IP Core Design

Homework 3

Virtual Prototype
Of
Baseline JPEG Software Encoder

Instructor: Prof. Chein-Wei Jen
Announcement: 2001.11.26

Many problems can arise during the system integration process. Moving the system integration phase forward in the design cycle would help in detecting these integration problems earlier. This can be achieved by creating a HW/SW co-verification environment (Figure 1 (a)) early in the design cycle. Soft (or virtual) prototype (Figure 1(b)) is one kind of such environment that it is a software design representation of the design being verified. Another example of such environment is Mentor Graphics Seamless Co-Verification Environment (CVE). In this homework, we make use of virtual prototype that the processor is modeled as an Instruction Set Simulator (ISS) and hardware functional blocks (especially those IPs you will do synthesizable HDL design in homework 4) are represented with C models. The virtual prototype allows designers to do the following:

- Make trade-offs by modifying system parameters and checking the results
- Test interrupt handlers
- Develop and test drivers for your IPs
- Test the correctness of the application algorithms.

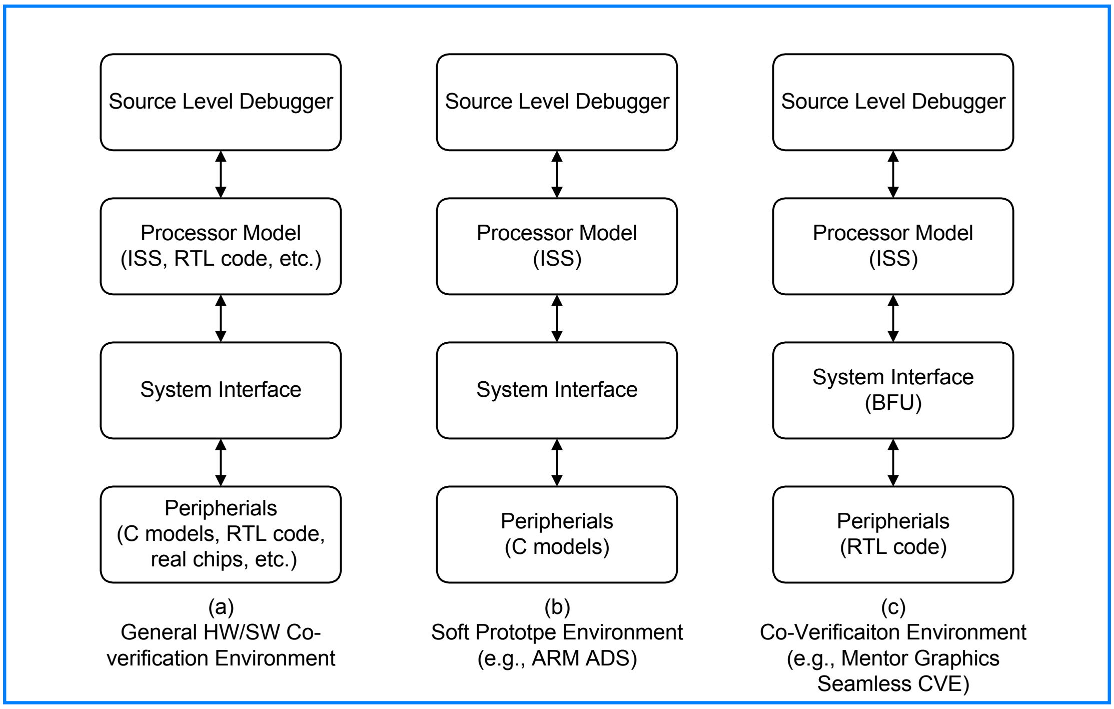
Source Level Debugger
Processor Model (ISS, RTL code, etc.)
System Interface
Peripherials (C models, RTL code, real chips, etc.)
(a)
General HW/SW Co-verification Environment
Source Level Debugger
Processor Model (ISS)
System Interface
Peripherials (C models)
(b)
Soft Prototpe Environment (e.g., ARM ADS)
Source Level Debugger
Processor Model (ISS)
System Interface (BFU)
Peripherials (RTL code)
(c)
Co-Verificaiton Environment (e.g., Mentor Graphics Seamless CVE)

Figure 1. HW/SW co-verification environments.

The software/firmware developed using the virtual prototype can be reconfigured for emulation and downloaded through an ICE to the rapid prototype or target hardware system for testing. Two of the limitations of the virtual prototype we have to know are:

- **Accuracy of models**. The hardware models are functionally correct but not pin-accurate. Also, it is often difficult to model exact cycle-accurate hardware designs.
- **Synchronization**. It is usually difficult to resolve the synchronization requirements of the peripheral data dependencies.

As the design evolves it is important to keep the software model in step so that the software development is based on the most accurate estimates of timing that are available. If the timing assumption built into the original software model is proved impossible to meet in the course of the detailed hardware design (e.g., homework 4), consistency between the models should be maintained.

Using the ARMulator, it is possible to build a complete, clock-cycle accurate software model of a system including MMU, physical memory, peripherals, and OS. Since this is likely to be the highest-level model of the system, it is one of the best places to perform the initial evaluation of design alternatives before detailed RTL design. Once the design is reasonably stable, hardware development will probably move into a timing-accurate design environment, but software development can continue using the ARMulator-based model.

Deliverable

Your deliverable has to include:

1. Report that describes your idea and result.
2. Source code of your JPEG encoder, hardware models, and drivers.
3. All setting and information required for regenerating the result shown in your report.

Design and model those portions of JPEG baseline encoder that you want to improve the coding process through hardware accelerators. All the details of the registers, memory addresses of these registers, bit definitions, and drivers (initialization and interrupt behavior) of your IP(s) should be included in the C models. To facilitate testbench migration from this functional level to lower levels, use the bit-true, fixed-point representations in the functional testbench.

Your report should include the description of the functionality of the IPs, the programmer's model (e.g., Section 4.6 of Integrator/AP User Guide), the behavior of the driver (initialization and interrupt behavior), and the coordination of hardware and software.

State your approaches, key ideas and results clearly and formally, and void redundant description. Your report can be written in Chinese or English. However, make sure your report is readable. A manual report won't degrade your score, unless it is scrabbled.

If your JPEG encoder is modified from an existing reference code, please acknowledge in the last section of your documentation that you've used the reference code.

Important Date

Due : 5:00 p.m. Dec. 10, 2001

For more information

- The contents of this document: Kun-Bin Lee
- ARM development tools: contact the TA with the number = your team number %5

No.		Email	Ext.
0	Kun-Bin Lee	kblee@yankees.ee.nctu.edu.tw	54225
1	Yuan-Chung Lee	yzlee@yankees.ee.nctu.edu.tw	54225
2	Jih Yiing Lin	jinlin@yankees.ee.nctu.edu.tw	54243
3	Nelson Yen-Chung Chang	ycchang@yankees.ee.nctu.edu.tw	54243
4	Tzung-Shian Yang	tsyang@yankees.ee.nctu.edu.tw	54243