# Low-Voltage Swing Logic Circuits for a Pentium® 4 Processor Integer Core

Daniel J. Deleganes, Micah Barany, George Geannopoulos, Kurt Kreitzer, Matthew Morrise, Dan Milliron, Anant P. Singh, and Sapumal Wijeratne

***Abstract*—The Pentium® 4 processor architecture uses a 2× frequency core clock to implement low latency integer operations. Low-voltage-swing (LVS) logic circuits implemented in 90-nm technology meet the frequency demands of a third-generation integer-core design.**

***Index Terms*—Adders, integrated circuit design, microprocessors.**

## I. Introduction

THE Pentium 4 processor architecture uses ultra-low-latency integer operands to achieve performance [1]. This is accomplished by running the integer core at twice the core frequency of the processor. For example, a 3.50-GHz processor has the integer logic functioning at 7 GHz. The introduction of 90-nm technology [2] enables steadily increasing transistor speeds over the life of the process. Described in this paper is the implementation of a 2× frequency integer logic core using low-voltage-swing (i.e., differential-voltage small-signal) logic (LVS). This circuit topology is designed explicitly to take advantage of the frequency headroom enabled by 90–nm technology [3], [4]. A brief introduction of the processor and integer core will be given followed by details of the LVS technology. Included are discussions of the basic circuit topology, pre-silicon verification, and implementation examples. Finally, post-silicon data will be shown.

## II. Integer Core

The integer core being described is part of the third-generation Pentium 4 processor on 90-nm technology. The design has 1-MB L2 cache, 16-kb L1 cache, 125 M transistors, and 112-mm$^2$ die size. Moving the architecture onto 90-nm technology presents many challenges. This paper focuses on those needed to move the integer core component onto 90-nm technology.

The integer core contains the fast data path which executes all low latency integer operations for the machine. The data path includes arithmetic logic units (ALUs), the bypass and flag logic to support those ALUs, and the integer register file. The level one (L1) data cache and the address generation units (AGUs) are also part of the integer core. To give a relative feel of size, the total number of drawn devices in the integer core exceeds that of a Pentium Pro processor.

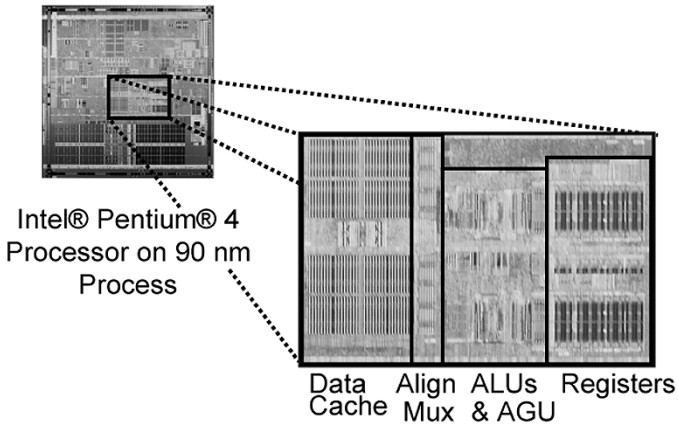


Fig. 1. Die photograph with integer core magnified.

The integer core utilizes an internal clock running at twice the frequency of the core clock. This "2× frequency" clock is referred to as "Fast Clock" or by commonly by the acronym FCLK. Designing at such clock rates is extremely difficult. For example, consider logic depth. Between sequential elements such as latches, there is time for at most six logic stages. After accounting for clock variation and the timing requirements of driving and sampling sequentials, the designer is left with only two stages for logical work. Another significant complexity is the clock network itself. Guaranteeing robust FCLK distribution through end-of-life process capabilities requires detailed analysis. Because of such challenges, FCLK usage on the processor is exclusive to the integer core and usage is targeted. In final implementation, one third of the integer core blocks are using the slower core clock frequency.

Fig. 1 shows a die photograph of the processor on 90-nm process. The integer core lives in the center of the die. A magnified view shows the relative positions of the integer core building blocks. There are two distinct 32-bit FCLK execution data paths staggered by one clock to implement 64-bit operations. Each datapath has two ALUs. ALU0 can perform either a fast add or a fast Boolean operation during a cycle. ALU1 can perform either a fast add or a fast shift/rotate during a cycle. The AGUs in each 32-bit datapath are distinct designs, whereas the ALUs are essentially shared circuits.

Fig. 2 shows a block diagram of the structures in essentially the same placements as the die photo. Illustrated is the integer core load loop, which is a fundamental pipeline in the machine

Manuscript received April 15, 2004; revised July 30, 2004.

The authors are with Intel Corporation, Hillsboro, OR 97123 USA (e-mail: daniel.j.deleganes@intel.com; mbarany@ichips.intel.com; george.geannopoulos@intel.com; kurt.kreitzer@intel.com; matthew.c.morrise@intel.com; daniel.e.milliron@intel.com; anant.p.singh@intel.com; sapumal.wijeratne@intel.com).

Digital Object Identifier 10.1109/JSSC.2004.838020

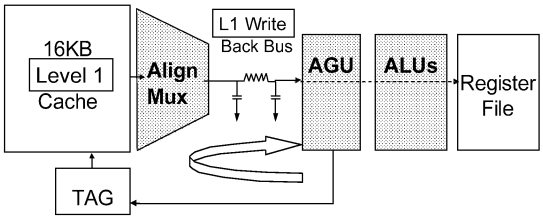


Fig. 2. Integer core loop. Shaded boxes are implemented using LVS circuit technology.

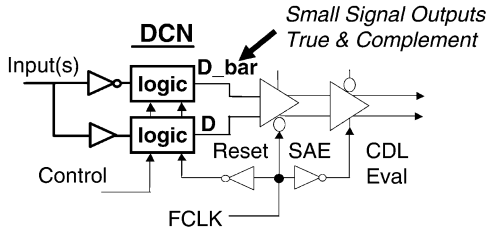


Fig. 3. Block diagram of chosen LVS circuit topology.

critical to performance. The loop is initiated by the AGU address output used for a data cache lookup. The cache data output is aligned and presented to the AGU or ALUs for use as a possible input source. Finally, results are written back to the register file or used to initiate another cache look up. To meet the low-latency demands of the pipeline, several structures must produce 16- or 32-bit results in an FCLK phase. The shaded boxes (alignment multiplexer, AGUs, and ALUs) use LVS circuit technology to meet these low-latency demands.

## III. LVS Technology

The foundation of LVS technology is the use of deep N-channel pass gate chains to implement logic. The chain depth—that is, the number of pass gates in series—is such that the output voltage levels are very small averaging about 10% to 20% of VCC. Parallel pass-gate chains are thus used to produce both true and complement outputs and the voltage differential between the outputs is sensed. In the integer core, the logic functions are varied and are in cases random. The guideline for depth is no more than six transistors in series but there are no guidelines for the number of parallel transistors. Thus, the function width is not restricted. The resulting circuit is like a single gate with up to 5 000 transistors.

### A. Basic Topology

Fig. 3 shows a block diagram of the chosen LVS circuit topology [5]. Inverters drive complementary full rail data signals into the parallel N-channel pass gate logic chains. The outputs of the parallel pass-gate chains are small signal. Throughout this paper, the parallel pass-gate network—encompassing both chains—will be referred to as a Diffusion Connected Network (DCN). The outputs of the DCN are sensed and restored to full rail outputs.

### B. DCN

The DCN is reset low and equalized by N-channel devices. During small-signal development, one side of the DCN evaluates high while the other side attempts to keep the network low. The N-channel pass-gate system was adapted due to the comparative advantage of N-channel linear region characteristics. The reset low system brings the advantage of significantly smaller reset and equalize devices minimizing overall capacitance in the network. Fig. 4 shows the ALU input source multiplexer (mux) DCN as an example.

The gates of the DCN devices are driven by either static or domino circuits. The first pass gate in the network is termed the "Thru Gate." It is controlled by reset or by a logically qualified reset clock. The Thru Gate topology eliminates any contention between data and reset on either side of reset window and reduces the complexity of the CMOS to LVS interface to that of a standard CMOS to domino interface. After the Thru Gate in the ALU source mux example, the data goes through a 5-to-1 mux to select the proper source followed by an $M$-to-1 conditioning mux. The $M$ value depends on the ALU and the functional block in the ALU. An example of conditioning here may be zeroing out bits in the data. This simple example of a DCN is straightforward and for all input to output paths has only three N pass gates in series.

Illustrated in Fig. 5 without any supporting reset, equalization, or clocking devices is an example of random logic in a DCN. The "True" logical function is shown implemented on both sides of the DCN. On one side, the inverse of the reset clock supplies a high input level into this True function, while on the other parallel DCN network, a zero is driven into the True function. (The reset clock is used as a high input level source since it provides a monotonically switching input during evaluation.) The equivalent holds for the "Complement" logical function. It is implemented on *both* sides of the parallel chains with opposite input levels. At every node along the chain, there is a node matching it in the parallel chain. This node-for-node matching along the parallel networks is critical when sensing differential voltage at the outputs. Any circuit mismatch or unbalance between the parallel networks will result in built-in differential noise. This example remains a simplistic illustration since the longest path from data-in to data-out is three series devices and the width does not exceed two devices. Actual circuits in the integer core are up to six series devices deep and the width is unconstrained.

### C. Sensing and Gain

The DCN ends in a high-speed low-gain ratio'd P-type sense amplifier feeding a level restore stage implemented as a cross-coupled domino logic (CDL) gate. Together these stages amplify 10% VCC differential (average) to full rail outputs.

Fig. 6 shows a schematic of a typical sense amp and CDL. The ratio'd P-type sense amp is quick but burns power throughout evaluate. Lower power designs are used if speed can be sacrificed. The output of the CDL can go to CMOS or can feed into another LVS network as data or control.

In practice, the CDL stage is often used to implement logical functions combining the outputs of up to four different sense amps. Shown in Fig. 7 is a 2-to-1 mux with small signal selects used in the ALU1 rotator. Two data DCNs and sense amps are being chosen by the output of a third select DCN. The CDL mux in the center of the figure is the level restore stage for all three

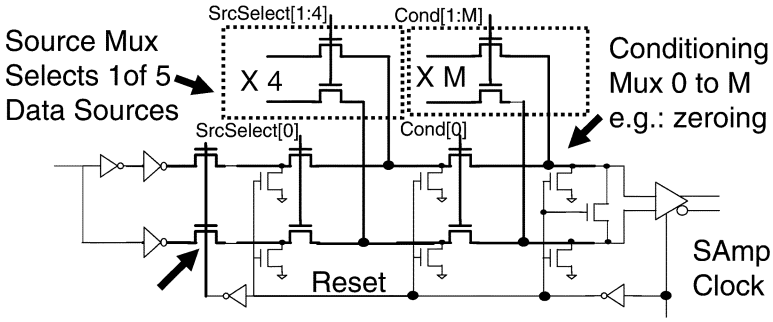


Fig. 4. Example of a DCN taken from the ALU input source mux.

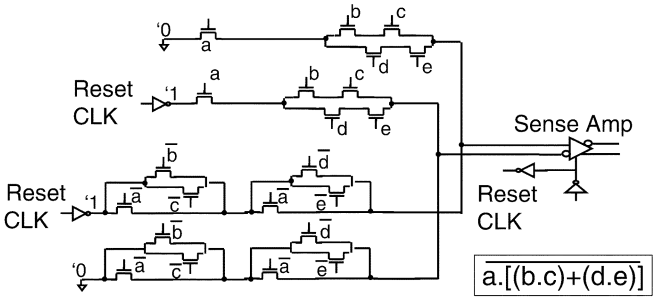


Fig. 5. Illustration of a complex random logic function DCN.

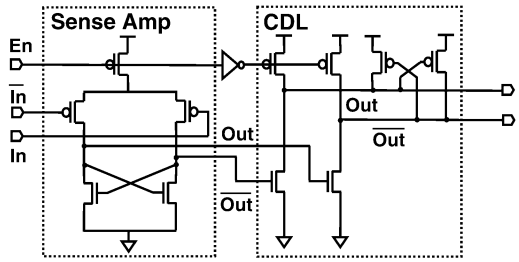


Fig. 6. Typical sense amp and CDL pair.

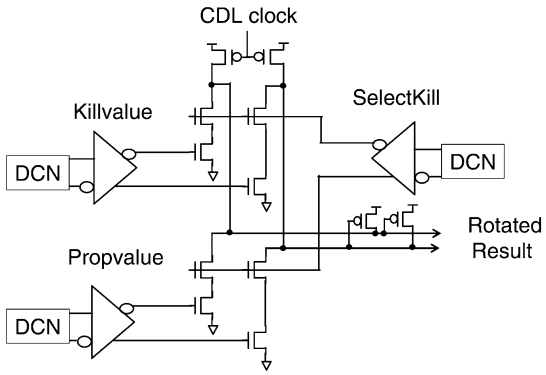


Fig. 7. ALU1 rotator 2-to-1 CDL mux.

DCNs. This combining of multiple LVS networks into a single CDL is widely used in the shifter/rotator since the data inputs and control inputs arrive together in time.

### D. Advantages

Thus, the chosen LVS system delivers compelling advantages for implementing high-frequency logic circuits.

The DCN logic function signal development is allocated less than 35% of a FCLK clock phase, equivalent to two nominal inverter delays. During this time it implements up to six effective logic stages. The gain stages cost less than two stage delays and a seventh logic function can be implemented in the CDL itself. Thus, in the time of four nominal gate delays there is the potential for up to seven logic stages, each of which may be very wide.

In small-signal differential voltage systems, wire resistance of long routes has a low cost. The LVS DCN transistors operate in the linear region—they have a high ON resistance. Sensitivity studies demonstrate that wire resistance is a relatively small percentage of the total resistance in most LVS small signal networks. This leads to usage of thin wires throughout the small signal data paths minimizing capacitance and enabling compact LVS datapath layout. Compact LVS datapath layout relaxes constraints on any large signal datapath controls.

A LVS DCN is essentially one large stacked gate. This means that from a source-drain leakage perspective the overall structure has a much lower effective device width dissipating current than a multi-staged CMOS or domino design.

## IV. Pre-SI Verification

In microprocessor design, a novel circuit topology needs to produce timing requirements compatible with the tool suite analyzing the rest of the design. For LVS this requires taking an analog small signal system and characterizing it to have digital timing relationships.

### A. Simulation Failure Criteria

The first step to analyzing the circuits is to create the digital pass/fail criteria. In the LVS circuit topology described, the output low-voltage level $(V_{\rm ol})$ of the low-gain sense amp is above $V_{\rm ss}$. The $V_{\rm ol}$ level is directly dependent on the amount of differential signal presented to the ratio'd sense amp. This non-$V_{\rm ss}$ $V_{\rm OL}$ is amplified by the following CDL causing a false discharge on the CDL output which is intended to remain high. In a circuit which resolves properly the cross-coupled CDL keepers fight back resulting in a low going "glitch" on the logical high output waveform. If the circuit does not resolve, then the glitch becomes a full discharge—meaning both sides of the CDL evaluate low. Simulation waveforms showing this "glitch" in a properly functioning circuit are in Fig. 8. The glitch magnitude is one fail criterion used to calculate the circuit setup time.

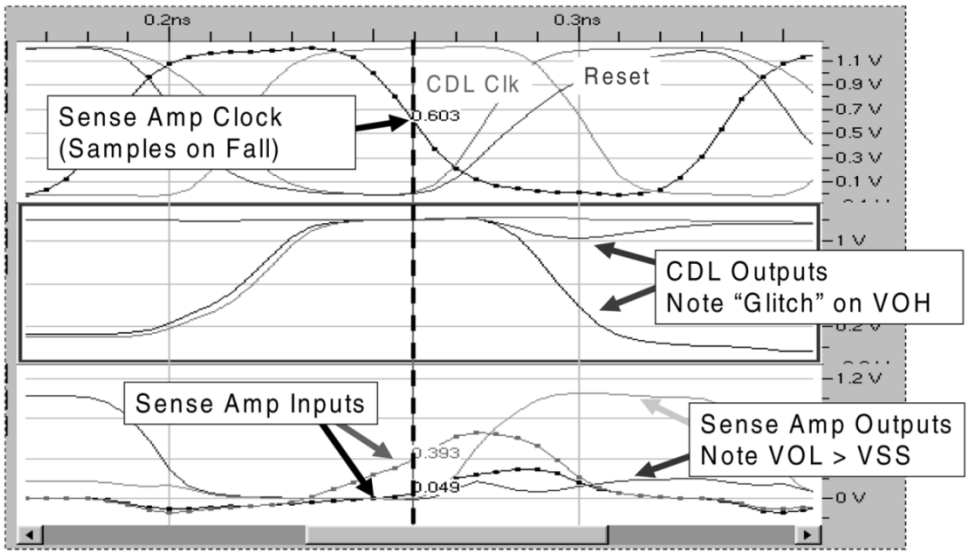


Fig. 8. Simulation waveforms from a properly functioning LVS circuit.

The second fail criteria used is to measure the input voltage differential at the sense amp when it is clocked and check against a minimum needed to overcome differential noise sources, dc offset of the sense amp and process variation. This minimum differential voltage is typically 8% to 10% of the supply and is determined using statistical analysis of the first-order variables per circuit. A more exacting method would be to statistically measure differential current integrated over time for each timing arch. However, this is a much harder measurement to make and quantify as a digital pass/fail for widely varying topologies.

Through simulation it was concluded that combining the two fail criteria—CDL glitch magnitude and minimum voltage differential—leads to a robust method that adequately finds the fail points of a LVS system. Which criteria of the two criteria fails first (i.e., is the limiter) varies between circuits.

### *B. Analysis*

An in-house LVS timing and noise tool (LVSTNT) was created to analyze the LVS circuits. The tool uses a patented pruning algorithm [6] to exhaustively search all circuit paths constrained by logic equations embedded as schematic properties. The RTL model incorporates equivalent equations that are formally verified against the schematic versions.

The algorithm starts at each input and traces through pass gates. As each pass gate is encountered, it is determined from the logical relationships and previous state if the gate is on or off. If this determination can not be made the pass gate is turned on and a duplicate of the path with the pass gate off is placed in the queue for later tracing.

Using the same core path analysis and simulation engines LVSTNT performs several different analyses. Timing analysis using the above described fail criteria is done to determine setup and hold times. Noise analysis with spurious input values is done to flag sensitivities and appropriately tax the timing arcs. Reset analysis is performed to determine if reset time was sufficient to reset the network to ground. Sensitivity analysis is done to determine the effects of small changes in input and clock arrival times and waveforms. Power analysis is done to determine the maximum power consumed by the block. In addition, because an exhaustive path search is done, LVSTNT performs *de facto* logic verification as a precursor to formal verification done later in the tool flow.

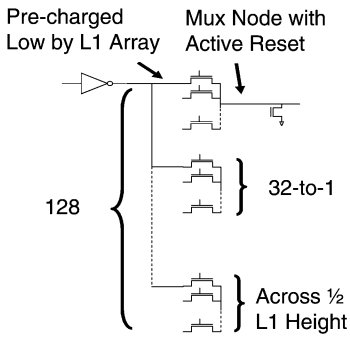


Fig. 9. LVS alignment mux topology.

After pruning, the rotator/shifter described later still exceeds 42 000 paths for verification. Including all circuit-induced differential noise scenarios increases the path count 10×. This number of paths requires days of CPU time to simulate. Such long turn-around times for timing analysis would be prohibitive, so the LVSTNT tool uses parallel processing to partition the job of tracing and simulating signal paths. It then merges the individual timing analysis results into a complete analysis report. Using a pool of hundreds of compute servers, the turn-around time for a full analysis was improved by a typical factor of 70.

## V. Physical Layout

Extremely high-quality layout is necessary for random small-signal logic, especially for a product in high-volume manufacturing. For the LVS design described, matching analog layout rules are required in three dimensions up to the layer above the last used for small-signal interconnect. These rules ensure that all physical noise sources such as leakage, gate-to-drain coupling, wire coupling, and process variation are common mode. An in-house analog layout rule checker

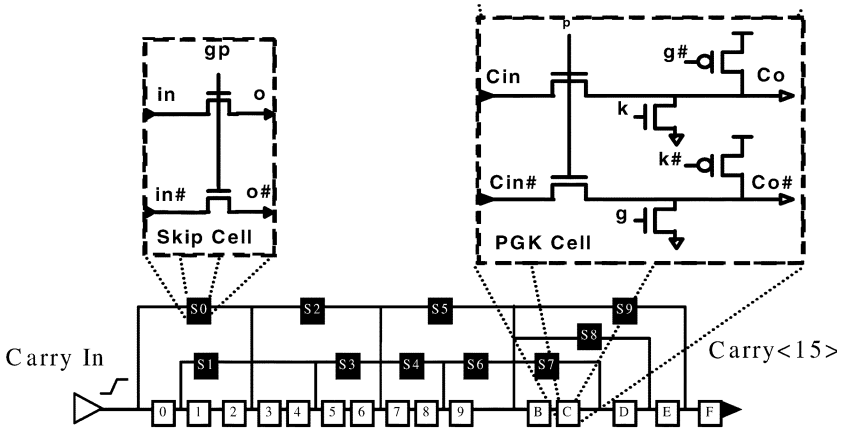


Fig. 10. LVS 16-bit adder carry-skip chain.

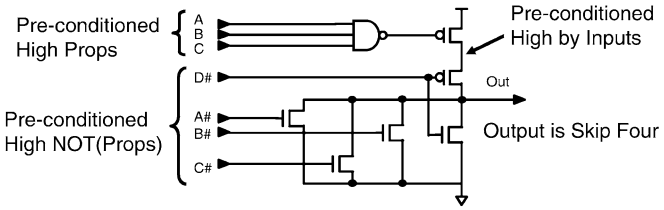


Fig. 11. P-interruptable ratio'd NOR gate used to generate group propagate (skip) control signals.

(LVSLRC) enforces adherence to matching rules. The checker guarantees, during construction, a one-to-one layout geometry matching of differential nodes. Included during analysis are process patterning, variation issues, consistent shielding and geometric forcing of all signal attackers to be common mode. In this way, essentially random logic is laid out to have minimal differential noise. Pre-silicon the results from post layout extraction tools are used to verify the resulting layout quality.

## VI. Implementation Examples

The integer core has four main LVS structures. The L1 data cache alignment mux uses the core clock to produce 32:1 data alignment in less than one phase of a $2\times$ frequency clock. There are two ALUs. ALU0 has a 32-bit LVS adder and LVS Boolean logic functions. ALU1 has the same 32-bit LVS adder and a fast 32-bit LVS shifter/rotator [7]. Low-latency integer shifts and rotates were enabled by LVS technology and are new to the architecture. The fourth structure is the address generation unit. The AGU pipeline cascades back-to-back LVS circuits through four phases of the $2\times$ frequency clock. There are several LVS adders and carry chains in the block all of which share a common topology with the ALU adder. These four LVS structures make up over 20% of the integer core being described.

### A. 32:1 Data Alignment Mux

LVS technology enables the nontraditional implementation of the L1 data alignment mux shown in Fig. 9. The circuit topology includes 128 individual 32:1 dual rail muxes. This function is accomplished in a single stage of logic by distributing the mux node over half the height of the L1. The large capacitance of the mux node and the large resistance of the distributed network results in small signal that is amplified to rail by the two gain stages described previously. Duplicating and distributing the mux reset and select logic along the height of the block ensures signal integrity of the large signal control. This single-stage alignment mux, operating in less than one $2\times$ clock phase, directly addresses a critical point in the integer core load loop described earlier.

### B. Adders

The carry propagation *RC* delays for a 32-bit LVS adder are too slow to sustain the frequency targets of the integer core. For this reason, all 32-bit ALU and AGU adders are built upon a common 16-bit LVS adder core. The 32-bit ALU for example employs three of these cores in a carry-select configuration. This common core is allocated only one $2\times$ clock phase to compute a 16-bit add. Thus, the add portion of the AGU and ALU operations in actuality are operating at twice the quoted frequency of the integer core.

*1) Carry Chain:* The LVS carry chain for a 16-bit adder is shown in Fig. 10. It is built upon 16-cascaded LVS PGK cells 0-F with carry-skip pass gates S0–S9 positioned such that any carry propagation path traverses through no more than six series devices. The LVS cells that make up the LVS carry-chain are basic pass-gate logic configurations with generate and kill transistors at each bit position.

The LVS XOR gates hookup to each polarity of the carry$[n]$ nodes along the chain to produce the sum $[n+1]$ result. The typical critical path begins with the turning ON of the s0 skip pass gate that allows the "Cin" to charge up the reset-low carry-chain and develop differential at the inputs of 17 PMOS sense amplifiers that sense the 16 sum and final carry results along this structure.

*2) Solving the Critical Path:* Typically, the adder critical path goes through the skip gate controls which are formed by ANDing together the propagates being skipped over. These group propagates generated by wide NORs by definition come one gate delay after the propagates themselves. A novel p-interruptible ratio'd-NOR gate (RP-NOR), illustrated in Fig. 11, was designed to implement fast NOR2–NOR5 gates. The evaluate edge for this gate is extremely fast when compared to static NOR structures since the P stack is only two high.

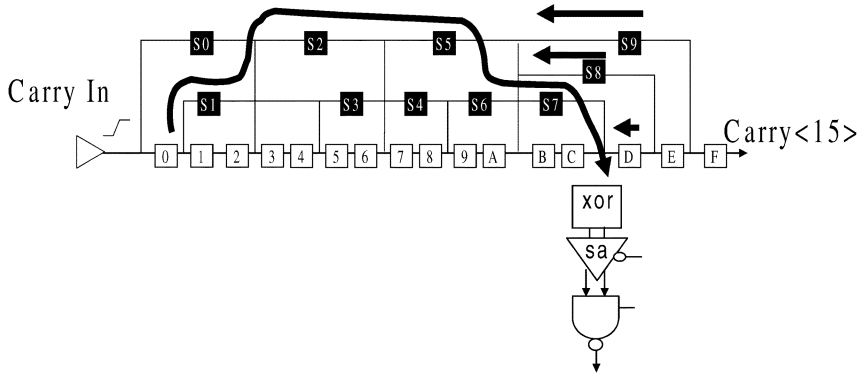


Fig. 12. Example of a differential noise scenario for the 16-bit adder carry-chain simulated and analyzed using in-house tools.

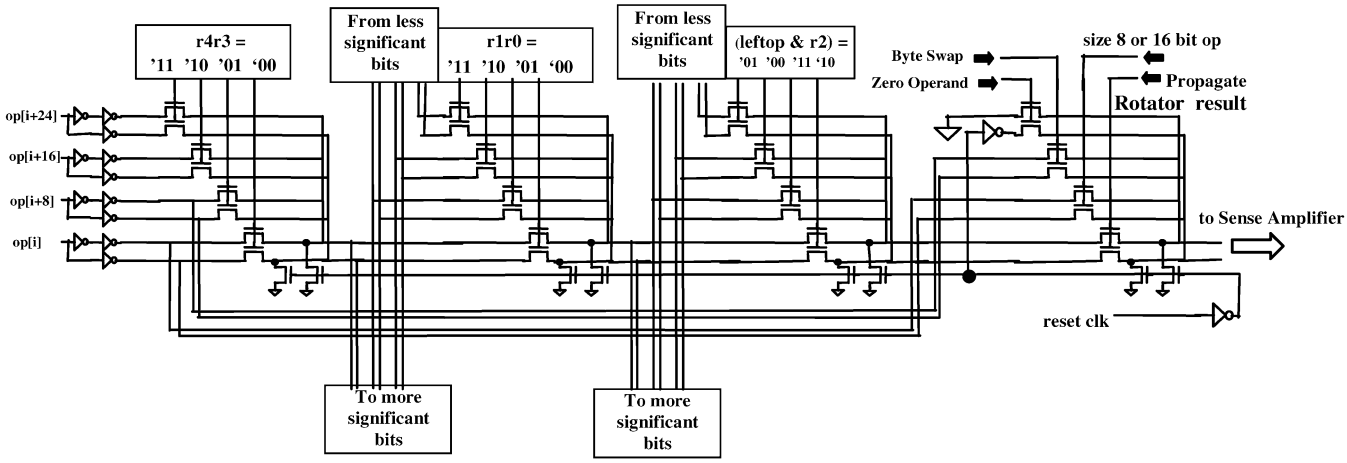


Fig. 13. Bit slice of the LVS rotator data path.

However, for certain input combinations meant to produce an output low level, the pull-up and pull-down networks are on simultaneously. For these cases, the gate's output can produce a steady-state noise source unacceptable for controlling a LVS DCN. Thus, pre-charged high true and complement inputs are used to logically control an interrupt device in the p-stack disabling contention within one gate delay. To the DCN carry chain, this creates a differential noise source during a narrow pulse approximately one gate delay wide. This noise pulse occurs at the onset of LVS evaluation leaving the carry-chain a significant amount of time to recover and develop positive differential unimpeded by any further contention-induced differential noise.

*3) Differential Noise Example:* Illustrated in Fig. 12 is an example of a differential noise scenario handled during pre-silicon analysis by the in-house LVSTNT tool. Shown is a generate at bit 0 in the carry chain driving signal to a sum at bit 12. Multiple devices that are digitally off can drive small amounts of differential noise into the network close to the sense amp. This is a very dangerous scenario where multiple back contention sources near the sense amp may add up to be large enough that the driver at the other end of the resistive network is unable to overcome the differential noise at any frequency. These types of noise scenarios are very concerning and are a primary motivation for creating the in house noise tool and for limiting the DCN to six series gates.

### *C. Rotator*

The ALU rotator/shifter circuits can be thought of as a series of muxes wired with long interconnect. These muxes steer an operand over the data path length and width. The mux nature and the distributed RC make a rotator topology particularly suited to LVS technology.

Fig. 13 shows the bit slice of the LVS rotator data path. Full swing single ended operands are converted to true and complementary signals with a local inverter and steered through four stages of differential 4-to-1 muxes implemented as LVS DCNs.

The DCN control signals calculated from the shift count logic are pre-charged low and are driven by static stages. The monotonic switching of the control signals enables time borrowing into the evaluation phase.

The operand being steered through the datapath and the shift (or rotate) count signals described above present themselves at the ALU1 inputs simultaneously. A LVS source select mux similar to the structure shown in Fig. 4 selects the count bits from one of the five possible sources and performs a full decode. After the source mux a complex CDL gain stage and following sense amplifier completes the full decode. The resulting control signals, after one static logic stage are used as selects for the series of 4-to-1 LVS muxes.

The rotator datapath described above produces two outputs per bit, a "propvalue" and a "killvalue." The propvalue is the

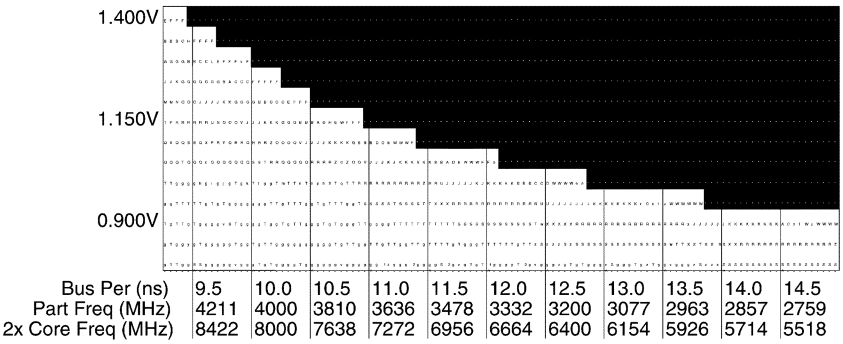


Fig. 14. Post silicon voltage–frequency shmoo plot of isolated FCLK (2× freq.) integer core demonstarting 8 GHz, 70 C, 1.3 V operation (450 directed patterns).

actual rotated or shifted data, and the killvalue represents the 0 or the size-based most significant bit. A separate LVS DCN (up to six pass gates deep) implements the complex shifter logic and produces the kill signal. A complex 2-to-1 CDL mux (Fig. 7.) selects between the two small signal data signals (propvalue and killvalue) based on the small-signal select (kill) to produce the rotated (or shifted) result.

This rotator/shifter can logically be described as 32 sets of 32-to-1 muxes. The time allocated is about one and a half nominal inverter delays. A traditional domino or static implementation has too many logic stages to fit in this design space without incurring large area, static power, and power race complexities.

## VII. Results

The processor with the described LVS integer core has been fabricated on 90-nm technology. The processor includes test hardware that isolates the integer core on silicon. Fig. 14 shows the isolated core's voltage versus frequency shmoo running all directed tests.

The LVS integer core exceeds 8 GHz on initial silicon at 70 C junction and 1.30 V pin. The core is expected to reach $>10$ GHz with process and post silicon optimizations. The integer core contains over 6.8 million nonarray physical transistors. The core area is 14070.8 sq. mils.

## VIII. Conclusion

This paper covered the usage of low-voltage swing technology to achieve very high-frequency circuits. These circuits have been used to build a 2× frequency integer core that extends the processor onto 90-nm technology. The design has been fabricated and is in production. Measured silicon results of the integer core hit frequency expectations and show headroom to scale further with the process.

## Acknowledgment

The authors would like to acknowledge the efforts of the many design engineers, physical design specialists, and tool automation engineers at Intel in Hillsboro, OR, who contributed to the implementation of LVS technology. Their dedication, hard work, and attention to quality have been outstanding! We thank them for allowing us the opportunity to share their results.

## References

[1] D. Sager *et al.*, "A 0.18 $\mu$m CMOS IA32 microprocessor with a 4 GHz integer execution unit," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2001, pp. 324–325, 461.

[2] S. Thompson *et al.*, "A 90 nm technology featuring 50 nm strained silicon channel transistors. . .," in *IEDM Tech. Dig.*, 2002, pp. 61–64.

[3] D. Deleganes *et al.*, "Low-voltage-swing logic circuits for a 7GHz x86 integer core," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2004, p. 154.

[4] J. Schutz and C. Webb, "A scalable x86 CPU design for 90 nm process," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, Feb. 2004, p. 62.

[5] T. Sakurai *et al.*, "Low-power CMOS design through Vth control and low-swing circuits," in *Proc. Int. Symp. Low Power Electronics and Design (ISLPED)*, Aug. 1997, pp. 1–6.

[6] M. Morrise *et al.*, " Algorithm for finding vectors to stimulate all paths and arcs through an LVS gate," U.S. Patent 6,557,149, Apr. 29, 2003.

[7] A. Singh *et al.*, "A mixed signal rotator/shifter for 8GHz Intel Pentium 4 integer core," in *Symp. VLSI Circuits*, June 2004, pp. 394–397.

[8] D. Deleganes *et al.*, "LVS technology for the Intel Pentium 4 processor on 90 nm technology," *Intel Technol. J.*, Feb. 2004.

**Daniel J. Deleganes** received the B.S.E.E. degree from the University of Washington, Seattle, in 1988 and the M.S.E.E. degree from Cornell University, Ithaca, NY, in 1989.

He is the Project Manager and Technical Lead of the Intel Pentium 4 processor integer execution cluster. He technically directed the LVS program from basic feasibility through actual implementation. At Intel, he contributed to the designs of high-speed Intel i486 and Intel Pentium processor second level cache SRAM families, several Pentium processor generations, and all Pentium 4 processor generations. He presently is involved with circuit technology and methodology path finding for Intel's next-generation microprocessor.

**Micah Barany** received the B.S. degree in engineering physics from the University of California, San Diego, in 1989, and the M.S.E.E. degree from Stanford University, Stanford, CA, in 1993.

He is the Micro Architecture Manager of the Pentium 4 processor integer execution cluster. At Intel, he contributed to Intel i386 SX and Intel i486 SL microprocessor families, and several Pentium and Pentium 4 processor generations.

**George L. Geannopoulos** received the B.S.E.E. degree from the University of Illinois at Urbana-Champaign.

He co-managed the LVS design team and is currently managing the PLL team for a Pentium 4 processor proliferation. He joined Intel in 1994. His interests include high-speed circuits, PLL, clock generation and analog design. Prior to joining Intel, he worked at Bipolar Integrated Technologies as a Design Manager designing VLSI ECL RISC FPUs, FPCs, and register files. He also worked at MMI/AMD designing programmable array logic (PALs).

**Kurt Kreitzer** received the B.S.C.E. degree from Oregon State University, Corvallis, in 1994.

He currently manages the LVS design team at Intel. During initial design phases, he drove the specification and development of the LVS design tools and methodologies. After working on the Pentium Pro, he created the modular SRAM array used throughout the Pentium 4 and other projects and implemented the Pentium 4 Trace Cache.

**Matt Morrise** received the B.S. and M.S. degrees in mathematics from Brigham Young University, Provo, UT, in 1981 and 1984, respectively.

He designed and coded the timing tool used on all LVS blocks at Intel. Prior to joining Intel, he worked at Signetics Corporation on a variety of CAD tools. He now develops timing and clocking tools for future microprocessor projects at Intel.

**Dan Milliron** received the B.S.E.E. and B.S.C.S degrees from the Massachusetts Institute of Technology, Cambridge, in 1986.

He is a software engineer developing CAD tools for microprocessor design. He joined Intel in 2000 and began working on the LVS timing and noise analysis tool, concentrating on parallel simulation processing and dynamic noise analysis. He continues to contribute at Intel to software architecture and algorithm design for future microprocessor projects.

**Anant P. Singh** received the B.S.E.E. degree from Delhi University, India, and the M.S.E.E. degree from the University of Washington, Seattle.

He is a Senior Designer on the LVS team at Intel. His recent focus is in mixed-signal design where he has worked to define, implement, and productize LVS circuit technology. Previously, he contributed circuit designs on Pentium III processors. Prior to joining Intel, he worked in the field of control systems and automation.

**Sapumal B. Wijeratne** received the B.S.E.E. degree from Lafayette College, Easton, PA, in 1984 and the M.S.E.E. degree from Purdue University, West Lafayette, IN, in 1986.

He co-designed the LVS AGU/ALUs. Prior to this work, he held numerous technical leadership positions including methodology lead for domino circuits and register files. He is currently co-managing the an integer execution core design for a Pentium 4 microprocessor proliferation at Intel.