Architectural Exploration Using Verilog-Based Power Estimation: A Case Study of the IDCT

Thucydides Xanthopoulos, Yoshifumi Yaoi, Anantha Chandrakasan
Department of Electrical Engineeering and Computer Science
Massachusetts Institute of Technology, Cambridge

ABSTRACT

We describe an architectural design space exploration methodology that minimizes the energy dissipation of digital circuits. The centerpiece of our methodology is a Verilog-based power estimation tool, Pythia, that blends the accuracy of low-level circuit simulators such as powermill with the speed of high level power estimators geared to design exploration. Pythia takes into account voltage-dependent capacitive nonlinearities and supports runtime adaptation of supply voltage. It employs a hybrid modeling aproach in which low-level simulation of logic gates and flip-flops can be combined with high level macromodels for memory structures where the energy per access is not as sensitive to input data statistics. The speed and accuracy of Pythia has enabled a detailed case study of two different approaches for the computation of the Inverse Discrete Cosine Transform, an integral component of the MPEG video coding algorithm. One approach uses conventional methods and the other exploits signal statistics to dynamically minimize the average number of operations required and the operating supply voltage.

1. Introduction

Low power design has become increasingly important due to the recent proliferation of portable electronic devices. One approach to reduce the power dissipation in a signal processing system is to exploit the distribution of the data that the system operates on. The computational algorithm as well as the architectural implementation are optimized based on the properties of the input data stream so as to minimize the total number of operations required.

One example where this technique can be applied is the Inverse Discrete Cosine Transform (IDCT) unit of a typical MPEG video decompression system. Compressed video data contains a large percentage of zero-valued spatial frequency (DCT) coefficients. The skewed distribution of frequency domain video data is the primary reason for the use of transform coding in the MPEG compression standard. Zeroes can be transmitted in a run-length coded form with a high degree of compression. The histograms in Figure 1 show the relative occurence of non-zero coefficients for two typical MPEG-2 sequences. The average in both cases is between 6 and 7 non-zero coefficients per 8x8 block.

In the study that follows we will investigate the energy dissipation of two different approaches for the IDCT: The Data-Driven IDCT (DDIDCT) and a conventional approach based on Distributed Arithmetic (DA) principles. The energy dissipation of the DDIDCT approach exhibits strong correlation with the number of non-zero coefficients in the compressed data stream and employs techniques such as adaptive supply voltage and clock period as well as global clock gating to exploit temporal variations in the computational load. The DA approach exhibits a smaller correlation with the number of non-zero DCT coefficients and always performs a fixed number of operations per block.

We have developed a Verilog-based power estimator, Pythia, which estimates the total energy dissipated during a simulation by monitoring all signal transitions. Pythia, as opposed to other similar transition-based power estimators takes capacitance vs. voltage nonlinearities into account. It can monitor and filter glitches. Furthermore, it supports changing the supply voltage during simulation runtime which enables modeling of adaptive supply systems. Pythia is very accurate when compared to transistor-based power simulators such as powermill and requires much less CPU time (10x-100x). As a result, it has enabled this study which requires both accuracy and speed since a large number of input vectors must be simulated to determine data dependencies in the energy dissipation of both systems under consideration.

2. Case Study: Conventional vs. Data-Driven IDCT

This section describes the two IDCT architectures under consideration, identifying some of the requirements on the power estimation tool.

2.1 Distributed Arithmetic

Conventional approaches to the computation of the 2-Dimensional IDCT on an 8x8 block use row column decomposition: First, the 8-point 1D IDCT of each row is computed, then the result is transposed and the 8-point 1D IDCT of each column is computed. This process is equivalent to a 64-point 2D IDCT of the entire 8x8 block. Such a system is shown in Figure 2. Our first approach falls into this category.

The computation of the 1D IDCT on a block row or column is essentially a multiplication of an 8-element vector (DCT coefficients) with an 8x8 matrix (reconstruction constants). Distributed Arithmetic [1] (DA) is a bit-serial computational operation that

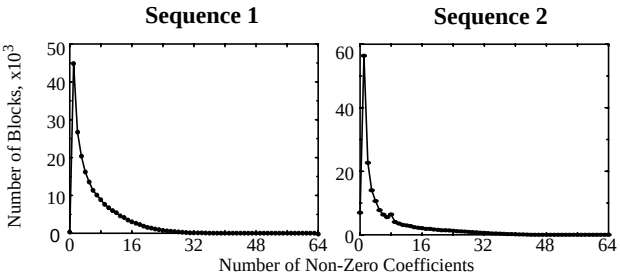


Figure 1: Histograms of Non-Zero DCT Coefficients

DAC 97, Anaheim, California
(c) 1997 ACM 0-89791-920-3/97/06 ..$3.50

Figure 2: IDCT with Row-Column Decomposition

performs the dot product of two vectors (one vector must be a constant) in parallel without the use of a multiplier. Linear combinations of the constant vector elements are stored in a ROM which is addressed by the elements of the data vector in a bit-serial fashion. An accumulator with an arithmetic shifter are used to accumulate the partial sums. After a number of cycles equal to the bitwidth of the data vector elements, the accumulator register contains the dot product of the two vectors. This simple mechanism is illustrated in Figure 3 for a dot product of two four-element vectors.

Figure 4 shows in detail our implementation of the 1D IDCT using an optimized DA-based approach [2]. Input DCT coefficients are clocked into the ppreg{0-6} registers before being parallel-loaded into the parallel-in serial-out registers (psreg{0_0-7_1}). There are 8 Distributed Arithmetic processing units each one consisting of 2 ROMs, one adder that combines the contents of the ROMs and one accumulator with an arithmetic shifter. This scheme employs a well known DA technique which involves splitting the ROM in two halves to reduce the latency of the computation and the ROM size. In this case. the result of the dot product appears in the accumulators in a number of cycles equal to half the bitwidth of the incoming coefficients. After all eight dot-products have converged in the accumulators (A0-A7), they are combined once more through 4 adders and 4 subtractors before being loaded into the final row of shift registers. After this stage, the 8-element block row will be written in the transposition memory (TRAM) (Figure 2). When the TRAM is full, the block will be read out in transposed form and will be fed into a similar 1D IDCT structure which will compute the IDCT on the columns.

This IDCT approach *does not* exploit the skewed distribution of video data. This algorithm performs a fixed number of additions and memory operations per block independent of the number of zero coefficients present.

2.2 Data-Driven IDCT

The second architecture is based on a direct algorithm (as opposed to row-column decomposition) proposed by McMillan and Westover [3]. This algorithm can be formulated as follows:

$$\vec{x} = y_0\vec{C}_0 + y_1\vec{C}_1 + \ldots + y_{63}\vec{C}_{63} \quad \textbf{(1)}$$

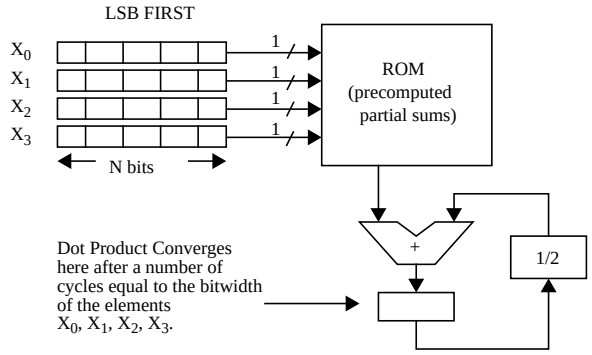


Figure 3: Illustration of Dot Product Using DA

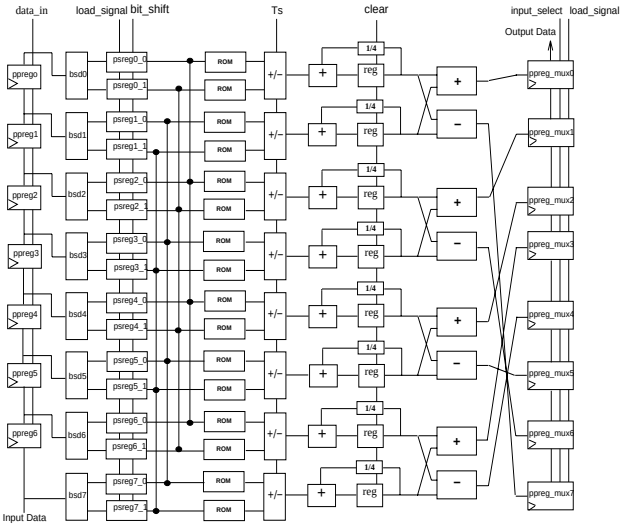


Figure 4: 1D IDCT Distributed Arithmetic Unit

where $\vec{x}$ is the 64-element reconstructed image block, $y_0, y_1, \ldots, y_{63}$ are the DCT coefficients (mostly zeroes) and $\vec{C}_0, \vec{C}_1, \ldots, \vec{C}_{63}$ are 64-element constant reconstruction vectors.

As seen from Equation 1, each DCT coefficient is treated individually. Using this formulation combined with the observation that multiplication with a zero is a NOP, we can adaptively minimize the number of switching events by processing only the non-zero coefficients. The DDIDCT architecture exploits this fact to minimize energy dissipation.

A block diagram of this approach is shown in Figure 5. Data is pushed into one of two FIFOs while coefficients are read and processed from the FIFO not currently being written to (ping-pong buffer.) Only non-zero values are pushed into the FIFOs and they are annotated with their position (0-63) within the 64-element block. Coefficients are represented in 12-bit sign magnitude form. When read from the FIFO, the coefficients are scaled with the appropriate constant recostruction vector $\vec{C}_0, \vec{C}_1, \ldots, \vec{C}_{63}$ whose element values are looked up in a ROM indexed by the coefficient position in the block. The scaled values are presented to an array of 64 accumulators under local distributed control. Each accumulator chooses to add, subtract or disregard the particular scaled coefficient presented based on its control logic. Each non-zero coefficient needs between 1 and 10 cycles (this depends on the redundancy in the reconstruction vectors of Equation 1) for scaling and accumulation in the result registers. As a result, the multiplier and the 64 accumulators must be clocked faster than the incoming sample rate.

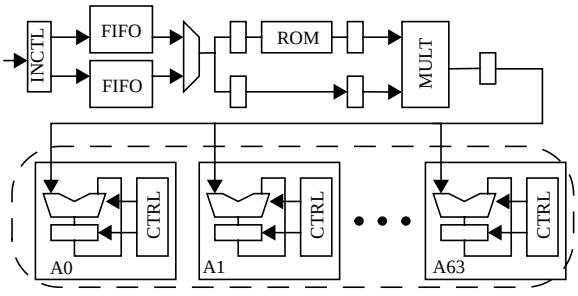


Figure 5: Data-Driven IDCT Block Diagram

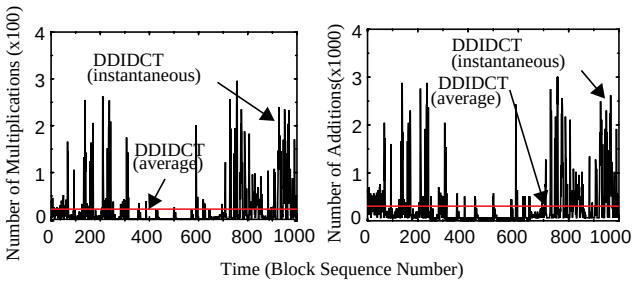


Figure 6: Number of Ops/Block in the DDIDCT Approach

Figure 6 shows the number of multiplications and additions performed per block for 1,000 blocks of compressed video data. Although a direct comparison cannot be made with the DA approach which does not involve multiplication, if we assume that one multiplication is equivalent to about five additions we see that the DDIDCT performs 514 operations per block on average as opposed to 2176 for the DA approach. This indicates that that the DDIDCT may be a promising approach for energy efficiency. Further and more accurate investigation is therefore necessary.

The DDIDCT algorithm exhibits a variable number of operations per block. This can be exploited in two ways for energy efficiency. Either the system can be powered down (clock gating) when the processing load is less than the peak, or the measure of processing load can be used as a control input to an embedded DC-DC converter which will dynamically adjust the system power supply. When the supply drops, the propagation delays within the system will increase. A variable clock must also be employed in order to track the delay of the critical path. The combinations of supply voltage and clock period must be picked in such a way so that the system always finishes the computation before the next sample becomes available. The feasibility of such an approach has been demonstrated in [4], [5]. We chose to adopt the variable supply idea for the implementation of the DDIDCT.

For implementation reasons, the power supply and the clock do not vary continuously with the computational load. Instead, there are 4 discrete values between 1.2 V and 3.3 V that the supply can take, and four clock periods (0.4 f_{sample}- 6.5 f_{sample}). Figure 7 shows the energy savings achieved vs. the case of simply gating the clock when when the load is less than the peak. The staircase function is the curve that the system follows given that the power supply changes at four discrete values. If the computation finishes before the next block becomes available (this occurs because of the four discrete clock periods available) , the system is powered down. Given our choice in placing the 4 thresholds, we see that the system curve follows closely the ideal curve in the region where the block frequency is highest.

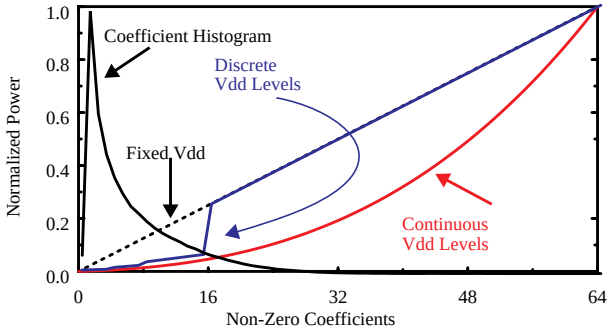


Figure 7: Energy Savings with Variable Supply Voltage

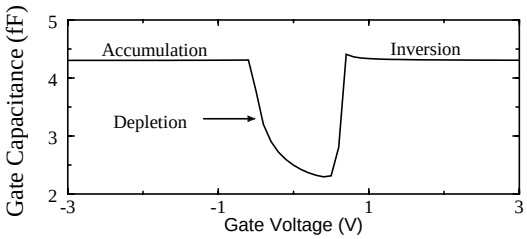


Figure 8: Min Size NMOS CV Plot (0.5µm process)

3. Power Estimation

The two systems described above are reasonably large (~200,000 transistors) and need to be simulated for a large number of compressed video blocks so that conclusions regarding correlation of energy dissipation vs. processing load can be drawn. The need for a fast and accurate power simulator for such cases has motivated us to develop Pythia.

Pythia works with a structural Verilog circuit description. It uses a collection of Verilog-PLI access and utility routines to traverse circuit hiearchy, extract user-provided parameters and monitor signal transitions. Each net in the design is annotated with four different capacitance values:

1. A gate capacitance value indicating the total gate capacitance attached to the net.
2. An nMOS drain capacitance value indicating the total n-drain to p-substrate or p-well junction capacitance attached to the net.
3. A pMOS drain capacitance value indicating the total p-drain to n-well or n-substrate junction capacitance attached to the net.
4. A routing capacitance indicating the total metal routing capacitance of the net.

During each net transition, there is a different energy contribution from each particular capacitive component. Only the routing capacitance is assumed to remain constant throughout the node voltage swing.

A unique feature of Pythia compared to other switched-capacitance based power estimators is that it takes capacitance vs. voltage non-linearities into account and it treats each component of a capacitive net differently instead of lumping all components into one big capacitor. This is necessary since a gate capacitance has a different voltage dependence than a junction capacitance. This approach has resulted in increased accuracy without compromising speed.

3.1 Capacitance vs. Voltage Nonlinearities

Figure 8 shows a typical minimum size transistor CV plot for a 0.5 µm process. Gate capacitance is clearly not constant over the voltage swing from 0 to V_{DD}. Pythia maintains a table of precomputed coefficients corresponding to various power supplies such that :

$$C'_{OX} = aC_g(V_{DD}) = \frac{1}{V_{DD}}\int_0^{V_{DD}} C_g(V)\,dV \qquad \textbf{(2)}$$

where α is the precomputed coefficient for each power supply V_{DD} and $C_g(V)$ is the gate unit area capacitance (voltage variable as shown in Figure 1). C'_{OX} is the effective value of gate unit area capacitance used in subsequent calculations.

While computing the gate capacitance, Pythia makes the assumption that the transistors are mostly in the linear region of

operation. This is true when the power supply is significantly larger than twice the transistor threshold voltage V_T. If we accept this assumption then the total gate-to-channel capacitance (WLC'_{OX}) must be equally divided between the source and the drain. Then, the drain component is multiplied by a factor of 2 to account for the large signal version of the Miller effect. Therefore, the total gate capacitance referred to ground used in equation is:

$$C_{gate}(V_{DD}) = \frac{1}{2}WLC'_{ox} + 2 \times \frac{1}{2}WLC'_{ox} = \frac{3}{2}WLC'_{ox} \quad (3)$$

Drain and source overlap capacitances (C_{GDO}, G_{GSO}) are also added to the total capacitance seen at the gate. These capacitances are constant throughout the swing and are not included in the integral of Equation 2. The drain overlap is multiplied again by a Miller factor of 2 before being referred to ground:

$$C_{gate-total} = C_{gate}(V_{DD}) + C_{GSO} + 2 \times C_{GDO} \quad (4)$$

Junction capacitances constitute a significant component of the total circuit switched capacitance and they also exhibit non-linear variation with voltage. Each design net is annotated with two types of juction capacitance: N-drain to p-substrate and p-drain to n-well. Both capacitances are referred to ground but they vary differently with voltage due to differences in technology parameters.

The energy drawn from the power supply while charging the n-drain capacitance is given by the following formula:

$$E_{drain} = \int V_{DD} i(t)\,dt = V_{DD}\int \frac{d}{dt}q(t)\,dt = V_{DD}\int_0^{q(V_{DD})} dq \quad (5)$$

To compute the energy, we need to find the total charge on the junction drain capacitance when the voltage across it reaches V_{DD}:

$$V_{DD}\int_0^{q(V_{DD})} dq = V_{DD}\int_0^{V_{DD}} C(V)\,dV \quad (6)$$

The voltage variable junction capacitance is given by the following formula as a function of voltage:

$$C(V) = C_{j0}\left(1 + \frac{V}{\phi_0}\right)^{-M} \quad (7)$$

where C_{j0} is the junction zero-bias capacitance, ϕ_0 is the junction built-in potential and M is a positive number <1 representing the doping gradient of the junction.

Solving the integral of Equation 6 yields:

$$E_{junction} = V_{DD} \times \frac{C_{j0}\phi_0}{-M+1}\left[\left(1 + \frac{V_{DD}}{\phi_0}\right)^{-M+1} - 1\right] \quad (8)$$

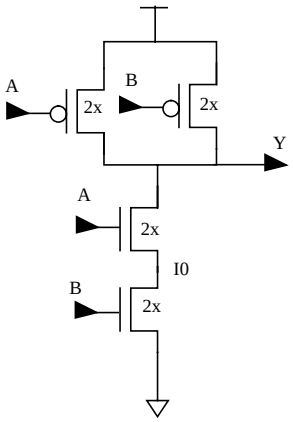
Equation 8 is applied to both area and perimeter parasitics for nMOS and pMOS devices.

All technology parameters (C_{j0}, M, t_{OX}, ϕ_0 etc.) are read from spice level 1 models for the target process.

3.2 Verilog Description and Cell Library Annotation

The Verilog circuit description must comply with two important requirements:

1. The entire description must be 100% structural
2. The lowest level Verilog modules must be Verilog cells (defined between ‘celldefine and ‘endcelldefine statements.)




```
‘celldefine

module nand2 (Y, A, B);
output Y;
input A;
input B;

nand #1 (Y, A, B);

specify
specparam GateCap$A=4;
specparam GateCap$B= 4;
specparam nDrainCap$Y=2;
specparam pDrainCap$Y=4;
specparam PythiaID= “NAND2”;
endspecify

endmodule

‘endcelldefine
```

Figure 9: Verilog Description of a NAND2 Gate

Moreover, they should only contain built-in or user-defined Verilog primitives.

The Verilog-XL interpreter after parsing the netlist creates an internal representation of the circuit. Verilog provides a collection of access and utility routines for accessing and extracting information out of the internal circuit representation. Pythia relies on these routines to descend the hierarchy, build a list of all the circuit nets annotated with capacitance values and place probes that will cause a function callback every time there is a change of value in a design net.

Every Verilog cell must be annotated with certain terminal attributes. The user must specify the total gate capacitance attached to each cell input. Moreover, the total n-drain to p-substrate and p-drain to n-well junction capacitances for each output must be specified. All these attributes are specified in terms of multiples of a minimum-sized transistor. Technology specific information will be used for the derivation of the exact capacitance values.

Figure 9 shows an annotated Verilog cell of a NAND2 gate along with the corresponding schematic. Inputs A and B are attached to four units of gate capacitance (one 2x nMOS and one 2x pMOS). Output Y is attached to 2 units of n-drain capacitance (one 2x nMOS) and 4 units of p-drain capacitance (two 2x pMOS). The PythiaID parameter links this Verilog cell to an internal representation that models the switching of internal node I0 (Figure 9) so that this switched capacitance can be taken into account in the power estimation. Pythia takes into account reduced swing in internal nodes (i.e. node I0 swings only up to V_{DD}-V_T).

3.3 Changing the Power Supply at Simulation Runtime

Pythia supports changing the power supply during simulation runtime. This is achieved by making the supply and the voltage swing a local property of each Pythia net structure. When a $change_supply([module], [new_supply]) Verilog call is encountered, the net structures for all relevant nets are accessed and the local properties are altered. Extensive hashing and cacheing techniques are used to speed the process of net structure retrieval for large designs. The $change_supply call can be invoked on different modules within the Verilog netlist to model multiple different supply nets on the chip.

3.4 Hybrid System Modelling

Pythia employs a hybrid approach in modeling the system under power evaluation. Memory-based structures (RAMs/ ROMs) are modelled at a higher level since such modules are most often

modelled behaviorally in Verilog for simulation speed purposes and no information regarding each capacitive node is available to Pythia. Pythia models memory structures as black boxes that include tabulated data which can be easily used to compute energy per access..

V_{DD}	Words	Bits	Blocks	HSPICE	PYTHIA
1.5 V	256	8	1	28.80 pJ	25.65 pJ
2.2 V	256	8	1	59.85 pJ	58.62 pJ
3.3 V	256	8	1	137.91 pJ	136.34 pJ
5.0 V	256	8	1	348.28 pJ	317.22 pJ

Table 1. Comparison of Pythia vs. Hspice Results for ROM Energy Dissipation

Table 1 compares energy per access numbers produced by a Pythia high level ROM model to HSPICE simulation data.

3.5 Power Estimation Accuracy

Table 2 compares results from a powermill run on the entire DDIDCT system using an extracted 200,000 transistor netlist with a Pythia simulation of the same system with the same test vectors at the Verilog gate level. A fixed supply of 3.3V and an input sample rate of 10 Msamples/sec has been used for both simulations. During this simulation run, three 8x8 blocks of video data have been fed into the system.

	Average Power	Execution Time
Powermill	16.137 mW	10,925 sec
Pythia	15.8 mW	475 sec

Table 2. Comparison of Powermill and Pythia Results

Pythia comes remarkably close to the power number reported by powermill. It is slightly lower because it does not include leakage and short circuit power dissipation which powermill takes into account.

Figure 10 shows a profile of the power supply current for 200 ns of simulation time for both tools. Current bursts occur at about 10 ns intervals because the DDIDCT system runs on an internal 50 MHz clock. We see that the current profiles are very similar. Pythia thinks that transitions occur a few nanoseconds later than powermill does. This is expected since Pythia works with user defined gate delays at the Verilog level which are conservative in this case. The area under the current plots though which represents average power dissipation is very similar.

4. Simulation Results

Both architectures described in Section 2 have been implemented in schematic form using a standard cell library appropriately annotated for Pythia. Verilog netlists have been automatically extracted from the schematics and simulations have been run using compressed video data as input stimuli.

Figure 11 shows the power dissipation profile for the two architectures for a sequence of 10 blocks of video data. The simulation has been run for 700 clock cycles. A new block is fed into both systems every 64 cycles. The DA has a power supply of 2.2 V and the DDIDCT runs from a fixed 3.3 V supply. A sample rate of 10 Msamples/sec is applied to both systems. The bursts in the DDIDCT power profile occur as a result of clock gating when the accumulator array finishes the computation before the arrival of the next block due to the four discrete clock frequencies available. The instantaneous power dissipation is higher than the DA case, because a faster worst case internal clock is required. As seen from the figure, power dissipation is strongly correlated with the number of non-zero coefficients in the block being processed. In the case of 64 non-zero coefficients, the entire block time (64 cycles) is required to complete the transform computation and power consumption is maximized.

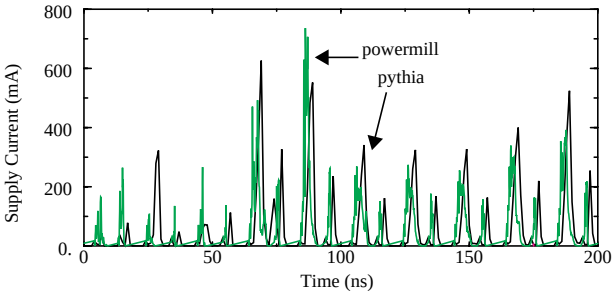


Figure 10: Supply Current Profile from Pythia and Powermill

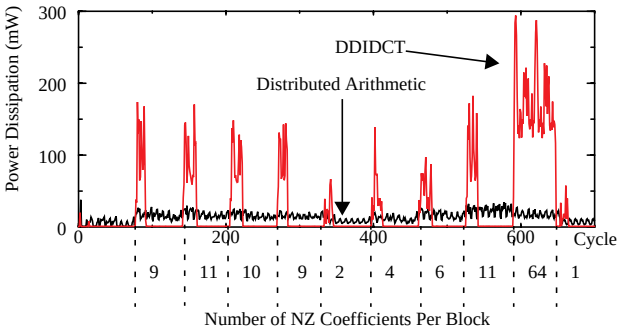


Figure 11: Power Dissipation Profile for the 2 Architectures (Fixed V_{DD})

On the other hand, the DA approach exhibits negligible power consumption correlation with the number of non-zero coefficients since it always performs a fixed number of operations per block, regardless of signal statistics. Power dissipation does not exhibit bursts and is almost constant throughout the entire time available for computation. No clock gating is employed in this approach.

Figure 12 shows the same data as Figure 11, but this time we have enabled the adaptive supply modeling in Pythia. The DDIDCT supply voltage does not stay fixed at 3.3V but instead assumes four discrete values (1.2V, 1.9V, 2.6V, 3.3V) which depend on the number of non-zero coefficients present in the block processed. As we see from the figure, this has the effect of lowering the power bursts for 9 out of 10 input blocks. In addition, blocks 5, 7 and 10 exhibit smaller peak power dissipation than the DA case. We conclude from these preliminary experiments that the Data-Driven IDCT can be more energy efficient than the mainstream Distributed Arithmetic approach if the average number of non-zero coefficients per block in the video stream is small enough.

In order to quantify the previous conclusion, we picked 64 blocks out of an MPEG-2 video sequence containing 1,2,..., 64

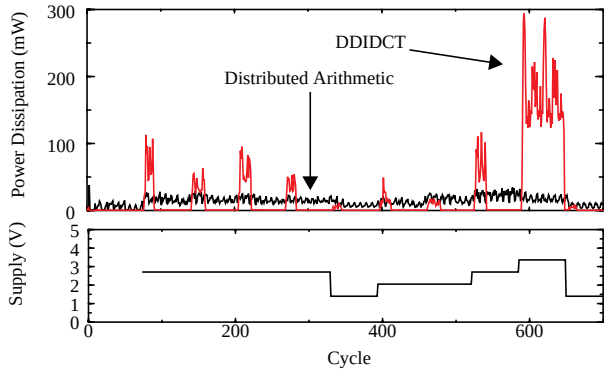


Figure 12: Power Dissipation Profile for the 2 Architectures (Variable V_{DD})

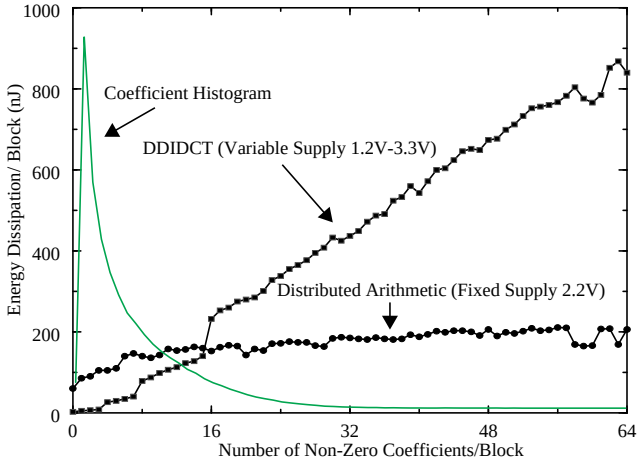


Figure 13: Pythia Results for Energy Dissipation vs. NZ Coefficients for both IDCTs

non-zero coefficients respectively and fed them into both systems. Figure 13 plots energy dissipation vs. number of non-zero coefficients for both approaches. This plot confirms our expectations about energy dissipation correlation with computational load. The DA approach does not exhibit a strong correlation. Energy dissipation is slightly reduced in the presence of a large number of zero coefficients. This is because although the total number of operations per block is fixed, the presence of zero inputs reduces global switching and the total switched capacitance is minimized. Energy dissipation quickly reaches a relatively steady state after 10 non-zero samples. The reason that the steady state is reached with a few non-zero samples is that most zeroes are absorbed after the first 1D IDCT computation. The second stage sees a matrix of mostly non-zero elements and the average number of global switching events is unaffected.

On the other hand, the energy dissipation of the DDIDCT approach is strongly correlated with the processing load. In this simulation the DDIDCT has a variable supply voltage of four discrete levels (1.2 V, 1.9 V, 2.6 V, 3.3 V). The corresponding load thresholds are set at 2, 4, 8, and 16 non-zero coefficients respectively. Abrupt discontinuities can be observed in the diagram around these thresholds. The energy curve is not entirely monotonic because the energy dissipation in the DDIDCT does not only depend on the number of non-zero samples but also on the position of these samples within the block. Different coefficients usually require a different number of scalings which depends on the degree of redundancy of each reconstrunction vector in Equation 1.

We observe that the two energy curves cross at a processing load of 15 DCT coefficients. Within the range of 1 to 15 DCT coefficients, energy savings ranging from a factor of 32.5 to a factor of 1.14 may be achieved with the DDIDCT approach vs. the conventional DA approach. On the same figure we have overlayed the block frequency diagram of Sequence 1 in Figure 1 to show that the majority of blocks fall into the region of energy savings. The average number of non-zero coefficients per block for this sequence is 6.37.

The conclusion of our case study is that the Data-Driven approach can yield substantial energy savings for highly compressed video sequences. For an average number of 6-7 non-zero coefficients per block, energy savings of a factor of 3 to 4 can be achieved based on our experimental data. An interesting application for such a system may be a low power video terminal used for playback of high quality coded video produced with sophisticated motion estimation techniques that yield a high degree of coding efficiency. This IDCT approach though is not suited for random DCT data or poorly coded video which results in a large number of DCT samples transmitted to the decoder. In such a case, the conventional Distributed Arithmetic approach should be used instead.

5. Conclusion

In this paper we have presented a power estimation methodology for digital systems. We have developed a Verilog based power estimation tool called Pythia which uses the Verilog-XL simulation engine to accumulate energy dissipation due to switched capacitance on all circuit nets. Unlike similar switched-capacitance based power estimators Pythia takes nonlinear voltage variable capacitances into account (gate and junction capacitances) for improved accuracy. Moreover, it supports runtime adaptation of supply voltage and can also model multiple supply voltages per system.

The speed and accuracy of Pythia has enabled us to compare two large (~200,000 transistors) digital systems that compute the 2D IDCT on an 8x8 data block for a large number of test inputs. Using Pythia's results we drew important conclusions about the weaknesses and strengths of each system from the perspective of power consumption.

6. Acknowledgment

This research is sponsored by SHARP Corporation. Thucydides Xanthopoulos is supported by a National Semiconductor graduate fellowship. The authors wish to thank Debashis Saha for his efforts in making Pythia available over the World Wide Web.

7. References

[1] S. White, "Applications of Distributed Arithmetic to Digital Signal Processing: A Tutorial Review", IEEE ASSP Magazine, pp. 4-19, July 1989.

[2] S. Uramoto et al. "A 100 MHz 2-D Discrete Cosine Transform Core Processor", IEEE Journal of Solid State Circuits, pp. 492-498, April 1992.

[3] L. McMillan and L. A. Westover, "A Forward Mapping Realization of the Inverse Discrete Cosine Transform", Proceedings of the Data Compression Conference (DCC '92), March 1992.

[4] V. Gutnik and A.P. Chandrakasan, "An Efficient Controller for Variable Supply Voltage Low Power Processing", Symposium on VLSI Circuits, June 1996.

[5] G. Wei and M. Horowitz, "A Low Power Switching Power Supply for Self-clocked Systems", International Symposium on Low Power Electronics and Design, pp. 313-318, 1996.