Quality-Driven Design for Video Applications

Cao, Yun
Department of Computer Science and Communication Engineering, Kyushu University

Yasuura, Hiroto
Graduate School of Information Science and Electrical Engineering, Kyushu University

出版情報：IEICE Transactions on Fundamentals of Electronics. E85-A (12), pp.2568-2576, 2002-12. 電子情報通信学会
バージョン：
権利関係：

PAPER *Special Issue on VLSI Design and CAD Algorithms*

Quality-Driven Design for Video Applications

Yun Cao[†], *and* **Hiroto Yasuura**[††], *Member*

SUMMARY This paper presents a novel system-level design methodology, called quality-driven design, by which application-specific optimization can be achieved; furthermore the entire functionality can be shared to maximize design reuse. As a case of study, this paper focuses on quality-driven design for video applications and introduces an output quality adaptive approach based on variable bitwidth optimization to explore a new design space. MPEG2 video is used as the driver application to illustrate the potential of the presented methodology. Experimental results show the effectiveness of the methodology.

key words: Quality-Driven Design, Variable Analysis, Design Reuse, Application-Specific Optimization

1. Introduction and Motivation

Semiconductor technology allows designers to build chips with millions of transistors. Future advancements in technology will allow us to complete a system on a single piece of silicon. Moore's Law has enabled us to increase the complexity of integrated circuits at an exponential rate. At the same time, market competition is shortening the product lifetime and therefore the design cycle must fit into a fraction of the time. Furthermore, an exponentially more number of devices have to be coped with. These result in the gap between manufacturing capability and design productivity. All of these problems leave the designers to develop and apply a new generation of more suitable design paradigms, methods and tools.

Extensive research on system-level design methodology has been conducted in these recent years. Such methodologies can be divided into two groups roughly. One group is design optimization methodologies [1] [2] [4] [3] [5], which optimize performance and/or power/energy and/or cost in various ways. Another group is design reuse and core/IP methodologies [6] [7] [8], which target to reduce design complexity and save design-cycle time throughout the design process to reduce the gap between technology advancement and design productivity.

This paper presents quality-driven design, a system-level design methodology targeting both design optimization and design reuse to explore a new design space. On the one hand, quality-driven design is a kind of user-application-oriented design, it basically consists of building the quality models, using them for constructing, selecting and improving the design solutions, which satisfy the users' requirements and are optimal for applications. Therefore, one of its targets is furthering optimization by using characteristics of applications. On the other hand, by sharing functionalities for different system design, design reuse can be achieved. In addition, parameterized function modules are used to design systems, which decrease design complexity and design time to meet the requirement of time-to-market. Therefore, quality-driven design paradigm contributes to modeling, analysis, optimization and design reuse.

Quality-drive Design proposes a new design constraint, *quality*, different with general design constraints, performance, power/energy and cost. In addition, it involves in the following key techniques:

* Parameter Selection
* Measurement of Output Quality
* Optimization under Quality Constraint
* Design of Function Module

What are the parameters should be selected? They are must be *quality sensitive*, while do not change the functionality of the system. Such as variable bitwidth, datapath width, the number of iteration and so on. Measurement of output quality is an important research area, the paper [18] discussed the problem. Under the constraint, quality, optimizing design for Hardware/ Software to achieve high performance and/or low power/energy and/or low cost is also a key technique for quality-driven design. In addition, designing of function module for maximizing reuse is a key issue.

As a case of study, this paper focuses on quality-driven for video applications. An output quality adaptive approach by variable bitwidth optimization is proposed. Video quality measurement is discussed. MPEG2 video is used as the driver application. Experimental results show the effectiveness of the proposed methodology.

The rest of this paper is organized as follows. Next section presents quality-driven design methodology. Section 3 presents quality-driven design for video applications by variable bitwidth optimization. Experiments and results are reported in Section 4. Finally, Section 5 concludes our work.

Manuscript received March 15, 2002.
Manuscript revised May 18, 2002.

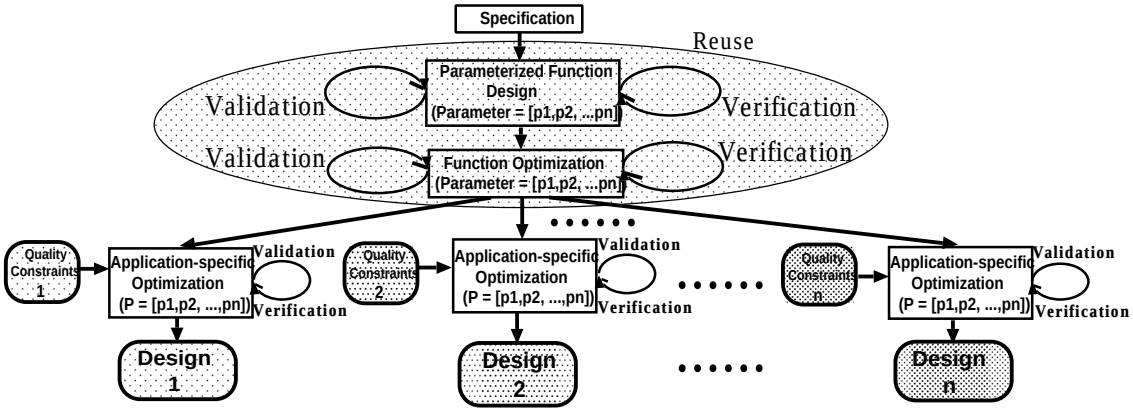

Fig. 1 A flow of quality-driven design

2. Quality-driven Design (QDD)

As the complexity of the products under design increases, the development efforts increase rapidly. However, development time must be shorter and shorter to meet time-to-market requirements. Short development times often imply to meet changes in specifications. Hence, flexibility, the capacity of the platform to adapt to different functionalities without significant changes, is a very important criterion. In order to introduce such a design platform to meet the requirement of time-to-market while keeping design quality, quality-driven design is presented.

An essential concept of the quality-driven design methodology is the "quality constraint" in design process, which allows new effective design space exploration. Quality-driven design components are within a certain degree of parameterization. Therefore it can reduce design costs and development time; in the meantime design reuse can be achieved by sharing functionalities. The overall goal of quality-driven design can be summarized as follows:

Minimize: Development time, product cost and power/energy consumption

Subject to: Constraints on quality, performance and functionality of the system

Figure 1 shows the presented flow for quality-driven design. It mainly consists two phases. In the first phase, from function specification of a system, we do a function design, which is validated and verified completely with sufficient output quality. The function design has several design parameters, such as datapath width, the number of iteration, supply voltage and clock frequency and so on. Then we optimize the function design under general constraints, performance/power/cost on system-level. The availability of parametric modules allows the system designer to explore different algorithms and architectures and to choose the most efficient solution in terms of both system and technological realization.

In the second phase, we do application-specific optimization. For a given requirement of output quality to the system, we tune each parameter to optimize performance/power/cost while keeping the given quality constraint. For the system implementation, we can use hardware, software and hardware/software codesign.

Function design can be reused for different system design, because we use parameterized components. Functional correctness is guaranteed by preverified function design. Therefore, we can reduce design time consumed by function design, function validation and function verification through sharing functionality.

For a chosen application, systems can differ very much in their input-output sub-systems. In the dashboard example, low-end and high-end products have completely different I/O requirement: the former use simple display and serial communication while the latter use very complex display and communication devices. Nevertheless, functionality is shared between the two classes of products and a common platform introduced by quality-driven design could be used to application specification design while achieving design reuse.

The aim of quality-driven design is to introduce a methodology to produce a design platform that can be easily programmed to meet a variety of applications. We are currently focusing on video applications. We aim to enable significant design space exploration by means of a highly automated environment for application specific design.

3. QDDV by Bitwidth Optimization

As a case of study, this section presents quality-driven design for video applications (QDDV) by bitwidth optimization. Mpeg2 video is used as the application and computation precision is used as the design parameter.

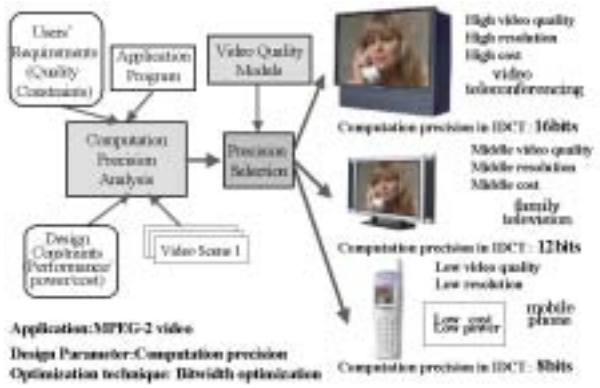

Fig. 2 A flow of QDDV

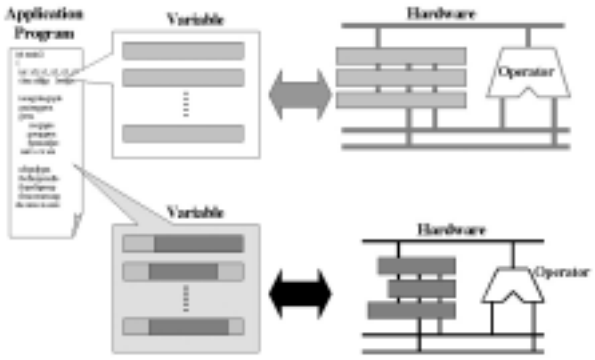

Fig. 3 Effects of variable bitwidth on hardware

3.1 Basic Approach

For digital system design, the bitwidth of a data computed in a system is one of the most important design parameters related with performance, power and cost of the system [14]. Datapath width and size of memories strongly depend on the bitwidth of the data. System designers often spend much time to analyze the bitwidth of a data required in the computation of a system [15]. Hardware designers of portable multimedia devices usually reduce datapath width (bitwidth of registers and bitwidth of operation units) [13]. Programmers of embedded systems work hard for adjustment of the bitwidth of a variable to keep the accuracy of computation. By controlling datapath width, we can reduce cost [5] and power consumption drastically [12]. Furthermore, we can choose the computation precision really required for each application to further optimization for application-specific design. In video processing, for instance, the required quality of video, such as resolution and levels of color, strongly depend on the characteristics of output display devices. We can reduce the computation precision in a target application program, if the reduction does not induce decrease of output quality. It means that we can design a video system with the minimum hardware and energy consumption by eliminating redundant computation. We call the design methodology, quality-driven design for video applications(QDDV).

Figure 2 shows the flow of our presented QDDV. In the first phase of a system design, we concentrate on the implementation of the functionality of the system and optimization for general constraints, performance/power/cost. Initial designs are written in high-level language, such as C in which most variables are assumed to be 32 bits. After the function design is validated and verified, we will enter the second phase for application-specific optimization. In this phase, we analyze bitwidth of variables in the application program, tune the design parameter, computation precision to adapt to the output quality, and do bitwidth optimization under the given quality constraint. Using QDDV, we can design different video applications with application-specific optimization while decreasing design complexity and design time.

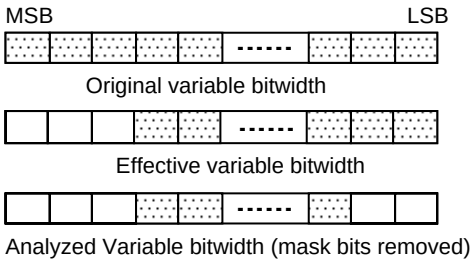

Fig. 4 Variable bitwidth analysis

Our methodology is completely general, to illustrate its technicalities, we consider the MPEG-2 video application throughout the paper. Figure 2 shows three different cases of video applications with the same functionality (MPEG-2 video decoder). In case 1, for the video conferencing with big display, so it need good quality image (computation precision of idct is 16bits). In case 2, for family television, however, needs medium quality image (computation precision of idct is 12bits). Case 3 for mobile phone, needs comparatively low quality (computation precision of idct is 8bits). If the three applications use same computation precision, it means that the computation precision must meet the requirement of video conferencing with big output display, which is "redundant" for family TV and mobile phone, result in redundant performance/power/cost. Therefore, QDDV can further optimization for system design by application-specific optimization; QDDV can improve design reuse by sharing functionalities.

3.2 Variable Bitwidth Analysis

Bitwidth has drastic effects on hardware shown in figure 3, and it also affects software strongly [10] [9] [11]. Therefore, variable bitwidth analysis is necessary [14].

It is a technique to analyze the maximum bit length of each variable in a program or an HDL description. In this section, we discuss practical methods of variable bitwidth analysis in combination of a static approach and simulation based dynamic approach.

The definition of the variable bitwidth analysis is as follows: for given inputs of a system and requirements of output quality, determine the smallest bitwidth of each variable. We have two different sub problems shown in figure 4. When the requirements of output quality are very strict, we have to find the bitwidth of variable without loss of accuracy in the computation, which called effective bitwidth analysis. 2) When the requirements of output quality are not so strict, we can choose the bitwidth of variable under the consideration of trade-off with performance/power/cost, which called output-quality-based bitwidth analysis.

3.2.1 Effective bitwidth analysis

In order to optimize performance/power/cost of a system under a given quality constraint, the effective bitwidth of each variable in an application program needs to be analyzed. This section explains methods to analyze effective bitwidth of variables in C programs. In this paper, we define *effective bitwidth* as the smallest size which can hold both maximum and minimum values of a variable [15]. In many cases, some bits of a variable are never used during execution of a program. Therefore, by analyzing effective bitwidth of variables we can reduce unused bits to reduce power consumption and cost. If a variable x of unsigned integer type whose value is in [0, 2000], then the number of necessary bits of x is 11, because the 11-bit size is large enough to hold any value in [0, 2000]. We use two methods to analyze effective bitwidth of variables. One is dynamic analysis, the other is static analysis.

For static analysis, when the maximum value of an unsigned integer variable x is n_{max}, the effective bitwidth of x, $e(x)$, is given as follows:

$$e(x) = \log_2(n_{max} + 1) \tag{1}$$

For a signed integer x with a maximum value n_{max} and a minimum value n_{min}, $e(x)$ is defined as follows:

$$e(x) = \lceil \log_2 \mathcal{N} \rceil + 1 \tag{2}$$

where

$$\mathcal{N} = \max(|n_{max}| + 1, |n_{min}|) \tag{3}$$

Static analysis is an efficient method to analyze the effective bitwidth of variables. However, in many cases when we can not predict the assigned value of a variable unless we execute the program, such as the case of unbounded loops, static analysis becomes insufficient. As a solution to this problem, we adopted dynamic analysis. In dynamic analysis, we execute the program and monitor the values assigned to each variable, and then analyze the required bitwidth of the variable.

F_0	:	a formula for analysis
x	:	a variable on the left of F_0
F_n	:	the nth formula from F_0
x_{F_n}	:	the x with minimal LBMW(x)
v_{F_n}	:	the variable on the left of F_n
$LBMW(x)$	:	the removable low bitwidth
$N_{OutSpec}(x)$	:	the given mask bitwidth by output quality

```
n = 1;
while (F_n exists) {
    if (LBMW(x) > LBMW(x_{F_n}))
    LBMW(x) = LBMW(x_{F_n});
    n = n + 1;
    if (x = v_{F_n}) {
        if (LBMW(x) > LBMW(x_{F_n})) {
            LBMW(x) = LBMW(x_{F_n});
        }
    }
}
LBMW(x) = N_{OutSpec}(x);
```

Fig. 5 Pseudo code for mask bitwidth analysis

3.2.2 Output-quality-based bitwidth analysis

By considering characteristics of each application and given output quality constraints, we analyze the variable bitwidth more tightly to further design optimization. In this paper, we define that *mask bits* are the low bits of variables, which can be removed, while do not affect the designated computation precision.

We make following assumption for the application program:

* The effective bitwidth (dynamic range) of a variable has been known
* The output accuracy is designated for each basic function block
* No floating computation, no pointer computation

For a variable x, we make following definition:
$e(x)$: effective bitwidth
$n(x)$: mask bitwidth

Arithmetic Operations

Addition, subtraction: $z = x + y$, $z = x - y$, using the result variable z to calculate $n(x)$ and $n(y)$

$$n(x) = n(z) - 1$$
$$n(y) = n(z) - 1$$

Multiplication: $z = x \times y$

Table 1 MOS grading scale

Scale	Impairment
5	Imperceptible
4	perceptible, but not annoying
3	slightly annoying
2	annoying
1	very annoying

Table 2 MPEG-2 test bitstreams

Bitstream name (Abbreviation)	resolution (pixels*lines)	bit rate (Mbit/sec)
flwr015	352*240	1.5
flwr400	704*480	40
mobl015	352*240	1.5
mobl400	704*480	40
susi015	352*240	1.5
susi400	704*480	40
sonyct1	352*224	1.5
sonyct2	704*480	40
tek3	512*512	3.5

$$\begin{array}{lll} n(x) & = & n(z) - e(y) + 1 \\ & (& \text{if } n(x) < 0 \text{ then } n(x) = 0) \\ n(y) & = & n(z) - e(x) + 1 \\ & (& \text{if } n(y) < 0 \text{ then } n(y) = 0) \end{array}$$

Logical Operations

AND, OR, XOR, NOR

Because they are bit operation, so no change in mask bitwidth. We can use the result variable z to calculate $n(x)$ and $n(y)$.

$$\begin{array}{lll} n(x) & = & n(z) \\ n(y) & = & n(z) \end{array}$$

Left shift $z = x << y$

$$\begin{array}{lll} n(x) & = & n(z) - y_{max} \\ n(y) & = & 0 \end{array}$$

Right shift $z = x >> y$

$$\begin{array}{lll} n(x) & = & n(z) + y_{min} \\ n(y) & = & 0 \end{array}$$

Figure 5 shows the pseudo code for mask bitwidth $LBMW(x)$ analysis. The basic rule to analyze the variable in basic block is that using the formula above to start from the end of basic block to the head of the block. We call it output quality adaptive approach. The mask bitwidth $LBMW(x)$ of a variable x is analyzed by the basic block which is in, and the analysis result is calculated by formula above or given by the output bitwidth of the basic block. For a conditional statement, the *then* and the *else* parts are analyzed separately. If both of the two branches have the results of the same variable, the less one will be used.

3.3 Measurement of Video Quality

In order to bring quality-driven design into effect, the definition of quality has to be solved. It should be precise enough to enable the systematic consideration, measurement and comparison of quality, which are necessary for quality-driven design. For quality-driven design for video applications, we use well-known two measures, the subjective measure Mean Opinion Scale (MOS) shown in table 1 and objective measure peak signal-to-noise ratio (PSNR). We measure the video quality of MPEG-2 video system by subjective and objective metric, which focuses on the effects of computation precision. We change the computation-precision of IDCT, the kernel program of MPEG-2 video decoder and measure video quality by some experiments. The test bitstream shown in table 2 is used in our experiments.

3.3.1 Subjective Measure

The subjective assessment of video quality has drawn attention of a number of researchers for many years, principally in relation to evaluation of new transmission or coding schemes, and in the development of advanced television standards. The standardization committees of the ISO, and in particular the CCIR, have published recommendations on the assessment of picture quality in television [16] [17] .

In our work, we follow closely the CCIR 500 recommendations with respect to subjective scales and experimental conditions. Ten observers participated in the experiment. We make use of the numerical scores (5-point (MOS) impairment scale) associated with the impairment descriptors for the computation of average MOS scores.

Because the subjective perception of noise and the behavior of MPEG-2 systems are influenced by scene attributes such as spatial detail, amount and complexity of motion, brightness, and contrast, test scenes that spanned a range of these attributes are selected.

In the experiments, the observers are asked to assign a score A(i,k) to each test bitstream, where A(i,k) is the score given by the i_{th} observer to test bitstream k. The scores are averaged to obtain the MOS value for specific image.

$$MOS(k) = \frac{1}{n} \sum_{i=1}^{n} A(i,k) \tag{4}$$

where n denotes the number of observers.

We changed the computation precision of IDCT, a kernel program in MPEG-2 video decoder and got the results of figure 6, which shows the relationship among the internal variable bitwidth in IDCT, MOSs and 9 clips of input bitstreams shown in Table 2. Here, each point in the graph represents the average quality of a MPEG-2 video decoder system, which was obtained by averaging the subjective MOSs using formular (4).

Subjective assessment tests are widely used to evaluate the picture quality of coded images, but careful

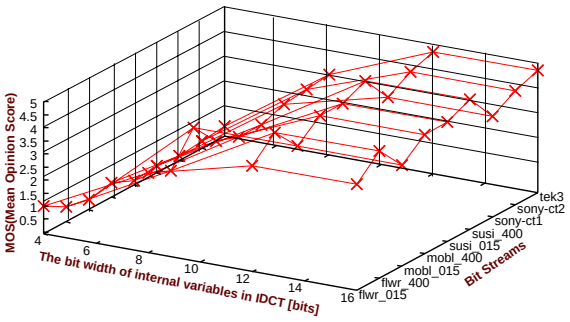

Fig. 6 MOSs of bitstreams for different internal variable bit length in IDCT

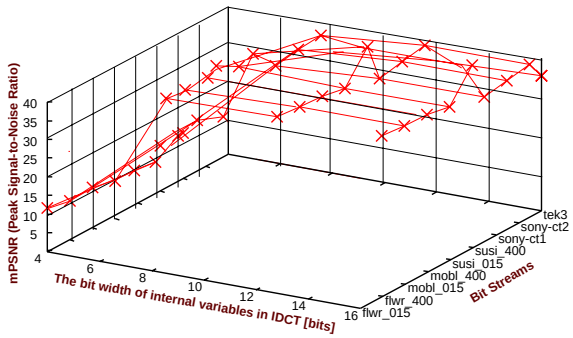

Fig. 7 mPSNRs of bitstreams for different internal variable bit length in IDCT

subjective assessments of quality are experimentally difficult and lengthy, and the results obtained may vary depending on the test conditions. Further, subjective assessments provide no constructive methods for performance improvement, and are difficult to use as part of the design process. However, the most fundamental quality measures for digital video are the subjective responses of human viewers to delivered images and subjective tests remain the only viable reference point for validating objective measures.

3.3.2 Objective Measure

Objective measures of picture quality can make the comparison of coded images, and also have the possibility of successive adjustments to improve or optimize the picture quality for a desired quality of service. The objective assessment of performance both with respect to bit rate and image quality would also lead to a more systematic design of video systems. An objective model that produces overall quality estimates would have to account for application-specific effects. The influence on accuracy of measurement is the changing expectations of people over time. For these reasons, objective video quality measurement is valid only if the application and viewer population are well defined.

PSNR is often used to specify the signal-to-noise ratio of a video signal. This method has the advantage of removing the signal power, which varies from scene to scene from the signal-to-noise-ratio (SNR) calculation so that a given SNR is indicative of some fixed amount of noise power. We calculate PSNR according to the following formulation. Assume a source image f(i,j) is given that contains N by N pixels. Then we can get a decoded image F(i,j) using original decoder and we also can get an image G(i,j) by using the decoder with changed computation precision.

The mean squared error (MSE) of the decoded image is computed as follows, the summation is over all pixels.

$$MSE = \frac{\sum[G(i,j) - F(i,j)]^2}{N^2} \tag{5}$$

$$PSNR = 10 \times \log_{10}[\frac{1}{MSE} \times 255^2][dB] \tag{6}$$

where, $PSNR$: Ratio of peak signal to noise

$$mPSNR = \frac{\sum PSNR_{G(i,j)}}{N^2} \tag{7}$$

where, $mPSNR$: Mean Ratio of peak signal to noise

Figure 7 shows the relationship among the internal variable bit length in IDCT, mPSNRs and 9 clips of input bitstreams shown in Table 2. Here, each point in the graph represents the average quality of a MPEG-2 video decoder system, which was obtained by averaging the objective PSNRs using formular (7).

The subjective quality perceived by the users that determines whether an application is adopted. The ultimate benchmark would be for objective measures to replace subjective experiments altogether. We measure video quality by conducting simultaneous subjective and objective tests.

3.4 Problem Formulation

The main concepts of QDDV can be formulated as follows:

$$\begin{array}{ll} Given & An Application Program \\ Minimize & E(CP),\ A(CP) \\ subject\ to & PSNR(CP) \geqq S_{cst} \\ & MOS(CP) \geqq M_{cst} \\ & P(CP) \geqq P(CP)_{cst} \end{array}$$

$E(CP)$ (energy consumption), $A(CP)(area)$, $PSNR(CP)$(peak signal-to-noise ratio), $MOS(CP)$ (mean opinion scale) and $P(CP)$ (performance) are functions of the computation precision CP, S_{cst}, M_{cst} and P_{cst} are the constraints on PSNR, MOS and performance P respectively.

Fig. 8 Experiments for QDDV

Table 3 idct.c

Function	# variable	# line	Description
idct	1	11	2 dimension IDCT
idctrow	9	54	row IDCT
idctcol	9	54	column IDCT

4. Experiments and Results

To evaluate our methodology, this section presents experiments and results. Video quality measurements of MPEG-2 video decode system by changing the computation precisions of IDCT, the kernel program of MPEG-2 video decoder are conducted for QDDV shown in section 3.3. This section shows design results for four kinds of systems (scenarios) using the presented methodology QDDV. We use Mpeg2 video as the functionality, the computation precision of idct as the design parameter and hardware as the design implementation. Variable bitwidth optimization is used.

Mpeg2decode C source program from the MPEG Software Simulation Group is used. It is a player for MPEG-1 and MPEG-2 video bitstreams. Mpeg2decode is an implementation of an ISO/IEC 13818-2 decoder, which emphasizes on correct implementation of the MPEG standard and comprehensive code structure. The MPEG-2 core consists of several function blocks such as IDCT, a couple of motion estimation blocks, a motion compensation block, variable length encoding, decoding blocks and so on. IDCT is the kernel part of Mpeg2decode for computation, it consists of three functions, which are *idct*() of two dimension IDCT with 11 lines, 1 variable, *idctrow*() of row IDCT with 54 lines, 9 variables and *idctcol*() of column IDCT with 54 lines, 9 variables, shown in table 3.

Our methodology is completely general, however this section only focus on IDCT, one of the heaviest computation parts in MPEG-2 video decoder system

Table 5 Results of QDDV

	Area		Power	
Quality	Area (μm^2)	Saving	Consumption	Saving
A	.002210	-	22.49 mw	-
B	.002076	6.1%	21.38 mw	4.9%
C	.001930	12.7%	20.29 mw	9.8%
D	.001823	17.5%	19.62 mw	12.6%

although the presented methodology can be used in other blocks such as motion compensation and so on. Figure 8 shows the flow used in our experiments. We explore four different systems A, B, C and D. The difference among the four systems is the required image quality (here, we use video quality measure, PSNR). We assumed that for System A the required PSNR is 38.613dB, for System B is 38.610dB, for System C is 38.501dB, and for System D is 31.623dB. In real design, we can build a library in which function modules are scalable by PSNR. Here we mean that the function modules of IDCT whose precisions are scalable by PSNR. Therefore we can determine the precision of IDCT by the given PSNR. Then we apply the variable bitwidth analysis techniques described in section 3.2 and got the table 4 of results for variable bitwidth analysis. Column "Effective Size" shows the effective bitwidth of each variable, where computation of idct is 16bits. When design B system, we got this results for optimization. Column "Size (mask bits 4)" means the bitwidth of which the low 4bits are removed, where computation precision of idct is 12bits. Similarly, Column "Size (mask bits 8)" means the bitwidth of which the low 8bits are removed, where computation of idct is 8bits. They are got for C and D system design optimization respectively.

We rewrite the program of IDCT using the variable bitwidth results for each application using VHDL language, and synthesize those using Synopsys Behavior Compiler respectively. At last, we got the table 5, which shows the design results using the proposed quality-driven design for the four systems A, B, C, D with four kinds of quality constraints. The design results(area and power) for the four designs are achieved by using HITACH 0.35 μm CMOS technology and Synopsys Design Compiler. From this table, we can see that compared to the high quality system A, system B, C, D achieved power and area reduction. The reduction achieved seems smaller than expected. We think that it should be improved if we use improved behavior synthesis tools.

It is well known that IDCT computation precision is defined in MPEG-2 standard Annex A. Generally, MPEG-2 decoder which used in family television application must be satisfied the precision. However, this work presents a new methodology which changes the precision of IDCT according users requirement. We think it is reasonable. It brings designers a new design

Table 4 Results of variable bitwidth analysis

Function	Variable	Type in C	Effective size	Size(mask bits 4)	Size(mask bits 8)
idct	i	int(32bits)	3 bits	3 bits	3 bits
idctrow	x0	int(32bits)	30 bits	25 bits	23 bits
	x1	int(32bits)	30 bits	25 bits	23 bits
	x2	int(32bits)	29 bits	25 bits	25 bits
	x3	int(32bits)	30 bits	25 bits	25 bits
	x4	int(32bits)	30 bits	23 bits	23 bits
	x5	int(32bits)	30 bits	24 bits	24 bits
	x6	int(32bits)	30 bits	24 bits	24 bits
	x7	int(32bits)	30 bits	25 bits	24 bits
	x8	int(32bits)	29 bits	24 bits	24 bits
idctcol	x0	int(32bits)	27 bits	23 bits	23 bits
	x1	int(32bits)	28 bits	25 bits	23 bits
	x2	int(32bits)	26 bits	25 bits	25 bits
	x3	int(32bits)	27 bits	25 bits	25 bits
	x4	int(32bits)	27 bits	25 bits	25 bits
	x5	int(32bits)	28 bits	25 bits	25 bits
	x6	int(32bits)	28 bits	25 bits	25 bits
	x7	int(32bits)	27 bits	24 bits	24 bits
	x8	int(32bits)	29 bits	24 bits	23 bits

space.

5. Conclusions

This paper presents quality-driven design methodology and proposes a case of study quality-driven design for video application by bitwidth optimization. In our experiments on Mpeg2 video decoder systems, we designed four systems under given four quality constraints and the experimental results show that reducing computation precision while providing certain video quality to design system is possible, and we believe that this research is perspective because it can reduce a lot redundancies, which results in reduction of cost including power consumption and areas of hardware. Furthermore, parameterized functionality can be reuse for system design to decrease design complexity and design time.

References

[1] A. Nandi, R. Marculescu, "System-Level Power/Performance Analysis for Embedded Systems Design", Proc. of the 38th ACM/IEEE Design Automation Conference, pp.599-604, June 2001.

[2] T. Givargis, F. Vahid and J. Henkel, "System-level Exploration for Pareto-optimal configurations in Parameterized Systems-on-chip", Proc. of ACM/IEEE International Conference on Computer Aided Design, pp.25-30, Nov. 2001.

[3] W. Wang, A. Raghunathan, G. Lakshminarayana and N. K. Jha, "Input Space Adaptive Design: A High-level Methodology for Energy and Performance Optimization", Proc. of the 38th ACM/IEEE Design Automation Conference, June 2001.

[4] D. Shin, J. Kim, S. Lee, "Low-energy intra-task voltage scheduling using static timing analysis", Proc. of the 38th ACM/IEEE Design Automation Conference, pp.438-443, June 2001.

[5] B. Shackleford, M. Yasuda, E. Okushi, H. Koizzumi, H. Tomiyama and H. Yasuura, "Memory-CPU Size Optimization for Embedded System Designs", Proc. of the 34th ACM/IEEE Design Automation Conference, pp.246-251, June 1997.

[6] Reinaldo A. Bergamaschi, William R. Lee, "Designing Systems-on-chip using cores", Proc. of the 37th ACM/IEEE Design Automation Conference, pp.420-425, June 2000.

[7] T. Grahm and B. Clark, "SoC Integration of Reusable Baseband Bluetooth IP", Proc. of the 38th ACM/IEEE Design Automation Conference, pp.256-261, June 2001.

[8] S. Meguerdicchian, F. Koushanfar, A. Morge, D. Petraanovic, M. Potkonjak, " MetaCores: design and optimization techniques", Proc. of the 38th ACM/IEEE Design Automation Conference, pp.585-590, June 2001.

[9] S. Mahlke, R. Ravindran, M. Schansker, R. Schreiber and T. Sherwood, "Bitwidth Congnizant Architecture Synthesus of Custom Hardware Accelerators", IEEE Transactions on Computer-aided Design of Integrated Circuit and Systems, Vol.20, No.11, Nov. 2001.

[10] H. Yasuura, H. Tomiyama, A. Inoue and F. N. Eko, "Embedded System Design Using Soft-core Processor and Valen-C", IPSJ.Info. Sci. Eng. vol. 14, pp.587-603, Sept. 1998.

[11] D. Brooks and M. Martonosi, " Dynamically Exploiting Narrow Width Operands to Improve Processor Power and Performance", Proc. of the Fifth International Symposium on High Performance Computer Architecture, 1999.

[12] Yun Cao, Hiroto. Yasuura, "A System-level Energy Minimization Using Datapath Optimization" Proc. of ACM/IEEE International Symposium on Low Power Electronics and Design, 2001.

[13] Clark N. Taylor, Sujit Dey, and Debashis Panigrahi, "Energy/Latency/Image Quality Tradeoffs in Enabling Mobile Multimedia Communication", *Proc. of Software Radio: Technologies and Services*, Enrico Del Re, Springer Verlag Ltd., pp.55-66, January 2001.

[14] Marc-Andre Cantin and Yvon Savaria, "An Automatic Word Length Determination Method", *Proc. of The IEEE International Symposium on Circuit and Systems*, V53-V56, May. 2001.

[15] H. Yamashita, H. Yasuura, F. N. Eko, and Yun Cao, "Variable Size Analysis and Validation of Computation Quality",

Proc. of Workshop on High-Level Design Validation and Test, HLDVT00, pp.95–100, Nov. 2000.
[16] CCIR, "Rec. 500-2, Method for the subjective assessment of the quality of television pictures", In *Recommendations and reports of the CCIR*, Geneva, 1982.
[17] http://www.ccir.ed.ac.uk/
[18] Yun Cao and Hiroto Yasuura, "Video Quality Modeling for Quality-driven Design", the 10th Workshop on System and System Integration of Mixed Technologies (SASIMI 2001), Oct. 2001.

Yun Cao is a PhD candidate in Department of Computer Science and Communication Engineering, Kyushu University, Japan. She received her B.E. degree of Electronics and Information Engineering from Huazhong University of Science and Technology, China, in 1989. Her research interests are hardware/software codesign, low power/low energy system design and system design methodology. She is a student member of IPSJ, IEEE and ACM.

Hiroto Yasuura is a professor of Department of Computer Science and Communication Engineering, Graduate School of Information Science and Electrical Engineering, and the director of System LSI Center of Kyushu University, Fukuoka, Japan. He received the B.E., M.E., and PhD. degrees in computer science from Kyoto University. His current interests include parallel computer architectures, hardware algorithms for VLSI, VLSI CAD, and system design methodology. He is a member of IPSJ, IEEE and ACM.