Off-chip Latency-Driven Dynamic Voltage and Frequency Scaling for an MPEG Decoding

Kihwan Choi
Ramakrishna Soma
Massoud Pedram

Dept. of Electrical Engineering
University of Southern California

Outline

- Dynamic Voltage and Frequency Scaling (DVFS)
- Workload Decomposition
- Proposed Off-chip Latency-Driven DVFS Policy
- Experimental Results
- Conclusion

Background

- DVFS is a method through which variable amount of energy is allocated to perform a task
- Power consumption of a digital CMOS circuit is:

$$P = \alpha \cdot C_{eff} \cdot V^2 \cdot f$$

α : switching factor
C_{eff} : effective capacitance
V : operating voltage
f : operating frequency

- Energy required to run a task during T is:

$$E = P \cdot T \propto V^2 \quad \text{(assuming } f \propto V,\ T \propto f^{-1}\text{)}$$

- Lowering *V* (while simultaneously and proportionately cutting f) causes a quadratic reduction in *E*

Overview of Prior Work

- DVFS techniques may be classified based on three factors:
  - Constraint type : real-time (critical) vs. non real-time (non critical)
  - Scaling granularity : inter-task (coarse) vs. intra-task (fine)
  - Policy determination : static (offline) vs. dynamic (online)
- The target CPU frequency is calculated as follows:
  - Given a task with workload, W, and latency constraint, D
  - Suppose:
    - W is specified as the number of CPU clock cycles needed to complete the task
    - An inverse-linear relationship between the execution time and the CPU frequency exists, i.e., $T_{task} = W/f_{cpu}$
  - f_{target} is hence calculated as W/D (Note that T_{task} = D)

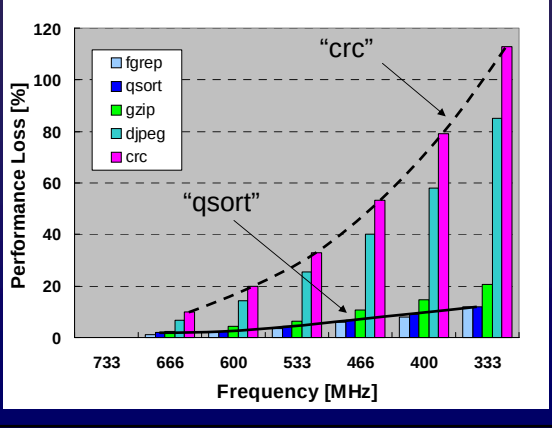
Summary of the Proposed DVFS Method

- Our proposed DVFS method for MPEG decoding is
  - *Dynamic, Intra-task* and *real-time*
- The proposed method results in significantly higher energy saving compared to the previous approaches. This is due to:
  - Accurate *estimation of the task execution time variation* as the CPU frequency is varied
  - This is in turn achieved by *decomposing the workload* into *on-chip* and *off-chip* components
  - Dynamic profiling data provided by *embedded performance monitor unit* on the CPU is used to guide the estimation

Motivation for Workload Decomposition

- CPU-bound vs. memory-bound applications
  - Shows different execution time variation according to the CPU frequency, ranging from 733MHz to 333MHz

"djpeg" & "crc" : CPU-bound "qsort" & "fgrep" : memory-bound



For CPU-bound applications, we have:

$$T_{task} = \frac{W_{task}}{f_{cpu}}$$

$$f_{cpu} = \frac{W_{task}}{D}$$

For memory-bound applications, these relations do not seem to hold

Workload Decomposition

- A program execution sequence consists of on-chip and off-chip workloads
  - On-chip workload, W_{on} : work performed inside the CPU (e.g., register-register instruction, ALU operation)
  - Off-chip workload, W_{off} : work performed outside the CPU (e.g., cache miss and subsequent access to main memory)

$$W_{task} = W_{on} + W_{off}$$

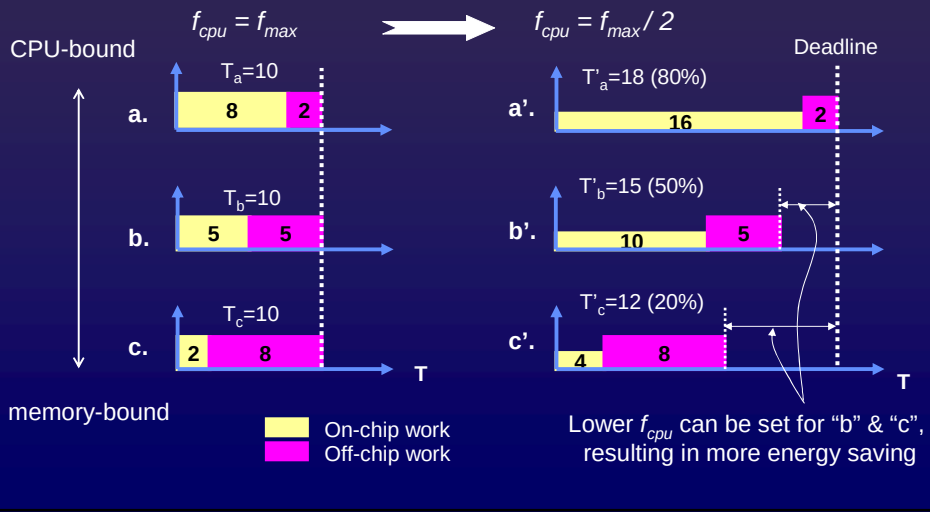
- An external memory access is asynchronous to the CPU
  - The change in the task execution time due to CPU frequency scaling is strongly dependent on the workload composition in a task

$$T_{task} = \frac{W_{on}}{f_{cpu}} + \frac{W_{off}}{f_{mem}} \longrightarrow \frac{\partial T_{task}}{\partial f_{cpu}} = -\frac{W_{on}}{f_{cpu}^{2}}$$

Variable (333MHz to 733MHz) *Fixed* (100MHz for SDRAM access)

Energy Saving as a Function of Application Type

- CPU energy can be saved with lower performance loss for memory-bound application programs



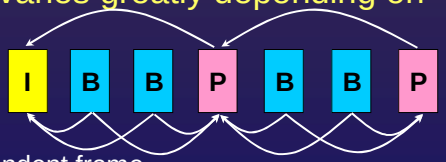
Performance Monitoring Unit (PMU)

- PMU on the XScale can provide values of some 20 dynamic events during execution of a program
  - cache hit/miss
  - TLB hit/miss
  - no. of external memory access
  - no. of instructions being executed
  - branch mis-prediction
  - data stall
- Any two events can be monitored and reported at the same time
- For DVFS policy setting in addition to
  - no. of clock counts (CCNT)

  we make use of the following event statistics:
  - no. of instructions being executed (INSTR)
  - no. of external memory access (MEM)

DVFS for MPEG Decoding

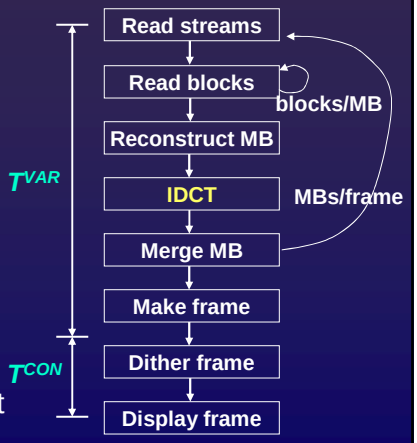
- "*Low-energy consumption*" and "high *quality of service (QoS)*" are key requirements for an MPEG decoder used in battery-powered electronic systems
- Decoding time per frame varies greatly depending on the frame type



  - Three frame types exist:
    - I-frame, which is an independent frame
    - P-frame, which has only one reference frame
    - B-frame, which has two reference frames
  - The workload generated by each frame type must be accurately estimated for DVFS to be effective
- Frame rate may be used to set the timing constraint
  - To decode a frame @10 fps, we must set the timing constraint to 100ms.

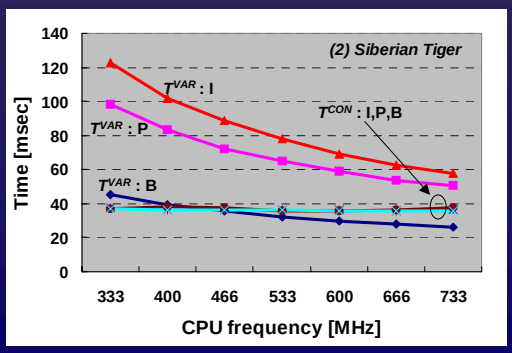
MPEG Decoding

- Memory-bound operations are dominant during the "dithering" and "display frame" steps (T^{CON})
- During the other steps, both on-chip and off-chip works are performed ($T^{VAR} = T^{ON} + T^{OFF}$)
- Divide the decoding time into:
  - T^{VAR} : CPU-frequency dependent component
  - T^{CON} : CPU-frequency independent component



Variation in the MPEG Decoding Time

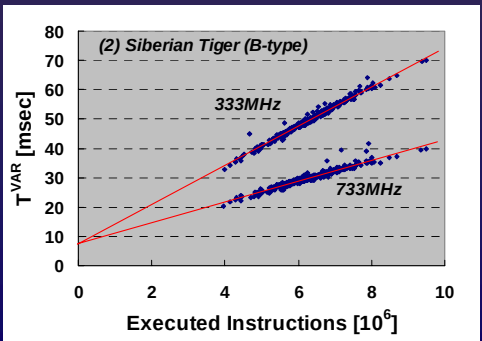
- Data for a test video clip, "*Siberian Tiger*"
- T^{CON} and T^{VAR} for the three frame types for different CPU frequencies



- Note that T^{CON} is nearly constant for all frame types
  - It is easily obtained after decoding a single frame

Extracting T^{OFF} from T^{VAR}

- T^{OFF} is independent of the CPU clock frequency
- Contour plots of T^{VAR} versus number of executed instructions, INSTR, for different CPU frequencies



Regression equation

$$T^{VAR} = T^{ON} + T^{OFF}$$
$$= \left(\frac{W^{ON}}{f_{cpu}} \right) + T^{OFF}$$
$$= \left(\frac{CPI_{AVG}^{ON} \cdot INSTR}{f_{cpu}} \right) + T^{OFF}$$

- Y-intercept of this 2-D plot gives T^{OFF}

T^{OFF} to T^{VAR} Ratio

- This table reports ratio of T^{OFF} to T^{VAR} as a percentage for different video clips

Test video	Frame size	Frame type		
		I	P	B
(1) Terminator 2	352 X 240	3.49 %	11.60 %	40.58 %
(2) Siberian Tiger	320 X 240	7.96 %	11.87 %	25.74 %
(3) Deploy	352 X 288	15.01 %	58.01 %	47.19 %
(4) Wg_wt	304 X 224	10.12 %	43.95 %	-
(5) Badboy2	480 X 208	20.64 %	38.85 %	50.76 %
(6) Final3	160 X 120	26.11 %	36.80 %	59.34 %

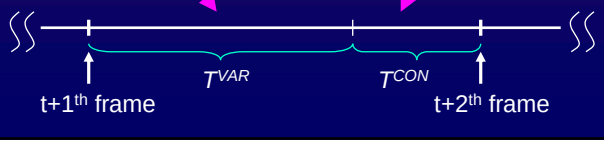
- There is frequent block data transfer for B and P type frames, so they have higher T^{OFF}
- T^{OFF} to T^{VAR} ratio varies greatly based on the decoded video clip

Proposed DVFS Policy

- Decoding time of a frame
  - $T^{VAR}+T^{CON} = (T^{ON}+T^{OFF}) + T^{CON}$
- To extract T^{OFF} from T^{VAR} for each frame
  - We maintain a regression equation for each frame-type and use a moving-average or weighted-average of INSTR statistics to predict INSTR for the next time slot
- We set the CPU frequency during T^{VAR} by equation given below. Note that we set the lowest CPU frequency, f_{min}, during T^{CON}

$$f_{cpu}^{t+1} = \frac{INSTR_{t+1}^{EXP} \cdot CPI_{AVG}^{ON}}{D - T^{CON} - T_{EXP}^{OFF}}$$

$$f_{cpu}^{t+1} = f_{min}$$



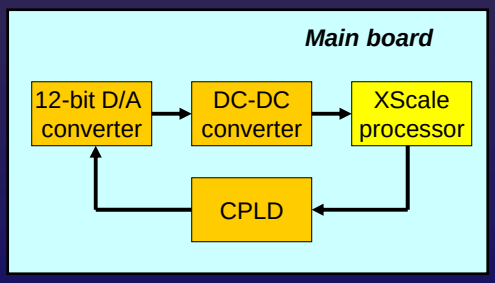
Inter-frame Error Compensation

- The QoS constraint e.g., the frame rate is also important in the MPEG decoding
- Workload prediction is not always perfect
- Error diffusion
  - If a (positive or negative) slack exists in the current frame, we will diffuse it into the next frame
  - This scheme can result in local QoS variation, but meets a global QoS target
- Target CPU frequency during T^{VAR} with error compensation:

$$f_{cpu}^{t+1} = \frac{INSTR_{t+1}^{EXP} \cdot CPI_{AVG}^{ON}}{D - T^{CON} - T_{EXP}^{OFF} + T_t^{SLACK}}$$

Implementation (I)

- Block diagram of variable voltage generator

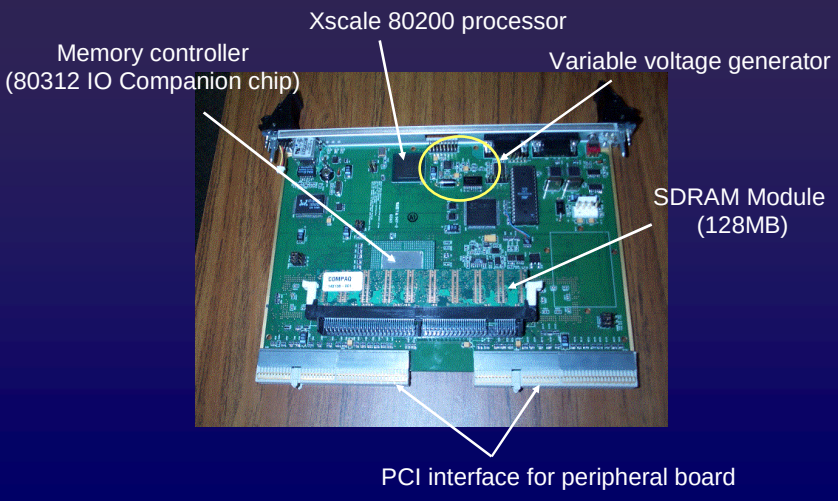


Frequency [MHz]	Voltage [V]
333	0.91
400	0.99
466	1.05
533	1.12
600	1.19
666	1.26
733	1.49

- A public software, *"mpeg_play"*, from UC-Berkeley was adopted and modified

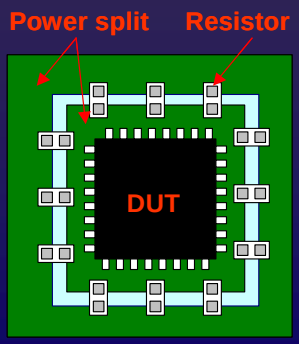
Implementation (II)

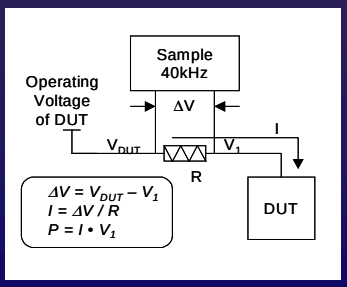
- Apollo Testbed II – Main board (USC, SNU)



Implementation (III)

- Power measurement with DAQ (Data Acquisition board)

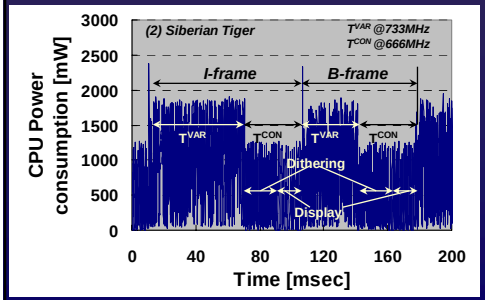




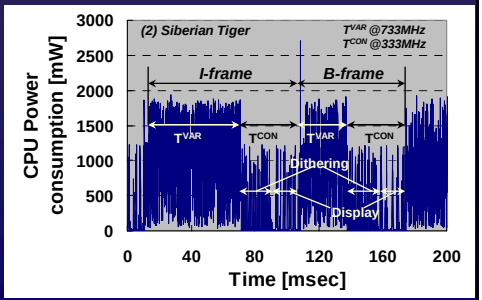
Experimental Results (I)

- Decoding time and power consumption at different CPU frequencies and voltage levels

T^{VAR} @733MHz, T^{CON} @666MHz



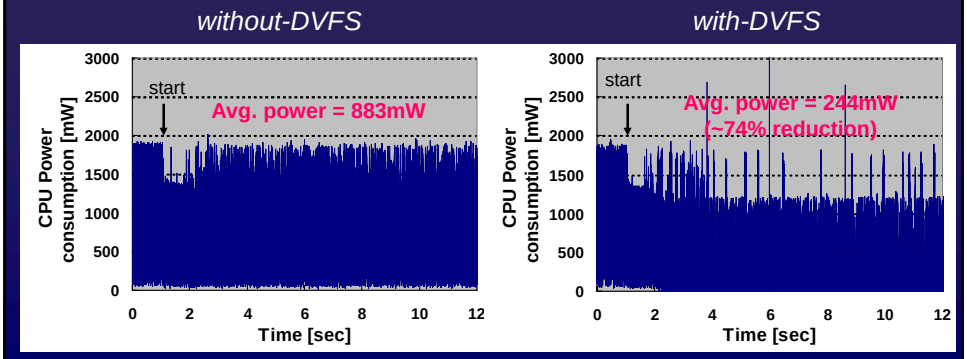
T^{VAR} @733MHz, T^{CON} @333MHz



avg. power consumption during T^{CON} : 510mW to 186mW (64% reduction)

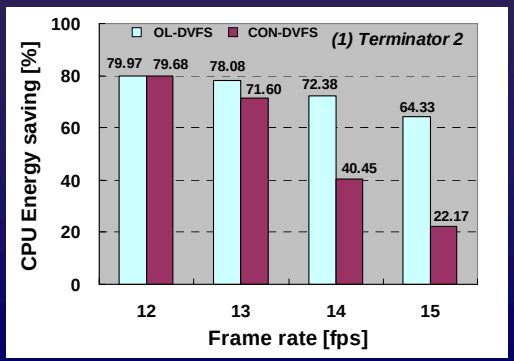
Experimental Results (II)

- CPU power consumption with the proposed DVFS
  - "Terminator 2" @ 14fps



Experimental Results (III)

- CPU energy saving comparison
  - OL-DVFS : Off-chip Latency-Driven DVFS
  - CON-DVFS : Conventional DVFS (no workload partitioning)



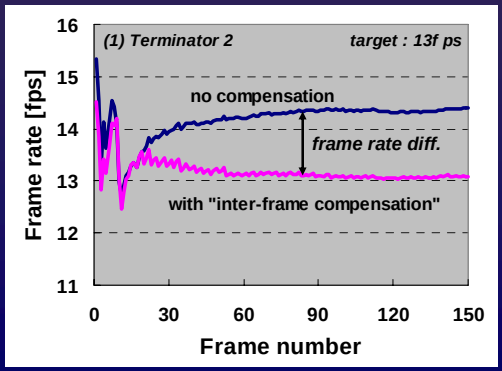
Experimental Results (IV)

- OL-DVFS vs. CON-DVFS
  - Numbers in parenthesis of the first column are for (6)

fps	(1) Terminator		(2) SiberianTiger		(3) Deploy		(4) Wg_wt		(5) Badboy2		(6) Final3	
	CON	OL	CON	OL	CON	OL	CON	OL	CON	OL	CON	OL
10	-	-	73.15	77.78	-	-	-	-	-	-	-	-
11(27)	80.46	80.75	55.49	71.39	-	-	-	-	-	-	80.88	82.62
12 (28)	79.68	79.97	43.39	60.66	-	-	-	-	79.33	79.45	82.04	82.63
13 (29)	71.60	78.08	25.36	49.54	-	-	75.27	77.74	78.85	79.48	81.85	81.96
14 (30)	40.45	72.38	-	-	57.94	75.69	60.59	73.18	71.34	75.16	81.65	81.99
15	22.17	64.33	-	-	35.53	64.44	41.33	66.99	46.99	61.64	-	-
16	-	-	-	-	4.24	61.41	28.23	57.84	-	-	-	-

Experimental Results (V)

- Frame rate variation
  - With the proposed "*inter-frame compensation*", the target frame rate is achieved with lower computational workload



Conclusion

- An off-chip latency driven DVFS technique for an MPEG decoding was proposed and implemented in an XScale-based platform
  - On-chip and off-chip workloads are separated at run time using dynamic profiling data from an embedded hardware unit
  - To guarantee a global QoS for MPEG decoding, a novel inter-frame compensation technique based on inter-frame error diffusion was proposed
- Based on actual current measurements in the testbed
  - Significant CPU energy saving ranging from 50% to 80% was achieved