Soft Core – An Approach to Embedded System Design

Archana Mahatme
Dept. of Electronics and Communication
K.I.T.S., Ramtek,Dt.Nagpur, India
e-mail: mahatme.archana@gmail.com

V.P.Mahatme
Dept. of Computer Technology
K.I.T.S., Ramtek,Dt.Nagpur, India
e-mail: mahatme.vilas@gmail.com

Abstract— An embedded system is a computer system designed to perform one or a few dedicated functions often with real-time computing constraints. To handle a particular task, it requires powerful processors. As the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Design reuse is an important part of today's design practice. Reuse may be implemented using Intellectual Property (IP). An IP-core is a hardware description language (HDL) model of a specific processor that can be customized for a given application and synthesized for an ASIC or FPGA target. Soft core processors provide several advantages over custom designed processors such as reduced cost, flexibility, platform independence and greater immunity to obsolescence. This paper presents a survey of soft-core processors that are used in embedded systems. Several soft-core processors available from commercial vendors and open-source communities are reviewed.

Keywords- *Embedded system, Intellectual Property, HDL, SoC*

*****

I. INTRODUCTION

Embedded systems are hardware and software components working together to perform a specific application. They play vital role in our everyday life and have lot of application in various fields such as in medical field, industrial control system, computer networking, telecommunications systems, consumer electronics and many household devices. The hardware platform for embedded system consists of a microprocessor, on board memory, output display, input device and application software. The design of embedded system is also becoming increasingly difficult due to the constraints on area usage, size, power consumption and performance. In addition many embedded system developers are faced with tight time to market deadlines. Due to rapidly growing chip technology, it is possible to have several millions of gates in a chip. At the same time it is hard to design a complete system from scratch. Therefore the idea of design reuse becomes a very important concept in design methodology. Design reuse is the inclusion of previously designed components in software and hardware. The term is more frequently used in hardware development. Design reuse makes it faster and cheaper to design and build a new product. Reused components are already designed and which are tested for reliability. So, reusing the existing blocks reduce the possibility of failure based on design and verification of a block for first time. It accelerates the development of new products to meet today's time-to-market challenges. Developers can reuse a component in both similar and completely different applications. For example, a component used as part of a central processing unit for a PC could be reused in a handheld device. In hardware development, components in design reuse are called IP cores that are Intellectual Property cores. IP core is entirely portable. It can be easily inserted into any vendor technology or design methodology. IP cores are part of the electronic design automation.Universal Asynchronous Receiver/Transmitter (UARTs), central processing units, Ethernet controllers, and PCI interfaces are all examples of IP cores. Soft core processors are microprocessors whose architecture and behavior are fully described using synthesizable hardware description language. They can be synthesized for any application specific integrated circuit or field programmable gate array. Thus they provide substantial amount of flexibility to designers. This paper presents a survey of the available soft core processors that are used to design and implement embedded system.

II. TYPES OF IP CORE

Three types of IP cores exist, providing different optimization levels and flexibility of reuse.

A. Hard core

Hard cores are physical manifestations of the IP design. Hard-cores are mask and technology dependent modules that already have physical layout information which give predictable performance. Such cores, whether analog or digital, are called hard cores because the core's application function cannot be meaningfully modified by chip designers. The key deliverable is a fully verified layout in Graphical Design System II (GDSII) format along with a design for a test structure and test patterns. Transistor layouts must obey the target foundry's process design rules, and hence hard cores delivered for one foundry's process cannot be easily ported to a different process or foundry [1]. Merchant foundry operators such as IBM Fujitsu Samaung etc. offer a variety of hard-macro IP functions built for their own foundry process, helping to ensure customer lock-in .Thus it is technology dependent. The drawback of hard-core is that the cores can not be customized for a particular design application. Provide minimum flexibility and portability in reconfiguration and integration across multiple designs and technologies. The advantage of this is that they offer better predictability of chip performance in terms of timing performance and area because of their low-level representation. Integration is simple and the core can be dropped into a SOC physical design with very less effort. These are best for plug-and-play applications.

B. *Firm core*

Firm-cores are delivered as a mix of RTL code and a technology dependent net-list. Like the hard cores, firm (sometimes called *semi-hard*) cores also carry placement data but are configurable to various applications. They come ready for routing analysis and do not present significant difficulties for floor-planning, placement, and routing. They have the same routability properties as soft-cores. They use the advantages of both hard cores and soft cores. Firm cores try to balance the optimization and fast reuse potential of hard cores and the flexibility and reconfigurability of soft cores. They have a higher level of optimization and are targeted for specific device architecture. They are less portable than soft cores.

C. *Soft core*

The most flexible among three cores is soft core. It exists either as a net list (a list of the logic gates and associated interconnections making up an integrated circuit) or hardware description language (HDL) such as VHDL or Verilog. The net list is a Boolean-algebra representation of the IP's logical function implemented as generic gates or specific standard cells. An IP core implemented as generic gates is portable to any process technology. A net list gives the IP core vendor reasonable protection against reverse engineering. A synthesizable soft-core consists of a set of technology-independent HDL files, synthesis constraints, test-bench and validation information and adequate information. Both, net list and synthesizable cores are called "soft cores", as both allow a Synthesis, Placement and Route (SPR) design flow. Soft IP specification can be both simulated and synthesized. These are analogous to high level programming languages such as C. IP cores delivered to chip makers as RTL permit chip designers to modify designs (at the functional level), though many IP vendors offer no warranty or support for modified designs. Thus soft core offer highest degree of modification, flexibility and the requirements of a specific design application. For example, a soft processor core allows the core user to reconfigure the features of the processor, such as its instruction set, caches, communication mechanisms, and interrupt mechanisms to make the processor core more suitable for a particular System on Chip (SoC) application. However, as opposed to a hard core user, a soft core user must synthesize, optimize, validate, and develop tests for the soft core before integrating it in the SoC being designed. Being synthesizable, are compatible with the ASIC design flow. Therefore, the quality of a soft IP is highly dependent on the effort needed in the IP integration stage of SoC design.

III. SURVEY OF SOFT CORE PROCESSOR

There are two sources of IP core, commercial and open source. IP core available under license version are commercial cores. Some of the well known commercial cores are Altera's NiosII, Xillinx, Microblaze and Picoblaze and Tensilica's Xtensa. An open source soft core processor is freely available under GNU LGPL license and can be freely downloaded across the internet. "OpenCores.org" offers a wide variety of designs, mostly written in VHDL or Verilog. The open source cores are LEON by Gaisler Research, OpenRISC 1000 from opencores.org. Available soft-core processores provided by commercial vendors and open source communities are reviewed.

A. *Commercial Cores*

The two leading FPGA vendors , Altera and Xilinx pioneered this market several years ago by introducing processor cores specifically designed for their own programmable-logic fabrics. NiosII, and Microblaze, Picoblaze are the leading soft-core processors provided by Altrea, and Xilinx respectively. Newcomers to this market segment are Freescale Semiconductor and Tensilica. Freescale is licensing the ColdFire-V1 and Tensilica is licensing the Diamond Standard 106Micro. The ColdFire-V1 is an older embedded processor that dates back to Freescale's predecessor, Motorola which derived ColdFire from the 68K architecture in the 1990s.The Diamond 106Micro is a preconfigured microcontroller core based on Tensilica's Xtensa configurable-processor architecture. Another relatively recent arrival in this market segment is the ARM Cortex-M1, a microcontroller core designed specifically for FPGAs.

- Nios II Altera processor
  NiosII is a proprietary 32-bit RISC architecture and a processor core developed by Altera. It provides full 32-bit instruction set, 32 general-purpose registers and 32 external interrupt sources. It has dedicated instructions for computing 64-bit and 128-bit products of multiplication and floating-point instructions for single-precision floating-point operations. It has access to a variety of on-chip peripherals and interfaces to off-chip memories and peripherals. It also provide hardware-assisted debug module enabling processor start, stop, step and trace under integrated development environment (IDE) control. Instruction set architecture (ISA) is compatible across all NiosII processor systems and has performance up to 250 DMIPS [3]. Number of logic elements occupied by NiosII core is comparable to that of MicroBlaze, except for NiosII economy version. It uses significantly less logic elements. Nios II can only be used on Altera FPGAs. They also provide associated CAD tools such as Quartus II and System-On-Programmable Chip(SOPC) builder that allow designers to synthesis, program and debug their designs and build embedded systems on Altera's FPGAs[2]. GNU toolchain and Linux operating system has been ported to NiosII. NiosII soft core comes in three versions fast, standard and economy. Each core version modifies the number of pipeline stages, instruction and data cache memories and hardware components for multiply and divide operations. NiosII fast is optimized for maximum performance. It employs a 6-stage pipeline, and executes one instruction per cycle. It alsoincludes separate instruction and data caches. It support Memory Management Unit (MMU) and Memory Protection Unit(MPU). It provides hardware multiply, divide and shift operations. NiosII standard employs 5-stage pipeline and executes one instruction per cycle .It includes instruction cache and hardware multiply, divide & shift operations.MMU and MPU are absent. NiosII economy is designed to occupy as little area as possible. It doesn't use pipeline, executes one instruction in 6 cycles and no supplementary arithmetic blocks.

- Microblaze and Picoblaze Xilinx Microprocessor

MicroBlaze is a proprietary 32-bit RISC architecture and soft CPU core designed by Xilinx for use in their FPGAs. Xilinx Incorporation is the maker of Spartan and Virtex families of FPGA. There are also open-source clones of MicroBlaze, namely aeMB and OpenFire, which are device-independent and binary compatible with the original. MicroBlaze has 32 bit instructions, a 3-stage pipeline, a 32 register wide register file, a shift unit and two levels of interrupt with Harvard architecture. Memory can reside on chip or as an external peripheral. On-chip memory can be accessed by Microblaze using a local memory bus, which provides single access to the memory. Also, a general purpose interface known as on-chip peripheral bus can be used to interface Microblaze with both on-chip and off-chip memories as well as other peripherals. This basic design can then be configured with more advanced features to tailor to the exact needs of the target embedded application such as barrel shifter, divider, multiplier, single precision floating-point unit, instruction and data caches, exception handling, debug logic, fast simplex link interfaces and others [4]. The fast simplex link can be used to connect custom designed hardware modules directly to the pipeline in order to accelerate time critical tasks. The Microblaze soft core processor is targeted for the Virtex and Spartan families of FPGAs only. Xilinx also offers the Embedded Development Kit which includes Xilinx platform studio and a set of IP cores that are required for developing embedded system using Microblaze. It features a five stage pipeline, with most of the instructions completing in a single cycle. Although lacking a memory management unit, and so unable to run full Linux, several operating systems have been ported to the MicroBlaze including uClinux and FreeRTOS. Xilinx includes as part of its MicroBlaze development package, a MicroBlaze GNU C Compiler (MB-GCC) which allows programmers to use the C programming language to write programs for the architecture. Being developed specially for FPGA, MicroBlaze designs occupy a few times less area than LEON or OpenRISC-based designs, making it particularly suitable for mid-density FPGAs. The original MicroBlaze implements the features unsupported by the open-source clones, such as floating-point unit or 5-stage pipeline. Open-source clones implement only 3-stage pipeline.There is a port of the UClinux operating system to the MicroBlaze architecture. PicoBlaze is a proprietary 8-bit RISC architecture and a CPU core developed by Xilinx. Although PicoBlaze is available free of charge, the core is tied to a Xilinx architecture. There is also an open-source clone named PacoBlaze, which is a device-independent CPU core binary-compatible with the original PicoBlaze. PicoBlaze was designed to operate in low-density FPGAs and occupy about 100 Spartan/Virtex slices.

- ColdFire-V1
  Freescale's ColdFire-V1 processor is available from IPextreme, an independent company that handles IP licensing for Freescale and other IP sources. IPextreme packages the cores, provides tech support and collects royalties due. Freescale has tweaked one version of the ColdFire-V1 specifically for Altera Cyclone III FPGAs. This version is available from IPextreme's online Core Store without a licensing fee. Another version of the ColdFire-V1 works with virtually any FPGA. Both versions incur royalties. In addition to its low cost, the ColdFire-V1 is attractive to developers familiar with the long-established 68K architecture.

- Diamond Standard Processor Cores
  Tensilica's Diamond Standard Processor core family consists of four ready-to-use cores available to eASIC customers. The diamond Standard Series cores are available as synthesizable HDL description of the processors written in Verilog. These cores range from area-efficient, low-power controllers to the industry's highest performance licensable DSP and popular audio processor. This family covers a very broad range of performance/power options. It is based on Tensilica's highly efficient Xtensa configurable 32-bit RISC processor. Tensilica has emphasized instruction set configurability as the primary feature that distinguishes Xtensa from other core offerings [5]. Tensilica's Diamond 106Micro core is available through the Synplicity ReadyIP program. The preconfigured 106Micro is Tensilica's smallest core, lowest power 32-bit RISC controller intended mainly for microcontroller applications. Developers can implement and distribute the 106Micro in as many as 10,000 FPGAs without paying a licensing fee or chip royalties. The108mini is a low-power RISC controller with built-in DSP capabilities. 212GP is a flexible mid-range RISC controller with flexible memory choices. 232L is a mid-range CPU with Memory Management Unit for Linux OS support. All Diamond Standard Processor cores offers the designer a set of predefined parameters which they configure in order to tailor the processor to the intended application. They are also extensible so that designers can also invent custom instructions and execution units and integrate them directly into processor core. All Diamond Standard Processor cores share a common base of 16/24-bit instructions. Some Diamond processors add VLIW-style 64-bit instructions. Tensilica's VLIW capability allows the issue of multiple operations per instructions, boosting the processor's parallel execution abilities and application performance.

- Cortex-M1 ARM
  Cortex-M1 processor is an implementation of the proprietary 32-bit ARMv6 architecture designed for FPGA. It requires license from ARM Limited. There are Actel flash-based FPGA chips which are shipped with Cortex-M1 license included. Cortex-M1 can be used with Xilinx and Altera FPGAs. Advanced RISC Machine (ARM) is a popular RISC architecture particularly suitable for applications that demands low power consumption. Many operating systems have been ported to ARM, including Linux and Windows. We now have quad core Cortex A9 processors and more [5].

B. Open Source processor

Open source cores are IP components that are freely available in open source community. Usually these types of

cores are used in academic research as well as in the development of embedded systems. Leon processor, OpenRISC 1200, LatticeMicro32, S1 core, DSPuva16 are available in the open source community.

- Leon3 Gaisler Microprocessor
  LEON3 is an open-source synthesizable VHDL model of the SPARCv8 32-bit architecture with Harvard architecture designed by Aeroflex Gaisler AB [6]. It is a part of the GRLIB open-source IP core library. The model is highly configurable and particularly suitable for system-on-a-chip (SOC) designs. The full source code is available under the GNU LGPL license, allowing free and unlimited use in both research and commercial applications. The LEON3 processor has the features such as SPARC V8 compliant integer unit with 5-stage pipeline, hardware multiply, divide and MAC units, interface to the Meiko FPU and custom co-processors, separate instruction and data cache. It has set-associative caches: 1 - 4 sets, 1 - 64 Kbytes/set, Random, LRR or LRU replacement, data cache snooping, AMBA-2.0 AHB and APB on-chip buses, 8/16/32-bits memory controller for external PROM and SRAM, 32-bits PC133 SDRAM controller. It also has on-chip peripherals such as uarts, timers, interrupt controller and 16-bit I/O port, advanced on-chip debug support unit and trace buffer and power-down mode. The processor is extensively configurable and can be efficiently implemented on both FPGAs and ASIC technologies. The only technology-specific mega-cells needed are RAM cells for caches and register file. The Gaisler Research LEON3 allow developers to optimize the core for either standard-cell or programmable-logic implementations. When targeting FPGAs, the LEON3 uses modified I/O pads, memory cells, clock generators, pipeline stages, register files, multipliers, adders, and other structures. As LEON3 is configurable, developers can change the VDHL source code if they wish. For educational users, the LEON3 processor is free. In either case, LEON3 software-development tools are free. A big attraction of the LEON3 is that it includes an extensive library of peripheral IP called GRLIB. This library includes AMBA bus controllers, memory controllers, DMA controllers, a programmable UART and a 10–100Mb/s ethernet controller. Source code for GRLIB peripherals is available under a GNU General Public License.

- OpenRISC 1200 Processor
  OpenRISC 1200 is an open-source 32-bit processing core that implements the OpenRISC 1000 RISC architecture. It is designed with emphasis on performance, simplicity, low power requirements, scalability and versatility [7].The OR1200 processor is a 32-bit load and store RISC architecture developed by the OpenCores.org community. The processor features a Harvard architecture containing separate data and instruction caches. The OpenRISC architecture uses a 5-stage pipeline and its own instruction set.GNU tool chain and Linux operating system has been ported to the OpenRISC architecture. OpenRISC 1200 is available under the terms of the LGPL license, making it possible to use it in proprietary projects.

- LatticeMico32
  LatticeMico 32 is an open-source 32-bit RISC architecture and a CPU core developed by Lattice Semiconductor. Lattice created a portable design that can be used on other manufacturers' FPGAs or even on an ASIC,otherwise it is comparable to MicroBlaze and NiosII standard. LatticeMico32 doesn't include a floating-point unit. GNU toolchain and Linux operating system has been ported to LatticeMico32 architecture. Lattice Mico System is based on Ecllipse C/C++ Development environment, which is an industry open source development and application framework for building software[8].

- S1 core
  S1core is an open-source implementation of the SPARCv9 architecture. On March 21, 2006 Sun Microsystems, the developer of the SPARC architecture, released their UltraSPARCE T1 microprocessor to public under the terms of the GNU GPL. UltraSPARC T1 was an 8-core implementation of the SPARCv9 64-bit architecture, designed to run in power-critical environments. UltraSPARC T1 itself is too huge to be implemented on FPGA. S1core is a cut-down version of the UltraSPARC T1 containing just one CPU core and additional Wishbone bus controller. Even then, S1 Core occupies 37000-60000 Virtex-5 LUTs depending on the number of supported threads and L1 cache blocks and is probably too big for most SoC designs. Nevertheless, it is probably the only available 64-bit CPU core for FPGA. The GPL license ensures that this core can be used for free, but it requires that you give away the source code of the whole design when distributing S1 Core based devices.

- *DSPuva16*
  DSUPuva16 is an open-source 16-bit RISC core designed by Santiago de Pablo, University of Valladolid. It has an interesting feature of having a dedicated hardware multiplier accumulator (MAC) and still occupying small area, making it possible to use it even in low-density FPGAs. The downside is that the CPU supports quite a small program memory and no data memory except internal registers. Therefore this processor can be utilized to implement simple algorithms that nevertheless include many multiplication operations. One possible application is automatic control. 1-CORE Technologies has made DSPuva16 available from its website for the general public .

IV. CONCLUSION

The use of soft cores processor holds many advantages for the designer of an embedded system over custom designed processor. The soft core processors are flexible and can be customized for a specific application with relative ease. An IP core should be entirely portable that is, able to easily inserted into any vendor technology or design methodology. Thus, soft core processor are technology independent and can be synthesized for any target ASIC and FPGA technology. The architecture and behavior of soft core processor are described at a higher abstraction level using HDL. Therefore it becomes much easier to understand the overall design. It accelerates the development of new products to meet today's time-to-market

challenges. The advantage of reusing the existing blocks is to reduce the possibility of failure based on design and verification of a block for first time.

REFERENCES

[1] Eli Greenbaum, "Open Source Semiconductor Core Licensing",Hardvard Journal of Law and Technology. Vol. 25, No.1, 2011.

[2] Jason G.Tong,Ian D.L. Anderson and Mohammed A.S.Khalid, "Soft-core processors for embedded System",18th International Conference on Microelectronics(ICM)2006.

[3] NiosII Processor Reference Handbook, February 2014.

[4] Rod Jesman,Fernando MartinezVallina,Jafer Saniie,"Microblaze Tutorial Creating a Simple Embedded System and Adding Custom Peripherals Using Xilinx EDK Software Tools"Illinois Institute of Technology.

[5] http://www.androidauthority.com/arm-a9-vs-arm-a15-87559 acessed on August 2014

[6] The Leon Processor User's Manual, Version 2.3.5,July 2001.

[7] "OpenRISC1200 IP Core Specification"Opencores.org accessed from www.opencores.org

[8] http://www.latticesemi.com/Products/DesignSoftwareAndIP.aspx acessed on October 2014.