Parallel Dynamic Voltage and Frequency Scaling for Stream Decoding using a Multicore Embedded System

Ying-Xun Lai, Yueh-Min Huang
Department of Engineering Science
National Cheng Kung University
Tainan, Taiwan
eetaddy@gmail.com
huang@mail.ncku.edu.tw

Chin-Feng Lai
College of Electrical Engineering
and Computer Science
National ILan University
I-Lan, Taiwan
cinfon@ieee.org

Ljiljana Trajković
School of Engineering Science
Simon Fraser University
Vancouver, Canada
ljilja@cs.sfu.ca

***Abstract*—Parallel structures may be used to increase a system processing speed in case of large amount of data or highly complex calculations. Dynamic Voltage and Frequency Scaling (DVFS) may be used for simpler calculations in order to decrease the system voltage or frequency and achieve lower power consumption. Combining these two mechanisms may lead to higher efficiency and lower power consumption. In this paper, we introduce a parallel decoding process with Digital Signal Processing (DSP) for power efficiency in a heterogeneous multi-core embedded system. We describe a parallel low-power design on the system level. Under the condition of preserving the original decoding process, we manage the size of the system's multimedia buffer by considering the spontaneous streaming transfer and tuning the decoding process scheduling time by using the DVFS system in order to decrease the multimedia data dependency and achieve a multi-core embedded system with accurate and low-power detection mechanism.**

I. INTRODUCTION

Many current products employ embedded systems. The improved quality of commercial products and demand for multimedia applications require increasing number of data operations. Due to the demand for higher system frequency, the newly proposed hardware embedded systems begun using multi-core designs. These new architectures pose many challenges to developers:

1. Many embedded multimedia applications exhibit dependency problems during decoding processes that refer to the previous segment to perform decoding. Developers of multi-core systems need to consider how to effectively distribute data to different cores for processing and how to avoid dependency problems.

2. Compared to single core platforms, multi-core systems need a power managing mechanism to prevent excess power usage, especially in case of embedded systems such as handheld and battery devices. Dynamic Voltage and Frequency Scaling (DVFS) is a viable solution: it dynamically adjusts the system voltage or frequency during low calculation applications and it is effective for decreasing power consumption. The design challenge is to predict the system voltage or frequency with a running application process in order to achieve low power.

3. An important and realistic problem for a system developer is the overhead time to be invested to change the single core system platform to a fully working system. This remains a major issue to for developers and manufacturers.

In this paper, we introduce a front-wave parallel power management stream decoding system. We consider the entire system structure. While preserving the original decoding process with a single core, the parallel decoding implementation is achieved by using a simple yet effective concept: using the buffer management mechanism under the acceptable limits for the end-user to remove the time slack and data loss with estimation error. We then combine parallel processing and buffer management for adjusting both the system voltage and frequency according to parameters received from the two mechanisms.

In Section II, we introduce the DVFS system and review related proposals in the area of parallel structures and single core decoding procedures. In Section III, we introduce the front-end parallel DVFS system and describe system structure and module design. The implementation of the experimental platform and power efficiency prediction is given in Section IV. We conclude with Section V.

II. RELATED WORK AND BACKGROUND

We classify related proposals in two types: parallel decoding and the DVFS system.

A. Parallel Decoding

There are many designs that employ parallel decoding. If a complete decoding frame is used as a separation point, the proposed designs may be grouped into two main types: front-wave parallel processing and internal parallel processing.

- *Front-Wave Parallel Processing*

In front-wave parallel processing, parallel distribution with the decoding frame data is performed first. The decoder is then used for decoding calculations, where one video segment is split by Group of Pictures (GoP) [1], [2] and each GoP is distributed to a processor for decoding. Flierl et al., [3] proposed a B frame parallel decoding method. The main

concept is that B frames are not referenced by other frames and, hence, may be distributed to different processors for decoding. However, this method is not applicable to H.264 since B frames may be referenced by other frames in H.264 decoding.

• *Internal Parallel Processing*

The front-wave processing structure first completes splitting the data before delivering data to the system for processing. In contrast, the internal parallel processing delivers the frame to the system and allows the system to perform the splitting process. This splitting has the advantage because letting the system to do the process scheduling and planning may produce better parallel decoding efficiency [4], [5]. However, the drawback is usually that the entire decoding structure needs to be changed. There are H.264 decoding proposals that perform splitting according to slices [6], because slices are the smallest independent decoding units. Using slices to separate the decoding frames can produce good parallel decoding efficiency. Van der Tol et al., [7] achieve parallel structure by taking each decoding procedure and separating it into various tasks and by assigning different tasks to different decoders.

B. *Dynamic Voltage and Frequency Scaling (DVFS)*

For many commercial electronic systems, a good power manager is a necessity, especially for handheld system or battery-based devices. Various dynamic power management systems have been proposed [8], [11]. These management systems dynamically adjust the system voltage or frequency to complete process with the smallest power consumption before the deadline. We calculate the power consumption of the processor using the CMOS manufacturing technology as:

$$P = C_{eff} * {V_{dd}}^2 * f \,, \tag{1}$$

where C_{eff} is the effective switched capacitance, V_{dd} is the operating voltage, and f is the operating frequency. T_{Proc} defines the time period of completing process, is calculated as:

$$T_{Proc} = C/f = C\frac{V_{dd}}{K*(V_{dd}-V_t)^a} \,. \tag{2}$$

According to the energy equation:

$$E = P * T_{Proc} \simeq C_{eff} * {V_{dd}}^2 \,. \tag{3}$$

Hence, we can reduce the power consumption by adjusting the system voltage or frequency to achieve T_{Proc} without a time slack and to ensure that the system does not lag while playing the multimedia data before the deadline.

III. Proposed Parallel Decoder Streaming Process

In this Section, we provide description of the parallel DVFS decoding system and its design. We also introduce a model for the parallel structure and the DVFS mechanism for stream processing.

A. *System Overview*

The diagram of the proposed system is shown in Fig. 1. As the date stream enters the heterogeneous multi-core platform, the Micro Processing Unit (MPU) takes the data stream and performs parallel scheduling. The DVFS system decoding prediction is performed according to the video dependency and video format. In heterogeneous multi-core platforms, the decoding process is added to the video and audio parts of the digital signal processing (DSP) system. This proposal focuses on a single MPU with a multiple DSP core structure platform and addresses parallel decoding. We use MPU to manage system parallel planning with the DVFS prediction and settings. Using the front-wave process design, the DSP decoding need not be changed to implement the DVFS system process.

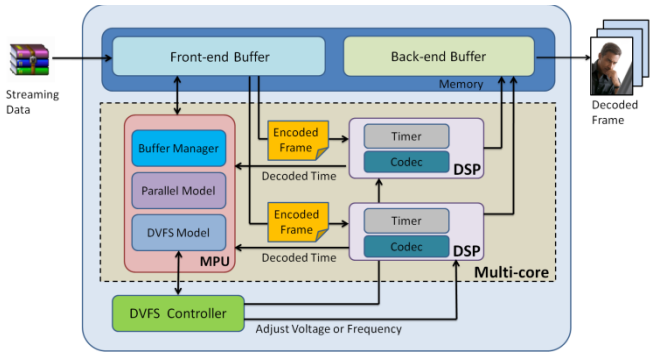

Fig. 1 The architecture of the proposed system.

B. *Parallel DVFS on Stream Decoding*

In past proposals, the main approach to decrease energy consumption in multimedia decoding system included: 1. reducing time slack and 2. correctly predicting the system voltage or frequency to process the next frame. To achieve these two goals, we use a simple yet applicable concept. To build the entire DVFS structure, we utilize two buffers: front-end and back-end. Using the front-end buffer achieves parallel mechanism and also easily erases time slack. We define as a deadline the set time of decoding for each frame. We do not change the scheduling deadline and use an effective method to predict the system voltage or frequency. We then adjust the system voltage and frequency according to the predicted values and the priorities of decoding tasks.

This proposal combines the offline method and an online mechanism to decrease prediction error rate. Before the decoding begins, the system employs the DVFS model, encoded frame format and the previous frame size, and decoded time to determine the initial system voltage and frequency. The DSP-end will dynamically adjust the system voltage or frequency according to the time spent for executing each function and the relevant information from the decoding process.

C. *DVFS Algorithm for Independent Frames*

The system voltage should be chosen to achieve a low-power system. In embedded systems, the DVFS hardware module usually provides several set voltages and respective frequencies to allow the developer to use software tools to control the system voltage or frequency. Let $\upsilon = \{V_1, \dots, V_{max}\}$ be the adjustable voltage provided by the hardware and $F = \{f_1^{V_x}, \dots, f_{max}^{V_x}\}$ be the adjustable frequency under V_x voltage. If the system starts from frequency $f_{max}^{V_x}$

and achieves $T_{\max}^{V_x}$, we can calculate based on (2) the workload that is estimated according to past frame defined by system frequency and cycle as:

$$C = T_{\max}^{V_x} * f_{\max} = T_n^{V_x} * f_n , \quad (4)$$

where f_n is the chosen frequency and T_n is the respective workload. Predictions are made according to the worst-case scenario and satisfy condition:

$$T_n \le T_{dead} < T_{n+1} . \quad (5)$$

In an ideal situation, the predicted workload satisfies the applied deadline. If the system frequency cannot satisfy T_n=T_{dead}, time slack occurs. We propose here a time-oriented prediction that differs from the worst-case prediction rule. It is based on finding the closest T_{dead} frequency given by the inequality:

$$|T_{dead} - T_n| < |T_{dead} - T_{n-1}| \&\& |T_{dead} - T_{n+1}| . \quad (6)$$

This prevents estimation errors and the next time segment may be estimated as:

$$T_{new} = (T_{dead} - T_n) + T . \quad (7)$$

We then use T_{new} to estimate the system voltage and frequency of the next time segment.

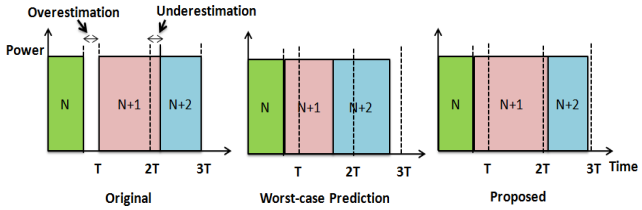

Fig. 2 The DVFS algorithm for independent frames.

D. DVFS Algorithm for Dependent Frames

In this Section, we discuss the immediate data dependency issue in order to set a suitable system voltage and frequency to reduce system delay. Since the encryption decoding process that involves changing and re-ordering data in one frame has no data dependency issues with other frames, the entire system may begin decoding when the third column of the reference frame completes decoding to ensure that the reference part is entirely decoded avoiding data miss. Hence, we only need to ensure that the column number remains above 3, as shown in:

$$T_{ED}^{dec} = T_{ED}^{ref} + T_{3C}^{ref} , \quad (8)$$

where T_{ED}^{dec} is the time spent for encryption decoding, T_{ED}^{ref} is the reference time needed for encryption decoding, and T_{3C}^{ref} is the time needed to decode the first three columns of the frame. We use (8) and the system average priority to determine T_{ED}^{ref} and T_{3C}^{ref}:

$$T_{ED}^{ref} = T * \frac{W_{ED}}{W_{dec}} \quad (9)$$

$$T_{3C}^{ref} = \left(T - T_{ED}^{ref}\right) * \frac{3}{C_{total}} , \quad (10)$$

where C_{total} is the frame column number that can determine the system voltage and frequency needed by the decoding frame in the encryption decoding. In ideal circumstances, this can correct the data dependency issue. However, in realistic decoding schemes, various frame format decoding schemes have different speeds.

IV. Implementation and Analysis

A. Implemention of the DVFS Algorithm on a Heterogeneous Multicore Platform

The proposed system employs as the hardware platform the Parallel Architecture Core (PAC) Duo developed by the Industrial Technology Research Institute (ITRI), Taiwan. It implements the proposed power efficiency perceptive system on the Android OpenCORE. The system structure and operating procedure are shown in Fig. 3. The system operates from the upper application layer Android Package (APK) that calls the OpenCORE multimedia framework to perform video playback. The OpenCORE is responsible for coordinating the DSP for processing. It employs the DVFS predictor decoder load, which is based on the previous frame size, and decoded time to perform prediction of the appropriate DVFS level. It then uses I/O controller to transfer data to the DSP Power Management Driver and performs DSP voltage and frequency control for the DVFS controller to achieve coordination. Finally, the OpenCORE activates DSP to perform video decoding. All bit streams are 30 fps, with Common International Format (CIF) resolution for a total of 300 frames.

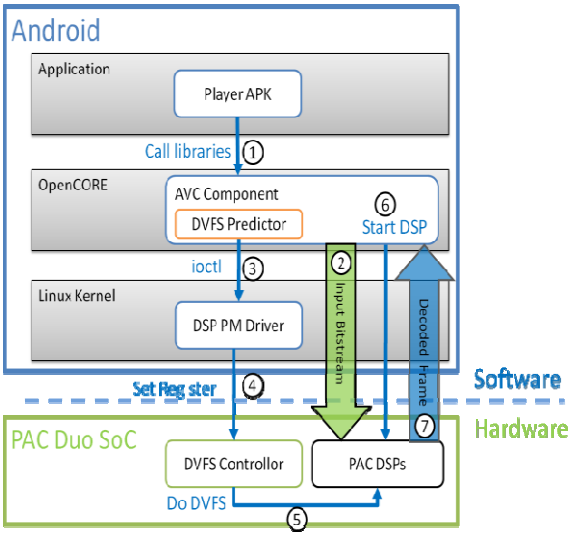

Fig. 3 The Android system structure and procedure.

B. Bitrate Effects on Energy Consumption

Different bit rates affect the decoding: the higher the bit rate the larger the frame size. Hence, the system needs a higher frequency to complete decoding. We tested 200 kbps, 400 kbps, and 600 kbps bit rates and measured their energy consumption compared to the baseline without proposed mechanism, as shown in Fig. 4.

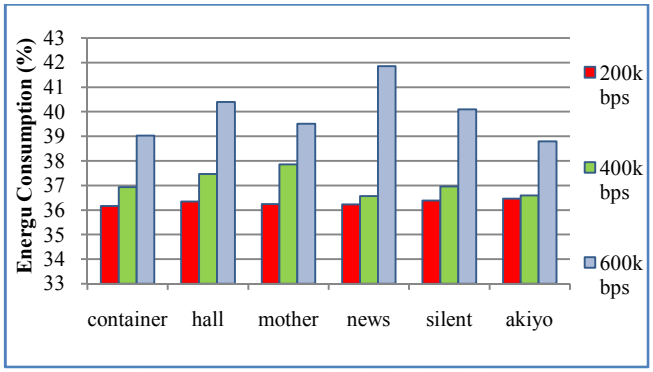

Fig. 4 Energy consumption for different bit rates.

As predicted, the power consumption increases with the bit rate. However, the designed prediction module for the three bit rates still has between 36.2% and 41.9% smaller energy consumption.

C. Deadline Miss Analysis

A deadline miss occurs when a frame does not complete decoding before the deadline limit. It may be seen as the marker for tuning the DVFS algorithm. After using the proposed prediction module, there are different levels of a deadline miss. When the module predicts a high DSP load, less energy will be saved even though the deadline miss ratio is small. For example, in the case of a news bit stream, the energy consumption is the highest even though the deadline ratio is the smallest. However, if the deadline miss error rates are added, as shown in Fig. 5, prediction modules other than news have the bit stream error within 5%.

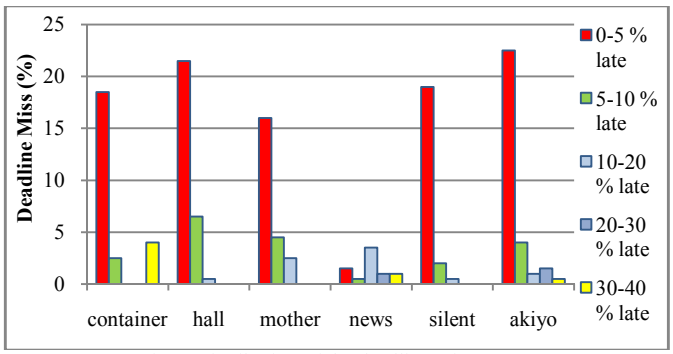

Fig. 5 Distribution of the deadline miss rates.

V. Conclusions

In this paper, we introduced a parallel decoder streaming process for power efficiency perception in a multi-core embedded system by combining multi-core scheduling and a DVFS mechanism to provide a highly efficient and energy multi-media decoding mechanism. The DVFS decreases the system power usage through scheduling and correcting calculations and resolves the multimedia data dependency issues. This mechanism was implemented on the Android system. We also analyze the effectiveness of the developed platform. The experimental results show the decrease of 36.2% to 41.9% in power usage. The proposed framework provides a new approach for integrating power efficient oriented mechanisms by tuning the system voltage or frequency in multicore embedded systems.

References

[1] T. Olivares, F. J. Quiles, P. Cuenca, L. Orozco-Barbosa, and I. Ahmad, "Study of data distribution techniques for the implementation of an MPEG-2 video encoder," *Parallel and Distributed Computing Systems '99*, MIT, Boston, MA, USA, pp. 537–542, Nov. 1999.

[2] D. Farin, N. Mache, and H. N. Peter, "SAMPEG, a scene adaptive parallel MPEG-2 software encoder," *SPIE Visual Communications and Image Processing*, pp. 272–283, Jan. 2001.

[3] M. Flierl and B. Girod, "Generalized B pictures and the draft H. 264/AVC video-compression standard," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 587–597, Jul. 2003.

[4] K. Asanovic, K. Keutzer, D. A. Patterson, et al., "The landscape of parallel computing research: a view from Berkeley," *Tech. Rep. UCB/EECS-2006-183, EECS Department*, University of California, Berkeley, Dec. 2006.

[5] C. Jike, N. Satish, B. Catanzaro, K. Ravindran, and K. Keutzer, "Efficient parallelization of H.264 decoding with macro block level scheduling," *IEEE International Conference on Multimedia and Expo*, pp.1874–1877, July 2007.

[6] A. Azevedo, C. Meenderinck, B. Juurlink, A. Terechko, J. Hoogerbrugge, M. Alvarez, and A. Rammirez, "Parallel H.264 decoding on an embedded multicore processor," in *Proc. 4th International Conference on High Performance and Embedded Architectures and Compilers, HiPEAC*, Paphos, Cyprus, pp. 404–418, Jan. 2009.

[7] E. van der Tol, E. Jaspers, and R. Gelderblom, "Mapping of H.264 decoding on a multiprocessor architecture," *Image and Video Communications and Processing 2003*, pp. 707–718, May 2003.

[8] S. Yaldiz, A. Demir, and S. Tasiran, "Stochastic modeling and optimization for energy management in multicore systems: a video decoding case study," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 27, pp. 1264–1277, July 2008.

[9] P. Francesco, P. Antonio, B. Davide, B. Luca, and M. Poletc, "Energy-efficient multiprocessor systems-on-chip for embedded computing: exploring programming models and their architectural support," *IEEE Trans. on Computers*, vol. 56, pp. 606–621, May 2007.

[10] S. Lee, "Low-power video decoding on variable voltage processor for mobile multimedia applications," *ETRI Journal*, vol. 27, pp. 504–510, Oct. 2005.

[11] E. Nurvitadhi, B. Lee, C. Yu, and M. Kim, "A comparative study of dynamic voltage scaling techniques for low-power video decoding," *International Conference on Embedded Systems and Applications*, pp. 23–26, June 2003.