行政院國家科學委員會專題研究計畫　成果報告

子計畫六：用於奈米晶片系統設計之功率意識高階合成研究 (3/3)

計畫類別：	整合型計畫
計畫編號：	NSC94-2220-E-009-023-
執行期間：	94 年 08 月 01 日至 95 年 07 月 31 日
執行單位：	國立交通大學電機與控制工程學系(所)

計畫主持人：	董蘭榮
計畫參與人員：	楊學之、江宗錫、宋岳璋、林耕興、吳智偉、林盟淳、賴信丞、呂文豪、林毅慧、黃仕捷

報告類型：	完整報告
報告附件：	出席國際會議研究心得報告及發表論文
處理方式：	本計畫可公開查詢

中　華　民　國 95 年 10 月 24 日

行政院國家科學委員會補助專題研究計畫 ■ 成 果 報 告
□期中進度報告

先進電子設計自動化技術研發

子計畫六：用於奈米晶片系統設計之功率意識高階合成研究

(3/3)

計畫類別：□ 個別型計畫　　■ 整合型計畫
計畫編號：NSC 94－2220－E－009－023－
執行期間：　　94 年　8 月　1 日至　　95 年　7　月　31 日

計畫主持人：董蘭榮
共同主持人：
計畫參與人員： 楊學之、江宗錫、宋岳璋、林耕興、吳智偉、林盟淳、賴信丞、呂文豪、林毅慧、黃仕捷

成果報告類型(依經費核定清單規定繳交)：□精簡報告　■完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
■出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
■涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立交通大學電機與控制工程學系

中　華　民　國　九十五　年　十　月　十九　日

中英文摘要

（一） 計畫中文摘要

在 CMOS 電路尺寸逼近極限時，電路的功率消耗與效能品質的平衡考量就顯得非常重要。特別是隨著奈米技術的大幅進展，能意識到耗能條件的設計方法成為很重要的關鍵。所謂功率意識設計除了考慮平均的功率消耗外同時也包含了瞬間的功率消耗狀態，例如：峰值功率，功率梯度等。目前考慮瞬間功率消耗的設計方法僅限於電晶體或是邏輯閘階層的設計。然而，如果在系統設計階段就功率意識找出解答，積體電路的設計將可明顯地提升低階技術的功率意識最佳化程度。本計畫的目的就在於探索可用於功率意識系統的高階合成方法來管理並減少暫態功率。

關鍵詞: 功率意識；高階合成；系統晶片；電腦輔助設計

（二） 計畫英文摘要

As we get closer to the limits of scaling in CMOS circuits, it is imperative to consider power/performance trade-offs and to develop appropriate power aware methodologies and techniques for embedded systems. The use of nanometer technologies is making it increasingly important to consider transient characteristics of a circuit's power dissipation (e.g., peak power, and power gradient or differential) in addition to its average power consumption. State-of-the-art transient power analysis and reduction approaches are mainly at the transistor- and gate-levels. However, we believe architectural solutions to transient power problems may complement and significantly extend the scope of lower-level techniques, as was the case with average power minimization. This project intends to exploit high-level synthesis approach to transient power management and reduction in that a power-aware high-level synthesis can impact the cycle-by-cycle peak power and peak power differential for the synthesized implementation.

Keywords: Power-aware system; High-level synthesis; SOC; CAD

目錄

報告內容

一、 前言

With increasing demand of portable, power-aware multimedia devices, an architecture that can be flexible in both power consumption and performance is highly required. As we get closer to the limits of scaling in CMOS circuits, it is imperative to consider power/performance trade-offs and to develop appropriate power aware methodologies and techniques for embedded systems. The use of nanometer technologies is making it increasingly important to consider transient characteristics of a circuit's power dissipation (e.g., peak power, and power gradient or differential) in addition to its average power consumption. State-of-the-art transient power analysis and reduction approaches are mainly at the transistor- and gate-levels. However, we believe architectural solutions to transient power problems may complement and significantly extend the scope of lower-level techniques, as was the case with average power minimization.

When circuit technology scales through shrinking the transistor feature size by a factor of x, the capacitance is reduced by x and the supply voltage by x^2. Therefore, power decreases by a factor of x^3, provided the frequency remains the same. Unfortunately, with each generational scaling of the feature size, more complex, aggressive designs are used. These designs employ higher clock frequency, larger chip area and higher total number of transistors due to the use of more aggressive speculative execution. The result is a significant increase in power dissipation. On the other hand, aggressive, complex designs increase the opportunities available fore power management: there are more individual units which can be placed on standby when not needed by the application.

Another worrying trend is the increase in power density. Considering the Intel family of microprocessors, for instance, the power density is expressed in terms of watts/cm^2 : the current generation is getting close to the power density of a nuclear reactor. This results in more expensive cooling mechanisms and reduced reliability. The increase in total power dissipation as well as power density means that traditional power management policies centered only at the device and VLSI levels are no longer sufficient. As a result, power has propagated as an important design constraint to the higher levels. Therefore, this project intends to exploit high-level synthesis approach to transient power management and reduction in that a power-aware high-level synthesis can impact the cycle-by-cycle peak power and peak power differential for the synthesized implementation.

二、 研究目的

As mentioned above, with increasing demand of portable, power-aware multimedia devices, an architecture that can be flexible in both power consumption and performance is highly required. This project will first investigate and characterize power consumption of battery components and then come up with high level synthesis approaches to balance the power dissipation and performance and thus save the power consumption while maintaining required system performance. Given the transient power constraints, the proposed project has four goals: to have the longest battery lifetime while achieving the performance goals, to deliver task schedule and resource allocation automatically, and to synthesize the SOC architectures at system level.

三、 文獻探討

Currently, power-aware systems research at the architectural level for power saving is concentrated on the following issues: instruction set architecture (ISA) selection, instruction caches (I-cache) and the system bus, voltage and frequency scaling, battery-consciousness, and task movement.

1. **Instruction Set Architecture (ISA) Level**: This is an active research area in the context of general-purpose architectures; various researchers have commented on the need to take power and energy into account in ISA design. However, not much effort has been devoted to power-aware ISA design. Paper [1] employs a fine-grained off-line scheduling approach which saves power by combining multiple instructions into on complex but lower power instruction or by using low-power versions of instructions while considering task deadlines. The proposed scheme assumes that the ISA is sufficiently flexible; however, in practice there is not much scope for the existence of complex instructions which are functionally equivalent to a group of simpler instructions in the ISA design.
2. **I-cache and Buses**: The control path, which governs the fetch, issue and retiring of instructions, is quite simple in typical embedded processors and occupies a relatively small portion of the chip area. The caches take up most of the chip area [2] and are responsible for a considerable percentage of the energy dissipation even though memory is more energy efficient than control logic. Paper [3] compresses the instructions in memory. This saves instruction fetch energy by using fewer bits on a fetch. An alternative strategy by paper [4] also concentrates on saving instruction energy. The authors employ a loop cache and keep the tight loop in a small loop cache instead of accessing a larger block.

This paper shows the usefulness of augmenting an ISA in a power-aware fashion.

3. **Voltage and Frequency Scaling**: In general, complex systems are typically over-designed, provisioning resources for the worst-case execution time. Since tasks rarely execute up to their worst case, there is significant scope for power and energy savings using dynamic voltage and frequency scaling. Papers [5]-[8] are in this category.
4. **Battery Consciousness**: The most important issues to be considered for battery-driven systems are the total battery capacity and the battery discharge profile. The latter is important in devising battery-aware schemes that are guided by the discharge profile. Paper [9] considers distributed real-time systems and develop battery model, which is used in two scheduling schemes: first they optimize the battery discharge power profile, and then they use voltage scaling for distributed real-time systems. The overall objective is to extend the battery lifespan while meeting task deadlines and precedence requirements. The authors claim that mitigating battery capacity loss requires reducing the discharge current level and shaping its distribution.
5. **Task Movement**: Task movement is important in real-time systems for fault-tolerance or load balancing purposes. However, power efficient task movement heuristics have not been extensively investigated. One exception is the work of paper [10]. The paper is based on the observation that a set of processors can operate at a lower power level than a single one with the same performance if there is enough parallelism.

四、 研究方法

We consider the project as three parts: transient power management thru high-level synthesis, system-level power-aware design automation, and high-level synthesis for adaptive power-quality tradeoff in energy-aware multimedia embedded systems. The yearly schedule is shown as follows:

1st Year:

1. Study on power characteristics of battery-based system.
2. Develop static scheduling algorithm under transient power constraints.
3. Demonstrate the proposed scheduling technique using state-of-the-art commercial design flow.

2nd Year:

1. Study on power aware Instruction Set Architecture (ISA).
2. Develop automated ISA selection for power aware systems.

3. Develop power-aware bus encoding techniques.
4. Develop the power-aware scheduling algorithm for dynamic voltage and frequency scaling.

3rd Year:

1. Study on battery-conscious multimedia systems.
2. Develop the adaptive power-quality tradeoff algorithm.
3. Develop the high-level synthesis for adaptive power management.

By the end of this project, we would expect as follows:

1. Publications: There will be at least two papers published in major international conferences each year. We will publish at least two academic journal papers in support of this three-year project. Also, there will be at least two Ph.D. dissertations and six master theses funded by the project.
2. CAD environment: We are going to build up a high-level synthesis tool driven by techniques from this project and embedded the tool into state-of-the-art commercial design flow.
3. Training: There will be two Ph.D. students and six master students earned their degrees within the execution period of this project.

五、 結果與討論

In this year, the project has resulted in five journal papers and one conference paper:

1. Hsien-Wen Cheng and Lan-Rong Dung, "A Power-Aware Motion Estimation Architecture Using Content-based Subsampling," Journal of Information Science and Engineering, vol. 22, no. 4, pp. 799-818, 2006.

2. Hsien-Wen Cheng and Lan-Rong Dung, "A Content-based Methodology for Power-Aware Motion Estimation Architecture," IEEE transactions on Circuits and Systems II, vol.52, No.10, pp.631-635, 2006.

3. Lan-Rong Dung and Hsueh-Chih Yang, "A Parallel-In Folding Technique for High-Order FIR Filter Implementation," accepted by IEICE transactions on Fundamentals.

4. Tsung-Hsi Chiang, and Lan-Rong Dung, "System level verification on high-level synthesis of dataflow algorithms using Petri net," accepted by WSEAS transactions on Circuits and Systems.

5. Chuan-Sheng Lin and Lan-Rong Dung, "A NAND Flash Memory Controller for SD/MMC Flash Memory Card," to be appeared in IEEE transactions on Magnetics

6. Tsung-Hsi Chiang, and Lan-Rong Dung, "System-Level Verification on High-Level Synthesis of Dataflow Graph," ISCAS 2006.

The following pages are the articles.

A Power-Aware Motion Estimation Architecture Using Content-based Subsampling*

HSIEN-WEN CHENG AND LAN-RONG DUNG
Department of Electrical and Control Engineering
National Chiao Tung University
Hsinchu, 300 Taiwan
E-mail: lennon@cn.nctu.edu.tw

This paper presents a novel power-aware motion estimation architecture for battery-powered multimedia devices. As the battery status changes, the proposed architecture adaptively performs graceful tradeoffs between power consumption and compression quality. The tradeoffs are considered to be graceful in that the proposed architecture is scalable with changing conditions and the compression quality is slightly degraded as the available energy is depleted. The key to such tradeoffs lies in a content-based subsample algorithm, first proposed in this paper. As the available energy decreases, the algorithm raises the subsample rate for maximizing the battery lifetime. Differently from the existing subsample algorithms, the content-based algorithm first extracts edge pixels from a macro-block and then subsamples the remaining low-frequency part. By doing so, we can alleviate the aliasing problem and, thus, limit the quality degradation as the subsample rate increases. Given a power consumption mode, the proposed architecture first performs edge extraction to generate a turn-off mask and then uses the turn-off mask to reduce the switch activities of processing elements (PEs) in a semi-systolic array. The reduction of switch activities results in significant power consumption savings. To achieve a high degree of scalability and qualified power-awareness, we use an adaptive control mechanism to set the threshold value for edge determination and make the reduction of switch activities rather stationary. As shown by experimental results, the architecture can dynamically operate in different power consumption modes with little quality degradation according to the remaining capacity of the battery pack while the power overhead of edge extraction is kept under 0.8%

Keywords: motion estimation, image processing, VLSI architecture, video compression, power-aware system

1. INTRODUCTION

Motion estimation (ME) has been notably recognized as the most critical part of many video compression applications, such as MPEG standards and H.26x [1], since it tends to dominate the computational and hence power requirements. With increasing demand for battery-powered multimedia devices, an ME architecture that can be flexible in both power consumption and compression quality is highly required. This requirement is driven by the user-centric perspective [2]. Basically, users have two views on using portable devices. Sometimes, users want extremely high video quality at the cost of reduced battery lifetime. At other times, users want acceptable quality with extended battery lifetime. This paper, therefore, presents a novel power-aware ME architecture that

Received February 9, 2004; revised July 6, 2004; accepted July 27, 2004.
Communicated by Pau-Choo Chung.
* This work was supported in part by the National Science Council of Taiwan, R.O.C., under grant No. NSC 92-2220-E-009-033.

uses a content-based subsample algorithm, which can adaptively perform tradeoffs between power consumption and compression quality as the battery status changes. The proposed architecture is driven by a content-based subsample algorithm that allows the architecture to work in different power consumption modes with acceptable quality degradation. Since the control mechanism and data sequences in different power consumption modes are the same in the architecture, the power-aware algorithm can switch power consumption modes very smoothly on the fly. The block diagram shown in Fig. 1 illustrates a typical application of the proposed power-aware ME architecture. The host processor monitors the remaining capacity of the battery pack and switches power consumption modes. According to the power mode, the power-aware architecture sets the subsample rate and calculates the motion vector (MV) for motion compensation. Note that most portable multimedia devices, in practice, have a battery monitor unit and power management subroutines. The host processor and battery monitor unit should not be considered as the overhead of using the power-aware architecture.

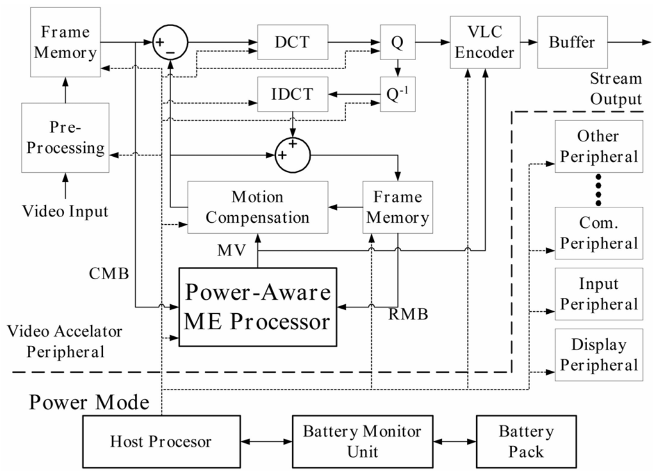


Fig. 1. The system block diagram of a portable, battery-powered multimedia device.

Many published papers have presented efficient algorithms for VLSI implementation of motion estimation, based on either high performance or low power design. However, most of them cannot dynamically adapt the compression quality to different power consumption modes. Among these proposed algorithms, the Full-Search Block-Matching (FSBM) algorithm with the Sum of Absolute Difference (SAD) criterion is the most popular approach to motion estimation because of its good quality. It is particularly attractive when extremely high quality is required. Many types of architectures have been proposed for the implementation of FSBM algorithms [3-6]. However, they require a huge number of *comparison/difference* operations and result in a large computation load and high power consumption. To reduce the computational complexity of FSBM, researchers have proposed various fast algorithms. They either reduce the number of search steps [7-12] or simplify the calculation of the error criterion [13-16]. By combining step-reduction and criterion-simplification, some proposed two-phase algorithms balance the performance between complexity and quality [17-19]. They first use FSBM with a simplified matching criterion to generate candidate vectors and then select the best motion vector from among these candidates using the SAD criterion. These fast-search algorithms successfully improved the block matching speed while limiting the quality degradation, thus achieving low power implementation. However, a low power implementation is not necessarily a power-aware system in that a power-aware system should adaptively modify its behavior according to the change of the power/energy status and achieve a balance between quality and battery life [20]. The requirement of ME algorithms to be suitable for power-aware designs is high degree of scalability in performance tradeoffs. Unfortunately, the fast algorithms mentioned above do not meet this requirement.

The authors in [21, 22] presented subsample algorithms that significantly reduce the computation cost with low quality degradation. The reduction of the computation cost implies a savings in power consumption. Since the power consumption can be reduced by simply increasing the subsample rate, the subsample algorithms have a high degree of scalability and are very suitable for power-aware ME architectures. However, applying subsample algorithms for power-aware architectures may suffer from aliasing problem in the high frequency band. The aliasing problem degrades the compression quality rapidly as the subsample rate increases. To alleviate this problem, we extend traditional subsample algorithms to obtain a content-based algorithm, called the content-based subsample algorithm (CSA). In this algorithm, we first use edge extraction techniques to separate the high-frequency band from a macro-block and then subsample the low-frequency band only. By combining the edge pixels and subsample pixels, the algorithm generates a turn-on mask for the architecture to limit the switch activities of processing elements (PEs) in a semi-systolic array. By doing so, we can achieve significant power consumption savings and limit the quality degradation as the subsample rate increases. Because the number of high-frequency pixels varies with different video clips, we use an adaptive control mechanism to set a threshold value for edge determination and make the number of masked pixels stationary for a given power mode.

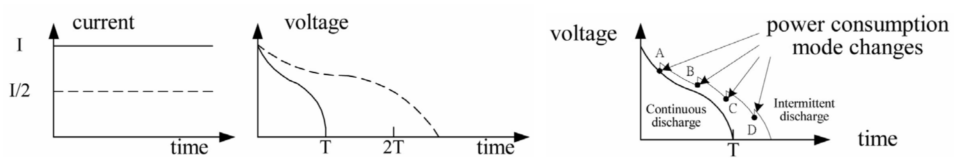
The CSA can be used in most existing ME architectures by turning off PEs according to the subsample rate. In this paper, we present a semi-systolic architecture with gated PEs. The proposed architecture shows that the CSA algorithm can dynamically alter the subsample rate as the power consumption mode changes. As shown by experimental results, the proposed architecture can work in different power consumption modes with acceptable and smooth quality degradation while keeping the power overhead of edge extraction under 0.8%.

The rest of the paper is organized as follows. In section 2, we introduce the background of the power-aware paradigm. Section 3 presents subsample algorithms in detail. Section 4 describes the proposed power-aware architecture and gives experimental results. Finally, in section 5, we draw conclusions of this work.

2. BACKGROUND

2.1 Battery Properties

One may simply consider a battery as a capacitor in which the charge capacity is linearly proportional to the output voltage. However, in practice, the behavior of a battery is less than ideal due to the variation in voltage and capacity. Two other important properties of batteries are the rate capacity effect and recovery effect [23]. The first effect means that the capacity of a battery is dependent on the discharging rate, and the second one means that a battery with an intermittent load may have a larger capacity than one with a continuous load. Fig. 2 (a) illustrates the rate capacity effect by plotting the cell voltage of two different discharging loads as time advances. As shown by the curves, when the load is halved the battery life can be more than two times longer. Fig. 2 (b) shows the recovery effect, in where the reduction of the load causes a raise of the voltage. Therefore, one can extend the battery lifetime by gradually stepping down the power dissipation. The Intel® SpeedStep™ technology, for instance, which is widely used in mobile CPUs, adopts the same strategy to extend the battery lifetime [24]. This technology changes the power consumption mode by scaling down the supplied voltage and operating frequency, hence degrading the performance in order to increase the battery lifetime.



(a) The rate capacity effect [25]. (b) The recovery effect.

Fig. 2. Non-ideal battery properties.

From these two properties of batteries, we can learn two things. First, we can reduce the load to achieve a longer battery lifetime because halving the current can more than double the battery lifetime. Second, optimal performance can be achieved when the battery is fully charged because the battery capacity can be recovered later by reducing the load. These properties provide strong motivation for developing power-aware designs and reason out the requirement of power-aware architecture – high degree of scalability in energy-quality tradeoffs.

2.2 Power Model

One can consider the major power consumption of a CMOS gate i as in Eq. (1), where C_i is the output capacitance, f_i is the operation frequency, $r_i(0 \leftrightarrow 1)$ is the switch activity of gate i, α and κ are constants:

$$P_{gate_i} = \alpha \cdot C_i \cdot f_i \cdot V_{DD}^2 = \kappa \cdot C_i \cdot r_i(0 \leftrightarrow 1). \qquad (1)$$

For an execution unit EU_j in a VLSI system, the power consumption can be computed using Eq. (2), where $N_{gate,j}$ is the gate count of EU_j:

$$P_{EU_j} = \sum_{i=1}^{N_{gate,j}} \kappa \cdot C_i^j \cdot r_i^j(0 \leftrightarrow 1). \qquad (2)$$

After considering the activity of execution units, the total power consumption can be expressed as in Eq. (3) and approximated as in Eq. (5) by assuming that the switch activities are uniform within an execution unit; that is, $r_i^k(0 \leftrightarrow 1) = r^k(0 \leftrightarrow 1)$, $\forall r_i^k(0 \leftrightarrow 1)$. Since the average output capacitances of each execution unit (C_{avg}^k) are nearly the same as the average output capacitances of the total system (C_{avg}), the total power consumption can be approximated to Eq. (8). Therefore, we can obtain an approximate power estimation model as shown in Eq. (9), where ε_{gp} is defined as the gate power coefficient. In this paper, we use the gate power coefficient as the unit for estimating power dissipation:

$$P_{total} = \sum_{\forall \text{inactive} EU_j} P_{EU_j} + \sum_{\forall \text{active} EU_k} P_{EU_k} \tag{3}$$

$$= \sum_{\forall \text{inactive} EU_j} \kappa \sum_{i=1}^{N_{gate,j}} C_i^j \cdot 0 + \sum_{\forall \text{active} EU_k} \kappa \sum_{i=1}^{N_{gate,k}} C_i^k \cdot r_i^k(0 \leftrightarrow 1) \tag{4}$$

$$\cong \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \sum_{i=1}^{N_{gate,k}} C_i^k \tag{5}$$

$$= \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times \frac{\sum_{i=1}^{N_{gate,k}} C_i^k}{N_{gate,k}} \times N_{gate,k} \tag{6}$$

$$= \kappa \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times C_{avg}^k \times N_{gate,k} \tag{7}$$

$$\cong (\kappa \cdot C_{avg}) \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times N_{gate,k} \tag{8}$$

$$= \varepsilon_{gp} \sum_{\forall \text{active} EU_k} r^k(0 \leftrightarrow 1) \times N_{gate,k}. \tag{9}$$

3. SUBSAMPLE ALGORITHMS

3.1 Generic Subsample Algorithm

Many published papers have presented efficient algorithms for VLSI implementation of motion estimation [1, 3, 5, 6, 15, 19]. The FSBM algorithm with the SAD criterion is the most popular approach to motion estimation because of its good quality and regular data path. The algorithm uses Eqs. (10) and (11) to compare each current macro-block (CMB) with all the reference macro-blocks (RMB) in the search area to determine the best match and the motion vector is found in Eq. (11):

$$SAD(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} | S(i+u, j+v) - R(i, j) |, \ -p \le u, v \le p-1. \tag{10}$$

The motion vector is found using Eq. (11):

$$\overrightarrow{MV} = (u,\ v)\,|_{\min_{-p\le u,v\le p-1} SAD(u,v)}, \tag{11}$$

where the macro-block size is N-by-N and $R(i, j)$ is the luminance value at (i, j) of the current macro-block (CMB). $S(i + u, j + v)$ is the luminance value at (i, j) of the reference macro-block (RMB), which offsets (u, v) from the CMB in the search area $2p$-by-$2p$.

Much research has addressed subsample techniques for motion estimation in order to reduce the computation load of FSBM [21, 22]. Liu and Zaccarin, pioneers in developing subsample algorithms, applied 4-to-1 subsampling to FSBM and significantly reduced the computational load. As shown by simulation results, the 4-to-1 subsample algorithm reduces the computational load significantly while keeping the quality similar to that with exhaustive search [21]. Here, we will present a generic subsample algorithm in which the subsample rate ranges from 4-to-1 to 1-to-1. The generic subsample algorithm uses Eq. (12) as a matching criterion, called the subsample sum of absolute difference (SSAD), where $SM_{8:m}$ is the subsample mask for the subsample rate 8-to-m as shown in Eq. (13):

$$SSAD_{8:m}(u,\ v) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} |\, SM_{8:m}(i,\ j)\cdot[S(i+u,\ j+v) - R(i,\ j)]\,|, \quad \text{for } -p \le u, v \le p-1, \tag{12}$$

$$SM_{8:m}(i, j) = BM_{8:m}(i \bmod 4, j \bmod 4). \tag{13}$$

The subsample mask $SM_{8:m}$ is generated from a basic mask as shown in Eq. (14):

$$BM_{8:m} = \begin{bmatrix} u(m-2) & u(m-5) & u(m-2) & u(m-6) \\ u(m-3) & u(m-7) & u(m-4) & u(m-8) \\ u(m-2) & u(m-5) & u(m-2) & u(m-6) \\ u(m-3) & u(m-7) & u(m-4) & u(m-8) \end{bmatrix}, \tag{14}$$

where $u(n)$, is a step function; that is,

$$u(n) = \begin{cases} 1, & \text{for } n \ge 0 \\ 0, & \text{for } n < 0 \end{cases}.$$

For example, consider the subsample rate 8-to-6. The subsample mask $SM_{8:6}$ can be expressed in Eq. (15) and is illustrated in Fig. 3:

$$SM_{8:6} = \begin{bmatrix} 1&1&1&1&1&1&1&1 \\ 1&0&1&0&1&0&1&0 \\ 1&1&1&1&1&1&1&1 \\ 1&0&1&0&1&0&1&0 \\ 1&1&1&1&1&1&1&1 \\ 1&0&1&0&1&0&1&0 \\ 1&1&1&1&1&1&1&1 \\ 1&0&1&0&1&0&1&0 \end{bmatrix}. \tag{15}$$

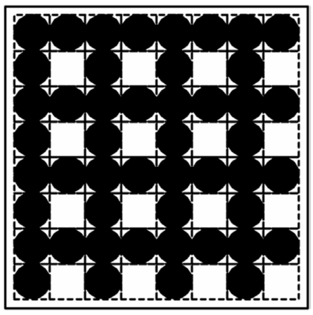


Fig. 3. The subsample mask of the subsample rate 8-to-6.

Given a subsample mask, the computational cost of the SSAD calculation can be lower than that of the SAD calculation. Since a reduction of computational cost implies reduced power consumption, the generic subsample algorithm allows the system power to scale with the changing subsample rate. The higher the subsample rate, the greater the number of inactive execution units (EUs). Accordingly, the power consumption of the system is proportional to the inverse of the subsample rate. Due to its flexibility in achieving an energy-quality tradeoff, the generic subsample algorithm is suitable for implementing power-aware architectures. However, the algorithm suffers from the aliasing problem in the high frequency band. The aliasing problem will degrade the MV quality and result in considerable quality degradation when the high-frequency band is messed up.

3.2 Content-Based Subsample Algorithm

As mentioned above, the generic subsample algorithm suffers from the aliasing problem due to the high subsample rate, leading to considerable quality degradation because the high frequency band is messed up. To alleviate this problem, we propose using the content-based subsample algorithm (CSA), which only subsamples the low-frequency band. The CSA procedure is shown in Fig. 4. We first use edge extraction to separate high-frequency pixels (or edge pixels) from a macro-block and then subsample the remaining pixels (or low-frequency pixels). The determination of edge pixels starts with gradient filtering. Three popular gradient filters [26] were also used here to execute the content-based algorithm; they are the high-pass gradient filter, the Sobel gradient filter, and the morphological gradient filter. Eqs. (16) to (18) show the calculations of the three gradient filters:

High-Pass Gradient Filter:

$$G_{hpf}(i, j) = |MF(HPF_{mask}, R)(i, j)|, \tag{16}$$

where $HPF_{mask} = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}.$

```
// frame: $t$
**Input current and reference frames, $W \times H$;**
for ($y = 0$; $y < W/N$; $y$++) {
    for ($x = 0$; $x < H/N$; $x$++) {
        **Perform gradient filtering;**
        **Calculate the edge threshold:**
            $threshold = m_1^t(x, y) \cdot \max\{G(i, j)\} + (1 - m_1^t(x, y)) \cdot \min\{G(i, j)\}$
        **Determine edge pixels and edge mask;**
        **Generate content-based subsample mask (GSM);**
        $edge\_cnt$ = total edges of CSM;
        // update threshold parameter for the next frame
        $m_1^{t+1}(x, y) = m_1^t(x, y) + K_p \cdot (csm\_cnt - trg\_cnt)$;
        if ($m_1^{t+1}(x, y) < 0$) {$m_1^{t+1}(x, y) = 0$};
        if ($m_1^{t+1}(x, y) > 1$) {$m_1^{t+1}(x, y) = 1$};
        // find $MV$
        $SSAD_{\min}(x, y) = \infty$;
            for ($u = -p$; $u < p$; $u$++) {
                for ($v = -p$; $v < p$; $v$++) {
                    $SSAD(u, v) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} | CSM(i, j) \cdot (S(i+u, j+v) - R(i, j)) |$;
                    if $SSAD_{\min}(x, y) > SSAD(u, v)$
                      {$SSAD_{\min}(x, y) = SSAD(u, v)$; $MV(x, y) = (u, v)$;}
                } // for loop index $v$
            } // for loop index $u$
    } // for loop index $x$
} // for loop index $y$
```

Fig. 4. The content-based subsample algorithm.

Sobel Gradient Filter:

$$G_{sobel}(i, j) = |MF(SX_{mask}, R)(i, j)| + |MF(SY_{mask}, R)(i, j)|, \tag{17}$$

where $SX_{mask} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$ and $SY_{mask} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$.

Morphological Gradient Filter:

$$G_{morphological} = (R \oplus B) - (R \ominus B), \tag{18}$$

where $B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$, and the operations "$\oplus$" and "$\ominus$" denote morphological dilation and erosion.

In Eqs. (16) and (18), the $MF(\cdot)$ function is the mask filter operation as shown in Eq. (19):

$$MF(M,R)(i,\ j)=\sum_{p=-1}^{1}\sum_{q=-1}^{1}M(p+1,\ q+1)\cdot R(i+p,\ j+q), \tag{19}$$

where M is a 3-by-3 mask and $R(i, j)$ is the luminance value at (i, j).

After obtaining the gradients, G, instead of using a constant threshold, we use a floating threshold to determine the edge pixels of the CMB. The floating threshold makes edge extraction more robust when video content varies. Eq. (21) shows the calculation of the floating threshold:

$$threshold = m_1^t(x, y)\cdot\max\{G(i, j)\} + (1 - m_1^t(x, y))\cdot\min\{G(i, j)\},\ \text{for } 0 \le m_1^t \le 1, \tag{20}$$

where $m_1^t(x, y)$ is the threshold parameter of macro-block (x, y) in the *t-th* frame.

Following the threshold setting step, the algorithm uses the threshold value to pick the edge pixels and produce the edge mask as shown in Eq. (21):

$$EdgeMask(i,\ j)=\begin{cases}1, & \text{for } G(i,\ j)\ge threshold\\ 0, & \text{otherwise}\end{cases}. \tag{21}$$

Finally, the contend-based subsample mask (CSM) is generated by merging the edge mask and the subsample mask, as shown in Eq. (22). In Eq. (22), the operator $\vee$ means logic a OR operation. According to the calculation of the CSM, the subsample rate in the CSA (CSR), denoted as R_s, is N^2-to-*csm_cnt*, where *csm_cnt* is the number of 1's in CSM and N^2 is the macro-block size. Fig. 5 shows an example of a CSM where the subsample rate is 64-to-27:

$$CSM(i, j) = SM_{8:m}(i, j) \vee EdgeMask(i, j),\ 0 \le i, j \le N - 1. \tag{22}$$

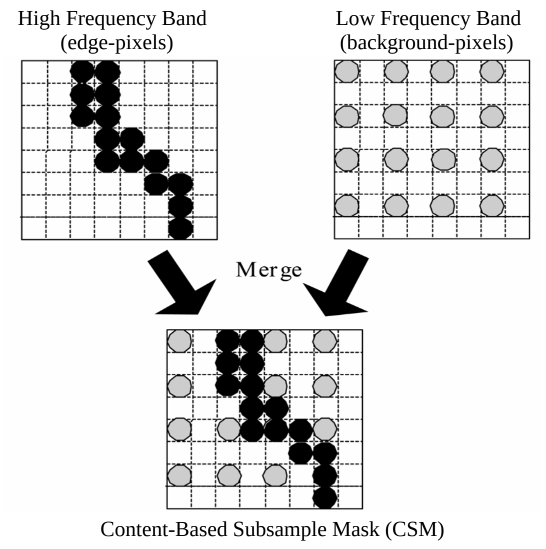


Fig. 5. The components of a content-based subsample mask (CSM).

Once the CSM is generated, the algorithm can then determine the motion vection (MV) with the content subsample sum of the absolute difference (CSSAD) criterion. The CSSAD criterion is similar to SSAD mentioned in section 3.1 and shown in Eq. (23):

$$CSSAD_{8:m}(u,\ v) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} | CSM_{8:m}(i,\ j) \cdot [S(i+u,\ j+v) - R(i,\ j)] |,$$
$$\text{for } -p \le u,\ v \le p-1. \tag{23}$$

The results of simulation show that the CSA can significantly reduce the computation complexity with little quality degradation. However, there will exist a non-stationary problem with CSA when a power-aware architecture is implemented if the designer uses constant threshold parameters m_1^t and statically sets the floating threshold for a given power mode. Since different video clips with the same threshold parameters will have different subsample rates, setting the threshold value without considering the content variation of the video clip will make the subsample rate non-stationary; that is, power consumption will not converge within a narrow range for a given power mode. The divergence of power consumption can result in a poor power-awareness. To solve this non-stationary problem, we use an adaptive control mechanism to adaptively adjust the threshold parameters so that the subsample rate can be stationary. The adaptive control mechanism used here is a run-time process that adjusts the threshold parameters fittingly according to the difference between the current subsample rate and the desired subsample rate (or target subsample rate).

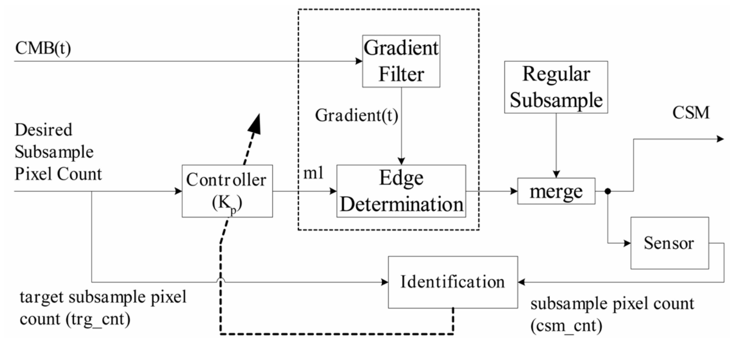


Fig. 6. A block diagram of the edge-extraction unit with an adaptive control mechanism.

Fig. 6 shows a block diagram of the adaptive control mechanism. Given the battery status, the host processor sets the power mode and the target subsample rate as well. The target subsample rate is N^2-to-*trg_cnt*, where *trg_cnt* is the target number of 1's in the CSM. Then, the controller recursively updates the threshold parameter, $m_1^{t+1}(x, y)$, based on the current $m_1^t(x, y)$ and the difference of *csm_cnt* and *trg_cnt*, as shown in Eq. (24):

$$\begin{aligned} &m_1^{t+1}(x, y) = m_1^t(x, y) + K_p \cdot (csm\_cnt - trg\_cnt); \\ &\text{if } (m_1^{t+1}(x, y) < 0)\ \{m_1^{t+1}(x, y) = 0\}; \\ &\text{if } (m_1^{t+1}(x, y) > 1)\ \{m_1^{t+1}(x, y) = 1\}; \end{aligned} \tag{24}$$

where $m_1^{t+1}(x, y)$ is the threshold parameter of macro-block (x, y) in the $(t + 1)$-*th* frame and K_p is the control parameter. The control parameter K_p will affect the settling time and steady-state error of the subsample rate.

3.3 Simulation Results

Figs. 7 and 8 show the simulation results for four 352-by-288 MPEG clips with the parameters $N = 16$ and $p = 32$. The control parameter K_p was set as 0.3. The target subsample rates were set to (4:1), (8:3), (2:1), (8:5), (4:3), (8:7), and (1:1); that is, the target subsample pixel counts were 64, 96, 128, 160, 192, 224, and 256, respectively. Note that the target subsample pixel counts were proportional to the power consumption. Thus, the figures can also be read as charts of power versus PSNR. The dashed lines indicate the results obtained using the generic subsample algorithm, and the solid lines indicate the results obtained using the content-based subsample algorithm with the three gradient filters. As shown by the results, the quality degradation due to the content-based algorithm was less than that due to the generic subsample algorithm, and the type of gradient filter did not significantly affect the performance of the proposed algorithm. In addition, the adaptive control mechanism kept the subsample rate quite stationary. Tables 1 to 3 show the CSR errors with four 40-frame clips. From the results shown in tables, the

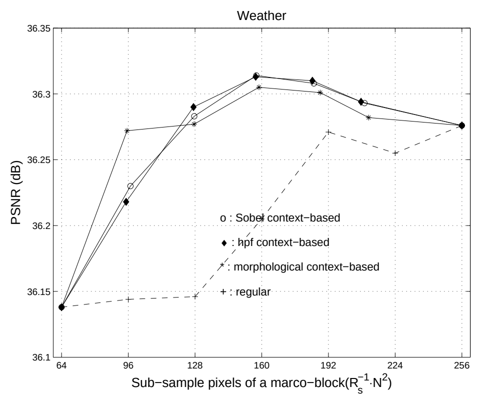


Fig. 7. The quality degradation of the weather clip.

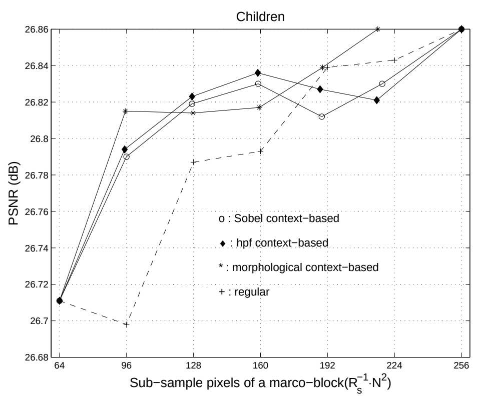


Fig. 8. The quality degradation of the children clip.

Table 1. The CSR error of the content-based subsample algorithm (where the control parameter K_p = 0.3).

Video Clip	Weather		News		Table-Tennis		Children	
Target CSR	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error
96	95.416	0.61%	95.366	0.66%	95.398	0.63%	95.490	0.53%
128	127.521	0.37%	127.720	0.22%	127.467	0.42%	127.754	0.19%
160	158.678	0.83%	159.795	0.13%	159.533	0.29%	159.430	0.36%
192	188.037	2.06%	191.313	0.36%	191.429	0.30%	189.632	1.23%
224	211.274	5.68%	221.224	1.24%	222.893	0.49%	216.001	3.57%

The edge-extraction unit uses the high-pass gradient filter.

Table 2. The CSR error of the content-based subsample algorithm (where the control parameter $K_p = 0.3$).

Video Clip	Weather		News		Table-Tennis		Children	
Target CSR	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error
96	94.985	1.06%	94.728	1.33%	95.361	0.67%	95.114	0.92%
128	127.264	0.58%	127.688	0.24%	127.445	0.43%	127.304	0.54%
160	157.104	1.81%	159.766	0.15%	159.415	0.37%	159.702	0.381%
192	184.295	4.01%	191.183	0.43%	191.249	0.39%	188.451	1.85%
224	207.612	7.32%	220.949	1.36%	222.685	0.59%	215.536	3.78%

The edge-extraction unit uses the Sobel gradient filter.

Table 3. The CSR error of the content-based subsample algorithm (where the control parameter $K_p = 0.3$).

Video Clip	Weather		News		Table-Tennis		Children	
Target CSR	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error	Average CSR	CSR Error
96	97.072	1.12%	95.547	0.47%	96.040	0.04%	95.959	0.04%
128	127.626	0.29%	127.718	0.22%	127.120	0.69%	127.267	0.57%
160	157.401	1.62%	159.659	0.21%	158.986	0.63%	158.885	0.70%
192	185.013	3.64%	191.477	0.27%	190.765	0.64%	189.279	1.42%
224	209.300	6.56%	222.434	0.70%	222.405	0.71%	218.118	2.63%

The edge-extraction unit uses the morphological gradient filter.

average CSR error was as low as 1.12%, and the CSR error variance was as low as 0.00024. Because the subsample rate could be kept nearly stationary with given target subsample rate and power mode, we conclude that the power-awareness of the proposed algorithm is very good, and that the CSA can be applied in a power-aware architecture. The results also show that the selection of K_p was proper for controlling the threshold parameters. The following section will further address on the selection of the control parameter.

3.4 Selection of the Control Parameter

As mentioned in sections 3.3 and 3.4, we use an adaptive control mechanism to obtain a stationary subsample rate while keeping the quality acceptable. However, if the control parameter is not properly selected, the settling time will be too long to achieve real-time switching, and the CSR error will be so large as to make the setting of the power consumption mode inaccurate and the power-awareness worse. The control parameter, K_p, in Fig. 6 is the major factor affecting the settling time and the CSR error. After four 30-frame video clips were simulated with 1:1 of the initial subsample rate and 8:5 of the target subsample rate, the effects of the K_p selections were as shown in Figs. 9 and 10. Obviously, the higher the value of K_p, the shorter the settling time and the worse the stability of the CSR. As shown by the results, the suitable range of K_p was from 0.1 to 0.5.

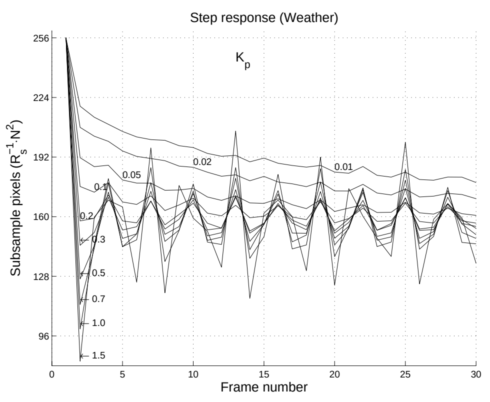


Fig. 9. The step response for varying K_p in the case of the weather clip.

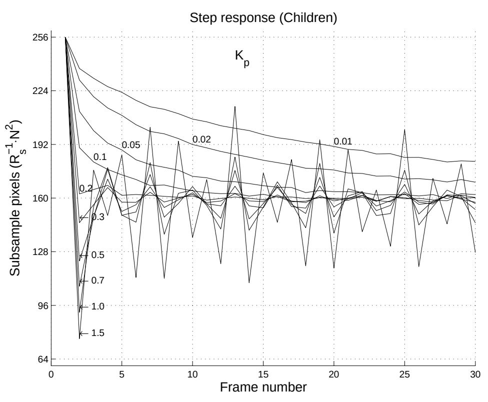


Fig. 10. The step response for varying K_p in the case of the children clip.

4. THE POWER-AWARE ARCHITECTURE

4.1 System Architecture

According to the content-based subsample algorithm, we present a semi-systolic architecture in Fig. 11, which is based on existing architectures, such as that in [5]. The architecture contains an edge-extraction unit (EXU), an array of processing elements (PEs), a parallel adder tree (PAT), a shift register array (SRA), and a motion-vector selector (MVS). Given the power consumption mode, the EXU extracts high-frequency (or edge) pixels from the current macro-block (CMB) and generates 0-1 content-based subsample masks (CSM) for the PE array to disable or enable processing elements (PEs). The structure of the PE array, as shown in Fig. 12, is used to accumulate absolute pixel differences column by column while the parallel adder tree sums up all the results to generate the value of the CSSAD. The MVS then performs a compare-and-select operation to select the best motion vector.

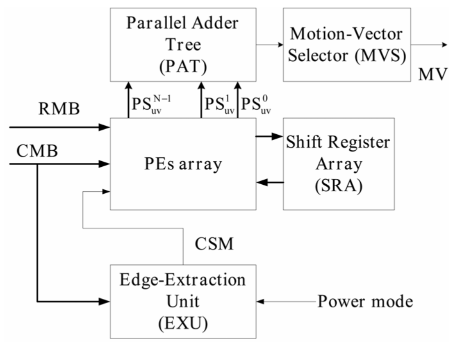

Parallel Adder Tree (PAT)
Motion-Vector Selector (MVS)
MV
PS_{uv}^{N-1}
PS_{uv}^{1}
PS_{uv}^{0}
RMB
CMB
PEs array
Shift Register Array (SRA)
CSM
Edge-Extraction Unit (EXU)
Power mode

Fig. 11. The system block diagram.

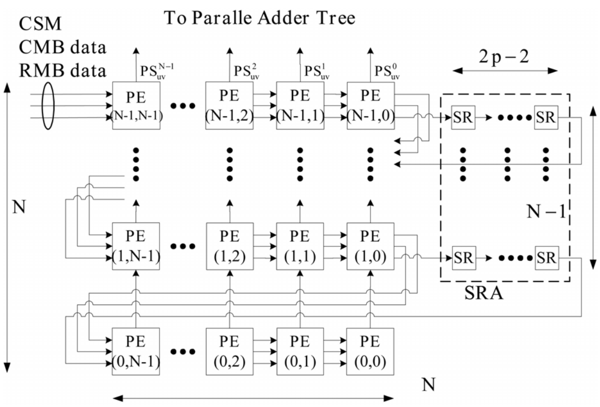


Fig. 12. The architecture of the PE array and shift register array.

Based on the semi-systolic architecture with the content-based subsample algorithm, the architecture dynamically disables some processing elements to reduce the power consumption since we assume the major power consumption is mainly determined by the switch activity of the system [15]. After edge extraction is performed first, a threshold is set as the criterion for determining whether or not to enable/disable processing elements, thus dynamically changing the switch activities of the system to reduce the power consumption. Fig. 13 shows the PE structure and indicates how the CSM disables/enables processing elements. The CSM disables the PE by using the block element (BE), implemented by means of AND gates. The BEs can nullify the input signals of data path, which consists of the absolute difference unit ($|a - b|$) and the Adder unit. When a PE is disabled during a MV searching iteration, the circuits in the PE remain still until the next iteration starts; thus, the consumption of transient power can be reduced.

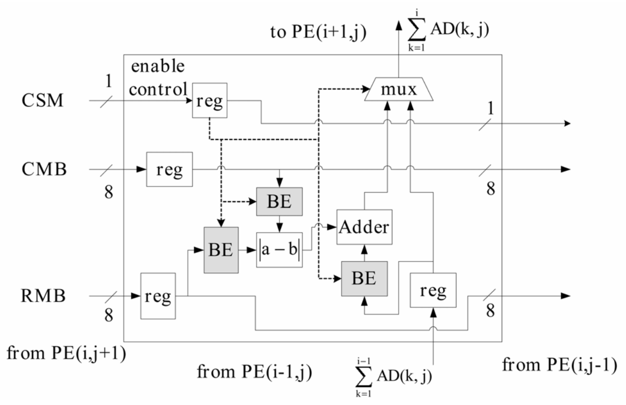


Fig. 13. The structure of a PE.

The edge-extraction unit contains two blocks: a gradient filter and a CSM generator. The gradient filter is implemented based on one of Eqs. (16) to (18). The proposed architecture only requires that a single gradient filter be embedded. However, we will show all of their implementations for the purpose of estimating the overheads and making a comparison. Figs. 14 to 16 illustrate the implementations of high-pass, Sobel, and morphological gradient filters respectively. Multiplexers are used to prevent boundary errors from occurring with border pixels of the CMB. The black dot in each multiplexer indicates the switching path used when processing a border pixel. Fig. 17 shows the structure of the CSM generator. The CSM generator first determines the threshold according to the gradient values and the power mode, and then generates the CSM by OR-merging the regular subsample pattern and the edge pattern, as shown in Eqs. (21) and (22).

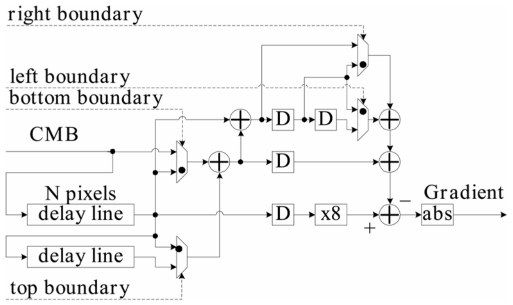


Fig. 14. The architecture of the high-pass gradient filter.

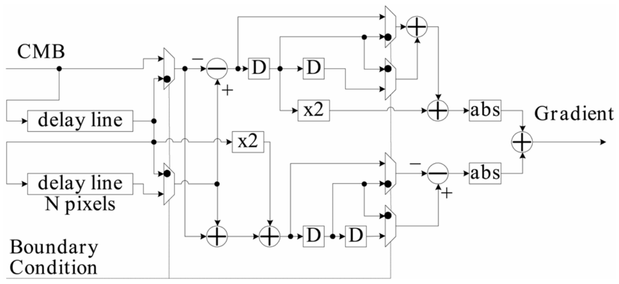


Fig. 15. The architecture of the Sobel gradient filter.

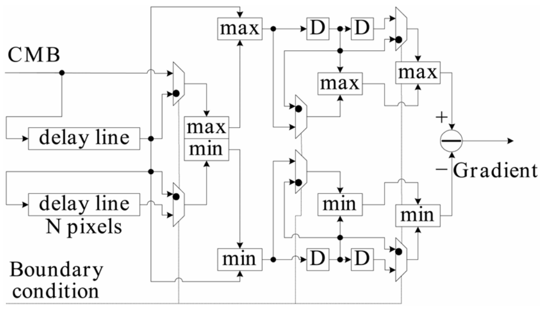


Fig. 16. The architecture of the morphological gradient filter.

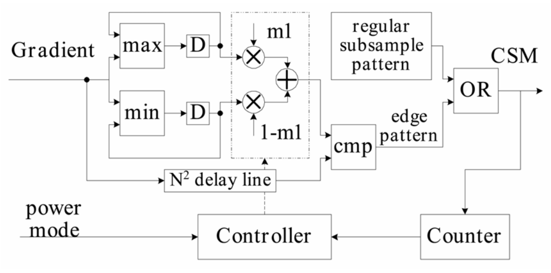
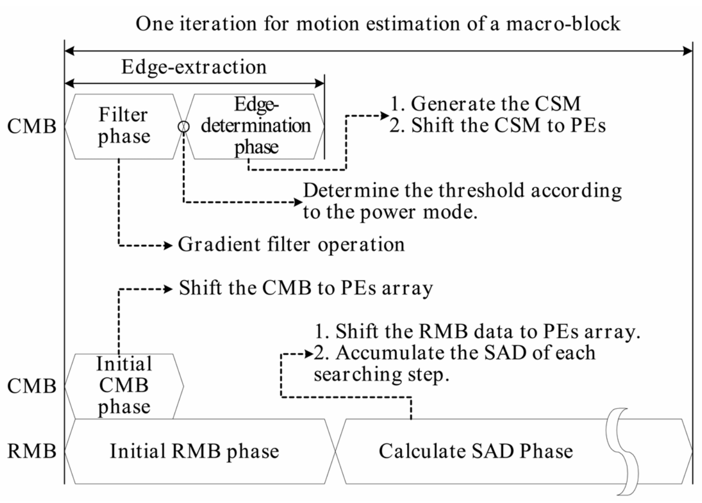


Fig. 17. The architecture of edge-determination and the CSM generator.

4.2 Execution of Power-Aware ME

Power-aware motion estimation is performed in five phases: the initial CMB phase, initial RMB phase, SAD calculation phase, filtering phase, and edge-determination phase. The initial CMB phase involves loading the CMB data into a PE array, while the initial RMB phase involves filling up the PE array with RMB data to start the SAD calculation. As shown in Fig. 18, the initial CMB phase and initial RMB phase are executed in parallel with edge extraction; thus, the timing overhead of edge extraction is hidden by the initial phases. For $p > 8$, the timing overhead of edge extraction is zero.


One iteration for motion estimation of a macro-block
Edge-extraction
CMB
Filter phase
Edge-determination phase
1. Generate the CSM
2. Shift the CSM to PEs
Determine the threshold according to the power mode.
Gradient filter operation
Shift the CMB to PEs array
1. Shift the RMB data to PEs array.
2. Accumulate the SAD of each searching step.
CMB
Initial CMB phase
RMB
Initial RMB phase
Calculate SAD Phase
Fig. 18. The execution phases of the power-aware architecture.

4.3 Experimental Results

Table 4 shows the synthesis results obtained using the TSMC 1P4M 0.35*um* cell library, where the symbol R_s denotes the content-based subsample rate and ε_{gp} is the gate power coefficient defined in Eq. (9). Compared with the general semi-systolic architecture [5], the edge extraction unit (EXU) of the proposed architecture has the major

Table 4. Power analysis of the power-aware architecture.

EU_i	PE array		SRA	PAT + MVS	EXU
	AD + Adder	Other			
Gate Count G^i	117,760	58,708	44,640	1,800	17,121
$r^i(0\leftrightarrow1)$	$4p^2R_s^{-1}=4096R_s^{-1}$	$4p^2=4096$	$4p^2=4096$	$4p^2=4096$	$N^2=256$
$P^i_{consumption}$	$4.8e8 \cdot R_s^{-1}$	$2.4e8$	$1.8e8$	$7.37e6$	$4.38e6$
$P^{all}_{consumption}(\varepsilon_{gp})$	$4.8e8 \cdot R_s^{-1} + 4.3e8$				

N = 16 and p = 32. Cell library: TSMC 0.35*um* process.

overhead for the power-aware function. As mentioned above, we used one of the three gradient filters here to implement the EXU. As for the synthesis results, the gate counts of the three gradient filters were 595.33, 793.77, and 727.63, respectively. The variance of these values is very small compared to the overall gate count of the EXU. For instance, the gate count of EXU with the high-pass filter was equal to 14745. This number is much larger than the variance. This means that the selection of a gradient filter does not affect the overhead estimation very much. Therefore, we used the high pass filter to estimate the performance overhead caused by the EXU. From Table 4, one can see that the area overhead of EXU was 7.68%, while the worst-case power overhead was only 0.8% when the subsample rate was 4-to-1 for motion estimation with N = 16 and p = 32.

Fig. 19 shows the results for the video clip "table-tennis" under switching of the power consumption mode. The target subsample pixel count was reduced by 48 every 40 frames, and the control parameter K_p was set to 0.3. The results show that the adaptive control mechanism could enable the power consumption to reach the target level within 10 frames. According to the battery properties described in section 2, the curve shows that our power-aware architecture can extend the battery lifetime by gradually degrading the quality. A, B, C, and D correspond to the switching points in Fig. 2 (b), respectively.

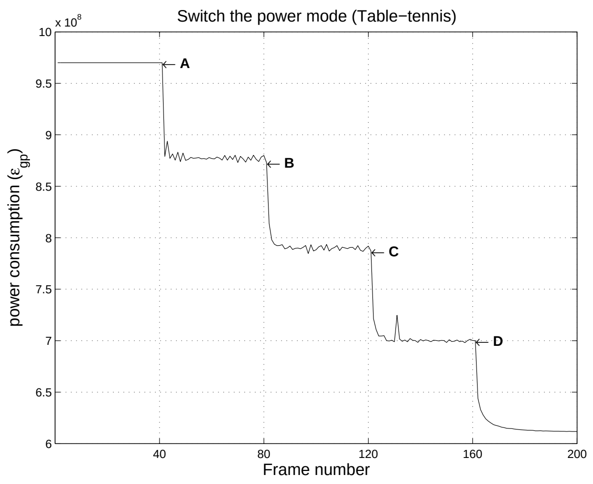


Fig. 19. An application involving switching of the power mode in the case of the video clip "table-tennis."

5. CONCLUSIONS

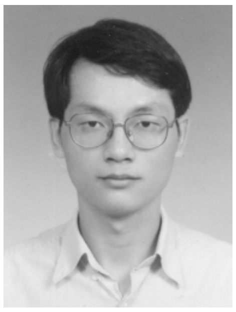
Motivated by the battery properties and the power-aware paradigm, this paper has presented an architecture-level power-aware technique based on a novel content-based subsample algorithm. When the battery capacity is full, the proposed ME architecture turns on all the PEs to provide the best compression quality. In contrast, when the battery capacity is short for full operation, instead of exhibiting an all-or-none behavior, the proposed architecture shifts to a lower power consumption mode by disabling some PEs in order to extend the battery lifetime with little quality degradation. Switching of power consumption mode can be smoothly accomplished; thus, the proposed architecture makes it possible to switch the power consumption mode with acceptable quality degradation. Although edge extraction plays a crucial role to dynamically adjusting the power consumption mode, it does not introduce much power dissipation and the timing overhead can be neglected. As shown by the simulation results, the proposed algorithm successfully improves the compression quality of the generic subsample algorithm and switches the power consumption mode by adaptively adjusting the threshold parameters.

REFERENCES

1. P. Raghavan and C. Chakrabarti, "Battery-friendly design of signal processing algorithms," in *Proceedings of the IEEE Workshop on Signal Processing Systems*, 2003, pp. 304-309.
2. M. J. Chen, L. G. Chen, and T. D. Chiueh, "One-dimensional full search motion estimation algorithm for video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 4, 1994, pp. 504-509.
3. B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 3, 1993, pp. 148-157.
4. H. W. Cheng and L. R. Dung, "EFBLA: a two-phase matching algorithm for fast motion estimation," *Advances in Multimedia Information Processing – PCM*, LNCS 2532, 2002, pp. 112-119.
5. D. Linden, *Handbook of Batteries*, 2nd ed., McGraw-Hill, Inc., New York, 1995.
6. R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, New York, 1993.
7. C. H. Hsieh and T. P. Lin, "VLSI architecture for block-matching motion estimation algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 2, 1992, pp. 169-175.
8. K. Sauer and B. Schwartz, "Efficient block motion estimation using integral projections," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 6, 1996, pp. 513-518.
9. J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Transactions on Communications*, Vol. COM-29, 1981, pp. 1799-1808.
10. J. N. Kim, S. C. Byun, Y. H. Kim, and B. H. Ahn, "Fast full search motion estimation algorithm using early detection of impossible candidate vectors," *IEEE Transactions*

on Signal Processing, Vol. 50, 2002, pp. 2355-2365.
11. T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishigura, "Motion compensated interframe coding for video conferencing," in *Proceedings of the IEEE National Telecommunication Conference*, Vol. 4, 1981, pp. G5.3.1-G5.3.5.
12. Y. K. Lai and L. G. Chen, "A data-interlacing architecture with two-dimensional data-reuse for full-search block-matching algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, 1998, pp. 124-127.
13. S. Lee and S. I. Chae, "Motion estimation algorithm using low-resolution quantization," *IEE Electronic Letters*, Vol. 32, 1996, pp, 647-648.
14. J. H. Luo, C. N. Wang, and T. Chiang, "A novel all-binary motion estimation (ABME) with optimized hardware architectures," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, 2002, pp. 700-712.
15. *Mobile Pentium III Processor in BGA2 and Micro-PGA2 Packages Datasheet*, Intel Corporation, pp. 55.
16. P. Kuhn, *Algorithms*, *Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation*, Kluwer Academic Publishers, U.S.A., 1999.
17. R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 4, 1994, pp. 438-442.
18. C. K. Cheung and L. M. Po, "Normalized partial distortion search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, 2000, pp. 417-422.
19. S. Unsal and I. Koren, "System-level power-aware design techniques in real-time systems," in *Proceedings of the IEEE Special Issue on Real-Time Systems*, Vol. 91, 2003, pp. 1055-1069.
20. W. Li and E. Salari, "Succesive elimination algorithm for motion estimation," *IEEE Transactions on Image Processing*, Vol. 4, 1995, pp. 105-107.
21. J. Y. Tham, S. Ranganath, M. Ranganath, and A. A. Kassim, "A novel unrestricted center-biased diamond search algorithm for block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, 1998, pp. 369-377.
22. J. C. Tuan, T. S. Chang, and C. W. Jen, "On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, pp. 61-72.
23. V. L. Do and K. Y. Yun, "A low-power VLSI architecture for full-search block-matching motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 8, 1998, pp. 393-398.
24. K. M. Yang, M. T. Sun, and L. Wu, "A family of VLSI designs for the motion compensation block-matching algorithm," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 36, 1989, pp. 1317-1325.
25. W. Zhang, R. Zhou, and T. Kondo, "Low-power motion-estimation architecture based on a novel early-jump-out technique," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, Vol. 5, 2001, pp. 187-190.
26. C. Zhu, X. Lin, and L. P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 12, 2002, pp. 349-355.

27. M. Bhardwaj, R. Min, and A. P. Chandrakasan, “Quantifying and enhancing power awareness of VLSI systems,” *IEEE Transactions on Very Large Scale Integration Systems*, Vol. 9, 2001, pp. 757-772.
28. C. L. Su and C. W. Jen, “Motion estimation using MSD-first processing,” *IEE Proceedings of Circuits*, *Devices and Systems*, Vol. 150, 2003, pp. 124-133.



Hsien-Wen Cheng (鄭顯文) was born in 1968. He received the B.S. degree in Control Engineering from National Chiao Tung University, Hsinchu, Taiwan, R.O.C. in 1992. He joined the AVerMedia Technologies Inc. from 1994 to 2001. He is currently working toward the Ph.D. degree in the Electrical and Control Engineer, National Chiao Tung University. His research interests are video/image compression, motion estimation, VLSI architecture, and digital signal processing.

Lan-Rong Dung (董蘭榮) was born in 1966. He received a B.S.E.E. and the Best Student Award from Feng Chia University, Taiwan, in 1988, an M.S. in Electronics Engineering from National Chiao Tung University, Taiwan, in 1990, and Ph.D. in Electrical and Computer Engineering from Georgia Institute of Technology, in 1997. From 1997 to 1999 he was with Rockwell Science Center, Thousand Oaks, CA, as a Member of the Technical Staff. He joined the faculty of National Chiao Tung University, Taiwan in 1999 where he is currently an Associate Professor in the Department of Electrical and Control Engineering. He received the VHDL International Outstanding Dissertation Award celebrating in Washington DC in October, 1997. His current research interests include VLSI design, digital signal processing, hardware-software codesign, and System-on-Chip architecture. He is a member of Computer and Signal Processing societies of the IEEE.

A Content-Based Methodology for Power-Aware Motion Estimation Architecture

Hsien-Wen Cheng and Lan-Rong Dung, *Member, IEEE*

***Abstract*—This paper presents a novel power-aware motion estimation algorithm, called adaptive content-based subsample algorithm (ACSA), for battery-powered multimedia devices. While the battery status changes, the architecture adaptively performs graceful tradeoffs between power consumption and compression quality. As the available energy decreases, the algorithm raises the subsample rate for maximizing battery lifetime. Differing from the existing subsample algorithms, the content-based algorithm first extracts edge pixels from a macro-block and then subsamples the remaining low-frequency part. In this way, we can alleviate the aliasing problem and thus keep the quality degradation low as the subsample rate increases. As shown in experimental results, the architecture can dynamically operate at different power consumption modes with little quality degradation according to the remaining capacity of battery pack while the power overhead of edge extraction is under 0.8%.**

***Index Terms*—Content-based image processing, motion estimation (ME), power-aware system, subsample algorithm, very large-scale integration (VLSI) architecture, video compression, VLSI image processing.**

I. Introduction

Motion estimation (ME) has been notably recognized as the most critical part in many video compression applications, such as MPEG standards and H.26x [1], which leads to dominant computational and, hence, power requirements. With increasing demand of portable multimedia devices, recently, a power-aware ME that can be flexible in both power consumption and compression quality is highly required [2]. Fig. 1 illustrates a typical block diagram of the portable multimedia system powered by battery. In practice, a battery has the two most important nonideal properties, which are the rate capacity effect and recovery effect [3]. Fig. 2, as an example, illustrates that the system can extend the battery lifetime by gradually stepping down the power dissipation. Normally, system designers can choose points A, B, C, and D according to the discharging profile provided by battery manufacturer. When the battery monitor unit detects the voltage drop of battery, the host processor will change the power mode accordingly. By changing the operation mode of all power-aware components simultaneously, the architecture adapts the power dissipation to battery status and, hence, raises battery performance.

Manuscript received February 24, 2004; revised September 13, 2004 and February 22, 2005. This work was supported in part by Taiwan MOE Program for Promoting Academic Excellent of Universities under Grant 91-E-FA06-4-4 and the National Science Council, Taiwan, R.O.C., under Grant NSC 93-2220-E-009-023. This paper was recommended by Associate Editor S.-F. Chang.

The authors are with the Department of Electrical and Control Engineering, National Chiao Tung University, Hsinchu 300, Taiwan, R.O.C. (e-mail: jswcheng.ece89g@nctu.edu.tw; lennon@cn.nctu.edu.tw).

Digital Object Identifier 10.1109/TCSII.2005.850771

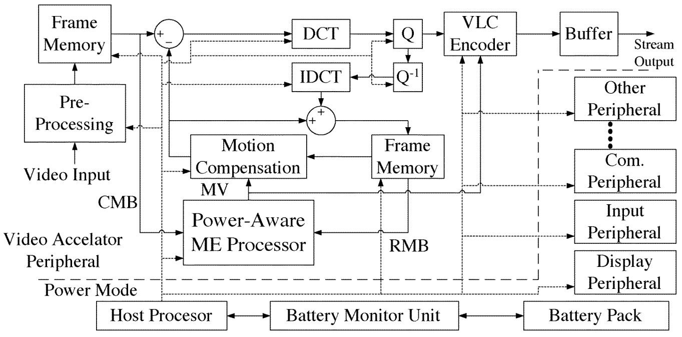


Fig. 1. System block diagram of a portable, battery-powered multimedia device.

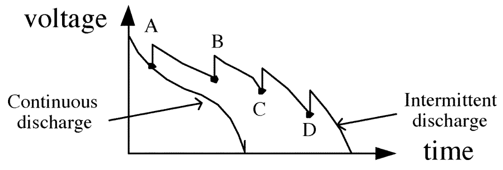


Fig. 2. Diagram representing the nonlinear discharging properties of battery.

This paper, therefore, intends to presents a power-aware ME architecture driven by an adaptive content-based subsample algorithm to lengthen the battery lifetime.

Many published papers have presented efficient algorithms for very large-scale integration (VLSI) implementation of motion estimation on high performance or low-power design. Yet most of them cannot dynamically adapt the compression quality to different power consumption modes. Among these proposed algorithms, the full-search block-matching (FSBM) algorithm with the sum of absolute difference (SAD) criterion is the most popular approach for ME because of its considerably good quality. It is particularly attractive to those who require extremely high quality. However, the huge number of *comparison/difference* operations results in a high computation load and power consumption [4], [5]. To reduce the computational complexity of FSBM, researchers have proposed various fast algorithms. They either reduce search steps [6]–[8] or simplify calculations of error criterion [9], [10]. These fast-search algorithms have successfully improved the block matching speed and, thus, led to a low-power implementation. However, a low-power implementation is not necessarily a power-aware system, in which the implementation should adaptively modify its behavior to balance the performance between quality and battery life [11]. The requirement for ME algorithms to be suitable for power-aware design is a high degree of scalability in performance tradeoffs. Unfortunately, the fast algorithms mentioned above do not meet this requirement.

The subsample algorithms present in [12] and [13] are very suitable for power-aware ME architecture because of their highly scalable characteristic. As the subsample rate increasing, the unsampled processing elements in a semisystolic array will be disabled to save the switch activities, that is, the power consumption of the architecture will be decreased correspondingly. However, applying subsample algorithms for power-aware architecture may suffer from the aliasing problem in the high-frequency band. The aliasing problem degrades the compression quality rapidly as the subsample rate increases. To alleviate the problem, we extend traditional subsample algorithms to an adaptive content-based subsample algorithm (ACSA). In the ACSA, we first use edge extraction techniques to separate the high-frequency band from a macro-block and then subsample the low-frequency band only. By merging the edge pixels and subsample nonedge pixels, the algorithm generates a turn-off mask for the architecture to disable the switching activities of processing elements (PEs) in a semisystolic array. This content-based algorithm keeps the quality degradation low as the subsample rate increases. Because the number of high-frequency pixels varies with different video clips, we introduce an adaptive control mechanism to set the threshold value for edge determination and make the number of masked pixels stationary for a given power mode. The ACSA can be implemented in most existing ME architectures by turning off PEs according to the subsample mask. In this paper, we present a semisystolic architecture with gated PEs. The simulation results show that the ACSA can dynamically alter the subsample rate as the power consumption mode changes and the architecture can work at different power consumption modes with acceptable and smooth quality degradation while the power overhead of edge extraction is under 0.8%.

II. Subsample Algorithms

A. Generic Subsample Algorithm

Here, we present a generic subsample algorithm (GSA) in which the subsample rate ranges from 4:1 to 1:1. The GSA uses

$$\mathrm{SSAD}_{8:m}(u,v) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} \left|SM_{8:m}(i,j)\,(S(i+u,j+v)-R(i,j))\right| \tag{1}$$

as a matching criterion, called the subsample sum of absolute difference (SSAD), where $SM_{8:m}$ is the subsample mask for the subsample rate 8 to m as shown in

$$SM_{8:m}(i,j) = BM_{8:m}(i \bmod 4, j \bmod 4). \tag{2}$$

where the macro-block size is N-by-N, $R(i,j)$ is the luminance value at (i,j) of the current macro-block, and $S(i+u,j+v)$ is the luminance value at (i,j) of the reference macro-block which offsets (u,v) from the current macro-block in the searching area $2p \times 2p$.The subsample mask $SM_{8:m}$ is generated from the basic mask

$$BM_{8:m} = \begin{bmatrix} u(m-2) & u(m-5) & u(m-2) & u(m-6) \\ u(m-3) & u(m-7) & u(m-4) & u(m-8) \\ u(m-2) & u(m-5) & u(m-2) & u(m-6) \\ u(m-3) & u(m-7) & u(m-4) & u(m-8) \end{bmatrix}$$

```
//frame:$t$
**Input current and reference frames,** $W \times H$ ;
for($y = 0$; $y < W/N$ ; $y{+}{+}$ ){
  for($x = 0$; $x < H/N$ ; $x{+}{+}$ ){
    **Perform gradient filtering**;
    **Calculate the edge threshold :**
      $E_{th} = m_1^t(x,y) \cdot \max\{G(i,j)\} + (1 - m_1^t(x,y)) \cdot \min\{G(i,j)\}$
    **Determine edge pixels and edge mask**;
    **Generate adaptive content - based subsample mask** ($ACSM$);
    $acsm\_cnt = total\ edges\ of\ ACSM$;
    //update threshold parameter for the next frame
    $m_1^{t+1}(x,y) = m_1^t(x,y) + K_p \cdot (acsm\_cnt - trg\_cnt)$;
    if ($m_1^{t+1}(x,y) < 0$) {$m_1^{t+1}(x,y) = 0$};
    if ($m_1^{t+1}(x,y) > 1$) {$m_1^{t+1}(x,y) = 1$};
    //find MV
    $ACSSAD_{\min}(x,y) = \infty$;
    for($u = -p$; $u < p$; $u{+}{+}$ ){
      for($v = -p$; $v < p$; $v{+}{+}$ ){
        $ACSSAD(u,v) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} \left|ACSM(i,j) \cdot (S(i+u,j+v) - R(i,j))\right|$;
        if $ACSSAD_{\min}(x,y) > ACSSAD(u,v)$
          {$ACSSAD_{\min}(x,y) = ACSSAD(u,v)$; $MV(x,y) = (u,v)$; }
    }}}}//for loop index $v,u,x,y$
```

Fig. 3. Procedure of the ACSA.

where

$$u(n) = 1, \qquad \text{for } n \geq 0 \text{ and } 0 \text{ for } n < 0. \tag{3}$$

Due to its flexibility in energy-quality tradeoffs, the GSA is suitable for the implementation of power-aware architectures. The power consumption of the architecture is proportional to the inverse of the subsample rate. However, the algorithm suffers from the aliasing problem which will degrade the ME quality and results in a considerable degradation of quality when the high-frequency band is messed up.

B. Adaptive Content-Based Subsample Algorithm

As mentioned above, the GSA has an aliasing problem for a high subsample rate and leads to considerable quality degradation because the high-frequency band is interfered with. To alleviate the problem, we propose an ACSA that only subsamples the low-frequency band. The procedure of the ACSA is described in Fig. 3. We first use edge extraction to separate high-frequency pixels (or edge pixels) from a macro-block and then subsample the remaining pixels (or low-frequency pixels). The determination of edge pixels starts from gradient filtering. In this paper, we use three popular gradient filters to exercise the content-based algorithm; they are the high-pass gradient filter, the Sobel gradient filter, and the morphological gradient filter. After obtaining the gradients denoted as G, we use a floating threshold E_{th} to determine the edge pixels of the current macro-block. The floating threshold makes the edge extraction more robust with video content varying than the constant threshold does. The following equation describes the calculation of E_{th} for each macro-block (x,y) in the tth frame:

$$E_{\mathrm{th}} = m^t \cdot \max\{G(i,j)\} + (1-m^t) \cdot \min\{G(i,j)\}, \quad \text{for } 0 \leq m^t \leq 1 \text{ and } 0 \leq i,\ j < N. \tag{4}$$

Following the threshold setting step, the algorithm uses the threshold value to pick the edge pixels and produces the edge mask as shown as

$$\text{EdgeMask}(i,j) = \begin{cases} 1, & \text{for } G(i,j) \geq E_{\text{th}} \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

Finally, the adaptive content-based subsample mask (ACSM) is generated by merging the edge mask and the subsample mask, as shown in

$$\text{ACSM}(i,j) = SM_{4:1}(i,j) \vee \text{EdgeMask}(i,j), \quad 0 \leq i,\ j \leq N-1. \quad (6)$$

The operator $\vee$ means logic OR operation. According to the calculation of ACSM, the subsample rate of the ACSA R_{ACSA} is equal to N^2 to $acsm\_cnt$, where $acsm\_cnt$ is the number of 1's in ACSM. Once the ACSM is generated, the algorithm can then determine the motion vection (MV) with the adaptive content-based subsample sum of absolute difference (ACSSAD) criterion. The following equation shows the ACSSAD criterion:

$$\text{ACSSAD}(u,v) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} \left|\text{ACSM}(i,j)\,(S(i+u,j+v) - R(i,j))\right| \quad (7)$$

Although the content-based algorithm can perform high-power scalability and alleviate the mess in high frequency, there exists a nonstationary problem if the designer uses a constant threshold parameter m^t to statically derive the floating threshold for edge-extracting. Since different macro-blocks with the same threshold parameter will have different R_{ACSA} values, setting the threshold parameters without considering the content variation of macro-blocks will make the subsample rate nonstationary; that is, the power consumption will not be converged within a narrow range for a given power mode. For example, the subsample rate of the Weather clip can vary between 256:67 and 256:224 in the first frame while m^t is set as 0.1. To solve this problem, this paper introduces an adaptive control mechanism to adjust the threshold parameter so that the subsample rate R_{ACSA} can be stationary. The adaptive control mechanism is a run-time process that adjusts the threshold parameter fittingly according to the difference between the current subsample rate and the desired subsample rate (or target subsample rate).

Fig. 4 shows the block diagram of the adaptive control mechanism. Given the battery status, the host processor sets the power mode and the target subsample rate as well. The target subsample rate is N^2 to trg_cnt, where trg_cnt is the target number of 1's in the ACSM. Then, the controller recursively updates the threshold parameter $m^{t+1}(x,y)$ based on the current $m^t(x,y)$ and the difference of $acsm\_cnt$ and trg_cnt, as shown in Fig. 3 under the line "//update threshold parameter for the next frame."

III. Power-Aware Architecture

According to the ACSA, we present a semisystolic architecture as shown in Fig. 5, based on existed architectures, such as [4]. The architecture contains an edge-extraction unit (EXU), an array of processing elements (PEs), a parallel adder tree (PAT), a shift register array (SRA), and a motion-vector selector (MVS).

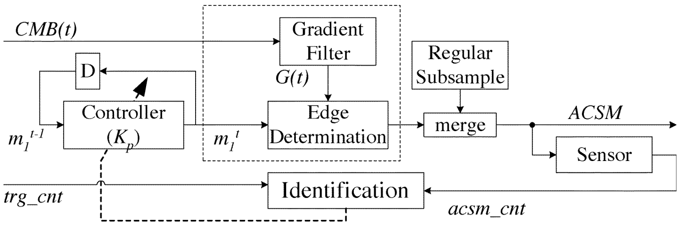


Fig. 4. Block diagram of the edge-extraction unit with adaptive control mechanism.

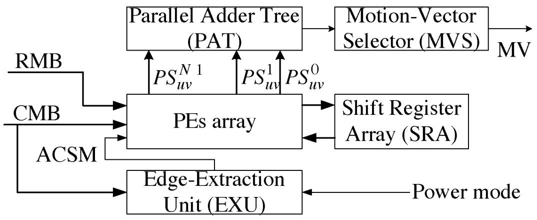

Parallel Adder Tree (PAT)
Motion-Vector Selector (MVS)
MV
RMB
PS_{uv}^{N-1}
PS_{uv}^1
PS_{uv}^0
CMB
PEs array
Shift Register Array (SRA)
ACSM
Edge-Extraction Unit (EXU)
Power mode

Fig. 5. Block diagram of the power-aware ME architecture driven by the ACSA.

Given the power consumption mode, the EXU, which contains the gradient filter and the ACSM generator, extracts high-frequency (or edge) pixels from the current macro-block and generates 0–1 ACSM to disable or enable PEs. The PE array is used to accumulate absolute pixel differences column by column while the parallel adder tree sums up all the results to generate the value of ACSSAD. Each PE has a datapath that consists of the absolute difference unit ($|a-b|$) and the Adder unit. Upon the ACSSAD is generated, the MVS then performs compare-and-select operation to select the best motion vector.

To start with, by performing the edge extraction, the architecture first generates ACSM to determine whether to enable/disable PEs and thus dynamically changes the switching activities of system to reduce the power consumption. The ACSM disables the PE by using a block element (BE) that is implemented by AND gates. The BEs can nullify the input signals of data path. When a PE is disabled during an MV searching iteration, the circuit in the PE remains still until the next iteration starts and, thus, the consumption of transient power can be saved.

IV. Power Model

One can consider the major power consumption of a CMOS gate i as

$$P_{\text{gate}_i} = \alpha \cdot C_i \cdot f_i \cdot V_{DD}^2 = \kappa \cdot C_i \cdot r_i(\updownarrow) \quad (8)$$

where C_i is the output capacitance, f_i is the operation frequency, $r_i(\updownarrow)$ is the switch activity of gate i, and α and κ are constants. For an active execution unit EU_k in a VLSI system, the power consumption can be shown in

$$P_{\text{EU}_k} = \sum_{i=1}^{N_{\text{gate},\text{EU}_k}} \kappa \cdot C_i^k \cdot r_i^k(\updownarrow) \quad (9)$$

where $N_{\text{gate},\text{EU}_k}$ is the gate count of EU_k. The following equations demonstrate the total power consumption:

$$P_{\text{total}} = \sum_{\forall \text{active}\text{EU}_k} \kappa \sum_{i=1}^{N_{\text{gate},k}} C_i^k \cdot r_i^k(\updownarrow) \quad (10)$$

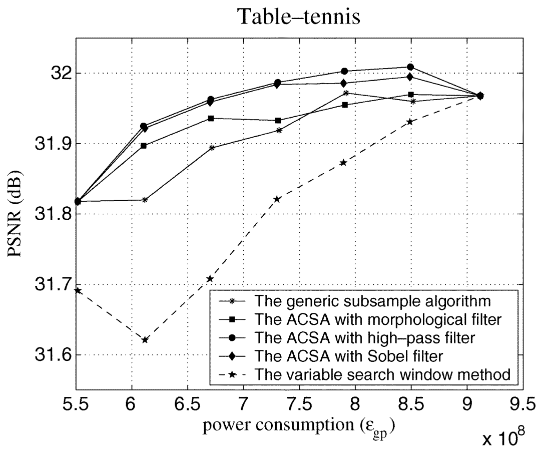


Fig. 6. Quality degradation of the “table tennis” clip.

$$\cong \kappa \sum_{\forall \text{active} \text{EU}_k} r^k(\updownarrow) \sum_{i=1}^{N_{\text{gate},k}} C_i^k \tag{11}$$

$$= \kappa \sum_{\forall \text{active} \text{EU}_k} r^k(\updownarrow) \cdot \frac{\sum_{i=1}^{N_{\text{gate},k}} C_i^k}{N_{\text{gate},k}} \cdot N_{\text{gate},k} \tag{12}$$

$$= \kappa \sum_{\forall \text{active} \text{EU}_k} r^k(\updownarrow) \cdot C_{\text{avg}}^k \cdot N_{\text{gate},k} \tag{13}$$

$$\cong (\kappa \cdot C_{\text{avg}}) \sum_{\forall \text{active} \text{EU}_k} r^k(\updownarrow) \cdot N_{\text{gate},k} \tag{14}$$

$$= \varepsilon_{\text{gp}} \sum_{\forall \text{active} \text{EU}_k} r^k(\updownarrow) \cdot N_{\text{gate},k}. \tag{15}$$

After considering the activity of EUs, the total power consumption can be expressed as (10) and approximated as (11) by assuming the switch activities are uniform within an execution unit; that is, $r_i^k(\updownarrow) = r^k(\updownarrow), \forall r_i^k(\updownarrow)$. Since the average output capacitances of each execution unit (C_{avg}^k) are nearly the same as the average output capacitances of total system (C_{avg}), the total power consumption can be approximated to (14). Therefore, we can obtain an approximated power estimation model shown in (15), where ε_{gp} is defined as the gate power coefficient. In this paper, we use the gate power coefficient as the unit for estimating power dissipation for the power-aware architecture.

V. Results

The peak signal-to-noise ratio (PSNR) of the motion-compensated predicted frame compared to the original frame is adopted as a performance measure. Fig. 6 shows the quality degradation of the CIF video clip “table tennis” with parameters $N = 16$ and $p = 32$. The target subsample rates of GSA and ACSA are set as (4:1), (8:3), (2:1), (8:5), (4:3), (8:7), and (1:1) respectively. In the simulation, we compare our approach with GSA and the variable-search-window approach. The variable-search-window approach is simply achieved by gradually setting the search parameter p to 16, 20, 22, 25, 27, 30, and 32. As shown in the results, the quality degradation of the ACSA is less than that of the GSA and the variable-search-window approach. Fig. 7 illustrates the improvement of ACSA over GSA. Because the ACSA alleviates the aliasing problem for the high-frequency band, the compensated frame with the ACSA has better quality than that with the GSA on the edge of the sportsman’s arm and thumb. In addition, we can find that the type of gradient filter does not make much difference to the performance of the proposed algorithm. Fig. 8 illustrates the stationary response with variable K_p. Obviously, the higher the value of K_p, the shorter the settling time and the worse the stability of the R_{ACSA}. The suitable range of K_p is from 0.1 to 0.5 after testing those four video clips. We analyzed the R_{ACSA} errors of ACSA in Table I. The average R_{ACSA} error is as low as 1.12% and the R_{ACSA} error variance is as low as 0.000 24. Because the subsample rate can be nearly stationary with a given target subsample rate and power mode, the power-awareness of the proposed algorithm is very good and the ACSA can be applied for power-aware architecture.

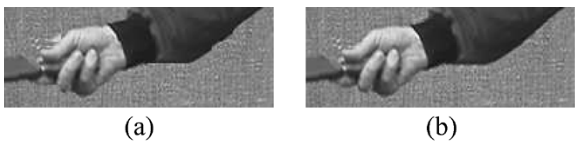


Fig. 7. Nineteenth motion-compensated frame of the “table-tennis” clip. (a) GSA with the 8-to-3 subsample rate and (b) ACSA with the 8-to-3 subsample rate.

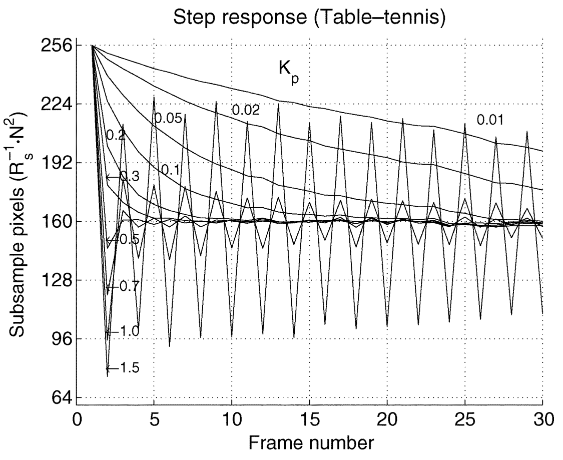


Fig. 8. Step response for varying K_p of the “table tennis” clip.

TABLE I
STATIONARY ERROR OF THE ACSA SUBSAMPLE RATE (R_{ACSA})

Target R_{ACSA}	Average R_{ACSA}	R_{ACSA} Error	Average R_{ACSA}	R_{ACSA} Error
Video Clip	Weather		News	
96	95.416	0.61%	95.366	0.66%
128	127.521	0.37%	127.720	0.22%
160	158.678	0.83%	159.795	0.13%
192	188.037	2.06%	191.313	0.36%
224	211.274	5.68%	221.224	1.24%
Video Clip	Table-Tennis		Children	
96	95.398	0.63%	95.490	0.53%
128	127.467	0.42%	127.754	0.19%
160	159.533	0.29%	159.430	0.36%
192	191.429	0.30%	189.632	1.23%
224	222.893	0.49%	216.001	3.57%

The edge-extraction unit uses the high-pass gradient filter. The control parameter $K_p = 0.3$.

Table II shows the synthesis result using the TSMC 1P4M 0.35-μm cell library, where the symbol R_s can be either the

TABLE II
POWER ANALYSIS OF THE POWER-AWARE ARCHITECTURE

EU_i	PEs array		SRA+PAT+MVS	EXU
	AD + Adder	Others		
N_{gate,EU_i}	117,760	58,708	46,440	17,121
$r^i(\updownarrow)$	$4p^2R_s^{-1}$	$4p^2$	$4p^2$	N^2
P_{EU_i}	$4.8 \cdot 10^8 \cdot R_s^{-1}$	$2.4 \cdot 10^8$	$1.9 \cdot 10^8$	$4.38 \cdot 10^6$
$P_{total}(\varepsilon_{gp})$	$4.8 \cdot 10^8 \cdot R_s^{-1} + 4.3 \cdot 10^8$			

Area Overhead $= N_{\mathrm{gate,EXU}}/(N_{\mathrm{gate,PEs\ array}} + N_{\mathrm{gate,SRA+PAT+MVS}})$
Power Overhead $= P_{\mathrm{EXU}}/(P_{\mathrm{PEs\ array}} + P_{\mathrm{SRA+PAT+MVS}})$
$N = 16$
$p = 32$
Cell library: TSMC 0.35-μm process.

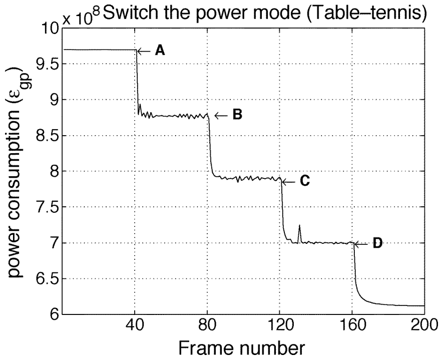


Fig. 9. Application for switching the power mode of the video clip "table-tennis."

adaptive content-based subsample rate or the generic subsample rate, and $\varepsilon_{\mathrm{gp}}$ is the gate power coefficient defined in (15). Comparing with the general semisystolic architecture [4], the edge-extraction unit (EXU) of the proposed architecture is the major overhead for the power-aware function. As mentioned above, this paper uses one of three gradient filters to implement the EXU. As per the synthesis results, the gate counts of the three gradient filters are 595.33, 793.77, and 727.63, respectively. The variance of these values are very little to the overall gate count of EXU. This means that the selection of the gradient filter does not affect the overhead estimation much. Therefore, we selectively use a high-pass filter to estimate the performance overhead caused by EXU which dominates the area/power overheads. From the results, the area overhead of EXU is 7.68% and the power overheads are 0.8% and 0.5% for 4-to-1 and 1-to-1 subsample rates, respectively. Fig. 9 shows an example of the video clip "table-tennis" for switching the power consumption mode. The target subsample pixel count is reduced by 48 every 40 frames. The result shows that the adaptive control mechanism can make the power consumption reach the target level within 10 frames. According to the battery properties described in Section I, the curve shows that our power-aware architecture can match the behavior by slowly and gradually degrading the quality. The marks A, B, C, and D correspond to the power-switching points.

VI. CONCLUSION

Motivated by the concept of battery properties and the power-aware paradigm, this paper presents an architecture-level power-aware technique based on a novel adaptive content-based subsample algorithm. When the battery is in the status of full capacity, the proposed ME architecture will turn on all of the PEs to provide the best compression quality. In contrast, when the battery capacity is short for full operation, instead of exhibiting an all-or-none behavior, the proposed architecture will shift to lower power consumption mode by disabling some PEs to extend the battery lifetime with little quality degradation. As shown in the simulation results, the proposed algorithm successfully improves the compression quality of the generic subsample algorithm and does not introduce much power dissipation.

ACKNOWLEDGMENT

The authors would like to thank National Chip Implementation Center (CIC), Taiwan, R.O.C., for technical support.

REFERENCES

[1] P. Kuhn, *Algorithms, Complexity Analysis And VLSI Architectures for MPEG-4 Motion Estimation*. Norwell, MA: Kluwer, 1999.
[2] M. Bhardwaj, R. Min, and A.-P. Chandrakasan, "Quantifying and enhancing power awareness of VLSI systems," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 9, no. 6, pp. 757–772, Dec. 2001.
[3] D. Linden, *Handbook of Batteries*, 2nd ed. New York: McGraw-Hill, 1995.
[4] C.-H. Hsieh and T.-P. Lin, "VLSI architecture for block-matching motion estimation algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 2, no. 2, pp. 169–175, Jun. 1992.
[5] J.-C. Tuan, T.-S. Chang, and C.-W. Jen, "On the data reuse and memory bandwidth analysis for full-search block-matching VLSI architecture," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 1, pp. 61–72, Jan. 2002.
[6] R. Li, B. Zeng, and M.-L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 438–442, Aug. 1994.
[7] J.-Y. Tham, S. Ranganath, M. Ranganath, and A.-A. Kassim, "A novel unrestricted center-biased diamond search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 4, pp. 369–377, Aug. 1998.
[8] C. Zhu, X. Lin, and L.-P. Chau, "Hexagon-based search pattern for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 5, pp. 349–355, May 2002.
[9] K. Sauer and B. Schwartz, "Efficient block motion estimation using integral projections," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 5, pp. 513–518, Oct. 1996.
[10] J.-H. Luo, C.-N. Wang, and T. Chiang, "A novel all-binary motion estimation (ABME) with optimized hardware architectures," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 8, pp. 700–712, Aug. 2002.
[11] O.-S. Unsal and I. Koren, "System-level power-aware design techniques in real-time systems," *Proc. IEEE*, vol. 91, no. 7, pp. 1055–1069, Jul. 2003.
[12] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 2, pp. 148–157, Apr. 1993.
[13] C.-K. Cheung and L.-M. Po, "Normalized partial distortion search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 3, pp. 417–422, Apr. 2000.

PAPER

A Parallel-In Folding Technique for High-Order FIR Filter Implementation

Lan-Rong DUNG[†], ***Member and*** **Hsueh-Chih YANG**[†], ***Nonmember***

SUMMARY This paper presents a hardware-efficient folding technique for high-order FIR filtering while considering the tradeoff between the number of processing elements and throughput rate. Given the throughput rate, one can always employ the minimum number of processing elements for saving the implementation cost and figure out a folded architecture. However, applying inefficient folding techniques may result in costly switches and registers. Therefore, our work intends to evaluate the efficiency for folding techniques in terms of the number of registers, and the power dissipation of registers. As shown in the estimation results, while comparing with the published folded architectures under the same throughput rate, the proposed folding technique can turn out less power dissipation and low hardware complexity than the others. The proposed design has been implemented using TSMC 0.18 μm 1P6M technology. As seen in the post-layout simulation, our design can meet the requirement of IS-95 WCDMA pulse shaping FIR filter while the power consumption can be as low as 16.66 mW.
key words: *FIR, VLSI hardware, digital filters.*

1. Introduction

FIR filtering is an essential function in most DSP applications, such as telecommunication and multimedia systems. Its guaranteed stability and simple structure make FIR itself a popular technique for removing unwanted parts of signal. For quality-sensitive applications, the number of FIR taps is normally large, ranges from tens to hundreds. However, long-length (or high-order) FIR filters may result in costly hardware and hence severe power consumption problem. Furthermore, a long-length FIR architecture may suffer from the clock-skew problem.

For decades, many approaches have been proposed to reduce the hardware complexity of FIR filtering [1-14]. They can be classified into three categories: multiplier reduction [4], [5], [8], [9], [11], [12], multiplierless realization [1], [6], [13], and resource sharing (or folding) [2], [7], [10], [14], [15]. The first category can reduce the number of multiplications at the expense of increased number of additions or pre-/post-processing units. The second category mainly uses the distributed arithmetic(DA) technique to dismiss multipliers. Both categories are able to significantly reduce computational complexity but lack flexibility for cost-optimal FIR implementation. The third category is proposed for reducing the datapath cost when the throughput rate is lower than the maximum speed at which a full-length datapath can operate. That is, the folding techniques intend to trade excessive speed for datapath cost with high degree of flexibility.

[10] and [2] present the first systematic transformation technique to fold DSP algorithms. The folding technique was proposed for efficient resource sharing to meet both throughput and resource constraints, while it allows identical operations to be time-multiplexed by the same circuit unit. The technique is attractive not only for its area saving but also for the alleviation of the clock-skew problem. Following [10] and [2], [7] and [14] present folded FIR architectures for variable folding factor and bit-level pipelined implementations, respectively. Both of them take the advantage of folding technique to meet their objectives. The existed folded architectures mainly focus on the utilization of processing elements or data path; however, few emphasize the cost of registers and control units. Note that when the processing elements have been carefully allocated for performance constraints registers and control units will become the main issues to the efficiency of folding techniques. Thus, this paper presents two folding techniques based on algebraic and structural transformations. As shown in the estimation results, the proposed techniques can outperform the others in terms of cost and power dissipation.

[†]The authors are with the Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu 30010, Taiwan

2. Candidates of Folding Techniques

There are five folding techniques being targeted as candidates. Three of them are published in [7], [10], and [15] while the others are proposed in this paper at the first time. Later, we will use the following notations to explain the techniques.
m = the bitwidth of input sample
b = the bitwidth of coefficient
K = the number of filter taps
r = the number of multiplier-adders (MAs)
f = the folding factor ($f = \lceil \frac{K}{r} \rceil$)
T_{MA} = the execution time of MA unit
$xput_{spec}$ = the specified throughput

2.1 Existing folding techniques

Article [10] presents a systematic transformation technique to fold DSP algorithms. At first, it maps the nodes of the signal flow graph (SFG) into folding elements; each of them is composed of one or more identical operations. Given the folding factor f and a K-tap FIR, there exist $\lceil K/f \rceil$ (or r) processing elements (PEs) (or MA) in the target architecture, and up to f equally structured folding elements will be assigned to the same PE. The executing order of folding elements, being assigned to the same PE, is determined by the folding-order number; the folding-order number is set by the task scheduling of folding elements. Finally, the technique requires delay-insertion to synchronize the input of coefficients. The synchronization is realized by registers clocked at the sampling rate, a multiplexer, and a MOD f counter. Fig. 1 illustrates an example with K=6 and r=2. **The paper [15] presents a low power synthesis approach by using the unfolding technique with coefficient reordering. However, it uses the same hardware mapping strategy as [10]. Unfolding a filter, by a factor F will result in a F-parallel filter topology. The memory requirement is approximatively proportional to the unfolding factor F.**

Following the systematic folding technique, article [7] presents the folded bit-plane FIR architecture and folds the FIR filter algorithm bitwisely. The folded bit-plane FIR architecture enables the implementation of changeable folding factor onto a fixed-size systolic array. The systolic array reads the input word $x[n]$ in the manner of bit-serial, and generates the partial products bit-by-bit with the coefficients. The partial products will then be accumulated for the calculation of the result $y[n]$. In the folded bit-plane architecture, the folding factor f is equal to the programmable bit-width of coefficients and the throughput of the folded architecture can be increased by reducing the bit-width of coefficients. Fig. 2 is the diagram of the folded bit-plane FIR presented in [7].

2.2 The architecture of the serial-in folded FIR filter

The transfer function of the K-tap FIR filter can be reformulated as Eq. 1 by recursively calculating f-tap FIR filtering.

$$\sum_{k=0}^{K-1} h_k \cdot z^{-k} = \sum_{i=0}^{r-1} z^{-fi} \cdot \sum_{j=0}^{f-1} h_{fi+j} \cdot z^{-j} \tag{1}$$

where $f = \frac{K}{r}$. From Eq. 1, the K-tap FIR filter is composed of r short-length FIRs ($\sum_{j=0}^{f-1} h_{fi+j} \cdot z^{-j}$), and can be realized by the SFG as shown in Fig. 3(a). Given the folding factor f, the SFG can be then mapped onto r PEs by (1,0) projection. The folded architecture in Fig. 3(b) is called the serial-in folded FIR. In the architecture, each PE serially executes the short-length FIR on input samples and the coefficients h_{fi+j} are circular in cyclic shift

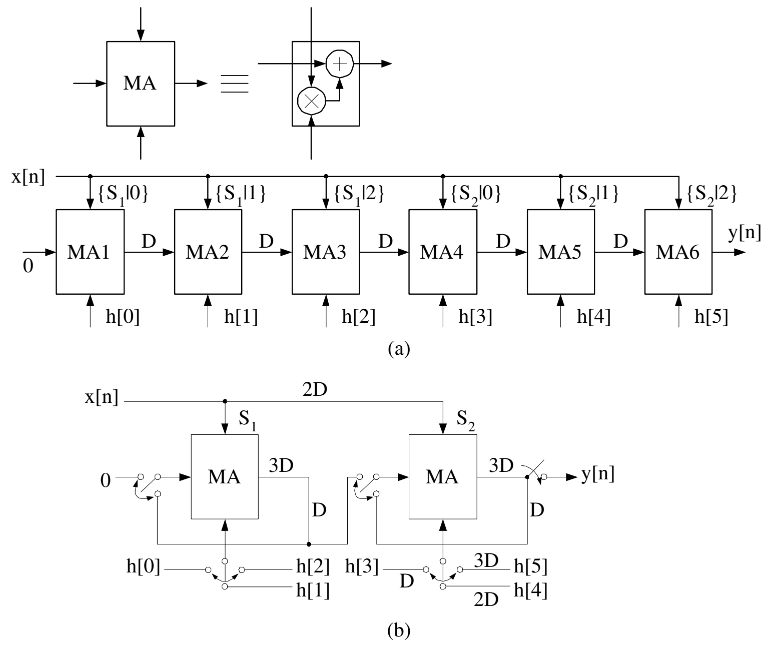


Fig. 1 A folding example of FIR filter. (a) A 6-tap FIR filter in the transposed form. (b) The folded architecture of Fig. 1(a) by using the technique presented in [10].

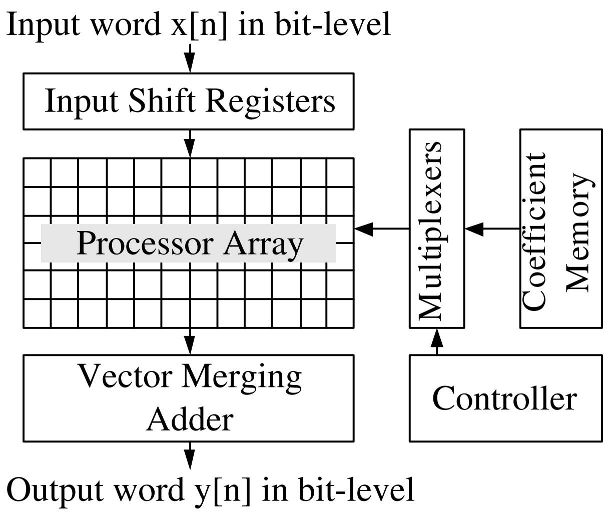
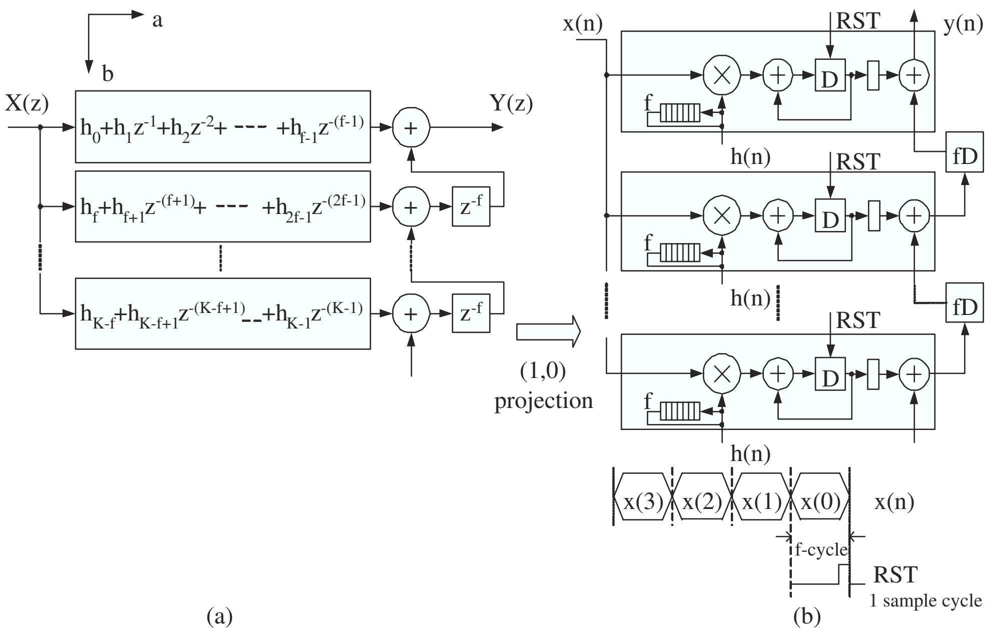


Fig. 2 The diagram of the folded bit-plane FIR architecture.



a
b
X(z)
Y(z)
h0+h1z-1+h2z-2+ --- +hf-1z-(f-1)
hf+hf+1z-(f+1)+ --- +h2f-1z-(2f-1)
hK-f+hK-f+1z-(K-f+1)--+hK-1z-(K-1)
z-f
(1,0) projection
x(n)
y(n)
RST
D
fD
f
h(n)
x(3) x(2) x(1) x(0)
f-cycle
1 sample cycle
(a)
(b)

Fig. 3 (a) The SFG of the reformulated K-tap FIR filter. (b) The serial-in folded architecture.

registers of PEs. As shown in Fig. 3(b), the clock rate is f times of the sampling rate and the the signal "RST" is asserted every f clock cycles. The signal "RST" is used to clear the accumulator before the new sample comes. Finally, the f-stage shift registers fD are used to buffer the results of short-length FIRs and realize the function of z^{-fi}.

2.3 The architecture of the parallel-in folded FIR filter

For an input sequence $x(n)$ and filter coefficients $h(n)$, the output sequence $y(n)$ is given by Eq. 2.

$$y(n) = \sum_k x(k)h(n-k) \tag{2}$$

According to the Eq. 2, the data path of FIR filtering can be presented by an SFG shown in the Fig. 4 where D is the delay element. Because there are only r MAs available for the folding factor f, the FIR filtering can execute r MA-nodes in parallel at the maximum within a signal cycle and requires $\lceil \frac{K}{r} \rceil$ (or f) cycles to finish an iteration. Hence, given r MAs, the iteration period is bounded by f cycles and f is the folding factor. To fold the execution of the K-tap FIR by f, we reconstructed a K-tap FIR to the r-split SFG as shown in the Fig. 5. At first, the delay elements of the original SFG are scaled by the folding factor f so each iteration of the K-tap requires f cycles. Then, we performed the retiming transforms on the edges in the cut-sets as shown in Fig. 5. **The cut-set is used to segment the retimed graph into f subgraphs(r-split graph). ($f = \lceil \frac{K}{r} \rceil$). Each subgraph has r or less than r MA operations. Afterward those f subgraphs would be executed in the same r MAs by turns. The terms r and f are defined as the number of hardware resource of MAs and how many equally structured operations will be assigned to the same hardware, respectively.** To execute the subgraphs in order, the FIR coefficients are scheduled as illustrated in the Fig. 6. Because the input samples are read into the MA array in parallel, the folding technique is called the parallel-in folded FIR.

Fig. 7 illustrates the parallel-in folded FIR architecture, where w is the bitwidth of data bus. Given r MAs, the folding factor becomes f and the K-tap FIR requires f cycles for an iteration. **There are K b-bit registers in coefficient register bank. The configuration is composed of r groups. ($f = \lceil \frac{K}{r} \rceil$). Each group contains f filter coefficients and simultaneously provides them one by one to the corresponding MA**

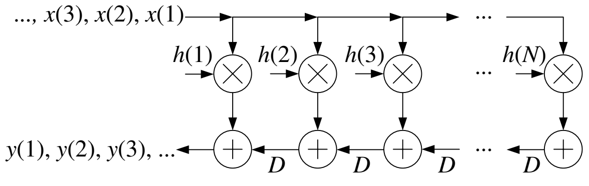


Fig. 4 The flow graph of FIR filtering.

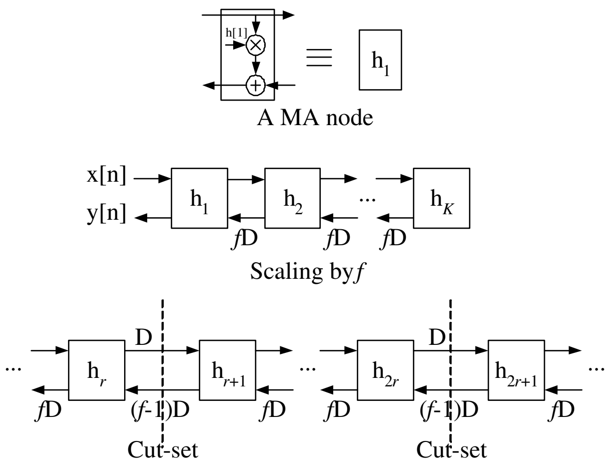


Fig. 5 The r-split FIR filtering.

cycle	MA1	MA2	···	MAr
1	h_0	h_1	···	h_{r-1}
2	h_r	h_{r+1}	···	h_{2r-1}
⋮	⋮	⋮	···	⋮
f	$h_{(f-1)r}$	$h_{(f-1)r+1}$	···	h_{fr-1}
⋮	⋮	⋮	···	⋮
⋮	⋮	⋮	···	⋮

Fig. 6 The scheduling of FIR coefficients for the parallel-in folded technique.

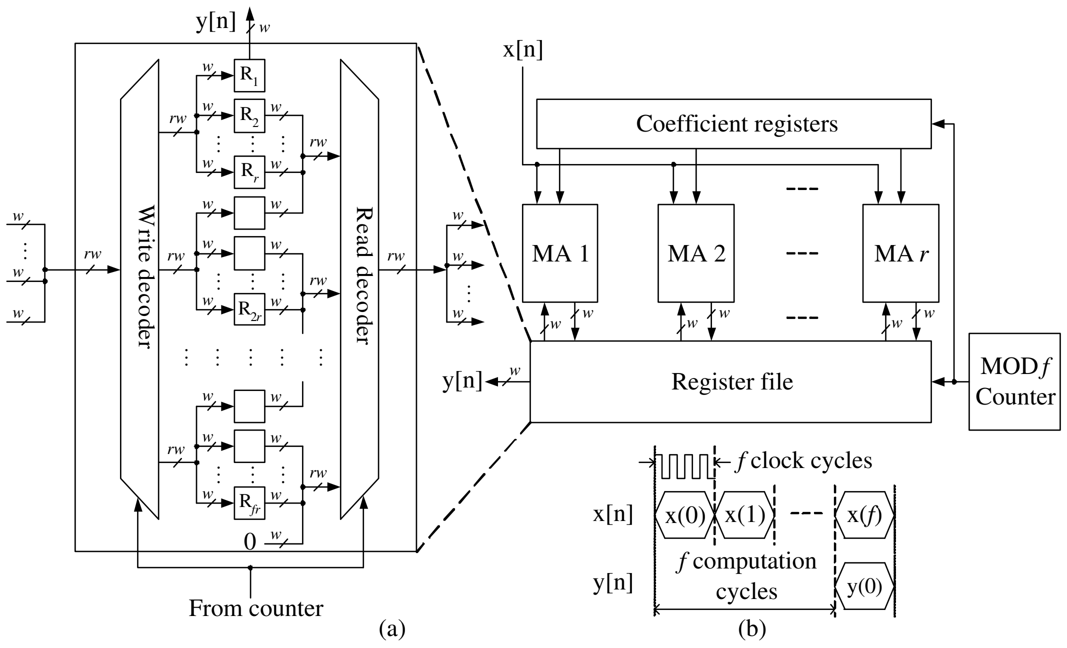


Fig. 7 (a) The architecture of the parallel-in folded FIR filter, and (b) the timing diagram.

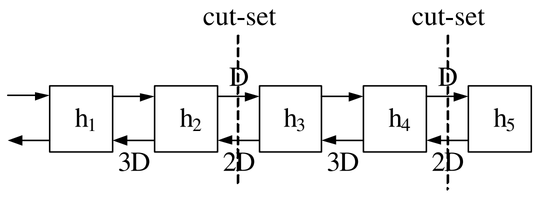


Fig. 8 The 2-split, 5-tap FIR filtering.

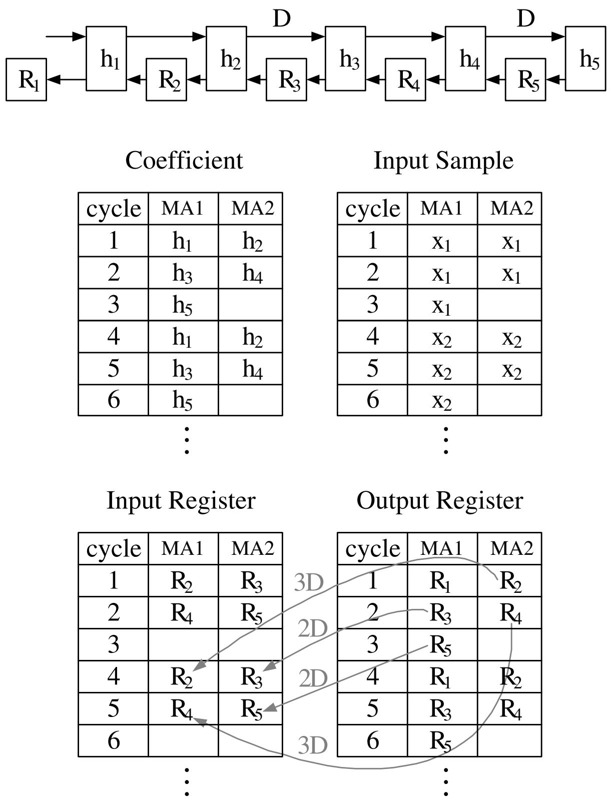
unit according to the scheduled order shown in Fig.6. The register file is used to buffer the output of MAs so that the subgraphs of r-split FIR can be executed recursively. The counter is used to schedule the inputs and outputs of MA array for the FIR working correctly. Fig. 8 and Fig. 9 demonstrate the scheduling for an example of 5-tap FIR with two MAs. Given $r = 2$, the folding factor f is equal to three and we can obtain the 2-split SFG of Fig. 8. To map the SFG to the parallel-in architecture, we replaced the delay elements with registers and generated the scheduling as shown in Fig. 9. Note that each delay element is not necessary to be realized as a register if one can properly schedule the data storage. Following the scheduling, the MA units can perform the function of Eq. 3 in each cycle and produce the output $y(n)$ in R_1 every three cycles.

$$R_i = x(n)h_i + R_{i+1} \quad i = 1, 2, 3, 4 \tag{3}$$

3. Comparison results

Given the folding factor f for K-tap FIR, all the folding techniques require the same number of MAs and has the same throughput rate and, hence, the efficiency of folding techniques is determined by size and power consumption of memory. At the stage of high-level synthesis, we consider the number of D-type flip-flops (DFFs) as the size of memory and the access number of DFFs per iteration as the power dissipation of memory. When making the comparison, we set the bitwidth of data bus w as $m + b + \lceil \log_2 K \rceil$ for full-precision FIR calculation.

Eq. 4 formulates the number of DFFs required by [10]. In [10], there are r MAs located on the accumulated



Coefficient

cycle	MA1	MA2
1	h_1	h_2
2	h_3	h_4
3	h_5	
4	h_1	h_2
5	h_3	h_4
6	h_5	

⋮

Input Sample

cycle	MA1	MA2
1	x_1	x_1
2	x_1	x_1
3	x_1	
4	x_2	x_2
5	x_2	x_2
6	x_2	

⋮

Input Register

cycle	MA1	MA2
1	R_2	R_3
2	R_4	R_5
3		
4	R_2	R_3
5	R_4	R_5
6		

⋮

Output Register

cycle	MA1	MA2
1	R_1	R_2
2	R_3	R_4
3	R_5	
4	R_1	R_2
5	R_3	R_4
6	R_5	

⋮

Fig. 9 The scheduling of FIR filtering.

loop. In Eq. 4, we use the term α_0 to express the number of DFFs for coefficient storage. Because each MA has $(f+1)$ registers with full precision to hold the accumulating results, the term α_1 means the total number of DFFs for r MAs. The term α_2 represents the number of DFFs for multiplexing the input samples to MAs. Because the coefficients are cyclically read by MAs, the number of registers for multiplexing coefficients (in each PE) increasingly varying with the MA changes. Hence the total number of registers for multiplexing coefficients can be calculated by q and the term α_3 represent the number of DFFs for multiplexing coefficients. Finally, the controller can be implemented as a counter and the term α_4 counts the number of DFFs required in the controller. **Eq. 5 shows the number of DFFs required by [15]. The unfolding factor, *F*, will result in a *F*-parallel filter topology. The memory requirement is approximatively proportional to the unfolding factor *F*.**

$$\#DFF_{[10]} = \underbrace{Kb}_{\alpha_0} + \underbrace{r(m+b+\lceil \log_2 K \rceil)(f+1)}_{\alpha_1} + \underbrace{(K-f)m}_{\alpha_2} + \underbrace{bq}_{\alpha_3} + \underbrace{\lceil \log_2 f \rceil}_{\alpha_4}, \text{where } q = \sum_{i=1}^{K-f} i. \tag{4}$$

$$\boxed{\#DFF_{[15]} = Kb + F(r(m+b+\lceil \log_2 K \rceil)(f+1) + (K-f)m + bq + \lceil \log_2 f \rceil), \text{where } q = \sum_{i=1}^{K-f} i} \tag{5}$$

Eq. 6 formulates the number of DFFs required by the folded bit-plane FIR architecture [7]. The folded bit-plane FIR arechitecture requires $K\times$ $(m+b+\lceil \log_2 K \rceil)$ PEs. Each PE performs 1-bit addition and needs two DFFs for the carry-out and sum signals. In addition, the length of the input shift register is $(m+b+\lceil \log_2 K \rceil)$. Thus, the total number of DFFs for PEs can be expressed by the term α_5. The term α_6 is the total number of DFFs for coefficient registers. Finally, the term α_7 represents the number of DFFs in the controller.

$$\#DFF_{[7]} = \underbrace{(m+b+\lceil \log_2 K \rceil)(2K+1)}_{\alpha_5} + \underbrace{Kb}_{\alpha_6} + \underbrace{\lceil \log_2 b \rceil}_{\alpha_7} \tag{6}$$

The formulations of our proposed folded FIR techniques are shown in Eq. 7 and Eq. 8. The details are as follows. The term α_8 represents the number of DFFs of the coefficient registers. The term α_9 counts the number of DFFs in accumulators and latches of the serial-in folded FIR. The term α_{10} is the number of DFFs for the out-loop delays, fD, as shown in Fig. 3. Because there is a counter to generate the signal "RST", the term α_{11} expresses the number of DFFs of the counter. For the parallel-in folded FIR, the term α_{12} gives the number of DFFs in the register file, α_{13} represents the number of DFFs of the coefficient registers, and α_{14} expresses the number of DFFs for the MOD-f counter.

$$\#DFF_{serial-in} = \underbrace{bfr}_{\alpha_8} + \underbrace{2r(m+b+\lceil \log_2 f \rceil)}_{\alpha_9} + \underbrace{f(r-1)(m+b+\lceil \log_2 K \rceil)}_{\alpha_{10}} + \underbrace{\lceil \log_2 f \rceil}_{\alpha_{11}} \tag{7}$$

$$\#DFF_{parallel-in} = \underbrace{K(m+b+\lceil \log_2 K \rceil)}_{\alpha_{12}} + \underbrace{Kb}_{\alpha_{13}} + \underbrace{\lceil \log_2 f \rceil}_{\alpha_{14}} \tag{8}$$

We estimated the power consumption of memory by counting the access number of registers of each computation iteration [3]. The following equations list the estimation results for five candidates.

$$\#reg\_access_{[10]} = f(r(m+b+\lceil \log_2 K \rceil)(f+1) + (K-f)m) \tag{9}$$

$$\boxed{\#reg\_access_{[15]} = F(f(r(m+b+\lceil \log_2 K \rceil)(f+1) + (K-f)m))} \tag{10}$$

$$\#reg\_access_{[7]} = m((m+b+\lceil \log_2 K \rceil)(2K+1) + K) \tag{11}$$

$$\#reg\_access_{serial-in} = f(m + (m+b+\lceil \log_2 f \rceil)(r(f+1)-f)) \tag{12}$$

$$\#reg\_access_{parallel-in} = 2rf(m+b+\lceil \log_2 K \rceil) + bfr \tag{13}$$

Additionally, we formulated the occupancy of multiplexers for five folded architectures by the

number of the 1-bit 2-to-1 multiplexer as follows:

$$\#MUX_{[7]} = 2Kw + Kb \tag{14}$$

$$\#MUX_{[10]} = r(w(f-2) + b(f-1)) + w \tag{15}$$

$$\#MUX_{[15]} = F(r(w(f-2) + b(f-1)) + w) \tag{16}$$

$$\#MUX_{serial-in} = rw \tag{17}$$

$$\#MUX_{parallel-in} = 2rwf \tag{18}$$

To graphically compare the candidates, we sketched the results for K=255 and m=8 (in log scale), as shown in Fig. 10, Fig. 11, and Fig. 12. As shown in Fig. 10, the serial-in folded FIR has the lowest memory requirement for large f while the parallel-in folded FIR has the edge for small f. With regard to the power consumption, the folded FIR of [10] and the serial-in folded FIR exponentially grows with the increasing value of folding factor. According to Fig. 11, the parallel-in folded FIR consumes the least power than others.

Taking the IS-95 WCDMA pulse shaping FIR filter, whose specification is tabulated in Table 1, as an example, we have implemented five architectures and estimated area requirements and power consumption by the Synopsys Design Analyzer and PrimePower. **VLSI design exists a trade-off between operational speed and silicon area occupation. The main argument in this paper is to save silicon area and power consumption based on the same speed. Basically, each architecture's critical-path delay is the latency between one MA and one multiplexer. We specified the clock constraint of each architecture in the synthesizing stage to let the comparisons make sense.** The target report of each architecture is summarized in Table 2. As we can see that the folding technique can save the area requirement but maximal resource sharing can lead to an increase in power consumption. However, our proposed folded architectures consume less power among all folded ones. The parallel-in folded FIR filter with the folding factor $f = 11$ was fabricated using a 0.18μm CMOS

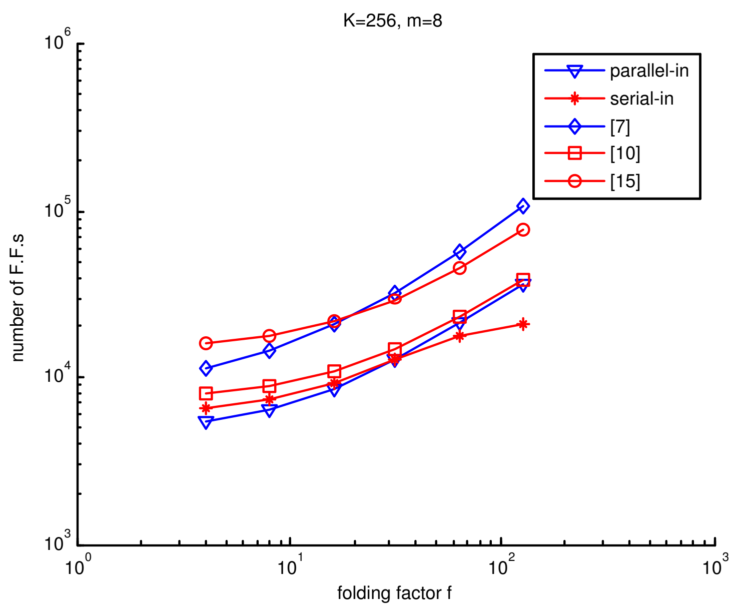


Fig. 10 Number of DFFs of folded architectures (in log scale)

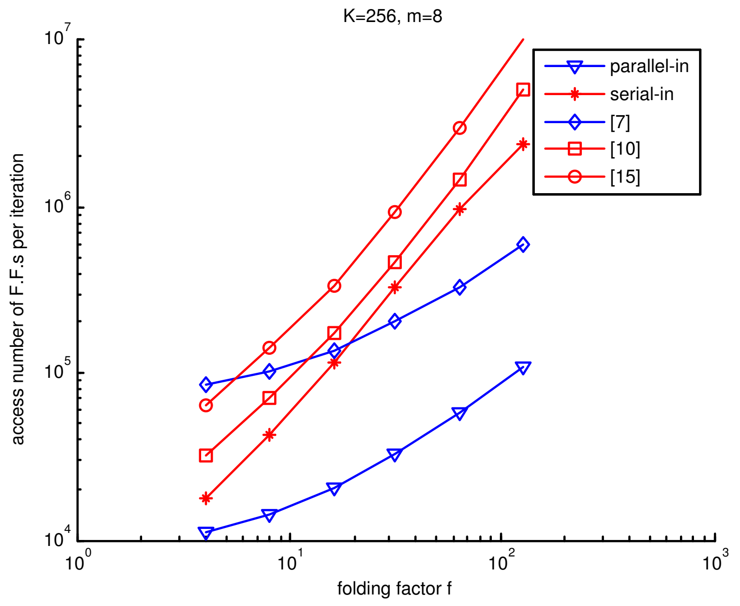


Fig. 11 Access number of DFFs per iteration (in log scale)

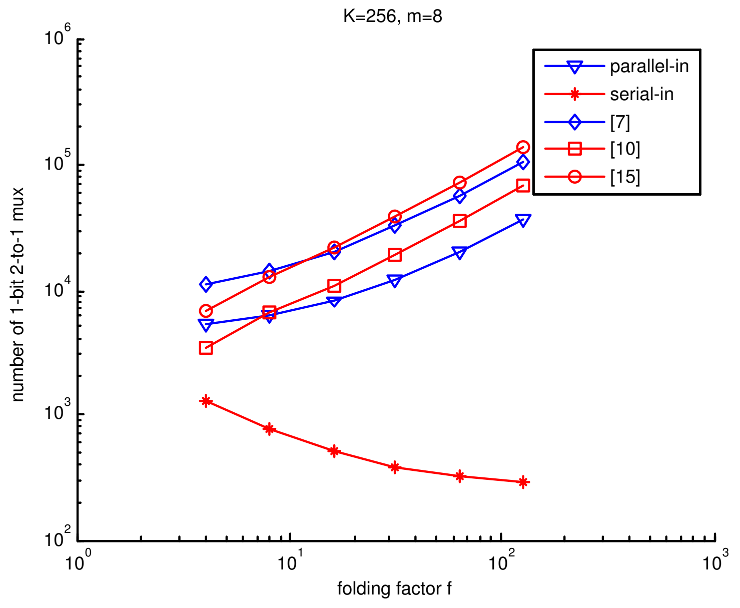


Fig. 12 Number of 1-bit 2-to-1 multiplexers of folded architectures (in log scale)

filter length	33 tap
throughtput	15.36 MSPS
passband edge	$0.1\pi\omega$
stopband edge	$0.28\pi\omega$
passband ripple	1.5 dB
stopband ripple	40 dB
input sample word-length	8 bit
coefficient word-length	16 bit

Table 1 IS-95 WCDMA pulse shaping FIR filter specification.

FIR Architecture	unfolded	[7]	[10]	[15] $(F = 2)$	serial-in	parallel-in
Area (μm^2)	2311048	2231788	865477	1627096	578116	758781
Power (mW)	6.79	49.9	34.8	70.2	25.45	16.66
Critical-path (ns)	65.1	4.07	5.92	5.92	5.92	5.92

Table 2 Area and power consumption comparisons.(An IS-95 WCDMA pulse shaping 33-tap FIR)

throughput	168.96 MSPS
power dissipation	16.66 mW
chip size	$1.411 \times 1.411 mm^2$
supply voltage(core/ring)	1.8 V/3.3 V

Table 3 Features of the IS-95 WCDMA pulse shaping FIR filter chip.

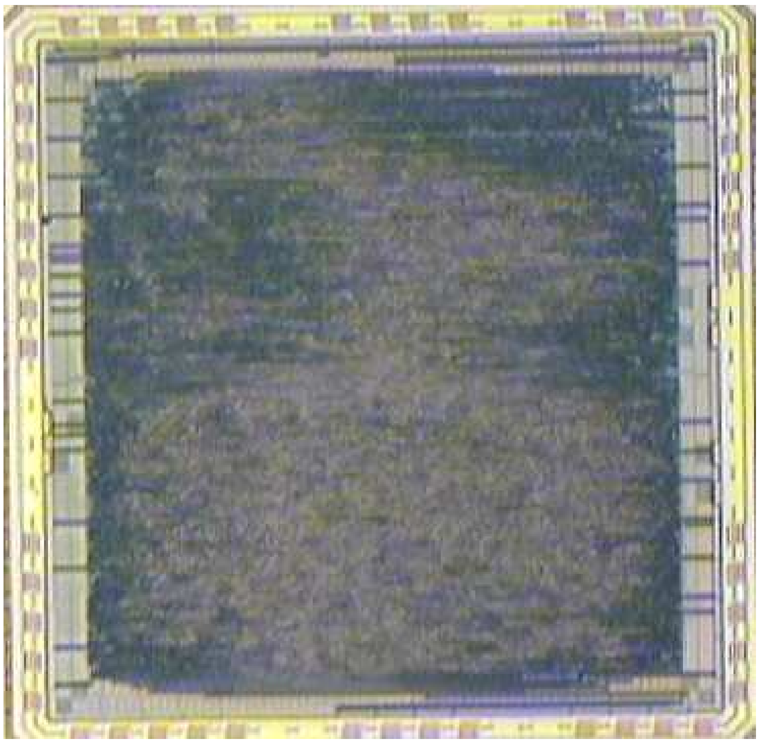


Fig. 13 Photomicrograph of IS-95 WCDMA pulse shaping FIR filter chip.

technology, packaged in 68-pin LCC, and successfully passed functional testing. The features of the implementation and the chip micrograph are given in Table 3 and Fig. 13, respectively.

4. Conclusion

Two novel systematic hardware-efficient folding techniques for high-order FIR filtering have been presented. The parallel-in folded design methodology was applied to the design of an IS-95 WCDMA pulse shaping FIR filter. It features a sample rate of 168.96 MSPS at a power dissipation of 16.66 mW in a $0.18\mu m$ CMOS technology. Under the same throughput rate, the proposed techniques enable the validation of the architecture of the folded FIR filter with minimal storage requirement and less power dissipation when comparing with that of the previous works in the literatures.

Acknowledgments

This work was supported by the National Science Council, R.O.C., under the grant number NSC 94-2220-E-009-023. We gratefully acknowledge the implementation work made by Ming-Feng Yaung.

References

[1] C. S. Burrus, “Digital filter structures described by distributed arithmic,” *IEEE Trans. on Circuits Syst.*, pp. 674–680, Dec. 1977.

[2] T. C. Denk and K. K. Parhi, “Synthesis of folded pipelined architectures for multirate DSP algorithms,” *IEEE Trans. on VLSI*, vol. 6, no. 4, pp. 595–607, Dec. 1998.

[3] Rashindra Manniesing, R. Kleihorst, A. V. D. Avoird, and Emile Hendriks “Power analysis of a general convolution algorithm mapped on a linear processor array,” *Journal of VLSI Signal Processing*, vol. 37, pp. 5–19, May 2004.

[4] S.-F. Lin, S.-C. Huang, F.-S. Yang, C.-W. Ku, and L.-G. Chen, “Power-efficient FIR filter architecture design for wireless embedded system,” *IEEE Trans. on Circuits Syst. II*, vol. 51, no. 1, pp. 21–25, Jan. 2004.

[5] E. Lueder, “Generation of equivalent block parallel digital filters and algorithms by a linear transformation,” in *IEEE Int. Symp. on Circuits and Systems*, , pp. 495–498, May 1993.

[6] M. Mehendale and S. D. Sherlekar, *VLSI SYNTHESIS OF DSP KERNELS- Algorithmic and Architectural Transformations.* KLUWER ACADEMIC PUBLISHERS, 2001.

[7] I. Milentijevic, V. Ciric, T. Tokic, and O. Vojinovic, “Folded bit-plane FIR filter architecture with changable folding factor,” in *IEEE Euromicro Sym. on Digital System Design*, pp. 45–52, Sept. 2002.

[8] Z. J. Mou and P. Duhamel, “Short-length FIR filters and their use in fast nonrecursive filtering,” *IEEE Trans. on Signal Processing*, vol. 39, no. 6, pp. 1322–1332, June 1991.

[9] A. V. Oppenheim and R. W. Schafer, *Discrete-time signal processing*, 2nd ed. PRENTICE HALL, 1999.

[10] K. K. Parhi, C.-Y. Wang, and A. P. Brown, “Synthesis of control circuits in folded pipelined DSP architecture,” *IEEE Journal of Solid-State Circuits*, vol. 27, no. 1, pp. 29–43, Jan. 1992.

[11] D. A. Parker and K. K. Parhi, “Low-area/power parallel FIR digital filter implementations,” *J. VLSI Signal Processing Syst.*, vol. 17, no. 1, pp. 75–92, 1997.

[12] D. N. Pearson and K. K. Parhi, “Low-power FIR digital filter architectures,” in *IEEE Int. Symp. on Circuits and Systems*, pp. 231–234, Apr. 1995.

[13] S. White, “Applications of distributed arithmetic to digital signal processing: A tutorial review,” *IEEE ASSP Magazine*, pp. 4–19, July 1989.

[14] P. Bougas, P. Kalivas, A. Tsirikos, and K. Z. Pekmestzi, “Pipelined Array-Based FIR Filter Folding,” *IEEE Trans. on Circuits and Systems-I*, vol. 52, no. 1, pp. 108–118, Jan. 2005.

[15] Vijay Sundararajan and K. K. Parhi, “SYNTHESIS OF LOW POWER FOLDED PROGRAMMABLE COEFFICIENT FIR DIGITAL FILTERS,” *in Proc. ASPDAC*, pp. 153–156, Jan. 2000.

Lan-Rong Dung was born in 1966. He received a BSEE and the Best Student Award from Feng Chia University, Taiwan, in 1988, an MS in electronics engineering from National Chiao Tung University, Taiwan, in 1990, and Ph.D. in electrical and computer engineering from Georgia Institute of Technology, in 1997. From 1997 to 1999 he was with Rockwell Science Center, Thousand Oaks, CA, as a Member of the Technical Staff. He joined the faculty of National Chiao Tung University, Taiwan in 1999 where he is currently an associate professor in the Department of Electrical and Control Engineering. He received the VHDL International Outstanding Dissertation Award celebrating in Washington DC in October, 1997. His current research interests include VLSI design, digital signal processing, hardware-software codesign, and System-on-Chip architecture. He is a member of Computer and Signal Processing societies of the IEEE.

Hsueh-Chih Yang was born in 1978. He received the B.S degree in Mechanical Engineering from National Central University, Taoyuan, Taiwan, R.O.C. in 2002. He is currently working toward the Ph.D degree in the Electrical and Control Engineer, National Chiao Tung University. His research interests are power aware system, VLSI architecture, and digital signal processing.

System Level Verification on High-Level Synthesis of Dataflow Algorithms Using Petri Net

Tsung-Hsi Chiang & Lan-Rong Dung
Department of Electrical and Control Engineering
National Chiao Tung University
300, Hsinchu City
Taiwan, R.O.C.
aries.ece89g@nctu.edu.tw

Abstract: - This paper presents a formal verification algorithm using the Petri Net theory to detect design errors for high-level synthesis of dataflow algorithms. Typically, given a dataflow algorithm and a set of architectural constraints, the high-level synthesis per-forms algorithmic transformation and produces the optimal scheduling. How to verify the correctness of high-level synthesis becomes a key issue before mapping the synthesis results onto a silicon. Many tools exist for RTL design, but few for high-level synthesis. Instead of applying boolean algebra, this paper adopts the Petri Net (PN) theory to verify the correctness of the synthesis result, because the Petri Net model has the nature of dataflow algorithms. Herein, we propose two approaches to realize the PN-based formal verification algorithm and conclude the best one who outperforms the others in terms of processing speed and resource usage.

Key-Words: - Formal verification; high-level synthesis; Petri Net; dataflow graph

1 Introduction

With increasing design complexity of digital signal processing system, verification becomes a more and more important within the design flow. In modern circuits, it is observed that up to 80% of the overall design costs are due to verification. Formal verification techniques which ensure 100% coverage of function and system model correctness have gained large attention. In [1,2], authors give excellent survey of major trends of formal verification techniques which can be classified into two categories, equivalence checking [2] and model checking [3]. Equivalence checking is used to proof the functional equivalence of two design representations modeled at the same or different levels of abstraction. Model checking is a process that checks the correctness of a design model with given properties. Although formal verification for logic synthesis has been studied very extensively, little work has been done for high-level synthesis.

This paper presents a novel verification algorithm to verify high-level synthesis (HLS) of dataflow algorithms. Given a dataflow graph (DFG) and architectural constraints, the HLS aims to generate the task schedule with processor assignment. Typically, the HLS performs algorithmic transformation, such as retiming, scaling, and unfolding, on the DFG to meet the architectural constraints, and allocates resources accordingly [4,5,6,7,8]. Both algorithmic transformation and resource allocation require complex procedures. These procedures are rather heuristic and error-prone. The integer linear programming (ILP), for instance, is one of the popular techniques applied for HLS. The success of ILP is relied on the completeness of clauses. Any mistake or incomplete description made in the ILP may result in an illegal solution and screw up following synthesis results. This paper intends to present a formal verification algorithm to unveil the faults produced in HLS.

In the proposed algorithm, we employed the Petri-Net model as the formal description to check the correctness of dataflow behavior. Petri Net model has the nature of dataflow computing, and hence a good tool for the verification of algorithmic transformations and datapath scheduling. The use of the Petri-Net is two-folded. First, the Petri-Net model of dataflow algorithm can hold the data dependence and hence any legal transformation has to conform to the firing rules of the Petri-Net model. Secondly, the scheduling candidate is correct if and only if the initiation of each task is allowed in the Petri-Net model. Comparing with the traditional model checking techniques, the first use can provide simple but thorough model for restructured algorithms while the second use can avoid false negative problems.

The inputs to the proposed formal verification are the system description and task schedule. The system description is basically a fully-specified flow graph (FSFG) [9]. The FSFG represents the behavioral specification of the dataflow algorithm which is also a design entry of HLS. In HLS, to meet the architectural constraints, the algorithmic transformation normally reconstructs the initial FSFG to find out optimal scheduling results. The reconstructed FSFG is admissible if and only if it is equivalent to the initial FSFG. To verify the correctness of the task schedule, the proposed algorithm first converts the initial FSFG to a Petri-Net model which is expressed by Petri-Net characteristic matrix, because any admissible reconstructed FSFG has to have the same characteristic matrix.

Another input is the schedule, the DUV (design under verification), generated by HLS. The schedule is expressed in the format of processor-time chart (or chart). The chart equally shows the firing sequence. The proposed verification uses the firing sequence to unveil the legal algorithmic transformations applied for the original FSFG. The legal algorithmic transformations will then be candidates to trace the firing sequence of the given schedule.

Based on the inputs, the proposed verification first extracts the initial firing pattern and uses it to determine the candidate reconstructed FSFGs. The candidates will then be verified with the Petri-Net model. If there exist at least one candidate who can allow the firing sequence to execute legally (without against the firing rules), the HLS result is claimed as a correct solution; otherwise, the verification will show the counter example in proof of the incorrectness. In this paper, we propose two approaches to realize the PN-based formal verification and conclude the best one who outperforms the others in terms of processing speed and resource usage.

The remainder of this paper is organized as follows. Section 2 describes some useful definition and proposed modeling technique. The proposed high-level verification technique and verification algorithms are presented in section 3. In section 4, we discuss the complexity analysis of two verification algorithms. In section 5 some experimental results are given. Section 6 gives the conclusions of this paper.

2 Definition and Modeling

In this section, we will discuss some useful definitions and proposed transformation technique to transform a FSFG into PN model.

2.1 Fully-Specified Signal Flow Graph

Fully-Specified Signal Flow Graph (FSFG) [9] or DFG is a natural paradigm for describing DSP algorithms. A FSFG $G_{FSFG}(V,E,D)$, where $V=\{v_1,\dots,v_n\}$ and $E=\{e_1,\dots,e_m\}$, is a three-tuple directed and edge-weighted graph which contains a vertex set V, a directed edge set E, and an ideal delay set D. Vertex set V represents atomic operation of functional units. A vertex may have a zero execution delay, such as the signal duplicator, or may be assumed to take non-zero unit time, such as adder or multiplier. Directed edge set E describes the direction of flow of data between functional units. Inter data dependencies between functional units are denoted by weighted edges. Figure 1, for instance, shows a third-order IIR filter in the form of FSFG.

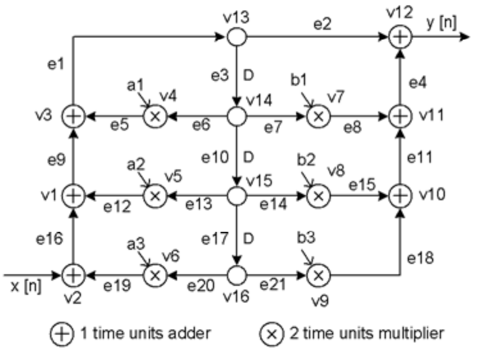


Fig. 1. A third-order IIR filter in the form of FSFG.

2.2 Petri Net Model

A Petri Net $G_{PN}(P,T,W,M)$ is a four-tuple [10], where $P=\{p_1,\dots,p_n\}$ and $T=\{t_1,\dots,t_m\}$ are finite sets of place and transition, W is the weighted flow relation, and M_0 is the initial marking. A marking is a function $M{:}P\rightarrow Z$. If $M(p_i)=k$ for place p_i, we will say that p_i is marked with k tokens. If $W(u,v)>0$, then there is an arc from u to v with weight $W(u,v)$. Usually, matrix representation gives a complete characterization of Petri Net. The characteristic matrix of PN is defined by incidence matrix A, which is a $|P|\times|T|$-matrix with entries

$$A_{ij}=W\left(tr_j,p_i\right)-W\left(p_i,tr_j\right) \tag{2}$$

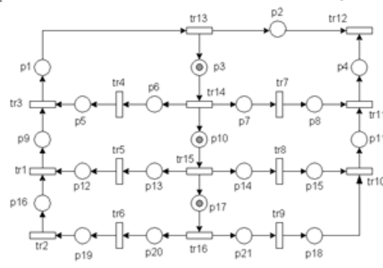
Marking m_0 is an $|P|\times 1$ column vector with entities $m_0(i)=M(p_i)$, $\forall p_i \in P$. We say that is a *valid marking* if and only if $m_0(p_i)\geq 0$. Let $x_j=\{tr_j\}=(\dots,0,1,0,\dots)$ be the unit $|T|\times 1$ column vector, which is zero everywhere except in the j-th element. Transition t_j can be represented by the column vector x_j. We say that t_j is enabled at a marking m_0 if $m_0\geq A\cdot x_i$ for every element of m_0 is a non-negative integer. And the result m' of firing *enabled* transition t_j in a marking m_0 is represented by

$$m'=m_0+A\cdot x_j \tag{3}$$

2.3 Transformation from FSFG to PN

The FSFG is attractive to algorithm developers because it directly models the equations of DSP algorithm. Yet, it does not sufficiently unveil the dynamical behaviour and the implementation limits in terms of the degree of parallelism and the memory requirement. Thus, we use Petri Net to model DSP algorithms. It also allows us to discover the characteristic of the target architecture and to observe the dynamical behaviour of the algorithm.

The FSFG $G_{FSFG}(V,E,D)$ of a DSP algorithm can be modelled as PN $G_{PN}(P,T,W,M_0)$ by applying following rules. First, functional element set V and edge E of FSFG can be transformed into the transition set T and the place P with respect. Since, each place in PN has only one output, the pseudo transition of each fork edge will be added as source duplicators. At last, the delay element set D of edge in FSFG is corresponded to the number of tokens of place in PN. By applying above transition rules, an example in Figure 2 shows the PN model of the third-order IIR filter in Figure 1. The characteristic matrix A with the initial marking m shows the matrix representation of the PN model, for instance.



$$[A] = \begin{array}{c|cccccccccccccccc}
P/T & tr1 & tr2 & tr3 & tr4 & tr5 & tr6 & tr7 & tr8 & tr9 & tr10 & tr11 & tr12 & tr13 & tr14 & tr15 & tr16 \\
\hline
p1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\
p2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\
p3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\
p4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
p5 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
p6 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
p7 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
p8 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
p9 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
p10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\
p11 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\
p12 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
p13 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
p14 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
p15 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
p16 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
p17 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\
p18 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
p19 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
p20 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
p21 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}
\quad m = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Fig. 2. A PN graph and the matrix representation to the third-order IIR filter of Figure 1.

2.4 Schedules to the FSFG

In HLS, a FSFG design may contain cycles to model a DSP application with loops. The intra-iteration precedence relation is represented by the edge without delay and the inter-iteration precedence relation is represented by the edge with delays. Given an edge $e(v_i,v_j)\in E$ in FSFG design, $d(e)$ means the data used as inputs in node v_j are generated by node v_i at $d(e)$ inter-iteration before. A *static* schedule of a cycle FSFG is a repeated pattern of an execution of the corresponding loop. And a static schedule must obey the precedence relations of the directed acyclic graph (DAG) portion of a FSFG design that is obtained by removing all edges with delays from that FSFG.

Let d_j^i be the execution delay for each task node op_j^i, the length $le(S)$ of a schedule S is the latest finish time of all the operations scheduled, that is $le(S)=max\{\varphi(op_j^i)+d_j^i-1 \mid \forall\ op_j^i\in V\}$. For each task node $op_j^i\in V$, a schedule of the FSFG design is given as following:

- Start time: $t_j^i=\varphi\left(op_j^i\right), \varphi:V\to Z^+=\{1,2,\ldots\}$
- Execution delay: $d_j^i\in Z=\{0,1,2,\ldots\}$
- Finish time: $\varepsilon_j^i=\varphi\left(op_j^i\right)+d_j^i-1$
- Task assignment: $pe_j^i=\tau(op_j^i), \tau:V\to\{1,2,\ldots,n_{res}\}$
- Length of the schedule:
  $le(S)=\max\left\{\varphi\left(op_j^i\right)+d_j^i-1 \mid \forall op_j^i\in V\right\}$
- The earliest task-finished step:
  $t_{eff}=\min\left\{\varphi\left(op_j^i\right)+d_j^i-1 \mid \forall op_j^i\in V\right\}$

3 High-Level Verification

In this section, proposed two-stage verification technique is introduced. The algorithms to both stages are also presented separately as the implementations.

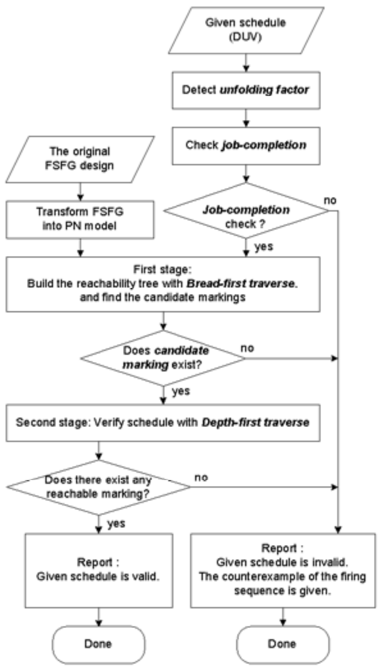
3.1 Verification Flow

A flowchart illustrating our verification flow is shown in Figure 3. There are two inputs to the flow: a given schedule and the original FSFG. The given schedule is the DUV (design under verification) that needs to be verified. The original FSFG reserves the characteristics of the system that the DUV must be satisfied. The proposed verification method tries to find the correct restructured FSFG, which is candidate to the DUV at the first stage, and then, it checks whether the execution sequence, the DUV, of the PN model corresponded to the candidate is satisfied at the second stage. Before introducing two-stage verification method, we address the preprocessing on both inputs separately.

One of the inputs is the given schedule. In system-level design flow, designers may use unfolding algorithm to pursue perfect FSFG achieving iteration period bound on their original FSFG design. Usually, the FSFG of the DSP algorithm describes one iteration of the computation. By applying unfolding algorithm on the FSFG is to unfold the original FSFG by a factor f which implies f consecutive iterations of the design. In contrast, we perform *unfolding checking* in our verification flow to detect the *unfolding factor* f

from given schedule. Another input to the verification flow is the original FSFG graph. It is transformed into a PN model by proposed transformation rules.

In PN domain, the markings, which can be reached from the initial marking, can be seen as the retimed FSFGs of the original design. Some reachable markings are the correct restructured FSFGs for the given schedule. These markings, which dominate the correctness of the given schedule, are said to be the *candidate markings*. In order to find the candidate markings, *Breadth-First algorithm* is used to traverse all the markings of PN reachability tree at the first stage. If the candidate marking does not exist, it means the correct retimed FSFG does not exist, it reports the given schedule is not valid due to absent of the candidate marking. If the candidate marking exists, we continuously apply *Depth-First algorithm* on each candidate marking.



Given schedule (DUV)
Detect unfolding factor
Check job-completion
The original FSFG design
Job-completion check ?
no
yes
Transform FSFG into PN model
First stage: Build the reachability tree with Bread-first traverse. and find the candidate markings
Does candidate marking exist?
no
yes
Second stage: Verify schedule with Depth-first traverse
Does there exist any reachable marking?
no
yes
Report : Given schedule is valid.
Report : Given schedule is invalid. The counterexample of the firing sequence is given.
Done
Done

Fig. 3. Flowchart for the proposed high-level verification method.

The given schedule is valid if there exists an initial marking, the candidate marking, of the PN model leading a firing sequence of the schedule valid. At the second state, we apply Depth-First traverse procedure on each candidate marking to check whether the given schedule is valid by checking the firing sequence of the schedule. At last, if it exists such candidate marking, the flow is done and reports given schedule is valid, or a counterexample of invalid firing sequence is reported if given schedule is invalid.

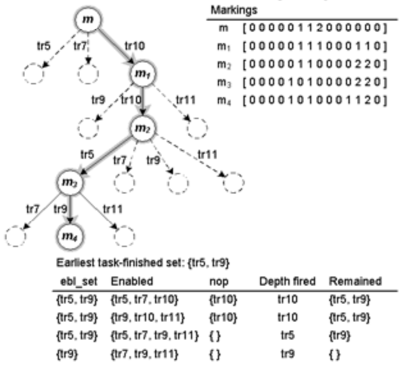
3.2 The Candidate Marking

The candidate marking set is a subset of the reachable marking set of a Petri Net. A candidate marking is probably the correct initial marking, it also means correct retimed FSFG, which leads the firing sequence of a given schedule being valid. Let S be a schedule of a FSFG. The earliest task-finished set *etf_set* of S are the tasks which are finished at the earliest task-finished step t_{etf} in S, such that

$$etf\_set = \left\{ op_j^i \mid \varepsilon_j^i = t_{etf}, \forall op_j^i \in V \right\}. \qquad (4)$$

Marking m is defined as a ***candidate marking***, if marking m and firing sequence σ satisfies Definition 1.

Definition 1 Marking m is said to be a ***candidate marking*** if and only if there exists a firing sequence σ:$tr_1 \ldots tr_k$, such that for all tasks $op \in etf\_set$ are covered by all the transitions in σ, i.e. $etf\_set \subseteq \sigma$. And it is also satisfied that each firing transition $tr_j \in \sigma$ is either a pseudo transition, d_j=0, or an earliest task-finished transition, $\varepsilon_j = t_{etf}$.



Markings	
m	[0 0 0 0 0 1 1 2 0 0 0 0 0 0]
m_1	[0 0 0 0 0 1 1 1 0 0 0 1 1 0]
m_2	[0 0 0 0 0 1 1 0 0 0 0 2 2 0]
m_3	[0 0 0 0 1 0 1 0 0 0 0 2 2 0]
m_4	[0 0 0 0 1 0 1 0 0 0 1 1 2 0]

Earliest task-finished set: {tr5, tr9}

ebl_set	Enabled	nop	Depth fired	Remained
{tr5, tr9}	{tr5, tr7, tr10}	{tr10}	tr10	{tr5, tr9}
{tr5, tr9}	{tr9, tr10, tr11}	{tr10}	tr10	{tr5, tr9}
{tr5, tr9}	{tr5, tr7, tr9, tr11}	{ }	tr5	{tr9}
{tr9}	{tr7, tr9, tr11}	{ }	tr9	{ }

Fig. 4. Check whether a marking is a candidate marking.

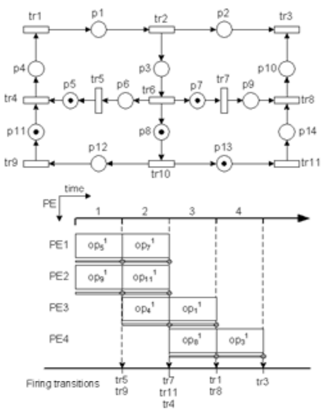


Fig. 5. An example schedule and 2nd order IIR filter.

As an example, Figure 4 shows the procedure to check whether a marking is candidate. For a given schedule in Figure 5, the earliest task-finished step is t_{etf}=1, and the earliest task-finished set is $etf\_set=\{op_j^i | \varepsilon_j^i = t_{etf} = 1\} = \{v_5, v_9\}$. Marking m = [0 0 0 0 0 1 1 2 0 0 0 0 0 0] is said to be a ***candidate***

marking of the corresponded PN model. Since, there exists a firing sequence σ:tr_{10} tr_{10} tr_5 tr_9, $m \xrightarrow{\sigma} m_4$, such that $etf\_set \subseteq \sigma$. The markings, $m_1 \ldots m_k$, of the firing sequence, $m \xrightarrow{tr_{10}} m_1 \xrightarrow{tr_{10}} m_2 \xrightarrow{tr_5} m_3 \xrightarrow{tr_9} m_4$, are valid states.

3.3 Marking sets

The proposed high-level verification method includes two stages: the Breadth-First and the Depth-First traverse procedures. At the first stage, the Breadth-First traverse procedure tries to find candidate markings, the correct retimed FSFGs, from reachability tree. At the second stage, the Depth-First traverse procedure verifies given schedule by checking the candidate markings. Since, the nodes of reachability tree are exponential growth with the height of the tree, two-stage method is the better policy. The verification method shortens the searching space by finding candidate markings at the first stage. At the second stage, it verifies given schedule by checking candidate markings rather than all the reachable markings of reachability tree.

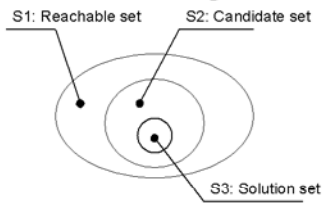


Fig. 6. The relation between reachable, candidate and solution marking sets.

Assuming there are *n* operations in a given FSFG, and hence there are *n* transitions in the corresponded PN model. Let *f* be the *unfolding factor* of a given schedule while designers performing *unfolding* technique on their FSFG design. At the first stage, the procedure tries to find the candidate marking set from the reachable marking set from the reachability tree and fires each transition once each time. The height of each marking in reachability tree is the distance from the root node to itself. Since, during one iteration period of the schedule *S*, *le*(*S*), each scheduled task must be fired once, the height can also be seen as the number of transitions that have been fired since the root node. Thus, for an *n*-tasks schedule, the *upper height-bound* of the reachability tree is bounded by $H_{up}=f{\cdot}n$. At the second stage, it continually finds the solution marking set from the candidate marking set. The set relation between three marking sets is shown in Figure 6, that is $S3 \subseteq S2 \subseteq S1$. The purpose of the first stage is trying to reduce the searching space from reachable marking set *S1* to candidate marking set *S2*, while the second stage is trying to find solution marking set *S3* from candidate marking set *S2*.

3.4 First stage: Breadth-First Traverse

At the first stage of the verification method, we apply Breadth-First traverse procedure to find the candidate markings from the reachability tree. Two approaches, which include the early-terminated and the optimal approaches, are proposed in this paper and discussed in the following sections.

3.4.1 The early-terminated approach

The second approach to verify a schedule of a given FSFG is called the early-terminated approach which improves the exhaustive approach. Before introducing the improved approach, we first consider Lemma 1.

Lemma 1 Let T_{tree} be a reachability tree which is bounded by upper height-bound H_{up} and m_1 be any one of the candidate markings in T_{tree}. For any other candidate marking m_2 in the successor path of marking m_1, m_2 is in the solution marking set *S3* if and only if m_1 is in *S3*.

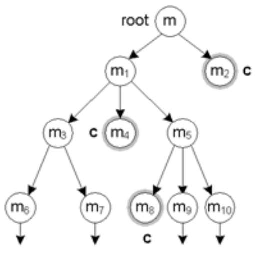


Fig.8. The traverse order of early-terminated approach.

The early-terminated approach uses Lemma 1. It tries to minimize the size of candidate set *S2* from reachable set *S1*. The difference between the exhaustive and the early-terminated approaches is that when an enqueued unvisited marking is candidate, the early-terminated approach ignores the candidate marking and marks as a visited node. Then, it proceeds other unvisited nodes in queue *Q* until all the markings have been visited. In Figure 8, as an example, the traverse order of the early-terminated approach is m, m_1, m_2, m_3, …, m_{10}.

3.4.2 The optimal approach

The second approach to verify a schedule of a given FSFG is the optimal approach which is improved from the early-terminated approach. In order to reduce reachable marking set *S1* of the reachability tree, it tries to merge the redundant nodes when it proceeds Breadth-First traverse.

Let m be an unvisited node to be processed. If m is a candidate marking, it ignores this node by using Lemma 1 and proceeds other unvisited nodes in queue. If m is not a candidate marking, it applies *negative test* to find enabled set of transitions and creates new node on each enabled transition. For each new produced node with marking m', if there exists another node in the reachability tree, and has the same marking associated with it, then the node with marking m' is a duplicate node. Since, the marking m' has appeared in the tree, this new produced node is redundant. Then, it merges this redundant node to the existential node and creates transition link from marking m to the existential node. As an example in Figure 9, when it proceeds marking m_5, it founds the new created node with marking m_7 is a duplicate node. It merges these nodes and creates transition from m_5 to m_7. Then, it continually proceeds other unvisited nodes in queue.

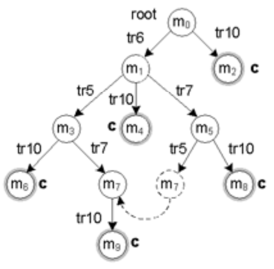


Fig. 9. Merge the redundant node in optimal traverse approach.

3.5 Second stage: Depth-First traverse method

At the second stage, we apply Depth-First traverse procedure to verify a schedule on candidate markings rather than all reachable markings in PN model. As showing in Figure 10, a candidate marking ***m*** which is found in the first stage is probably the correct marking, the correct retimed FSFG that leads a given schedule being valid. For a given schedule in Figure 5, task tr_5 and task tr_9 are scheduled and finished at the first step of the schedule. The procedure tries to fire one transition of these scheduled tasks or enabled *nop* operations once each time during the first scheduled step. At the end of the first step, marking m_1 is obtained from candidate marking m by firing transition sequence σ:tr_6 tr_5 tr_{10} tr_9, that is $m \overset{\sigma}{\rightarrow} m_1$, where transition tr_6 and tr_{10} are *nop* operations. The procedure continually traverses entire length of the schedule step-by-step until all the scheduled tasks are fired. A given schedule is said to be valid if and only if all the markings in the traverse path are valid.

4 The Complexity Analysis

Assuming there are n non-nop operations in a given FSFG. Let f be the *unfolding factor* of a given schedule. As described in previous section, the upper-height of the reachability tree of the corresponded PN model is bounded by $H_{up} = f \times n$. The complexity analysis of the proposed two-stage verification method is discussed as following.

At the first stage, two approaches are proposed including the early-terminated and the optimal traverse methods. Considering each node in the reachability tree has n enabled transitions in worse case, the level 0 (the root node) has one node.

Level 1 has n nodes

Level 2 has $(n)(n)=n^2$ nodes

……

Level $f{\times}n$ has $(n^{f \cdot n-1})(n)=n^{f \cdot n}$ nodes

The total number of nodes is:

$$1+n+n^2+\ldots+n^{f\cdot n}=\left(n^{f\cdot n+1}-1\right)/\left(n-1\right) \qquad (5)$$

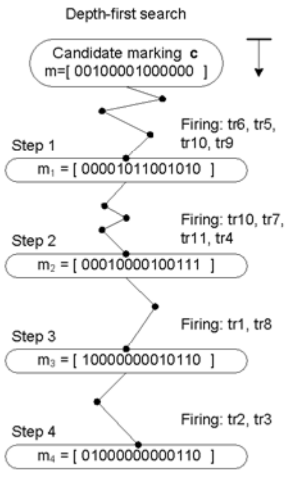


Fig. 10. Verify schedule with Depth-First search algorithm

In the first approach, the early-terminated approach, the algorithm stops traversing a node while it is candidate. Let p , $p \le (f \cdot n)$, be the deepest level that Breadth-First traverse procedure can reach. The complexity of the second approach is $O(N^p), p \le f \cdot n$.

In the second approach, the optimal approach, the algorithm merges duplicate markings in order to reduce the reachable marking set of the reachability tree. Let $x \in Z = \{1, 2, \cdots\}$ be the merging radio in the reachability tree. The complexity of the three approach is $O((N/x)^p), p \le f \cdot n$. Thus, the relation of the complexity between two approaches is:

$$O\left(N^p\right) > O\left(\left(N/x\right)^p\right). \tag{6}$$

At the second stage, the algorithm performs Depth-First traverse to verify a given schedule by checking the firing sequence, which contains $f \cdot n$ transitions , of the PN model. Thus, the complexity is $O(f \cdot n)$, in worse case.

5 Experimental Results

We have implemented these three approaches as the proposed formal verification algorithms. Each of these approaches is applied to several dataflow algorithms. Figure 11 shows the statistics of these designs.

Design name	#vertices	#edges	#init. delays	Size of PN (Places x Trans.)	Schedule length	Unfolding factor
iir2d-sch1	8	14	2	(14 x 11)	6	1
iir2d-sch2	8	14	2	(14 x 11)	4	1
iir2d-sch3	8	14	2	(14 x 11)	4	1
iir2d-sch4	8	14	2	(14 x 11)	4	1
iir3d-sch1	12	21	3	(21 x 16)	6	1
iir3d-sch2	12	21	3	(21 x 16)	6	1
p243-sch1	5	7	5	(7 x 5)	96	6
p243-sch2	5	7	5	(7 x 5)	96	6
ewf-sch1	34	47	0	(47 x 34)	40	1
ewf-sch2	34	47	0	(47 x 34)	40	1

Fig. 11. The statistics of test designs

Test schedule	Early-terminated		Optimal	
	Time (sec)	Res. usage	Time (sec)	Res. usage
Iir2d-sch1	0.17	16	0.19	16
Iir2d-sch2	0.2	14	0.2	14
Iir2d-sch3	24.80	34084	0.33	244
Iir2d-sch4	19.845	35720	0.32	293
Iir3d-sch1	0.19	19	0.21	19
Iir3d-sch2	0.18	19	0.21	19
p243-sch1	0.2	32	0.21	32
p243-sch2	0.22	32	0.21	32
ewf-sch1	0.26	36	0.28	36
ewf-sch2	0.3	36	0.28	36

Fig. 12. The experimental results

Design *iir2d-sch1* to *iir2d-sch4* and design *iir3d-sch1* to *iir3d-sch1* [9] are the second-order and the third-order Infinite Impulse Response filters. Design *p243* [9] is a design with *unfolding factor* 6, the lengths of schedule *p243-sch1* and *p243-sch2* are both 96 steps. Design *ewf-sch1* and *ewf-sch2* are low power schedules for the Elliptic Wave Filter in [5]. Figure 12 shows the experimental results of using three approaches. The optimal approach outperforms the others in terms of time and resource usage.

6 Conclusion

This paper aims to exploit formal verification techniques for high-level synthesis. In the top-down design flow, design errors should be removed as early as possible; otherwise, errors detected at the later stages will result a costly, time-consuming redesign cycles. Although formal verification for logic synthesis has been studied very extensively, little work has been done for high-level synthesis. The paper presents a novel verification flow that can efficiently detect the design errors from the results of high-level synthesis. As shown in the experimental results, we can apply the optimal approach for the first phase to efficiently verify complex design cases.

7 Acknowledgement

This work was supported by the National Science Council, R.O.C., under the grant number NSC 94-2220-E-009-039.

References:

[1] A. Gupta, "Formal hardware verification methods: a survey," *Formal Methods in System Design*, vol. 1, pp. 151–238, 1992.

[2] C. Kern and M. Greenstreet, "Formal verification in hardware design: a survey," *ACM Transactions on Design Automation of E. Systems*, vol. 4, pp. 123–193, Apr. 1999.

[3] E. M. Clarke, O. Grumberg, and D. A. Peled, *Model checking*, The MIT Press, 1999.

[4] V. K. Madisetti and B. A. Curtis, "A uantitative methodology for rapid prototyping and high-level synthesis of signal processing algorithms," *IEEE Transactions on signal processing*, vol. 32, no. 11, pp. 3188–23 208, *Nov.* 1994.

[5] L.-R. Dung and H.-C. Yang, "On multiple-voltage high-level synthesis using algorithmic transformations," *IEICE Transactions on Fundamentals*, 2004.

[6] K. Ito, L. E. Lucke, and K. K. Parhi, "Ilp-based cost-optimal dsp synthesis with module selection and data format conversion," *IEEE Transactions on Very Large Integration Systems*, vol. 6, no. 4, pp. 582–594, Dec. 1998.

[7] K. K. Parhi, "High-level algorithm and architecture transformations for dsp synthesis," *Journal of VLSI signal processing*, vol. 9, pp. 121–143, 1995.

[8] L.-F. Chao and E. H.-M. Sha, "Scheduling data-flow graphs via retiming and unfolding," *IEEE Transactions on parallel and distributed systems*, vol. 8, no. 12, pp. 1259–1267, Dec. 1997.

[9] V. K. Madisetti, *VLSI Digital Signal Processors*, IEEE Press, 1995.

[10] W. Reisig and G. Rozenberg, *Lectures on Petri Nets I: Basic Models* Springer-Verlag, 1998.

[11] K. Parhi and D. Messerschmitt, "Static rate-optimal scheduling of iterative data-flow programs via optimum unfolding," *IEEE Transactions on Computers*, vol. 40, no. 2, pp. 178–195, Feb. 1991.

[12] C. Hwang, J. Lee, and Y. Hsu, "A formal approach to the scheduling problem in high level synthesis," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 10, pp. 464–475, Apr. 1991.

A NAND Flash Memory Controller for SD/MMC Flash Memory Card

Chuan-Sheng Lin(1), *Student Member*, and Lan-Rong Dung(2), *Member, IEEE*

Abstract—**In this paper, a novel NAND Flash Memory Controller was designed. A t-EC w-bit Parallel BCH ECC code was designed for correcting the random bit errors of the flash memory chip, which is suitable for the randomly bit errors property and parallel I/O interface of the NAND type Flash memory. A Code-Banking mechanism was designed for the trade-offs between the controller cost and the ISP (In System Programmability) support. With the ISP functionality and the Flash Parameters programmed in the reserved area of the Flash Memory chip during the card production stage, the function for supporting various kinds of NAND Flash memory could be provided by a single controller. In addition, built-in defect management and wear-leveling algorithm enhanced the product life cycle and reliability. Dual Channel accessing of the Flash memory provided the good performance in data transfer rate. With respect to the proposed controller architecture, a real SD/MMC flash memory card controller chip was designed and implemented with UMC 0.18um CMOS process. Experimental results show the designed circuit can fully comply with the system specifications and shows the good performances.**

Index Terms—**NAND Flash memory controller, BCH ECC Codes, Non-volatile memory, Flash Storage Systems.**

I. Introduction

As a semiconductor memory device capable of nonvolatile data storage even after removing the power supply, NAND flash has gained popularity in a variety of applications, like removable memory cards for portable devices, MP3 players, digital still cameras, and mobile handsets. The emerging multi-media applications demand for higher density and lower cost/MB, and drive the continuous process technology shrink and the MLC (multi level cell) technology adoption. Nevertheless, the narrower line pitch due to process technology shrink may induce severer cross interference between memory cells. No matter the requirement for multiple energy states also leaves MLC technology with a lower margin of error to read the bits. Since the NAND flash is operated through page (e.g. 2 Kbytes) programming and block (e.g. 128 Kbytes) erasing to satisfy the fast data transfer rate for mass data storage, an intrinsic random bit error makes the whole block be marked as "bad" and can no longer be utilized. Once the number of bad blocks exceeds a certain value that the controller chip can manage, the NAND flash chip is declared fail. A NAND Controller is required to handle the bit errors, the bad blocks, maintaining the high data accessing speed, flash memory management, etc. The appropriateness of a NAND controller can enhance the reliability and increase the endurance cycles of the flash memory. In addition, the system performance and product lifetime can also be improved by incorporating an excellent NAND Flash Controller in the NAND Flash Storage Systems.

The Systolic Array architecture has been applied to the regular and iterative VLSI architecture, like RS encoders and decoders, and showed good performance [10]. The systematic design approach of a Systolic Array Processor can make the circuit design easy for implementation and do the pipelining to fit the system level design specifications. In this paper, we presented a *t*-EC *w*-bit Parallel BCH ECC code with incorporating the Systolic Array architecture. The good performances were shown by the real chip realization.

II. The Controller Architecture

The functional block diagram of the NAND Flash Controller was shown in Figure 1. The major functions of the Controller can be divided as: the t-EC w-bit Parallel BCH ECC Circuit, the Code-Banking structure and Firmware In-System Programmable (ISP), the Defect Management and Wear-leveling Algorithm, and the Dual Channel and Multi-Buffering mechanism. The ECC Circuit was designed here to enhance the data integrity and reliability of the data stored in the Flash memory. The Code-banking structure for the Micro-controller complying with firmware ISP function can provide the firmware upgrade to support various kind of Flash memory. The Defect Management Algorithm can increase the yield of the Flash memory and prolong the product life cycle by replacing the defected (Bad) block with the reserved virgin (clean data) blocks. The Wear-leveling Algorithm was also introduced to prolong the product life cycle by preventing the Flash memory blocks from un-balanced usage. The Dual Channel and Multi-Buffering mechanism was designed to increase the data transfer rate at the Flash memory side and to fit the maximum bandwidth of the host-side interface bus.

(1) Chuan-Sheng Lin is with the Prolific Technology, inc., Hsinchu, Taiwan. He is also working toward to Ph.D. degree in the Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan (e-mail: chanson@ntu.edu.tw).

(2) Lan-Rong Dung is with the Department of Electrical and Control Engineering, National Chiao Tung University, Hsinchu, Taiwan, (phone: +886-3-5131567; e-mail: lennon@faculty.nctu.edu.tw). (Lan-Rong Dung provides phone number because he is the corresponding/submitting author.)

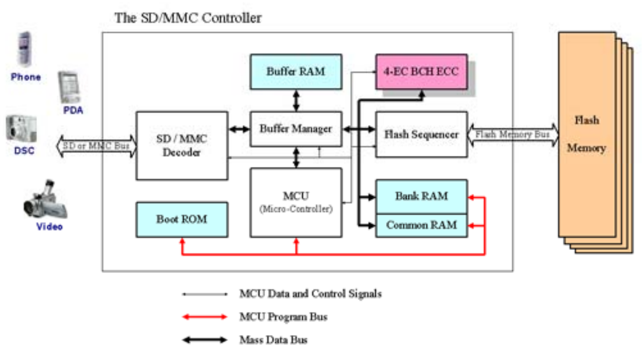


Fig. 1. The Block Diagram of the NAND Flash Controller for SD/MMC Card

A. The t-EC BCH ECC Code Construction

For a typical *t*-EC BCH Code, the Generator Polynomial can be expressed as

$$G(x) = m_1(x)m_3(x)m_5(x)\cdots m_{2t-1}(x) \quad \ldots\ldots\ldots \quad (1)$$

, where $m_j(x),\ j = 1, 3, 5, 2t-1$ are the minimal polynomials in GF(2^m).

By expanding the product of the left side of equation (1), the equation can be expressed as

$$G(x) = \sum_{i=0}^{mt-1} a_i \cdot x^i \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (2)$$

, where a_i 's are coefficients of the Generator Polynomial $G(x)$, $a_i \in GF(2)$. In cyclic operation, $G(x)$ can be implemented by a set of registers and XOR gates. Totally, $m \cdot t$ registers are necessary to fulfill the cyclic operations. The equation (2) can be expressed as a matrix format as:

$$[\text{Reg}]_{i+1} = G \cdot [\text{Reg}]_i + g \cdot D_i \quad \ldots\ldots\ldots\ldots\ldots\ldots \quad (3)$$

The detailed Matrix form can be written as follows,

$$\begin{bmatrix} R_{mt} \\ R_{mt-1} \\ \vdots \\ R_2 \\ R_1 \end{bmatrix}_{i+1} = \begin{bmatrix} a_{mt-1} & 1 & 0 & \cdots & 0 \\ a_{mt-2} & 0 & 1 & & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ a_1 & 0 & 0 & \cdots & 1 \\ a_0 & 0 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} R_{mt} \\ R_{mt-1} \\ \vdots \\ R_2 \\ R_1 \end{bmatrix}_i \oplus \begin{bmatrix} a_{mt-1} \\ a_{mt-2} \\ \vdots \\ a_1 \\ a_0 \end{bmatrix} \cdot D_i \quad (4)$$

In serial connection, the matrix equation can be expressed as in equation (4). Moreover, for a continuously serial data input stream, $\{\ldots, D_i, D_{i+1}, D_{i+2}, \ldots, D_{i+w-1}, \ldots\}$, the *w*-bit parallel formula can be formed as:

$$\begin{aligned} [\text{Reg}]_{i+1} &= G \cdot [\text{Reg}]_i + g \cdot D_i \\ [\text{Reg}]_{i+2} &= G \cdot [\text{Reg}]_{i+1} + g \cdot D_{i+1} \\ &\vdots \\ [\text{Reg}]_{i+w} &= G \cdot [\text{Reg}]_{i+w-1} + g \cdot D_{i+w-1} \end{aligned}$$

By *w*-times functional composition of the above equations, an analytical equation was deducted as in equation (5).

$$[\text{Reg}]_{i+w} = G^w \cdot [\text{Reg}]_i + \sum_{j=0}^{w-1} G^j \cdot g \cdot D_{i+w-j-1} \quad (5)$$

Based on the same concept and operations, the induction of the parallel Syndrome Polynomials is similar to the Generator Polynomials.

From the Matrix operation of the equation (5), a general basic equation for the Systolic Array Processing can be expressed as

$$R_{i+1,j} = a_i \cdot R_{i,mt} + a_i \cdot D_i + R_{i,j-1} \quad \ldots\ldots\ldots. \quad (6)$$

, where the boundary conditions are

$$R_{i+1,mt} = a_i \cdot R_{i,mt} + a_i \cdot D_i + R_{i,mt-1} \quad \ldots. \quad (7)$$

$$R_{i+1,1} = a_i \cdot R_{i,mt} + a_i \cdot D_i \quad \ldots. \quad (8)$$

Based on the general basic equation of a *t*-EC *w*-bit parallel BCH code, the basic operation module for the matrix array can be constructed by two AND gates and two XOR gates. The coefficients a_i's are determined by the Generator Polynomial or the Syndrome Generator Polynomials of the BCH code. To complete the matrix equation (4) by the basic operation module, an Array Architecture is adopted. The Array Architecture of the *n*-bit input data stream for the BCH Generator Polynomial or Syndrome Generator Polynomials is shown as in Figure 2.

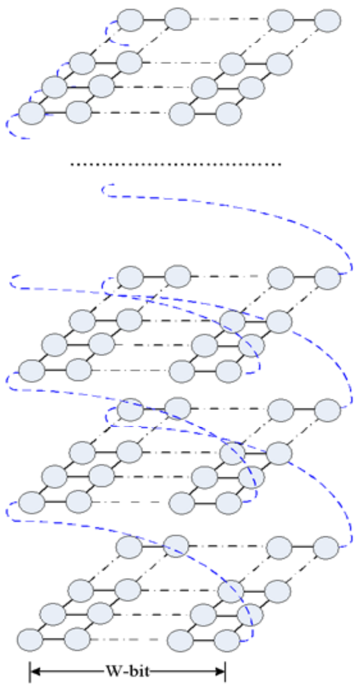


Fig. 2. The *w*-bit Folded Structure for *w*-bit Parallel Operation.

B. *The Code-Banking and various NAND Flash Memory support*

The architecture for the Code-banking was shown in Figure 3. The Boot ROM was the Masked Read Only Memory which stored the Boot codes for the Micro-controller in Booting. By the Code-Banking architecture, the whole system firmware can be separated by several banks, Bank Code #0, 1, …, n. The Banked Codes will be executed Bank by Bank when loaded by the Code Loader to the Bank RAM. The System Firmware of the micro-controller can be upgraded from the Host side.

To support the various kinds of the NAND Flash memory in a controller, a specified Flash Parameters was created to record some system operated parameters of the NAND Flash memory. The Table was started by a Start-Tag, and ended by an End-of-Table Flag. The useful parameters such as: Total Capacity, Total Physical Blocks, Pages per Block, etc. With the Code-Banking architecture and the Specified Flash memory parameters, the various kinds of NAND Flash memory can be supported in the same controller chip and be accessed in an optimal way respectively.

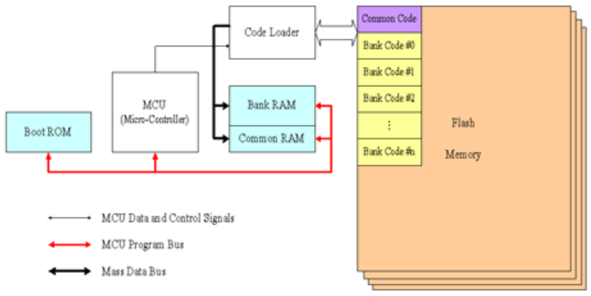


Fig. 3. The Code-Banking Architecture.

III. The Chip Implementation

A. *The Controller Chip Implementation*

The functional block diagram of the designed NAND Flash Controller for SD/MMC memory card was discussed in Section II, Figure 1. With the chip architecture designed, the circuit of the chip was designed and implemented to the UMC 0.18 um CMOS Process. The chip micrograph of the designed NAND Flash controller is shown in Figure 4.

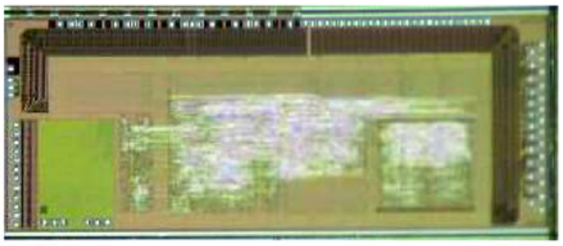


Fig. 4. The micrograph of the controller chip.

B. *The Experiment and Test Results*

The System performance of the designed NAND Controller was evaluated by the certified platform of MMCA (http://www.mmca.org) and SDA (http://www.sdcard.org), the test platform was provided by Testmetrix, Inc. (http://www.testmetrix.com). The Summary of the System

TABLE I
Results of the System Performance

Test Items	Results
1. Sequential Multi-Block Read: MMC 8-bit, 52 MHz	42.33 MB/s
2. Sequential Multi-Block Write: MMC 8-bit, 52 MHz	19.86 MB/s
3. Random Multi-Block Read: MMC 8-bit, 52 MHz	41.03 MB/s
4. Random Multi-Block Write: MMC 8-bit, 52 MHz	12.77 MB/s
5. Suspend Current: 3.3V	120 uA
6. Operating Current at Sustained Data Write: 3.3V	30.95 mA
7. Operating Current at Sustained Data Read: 3.3V	20.23 mA

*The test conditions: 1. Test Platform: Testmetrix for SD/MMC Card; 2. Flash Configuration: 2x Samsung K9F1G08U0M in Dual Channel

Performance of designed Controller is shown in Table I.

IV. Conclusions

A NAND Flash memory controller for SD/MMC memory card was presented. The *t*-EC *w*-bit Parallel BCH ECC by using Systolic Array was presented, and it can be easily applied to a general *t*-EC *w*-bit parallel BCH ECC circuit application. The Code-Banking and ISP capability was presented and discussed for supporting the various kinds of the NAND Flash memory. The real chip realization and the test results by experiment show the good performances of the controller chip.

References

[1] Paolo Cappelletti, Carla Golla, Piero Olivo and Enrico Zanoni, Flash Memories, Kluwer Academic Publishers, 2000.

[2] D.R.Hankerson, D.G.Hoffman, D.A.Leonard, C.C.Lindner, K.T.Phelps, C.A.Rodger, J.R.Wall, Coding Theory and Cryptography, Marcel Dekker Inc., 2000.

[3] Hsie-Chia Chang, Chien-Ching Lin, Tien-Yuan Hsiao, Jieh-Tsorng Wu and Ta-Hui, Wang, "Multi-Level Memory Systems Using Error Control Codes," ISCAS 2004, pp. II 393-396.

[4] Tong-Bi Pei and Charles Zukowski, "High-speed Parallel CRC Circuits in VLSI," IEEE Trans on Communications, Vol. 40, No. 4, Apr. 1992. pp 653-657.

[5] Xinmiao Zhang and K. K. Parhi, "High-Speed Architectures for Parallel Long BCH Encoders," IEEE Trans on Very Large Scale Integration (VLSI) Systems, Vol. 12, No. 5, Jul. 2005. pp 872-877.

[6] K. K. Parhi, "Eliminating the fan-out bottleneck in parallel long BCH encoders," Journal, IEEE Trans on Circuits and Systems, 3(51), pp. 512-516, 2004.

[7] Jun Zhang, Zhi-Gong Wang, Qing-Sheng Hu and Jie Xiao, "Optimized Design for High-speed Parallel BCH Encoder," IEEE Int. Workshop VLSI Design & Video Tech. 2005, pp. 97-100.

[8] Yanni Chen and K. K. Parhi, "Small Area Parallel Chien Search Architectures for Long BCH Codes," IEEE Trans on Very Large Scale Integration (VLSI) Systems, Vol. 12, No. 5, May 2004, pp. 545-549.

[9] Yuejian Wu, "Low Power Decoding of BCH Codes," INCAS 2004, pp. II 369-372.

[10] Keiichi Iwamura, Yasunori Dohi and Hideki Imai, "A Design of Reed-Solomon Decoder with Systolic-Array Structure," Journal, IEEE Trans on Computers, Vol. 44, No. 1. Jan. 1995, pp. 118-122.

System-Level Verification on High-Level Synthesis of Dataflow Graph

Tsung-Hsi Chiang
Department of Electrical and Control Engineering
National Chiao Tung University
Hsinchu 30010, Taiwan
Email: aries.ece89g@nctu.edu.tw

Lan-Rong Dung
Department of Electrical and Control Engineering
National Chiao Tung University
Hsinchu 30010, Taiwan
Email: lennon@faculty.nctu.edu.tw

Abstract— **This paper presents a system-level verification algorithm using the Petri Net theory to detect design errors for high-level synthesis of dataflow graphs. Typically, given a dataflow graph and a set of architectural constraints, the high-level synthesis performs algorithmic transformation and produces the optimal scheduling. How to verify the correctness of high-level synthesis becomes a key issue before mapping the synthesis results onto a silicon. Many tools exist for RTL design, but few for high-level synthesis. Instead of applying Boolean algebra, this paper adopts the Petri Net theory to verify the correctness of the synthesis result. Herein, we propose three approaches to realize the Petri Net based formal verification algorithm and identify the best one that outperforms the others in terms of processing speed and resource usage.**

I. Introduction

This paper presents a novel verification algorithm to verify high-level synthesis (HLS) of dataflow algorithms. Given a dataflow algorithms and architecture constraints, the HLS aims to perform algorithmic transformation and allocation and finally generates the task schedule and processor assignment. It typically requires complex procedures. These procedures are rather heuristic and error-prone. The integer linear programming (ILP), for instance, is one of the popular techniques applied for HLS. The success of ILP relies on the completeness of clauses. Any mistake or incomplete description made in the ILP may result in an illegal solution and affect the correctness of following synthesis results. This paper intends to present a formal verification algorithm to unveil the faults produced in HLS.

In the proposed algorithm, we employ the Petri-Net model as the formal description to check the correctness of dataflow behavior. Petri net model has the nature of dataflow computing, and hence is a good tool for the verification of algorithmic transformations and datapath scheduling. The use of the Petri-Net is two-folded. First, the Petri-Net model of dataflow algorithm can hold the data dependence and hence any legal transformation has to conform to the firing rules of the Petri-Net model. Secondly, the scheduling candidates is correct if and only if the initiation of each task is allowed in the Petri-Net model. Comparing with the traditional model checking techniques, the first use can provide simple but thorough model for restructured algorithms while the second use can avoid false negative problems.

The inputs to the proposed formal verification are the system description and task schedule. The system description is basically a dataflow graph (DFG). Given a DFG, the proposed algorithm converts it to a Petri-Net model which is expressed by Petri-Net characteristic matrix. Later, the matrix will be used to verify primary properties, such as reachability, liveness, safeness, and boundedness. The other input is the schedule generated by HLS. The proposed verification starts from the extraction of firing status and then find out the legal algorithmic transformations applied for the original DFG. The legal algorithmic transformations will then be candidates to trace the firing sequence of the given schedule. If there exist at least one candidate who can allow the firing sequence to execute legally (without against the firing rules), the HLS result is claimed as a correct solution; otherwise, the verification will show the counter example in proof of the incorrectness. In addition, we propose three approaches to realize the Petri Net (PN) based formal verification and conclude the best one who outperforms the others in terms of processing speed and resource usage.

II. Definitions and properties

In the following section, we will discuss some useful properties and proposed transformation technique to transform a FSFG into PN model.

A. Fully-Specified Flow Graph (FSFG)

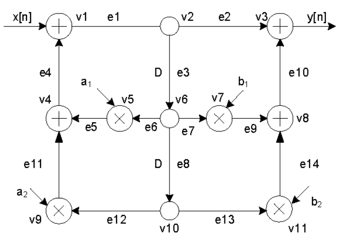


Fig. 1. A second order IIR filter in the form of FSFG

Fully-Specified Flow Graph (FSFG) [1] or DFG is a natural paradigm for describing dataflow algorithms. A FSFG

$G_{FSFG}(V,E,D)$, where $V = \{v_1,\ldots,v_m\}$ and $E = \{e_1,\ldots,e_n\}$, is a three-tuple directed and edge-weighted graph which contains a vertex set V, a directed edge set E, and a ideal delay set D. Vertex set V represents atomic operation of functional units. Directed edge set E describes the direction of flow of data between functional units. Inter data dependencies between functional units are represented by weighted edges. Figure 1, for instance, shows a second order IIR filter in the form of FSFG.

B. Transformation from FSFG to PN model

A Petri net $G_{PN}(P,T,W,M_0)$ is a four-tuple [2], where $P = \{p_1,\ldots,p_n\}$ and $T = \{t_1,\ldots,t_m\}$ are finite sets of place and transition, W is the weighted flow relation, and M_0 is the initial marking. A marking is a function $M : P \to \mathbb{Z}$. If $M(p_i) = k$ for place p_i, we will say that p_i is marked with k tokens. A marking M is said to be a valid state if and only if $M(p_i) \geqslant 0$, $\forall p_i \in P$. If $W(u,v) > 0$, then there is an arc from u to v with weight $W(u,v)$.

The FSFG $G_{FSFG}(V,E,D)$ of a DSP algorithm can be modeled as PN $G_{PN}(P,T,W,M_0)$ by applying the following rules. First, functional element set V and edge set E of FSFG can be transformed into the transition set T and the place set P with respect. Since, each place in PN has only one output, the pseudo transition of each fork edge will be added as source duplicators. At last, the delay element set D of edge in FSFG is corresponded to the number of tokens of place in PN. By applying above transformation rules, an example in Figure 2 shows the PN model of the second order IIR filter in Figure 1. The characteristic matrix A with the initial marking m shows the matrix representation of the PN model, for instance.

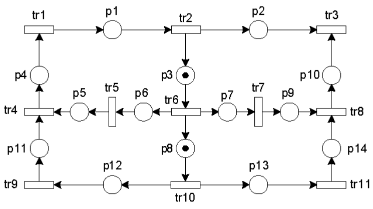


Fig. 2. PN models of second order IIR filter in static and dynamical phase

C. Schedule of a given FSFG

Before mapping an FSFG design into a hardware, the execution start time for each task must be determined. An *iteration* is simply defined as a duration that each task node in the FSFG executes once. Let op_j^i be a task, which is corresponded to each vertex $v_j \in V$ of a given FSFG in the ith iteration. A schedule of the given FSFG, $G_{FSFG} = (V,E,D)$, is a function $\varphi : V \to \mathbb{Z}^+$, which arranges each task node op_j^i to begin its execution at the time step $\varphi(op_j^i)$, where $\mathbb{Z}^+ = \{1,2,\ldots\}$ is the positive integers.

Assuming d_j is the execution delay for each task node op_j^i, the length $le(S)$ of a schedule S is the latest finish time of all the operations scheduled, that is $le(S) = \max\{\varphi(op_j^i) + d_j - 1 | \forall op_j^i \in V\}$. For each task node $op_j^i \in V$, a schedule of the given FSFG is as following:

- Start time: $t_j^i = \varphi(op_j^i)$, $\varphi : V \to \mathbb{Z}^+ = \{1,2,\ldots\}$
- Execution delay: $d_j \in \mathbb{Z} = \{0,1,2,\ldots\}$
- Finish time: $\varepsilon_j = \varphi(op_j^i) + d_j - 1$
- Length of the schedule:
  $le(S) = \max\{\varphi(op_j^i) + d_j - 1 | \forall op_j^i \in V\}$
- The earliest task-finished step:
  $t_{etf} = \min\{\varphi(op_j^i) + d_j - 1 | \forall op_j^i \in V\}$

III. High-Level Verification Flow

A. Verification Flow

The flowchart for the proposed high-level verification is shown in Figure 3. In the beginning, the inputs to the flowchart are the original FSFG and the schedule scheme that needs to be verified. Then, a FSFG is converted into PN model by using proposed transformation. Since designers, may apply unfolding algorithm [1], [3]–[5] on FSFG in their design flow to pursue perfect FSFG achieving *iteration period bound*, we apply unfolding checking to detect the *unfolding factor* from given schedule and FSFG. And then, three check points which determine whether given schedule is valid are proceeded in turn.

First, the start time of each operation in FSFG must be determined in a given schedule. Thus, we perform job completion checking to make sure each task-node in FSFG has be scheduled. A given schedule is valid if and only if there exist an initial marking of the Petri net transformed by the original FSFG that makes the firing sequence of a given schedule valid. In order to find valid solution marking from the Petri net, we build the reachability tree, which contains all reachable markings of the Petri net, by using *Breadth-First* traverse method. Since the nodes of reachability tree are exponential growth with the height of the tree, we choose the candidate markings from all the reachable markings in the reachability tree filtering out the useless markings. If candidate marking exists, *Depth-First* traverse method is used to verify whether given schedule is valid. At last, if there exists reachable markings leading a firing sequence of schedule valid, the flow is done and we report schedule is valid, or a counterexample of invalid firing sequence is reported if given schedule is invalid.

B. The candidate marking

Candidate marking set is defined as a subset of reachable marking set of a Petri net. The candidate markings are probably the correct initial markings that lead the firing sequence of a given schedule being valid.

Let S be a schedule of a FSFG. The earliest task-finished set est_set of S are the tasks which are finished at the earliest task-finished step t_{est} in S, such that

$$est\_set = \{op_i | \varepsilon_i = t_{est},\ \forall op_i \in V\}. \tag{1}$$

Assuming m is a marking of the corresponded PN model. A firing sequence $\sigma : tr_1 \ldots tr_k$ of transitions is denoted by $m \xrightarrow{\sigma} m_k$, $m_1 \ldots m_k$ are valid states, such that

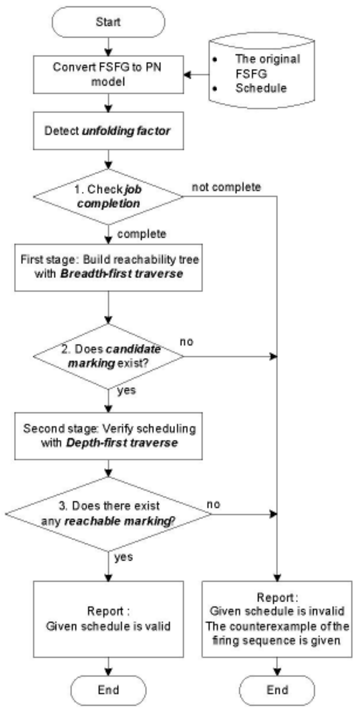

Start
Convert FSFG to PN model
The original FSFG
Schedule
Detect *unfolding factor*
1. Check *job completion*
not complete
complete
First stage: Build reachability tree with *Breadth-first traverse*
2. Does *candidate marking* exist?
no
yes
Second stage: Verify scheduling with *Depth-first traverse*
3. Does there exist any *reachable marking*?
no
yes
Report : Given schedule is valid
Report : Given schedule is invalid The counterexample of the firing sequence is given
End
End

Fig. 3. Flowchart for the proposed high-level verification

$$m \xrightarrow{tr_1} m_1 \xrightarrow{tr_2} \cdots \xrightarrow{tr_k} m_k.$$

C. Proposed Verification Method

The proposed high-level verification method includes two stages: the *Breadth-First* and the *Depth-First* traverse procedures. Since the nodes of reachability tree are exponential growth with the height of the tree, the policy to shorten the searching space is to find candidate markings at the first stage. At the second stage, it verifies given schedule by checking candidate markings rather than all the reachable markings.

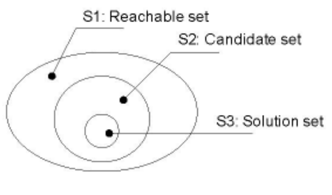


Fig. 4. The relation between reachable, candidate and solution marking sets

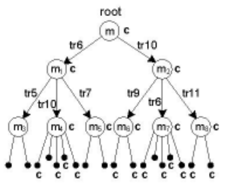
Assuming there are n operations in a given FSFG and n transitions in the corresponded PN model. Let f be the *unfolding factor* of a given schedule. At the first stage, the algorithm tries to find the candidate marking set from the reachable marking set in reachability tree and fires each transition once each time. The height of each node in the reachability tree is the distance from the root node to itself. Since, during one iteration period of the schedule S, $le(S)$, each scheduled task must be fired one time, the height can also be seen as the number of transitions that have been fired from the root node. Thus, for an n-tasks schedule, the *upper height-bound* of the reachability tree is bounded by $H_{up} = f \times n$. At the second stage, it continually finds the solution marking set from the candidate marking set. The set relation between three marking sets is shown in Figure 4, that is $S3 \subseteq S2 \subseteq S1$. The purpose of the first stage is trying to reduce the searching space from reachable marking set $S1$ to candidate marking set $S2$, while the second stage is trying to find solution marking set $S3$ from candidate marking set $S2$.

IV. Three Approaches for Candidate Search

At the first stage, three approaches including the exhaustive, the early-terminated, and the optimal methods are proposed. We will discuss in the following sections.

A. Exhaustive approach

The first approach to verify a schedule of a given FSFG is the exhaustive method. The reachability tree represents the reachable set of a Petri net. In Figure 5, as an example, each node in the tree associated with a marking of the Petri net. The root node of the reachability tree is the initial marking m of a given PN model. In this marking, two transitions are enabled: tr_6 and tr_{10}. For each enabled transition, the algorithm creates a new node in the reachability tree for the reachable markings which result from firing both transitions. An arc, labeled by the transition fired, leads from the initial marking to each of the new nodes. Then it applies candidate checking for each new node to check whether a marking is a candidate. The procedure continually builds the reachability tree for each new produced node until reach the *upper height-bound* H_{up}.



#.	States	Candidate	Branches
m	0 0 1 0 0 0 0 1 0 0 0 0 0 0	C	{tr6, tr10}
m_1	0 0 0 0 0 1 1 2 0 0 0 0 0 0	C	{tr5, tr10, tr7}
m_2	0 0 1 0 0 0 0 0 0 0 0 1 1 0	C	{tr9, tr6, tr11}
m_3	0 0 0 0 1 0 1 2 0 0 0 0 0 0	-	{tr10, tr7}
m_4	0 0 0 0 0 1 1 1 0 0 0 1 1 0	C	{tr9,tr5,tr10,tr11,tr7}
m_5	0 0 0 0 0 1 0 2 1 0 0 0 0 0	C	{tr5, tr10}
m_6	0 0 1 0 0 0 0 0 0 0 1 0 1 0	C	{tr6, tr11}
m_7	0 0 0 0 0 1 1 1 0 0 0 1 1 0	C	{tr9,tr5,tr10,tr11,tr7}
m_8	0 0 1 0 0 0 0 0 0 0 0 1 0 1	C	{tr9, tr6}
	∘ ∘ ∘ ∘		∘ ∘ ∘ ∘

Fig. 5. Build reachability tree with breadth-first search algorithm

B. Early-terminated approach

The second approach is the early-terminated traverse method. We first consider Lemma 1.

Lemma 1: Let T_{tree} be a reachability tree which is bounded by upper height-bound H_{up} and m_1 be any one of the candidate markings in T_{tree}. For any other candidate marking

57

m_2 in the successor path of marking m_1, m_2 is in the solution marking set $S3$ if and only if m_1 is in $S3$.

The early-terminated traverse is based on exhaustive approach and uses Lemma 1. It tries to minimize the size of candidate set $S2$ from reachable set $S1$. In early-terminated approach, when a marking is a candidate, the algorithm ignores this candidate marking. In Figure 6, as an example, the traverse order of early-terminated approach is $m, m_1, m_2, m_3, \ldots, m_{10}$.

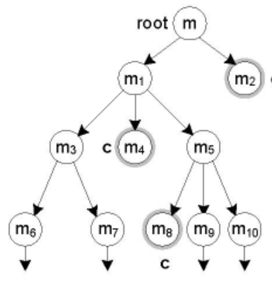


Fig. 6. The traverse order of early-terminated approach

C. Optimal approach

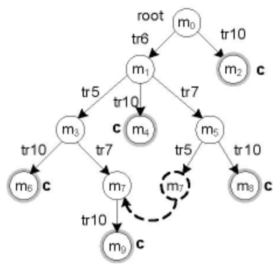


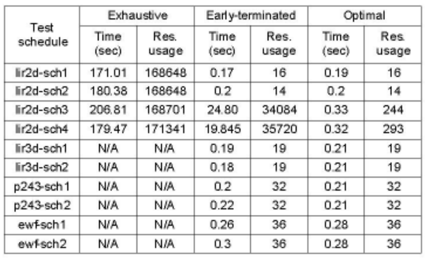
Fig. 7. Merge the redundant node in optimal traverse approach

In order to reduce reachable set $S1$ of the reachability tree, the optimal method tries to merge the redundant nodes when it proceeds *breadth-first* traverse.

Let m be an unvisited node to be processed. If m is a candidate marking, the algorithm ignores this node by using Lemma 1 and proceeds other unvisited nodes in queue. If m is not a candidate marking, the algorithm applies $negative\ test$ to find enabled set of transitions and creates a new node on each enabled transition. For each new produced node with marking $m\prime$, if there exists another node in the reachability tree, and has the same marking associated with it, then the node with marking $m\prime$ is a duplicated node. Since, the marking $m\prime$ has appeared in the tree, this new produced node is redundant. Then, the algorithm merges this redundant node to the existential node and creates a transition link from marking m to the existential node. As an example in Figure 7, when algorithm proceeds marking m_5, it founds the new created node with marking m_7 is a duplicate node. The algorithm merges these nodes and creates a transition from m_5 to m_7. Then, it continually proceeds other unvisited nodes in queue.

V. Experimental Results

We have implemented these three approaches in our study. Each of these approaches is applied to several dataflow algorithms. Figure 8 shows the experimental results of using three approaches. Design *iir2d-sch1* to *iir3d-sch2* are the second and third order Infinite Impulse Response filters. Design *p243* [1] is a design with *unfolding factor* 6. Design *ewf-sch1* and *ewf-sch2* are low power schedules for the Elliptic Wave Filter in [6]. There are two columns on each approach including execution time in seconds and the resource usage in unit number of allocated nodes of the reachability tree. According to experimental results, exhaustive approach suffers the state explosion problem while it traverses the reachability tree. In *iir3d*, *p243* and *ewf* cases, optimal approach takes more benefit than early-terminated approach. According to experimental results, the optimal approach outperforms the others in terms of time and resource usage in average.



Test schedule	Exhaustive		Early-terminated		Optimal	
	Time (sec)	Res. usage	Time (sec)	Res. usage	Time (sec)	Res. usage
Iir2d-sch1	171.01	168648	0.17	16	0.19	16
Iir2d-sch2	180.38	168648	0.2	14	0.2	14
Iir2d-sch3	206.81	168701	24.80	34084	0.33	244
Iir2d-sch4	179.47	171341	19.845	35720	0.32	293
Iir3d-sch1	N/A	N/A	0.19	19	0.21	19
Iir3d-sch2	N/A	N/A	0.18	19	0.21	19
p243-sch1	N/A	N/A	0.2	32	0.21	32
p243-sch2	N/A	N/A	0.22	32	0.21	32
ewf-sch1	N/A	N/A	0.26	36	0.28	36
ewf-sch2	N/A	N/A	0.3	36	0.28	36

Fig. 8. The experimental results

VI. Conclusion

This paper aims to exploit formal verification techniques for high-level synthesis. In the top-down design flow, design errors should be removed as early as possible; otherwise, errors detected at the later stages will result a costly, time-consuming redesign cycles. Although formal verification for logic synthesis has been studied very extensively, little work has been done for high-level synthesis. As shown in the experimental results, we can apply the optimal approach for the first phase to efficiently verify complex design cases.

References

[1] V. K. Madisetti, *VLSI Digital Signal Processors*. IEEE Press, 1995.
[2] W. Reisig and G. Rozenberg, *Lectures on Petri nets I: Basic Models*. Springer-Verlag, 1998.
[3] K. Parhi and D. Messerschmitt, "Static rate-optimal scheduling of iterative data-flow programs via optimum unfolding," *IEEE Transactions on Computers*, vol. 40, no. 2, pp. 178–195, Feb. 1991.
[4] L.-F. Chao and E. H.-M. Sha, "Scheduling data-flow graphs via retiming and unfolding," *IEEE Transactions on parallel and distributed systems*, vol. 8, no. 12, pp. 1259–1267, Dec. 1997.
[5] K. Parhi, "Algorithm transformations for concurrent processors," *In Proceedings of the IEEE*, vol. 77, no. 12, pp. 1879–1895, Dec. 1989.
[6] L.-R. Dung and H.-C. Yang, "On multiple-voltage high-level synthesis using algorithmic transformations," *IEICE Transactions on Fundamentals*, 2004.

參考文獻

[1] S.-T. Cheng, C.-M. Chen, J.-W. Hwang, “Low-Power Design for Real-Time Systems,” Real-Time Systems, Vol.15, pp.131-148, Kluwer Academic Publishers, 1998.

[2] K. Danckaert, F. Catthoor, H. De Man, “System Level Memory Optimization for Hardware-Software Co-Design,” Proceedi9ngs of the 5th International Workshop on Hardware/Software Co-Design, 1997, pp. 55-64.

[3] L. Benini, A. Macii, E. Macii, M. Poncino, “Selective Instruction Compression for Memory Energy Reduction in Embedded Systems,” International Symposium on Low-Power Electronics and Design, Aug. 1999, pp. 206-211.

[4] L. H. Lee, B. Moyer, J. Arends, “Instruction Fetch Energy Reduction Using Loop Caches for Embedded Applications with Small Tight Loops,” International Symposium on Low-Power Electronics and Design, Aug. 1999, pp. 267-269.

[5] F. Yao, A. Demers, S. Shenker, “A Scheduling Model for Reduced CPU Energy,” IEEE Annual Foundations of Computer Science, pp. 374-382, 1995.

[6] D. Kirovski, M. Potkonjak, “System-Level Synthesis of Low-Power Hard Real-Time Systems,” Design Automation Conference, 1997, pp. 697-702.

［7］S. Lee, T. Sakurai, “Run-Time Voltage Hopping for Low-Power Real-Time Systems,” DAC’00, June, 2000, pp. 806-809.

［8］A. Manzak, C. Chakrabarti, “Variable Voltage Task Scheduling for Minimizing Energy or Minimizing Power,” ISLPED’01, 2001, pp. 279-282.

[9] J. Luo, N. K. Jha, "Battery-Aware Static Sheduling for Distributed Real-Time Embedded Systems," DAC’01, 2001, pp. 444-449.

[10] J. Liu, P. H. Chou, N. Bagherzadeh, F. Kurdahi, “Power-Aware Task Motion for Enhancing Dynamic Range of Embedded Systems with Renewable Energy Sources,” Workshop on Power-Aware Computer Systems, PACS’02

[11] “Mobile Pentium III Processor in BGA2 and Micro-PGA2 Packages Datasheet,” Intel Corporation, p55.

[12] K. M. Yang, M T. Sun, and L. Wu, “A family of VLSI designs for the motion compensation block-matching algorithm,” *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 36, no. 10, pp. 1317-1325, Oct. 1989.

[13] Yeong-Kang Lai, and Liang-Gee Chen, “A data-interlacing architecture

with two-dimensional data-reuse for full-search block-matching algorithm," *IEEE Trans. Circuits syst. Video Technol.*, Vol. 8, no. 2, pp. 124-127, Apr. 1998.

[14] Chaur-Heh Hsieh, and Ting-Pang Lin, "VLSI Architecture for Block-Matching Motion Estimation Algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 2, no. 2, pp. 169-175, Jun 1992.

[15] Jen-Chieh Tuan, Tian-Sheuan Chang, and Chein-Wei Jen, " On the Data Reuse and Memory Bandwidth Analysis for Full-Search Block-Matching VLSI Architecture," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 12, no. 1, pp. 61-72, Jan. 2002.

[16] Mei-Juan Chen, Liang-Gee Chen, and Tzi-Dar Chiueh, "One-dimensional full search motion estimation algorithm for video coding," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 4, no. 5, pp. 504-509, Oct. 1994.

[17] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishigura, "Motion compensated interframe coding for video conferencing," in *Proc. NTC'81*, New Orleans, LA, pp. G5.3.1-G5.3.5, Nov. 1981.

[18] J. R. Jain, and A. K. Jain, "Displacement measurement and its application in interframe image coding," IEEE Trans. Commun. Vol. COM-29, pp. 1799-1808, Dec. 1981.

[19] Renxiang Li, Bing Zeng, and Ming L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 4, no. 4, pp. 438-442, Aug. 1994.

[20] Ken Sauer, and Brian Schwartz, "Efficient block motion estimation using integral projections," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 6, no. 5, pp. 513-518, Oct. 1996.

[21] Viet L. Do, and Kenneth Y. Yun, "A Low-Power VLSI Architecture for Full-Search Block-Matching Motion Estimation," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 8, no. 4, pp. 393-398, Aug. 1998.

[22] W. Li, and E. Salari, "Succesive elimination algorithm for motion estimation," *IEEE Trans. Image Processing*, Vol. 4, no. 1, pp. 105-107, Jan. 1995.

[23] Wujian Zhang, Runde Zhou, and Kondo, T., "Low-power motion-estimation architecture based on a novel early-jump-out technique," *The IEEE International Symposium on Circuits and Systems*, Vol. 5 , pp. 187-190, 2001.

[24] Bede Liu, and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 3, no. 2, pp. 148-157, Arp. 1993.

[25] Chok-Kwan Cheung, and Lai-Man Po, “Normalized partial distortion search algorithm for block motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, Vol. 10, no. 3, pp. 417-422, Arp. 2000.

[26] Rafael C. Gonzalez, and Richard E. Woods, “Digital Image Processing,” Addison Wesley, Sep. 1993.

計畫成果自評

本計畫第二年已著手設計功率意識高階合成之應用。在影像處理部份，針對移動估測合成彈性功率調節電路。在通用數位信號處理部份，提出系統階層模型與驗證技術。在關鍵運算單元方面，已完成多模功率感知乘法器設計。本計畫之研究成果已發表或已接受發表為下列五篇期刊論文與一篇會議論文：

- Hsien-Wen Cheng and Lan-Rong Dung, "A Power-Aware Motion Estimation Architecture Using Content-based Subsampling," Journal of Information Science and Engineering, vol. 22, no. 4, pp. 799-818, 2006.
- Hsien-Wen Cheng and Lan-Rong Dung, "A Content-based Methodology for Power-Aware Motion Estimation Architecture," IEEE transactions on Circuits and Systems II, vol.52, No.10, pp.631-635, 2006.
- Lan-Rong Dung and Hsueh-Chih Yang, "A Parallel-In Folding Technique for High-Order FIR Filter Implementation," accepted by IEICE transactions on Fundamentals.
- Tsung-Hsi Chiang, and Lan-Rong Dung, "System level verification on high-level synthesis of dataflow algorithms using Petri net," accepted by WSEAS transactions on Circuits and Systems.
- Chuan-Sheng Lin and Lan-Rong Dung, "A NAND Flash Memory Controller for SD/MMC Flash Memory Card," to be appeared in IEEE transactions on Magnetics
- Tsung-Hsi Chiang, and Lan-Rong Dung, "System-Level Verification on High-Level Synthesis of Dataflow Graph," ISCAS 2006.

另外，部分研究成果正投稿于 IEEE 相關期刊。經由本計畫之執行正培養三名博士生及四位碩士生從事於高階合成相關之研究領域。