The Simulation of Dynamic Voltage Processor with MPEG decoding

JINAH SHIN SUNGIK JEON KYOIL CHUNG
Electronics and Telecommunications Research Institute
166 Gajeong-Dong, Yusung-Gu, Daejon
KOREA
{jashin, sijun, kyoil}@etri.re.kr http://www.etri.re.kr

Abstract: As the usage of personal mobile devices increase, there are many researches for more efficient and convenient mobile systems. Especially, reducing energy consumption is one of the critical issues for those systems owing to their portability. Dynamic voltage scaling (*DVS*) has been suggested to reduce the processor power which is highly consumed energy component. In this paper, we propose a couple of *DVS* algorithms for MPEG decoder using variance of MPEG frame decoding time. The processor power is measured out frame by frame using *Wattch*, power estimator based on instruction set simulator. As a result, our algorithm shows about 70% ~ 85% energy reduction of processor with *DVS* per frame unit.

Key-Words: DVS, Low-power system, Processor power reduction, SimpleScalar, Voltage scaling, Wattch

1 Introduction

The energy dissipation of embedded system has become critical issue due to the necessity of its longer battery life. System designers have to consider the methods to make longer battery lifetime in the portable environment, because users want to use mobile devices for a long time without recharging. One of the ways to increase battery lifetime is to take the low power states of portable system components [1].

Though power consumption of microprocessor is one of the most principal factors to affect the total energy consumption of one system [2], the energy consumption of processor is rapidly increases as the performance of that dramatically grow. While the Intel 386 processor consumed about 2 Watts of energy, a Pentium 4 use as much as 55 Watts [3]. For this problem, several techniques of processor power reduction are proposed: shut-down method, frequency-setting technique, dynamic voltage scaling and etc. Shut-down method have non-negligible overhead energy cost between shut down and wakeup process. Furthermore, it takes a time delay to transition between in and out the low power idle state [4]. Frequency-setting technique cannot expect the substantial energy reduction because low frequency takes longer time to complete a job [5]. Dynamic voltage scaling can achieve significant energy savings by dropping the supply voltage while processor runs less overhead. We will show the examples of variable voltage processor in the next section.

When *DVS* applied a processor, it is necessary to predict the accurate voltage level to finish a real-time job in the deadline. In the performance point of view, job delay can occur during dynamic voltage scaling. When a processor runs on the low voltage level, the speed of processor will decrease and job cannot meet the deadline. Our proposed algorithm estimates the running time of MPEG frame for parsing the header before decoding. The coefficient of determinations R^2 of the predictions are 0.9 or more approximately.

In this paper, we simulate *DVS* technique and estimate the power on *Wattch* [6] which is extended version of *SimpleScalar* instruction set processor simulator [7]. As workload of this simulation, we use MPEG system which is one of the public applications used in mobile devices.

The rest of this paper is organized as follow. After introducing previous related works in section 2, section 3 presents our proposed approach of *DVS*. Then section 4 explains power estimation techniques of processor for simulation. Section 5 shows the result the simulation. Finally, section 6 summarizes the main conclusions of this work.

2 Related Work

This section explains the techniques of dynamic voltage scaling. First, we describe previous power reduction researches and dynamic voltage scaling in section 2.1. Section 2.2 shows the practical examples of dynamic voltage scaling. 2.3 explains the fundamental MPEG concepts and characteristics.

2.1 Dynamic Voltage Scaling (*DVS*)

There are several components which mainly consumed the energy of mobile system. Processor, backlight for the LCD screen and hard disk drive are the top three power consuming components [2][8]. Especially the study of processor power consumption, the most critical factor of system power, is going on actively for low power system.

Weiser et al. proposed the voltage scheduling to reduce processor energy at [9]. The presented three algorithms which are OPT, FUTURE and PAST are the basic of all *DVS* methods. OPT takes the entire trace, and stretches all the run times to fill all the idle times. OPT algorithm is impractical because we cannot predict when the idle time starts and stops. FUTURE peers into the future only a small window, and optimizes energy over while window size and PAST looks a fixed window into the past and assume that the next window will be similar the previous one.

These algorithms are applied by Son et al [10]. In their research, *DVS-DM* is based on PAST algorithm with delay and drop factor, *DVS-PD* is based on FUTURE algorithm with predicting of GOP size.

Processor voltage has effect amount of quadratic effect to energy, while frequency is proportional to energy linearly. In CMOS design, total energy consumption is given by

$$E = V_{DD}^2 C_{eff} f T \qquad (1)$$

where V_{DD} is the supply voltage, C_{eff} is the average capacitance, f is the processor frequency and T is the total execution time. We can find out easily that the supply voltage is the most critical factor to reduce energy consumption and frequency can also affect that. Therefore, it can be concluded that decreasing supply voltage and frequency reduce energy consumption. However, lower voltage generates the circuit delay as [11].

$$\tau \propto \frac{V_{DD}}{(V_G - V_T)^2} \qquad (2)$$

where τ is the propagation delay of the CMOS transistor, V_T the threshold voltage, and V_G the input gate voltage. The propagation delay restricts the clock frequency in a processor at the certain voltage stage. Processor can operate at a lower supply voltage, only if the clock frequency is reduced to tolerate the increased delay.

From the Fig. 1, all tasks should be done until 15. Each tasks are done with fixed voltage processor, therefore if a task is done before starting next task,

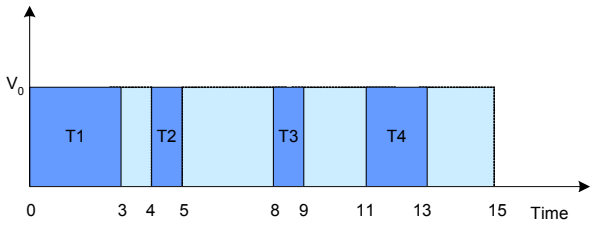

Fig. 1 CPU workload without energy reduction strategy

processor may stay with fixed supply voltage (V_0) in the idle time. When there is no processor energy reduction strategy, the energy is proportional to $V_0^2 \times 15$. However, if we apply energy reduction techniques to the above case, the energy saving can be shown according to each approaches. First, shut-down approach stops processor operation during idle execution time. In this case, the consumed energy during T1, T2, T3 and T4 are proportional to $V_0^2 \times (3+1+1+2)$ and it includes non-negligible voltage switching overhead [4]. The other technique is voltage scaling which supply voltage sets depending on the task load. Following Fig. 2 shows that voltage scaling method is applied to Fig. 1. Energy consumption in the Fig. 2 can be derived $V_1^2 \times 4 + V_2^2 \times (3+4) + V_3^2 \times 3.5$ approximately. According to [4], energy efficiency of DVS is better than shut-down.

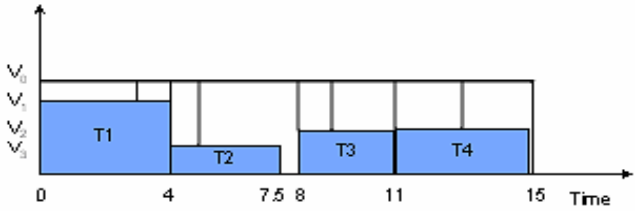

Fig. 2 CPU workload with voltage scaling strategy

As we have mentioned before, low voltage occurs circuit delay during processor operation. We have to consider circuit delay for saving power, because time is one of the factors to decide total energy consumption as represented in equation (1). Dynamic voltage scaling approach solves this problem removing idle operation time.

In equation (1), T can be broken up following equation with the definition that T can be represented by product of total number of cycle(N_{total}) and fixed unit clock time($t = 1/f$)

$$T = (N_{T1} + N_{T2} + N_{T3} + N_{T4} + N_{idle}) \cdot 1/f \qquad (3)$$

N_{T1}, N_{T2}, N_{T3} and N_{T4} are executed cycle number of each task and N_{idle} is the cycle number of idle operation represented light gray portion in Fig. 1. Let's assign f_1, f_3, and f_2 for T_1, T_2, and T_3 and T_4 to running frequency on supply voltage V_1, V_3, and V_2 respectively. For the idle area, zero frequency and voltage are assigned.

$$T=N_{T1}\cdot(1/f_1)+N_{T2}\cdot(1/f_3)+N_{T3}\cdot(1/f_2)+N_{T4}\cdot(1/f_2)+N_{idle}\cdot 0 \quad (4)$$

The total time T value is not change in spite of increase of the real operation owing to circuit delay. Therefore, the total execution time(T) does not effect to the total energy consumption. The power part, equation (5), is only considered in energy consumption of processor in the voltage scaling technique.

$$P = V_{DD}^{2} C_{eff} (f_1 + f_2 + f_3) \quad (5)$$

In equation(3), V_{DD} can be decreased by lower frequency dynamically, then the total power consumption is reduced as quadric of reduced voltage.

2.2 Dynamic Voltage Processor in practical

Mobile Intel Processor has 11~12 frequency stages at the different supply voltage. The processor accepts Intel SpeedStep technology for low power mobile system.

This processor has two performance modes and allows real-time dynamic switching of the voltage and frequency between the modes. This occurs by switching the core operating voltage and core processor speeds without resetting the system. There are two performance modes, *Maximum Performance* mode and *Battery Optimized Performance* mode. The former provides the best performance and runs at 600 or 650MHz. The latter provides the balance between performance and battery lifetime and operates at a lower frequency of 500Mhz [12].

Transmeta Crusoe has also variable voltage and frequency as represented in Table 1[13]. Transmeta *LongRun* power management technology continuously scales both the frequency and voltage of the processor according to the instantaneous performance demands of the system. It can detect different operating scenarios based on runtime performance information and the exploit these by adapting its power usage accordingly.

2.3 MPEG Decoding

MPEG(Moving Pictures Experts Group) is a group of people that meet under ISO to generate standards for digital video (sequences of images in time) and audio compression.

Frequency	Voltage	Power Consumption
667 MHz	1.6 V	5.3 W
600 MHz	1.5 V	4.2 W
533 MHz	1.35 V	3.0 W
400 MHz	1.225 V	1.9 W
300 MHz	1.2 V	1.3 W

Table 1 Clock frequency versus supply voltage for the Transmeta Crusoe processor

Standard MPEG stream is composed of three types of compressed frames: *I*, *P* and *B*. I frame is intra-coded pictures which is coded independently from all other frames. P frame is predictive-coded picture which is coded based on a prediction from a past I or P frame. And B frame is bi-directionally predictive-coded picture which is coded based on a prediction from a past and/or future I or P frames[14]. As a result, I frames are, on the average, the largest in size, followed by P frames, and finally B frames.

A Group of Pictures(GOP) is a sequence of frames from one I frame to the next I frame. Most of encoders use a fixed GOP pattern when compressing a video sequence, where the GOP pattern specifies the number and temporal order P and B frames between two successive I frames. Mostly, decoding time of I frame is also the longest and B frame has the shortest decoding time.

From decoding time point of view, I frames usually take longer than other frames while B frames take the least. On the contrary, *I-frame decoding time per byte* (*I_DTPB*) is the shortest while *B-frame decoding time per byte* (*B_DTPB*) is the longest because a large computational work is needed in motion prediction in case of B-frame and P-frame [15].

The reasons why we choose MPEG decoder for simulation workload are that MPEG is the best typical model of multimedia streaming applications which mobile devices use mostly and the workload is can be divided definitely by process overhead. Therefore, the MPEG decoder simulation can represent the tendency of streaming applications on mobile system and efficient energy reduction using dynamic voltage scaling.

3 Proposed Approach

In this section, we introduce the three *DVS* algorithms. The first one is proposed previous research[10], and others complement the efficiency of voltage scaling. We can show the results of those schemes in section 5.

Son and et al. propose *DVS-DM* and *DVS-PD*[10].

Both of those approaches do voltage scaling based on the GOP unit. *DVS-DM* sets and adjusts the voltage level by drop and delay rate in times past. *DVS-PD* determines the voltage by the predicting of the future GOP size from the header of GOP.

Generally, the variance of GOP is smaller than the variance of frame in one MPEG stream [14]. Therefore the case of *DVS* of frame by frame can be achieved more efficient when two voltage scaling techniques are applied. We propose two approaches about dynamic voltage scaling based on the frame characteristics. According to common MPEG frame characteristics, the decoding time of I frames is longer than other frames and P frames need more decoding time than B frames do. MPEG stream can be decoded with different clock frequencies depending on the frame type. The other scheme predict the decoding time of each frame by frame type and size. In our scheme, the cycle number of each frame is expected and the supply voltage is changed.

3.1 DVS-PFU (per frame unit) based on frame type

Fig. 3 and 4 shows the example of required clock cycle per each frame when typical MPEG movie clip, the former is StarWars movie trailer and the latter is the moving picture of ski, described below the figures run. In the first one, the average decoding clock cycle of I, P, and B frames are 715552, 533147, and 272160 respectively. The other clip has 760192, 503132, and 276694 clock cycles for each frame. We can see the difference of required clock cycle between each frame in these figures. For example, the largest number of clocks in Fig. 3 is 971,500 for an I frame and the smallest one is 269,000 for a B frame. There is a wide difference between those two values. If the voltage and frequency switching overheads are small, we can expect power reduction due to assign higher frequency to I frame and lower voltage to P frame or B frame.

There is problem which cannot handle the exception frame like small I frame or large B frame. This approach ignores the variation within frame type, therefore when large B frame is decoded, time delay can occur. This issue is caused by that the decoding time of B frame(*B_DTPB*) is larger than other frames.

3.2 DVS-PFU based on predicted decoding time

The frame size is the most important factor which decides the clock cycle of decoding a frame. A frame size can be known by parsing operation before

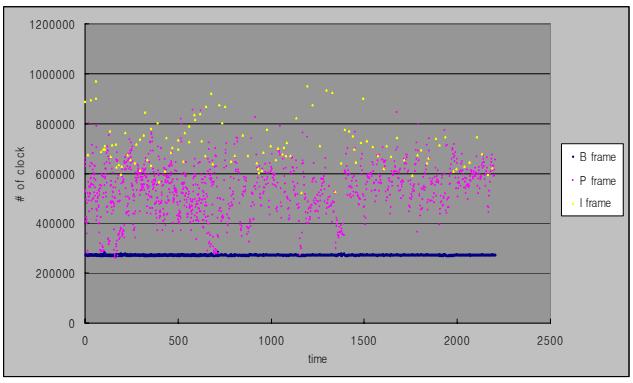

Fig. 3 Number of required clocks which each frames takes in StarWars Trailer

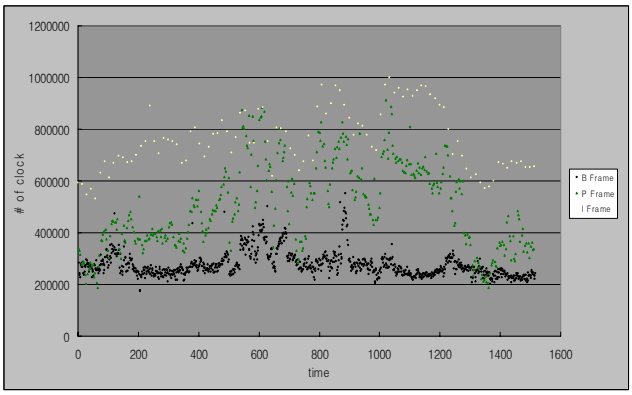

Fig. 4 Number of required clocks in the ski moving picture

decoding that frame. Therefore if we estimate a frame size before decoding the frame, the operation clock cycle of that can be found out.

Fig. 5 represents an example of the required decoding clocks according to the frame size. It can be simply analyzed to represent linear line shape. However, it is unsuitable that decide the decoding clock only with frame size because *I_DTBP* is shorter than *P_DTBP* and *B_DTBP*. If there are two frames with same size, one is I frame and the other is B frame, B frame has longer decoding time than I frame. Therefore, when this data is categorized by frame type, we can get more accurate linear regression models which infer required clock cycle by frame size. Moreover, this approach can be applied to different frame resolution because most MPEG stream represent similar regression model.

The regression model of Fig 5. is shown in Table 2. All regression models are positive linear. It means that the decoding time of frame is generally proportioned to frame size. However, due to the difference of decoding time per byte between frames, each frame type has an indivisual regression model : I frame has big intercept value and small gradient. On the contrary,

B frame has small intercept and large gradient.

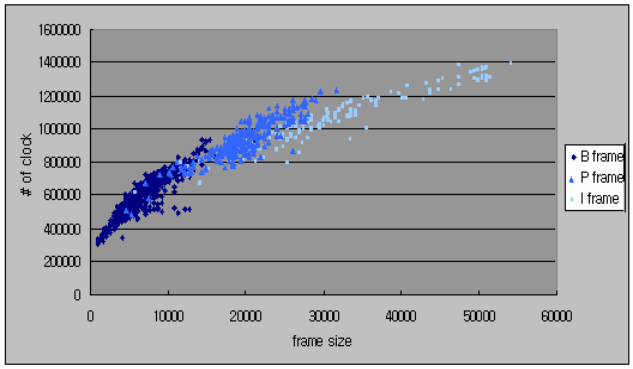

Fig. 5 Number of required clocks according to frame size

Frame Type	Regression model	R^2
I frame	$16x + 649365$	0.96
P frame	$25x + 404994$	0.89
B frame	$35x + 380179$	0.93

Table 2 Regression model for the expected decoding cycle of Fig. 5

4 Simulation

In this section, we present the implementation set up for our proposed approach, then show the power reduction of dynamic voltage scaling per frame unit and comparison with the result of *DVS-PD*.

SimpleScalar simulator performs fast, flexible, and accurate the simulation of *SimpleScalar* 5-stage pipeline architecture. Any C source codes can be compiled with *SimpleScalar*-GCC compiler and tested their performance which user wants to see. The statistics of performance of the given program can be obtained as the results of simulator, e.g. number of instruction, IPC, cache miss rate, and etc. The whole simulation environment parameters are followed the given values in [7].

Wattch is power model which runs with *SimpleScalar* simulator and the power consumption of *SimpleScalar* presents with performance statistics. We use *Wattch* for estimating the change of processor power in *DVS* algorithm.

As benchmark source, our target application MPEG decoder, from Berkeley University, is compiled with *SimpleScalar*-GCC. The decoder then executes on the *SimpleScalar* processor simulator which includes *Wattch*. For dynamic voltage scaling, the simulator has to be modified with some hardware configuration and several system calls are added to the simulator. Fig. 6 shows the experimental framwork of MPEG decoding on the *Wattch* power estimaotr.

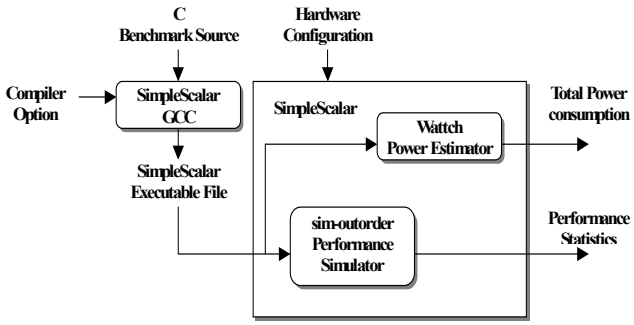

Fig. 6 Overall of experimental framwork

The number of cycles for performance measurement and the consumed power of the different movie clip can be gotten after the simulation. Each of them is obtained from *SimpleScalar* performance simulator and *Wattch* power estimator respectively. Moreover, different *DVS* techniques can be applied on this framework and are compared with another.

5 Result

There are four MPEG decoders to compare their performances with sample streams. The first one is original MPEG decoder without any modification, the second one is MPEG decoder with *DVS-PD*, the third implements DVS-PFU(*DVS* per frame unit) based on frame type, and the last one runs DVS-PFU based on predicted decoding time. Power consumption of each simulation can be gotten from the *Wattch* output. In our simulation, we assume that the *SimpleScalar* architecture runs on various voltage levels for supporting dynamic voltage scaling time and there is no delay to switch voltage level or frequency level, because the voltage switching delay is much smaller as compared with a frame decoding time[11].

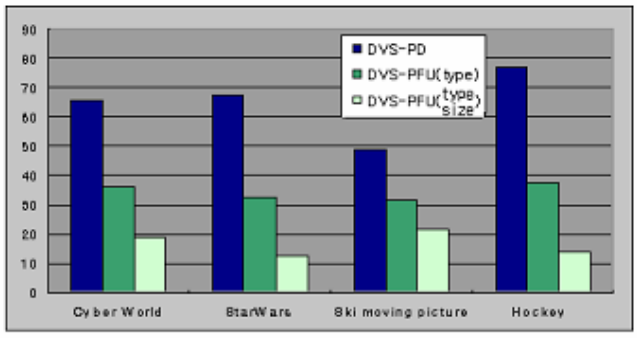

Fig. 7 Power Conusmption of MPEG streams

Our simulation shows that MPEG decoder with DVS-PFU with predicted decoding time consumes the least power among the four decoders and MPEG decoder with DVS-PFU with frame type is little better than *DVS-PD* approach. Fig. 7 shows the power consumption by MPEG decoders with several sample streams when the original processor power consumption is set to 100%.

The power consumption of *DVS-PD* decreases above 50% of original power except ski moving picture. It represents the worst performance among other *DVS* decoders due to its inefficiency that ignore the variation of workload within frames. Each frame type has different workload as the decoding time of I frame is the largest and the decoding time of B frame is the smallest. Using this feature, power consumption can save about 70%. However, this scheme cannot handle the exceptional frame. Therefore, voltage scaling is not accurate to the specific frame. The decoding clock of frame is analogized with frame size and type. With this characteristic, the high power saving up to 85% with DVS-PFU can be achieved from regression model which predict the decoding clock. This approach shows the most efficient power consumption as we can see in Fig. 7.

6 Conclusion

Dynamic voltage scaling, especially real-time scheduling technique can accomplish the ideal power consumption when the processor is running multimedia applications.

We compare four MPEG decoders: original MPEG decoder without any modification, MPEG decoder with *DVS-PD* which proposed in previous research, decoder with DVS-PFU scaled by frame type, and decoder which varies the voltage level by predicted frame decoding time.

Energy reduction is related with the decoding time of application. An amount of workload, for example I frame in MPEG stream, cannot achieve energy saving. DVS-PFU with frame type uses this characteristics of MPEG decoder, then the energy saving is expected during decoding of P or B frame which has less decoding time. The other approach of DVS-PFU complements the former approach which has limitation not to manage the exceptional frame. The proposed technique DVS-PFU based on predicted decoding time is simple but powerful methodology to reduce processor energy consumption. As a result, nearly 85% of the original consumed energy can be reduced on this experimentation compared with that DVS-PFU with frame type reduce about 70% and *DVS-PD* approach archives 50% power reduction.

References:

[1] J. Lorch, “A Complete Picture of the Energy Consumption of a Portable Computer,” *Master thesis*, Computer Science, University of California Berkeley, 1995

[2] J. Lorch, “Energy consumption of Apple Macintosh computers,” *IEEE Micro 18(6)*, November/December 1998, pp 54-63

[3] K. Flautner, S. Reinhardt, and T. Mudge, “Automatic Performance Setting for Dynamic Voltage Scaling,” *Int’l Conf. On Mobile Computing and Networking*, July 2001, pp. 260-271

[4] A. Acquaviva, L. Benini, and B. Ricco, “An Adaptive Algorithm for Low-Power Streaming Multimedia Processing," *Design Automation and Test in Europe*, March 2001, pp 273-279

[5] K. Govil, E. Chan, and H. Wasserman, “Comparing Algorithms for Dynamic Speed-Setting of a Low-Power CPU,” *Int’l Conf. on Mobile Computing and Networking*, November 1995, pp. 13-25

[6] D. Brooks, V. Tiwari, and M. Martonosi, “Wattch: A framework for Architectural-Level Power Analysis and Optimizations,” *27th ann. Int’l Symp. on Computer Architecture*, pp. 83-94, June 2000

[7] D. Burgerj, T. Austin, “The SimpleScalar Tool Set, Version 2.0,” *Tech. Rep. 1342*, CS Department, University of Wisconsin Madison, June 1997.

[8] G. Welch, “A Survey of Power Management Techniques in Mobile Computing Operating Systems,” *ACM Operating System Review 29(4)*, 1995, pp 47-56

[9] I. Weiser, B. Welch, A. Demers, and S. Shenker, “Scheduling for Reduced CPU Energy,” *SOSDI*, November 1994, pp 13-23

[10] D. Son, C. Yu, and H. Kim, “Dynamic Voltage Scaling on MPEG Decoding,” *Int’l Conf. on Parallel and Distributed Systems*, June 2001, pp.633-640

[11] J. Pouwelse, K. Langendoen, and H. Sips, “Dynamic Voltage Scaling on a Low-Power Microprocessor,” *Int’l Conf. on Mobile Computing and Networking*, July 2001, pp 251-259

[12] Mobile Intel PentiumIII Processor in BGA2 and Micro-PGA2 Packages Datasheet, Intel Corporation

[13] The Technology Behind Crusoe Processors, Transmeta Corporation

[14] O. Rose, “Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems,” *Technical Report 101*, University of Wuerzburg, Insitute of Computer Science, February 1995

[15] J. Lorch, A. Smith, “Improving Dynamic Voltage Scaling Algorithms with PACE,” *Proc. Of the ACM SIGMETRICS Conference*, June 2001, pp. 50-61