DECEMBER 1993

WRL
Research Report 93/8

A 300MHz 115W 32b Bipolar ECL Microprocessor

Norman P. Jouppi, Patrick Boyle, Jeremy Dion, Mary Jo Doherty, Alan Eustace, Ramsey Haddad, Robert Mayo, Suresh Menon, Louis Monier, Don Stark, Silvio Turrini, Leon Yang, John Fitch, William Hamburgen, Russell Kao, and Richard Swan

digital **Western Research Laboratory** 250 University Avenue Palo Alto, California 94301 USA

The Western Research Laboratory (WRL) is a computer systems research group that was founded by Digital Equipment Corporation in 1982. Our focus is computer science research relevant to the design and application of high performance scientific computers. We test our ideas by designing, building, and using real systems. The systems we build are research prototypes; they are not intended to become products.

There two other research laboratories located in Palo Alto, the Network Systems Laboratory (NSL) and the Systems Research Center (SRC). Other Digital research groups are located in Paris (PRL) and in Cambridge, Massachusetts (CRL).

Our research is directed towards mainstream high-performance computer systems. Our prototypes are intended to foreshadow the future computing environments used by many Digital customers. The long-term goal of WRL is to aid and accelerate the development of high-performance uni- and multi-processors. The research projects within WRL will address various aspects of high-performance computing.

We believe that significant advances in computer systems do not come from any single technological advance. Technologies, both hardware and software, do not all advance at the same pace. System design is the art of composing systems which use each level of technology in an appropriate balance. A major advance in overall system performance will require reexamination of all aspects of the system.

We do work in the design, fabrication and packaging of hardware; language processing and scaling issues in system software design; and the exploration of new applications areas that are opening up with the advent of higher performance systems. Researchers at WRL cooperate closely and move freely among the various levels of system design. This allows us to explore a wide range of tradeoffs to meet system goals.

We publish the results of our work in a variety of journals, conferences, research reports, and technical notes. This document is a research report. Research reports are normally accounts of completed research and may include material from earlier technical notes. We use technical notes for rapid distribution of technical material; usually this represents research in progress.

Research reports and technical notes may be ordered from us. You may mail your order to:

Technical Report Distribution
DEC Western Research Laboratory, WRL-2
250 University Avenue
Palo Alto, California 94301 USA

Reports and notes may also be ordered by electronic mail. Use one of the following addresses:

Digital E-net:	`DECWRL::WRL-TECHREPORTS`
Internet:	`WRL-Techreports@decwrl.dec.com`
UUCP:	`decwrl!wrl-techreports`

To obtain more details on ordering by electronic mail, send a message to one of these addresses with the word ‘‘`help`’’ in the Subject line; you will receive detailed instructions.

A 300MHz 115W 32b Bipolar ECL Microprocessor

Norman P. Jouppi, Patrick Boyle, Jeremy Dion, Mary Jo Doherty, Alan Eustace, Ramsey Haddad, Robert Mayo, Suresh Menon, Louis Monier, Don Stark, Silvio Turrini, Leon Yang, John Fitch, William Hamburgen, Russell Kao, and Richard Swan

December, 1993

Abstract

A full-custom single-chip bipolar ECL RISC microprocessor was implemented in a 1.0μm single-poly bipolar technology. This research prototype contains a CPU and on-chip 2KB instruction and 2KB data caches. Worst-case power dissipation with a nominal -5.2V supply is 115W. The chip has been designed for a worst-case clock frequency of 275MHz at a nominal supply. The chip verifies a new style of CAD tools developed during the design process, advanced packaging techniques for high-power microprocessors, and VLSI ECL circuit techniques.

This Research Report is a reprint of a paper appearing in the November 1993 issue of the IEEE Journal of Solid-State Circuits.

digital **Western Research Laboratory** 250 University Avenue Palo Alto, California 94301 USA

Table of Contents

List of Figures

List of Tables

1. Introduction

Bipolar ECL technology has historically been used to implement high speed communication circuits and mainframe computers built with gate arrays and multichip modules. These gate arrays have had low integration compared with full-custom CMOS microprocessors. In addition, multichip modules typically have a power dissipation limit of under 30 Watts per chip. This further limits both the integration and circuit speed available. Moreover, a gate array design style pays a significant penalty in terms of the number of gates in series required to implement a particular function because of the limited gate selection (typically under 100, including power options) available in a gate array macro library, in contrast to the billions of gate circuit functions available in a custom ECL technology. The combination of all of these factors makes it difficult for ECL multichip gate array machines to compete with full-custom CMOS microprocessors.

A full-custom design approach applied to ECL can provide logic density similar to full-custom CMOS. Full-custom ECL also provides added circuit speed by tailoring logic swings for specific circuits and allowing a wider range of circuit topologies. A full-custom ECL CPU and its caches can be integrated on a single die, and yields significantly higher performance. Although this die dissipates considerable power, it is only a single die, not a collection of many medium-power die as in a multichip CPU. A single high-power die can be cooled with a thermosiphon.

This paper describes a full-custom single-chip ECL RISC microprocessor [5] which has been implemented in a 1.0μm single-poly bipolar technology. The chip contains a CPU and on-chip instruction and data caches. The 15.4 x 12.6 mm die contains 468K bipolar transistors and 206K resistors. Worst-case power dissipation with a nominal -5.2V supply is 115W. The chip has been designed for a worst-case clock frequency of 275MHz at a nominal supply. It has 202 ECL 100K inputs, 157 ECL 100K outputs, and 254 power pads, and is packaged in a 504 pin plastic pin grid array. A subset of the MIPS R6000 architecture is implemented. No floating-point, memory management, or integer multiplication and division support is provided on-chip. The chip is a research prototype designed to verify a new style of CAD tools, advanced packaging techniques for high-power microprocessors, and VLSI ECL circuit techniques.

The chip was designed largely with CAD tools developed by members of the design team. The schematics are graphical representations of C++ programs. The layout consists of 554 different cells, of which 93 are hand-drawn. The remainder are automatically synthesized leaf cells or composite cells placed by program and routed automatically. Over half of the point-to-point connections within a typical synthesized leaf cell are made with <5 ohm/square silicide. Switch-level simulation was performed on a flat netlist of all transistors, resistors, and capacitors extracted from the entire chip layout, including the caches. A bipolar transistor-level timing analyzer was used to tune the performance of the design. Resistance and current of power supply and reference distribution networks were extracted and voltage drops calculated [9]. Drops on these networks were all less than 15mV. Another tool developed for the project verified noise margins and saturation margins on all circuits on the chip.

A novel thermosiphon-based cooling technology was developed for the chip [4]. A thermosiphon is a heatpipe without a wick. It relies on a combination of phase change and mass transport to provide much lower thermal resistances than are possible with solid conduction heat sinks. In addition, CAD software was developed to model the temperature profile of the die before tapeout.

2. Chip Overview

Figure 1 shows a photo of the chip after deposition of the fourth layer of metal, with a floorplan overlay. The breakdown of transistors and area usage are listed in Table 1. The die area and transistor counts are dominated by the 2KB instruction and 2KB data caches. The instruction and data caches each use a 16B line, contain byte parity, and are direct-mapped.

Because the fundamental goals of this phase of our work were the demonstration of CAD, packaging, and circuit design techniques, the microarchitecture was intentionally kept simple both at the expense of cycle time and performance. For example, caches with larger than 16B line sizes would have better performance, but the tradeoff was made to keep the machine simple by making the cache line size equal to the width of the external interface data read bus. This requires only one off-chip fetch per on-chip cache miss.

The machine has a relatively shallow 5 stage pipeline (Figure 2), each one clock cycle long. In instruction fetch (IF), the PC is sent to the 2KB instruction cache and the instruction at that address is read, cache tags are compared, instruction parity is checked, and the instruction is decoded. If a cache miss occurs (possibly as a result of a parity error), then a pipeline stall signal is sent to the whole chip and the machine stalls in the IF pipestage while the cache line is fetched. This and a similar path involving the data cache in the MEM pipestage are the cycle-time limiting paths for the machine. This is an example of a tradeoff favoring microarchitectural simplicity over cycle time and preformance. More complicated approaches allow instructions to advance to the next pipestage before cache hit or miss is known and then back up the pipeline on a miss, or split the cache access into several pipeline stages as the MIPS R4000 [8] does.

The rest of the pipeline operates as follows. In the RD pipestage the source operands are read from the register file. 32b integer addition, subtraction, logical, and shift operations as well as addressing calculations take place in the ALU pipestage. In the MEM pipestage the 2KB on-chip write-through data cache may be read or written. In the case of a data cache read, the parity is also checked in the MEM pipestage. In the event of a parity error, the cache signals a cache miss and the line is read again from off-chip. Since the data cache is write-through it contains no information which is not also present external to the chip. This allows soft and some hard failures in the instruction and data cache to be bypassed. Since the caches are a large percentage of die area and transistor count, this improves reliability significantly. Instruction results are written to the register file and store instructions place their result in the write buffer for transfer to the external cache in the WB pipestage.

The external interface is dominated by large unidirectional busses. A 128b bus is used for incoming data. A 64b bus is used for outgoing data. All external data busses have byte parity. Separate address busses of 32b and 14b are used to index the external cache data and tags, respectively. Two unidirectional 11b busses are used to read and write external cache tags. Scan data is input on an 8b bus and output on another 8b bus.

An on-chip PLL is used to generate a 1X to 8X multiple of an off-chip differential clock for use on-chip. All communication between the chip and board is synchronous to the external board clock. All I/O pads have flip-flops clocked with the board clock, so no chip pins transition at the chip clock rate. A typical external clock frequency is 100MHz.

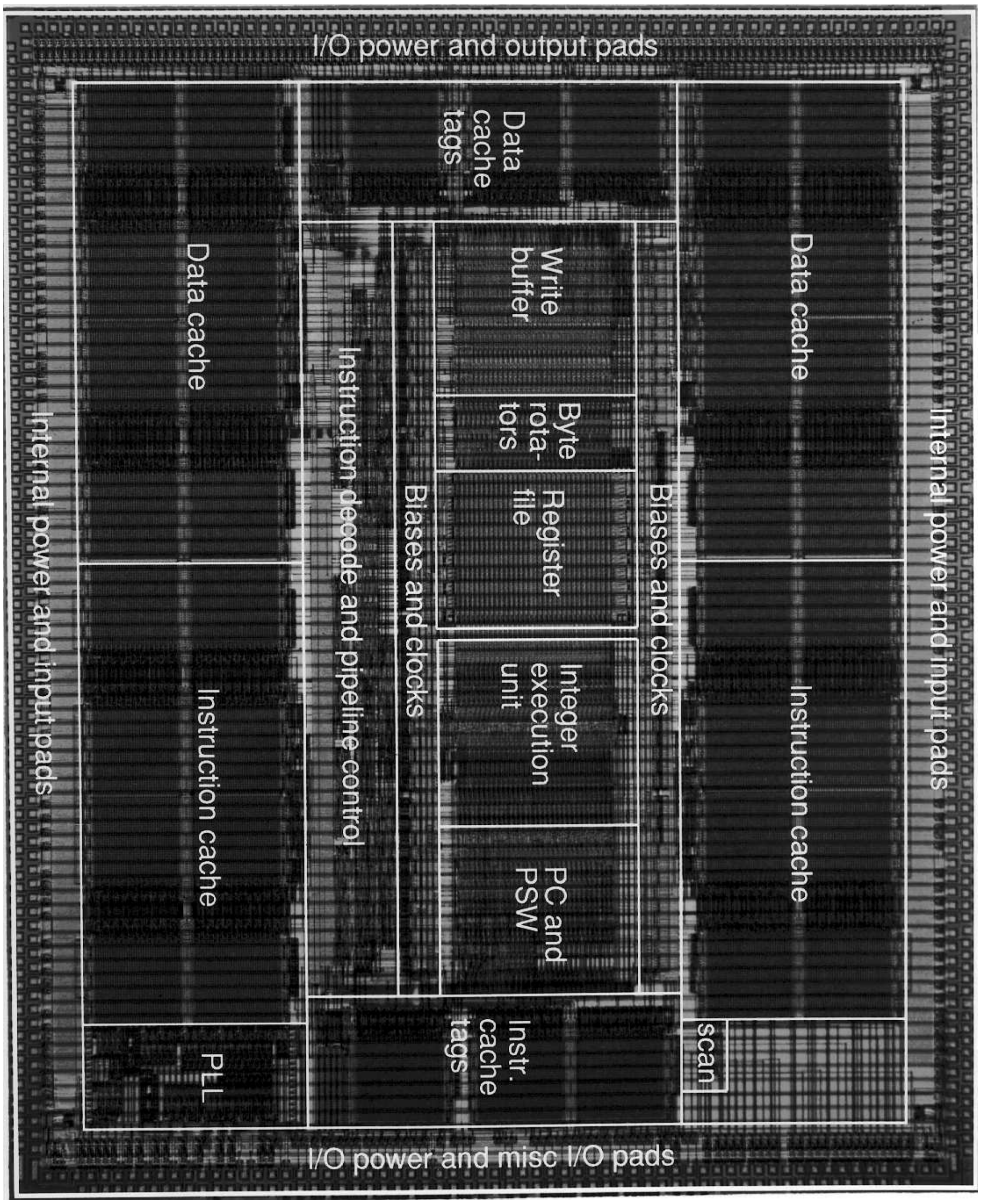

Figure 1: Die before gold metalization with floorplan

The chip is designed to work with a pipelined second-level board cache. One half of an external clock cycle is required to go from the microprocessor chip to flip-flops placed just before the second-level cache RAMs. One external pipestage is allocated for external cache RAM access, with data from the RAMs going directly to another set of flip-flops. Finally, one more half external clock cycle is used to get from the flip-flops at the output of the second-level cache RAMs back onto the CPU die. Since the external interface write path is only half a cache line wide, and the external cache is write-back, writes from the chip require one cycle to probe the external cache followed several cycles later by one cycle to write the data. Several write opera-

Unit	Bjts	Resistors	Power(W)	%Area	W/cm^2
Data cache	179,842	80,950	28.39	22.53	63.8
Instruction cache	179,766	80,790	27.71	21.88	64.1
Write buffer	12,369	4,157	3.79	1.96	97.9
Byte rotators	3,570	1,260	1.76	1.05	84.7
Register file	19,422	6,517	2.91	2.43	60.5
Integer execution	14,627	4,575	5.75	2.67	109.1
PC and PSW	10,769	3,728	3.24	2.05	80.0
Biases and clocks	13,101	5,271	8.28	4.36	96.1
Control logic	11,568	4,815	7.04	5.11	69.7
PLL	7,731	4,791	5.51	1.76	158.6
Scan control	471	189	0.11	0.09	64.6
Pads & interface	14,328	9,817	20.52	22.59	46.0
Routing or unused				11.52	0.0
Total	467,564	206,860	115.00	100.00	58.2

Table 1: Device counts, power, and area of each functional unit

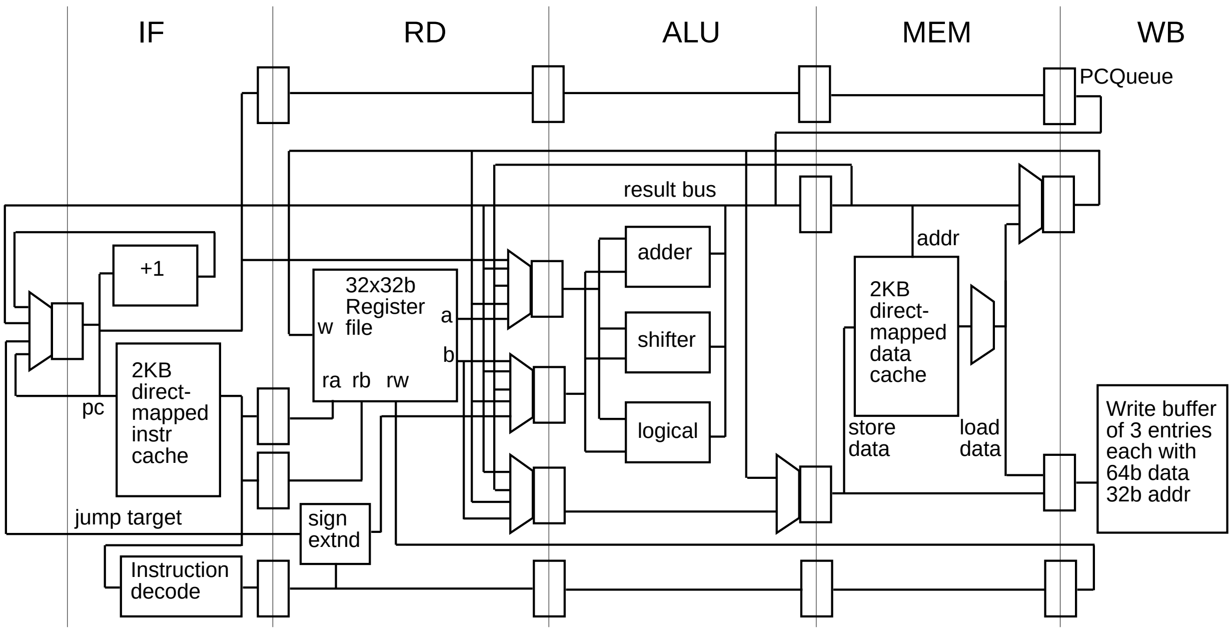

Figure 2: CPU pipeline and machine organization

tions can be in progress at the same time, since there are separate interfaces to the external tag RAMs and to the external data RAMs.

3. Bipolar Process Technology

The chip has been implemented in a 1.0µm single-poly bipolar technology. A summary of the bipolar transistor parameters for the most common transistor sizes used in the design is given in Table 2. The 1µm by 2µm emitter device is the minimum size device available. We used the minimum device in gates with switch currents of 100µA and 200µA, resulting in current densities of $50\mu A/\mu m^2$ and $100\mu A/\mu m^2$. The 100µA current-switch family was used only in non-speed critical sections. For larger speed-critical gates or gates driving larger loads we used a 1µm by 4µm emitter device running with a switch current of 600µA, resulting in a current density of $150\mu A/\mu m^2$.

	1um x 2um emitter	1um x 4um emitter
C_{js}	25ff	31ff
C_{jc}	8ff	13ff
C_{je}	6ff	10ff
R_e	52 ohms	44 ohms
R_c	116 ohms	64 ohms
R_b	462 ohms	278 ohms
t_F	14 ps	14 ps

Table 2: Transistor parameters

The metal widths, spacings, thickness, and resistivity used on the chip are shown in Table 3. The metals form an "inverted pyramid" structure, where the thickness and pitch increase while the resistivity decreases with distance from the substrate. This is a natural fit for VLSI chips where long distance communication and power distribution occur on the upper layers. For both of these the metal pitch is not a significant issue, but low resistance connections are important for keeping IR drops on power supplies low and for low-skew clock distribution. In particular, the thick metal 3 is crucial in obtaining our sub-50ps clock skew over the die. Lower layers are more useful for local interconnections, where the RC time constant of wires is not significant. Thus for local wires, fine pitch at the expense of resistance is also a good tradeoff. The router developed during this project was gridless, so it was able to handle different metal design rules on each layer without difficulty.

layer	width	spacing	thickness	resistivity
metal 1	1.8um	1.2um	0.7um	60 mohms/sq
metal 2	2.0um	1.7um	1.1um	29 mohms/sq
metal 3	3.0um	2.6um	2.1um	16 mohms/sq
metal 4	10.0um	5.0um	2.4um	13 mohms/sq

Table 3: Metal parameters

We also used a "metal 5" consisting of one mil thick gold TAB metalization. Gold bus bars covering the entire chip are used to distribute the 26A of supply current. A cross section of the gold metal is shown in Figure 3. The low resistance of the gold bus bars is the major factor in meeting a 17mV total IR drop budget on each supply. A die photo after deposition and patterning of the gold bus bars is shown in Figure 4.

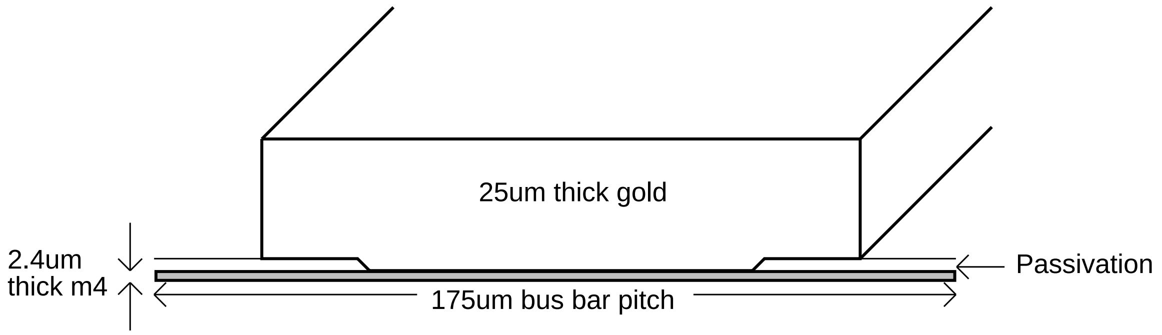

Figure 3: Cross section of gold bus bars

To get a better intuition of how the transistor parameters affect gate delays, let us briefly consider a gate delay modelbased on these parameters. P. K. Tien has developed a model of ECL inverter delay [10]:

$$delay = t_F + (6.7 \times R_b \times I_c \times t_F) + (2.6 \times R_b \times C_{bc}) + (2.6 \times C_{bc} \times V_{sw}/I_c) + (0.52 \times C_{cs} \times V_{sw}/I_c)$$

The first three terms of this equation cover the intrinsic switching time of a single transistor, while the last two terms cover the loading of the output by the transistor's collector-base and collector-substrate capacitances. The constant factor of 0.52 in the last term adjusts the collector-substrate junction capacitance which is given at zero-bias for the reverse bias found during operation. The constant factor of 2.6 in the second term adjusts the base-collector capacitance of the switching transistor and the Miller capacitance by 0.8 but leaves the emitter-follower base-collector capacitance at its zero-bias value.

For an inverter using the minimum size transistor, a 200µA switch current, and a single-ended swing (V_{sw}) of 575mV, this model predicts a delay of:

$$delay = 14 + 9 + 10 + 60 + 37 = 129ps$$

For an inverter using the 1um by 4um emitter device and a 600µA switch current, the model predicts a delay of:

$$delay = 14 + 16 + 9 + 32 + 15 = 87ps$$

The simulated delays of 105ps and 70ps for 200µA and 600µA inverters are within 25% of the delay values predicted by the model.

However, in ECL an inverter is usually a useless gate since gates can produce both true and complement outputs. If we consider more useful gates such as an 8 input multiplexor, then we can extend the model by adding a term of

$$fan\text{–}in_{delay} = 0.7 \times 0.8 \times C_{bc} \times V_{sw}/I_c + 0.7 \times 0.52 \times C_{cs} \times V_{sw}/I_c$$

for each additional fan-in. (This assumes the delay of interest is from a level one data input to the output, and not a select to the output.) The first term represents the additional loading on the load resistor of the gate from the collector-base capacitance of each additional transistor con-

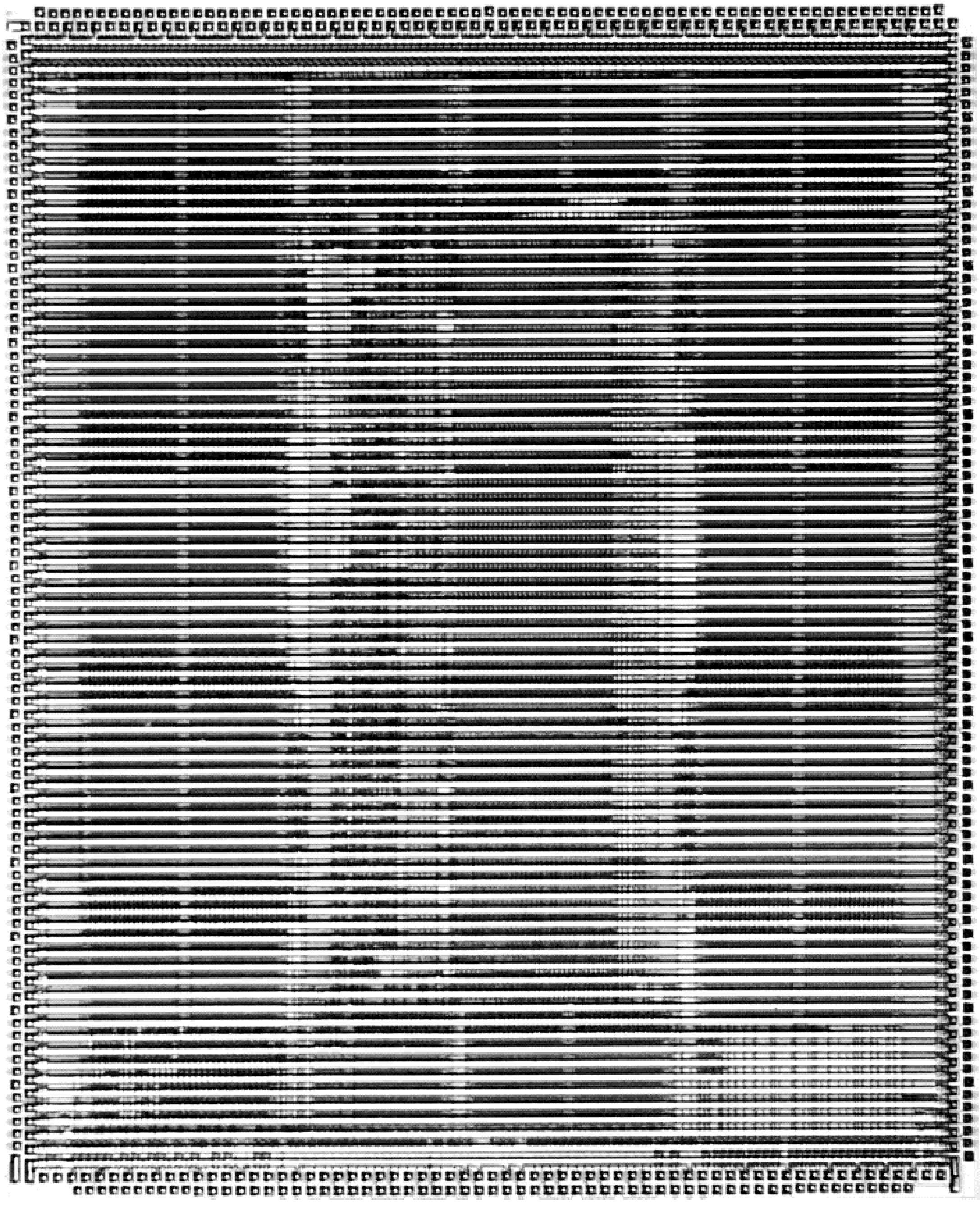

Figure 4: Die with gold bus bars

nected to the load resistor. The second term represents the additional load from the collector-substrate capacitance of each additional transistor. The 0.7 factor in each term approximates the reference to be midway between the high and low outputs ($0.5 = e^{-0.7}$), and that a successor gate can start to switch when the output of the previous gate crosses the reference. Based on this equation, we get a fan-in delay of 39ps per fan-in at a switch current of 200µA, for a total of 402ps for an 8-input multiplexor. For the 600µA gate family, this works out to a fan-in delay of 18ps per fan-in, for a total of 213ps for an 8-input multiplexor. This compares well to delays of 402ps and 232ps for the 200µA and 600µA gate families obtained from spice simulations. For each current switch family, we had families of emitter followers which used 1X, 2X, and 4X the switch current per output.

One important observation is that even for a simple inverter, gate delay terms involving the logic swing dominate the delay equation. This is even more pronounced for more complex and useful gates, such as the 8-input multiplexor. For example, according to this model 75% of the delay of a low power inverter is proportional to the logic swing. For a high power 8-input multiplexor, 82% of the gate delay is directly proportional to the logic swing. Thus we spent considerable effort during the design process to use as small a voltage swing as we could, and then to completely verify the noise margin in all the circuits of the chip.

4. Circuit Technology

A custom ECL design approach gave us the freedom to use the best features available in ECL to attain high performance.

An example of this is the heavy use of wired-ORs in the control logic and the datapath busses. A function implemented with wired-ORs has less delay in comparison to the same function implemented in an OR gate. Wired-ORs also require less wiring because only a single wire travels among the outputs being wire-ORed together, instead of many wires traveling to the inputs of an OR gate. Excessive IR drop between the shared current source and outputs being wired-ORed is a crucial issue. Typically the use of wired-ORs is not permitted in gate arrays since, depending on the placement of the cells, the IR drop could be excessive. By careful design of the wired-ORs, we were able to keep the IR drop on wired-OR wiring within a 10mV budget. The low resistance of metal 3 helped us achieve a low IR drop in many cases.

Another important feature in ECL is series-gating. Series-gating allows more functions to be performed in a gate with a single current switch. This can reduce the delay and area required to compute a function while saving power. The chip makes extensive use of three-level series gating, with the restriction that the bottom level is always differential. Thus the voltage at the collector of the current source is -4 V_{be} (-1 V_{be} for an emitter-follower and -3 V_{be} for the series gating). 478mV was dropped across the current source resistor. The V_{ce} of the current source was set to V_{be}, so that the current source did not saturate on switching transients. Hence a total of about $V_{swing} + 2*V_{be} + 0.5 = 2.7$ volts is required for emitter followers and their current sources, but only one V_{be} is required for each level of series-gating. Therefore, approaches which limit circuits to one level of series-gating to reduce the power supply and hence the power dissipation end up wasting a greater percentage of the power supply on the overhead of emitter followers and current sources than approaches using a higher supply (in our case -5.2V) with three levels of series gating.

In many gate array and standard cell designs, inputs to a cell are always assumed to be at level one. Hence, inputs which are used at the second or third level of series gating require level shifters in each cell. However, in many instances where an array of cells is used with a common input, level shifters can be factored out of individual cells. For example, if a 32b 4-input multiplexor is used in a datapath, the output of the gate generating the selector controls can be driven directly at the second level to all of the multiplexors. This saves power and the area required by level shifters in each cell.

Another benefit of custom ECL is that we were able to use different logic swings in different circuits depending on speed requirements and logic and layout constraints. For example, most logic portions of the chip use a single-ended swing of 575mV. However, a differential logic swing of 230mV was used in some logic circuits as well as the PLL. Differential logic was used in the PLL to avoid noise inherent in single-ended switching. The noise from single-ended ECL switching on the power supplies is already very small in comparison to CMOS circuits, but we felt the speed advantage of differential circuits coupled with their lower supply noise generation and common-mode noise rejection was important for low jitter in the PLL. Finally, the bitlines in the memories and specialized logic structures such as the barrel shifter use cascode circuits with swings of 30 to 70mV. Figure 5 is an example of a cascode circuit used in a shifter. The 100μA standby current sources effectively clamp nodes *c* and *c_* at one V_{be} below V_{cc}. Different data is selected by multiplexor arm selectors *Sel0* through *Sel31* driven at level two from a decoder, with one and only one mux selector high at a time. As the data is changed, the 200μA switch current source can be switched onto either node *c* or *c_*. This will result in a 3:1 variation in current through the cascode transistors. The variation of V_{be} with a change in current of *n*:1 is V_T*ln(n). For a 3:1 variation in current this results in about a 30mV swing on nodes *c* and *c_*. This means that *c* and *c_* can transition almost 20 times faster than if a full swing of 575mV were used. The output of this shifter is taken differentially. A 230mV swing from the 200μA switch current source appears on *out* and *out_*, along with a 115mV offset from V_{cc} due to the standby current sources. The cascode circuit effectively shields the capacitance of the high fan-in nodes *c* and *c_* from the outputs.

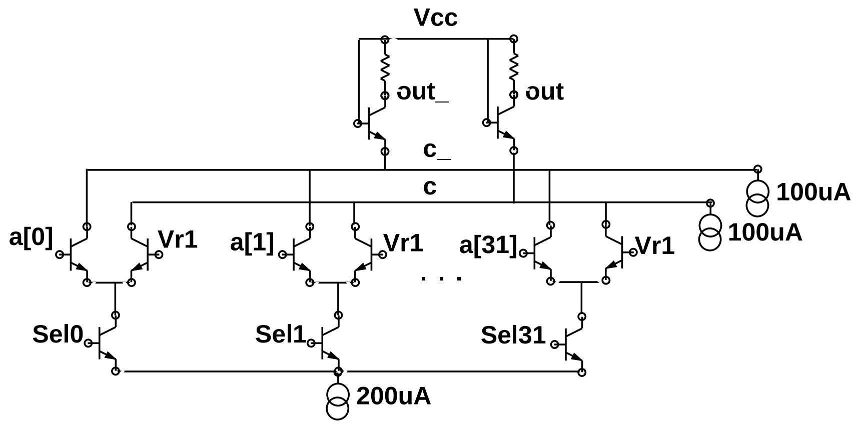

Figure 5: Cascode multiplexor circuit

4.1. Noise Margins

In addition to using different logic swings in different parts of the chip, we were careful to make each type of swing as small as possible. The high current drive available in an ECL emitter follower plus the short average wire lengths (due to the use of custom chip design techniques) meant most of the cycle time was spent in gate delay. Most of the gate delay time is proportional to the logic swing, so the chip speed is highly dependent on the logic swing. Figure 6 is a graphical representation of our single-ended noise budget. The reference is placed 267mV below the high output, and 300mV above a low output. 8mV is allowed for noise on the reference itself. The swing is composed of many terms. The adherence of the chip to each term in the noise budget was verified before tapeout by CAD tools that were developed for the project.

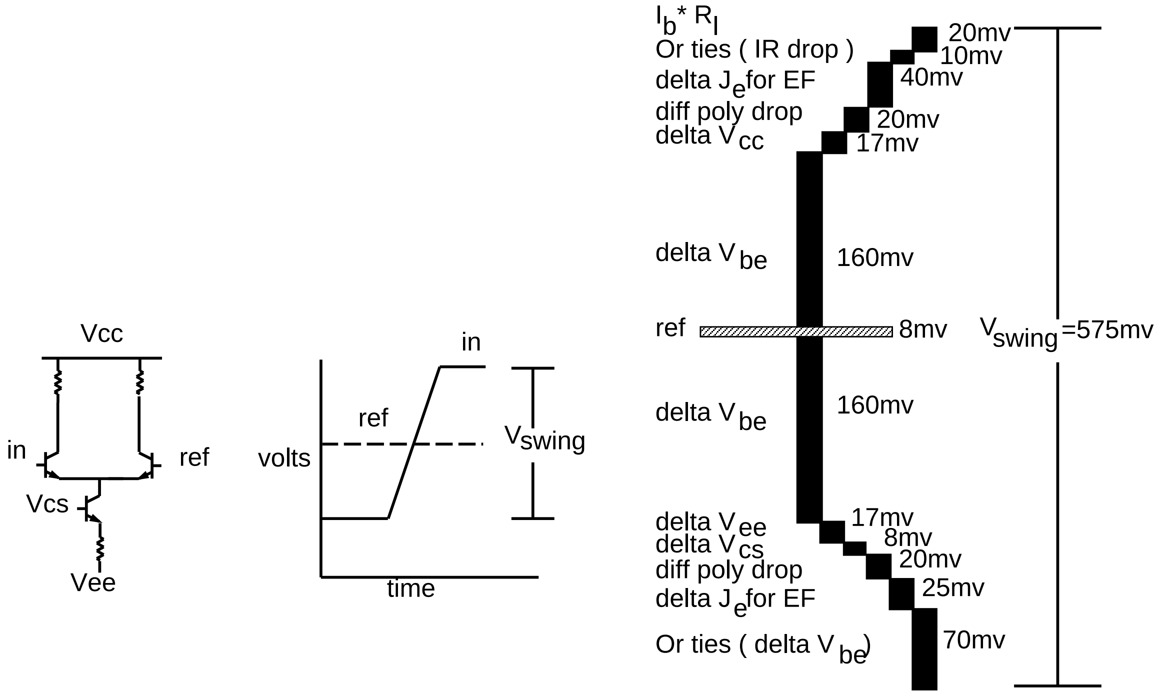

Figure 6: Breakdown of single-ended noise margin

The swing above the reference was determined by six terms. The I_b*R_l term is the drop on the swing resistor from the base current of an emitter follower. 10mV is allocated for IR drops on interconnect when using wired-ORs. Emitter followers in different circumstances are run at different current densities; this produces a variation of up to 40mV. Silicide is used extensively for intracell wiring. Since it has a resistance of up to 5 ohms per square, drops along silicide wiring between differential pairs can use up to 20mV. During automatic leaf cell generation the use of silicide is limited so that this term in the noise budget is not exceeded. 17mV is allowed for the total drop on V_{cc} between the chip bond pads and any device on the chip. Conformance with this requirement was verified before tapeout with a CAD tool we developed which is discussed in Section 5. The last term of 160mV is required to guarantee 100:1 switching of current in a receiving gate even if all other noise margin terms are used.

The swing below the reference was determined by six terms. As with the swing above the reference, a term of 160mV is required for 100:1 switching of current in a receiving gate. There is also a 17mV allowance for the total drop on V_{ee} between the bond pads and any device. Keeping this small reduces current variation between current sources. Since a base current flows from the bias generators to the current sources in each gate, an IR drop can occur on the bias routing. 8mV is allocated for this. The drops on V_{cs} and on V_{ee} are verified with the same tool used for verifying the delta V_{cc} term. Again, 20mV may be dropped on the low side in silicide routing within leaf cells. The variation in signals due to shifting with emitter followers or level shifters running at different current densities is split between the high side and the low side, with

25mV being budgeted on the low side. The last term is a delta V_{be} term occurring whenever current may be split among different numbers of transistors in OR structures depending on the state of the data inputs. This results in a variation on V_{be} of V_T*ln(n) for a *n*:1 variation. In this design we limited the maximum ORing to 8, resulting in a 70mV variation in V_{be}.

4.2. Clock Distribution

The chip uses a single-phase differential clock. All on-chip non-RAM state devices are flip-flops with scan. The PLL has additional circuitry for generating single-step clocks at high frequencies. The PLL also generates scan clocks for boundary scan of board clock flip-flops and interior scan of chip clock flip-flops. Since the entire die is covered with 1 mil thick gold bus bars, scan is essential for debugging the design.

The clock distribution network consists of three levels of H-trees (see Figure 7). Wires in this figure represent a differential clock signal. The vertical sections of the H-tree are implemented with the third metal layer, which is 2μm thick. Wires in the first level of H-tree are wider than minimum width to further reduce their intrinsic RC time constant. The center of the level-1 H-tree is driven by a large push-pull buffer. Successive levels of H-trees are connected by emitter followers. This results in lower delay, power, and skew than using buffers in the clock distribution network. Moreover, the third H-tree then has clocks at the proper level for use directly by flip-flops. Emitter followers are placed all along each H-tree to minimize delay, and different branches of the final H-tree are connected together to minimize skew. The chip contains 1623 flip-flops clocked by the chip clock, resulting in a total i_b of 6.2mA. The capacitive load on each wire of the differential pair at level 3 is 75pF. The two levels of emitter followers provide a large current gain, which minimizes IR drops in the first H-tree. The overall average delay from the output of the central clock buffer to flip-flop clock inputs is less than 100ps, and the skew between flip-flops is less than 50ps.

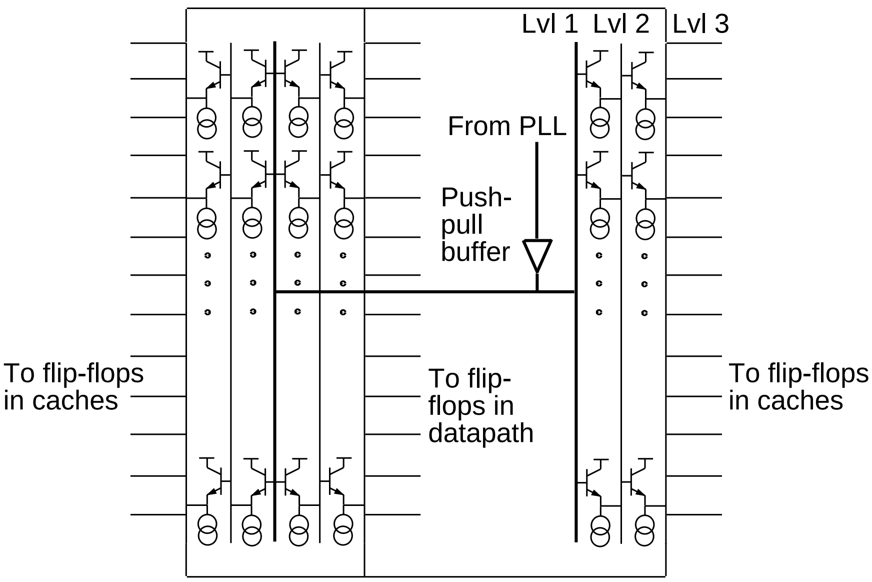

Figure 7: Clock distribution network

4.3. RAM Cell

There are several types of RAM cells available in a pure bipolar technology. PNP loaded cells have good density, but require write cycles that are longer than their read cycles. One of the key requirements on the RAM design for this chip were sub-2ns read and write. This directed the design to a pseudo-Schottky diode-clamped cell instead of a PNP cell. The RAM cell used in the caches is drawn in Figure 8. Diode clamping was accomplished with non-saturating NPN transistors, which keep the clamp current off the top word line. This RAM cell's area is 954μm^2. The unselected cell swing is 330mV with a standby current of 59µA. The data and instruction caches can simultaneously read or write 171 bits of data, parity, and tags. Although the pure bipolar cell provides high performance, the bit density is four or five times less than that available in a CMOS or BiCMOS cell.

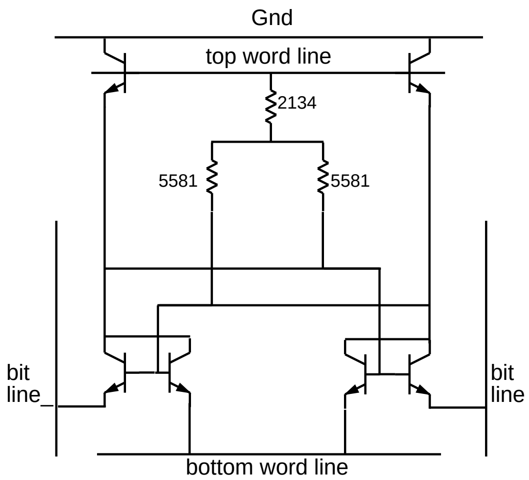

Figure 8: Cache RAM cell

4.4. Biases

The chip contains 30 internal master bias circuits, whose outputs are tied together. The master biases are distributed throughout the chip to minimize the difference in supply voltage and temperature between the biases and circuits. The master biases drive 420 internal slave biases. The slave biases generate the ECL reference signals and V_{cs}. Master and slave biases for ECL 100K I/O circuits are not temperature or voltage compensated, so they were placed in the four corners of the padframe for layout convenience.

4.5. Testing

The chip was designed for a worst-case clock frequency of 275MHz at a nominal -5.2V supply. Operation in excess of 335MHz has been observed with a -3.9V supply. Testing was typically performed with an internal clock four times faster than the external clock. Since no signals at the chip pins transition at the internal clock frequency, the test environment only needed to run up to 85MHz. A special-purpose board containing a dual-ported second-level cache was built for testing. The test board was built with off-the shelf ECL components.

Figure 9 is a schmoo plot of cycle time versus supply voltage. Unlike CMOS chips, the part operates faster with a lower supply voltage. This can be explained as follows. When operating at the reduced supply voltage of -3.9V, the base-collector junction of the current source transistor becomes forward biased, soft-saturating the current source NPN. This slightly reduces the current and hence the swing in the logic portions using three-level series gating. However, the current in the emitter followers is not reduced because of the smaller V_{be} stack on their current source transistors. This shows that there is significant noise margin present during operation with a nominal supply.

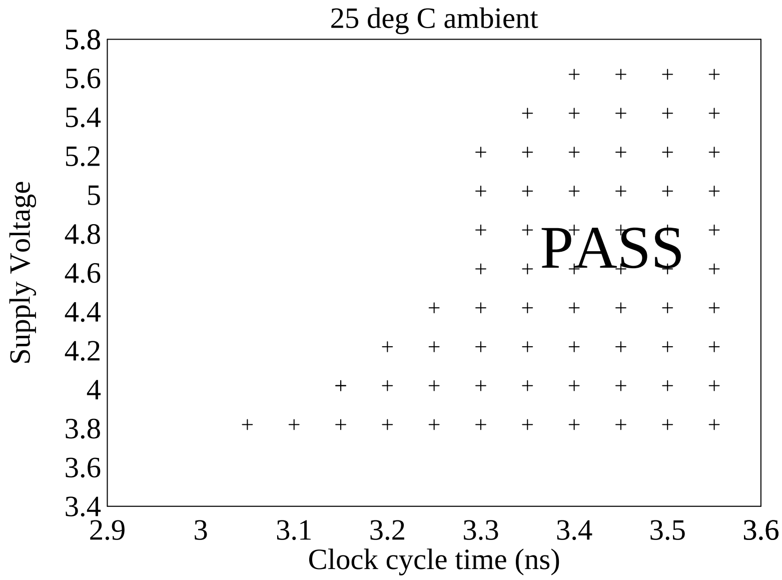

Figure 9: CPU operating frequency vs. supply voltage

5. CAD

The chip was designed largely with CAD tools developed by members of the design team [3]. Our tools allow us to control all aspects of the processor’s circuits and layout. All tools work at the device level, not the gate level, and there is no fixed gate library of circuits from which designers or synthesis tools must choose. Our system enables quick prototyping followed by incremental refinement. This allows us to do early and continual floorplanning, global performance tuning, and to track changes in technology.

We made several implementation choices that directed much of our effort. Perhaps the most important decision was to represent circuits as programs. Programs are simultaneously the most powerful and most modifiable descriptions we knew of. Much of the design process was cast as a problem in software development, permitting the use of standard programming tools to change and debug our design. We decided to develop many of our CAD tools as libraries which could be linked and run with the circuit design.

5.1. Design Capture

There is no single best way to describe circuits and logic. For analog circuits such as RAMs, schematic drawings of interconnected transistors are the most concise specification. For control logic, Boolean equations allow easiest debugging. For cell generators, such as a parameterized n-bit adder, a program is the most flexible representation. Rather than attempting to mix several different forms of circuit description, we chose to use their greatest common divisor, the program, and to translate schematics and Boolean equations into programs.

A circuit in our system is a C++ program. A procedure in this program is a cell generator. It can take arbitrary parameters, and returns the netlist for the requested cell. Many such generators take simple parameters, such as the amount of current drive to provide in the outputs, but some are quite complicated. For example, instead of having a library of OR-gates, we have an OR-gate generator, to which we pass the number of inputs, and a description of the outputs required. At this level, our form of description is quite like other hardware description languages.

Another library of C++ functions provides the syntax of Boolean equations, which are extensively used for control logic and for prototyping new blocks of logic. The library maps these into valid ECL gates (such as n-input OR/NOR gates). The mapping used in this chip was a fairly direct mapping that preserved the logic structure written by the user. The result of calling a cell generator defined by Boolean equations is a netlist identical to that which would be obtained by explicitly interconnecting a collection of gates, flip-flops and multiplexors; we trust the equation mapper to make this translation on parts of the circuit where the precise selection of the gates used is not critical.

Schematics are translated into a program which is the equivalent structural description of interconnected devices. This is done by analogy with programming. Our "source code" is a drawing produced by a conventional drawing editor which has no specialized knowledge of schematics, just as typical text editors have no knowledge of programs or text documents. We then translate this drawing into C++ code using a drawing interpreter called **drip** developed for the project which interprets lines as wires, and names as labels of wires and devices. It uses only visible cues in the drawing to parse it into devices and wires, and can put arbitrary code from the schematic, such as loops and tests, into the generated program. The example schematic in Figure 10 shows a cell generator containing a *for* loop. The output of **drip** is a program identical (except for verbosity) with one written by hand.

To generate a netlist for the entire chip, we translate all schematics into C++. The C++ source code is then compiled, as are the files containing Boolean equations and the hand-written cell generators, and all are linked with the CAD libraries. We also include a main program which calls the generator for the top-level cell of the chip. When the resulting program is executed it generates an in-memory data structure representing the circuit netlist.

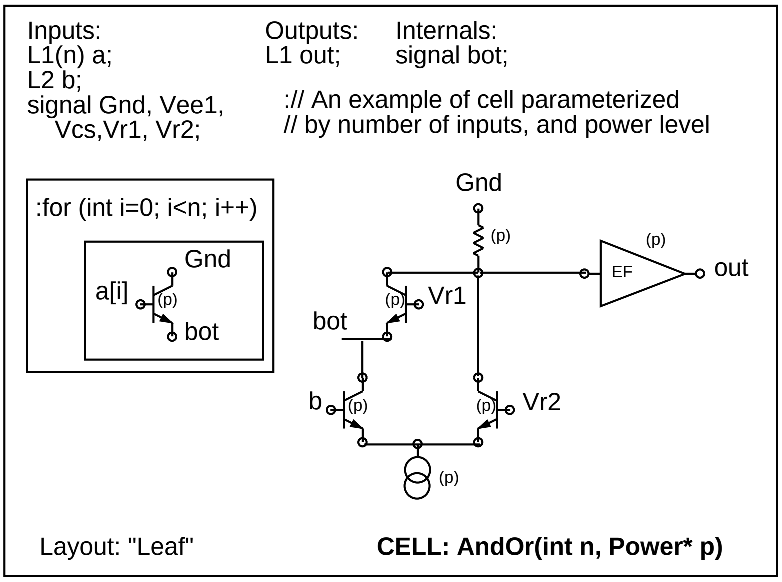

Figure 10: Typical cell schematic

5.2. Simulation

The CAD libraries contain procedures to traverse the in-memory netlist and produce input files for various simulators. We use **spice** for circuit simulation of analog circuits, and **bisim** [6] for switch-level simulation of digital circuits.

Bisim is an event driven switch-level simulator for ECL circuits written in conjunction with Stanford University. **Bisim** models node voltages using piecewise linear waveforms, and bipolar transistors using voltage controlled current switches, in order to simulate the behavior of ECL current steering trees. **Bisim** is also capable of mixed mode simulation. That is, circuits described at the transistor level can be freely mixed with higher level blocks whose logical behavior is described by C++ code.

Cells in the in-memory netlist can carry behavioral models provided as C++ routines by the designer, or generated automatically by the CAD system. In particular, a cell can both carry a behavioral model, and be defined as an interconnection of instances of lower level cells. The procedure which traverses the in-memory netlist to generate **bisim** input files allows the user to select the level of detail for simulation by specifying which of the behavioral models should be used, and which should be ignored in favor of lower-level models on subcells. The result is a **bisim** netlist at the level of detail requested by the user, and a file of C++ behavioral models which need to be linked with the simulator. Most of these behavioral models are at the gate level, because the library of generators for ECL gates automatically attaches a behavioral model to each gate. Large memory blocks carry hand-written behavioral models.

5.3. Generation of Layout

Layout can also be generated from a traversal of the in-memory netlist. In the same way that cells can carry behavioral models for simulation, cells also carry layout recipes which define how their layout is to be generated. A layout recipe is simply the name of a C++ procedure, so this mechanism is general and extensible. As new styles of layout are developed, they are described as new layout recipes which can be attached to cells. This recipe is an integral part of the definition of the cell, and is specified by the designer just like the wires defining the cell's interface. If the same circuit needs to be laid out in two different ways, it is described by a cell generator accepting the layout recipe as a parameter. Two different cells will result from calls supplying different layout recipe arguments. They will have identical netlists, but different layouts.

The layout process is a bottom-up, batch process. No interaction from the designer is required, since all the information needed to generate the layout is in the netlist and in the layout recipes. The generation of layout begins at leaf cells, which do not depend on layout of subcells. Leaf cells can either be hand-drawn or synthesized. Hand-drawn layouts are used for analog cells such as pads and voltage references, and in cases where hand-drawn layout offers a significant density advantage, such as memories and shifters. Only 17% of the cells in our design are hand-drawn, but they constitute well over half of the devices in the final layout. We used **magic** [7] for examining layout and creating the hand-drawn cells.

Most other leaf cells are synthesized. Typically, these cells are at the gate level, containing up to fifty transistors. The cell synthesizer produces finished layout given the netlist of devices, a set of templates for devices, and a vertical pitch. The placement uses no hints from the designer, but selects positions of the transistors and resistors which maximize the use of silicide interconnect. The resulting placement is very close to the density of hand designs. In part this is due to the regularity of current trees in ECL logic and the similar sizes of bipolar devices. Routing of leaf cells is done first in silicide. The connections which cannot be made in the planar silicide routing are given contacts to first-level metal, and then the router is called again to finish the routing using the metal layers. Placement and routing for a typical leaf cell takes under a minute on a DECStation 5000/200.

This style of leaf cell synthesis may be contrasted with semi-custom design. In our system there is no cell library, and any one of the enormous number of legal ECL gates may actually be used. The particular gate selection is determined by the parameters passed to the gate generator procedures during creation of the netlist. Figure 11 shows a schematic representation of the netlist produced by a call to a Mux-Flip-Flop generator with the following procedure call:

```
MuxFFHoldCell("T2_T2_T2", "T_T_L", "T_C", p, "Leaf2");
```

This procedure call requests a flip-flop with a built-in 4-input multiplexor. The first parameter "T2_T2_T2" specifies that the three select inputs will each be at level two, and that they are all high-true. Only three mux selectors are specified since the fourth default input always acts to recirculate the value held in the flop when no other selects are held high. The second parameter ("T_T_L") specifies that the data inputs are high-true, except for the third input which is a constant zero ("low"). The MuxFFHoldCell generator then knows to directly connect the collector of the third select arm to the true output, causing it to go low if selected. The third parameter ("T_C") species that both true and complement outputs of the cell are required. The fourth parameter specifies the power level of the switch, emitter followers, and any level shifters in the cell. The last parameter specifies the layout characteristics of the cell when layout is generated. In this case "Leaf2" specifies a format with two rows of transistors. There are similar generators for multiplexors, OR gates, AND gates, and other functions.

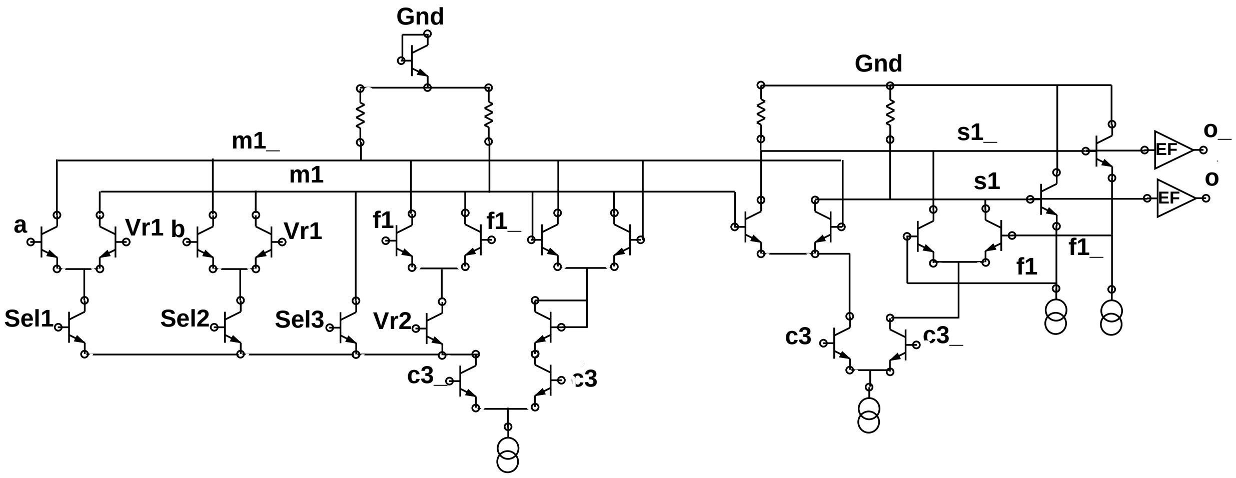

Figure 11: Flip-flop with built-in 4-input multiplexor

Figure 12 shows the MuxFFHoldCell after placement and silicide wiring. The horizontal bar at the top of the cell is a metal 3 V_{cc} bus bar; the metal 3 bus bar at the bottom of the cell is for V_{ee}. The cell is composed of three sections: a top resistor tray, an array of transistors, and a bottom resistor tray. The top resistor tray is primarily used for load resistors; the bottom tray is useful for current source resistors. Based on parameters to the leaf cell generator, layouts with different numbers of resistor trays and transistor rows can be generated. For example, some cells in the design have up to 5 transistor rows to make them pitch-matched to memory peripheral logic. We connect to leaf cell transistors in a collector-base-emitter (CBE) configuration because it allows planar wiring of the primary current paths within the cell (see Figure 13). The base of the transistor is connected to silicide on two sides, and the emitter silicide wraps down around the lower base connection. Since current primarily flows from the top to the bottom of the cell, when transistors are stacked, emitter and collector terminals are adjacent. This is ideal for the wiring of series-gating sections.

Figure 14 shows the same cell after routing. The intra-cell routing can be completed in metal 1 and a little metal 2. This leaves all of the metal 3 over the cell (except for the power bus bars) and most of metal 2 for inter-cell routing.

For non-leaf cells, the designer must choose an appropriate layout recipe. This will depend on how the cell is to fit in the floorplan of the chip. Layout recipes vary over the spectrum of greater convenience to greater control. At the convenience end, a fully automatic placer was used to generate layout for standard-cell-like blocks of control logic. At the control end, designers use a collection of recipes based on corner alignment of subcells to specify placement directly.

A large fraction of the development effort was spent on a general router based on a hybrid maze/line search principle [2]. An important point about the router is that it is used in each cell

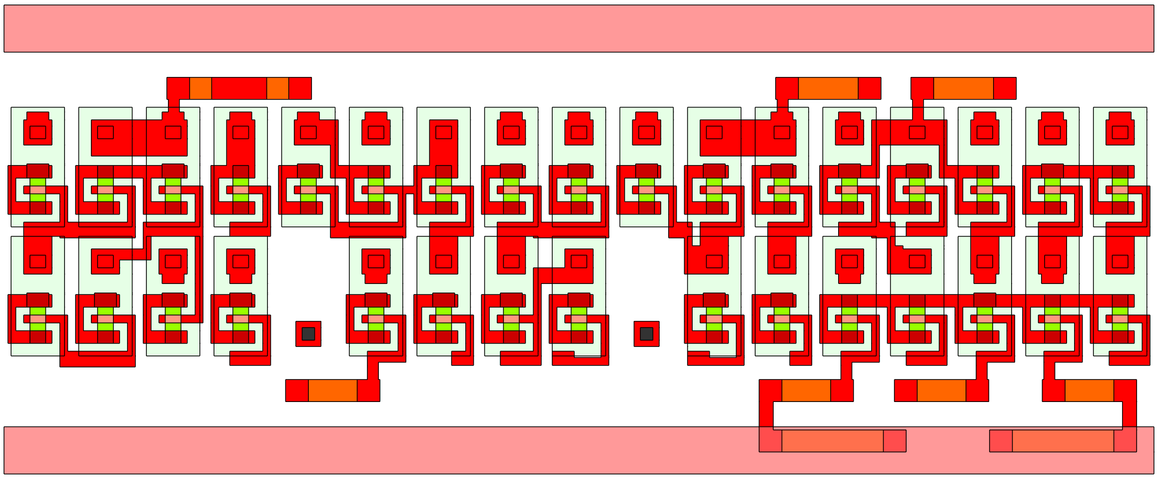

Figure 12: Gate generated with silicide routing

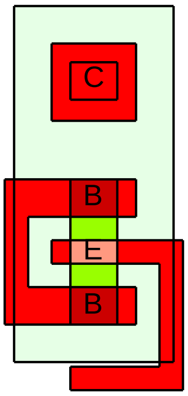

Figure 13: CBE transistor configuration

of the design to complete connections not made by placement. In general therefore, the router is adding wires to cells on top of wires already routed in the subcells. Routing over the top of active logic is one of the characteristics of custom VLSI, and is largely responsible for its density. The router reads design rules from the same file used by the layout editor and the design rule checker. It can generate routing with minimum dimensions and clearances from obstacles on all wiring layers simultaneously. There is only one router, and it is used repeatedly at all levels of the design, from routing silicide in the leaf cells to making millimeter-length connections at the chip level.

To recreate the chip layout including full routing from only the source schematics, source code, and hand-drawn cells takes about 10 hours on a DECStation 5000/200. However after a change to a cell in the design, only the cell being changed and all of its parents in the hierarchy will need to be regenerated because of cell caching. Thus it only requires a few hours to regenerate the chip layout after most changes.

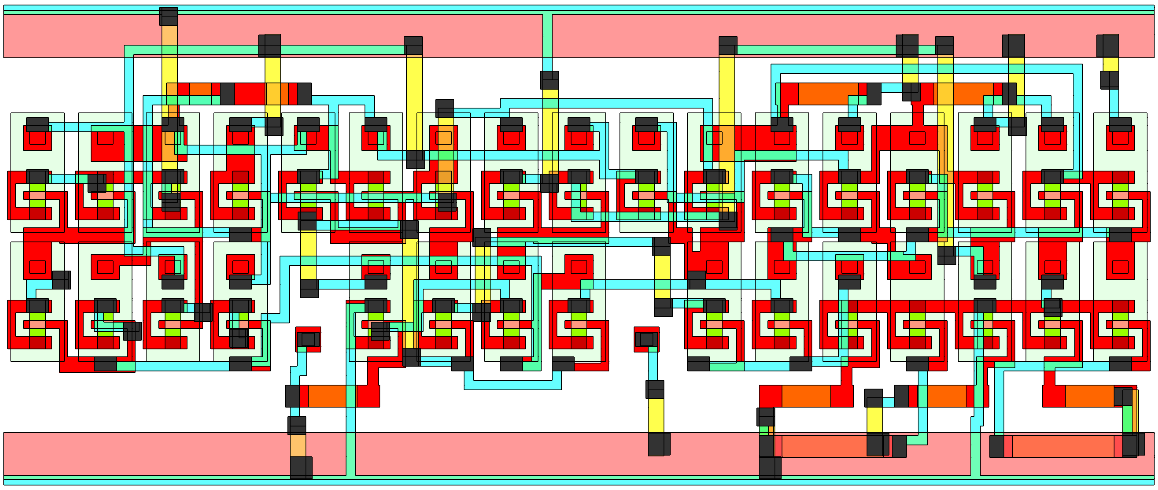

Figure 14: Gate with silicide and metal routing

5.4. Design Verification

Extensive simulation was performed with **bisim**, from the behavioral level through the transistor level. In the initial stages of the design behavioral models were used exclusively. Then, as blocks were more completely implemented, portions of the design could be flattened to verify those implementations. At the very end, as a final check of all the tools, the entire chip, including the memories, was extracted from the layout and simulated with **bisim** at the transistor level.

Besides **bisim**, three other tools were used extensively in design verification. **Ariel** verified IR drops on power supplies, biases, and crucial signals, **secure** verified transistor-level ECL design rules, and **bitv** verified ECL noise margins and provided transistor-level whole-chip timing verification.

Ariel has three main modules. The first extracts a resistance network for power networks and other crucial signals from the mask geometry. The second obtains the currents in each gate from the netlist; this is easy since the power dissipation of the chip is static. The third module puts the resistance network together with all the currents injected into the network and solves for the voltage drop at each gate. Figure 15 is a plot of the drop on all the V_{ee} wiring of the chip. In general, the drop increases when traveling towards the vertical center of the chip since both power rails are supplied from both the top and bottom of the chip. The worst IR drop is in the datapath, and is less than 15mV. This is within our 17mV noise margin allowance. Figure 16 shows the IR drops on the V_{cs} network. The centers of the RAM blocks appear white on this graph because they do not use V_{cs} and therefore there is no V_{cs} wiring over the RAMs. For V_{cs}, the worst drop is 9.5mV in the PLL. This is larger than our 8mV single-ended swing noise margin budget, but the PLL uses differential logic which has a 10mV delta V_{cs} noise margin. The second worst area for V_{cs} drops is the center of the datapath, but this is only 6mV, which is well within the 8mV noise budget.

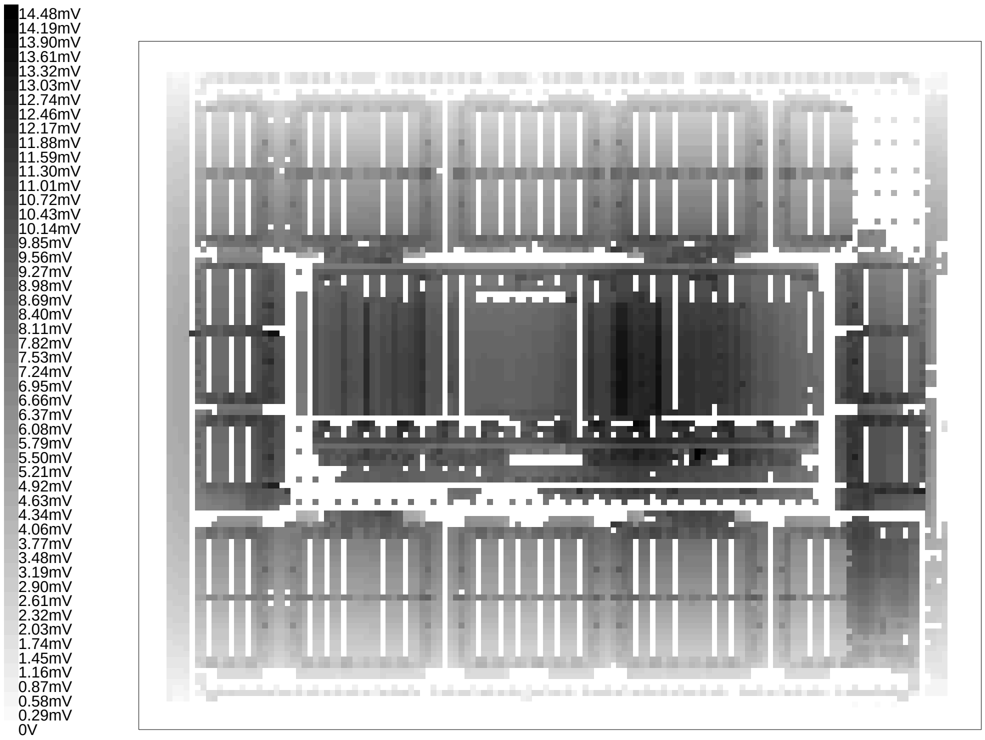

Figure 15: IR drops on V_{ee}

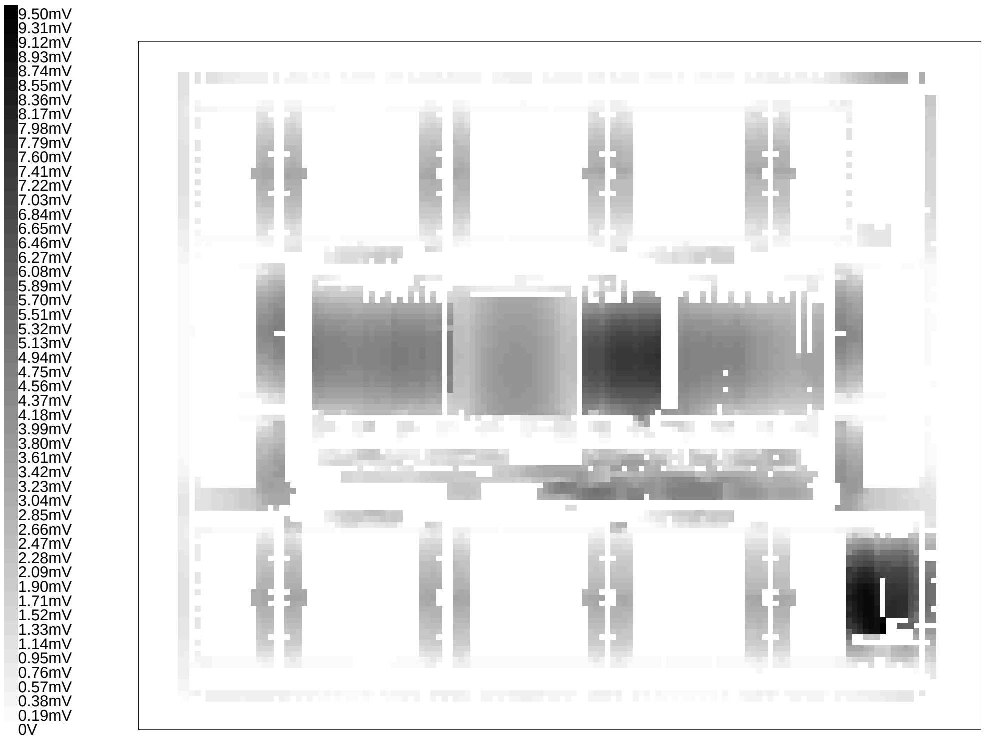

Figure 16: IR drops on V_{cs}

Secure is a static checker for ECL circuits that searches for syntactic errors in a design. It checks for shorted references, dangling device terminals, saturating transistors, excessively high current density, and devices that never turn on. It also calculates the circuit's power consumption for use by **ariel**.

Bitv is a pattern-independent transistor level circuit analysis and timing analysis tool that was also developed as part of the project. First, it reads a flat transistor netlist and parses the netlist into ECL gates. It then performs a static circuit analysis. This forms the basis for its verification of a number of terms in the noise margins: the I_b*R_l term, the delta J_e terms and the delta V_{be} term related to OR configurations. Finally, **bitv** performs a breadth-first traversal of the whole circuit to determine critical timing paths. **Bitv** computes separate delays for rising and falling edges. It can predict most ECL gate delays within +/- 20%. It was run on a complete chip netlist extracted from layout before tapeout, and predicted a worst-case clock frequency of 275Mhz.

5.5. CAD Summary

Perhaps the most important feature of the CAD system developed for this chip is that it allowed as much control of each piece of the design as needed by the designers. Critical circuits were designed and layed out by hand, while non-critical circuits could be machine-generated. As another example, the placement of cells in the datapath, memories, and the global floorplan were all specified by hand, but the placement of cells in non-critical blocks of control logic was done automatically. This flexibility is essential for enabling the construction of a high-performance design with a minimum of design effort.

6. Packaging

Figure 17 is a picture of a package and thermosiphon. Figure 18 is an exploded view of the package assembly [4]. Note that the package does not have a conventional slug. The chip is attached with low modulus epoxy to a copper spreader which serves as both a slug and the boiling surface for a thermosiphon (a heatpipe without a wick). This reduces the number of interfaces through which the heat must travel. In operation the fluid boils and vapor rises and condenses on the walls of the condenser, transferring its heat to the passing air. The combination of phase change and mass transport allows much lower thermal resistances than are possible with a solid heatsink and thermal conduction. The fluid (typically an alcohol-water mixture) is maintained at sub-atmospheric pressure, so that boiling will occur at low temperatures. The airflow required by the condenser is quite modest: quiet, low velocity air such as is typically found in a desktop workstation is sufficient. A typical maximum operating junction temperature is 100^{o}C. Under normal operation there is a temperature gradient of 25^{o}C across the die.

A CAD tool was developed during the project for predicting the die temperature during operation. **Thermap** uses a relaxation-based approach to solve a three-dimensional thermal model of the chip, die attach, and boiler, usually taking a few minutes on a DECStation 5000/200. Figure 19 shows the predicted die temperature profile. The temperature profile measured on operating chips with infrared photography (see Figure 20) is remarkably close to that predicted by the thermal model.

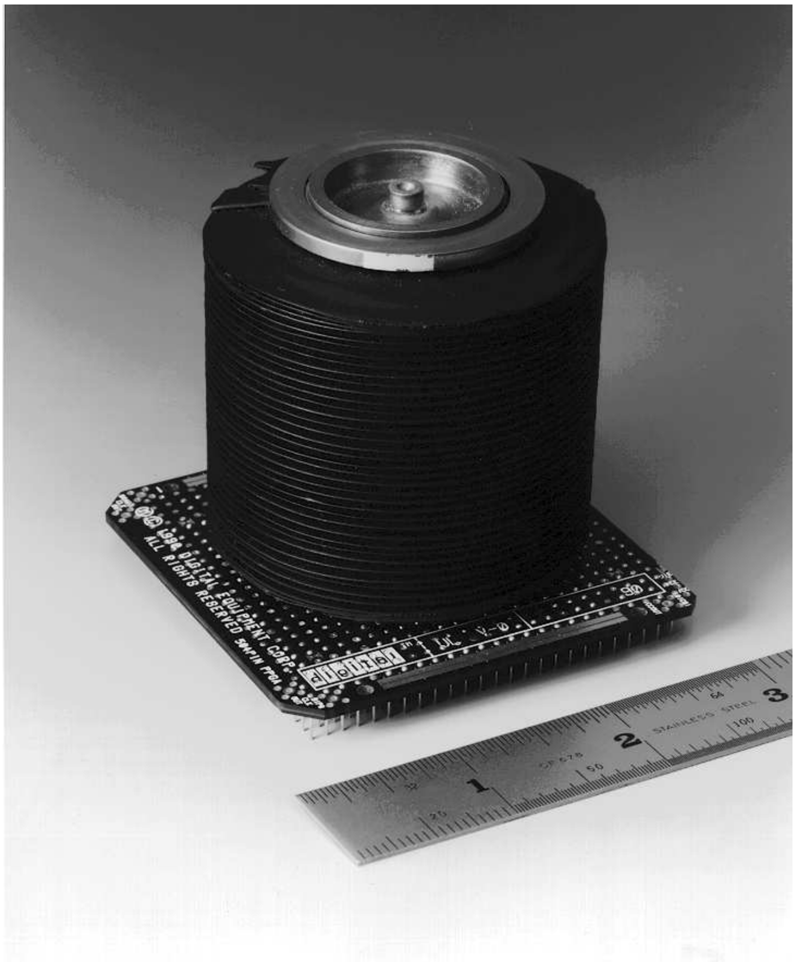

Figure 17: Package with thermosiphon

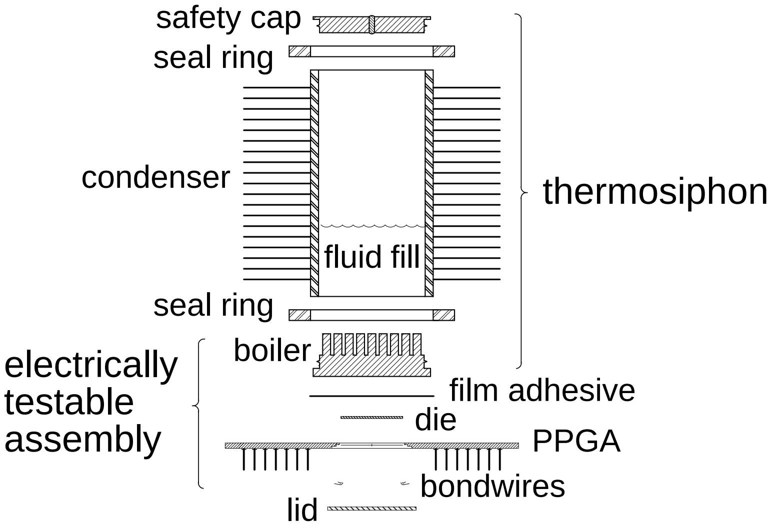

Figure 18: Exploded view of package assembly

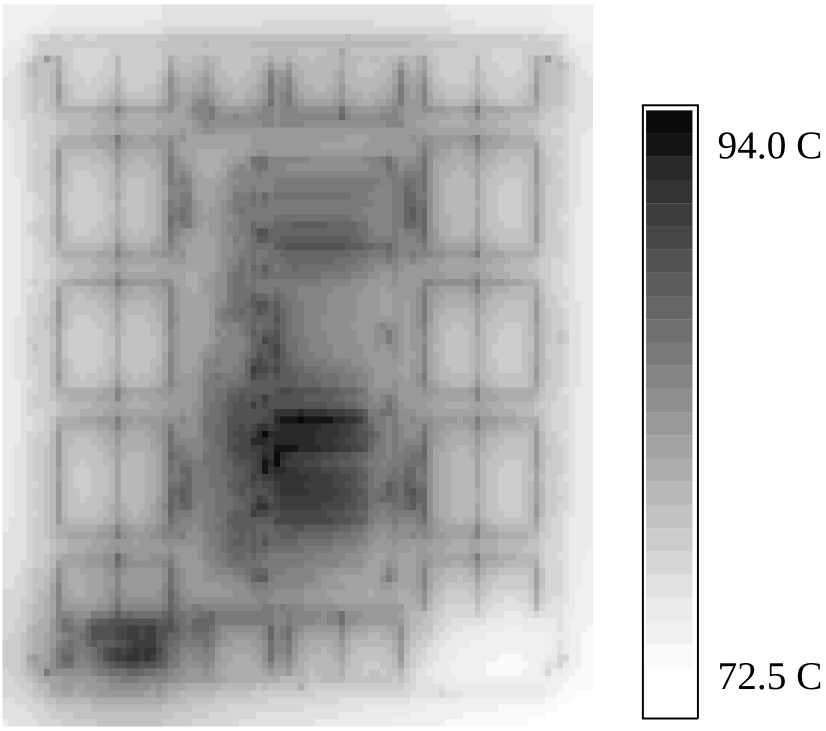

Figure 19: Numerical model of die temperature

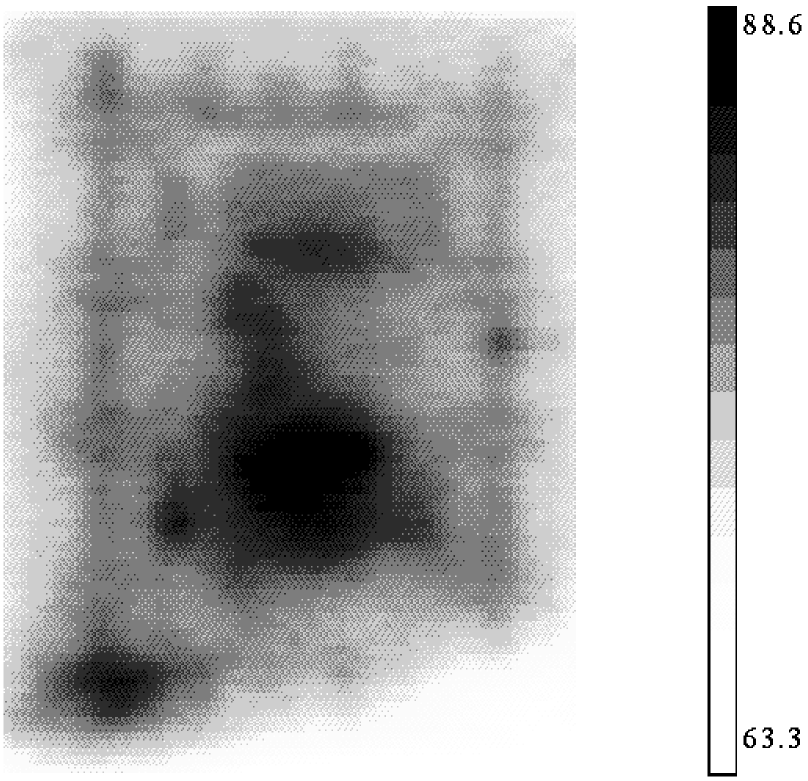

Figure 20: Infrared photograph of operating chip

The package is a plastic pin grid array (PPGA). Although not hermetic, the PPGA offers a number of advantages over ceramic packages, the most important of which is the ability to have 50 ohm impedance wiring on the package. All input pads include 50 ohm on-chip termination resistors to a -2V supply. Each termination resistor requires 29mW, and the total of 202 inputs requires 5.8W. This would be a prohibitive amount of power for many chips but in our case it is an insignificant fraction of the total chip power. The combination of the 50 ohm transmission line environment all the way to the bondwire and on-chip termination resistors allow ECL-level signal edge rates down to 150ps [1].

A two-row bondwire arrangement is used. The inner row of power pads is bonded with 1.8mil gold wires, and signals in the outer row are bonded with 1.25mil gold wires. Tests have shown that these larger gold power bond wires can have fusing currents in excess of 150A to a die the size of the processor.

7. Summary

This paper has described the application of full-custom design techniques to an ECL processor. The resulting logic density of large functional units such as datapaths is similar to full-custom CMOS when implemented with the same feature size lithography. Moreover, the circuit performance has been increased significantly by using different signal swings in different applications, and by using circuit topologies (such as low-swing cascode and wired-OR circuits) which are difficult to use in gate arrays.

A major part of this project was the development of a suite of CAD tools which allowed automatic layout generation of most of the design. Tools capable of switch-level simulation and timing verification of the entire chip at the transistor level were also developed. In addition, detailed verification of noise margins was performed on the whole die down to the mV. This CAD suite was based on the idea of representing the design as a C++ program. This allowed the full power, functionality, and mutability of a modern programming language to be exploited in the design process.

Finally, developments in high-performance packaging for this die included power distribution with gold bus bars, two row gold wire bonding, thermosiphon cooling, and CAD for accurate modeling of die temperatures before tapeout. The plastic pin grid array used in conjunction with on-chip termination of inputs enables very high performance off-chip communication.

Acknowledgements

The authors wish to thank Wayne Mack and Joel Bartlett for their help and assistance. The authors also want to thank Motorola Semiconductor ASIC Products Division and Advanced Custom Technology Center for fabrication and their contributions to the project.

References

[1] Patrick Boyle. *Electrical Evaluation of the BIPS-0 Package*. Technical Report TN-29, Digital Equipment Corporation Western Research Laboratory, July, 1992.

[2] Jeremy Dion. *Fast Printed Circuit Board Routing*. WRL Research Report 88/1, Digital Equipment Corporation Western Research Laboratory, 1988.

[3] Jeremy Dion and Louis Monier. *Design Tools for BIPS-0*. WRL Research Report 91/3, Digital Equipment Corporation Western Research Laboratory, December, 1992.

[4] William R. Hamburgen, and John S. Fitch. Packaging a 150W Bipolar ECL Microprocessor. *IEEE Components, Hybrids, and Manufacturing Technology* 16(1):pages 28-38, February, 1993.

[5] Norman P. Jouppi, et. al. A 300Mhz 115W 32b Bipolar ECL Microprocessor with On-Chip Caches. *International Solid-State Circuits Conference* :pages 84-85, February, 1993.

[6] Russell Kao, Bob Alverson, Mark Horowitz and Don Stark. Bisim: A Simulator for Custom ECL Circuits. In *IEEE International Conference on Computer-Aided Design*, pages 62-65. Santa Clara, California, November, 1988.

[7] Robert N. Mayo, Michael H. Arnold, Walter S. Scott, Don Stark, Gordon T. Hamachi. *1990 DECWRL/Livermore Magic Release*. WRL Research Report 90/7, Digital Equipment Corporation Western Research Laboratory, 1990.

[8] Sunil Mirapuri, Michael Woodacre, and Nader Vasseghi. The MIPS R4000 Processor. *IEEE Micro* 12(2):pages 10-22, April, 1992.

[9] Don Stark. *Analysis of Power Supply Networks in VLSI Circuits*. Technical Report 91/3, Digital Equipment Corporation Western Research Laboratory, April, 1991.

[10] P. K. Tien. Propagation Delay in High Speed Silicon Bipolar and GaAs HBT Digital Circuits. *International Journal of High Speed Electronics* 1(1):pages 101-124, March, 1990.

WRL Research Reports

‘‘Titan System Manual.’’
Michael J. K. Nielsen.
WRL Research Report 86/1, September 1986.

‘‘Global Register Allocation at Link Time.’’
David W. Wall.
WRL Research Report 86/3, October 1986.

‘‘Optimal Finned Heat Sinks.’’
William R. Hamburgen.
WRL Research Report 86/4, October 1986.

‘‘The Mahler Experience: Using an Intermediate Language as the Machine Description.’’
David W. Wall and Michael L. Powell.
WRL Research Report 87/1, August 1987.

‘‘The Packet Filter: An Efficient Mechanism for User-level Network Code.’’
Jeffrey C. Mogul, Richard F. Rashid, Michael J. Accetta.
WRL Research Report 87/2, November 1987.

‘‘Fragmentation Considered Harmful.’’
Christopher A. Kent, Jeffrey C. Mogul.
WRL Research Report 87/3, December 1987.

‘‘Cache Coherence in Distributed Systems.’’
Christopher A. Kent.
WRL Research Report 87/4, December 1987.

‘‘Register Windows vs. Register Allocation.’’
David W. Wall.
WRL Research Report 87/5, December 1987.

‘‘Editing Graphical Objects Using Procedural Representations.’’
Paul J. Asente.
WRL Research Report 87/6, November 1987.

‘‘The USENET Cookbook: an Experiment in Electronic Publication.’’
Brian K. Reid.
WRL Research Report 87/7, December 1987.

‘‘MultiTitan: Four Architecture Papers.’’
Norman P. Jouppi, Jeremy Dion, David Boggs, Michael J. K. Nielsen.
WRL Research Report 87/8, April 1988.

‘‘Fast Printed Circuit Board Routing.’’
Jeremy Dion.
WRL Research Report 88/1, March 1988.

‘‘Compacting Garbage Collection with Ambiguous Roots.’’
Joel F. Bartlett.
WRL Research Report 88/2, February 1988.

‘‘The Experimental Literature of The Internet: An Annotated Bibliography.’’
Jeffrey C. Mogul.
WRL Research Report 88/3, August 1988.

‘‘Measured Capacity of an Ethernet: Myths and Reality.’’
David R. Boggs, Jeffrey C. Mogul, Christopher A. Kent.
WRL Research Report 88/4, September 1988.

‘‘Visa Protocols for Controlling Inter-Organizational Datagram Flow: Extended Description.’’
Deborah Estrin, Jeffrey C. Mogul, Gene Tsudik, Kamaljit Anand.
WRL Research Report 88/5, December 1988.

‘‘SCHEME->C A Portable Scheme-to-C Compiler.’’
Joel F. Bartlett.
WRL Research Report 89/1, January 1989.

‘‘Optimal Group Distribution in Carry-Skip Adders.’’
Silvio Turrini.
WRL Research Report 89/2, February 1989.

‘‘Precise Robotic Paste Dot Dispensing.’’
William R. Hamburgen.
WRL Research Report 89/3, February 1989.

‘‘Simple and Flexible Datagram Access Controls for Unix-based Gateways.’’
Jeffrey C. Mogul.
WRL Research Report 89/4, March 1989.

‘‘Spritely NFS: Implementation and Performance of Cache-Consistency Protocols.’’
V. Srinivasan and Jeffrey C. Mogul.
WRL Research Report 89/5, May 1989.

‘‘Available Instruction-Level Parallelism for Superscalar and Superpipelined Machines.’’
Norman P. Jouppi and David W. Wall.
WRL Research Report 89/7, July 1989.

‘‘A Unified Vector/Scalar Floating-Point Architecture.’’
Norman P. Jouppi, Jonathan Bertoni, and David W. Wall.
WRL Research Report 89/8, July 1989.

‘‘Architectural and Organizational Tradeoffs in the Design of the MultiTitan CPU.’’
Norman P. Jouppi.
WRL Research Report 89/9, July 1989.

‘‘Integration and Packaging Plateaus of Processor Performance.’’
Norman P. Jouppi.
WRL Research Report 89/10, July 1989.

‘‘A 20-MIPS Sustained 32-bit CMOS Microprocessor with High Ratio of Sustained to Peak Performance.’’
Norman P. Jouppi and Jeffrey Y. F. Tang.
WRL Research Report 89/11, July 1989.

‘‘The Distribution of Instruction-Level and Machine Parallelism and Its Effect on Performance.’’
Norman P. Jouppi.
WRL Research Report 89/13, July 1989.

‘‘Long Address Traces from RISC Machines: Generation and Analysis.’’
Anita Borg, R.E.Kessler, Georgia Lazana, and David W. Wall.
WRL Research Report 89/14, September 1989.

‘‘Link-Time Code Modification.’’
David W. Wall.
WRL Research Report 89/17, September 1989.

‘‘Noise Issues in the ECL Circuit Family.’’
Jeffrey Y.F. Tang and J. Leon Yang.
WRL Research Report 90/1, January 1990.

‘‘Efficient Generation of Test Patterns Using Boolean Satisfiablilty.’’
Tracy Larrabee.
WRL Research Report 90/2, February 1990.

‘‘Two Papers on Test Pattern Generation.’’
Tracy Larrabee.
WRL Research Report 90/3, March 1990.

‘‘Virtual Memory vs. The File System.’’
Michael N. Nelson.
WRL Research Report 90/4, March 1990.

‘‘Efficient Use of Workstations for Passive Monitoring of Local Area Networks.’’
Jeffrey C. Mogul.
WRL Research Report 90/5, July 1990.

‘‘A One-Dimensional Thermal Model for the VAX 9000 Multi Chip Units.’’
John S. Fitch.
WRL Research Report 90/6, July 1990.

‘‘1990 DECWRL/Livermore Magic Release.’’
Robert N. Mayo, Michael H. Arnold, Walter S. Scott, Don Stark, Gordon T. Hamachi.
WRL Research Report 90/7, September 1990.

‘‘Pool Boiling Enhancement Techniques for Water at Low Pressure.’’
Wade R. McGillis, John S. Fitch, William R. Hamburgen, Van P. Carey.
WRL Research Report 90/9, December 1990.

‘‘Writing Fast X Servers for Dumb Color Frame Buffers.’’
Joel McCormack.
WRL Research Report 91/1, February 1991.

‘‘A Simulation Based Study of TLB Performance.’’
J. Bradley Chen, Anita Borg, Norman P. Jouppi.
WRL Research Report 91/2, November 1991.

‘‘Analysis of Power Supply Networks in VLSI Circuits.’’
Don Stark.
WRL Research Report 91/3, April 1991.

‘‘TurboChannel T1 Adapter.’’
David Boggs.
WRL Research Report 91/4, April 1991.

‘‘Procedure Merging with Instruction Caches.’’
Scott McFarling.
WRL Research Report 91/5, March 1991.

‘‘Don’t Fidget with Widgets, Draw!.’’
Joel Bartlett.
WRL Research Report 91/6, May 1991.

‘‘Pool Boiling on Small Heat Dissipating Elements in Water at Subatmospheric Pressure.’’
Wade R. McGillis, John S. Fitch, William R. Hamburgen, Van P. Carey.
WRL Research Report 91/7, June 1991.

‘‘Incremental, Generational Mostly-Copying Garbage Collection in Uncooperative Environments.’’
G. May Yip.
WRL Research Report 91/8, June 1991.

‘‘Interleaved Fin Thermal Connectors for Multichip Modules.’’
William R. Hamburgen.
WRL Research Report 91/9, August 1991.

‘‘Experience with a Software-defined Machine Architecture.’’
David W. Wall.
WRL Research Report 91/10, August 1991.

‘‘Network Locality at the Scale of Processes.’’
Jeffrey C. Mogul.
WRL Research Report 91/11, November 1991.

‘‘Cache Write Policies and Performance.’’
Norman P. Jouppi.
WRL Research Report 91/12, December 1991.

‘‘Packaging a 150 W Bipolar ECL Microprocessor.’’
William R. Hamburgen, John S. Fitch.
WRL Research Report 92/1, March 1992.

‘‘Observing TCP Dynamics in Real Networks.’’
Jeffrey C. Mogul.
WRL Research Report 92/2, April 1992.

‘‘Systems for Late Code Modification.’’
David W. Wall.
WRL Research Report 92/3, May 1992.

‘‘Piecewise Linear Models for Switch-Level Simulation.’’
Russell Kao.
WRL Research Report 92/5, September 1992.

‘‘A Practical System for Intermodule Code Optimization at Link-Time.’’
Amitabh Srivastava and David W. Wall.
WRL Research Report 92/6, December 1992.

‘‘A Smart Frame Buffer.’’
Joel McCormack & Bob McNamara.
WRL Research Report 93/1, January 1993.

‘‘Recovery in Spritely NFS.’’
Jeffrey C. Mogul.
WRL Research Report 93/2, June 1993.

‘‘Tradeoffs in Two-Level On-Chip Caching.’’
Norman P. Jouppi & Steven J.E. Wilton.
WRL Research Report 93/3, October 1993.

‘‘Unreachable Procedures in Object-oriented Programing.’’
Amitabh Srivastava.
WRL Research Report 93/4, August 1993.

‘‘Limits of Instruction-Level Parallelism.’’
David W. Wall.
WRL Research Report 93/6, November 1993.

‘‘Fluoroelastomer Pressure Pad Design for Microelectronic Applications.’’
Alberto Makino, William R. Hamburgen, John S. Fitch.
WRL Research Report 93/7, November 1993.

‘‘A 300MHz 115W 32b Bipolar ECL Microprocessor.’’
Norman P. Jouppi, Patrick Boyle, Jeremy Dion, Mary Jo Doherty, Alan Eustace, Ramsey Haddad, Robert Mayo, Suresh Menon, Louis Monier, Don Stark, Silvio Turrini, Leon Yang, John Fitch, William Hamburgen, Russell Kao, and Richard Swan.
WRL Research Report 93/8, December 1993.

‘‘Link-Time Optimization of Address Calculation on a 64-bit Architecture.’’
Amitabh Srivastava, David W. Wall.
WRL Research Report 94/1, February 1994.

‘‘ATOM: A System for Building Customized Program Analysis Tools.’’
Amitabh Srivastava, Alan Eustace.
WRL Research Report 94/2, March 1994.

‘‘Complexity/Performance Tradeoffs with Non-Blocking Loads.’’
Keith I. Farkas, Norman P. Jouppi.
WRL Research Report 94/3, March 1994.

‘‘A Better Update Policy.’’
Jeffrey C. Mogul.
WRL Research Report 94/4, April 1994.

‘‘Boolean Matching for Full-Custom ECL Gates.’’
Robert N. Mayo, Herve Touati.
WRL Research Report 94/5, April 1994.

WRL Technical Notes

‘‘TCP/IP PrintServer: Print Server Protocol.’’
Brian K. Reid and Christopher A. Kent.
WRL Technical Note TN-4, September 1988.

‘‘TCP/IP PrintServer: Server Architecture and Implementation.’’
Christopher A. Kent.
WRL Technical Note TN-7, November 1988.

‘‘Smart Code, Stupid Memory: A Fast X Server for a Dumb Color Frame Buffer.’’
Joel McCormack.
WRL Technical Note TN-9, September 1989.

‘‘Why Aren’t Operating Systems Getting Faster As Fast As Hardware?’’
John Ousterhout.
WRL Technical Note TN-11, October 1989.

‘‘Mostly-Copying Garbage Collection Picks Up Generations and C++.’’
Joel F. Bartlett.
WRL Technical Note TN-12, October 1989.

‘‘The Effect of Context Switches on Cache Performance.’’
Jeffrey C. Mogul and Anita Borg.
WRL Technical Note TN-16, December 1990.

‘‘MTOOL: A Method For Detecting Memory Bottlenecks.’’
Aaron Goldberg and John Hennessy.
WRL Technical Note TN-17, December 1990.

‘‘Predicting Program Behavior Using Real or Estimated Profiles.’’
David W. Wall.
WRL Technical Note TN-18, December 1990.

‘‘Cache Replacement with Dynamic Exclusion’’
Scott McFarling.
WRL Technical Note TN-22, November 1991.

‘‘Boiling Binary Mixtures at Subatmospheric Pressures’’
Wade R. McGillis, John S. Fitch, William R. Hamburgen, Van P. Carey.
WRL Technical Note TN-23, January 1992.

‘‘A Comparison of Acoustic and Infrared Inspection Techniques for Die Attach’’
John S. Fitch.
WRL Technical Note TN-24, January 1992.

‘‘TurboChannel Versatec Adapter’’
David Boggs.
WRL Technical Note TN-26, January 1992.

‘‘A Recovery Protocol For Spritely NFS’’
Jeffrey C. Mogul.
WRL Technical Note TN-27, April 1992.

‘‘Electrical Evaluation Of The BIPS-0 Package’’
Patrick D. Boyle.
WRL Technical Note TN-29, July 1992.

‘‘Transparent Controls for Interactive Graphics’’
Joel F. Bartlett.
WRL Technical Note TN-30, July 1992.

‘‘Design Tools for BIPS-0’’
Jeremy Dion & Louis Monier.
WRL Technical Note TN-32, December 1992.

‘‘Link-Time Optimization of Address Calculation on a 64-Bit Architecture’’
Amitabh Srivastava and David W. Wall.
WRL Technical Note TN-35, June 1993.

‘‘Combining Branch Predictors’’
Scott McFarling.
WRL Technical Note TN-36, June 1993.

‘‘Boolean Matching for Full-Custom ECL Gates’’
Robert N. Mayo and Herve Touati.
WRL Technical Note TN-37, June 1993.