EXHIBIT 1047

A. F. MURRAY, and P. B. DENYER, “A CMOS Design Strategy for Bit-Serial Signal Processing,”

IEEE JSSC, Vol. 20, Issue 3, pp. 746-753 (1985)

TRW Automotive U.S. LLC: EXHIBIT 1047
PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NUMBER 8,599,001
IPR2015-00436

# A CMOS Design Strategy for Bit-Serial Signal Processing

ALAN F. MURRAY AND PETER B. DENYER, MEMBER, IEEE

*Abstract* —We present a summary of the features and successes of a Silicon Compiler (FIRST) for LSI nMOS bit-serial signal processors. A replacement cell library of CMOS operators has been designed for the compilation of true VLSI bit-serial signal processors. The cell library is implemented in 2.5-μm bulk CMOS technology, and maintains a consistent performance of 20 MHz. We describe the design philosophy and style behind the CMOS cells, detailing the dynamic logic style used, its layout and testability. As an example of the capability of the library, we discuss a full-precision complex multiplier.

## I. Introduction

DESIGN complexity has become a dominant cost limit in the development of VLSI systems. Without new design methods and tools, advances in manufacturing capability and algorithm development will far exceed our capacity for design [1]. This effect is nowhere more pronounced than in the field of real-time signal processing; the continuous flow of data together with the complexity of many of the algorithms, imposes severe computational demands that often cannot be satisfied by general purpose machines or components.

We have addressed the issue of design complexity through the development of a powerful system synthesis tool—a silicon compiler, called FIRST. FIRST is more specialized than are most other silicon compilation/autolayout systems. This restricts its application (to real-time signal processing tasks), but at the same time permits functional integration density that is closer to hand-crafted custom than to other examples of automatic layout generation.

We describe a cell library implemented in 2.5-μm bulk CMOS technology (two-layer metal) whose operators are amenable to either silicon compilation or manual assembly to generate bit-serial signal processing functions, analogous to those served by FIRST.

Section II of this paper constitutes a summary of the aims, capabilities and achievements of the FIRST Silicon Compiler, as implemented in 5-μm nMOS technology. Section III describes the reimplementation of the cell library in 2.5-μm CMOS technology, giving details of the design and layout style, and illustrates the library's scope and performance by detailing the design of two of the major operators. Section IV shows how the inherently high level of testability exhibited by bit-serial primitives is carried through into the normally problematic regime of stuck-open faults in CMOS by the use of a dynamic design style.

## II. The FIRST Silicon Compiler

FIRST synthesizes signal processing chips and systems using exclusively bit-serial architectures. This architectural restriction has significant consequences in the simplicity and success of the environment. We shall later identify how restrictions and conventions such as these are key to a successful design methodology.

Bit-serial architectures [2] have major advantages over bit-parallel alternatives, especially in the areas of signal routing and communication (issues which are often important in signal processing applications), and in efficiency of computation through bit-level pipelining. These features are exemplified in a range of case studies given later.

### A. An Example

We begin with a simple example system to give an impression of the scope and use of FIRST. For the purpose of this example we consider an implementation of the four-region approximation [3] to the magnitude $M$ of a complex number $A+jB$

$$M=\text{greater}\left[\left(\frac{7G}{8}+\frac{L}{2}\right),G\right]$$

where,

$$G=\text{greater}[|A|,|B|]$$
$$L=\text{lesser}[|A|,|B|]. \qquad (1)$$

We view this algorithm as a functional flow-graph, as shown in Fig. 1. The oblong shapes are functional elements that perform fixed operations on the (bit-serial) data flowing through them. Now bit-serial elements exhibit a delay or latency of integer numbers of bit-times. It is important to equalize these delays and synchronize data paths entering each element. Simple delay elements are shown as circles in Fig. 1. We have deliberately reduced this algorithm to a network of functional operators (add, scale, delay, etc.) which are supported by FIRST. We call this the set of *primitive* operators. FIRST supports primitives from

Manuscript received November 5, 1984; revised January 16, 1985.

The authors are with the Department of Electrical Engineering, University of Edinburgh, The King's Buildings, Mayfield Road, Edinburgh EH9 3JL, Scotland.

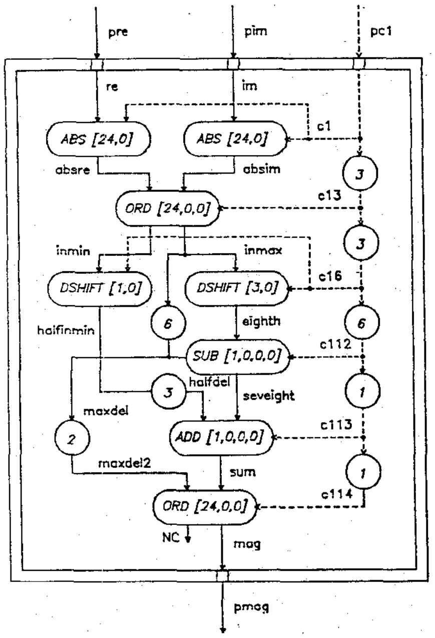

Fig. 1. Flow-graph of complex-to-magnitude example. Simple delay elements are shown as circles, while computational operators are represented by name.

which all systems are constructed which form a finite set of relatively high-level functions, well above the transistor or gate level.

FIRST is a tool which takes this functional flow-graph as input, instantiates the required set of bit-serial primitives, assembles these primitives, and routes the network of flow-graph connections between them. Finally, FIRST wires input/output pads, and power and clock services to complete a custom chip design. In effect this environment acts as a compiler, delivering low-level code (in this case VLSI mask artwork) from an initial high-level program (the flow-graph), while simultaneously simulating the chip's function.

The corresponding FIRST implementation of the complex-to-magnitude algorithm in silicon is shown in Fig. 2. As a VLSI device this is not particularly impressive; it contains only a few thousand transistors. However, as a demonstration of the design methodology it is; our design cycle was very fast and required no intimate knowledge of integrated circuit design.

The primitives themselves are irregular blocks of high-density layout, with *i/o* strategies configured to suit the routing channel convention. However, they are not held in FIRST as single "macrocells," but rather as collections of leaf cells and procedures for their assembly. In this way the primitive set is parameterized so that users may select for example, the size of a multiplier, the length of a delay element, etc.

This simple floor-plan style has proved effective in efficiently implementing many of the systems we have attempted. Later we show a selection of such designs which exemplify this feature. We find a typical overhead of

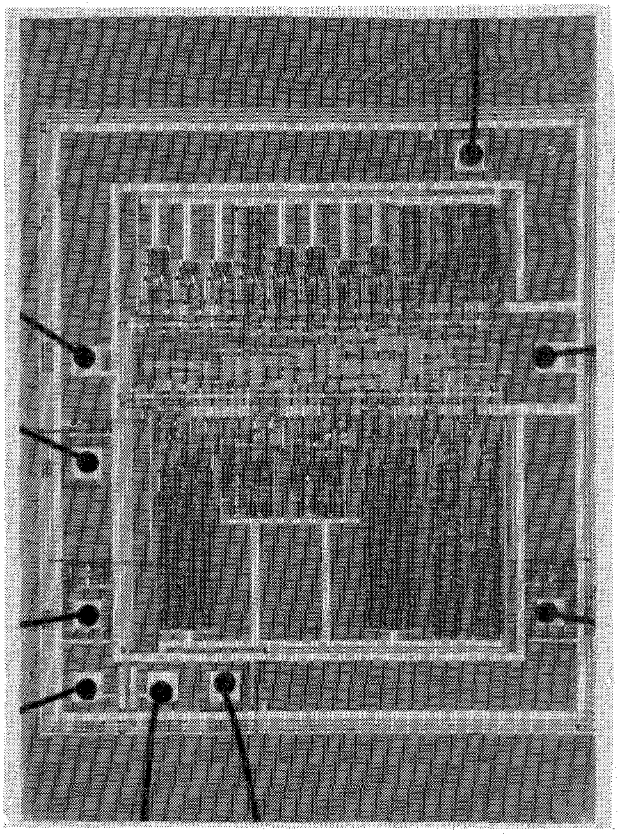

Fig. 2. FIRST (5-$\mu$m nMOS) chip synthesized from Fig. 1.

### B. Secrets of Success

The nMOS version of FIRST has been in use at Edinburgh University for some 18 months. Several designers have completed complex system designs in remarkably short timescales, vindicating our aims and approach.

We begin by setting rigorous conventions covering all aspects of the timing, electrical and numerical formats of signals throughout the system. If we now implement a base set of primitives that uniformly respect these conventions, then any connection of these primitives will communicate successfully, without requiring any further design attention to be devoted to the individual interfaces. Thus we can connect a flow-graph of elements, as in Fig. 1, and automatically achieve correct hardware functionality.

These advantages extend to elegant hierarchical design, because the communication conventions are automatically inherited by any network or subnetwork built exclusively from such components. Thus we are able to construct higher level operators as general hardware "routines."

These features make for a very speedy system design environment.

### C. A Resume of Case Studies

Table I summarizes the scope of applications addressed by FIRST as a selection of case system design studies. The information on transistor and chip counts relates to 5-$\mu$m nMOS technology. The advent of a CMOS primitive library (as discussed below) dramatically improves packing density

TABLE I
FIRST SYSTEM CASE STUDIES

| System | Computation MegaOps/sec | Chip Designs | Total Chips | Transistors (K) | FIRST Code (lines) |
|---|---|---|---|---|---|
| Speech Echo Canceller | 55 | 3 | 18 | 252 | 250 |
| Adaptive Lattice Filter | 10 | 5 | 5 | 30 | 500 |
| LDI Filter 5th Order | 15 | 1 | 1 | 7.5 | 160 |
| FFT 16-point 8 MHz | 150 | 6 | 43 | 160 | 300 |

Most significantly this table shows that impressive total computation rates can be realized by many slow bit-serial elements operating in parallel. The final two columns provide a comparison of the size of the system description code (FIRST input), and the transistor count of the resulting system.

## III. RE-IMPLEMENTATION IN CMOS TECHNOLOGY

Significant gains may be expected in mapping the techniques that we have demonstrated in nMOS technology, to an advanced 2.5-$\mu$m CMOS technology. Our design style uses a mixture of dynamic and static logic, using dynamic techniques where the benefits of the increased speed and reduction in transistor count (and therefore silicon area) can be realized and are significant. Where the fan-in of a logic gate is low, a static implementation may involve no more transistors than a dynamic one, particularly if the number of minterms in the Boolean expansion of the gate's function is also low. In such cases we use a static gate, to avoid additional clock distribution. We use dynamic latches everywhere. The dynamic logic structure used is based on that known variously as "NP" or "NORA" CMOS, modified to render it less sensitive to clock edge times [4], [5].

### *A. Design Style*

The logic evaluatory section of a gate consists of either n- or p-type transistors. The desired function is evaluated by "discharging" a precharged capacitance, conditional on either a positive logic (n-gate) or negative logic (p-gate) function of the gate's input variables. The technique is related to the "Domino CMOS" style, in which only n-gates are involved [6]. The mixed-gate approach is superior in that n- and p-gates may be cascaded without the need for the intervening inverter of Domino CMOS, and greater versatility is brought about by the positive and negative logic functions. In fact n-p, Domino, and static gates may be mixed, provided the rules for inclusion of each type are obeyed.

Fig. 3(a) shows an exclusive-NOR (XNOR) gate taken

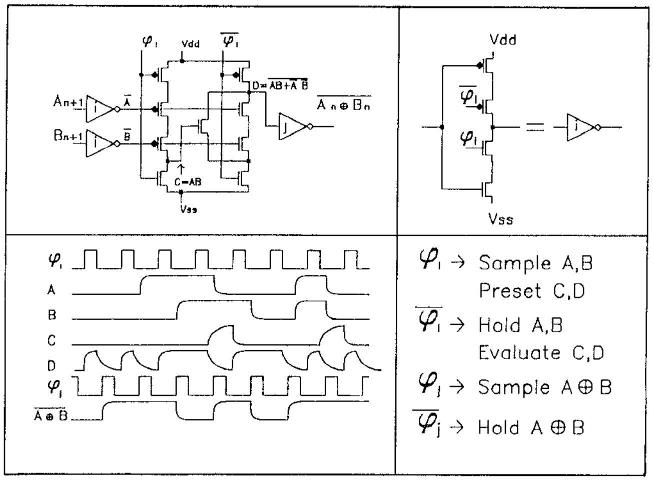

Fig. 3. Dynamic (mixed-gate) exclusıve NOR gate.

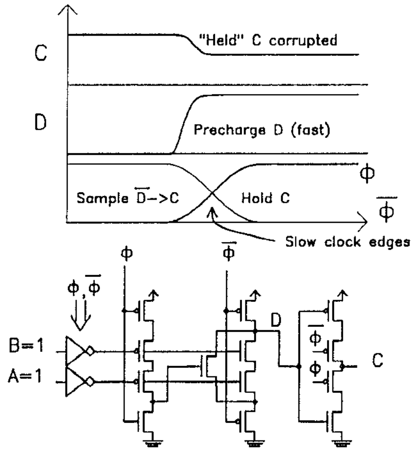

Fig 4. Data corruption due to slow clock edges in NORA/NP-CMOS.

used can be seen from Fig. 3(b) to consist of a "clocked inverter." Input values are sampled during $\phi i$, during which time the nodes $C$ and $D$ are precharged (or, more correctly, preset) low and high, respectively. During $\bar{\phi} i$, these nodes are conditionally "discharged," depending on the voltages on the gates of the transistors in the evaluation logic tree.

During the evaluation phase ($\phi i = 0$), node $C$ is pulled high if $\bar{A}$ and $\bar{B}$ are both low, forming the minterm $AB$. Simultaneously, node $D$ is discharged, conditional on either $C$ or both $\bar{A}$ and $\bar{B}$ being high, forming the exclusive OR function. During $\phi j$, this value is sampled by the clocked inverter to give a "held" XNOR during $\bar{\phi} j$. The effect of this is shown in Fig. 3(c) and 3(d). In common with all synchronous design systems, the output of a complete gate is presented one full clock cycle after its inputs. In addition to the $\phi i \rightarrow$ [n-p-gates] $\rightarrow \phi j$ ordering illustrated, gates may also be constructed which sample inputs on $\phi j$, and change outputs on $\phi i$. Furthermore, n- and p-sections may be cascaded internal to a gate provided ripple through times do not become excessive. With these capabilities, extremely tight pipelining of logic may be achieved.

In the original n-p or NORA scheme, $\phi j$ is simply $\bar{\phi} i$.

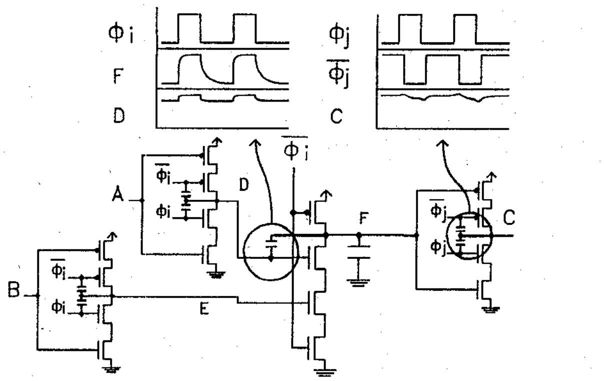

Fig. 5. Corruption of dynamically stored logic values due to capacitive (gate–drain) coupling.

that slow clock edges can result in corruption of the held output values from a gate, as the precharge nodes begin to be precharged before the output is fully held. This effect is shown by SPICE simulation in Fig. 4. The onset of the precharge on node "*D*" in the XNOR circuit of Fig. 4 causes a corruption of the incompletely "held" value on node *C* (the output of the clocked invertor), if the clock edges are "slow." Our simulations show that clock rise/fall times faster than 3 ns are required (appreciably less than a typical gate delay) if undesirably complicated rules for cell design are to be avoided.

Caution must, however, be exercised when designing dynamic sections even when our more conservative two-phase (plus inverses) nonoverlapping clocking scheme is adopted. The major effects to be avoided are as follows.

*Charge Redistribution*

Charge sharing can occur between the nodes of an n- or p-section which has inputs from a preceding p- or n-section (i.e., from an "internal" precharge/discharge node, rather than from a fully sampled and held node). This may be avoided by careful placement of such inputs within the logic evaluation tree such that they are as far as possible from the precharge node of the gate [5].

*Corruption of Sampled Values at the Inputs to a Dynamic Section*

Capacitive coupling (gate–drain) to the precharge node of a gate can cause corruption of the dynamically held values at the gate's inputs. In Fig. 5, the effect of a rapid pulldown of node *F* is shown, in the partial corruption of the dynamically held logical 1 on node *D*. Choice of transistor sizes within a logic tree matched to the capacitance of the input node minimizes this effect.

*Clock Breakthrough Leading to "Charge-Pumping"*

The effect of gate–drain coupling at a dynamic latch (ie., gate–drain coupling at the clocked transistors) can lead to insignificant, but creates the circumstances within which latchup can occur. This effect only comes into play when there is some stagger between the clock and its inverse, which there usually will be. The result is that the clocked inverter's output ripples around its correct value. In Fig. 5, an example of this form of charge-pumping is shown, where node *C* is made to ripple around its correct value of 5 V by the breakthrough of $\phi j$ and $\bar{\phi} j$. Again, carefully chosen transistor sizes minimize the coupling while preserving speed.

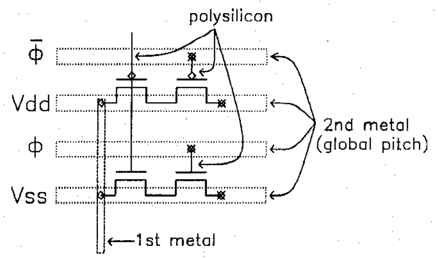

Fig. 6. Gate-matrix layout of clocked inverter.

These rules are not difficult to apply, and the clock signals are not as difficult to generate or distribute over the device, as would be signals requiring a maximum limit of 3 ns on clock edges.

### B. Layout Style

A structured layout style is used for the design of the cell library, to ease checking and to impose some consistency in the layout of low-level gates. The layout style is a relaxed version of that known as "gate-matrix" layout [7], within which the p-channel devices of a logic gate are grouped together and the n-channel devices are similarly grouped to minimize the effect of the large minimum p-channel-to-n-channel device separation [7]. Furthermore, the devices are arranged in rows with the source–drain direction horizontal (say), such that polysilicon gate "wires" may run vertically between p- and n-channel rows.

The resultant layout is dense, neat, and amenable to pitch-matching between gates, particularly with regard to power and clock distribution. In Fig. 6, a schematic representation is given, showing:

a) (mainly horizontal) strips of transistors forming the logic, or as in this example, the clocked invertor structure;
b) (mainly vertical) polysilicon, forming transistor gates and short-range intergate connections;
c) (mainly vertical) first layer metal, for longer-distance intercell interconnect, and to span the p-type to n-type boundary;
d) (almost exclusively horizontal) second-layer metal, for supply and clock lines, utilizing a global vertical pitch. Thus, low-level entities fit together like LEGO bricks, recouping in denser inter-cell interconnect

# Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

## Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

## Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

## Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

## API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

**LAW FIRMS**

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

**FINANCIAL INSTITUTIONS**

Litigation and bankruptcy checks for companies and debtors.

**E-DISCOVERY AND LEGAL VENDORS**

Sync your system to PACER to automate legal marketing.

**WHAT WILL YOU BUILD?** | **sales@docketalarm.com** | **1-866-77-FASTCASE**