

# Energy-Delay Tradeoffs in 32-bit Static Shifter Designs

Steven Huntzicker, Michael Dayringer, Justin Soprano, Anthony Weerasinghe, David Money Harris, and Dinesh Patil<sup>1</sup>

Harvey Mudd College, Claremont, CA 91711

{shuntzicker, mdayringer, jsoprano, aweerasinghe, david\_harris}@hmc.edu,  
Dinesh.D.Patil@sun.com

**Abstract**—This paper compares the energy-delay tradeoff curves of 32-bit static barrel and funnel shifters. The Stanford Circuit Optimization Tool (SCOT) is used to determine best transistor sizes in a 90 nm process. The paper evaluates the effect of multiplexer valency, circuit design, and physical placement. It also quantifies the costs of various shift operations. A funnel shifter using 4- and 8-input static multiplexer stages gives the best energy-delay tradeoff, with a knee at 440 ps (15 FO4 inverter delays) consuming 0.9 pJ per shift.

## I. INTRODUCTION

Shifters are an integral component of many arithmetic/logic units. Types of shifts include right and left rotates, logical shifts, and arithmetic shifts. Logical and arithmetic left shifts are identical, so a general purpose shifter performs five functions: ROR, ROL, LSR, LSL, and ASR. Shifters are challenging because they tend to be slower, larger, and more power-hungry than other arithmetic operations. Nevertheless, there has been relatively little research optimizing shifters for energy and delay.

Two predominant shifter architectures are the barrel [2] and funnel [1]. A barrel shifter performs a rotation. For shift operations, it then masks the necessary bits. A funnel shifter performs any type of shift or rotate by selecting the appropriate 32 bits from a 64-bit word.

Shifters are built from multiple levels of multiplexers. The theory of Logical Effort suggests that multiplexers with about 4 inputs are best [3], and some studies support this [2, 4, 5]. Zhu [6] suggests a benefit from refactoring two-input multiplexers using a technique called fanout-splitting. Hillebrand [7] and Zhu [6] suggest rearranging the physical placement of multiplexers in barrel shifters to reduce critical path wire length. Acken [8] studied a variety of design and circuit choices for barrel shifters. There has been no comprehensive study comparing barrel and funnel shifters and considering the broad range of proposed optimizations; this paper aims to fill that gap.

One challenge in such a study is accounting for the impact of transistor sizing on energy and delay. This study uses the Stanford Circuit Optimization Tool (SCOT) [9, 10] to optimize transistor sizing and generate energy-delay tradeoff curves for each shifter design considered in this study.

This paper describes the design space under consideration,

the base case designs, the optimization process, and the results, including the best shifter designs.

## II. DESIGN SPACE

Table 1 defines the inputs and outputs of a shifter. The degrees of freedom considered in this study include shifter architecture (funnel vs. barrel), types of shifts supported, arrival time of control signals, multiplexer design, shifter floorplan, and multiplexer valency.

Table 1. Shifter Inputs and Outputs

| Input/Output | Signal Name | Signal Description                      |
|--------------|-------------|-----------------------------------------|
| Input        | a[31:0]     | Input data                              |
| Input        | k[4:0]      | Amount to be shifted or rotated         |
| Input        | arithmetic  | 1: Arithmetic shift<br>0: Logical shift |
| Input        | left        | 1: Left direction<br>0: Right direction |
| Input        | shift       | 1: Shift<br>0: Rotate                   |
| Output       | y[31:0]     | Output data                             |

### A. Early Control

The shift type control signals are on the critical path for a funnel shifter. If control signals are available before the data to be shifted, control signal drivers can be downsized.

### B. Shift Types

Shifters can perform five different shift types, but some applications may not require all types. This paper explores how much energy or delay can be saved if only a subset of types are supported.

### C. Multiplexer Circuit Design

Multiplexers can be built from tristates, pass-transistors, and NAND gates. We also explore refactoring multiplexers using a technique called fanout-splitting [6].

### D. Multiplexer Valency

A 32-bit shifter can be constructed from 5 stages of 2-input multiplexers, each shifting by successive powers of two. Combining stages to use 4-input or 8-input multiplexers may give advantages in both energy and delay [2-5] because 4-input multiplexers can be faster and have less switching activity. For example, a 2-4-4 valency shifter has a single stage of 2-input multiplexers followed by two stages of 4-

<sup>1</sup> Sun Microsystems Laboratories

input multiplexers.

### E. Funnel Floorplans

A 32-bit funnel using 2-input multiplexers traditionally contains an input stage of 63 bits and funnel stages of 47, 39, 35, 33, and 32 bits. Higher valency designs also have stages wider than 32 bits. Different methods can be used to fit these wide stages in a 32-bit datapath. We examine which floorplan gives the shortest wire lengths and best results.

### F. Bit Swizzling

A barrel shifter has a few long wires at each stage of multiplexers that wrap around from the least to most significant bits. Hillebrand [7] and Zhu [6] show how to reduce the worst-case wire length by reordering the placement of these multiplexers. This comes at the expense of increasing the average wire length. In the Hillebrand arrangement, the outputs appear out of order, while in the Zhu arrangement, they appear in numerical order.

## III. BASE CASE DESIGNS

The basic 32-bit barrel and funnel shifters are shown in Fig. 1. Both contain several rows of shifting multiplexers. We define the base case to be the default choice for each degree of freedom so that we can study each design choice separately. The base case assumes all inputs arrive simultaneously, all 5 types of shifts are supported, and multiplexers are built from ganged-tristate inverters.

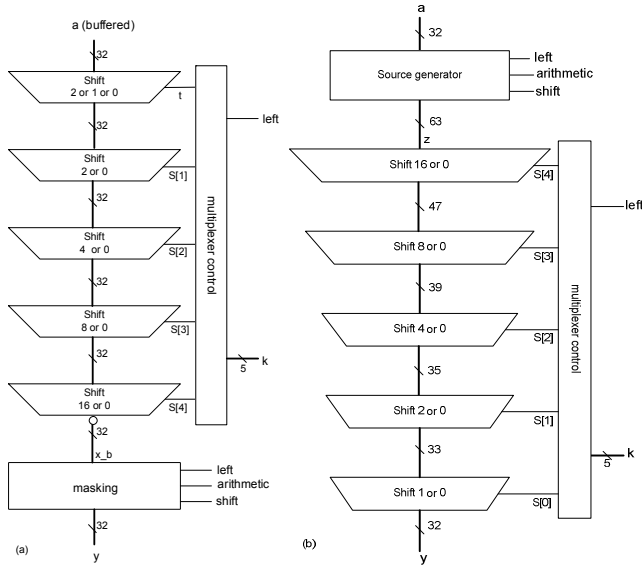


Fig. 1. Shifter architectures: (a) barrel, (b) funnel

The barrel shifter is a right rotator with masking after the rotation to squash unwanted bits for shifts. A left shift by  $k$  is equivalent to a right shift by  $32-k$ . Using two's complement arithmetic, the left shift amount should be  $\bar{k} + 1$ , where  $\bar{k}$  is the two's complement of  $k$ . To avoid an adder in the shift amount logic, the shift amount is simply complemented and the data is preshifted by 1 for left shifts. This preshift by 1 is incorporated into the first rotation stage creating a 3-2-2-2-2

valency design as our base case. Figure 2 shows the multiplexer control logic. The XOR gates conditionally invert the shift amount for left shifts. The first multiplexer is specially designed to accommodate the preshift.

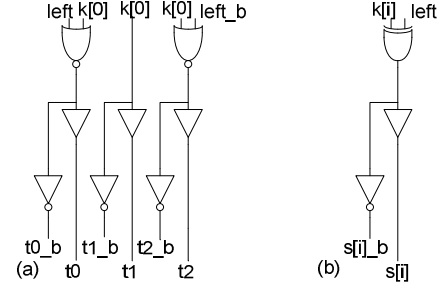


Fig. 2. Barrel multiplexer control: (a) mux3, (b) mux2

The mask is created using a binary-to-thermometer converter. For each bit of input, the select lines that would be driven for a shift of that amount are fed through AND gates. These signals then go through a unique OR tree to determine the actual shift amount and to carry the ones across the word. A 17-bit example of the binary-to-thermometer converter is shown in Fig. 3.

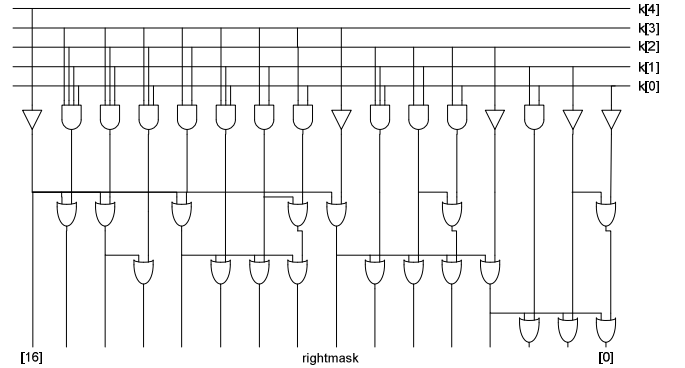


Fig. 3. Barrel mask generation

The final stage of the barrel masker determines the type of shift and masks out the unwanted bits, as shown in Fig. 4.

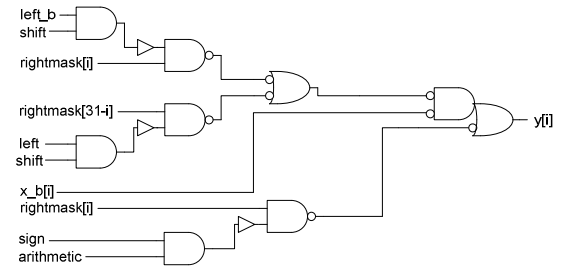


Fig. 4. One bit masking

$a[31]$  is renamed *sign* and used for sign extension in arithmetic right shifts. To handle left shifts,  $rightmask[31-i]$  is used to flip the mask. This masker was designed to add only one gate to the critical path from  $x_b$  to  $y$  while avoiding high energy devices such as multiplexers. The mask application is an inverting process; thus, the inverter can be eliminated from the final rotating multiplexer.

The funnel shifter performs a  $k$ -bit right shift by selecting bits  $k+31:0$  from a 63-bit input word,  $z$ , using 5 levels of 2-

input multiplexers [1]. The input word is chosen based on the shift type, as shown in Table 2.

Table 2. Funnel Shifter 63-bit Input Word

| Shift Type | z[62:32] | z[31] | z[30:0] |
|------------|----------|-------|---------|
| ROR        | a[30:0]  | a[31] | a[30:0] |
| LSR        | 0        | a[31] | a[30:0] |
| ASR        | sign     | a[31] | a[30:0] |
| ROL        | a[31:1]  | a[0]  | a[31:1] |
| LSL        | a[31:1]  | a[0]  | 0       |

Left shifts complement the shift amount to shift by  $31-k$ . A source generator is used to choose the word that goes through the funnel. Fig. 5 shows the design of the source generator; it is different for the upper, middle, and bottom groups of bits. Control signals are buffered before driving large loads.

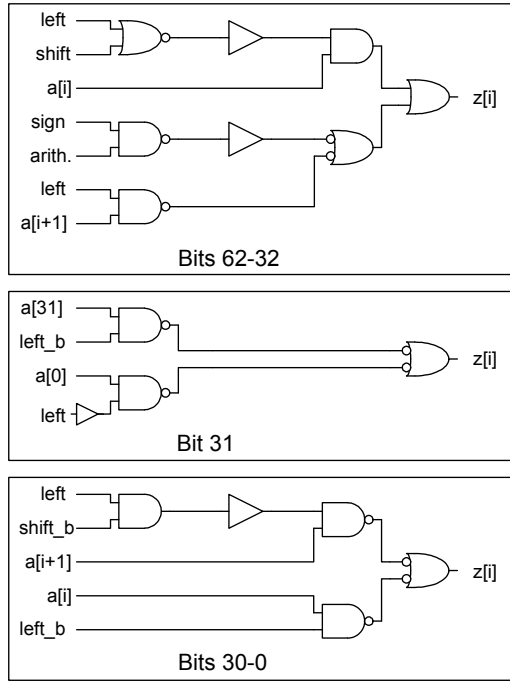


Fig. 5. Funnel source generator

#### IV. OPTIMIZATION PROCESS

The shifters were first modeled and verified in Verilog, then translated into a SPICE netlist format. The activity factors were extracted from IRSIM simulations. SCOT reads a modified SPICE file with the activity factors and computes transistor sizes that will minimize energy for a given delay.

The optimization targeted a 90 nm process with  $V_{DD} = 1.1$  V and a fanout-of-4 inverter (FO4) delay of 27 ps. Gate capacitance was 1.1 fF/ $\mu\text{m}$ . A bit pitch of  $80 \lambda$  ( $3.6 \mu\text{m}$ ) was assumed for the datapath. Wire capacitance was  $\frac{1}{4}$  of gate capacitance, corresponding to 0.99 fF/bit pitch.

##### A. Verilog Specification

All of the shifter designs were specified in structural Verilog and validated against directed and random test

vectors. Specific naming conventions were used to specify the location of each sub-circuit so a translation tool could calculate wire lengths.

##### B. Translation

The Verilog was translated into SPICE decks as input for SCOT using a custom script. The script also added wire capacitance between bit pitches. Studies confirmed that wire capacitance along a bit pitch was less significant, so it was ignored.

##### C. Activity Factors

The SPICE deck was translated into SIM format using a script, then simulated in IRSIM to obtain the activity factors for each node based on a set of 1000 random vectors. This models applications such as cryptography, but overestimates the activity factors for integer codes with shorter and less random inputs.

##### D. Optimization

SCOT optimizes a circuit for energy or delay by changing transistor sizes based on constraints of voltage, capacitance, and signal slope. The input and output capacitances were constrained to 5 fF and the minimum transistor width was set to  $3 \lambda$  ( $0.135 \mu\text{m}$ ). Each gate is allowed to be sized independently. Energy-delay curves were obtained by minimizing energy for a range of delays.

#### V. RESULTS

This section begins by discussing the activity factors of various nodes in a shifter. It then presents simulation results for each of the degrees of freedom in the design space.

##### A. Activity Factor Insights

The average activity factor over the entire shifter could be approximated at 0.25, which would correspond to random inputs with each node switching every other cycle on average. However, there are several groups of nodes that have activity factors that are significantly lower than 0.25. To obtain the most accurate results from simulation and to understand how to minimize dynamic power, it is necessary to use separate activity factors for every node in the system.

One place where there is a low activity factor is if  $n$ -bit one-hot multiplexers are used in a design. Each select line will have an activity factor of approximately  $1/(2n)$ . Because all of the select wires span the full width of the shifter, this is an important activity factor to take into account.

Another instance is in the mask creation in the barrel shifter. Because of the thermometer encoding, the probability of having a 1 at the node increases from 0 to 1 from the lsb to the msb. Therefore, there are low chances of switching and corresponding low activity factors near both extremes in the mask.

In the logic of the mask application in the barrel, and the input generator in the funnel, any gate that switches on a certain shift type (e.g. RSA) rather than any of the random inputs, will have a reduced activity factor. There are fewer

long wires in these areas, so this effect is relatively minor.

### B. Early Control

The funnel shifter can take advantage of early shift type signals (left, shift, and arithmetic) to perform part of the input generation logic before the data arrives. This substantially improves the best achievable delay from  $a$  to  $y$  in Fig. 1. The logic is small, so downsizing it has minor energy benefits.

The only control signal in the barrel shifter that is involved in the critical path is the *left* signal input to the preshift stage. Providing it early has negligible benefit unless the shift amount is also available early.

Fig. 6 shows the base-case performance of the funnel and barrel shifters and demonstrates how the funnel shifter benefits from early control signals.

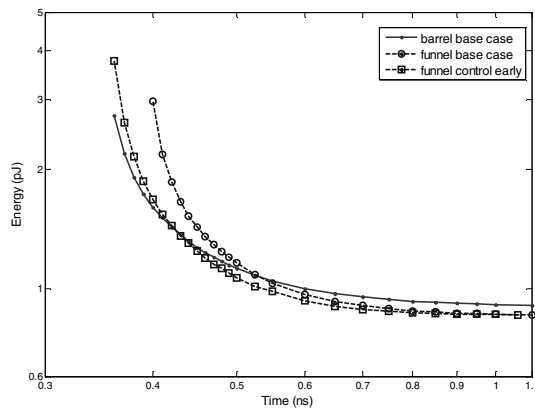


Fig. 6. Benefits of early control

### C. Shift Types

Fig. 7 compares the base cases for the funnel and barrel shifters against versions that perform only bidirectional shift operations (LSR, LSL, and ASR) and versions that perform only right rotates (ROR).

The barrel shifter performs a rotate and then masks off unwanted bits for shift operations. The only hardware benefit to removing the rotation operations is that two AND gates in the masker are eliminated. Thus, the two cases are almost indistinguishable.

A right-rotate only barrel eliminates the preshift and the mask stages, saving substantial energy and delay.

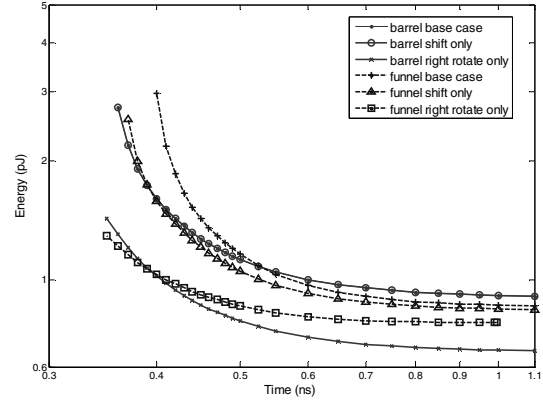


Fig. 7. Comparison of shift types

Removing rotation operations from the funnel eliminates 2 logic gates from the input generator, and reduces the critical path of the shifter.

A right-rotate only funnel shifter eliminates the input generator, leaving just the five funneling stages. This is similar to the barrel right rotator except that the funnel contains significantly more multiplexers in the early stages.

### D. Multiplexer Circuit Design

We considered three multiplexer designs used for two input devices: ganged-tristate, pass-transistor, and fanout-splitting. Examples of these designs are shown in Fig. 8.

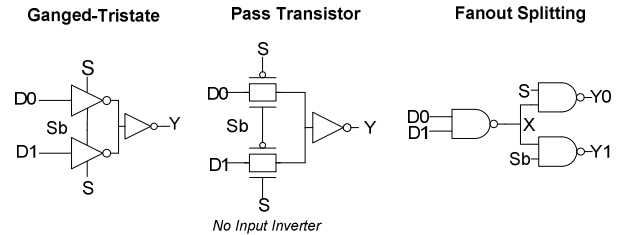


Fig. 8. Multiplexer circuit types

The base case ganged-tristate multiplexers use two tristates in parallel to choose one of the inputs.

The pass transistor multiplexers use a pair of transmission gates with inverters on the input, output, or both. Multiplexers with input inverters only were inferior to the ganged-tristates because they drove long wires through series transistors. Multiplexers with input and output inverters showed no benefit over ganged-tristates because the topology is essentially the same. Multiplexers with output inverters only are expected to be beneficial, but could not be easily modeled in SCOT.

Fanout-splitting multiplexers seek to increase the speed of the shifter by decoupling the shifting and non-shifting output paths of the multiplexer [4]. Because of the decoupling, each output must only drive a single input in the next stage. While 2 NAND gates have a higher logical effort than inverters, this process reduces the necessary drive by cutting the fanout from 2 to 1 driven gates and the average wire

length of any path. The activity factor for the output of a fanout gate is reduced from 1/4 to 3/16.

Fanout-splitting doesn't apply to 3-input multiplexers, so the first rotation stage in the barrel uses a one-hot multiplexer.

Fanout-splitting can also be applied to the funnel shifter. The same decoupling principles apply; the fanout and the loads, and activity factors are reduced. However, the same reduction of the critical path is not observed, because there are no long wrap around wires that can be cut out of the critical path.

Fig. 9 shows that the barrel fanout has lower energy for all delays than the barrel base case, and the funnel fanout has lower energy than the base case at short delays.

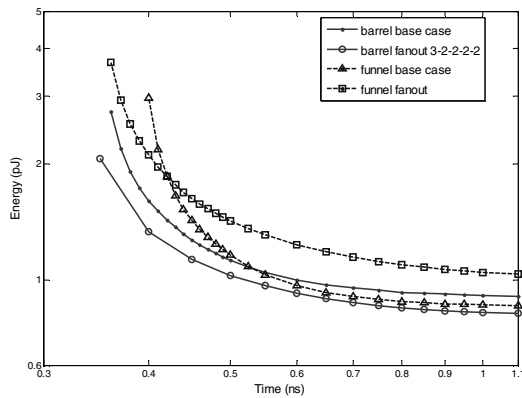


Fig. 9. Fanout-splitting comparison

### E. Multiplexer Valency

Logical Effort [3] suggests that wide multiplexers should be built from trees of 4-input multiplexers, so we considered 2-4-4 and 4-8 valency designs for the funnel, and 3-4-4 and 5-8 for the barrel. Note the additional input to the barrel to preshift for left shift operations.

In the barrel shifter, the smaller valency multiplexers are positioned first, because the control signal for the first multiplexer is on the critical path, and smaller multiplexers have less control logic. In the funnel, the smaller valencies are first so that the most wire is at the end of the shifting process with the large fan-in multiplexers.

In the barrel shifter, the 2-2-2-2-2 design uses 10 select wires (5 selects and their complements), each of which drives 64 tristates. The 2-4-4 design has 18 select wires, each 4 input device drives 32 tristates, and the 2 input drives 64 tristates. The 4-8 design has 24 select wires, each driving 32 tristates. Because the multiplexers tend to be sized small to keep the energy of the shifter low, the wire capacitance dominates gate capacitance. Increased valency adds wire capacitance to both shifter designs. However, the activity factors for the select lines are inversely proportional to the number of inputs to the multiplexer, and this counteracts the cost of the capacitance.

Fig. 10 shows that the two-stage designs with high-fan-in

multiplexers are consistently superior.

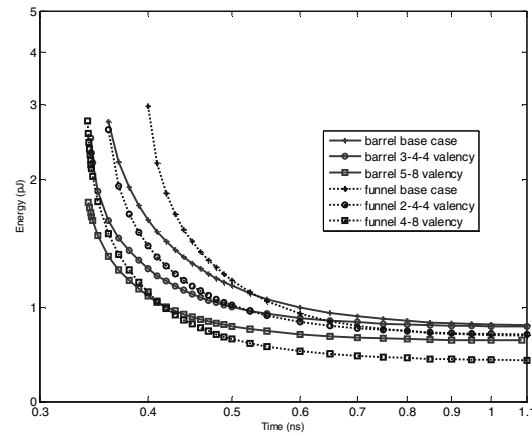


Fig. 10. Higher valency shifters

### F. Funnel Floorplans

We assumed the base case funnel design to have a naïve layout, as shown in Fig. 11. This layout is useful for understanding the functionality of the funnel shifter, but it does not conform to a rectangular datapath. It also has long wires spanning the first two rows of funneling multiplexers. There are two alternatives to allow the design to fit in 32 bits: folding the overhanging bits into new rows or compressing the rows to fit in a 32 bit data pitch.

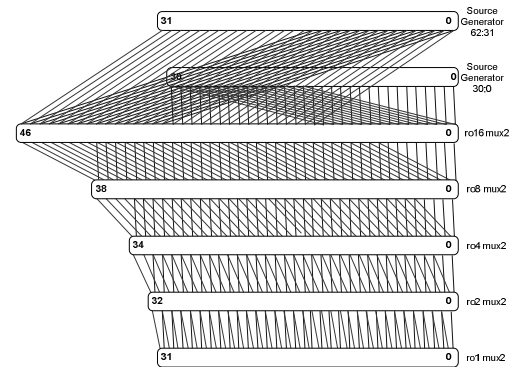


Fig. 11. 7-row funnel floorplan

One possible folding method will be referred to as the 11-row design. This design folds the extra width of each row into a new row. This limits each row to 32 bits and helps to bring the wires more in line, cutting wire length early in the shifter. The major failing of this layout is that it wastes space in the bottom half of the shifter, where some rows have only 1 or 3 bits occupied. A more compact 8-row folded design is proposed in Fig. 12, where the short rows are grouped together. This saves 3 rows for a small increase in the total wire capacitance.

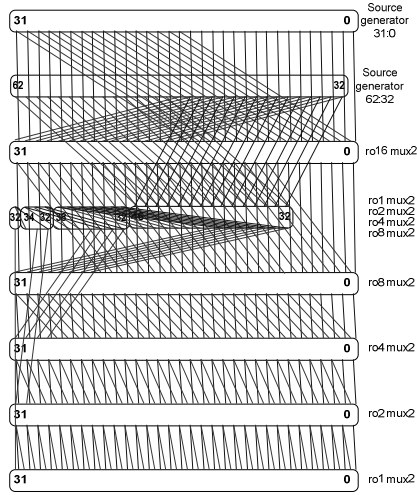


Fig. 12. 8-row funnel floorplan

Results for all three floorplan choices are shown in Fig. 13. As expected, the 11-row design offers the least energy, although we optimistically neglect the capacitance of wires between rows along the bit pitch. The 8-row design is competitive and offers a 25% savings in area.

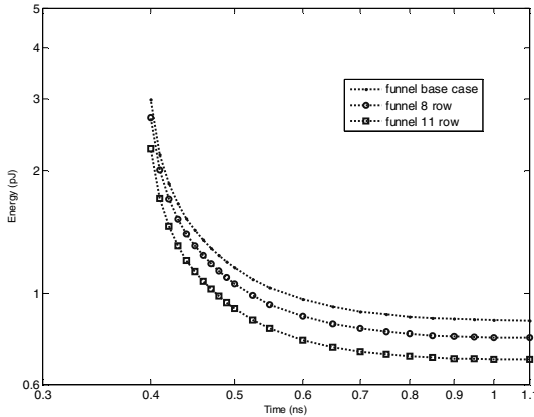


Fig. 13 Funnel floorplan results

Both the multiplexers and the source generator can be compressed, although the source generator is folded in the base case. However, compressing the shifting multiplexers was not found to be a profitable option, because the extra wire necessary to connect the inputs and outputs on the 32 bit pitch counteracted the savings from a compact area.

#### G. Bit Swizzling and Unique Sizings

The barrel shifter has long wrap around wires between multiplexers that rotate the least significant bits to the most significant bits. For example, Fig. 14(a) shows the wiring in an 8-bit barrel shifter

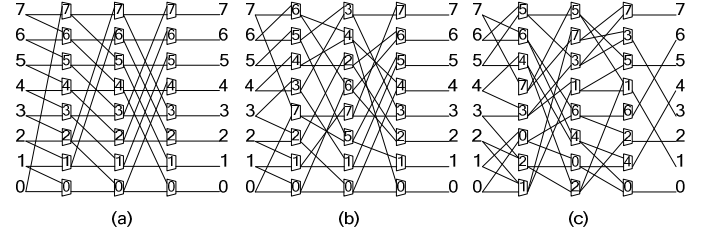


Fig. 14. 8 bit wiring: (a) base case (b) Zhu, (c) Hillebrand

The critical path for rotating bit 0 by  $k = 0$  involves a wire of length 7 in the first stage, 6 in the second, and 4 in the third. In general, the total accumulated wire load on the worst case path is  $O(n \log n)$ , as compared to  $O(n)$  on the average case. Zhu and Hillebrand propose reordering the multiplexers to reduce the total accumulated wire load on the critical path. Fig. (b) and (c) show the Zhu and Hillebrand wiring. Zhu's topology comes at the expense of increasing the average wire length. Hillebrand reduces the average wire length but the final stage of multiplexers are out of order. This is not a problem because they can be reordered before masking.

Another way of handling the long wires on certain multiplexers is to upsize those multiplexers. Sizing different gates in a row differently requires *uniquifying* the gates. This detracts from regularity in a datapath but helps performance significantly. The base cases and results in all the other sections assume uniquified gates. This section explores the benefits of swizzling and uniquifying the multiplexers in a barrel shifter.

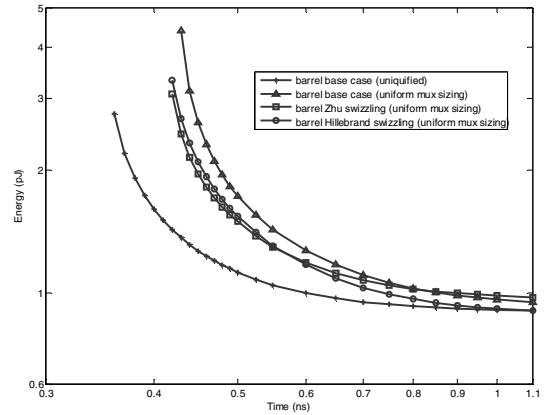


Fig. 15. Benefits of bit swizzling and uniquifying

Fig. 15 compares the barrel shifter base with unique and vs. uniform multiplexer sizing. The unique sizing offers a tremendous advantage because only a small number of multiplexers driving wraparound wires need to be upsized to greatly reduce delay. Both types of swizzling also offer an advantage when uniform multiplexer sizing is required; Hillebrand is slightly better than Zhu swizzling because it involves less total wire length. However, swizzling offers no benefits when the multiplexers are uniquified.

These swizzling methods do not apply to the funnel shifter because there are no wrap around wires, so the paths of all of the bits are the same length.

## H. Comparison of Results

The results so far considered each choice in the design space separately.

The 4-8 valency funnel design, which had the least energy and delay of the funnel shifters, has 7 bits that hang over the datapath, as shown in Figure 16. These bits could be crammed into the space next to the datapath in a typical layout, or folded into another row. Folding the 7 overhanging bits into a new row would allow for shorter wires, resulting in less overall wire energy.

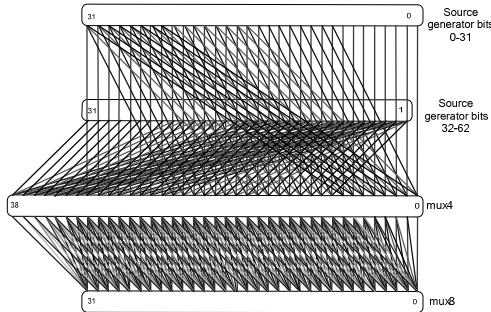


Fig. 16. Funnel 4-8 valency floorplan

Fig. 17 compiles all of the best results of shifters performing all shift types with all inputs arriving at the same time in the design space. The funnel 4-8 valency shifter with the longest row folded has the lowest energy, while the barrel 5-8 valency has the lowest delay. The two curves cross at a delay of 360 ps.

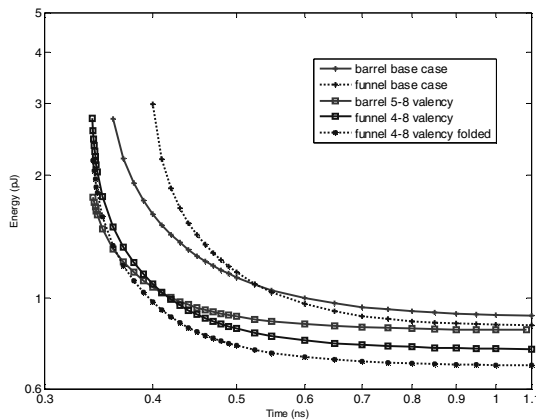


Fig. 17. Comparison of best designs

## VI. CONCLUSION

This work offers the designer guidance to select the best shifter design for a particular application. The results of our study show that a multiplexer valency of about 4 is best, with the two-stage 4-8 valency design working well. The funnel can achieve slightly lower energy (733 fJ) because the source generator has less hardware than the barrel masker. The barrel is best at minimum delay (340 ps, or 12.6 FO4 inverter delays) because the barrel has a shorter critical path than the funnel. The funnel shifter offers the best energy-delay product of 394 pJ-ns, although the barrel is

close at 441.

If only shift operations are needed, the funnel is preferable, while if only rotates are needed, a barrel with no masking is most efficient. Funnel shifters also benefit if the control signals arrive early.

There have been several other shifter optimizations proposed in the literature. Fanout-splitting is advantageous for valency-2 designs, but not competitive with higher-valency shifters. Bit swizzling offers modest gains for barrel shifters if all multiplexers are identically sized, but is not as good as simply upsizing the multiplexers that drive the long wrap-around wires.

An area for future research is to quantify the benefits of pass-transistor multiplexers. Also, all of the tests in this paper were performed with a fixed input and output load. It is possible that the relative position of the funnel and the barrel curves could be dependent upon the chosen loads.

## ACKNOWLEDGMENT

The authors would like to thank Ivan Sutherland for his technical insight and suggestions, and Sun Microsystems for supporting this work.

## REFERENCES

- [1] N. Weste and D. Harris. *CMOS VLSI Design: A Circuits and Systems Perspective*, Addison Welsey, 3<sup>rd</sup> Ed., 2004.
- [2] R. Davis, "The ILLIAC IV Processing Element," *IEEE Trans. Computers*, vol. C-18, no. 9, Sept. 1969, pp. 800-816.
- [3] I. Sutherland, R. Sproull, and D. Harris, *Logical Effort: Designing Fast CMOS Circuits*, Morgan Kaufmann, 1999.
- [4] G. Tharakan and S. Kang, "A new design of a fast barrel switch network," *IEEE J. Solid-State Circuits*, vol. 27, no. 2, Feb. 1992, pp. 217-221.
- [5] A. Singh, M. Barany, and D. Deleagnes, "A mixed signal rotator/shifter for 8 GHz Intel Pentium 4 integer core," *Symp. VLSI*, 2004, pp. 394-397.
- [6] H. Zhu, Y. Zhu, C. Cheng, and D. M. Harris, "An Interconnect-Centric Approach to Cyclic Shifter Design Using Fanout Splitting and Cell Order Optimization," *Asia and South Pacific Design Automation Conf.*, pp. 616-621, Jan. 2007.
- [7] M. A. Hillebrand, T. Schuriger, and P. Seidel, "How to Half Wire Lengths in the Layout of Cyclic Shifters," *Intl. Conf. on VLSI Design*, pp. 339-344, Jan. 2001.
- [8] K. Acken, M. Irwin, and R. Owens, "Power comparisons for barrel shifters," *Intl. Symp. Low Power Electronic Design*, 1996, pp. 209-212.
- [9] D. Patil, O. Azizi, M. Horowitz, R. Ho, and R. Ananthraman, "Robust energy-efficient adder topologies," *Comp. Arith. Symp.*, 2007, pp. 16-28.
- [10] D. Patil, S. Kim, "Stanford Circuit Optimization Tool (SCOT) User Guide," private correspondence with authors: ddpatil@stanford.edu, sjkim@stanford.edu.