Dynamic Power Reduction In Integrated Chips Using Inertial Filtering

Sarat Chandra dubba (m.tech) **K. Rajasekhar m.tech**

Abstract *– We propose a new method for dynamic power reduction in integrated integrated chips using techniques of inertial filtering. In a CMOS circuit ,the total energy consumption per signal transition at a given node with capacitance C is* $0.5CV^2$*. Since this is independent of the charging or discharging resistance, the signal delay can be varied without affecting the power consumption by varying the resistance. Keeping the gate delays, internal to standard cells, fixed we determine the values of necessary routing delays to eliminate all glitches by inertial filtering. To implement these delays we insert the required values of resistances through customized feedthrough cells. In spite of the increased resistance in the circuit, the overall power is reduced because the resistive delays suppress glitches without increasing the* $0.5CV^2$ *power per transition, and the critical path delay is unchanged. For the ISCAS '85 benchmark circuit, c2670, we achieve a 30% saving in average power consumption with 14% increase of the chip area. This saving is about the same as reported for a previously published custom design method for minimum dynamic power.*

.

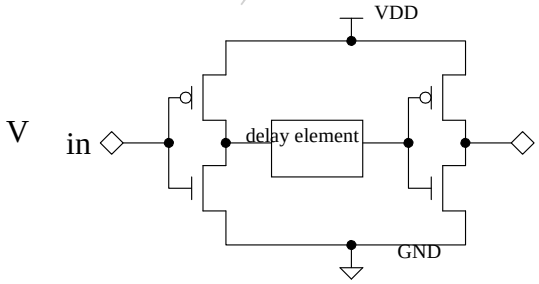
1.0 Introduction

Dynamic power is a major component in the overall power dissipation of a CMOS circuit. It can be reduced by minimizing the number of transitions of signals. Besides the logic transitions, glitches (or hazards) also consume power. Reducing these can save up to 30 to 70% of the total power dissipation. A general solution to glitch elimination involves gate delay manipulation to balance paths and to filter hazards as discussed in recent papers by Agrawal (1997), Agrawal *et al.* (1999) and Raja *et al.* (2002, 2003, 2004). These methods are not applicable to standard-cell ASICs, where delays of library cells cannot be arbitrarily changed. We design a resistive feedthrough cell whose delay can be customized. Using this, we accomplish path delay balancing and hazard filtering for ASICs. For details the reader may refer to Uppalapati (2004). Most

of the reported work on standard cell ASICs, i.e., by Dragone *et al.* (1998), Fisher *et al.* (1996), Hashimoto and Onodera (2001), Masgonty *et al.* (2001), Scott and Keutzer (1994), Sulistyo and Ha (2002), and Zhang *et al.* (2001), ignores the glitch suppression aspect of low-power design.

2.0 Design of a Delay Cell

To implement a delay element as standard cell that can fit into a 0.25µ CMOS cell library used in this work, we analyzed several delay elements. These were an inverter pair, an n-diffusion capacitor, a polysilicon resistor and a CMOS transmission gate. Among these, the inverter pair and the transmission gate have been used before by Mahapatra *et al.* (2000) and Raja *et al.* (2004). The delays of these elements are controlled by changing the *W/L* ratio of the transistors. The formulation might become non-linear, since changing the *W/L* ratio of a MOSFET changes the channel resistance as well as the associated parasitic capacitances. This motivated us to look at delay elements that are entirely capacitive or entirely resistive so that the RC time constant of the element can be controlled linearly. Thus, we tried an n-diffusion capacitor and a resistor made from polysilicon wire with a silicide blocking mask over it. The silicide blocking mask increases the resistivity of the polysilicon resistor. To simulate a realistic situation, each delay element is driven by an inverter gate and is also loaded with an inverter. The circuit set up is shown in Figure 1. The connection for the n-diffusion capacitor is different from that depicted in the figure. The capacitor is connected such that it increases the load capacitance of the driving inverter. The circuit simulator used to measure the delay and power consumption is the Spectre™ circuit-level simulator from Cadence (2003). The results of this experiment are presented in Figure 1. The inverter and the transmission gate cells are made of transistors with minimum sizes. The resistor cell was designed for a resistance of 15.4kΩ and the capacitor cell was designed for a capacitance of 2.7fF.



Element	Delay(ns)	Metric1	Metric2
Inverter pair	0.28	0.22	0.03
Diffusion capacitor	0.31	0.23	0.05
Poly resistor	0.44	0.33	0.11
CMOS trans. gate	0.35	0.22	0.16

Figure 1: Evaluation of delay elements.

The *average delay* in the table of Figure 1 is the average of the rise and fall delays of the delay element alone. This value is calculated as the difference

between the delay of the entire circuit shown in Figure 1 and the delay of the same circuit without the delay element. The delay is expressed in nanoseconds. The delay element should also be optimized for size and power consumption. We use two metrics for the circuit of Figure 1, delay/power (metric 1) and delay/area (metric 2). The delay values used to calculate both metrics are given in column 2 of the table in Figure 1. The power consumption values used to calculate metric 1 are expressed in µ*W*. Since the delay elements are implemented as standard cells with fixed height, the area is measured in terms of the number of grid units along the width. We observe that the polysilicon resistor introduced the maximum amount of average delay. Since the polysilicon resistor also had the highest metric 1 value, we can conclude that this element introduces maximum delay per unit power consumption of the circuit in Figure
1. Similarly, metric 2 indicates the amount of delay introduced per unit area overhead. Though the transmission gate had the highest metric 2 value, since saving power is our primary objective, we chose the polysilicon resistor in our effort to reduce the glitch power. We call this delay element a *resistive feedthrough cell*. At logic level this element was modeled as a fictitious buffer. This cell was designed as a parameterized cell, i.e., the physical design of the cell is generated on-the-fly according to the required delay.

Resistors are relatively easy to integrate and have been used extensively in the past to build analog circuits as discussed by Hastings (2001). The resistance of a rectangular slab of length *L*, width *W* and thickness *t* can be calculated in terms of a material specific constant called resistivity ρ, as $R=\rho L/(Wt)$. Integrated resistors consist of diffusions or depositions that can be modeled as films of constant thickness. It is therefore customary to combine resistivity and thickness into a single term called the *sheet resistance* R_s. The formula can be written as: $R = R_s\ L/W$, where $R_s=\rho/t$. The *L/W* ratio is given the units of *squares* and R_s of a material is given in units of Ω per square. Layout designers generally maintain uniform width and control the resistance by varying the length. This is done to avoid two main factors that affect the integrated resistor -- *width bias* and *non- uniform current flow*. The details of these effects can be found in a book by Hastings (2001). This practice has an indirect advantage that the delay across an integrated resistor can be formulated as a linear function of the length. Resistors are often folded in a serpentine manner employing rectangular turns. The rectangular turns are easy to draw and the spacing between the turns is easily adjusted. A rectangular corner adds a resistance of approximately 1/2 square.

In a CMOS process, a resistor can be implemented using metal layers, the polysilicon layer, diffusion layers or the n-well layer. Among these materials the polysilicon forms the best resistor. The narrower polysilicon pitch will result in a smaller layout and they are less susceptible to parasitics. However, the polysilicon used for constructing MOS gates is heavily doped to improve conductivity resulting in smaller sheet resistance. Most CMOS processes provide a *silicide blocking mask* to block the doping and increase the sheet resistance. For a 0.25µ CMOS process, we found that the silicided poly has a sheet resistance of 3.6Ω/square but with a silicide blocking mask the sheet resistance is 173.6Ω/square. We implemented the feedthrough cell as a serpentine path of poly with a silicide blocking mask over it. The width of the path is kept to a minimum. This cell is designed as a parameterized cell so that

delay can be varied continuously. The delay is controlled by varying the length of the poly wire. Since the delay depends only on the length of the poly wire, we get a linear formulation. Also, we get very high resolution since the length can be varied in steps of the minimum feature size (λ) of a particular technology. For the 0.25µ CMOS technology with λ=0.15µ, we found that the resistance of the feedthrough cell could be varied in steps of 95Ω. The layout of a resistive feedthrough cell to give a resistance of 20kΩ is shown in Figure 2. Since the polysilicon resisitive cell has a regular geometry, the automation of the physical design is fairly simple and these cells can be easily implemented as parameterized cells with the length of the polysilicon wire as the parameter. Compared to a library where all logic gates are implemented as parameterized cells, this approach is economical in design effort.

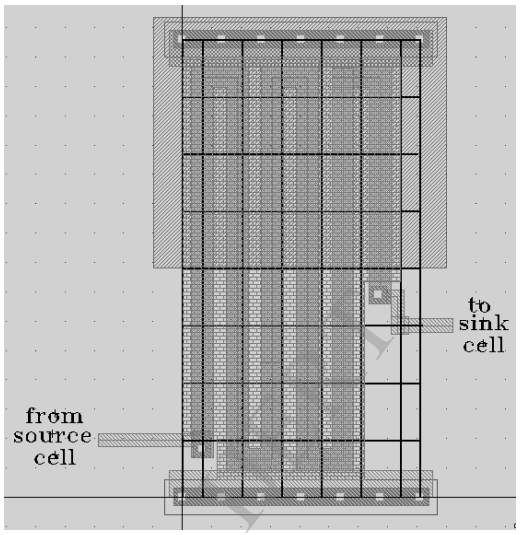


Figure 2: Layout of a resistive feedthrough cell.

3.0 Chip-Level Design

We introduce an extra low power design step into the regular standard cell design flow to produce a glitch-free design. The extra step uses a linear program to determine signal delays and then uses the resistive feedthrough cells to realize those delays at the circuit level. The resistive feedthrough cell insertion step is incorporated into the flow between the technology mapping and the place-and- route phases. The procedure of Figure 3 is explained below.

We modified the linear program (LP) of Raja *et al*. (2002, 2003) to treat gate/cell delays as fixed parameters but left the buffer delays as variables. The linear program gives an optimal distribution of these delays, which are actually the signal (routing) delays in Figure 3. The objective function of the modified LP still is to minimize the sum of the signal delays but the underlying implication of this function is different. Since these delays are implemented as resistors, they do not introduce extra transitions into the circuit. However, the feedthrough cells used to implement the signal delays will contribute to an

increase in the die area of the circuit. By minimizing the total signal delay, the increase in area is kept to a minimum. We infer that the result of the LP is indeed a globally optimal solution because of its strong similarity to Raja *et al*. (2002, 2003). We observed that the linear program gives the same solution even if the constraint on the overall delay of the circuit is relaxed. This observation can be theoretically explained with the help of the following theorem.

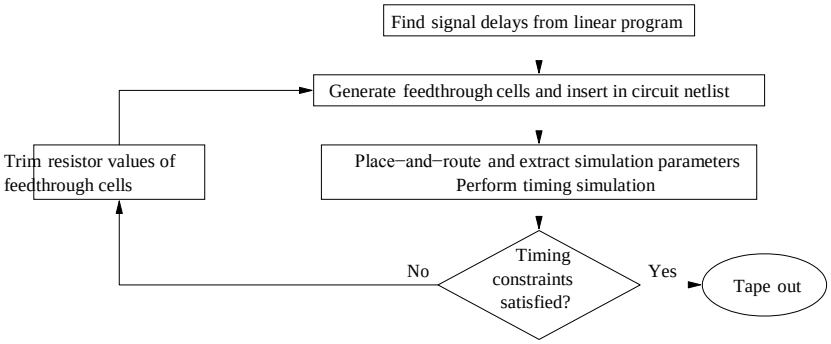


Figure 3: Resistive feedthrough cell insertion.

Theorem: *If the overall circuit delay is constrained not to increase, then a minimum transient energy design cannot be guaranteed by only increasing the gate output delays. See* Agrawal *et al*. (1999).

We infer that if the overall circuit delay is not to increase, then the only way to achieve a minimum transient energy design is by increasing the wire delays (or gate input delays). In our method glitches are removed by adjusting the wire delays. These wire delays are added only on non-critical paths since the goal is to equalize the signal arrival times at the inputs of each gate within the inertial delay of the gate. Thus, we can say that, if there is no upper bound on the wire delays, our modified linear program is capable of producing a minimum transient energy design without increasing the overall circuit delay.

After the LP finds the necessary signal delays, we determine the resistor values that will introduce those delays on signal lines. Delay across a resistor depends on the value of the resistor itself as well as the capacitive load at its output. The propagation delay of the *RC* circuit can be calculated from the Elmore delay formula (see Rabaey *et al.* (2003)): $t_p=0.69R_{eq}C_L$, where R_{eq} is the amount of resistance that is implemented using the feedthrough cells and C_L is the load capacitance that is seen by the feedthrough cell. The value of C_L depends on the number of pairs of transistors that the cell is feeding into (a pair consists of one *n*MOS and one *p*MOS transistor) since all of the standard cells are static CMOS designs. So, once we know the load capacitance and the propagation delay necessary we can calculate the feedthrough cell resistance using the Elmore formula. Though it is a reasonable linear approximation, for better accuracy we used data from the simulation of an *RC* circuit over a wide range of resistor and capacitor values. The data was stored as a three dimensional lookup table that could be used to find the resistance value for any given delay and load capacitance values. The delay model is shown in Figure 4 with the load capacitance, line resistance, and delay on *x*, *y* and *z* axes, respectively.

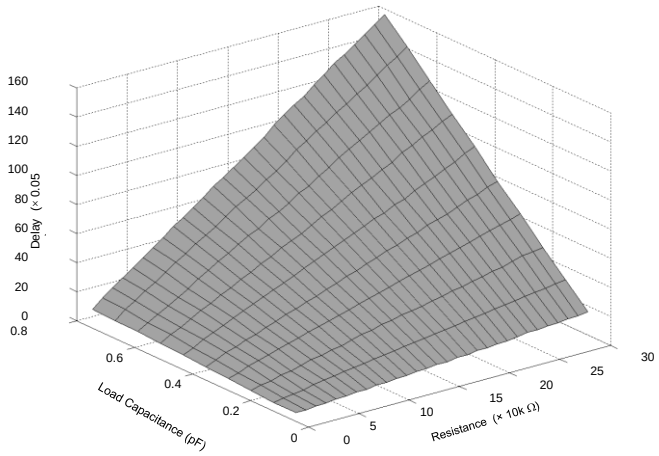


Figure 4: Delay model of the feedthrough cell.

A program first finds the load capacitance for the type of gates that are on the load. It then scans the delay values for that particular load and compares it with the required delay. If that delay falls between two data points then the program uses linear interpolation to find the resistance value.

4.0 Glitch-Free Physical Design

The resistances found from the lookup table are automatically designed as standard cells according to the procedure described above. These feedthrough cells are inserted into the original circuit by modifying a *Verilog* netlist of the circuit. The place-and-route layout of the modified netlist is then done using a commercial tool such as *Silicon Ensemble*^{TM} (2003). The circuit is extracted from layout and simulated to check for any violations in the timing behavior. If necessary, we trim the resistances of the feedthrough cells to correct the routing delays. The trimming can be done by reducing the silicide blocking mask over the polysilicon resistors to reduce the resistance value. If an increase in resistance is required, then, instead of introducing a new feedthrough cell that could change the entire physical design of the circuit we can change the metal interconnects into polysilicon routes.

We conducted experiments to verify that insertion of resistive feedthrough cells saves power by reducing the glitches. The *Verilog* netlist of a circuit and the timing information of the target standard cell library in ASCII format are given as inputs to our experimental setup to generate the layout of the minimum transient energy design. A C-program parses the circuit netlist and the cell delay file to output the LP in a language called AMPL. These files are used by the linear program solver called CPLEX (see Fourer *et al.* (1993)) to find the necessary wire delays. The wire delays are then used by a PERL script to generate the physical design of resistive feedthrough cells and to modify the initial circuit netlist by inserting the new delay elements. This netlist is then used for a final place-and-route by any commercial tool. In our experiments we used *Silicon Ensemble*^{TM} (2003). The target cell library used for the physical design of the modified netlist also includes the various delay elements generated.

Circuit name	Logic cells	No. of resistive feedtrough cells	Area increase (%)	Power saving (%)	
				ASIC	Custom
ALU4	90	37	29.5	23.7	N/A
c432	240	162	114.0*	50.0	35.0
c499	618	396	86.0*	32.0	29.0
c880	383	217	98.0*	43.0	44.0
c1355	546	414	22.0	68.3	56.0
c2670	1193	120	14.0	30.0	31.0

Table 2: ISCAS85 benchmarks in standard-cell.

The power dissipation was estimated in two ways. At logic level we used an event driven logic simulator to estimate the dynamic power consumption as discussed by Raja (2004). This program uses the cell delays and wire delays obtained from LP and the node capacitances extracted from layout to estimate the power dissipation before and after optimization. At post layout level, the *Spectre*^{TM} circuit simulator from Cadence (2003) was used to simulate the extracted circuit. The average power for various input vectors is then calculated by integrating the instantaneous current waveform.

The results obtained by logic simulation of various circuits are presented in Table 2. The dynamic power saving is the decrease in the average power due to random vectors as a percentage of the average power of the unoptimized circuit for the same vectors. These values are compared with a glitch removal technique for custom circuits that uses variable input delay gates described by Raja (2004). Table 2 shows that the average power saving is comparable to that achieved in the custom design of Raja *et al.* (2004). For circuits c432 and c1355 the power saved by our method is significantly greater. This is because the custom design had to insert 61 and 64 delay buffers in c432 and c1355, respectively. In our case the range of the resistance values that can be introduced on a signal line is large. Table 2 shows large area increases for three circuits, marked with asterisk (*). We should point out that the added resistances could have been reduced by selecting alternative larger delay logic cells available in the library. Then the area increase will be similar to other circuits. All optimized circuits had no increase in the input to output overall delay. We have assumed that the glitch removal technique has little or no impact on the short circuit and leakage power. To verify the validity of this assumption we performed the post layout circuit- level simulations for the 4-bit ALU circuit. The pre-optimized and post-optimized circuits were simulated for 1,000 random vectors with a cycle period of 10ns. Simulation gave a 22% power saving.

6.0 Conclusion

The dynamic power elimination objective is successfully incorporated into the standard cell library based design flow. This was done without re-design of the library. The new design flow is effective in designing minimum transient energy standard cell based digital CMOS circuits. Due to the high level of automation the flow permits, the performance of a standard cell based design is improved without sacrificing the fast design cycle time feature. Results have shown circuit design examples with, on average, 40% reduction in average power for no speed degradation. The increase in area, on average, was 58%. The power savings

were either similar to or more than those achieved in the custom design style. Despite the increase in the overall resistance in the circuit, the power dissipation is still lower than the original design because we are decreasing the average current drawn from the power source by reducing the number of transitions, while the power per transition remains the same. Seen another way, in general, increase of resistance reduces the power dissipation, which is V^2/R. Still exploration of other delay elements, especially, CMOS transmission gate is recommended. Although, in finer technologies the leakage power may be significant, the reduction of dynamic power will remain useful.

References

Raja, T., Agrawal, V.D. and Bushnell, M.L. (2004), CMOS Circuit Design for Minimum Dynamic Power and Highest Speed, in *Proc. Int. Conf. VLSI Design*, Jan., pp. 1035-1040. Scott, K. and Keutzer, K. (1994), Improving Cell Libraries for Synthesis, in *Proc. Custom*

Agrawal, V.D. (1997), Low Power Design by Hazard Filtering, in *Proc. Int. Conf. VLSI Design*, Jan., pp. 193-197.

Agrawal, V.D., Bushnell, M.L., Parthasarathy, G. and Ramadoss, R. (1999), Digital Circuit Design for Mimimum Transient Energy and Linear Programming Method, in *Proc. Int. Conf. VLSI Design*, Jan., pp. 434-439.

Dragone, N., Guardiani, C., Zafalon, R. and Meier, P. (1998), An Innovative Methodology for the Design Automation of Low Power Libraries,' in *Proc. Int. Workshop Power and Timing Models, Optimization and Simulation*, Oct.

Fourer, R., Gay, D.M. and Kernighan, B.W. (1993), *AMPL: A Modeling Language for Mathematical Programming*, South San Francisco, California: The Scientific Press. Hashimoto, M. and Onodera, H. (2001), Post-Layout Transistor Sizing for Power Reduction in Cell-Based Design, in *Proc. Asia South Pacific Design Automation Conf. (ASPDAC)*, pp. 359-365.

Hastings, A. (2001), *The Art of Analog Layout*, Upper Saddle River, New Jersey: Prentice Hall.

Mahapatra, N.R., Garimella, S.V., and Tareen, A. (2000), An Empirical and Analytical Comparison of Delay Elements and a New Delay Element Design, in *IEEE Workshop on VLSI*, Apr., pp. 81-86.

Masgonty, J.M., Cserveny, S., Arm, C., and Pfister, P.D. (2001), Low-Power Low-Voltage Standard Cell Libraries with a Limited Number of Cells, in *Proc. Int. Workshop Power and Timing Models, Optimization and Simulation*, Oct.

Rabaey, J.M., Chandrakasan, A. and Nikolic, B. (2003), *Digital Intergrated Circuits: A Design Perspective*, Upper Saddle River, New Jersey: Prentice Hall.

Raja, T. (2002), *A Reduced Constraint Set Linear Program for Low-Power Design of Digital Circuits*, Master's Thesis, Rutgers University, Dept. of ECE, New Brunswick, NJ, Mar.

Raja, T. (2004), *Minimum Dynamic Power CMOS Design with Variable Input Delay Logic*, PhD Thesis, Rutgers University, Dept. of ECE, New Brunswick, NJ, May.

Raja, T., Agrawal, V.D. and Bushnell, M.L. (2003), Minimum Dynamic Power CMOS Circuit Design by a Reduced Constraint Set Linear Program, in *Proc. Int. Conf. VLSI Design*, Jan., pp. 527-532. *Integrated Circuits Conf.*, May, pp. 128-131.

Uppalapati, S. (2004), *Low Power Design of Standard Cell Digital VLSI Circuits*, Master's Thesis, Rutgers University, Dept. of ECE, New Brunswick, NJ, Oct.

Wroblewski, A., Schimpfle, C.V., and Nassek, J.A. (2000), Automated Transistor Sizing Algorithm for Minimizing Spurious Switching Activities in CMOS Circuits, *Proc. Int. Symp. Circuits and Systems.*

Zhang, Y., Hu, X., and Chen, D.Z. (2001), Cell Selection from Technology Libraries for Minimizing Power, *Proc. of the Design Automation Conf.*, Jun., pp. 609-614.