processor fundamentals—architectures and features*. IEEE Press.
6. Hu, Y. H. (2002). *Programmable digital signal processors—architecture, programming, and applications*. Marcel Dekker Inc.
7. Fisher, J. A., Faraboschi, P., & Young, C. (2005). *Embedded computing—A VLIW approach to architecture, compiler, and tools*. Morgan Kaufmann.
8. Lin, T. J., Hsiao, P. C., Liu, C. W., & Jen, C. W. (2006). Area-efficient register organization for fully-synthesizable VLIW DSP cores. *International Journal of Electrical Engineering, 13*, 117–127.
9. Lin, T. J., Lee, C. C., Liu, C. W., & Jen, C. W. (2005). A novel register organization for VLIW digital signal processors. In *Proc. VLSI-TSA-DAT*, Apr. 2005, pp.337–340.
10. http://www.bdti.com/bdtimark/core_scores.pdf
11. Lin, T. J. et al. (2005). A unified processor architecture for RISC & VLIW DSP. In *Proc. GLSVLSI*, Apr. 2005, pp.50–55.
12. Liu, C. H. et al. (2005). Hierarchical instruction encoding for VLIW digital signal processors. In *Proc. ISCAS*, May 2005, pp.3503–3506.
13. Wu, C. et al. (2006). Integrating compiler and system toolkit flow for embedded VLIW DSP processors. In *Proc. RTCSA*, Aug. 2006, pp.215–222.
14. Lin, Y. C. et al. (2005). Compiler supports and optimizations for PAC VLIW DSP processors. In *Proc. LCPC*, Oct. 2005, pp.466–474.
15. Chen, C. K. et al. (2007). Enabling compiler flow for embedded VLIW DSP processors with distributed register files. In *Proc. LCTES*, pp.146–148.
16. http://ipf-orc.sourceforge.net/
17. Wu, C. J., Chen, S. Y., & Lee, J. K. (2007). Copy propagation optimizations for VLIW DSP processors with distributed register files. *Languages and Compilers for Parallel Computing* (*LNCS 4382*), pp.251–266, Jun. 2007.
18. Lin, Y. C., et al. (2008). Effective code generation for distributed and ping-pong register files: a case study on PAC VLIW DSP cores. *Journal of Signal Processing Systems, 51*, 269–288.
19. Lin, Y. J., You, Y. P., & Lee, J. K. (2007). PALF: compiler supports for irregular register files in clustered VLIW DSP processors. *Concurrency and Computation: Practice and Experience, 19*, 2391–2406.
20. Lu, C. H., Lin, Y. J., You, Y. P., & Lee, J. K. (2009). LC-GRFA: global register file assignment with local consciousness for VLIW DSP processors with non-uniform register files. *Concurrency and Computation: Practice and Experience, 21*, 101–114.
21. Zivojnovic, V., Martinez, J., Schläger, C., & Meyr, H. (1994). DSPstone: a DSP-oriented benchmarking methodology. In *Proc. ICSPAT*, Oct. 1994.
22. http://www.gnu.org/software/gdb/
23. *Eclipse Platform Technical Overview*, available online: http://www.eclipse.org/articles/Whitepaper-Platform-3.1/eclipse-platform-white paper.pdf
24. Hsieh, K. Y., Lin, Y. C., Huang, C. C., & Lee, J. K. (2008). Enhancing microkernel performance on VLIW DSP processors via multiset context switch. *Journal of Signal Processing Systems, 51*, 257–268.
25. Lai, C. Y., Lin, J. H., & Wang, Y. F. (2005). DVFS SoC architecture and implementation. *SoC Technology Journal, 3*, 84–91.

David Chih-Wei Chang received his Ph.D. degree in Electrical Engineering from the University of Southern California. He was a part-time professor in the Electrical Engineering Department at the San Jose State University. From 1985 to 1988, he was a Design Engineer at Citicorp/Quotron, designing tightly-coupled multiprocessors. From 1988 to 1991, he was a Sr. Staff Engineer at Siemens/Pyramid Technology, where he worked on RISC-based multiprocessor designs. From 1991 to 1997, he was a Sr. Manager at Fujitsu/HAL Computer Systems, where he worked on SPARC 64-bit 4-way superscalar, speculative out-of-order execution microprocessors. From 1997 to 1999, he was the Director of Engineering of Desktop Graphics and Integration Graphics (with VIA, ALI) at Trident Microsystems. From 1999 to 2002, he was the Vice President of Engineering and was responsible for the VoIP Network Processors development at empowerTel Networks. From 2002 to 2003, he was the Vice President of Engineering at Arcadia Design Systems, where he worked on JPEG2000 image devices and ARM-based SOC platforms. From 2003 to 2007, he served as the Deputy General Director of SoC Technology Center (STC), Industrial Technology Research Institute and the Director of Semiconductor Industry Promotion Office/IDB, MOEA in Taiwan. He is now in charge of the wireless connectivity products (Bluetooth, WLAN, GPS, FM) at MediaTek. Dr. Chang received the Outstanding Electrical Engineering Award from the Chinese Institute of Electrical Engineering in 2005. He holds four patents and has published 30+ technical papers in VLSI/computer/networking areas. He is a Sr. Member of IEEE.

Tay-Jyi Lin received the B.S. degree in Electrical and Control Engineering and the Ph.D. degree in Electronics Engineering, from National Chiao Tung University, Taiwan in 1998 and 2005, respectively. He is now with the Microelectronics and Information Systems Research Center, National Chiao Tung University, as a researcher assistant professor. His research interests include VLSI signal processing, low-power design methodology, and computer architecture.

Chung-Ju Wu received the B.S. degree in Computer Information and Science from National Chiao Tung University, Taiwan in 2001 and the M.S. degree in Computer Science from National Tsing Hua University, Taiwan in 2003, where he is currently working towards the Ph.D. degree. His current researches include GCC compiler porting, Open64 compiler optimizations, and system software for embedded SOC environments.

Jenq-Kuen Lee received the Ph.D. degree in Computer Science from Indiana University in 1992, where he also received the M.S. degree (1991) in Computer science. He is now a professor in the Department of Computer Science at National Tsing-Hua University, Taiwan, where he joined the department in 1992. He was a key member of the team who developed the first version of the pC++ language and SIGMA system while at Indiana University. His current research interests include optimizing compilers, compilers for low-power and embedded systems, parallel object-oriented languages and systems, and computer architectures.

Yuan-Hua Chu received the B.S. degree in Electrophysics and the M.S. degree in Electronics Engineering, from National Chiao Tung University, Taiwan in 1981 and 1983, respectively. He is now with the Design Automation Technology Division, Information and Communications Research Laboratories (ICL), Industrial Technology Research Institute (ITRI), as a technical director. His research interests include low-power design methodology and design automation.

An-Yeu Wu received the B.S. degree from National Taiwan University in 1987, and the M.S. and Ph.D. degrees from the University of Maryland, College Park in 1992 and 1995, respectively, all in Electrical Engineering. From August 1995 to July 1996, he was a Member of Technical Staff (MTS) at AT&T Bell Laboratories, Murray Hill, NJ, working on high-speed transmission IC designs. From 1996 to July 2000, he was with the Electrical Engineering Department of National Central University, Taiwan. In August 2000, he joined the faculty of the Department of Electrical Engineering and the Graduate Institute of Electronics Engineering, National Taiwan University (NTU), where he is currently a Professor. From August 2007 to January 2010, he was on leave from NTU and served as the Deputy General Director of SoC Technology Center (STC), Industrial Technology Research Institute (ITRI), where he supervised PAC (Parallel Architecture Core) and Android/multicore-related projects.