SUBNANOSECOND ARITHMETIC
Second Report

Michael J. Flynn
Giovanni De Micheli
Robert Dutton
R. Fabian Pease
Bruce Wooley

Technical Report CSL-TR-91-481

June 1991

Supported by the NSF under contract MTP 88-22961.

SUBNANOSECOND ARITHMETIC

Second Report

by
M. Flynn, G. De Micheli, R. Dutton,
F. Pease and B. Wooley

Technical Report CSL-TR-91-4xx
June 1991

Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
St anford University
Stanford, California 94305-4055

Abstract

The Stanford Nanosecond Arithmetic Project is targeted at realizing an arithmetic processor with performance approximately an order of magnitude faster than currently available technology. The realization of SNAP is predicated on an interdisciplinary approach and effort spanning research in algorithms, data representation, CAD, circuits and devices, and packaging. SNAP is visualized as an arithmetic coprocessor implemented on an active substrate containing several chips, each of which realize a particular arithmetic function. This year's report highlights recent results in the area of wave pipelining. We have fabricated a number of prototype die, implementing a multiplier slice. Cycle times below 5 ns were realized.

Key Words and Phrases: High-speed arithmetic, active substrate, BiCMOS, computer-generated layout, floating point, high level primitives, leading-one prediction, microcontacts, photoconduction, wave pipelining.

Contents

Overview

The Stanford nanosecond arithmetic processor (SNAP) project was begun more than two years ago in order to create a basic technology in high-speed arithmetic which would provide significant performance advantages in the area of floating-point arithmetic and arithmetic coprocessors.

Our original design remains unchanged. It consists of multiple chips on an active substrate (wafer). Each chip represents a functional unit in a high-speed arithmetic coprocessor. The functional units originally identified include (together with their target execution delays):

1. Floating-point adder (2-3 nsec).
2. Floating-point multiplier (5-8 nsec).
3. Floating-point divider (IO-15 nsec).
4. Register file (I-2nsec).
5. Control unit.
6. Integer unit (l-2 nsec).

Moreover, we had as an objective a one-nanosecond repetition rate made for communication between chips and for pipeline rates across chips.

As we enter the final year of our NSF contract we believe we have made significant progress toward achieving our very ambitious goals. During the past year we have completed our test chip based on the concept of wave pipelining, which uses delays intrinsic to the circuits themselves as a storage element in high-speed pipelined functional implementations. Wave pipelining is the target technology that we see as allowing us to achieve extremely high pipeline rates with functional units. We are currently measuring our first waveline implementation (a slice from a large multiplier). Simulations predict that two or more pipeline waves are possible, thus doubling the normal clock frequency without additional registers.

We have continued to work on all specified functional units. With the kind cooperation of several industrial partners we have arranged to build a CMOS floating-point adder and a BiCMOS floating-point multiplier. We are still in the process of soliciting corporate interest in a purely bipolar implementation of another floating-point multiplier design.

Our earlier and continuing work in CAD provided the basis for our wave pipelining implementations. Our CAD system and tools were able to automatically pad out, i.e., extend, short paths in logic and provide sufficient storage for us to achieve significantly improved repetition rates across logical functions. We continue to develop other function-specific CAD tools such as optimized routers for floating-point multiplication.

In the area of circuit design we focus on the BiCMOS technology. Longer term we believe that a marriage of CMOS and bipolar technology (probably current mode bipolar) will become a dominant technology in a high-speed area.

The design for our register file involves critical circuit design issues. The register file must be able to accommodate a minimum of 3 accesses per cycle from a functional unit in order to match that unit's execution speed. If two or more functional units are allowed to execute at the same time, the register bandwidth must be even higher. Thus, the register design problem involves issues of access time, cycle time, and number of access ports. We are currently building a test BiCMOS register file chip with multiple ports having access time approaching 1 ns. We are also exploring means by which large register files can be packaged without the I/O delays that severely compromise their performance.

The packaging area is another key ingredient in the SNAP project. We have focused on a design in which the functional unit chip was fastened to the active substrate by the surface tension of a thermally conductive fluid. This solves both the attachment and any potential thermal stress problems that might develop. Electrical contacts are to be made by microcontacts. Figure 1 contrasts the conventional contact with a microcontact. The extremely small area at the end of the microcontact creates very large forces on the substrate base pad, ensuring good electrical contact between the chip and the substrate. Microcontacts also offer the possibility of significantly greater contact density: 10 vs. 50-100 microns with conventional contacts. Over the past year we have looked at the issue of microcontact and noise.

We have also begun to look at busing technologies which could be available along the substrate that might support extremely fast signal propagation using high temperature superconductor signal propagation technology.

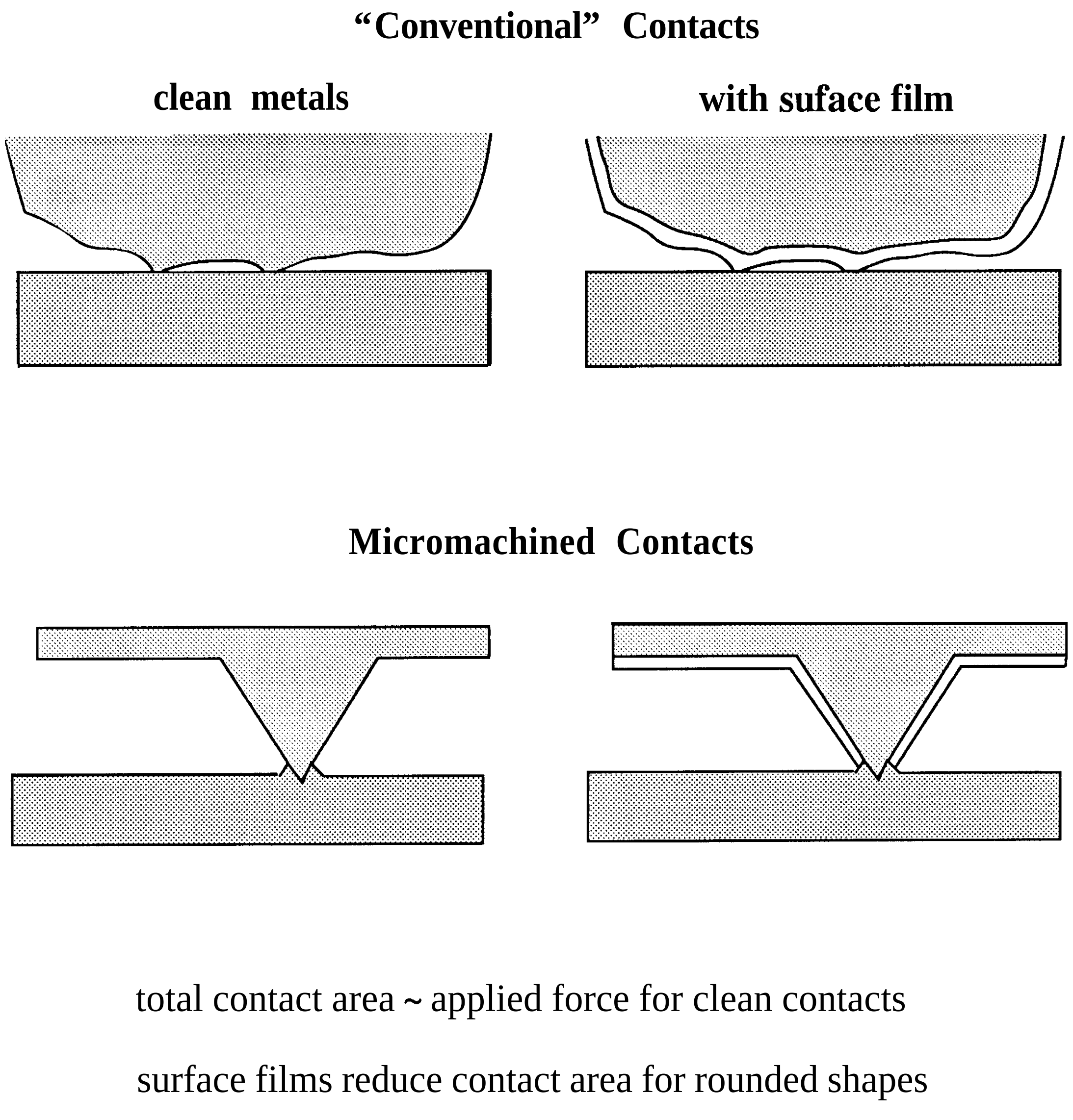

“Conventional” Contacts
clean metals
with suface film
Micromachined Contacts
total contact area ~ applied force for clean contacts
surface films reduce contact area for rounded shapes

Figure 1: Microcontacts.

1 Algorithms and Functional Units

1.1 Representation

1.1.1 IEEE **Rounding** (N. Quach)

Not all numbers are representable in a computer because of limited hardware precision. Rounding is an operation that maps an unrepresentable number into a representable one. The IEEE 754 standard is the current standard for rounding. A rounding method has been developed and described in a technical report, CSL-TR-91-459. In this method, rounding is cast as a prediction problem. The number of prediction schemes determines the total number of rounding schemes for a given hardware model.

Our rounding method [QTF91] has the following advantages:

- It is understandable, and therefore easily generalizable to other operations such as addition, division, and other number representations. (In the report, most emphasis has been placed on rounding for binary multipliers)
- It is possible to explore the solution space for a given hardware model.
- Implementations do not require additional hardware.

Analysis of the IEEE rounding modes for high-speed conventional binary multipliers reveals that round to infinity is more difficult to implement than the round to nearest mode; more adders are potentially needed. We have shown that redundant binary multipliers has two major advantages over conventional binary multipliers for rounding. First, the computation of the sticky bit consumes considerably less hardware. Second, implementing rounding to positive and minus infinity modes does not require the examination of the sticky bit, removing a possible worst-case path.

1.1.2 **Leading-One Prediction** (N. Quach)

Leading-one prediction (LOP) is a technique by which the number of leading zeros in a result can be determined in advance of its arrival. Such a circuit is crucial to achieving high speed in a floating-point adder. The description of past work on LOP in the literature is sketchy, at best.

In this work [QF91a] we develop the theory of LOP and then show two possible implementations. The first is similar to the one used in the IBM RS/6000 floating-point unit. The second is a more distributed version and consumes less hardware than the first. We then generalize the concept of LOP to a pattern detection problem, which includes the well-known carry lookahead in parallel addition. Through this generalization, techniques developed for adders may be applied for future pattern detection problems. Finally, we point out that the sticky bit computation in binary multipliers is also a pattern detection problem. We show an efficient way to compute the sticky bit. This work has been detailed in a technical report (CSL-TR-91-463).

1.2 Floating-Point Addition

1.2.1 IEEE Double Precision Floating-Point Adder
(N. Quach with R. Lee, HP Corp.)

A 53-bit floating-point adder is being designed and laid out, scheduled for fabrication at the Hewlett-Packard Corp. this summer. The algorithm used is an improved version of existing state-of-the-art algorithms. The algorithm has been described in Technical Report CSL-TR-90-442. The design of the adder itself will be described in another technical report when the adder is completed and evaluated.

Briefly, the adder being designed has the following desirable features:

- It has only a single significand addition step in the critical path.
- It consumes less hardware than most existing algorithms.
- It uses a modified Ling propagate adder.
- It uses an improved leading-one-detection (LOP) circuit.

In most conventional designs, parallelism is usually exploited by employing more hardware (e.g., more adders to compute all the possible outcomes). In the current design the parallelism is exploited through detailed analysis of the control and data flows; no extra hardware is used.

1.3 Floating-Point Multiplication

1.3.1 Multiplication Algorithms and Implementation (G. Bewick)

A new scheme for multiplication has been developed. This scheme is a variation of a high order Booth multiplication algorithm (i.e., one which requires the generation of '*difficult*' multiples of one of the operands, such as three times an operand-this normally requires the delay of a full-length carry propagate addition). This scheme uses a partially redundant representation for all ***of*** the *difficult* multiples. As a result the multiples can be generated using a series of small adders which all operate in parallel. These small adders are much faster than a single full-length adder. Using this scheme, most of the benefits of the smaller partial product reduction tree due to the higher order Booth algorithms are obtained, without having to pay the large time (and to a lesser extent, area) penalty of a full-length adder.

On the negative side, this algorithm has a physical implementation which is less regular than more conventional algorithms. In order to fully compare the different multiplication algorithms, an automated tool has been developed, which generates the physical layout of a multiplier tree, from general input parameters (such as the length of the multiplier desired, the algorithm to be used, etc.). The layout program uses information about wire delays and takes these delays into account as it lays out the multiplier tree. The result is a performance-driven layout, from which actual wiring capacitances can be obtained. Simulation runs can then compare the actual implementations of different multiplication algorithms. This is far superior to common comparisons done by completely ignoring wire delays, layout difficulties, and layout areas.

The layout program currently works with ECL (emitter coupled logic) 3/2 counters as the primitive element. The program is being modified to deal with CMOS counters (including sizing of drivers for longer wires). Modifications are also being implemented to allow the program to make use of some common circuit tricks (such as differential signals, current ramping in noncritical paths to reduce power) to give faster and lower power designs.

Preliminary results indicate that this new algorithm has about a 20% speed-power advantage over other algorithms, with about the same layout area (ECL implementations). Estimated area for a 53 by 53 multiplier tree (IEEE double precision) is about 4mm **x** 5mm using a process equivalent to 0.8 micron BiCMOS. The tree time is approximately 4-5 nsec, which when combined with a 2-nsec adder gives a multiplication time of 6-7 nsec. Further optimizations (as described above) have the potential of reducing the tree time to the vicinity of 2.5-4 nsecs.

At the moment CMOS designs have not been evaluated in detail, but it seems that the size of the multiplier trees generated by the layout program are comparable in size to trees described in the literature, using comparable technology.

1.3.2 Partial Product Reduction (P. Song, G. De Micheli)

An associated multiplier design project has led to the following results. We have developed two structures for partial-product reduction for IEEE standard floating-point multiplication, leading to structured layouts. Such layouts are desirable because they require shorter wires and therefore support faster operation. The former approach uses a reduction scheme based on a 28/5 counter followed by 5/5/4 counters. The latter uses a novel family of counters, called 9/2, 6/2, and 4/2, respectively. Both approaches use 3/2 counters as basic building blocks. They differ, however, in the way that the 3/2 counters are hierarchically combined to form the larger counters.

We have implemented and tested two vertical slices of a multiplier [DS91] using a 28/5 and a 5/5/4 counter. The blocks have been implemented in the *Sabre* technology of Signetics Inc. Circuits have been fabricated and tested successfully, showing a delay of about 10 nsec, including I/Os. These results are very encouraging, as they show that a fully parallel double-precision multiplier can be built with 12-15 nsec delay.

1 .4 Division

1.4.1 Fast Division Using A Reduced Number of Iterations (D. Wong)

We have developed a class of iterative integer division algorithms based on lookup table and Taylor-series approximations to the reciprocal. Using large lookup tables to estimate the reciprocal of the divisor to a certain number of bits, the algorithm can compute a 53-bit division in 4 or even 2 iterations. The algorithm iterates by using the reciprocal to find an approximate quotient and then subtracting the quotient multiplied by the divisor from the dividend to find a remaining dividend. Fast implementations can produce an average of either 14 or 27 bits per iteration, depending on whether the basic or advanced version of this method is implemented. Detailed theoretical analyses support the claimed accuracy per iteration. Speed estimates using state-of-the-art ECL components show that this method is faster than the Newton-Raphson technique and can produce 53-bit quotients of 53-bit numbers in about 28 or 22 ns for the basic and advanced versions [WF91].

In the next year, our group hopes to compare the various new division algorithms developed in the SNAP project and elsewhere and perhaps implement one of them in a demonstration chip.

1.4.2 The Expanded Redundancy Met hod for Division (E. Schwarz)

Research into unusual algorithms for division has been conducted with some very interesting findings. An algorithm for division based on Renato Stefanelli's work [1] has been enhanced and applied to high-radix (byte) division. This method is called the ***expanded redundancy method*** (ERM). It reduces the hardware requirements for division by replacing the quotient estimation hardware. Typically the quotient estimator is implemented with a ROM or PLA, but the ERM method uses a combinational network which creates a less accurate guess of the quotient.

Stefanelli describes division as the inversion of multiplication. The product of the quotient and the divisor is equal to the dividend. This multiplication can be expanded bit by bit into a partial product array. If a redundant notation is used for the quotient digits, then a restriction of no carry propagation between columns can be imposed. This creates separate linear equations, represented by summation of the columns of the array, which can be solved to yield the quotient digit values. Our algorithm proposes reductions to these equations and an efficient implementation.

Our study [2] has shown that this method requires slightly more iterations, but can be implemented in a very small area. On average the ERM requires 1.3% more iterations for a quadratic convergence division, 2.5% more for a constant convergence division, and 12% more for a nonrestoring high-radix division. This is in comparison to a ROM lookup-table implementation. In a CMOS implementation the ERM implementation requires only 320 gates for a division estimate or 122 gates for a reciprocal estimate and a ROM implementation would require 16K bytes or 128 bytes respectively. The ERM implementation offers an attractive alternative to using an off-chip ROM for the quotient estimate of a high-radix divider because of its small size and its small penalty in performance.

1.5 High Level Functions (M. Morf)

There has been increased interest in the use of high level arithmetic primitives to increase processor performance. These may be either joint primitives such as ***Multiply-Add*** or complex primitives such as ***sine*** or ***square root reciprocal.*** To support these studies work on mathematical and software tools, including symbolic computations, were carried out. Two complementary approaches were pursued to improve performance: the first speeds up a given set of hardware primitives and the matching arithmetic routines; the second attempts to find better primitives, hence, jointly optimizing primitives and arithmetic routines. The first approach serves as a baseline to explore incremental improvements, while the second approach attempts to achieve a more global optimization. It is in part motivated by a "top down" functional point of view, with an eye toward compiler support.

A number of applications were pursued, with several objectives in mind: to study the process of matching/mapping algorithms, architectures and arithmetic; to derive "natural" primitives; and to obtain statistical characterizations. Arithmetic studies included adders, multipliers, and symbolic computations (CORDIC algorithms, CMOS adders [5]). Device, circuit, and architecture modeling was a second set of topics. Some of the main tools to evaluate the performance of the various alternatives are statistical and analytic models for such parameters as speed, area, and power.

Higher level functions were further explored with the concept of ***information-preserving transformations*** (IPT's), e.g., CORDIC and Conservative Invertible (CI) functions, or X-gates (exchange-type gates, e.g., the Fredkin gate; see below). A number of these primitives have been explored with semiautomated algebraic and symbolic tools. General synthesis procedures continue to be developed for the algebraic level, logic level, and Finite State Machines (FSM). A major application of these concepts to high performance Lattice Gas (LG) computations, an alternative method for solving PDE's, has shown very promising results.

1.5.1 Accomplishments

Hardware Primitives. Speeding up the execution of scientific or signal processing (DSP) programs under power and chip area constraints involves speeding up frequently-used primitives (e.g., F-Add, F-Mult, and F-Div), reducing communication delays and decision bottlenecks, and speeding up arithmetic routines. Examples of analytic device models that have better than 10% accuracy can be obtained for delay and power estimates. For instance the inverse latency (computation rate) tends to be proportional to chip area and the square of the power, whereas the clock frequency (pipeline rate) tends to be inversely proportional to the area (per stage) and proportional to the power. Examples that support trend predictions are the observations that increasing parallelism in arithmetic units increase the throughput/area. The growth in area tends to be limited to factors of 2 to 4 (even in serial architectures, bits need to "wait on chip" to be processed).

Separate, optimized functional units lead to independently optimized, hence high-speed, primitives and independent parallelism, but the cost is increased in terms of power and chip area. Jointly optimized functional units tend to obtain their speed with more tightly coupled primitives and parallelism, allocating resources such as power and area to critical paths and relaxing the use of resources in less critical areas, typically equalizing speeds of different paths (a desirable property for wave pipelining, for instance).

Since in joint optimizations multiple, often conflicting, objectives are involved, comparisons can be based on "noninferior" solutions: a solution is noninferior if there exists no feasible design alternative that improves at least one design objective without degrading one other. A noninferior solution is a set of solutions that exclude all solutions that are uniformly worse ("the best of all possible bad solutions").

A concrete example: We are studying an AP (Arithmetic Processor) design based on independently optimized primitives (F-Add, F-Mult, F-Div, F-Sqrt) versus a CORDIC-function-based AP design. Let numbers be n-bit quantities. The independently optimized AP will have an area that is proportional to n^2 log ***n*** and a latency proportional to log ***n.*** A classical CORDIC-based design uses CP-Add, Shift and ROM as primitives, hence area and latency are proportional to n^2.

Pipelining the CORDIC doesn't reduce latency but can improve the pipeline rate by a factor of ***n,*** using faster adders increases area to n^3 log ***n*** while further improving the pipeline rate to match the latency of the independently optimized AP. In terms of speed, the pipelined classical CORDIC is in general an inferior solution. We have in the past proposed a number of ways to improve the speed of the CORDIC-type algorithms [6]. The basic problem is that the latency of the classical CORDIC consists of ***n*** Adds. Doubling the base halves the latency; however, the area also tends to be doubled. Hence, a balanced attack to reduce the number of iterations/stages while increasing the complexity of each stage (increasing total area) should help the speedup.

This leads to various hybrid schemes. One method is to combine ***n/2*** simple CORDIC steps with a final $n/2$ by ***n*** multiplication; the last step doubles the precision of the ***n/2*** CORDIC result. A new proposed method extends this idea backward to the first step by applying the first (binary) CORDIC step, then doubling the precision by doubling the base to 4 and applying a (quad) CORDIC step, continuing this CORDIC-type procedure by doubling the base every iteration. This doubling-type CORDIC requires log ***n*** steps, and if pipelined log ***n*** adders of increasing base and complexity the last one with base ***n/2*** dominating the chip area. This proposed

CORDIC algorithm leads to AP designs with a speed and area complexity that are similar to the baseline AP, hence the potential for at least a noninferior solution. Optimizing such a solution under given technology constraints has the potential for an AP that improves on the baseline design.

High Level Functions. The second approach is further explored in parallel from a higher-level function point of view, with an eye toward compiler support.

The notion here is to map applications onto high level functions and to implement or decompose these functions into lower level "natural" primitive functions, taking advantage of algebraic and other structural properties of the high level functions. Statistical information of these functions (at all levels) can be used as one of the guidelines to judge the "naturalness" of a primitive.

Clearly the set of lower level primitives needs to have at least the property of ***completeness,*** i.e., the property that any higher level function can be generated from these primitives. Sets of primitives can be evaluated by studying the complexity required to generate higher level functions as well as other competing sets of primitives.

The search for primitives even at the algebraic function level is most easily explained with a gate-level example involving X-type gates. The simplest X gate, the exchange or *Fredkin* gate, exchanges two bits depending on a third bit (the "control" bit); this conditional permutation is an invertible 3 by 3 transformation if the control bit is included. It is also a logically complete and conservative primitive function since the number of l's and O's (the Hamming weight) is preserved. There exists only one other such conservative, complete, and invertible 3 by 3 primitive; we termed it R-gate, because of its ***rotational symmetry*** (see group property below). We have shown that it takes two (primitive) X-gates to simulate a (primitive) R-gate, whereas it takes four R-gates to simulate an X-gate; in addition, an X (resp. R)-gate requires two (resp. three) 2-way MUX'S to realize them in hardware. One could conclude from this that X-gates are "more natural" primitives, both from a functional and from a hardware point of view.

The evaluation of sets of primitives involves transformation between such sets of primitives, in particular information-preserving transformations between these sets, or alternatively transformations between programs or higher level functions that are using these primitives.

Primitive IPT's are themselves good candidates for "natural primitives," e.g., the X/R-gates or the CORDIC functions discussed below. At the instruction set level, the X-gate corresponds to a conditional permutation or swap.

IPT's are at least good candidates for higher level primitives, enabling transformation/cross-compilers between different instruction sets or hardware primitives, since any function can be imbedded in an information-preserving function.

In the past year we have obtained statistical evidence that the use of nonminimal (e.g., boolean functions with more than three inputs and outputs) IPT primitives has certain advantages: In the automatic synthesis of functions in terms of IPT's, the use of nonminimal IPT primitives appears to regularize the search spaces in the decomposition process. The simplest synthesis algorithm is is based on randomly choosing IPT's and applying them as transformations to inputs and outputs (I/O spaces) of a given function, and choosing the IPT's that reduce the complexity of the given function. This reduction process is similar to the singular value computations of a matrix (a linear function/map). The complexity measures (or complexity indicators) we have considered for boolean functions range from byte counts of the algebraic expressions that represent the given function, to the number of min-terms, or Hamming-distance-based functions.

A number of observations were made in the use of nonminimal primitives during search-based synthesis procedures, and many specific and generic insights were garnered: Arithmetic functions have "natural" primitives such as xor-gates and X-gates; 3/2 counters are related to "interesting" 5 by 5 primitives; X- and R-gates are equal to the simplest Lattice Gas rules [7] and associated with the triangle group. An example of a very general insight is that sets of component functions with equal complexity form subsets that can be reduced independently and in parallel. This fact, among others, was used to decompose a polynomial root finder into a parallel program.

2 Basic Technology: Circuits and Wave Pipelining

2.1 Wave Pipelining (D. Wong)

Wave pipelining is a design method that can boost the pipeline rate of a system without using additional registers. In ordinary pipelined systems, there is one "wave" of data between register stages. When a new set of values is clocked into one set of registers, the values are allowed to propagate to the next set of registers before the first set is clocked again.

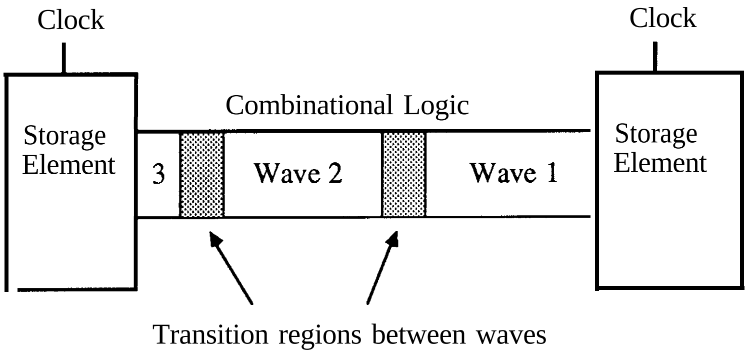


Figure 2: Wave pipelining.

In contrast, wave pipelining (Figure 2) uses multiple coherent "waves" of data between storage elements. This is achieved by clocking the system faster than the propagation delay between registers. In this method, the data values at the first set of registers are changed before the old data values have propagated to the next set of registers. The capacitance in the combinational logic circuit is being used to store values for pipelining.

Ideally, if all paths from input to output have equal delay, then the circuit's clock frequency is limited by rise/fall times, clock skew, and setup and hold times of the storage elements. In practice, due to the above limits and variations in fabrication, clock frequency can be increased by a factor of 2 to 3 using the best available design methods.

An analysis of circuit technologies shows that CML and super-buffered ECL are well suited for designing circuits with uniform delay. Standard ECL and static CMOS are not as good.

We have developed CAD algorithms to automatically equalize delays in combinational logic circuits to achieve wave pipelining. The algorithms adjust gate speeds and insert a minimal number of active delay elements to balance input-to-output path lengths in a circuit. For both normal and wave-pipelined circuits, the algorithms also optimally minimize power under delay constraints. These algorithms have been implemented in a set of CAD tools for wave pipelining in ECL/CML circuits.

Since February 1990, we have completed the design and fabrication of a test chip to demonstrate the concepts of wave pipelining. One of our

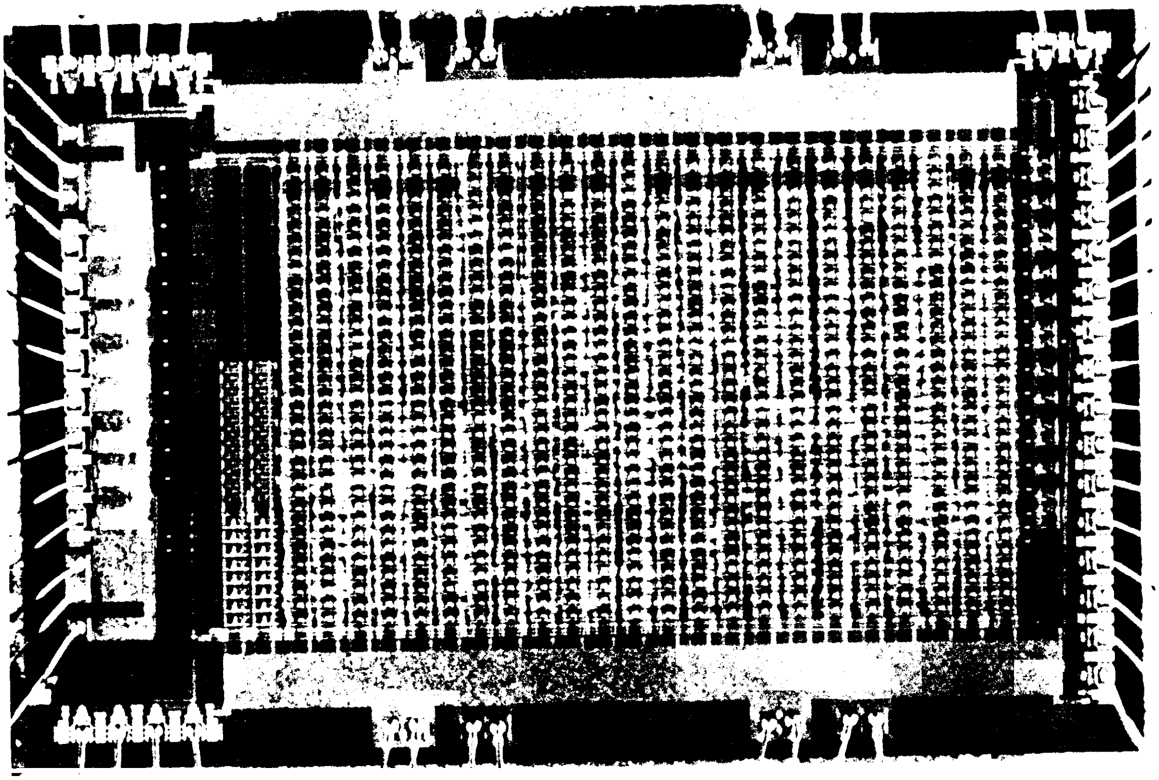


Figure 3: Photomicrograph of wave pipelining demonstration chip.

corporate sponsors, Signetics, has just finished manufacturing the chips in their commercial BiCMOS process called Qubic 1. A photomicrograph of a fabricated wave pipelining chip is shown in Figure 3.

Testing of the wave pipelining demonstration chips is being performed using a Trillium Delta-Master tester. Currently, we have preliminary test results which show that the chips actually function with the expected propagation delay and can support wave pipelining.

We have packaged 72 chips from two wafers from the same fabrication process run. Of these 26 have passed a low-speed functional test composed of 20000 pseudo-random vectors applied at 40 Mhz. The others had various functional defects. The longest path propagation delay for the functional chips averages 9.5-10.00 ns.

In a second test, the 20000 test vectors have been applied at 160 Mhz for a clock period of 6.25 ns. This is a factor of 1.64~ faster than the estimated normal clock period of 10.25 ns (97.5 Mhz). At 160 Mhz, all 25 functional chips have passed![1] This test verifies that wave pipelining works consistently for any input pattern and for all chips that pass low-speed functional tests.

In a third test, 40000 test vectors have been applied at various wave pipelining frequencies up to 240 Mhz. All 25 functional parts pass the test up to a frequency between 188 and 224 Mhz (clock period = 5.3 to 4.5 ns). This is 1.9x to 2.3~ faster than the normal pipelining frequency.

In the process of designing the chip, the CAD software for wave pipelining which was developed previously has been refined. The software has now been made to handle all the complexities associated with an actual design.

A theory has been developed which compares the performance of normal and wave pipelining in both transparent latch and edge-triggered register designs. One, two, and multi-phase clocking are considered. A complete description of this work may be found in Wong's dissertation.

In the next year, we hope to investigate the possibility of applying wave pipelining to some of the other chip designs from the SNAP project.

2.1.1 Other Research in Wave Pipelining

Since the start of our research in 1988, at least three other research groups following our work have begun independent efforts in the field of wave pipelining. Two principal efforts in CMOS have been undertaken by a group at North Carolina State under Prof. Wen-tai Liu, and by Fabian Klass and Prof. Hans Mulder at the Delft University of Technology in the Netherlands. Prof. Liu's group has shown that wave pipelining works in some special CMOS structures that act as FIFO's. These chips have been fabricated and fully tested. They plan to do some further work in general techniques for CMOS wave pipelining. Klass and Mulder have further developed the theory of CMOS wave pipelining, especially ways of minimizing the effects of the inherent delay variation in CMOS gates. At this time, they are designing a wave-pipelined 32-bit adder. At Amherst, MA, a third group is researching CAD algorithms for designing CMOS wave-pipelined circuits.

[1] Because the 26th functional chip was being photographed, it was not available for that test.

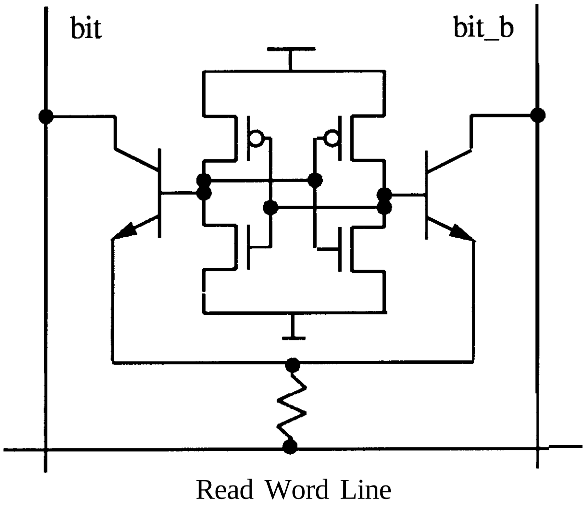


Figure 4: Bipolar access CMOS storage cell.

2.2 Register File Design (C. Chao, B. Wooley)

2.2.1 Achieving shorter access time

The access path to a register file can be broken into address buffering, decoding, word-line driving, bit-line sensing, and output buffering. Due to the large capacitance of the word and bit lines, driving the word lines and sensing the bit lines are the most time consuming operations. Faster access can be achieved by reducing the signal swings both on the word lines and the bit lines.

For MOS register (memory) cells with NMOS access transistors, the word line must swing from rail to rail in order to select a particular word. In addition, level conversion must be provided if ECL, or ECL-like, input/output interfaces are used in order to minimize interchip I/O delays. In a BiCMOS technology, bipolar transistors can be used to provide access to a CMOS storage cell in a fashion similar to that used in static bipolar memories and registers. This approach is illustrated in Figure 4, wherein data can be accessed with a word line swing of only a few hundred millivolts. Such an approach not only minimizes the word-line delay but also eliminates the need for ECL-to-CMOS level conversion in the access path.

In conventional CMOS memories, bit-line sensing is accomplished differentially by precharging the bit line pairs and then discharging one line

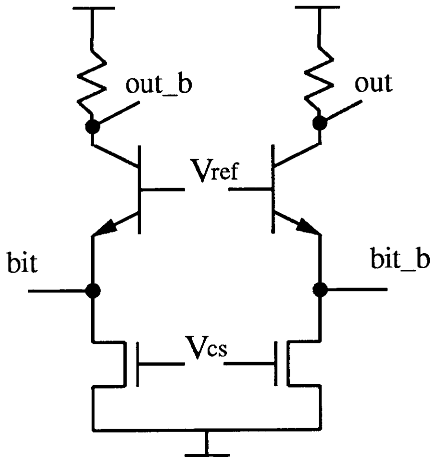


Figure 5: First stage sense amplifier.

in each pair through the memory cell. The cell current is necessarily low because the transistors must be kept small to conserve area; furthermore, the current must be pulled through the cell access transistor. Because of the low discharge current and the large bit-line capacitance, the change in the bit-line voltage, and thus the sensing of the cell, is relatively slow. In this research we have therefore adopted a current sensing approach similar to that used in bipolar memories whereby the bit-line discharge current is governed by bipolar access transistors rather than the MOS devices that form the basic cross-coupled storage element of the cell.

The first stage of the proposed sense amplifier is shown in Figure 5. In this circuit the two MOS current sources pull current through the common-base bipolar transistors, and both bit lines are clamped to a voltage one Vbe below Vref. When a cell is selected, one of the bit lines pulls additional current out of the bipolar transistor. The swing on the bit lines can be quite small, on the order of kT/q. To further reduce the bit-line sensing delay, a latch can be used as the second stage of the sense amplifier. Due to the positive feedback of the latch, its output voltages regenerate very quickly to full digit al levels.

2.2.2 Multiporting and Pipelining

Multiporting and pipelining are two effective ways of increasing the bandwidth of the register file. Conventional register files use three-port memory

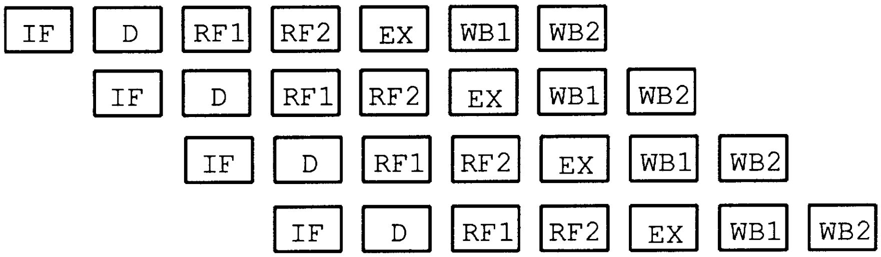


Figure 6: Instruction streams.

cells to provide two read accesses and one write access for the functional units. Since it is not pipelined, the cycle time is at least equal to the access time. In a pipelined, three-port register file, the cycle time can be made less than the access time. For example, Figure 6 shows the typical instruction streams for a processor with a two-stage pipelined register file. The cycle time is half of the access time.

Increasing the number of ports can also be effective in reducing the equivalent bandwidth of the register file. However, even a three-port design with the wide words being contemplated for this project requires that alternatives to conventional packaging technologies be devised. In particular, it will be essential to provide a large number of chip I/O's (at least several hundred), while maintaining I/O delays that are ultimately below one nanosecond. To accomplish this we are continuing to examine the use of both active and passive silicon substrates, together with novel chip attachment technologies.

3 Packaging

3.1 Microcontacts and Noise (J. Beale, F. Pease)

System performance is limited by our ability to distribute signals and power rapidly and reliably among the IC's comprising the system. Increasing the density of the chip-to-chip interconnect system is clearly highly desirable since, for a given system complexity, the length of each interconnect will be proportionately induced planarization, micro-spring contacts between chip and substrate and the use of silicon substrates to allow good heat sinking, high resolution fabrication of interconnect structures and the possibility of incorporating active circuitry into the substrate and so transfer the communication function from the chip to the substrate. One serious and poorly understood factor is the noise (by which we also mean internally generated interference) that can arise from a number of sources. We are initially investigating the noise that arises in microscopic pressure contacts.

Pressure contacts have two applications in the engineering of high-speed systems. The first is to allow the replacement of chips in multichip modules both during system assembly and checkout and during routine maintenance and troubleshooting. The second is for high-speed testing of chips prior to dicing and packaging. At present contact pads consume a significant fraction of the total area of high-speed die. Miniaturization of the contacts is obviously desirable along with the continued miniaturization of the rest of the circuitry. However, small pressure contacts can lead to increased noise and resistance. Our research is aimed at studying the electrical and mechanical behavior of such contacts down to the atomic level using such new tools as the scanning tunneling microscope (STM) and the atomic force microscope (AFM).

To study noise in microscopic contacts we originally set up a STM along with a spectrum analyzer to characterize noise spectra of tunneling currents under a variety of conditions and materials. In this way the contact can be modeled as an array of STM's operating under a variety of conditions; the variable parameters include tip-to-target distance, material, applied voltage, current and separation.

Those original experiments indicated in general the 1/f' (inverse frequency) spectral character of the noise but the STM proved unsuitable for this work because the current per se is used to control the tip-to-target distance.

To overcome this problem a new system has been designed and built which is, essentially, a combined STM and AFM. A schematic is shown in Figure 7. The vertical position of the tip is monitored optically with a resolution of 10 pm by means of an optical lever arrangement involving a laser beam, a miniature mirror carefully mounted on the microcantilever arm of the AFM and a split-field photodetector. The sample can be moved vertically independently by means of the piezoelectric transducer. In this way the tip-to-target current can be monitored independently of the tip-to-target distance. Moreover force and degree of indentation can be also measured and corresponding electrical properties (I, V, noise) monitored.

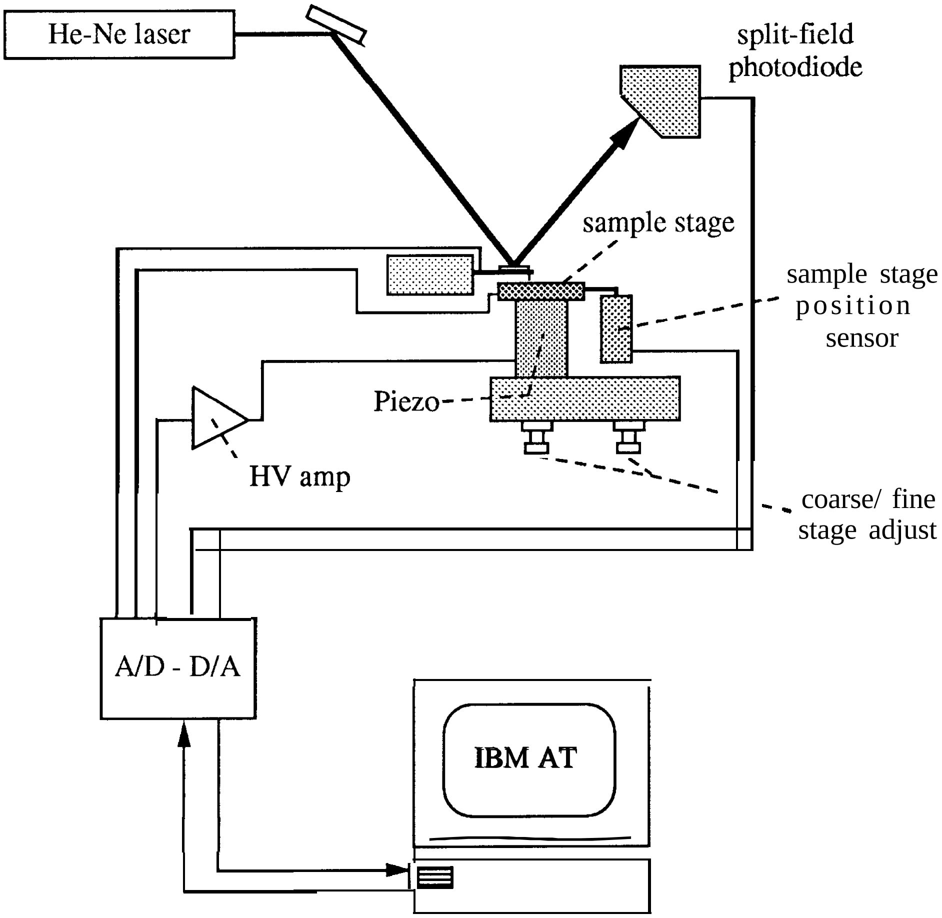


Figure 7: Combined AFM/STM contact force microscope.

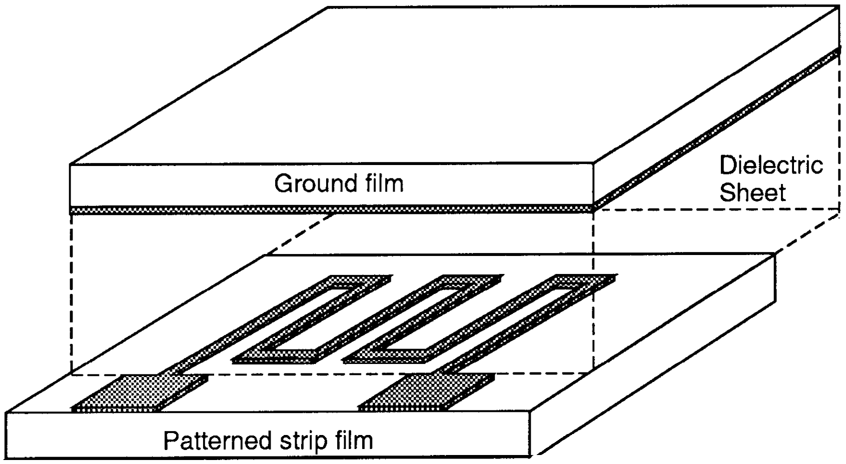


Figure 8: Schematic view of test structure. Two superconducting thin films are clamped together with a dielectric sheet between them to form a microstrip transmission line.

3.1.1 Superconducting Chip-to-Chip Interconnects

This project originated with support from the Semiconductor Research Corporation (SRC) but was incorporated into the SNAP program in late 1990.

Increasing the density of off-chip interconnects is one of the prime aims of advanced packaging research. The resistivity of the conductors is the fundamental impediment to achieving this density because the resistance per unit length of a transmission line increases as dimensions are scaled down, whereas inductance and capacitance per unit length are both unchanged on scaling. Thus, the ratio of line resistance to characteristic impedance increases as dimensions are scaled down, leading to worse dispersion and loss. With room temperature conductors the minimum signal-line-pitch (SLP) for 10 cm long terminated lines is about 40 microns.

However by employing superconductors the minimum useful SLP should be reduced tenfold [3] and maybe even more if the the frequencies of interest are below 100 GHz (almost certainly the case for SNAP circuits). Our experimental results on transmission lines made with YBCO films (Figure 8) indicate that the above predictions originally made for metallic superconductors below 10K should hold good for high temperature superconductors at 77K [4]. At this higher temperature superconducting lines and semiconductor devices clearly become compatible.

The research planned for this program includes a study of how such lines might impact the arithmetic performance of the systems planned and a study of the contacts between the normally conducting interconnects on top of the c-axis-oriented superconducting films and the underlying film which is usually only superconducting normal to the c-axis.

3.2 Photoconducting Interconnects (A. Hai, R. Dutton)

Optical devices fabricated with integrated circuits represent an alternative approach to high-speed data transmission (busing) and clocking. For example, such devices could be used to multiplex signals either for testing or to reduce pin count for certain I/O operations. Moreover, availability of such a technology could offer interesting alternatives for clock generation, especially on active substrate implementations. The design, fabrication, and testing of photoconductors, fabricated with standard ICs, is investigated here as a means to improve integrability with standard and mainstream silicon technologies. Initial results of the characterization of individual devices are also summarized.

3.3 Processing and optical characteristics

Part of this research focuses on determining how sensitive photoconductive devices are to processing conditions. Since these devices will be fabricated in an existing silicon device run, the designer may not be able to optimize the processing conditions. Knowing how the device characteristic degrades with processing parameters is therefore essential.

The basic structure of a photoconductive device is shown in Figure 9. It consists of an undoped polysilicon gap, contacted by two metal lines. The contacts between metal and poly can be doped for more linear characteristics. An interdigitated structure is also shown. The main design variables are the poly thickness, contact doping levels, different metal contacts, poly anneals, and different geometries.

The I-V curves of a typical photoconductive device in Stanford's BiCMOS process are shown in Figure 10. Different curves correspond to different values of total laser power incident on the chip. The device resistance changes by a factor between 10 to 30, typically. Such a modulation factor is more than sufficient to obtain complete on/off switching of detector and gating device.

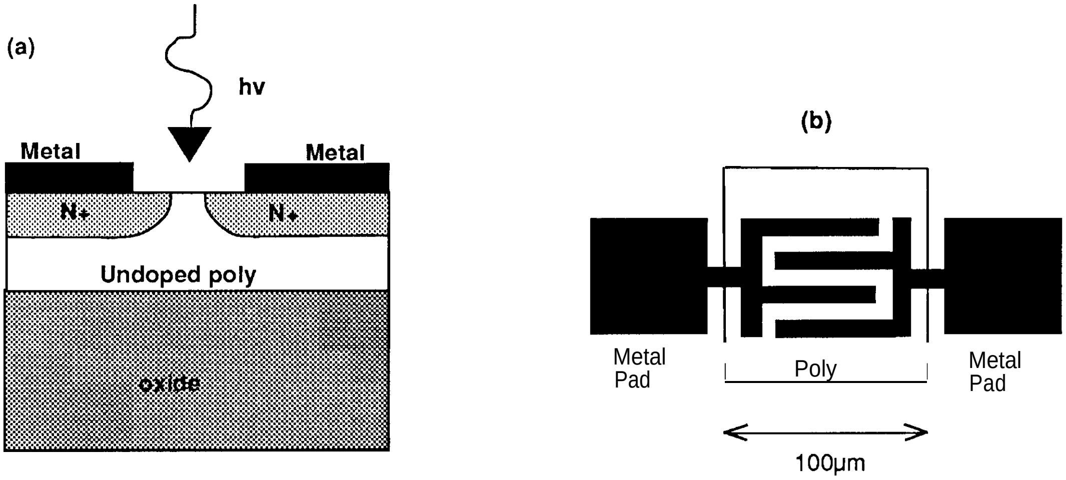


Figure 9: Photoconductors. (a) Cross-section. (b) Layout geometry.

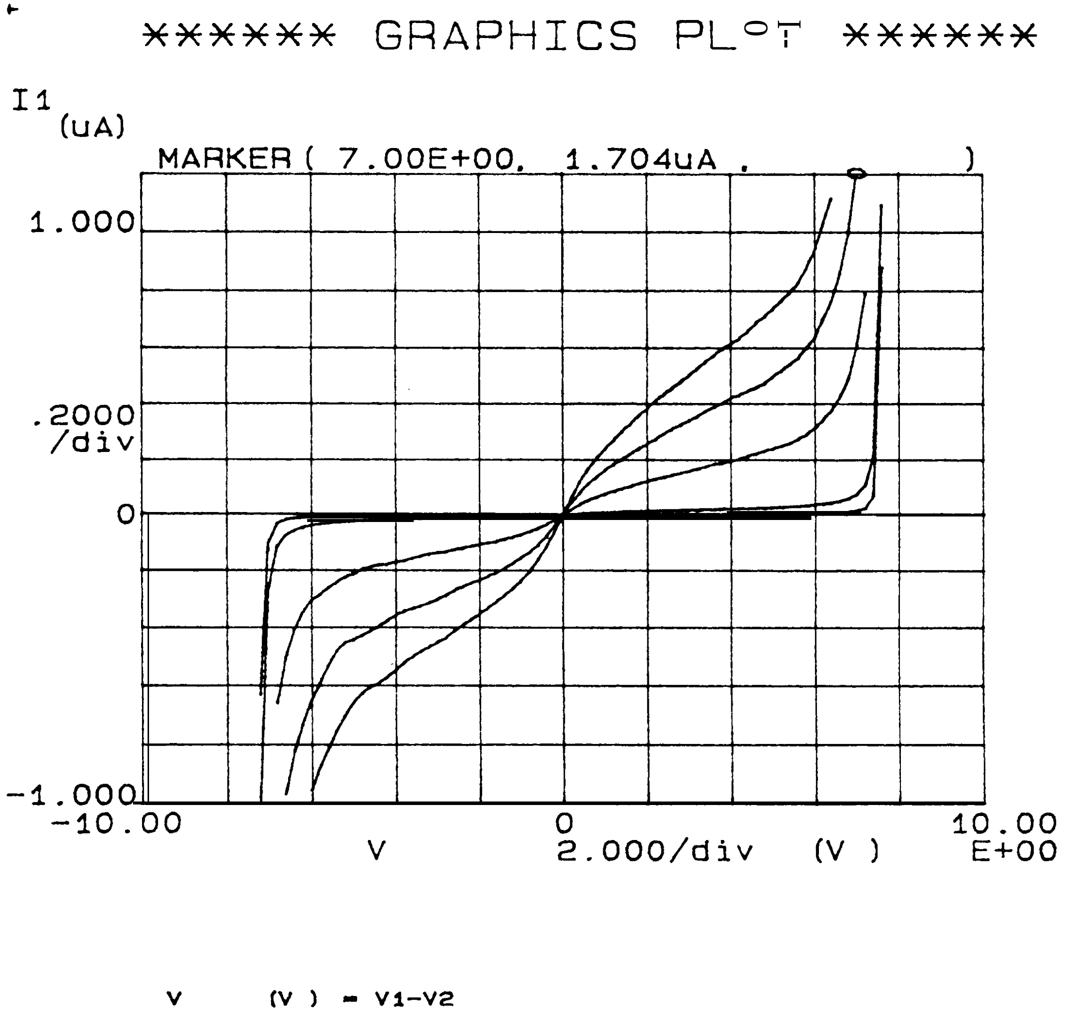
3.4 Electra-Optical Signal Detection

As a test vehicle Figure 11 shows a 32-bit electrical data bus being optically tested. As the laser is moved across the photoconductor on each line, a single output pin is driven high or low, showing the digital state of each line. With appropriate equipment, this method could be used for high-speed testing of buses on chips, with only one additional external pin. Alternatively, the optical detector is critical to the implementation of an optical data bus.

Circuits to test this application have been designed and should be ready for testing after the next BiCMOS run at Stanford.

The purpose of this project is to demonstrate the use of optical devices, fabricated with regular BiCMOS circuits. Ultimately, this area of research should contribute to a feasible design for optical interconnects.

L=4μ



```
Variable1:
 V1     -Ch1
 Linear sweep
 Start        -10.000V
 Stop          10.000V
 Step           .2000V

Constants:
 V2     -Ch2     0000V
```

ω 15-16:15

Power 45-5 mW
10 mW steps

File M57

Figure 10: Photoconductor I-V CURVES with varying laser power.

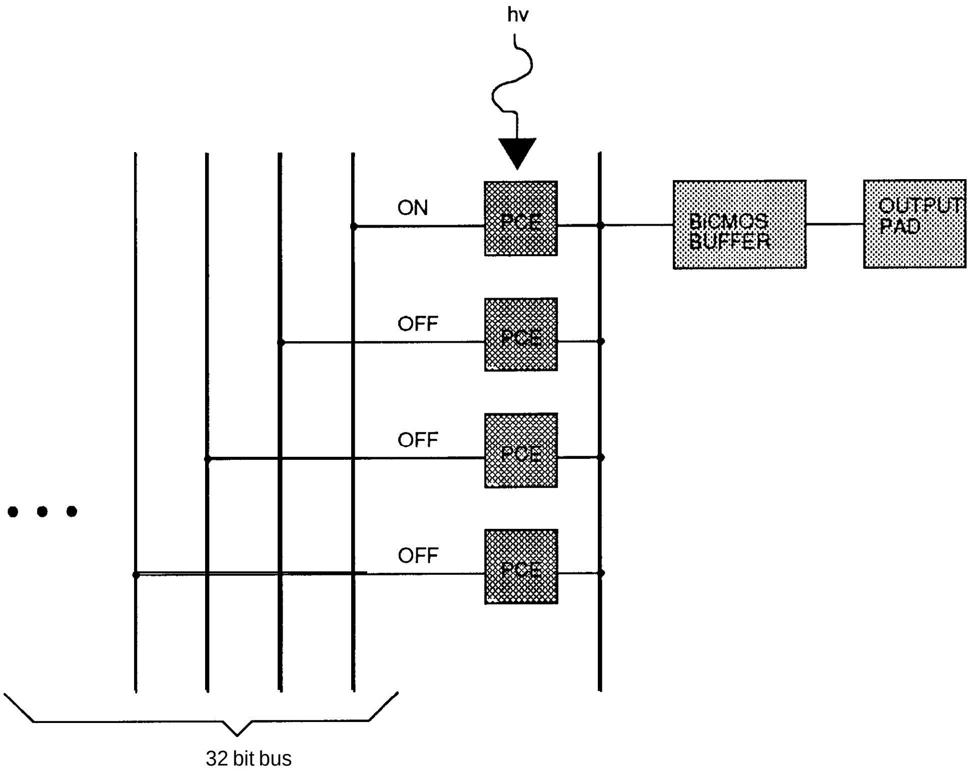


Figure 11: Testing a bus using photoconductive switch.

4 Industrial Cooperation

Our wave pipelining chip was built by Signetics with their kind support and cooperation. Hewlett-Packard Research has given us access to their fabrication facility for our floating-point adder functional unit. This is a state-of-the-art CMOS process. We have also tentatively been awarded an opportunity to fabricate a BiCMOS functional unit multiplier at SMOS in San Jose. This is another state-of-the-art CMOS technology with bipolar technology constructs.

References

[1] R. Stefanelli, "A Suggestion for a High-Speed Parallel Binary Divider," ***IEEE Trans. Comput.,*** Vol. C-21, No. 1, pp. 42-55, January 1972.

[2] E. M. Schwarz and M. J. Flynn, "Cost-Efficient High-Radix Division," ***Journal of VLSI Signal Processing,*** Special Issue on Computer Arithmetic, to be published August 1991.

[3] Kwon, 0. K., Langley, B. W., Beasley, M. P., and Pease, R. F. W., Superconductors as High Speed Chip-to-chip Interconnects, IEEE Elec. Dev. Lett. EDL 8, 582, 1987.

[4] Langley, B. W., and Pease, R. F. W., Paper presented at US/Japan Symposium on VLSI Technology, Honolulu, HI, June 1990.

[5] N. T. Quach and M. J. Flynn. High-speed addition in CMOS. Technical Report CSL-TR-90-415, Stanford University, February 1990.

[G] H. M. Ahmed, J.-M. Delosme, and M. Morf. "Highly Concurrent Computing Structures for Matrix Arithmetic and Signal Processing," IEEE ***Computer,*** Jan. 1982, pp. 65-82.

[7] F. F. Lee, M. J. Flynn, and M. Morf. "A VLSI Architecture for the FCHC Isometric Lattice Gas Model," Stanford University Technical Report CSL-TR-90-426, April 1990.

SNAP Publications (Year 2)

[DS91] G. De Micheli and P. Song. Circuit and architecture trade-offs for high-speed multiplication. ***IEEE Journal** of **Solid State Circuits,*** September 1991. (To be published).

[LP90] B. W. Langley and R. F. W. Pease. Superconducting interconnects for VLSI multi-chip system integration. In ***Proceedings, Symposium on VLSI Technology,*** Honolulu, HI, June 1990.

[MQF91] J. M. Mulder, Nhon T. Quach, and Michael J. Flynn. An area model for on-chip memories and its application. ***Journal*** of ***Solid State Circuits,*** 26(2), February 1991. Also published as CSL-TR-90-413.

[QF90] N. T. Quach and M. J. Flynn. An improved algorithm for high-speed floating-point addition. Technical Report CSL-TR-90-442, Stanford University, August 1990.

[QF91a] N. T. Quach and M. J. Flynn. Leading one prediction—implementation, generalization, and application. Technical Report CSL-TR-91-463, Stanford University, March 1991.

[QF91b] N. T. Quach and M. J. Flynn. The SNAP floating-point adder. Technical Report (in preparation), Stanford University, May 1991.

[QTF91] N. T. Quach, N. Takagi, and M. J. Flynn. On fast IEEE rounding. Technical Report CSL-TR-91-459, Stanford University, January 1991.

[SF91] E. M. Schwarz and M. J. Flynn. Cost-efficient high-radix division. ***Journal** of **VLSI Signal Processing,*** August 1991. Special Issue on Computer Arithmetic.

[Tak91] Naofumi Takagi. A radix-4 modular multiplication hardware algorithm. Technical Report CSL-TR-91-458, Stanford University, January 1991.

[WF91] D. Wong and M. Flynn. Fast division using accurate quotient approximations to reduce the number of iterations. In ***10th Symposium on Computer Arithmetic,*** Grenoble, France, June 1991. IEEE.

SNAP Ph.D. Thesis

Derek C. Wong, "Designing High-Performance Digital Circuits Using Wave Pipelining and Power Optimization," Electrical Engineering Department, Stanford University, June 1991.