A Reconfigurable Signal Processing IC with embedded FPGA and Multi-Port Flash Memory

M. Borgatti, L. Calì, G. De Sandre, B. Forêt, D. Iezzi, F. Lertora,
G. Muzzi, M. Pasotti, M. Poles and P.L. Rolandi

STMicroelectronics, Central R&D
Agrate Brianza, Italy
michele.borgatti@st.com

ABSTRACT
A 1GOPS dynamically reconfigurable processing unit with embedded Flash memory and SRAM-based FPGA targets image-voice processing and recognition applications. Code, data and FPGA bitstreams are stored in the embedded Flash memory and are independently accessible through 3 content-specific, 64-bit I/O ports with a peak read rate of 1.2GB/s. The system is implemented in a 0.18um, 2PL-6ML CMOS Flash technology, chip area is 70mm2.

Categories and Subject Descriptors
B.7.1 [**INTEGRATED CIRCUITS**]: Types and Design Styles – *Advanced technologies, Algorithms implemented in hardware, Gate arrays, Input/output circuits, Memory technologies, Microprocessors and microcomputers, VLSI.*

C.1.3 [**PROCESSOR ARCHITECTURES**]: Other Architecture Styles – *Adaptable architectures, Heterogeneous (hybrid) systems.*

B.3.1 [**MEMORY STRUCTURES**]: Semiconductor Memories.

C.3 [**SPECIAL-PURPOSE AND APPLICATION-BASED SYSTEMS**]: Signal processing systems.

General Terms
Design, Performance, Algorithms.

Keywords
Application-specific integrated circuits (ASICs), digital signal processors, field-programmable gate arrays (FPGAs), integrated circuit design, multimedia computing, reconfigurable architectures.

1. INTRODUCTION
Increasing complexity of system design and shorter time-to-market requirements are leading research towards the investigation of hybrid systems including processors enhanced by programmable logic [1][2]. Embedded programmable logic allows ASIC and ASSP vendors to broaden the appeal of their products. NRE reduction and shorter time-to-market are key to OEMs looking for faster turnaround and lower risk design solutions and technology. System integrators can also exploit hardware programmability for in-house product customization.

In this paper we present a pragmatic approach to introduce flexibility in system-chip design and exploit embedded programmable silicon fabrics to enhance system performances. In particular, enabling application-specific configurations to adapt the underlying hardware architecture to time-varying application demands can improve execution speed and reduce power consumption compared to a general-purpose programmable solution.

This paper describes a dynamically reconfigurable processing unit tightly connected to a Flash EEPROM memory subsystem. The reconfigurable processing unit targets image-voice processing and recognition application domains and is implemented by joining a configurable and extensible processor core and an SRAM-based embedded FPGA. Application-specific HW units are added and dynamically modified by embedded FPGA reconfiguration. By implementing application-specific vector processing instructions, the unit shows a peak computing power of 1GOPS. Efficient read-write-erase access to code, data and FPGA bitstreams is provided by a specific memory subsystem based on a modular 8Mb, 4-bank Flash memory. It features 3 content-specific I/O ports and delivers an aggregate peak read throughput of 1.2GB/s.

The proposed system has been built using a set of state-of-the-art IP cores and system design methodology. Design flows for system exploration and implementation are also described.

2. SYSTEM ARCHITECTURE
The system architecture is illustrated in Figure 1. The functional purposes of the embedded FPGA are: i) extension of the processor datapath supporting a set of additional special-purpose C-callable microprocessor instructions; ii) bus-mapped coprocessors (connected to the system bus through a master/slave

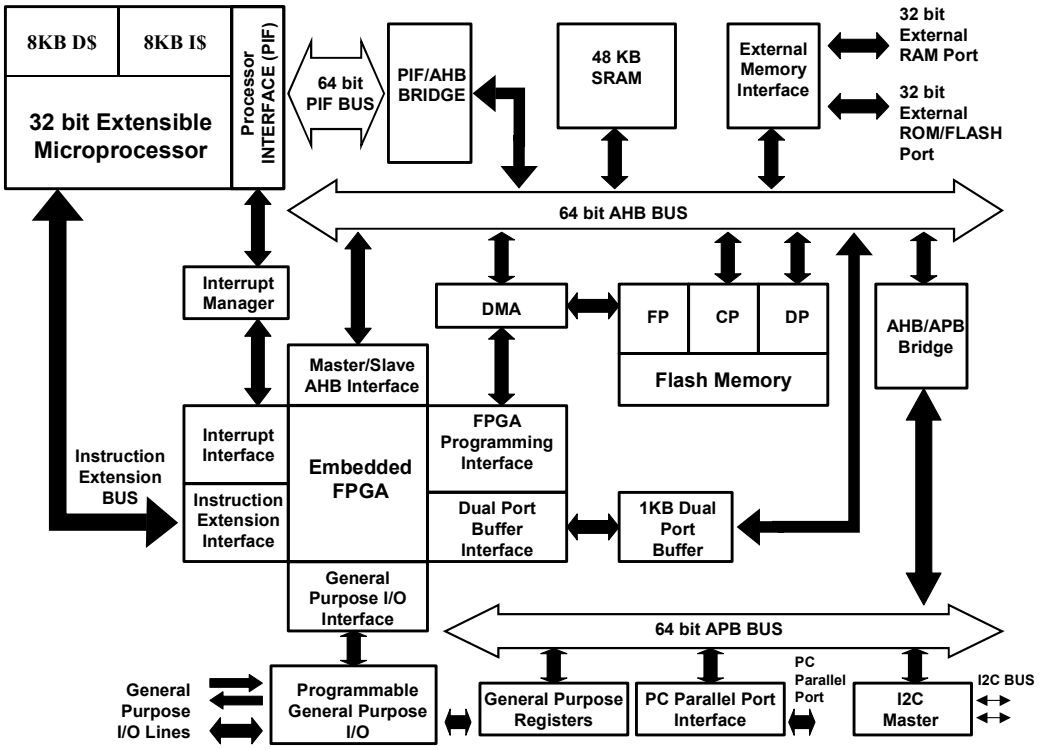

Figure 1. System Architecture.

interface); iii) flexible I/O (to connect external units or sensors featuring application-specific communication protocols).

Even though such different circuit purposes would require different kinds of programmable logic for best implementation of either arithmetic-dominated or control-dominated logic, we implemented a single programmable logic fabric to be shared among different purposes both in space (same configuration) and time (subsequent configurations). A single, high I/O count, fine-grain e-FPGA operates as a datapath for the microprocessor pipeline and as dedicated control logic for bus coprocessor and I/O control interface.

FPGA reconfiguration is concurrent to software execution. A local bus connects a dedicated 32-bit Flash memory port (FP) to the FPGA programming interface. A DMA channel handles the bitstream transfer while microprocessor fetches instructions and data from different Flash memory ports: 64-bit wide code port (CP) and data port (DP). To support streaming applications a 1kB dual-port buffer is used to interface fast decoding hardware and slower software running on the processor.

The memory sub-system architecture is shown in Figure 2. The modular memory (dotted line) includes charge pumps (Power Block), testability circuits (DFT), a power management arbiter (PMA) and a customizable array of N independent 2Mb flash memory modules, depending on the storage requirements (N=4 in the current implementation). The modular memory features (N+2) 128-bit target ports and implements a N-bank uniform memory.

An 8-bit microprocessor (uP) is devoted to handle complex file-system functions (defrag, compression, virtual erase, etc.) not natively supported by DP, and assists for built-in self test. A (N+2)x4 128-bit crossbar connects the modular memory with the four initiators (CP, DP, FP and uP) providing that three banks can be read in parallel at full speed. The memory space of the four modules is arranged in three programmable user-defined partitions, each one devoted to a port.

Each 2Mb flash memory module has a 128-bit IO data bus with 40ns access time, resulting in 400Mbyte/s, and a program/erase control unit. Simultaneous memory operations use the power management arbiter (PMA) for optimal scheduling. Available power and user-defined priorities are considered to schedule conflicting resource requests in a single clock cycle. The memory system allows up to four simultaneous operations (with a limit of one both for write and erase).

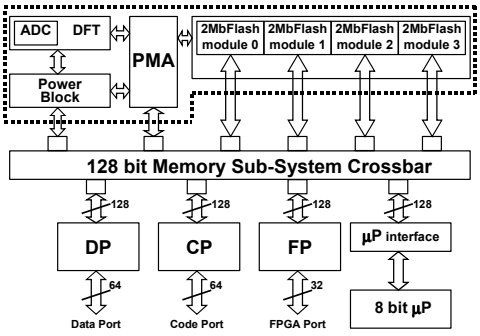

Figure 2. Flash Memory Architecture.

Figure 3 depicts the memory hierarchy and parallelism across the system. CP and DP are interfaced to the 64-bit, 800MB/s AHB system bus. At a system clock rate of 100MHz each I/O port can independently operate at maximum speed. So, an aggregate peak read rate of 1.2GB/s can be sustained as it is limited by memory access time. In the current implementation the e-FPGA reconfiguration takes 500us @ 100 MHz. 50MB/s average throughput out of the available 400MB/s are currently sustained by the e-FPGA configuration interface.

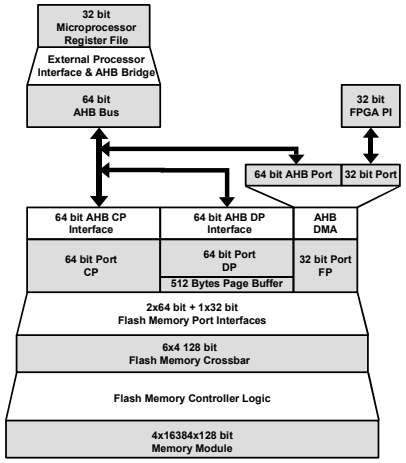

Figure 3. Memory Hierarchy.

System performance is evaluated for an image processing application (facial recognition) and a speech recognition application. More than 20 specific instructions were designed as C/assembly-callable functions, automatically translated to RTL, then synthesized and mapped to the e-FPGA. Figure 4 shows two examples of specific microprocessor extensions. On the right-hand side, an 8-issue, 8-bit, L2 calculation accounts for 23 8-bit arithmetic operations and 6 64-bit operations requiring about 10k ASIC equivalent gates. On the left-hand side, a datapath for an optimized fixed-point calculation of the square root accounts for 12 32-bit operations for about 2k ASIC equivalent gates.

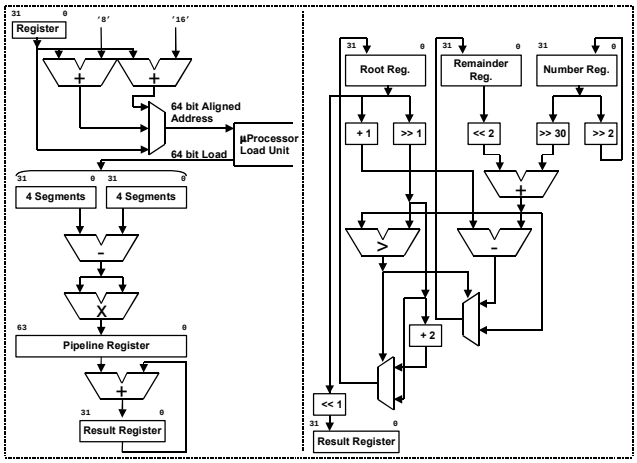

Figure 4. Added DSP instructions examples.

The overall performance improvements for the face recognition tasks are shown in Table 1. Execution time is compared for 32-bit RISC with basic DSP extensions (MAC, zero-overhead loops, etc) and the same processor enhanced with application-specific instructions. Measured speed-ups range from 1.8x to 10.6x (on the most-demanding task), with an overall improvement of 8.5x. Notice that switching between algorithm stages requires only one reconfiguration of the e-FPGA. Reconfiguration time is negligible. The speed-up factors take into account the possible multi-cycle clock penalty due to processor-FPGA synchronization in case of instruction extensions slower than the processor clock.

Energy efficiency figures are also depicted in Table 1. As the average power consumption of the system extended with the e-FPGA is slightly higher, the energy reduction for executing each of the tasks on its specific HW configuration (power-delay product improvement) results in an overall reduction of 6.7x. Only one task showed slightly worse total execution energy, though showing benefits on execution speed. Last column of Figure 5 reports the energy-delay improvement of each specific HW configuration compared to the general-purpose counterpart. Energy required for e-FPGA reconfiguration is always negligible. Measurements show the best energy efficiency in the range of several MOPS/mW at 1.8V supply. It lies between conventional ASIP/DSP and dedicated configurable hardware implementations [2].

Table 1. Benchmarks at 100MHz.

Algorit. Stage	RISC with basic DSP	RISC with uP extens.	Speed Up	Energy Reduct.	Energy Effic. Gain
Bayer Filter	58ms	24.7s	x 2.3	x 1.4	x 3.2
Edge Detect.	4.5ms	2.5s	x 1.8	x 0.95	x 1.7
Face Detect.	1.5s	382ms	x 4	x 2.9	x 11.6
Face Recog.	9.15s	860ms	x 10.6	x 9	x 95.4
Totals	10.7s	1.26s	x 8.5	x 6.7	

3. DESIGN FLOW AND SYSTEM INTEGRATION

3.1 The System-to-RTL Design Flow

In Figure 5 the design flow used for system architecture exploration and integration is described. The starting point is an untimed model of the system written in C/C++ code describing the desired functionality; at this stage the verification is done with simulations in CoWare N2C environment [3]. This methodology allows designers to validate the system specifications and consequently, with a progressive refinement of the functional blocks into hardware and software (partitioning process) and the generation of the HW/SW interface (interface synthesis), the verification of the system at a cycle accurate abstraction level. The microprocessor core is abstracted in the co-verification with its Instruction Set Simulator integrated into the simulation engine.

Extensive simulations of the system with the usage of the profiler (memory accesses, CPU load, exceptions) help in finding the computational kernels of the software running on the core (performance analysis).

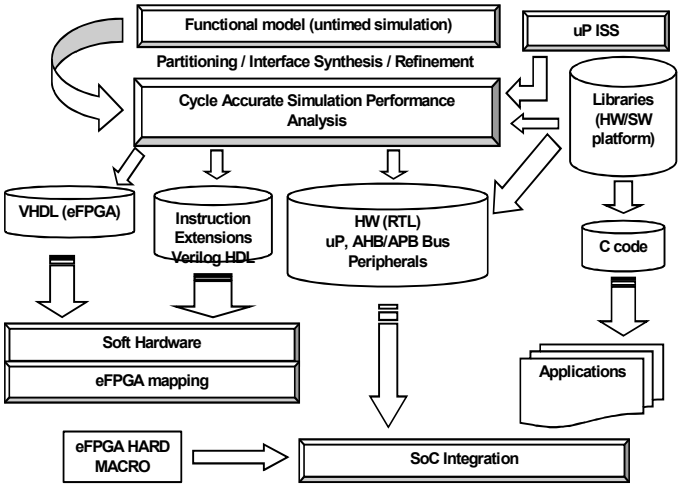

Figure 5. System to RTL Flow

At this point it is possible to group segments of codes that are the most time consuming as new instructions of the extensible processor. Those extensions of the Instruction Set can be easily mapped on the e-FPGA together with the VHDL code that results from the refinement process done after the HW/SW partition phase.

The system integration flow ends producing:

1. Soft Hardware to be mapped on the e-FPGA: HDL RTL code of instruction extensions, bus-mapped coprocessors and special purpose I/O peripherals.
2. Conventional fixed hardware: Microprocessor RTL code, AHB/APB bus and Peripherals.
3. Embedded Software (C code): Application software and low-level drivers for the hardware platform.

The C code generated by the flow described above is the final application while the RTL of the system with the e-FPGA hard macro goes into the SoC integration flow (RTL to layout).

3.2 The RTL-to-Layout Design Flow

In Figure 6 both silicon implementation flow and e-FPGA configuration flows are shown. These flows are run at different times. Once silicon implementation flow has produced the routed database its possible to implement e-FPGA flow that can be repeated for each different function built as a soft macro.

The RTL code of the CPU core, IP blocks and Interface modules (system bus) is synthesized and integrated with RAM blocks, Flash modules and FPGA hard macro in the floorplanning environment. To meet timing requirements at the boundary of the e-FPGA, a special care was taken during synthesis process for the logic cells that interfaces e-FPGA with the rest of the system. A particular set of constraints was specified to reach minimum delay of the hardwired logic. After the place and route stage, the final database is statically and dynamically verified against the RTL simulations in order to make verification at all levels of abstraction.

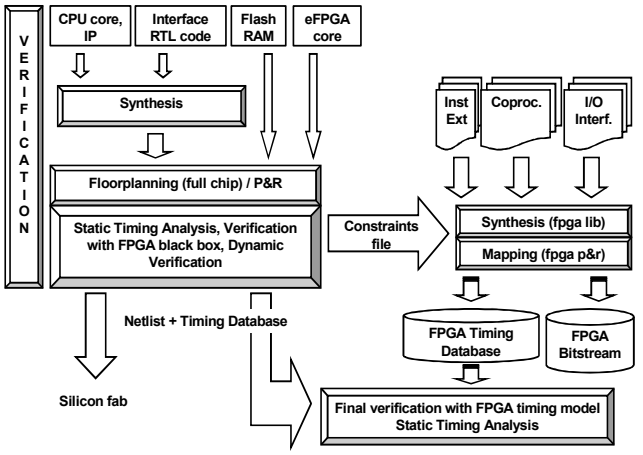

Figure 6. RTL to Layout Flow.

The timed database used for the verification, built after a parasitic extraction and a delay calculation process, allows knowing the effective delays at the boundary of the e-FPGA hard macro (all e-FPGA I/O pins are characterized with the static timing analyzer in the worst case condition). This information is exported in the e-FPGA flow as a constraint file and used during synthesis/mapping of the soft hardware by specific e-FPGA tools. This is done to correctly constrain the logic mapped on the e-FPGA with the real timing budget. Finally the generation of the bitstream and a timed view of the macro can be used for the final sign-off. Static timing analysis of the e-FPGA results in both a backannotated netlist and a timing view for full chip static timing analysis.

4. SYSTEM IMPLEMENTATION AND TEST

The full-chip is implemented in a 0.18um, 2-poly, 6-metal, CMOS embedded Flash technology. The layout of the system has been integrated using commercial place and route tools for digital ASIC. The chip is being tested and is fully functional at the clock rate of 125MHz (worst-case conditions). The processor system is able to reconfigure the e-FPGA at full speed. Reconfiguration takes about 500us at a clock rate of 100MHz. Technology and device characteristics are summarized in Table 2 and a chip micrograph is shown in Figure 7 with a floorplan view of system components. The system is being tested using both a face recognition application and a speech recognition application. As discussed in Section 2 we reported speedups of up to 8x using instruction extensions to accelerate face-recognition computing kernels. Additional 1.5x to 2x performance improvements are reported on specific I/O intensive tasks to interface an external CMOS camera and doing some image processing computations on-the-fly using the e-FPGA.

Table 2. Technology and chip characteristics.

Process	0.18 mm CMOS 2-Poly, 6-Metal Tunneling oxide: 10nmFlash cell size: 0.35mm2
Flash Memory (4x)	256Kbit x 9 Sectors Word: 128 bit Program Throughput: 1Mbyte/s Typ Read Rate: 400 Mbyte/s
SRAM memory	I$: 8kB (64-bit wide) D$: 8kB (64-bit wide) Buffers: 4x256B (8-bit wide) Main: 48kB (64-bit wide)
Chip size	8.4x8.4 mm2
e-FPGA size	8.2 mm2
Customizable I/O	24 general-purpose inputs 24 general-purpose outputs (tri-state) 8 general-purpose bidirs
Power supply	2.7-3.6V (I/O), 1.6-2.0V (core)

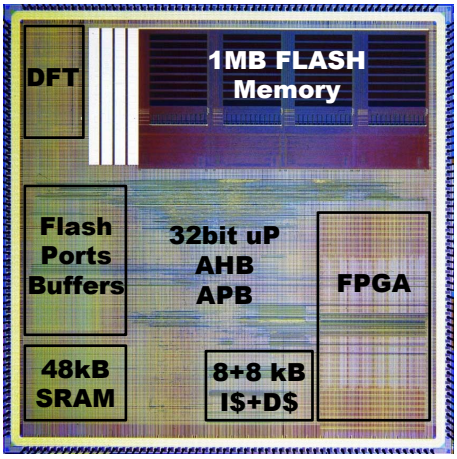

Figure 7. Chip Micrograph.

5. ACKNOWLEDGMENTS

The authors thank all the colleagues of NVM-DP Dept., A. Maurelli, F. Piazza and L. Fumagalli.

6. REFERENCES

[1] Young-Don Bae et al., "A Single-Chip Programmable Platform Base on A Multithreaded Processor and Configurable Logic Clusters", ISSCC 2002 Digest of Technical Papers, pp 336-337, Feb. 2002.

[2] Zhang et al., "A 1V Heterogeneous Reconfigurable Processor IC for Baseband Wireless Applications", ISSCC 2000 Digest of Technical Papers, pp 68-69,488, Feb. 2000.

[3] I.Bolsens, H.De Man, B. Lin, C.Van Rompaey, S.Vercauteren and D.Verkest, "Hardware/Software Co-Design of Digital Telecommunication Systems", Proceedings of the IEEE, Vol. 85, No. 3, March 1997, pp 391-418.