國立交通大學

電機與控制工程學系

碩 士 論 文

以平台架構為基礎之

JPEG2000 影像編碼矽智產設計

Platform-based Design and IP

Implementation of JPEG2000 Image Coding

研 究 生：鄭晏阡

指導教授：吳炳飛　博士

中 華 民 國 九 十 四 年 七 月

以平台架構為基礎之 JPEG2000 影像編碼矽智產設計

Platform-based Design and IP Implementation of JPEG2000 Image Coding

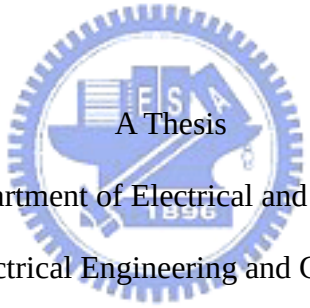
研 究 生：鄭晏阡　　Student：Yen-Chian Cheng

指導教授：吳炳飛 博士　　Advisor：Dr. Bing-Fei Wu

國 立 交 通 大 學

電 機 與 控 制 工 程 學 系

碩 士 論 文



A Thesis

Submitted to Department of Electrical and Control Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Electrical and Control Engineering

July 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年七月

AWARDS

- 本論文參與由教育部/國科會所主辦「九十二學年度第六屆大學院校矽智產(SIP)設計競賽」榮獲 SOFT IP 不定題組"佳作"。


獎牌

教育部
國科會
九十二學年度大學
校院矽智產(SIP)設計競賽
鄭晏阡、黃琪文、陳沛君、黃培恭
君榮獲不定題組SOFT IP類佳作
特頒此牌以資鼓勵

教育部部長
國科會主任委員

中華民國九十三年七月九日

AWARDS

- 本論文參與由旺宏電子/旺宏基金會所主辦「2004 年第五屆半導體設計與應用大賽」榮獲設計組”優等”。

以平台架構為基礎之 JPEG2000 影像編碼矽智產設計

學生：鄭晏阡　　指導教授：吳炳飛 博士

國立交通大學 電機與控制工程學系

摘　要

JPEG2000 影像壓縮技術可以提供我們高畫質，高解析度，高壓縮倍率的影像；但是相對地，其演算法比其他的靜態影像壓縮法複雜得多。所以將 JPEG2000 的演算法用硬體實現有其必要性和實用性。

JPEG2000 其中一項重要的特色就是精確的壓縮位元率控制，但是複雜的演算法不太適合硬體實作。本論文提出一個位元率控制的方法，非常適合實現在硬體上。此架構並且適用於傳統的 DWT 與本實驗室自行開發的 QCB-Based DWT 中。

在 SoC (System-on-a-Chip)的時代來臨之後，TTM (Time-To-Market)的觀念越來越為人所重視；為了解決這個問題，設計可以 Reusable 的 IP (Intellectual Property)就變得格外重要，尤其矽智財又牽涉到 SoC 的實現，因此 IP-Based Design 便孕育而生。但更進一步的，如何快速且有效率的整合這些 IPs，Platform-Based Design 才是重要關鍵。

本論文最後將 JPEG2000 實現成 IP 的形式，使用者可以依照自己的需求，參數化該 IP，該 IP 產生器便會產生出符合使用者規格的 JPEG2000 編碼器包含 RTL code、testbench、synthesis scrip 等；除此之外，也一併提供了包括 ASIC 和 FPGA 兩種設計流程中整套 tool chain 所有的 script files，增進設計自動化且縮短了整個設計的時程。

Platform-based Design and IP Implementation of JPEG2000 Image Coding

Student: Yen-Chian Cheng Advisor: Dr. Bing-Fei Wu

Department of Electrical and Control Engineering
National Chiao Tung University

ABSTRACT

JPEG2000 image compression technology can provide higher quality, higher resolution and higher compression ratio of an image. But JPEG2000 algorithm is more complex than the other still image compression relatively. It is necessary and usable to develop JPEG2000 with the hardware realization.

One of the important features of JPEG2000 is the accurate rate-control scheme but it is not suitable for hardware design. A rate-control scheme which is easy for hardware implementation is proposed in this thesis. The architecture is suitable for the traditional DWT and QCB-based DWT.

The coming of SoC (System-on-a-Chip) era has been urging the emphasis of the people towards the TTM (Time-To-Market) concept. To solve this problem, design of reusable IP (Intellectual Property) is more important. Especially IP is concern with realization of SoC. That is why IP-based design was appeared. Furthermore, the platform-based design is the key that how to integrate these IPs rapidly and efficiently.

Finally, we design JEPG2000 not only an IC but also an IP. Users can configure this IP with their requirements, and the JPEG2000 ModelGen will generate the RTL code, testbench and synthesis scrip user-defined. Moreover, the ModelGen also generates all script files of tools which in ASIC and FPGA design flow. It can decrease the development time and increase design automation.

ACKNOWLEDGEMENTS

首先一定要感謝台北科技大學專題老師**李文達**博士，是他帶學生進入 IC 設計的領域。進到了交大研究所，承蒙指導教授**吳炳飛**博士這兩年來對學生的提攜鼓勵及諄諄教誨，在 CSSP 裡不只學到了更進階的設計技巧，也學到了許多系統的觀念，使學生在專業知識及求學態度各方面獲益良多。感謝口試委員交大資工系**蔡淳仁**教授、其樂達科技**莊俊雄**博士，抽空前來參與學生論文口試及對本論文提供寶貴的意見與建議，得以使本論文更臻周延與完整。其中特別感謝交大電工系**黃俊達**老師，您開設的課程 IP Core Design 對學生論文幫助實大。

感謝 CSSP 的學長姐：**強哥**、**重甫**、618 的伙伴**世孟**、**馬哥**、**坤卿**等，有了你們使得 618 更顯溫暖；兩位超強的學姐**琪文**、**沛君**，跟妳們合作期間真的很愉快。同窗**俊傑**、**培恭**、**映伶**、**東龍**、**堯俊**、**則全**，很榮幸跟你們一起當同學。以及學弟妹**宗堯**、**小熊**、**元馨**，跟你們在一起的這一年很快樂。

另外感謝我的好友**士軒**、**弘忠**、**文佳**、**瑋柏**等及女友**琬貞**，很難得可以交到你們這群好朋友。最重要的當然要感謝我的爸、媽與妹妹，使我能在學習過程中無後顧之憂，晏阡一定會回報這無盡的恩情。

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

The need for high-performance image compression is becoming greater and greater as digital imagery finds its way into many areas of everyday life. While 20 years ago digital images were seldom used, today they are used in many diverse applications, including multimedia technology, digital photography, Internet viewing, image archiving, and medical imaging. In order to transmit and store digital images, the images must be compressed; otherwise, each image would require a huge amount of memory. Because digital images are used so extensively, image compression technology has surged forward in recent years. Much work has been put into developing an image compression standard that will have a broad range of functionality, as well as an excellent compression rate. The fruit of this work is an image compression standard known as JPEG2000.

In this chapter, the concept of JPEG2000, which is a new generation of still image compression standard are described in Section 1.1. Our proposed JPEG2000 hardware implementation is shown in Section 1.2, followed by the organization of this thesis.

1.1 Standard JPEG2000 Overview

JPEG2000 is a new standard for still image compression. It not only has better compression performance than the existing JPEG standard, but also provides many new features. The features included in JPEG2000 are listed below:

- Lossless and lossy compression.
- High compression ratio than current JPEG.
- Progressive transmission.
- Region-Of-Interest (ROI) decoding.
- Better error resilience.

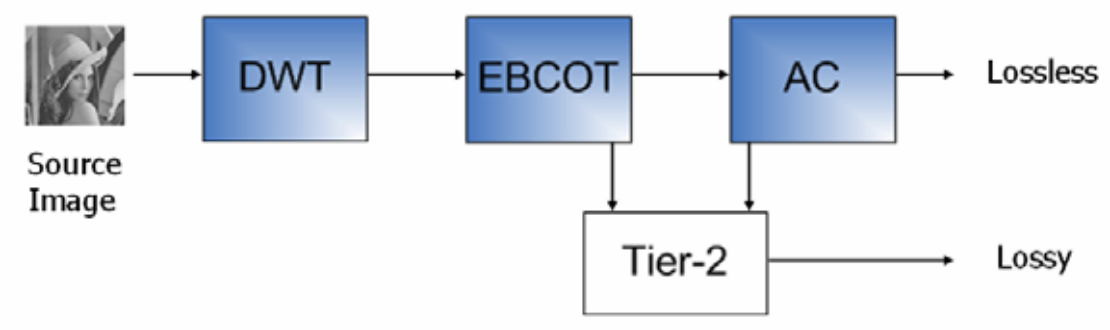


Figure 1-1 Block diagram of JPEG2000 encoding system.

Discrete Wavelet Transform (DWT) is adopted for the transform coding in JPEG2000 to transfer information from spatial domain to frequency domain. In the entropy coding part, JPEG2000 uses a new coding methodology called Embedded Block Coding with Optimized Truncation (EBCOT) as its entropy encoder. Moreover, after the block coding, Post-Compression Rate-Distortion Optimized (PCRD-Opt) rate control is exploited to obtain the best image quality and accurate rate control.

In most still image compression standards, rate control and compression ratio is adjusted by different quantization step sizes. However, the compression ratio of JPEG2000 standard can be controlled by the post-compression rate control procedure, so the quantization is usually ignored or used for weighting modification.

1.2 Our Proposed JPEG2000

1.2.1 QCB-Based DWT

The basic idea of QCB-based DWT is to divide a tile image into several QCB (quad code block) size images to perform the DWT process. Thus, the EBCOT can execute the bit-plane coding immediately after three code blocks are produced.

QCB-based DWT cuts the input tile image into four parts as in Figure 1-2 left side. After it encoded part 1, it generates three code blocks to EBCOT as in Figure 1-2 right side. When the part 2 is coded, another three code blocks to EBCOT are generated again. After the part 2 is coded, the next is part 3 and part 4; the encoding flow is as follows:

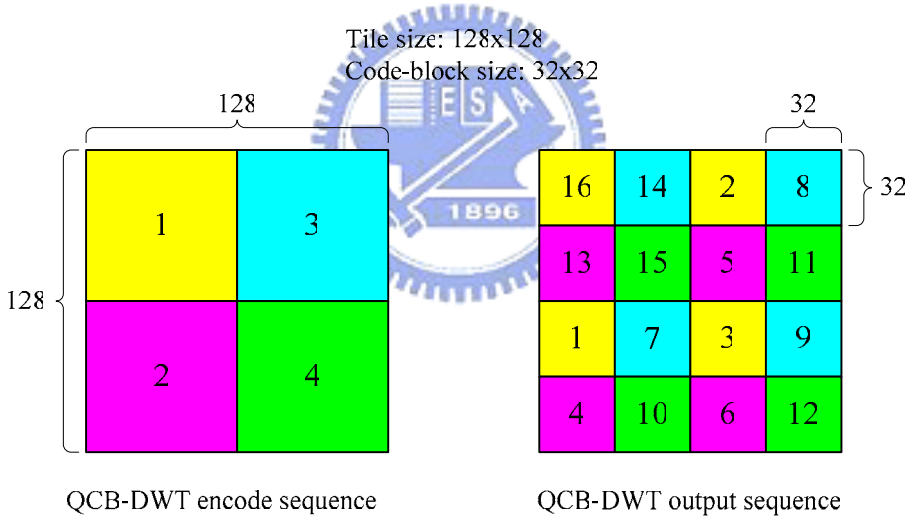


Figure 1-2 QCB-based DWT encode and output sequence.

Base on this algorithm, when QCB-based DWT finishes encoding the quarter of the tile image, EBCOT could start to encode the code block data (the code block size is 32x32 bytes). That is why QCB-based DWT just needs the quarter of compute time than the traditional DWT to output the coefficients to EBCOT and this feature does increase the overall system performance.

1.2.2 Pass Parallel EBCOT

EBCOT Tier-1 is the entropy coder in the JPEG2000; it transforms the output coefficients of DWT to the optimized single bit-stream.

The pass parallel architecture is used in this design, that is, the three coding passes are supposed to be coded in order originally, but it is not needed to code in this way right now. In the pass parallel architecture, we can encode the three passes in every bit plane at the same time and can save about 25% processing time and reduce 4K bits of internal memories when code block size is 32x32.

1.2.3 Four-stage Pipeline AC

Arithmetic Coding (AC) is also the entropy coder in the JPEG2000, it cooperates with EBCOT. AC receives the context label and symbol from EBCOT, then does the encoding operation and output the compressed image data. The pipeline architecture is used in AC hardware implementation, the throughput can increase at least 10 times than the standard and the AC can receive one CX-D pair per clock cycle.

1.2.4 Tier-2

In this thesis, we proposed a configurable rate control architecture. In traditional DWT, we can use Real-Time Rate-Distortion Optimized (RTRD-Opt) method to reduce computation of EBCOT Tier-1. On the other hand, it is more appropriate to use Imitative Post-Compression Rate-Distortion Optimized (IPCRD-Opt) method in QCB-based DWT architecture. Fuller discussion will be presented in the next chapter.

1.3 Thesis Organization

This thesis is composed of five chapters. The Tier-2 detailed implementation guide for JPEG2000 system is described in next chapter. In chapter 3, JPEG2000 IP and standard soft IP design flow are performed. Based on the JPEG2000 IP design, System-on-a-Chip (SoC) integration method and verification strategy are described in chapter 4. In the last chapter, conclusions and future works are given.

CHAPTER 2
TIER-2 METHOD AND HARDWARE DESIGN

The code-stream generated by the arithmetic encoder, together with the distortion metrics, allows JPEG2000 to selectively build the final bit-stream at the post-processing stage. This process is driven by two user-defined parameters:

- **Compression ratio:** This Tier-2 stage selects incoming packets to attain a user-specified compression ratio. The algorithm rejects packets that do not sufficiently improve the compression distortion. This mechanism allows precise control of the generated compressed file size, while maintaining good image quality.
- **Progression order:** JPEG2000 allows an initial preview of a picture with the first portion of the bit-stream. Decoding subsequent parts of the compressed file progressively refines the image. JPEG2000 also standardizes various refinement orders by prioritizing an image characteristic, such as quality or resolution. The Tier-2 stage achieves the desired progression order by reordering incoming packets.

In this chapter, an introduction of standard JPEG2000 rate control scheme is first given. In Section 2.2, we will introduce two proposed hardware implementation methods. Moreover, the detail implementation guide will be described in Section 2.3. Finally, result is given in Section 2.4.

2.1 Standard PCRD Optimized Rate Control

If we want to compress an image in high compression ratio, we need to compress an image by lossy compression, which means that we cannot perfectly reconstruct the original image. Although we cannot perfectly reconstruct the original image by lossy compression, we can reconstruct an image visually close to the original image to make the viewer unable to differentiate between the original image and the reconstructed image.

The rate control method recommended in JPEG2000 standard is quite different from other compression standards since almost all of them adopt quantization scheme for their rate control. However, it cannot control the bit-rate very accurate, and the obtained bit-stream is not optimal. Therefore, we will make a detailed explanation for the so-called Post-Compression Rate-Distortion Optimized (PCRD-Opt) rate control algorithm in JPEG2000.

Just as its name, PCRD-Opt is performed after all embedded bit-streams of code-blocks are generated. At this time, PCRD-Opt can use all information of all bit-streams of all code-blocks to find optimal truncation points for all embedded bit-streams by some length constraint. The concept of PCRD-Opt is briefly introduced as follows.

Because we wish to minimize overall distortion by the sum of bit-streams not exceeding maximal length $R_{\max}$, we need to select any set of truncation point, $\{T_i\}$,

such that

$$\sum_i R_i^{(T_i)} \le R_{\max} \tag{2-1}$$

With minimum

$$D_{\min} = \sum_i D_i^{(T_i)} \tag{2-2}$$

The above problem is equivalent to finding a set of truncation points, $\{T_{i,\lambda}\}$, which minimizes

$$D(\lambda) + \lambda R(\lambda) = \sum_i D_i^{(T_{i,\lambda})} + \lambda \sum_i R_i^{(T_{i,\lambda})} = \sum_i [D_i^{(T_{i,\lambda})} + \lambda R_i^{(T_{i,\lambda})}] \tag{2-3}$$

for some $\lambda > 0$ constrained by $R(\lambda) \le R_{\max}$.

For some given λ, we can find $T_{i,\lambda}^{opt}$ for each code-block B_i which minimize $D_i^{(T)} + \lambda R_i^{(T)}$; and $T_{i,\lambda}^{opt}$ for all code-block B_i minimize Equation (2-3) given some λ. However, the minimum of Equation (2-3) for some given λ is no need to be the minimum of Equation (2-3) constrained by $R(\lambda) \le R_{\max}$. Therefore, we need to find λ^{opt} which minimizes Equation (2-3) constrained by $\sum_i R_i^{(T_i)} \le R_{\max}$; and $T_{i,\lambda^{opt}}^{opt}$ is the optimal truncation point of the bit-stream of code-block B_i. Because $R(\lambda)$ is monotonic in λ, we can search for λ^{opt} by bisection algorithm in the interval $(\lambda^{\min}, \lambda^{\max})$.

Now we reduce the problem of finding a set of optimal truncation points to the problem of finding $T_{i,\lambda}^{opt}$ for each code-block B_i which minimize $D_i^{(T)} + \lambda R_i^{(T)}$ given some λ.

The concept of the R-D slope calculation is illustrated in Figure 2-1. In a code-block, Tier-1 adopts 3 pass coding scheme for each non-zero bit-plane. Then each pass will produce a piece of bit-stream. Actually each piece of bit-stream is a possible truncation segment. However, we need an evaluation factor to determine how

to truncate all the bit-streams. That is the R-D slope.

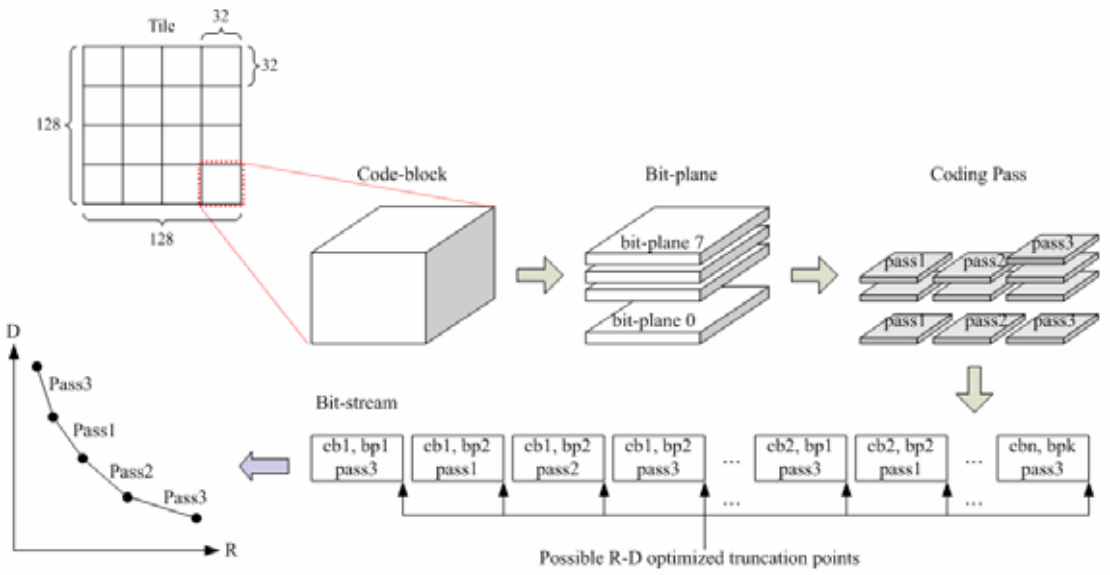


Figure 2-1 The coding structure of EBCOT.

2.1.1 Convex Hull Cancellation

In order to ensure that all bit-streams having contribution for reconstructed image. In EBCOT Tier-2, R-D optimized rate control should make sure that the slope values in a code-block are monotonically decreased. Therefore, the task of this part is to perform the convex hull cancellation if necessary.

To put the matter simply, there are three conditions that lead to a convex hull needed to be cancelled, and we list them below and depict them in Figure 2-2.

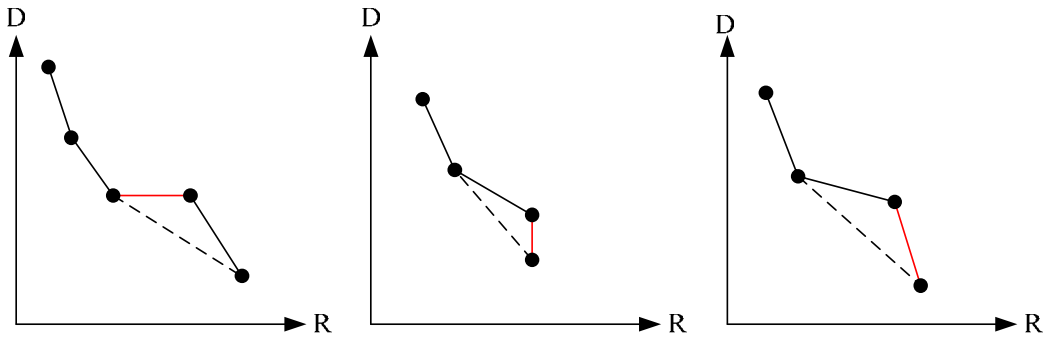


Figure 2-2 Different types of convex hull cancellation.

- If there is a pass whose delta distortion is not positive but byte count is increased, obviously we don't want this pass because getting it will pay more bytes but decrease the quality.
- We would like to include this kind of passes very much. It means that after including this pass, the overall distortion can be decreased without paying any byte overhead, even able to reduce the count.
- If the slope of the pass is large than or equal to that of the previous pass, they should be merged.

2.1.2 Evaluation of Compression Performance

To evaluate the performance of different image compression systems, we use several methods for different situations.

- The Compression Ratio (CR) metric is defined as:

$$CR = \frac{\text{original image size in bits}}{\text{compressed image size in bits}} \tag{2.3}$$

- The Bit Rate (BR) is defined as:

$$BR = \text{bpp=bits per pixel} \tag{2.4}$$

- The distortion between the original image and the reconstructed image can be expressed in terms of Mean Squared Error (MSE) or Peak Signal-to-Noise Ratio (PSNR) defined as:

$$MSE = \frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}(f_{x,y} - \hat{f}_{x,y})^2 \tag{2.5}$$

$$PSNR = 10\log_{10}\left(\frac{(2^p-1)^2}{MSE}\right) \quad dB \tag{2.6}$$

$f_{x,y}$ represents the magnitude of the pixel (x, y) in the original image while

$\hat{f}_{x,y}$ represents the magnitude of the pixel (x, y) in the reconstructed image. The dimension of the image is $M \times N$ and each pixel originally uses P bits to represent its magnitude.

In general, for lossless compression there is no distortion between the original image and the reconstructed image. Therefore, the only factor should be concerned is the amount of compression achieved, that is CR or BR. Obviously, the large CR (or smaller BR) represent the better compression performance. For lossy compression, the reconstructed image is only an approximation to the original. We can fix the compressed file size (CR or BR), and then compare the MSE or PSNR values among different compression systems. Obviously, smaller MSN or larger PSNR values correspond to less distortion and higher image quality.

A complete context formation module should include the distortion estimation, which is the core the rate distortion. In our design, it is difficult to compute the distortion of pass 1 and pass 2, so it is restricted to distortion of pass 3 now. In this situation, the truncation points must fall in the end of bit-plane. Hence, the lossy compression of JPEG2000 performs poorly than truncation points could fall in the end of every pass.

2.2 Our Proposed

One of the superior features in JPEG2000 is the ability of accurate rate-distortion optimized rate control. In fact, the PCRD-Opt algorithm is complicated and not suitable for the hardware implementation. In this section, we introduce two kinds of rate control scheme which appropriately be used in traditional DWT and QCB-based DWT.

2.2.1 Real-Time Rate-Distortion Optimized

In PCRD-Opt method, the R-D slope information at all the feasible truncation points are pre-computed and store in memory. This requires Tier-1 encoding of all the quantized coefficients and the storage of the whole encoded bit-stream in memory buffer even thought a large portion of them will not be included in the code-stream after the optimal truncation.

Figure 2-3 indicates that the traditional DWT encode and output sequence. We can see that the most significant code-block in LL band will be coding by EBCOT Tier-1 and output first. When the code-block was done, Tier-2 can truncate the output bit-stream and stall the operation of Tier-1 while the bit-streams are no contribution for reconstructed image. Generally speaking, the LL band coefficients are usually more important than LH, HL and HH band of wavelet coefficients. It would be better to say that if Tier-2 can get the code-block in LL band first, the truncation threshold may be accurate.

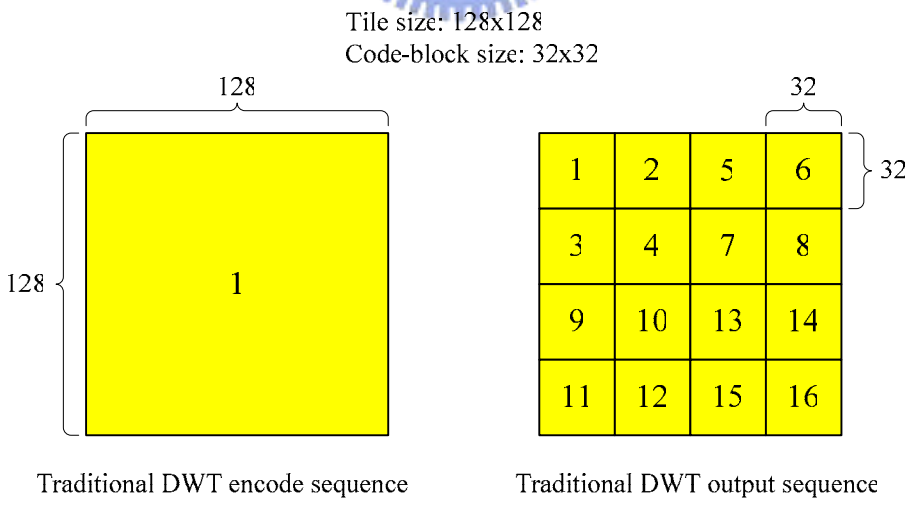


Figure 2-3 Traditional DWT encode and output sequence.

In RTRD-Opt architecture, code-blocks output sequence is one by one. So it only one pair of hardware for calculation of delta D, delta R, R-D slope and convex hull cancellation.

However, without using the actual R-D slopes of the whole image/tile, it is impossible to find the optimal truncation points. The cause is that when we get all R-D slope information and decide to truncate these bit-streams in this code-block, all operation of Tier-1 was done. To solve this problem, some mechanism to predict the truncation of next code-block is needed. So the problem now becomes finding the "good enough" truncation points but not the optimal.

2.2.2 Imitative PCRD Optimized

Because our JPEG2000 is QCB-based DWT architecture, it is not suitable for the RTRD-Opt. In QCB-based DWT, the most significant code-block will be coding in the end. Figure 2-4 indicates that the QCB-based DWT encode and output sequence. We can get the code-blocks from LH band, HL band and HH band but these code-blocks is not weighty than the code-blocks in LL band. Tier-2 may work now and decide a truncation threshold which is not optimal. For reasons mentioned above, we can't save the encoding time of Tier-1 in this operation sequence.

Thus, there is a better way for Tier-2 start while all code-block is done. It is not standard PCRD-Opt method but likely. The differentiation of them is how to generate the truncation threshold. Standard PCRD-Opt method has uncertain iteration loop that is not suitable for hardware implementation. In Imitative PCRD-Opt (IPCRD-Opt), we create a slope table to store R-D slope and rate information. The slope table will be updated when code-block done every time. That is to say, we can find the optimal truncation threshold easily in this slope table without any iteration loop.

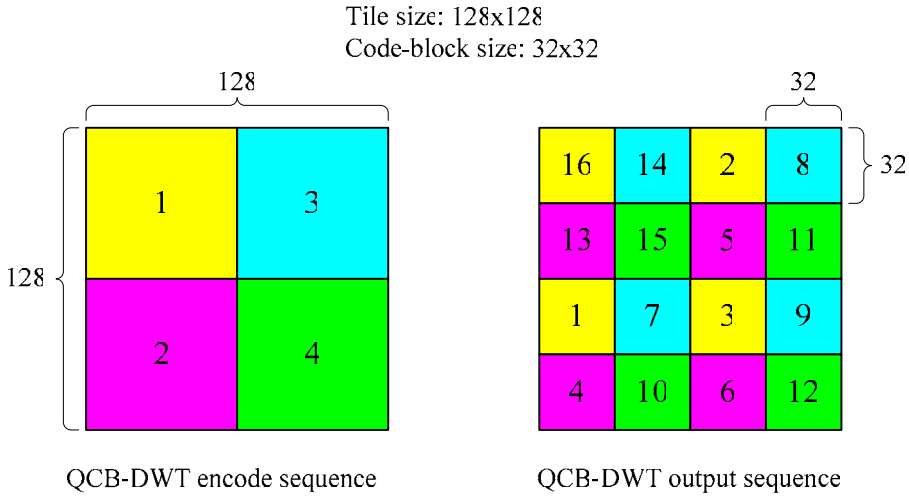


Figure 2-4 QCB-based DWT encode and output sequence.

In IPCRD-Opt architecture, there are three code-blocks output simultaneously. So it needs three pair of hardware for calculation of delta D, delta R, R-D slope and convex hull cancellation.

2.3 Implementation Guide

In this section, we will focus on implementation guide of JPEG2000 Tier-2. They are including flow chart, architecture, finite state machine, pin definition and so on.

We will take an example to illustrate slope table operation. Assume that the tile size is 128x128, 8-bits per pixel and the compression ratio is 64. It is say that data capacity reconstructed must less than 256 bytes. After three code-blocks coding, the byte count and truncation point is show below.

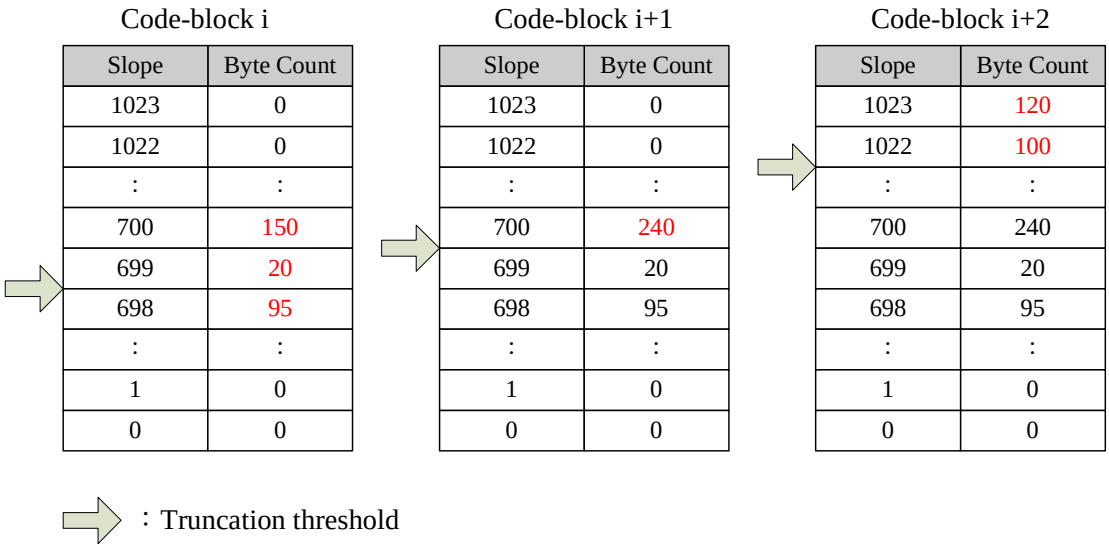


Figure 2-5 An example of the slope table.

It will be clear from these examples that we can find the optimal truncation threshold easily in this slope table without any iteration loop. But an extra memory to store these information is needed.

2.3.1 Flow Chart

Here is a flow chart which shows that the RTRD-Opt method operation flow. The main idea is to build a slope table by accumulating the number of bytes of each pass for each slope value and update it after finishing the coding of each code-block. When update slope byte every time, it will decide a threshold which be satisfied equation (2.1) and (2.2). Finally, data formation procedure will reconstruct and output data when all code-blocks were done.

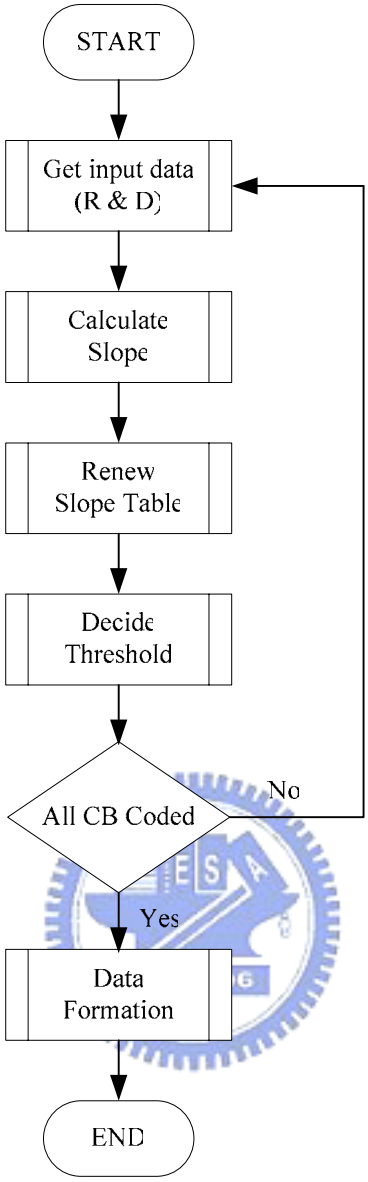


Figure 2-6 RTRD-Opt flow chart of JPEG2000 EBCOT Tier-2.

The Figure 2-7 below illustrates that IPCRD-Opt method operation flow. We can see that Tier-2 decides the threshold when all code-blocks were done.

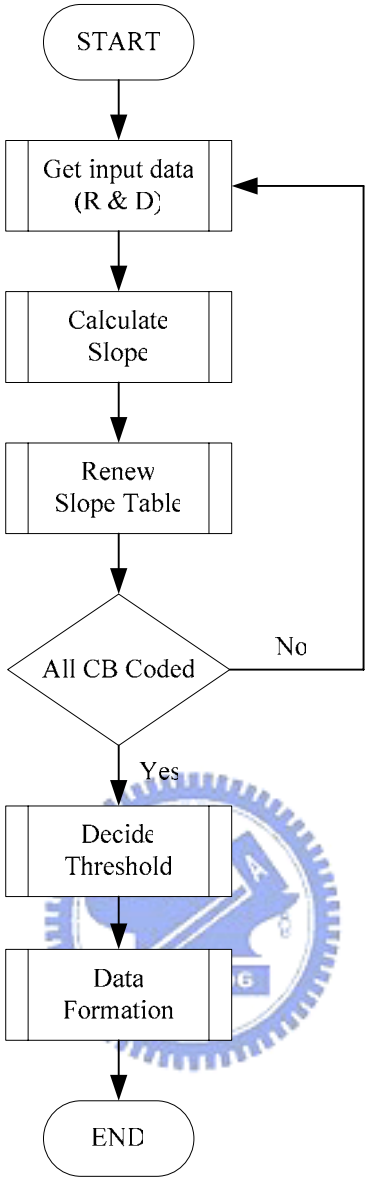


Figure 2-7 IPCRD-Opt flow chart of JPEG2000 EBCOT Tier-2.

2.3.2 Architecture

The Figure 2-8 illustrates us that the architecture of Tier-2. The Tier-2 hardware consists of slope calculation module, renew slope table module, threshold generator module, R-D optimized module and data formation module. The differentiation of RTRD-Opt method and IPCRD-Opt method are Tier-1 stall mechanism and hardware implementation cost.

The items listed below are detailed explanation of every sub-block in Tier-2.

- **Slope Calculation:**

  - Calculate delta distortion and delta rate.

  - Calculate R-D slope.

  - Convex hull cancellation.

  - Stall EBCOT Tier-1 (RTRD-Opt only).

- **Renew Slope Table:**

  - Renew slope table.

- **Threshold Generator:**

  - Generate optimal threshold.

- **R & D Optimized:**

  - Truncate AC bit-stream and update header information.

- **Data Formation:**

  - Reordering and output AC bit-stream and header information (FIFO structure).

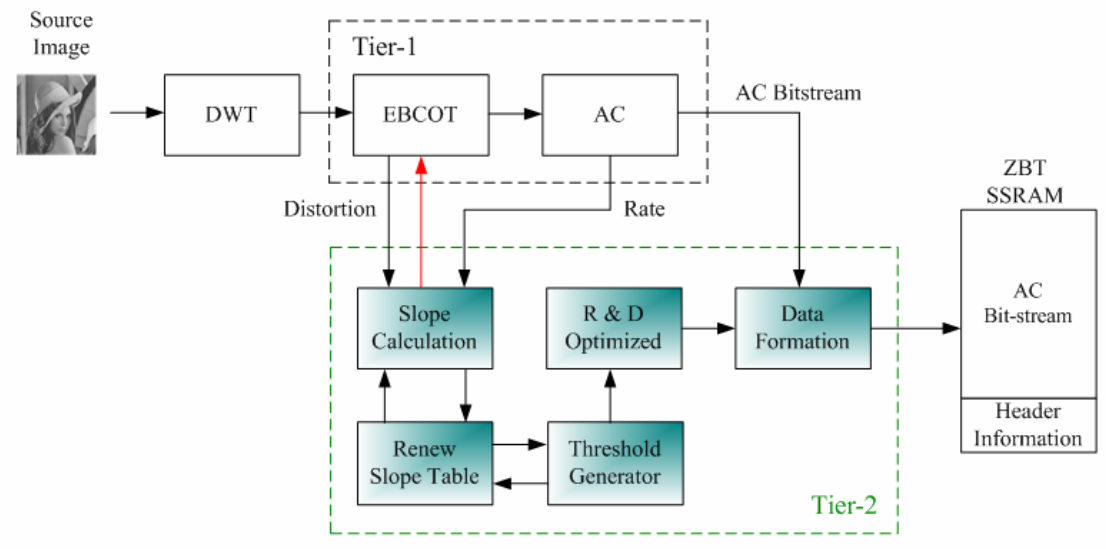


Figure 2-8 Architecture of JPEG2000 EBCOT Tier-2.

2.3.3 Finite State Machine

This figure indicates that the finite state machine of Tier-2.

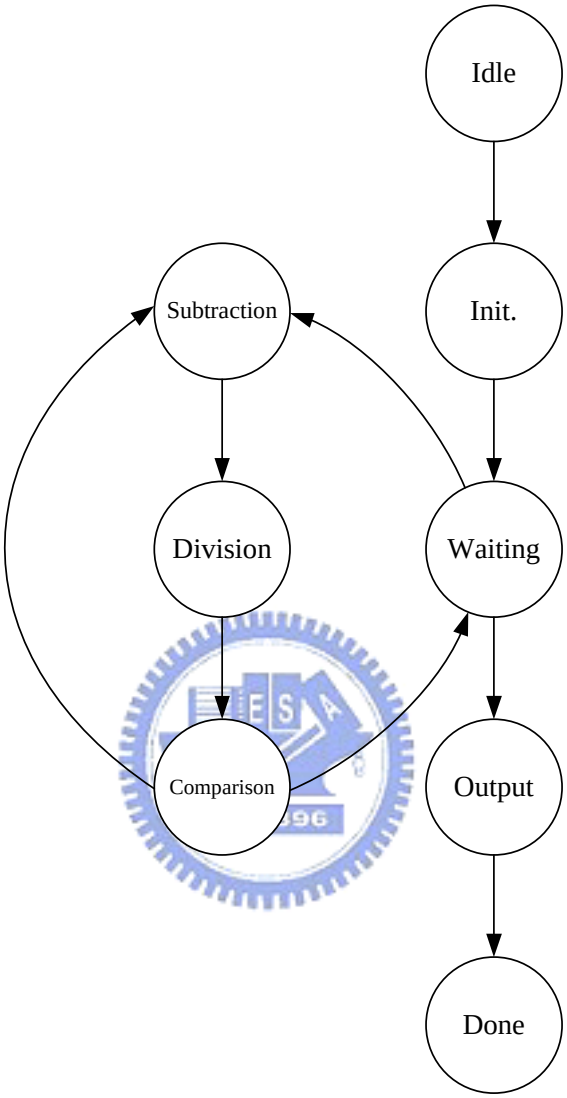


Figure 2-9 Finite state machine of JPEG2000 EBCOT Tier-2.

The items listed below are detailed explanation of FSM in Tier-2.

- **Idle:** This state illustrates that Tier-2 will do nothing.
- **Init:** This state illustrates that all signals and variables will be initialized.
- **Waiting:** This state illustrates that Tier-2 is waiting for input data (D and R).
- **Subtraction:** This state illustrates that Tier-2 is calculating delta D and delta R.
- **Division:** This state illustrates that Tier-2 is calculating R-D slope.

- **Comparison:** This state illustrates that Tier-2 will do the convex hull cancellation or not.
- **Output:** This state illustrates that Tier-2 will output AC bit-stream and header information.
- **Done:** This state illustrates that Tier-2 operation was done.

2.3.4 Pin Definition

The table and figure below illustrate that pin definition of Tier-2.

Table 2-1 Pin definition of JPEG2000 EBCOT Tier-2.

Signal Name	Direction	Width	Description
Mode	input	1	H: RTRD, L: PCRD
clk	input	1	Clock signal
rst	input	1	Reset signal
CR	input	3	Compression ratio
Distortion_1	input	32	Distortion of code-block 1
Dis_1_valid	input	1	Distortion valid signal of code-block 1
Rate_1	input	16	Rate of code-block 1
Rate_1_valid	input	1	Rate valid signal of code-block 1
Distortion_2	input	32	Distortion of code-block 2
Dis_2_valid	input	1	Distortion valid signal of code-block 2
Rate_2	input	16	Rate of code-block 2
Rate_2_valid	input	1	Rate valid signal of code-block 2
Distortion_3	input	32	Distortion of code-block 3
Dis_3_valid	input	1	Distortion valid signal of code-block 3
Rate_3	input	16	Rate of code-block 3
Rate_3_valid	input	1	Rate valid signal of code-block 3
Header_Info	output	32	Header information
Address	output	32	Output address
RAM_EN	output	1	ZBT SSRAM enable signal

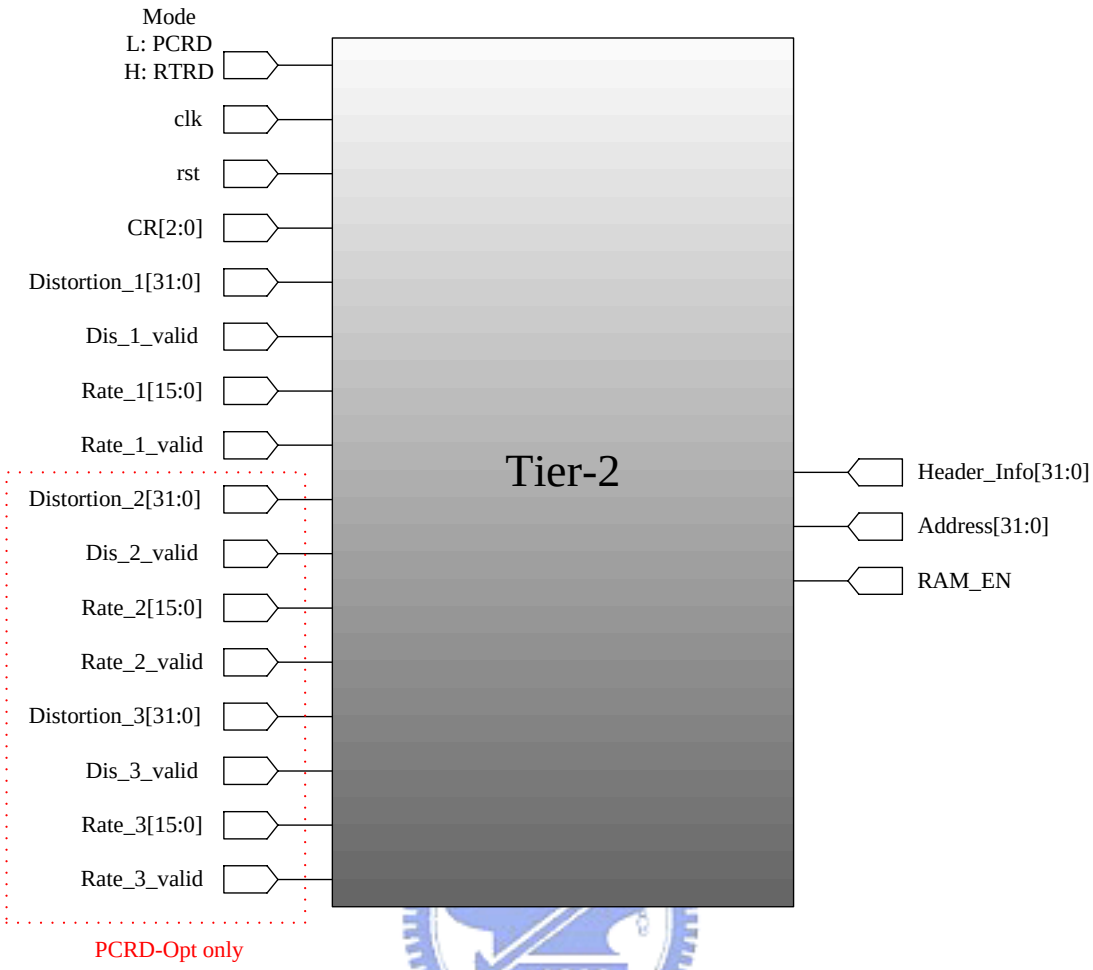
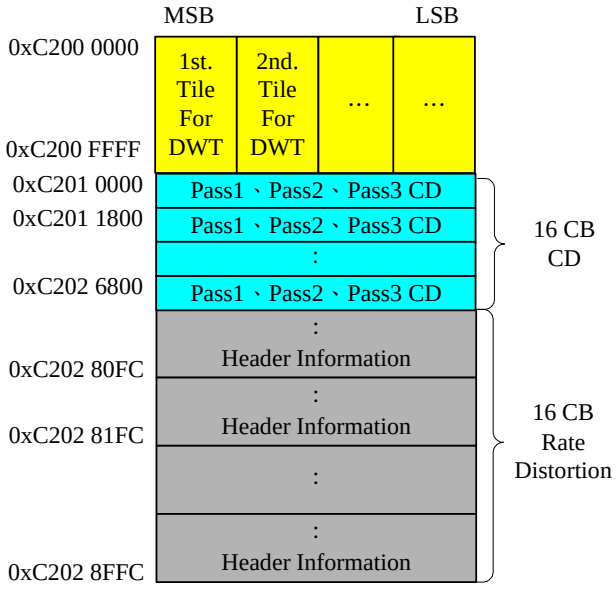


Figure 2-10 Pin definition of JPEG2000 EBCOT Tier-2.

2.3.5 SSRAM Memory Map

Memory configuration and usage are very important to embedded SoCs. ARM Integrator system is memory map architecture so we must plan a range of memory to store output data. User can find the compressed bit-stream and packet header information at the defined memory address.

The logic module provides the address space from 0xC2000000 to 0xC2100000 of 1 MB ZBT SSRAM. Figure 2-11 below is the memory allocation of the SSRAM for JPEG2000 coprocessor.



Header Information：

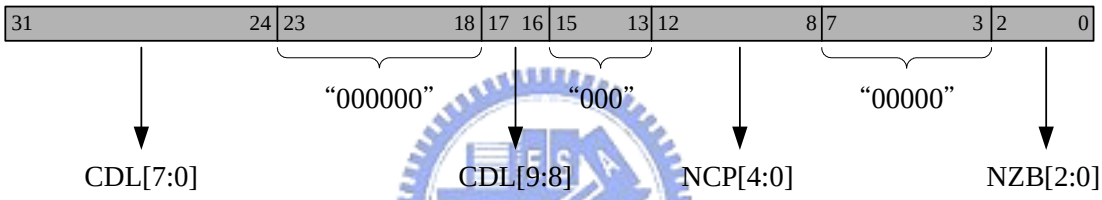


Figure 2-11 Memory allocation of SSRAM in logic module.

- **CDL:** Indicates length of AC output bit-stream.
- **NCP:** Indicates number of coding pass.
- **NZB:** Indicates number of zero bit-plane.
- **0xC2000000~0xC200FFFF:** Input tile image for QDWT.
- **0xC2010000~0xC2028000:** The compressed bit-stream from entropy coder. They are in bit-plane order and pass order. We allocate 256 bytes for every bit-plane pass in a code-block.
- **0xC2028000~0xC2029000:** For rate, distortion and header information in every code-block. It is allocated 256 bytes for every rate and distortion. The header information only uses 4 bytes and is located in the fixed address predefined.

2.4 Results

Figure 2-12 and 2-13 show PSNR comparison results between the proposed IPCRD-Opt method and the Jasper's JPEG2000 software VM. This demonstrates the proposed scheme only degrades about 1 dB from the rate-distortion optimization scheme, which can be hardly implemented as a practical system.

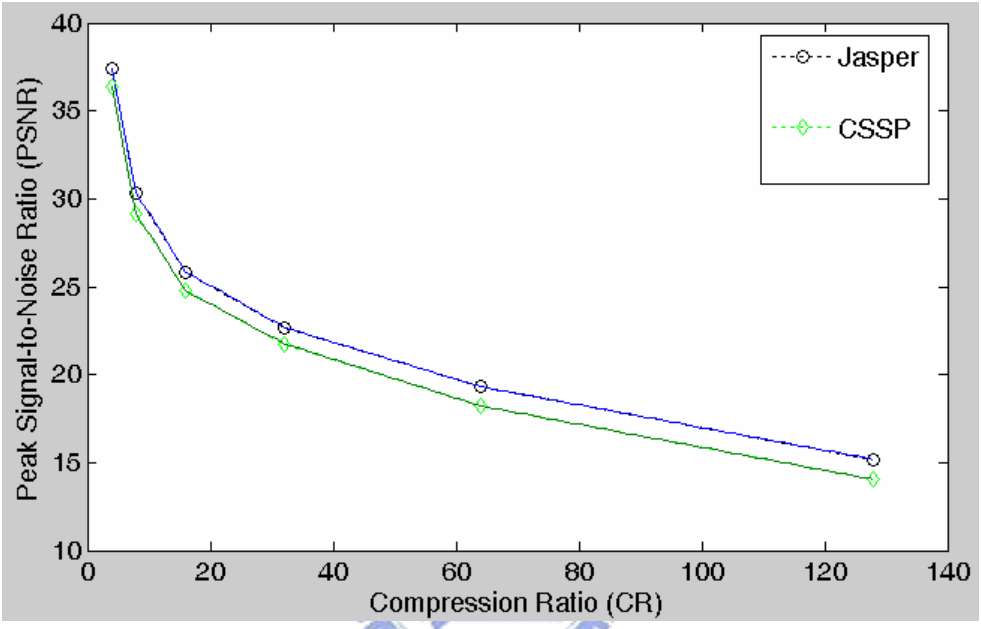


Figure 2-12 PSNR of Lenna 128x128.

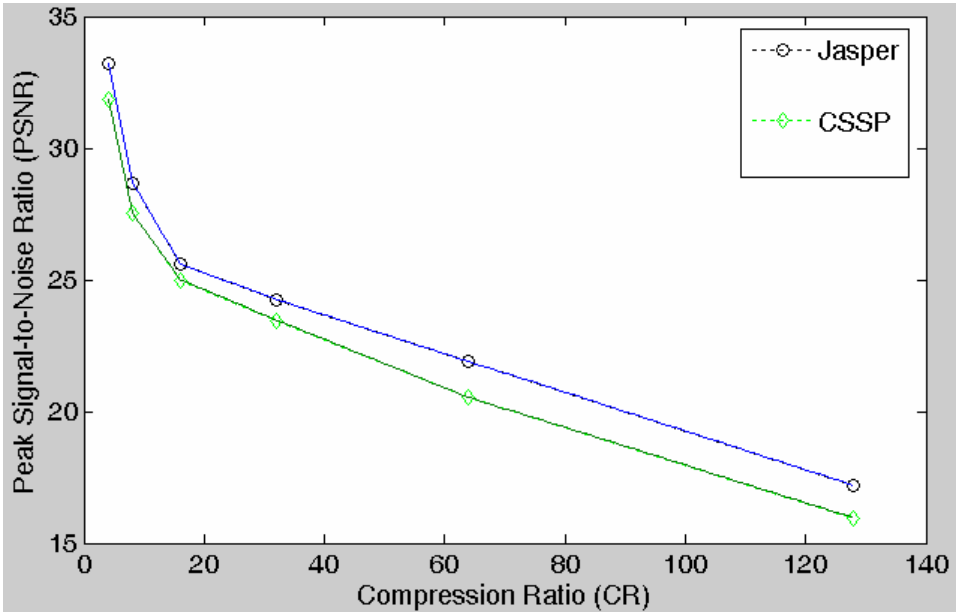


Figure 2-13 PSNR of Baboon 128x128.

CHAPTER 3
JPEG2000 IP IMPLEMENTATION

The never ending increase of silicon capacity available to system and IC designers, as predicted by Moore's Law, brings on a cyclical crisis in design methodology and engineering productivity generating a ripple effect through the EDA and electronics industries. The SoC era will need more than available silicon to become a reality. A new design methodology roadmap based on Intellectual Property (IP) reuse needs to emerge.

In this chapter, we introduce the concept of IP-based and platform-based design from Section 3.1 through Section 3.3. On-chip bus AHB and AHB-Lite specification is shown in Section 3.4. In Section 3.5, we will introduce IP Qualification guideline. Moreover, we construct a JPEG2000 encoder IP core generator in Section 3.6.

3.1 Why IP Reuse

IP-based design is more and more important in SoC era. We can decrease design time with reusable IPs. Without a design reuse or IP reuse strategy it would be hard to compete favorably in the SoC era.

Generally speaking, SIP may be divided into three types. In this thesis, we will focus on soft IP.

- Soft IP indicates that IP designed in the form of synthesizable HDL code (in

Verilog or VHDL format).

- Firm IP indicates that IP delivered in the form of gate-level netlist after synthesis.
- Hard IP indicates that IP delivered generally in the form of GDSII format, which are fully placed, routed and optimized for power, size, or performance, and mapped to a specific process technology.

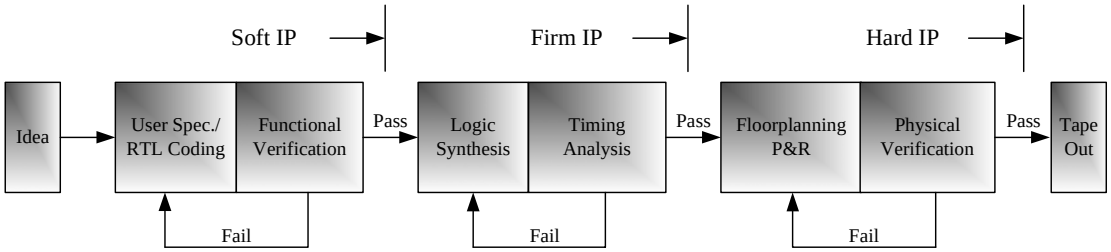


Figure 3-1 Classification of IP.

3.2 IP-Based and Platform-Based Design

Platform-Based Design (PBD) methodology was lately brought forward within SoC design field to meet the increasing complexity on single chip and pressure on Time-To-Market (TTM). The system level design concept of SoC is more and more important. Therefore, an efficient platform-based design flow is proposed in this section.

3.3 Platform

When a project starts, one of the most important parts is to build a working prototype. While this is often left until the hardware design is complete and a prototype PCB can be manufactured, this creates enormous uncertainly during a large part of the project. Software engineers are not entirely sure what they are targeting. Hardware engineers are not completely sure that their design will be correct.

Early prototyping is one solution to these problems. With an ARM Integrator

board, an engineer can make up a prototype of large parts of the system long before a board design exists.

Based on this SoC platform, hardware and software can be co-designed, co-simulated and co-verified. The prototyping environment can be partially reused for the system with the same platform. Therefore, the system can be built, verified and implemented rapidly by using this platform design.

3.3.1 Motherboard (Integrator/AP)

Integrator/AP is very similar to general personal computer motherboard; it has three PCI slots, two COM ports, PS/2 port. Besides, AP has two sockets, one is for core modules, and the other is for logic modules. We can stack up four core/logic modules on the individual socket, that is, we can use more than two core/logic modules at the same time, and this increases the applicability of the Integrator.

3.3.2 Core Module (Integrator/CM920T)

ARM core is put on the core module; we can change different core modules according to different ARM cores. There are 32MB flash, 1MB SSRAM, 128MB SDRAM on the core module, Linux OS can be ported to develop the Application Program Interface (API) and the drivers for the peripherals.

3.3.3 Logic Module (Integrator/LM-EP20K600E+)

There are an Altera FPGA, the content in which is about 1,000,000, 1MB SSRAM, 32MB flash on the logic module. The hardware of the JPEG2000 coprocessor is put here.

3.4 On-chip Bus: AMBA

The Advanced Microcontroller Bus Architecture (AMBA) specification defines an on-chip communications standard for designing high-performance embedded microcontrollers. On-Chip Bus (OCB) is the communication backbone with IPs in SoC. Data and control messages are passed through on-chip bus to all computational units and peripheral devices connected on the bus.

3.4.1 Typical AMBA AHB System

Three distinct buses are defined within the AMBA specification:

- **Advanced High-performance Bus (AHB):**

The AMBA AHB is for high-performance, high clock frequency system modules. The AHB acts as the high-performance system backbone bus. AHB supports the efficient connection of processors, on-chip memories and off-chip external memory interfaces with low-power peripheral macrocell functions. AHB is also specified to ensure ease of use in an efficient design flow using synthesis and automated test techniques.

- **Advanced System Bus (ASB):**

The AMBA ASB is for high-performance system modules. AMBA ASB is an alternative system bus suitable for use where the high-performance features of AHB are not required. ASB also supports the efficient connection of processors, on-chip memories and off-chip external memory interfaces with low-power peripheral macrocell functions.

- **Advanced Peripheral Bus (APB):**

The AMBA APB is for low-power peripherals. AMBA APB is optimized for minimal power consumption and reduced interface complexity to support peripheral

functions. APB can be used in conjunction with either version of the system bus.

The AMBA specification has been derived to satisfy four key requirements:

- To facilitate the right-first-time development of embedded microcontroller products with one or more CPUs or signal processors.
- To be technology-independent and ensure that highly reusable peripheral and system macrocells can be migrated across a diverse range of IC processes and be appropriate for full-custom, standard cell and gate array technologies.
- To encourage modular system design to improve processor independence, providing a development road-map for advanced cached CPU cores and the development of peripheral libraries.
- To minimize the silicon infrastructure required to support efficient on-chip and off-chip communication for both operation and manufacturing test.

An AMBA-based microcontroller typically consists of a high-performance system backbone bus (AMBA AHB or AMBA ASB), able to sustain the external memory bandwidth, on which the CPU, on-chip memory and other Direct Memory Access (DMA) devices reside. This bus provides a high-bandwidth interface between the elements that are involved in the majority of transfers. Also located on the high-performance bus is a bridge to the lower bandwidth APB, where most of the peripheral devices in the system are located (see Figure 3-2).

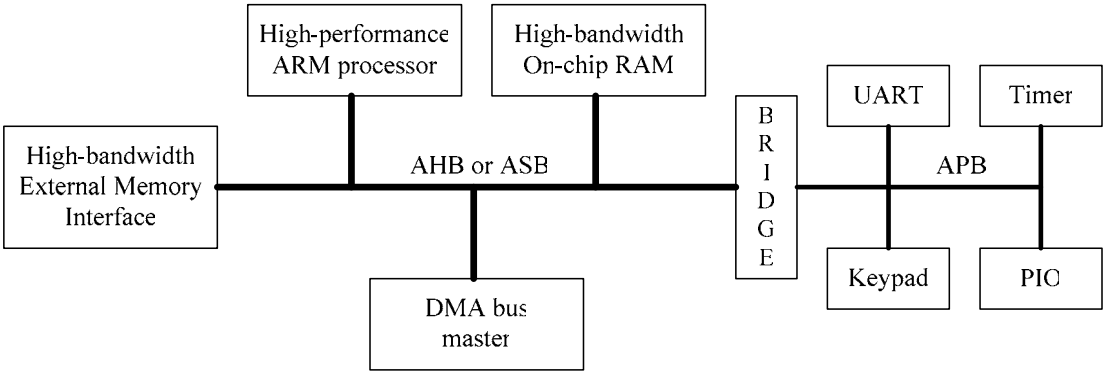


Figure 3-2 A typical AMBA system.

3.4.2 Single-Master AHB System: AHB-Lite

Actually, our system is based on AHB-Lite signal-master system. AHB-Lite is a subset of the full AHB specification and is intended for use in designs where only a single bus master is used. This can either be a simple single-master system, as shown below or a Multi-layer AHB system where there is only one AHB master per layer. Figure 3-3 shows a block diagram of the single-master system.

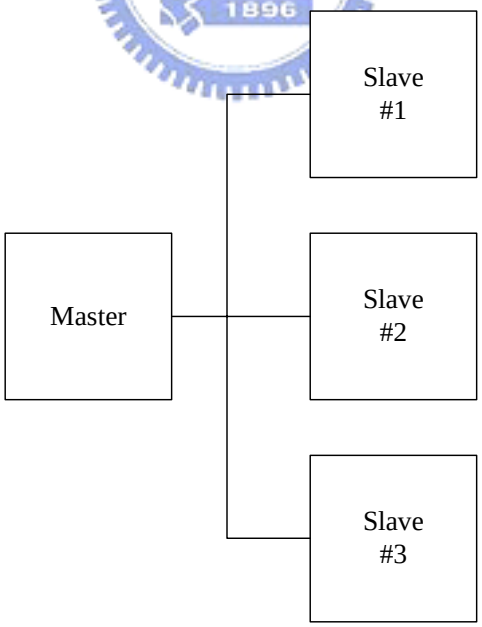


Figure 3-3 AHB-Lite single-master system.

AHB-Lite simplifies the AHB specification by removing the protocol required for multiple bus masters, which includes the Request/Grant protocol to the Arbiter and the Split/Retry responses from slaves.

Masters designed to the AHB-Lite interface specification can be significantly simpler in terms of interface design, compared to a full AHB master. AHB-Lite enables faster design and verification of these masters and the addition of a standard off-the-shelf bus mastering wrapper can be used to convert an AHB-Lite master for use in a full AHB system.

Any master that is already designed to the full AHB specification can be used in an AHB-Lite system with no modification.

The majority of AHB slaves can be used interchangeably in either an AHB or AHB-Lite system. This is because AHB slaves that do not use either the Split or Retry response are automatically compatible with both the full AHB and the AHB-Lite specification. It is only existing AHB slaves that do use Split/Retry responses that require an additional standard off-the-shelf wrapper to be used in an AHB-Lite system.

Any slave designed for use in an AHB-Lite system works in both a full AHB and an AHB-Lite design.

The AHB-Lite specification differs from the full AHB specification in the following ways:

- Only one master. There is only one source of address, control, and write data, son no Master-to-Slave multiplexer is required.
- No arbiter. None of the signals associated with the arbiter are used.
- Master has no **HBUSREQ** output. If such an output exists on a master, it is left unconnected.
- Master has no **HGRANT** input. If such an input exists on a master, it is tied

HIGH.

- Slaves must not produce either a Split or Retry response.
- The AHB-Lite lock signal is the same as **HMASTLOCK** and it has the same timing as the address bus and other control signals. If a master has an **HLOCK** output, it can be retimed to generate **HMASTLOCK**.
- The AHB-Lite lock signal must remain stable throughout a burst of transfers, in the same way that other control signals must remain constant throughout a burst.

Table 3-1 shows how masters and slaves designed for use in either full AHB or AHB-Lite can be used interchangeably in different systems.

Table 3-1 AHB-Lite interchangeability.

Component	Full AHB system	AHB-Lite system
Full AHB master	V	V
AHB-Lite master	Use standard AHB master wrapper	V
AHB slave (no Split/Retry)	V	V
AHB slave (with Split/Retry)	V	Use standard AHB slave wrapper

The advantage of using the AHB-Lite protocol is that the bus master does not have to support the following cases:

- Losing ownership of the bus. The clock enable for the master can simply be derived from the HREADY signal on the bus.
- Early terminated bursts. There is no requirement for the master to rebuild a burst due to early termination, because the master always has access to the bus.
- Split or Retry transfer responses. There is no requirement for the master to retain the address of the last transfer to be able to restart a previous transfer.

Figure 3-4 shows a more detailed block diagram, including Decoder and Slave-to-Master multiplexer connections.

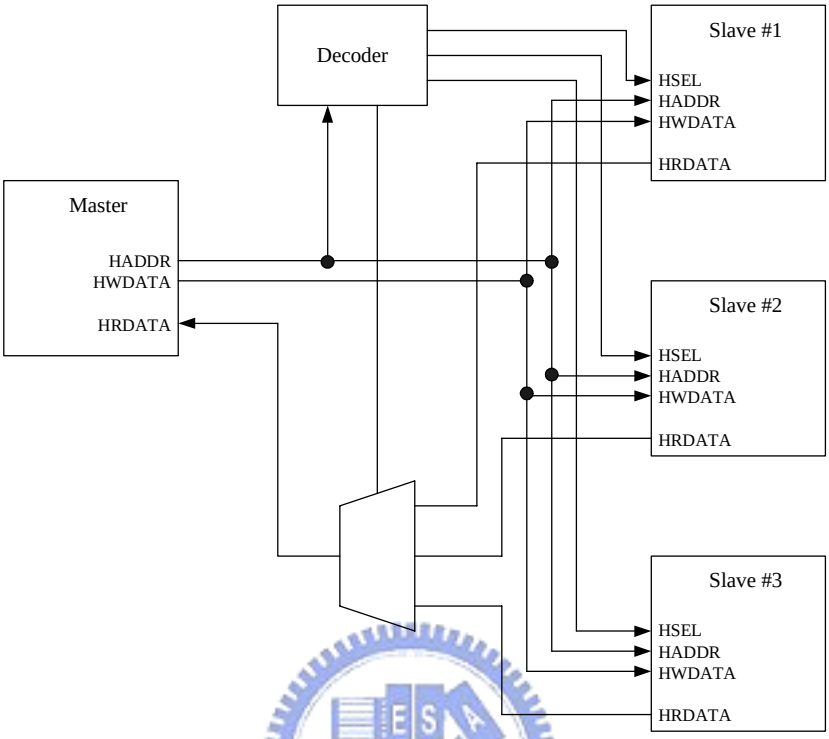


Figure 3-4 AHB-Lite block diagram.

3.5 IP Qualification Guideline Overview

A large number of today's IC products have evolved to SoC. As the trend has changed, so does the way IC products are designed. SoC are composed of one, and often several design units, including processor on chip, large amount of memory, bus-based peripherals, coprocessors and other I/O channels. There design units which are called IP, a block-based design or a reused macro that are emphasized for design reuse.

The general rules proposed in the IP Qualification (IPQ) guidelines are a set of best practices for creating reusable designs for use in an SoC design methodology. There practices are based on several reusable methodology literatures and experiences from Steering Committee members of IPQ Alliance in developing reusable designs.

Reusable macros that have already been designed and verified aid users to be aware of all need-to-know issues in advance. If the blocks do not conform to this standard for reusable methodology, the efforts for integrating pre-existing blocks into new SoC could become excessively high.

The quality criteria, which have to be taken into account, come from various sources: The Reuse Methodology Manual (RMM) contains a set of rules and guidelines that help ensure that a design is reusable and technology-independent. IPQ describes that language subset of VHDL or Verilog that are synthesizable and verifiable with any compliant tool. Further efforts on quality are under way in the Virtual Socket Interface Alliance (VSIA).

3.5.1 Soft IP Design Flow

In standard soft IP design flow, start in design specification definition, IP creators must follow the rules in design style guidelines, which is a section of soft IP design guidelines, to maximize the reusability of IP. In RTL coding phase, IP creators have to follow the rules in coding style guidelines to increase the readability, testability and reusability of the IP. When the design process stepping into synthesis. IP creators must follow the synthesis script coding guidelines, to increase the readability and reusability of the synthesis scripts template of the IP.

The soft IP is designed via the design flow illustrated in the Figure 3-5 below.

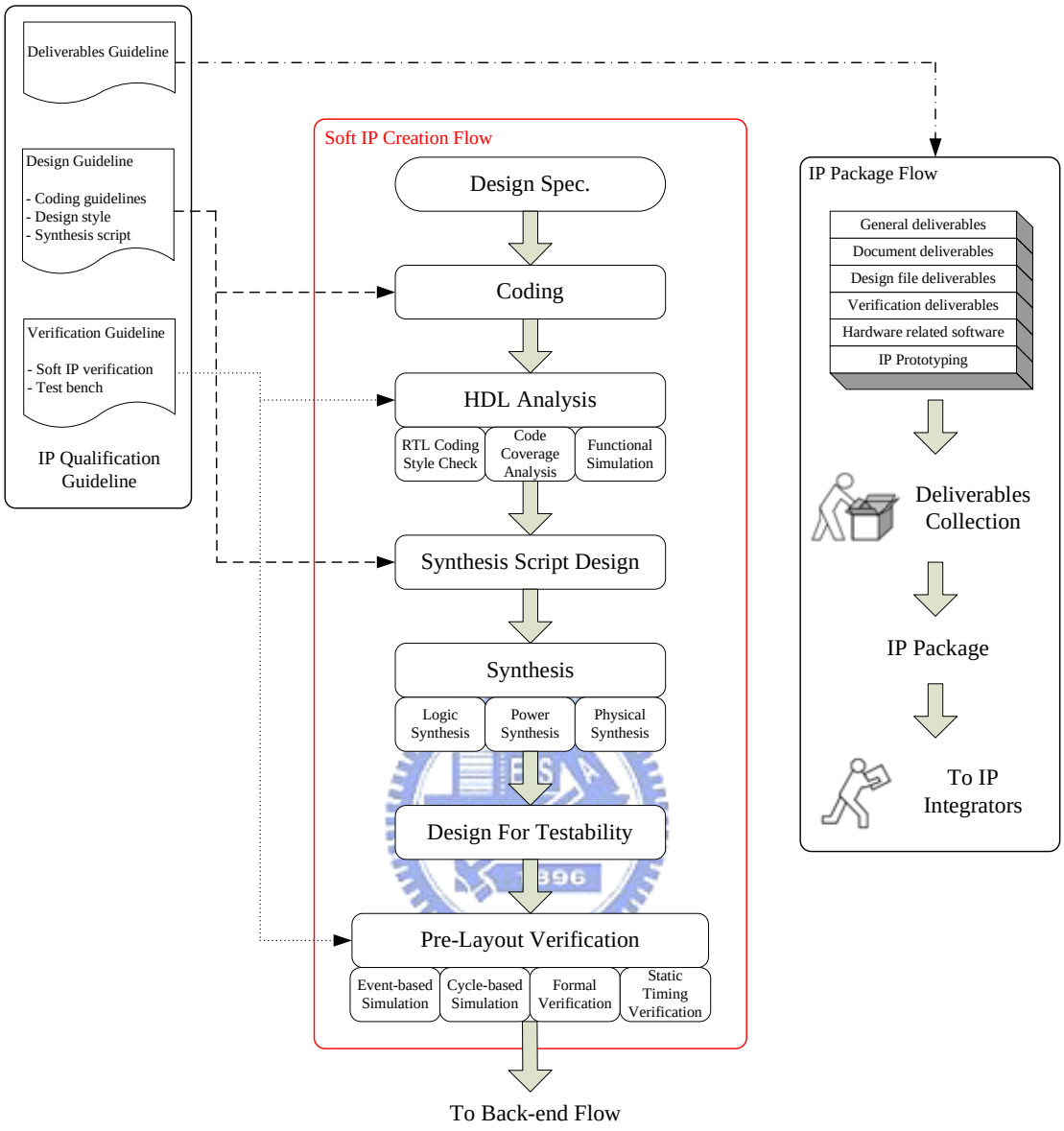


Figure 3-5 Soft IP design flow.

3.5.2 IP Qualification Guideline

These guidelines are no longer a proposal. They are solutions practiced in today's IP-based SoC design. The guidelines will become the basis for industry-wide solutions to accelerating the best practice to develop reusable and higher quality IP, and consequently contribute to IP exchanges.

The scope of these guidelines is that reuse-based design requires explicit

reusable methodology for reusable macro that is easily to be integrated into SoC designs and facilitates reusable IP exchange. The IP Qualification Guidelines classified the reusable methodology into three categories:

- **Design guidelines:** The design guidelines include coding rules and design issues. Soft IP that follows the rules can ensure that the HDL code is readable, portable and reusable. In addition, these rules also help achieve optimal synthesis and simulation results. The guidelines are categorized as follows:

  - HDL (Verilog & VHDL) coding guidelines.

  - Design style guidelines.

  - Synthesis script guidelines.

- **Verification guidelines:** In verification guidelines, a set of rules are provided which need to be followed by IP creators to improve the verification quality of the IP. The guidelines are categorized as follows:

  - Soft IP verification guidelines.

  - Coding guidelines in writing testbench codes.

  - IP prototyping.

- **Deliverable guidelines:** In deliverable guidelines, the rules which are needed to be followed by IP creators to ease the reuse process of the IP. The guidelines are categorized as follows:

  - General deliverables.

  - Documentation deliverables.

  - Design files deliverables.

  - Verification deliverables.

  - Hardware related software deliverables.

  - IP prototyping deliverables.

There are numerous rules contained in this three guidelines, the level of rules are classified into three classes,

- **Mandatory 1 (M1):**

  - Rules must be followed. If not, design must be fixed.

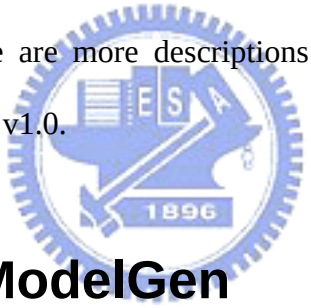
- **Mandatory 2 (M2):**

  - Rules should be followed. If not, documentation must be provided.

- **Recommended (R):**

  - Rules are recommended to be followed in the design.

Although there rules are classified into three severity levels, all of the rules in the guidelines are strongly encouraged to be followed. We are not concerned here with these detail guidelines. There are more descriptions of these guidelines in the IP Qualification Guidelines 2003 v1.0.



3.6 JPEG2000 ModelGen

In this section, we provide a friendly Graphics User Interface (GUI) with JPEG2000 coprocessor. User can configure this JPEG2000 IP easily via this GUI.

In the first page as Figure 3-6, user can select if they want AMBA wrapper or not. Asynchronous or synchronous reset signal can be configured. Figure 3-7 and Figure 3-8 indicate that user can select different synthesis optimization goal via one pair or three pair entropy coder architecture. Moreover, the JPEG2000 ModelGen also provides four modules listed below to conform to user's requirements.

- **Un-timed behavior model (C/C++ language).**

  This model is provided with high level C/C++ language. It is a behavior model which is function correct and no timing information. With this model, user can

develop software early.

- **Cycle-accurate, synthesizable RTL model (VHDL language).**

  This model is provided with Hardware Description Language (HDL). The basic deliverable of soft IP is RTL code. The model is cycle-accurate and synthesizable. User can use this model to design his product. The RTL code can be developed in both ASIC or FPGA target.

- **Timing-accurate, after synthesis gate-level model (gate-level netlist).**

  This model is provided with gate-level netlist after synthesis. It is timing-accurate model which includes gate delay information. Moreover, it also has power and area information. User can put this netlist to run back-end design flow directly.

- **After placement and routing, physical model (GDSII format).**

  This model is provided with GDSII format after placement and routing. The model passes the DRC and LVS check and post-layout simulation. User can integrate this macro like hard IP quickly.

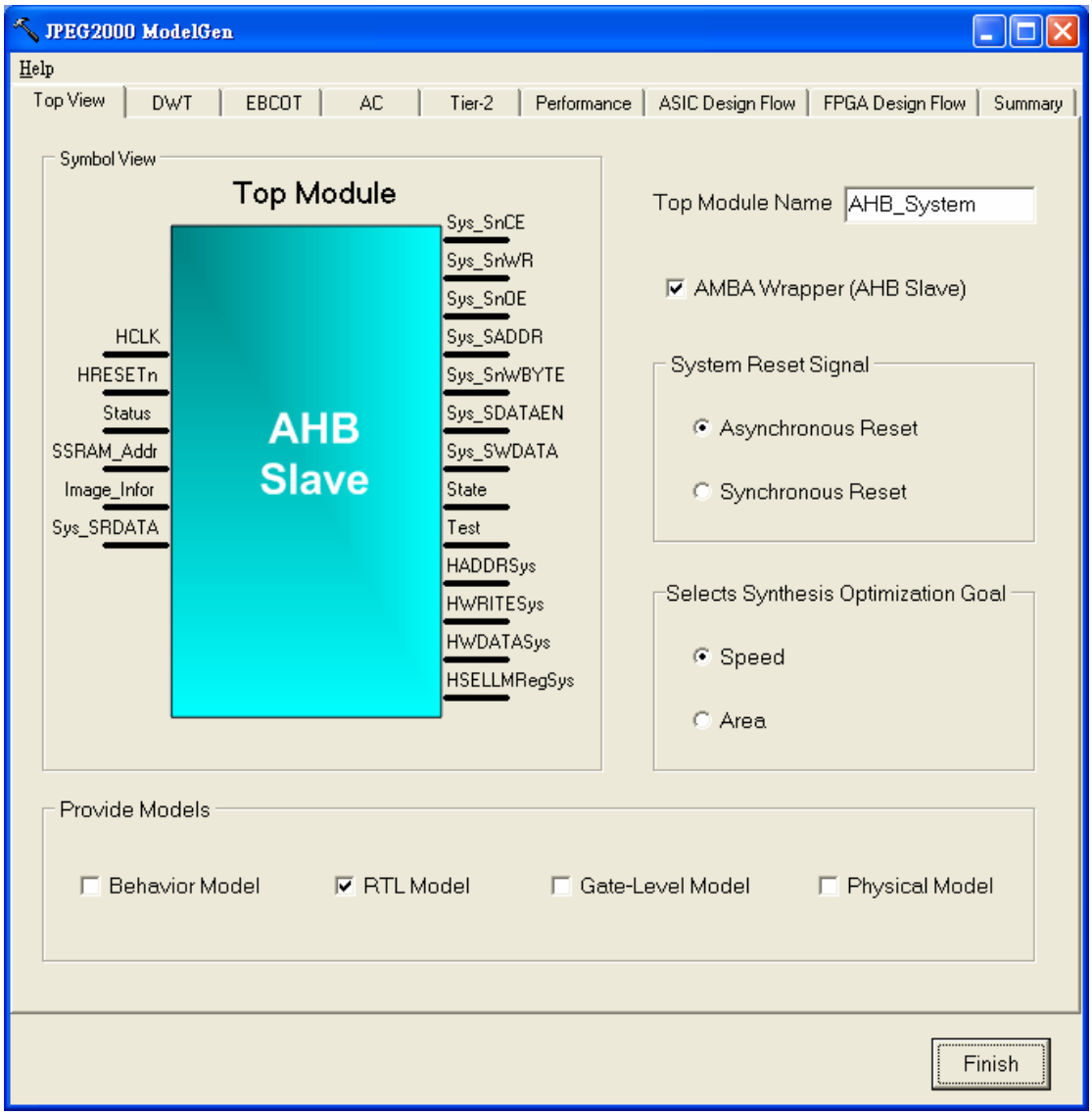


Figure 3-6 JPEG2000 ModelGen-1.

- **3 Pair Entropy Coder:** User can use three pair entropy coder to improve throughput but cost will increase.

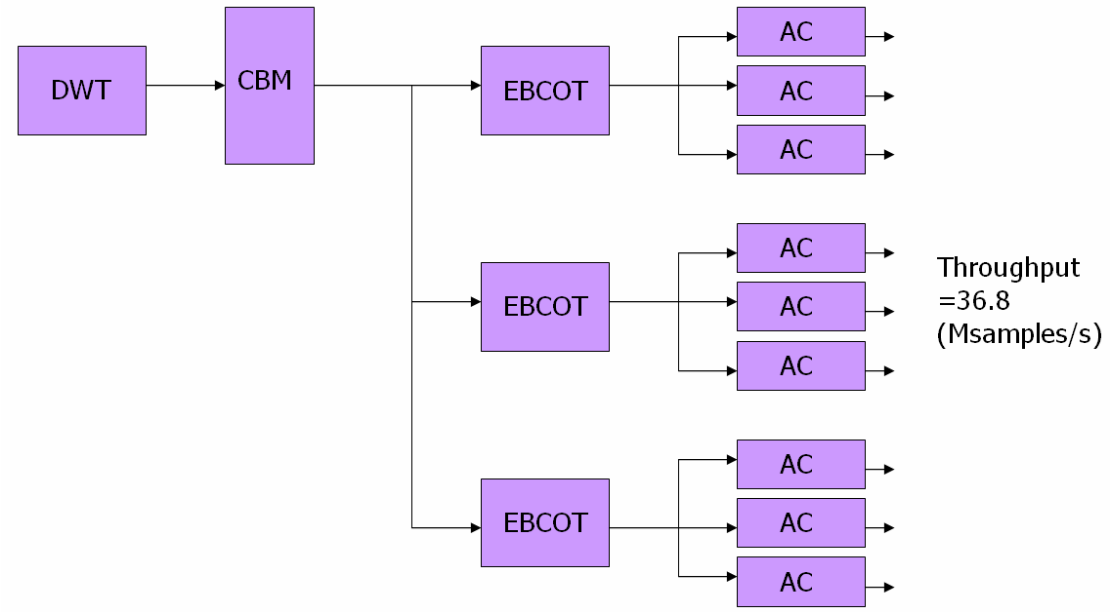


Figure 3-7 3 pair of entropy coder.

- **1 Pair Entropy Coder:** If user is concerned about area, they can use a pair entropy coder to decrease chip size.

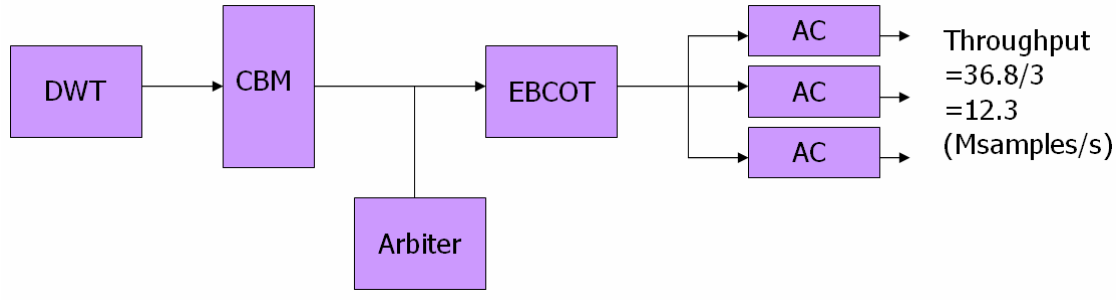


Figure 3-8 1 pair of entropy coder.

It is a compression performance pre-view window in this page as Figure 3-9. Left of page is source image and right of page is target image. User can load an image and set compression ratio to see compression performance.

Figure 3-9 JPEG2000 ModelGen-2.

In this page as Figure 3-10, the JPEG2000 ModelGen provides a post-synthesis performance estimation chart. User can choose two technology libraries to get the speed, area and power information before synthesis. It would be better to say that user can obtain the post-synthesis performance information without doing synthesis. It can reduce the iteration from RT level to gate-level.

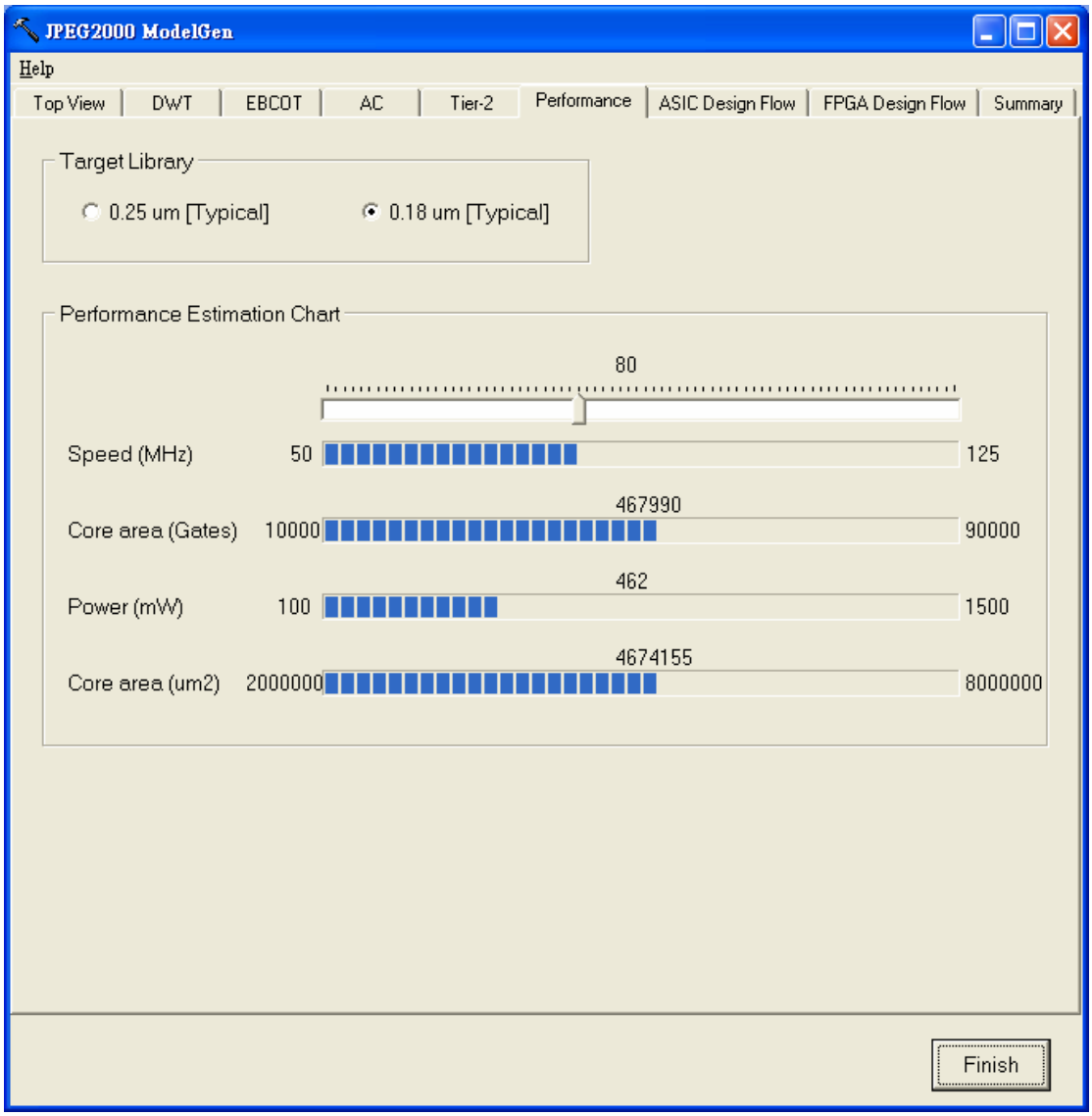


Figure 3-10 JPEG2000 ModelGen-3.

Figure 3-11 and Figure 3-12 indicate that the JPEG2000 ModelGen provides all tools' script files of ASIC and FPGA design flow. It can guarantee the objective user-defined and increase design automation.

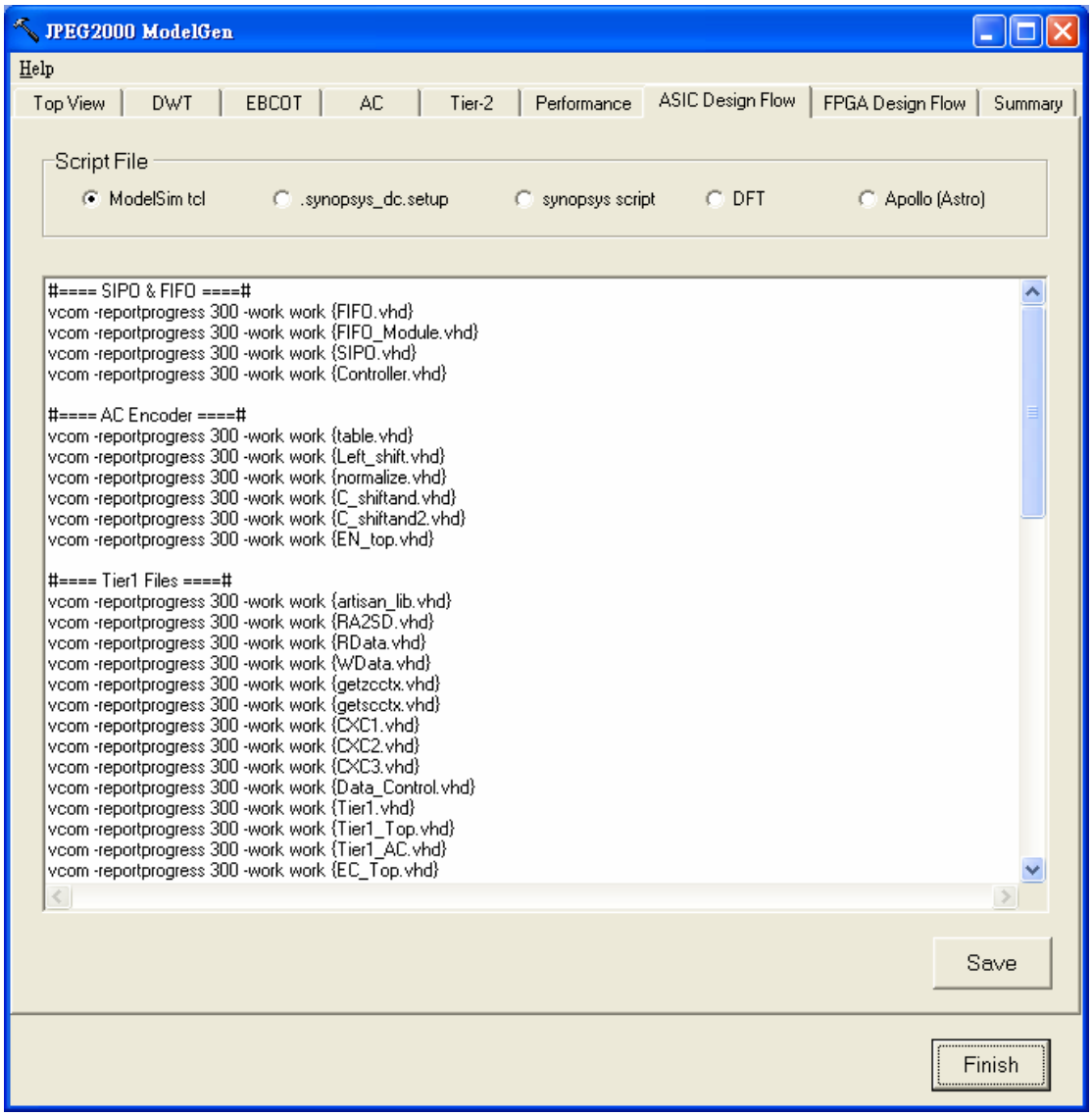


Figure 3-11 JPEG2000 ModelGen-4.

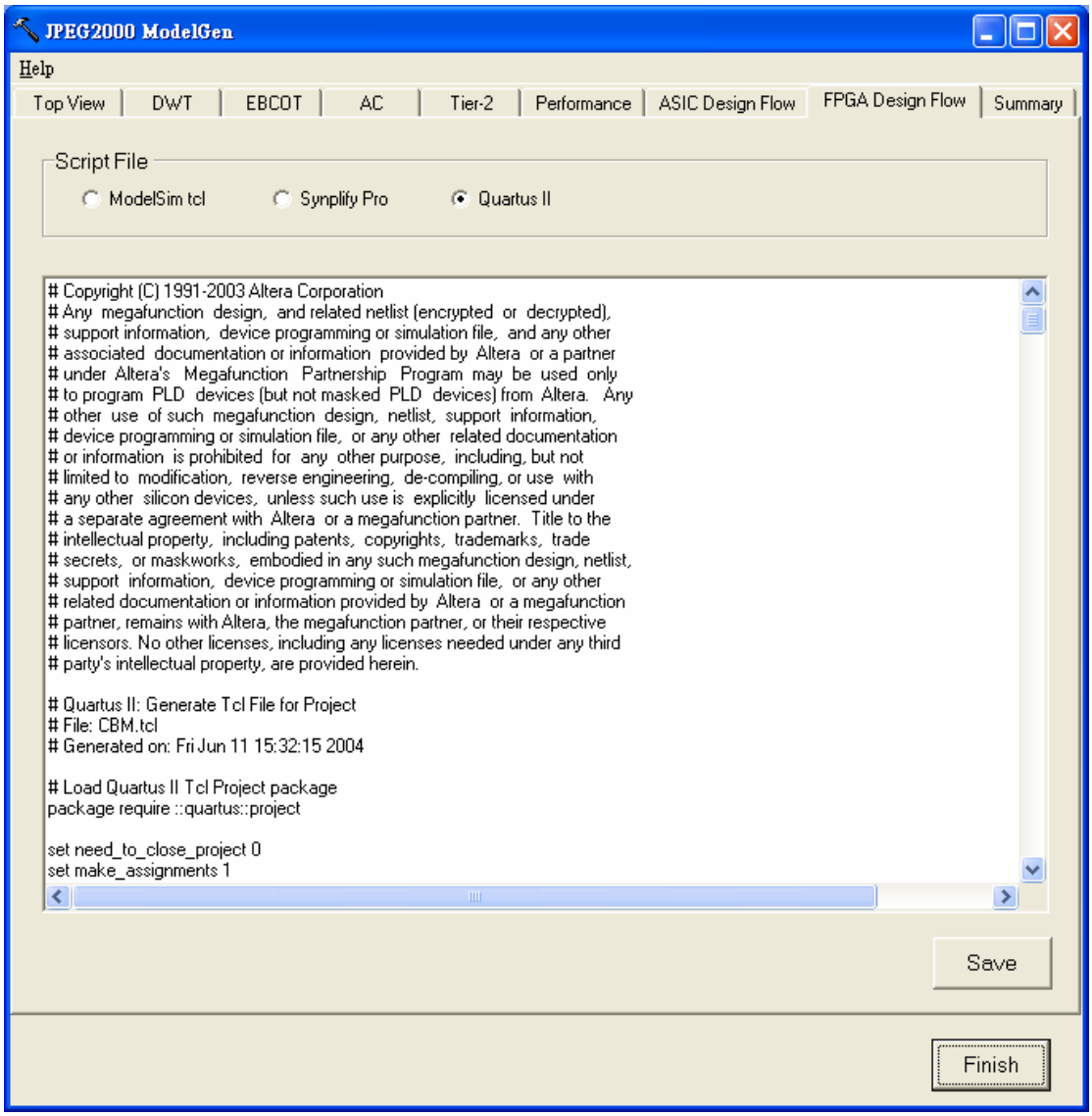


Figure 3-12 JPEG2000 ModelGen-5.

CHAPTER 4
SYSTEM INTEGRATION WITH REUSABLE JPEG2000 IP

IP reuse is one of the keys for SoC design productivity improvement. These deep submicron designs rely on increasingly sophisticated design tools and the reuse of preexisting IP. The success or failure of SoC projects rest squarely quality of the IP and the ability of the tools that integrate them.

In this chapter, we will discuss how to integrate JPEG2000 encoder with other IPs first. The verification strategy in the design flow is then discussed in Section 4.2. Finally, result is given in Section 4.3.

4.1 Novel SoC Design Flow

New process technologies enable IC devices of extremely small sizes and very high complexity. SoC designs incorporate whole systems on a single chip, containing processors, embedded memory, peripherals, and other system blocks. Yet as complexity increases, the time to get these devices to market continues to shrink. Producing these highly complex designs requires hierarchical design teams employing a hierarchical design flow.

The flow shown in Figure 4-1 illustrates the sequence of steps that a novel SoC design. The detail of the design flow is listed below:

- **System level:**

  - Specification development.

  - System Partition.

  - Virtual prototyping.

  - HW/SW co-design and co-simulation.

  - Rapid prototyping.

  - HW/SW co-verification.

- **Register transfer level:**

  - RTL code development.

  - Functional simulation.

  - Coding style rule check.

  - Code coverage.

  - Pre-layout power estimation.

- **Gate level:**

  - Logic synthesis and optimization.

  - Memory BIST.

  - Design for testability.

  - ATPG.

  - Fault coverage.

  - Timing verification.

  - Pre-layout simulation.

- **Physical level:**

  - Place and route.

  - On-line DRC and LVS check.

  - Off-line DRC and LVS check.

  - Post-layout power estimation.

  - post-layout simulation.

- **Chip level:**

  - Silicon verification.

  - Testing.

  - Chip integration.

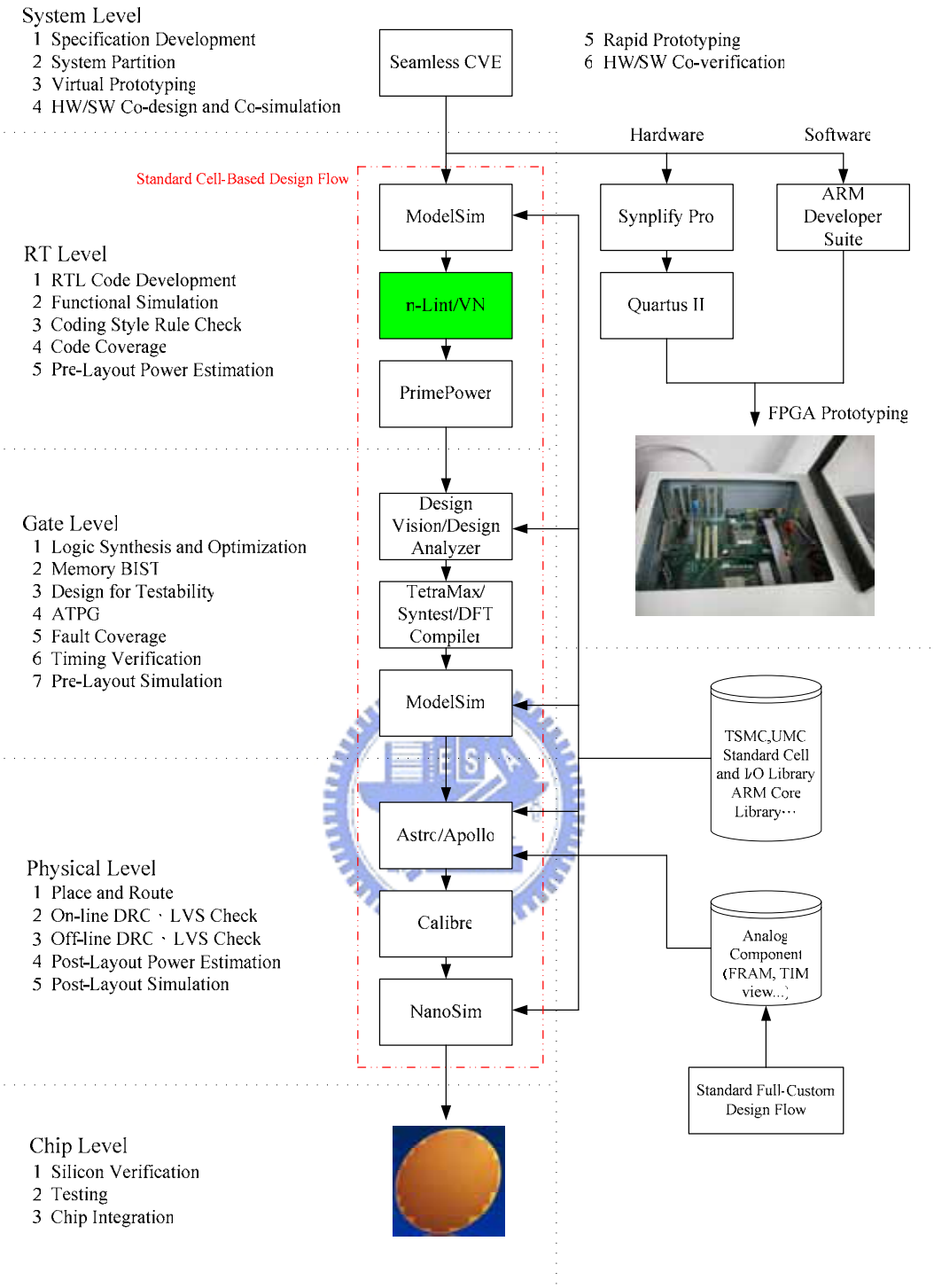


Figure 4-1 Novel SoC design flow.

4.1.1 How to Use ARM Hard Macro

The ASIC design flow diagram below shows the anticipated flow on the partner's side when implementing the ARM922T.

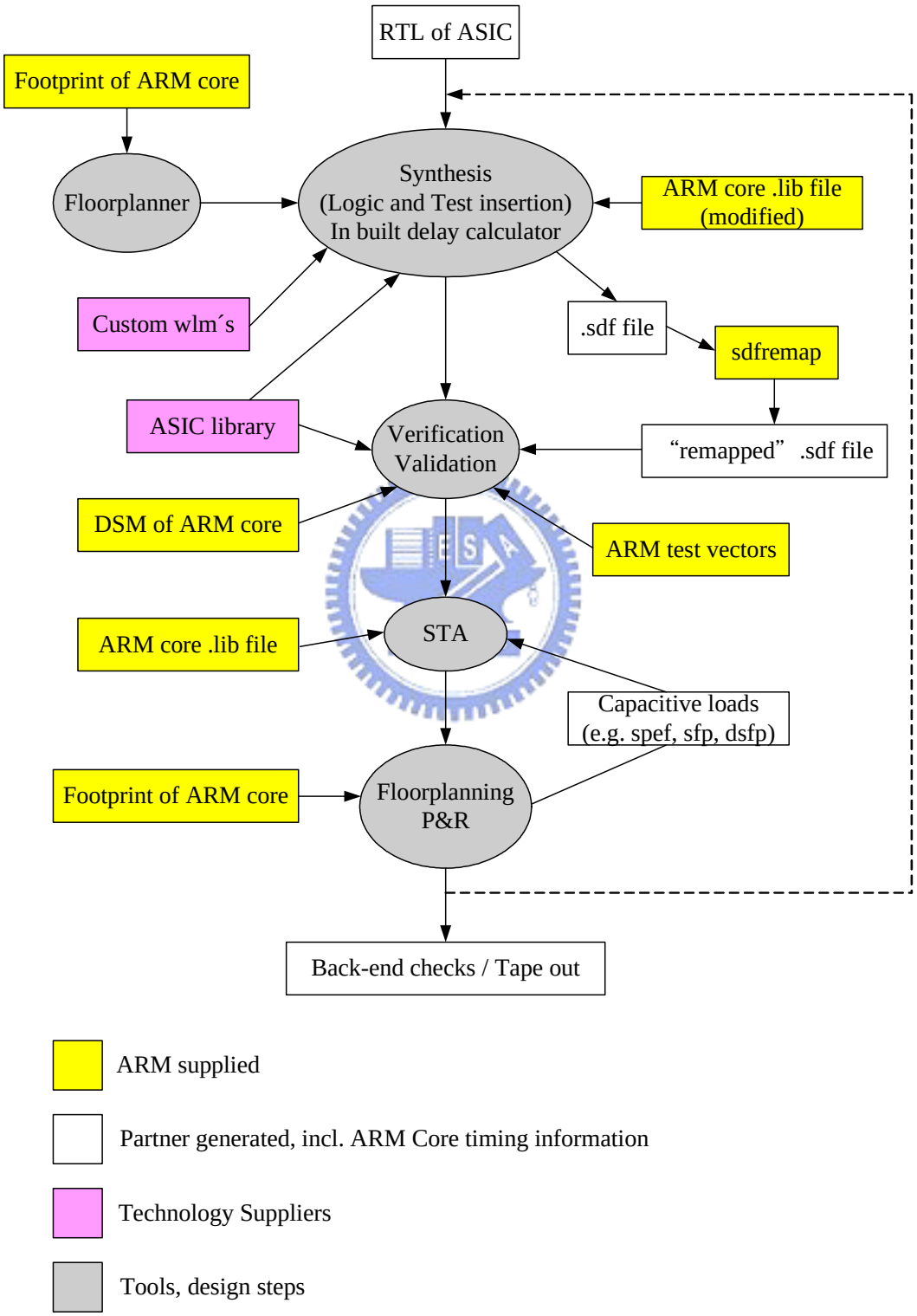


Figure 4-2 ASIC design flow implementing an ARM hard macro.

This table describes some common steps in SoC design flows and which of these design flow steps are supported by ARM.

Table 4-1 The design flow steps are supported by ARM.

Activity	Task	Methodology	Deliverables	Status of ARM support
Design Entry	RTL Capture	Text	Design Signoff Model	Fully supported. The designer will instantiate the ARM DSM, either as Verilog or VHDL.
Logic Implementation	Logic synthesis	Hierarchical or flat	Synopsys .lib file	Supported, some manual intervention required, to switch in and out the appropriate .lib file.
	Test synthesis	Scan insertion		Supported up to boundary of AHB wrapper. ARM core itself cannot be scan-inserted.
Physical Implementation	Floorplanning	Gate level or RTL level	Outline-only GDS2 Avant! .fram Cadence abstract	Fully supported. .lib file can be used to generate tool-specific timing file (e.g. .tlf, .clf) for timing driven placement.
	Place and route	Block hierarchical	Outline-only GDS2 Avant! .fram Cadence abstract	Fully supported. .lib file can be used to generate tool-specific timing file (e.g. .tlf, .clf) for timing driven placement.
	Full custom layout	Manual	Outline-only GDS2	Supported. User should note that the core should be treated as a no-go area. User should keep outside half a design rule away from the boundary.
	Parasitic extraction	Flat	Outline-only GDS2	
Verification	Functional verification	Dynamic simulation	Design Signoff model	Fully supported.
	Timing verification	Static	Synopsys .lib	Supported, some manual intervention required, to switch in and out the appropriate .lib file.
		Dynamic		Fully supported.
	Physical verification	DRC	Outline-only GDS2	Supported. User should note that the core should be treated as a no-go area. User should keep outside half a design rule away from the boundary.
		ERC	Outline-only GDS2	
		LVS	Outline-only GDS2 Pin-only HSPICE netlist	
Productisation	Manufacturing test vector generation	TIC	TIC vectors	Fully supported.

4.2 Verification Strategy

IP reuse requires a complete verification strategy. When system is more and more complex, verification is more and more difficult. It is important that how to verify the system is correct or not. In this section, we will focus on our verification strategy.

4.2.1 Hardware/Software Co-work

Hardware/Software co-design, co-simulation and co-verification create a virtual prototyping of your embedded system months before the physical prototyping is available. By executing your embedded software on simulated hardware, the Seamless co-verification environment allows you to fully verify the hardware/software interface and accelerate the debug of firmware.

Seamless CVE encourages cooperation between hardware and firmware developers, resulting in early detection and correction of errors. Many of the design problems exposed during system integration aren't attributed to software or hardware, but the interaction between the two. Exercising the initialization code, hardware diagnostics, device drivers, board support packages, and the RTOS before tape out will expose most hardware/software interface errors and reducing system integration time.

Figure 4-3 shows the major components of the Seamless CVE co-verification architecture. These components work together as follows:

- **Software simulator** or **Host Code Execution** (HCE) performs the software portion of a co-verification session. The software simulator executes machine code that you produce when you cross-assemble or compile your target software for the specific processor.

- **Co-verification Kernel** controls communications between the software and logic simulation. It permits you to set up various aspects of the co-verification session through the Seamless CVE session window and it coordinates access to memory through the Coherent Memory Server.
- **Logic Simulator** and **Logic Simulation Kernel** perform the hardware portion of a co-verification session. You instantiate one or more processor bus-interface models and some memory models into the hardware design, which is typically expressed in a HDL. The Logic Simulator controls execution of the design. Then the Logic Simulation Kernel executes the HDL simulation of the design.

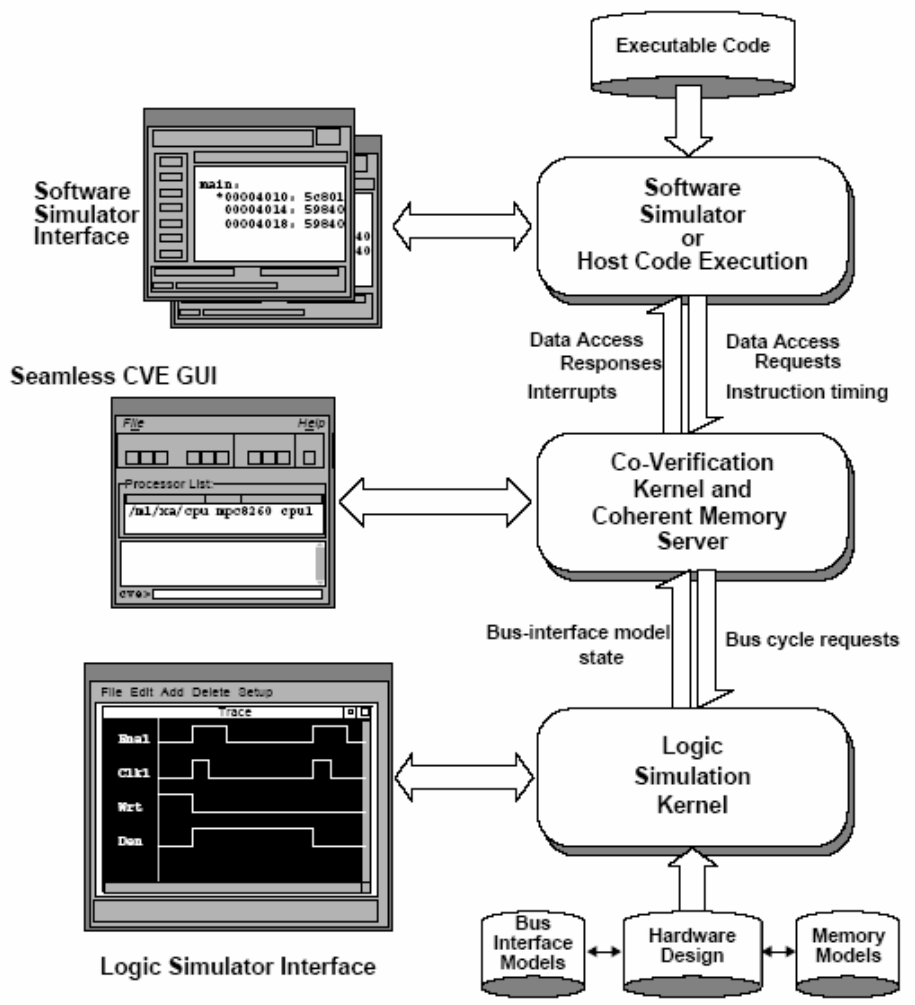


Figure 4-3 Seamless CVE architecture.

4.2.2 Functional Verification

The first point to notice is the correctness of functionality. If this part is not correct, other verification does not need to do. This result must to be compared with other level result, such as behavior level, gate level (pre-layout simulation) and physical level (post-layout simulation).

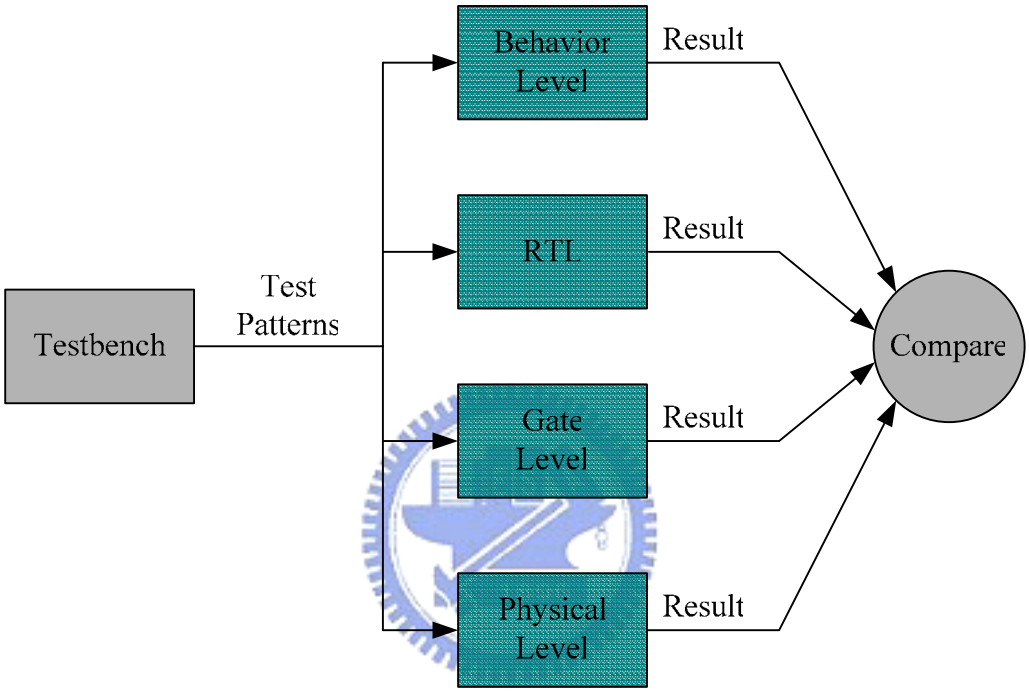


Figure 4-4 Functional verification.

4.2.3 FPGA Prototyping

The LM contains several components to facilitate rapid prototyping (Figure 4-5). The primary component is a FPGA from Altera or Xilinx. A configuration PLD and a flash memory are presented to store the FPGA configurations. A ZBT SSRAM of 1MB is provided for local storage. There's a prototyping grid where the user can attach small circuits to LM. The system bus connector provides connection to AP motherboard or to other modules.

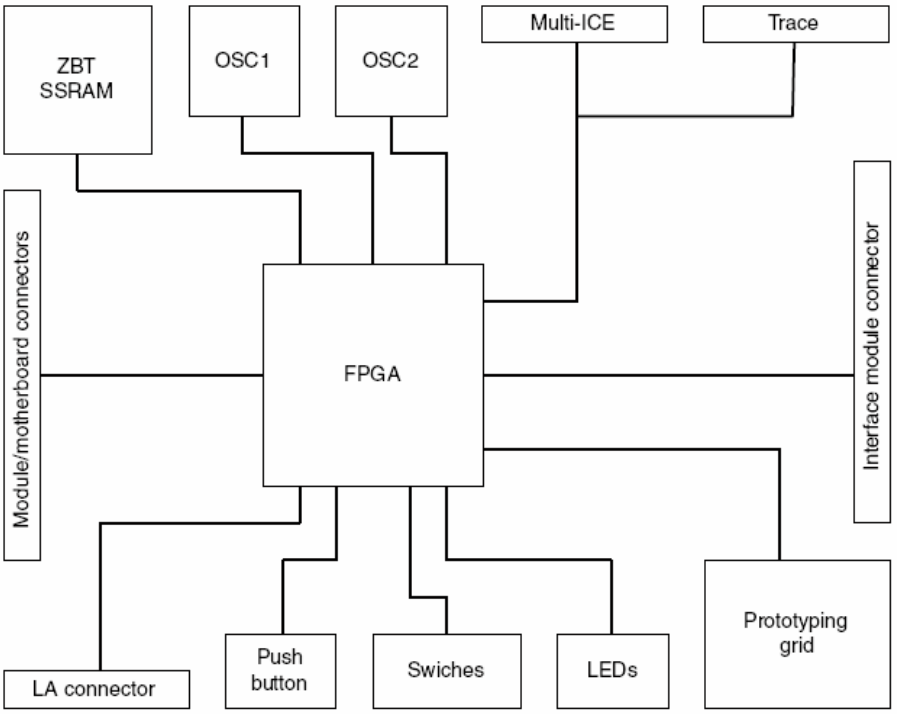


Figure 4-5 The architecture of a logic module.

Using FPGA not only to verify the SoC design itself but also to build early prototypes of the final product, to support hardware/software development and to improve the verification of system concepts are the key to overcoming this challenge. The reasons of rapid prototyping are:

- To increase effective communication.
- To decrease development time.
- To decrease costly mistakes.
- To extend product lifetime by adding necessary features and eliminating redundant features early in the design

Here is a figure which shows the JPEG2000 FPGA prototyping verification environment. In the first place, we must split the image to RGB from color image and they are input of software encoder. In the second place, the software output

coefficients are input of hardware encoder. Then, we must check the hardware output data is correct or not. Furthermore, finish the Human Visual System (HVS) check and Peak Signal-to-Noise Ratio (PSNR) check.

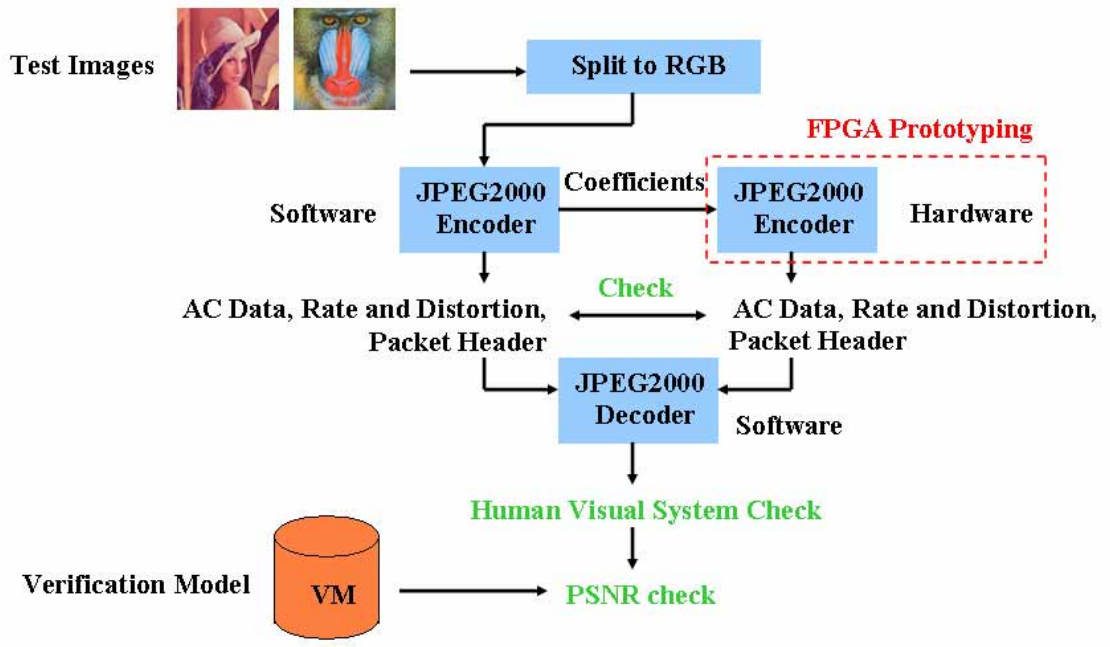


Figure 4-6 Rapid prototyping verification environment.

4.2.4 Coding Style Rule Check

A programmable rule checker has been integrated in the IP qualification framework. The predefined and configurable rules of this tool are used as much as possible and allow covering a large portion of the coding standard. Additional rules are currently being implemented using the programmability features of the tool.

The SpringSoft® nLint is used for static lint checking. A lint tool can find errors and warnings in many aspects including naming, synthesis, simulation and DFT issues. Common syntax errors, such as typing errors, unmatched bus width, undeclared objects, can be quickly located. Some logical errors like unreachable state can be found. The lint tool also indicates bad coding styles that may lead to poor reusability. By using the lint tool, many un-necessary iterations of simulation can be

saved.

In rule check, our design passes the lint tool checking with all rules defined by IPQ Alliance.

4.2.5 Code Coverage

Generally speaking, a coverage-driven verification methodology makes the verification flow more complete and efficient, and coverage report gives us a sense of the good and the bad of our HDL design and test bench.

The coverage-driven verification can be performed using several coverage metrics. A simple example of these metrics is the code coverage. By investigating the code coverage helps the designer find untested or redundant code in early stage of development and the quality of the stimuli can be measured. Therefore coverage gives the information that you need to know when you are ready for RTL sign-off. With a high coverage score, you can have more confidence that the code, in passing, works correctly.

In our design, most basic statement coverage is 100%. It means that the test patterns created by test bench are sufficient.

4.2.6 Pre-layout Power Estimation

Synopsys® PrimePower achieves a high level of accuracy with precise modeling of power dissipation. It is an advanced solution for designers who are developing leading-edge products for power-critical applications such as portable, computing, and telecommunications. PrimePower is a gate-level power analysis tool that is used by designers to explore power problems in architectures and implementations for accurate, dynamic power analysis for multimillion-gate designs.

In our design, we use PrimePower to calculate power consumption. The result is more accurate than the result in Synopsys® Design Compiler.

4.2.7 Design For Testability

The scan design or Design-For-Testability (DFT) has been a standard in ASIC design flow for most designers. Obtaining high fault coverage for a design depends on the quality of the implemented DFT logic.

DFT Compiler enables designers to conduct in-depth testability analysis at the Register Transfer Level (RTL), to implement the most effective test structures at the hierarchical block level, and, if necessary, to automatically repair test design rule checking violations at the gate-level.

In our design, fault coverage is up to 95%. Namely the higher fault coverage we reach, the better testability we get. The Figure 4-7 shows the block diagram of DFT.

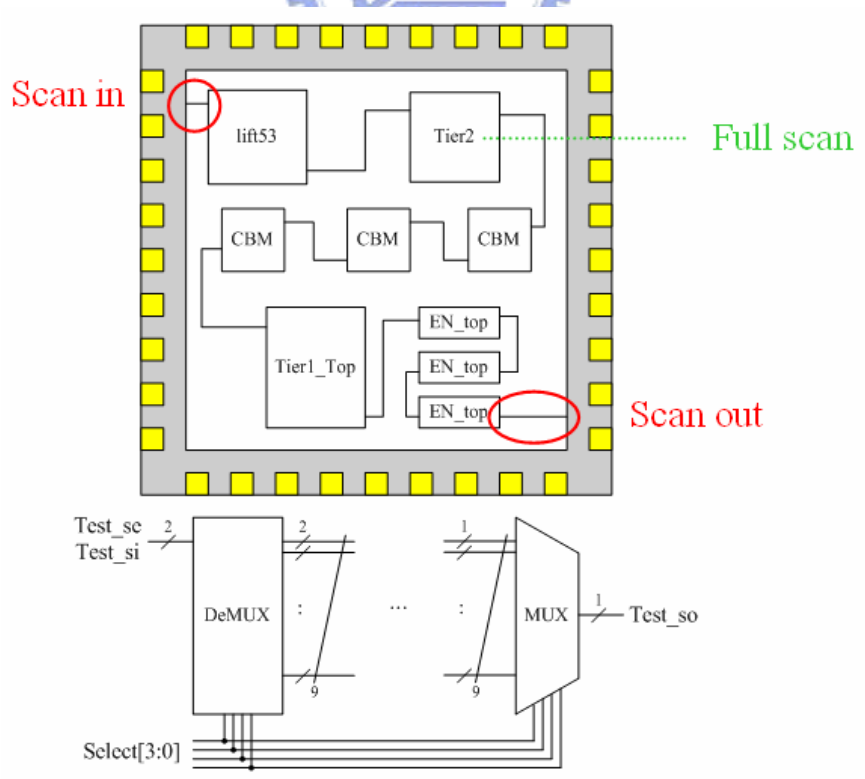


Figure 4-7 Block diagram of DFT.

4.2.8 Memory BIST

The SynTest SRAM BIST is an easy to use and power tool for creating Build In Self Test (BIST) circuit for the embedded memory of type SRAM and ROM. The embedded memory is difficult to test because its IO's are not easy to access through the primary inputs/outputs. The difficulties increase when more embedded memories are used in the design. Also, the embedded memory can not be tested at-speed through the testers. The SynTest SRAM BIST solves the problems by creating a test generation circuitry to test the embedded memory automatically.

The SynTest SRAM BIST creates a wrapper to include the BIST logic and the memory to form a bisted memory which can directly replace the original memory. This approach minimizes the re-routing effort when adding the BIST into the design. For multiple memories sharing one BIST controller, a top module is created to include all memories and the BIST controller. This top module can be used to replace all memory instances or to be used as the connection file.

There are 8.4K bytes Artisan SRAM in JPEG2000 coprocessor. The Table 4-2 below illustrates that the area overhead of memory BIST; And the Figure 4-8 shows the block diagram of memory BIST.

Table 4-2 The area overhead of memory BIST.

	Single-Port		Dual-Port			
	CBM_BUF	RA1SH_COL_64	RA2SH	RA2SH_BUF_40	RA2SH_BUF_64	RA2SH_LL_4096
RAM Spec.	1024*9	64*32	1024*2	40*16	64*16	4096*16
Total Area(mm2)	0.13	0.09	0.16	0.08	0.09	1.20
BIST Area(B)	0.02	0.02	0.01	0.01	0.01	0.01
RAM Area(R)	0.11	0.07	0.15	0.07	0.08	1.19
B/R(%)	18.2	28.5	6.7	14.2	12.5	0.9
Test Time(ns)	532,621	33,421	266,381	20,941	33,421	2,130,061
# of SRAM	x6	x2	x3	x1	x1	x1

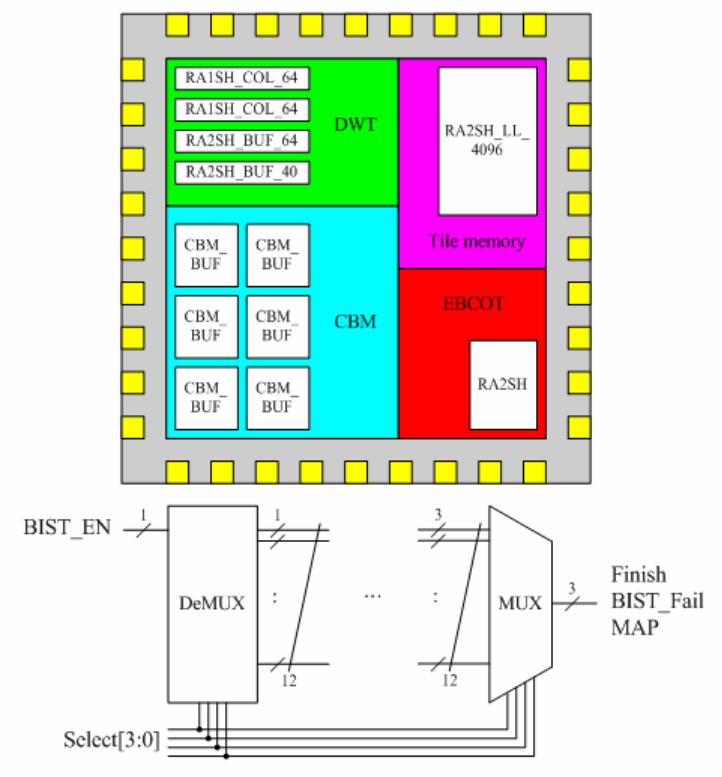


Figure 4-8 Block diagram of memory BIST.

4.2.9 Physical Verification

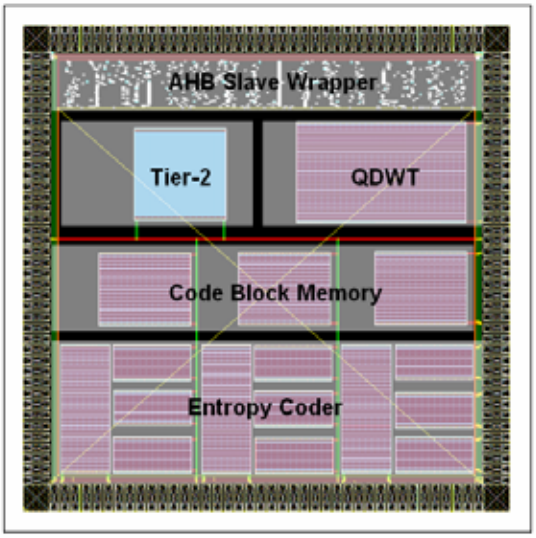
Physical design for deep submicron is most difficult in SoC. The most important issue is timing closure. Timing of the design is not easy to meet in physical level. To solve this question, robust Electronic Design Automation (EDA) tools and good coding style are needed.

In physical verification, we do Automatic Placement and Routing (APR), on-line Design Rule Check (DRC) and Layout Versus Schematic (LVS) using Synopsys® Astro; off-line DRC and LVS using Mentor Graphics® Calibre. Finally, we pass the Post-layout Simulation (Post-Sim.) using ModelSim.

4.3 Results

Judging from the above, verification will be completed as far as possible in system level in SoC, especially create a correct FPGA prototyping. This prototyping will be compared with the result of other layer. Thus we can decrease the loading of verification in physical level.

The Figure 4-7 below illustrates that implementation of JPEG2000 coprocessor. In this chip, CMOS UMC 0.18um (1P6M) technology is used. The chip size contains 335K gate count is $3mm \times 3mm$. The operation frequency can reach 50 MHz. With the operation frequency, power consumption is 100 mW. Moreover, on-chip memory is 8.4 Kbytes.



Chip Feature	
Technology	UMC 0.18um (1P6M)
Package	160-pin CQFP
Chip size	<3mm * 3mm
Gate Count	335K gates
Operation freq.	50MHz
Power consumption	<100mW
On-Chip Memory	8.4 KByte

Figure 4-9 Chip feature of JPEG2000 coprocessor.

Here is a diagram which illustrates the platform-based architecture for JPEG2000 encoding system. Figure 4-8 shows an overview of our JPEG2000 encoder platform. Mainly, it includes an ARM9 CPU for the data flow control and generating packet header production. The CPU communicates with the JPEG200 coprocessor via a 32-bit AHB bus.

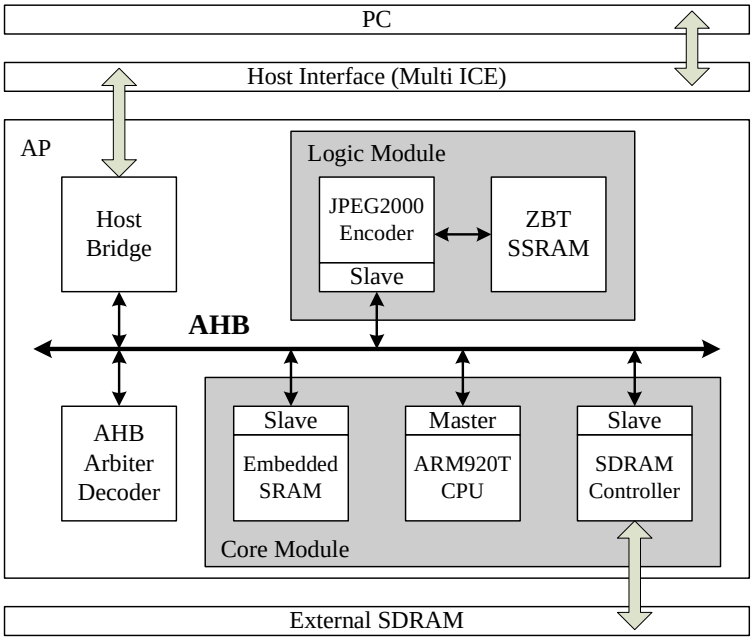
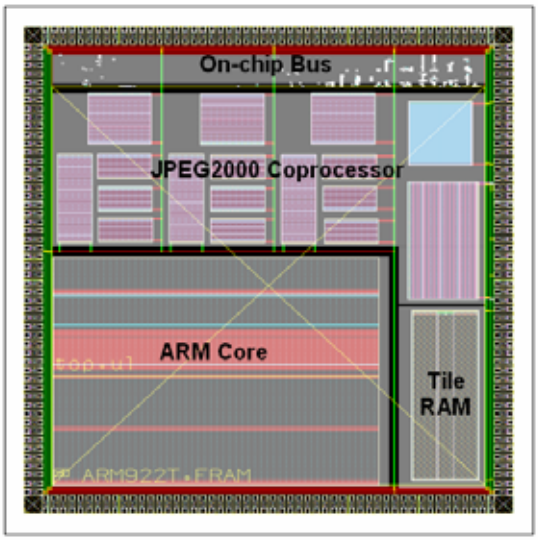


Figure 4-10 The proposed AMBA-based JPEG2000 encoder architecture.

The Figure 4-9 below illustrates that implementation of an SoC design case. In this chip, CMOS UMC 0.18um (1P6M) technology is used. The chip size contains 2M gate count is $5mm \times 5mm$. The operation frequency can reach 50 MHz. With the operation frequency, power consumption is 150 mW (ARM core excluded). Moreover, on-chip memory is 16.6 Kbytes.



Chip Feature	
Technology	UMC 0.18um (1P6M)
Package	208-pin CQFP
Chip size	<5mm * 5mm
Gate Count	2M gates
Operation freq.	50MHz
Power consumption	150mW
On-Chip Memory	16.6 KByte

Figure 4-11 An SoC design case with JPEG2000 coprocessor.

CHAPTER 5
CONCLUSIONS AND FUTURE WORKS

JPEG2000 is the superior still image compression standard due to its excellent compression efficiency and numerous novel features. One of important features of JPEG2000 is accurate rate-control scheme but it is not suitable for hardware design. As for other rate-control schemes such as quantization, they suffer the serious PSNR degradation and inaccurate rate control although their implementation is quite simple.

In this thesis, we proposed a configurable rate control architecture. In traditional DWT, we can use Real-Time Rate-Distortion Optimized (RTRD-Opt) method to reduce computation of EBCOT Tier-1. In addition, it is more appropriate to use Imitative Post-Compression Rate-Distortion Optimized (IPCRD-Opt) method in QCB-based DWT architecture.

In addition to this, we design JPEG2000 coprocessor not only an IC but also an IP. It is fact that IP design is more time-consuming than IC design. But if we design a complete IP, it is saving more development cycle when second time reuse. Finally we transplant the ARM integrator system to an SoC. The most difficult things are complete verification.

Here is a figure which shows the final objective of RTL design. To begin with,

the RTL code should be synthesizable. Then, it is better to write reusable RTL code. The reason is why we design a JPEG2000 IP but not an IC. Finally, if the RTL code is verifiable, we can integrate and verify the IP easily. It is very important when design a VLSI system.

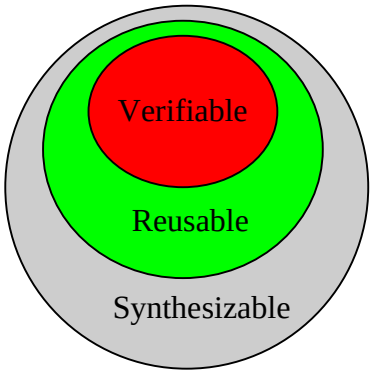


Figure 5-1 The final objective of RTL design.

The Figure 5-2 below illustrates that design layer of an IP. Bottom layer is fundamentals which everyone can study while he is a college student. Top layer is domain-spec., technology know how which everyone can research while he is a graduate student. But it is also very important that the middle layers (implementation skills for ICs/IPs) which everyone may ignore easily. That is to say if we want our design better, these two layers are key points.

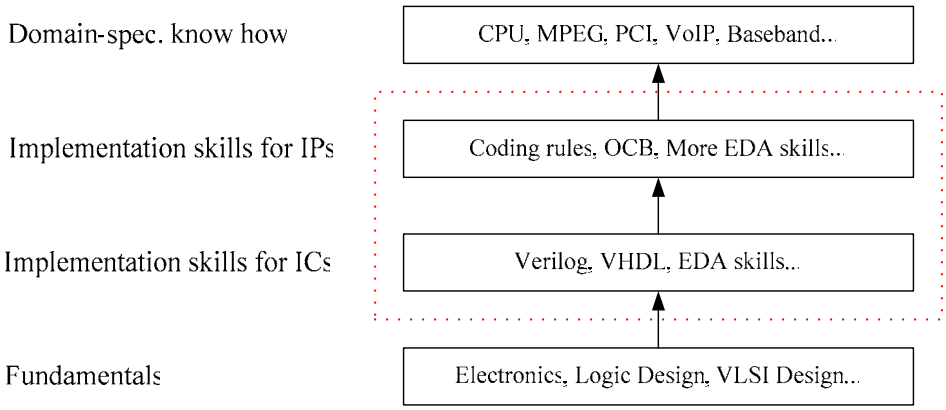


Figure 5-2 Design layer of an IP.

IP reusability has been a key issue in SoC design for many years. We have faced many situations where IP have being developed for a specific application but with the emphasis on reusability. Formal verification results a key approach to guarantee the correctness of the IP versus the specification.

REFERENCES

[1] ISO/IEC JTC1/SC29 WG 1 N1684, "JPEG2000 Part I Final Committee DraftVersion 1.0."

[2] Te-Hao Chang, Chung-Jr Lian, Hong-Hui Chen, Jing-Ying Chang, Liang-Gee Chen, "Effective hardware-oriented technique for the rate control of JPEG2000 encoding", *IEEE International Symposium on Circuits and Systems(ISCAS'03)*, Vol. 2, pp. 25-28, May 2003.

[3] Yeung Y.M., An O.C., Chang A., "Successive bit-plane rate allocation technique for JPEG2000 image coding", *IEEE International Conference Acoustics, Speech, and Signal Processing(ICASSP '03)*, Vol 3, pp. 6-10 April 2003.

[4] Masuzaki T., Tsutsui H., Izumi T., Onoye T., Nakamura Y., "JPEG2000 adaptive rate control for embedded systems", *IEEE International Symposium on Circuits and Systems(ISCAS'02)*, Vol 4, pp. 26-29 May 2002.

[5] B. F. Wu and C. F. Lin, "Analysis and Architecture Design for High Performance JPEG2000 Coprocessor," in Proc. *IEEE International Symposium on Circuits and Systems*, vol. 2, pp.225-228, May, 2004.

[6] Wei-Chang Tsai, Chun-Ming Huang, Jiann-Jenn Wang, Chen-Yi Lee, "Infrastructure for education and research of SOC/IP in Taiwan", *IEEE International Conference Microelectronic Systems Education*, pp. 150-151, June 2003.

[7] T. H. Chang, "Efficient JPEG2000 Encoder Design With Novel Rate-Distortion Optimizing Techniques," *Master's Thesis, Dept. of EE, National Taiwan University*, June 2003.

[8] C. W. Huang, "AHB-based JPEG2000 Coprocessor System Design," *Master's Thesis, Dept. of ECE, National Chiao Tung University*, June 2004.

[9] P. C. Chen, "Design of the efficient Pass-Parallel Context Formation Codec for JPEG2000," *Master's Thesis, Dept. of ECE, National Chiao Tung University*, July 2004.

[10] Virtual Socket Interface Alliance (VSIA); VSIA Homepate; 2001; http://www.vsi.org

[11] IP Qualification Alliance “IP Qualification Guidelines(Verilog Version)” Version 1.0, Release Date: Dec. 19, 2003.
[12] IP Qualification Alliance “IP Qualification Guidelines(VHDL Version)” Version 1.0, Release Date: Oct. 22, 2004.
[13] Keating, M.; Bricaud, P.; Reuse Methodology Manual for System-on-a-Chip Designs; Third Edition; Kluwer Academic Publishers; 2002.
[14] 黃俊達, IP Core Design 上課講義, 2004 Fall, Dep. of EE, National Chiao Tung University.
[15] 張添烜, SOC Design Lab 上課講義, 2005 Spring, Dep. of EE, National Chiao Tung University.
[16] CIC(Chip Implementation Center)寒暑期上課講義。
[17] 吳炳飛 胡益強 瞿忠正 蘇崇彥，JPEG2000 影像壓縮技術，全華科技書股份有限公司，2003 年 4 月初版。
[18] 戴顯權 陳政一，JPEG2000，紳藍出版社，2002 年 11 月。
[19] ARM Documents http://www.arm.com
[20] JPEG2000 Information http://www.jpeg.org/jpeg2000/index.html
[21] JasPer Software http://www.ece.uvic.ca/~mdadams/jasper/