FEAT'URES

FEATURE ARTICLE

Ron Stence

Speed upYour M68040 with an Interleaved Bursting EPROM Interface

Looking to improve the performance of your latest MC68040 or MC68060 design without locking yourself into a sole source of specialized memory devices? Explore one alternative that uses standard EPROMs.

An interleaved EPROM memory interface provides an impressive price/performance ratio for computer systems that require high-performance, EPROM-based program storage. During this article, I will explore the details of implementing an interleaved EPROM memory for systems based on the M68040 family of devices and the MC68060 processor. Since the MC68040, MC68LC040, and MC68EC040 are each pin, bus, and integer unit compatible, my discussion will apply to each of these devices. These processors will be referred to as an M68040 integer unit or as an M68040 processor.

In a design based on a 25MHz M68040 integer unit, utilization of the bursting capabilities causes a significant improvement in performance without more than an incremental increase in the overall system cost. Recently, I completed an evaluation of various memory systems capable of supporting the burst mode of the M68040 processors. Currently, some ROM manufacturers are focusing development efforts toward creating and marketing bursting EPROM technology. The cost for these devices is often prohibitive, while the memory

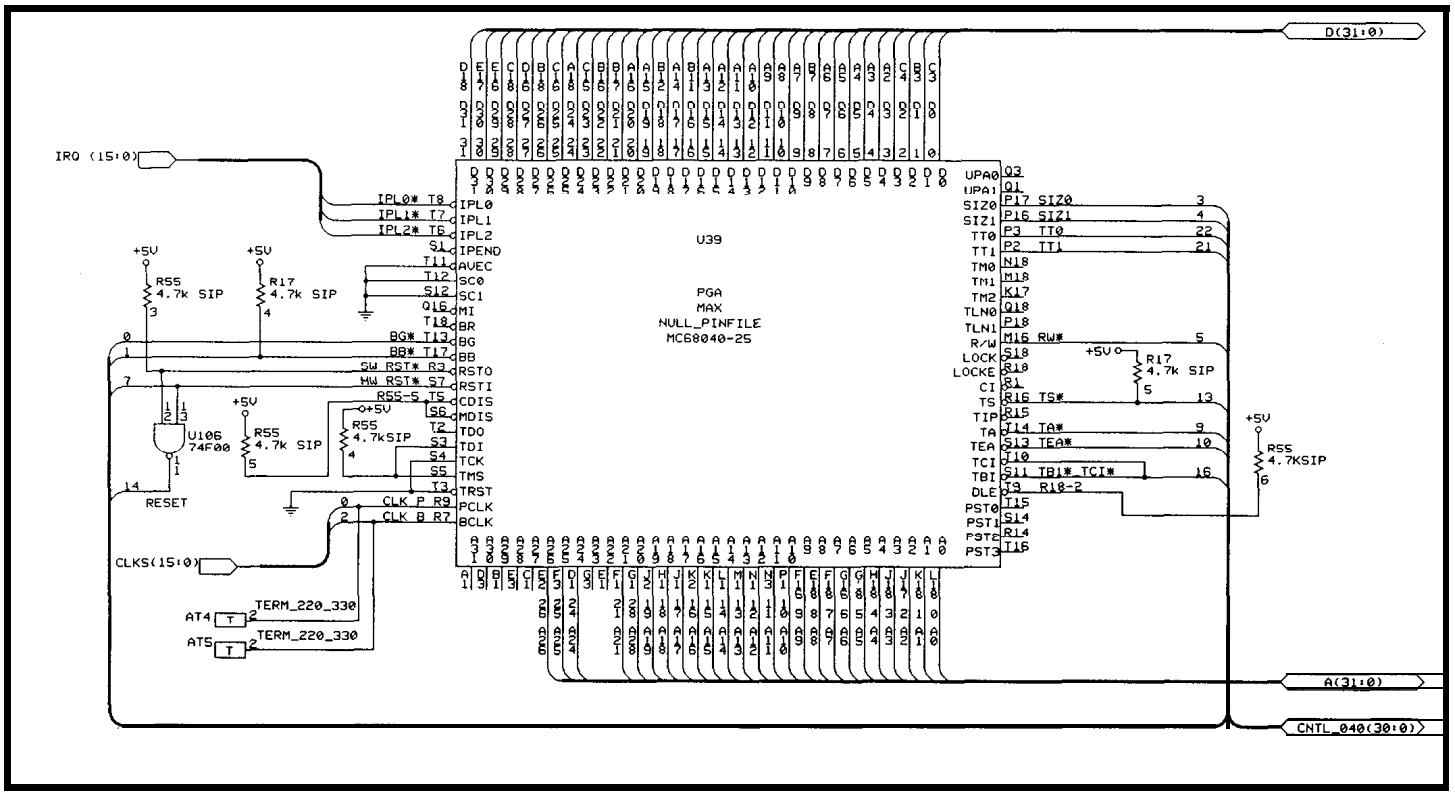


Figure 1a--At the core of the *sample design* is a Motorola *MC68040* processor. Performance can be improved by using an interleaved bursting EPROM interface.

size is limited when compared with other EPROM devices. Costs are also high because second sources for bursting EPROMs are not typically available. In this article, I will address other methods that can be used to create burst-enabled memory and provide data to support the performance improvements you may expect from an interleaved-based design.

CONSIDERING THE ALTERNATIVES

Several types of memory interfaces can be used to interface processors with today's memory devices. Most of these techniques are well known since EPROM technology has not changed much in the past few years. Notable changes in this arena have primarily focused on increasing memory size and decreasing the access time. Recently, several EPROM manufacturers have begun releasing bursting EPROMs. Designers should be careful about specifying the current offering of bursting EPROMs into their products due to the lack of second sources, limited memory size, and higher costs of these devices.

Instead of using bursting EPROMs, I have designed a two-way interleaved EPROM system which provides a burst-enabled memory system. This system can be expanded to create a four-way interleaved EPROM interface, which requires four banks of EPROMs. The more traditional linear pattern uses a single bank and does not allow for bursting. A third nonbursting alternative worthy of consideration is to use a single EPROM and an MC68 150 dynamic bus sizer to build the 32-bit word.

Bursting refers to a method used for memory accesses and also to a microprocessor's ability to make multiple accesses quickly. A burst begins with a read request to a single address that is referred to as the ***initial seed address.*** A memory system capable of bursting should provide a significant improvement in the access time of subsequent accesses after the initial seed has been accessed. Bursting may not be able to provide a significant improvement in performance if the information, data, or instructions are highly fragmented or if a zero-wait-state memory system is available. Bursting will have the greatest positive impact when all memory references are linear and then where the access penalty is large for the initial access and minimal for each subsequent burst access.

Typically, all data and instructions benefit from bursting. However, when data or instructions are highly fragmented, bursting can have a negative effect. For example, if a pointer is generated to read a single value from a memory location, then the other three accesses may never be used. In this situation, the unused cycles can stall the integer unit while the bus could have been used for other required memory accesses. When a zero-wait-state memory system is used, the M68040 integer unit will perform as well as, or better than, when the bursting function is used.

BURSTING THE M68040

The M68040 integer unit has a Harvard-style architecture that allows the six-stage pipeline and two integrated on-chip caches to operate independently. The M68040 processor has an integrated on-chip 4K-byte instruction cache and a 4K-byte data cache. Both caches are four-way-set-associative caches composed of four 32-bit words per cache line.

The relatively large instruction and data caches can each sustain approximately 90% hit rates. The hit rate varies depending on the instruction and data mixes. With a bursting

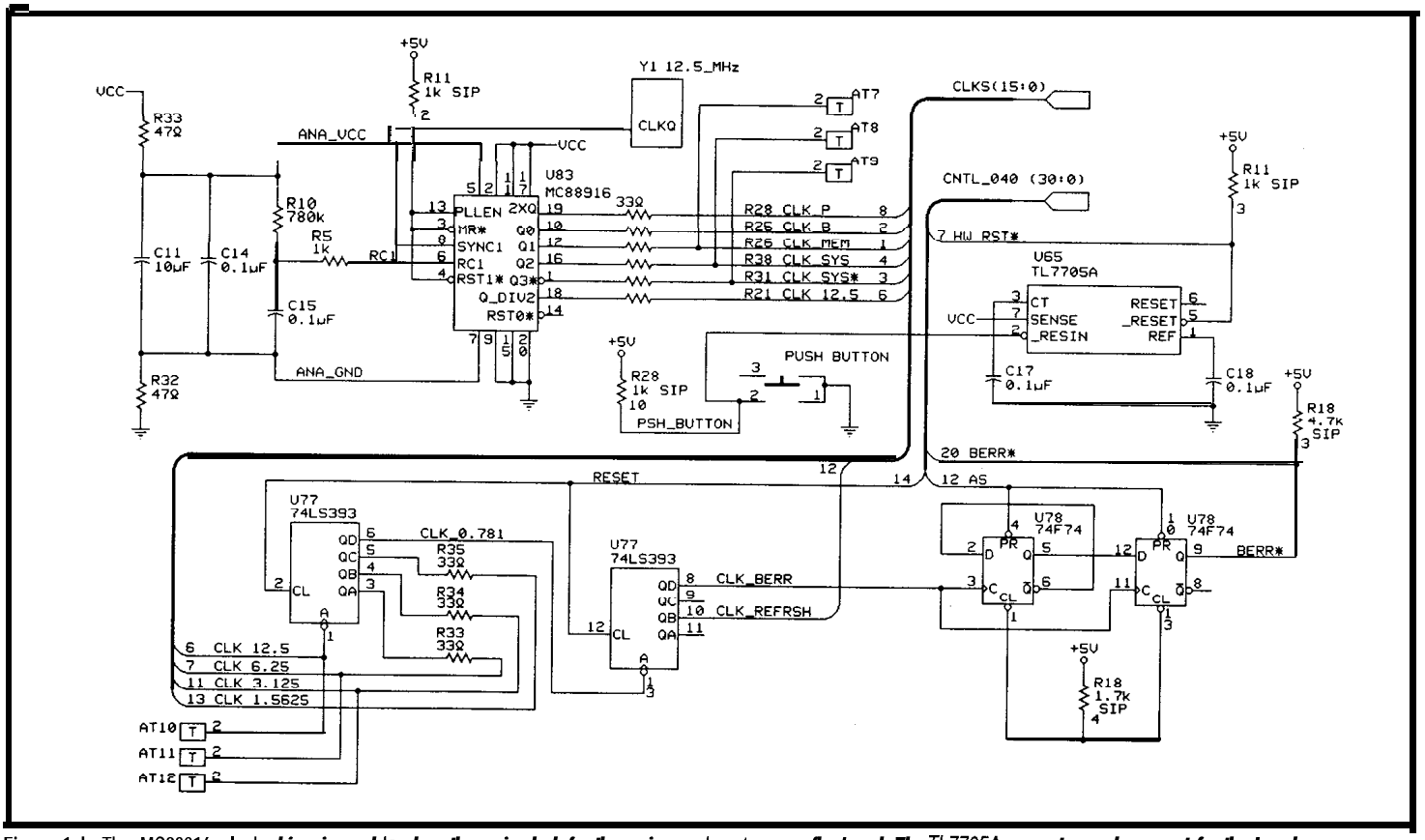


Figure 1 b–The MC88916 clock ***driver is used to skew the main clock for the various subsystems on the board. The TL7705A generates a clean reset for the board.***

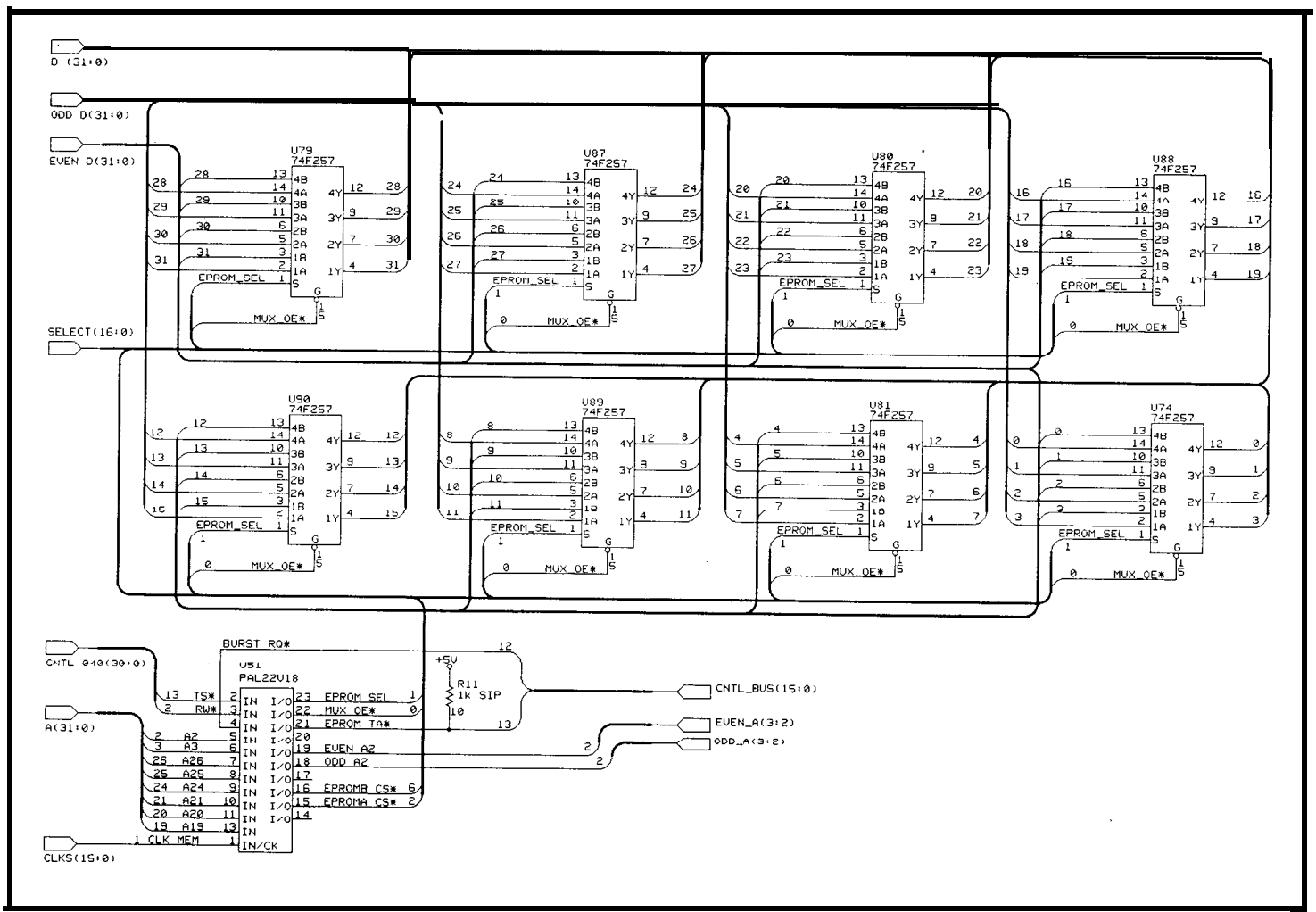


Figure 1 ***c-A 22V10 PAL implements both state machine and non-state machine signals. The state machine controls the 74F257 multiplexers to switch banks of EPROMs.***

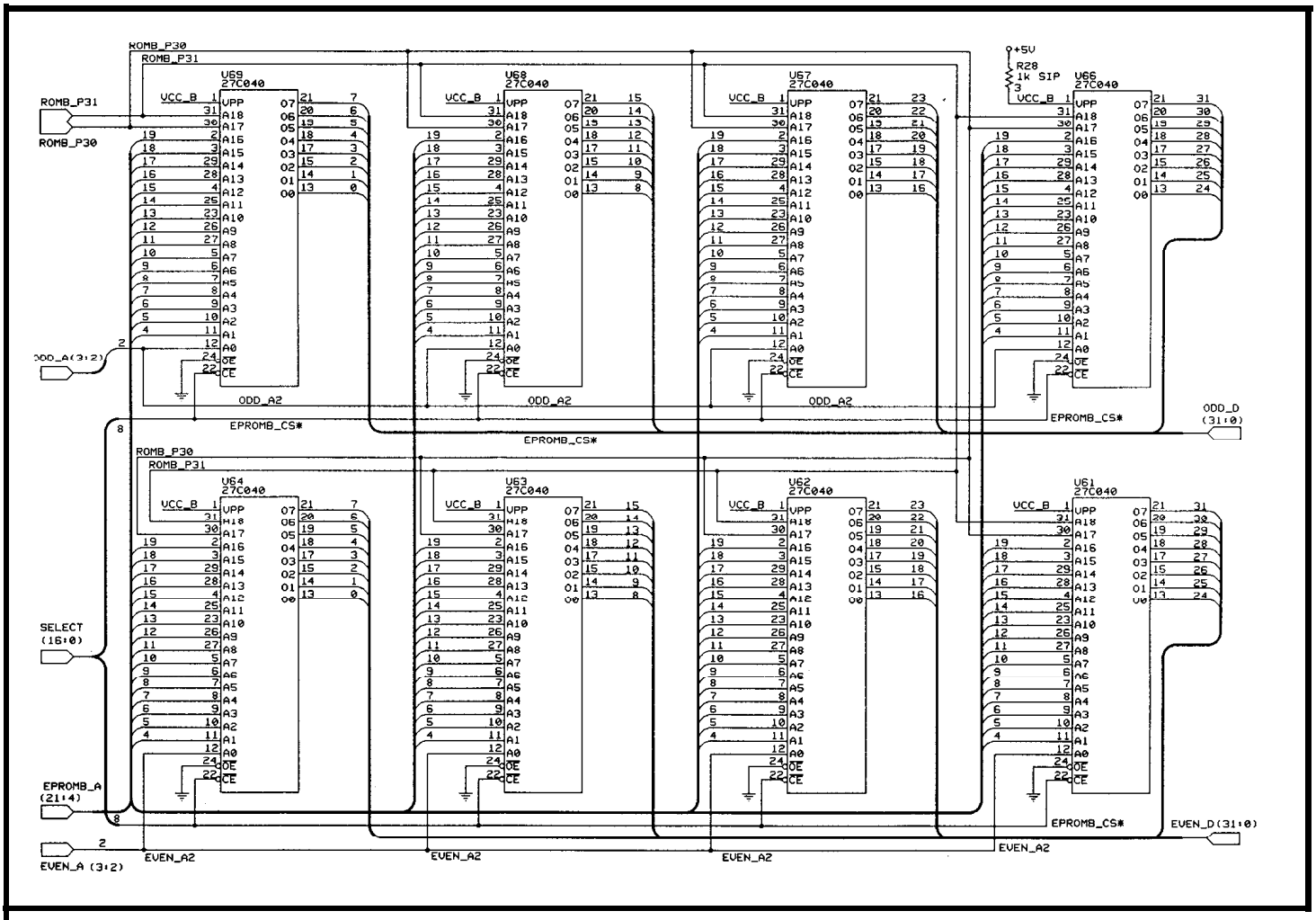


Figure 1—*Two banks of off-the-shelf 512K-byte EPROMs* ***form the core of the*** memory *section.*

EPROM interface, the penalty for an instruction or data cache miss is minimized. This is due to the significant reduction in total bus traffic. With the reduction, the instructions or data that are required to keep the M68040 busy can be fetched from memory in a minimal amount of time.

The M68040 integer unit will access four 32-bit words for each single burst sequence. The internal caches are organized to store four 32-bit words on a single cache line. The M68040 integer unit will always begin with the access that will contain the data or instruction for which the integer unit is waiting. A bursting sequence will always begin on a hex address ending with any of the following values: $0, $4, $8, or $C. Hardware is required to increment address bits A2 and A3 to obtain the additional three addresses. For example, if hex address $0000 1008 is the initial seed address, the processor will supply the seed address $0000 1008. External hardware will be required to generate and supply the "artificial addresses" $lOOC, $1000, and $1004.

The M68040 has a nonmultiplexed address and data bus, reducing the difficulty and improving the efficiency of the interleaved design. For example, a 25MHz bursting interface with a 5: 1:3: 1 (five-clock initial, one-clock second access, three-clock third access, and one-clock final access) profile can be done by using eight 120-ns EPROMs and eight multiplexers. The schematic in Figure 1 demonstrates how to implement this kind of memory system. No additional address latches are required.

Due to the relatively high-speed bus on the M68040 integer unit and the slow turn-off time of EPROMs, bus contention is highly possible. As a reminder, recall that bus contention occurs when two devices are attempting to drive the bus at the same time. Bus contention can produce three problems: increased system noise, an increase in the total system power, and system failure when the bus drivers blow out.

Memory Access	Total Number of Clocks	M68040 Performance
2:2:2:2 No Burst	8 clocks	100.0%
4:1:1:1	7 clocks	99.7%
4:1:2:1	8 clocks	97.9%
5:1:1:1	8 clocks	96.7%
4:2:2:2	10 clocks	93.4%
5:1:3:1	9 clocks	92.9%
5:2:2:2	11 clocks	90.1%
3:3:3:3 No Burst	12 clocks	89.8 %
5:3:3:3	14 clocks	83.4%
4:4:4:4 No Burst	16 clocks	80.2%

Table 1—*Awhole* ***range*** of ***bursting access*** arrangements is ***possible.*** Which you ***choose*** depends ***on your*** design.

PERFORMANCE CONSIDERATIONS

The 5:1:3:1 bursting access will yield about 93% of the maximum sustained performance available from the

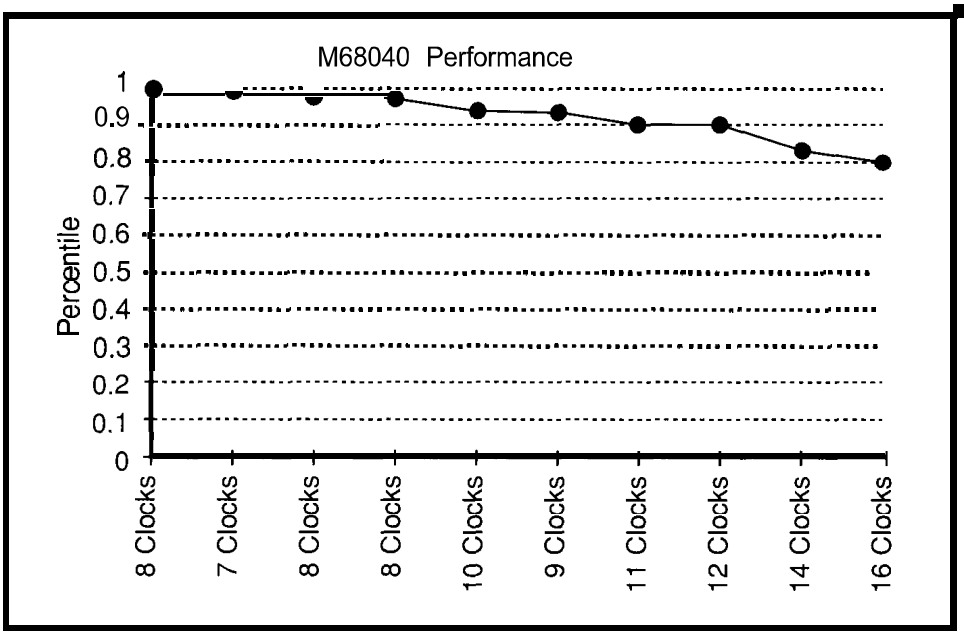


Figure 2-A 2222 *with* ***no burst yields the*** best performance ***while the performances of*** *other* ***ratios and numbers of clocks*** *fall* ***off*** *slightly.*

M68040 integer unit. See Table 1 for the memory interface derating chart. A graphic representation of the information contained in columns two and three of the table is shown in Figure 2.

Three points should be considered in making the entire burst access as efficient as possible. The most important part of the bursting sequence is the first access. Insertion of an additional clock cycle will typically incur a 3% degradation in performance.

The next most critical point is the second access. Often the M68040 integer unit will be starving for the first access. Once that has been accomplished, the M68040 integer unit will typically accept the second access into the first stages of the instruction pipe. When the second access is delayed by too many clocks, a *bubble* can occur in the instruction pipe. This will degrade the possible performance of the machine.

The third point to be considered is the total number of clocks for the complete burst transfer. The third and fourth transfers can degrade the possible performance by one or two percent for every additional clock cycle that is added. Typically, this is caused by stalls in data or nonsequential instruction fetches or data writes that have filled the write-back buffer.

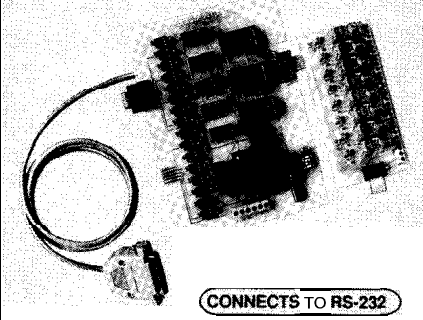
The M68040 integer unit is less susceptible to memory degradation than many other processors available today. This is due to three factors: the M68000 processors have a relatively high code density; M68000 code is typically half the size of a RISC processor; and a RISC processor is a load-

M86040 Address A3	A2	Even Bank A2	Odd Bank A2	Multiplexer Select
0	0	0	0	Even Bank
		1	1	Odd Bank
		0	0	Even Bank
		1	1	Odd Bank
0	1	1	0	Odd Bank
		0	1	Even Bank
		1	0	Odd Bank
		0	1	Even Bank
1	0	1	1	Odd Bank
		0	0	Even Bank
		1	1	Odd Bank
		0	0	Even Bank
1	1	0	1	Even Bank
		1	0	Odd Bank
		0	1	Even Bank
		1	0	Odd Bank

Table 2—*The* ***interleaved EPROM interface accesses two banks of memory: an even bank and an odd bank. Depending on address bit A3 from*** *the* ***processor, the interface*** *will* ***access the even or odd bank first, followed by the*** opposite ***bank.***

RELAY INTERFACE

PROVIDES SOFTWARE CONTROL OF RELAYS

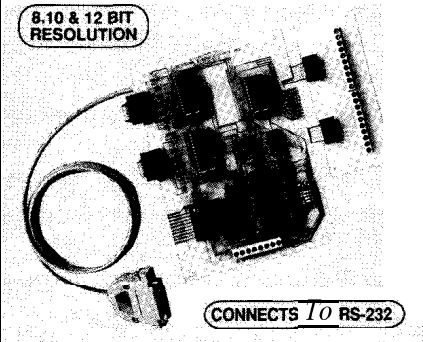


CONNECTS TO RS-232

AR-16 RELAY INTERFACE ..... $ 99.95
Two 8 channel relay output ports are provided for control of up to 16 relays (expandable to 128 relays using EX-16 expansion cards). Each relay output port connects to a relay card or terminal block. A variety of relay cards and relays are stocked. Call for more info. RS-422 available (distances to 4,000 feet). PS-4 port selector may be used to control satellite AR-1 6 interfaces. (up to 16,384 relays)
RD-8 REED RELAY CARD (8 relays, 10 VA).....$ 49.95
RH-8 RELAY CARD (10 amp SPDT 277 VAC).....$ 69.95
EX-16 RELAY EXPANSION CARD (16 channel)...$ 59.95

ANALOG TO DIGITAL

8,10 & 12 BIT RESOLUTION



CONNECTS To RS-232

ADC-16 A/D CONVERTER (16 channel, 8 bit)..$ 99.95
Input temperature, voltage, amperage, pressure,energy usage, energy demand, light levels, joystick movement and a wide variety of other types of analog signals. Inputs may be expanded to 32 analog or 128 status inputs using the AD-16 or ST-32 expansion cards. 112 relays may be controlled using EX-16 expansion cards. Analog inputs may be configured for temperature input using the TE-8 temperature input conversion. RS-422 available. PS-4 port selector may be used to connect satellite ADC-16 interfaces (up to 4,096 analog inputs/16,384 status inputs and 14,336 relays). Call for info on 10 & 12 bit converters. (terminal block and cable sold separately)
ST-32 STATUS EXPANSION CARD.....$ 79.96
Input on/off status of relays. switches, HVAC equipment, thermostats, security devices. smoke detectors and other devices including keypads and binary coded outputs. Provides 32 statue inputs (opto isolators sold separately).
TE% TEMPERATURE INPUT CONVERSION $49.66
Includes 8 temperature sensors 8 terminal block. Temperature range is minus 40 to 145 degrees F.
PS-4 PORT SELECTOR (4 channels RS-422) . . . ... $ 79.95
converts an RS-232 port into 4 selectable AS-422 ports.
TOUCH TONE DECODER and other serial interfacing products available. Call for free information packet.

- FULL TECHNICAL SUPPORT.. .Provided over the telephone by our staff. EACH ORDER INCLUDES A FREE DISK WITH PROGRAMMING EXAMPLES IN BASIC, C AND ASSEMBLY LANGUAGE. A detailed technical reference manual is also included.
- HIGH RELIABILITY...engineered for continuous 24 hour industrial applications. All ICs socketed.
- Use with IBM and compatibles, Tandy, Apple, Mac and most other computers with RS-232 or RS-422 ports. All standard baud rates and protocols may be used (50 to 19,200 baud).

Use our 800 number to order FREE INFORMATION PACKET. Technical Information (614) 464-4470.

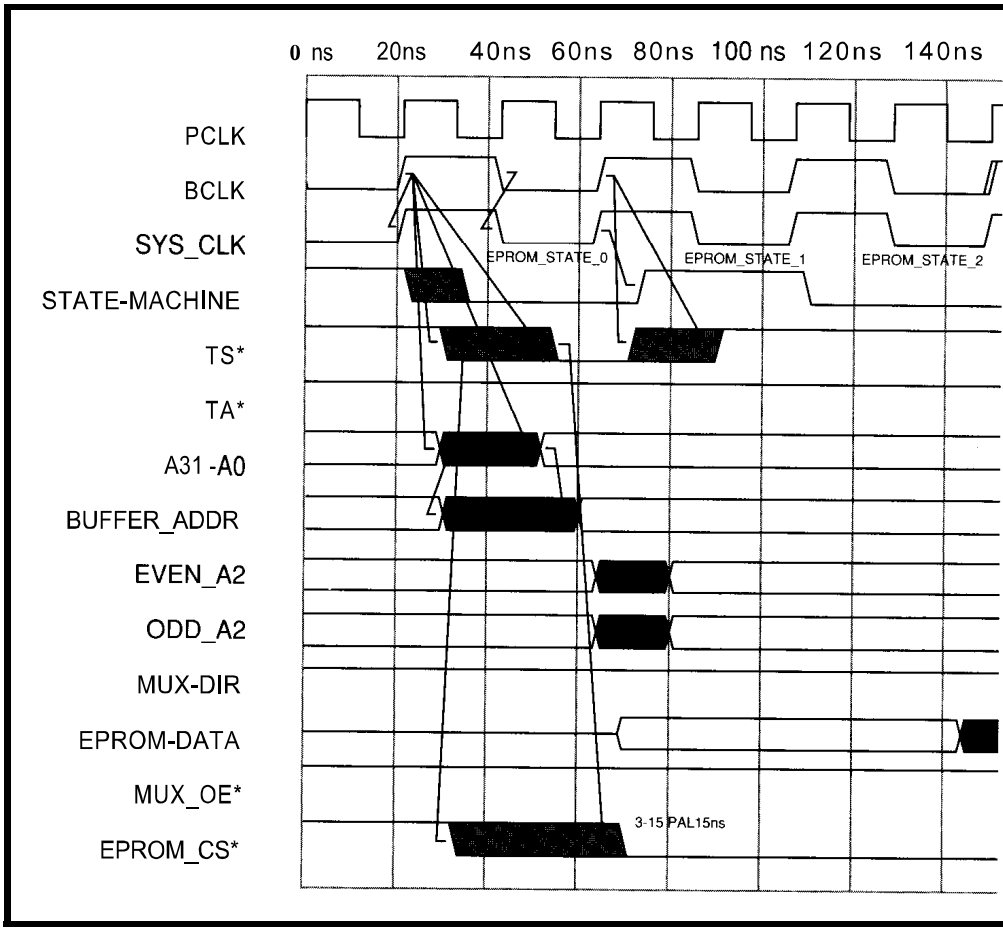
24 HOUR ORDER LINE (800) 842-7714
Visa-Mastercard-American Express-COD

International & Domestic FAX (614) 464-9656
Use for information, technical support & orders

ELECTRONIC ENERGY CONTROL, INC.
380 South Fifth Street, Suite 604
Columbus, Ohio 43215

#109

Figure **3-Precise timing of the interleaved EPROM interface is critical for successful operation. The state machine in the PAL handles all the tricky timing sequences.**



store based architecture, which does not have the ability to modify memory directly.

On a RISC processor, all instructions are a fixed length, resulting in wasted bits in the instruction fields. M68000 instructions can be 16 bits or larger, depending on the addressing mode or immediate data fields that are included in the instruction. This results in very dense instruction fetches and more efficient utilization of the instruction cache. The second factor is the relatively large instruction and data caches integrated on-chip. The third factor is the bursting interface coupled to the six-stage integer instruction pipe.

When a burst is initiated, a potential data write is delayed while the instruction(s) are fetched. Because data writes can be delayed, the M68040 can fetch instructions or data that are needed to keep the instruction pipeline full. The M68040 integer unit requests the required instruction first and bursts in the remaining three fetches. Typically, the M68040 integer unit will need the next instruction within one to four clocks after fetching the first instruction.

After bursting, the M68040 integer unit has four or more clock cycles when the integer unit will not require the data bus for instruction fetches. This allows the processor to update memory or fetch new data. The M68040 integer unit will use 50–90% of the total available bus bandwidth. The lower bus utilization allows other devices such as DRAM refresh, DMA, or slow peripherals to take advantage of the available bandwidth without impacting performance.

The interleaved EPROM interface accesses two banks of memory-an even bank and an odd bank. Depending on address bit A3 from the M68040 processor, the interleaved EPROM interface will access the even or odd bank first, followed by the opposite bank. See Table 2 for the incremental burst-sequence decode. The interleave control PAL decodes the state of the M68040 integer unit's address bits A3 and A2 to decode which bank of memory (the even or odd bank] should be selected for the M68040 integer unit's data bus. During a bursting sequence, both banks of memory are each accessed twice. The four 32-bit words fill a cache line. The interleaved control PAL increments the even A2 and the odd A2 signals to provide the two accesses with different addresses for each EPROM bank. The computation of the timing required for the interleaved EPROM memory system is greatly simplified by the synchronous bus on the M68040 integer unit.

TIMING CONSIDERATIONS

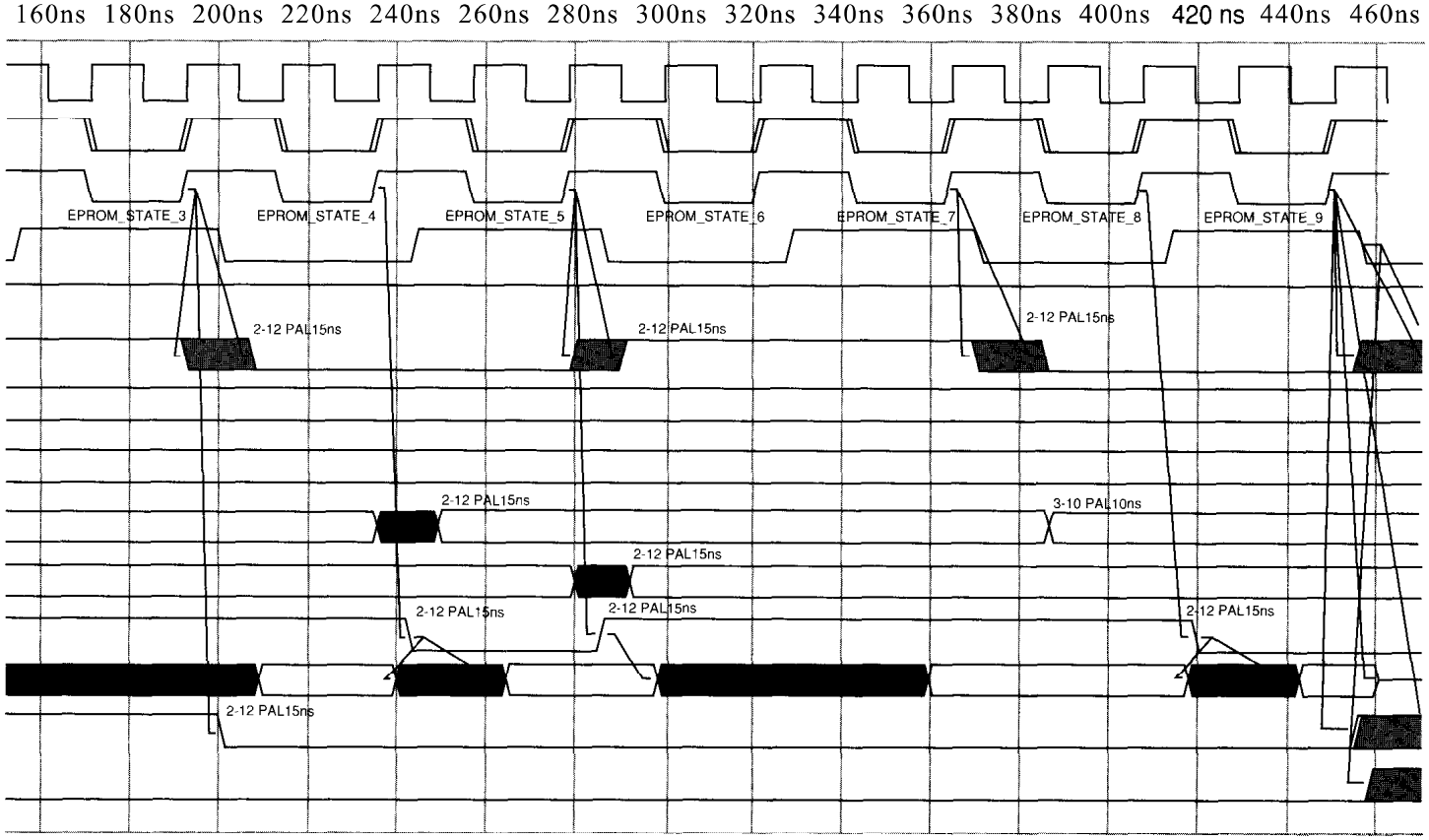
The first access begins at the rising edge of the BCLK. The signal TS* is asserted for 10 ns or more. This condition causes the state machine in the interleaved control PAL to go to state ES1. Assertion of the EVEN_A2 and ODD_A2 to the correct logic levels occurs between 2 ns and 15 ns after the rising edge of the BCLK when a 15 ns PAL is used.

$$t_{AVAIL} = (N - 1) \times t_1$$
$$t_{Rq} = t_{PALa} + t_{EPROMa} + t_{74F257P}$$
$$t_{VALID} = t_{AVAIL} - t_{Rq} - t_{SKEW} - t_{SETUP}$$

where:

t_{AVAIL} = total time available
t_{Rq} = time required by interface logic and memory
t_{VALID} = time available remaining
N = number of clock cycles
t, = time period of the bus clock
t_{PALa} = PAL logic access time to valid outputs (15 ns)
t_{EPROMa} = EPROM access time to data valid (120 ns)
$t_{74F257p}$ = maximum propagation delay for 74F257 (7 ns)
t_{SKEW} = maximum clock skew
t_{SETUP} = maximum setup time on M68040 (5 ns)

Computation of equation t_{AVAIL} is 160 ns when a 25-MHz M68040 integer unit with a five-clock initial access is used. Computation of equation t_{Rq} is 142 ns and t_{VALID} is 18 ns (specification 16 of the M68040). The hold time requires that data be held valid for a minimum of 4 ns after the rising edge of BCLK.



ECAL Universal Assembly Language Development System

- ▶ Alternative to Real-Time Emulator
- ▶ Support for 805 1.8096, and 186
- ▶ Support for 170+ additional processors
- ▶ User control of syntax and instructions
- ▶ Extremely fast assembly--? Kbytes/sec
- ▶ Integrated split-screen editor or command-line assembly supported
- ▶ Integrated linker/loader
- ▶ Instruction trace and I/O windows
- ▶ Monitor and RS-232 corn. windows
- ▶ Single micro processor versions available
- ▶ Optional EPROM emulator and programmer
- ▶ Source-level debugger

Product Information

ECAL is a complete assembly-language development system that provides all the tools needed to assemble, lmk, load. run, and debug your project for over 170 processors. By using user-editable control files, the ECAL macroassembler in its full configuration can handle 4-,8-,16-,or 32-bit microprocessors with unsurpassed speed and consistency.

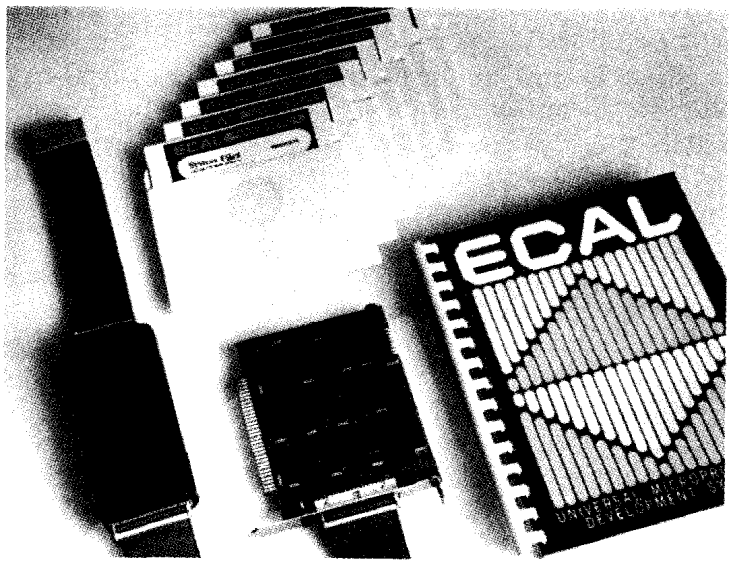
Using the familiar DOS-bated text windows. you can edit, assemble. set breakpoint,. trace execution, watch registers and I/O, and communicate with your target's serial port in separate closable windows. If you prefer to use other tools, with a few keystrokes. ECAL will incorporate your previous work into its consistent and intuitive environment.

The free ECAL evaluation program features all of the ECAL tools for all ot the supported micros. giving you a true sampling of ECAL development cycle (source and object length limited).

VAIL Silicon Tools sells and supports ECAL and can bundle ECAL with additional hardware and software to satisfy your need for economical project development tools.

Ordering Information

- 05-0200-010 ECAL OAS
- 05-0200-020 ECAL with EPROM Emulator
- 05-0200-XXX ECAL Single Processor



Contact

Vail Silicon Tools
692-A S. Military Trail
Deerfield Beach, FL 33442
Tel: (305) 570-5580
Fax: (305) 428-1811

Vail
Silicon Tools

#110

$$t_{HOLD} = t_{74F257h} + t_{EPROMh} + t_{PALh} - t_{SKEW}$$

where:

t_{PALh} = PAL logic minimum propagation time (2 ns)

t_{EPROMh} = EPROM hold time (0.0 ns)

$t_{74F257h}$ = minimum propagation delay for 74F257 (2 ns)

Calculation of equation t_{HOLD} is 0.0 ns for all four accesses. Refer to the hold time timing diagram, shown in Figure 3 for additional information. The 0.0 ns of margin on the hold time does not cause a problem when the following considerations are taken into account.

First, the EPROM manufacturers specify 0.0 ns hold time from chip select negation to data invalid. This is not very realistic and some EPROM vendors are beginning to specify 4 ns (or more) hold time. Second, the clock driver should be placed near the M68040 integer unit to reduce transmission line effects and EMI noise. This results in the BCLK trace to the M68040 integer unit being significantly shorter than the trace to the interleaved control PAL which adds to the total data hold time.

The clock driver chosen can make a significant difference in the clock skew time. When using a MC889 15 or MC88916 clock driver, the output chosen can further reduce the skew between the BCLK and the clock pin of the interleaved control PAL. For example, the MC88915 clock driver's Q4 clocks 34 picoseconds or more (up to 275 ps) before QO. When the Q4 output from the MC88915 is used to drive the M68040 BCLK pin and the QO is used to drive the interleaved control PAL, the BCLK will always lead the PAL logic output. When using the MC88915 to drive the system clock, the clock skew becomes 0.034 ns positive. If the M68040 integer unit cannot be used with the MC88915 or MC88916 in this configuration, a delay can be introduced into the interleaved control PAL clock. A typical delay device might be a MC74F08, where the low-to-high specification is 3.0 ns minimum and 6.6 ns maximum. The interleaved control PAL clock should not exceed an 8.7-ns delay because the

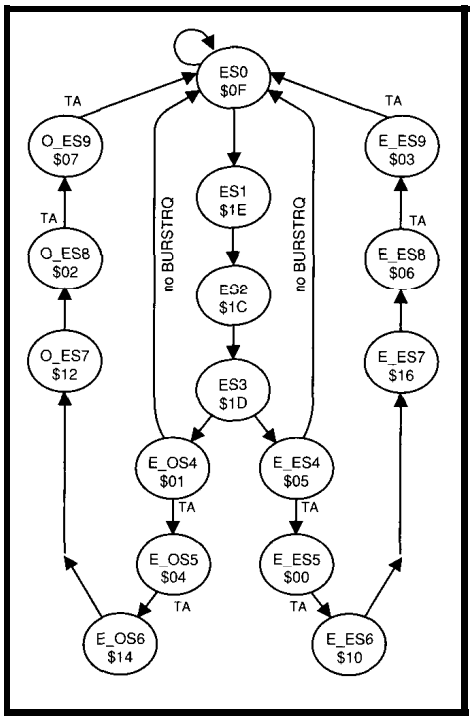


Figure 4—*Interleaved* ***EPROM*** *State* Machine ***Diagram*** *for the M68D040/LC040/EC040*

interleaved control PAL could miss the TS . signal from the M68040 integer unit.

Finally, the trace length and capacitance of the control signals and data bus need to be at a minimum to require any calculations to be performed on the trace or wire delay. Furthermore, the trace and device capacitance contribute to the total hold time. The hold time of 0.0 ns should not be a problem when a high-resolution clock driver is used. The maximum clock skew should never exceed 1 ns or BCLK will lead the interleaved control PAL clock.

The second access has one additional clock period to access the EPROM. The second access setup time (t_{VAIL}) is 240 ns and the t_{VALID} is 63 ns. An additional quick check should be performed on t_{HOLD} when the multiplexer is changed from one input to the other. The minimum multiplexer select time is 3 ns. The interleaved control PAL clock-to-state change is 2 ns. When using the MC88915 or MC88915 clock driver described above, the clock skew is negligible and the t_{HOLD} time is 1 ns.

The third access has timing that is similar to the first and second accesses. The t_{Rq} for the third access is 142 ns, so t_{VALID} is 18 ns. A similar argument can be made for the fourth and final access.

THE PROCESSOR'S PAL

The PAL equations to control the interleaved EPROM interface fit into a single 22VIO device. The code used to program this device is shown in Listing 1. The inputs to the PAL are a 25-MHz system clock, TS*, R/W*, and address signals from the M68040 processor. The last input is a combination of several signals to create the BURSTRQ' signal. The BURSTRQ . signal is asserted low when the M68040 processor attempts to burst, indicated by the assertion of SIZE1 and SIZE0 signals to a high logic level. BURSTRQ* can be derived inside the 22V10 with an equation, or a 74F00 can be used to conserve input pins.

The outputs of the interleaved EPROM PAL are in two groups. The first group is the non-state machine outputs and the second group contains the state machine outputs. The signals in the non-state machine group are the EPROM chip select (EPROM_CS*) and the even and odd address bit 2 (EVEN_A2 and ODD_A2). The signal EPROM_CS * will assert when the address is valid and the TS* signal is asserted. The EPROM_CS* signal remains asserted until the state machine returns to the ES0 or the starting state. The chip select signal will assert only during a read operation. A write causes the EPROM_CS' signal to negate and the state machine to go to ESO.

The EVEN_A2 and ODD_A2 signals begin the toggling process based on the original value of the A3 and A2 address bits from the M68040 processor. For this discussion, an address will be considered "even" when the address from the M68040 processor decodes to the even bank first. An "odd" address will decode to the odd bank first. All first accesses beginning with a $0 or $C will begin with the even bank. All first accesses beginning with a $4 or $8 will begin in the odd bank of EPROM. For example, the address $0000 0004 causes the first access to occur in the odd bank. If the first access were memory location $0000 OOOC, the access would be even.

Listing 1--The *22V10 PAL handles both state machine and non-state machine signals. It contains the main control logic for the interface.*

```
MODULE  INTERLEAVED-EPROM:

flag '-r3'

TITLE ' Interleaved EPROM PAL Eq.
        Ron Stence      Motorola Inc.'

inter device 'p22v10';

"-----------------"
"DEFINE INPUT PINS "


clk         pin 1;              "25MHz system clock
~ts         pin 2:              "68EC040 transfer start
read        pin 3;              "68EC040 read-write
~burstrq    pin 4;              "68EC040 wants to burst
a2,a3       pin 5,6;            "address bus
Address     pin 7,8,9,10,11; "address bus

"-- ---------------"
"DEFINE OUTPUT PINS "


~eprom_cs   pin 16;             "EPROM Chip Select
e_a2        pin 19;             "even EPROM A2
odd_a2      pin 18;             "odd EPROM A2
q1,q0       pin 17,20;          "state machine outputs
~eprom_ta   pin 21:             "EPROM Transfer Acknowledge
~mux_oe     pin 22;             "EPROM data MUX output enable
mux_sel     pin 23;             "mux sel=1:Even EPROM
                                "mux sel=0: Odd EPROM


"DEFINE CONSTANTS "
"-----------------"

X, C, Z, K  =. X. , . C. , . Z. , . K. :

"-----------------------"
"DEFINE INTERNAL NODES  "


RESET node 25:  "A Write to EPROM will Reset this state machine!


"DEFINE VECTORS "


eprom_s = [~eprom_ta,~mux_oe,mux_sel,q1,q0];
 ES0    = [1,  1,   1,   1,   1];" ^h1F
 ES1    = [ 1,  1,   1,   1,   0];" ^h1E
 ES2    = [ 1,  1.   1,   0,   0];" ^h1C
 ES3    = [ 1,  1,   1,   0,   1];" ^h1D
 E_ES4  =    [ 0, 0, 1, 0,  1];" ^h05, assert TA to even EPROM
 E_ES5  =  [0,0,0,0,     01:" ^h00, assert TA to odd EPROM
 E_ES6  = [ 1,  0,  0,   0,   0];" ^h10
 E_ES7  = [ 1,  0,  1,   1,  01;" ^h16
 E_ES8 = [  0,  0,  1,    1,  01;" ^h06, assert TA to even EPROM
 E_ES9  =  [0,0,0,   1,    1];" ^h03, assert TA to odd EPROM

 E_OS4  =    [ 0, 0, 0, 0,  1];" ^h01, assert TA to odd EPROM
 E_OS5  = [0,0,  1,   0,   01;" ^h04, assert TA to even EPROM
 E_OS6  = [ 1,  0,  1,   0,  01;" ^h14
 E_OS7  = [ 1,  0,  0,   1,  01;" ^h12
 E_OS8 = [  0,  0,  0,    1,  01;" ^h02, assert TA to odd EPROM
```

(continued)

The PAL equation for EVEN_A2 begins with the signal asserted to a low logic level when the address is "even." The basis of the PAL equation for the signal EVEN_A2 is an exclusive-OR. The signal EVEN_A2 is used for the first and third accesses when the address is "even." Therefore, EVEN_A2 will be toggled immediately following the first access to allow the even EPROM bank to begin the next access. Once the third access has been terminated, the signal EVEN-A2 becomes a "don't care." When an "odd" access is initiated, the signal EVEN_A2 will begin with a high logic level and the toggling will begin at the end of the second access.

The signal ODD_A2 follows the initial value of the signal A3 from the M68040 processor for both "even" and "odd" addresses. When the address is "odd," the first access will be to the odd bank. At the end of the first and third accesses, ODD_A2 will toggle. When the address is "even," the first access will be to the even bank and the signal ODD_A2 will toggle at the end of the second access.

The EPROM state machine is composed of five bits. Two bits were used to create unique or distinctive state machine outputs (Q1 and QO) so a Grey code type of count could be used. However, not all state changes are true Grey code. This does not cause a problem since all signals using the state machine outputs are registered. The next bit is the multiplexer select bit (MUX_SEL). When this bit is a logic high, the multiplexer selects the even bank. When the MUX_SEL bit is a logic low, the multiplexer selects the odd bank. The fourth bit is the multiplexer output enable bit (MUX_OE*). When this bit is asserted to a logic low, the multiplexer drives the data bus of the M68040 processor. When the MUX_OE* bit is negated to a logic high, the multiplexer becomes a high-impedance output. The final output of the EPROM state machine is the acknowledge signal to the M68040 processor (EPROM_TA*).

The output pin EPROM_TA' uses an open-collector-type output. This will reduce the number of inputs to the acknowledge PAL equation and

allows other devices to be connected together to drive EPROM_TA* to a low logic level. Other signals that can be connected to the EPROM_TA* signal net are the acknowledge from the interrupt PAL or from other peripherals PALs. A pull-up resistor is required when this configuration is used.

STATE MACHINE CAVEATS

The state machine (refer to Figure 4) can be modified easily to increase or decrease the number of wait states. To increase the number of wait states, an additional state should be added before states ES3, E_ES7, and E_OS7. To decrease the number of wait states, remove states ES3, E_ES7, and E_OS7.

When changing the state machine, two points must be considered. The Grey code may need modification to minimize state changes. The Q1 and QO outputs can be modified to minimize the number of state changes without impacting the operation of the PAL equation. Second, the state machine is running ahead of the M68040 processor. For example, the EPROM_TA' is asserted during state E_ES4. However, the M68040 processor will actually accept the data and TA* signal on the rising edge of the BCLK. The state machine will exit the E-ES4 state at the same time. Therefore, the modification to the state machine should take place when the state machine is going through wait states.

USING THIS NEW TOOL

Several types of applications are well suited to an interleaved-EPROM-based design, such as laptop computers, low-power computers, embedded controllers, and laser printers. A laser-printer-based application can benefit greatly from an interleaved EPROM interface due to the cost sensitivity of the laser printer market while being driven to provide higher performance for the same system cost. Laser printers typically will not copy instructions and fonts into DRAM. The interleaved EPROM interface will provide DRAM equivalent performance without a significant increase in the cost of the system.

Listing 1-continued

```
E_OS9  = [ 0,  0,  1,  1,  11;" ^h07, assert TA to even EPROM

addr = [Address];

state-diagram eprom_s

state  ES0:
            if (!~ts & (addr == Address)) then ES1;
            else ES0:
state ES1:
            goto ES2;
state ES2:
            goto E3;
state  ES3:
            if (!a2) then E_ES4;
            else E_OS4;
state E_ES4:                    "Assert TA, MUX sel = 1
            if (!~burstrq)  then E_ES5
            else ES0;           "no burst
state E_ES5:                    "Assert TA, MUX sel = 0
            goto E_ES6;
state E_ES6:
            goto E_ES7;
state E_ES7:                     "Assert TA, MUX sel = 1
            goto E_ES8;
state E_ES8:                     "Assert TA, MUX sel = 0
            goto E_ES9;
state E_ES9:
            goto ES0;

state E_OS4:                     "Assert TA, MUX sel = 0
            if (!~burstrq)  then E_OS5
            else ES0:           "no burst
state E_OS5:                    "Assert TA, MUX sel = 1
            got0 E_OS6;
state E_OS6:
            goto E_OS7
state E_OS7:                    "Assert TA, MUX sel = 0
            goto E_OS8;
state E_OS8:                    "Assert TA, MUX sel = 1
            goto E_OS9;
state E_OS9:
            goto ES0;


"PAL EQUATIONS"
"--------------"

equations

RESET = !read;

e_a2 := ( ((a3 $ a2)
  & (!~ts # (eprom_s == ES0) # (eprom_s == ES1)
         # (eprom_s == ES2) # (eprom_s == ES3))
         # (e_a2 & ((eprom_s == E_ES6) # (eprom_s == E_ES7)
         # (eprom_s == E_ES8) # (eprom_s == E_OS6)
         # (eprom_s == E_OS7) # (eprom_s == E_OS8)
         # (eprom_s == E_OS9) )) # (!a3 & ((eprom_s == E_ES4)
         # (eprom_s == E_OS4))) # ((a3 !$ a2) & ((eprom_s == E_ES5)
         # (eprom_s == E_OS5))) );

odd_a2 := ( ((a3 $ a2) & ((eprom_s == E_ES4)
           # (eprom_s == E_OS4)))
           # (a3 & (!~ts # (eprom_s == ES0) # (eprom_s == ES1)
           # (eprom_s == ES2) # (eprom_s == ES3)))
           # (!a3 & ((eprom_s == E_ES5) # (eprom_s == E_ES6)
```

(continued)

Listing 1 - continued

```
            # (eprom_s == E_ES7) # (eprom_s == E_ES8)
            # (eprom_s == E_ES9) # (eprom_s == E_OS5)
            # (eprom_s == E_OS6) # (eprom_s == E_OS7)
            # (eprom_s == E_OS8) # (eprom_s == E_OS9))))

!~eprom_cs = read & ((!~ts & (addr == Address) )
            # ((eprom_s != ES0) & (addr == Address)));
enable ~eprom_ta = (!~eprom_ta);

end INTERLEAVED-EPROM;
```

The interleaved EPROM interface has several positive points to be considered. The most significant is a dramatic increase in system performance while reducing bus utilization and memory latency on the M68040 integer unit. Second, the total system power can be reduced when the EPROMs do not have to be copied to a higher-speed memory system. When the EPROMs are not copied, power and money can be saved while not supporting a second memory system. Third, there will be a reduction of bus contention between the EPROMs and the M68040 integer unit. Finally, when data bus buffers are required, the change to multiplexers will have little or no effect to the cost of the system.

An interleaved EPROM interface does have several negative attributes which should be considered before using it in a design. First, there is an increased number of EPROMs included in the design. This technique should not be considered when the design will be using eight or more EPROMs. Second, there is an increased number of support devices such as multiplexers. Third, the increased complexity in the control logic could result in an increase in the board size. Finally, the potential increase in total system cost should be considered.

The change to an interleaved EPROM interface can be done with a small increase in complexity and system logic. The positive and negative attributes, such as total system size, cost, and performance levels, should be considered before changing to an interleaved interface. A requirement to make a dramatic improvement in the system performance could make an interleaved-EPROM-based design a wise solution.

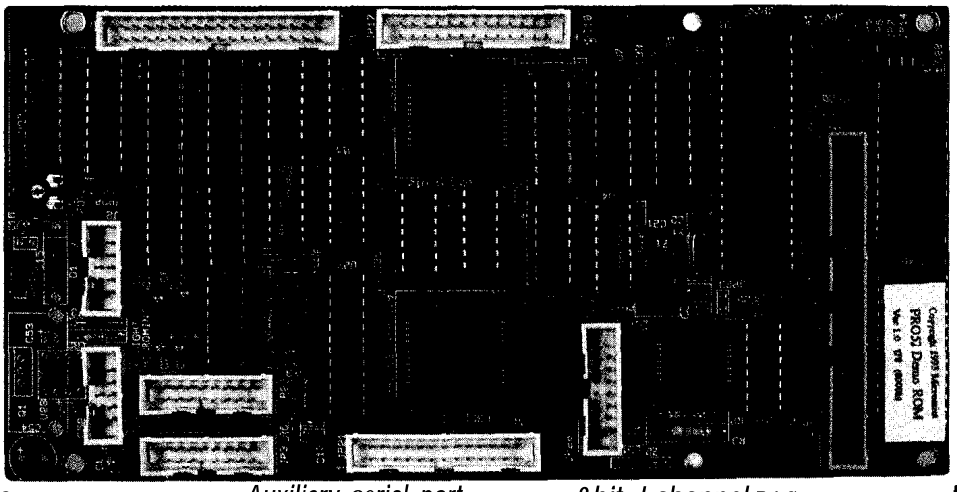
Ron Stence holds a B.S. in Computer Science from the Department of Engineering at Texas A&M University. He is currently a senior systems applications designer in Motorola's 68000 marketing applications group.

IRS

401 Very Useful
402 Moderately Useful
403 Not Useful

PRO = Positively Rampant Optimization

4 x 5 keypad input
4 x 20 or 8 x 40 LCD display output
Hardware real-time clock
24-bit parallel TTL I/O
CMOS PROCESSORS
• 80C31/Processor or
• 80C52 with BASIC52
8/32K RAM



PRO31 prices start a5 low as $289 ($219/100), includes SK RAM, watchdog, 56 bits I/O, S-bit ADC, and LCD interface. And it only costs $10 more for PRO52!

8/32K ROM

Power inputs
- Single supply or multisupply option
Console serial port
Auxiliary serial port
24-bit parallel TTL I/O
Hardware watchdog
&bit, I-channel DAC
8- or10-bit,8-channel A DC
EXPANSION HEADER
• Port 1 processor lines: 8-bit 77L
• Six decoded I/O address strobes
• Eight address lines

MICRO MINT PRO31/52 MICROMINT, INC.

4 Park Street • Vernon, CT 06066 • (203) 871-6170 • Fax (203) 872-2204
in Europe: (44) 0285-658122 . in Canada: (514) 336-9426 . in Australia: (02) 888-6401 • Distributor Inquiries Welcome!