SoC Design Laboratory
Term Project Part I
Baseline JPEG Software Codec

Instructor: Tian-Sheuan Chang
Announcement: 2004.03.09
Due date: 2004.04.06

Overview of term project

In term project, we will take the baseline JPEG codec in ARM-based platform system as an example to practice the design flow in SoC. We divide the project into three parts, and the goal of each part is described as follow.

Part I: Design a baseline JPEG software codec in C/C++ and port it to ARM core, which can be ARM7TDMI, ARM720T, or ARM922T. First of all, run the C/C++ design in ARM instruction set simulator, ARMulator. After that, verify your design in the target environment, ARM Integrator. You need to refine your C/C++ design under the consideration of your target environment.

Part II: Make use of virtual prototype to integrate/verify the hardware and software of baseline JPEG codec system. Then implement the target block as an AMBA AHB-compliant soft IP.

Part III: Verify your soft IP in target environment.

In part I, you can either write your own design or modify an existing reference code. Be aware of the differences between the ARMulator environment and the target platform (i.e., ARM development boards). Also, the data structures and the partition of function calls should be carefully defined because portions of your design will be implemented as hardware components in part II and mapped to FPGA in part III.

Term Project Part I

The term project of this course begins with a reference baseline JPEG software codec [1](The JPEG encoder/decoder specification can refer to [2][3]). Then try to optimize this design under the consideration of ARM core's features to get better performance, fewer memory requirements (includes the program itself and the temporal memory for data processing), or even fewer power consumption without sacrificing the image quality. The example approaches may possibly be helpful in such optimization:

- Select or modify the algorithms or the code segments used in JPEG to fit to ARM's architecture. By taking constraints of the ARM core hardware resources into consideration, some algorithms may be more suitable for ARM core than others. An example of such consideration can be found in [4].
- Create SIMD operations. Though current ARM architecture has no specific instructions to support single-instruction, multiple-data (SIMD) operation, certain SIMD operations can be synthesized using a sequence of normal ARM

instructions [5].
- Use ARM/Thumb mode for different code segments.

Next, port your own software JPEG codec to ARM hardware development system. To access the BMP file, you may load the portion of the picture (e.g., line by line or block by block) from the host, process the data, and write result to the host. If the performance of your design is constrained by the file I/O, you may load the whole picture into memory first and then process it.

You have to take the following considerations into account:
- performance & constraint of different types of memory (SSRAM, SDRAM, and Flash) with cache disabled.
- data alignment and data layout (in which type of memory)
- available data bus and memory bandwidth

[1] http://twins.ee.nctu.edu.tw/courses/soclab_04/index.html
[2] http://www.twins.ee.nctu.edu.tw/
[3] JPEG image compression FAQ, part 2/2, http://www.faqs.org/faqs/jpeg-faq
[4] Tadashi Sakamoto and Tomohiro Hase, "Software JPEG for a 32-bit MCU with dual issue," IEEE Transactions on Consumer Electronics, Vol. 44 Issue: 4, Nov. 1998, pp. 1334 -1341.
[5] Alan Lewis and Paul Carpenter, "Optimizing digital video codecs in ARM cores," EE Times, Sep. 20, 2001.

Deliverable

Your deliverable has to include:

1. Report that describes your idea, result, and improvement in ARMulaor and ARM integrator respectively.

   ARMulator:
   - You have to clearly point out the differences of code segments between reference design and your optimized design. Any improvements of the results outside these code segments will not be recognized. You also need to explain and analyze the superiority of the optimized design over the workable one. You also need to prove or convince TA of your claim.
   - Summarize your improvement of the memory requirement, profiling, and statistics in table format. Separate the result of statistics of file I/O routines from JPEG kernel.
   - Compare and discuss the results of ARM7TDMI, ARM720T, and ARM922T.

   ARM integrator:
   - Point out the differences of code segments between your design in ARMulator and ARM integrator, and explain the reasons.
   - Performance: use the timers/counters or the time function to record the time your program spends and show it on the host console. Please annotate that

cache is enable or not.

- Memory requirement: describe your memory organization for each stage of data processing in detail and explain how it works. Evaluate the maximum memory requirement during your program. Note that the same memory space can be shared with different data structures if their life times are not overlapped. Also, the memory requirement for the program itself and variables have to be mentioned if you modify your program.
- Compare and discuss the results in ARMulator and ARM integrator.
- Evaluate which parts should be improved.

2. Source code of your design and all setting and information required for regenerating the result shown in your report
3. Please acknowledge in the last section of your report that you've used the reference code and related documents.

State your approaches, key ideas and results clearly and formally, and avoid 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.

For more information

- The contents of this document: Hui-Cheng Hsu, huijane@twins.ee.nctu.edu.tw