Dynamic Voltage Scaling Techniques for Power-efficient MPEG Decoding

Masters Thesis

Presented by
Wissam Chedid

Advisor
Dr. Chansu Yu

December 2nd, 2003

Outline

- Power trends
- Power Management, Dynamic Voltage Scaling (DVS)
- MPEG
- DVS for MPEG decoding
- Experimental framework
- Results
- Conclusion

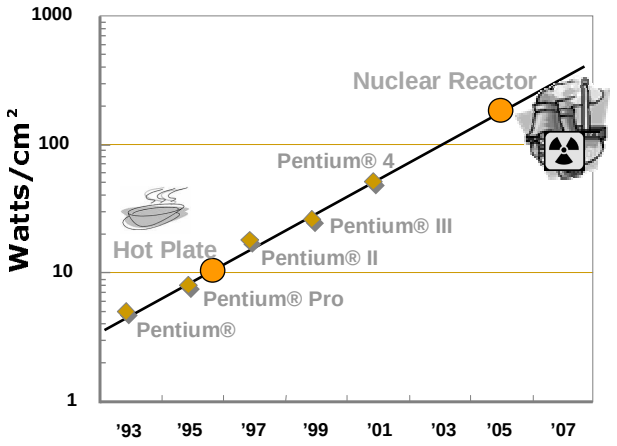
Outline

- Power trends
- Power Management, Dynamic Voltage Scaling (DVS)
- MPEG
- DVS for MPEG decoding
- Experimental framework
- Results
- Conclusion

Power Trends

- Power= CV^2f, Power density= CV^2f/Area
- New processors:
  - More integration ➡ Higher C
  - Faster clocks ➡ Higher f
  - Smaller die ➡ Smaller Area
- More power consumption
- More and more power density

Power Trends



* "New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies" – Fred Pollack, Intel Corp. Micro32 conference key note - 1999. Courtesy Avi Mendelson, Intel.

Outline

- Power trends
- Power Management, Dynamic Voltage Scaling (DVS)
- MPEG
- DVS for MPEG decoding
- Experimental framework
- Results
- Conclusion

Power Management Techniques

- Static Power Management
  - Offline, during design time.
  - Estimate power consumption and find power-efficient improvements on different hardware and software levels.
- Dynamic Power Management
  - Online, during runtime.
  - Monitor running system behavior and apply power-reduction techniques accordingly.

Dynamic Voltage Scaling (DVS)

- Power= CV^2f
- Reducing supply voltage V will reduce power quadratically
- Problem:
  - It incurs more propagation delay and thus, forces a reduction in clock frequency f,

Dynamic Voltage Scaling (DVS)

Frequency	Voltage
667 MHz	1.6 V
600 MHz	1.5 V
533 MHz	1.35 V
400 MHz	1.225 V
300 MHz	1.2 V

Transmeta Crusoe processor

Frequency	Voltage
733 MHz	1.15 V
677 MHz	1.15 V
533 MHz	1.05 V
450 MHz	1.05 V
300 MHz	.975 V

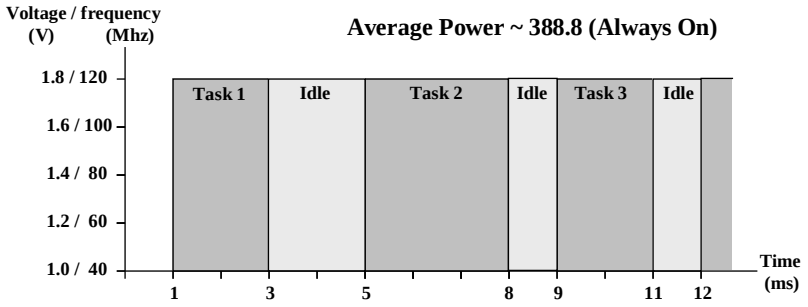
Mobile Intel PentiumIII processor

Dynamic Voltage Scaling (DVS)

- Power= CV^2f
- Reducing supply voltage V will reduce power quadratically
- Problem:
  - It incurs more propagation delay and thus, forces a reduction in clock frequency f,
  - which means longer execution time, i.e performance degradation ?

Example: Always-On

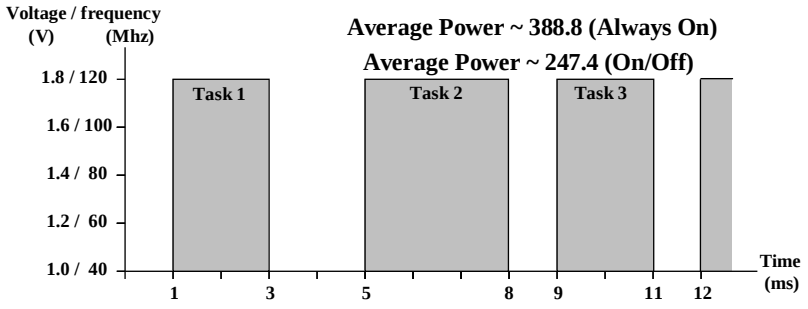
- Power is still consumed during idle time



Dynamic Voltage Scaling Techniques for Power-efficient MPEG Decoding

Example: On/Off mechanism

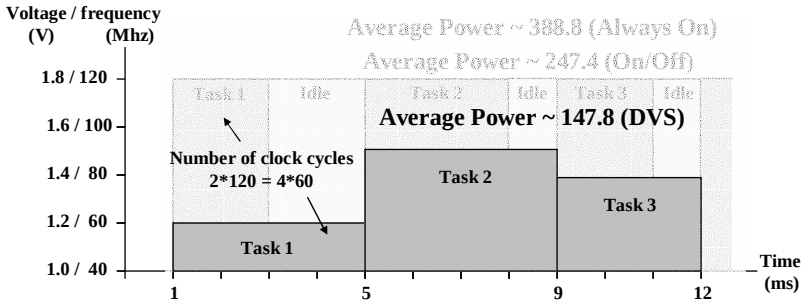
- Turn off the CPU during idle time



Dynamic Voltage Scaling Techniques for Power-efficient MPEG Decoding

Example: DVS

- DVS adjusts V and f to replace power-wasting idle time with useful execution cycles



Dynamic Voltage Scaling (DVS)

- Two questions for applying DVS :
  - Q: which applications are targeted by DVS ?
    A: DVS is best used for applications whose workload varies over time, like MPEG decoding.
  - Q: Who is going to predict the workload and adjust the voltage and frequency accordingly ?
    A: Workload can be predicted on different levels: CPU, OS, and application.
- Our scheme is an application-based DVS for MPEG decoding

Outline

- Power trends
- Power Management, Dynamic Voltage Scaling (DVS)
- MPEG
  - ✓ Overview
  - ✓ MPEG encoding
  - ✓ MPEG decoding
- DVS for
- Experin
- Results
- Conclusion

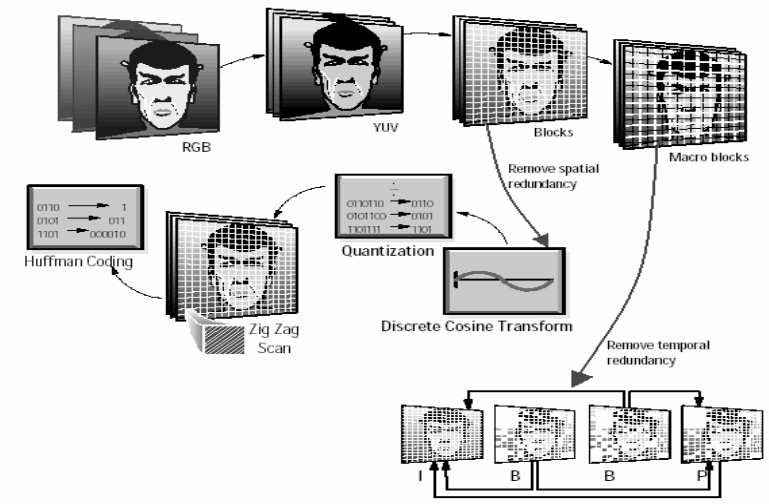
MPEG

- MPEG (Moving Picture Expert Group) is an ISO/IEC (International Organization for Standardization/International Electro-technical Commission) standard.
- MPEG video compression is used in many current and emerging products (HDTV, DVD players, video conferencing, PDAs, etc.)
- MPEG video requires:
  - Less storage space
  - Less transmission bandwidth

MPEG: Encoding

- MPEG compression relies on:
  - Eye's inability to resolve high frequency color changes
  - Spatial redundancy (or redundancy within each frame)
  - Temporal redundancy (or redundancy between adjacent frames)

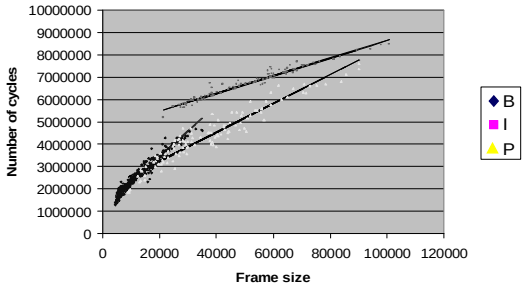
MPEG: Encoding



* "MPEG-2 : The Basics of how it Works", *Hewlett Packard Lab*

MPEG: Decoding

- MPEG decoding facts:
  - Different frame types (I,P,B) require different decoding time per byte.
  - Linear relationship between the 'number of clock cycles' and the 'frame size'.



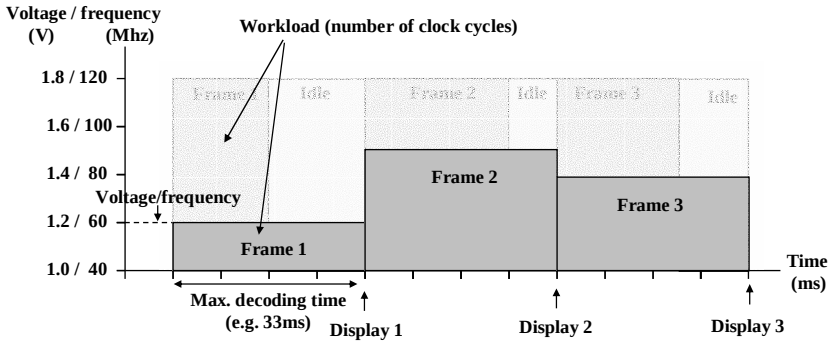
Outline

- Power trends
- Power Management, Dynamic Voltage Scaling (DVS)
- MPEG
- DVS for MPEG decoding
- Experimental framework
- Results
- Conclusion

✓ Key issues
✓ Workload estimation
✓ Voltage averaging

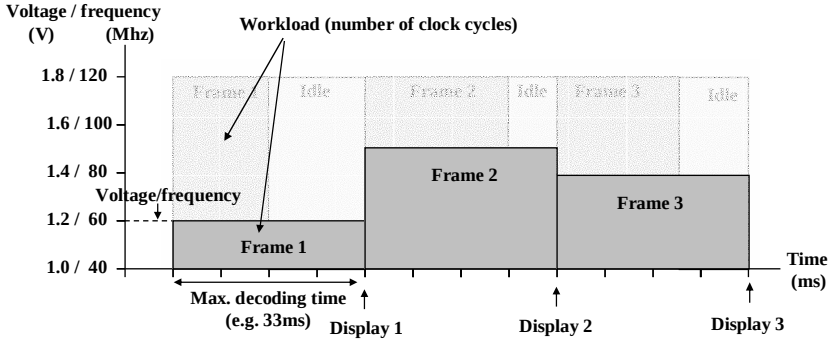
DVS for MPEG

- Key issues:
  - Accurate prediction of future workload
  - Voltage/frequency setting
  - Decode frame on time for display



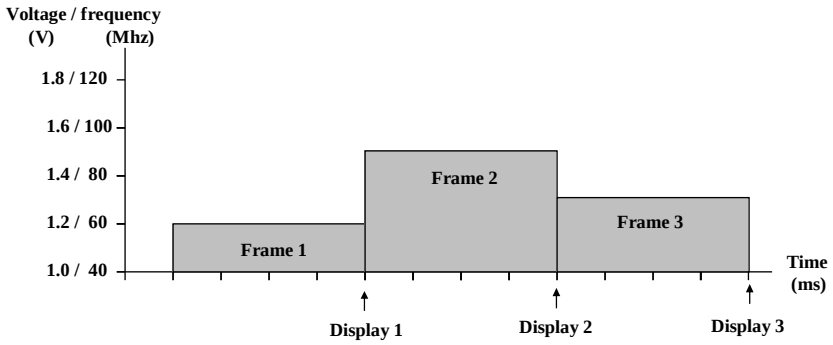
DVS for MPEG

- Solution:
  - Frame size and type ➡ Number of clock cycles
  - MPEG frame rate ➡ Max. decoding time
  - V ~ f = number of cycles / Max. decoding time



DVS for MPEG

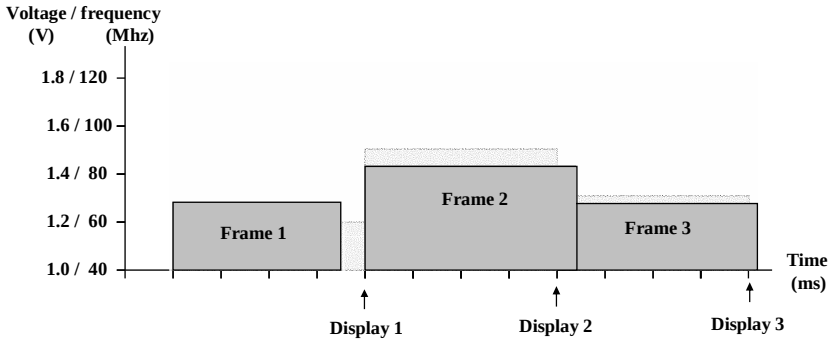
- DVS Problems: voltage/frequency are <u>estimated</u> to decode a frame just on time for display.



Dynamic Voltage Scaling Techniques for Power-efficient MPEG Decoding

DVS for MPEG

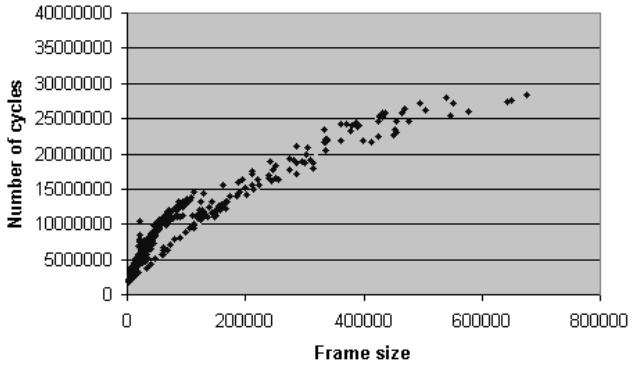
- DVS Problems:
  - Frame over-estimated ➡ more power consumed
  - Frame under-estimated ➡ miss display ➡ Dropped



Dynamic Voltage Scaling Techniques for Power-efficient MPEG Decoding

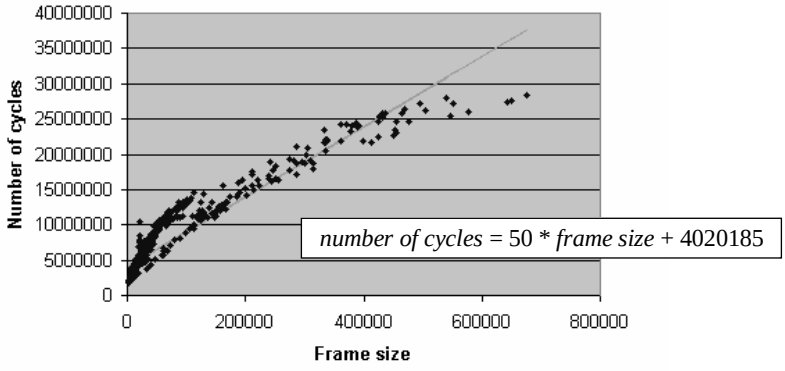
Workload Estimation

- Proposed DVS estimation techniques



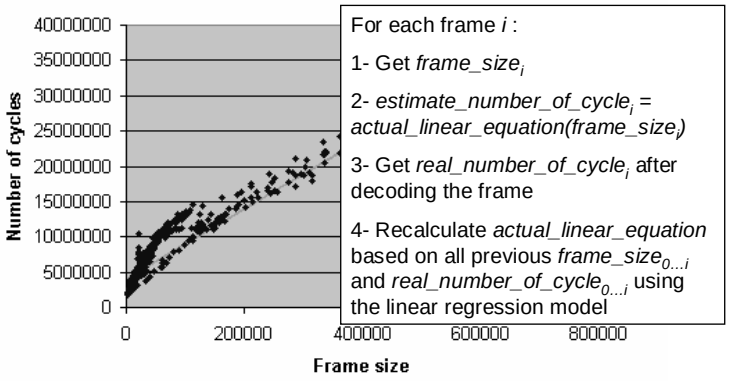
Workload Estimation: *regression*

- *Regression* (previous work)



Workload Estimation: *regression*

- *Regression* algorithm:



For each frame *i* :

1- Get $frame\_size_i$

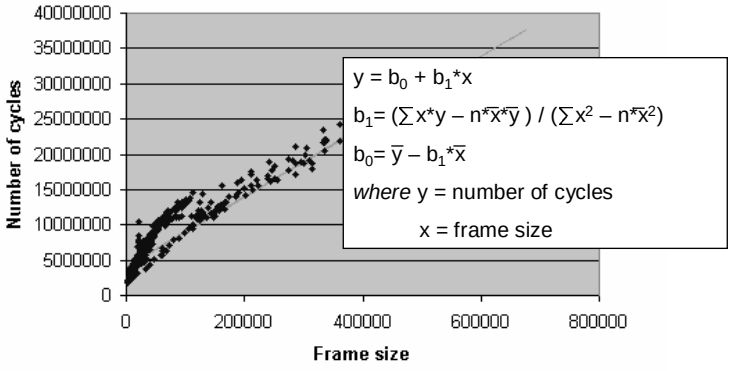
2- $estimate\_number\_of\_cycle_i$ = $actual\_linear\_equation(frame\_size_i)$

3- Get $real\_number\_of\_cycle_i$ after decoding the frame

4- Recalculate *actual_linear_equation* based on all previous $frame\_size_{0...i}$ and $real\_number\_of\_cycle_{0...i}$ using the linear regression model

Workload Estimation: *regression*

- *Regression* algorithm main problem: very computationally expensive



$y = b_0 + b_1 * x$

$b_1 = (\sum x*y - n*\bar{x}*\bar{y}) / (\sum x^2 - n*\bar{x}^2)$

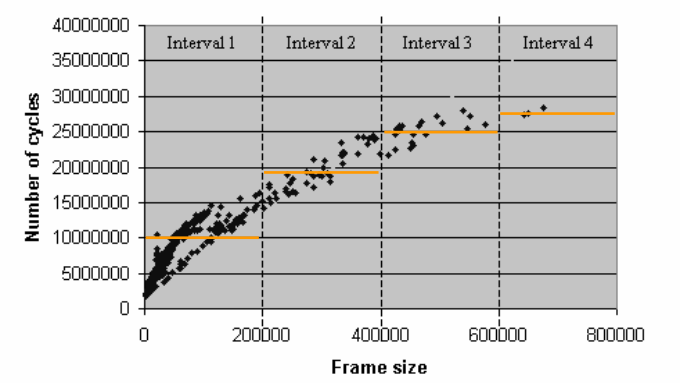
$b_0 = \bar{y} - b_1*\bar{x}$

where y = number of cycles

x = frame size

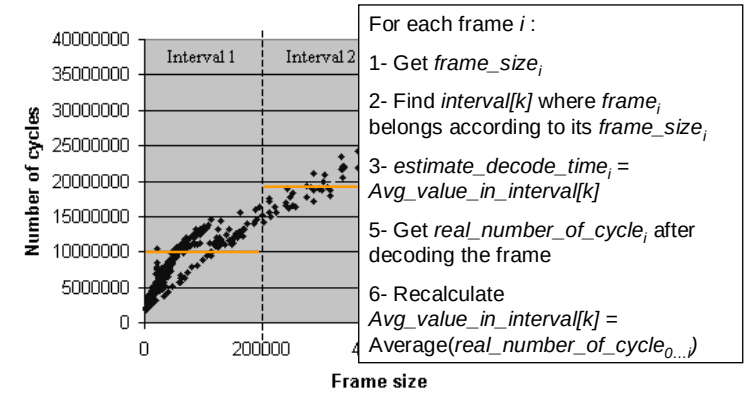
Workload Estimation: *interval-avg*

- *Interval-avg* (proposed technique) with simple average calculation in each interval



Workload Estimation: *interval-avg*

- *Interval-avg* (proposed technique) with simple average calculation



For each frame *i* :

1- Get $frame\_size_i$

2- Find *interval[k]* where $frame_i$ belongs according to its $frame\_size_i$

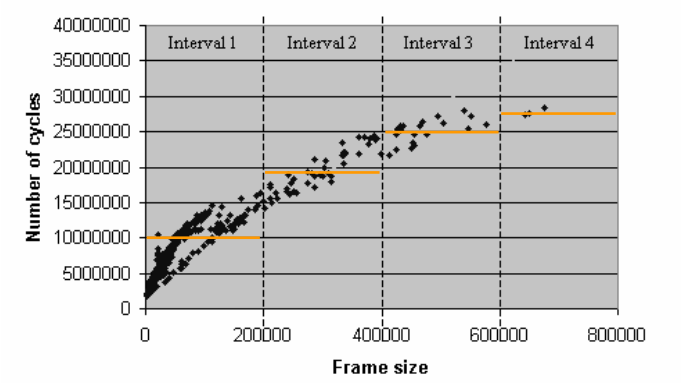
3- $estimate\_decode\_time_i$ = *Avg_value_in_interval[k]*

5- Get $real\_number\_of\_cycle_i$ after decoding the frame

6- Recalculate *Avg_value_in_interval[k]* = Average($real\_number\_of\_cycle_{0...i}$)

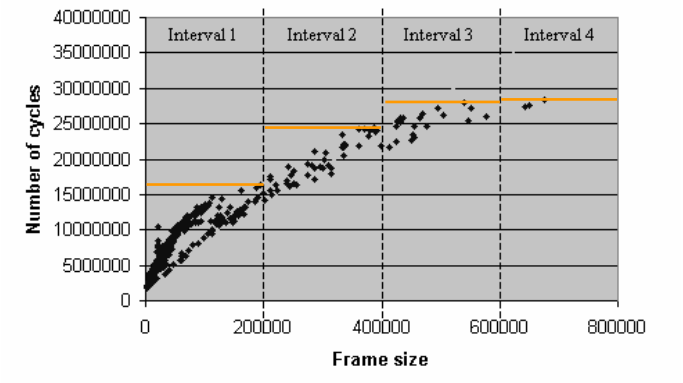
Workload Estimation: *interval-avg*

- *Interval-avg* problem: many frames will be under-estimated and dropped



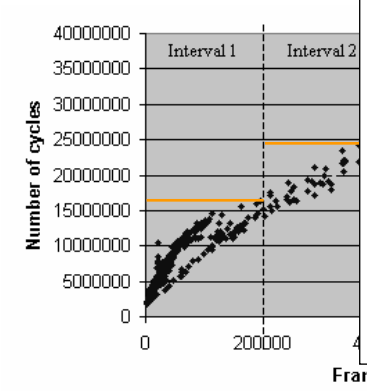
Workload Estimation: *interval-max*

- *Interval-max* (proposed technique) with simple and safe estimation



Workload Estimation: *interval-max*

- *Interval-max* (proposed technique) with simple and safe estimation



For each frame *i* :

1- Get $frame\_size_i$

2- Find *interval[k]* where $frame_i$ belongs according to its $frame\_size_i$

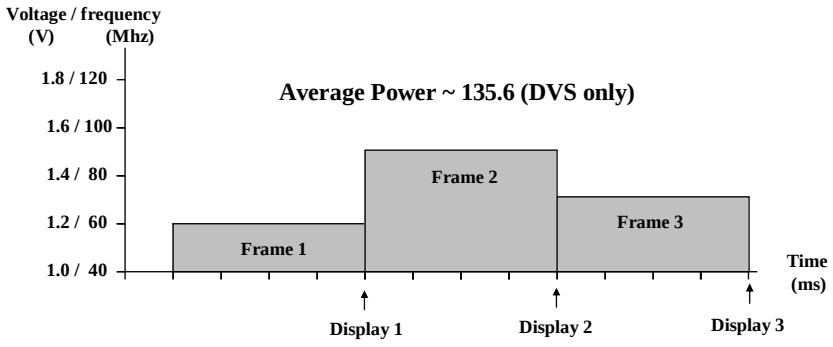
3- $estimate\_decode\_time_i$ = *Max_value_in_interval[k]*

5- Get $real\_number\_of\_cycle_i$ after decoding the frame

6- If $real\_number\_of\_cycle_i$ > *Max_value_in_interval[k]* then *Max_value_in_interval[k]* = $real\_number\_of\_cycle_i$

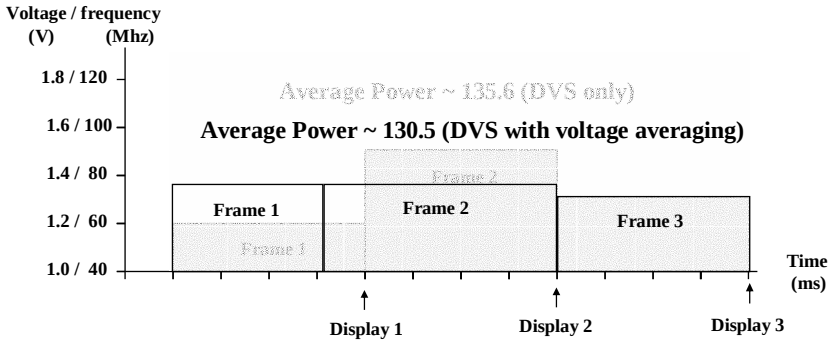
Voltage Averaging Technique

- Voltage averaging is implemented on top of the previous estimation techniques for more power saving



Voltage Averaging Technique

- Voltage averaging is implemented on top of the previous estimation techniques for more power saving



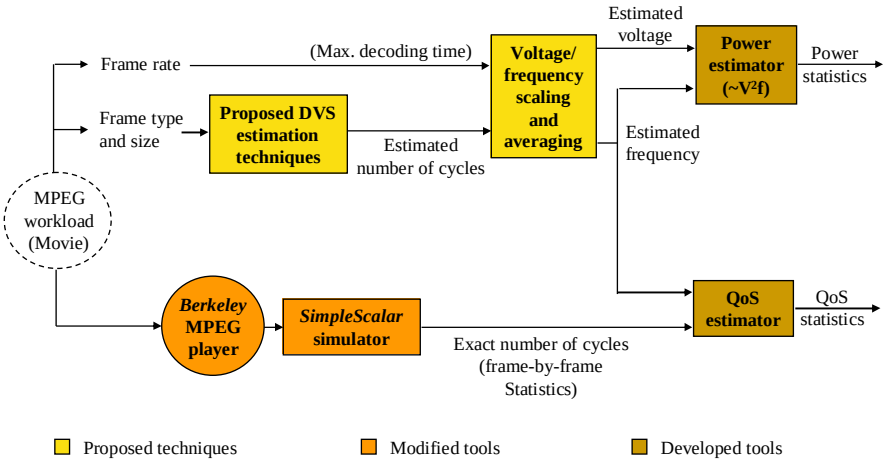
Dynamic Voltage Scaling Techniques for Power-efficient MPEG Decoding

Outline

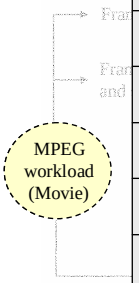
- Power trends
- Power Management, Dynamic Voltage Scaling (DVS)
- MPEG
- DVS for MPEG decoding
- Experimental framework
- Results
- Conclusion

✓ Simulation framework
✓ Simulation workload

Experimental Framework



Simulation Workload: Movie clips

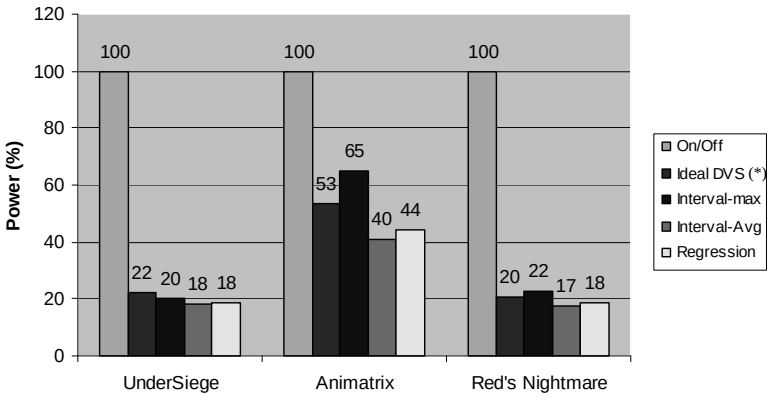


Title	*Red's Nightmare*	*The Animatrix*	*UnderSiege*
Type	Animation (low motion)	Animation (High motion)	Movie (High motion)
Frame Rate	25 fps	23.976 fps	30 fps
Resolution	320x240	592x252	352x240
I frames	41	107	122
P frames	81	428	122
B frames	1088	1070	486

Outline

- Power trends
- Power Management, Dynamic Voltage Scaling (DVS)
- MPEG
- DVS for MPEG decoding
- Experimental framework
- Results
  - ✓ Power
  - ✓ QoS
- Conclusio

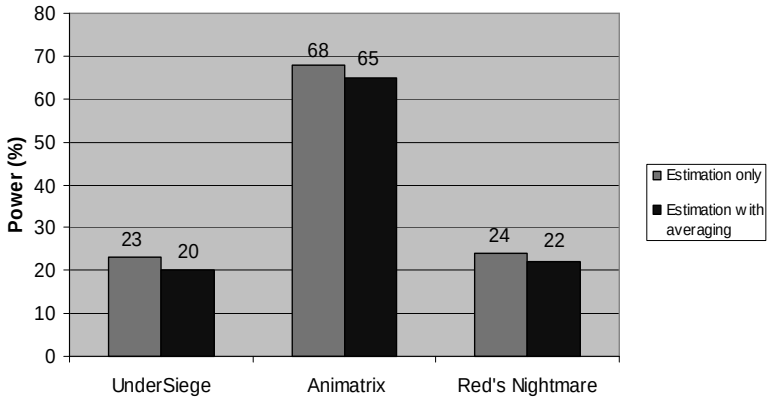
Simulation Results: Power



(*) Ideal DVS is the regular DVS with ideal perfect estimation

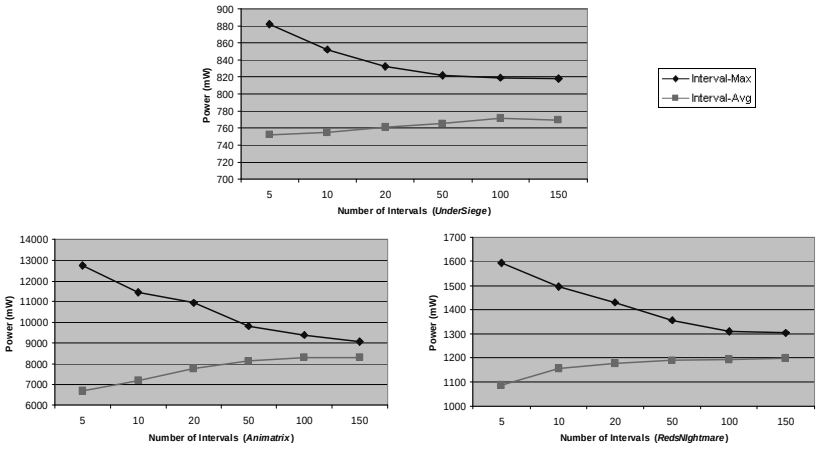
Simulation Results: Power

- Voltage averaging effect on Power

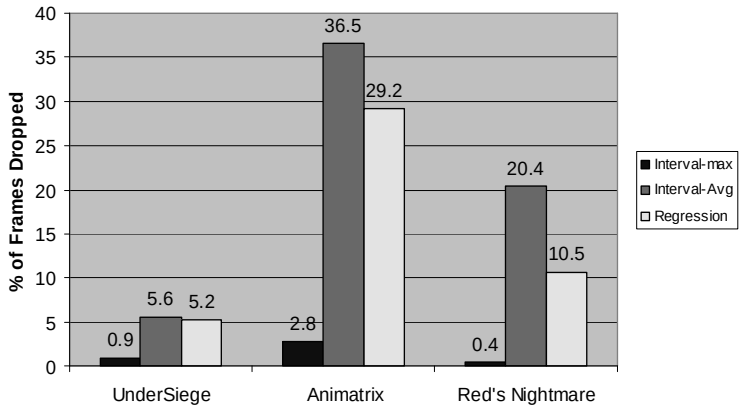


Simulation Results: Power

- Interval effect on Power

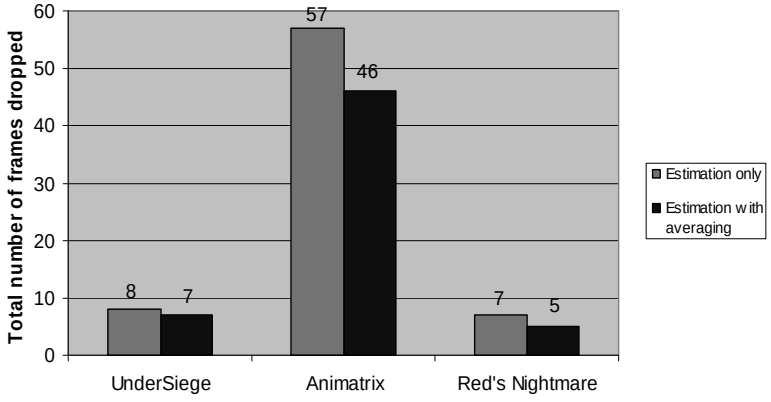


Simulation Results: QoS



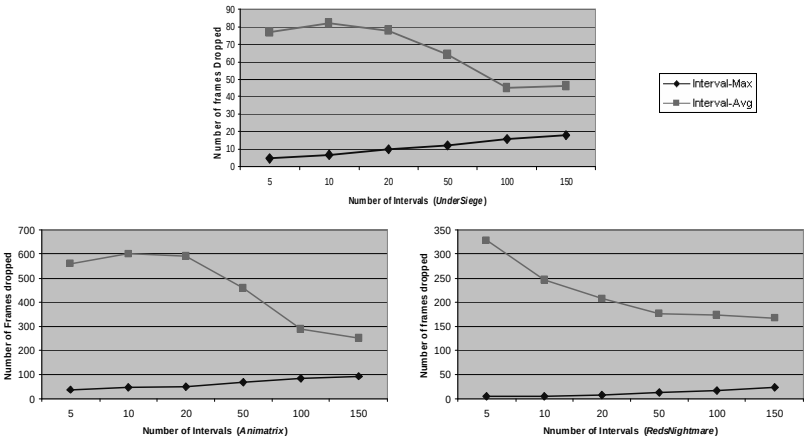
Simulation Results: QoS

- Voltage averaging effect on QoS



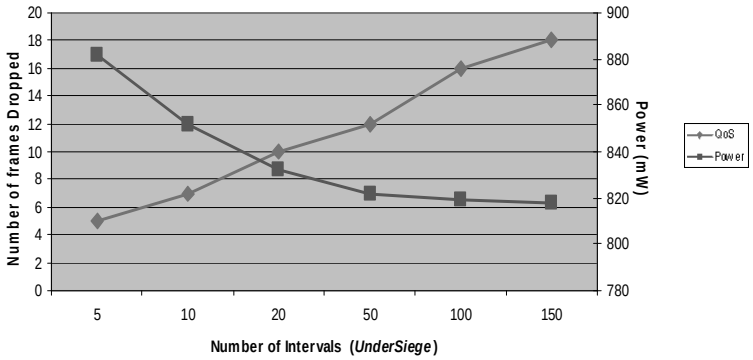
Simulation Results: QoS

- Interval effect on QoS



Simulation Results

- Power/QoS tradeoff for different interval-size



Outline

- Power trends
- Power Management, Dynamic Voltage Scaling (DVS)
- MPEG
- DVS for MPEG decoding
- Experimental framework
- Results
- Conclusion

Conclusion

- DVS estimation techniques for power-efficient MPEG decoding: *regression*, *interval-max*, and *interval-avg*
- Voltage averaging technique
- Results:
  - Up to 83% power saving over On/Off mechanism
  - As low as 0.4% frames dropped

Conclusion

- *Interval-max* algorithm, although the simplest to implement, looks to be the most promising:
  - Best QoS (~0.4% frames dropped)
  - Good power saving (~80% over On/Off)
  - Perfectly predictable when changing 'interval size' to balance Power/QoS factors
  - Best candidate for future work researching this Power/QoS tradeoffs

Thank You