A Survey of Open Source Processors for FPGAs

Rui Jia[†,‡], Colin Yu Lin[†], Zhenhong Guo[†,‡], Rui Chen[†,‡], Fei Wang[†], Tongqiang Gao[†], and Haigang Yang[†,§]
† System on Programmable Chip Research Department,
Institute of Electronics, Chinese Academy of Sciences, Beijing, China
‡ The University of Chinese Academy of Sciences, Beijing, China
§ Corresponding Author: yanghg@mail.ie.ac.cn

***Abstract*—FPGA-based SoCs have been gaining great favors among traditional applications and expanding the application domains to some new areas. Reusing and sharing components is an attractive and pratical methodology for system designers to reduce the design complexity of the SoC architectures. Open source hardware has become an effective method of improving the design productivity. As the core functional component, processors affects the performance of SoC systems. This paper investigates existing open source processors, and gives an overview. From the points of usability and stability, the main features of open source processors are summarized. Following these features, some open source processors with high usability and stability are selected. These open source processors and existing vendor-provided soft processors are implemented on Stratix V and Virtex-7 FPGAs using corresponding EDA tools, Quartus II and ISE. The implementation results are compared and discussed.**

I. Introduction

FPGAs offer many advantages such as reduced design cost, flexibility and short time to market compared to ASIC [1]. Traditional applications including industry control, communication, medical and signal processing systems accelerate the development of FPGA devices. As mentioned in [2], processors combined into FPGAs are sound both technically and commercially, and FPGA-based SoCs have become a technology trend for reconfigurable systems [3]. The rapid development of FPGA-based SoC also makes it possible to expand the application domains of FPGAs to some new areas, such as cloud computing, mobile market and database center [4]. In the meantime, the implementations of FPGA-based SoC systems and designs become more and more ambitious. The most attractive and practical methodology for SoC designer is to select reusable components and IPs to integrate the systems [5]. As the most important component, processor should be designed, verified, tested and shared in module level to increase the design productivity [6].

Being a significant access to shared IPs, open source hardware is becoming an effective method for improving the design productivity [7]. Also, the open source hardware available today is a valuable resource [8]. It is convenient to learn and innovate on an IP which is open with free and publicly available source files. If processors are shared in an open source form, it can be obtained and optimized for different applications. What is more, processors designed with excellent usability can be picked out to speed up the development process. Needless to say, the feature of stability for a processor guarantees the performance and reliability of the whole SoC system.

Soft core processors are microprocessors whose architecture and behavior are expressed in HDL [9]. It is convenient for designers to implement a soft core processor on all kinds of FPGA-based SoC systems. There are some advantages to use soft cores. First, the multiple physical components for a system can be reduced through using a single FPGA device [9]. Second, the "visibility" of soft cores is so high that system designers can not only understand the processor architecture but also realize deep hardware customization for different application domains [10]. Third, due to the remarkable flexibility, soft cores can be implemented on any target FPGA devices and ASIC technology [11].

However, the numerous open source processors make the selection process difficult. Besides features like reliability and visibility, usability and stability should be taken into consideration when choosing open source processors. Furthermore, open source processors with different ISAs, licenses and tool chains are selected for different SoC implementations.

In this survey, we review 178 open source processors and give out some principles of how to choose suitable open source processors. The differences between existing vendor-provided soft cores and open source processors are explored by implementing them on Stratix V and Virtex-7 FPGAs using corresponding EDA tools, Quartus II and ISE, and the implementation results are compared and discussed.

The major contributions of this survey paper are:

- An overview of open source processors which can be found through open source communities, and their classifications.
- A method to select open source processors with high stability and usability from all surveyed processors.
- Several selected open source processors and existing vendor-provided soft processors are implemented and compared.

The remainder of this paper is organized as follows. An overview of open source soft processors is presented in next section. Section 3 details the selection method of open source processor for FPGA-based SoC implementations. Section 4 implements the selected open source processors and compares with existing vendor-provided soft cores. In section 5, the implementation results and comparisons are discussed. Section 6 concludes the paper.

II. An Overview of Open Source Soft Processors

From our investigation, the number of open source soft processors increases rapidly in recent years. Open source soft processor gained wide popularity in embedded systems.

Commercial vendors and open source communities proposed a great number of open source soft processors. Such processors have different features. As a result, it is a great challenge for embedded system developers to select a proper processor from the plenty of candidates. In this section, we give an overview of open source soft processors from the aspects of usability and stability.

Totally 178 processors are discussed in this paper, and they all come from processor projects on OpenCores [12], and soft microprocessor and open processors on Wikipedia [13]. Source code of processors from OpenCores are available directly. For the others, the source code can be get through Internet. In the 178 processors, 4 processors have no source code, 1 is targeted for CPLD, 2 are written in C/C++, and 2 are designed as interface translators. These 9 processors are beyond the scope of open source soft processors for FPGAs. So, the total number of open source soft processors needed to be further considered decreases to 169.

The development process of open source processor lasts a long time. Generally, the life cycle of a open source project are categorized into 6 stages, including planning, pre-alpha, alpha, beta, stable, and mature [14].

The development stage determines the stability of processor. In the stable stage, the processor is feature-complete and it has few bugs. Therefore, only when the project is in stable stage, the processor is reliable. Based on investigation, only 68 open source processors in stable can be used conveniently.

Although these 68 processors are stable, many other factors determine the usability and stability of the processor. For the selected 68 stable processors, we give a detailed description from the aspects of license, instruction set architecture (ISA), compiler and assembler, verification and design documentation in the rest of this section.

A. License

All software is subject to copyright law by default. License is the permission that owner of the software give to people to copy and modify the software [14]. Thus, license of the processor is one of the most important factors when choosing an open source soft processor, because it determines what users can or cannot do with the processor. Most of licenses of open source processor are ported from open source software [15].

The license of processors cover GNU General Public License (GPL), GNU Lesser General Public License (LGPL), Berkeley Software Division (BSD), and Creative Commons-Attribution (CC-BY).

B. Instruction Set Architecture (ISA)

ISA serves as the interface between hardware and software. ISA with good programmability makes it easy to express program efficiently. Besides building the whole architecture with the features of implementability and compatibility, the ISA should be designed with free ISA instead of proprietary ones for open source processors. ISAs of the open source processors are surveyed and sorted in Table I. In the table, most of the ISAs are property of commercial corporations. The "unnamed" ISAs are designed by limited developers and they are not widely used.

TABLE I. INSTRUCTION SET ARCHITECTURES OF OPEN SOURCE PROCESSORS

	Provider	*ISA*	*Number*
Free	ARM	ARMv2	*1*
	Lattice Semiconductor	LatticeMicro32	*1*
	Open Cores	DLX	*1*
		OpenRISC	*3*
		unnamed	*14*
	Sun Microsystems	SPARCv8	*1*
		SPARCv9	*4*
Proprietary	Atmel	AVR	*5*
	Hitachi	SuperH-2	*1*
	Intel	8080	*2*
		8088	*3*
		80186	*1*
		MCS-48	*1*
	Microchip Technology	PIC-baseline family	*5*
	MIPS Technology	MIPS I	*6*
		MIPS 16	*1*
		MIPS 32	*4*
	MOS Technology	6502	*3*
	Motorola	6800	*1*
		68000	*1*
		68HC05	*1*
		68HC11	*1*
		68HC08	*1*
	National Semiconductor	COP400	*1*
	Synopsys	ARC	*3*
	Texas Instruments	MSP430	*1*
	Xilinx	MicroBlaze	*1*
	Zilog	Z80	*4*
Total			*68*

C. Compiler and Assembler

Compiler and assembler are the important components of development tool chain. Compiler translates source code from advanced language such as C or C++ into assembly language. Assembler is used to translate the code from assembly language into machine code. Compiler and assembler determine the usability of the processors. Besides, a stable and open source compiler or assembler attracts more users. Among the processors surveyed, 38 processors have both compiler and assembler, 12 processors provide only assembler, and 18 processors do not provide either assembler or compiler.

D. Verification

Verified processors is more reliable and stable. In the 68 processors, about 68% processors have been verified on FPGAs. They were implemented on FPGA board from Xilinx, Altera or Actel, and application programs were runs to test the logic function. 7 of the processors have also been verified as ASIC. The verified processors are provided with verification results including area consumption and performance, which are useful as reference for the developer to incorporate the processor into a complex system.

E. Design Documentation

When deciding to use an open source hardware product instead of a proprietary one, availability of design files and documents weighs heavy [16]. The design files of processors often include the description of the processors architecture, instruction set and the process of verification of the processors. We look into the processors and present our results in Although design documentation is not necessary, it takes less effort for user to understand the processor and add it into their embedded system if the processor has a detailed design document.

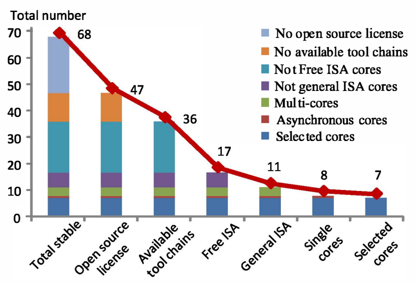

Fig. 1. Selecting Process of Open Source Processors.

F. Summary

Features of open source soft processors are summarized in Table II. The table consists of 68 stable soft processors under open source license and every processor is descried from the aspects of license, ISA, compiler & assembler, ASIC/FPGA verification and design documentation. License, ISA, compiler & assembler and design documentation determine the usability of the processor, and ASIC/FPGA verification reflects the stability. In Table II, most processors have proprietary ISA and only 10 processors have free ISA excluding the "unnamed".

III. Selection of Open Source Processors

Features of the open source processors are summarized in Table II. Many of the open source processors have very low usability, due to lacking of existing compiler and assembler, restrictions on license and ISA. Therefore, in this section, we select processors based on three most significant items including license, availability of compiler and assembler, and free ISA.

The process of selecting open source processors is presented in Fig. 1. First and the most important factor we take into consideration is license. Form the total 68 stable processors, 47 processors under open source license are chosen. Because open source license gives user the permission to use, modify the source code of the processor.

Furthermore, compiler and assembler are the essential tools for users to build an embedded system. From the selected 47 processors under open source licenses, we choose 36 processors with available compiler and assembler. With existing compiler and assembler, it is easy for designers to compile application program instead of developing own compiler and assembler.

Last but not least, ISA is not an neglectable item especially when using the processor for commercial purpose. From the selected 35 processors, 17 processors with free ISA are chosen. Among them, ISA's of 14 processors are unnamed, which are designed by limited developers and are not widely used. We tend to choose 11 processors with general ISA because they have stable and well-supported software development tools. The 11 selected processors are marked in red in Table II.

We look into the 11 selected processors. OpenSPARC T1, OpenSPARC T2 and OpenRISC 1200 HP are multi-core processors. S1 is the cut-down version of OpenSPARC T1 containing just one CPU core. OpenRISC 1200 HP can instantiate multiple OpenRISC 1200 in the same design. ASPIDA processor implements an asynchronous IP of DLX instruction set architecture which are controlled by a set of logic instead of a global clock. As we focus on single-core synchronous processor in this paper, these 3 multi-core processors and 1 asynchronous processor are not studied.

Following this process, 7 open source processors are selected. They are Amber [17], LatticeMicro32 (LM32) [18], S1 [19], Altor32 [20], OpenRISC 1200 (OR1200) [21], LEON2 [22] and LEON3 [23].

The features of bit-width, pipeline depth, complex computation unit, cache architecture and memory management unit (MMU) of the 7 selected open source processors are listed in Table III. These selected processors can be used under open source license, and they are supported with compatible compilers and assemblers. The ISA's of these open source processors are all free.

IV. Implementations and Comparisons

In this section, features of selected open source processors and soft cores provided by FPGA vendors are briefly reviewed firstly. Then two comparison works are presented: i) All selected processors and Nios II provided by Altera are implemented using Altera EDA platform, Quartus II, and the implementation results are compared and discussed. ii) The selected processors and MicroBlaze supplied by Xilinx are also implemented with Xilinx EDA tools, ISE Design Suite for comparisons. After that, more detailed discussions about open source processors and commercial soft cores are presented in next section.

A. Processor Descriptions and Platform Configurations

Features of selected processors are summarized in Table III. The complex computation unit, total size of cache and MMU types of some processors can be configured according to different demands of application. The default settings of all these configurations by the source code providers is used in our implementations, which are labeled in Table III with "(d)".

Three versions of Nios II cores are provided by Altera, economic, standard and fast cores, which are noted as Nios II/e, Nios II/s and Nios II/f, respectively. Nios II/e is designed to achieve a small core size and it has a limited set of features. For high performance, Nios II/f can be more flexibly configured for different performance requirement. Balanced performance and size can be achieved by using Nios II/s. MicroBlaze is the most wildly used soft core provided by Xilinx. MicroBlaze can be configured to enable functions such as pipeline depth, data cache size, instruction cache size, FPU, hardware multiplier, hardware divider, hardware barrel shifter, local memory bus data/instruction interface. MicroBlaze can also be used as a dual-core processor and it is designed with plenty of interfaces. The features of Nios II and MicroBlaze are summarized in Table IV, and the default settings used in our implementation are labeled with "(d)".

In order to compare the selected open source processors with commercial soft processors, all selected processors and Nios II e/s/f are implemented on an Altera Stratix V FPGA, and all selected processors and MicroBlaze are implemented on a Xilinx Virtex-7 FPGA. In the following two subsections,

TABLE II. OPEN SOURCE SOFT PROCESSORS, AND THEIR FEATURES: LICENSE, ISA, COMPILER & ASSEMBLER, VERIFICATION, AND DESIGN DOCUMENTATION, SELECTED PROCESSORS IN RED.

Processorss	*License*	*ISA*	*Compiler & Assembler*	*Verification*	*Documents*
16-bit Microcontroller	No license	unnamed	C compiler and assembler		√
16-bit CPU based on Caxton Foxter's Blue	GPL	unnamed	A cross compiler		
68hc05	No license	Motorala's MC68HC05	unknown	FPGA	
68hc08	No license	Motorala's MC68HC08	unknown	FPGA	
8080 Compitable CPU	Public domain	Intel's 8080	MI C/Micro Basic	FPGA	
AEMB	Modified BSD	Xilinx's MicroBlaze	GCC tool chain	FPGA	√
Altor32	LGPL	OpenRISC	GCC tool chain	FPGA	
Amber	LGPL	ARMv2	Sourcery G++	FPGA	
Aquarius	GPL	Hitachi's SuperH-2	GNU C for SuperH	FPGA	√
ASPIDA sync/async DLX	LGPL	DLX	Dlxgcc+dlxassembler	ASIC&FPGA	√
AVR Core	No license	Atmel's AT mega 103	WinAVR		
AVR Hyper pipelined	LGPL	Atmel'sAVR	WinAVR		√
AX8mcu	No license	Atmel's 90S1200/2313	unknown		
ClaiRISC	No license	PIC 12-bit	HiTechPICC	FPGA	
Cpu Generator	No license	unnamed	Built-in assembler		√
cpu6502_tc – R6502	GPL	MOS Technoloty's 6502	unknown	FPGA	√
CPU86	GPL	Intel's 8088	Any 8088 assembler		
Data Flow Processor	No license	unnamed	unknown	FPGA	
Educational16-bit MIPS	LGPL	MIPS16	A assembler written in JAVA		√
Educational RISC	No license	unnamed	unknown	FPGA	√
FORTH processor with Java compiler	LGPL	unnamed	FORTH-assembler and a java compiler	FPGA	√
HC11 Compatible – Gator μ Processor	LGPL	Motorala's 68HC11	HC11 C/C++ Compiler includeDCC	FPGA	
HIVE	Others	unnamed	unknown		
LatticeMicro32	GPL	LatticeMicro32	gcc+binutils	FPGA	√
LEM1_9	LGPL	unnamed	Assembler written in C#	FPGA	√
LEON2	LGPL	SPARCv8	BCC	ASIC&FPGA	√
LEON3	GPL	SPARCv9	BCC	ASIC&FPGA	√
Leros	BSD	unnamed	Compiler written in JAVA		√
Light 8080 compatible	GPL	Intel's 8080	Small -C	FPGA	√
McAdam's RISC	GPL	unnamed	spr assembler	FPGA	√
MicroSimplez	LGPL	unnamed	unknown	FPGA	
MiniMIPS	LGPL	MIPS I	gasm	FPGA	√
Mini-RISC core	No license	PIC 12-bit	Free MPLAB		
MIPS_enhanced	LGPL	MIPS I	gcc-elf-mips	FPGA	
MIPS32 Release1	LGPL	MIPS32	GCC tool chain		√
Mips 789	No license	MIPS I	gcc-elf-mips	FPGA	√
Mips-FaultTolerant	LGPL	MIPS32	unknown	FPGA	√
mipsr2000	LGPL	MIPS32	unknown	FPGA	√
Navré AVR clone (8-bit RISC)	GPL	Atmel'sAVR	unknown	FPGA	
Next 80186	LGPL	Intel's 80186	Macro Assembler	FPGA	√
Next Z80	LGPL	Zilog's Z80	unknown	FPGA	
Open8 μ RISC	BSD	Synopsys's ARC	Hi-Tech compiler+binutils	FPGA	
OpenMSP430	BSD	TI's MPS430	MSPGCC	ASIC&FPGA	√
OpenRISC 1000	LGPL	OpenRISC	GCC toolchain	ASIC&FPGA	√
OpenRisc 1200HP	LGPL	OpenRISC	GCC toolchain		√
OpenSPARC T1	GPL	SPARCv9	Solaris Studio	ASIC&FPGA	√
OpenSPARC T2	GPL	SPARCv9	Solaris Studio	ASIC&FPGA	√
Plasma	Public domain	MIPS I	gcc-elf-mips	FPGA	√
PPx16	No license	PIC 12-bit	HiTech PICC	FPGA	
RISC Microcontroller	No license	Atmel's AT90S1200	AVR assembler and AVR studio		√
RISC 16f84	CC-BY	PIC 12-bit	HiTech PICC	FPGA	
RISC5x	LGPL	PIC 12-bit	HiTechPICC	FPGA	
S1 Core	GPL	SPARCv9	GCC toolchain		√
SAYEH educational	LGPL	unnamed	unknown	FPGA	√
System68	No license	Motorala's 6800/6801	unknown		
T400 μ Controller	GPL	National Semiconductor's COP400	Macro assembler	FPGA	√
T48 μ Controller	GPL	Intel'sMCS-48	Macro assembler	FPGA	√
T65 CPU	No License	MOS Technology's 6502 65c02 65c816	unknown	FPGA	
T80 CPU	No license	Zilog's Z80 8080	Z88dk cross compiler		
TG68	LGPL	Motorala's 68000	unknown	FPGA	
Tiny64	No license	unnamed	assembler		
UCore	No license	MIPS32R2	GCC toolchain	FPGA	
Wishbone Z80	No license	Zilog's Z80	AS80 assembler		√
Y80e	BSD	Zilog's Z80	unknown		√
YACC	No license	MIPS I	gcc-elf-mips	FPGA	√
Yellow Star	No license	MIPS I	GCC toolchain		√
ZPU	FreeBSD+GPL	unnamed	GCC toolchain	FPGA	

TABLE III. BIT-WIDTH, PIPELINE DEPTH, COMPLEX COMPUTATION UNIT, CACHE ARCHITECTURE AND MEMORY MANAGEMENT UNIT (MMU) OF THE 7 SELECTED OPEN SOURCE PROCESSORS: AMBER, LATTICEMICRO32 (LM32), S1, ALTOR32, OPENRISC 1200 (OR1200), LEON2 AND LEON3.

			Amber23	*Amber25*	*LM32*	*S1*	*Altor32*	*OR1200*	*LEON2*	*LEON3*
Processor Architecture		*Bit-width*	32	32	32	64	32	32	32	32
		Pipeline Depth	3	5	6	6	5	5	5	7
	Complex Computation Unit	*Multiplier*	√	√	Optional, √ (d)	√	√	√	√	√
		Divider	N/A	N/A	Optional, √ (d)	√	N/A	N/A	√	√
		FPU	N/A	N/A	N/A	√	N/A	N/A	√	√
		Register	15	15	32	640	32	32	40-520, 136(d)	40-520,136(d)
	Cache	*Total Size/Kbytes*	8(d),12,16,32	16,24,32(d)	0,1,2,4(d),8	32(d)	16(d)	16(d)	8(d)	0.008(d)-64
		Instruction Cache/Kbytes	8(d),12,16,32	8,12,16(d),32	0,1,2(d),4,8	16(d)	8(d)	8(d)	4(d)	0.004(d)-32
		Data Cache/Kbytes		8,12,16(d),32	0,1,2(d),4,8	16(d)	8(d)	8(d)	4(d)	0.004(d)-32
		Sets	256(d)	256(d)	128,256,512(d),...	128(d)	256(d)	512(d)	128(d)	1(d)-256
		Associativity/Ways	2(d),3,4 or 8	2,3,4(d) or 8	1(d),2	4(d)	1(d)	1(d)	1(d)-4	1(d)-4
		Byte-per-line/Bytes	16	16	4(d),8,16	32(d)	32	16(d)	32(d)	4(d)-8
		Write policy	Write through	Write through	Write through	Write through	Write through	Write through	Write through	Write through
		Replacement policy	Read- miss	Read- miss	Read- miss	LRU	Read- miss	LRU	LRU,LRR,Random(d)	LRU,LRR,Random
	MMU	*Shared/Separate TLB*	N/A	N/A	N/A	Separate	N/A	Separate	N/A(d),Optional	N/A(d),Optional
		No. of Instruction TLB entries	N/A	N/A	N/A	64	N/A	16,32,64(d),128	2-32,N/A(d)	2-32
		No. of Data TLB entries	N/A	N/A	N/A	64	N/A	16,32,64(d),128	2-32,N/A(d)	2-32
		No. of Shared TLB entries	N/A	N/A	N/A	N/A	N/A	N/A	2-32,N/A(d)	2-32
Interface			Wishbone	Wishbone	Wishbone	Wishbone /Amba	Wishbone	Wishbone	Amba	Wishbone /Amba

the implementation results on the two different devices are presented.

B. *Implementations and Comparisons on Altera FPGA*

Implementation results of open source processors and Nios II e/s/f on an Altera Stratix V FPGA are shown in Table V. Nios II/e/s/f can be implemented with less ALMs and registers than all the open source processors. LEON3 needs the least resources in all selected open source processors. The average ALMs and registers of Nios II e/s/f is 19% and 29% of the averages of all open source processors. S1 and LEON2 need much more ALMs, registers and total block memory bits than all other processors, and the maximum frequency of S1 is the lowest.

F_{max} of Nios II/e/s/f are higher than all open source processors. F_{max} of Nios II/e is the highest and F_{max} of Nios II/s is the lowest in all three versions of Nios II. LEON3 with 7 stages of pipeline and Amber23 with a 3-stage pipeline achieve the highest and the lowest frequencies in all open source processors, respectively. F_{max} of open source processor is proportionate to the pipeline depth.

In Table V, the core static thermal power dissipations P_{CS} of all processors are nearly the same, because they are mainly decided by the chosen FPGA device. Total block memory bits

TABLE IV. BIT-WIDTH, PIPELINE DEPTH, COMPLEX COMPUTATION UNIT, CACHE ARCHITECTURE AND MEMORY MANAGEMENT UNIT (MMU) OF FPGA VENDOR-PROVIDED SOFT PROCESSORS: NIOS II E/S/F FROM ALTER AND SINGLE/DUAL MICROBLAZE FROM XILINX.

			Altera			Xilinx	
			Nios II/e	*Nios II/s*	*Nios II/f*	*MicroBlaze/single*	*MicroBlaze/dual*
Processor Architecture		*Bit-width*	32	32	32	32	32
		Pipeline Depth	N/A	5	6	3/5(d)	3/5(d)
	Complex Computation Unit	*Multiplier*	N/A	Optional,N/A(d)	Optional, √ (d)	Optional, √ (d)	Optional, √ (d)
		Divider	N/A	Optional,N/A(d)	Optional, √ (d)	Optional,N/A(d)	Optional,N/A(d)
		FPU	N/A	N/A	N/A	Optional,N/A(d)	Optional,N/A(d)
		Register	32	32	32	32	32
	Cache	*Total Size/Kbytes*	N/A	2(d),4,...	4(d),8,...	16,32,N/A(d)	16,32,N/A(d)
		Instruction Cache/KBytes	N/A	2(d),4	2(d),4,...	8,16,N/A(d)	8,16,N/A(d)
		Data Cache/KBytes	N/A	N/A	2(d),4,...	8,16,N/A(d)	8,16,N/A(d)
		Sets	N/A	64(d),128	64(d),128,...	512	512
		Associativity/Ways	N/A	1(d)	1(d)	1	1
		Byte-per-line/Bytes	N/A	32(d)	4,16,32(d),...	16(d),32	16(d),32
		Writepolicy	N/A	N/A	Write through	Write through(d),Write Back	Write through(d),Write Back
		Replacement policy	N/A	unkown	unkown	unkown	unkown
	MMU	*Shared/Separate TLB*	N/A	N/A	Shared+Separate	Separate+Shared	Separate+Shared
		No. of Instruction TLB entries	N/A	N/A	6(d)	Optional,N/A(d)	Optional,N/A(d)
		No. of Data TLB entries	N/A	N/A	4(d)	Optional,N/A(d)	Optional,N/A(d)
		No. of Shared TLB entries	N/A	N/A	128(d)	64,N/A(d)	64,N/A(d)
		Interface	Avalon	Avalon	Avalon	PLB,AXI(d),LMB,FSL,XCL	PLB,AXI(d),LMB,FSL,XCL

TABLE V. IMPLEMENTATIONS RESULTS OF OPEN SOURCE PROCESSORS AND NIOS II E/S/F ON AN ALTERA STRATIX V FPGA: ALMS, TOTAL REGISTERS (REGS), TOTAL BLOCK MEMORY BITS (T_{BMBs}), MAXIMUM FREQUENCY (F_{max}), CORE DYNAMIC/STATIC THERMAL POWER DISSIPATION (P_{CS}/P_{CD}), I/O THERMAL POWER DISSIPATION ($P_{I/O}$), ANALYSIS & SYNTHESIS CPU TIME ($T_{A\&S}$), AND FITTER CPU TIME (T_F).

	Amber23	*Amber25*	*LM32*	*S1*	*AltOR32*	*OR1200*	*LEON2*	*LEON3*	*Nios II/e*	*Nios II/s*	*Nios II/f*
ALMs	3073	5278	2141	39519	2014	2428	5678	1239	409	590	811
REGs	1875	3221	2468	55061	1754	1191	10546	1272	591	862	1347
T_{BMB}/ bits	152,575	305,664	52,736	270,432	69,632	156,288	72704	8,704	10,240	27,200	44,544
F_{max}(100 ℃)/MHz	84.73	96.11	179.92	52.32	94.64	110.3	158.5	212.27	367.51	260.48	297.89
F_{max}(-40 ℃)/MHz	83.08	96.15	171.14	56.15	91.99	107.4	158.55	209.95	337.04	250.38	288.68
P_{CS}/ mW	1738.22	1744.96	1735.37	1751.5	1735.08	1737.05	1738.29	1733.99	1733.48	1733.87	1734.5
P_{CD}/ mW	48.44	90.38	24.13	369.47	17.71	30.22	61.56	13.31	10.46	12.28	16.56
$P_{I/O}$/ mW	36.13	62.51	48.38	50.25	43.98	56.56	35.90	43.31	41.91	42.50	42.36
P_t/ mW	1822.78	1897.85	1807.87	2171.22	1796.77	1823.84	1835.75	1790.61	1785.85	1788.66	1793.42
$T_{A\&S}$	4min16s	7min3s	1min22s	12min52s	1min44s	1min12s	1min35s	38s	15s	15s	35s
T_F	20min35s	23min19s	29min17s	1h27min27s	29min35s	20min45s	34min36s	17min47s	25min35s	16min12s	17min19s

(T_{BMBs}) are mainly decided by total cache size. The values of T_{BMBs} vary directly with the total size of caches.

C. Implementations and Comparisons on Xilinx FPGA

The implementation results of open source processors and single- and dual-core MicroBlaze's on a Virtex-7 FPGA are available in Table VI. Dual-core MicroBlaze needs almost twice of the resources of the single-core version. MicroBlaze can be implemented with less LUTs and registers than all open source processors. LEON3 needs the least resources in all selected open source processors. The LUTs and registers of MicroBlaze is 41% and 58% of the averages of all open source processors.

The F_{max} of MicroBlaze is higher than all open source processors except LEON3. LEON3 with 7 stages of pipeline and Amber23 with a 3-stage pipeline achieve the highest and lowest frequencies in all open source processors, respectively. F_{max} of open source processor is proportionate to the pipeline depth.

In Table VI, the quiescent power (P_s) of all processors are nearly the same, because they are mainly decided by the chosen FPGA device. P_d is proportionate to LUTs, and P_d of MicroBlaze is less than all open source processors. The number of RAMB36E1's (N_{R36E1}) and the number of RAMB18E1's (N_{R18E1}) are mainly decided by total cache size. The values of N_R vary directly with the total size of caches.

V. DISCUSSIONS

Based on the implementation results in last section, we discuss the two different categories of processors, open source versus commercial.

A. Open Source or Not?

Open source processors are free to access the source code, while commercial ones cannot be referred to. Thus open source processors with excellent usability can be used to explore the strategies of optimizations for research and engineering. For applications requiring deep customizations, open source processors can be chosen to accelerate the process of project development.

B. Architecture versus Implementation

Commercial soft cores are superior to open source processors in logic utilization. For open source processors, logic utilization mainly depends on the architecture parameters such as number of registers, cache size, type of MMU and so on. ISA of open source processor also contributes to the logic utilization.

TABLE VI. IMPLEMENTATIONS RESULTS OF OPEN SOURCE PROCESSORS AND SINGLE- AND DUAL-CORE MICROBLAZE'S ON A XILINX VIRTEX-7 FPGA: LOOKUP TABLES (LUTS), REGISTERS (REGS), RAMB36E1'S (N_{R36E1}), RAMB18E1'S (N_{R18E1}), MAXIMUM FREQUENCY (F_{max}), DYNAMIC POWER (P_d), QUIESCENT POWER (P_s), SYNTHESIS & TRANSLATE TIME ($T_{S\&T}$), AND MAPPING & P&R TIME ($T_{M\&P}$).

		Amber23	**Amber25**	**LM32**	**S1**	**AltOR32**	**OR1200**	**LEON2**	**LEON3**	**MicroBlaze/single**	**MicroBlaze/dual**
LUTs		3274	6592	3281	56690	2742	3669	3674	2522	1491	3507
Regs		2306	3409	2233	38028	1629	1276	1559	1144	1114	2485
Mem	N_{R36E1}	8	16	4	48	2	4	2	0	2	2
	N_{R18E1}	4	8	0	17	1	2	3	2	0	0
F_{max}(25 ℃)/MHz		114.59	117.40	204.3	65.77	113.27	143.37	192.82	238.83	233.1	216.92
P_s/ mW		429	435	428	476	430	431	430	431	487	487
P_d/ mW		23	83	18	489	37	44	34	44	12	14
P_t/mW		452	519	446	965	467	475	464	474	499	501
$T_{S\&T}$		1min20s	1min45s	15s	11min41s	1min55s	1min37s	1min21s	1min20s	1min8s	2min3s
$T_{F\&P}$		5min45s	7min57s	4min24s	38min37s	7min39s	6min15s	4min42s	4min26s	3min15s	4min3s

Similarly, the maximum frequency of commercial soft cores which are optimized by venders are higher than open source processors. The larger pipeline depth of open source processor, the higher maximum frequency can be achieved. Amber25, OR1200 and Altor32 designed with a same pipeline depth reach a close maximum frequency.

The on-chip memory utilization depends on the total cache size. Open source processors configured with a larger cache need more FPGA on-chip memory blocks. Both Amber25 and S1 have a largest total cache size of 32 KBytes, as a result, they utilize most memory blocks.

C. Configurability and Conveniency

Commercial soft cores are all designed using user-friendly EDA tools from FPGA vendors. These tools provide not only soft cores with great configurability but also lots of optimized IPs. By appropriate configurations, systems can be constructed with a suitable soft core, minimal peripherals and highly optimized hardware.

Born with prominent "visibility", open source processors have the advantages of customization and exploration for optimization in engineering and research. What is more, engineers and researchers can experiment and verify any possible methods for their deep customized optimizations.

VI. CONCLUSION

FPGA-based SoC designs can be accelerated by reusing IPs. As the most important component, processor should be designed, verified, tested and shared in module level to increase the design productivity. Soft cores for FPGAs can be categorized into commercial and open source ones. This paper first gives an overview about open source processors. From the points of usability and stability, a selection process of open source processors is presented. The selected open source processors and existing commercial soft processors are implemented, compared and discussed.

ACKNOWLEDGMENT

This research is funded by National Natural Science Foundation of China (61271149, 61106033), and also supported by National Science and Technology Major Project of China (2013ZX03006004).

REFERENCES

[1] I. Kuon, R. Tessier, and J. Rose, "FPGA architecture: Survey and challenges," *Foundations and Trends in Electronic Design Automation*, vol. 2, no. 2, pp. 135–253, 2008.

[2] N. Tredennick and B. Shimamoto, "Prospects for reconfigurable systems," 2013.

[3] R. Wilson. (2011) CPUs in FPGAs: Many faces to a trend. [Online]. Available: http://www.edn.com/electronics-news/4369558/CPUs-in-FPGAs-many-faces-to-a-trend

[4] First time ever at ACM/SIGDA conference: FPGAs for mobile apps. [Online]. Available: http://www.eetimes.com/

[5] R. Saleh, S. Wilton *et al.*, "System-on-chip: Reuse and integration," *Proceedings of the IEEE*, vol. 94, no. 6, pp. 1050–1069, 2006.

[6] M. G. Kamte and M. P. Bhaskar, "Design and implementation of reusable processor independent IP core for SoC platform," *Int. J. on Recent Trends in Engineering & Technology*, vol. 5, no. 02, 2011.

[7] R. Viseur, "From open source software to open source hardware," in *Open Source Systems: Long-Term Sustainability.* Springer, 2012, pp. 286–291.

[8] S. Davidson, "Open-source hardware," *IEEE Design & Test of Computers*, vol. 21, no. 5, pp. 456–456, 2004.

[9] J. G. Tong, I. D. Anderson, and M. A. Khalid, "Soft-core processors for embedded systems," in *ICM 2006.* IEEE, 2006, pp. 170–173.

[10] P. Yiannacouras, J. G. Steffan *et al.*, "VESPA: Portable, scalable, and flexible FPGA-based vector processors," in *CASES 2008*, pp. 61–70.

[11] A. Prakash, S.-K. Lam, A. K. Singh, and T. Srikanthan, "Rapid design exploration framework for application-aware customization of soft core processors," in *FPL 2009.* IEEE, 2009, pp. 539–542.

[12] [Online]. Available: http://www.opencores/projects

[13] [Online]. Available: http://www.en.wikipedia.org/wiki/Soft_microprocessor

[14] R. Fontana, B. M. Kuhn *et al.*, *A Legal Issues Primer for Open Source and Free Software Projects.* Software Freedom Law Center, 2008.

[15] [Online]. Available: http://www.en.wikipedia.org/wiki/OpenCores

[16] OSHW community survey 2013. [Online]. Available: http://www.oshwa.org/oshw-community-survey-2013

[17] Amber 2 core specification. [Online]. Available: http://opencores.org/

[18] LatticeMico32 processor reference manual. [Online]. Available: https://www.yumpu.com/en/document/

[19] Simply RISC S1 core specification. [Online]. Available: http://www.srisc.com/download/

[20] [Online]. Available: http://opencores.org/project,altor32

[21] OpenRISC 1200 IP core specification. [Online]. Available: http://openrisc.net/or1200-spec.html

[22] LEON2 processor users manual. [Online]. Available: http://www.dit.upm.es/~str/ork/documents/

[23] Aeroflex Gaisler. (2010) SPARC V8 32-bit processor LEON3/LEON3-FT companion core data sheet, march, version 1.1. [Online]. Available: http://prod.sandia.gov/techlib/