ARM-Based SoC Prototyping Platform Using Aptix

Chang-An Tsai, Yu-Te Chou, Yu-Tsang Chang, Lan-Da Van, and Chun-Ming Huang
E-mail: {changan, steven, carven, ldvan, cmhuang}@cic.org.tw
Chip Implementation Center (CIC), National Applied Research Laboratories,
No. 26, Prosperity Rd. 1, Science Park, Hsinchu, Taiwan, R.O.C.

ABSTRACT: With the consecutive progress in the process technology and state-of-the-art design methodology, multi-million gate-counts system-on-a-chip (SoC) design can be realistically fulfilled. However, the rapid prototyping verification and integration for SoC designs reveal a big challenge. In this paper, we propose an ARM-based SoC prototyping platform using Aptix to accelerate the verification and integration process. The developed infrastructure in the prototyping platform consists of an ARM920T core module, a refined EASY system, an AHB-compliant slave component and many reconfigurable function blocks. On one hand, before silicon proof, since ARM Inc. only provides the design sign-off model for the ARM processor core, designer needs to spend lots of time to simulate and verify the ARM-based SoC design with software-based simulation method. On the other hand, due to many available intellectual property (IP) designs from the public domain and academics, it is desired to shrink the integration time via a common platform. Thus, the proposed ARM-based SoC prototyping platform not only speeds up ARM-based SoC system verification using real hardware acceleration instead of the design sign-off model, but also provides the integration facility via AMBA bus for distinct IP components. Importantly, the ARM-based SoC prototyping platform is very suitable for computation-intensive multimedia applications such as JPEG- and MPEG-oriented SoC designs. To demonstrate the performance and functionality of this platform, we use JPEG as our reference application to expose how JPEG can be rapidly prototyped via hardware/software codesign and accelerated hardware simulation. Through ARM-based SoC prototyping platform, the experimental results show that the system verification can be significantly speeded up by a factor of 1410 compared with that using the design sign-off model. Furthermore, National Chip Implementation Center (CIC) has built up the web-based mechanism including remote uploading as well as simulating designs and tutorial training. Through this channel, more education and research opportunities can be provided to the professors and students.

1. INTRODUCTION

With the advent of semiconductor process and EDA tools technology, IC designers can integrate more functions such as RISC processor, DSP processor, memory, etc. into a single chip. The complexity of a system-on-a-chip (SoC) design is high enough such that the verification time becomes a critical issue. To cope with such high-complicated SoC design, the feasible solution is to provide a generic platform that can easily integrate these distinct reusable intellectual properties (IP's). The IP's are combined with standardized on-chip interconnects protocol such as Advanced Microcontroller Bus Architecture (AMBA) [1] for the ARM and CoreConnect™ [2] for the PowerPC. Due to the off-the-shelf and low-power features, CIC selects AMBA University Kit (AUK) [3], which includes ARM920T processor and AMBA bus, as our SoC/IP platform and release AUK to academia in Taiwan. The ARM920T is a powerful RISC processor and AMBA enables reusable IP's, such as CPU, peripherals, and memory controllers to be connected together through the same bus protocol. AMBA specification defines two buses. One is a 32-bit Advanced High Performance Bus (AHB) with the feature of high-speed, high-bandwidth, and multi-master. The other is a 32-bit low-power, low-speed, and un-clocked bus that is referred to as Advanced Peripheral Bus (APB). APB bus plays a role of the secondary bus that communicated with AHB via a bridge. The AUK provides simulation and development environment for AMBA-based system and modules. There exist various AHB components and APB peripheral devices. In order to verify the integration methodology for SoC design, we embedded these distinct IP's within AUK as an AMBA AHB slaves and the ARM920T core processor within AUK as an AMBA AHB master used in this simulation. The system-level simulation aims to verify the correct IP behaviour before the silicon proof. AUK provides an alternative platform for ARM-based SoC design.

However, AUK only can provide designers ARM processor in sign-off model and AMBA circuits in RTL model. Such RTL-level simulation would take a lot of time to verify the functionality of multi-million ARM-based SoC design. Besides the pure RTL-level simulation, there exist the hardware/software co-simulation tools like Mentor-Seamless CVE [4]. Simulation time can be alleviated by using C-model instead of sign-off model of the ARM processor. However, the rest parts of the SoC design in RTL model have to be verified by RTL simulator. Hence, the simulation time cannot be competed with that of FPGA-based prototyping system such as Aptix [5]. Aptix provides a reconfigurable prototyping platform for SoC design as shown in Fig. 1. It consists of 4 high-density Xilinx Virtex-II 6000 FPGA [6], 2 SRAM modules, 1 LED module, and 1 ARM920T core module [7].

Fig 1. Aptix reconfigurable prototyping platform.

Up to date, these doesn't exist high-integration and low-verification-time SoC platform for academia in Taiwan. Thus, in this article, we propose an ARM-based SoC prototyping platform using Aptix to accelerate the verification and integration process. The developed infrastructure in the prototyping platform consists of an ARM920T core module, a refined example AMBA system (EASY), an AHB-compliant slave component and many reconfigurable function blocks. The organization is as follows: We present our ARM-based SoC prototyping platform in Section 2, and show the CIC ARM-based SoC prototyping platform in Section 3. Section 4 details how to run a JPEG encoding application on CIC ARM-based SoC prototyping platform and corresponding experimental and comparison results are presented in Section 5. Finally, we conclude the paper and future work in Section 6.

2. DEVELOPMENT FLOW OF ARM-BASED SOC PROTOTYPING PLATFORM

Fig. 2 reveals the development flow of ARM-based SoC prototyping platform. In the first step, the specification and the essential functionality of peripherals have to be identified in the SoC system. The hardware blocks and software models as exposed in Fig. 2 are compliant with AMBA specifications and ARM system architecture, respectively. Furthermore, we illustrate the hardware design flow and software development flow for ARM-based SoC prototyping platform as following:

A. HARDWARE DESIGN FLOW:
The hierarchical RTL design for SoC system including the ARM920T core module, SRAM, AMBA system and soft IP's are described in structural Verilog-HDL. Next, the RTL designs are synthesized to generate the EDIF netlists via Synplify_Pro [8]. In this step, the RTL designs for SoC system are partitioned into several modules including modified soft IP's and custom blocks that are mapped to hardware modules such as FPGA module, SRAM and ARM920T core module on Aptix platform. Finally, the EDIF netlists are imported into System Explorer to configure the Aptix platform. In this step, the FPGA place & route and FPIC routing [5] are performed to integrate all hardware modules on Aptix platform. Furthermore, we can probe the internal signals on the platform by using the Agilent logic analyzer.

B. SOFTWARE DEVELOPMENT FLOW:
The software activities can be developed in parallel with the hardware design. First, the application source code and hardware drivers are developed until system analysis is done, where all peripherals and hardware function blocks are memory mapped. The source codes are coded as assembly and/or C/C++ language. Next, those source codes are compiled to several object files, and then the object files are linked to generate executable AXD code for ARM920T processor.

After completing the hardware design and software development flows, the hardware and software can be verified simultaneously. As shown in Fig 2, the executable file is run at the ARM920T core module through the MultiICE cable. We debug the software in AXD environment and probe the internal hardware signal by using the logic analyzer. Therefore, we can complete the SoC co-verification in such platform.

3. CIC ARM-BASED SOC PROTOTYPING PLATFORM

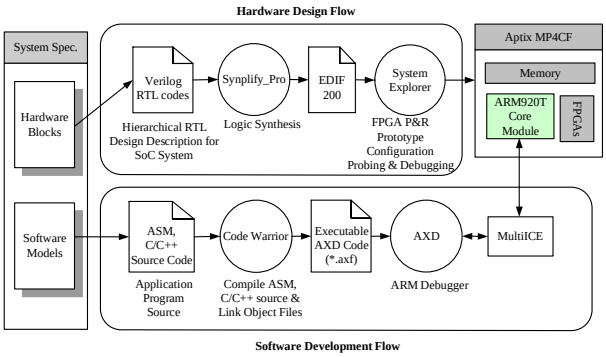

Fig.2. Development/design flow of ARM-based SoC prototyping platform.

Based on the development flow as above mentioned, CIC proposes a reference platform that called as CIC ARM-based SoC prototyping platform to reduce system development and verification time. As depicted in Fig. 3, the CIC ARM-based SoC prototyping platform consists of ARM920T core module, SRAM, LED module and soft cores in Xilinx FPGAs. The soft cores feature the parameterized Verilog-HDL code which can be synthesized and modified. The AUK obtained from ARM Inc. provides a set of parameterized RTL codes that realize an EASY system. Some of RTL codes involving MuxS2M, SMI, MuxM2S, IntMem, Arbiter and Decoder circuits in AUK have to be modified for the SoC prototyping platform.

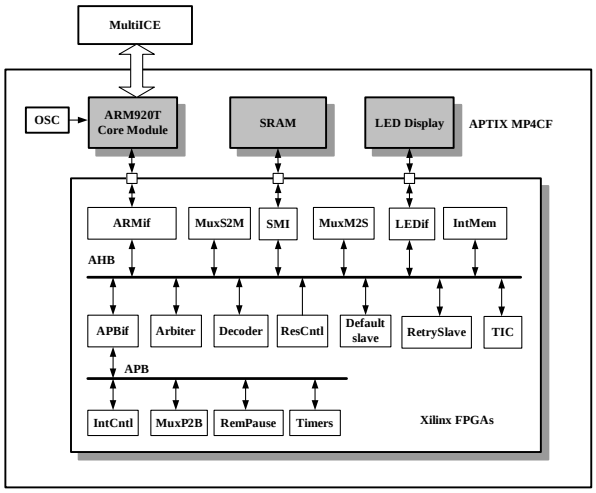

Fig. 3. The block diagram of CIC ARM-based SoC prototyping platform.

The AMBA in the reference design contains an AHB bus and an APB bus. Currently there exist several hard and soft cores that need to connect to AHB bus, such as the ARM920T core module, test interface controller (TIC), arbiter, decoder, SRAM, LED modules, and internal memory. There are also several soft cores connecting to APB bus such as interrupt controller and timers. Except the ARM920T core module, SRAM and LED modules, all reconfigurable soft cores connecting AHB or APB are implemented in Xilinx FPGAs. In this platform, we can integrate any AMBA-compliant IP's, DSP processors or peripherals to build up the individual SoC system.

In the reference design, there are two masters and several slaves on the AHB bus. The arbiter function grants the master ARM920T or TIC to access the bus. The ARM920T is the default master and connects to AHB bus through the ARM

interface (ARMif) that is used to map core module signals to the AHB bus. The TIC is a state machine that provides a system test for AMBA AHB bus. It reads test pattern and address data from the external data bus to check the internal components separately. The internal memory is a littile-endian model of 1KBx32 SRAM, which is addressable in byte, half-word or word mode. The static memory interface (SMI) is a programmable memory interface that connects the external memory bus to AHB. The default slave is responsible for transfers whose address is located on the undefined regions of memory. The retry slave is a rudimentary module that is used to demonstrate how to build an AHB slave. The LED interface is a simple AHB slave module that is utilized to connect LED module to AHB. The APB interface (APBif) that is the only master on APB bus plays a role of the bridge between AHB and APB bus and operates as AHB slave. The memory mapping of the aforementioned AHB slaves or peripherals is defined in advanced for the designated CIC ARM-based SoC prototyping platform. The designer can modify the memory mapping of AHB slaves or peripherals to adapt to distinct application SoC platform.

The memory mapping of CIC ARM-based SoC prototyping platform is depicted in Fig. 4. The address range from 0x00000000 to 0x10FFFFFF is defined as local SSRAM, SDRAM and core module control registers [7]. The local SDRAM could be accessed only by ARM processor core module at address from 0x00100000 to 0x0FFFFFFF. The maximum SDRAM size of 256 MB is already fitted on core module but the lowest 1 MB is not available. This is because the lowest 1MB is hidden by the local SSRAM. The core module registers allow ARM processor to determine its environment and to control the core module operations. The core module has a fixed memory mapping; hence, other slaves or peripherals cannot be defined in this region. The address range from 0x11000000 to 0xFFFFFFFF is defined as external RAM, retry slave, APB slave, etc. The external RAM is connected to SMI, which is able to handle up to 256-MB SRAM and ROM. CIC furnishes two 512Kx40 SRAM modules that can be used as external RAM.

Through the CIC ARM-based SoC prototyping platform, users can quickly modify the RTL codes for different benchmarks and rapidly accomplish hardware and software co-simulation. Moreover, CIC provides 4 advanced Xilinx Virtex Ⅱ-v6000 FPGAs to implement large-scale soft IP's. The CIC ARM-based SoC prototyping platform is built for sophisticated SoC design using multiple FPGA devices. Most importantly, an expandable and flexible SoC platform can be easily obtained while more FPGA or other hardware modules are demanded.

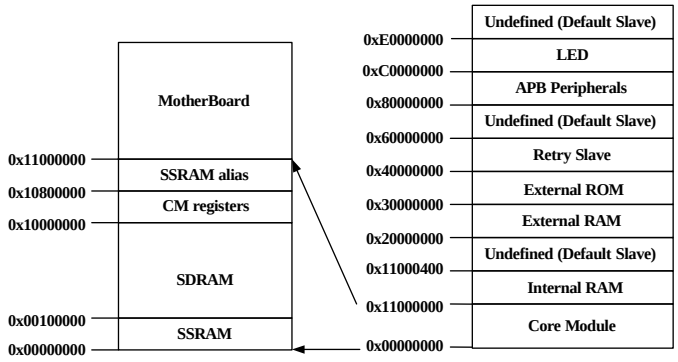

Fig. 4. Memory mapping of CIC ARM-based SoC prototyping platform.

4. A JPEG APPLICATION ON CIC ARM-BASED SOC PROTOTYPING PLATFORM

To demonstrate the effectiveness and performance of the proposed ARM-based prototyping platform, the computation-intensive JPEG SoC design is used to validate the functionality through the millions of input patterns and running cycles. For such multimedia-specific application, many computation-intensive tasks are performed by dedicate hardware and others are executed on the processor. Thus, the real time operation can be achieved. Without loss of generality, we applied the JPEG application as shown in Fig. 5 to the CIC ARM-based SoC prototyping platform. It is known that the JPEG encoding is a lossy compression scheme based on DCT computation. The JPEG program in C language carries out the RGB image compression and it is intended to execute on ARM processor. Through the profiling of JPEG, it is found that 2-D DCT/IDCT has the most computation load. In order to improve the simulation performance, the 2-D DCT/IDCT that implemented in Xilinx FPGA is designed as hardware accelerator.

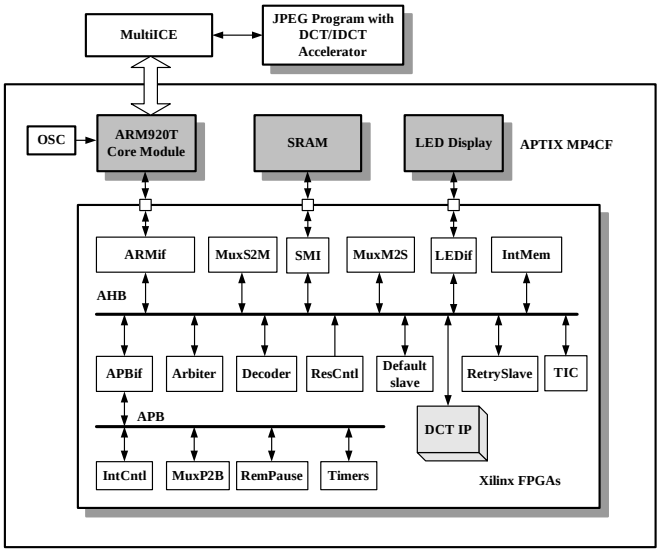

Fig. 5. A JPEG application applied to CIC ARM-based SoC design platform.

The 2-D DCT/IDCT IP can be easily integrated with CIC ARM-based SoC design platform by operating as a slave on AHB bus through an AMBA-compliant wrapper. So as to operating correctly, we define memory mapping address as 0x60000000~0x70000000. ARM processor acts as a master on AHB bus; therefore, ARM920T can issue the read and write data or control signal to 2-D DCT/IDCT IP according to the bus address. ARM uses the polling mechanism to know whether the DCT/IDCT completes the job or not. Once the DCT/IDCT completes the computation, ARM retrieves the results from DCT/IDCT through AHB bus.

5. EXPERIMENTAL AND COMPARISON RESULTS

The comparison results in terms of the number of instruction cycles and simulation time for JPEG encoding among four different cases are exposed in Table 1. The instruction count and simulation time mean how many ARM instruction cycles needed and the time for encoding an RGB image, respectively, in JPEG encoding. A RGB picture with the resolution of 704x416 is treated as the tested pattern in this experiment. JPEG encoding is implemented in C language, and it is

compiled and linked to generate the executable AXD code for ARM920T processor. As shown in Table 1, totally 169,781,925 ARM instructions are needed to complete JPEG encoding. However, in case the DCT IP is used to accelerate, only 64,921,783 ARM instructions are obtained. The experimental environment is described in Table 2.

Table 1. Comparison of experimental results among four cases.

Case	Instruction Count	Simulation time (sec)
RTL Simulation (AUK)	169,781,925	81,787
RTL Simulation (AUK + DCT Acceleration)	64,921,783	82,887*
CIC SoC Platform	169,781,925	139
CIC SoC Platform with DCT Acceleration	64,921,783	58

* Since bus transfer action is frequent, the simulation time of RTL Simulation (AUK + DCT Acceleration) is larger than that of RTL Simulation (AUK).

Based on Tables 1 and 2, the experimental results are demonstrated as below.

1. RTL Simulation (AUK): The JPEG encoder is executed by software and implemented in EASY environment. The target is simulated via Cadence Verilog-XL in Sun Fire 6800 workstation. The simulation time of 81,787 seconds is acquired.
2. RTL Simulation (AUK+DCT Acceleration): The JPEG encoder is executed by software except the DCT computation that is assigned to DCT IP. The target is implemented in EASY environment and simulated via Cadence Verilog-XL in Sun Fire 6800 workstation. The required simulation time is 82,887 seconds.
3. CIC SoC platform: The JPEG application is implemented by using the CIC ARM-based SoC platform. The JPEG encoder is executed by software on ARM920T core module via MultiICE and the simulation time is 139 seconds.
4. CIC SoC platform with DCT Acceleration: The JPEG application is implemented by CIC ARM-based SoC platform. The JPEG encoder is executed by software except the DCT computation, where the DCT computation is implemented in Xilinx FPGA. The software is executed on ARM920T core module via MultiICE and the simulation time is 58 seconds.

As the result in Table 1, the number of instructions of JPEG encoding with DCT IP acceleration is improved 62% compared to JPEG encoding without DCT IP acceleration. On the other hand, compared with AUK platform, the simulation time of CIC SoC Platform with DCT Acceleration can be speeded up by a factor of 1410. Therefore, the simulation result could be obtained in minutes instead of days even weeks.

6. CONCLUSION AND PERSPECTIVE

The CIC ARM-based SoC prototyping platform is presented in this paper including the design flow, design methodology, a design example and testing environment. By using the CIC ARM-based SoC prototyping platform, it could speeds up verification flow for complex SoC design and increase design reliability before silicon proof. CIC has released web-based mechanism including remote uploading as well as simulating designs and tutorial training for the professors and students since May 2004 as shown in Fig. 6. To view the details, please see http://www.cic.org.tw. Future work will focus on the multiprocessor-based SoC rapid prototyping platform development.

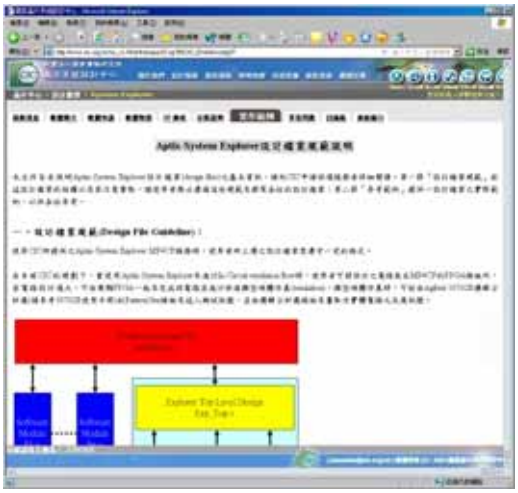

Fig. 6. The Aptix-System Explorer tutorial website constructed by CIC (in traditional Chinese version)

REFERENCE

1. *"AMBA Specification Rev 2.0"*, ARM Ltd., (1999)
2. *"The CoreConnect Bus Architecture"*, www.ibm.com, (1999)
3. *"The AMBA University Kit Technical Reference Manual"*, ARM Ltd., (2001)
4. *"Mentor Graphics-Seamless CVE User and Reference Manual"*, www.mentor.com, (2001)
5. *"Aptix-MP4CF System Explorer User and Reference Manual"*, www.aptix.com, (2000)
6. *"Xilinx-Platform FPGA Virtex-II datasheet"*, www.xilinx.com, (2000)
7. *"Integrator /CM920T-ETM User Guide. "*, ARM Ltd., (2000)
8. *"Synplicity-Synplify Pro User and Reference Manual"*, www.synplicity.com,(2001)

Table 2. Verification environment among four cases.

Case	Simulation Environment	Simulator/ Emulator	DCT Acceleration
RTL Simulation (AUK)	Sun Fire 6800, 20 UltraSPARCIII CPU, 40 GB memory, 2 TB storage	Cadence Verilog-XL	No
RTL Simulation (AUK + DCT Acceleration)	Sun Fire 6800, 20 UltraSPARCIII CPU, 40 GB memory, 2 TB storage	Cadence Verilog-XL	Yes
CIC SoC Platform	Aptix-MP4CF System Explorer	MultiICE	No
CIC SoC Platform with DCT Acceleration	Aptix-MP4CF System Explorer	MultiICE	Yes