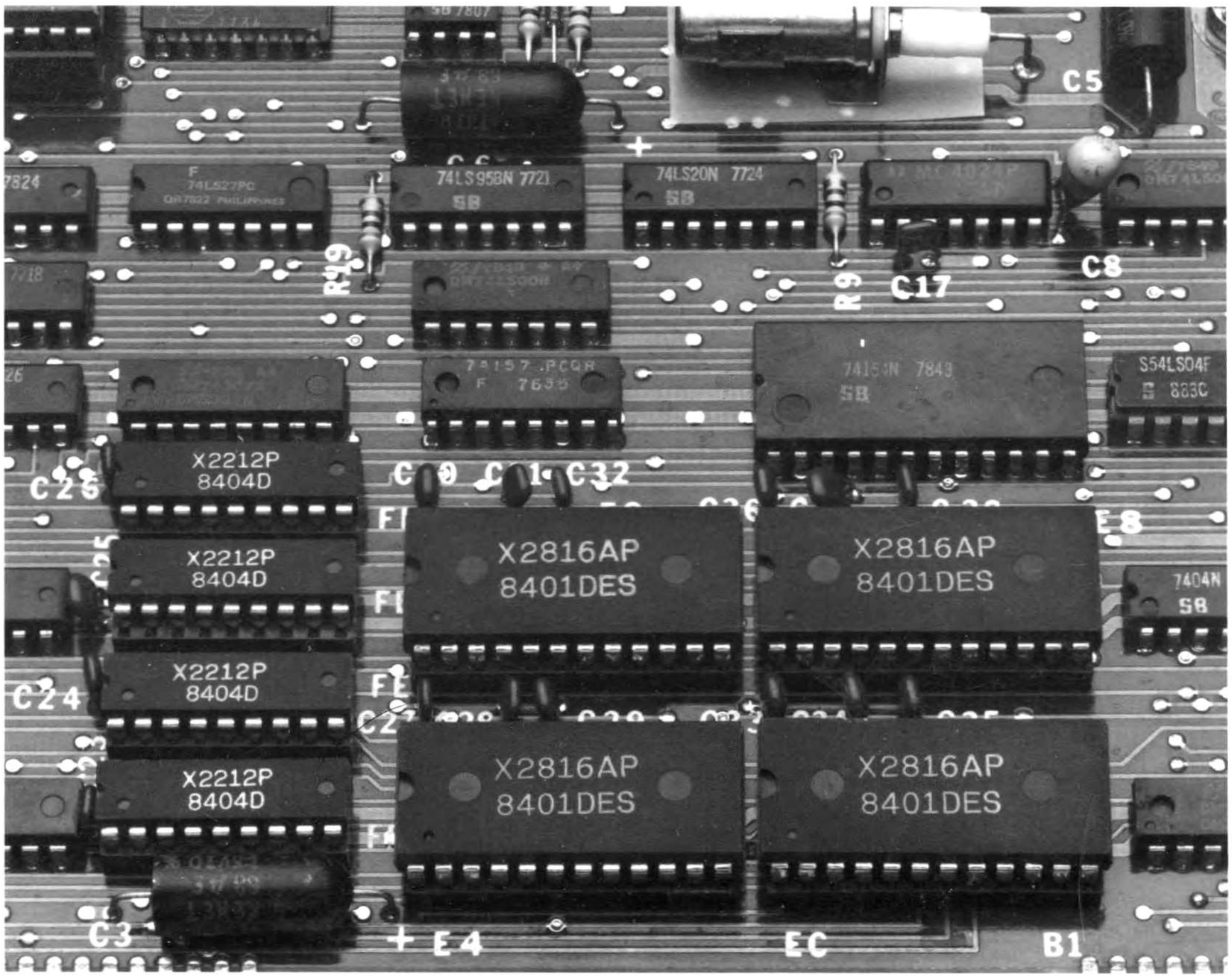
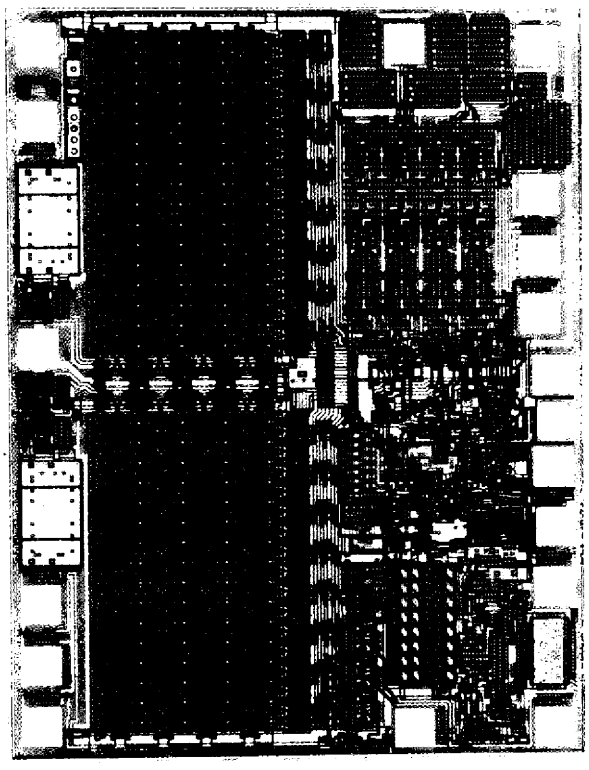
AB 301



APPLICATIONS BOOK

TABLE OF CONTENTS

For data sheet specifications on Xicor products see Data Book (DB201).



XICOR NOVRAM FAMILY, EASIER THAN EVER TO USE

BY GEORGE LANDERS

Introduction

The purpose of this application note is to take the user through the internal operation of NOVRAMs as well as their external operation. These devices are finding their way into many diverse applications due to their ease of use. The major features of the XICOR family of NOVRAMs are 1) only 5 volts is required for all operations including programming, 2) only TTL signals are required and 3) all pulse widths are short (< 450 ns).

Basically a NOVRAM is a memory device that has a static RAM overlaid with an EEPROM (Electrically Erasable Programmable ROM). The operation of the RAM is identical with other popular static RAMs such as the 2102A and the 2114. Figure 1 shows the block diagram of the XICOR NOVRAM family. NOVRAMs have $\overline{CS}$ and $\overline{WE}$ pins in common with their standard static RAM cousins but also have two additional control pins: $\overline{STORE}$ and $\overline{RECALL}$. The $\overline{STORE}$ and $\overline{RECALL}$ pins control movement of data between the RAM and the EEPROM.

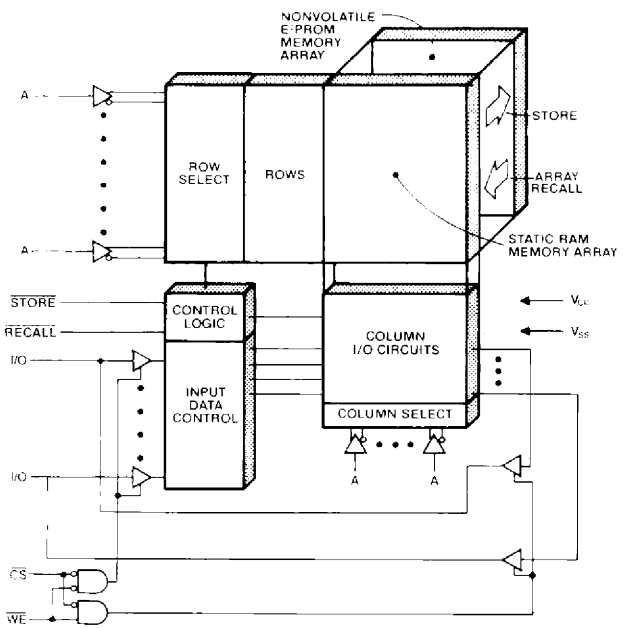


Figure 1) Block diagram of the XICOR NOVRAM family

The $\overline{STORE}$ pin is used to transfer the contents of the RAM to the EEPROM in a single operation. The entire contents of the RAM are transferred with one STORE operation. After a STORE operation is completed the original data is still in the RAM as well as the EEPROM.

The $\overline{RECALL}$ pin is used to transfer the contents of the EEPROM back to the RAM. When this is done, whatever data were in the RAM prior to the RECALL operation are totally replaced by the contents of the EEPROM. The STORE and RECALL operations function on the entire contents of the memory and not on a word by word basis. After either operation the contents of both the RAM and the EEPROM will be the same.

This may seem too good to be true; however, with XICOR's family of NOVRAMs the life of the systems designer is made even easier. Only a single five volt power supply is required for all operations including the STORE and RECALL operations. All addresses, data lines and control pins are TTL compatible and all pulse widths are short enough that most microprocessors do not require wait states. There are no high voltages or long pulse widths required which will inhibit the designer from designing a system with clean operation.

Technology

The XICOR NOVRAM stores its nonvolatile data during periods of power off by the absence or presence of charges on floating polysilicon gates. This is the same structure that is widely used in UV-EPROMS. The floating gate is an island of polysilicon surrounded by oxides with thicknesses of about 800 A°. Charge can be injected or removed from the floating gate by applying electric fields of sufficient strength to cause electron tunneling through the oxides. At normal field strengths the charges are permanently trapped on the floating gate even when power is removed.

The XICOR family of NOVRAMs uses three layers of polysilicon; the second layer is the floating gate. This structure employs a phenomenon known as Fowler-Nordheim tunneling. This form of tunneling is described in Vol. 40 No. 1 (Jan. 1969, pg. 278) of the Journal of Applied Physics and Vol. 27 No. 9 (Nov. 1975, pg. 505) of the Applied Physics Letters. In XICOR NOVRAMs this tunneling is enhanced by the use of textured polysilicon surfaces to generate higher field strengths at the surface to enhance electron injection into the oxide. The alternative to field enhancement by textured surfaces is to use ultrathin oxide layers in order to conduct the charge. The use by XICOR of standard oxide thickness gives XICOR a very manufacturable product, thus ensuring its low cost and volume delivery.

Figure 2. shows the circuit of the NOVRAM cell containing a conventional 6 transistor static RAM cell and a floating gate EEPROM cell with 2 additional transistors to control the action of data transfer. The floating gate (POLY 2) is connected to the rest of the circuit only through capacitance. Electrons are moved to the floating gate by tunneling from POLY 1 to POLY 2 and removed by tunneling from POLY 2 to POLY 3.

The capacitance ratios are the key to the operation of the transfer of data from RAM to EEPROM. If node

N1 is LOW, transistor Q7 is turned OFF. This allows the junction between CC2 and CC3 to float. Since the combined capacitance of CC2 and CC3 are larger than CP the floating gate follows the Internal Store Voltage node. When the voltage on the floating gate is high enough electrons are tunneled from POLY 1 to POLY 2 and the floating gate is negatively charged.

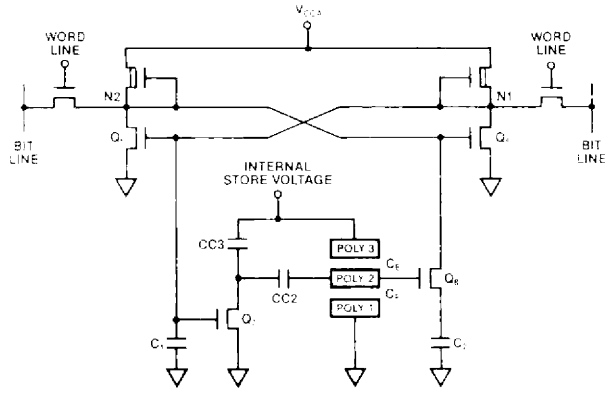


Figure 2) Circuit diagram of a NOVRAM cell.

If node N1 is HIGH, transistor Q7 is turned ON which grounds the junction between CC2 and CC3. Since CC2 is larger than CE, CC2 holds the floating gate near ground when the Internal Store Voltage node is pulled HIGH. This creates a sufficient field between POLY 2 and POLY 3 to tunnel electrons away from the floating gate leaving it with a positive charge.

The RECALL operation also takes advantage of capacitance ratios. The value of C2 in Figure 2 is larger than that of C1. When the external RECALL command is received, the internal power supply, VCCA, is first pulled LOW to equalize the voltage on N1 and N2. When the internal power node is allowed to rise, the node which has the lighter loading will rise more rapidly and the gain of the flip-flop will cause it to latch HIGH and the opposite node to latch LOW. If the floating gate has a positive charge C2, is connected to N2 through Q8 and N2 will latch LOW. If the floating gate has a negative charge Q8, is turned OFF and N1 will have the heavier loading.

The Xicor NOVRAM Family

XICOR's family of NOVRAMs contains three members with identical operating characteristics. The three parts offer the designer a choice of memory organization. The X2201A, X2210 andX2212 are organized 1024 x 1, 64 x 4 and 256 x 4 respectively. All three devices are packaged in 18 pin DIPs with 300 mil centers. Figure 3 shows the pin configuration of the three different NOVRAMs.

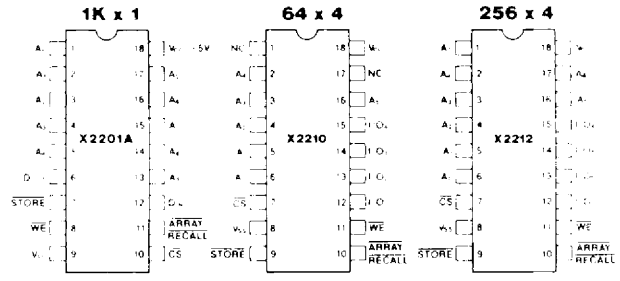


Figure 3) Pin configuration of the XICOR NOVRAM family.

The only functional difference between the three devices is that the X2201A has separate Data I/O lines while the two 4 bit wide parts have common I/O. Additionally, the X2210 and the X2212 are pin compatible. The two unused pins on the X2210 are used for the two higher order addresses on the X2212. The control pins $\overline{STORE}$, $\overline{RECALL}$, $\overline{CS}$ and $\overline{WE}$ operate identically on all three parts.

Write Operation

The WRITE operation is initiated by applying valid addresses followed by both $\overline{CS}$ and $\overline{WE}$ going LOW. On the 300ns version access time version, at least one of these two signals must remain HIGH until the addresses are valid. $\overline{CS}$ and $\overline{WE}$ must remain LOW simultaneously for 100ns.

An easy way to look at this is to consider the internal write command as the simultaneous LOW of $\overline{CS}$ and $\overline{WE}$. The internal write command is started by the last edge down and terminated by the first edge up. Valid addresses must overlap this internal write command. Data must be referenced to the first positive edge of $\overline{CS}$ or $\overline{WE}$. The timing required for writing to the RAM is shown in Figure 4.

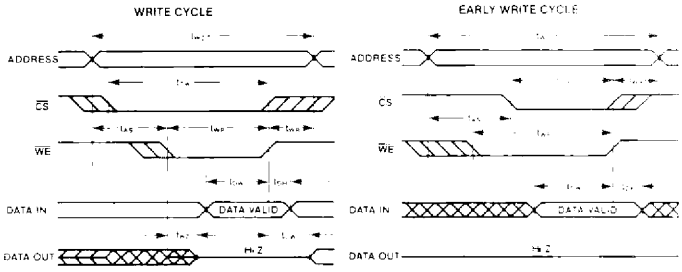


Figure 4) Timing diagram for Writing to the RAM.

Read Operation

The READ operation is the easiest of the four operations performed by the NOVRAM. In the case of the 300ns access time version, data will be valid at the outputs 300ns after valid addresses or 200ns after $\overline{CS}$ goes LOW, whichever is later.

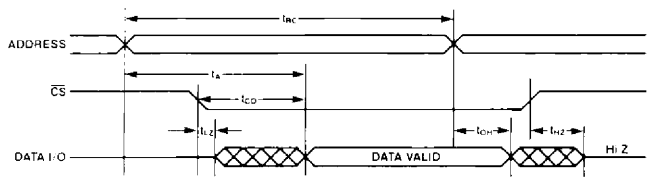


Figure 5) Timing diagram for Reading from the RAM.

Store Operation

The STORE operation is initiated by the application of an active LOW TTL pulse of 100ns or greater on the $\overline{STORE}$ pin. As long as the power supply remains within its specification for 10ms after the beginning of the STORE pulse the contents of the RAM will have been transferred to the EEPROM in total. The STORE operation cannot be terminated once initiated except by removing the power supply. This can not be counted upon to rapidly terminate the STORE operation since the user cannot determine how far the STORE operation the device has progressed. Additionally, if the power supply drops below the specification during the 10ms the integrity of the STORE operation is not assured. Figure 6 shows the timing diagram for transferring data from the RAM to the EEPROM.

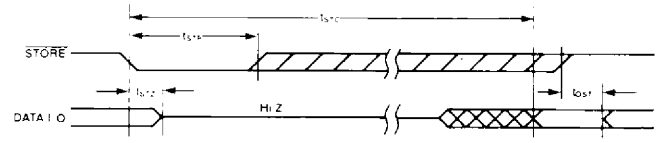


Figure 6) Timing diagram for the STORE operation where data is transferred from the RAM to the EEPROM.

The STORE operation takes precedence over all other operations except the RECALL operation. If the RECALL operation has begun, any STORE command is ignored as long as the $\overline{RECALL}$ pin remains LOW. Once the STORE operation has started, taking the $\overline{RECALL}$ pin LOW has no effect and the STORE operation will be completed. If a READ or WRITE cycle is in progress when a STORE command is received, that cycle is terminated. The data in the selected RAM address during an interrupted WRITE cycle would be indeterminate.

During the 10ms of the STORE operation the NOVRAM should not be accessed for any other operation as it would not be known if the internal STORE operation was completed or not. If the internal STORE operation was completed before the 10ms and another operation command was entered, that command would be executed. However, if the internal STORE operation was not completed and another operation command was received, the later command would be ignored. During the STORE operation the outputs of the NOVRAM are in the floating state.

Recall Operation

The RECALL operation is initiated by the application of an active LOW TTL pulse of 450ns or greater on the $\overline{RECALL}$ pin. The positive going edge of this pulse determines when it is possible to read data from the RAM. Valid data from the RAM can be viewed on the outputs of the NOVRAM 750ns after the rising edge of $\overline{RECALL}$ or 300ns after application of valid addresses, whichever comes latest. FIGURE 7 shows the timing requirements for transferring data from the EEPROM to the RAM.

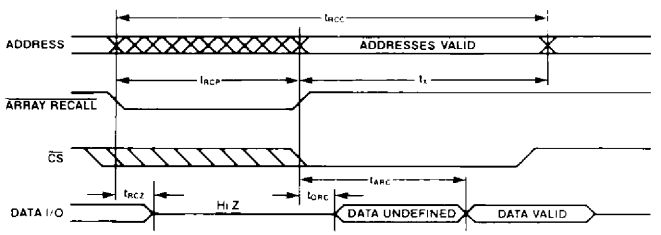


Figure 7) Timing diagram for the RECALL operation.

The RECALL operation takes precedence over all other operations. RECALL will terminate a READ or WRITE cycle if applied in the middle of either cycle. The RECALL operation can take precedence over the STORE operation only if the RECALL command was received prior to the STORE command.

Hooking up the NOVRAM

Now that the basics of the four NOVRAM operations have been described, we may discuss using these unique parts in a system. Let's first discuss a system in in which we connect the NOVRAMs to a 6502. We'll assume that the system powers up and down cleanly. The problem of systems where this does not occur will be dealt with later.

The 6502 microprocessor uses memory space for I/O functions. The design uses one set of addresses for the READ and WRITE operations and other blocks of addresses to initiate the STORE and RECALL operations. An APPLE II* computer was used as the 6502 computer because it has address space already decoded for I/O functions and convenient card slots to communicate with these decoder outputs.

The RAM was placed in the address space starting at HEX address $C800 by connecting pin 20 on the APPLE II peripheral connector to the $\overline{CS}$ on the NOVRAM. This pin is activated when any of the 2048 bytes starting at $C800 are accessed. Peripheral slot 2 was selected for the NOVRAM design. On slot 2 pin 41 is activated whenever the 16 bytes located at HEX address $C0A0 are accessed. This address space is sent only to slot 2 and was tied to the $\overline{STORE}$ pin on the NOVRAM. PIN 1 also has a unique address space for slot 2 and was tied to the RECALL pin. This space is the 256 bytes starting at HEX address $C200.

Figure 8 shows the connections of the card to be plugged into peripheral card slot 2. Reading and writing are accomplished from the BASIC programming language by PEEK and POKE instructions to HEX addresses starting at $C800. The STORE operation is called by either a PEEK or a POKE instruction to any of the 16 addresses starting at $C0A0. $\overline{RECALL}$ is initiated by accessing any of the 256 bytes starting at $C200.

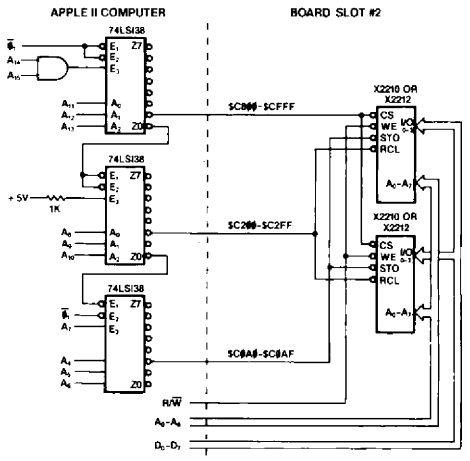


Figure 8) Connections for operating the NOVRAM on the I/O Bus of an APPLE II computer.

The APPLESOFT* program in Table 1 demonstrates how a NOVRAM can be accessed easily. The statement 210 is a Write operation to HEX address $C200 while statement 310 is a Write to HEX address $C0A0. The $\overline{CS}$ pin is activated with statement 430 and statement 580 respectively. This program was written for the X2210 but it could be easily modified for the X2212 or even eight X2201s.

```
5  TEXT : HOME
10 A$ = "@1 WELCOME TO THE WORLD'
     S EASIEST TO USENONVOLATILE
     STATIC RAM!!"
20 B$ = "@2 ONLY A SINGLE 5 VOLT
     POWER SUPPLY IS REQUIRED FOR
     OPERATIONS!"
30 C$ = "@3 ALL INPUTS AND OUTPUT
     S REQUIRE ONLY  TTL INTERFAC
     E SIGNALS!!!"
40 F$ = "@4 ALL TIMING SIGNALS AR
     E MICROPROCESSORCOMPATIBLE-N
     O STRECHING!"
50 G$ = "@5 3 SIZES AVAILABLE:(X2
     201)1024X1,      (X2212)256X4
     ,(X2210)64X4"
60  VTAB 5: HTAB 11: PRINT "XICOR
     'S X2210 DEMO"
75  PRINT : PRINT : HTAB 5: PRINT
     "1=RECALL";: HTAB 25: PRINT
     "3=READ"
80  PRINT : HTAB 5: PRINT "2=STOR
     E";: HTAB 25: PRINT "4=WRITE
     "
85  PRINT : HTAB 17: PRINT "5=EXI
     T"
90  VTAB 15: HTAB 12: INPUT "SELE
     CT ONE?,";A
100  IF A < 1 OR A > 5 THEN  HOME
     : VTAB 5: HTAB 5: PRINT "ONL
     Y 1-5 ALLOWED---TRY AGAIN": GOTO
      75
110  ON A GOTO 200,300,400,500,60
     0
200  REM  ***********************
201  REM  * RECALL SUBROUTINE *
202  REM  ***********************
210  POKE  - 15872,0
215  HOME : VTAB 5: HTAB 14
216  PRINT "RECALL COMPLETE"
218  FOR I = 1 TO 1000: NEXT I
220  HOME : GOTO 60
300  REM  ***********************
301  REM  * STORE SUBROUTINE *
302  REM  ***********************
310  POKE  - 16224,0
315  HOME : VTAB 5: HTAB 14: PRINT
     "STORE COMPLETE"
320  FOR I = 1 TO 1000: NEXT I
325  HOME : GOTO 60
400  REM  ********************
401  REM  * READ SUBROUTINE *
402  REM  ********************
422  HOME
423  VTAB 5: HTAB 12: PRINT "RESU
     LTS FROM RAM": PRINT : PRINT

425  FOR I = 1 TO 64
430  PRINT  CHR$ ( PEEK ( - 14336
      + I));
435  NEXT I
440  FOR I = 1 TO 5000: NEXT I
445  HOME : GOTO 60
500  REM  ***********************
501  REM  * WRITE SUBROUTINE *
502  REM  ***********************
505  HOME : VTAB 5: HTAB 6: PRINT
     "PRESELECTED MESSAGES (1-5)"
     : PRINT : PRINT
515  PRINT : PRINT : HTAB 4: PRINT
     "ENTER MESSAGE FROM KEYBOARD
      (6)"
520  PRINT : PRINT : HTAB 11: INPUT
     "SELECT ONE (1-6)?<";A
525  ON A GOTO 531,532,533,534,53
     5,540
531 Q$ = A$: GOTO 548
532 Q$ = B$: GOTO 548
533 Q$ = C$: GOTO 548
534 Q$ = F$: GOTO 548
535 Q$ = G$: GOTO 548
540  PRINT : PRINT : HTAB 5: PRINT
     "INPUT MESSAGE-64CHARACTER$
     MAX": INPUT Q$
545 Q$ = Q$ + "
                                "

548  HOME : VTAB 5: HTAB 12: PRINT
     "WRITTEN INTO RAM": PRINT : PRINT

550 Q$ =  LEFT$ (Q$,64)
570  FOR I = 1 TO 64
575 Z$ =  MID$ (Q$,I,1):Z =  ASC
     (Z$)
580  POKE  - 14336 + I,Z
582  PRINT Z$;
585  NEXT I
590  FOR I = 1 TO 5000: NEXT I
595  HOME : GOTO 60
600  END
```

Table 1) An APPLESOFT BASIC program for demonstrating the four operations of NOVRAMs.

* APPLE II and APPLESOFT are trademarks of Apple Computers.

Protection Against Inadvertent Store

The circuit described in the previous section assumes that the system is powered up and down in an orderly manner. This would mean that the microprocessor would never generate addresses unless they were part of the program. Unfortunately real systems do not operate in this ideal manner. Although the circuit described above has not produced a fault during extensive testing, the possibility exists that the hex addresses $C0A0-$C0AF could come up during power up, or during a brown out when the supply dropped below the operating specification, or during a power failure.

Several methods can be used to insure that the NOVRAM does not react to errors produced by the system when it is out of its operating specification. Setting the $\overline{\text{RECALL}}$ pin LOW to block a STORE operation is the easiest. Holding the $\overline{\text{STORE}}$ pin between VIH MIN and the falling power supply is another.

Most microprocessors are not totally under control for the first few cycles after power up. Their early addresses depend on what is in the registers after the System Reset pulse terminates. There is a possibility that these registers can cause one of the early addresses to be the same selected for a STORE operation. In this case the circuit shown in Figure 8 could cause the EEPROM to be written with false data during the power up operation. Figure 8 allows the STORE operation to be initiated if any of 16 addresses is selected for either a read or a write.

Although microprocessors can put out uncontrolled addresses they do not put out uncontrolled write commands. By ANDing the System Write line with the System STORE command, the NOVRAM would recognize a STORE operation only on a machine write cycle.

A potential danger in the use of the above schemes is the fact that three state TTL gates are not under total control while the power supply is coming up. This could produce glitches on the STORE pin even though no Write command was received at the input of the gate. A more positive way to insure that the $\overline{\text{STORE}}$ pin follows the power supply as the voltage increases is to use an open collector NAND gate with one of the inputs provided by a signal that determines power supply status (Circuitry for power supply status will be covered later in this application note). If one input of an open collector NAND gate is held LOW the output transistor is turned OFF since it can not receive base current. Pulling the output of this gate to the power supply of the NOVRAM through a pullup resistor will then insure that the output follows the power supply with no glitches.

Carrying the use of a power supply status signal one step further, would be to use it to hold the $\overline{\text{RECALL}}$ pin LOW in addition to holding the $\overline{\text{STORE}}$ pin HIGH. A direct connection of this status signal to the $\overline{\text{RECALL}}$ pin is all that is necessary as shown in Figure 9. This circuit has a more positive control of the NOVRAM since it takes two actions to prevent an inadvertent STORE operation.

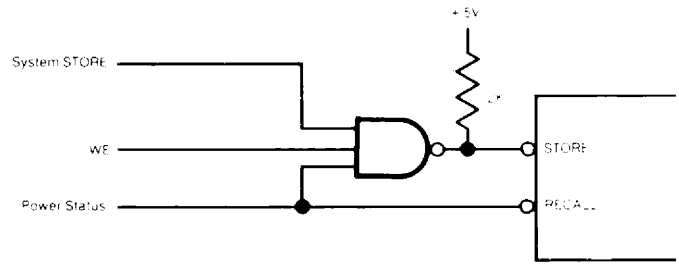


Figure 9) A power supply status line held LOW can insure that both the $\overline{\text{RECALL}}$ pin is held LOW and the $\overline{\text{STORE}}$ pin is held HIGH.

An example of a basic circuit to monitor the power supply status is shown in Figure 10. The output of this two-stage circuit is held LOW whenever the power supply is below 4.5 Volts. This same technique can be used with a Zener diode and an operational amplifier. The designer is cautioned to consider temperature effects.

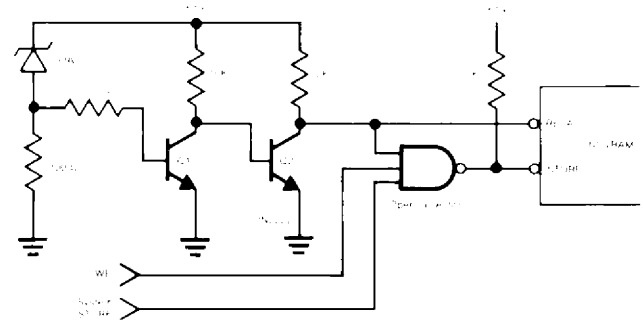


Figure 10) The Zener power supply detector is used in combination with the open collector on the $\overline{\text{STORE}}$ pin to provide protection against an inadvertent STORE operation during power up and power down.

Another method of power supply status is to assume that the only power supply fault which requires insuring that wrong data is not stored is the loss of the AC line voltage. Many commercially available AC line fault monitors are on the market. Two of these line fault monitors are the MID 400 Power Line Monitor from General Instrument and the SG1547 from Silicon General. Additionally, many commercially available power supplies have a power fail signal either as a standard feature or as an option.

The circuit shown in Figure 11 shows another type of power supply status detector. This circuit is a low cost solution but it should be used only to take the $\overline{\text{RECALL}}$

pin LOW because it does not provide adequate drive for a TTL gate.

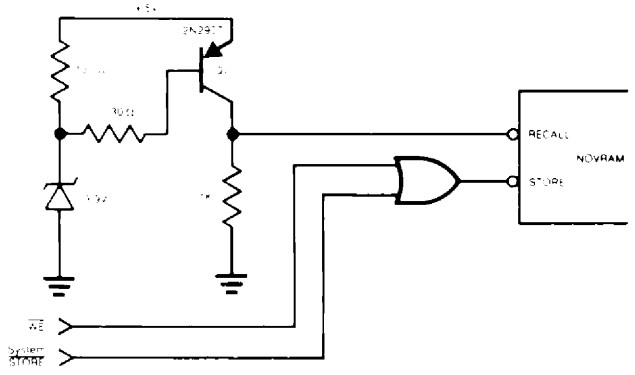


Figure 11) Another form of power supply status detector to drive the $\overline{\text{RECALL}}$ pin LOW when the supply drops below 4.5Volts.

Some other schemes to protect against inadvertent STORE operations are the use of jumpers, cables and/or switches. The STORE signal is transmitted through the jumper or switch which is normally open unless it is desired to change the data in the EEPROM. During normal operation the only component attached to the $\overline{\text{STORE}}$ pin is a resistor to the power supply.

A more comprehensive discussion of power supply status circuitry can be found in XICOR's Ap Note #102. This note covers those requirements that STORE data at power failure.

Applications

Most microprocessor systems have need for some form of nonvolatile memory to store important data such as:

1) Calibration constants
2) Set-up configuration information
3) User system ID
4) Changeable programs/firmware
5) System status
6) Accounting information
7) Error conditions

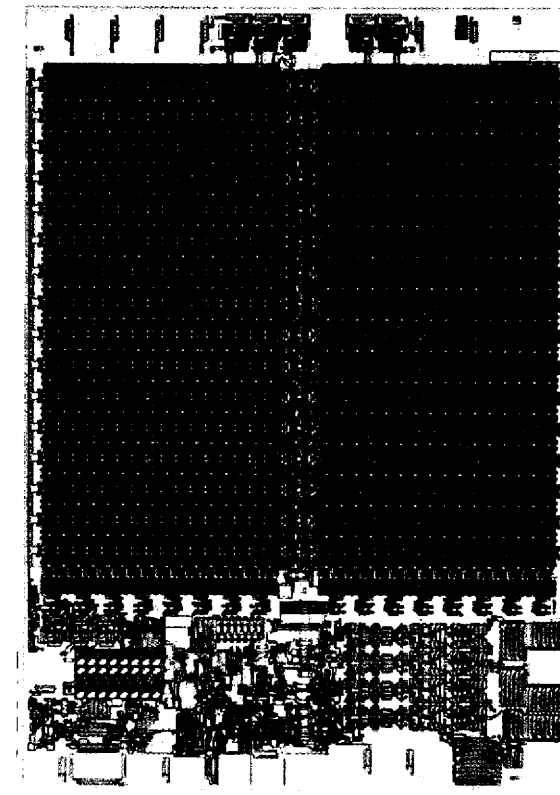
The types of equipment that are today being designed to include NOVRAMs vary through the complete line of electronic equipment. Some of these systems are:

1) Computer peripherals/terminals etc.
2) Automatic tellers/transaction terminals
3) Point-of-sale terminals
4) Smart scales
5) Vending machines and games (i.e. arcade games, slot machines)
6) Meters (i.e. utilities, taxi, gas pumps)
7) Process control
8) Robots
9) Instruments (musical, medical, test, avionics)
10) Communications
11) Transducers/load cells
12) Automotive: odometers, engine control
13) Office equipment: copiers, word processors
14) Military products

Some of the competitive products being replaced by NOVRAMs in systems are:

1) DIP switches
2) Thumbwheel switches
3) CMOS with batteries
4) EAROM
5) EEPROM
6) Potentiometers

One should let his imagination soar when thinking applications for these unique NOVRAMs. XICOR is always interested in application ideas that represent both normal and off-beat uses of these NOVRAMs. Any ideas sent to XICOR will be greatly appreciated.



STORING DATA IN XICOR NOVRAMS DURING POWER FAILURE

BY GEORGE LANDERS
AND JÜRG RUDIN

Introduction

Many systems require that some data be saved at the time of a power failure. This can be a wide variety of data such as accounting information, system status, program counter or register contents, or security information. The XICOR NOVRAM family is ideal for applications of this type as a NOVRAM* memory can save the entire contents of its RAM into an on-chip EEPROM in only 10ms and with a single 5V power supply.

This application note addresses several methods of detecting a power failure and insuring that the regulated + 5 Volts supply remains within specification for at least the 10ms required to complete the STORE operation. Various schemes of detecting the absence of ac and the decline of unregulated dc are discussed. The use of either hardware or software control is also covered. Since each system and power supply has individual requirements, it will be up to the system designer to select the circuit approach best suited to the application. None of the circuits shown are necessarily better in all cases and indeed the ideal circuit for a particular situation may not even be shown.

Store Requirements

The requirements to save data in the NOVRAM memory are very straightforward. It is only required that a low level TTL pulse, 100ns or greater duration, be applied to the $\overline{\text{STORE}}$ pin and the power supply remain above 4.5 Volts for 10ms after the beginning of the STORE pulse. Figure 1 shows the relationship between the STORE pulse and the + 5 Volt power supply. Once the STORE operation has started it can not be terminated unless the supply drops below + 4.5V.

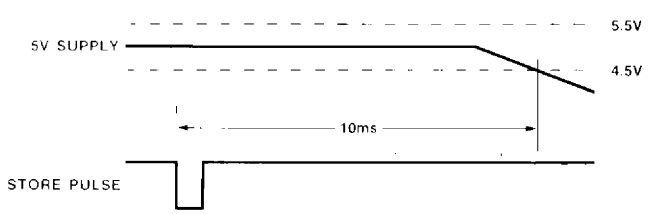


Figure 1) The + 5 Volt power supply must remain above + 4.5 Volts for 10ms after the STORE pulse starts.

During the STORE operation, the NOVRAM memory is not on the bus allowing the microprocessor to complete other tasks that may be required by the system for an orderly shutdown. The STORE operation is determined by the falling edge of the STORE pulse. If the $\overline{\text{STORE}}$ pin is held LOW, another STORE cycle will not be started. The $\overline{\text{STORE}}$ pin must be brought HIGH before another STORE operation can occur.

Since the regulated + 5 Volt power supply must remain above + 4.5 Volts for 10ms, some other point must be monitored for power failure. The detection of regulated dc dropping is already too late to provide an assured 10ms of at least + 4.5 Volts. Two possible points to monitor for an early indication of declining power are the ac to the power supply and/or the raw or unregulated dc to the regulator itself.

Detecting the absence of ac can be performed either before or after the transformer. If it is done in front of the transformer, it is necessary that the signal be electrically decoupled from the dc portions of the system. This can be accomplished by using optoelectronic components which transmit light signals between two points with differing electrical references.

If detection is accomplished after the transformer, it needs to be isolated from the heavily filtered unregulated dc of the main power supply as the response time there is very slow. This is done by using either a separate transformer tap or two extra diodes to isolate the signal from the bridge. Figure 2 shows the relationship necessary between the ac, the + 5 Volt supply and the STORE pulse.

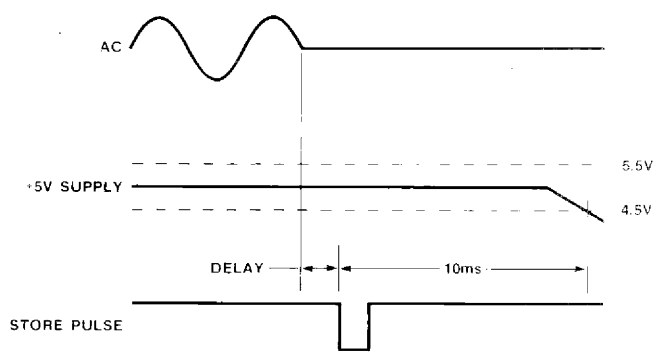


Figure 2) Missing ac must trigger a STORE pulse at least 10ms prior to the regulated dc dropping below specification.

The other technique for early detection of failing power is to sense the unregulated or raw dc in front of the regulator. The regulator has a range of input voltage for which it will maintain the output within fixed limits. If a trip point is set up below the normal input voltage, a STORE signal can be sent to the NOVRAM memory with enough time to insure that the + 5 Volts stays within specification for at least 10ms as shown in Figure 3.

*NOVRAM is Xicor's nonvolatile static RAM device.

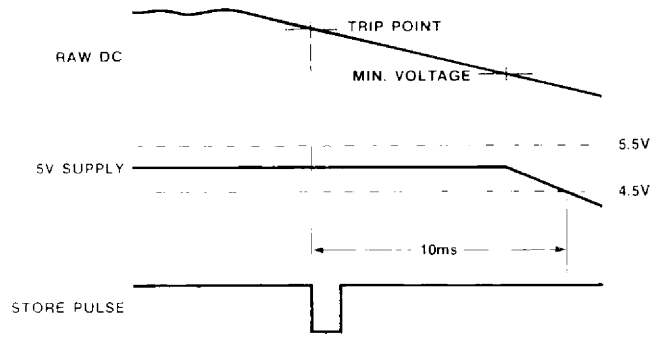


Figure 3) Falling dc is detected to trigger a STORE pulse at least 10ms before the +5 Volt supply drops below 4.5 Volts.

AC Power Failure Detect Circuits

It is possible to detect the loss of ac power with some very simple circuits. Four differing approaches are discussed in this section. These include a simple transistor zero crossing detector, two optoelectronic detectors and a CMOS Schmitt trigger method.

The circuit shown in Figure 4 is a low cost technique for detecting the loss of ac. It uses two diodes to isolate the circuit from the filter capacitor in front of the regulator. The resistor R2 is selected to limit the base current of Q1 and is dependent on the value of the ac signal.

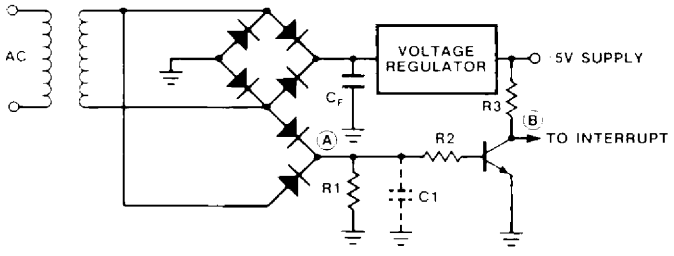


Figure 4) A circuit to detect the absence of ac.

When the full wave rectified ac drops toward zero, the transistor Q1 loses base current and the output goes HIGH. If the capacitor is not used, the circuit produces a short HIGH pulse every 8.3ms. This signal is then used to interupt the microprocessor. Since the ac has probably not been lost, the microprocessor starts a subroutine to determine if this is a true power loss event. This subroutine produces a short delay (2ms) and looks at the detector again. If the output of the detector is still HIGH the power loss is real and the microprocessor issues to the NOVRAM memory a STORE pulse. The waveform produced by the circuit without the capacitor is shown in Figure 5.

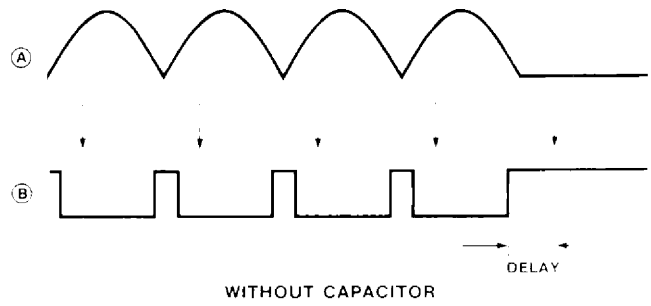


Figure 5) Waveforms in the circuit of Figure 4 without the capacitor. The microprocessor is interupted every 8.3ms and must check 2ms later to determine if the power loss is real.

The disadvantage of using an interupt every half cycle is that the available processing time is reduced by nearly 25%. In many applications this is undesirable. A capacitor in the circuit of Figure 4 eliminates the pulses every half cycle. The value of that capacitor depends on many factors. Among these are the holdup time of the power supply, the desired delay of missing ac before triggering STORE and the values of R1 and R2. Figure 6 shows the signals under these conditions. In the lab circuit the values were R1 = 33K Ohms, R2 = 10K Ohms, C1 = .1 microfarad and a full wave rectified ac of 20 volts peak. This combination produced a delay of 2ms from the time that a half cycle was missing.

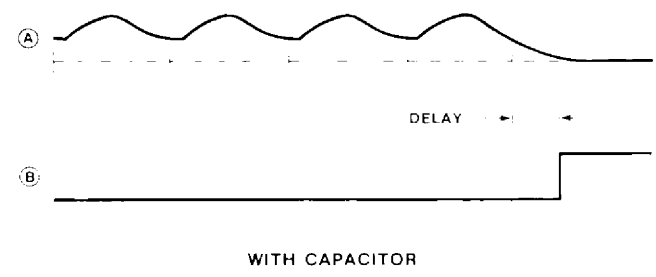


Figure 6) A capacitor in the circuit of Figure 4 fills in the valley in the rectified ac to produce a pulse only when ac disappears.

Care should be taken to insure that the time rise of the output of any detector circuit is fast enough for the microprocessor. Extra logic gates can be used to improve the timerise.

An approach that allows direct connection to the ac power line is shown in Figure 7. In this circuit the ac line is connected through a resistor to two optoelectronic isolators connected with their diodes back to back.

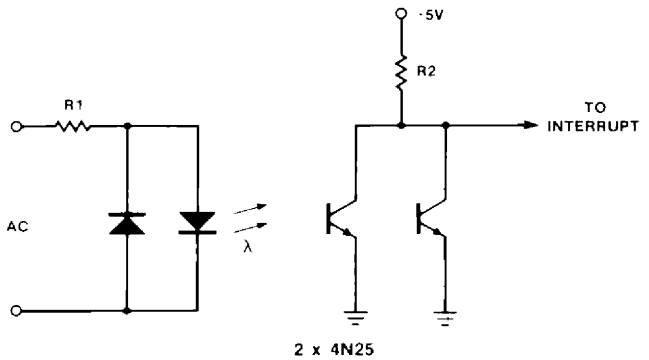


Figure 7) Two optoelectronic isolators are connected with their diodes back to back to provide an interupt signal.

This circuit produces a positive pulse each time that ac line has a zero crossing as shown in Figure 8.

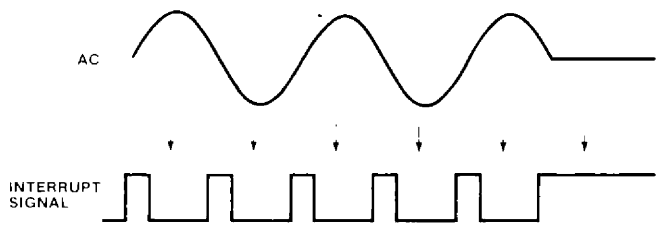


Figure 8) The circuit of Figure 7 produces positive pulses every 8.3ms until ac power failure and then remains HIGH until the dc decays.

This circuit is similar to the one in Figure 4 in that it produces an interupt every 8.3 ms. This ties up the microprocessor during the delay time when the output of the detector must be resampled. In addition to the method of adding a capacitor to the circuit in Figure 4, a missing pulse detector similar to that shown in Figure 9 can be used.

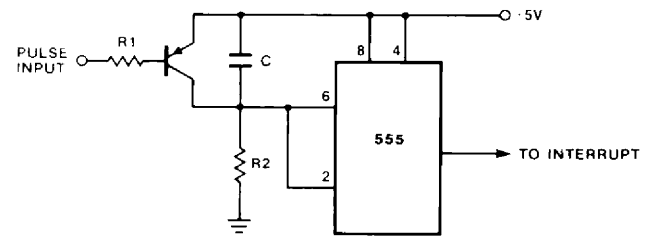


Figure 9) A 555 timer connected as a missing pulse detector.

Another circuit for connecting directly to the ac power line is shown in Figure 10. This circuit uses the MID400 from General Instruments. It requires only two resistors to provide a clean interupt signal.

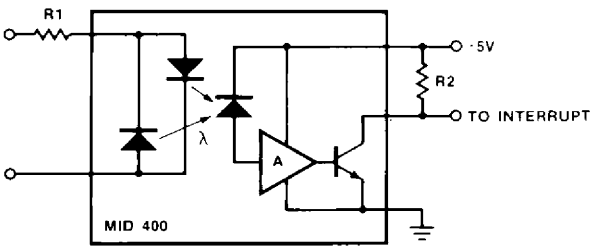


Figure 10) The MID400 requires only two resistors to connect directly to the ac power line.

The waveforms are shown in Figure 11. The turn on and turn off delays can be adjusted with a capacitor. (see the G.I. data sheet for details)

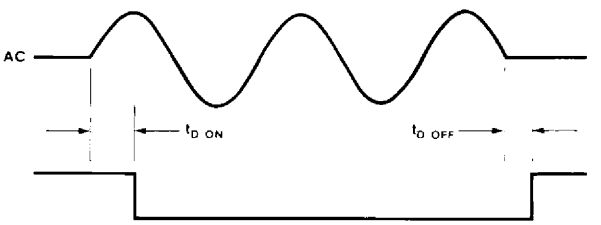


Figure 11) Waveforms for the MID400.

The last ac detect circuit to be discussed uses a CMOS Schmitt Trigger as both a full wave ac low voltage detector and a missing pulse detector. This circuit is shown in Figure 12.

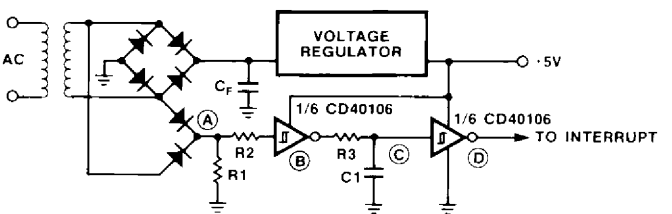


Figure 12) Two CMOS Schmitt Triggers are used to detect the absence of ac.

Care should be taken to insure that the input voltage of the CMOS Schmitt Triggers does not exceed the 5 Volt power supply. If this occurs, the possibility of latch-up exists which can be destructive to the CMOS cir-

cuits. A resistor in series between the bridge and R1 will limit the input voltage.

The ac input voltage should be as high as possible to provide narrow pulsewidths out of the first gate. The values of R3 and C1 determine the delay from the output. The waveforms can be seen in Figure 13.

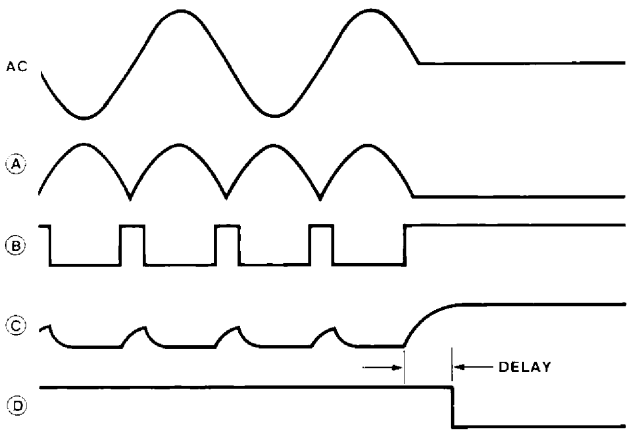


Figure 13) The delay established by R3 and C1 must be longer than the pulsewidth out of the first gate.

DC Power Failure Detect Circuits

Detecting the decline of the raw dc does not involve missing pulses or capacitor delays. The first circuit shown in Figure 14 uses a Zener diode to set a trigger point. This trigger point should be as high as possible without being high enough for the normal range of unregulated dc to trip it. The value of the diode should be selected to be equal to the trip point desired minus .7 volts for the base-emitter drop.

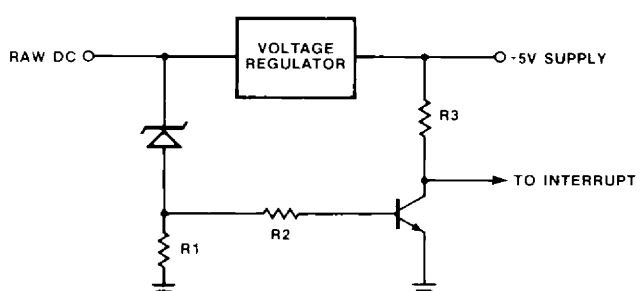


Figure 14) A Zener diode in series with a base-emitter diode establishes the trip point of this dc detector.

A PNP transistor can be used as shown in Figure 15. In this circuit the value of the Zener diode is the desired trip point minus 5 Volts plus .7 Volts for the base-emitter drop.

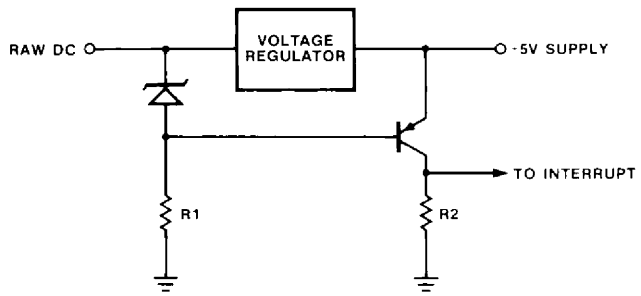


Figure 15) A dropping unregulated dc turns on the PNP transistor.

Care should be used with these last two circuits to insure that all tolerances and temperature coefficients have been considered.

The last circuit that will be discussed in this application note uses an operational amplifier and a Zener diode with some resistors as shown in Figure 16. The circuit trips when the raw dc drops the junction of R1 and R2 to the value of the Zener diode. This circuit can provide either a positive or negative interupt signal depending on the connection of the two amplifier inputs.

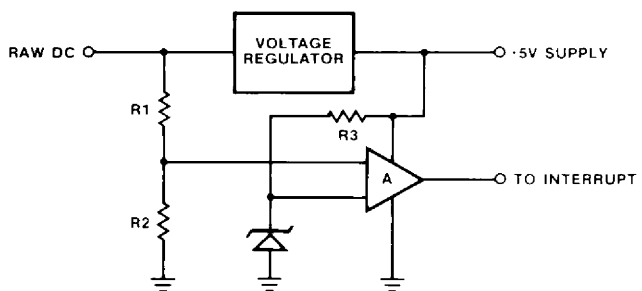


Figure 16) R1 and R2 with the Zener diode determine the trip point of this circuit.

Notes On Filter Capacitors

The value of the filter capacitor on the unregulated dc to ground should be high enough to ensure that the regulator remains in regulation for at least 10ms after the STORE pulse has been sent. This is dependent on the value of the trip point, the lowest input voltage to the regulator and the load. For an example consider a system with a 300mA load, a 15 Volt trip point and a 7 Volt lowest regulation voltage. Using the $I = CdV/dT$ equation we have: $C = 300 \times (10/8) \times 10^{-6} =$ 375microfarads.

Notes On Software

When using a microprocessor interupt to issue the STORE command, the program should branch to a STORE subroutine. This subroutine contains a short delay followed by a test of the power supply detector to insure that the power failure is valid. This can be seen in Figure 17.

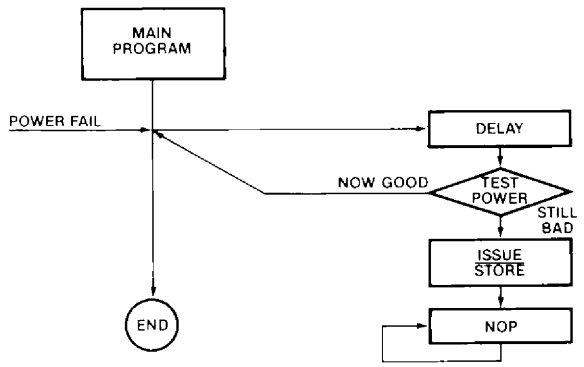
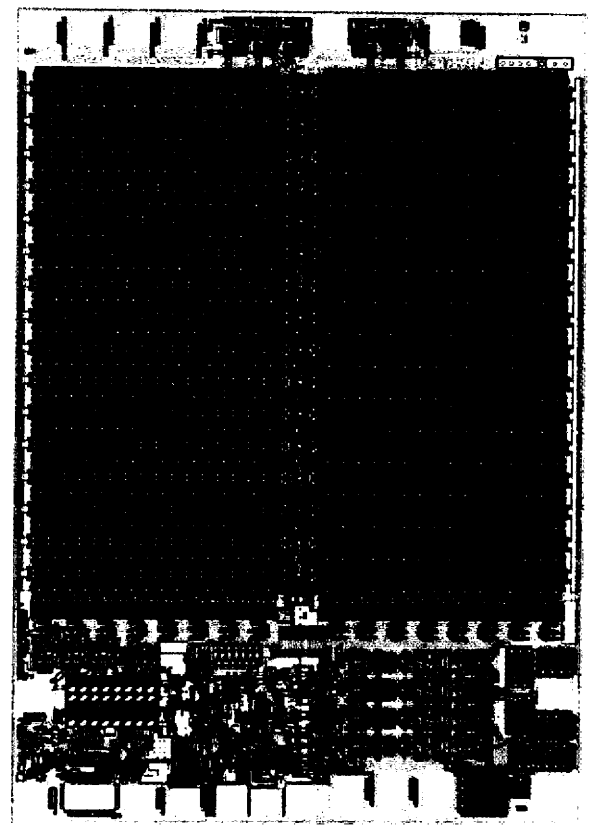


Figure 17: The program branches to a STORE subroutine when a power failure is detected.

Note: The exact values of components in the preceding drawings depend on system conditions such as secondary AC voltage and power supply hold-up time.

Some lab values for the drawings to use as starting points are as follows:

Figure 7. $R_1 = 12k\Omega$, $R_2 = 2k\Omega$, AC = 120V
Figure 9. $R_1 = 5k\Omega$, $R_2 = 3.3k\Omega$, $C = 1\mu F$, Time delay = 2msec
Figure 10. $R_1 = 22k\Omega$, $R_2 = 2k\Omega$, AC = 120V
Figure 12. $R_1 = 20k\Omega$, $R_2 = 20k\Omega$, $R_3 = 10k\Omega$, $C_1 = .2\mu F$, Rectified AC = 20V, Time Delay = 2msec
Figure 14. Zener = 10V, $R_1 = 1k\Omega$, $R_2 = 2k\Omega$, $R_3 = 2k\Omega$, Trip Point = 10.7V
Figure 15. Zener = 10V, $R_1 = 2k\Omega$, $R_2 = 2k\Omega$, Trip Point = 14.3V
Figure 16. Zener = 3.3V, $R_1 = 20k\Omega$, $R_2 = 6k\Omega$, $R_3 = 2k\Omega$, Trip Point = 14.3V



XICOR REPLACES DIP SWITCHES AND TRIMMERS WITH NOVRAM MEMORIES

BY GEORGE LANDERS

Introduction

The desire to replace mechanical components in electronic systems for purposes of increased reliability, lower costs and ease of maintainability has spread to DIP switches and trimming potentiometers or trimmers. The component that makes this replacement possible is the NOVRAM memory from Xicor. The NOVRAM memory is a device that has two memories in parallel, a standard static RAM and a nonvolatile electrically erasable programmable read only memory (EEPROM). The EEPROM portion of the NOVRAM memory holds data that is equivalent to the settings of the now obsolete DIP switches and trimmers.

What Is A NOVRAM Memory?

A NOVRAM memory, as stated previously, is two memories in a single unit. The standard static RAM has a nonvolatile EEPROM cell associated with each RAM cell. Figure 1 shows a block diagram of a typical NOVRAM memory.

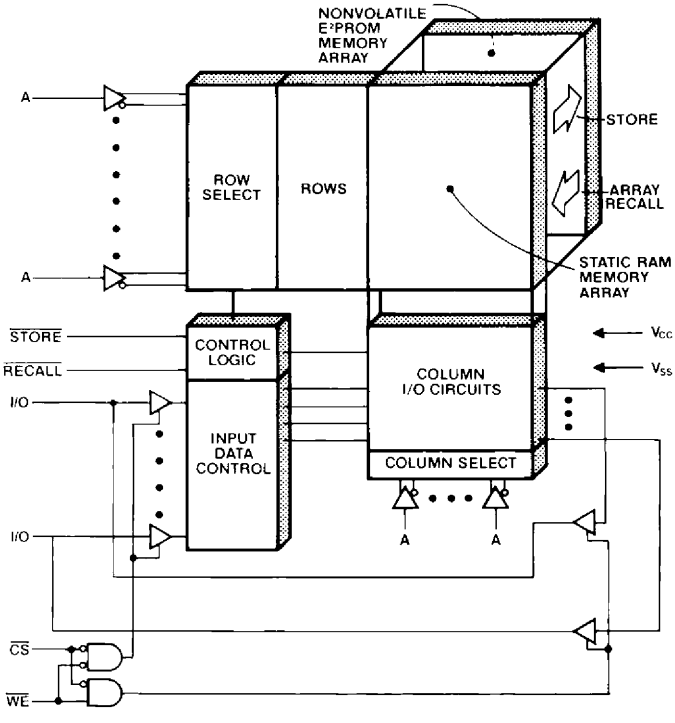


Figure 1) NOVRAM memory block diagram.

There are two additional pins on a NOVRAM memory device that do not appear on an ordinary static RAM. These two pins are called $\overline{\text{STORE}}$ and $\overline{\text{RECALL}}$. The $\overline{\text{STORE}}$ pin is used to transfer the entire contents of the RAM to the EEPROM as a single block. This operation is performed in parallel. The $\overline{\text{RECALL}}$ pin is used to transfer the entire contents of the EEPROM back to the RAM. At the end of either operation, the contents of the two memories are identical.

Access to the EEPROM data is through the RAM portion. To alter the contents of the EEPROM, the data must first be written into the RAM and then transferred to the EEPROM with a Store operation. To use the contents of the EEPROM in the system, perform a Recall operation and then the contents of the RAM may be read. Once data is stored in the EEPROM, the RAM can be used as an entirely separate and independent memory. Some users put configuration data into the EEPROM and then use the RAM as a separate scratchpad.

Besides these operational features, the NOVRAM memory has some unique electrical features. These devices are the world's easiest-to-use nonvolatile components in that they operate with only a single 5-volt power supply, simple TTL level pulses and short pulse widths (<450 ns). Even for operations such as the Store operation, which takes 10 ms to complete, it only requires a low level TTL pulse of 100 ns or greater to initiate. During the remaining time, the NOVRAM memory is not on the bus, which frees the microprocessor and the bus for other tasks. Complete details of the operation of NOVRAM memories can be found in the individual data sheets and application note AN101.

Replacing DIP Switches With NOVRAM Memories

DIP switches and thumbwheel switches have been used in systems to provide alterable, nonvolatile data. Some uses of this data are to set up configuration parameters and to provide calibration constants. The apparent low cost of these components is one of their attractive features. The drawback is that costs of these components do not end with installation.

The biggest cost of these mechanical, nonvolatile components is in post-installation service. A simple change of a DIP switch setting can require a technician to visit the equipment, disassemble the unit, throw the switch and reassemble the equipment. This could easily run the total use costs to well over 10 times the installed cost. A solution to the problems presented by DIP switches is to use a NOVRAM memory to hold valuable configuration or calibration data. In addition to a lower-cost, easier, and more secure method of changing data, NOVRAM memories cost less at the installed level.

The disadvantages of using DIP switches in modern electronic systems accumulate through each step of the manufacturing process. The first stage of NOVRAM memory advantages starts right at system concept and design. Since the density of NOVRAM memories is significantly greater (up to 1024 switches in a single low-cost DIP package), more functional options can be added to enhance the total value of the system. Features such as electronic unit type signature can be added for a small software cost, with no extra components. No special access needs to be provided to change the NOVRAM memory, as all changes can be made from a keyboard or over phone lines. This can not be said for DIP switches, which require disassembly or special doors or hatches to provide access.

At incoming inspection it is difficult to completely test a package of DIP switches for all possible combinations, or even as individual switches. The NOVRAM memory, on the other hand is tested by automatic test equipment both quickly and thoroughly. The NOVRAM memories are 100% tested by Xicor and can be further tested at whatever levels the user desires, including the quick testing of all the options that were designed into the equipment. In the case of the DIP switch, this would require manually setting each option, rather than have the final system test equipment take care of the task.

The assembly operation is made more difficult when trying to wave solder or clean a board containing DIP switches. These operations can cause contamination in the degreasing step. This is true, even on the components that have tape or other cover for protection, as these are extra items to handle or become lost. Again, the NOVRAM memory exhibits none of these problems in that they are in sealed packages like the rest of the semiconductor components that make up the bulk of the system.

Once the system is in the field, the advantages of the NOVRAM memory are further enhanced. The basic reliability improvement of semiconductors over mechanical components is well known. Equipment warranties can be enforced since there is no need for a customer to open the equipment. The greatest advantage of all comes in service. No longer is it necessary for a technician to travel to the users site to change the setting on DIP switches as this can be accomplished over a phone hookup.

In addition to all of the above cost savings and system benefits in using NOVRAM memories, the basic component cost is also very low. Figure 2 shows a typical interface for DIP switches in a microprocessor system. Each package of 8 switches requires a decoder port and 8 diodes to provide isolation from other switches.

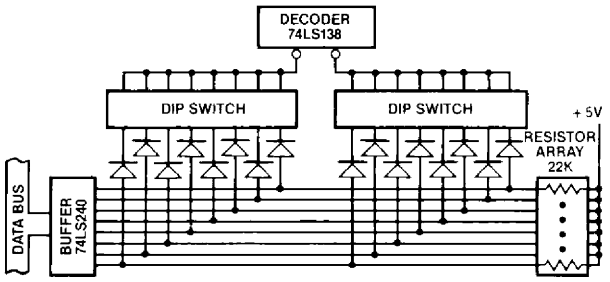


Figure 2) Typical DIP switch interface for multiple packages.

An octal buffer and 8 pull up resistors are required for any quantity of switch packages in a given system. Matrix schemes could be applied to reduce the decoder ports at the cost of more buffers but, by then, the costs will be much greater than those of using NOVRAM memories.

The assembly costs include incoming inspection, handling, inventory, board real estate, and final inspection. These costs are variable depending on volume and other factors.

The interface of a NOVRAM memory to a microprocessor is shown in Figure 3.

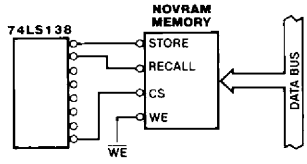


Figure 3) The typical NOVRAM memory interface requires only 3 decoder ports for any number of switches up to 1024.

This setup requires 3 decoder ports for any number of switches up to 1024 and then starts adding a single port for each additional package of 256 or 1024 switches.

The plot in Figure 4 shows that system costs using NOVRAM memories remain constant as the equivalent packages of DIP switches required are increased. These costs include the costs of all associated components, assembly and testing.

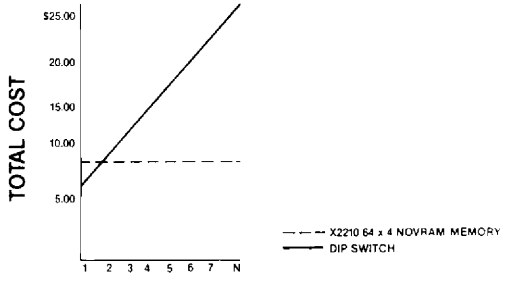


Figure 4) Relative costs of using NOVRAM memories as opposed to DIP switches as a function of packages of 8 switches required.

The plot shows that whenever the required number of DIP switch packages of 8 switches exceeds 1, the NOVRAM approach is lower in cost at the systems level.

The cost of using DIP switches rises constantly as the number of required packages increases. An actual cost crossover occurs between 1 and 2 packages of 8 switches. Designers can derive their actual costs by calculating the two approaches based on the costs at their firm. These costs should include, in addition to the component costs, all costs associated with incoming inspection, warehousing, assembly and system tests. One will find that the crossover between 1 and 2 packages of 8 switches is consistent and favors the NOVRAM memory approach.

Where And Why Trimmers Are Used

The trimming potentiometer or trimmer is a 3-terminal device that can be connected in many different configurations. The purpose of the trimmer in the analog circuit is to make a fine adjustment of a current or a voltage. This current or voltage is then used in analog circuits to compensate for component variations in frequency, gain, offset, voltage or current.

Like the DIP switch, the trimmer appears quite inexpensive when one considers only the purchased price. In actuality it can be one of the most expensive components on the card when the costs of field calibration are taken into consideration. It takes only one service call (considered by some to cost approximately $200) due to a changed setting caused by vibration, humidity or even well-intentioned user tampering, to run the cost of using the trimmer to high levels. In addition, the trimmer requires equipment disassembly and the skilled use of a screwdriver. This skill adds to the cost of owning the equipment.

Taking an 'all costs considered' approach is one way manufacturers are reducing the cost of equipment ownership as a function of performance. Although the end customer wants equipment that is low in purchase price and service costs while delivering a high level of performance, they will purchase a more expensive piece of equipment if they believe that the service costs and possible downtime will be reduced.

The functions of the trimmer can be duplicated quite well by a NOVRAM memory combined with a Digital-to-Analog Converter (DAC). A DAC is a device that delivers a voltage at the output that is a function of a digital signal at the input. In a microprocessor system, this is a variable voltage source that is under the control of the program. While the DAC cannot exactly duplicate the 3 terminals of the trimmer, the circuit can be modified to provide equivalent results.

The NOVRAM memory provides settings for the DAC that are free from problems of humidity and vibration, as well as holding onto those settings during times of no power. Once the NOVRAM memory/DAC combination is in the circuit, the calibration can be made automatic by closing the loop since all mechanical adjustments are eliminated. A self-calibrating system can eliminate all expensive service calls for recalibration.

Duplicating The Function Of The Trimmer

This section will demonstrate a few simple concepts for using a NOVRAM memory and a DAC in combination to modify important circuit parameters. As previously mentioned, a trimmer adjusts small variations of frequency, gain, offset, voltage or current. By properly interfacing the output voltage of a DAC in the analog circuit, these functions can be easily duplicated.

The first example shown will demonstrate how to effect a small adjustment in voltage that can be used as a reference or for some other need. Figure 5 shows an operational amplifier connected to provide a small amount of trim to the output. The 9.9k and 100 ohm resistors provide a division by 100 of the DAC output.

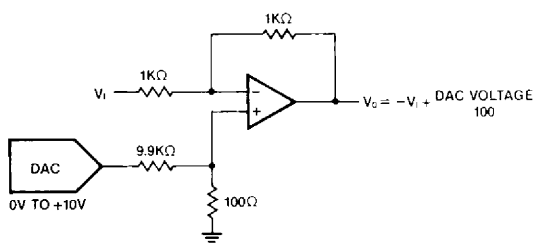


Figure 5) The DAC output provides up to 100 millivolts of trim for the operational amplifier output voltage.

If the DAC can be adjusted from 0 to 10 volts, this voltage divider provides an offset of up to 100 millivolts. The operational amplifier offset adds this amount to the output, providing up to 100 millivolts of reference voltage trim.

The next example will show how to provide a small offset for a fixed gain amplifier. Figure 6 shows the operational amplifier connected as an inverting amplifier with a gain of 10.

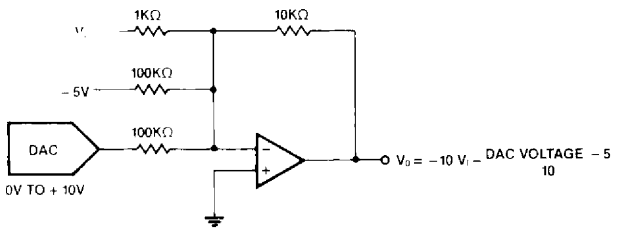


Figure 6) The DAC output provides an offset of 0.5 volt for a fixed gain amplifier.

The fixed gain is established by the 1K and 10K resistors. As the DAC output is varied from 0 through 10 volts, this voltage, combined with the −5 volts, reduces the amplifier output by $^1/_{10}$ of the difference. This gives a fixed offset of up to 0.5 volt in either direction.

The third example will show a different use of a DAC to change the operational amplifier gain. This example uses a CMOS DAC with the ladder network in the amplifier feedback loop.

A short course in CMOS DACs is in order at this time. Figure 7 shows a simple 3-switch CMOS DAC.

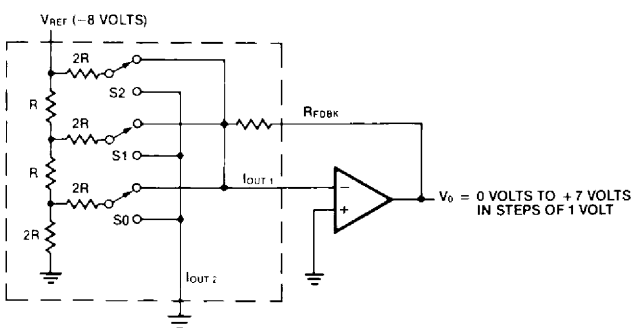


Figure 7) A simplified 3-switch CMOS DAC.

The outputs of a CMOS DAC are in the form of current. The sum of the two output currents is always a constant. In the case of Figure 7, this sum is $^7/_8$ x V_{REF}/R. Both current outputs must look into a ground potential. In Figure 7 the I_{OUT1} pin is tied to the summing junction of an operational amplifier while the I_{OUT2} pin is tied to system ground. The internally provided feedback resistor should be used with an amplifier since its temperature coefficient is identical to the other resistors on the DAC chip. The DAC switches are operated by standard 5 volt logic levels. The amplifier output in Figure 7 will vary from 0 to 7 volts in 1 volt increments depending on the setting of switches S0, S1 and S2. These switches in the 'up' position add 1, 2 and 4 volts, respectively, to the amplifier output. In the positions shown, the amplifier output is 7 volts.

Figure 8 shows the CMOS DAC of Figure 7 in a slightly different configuration.

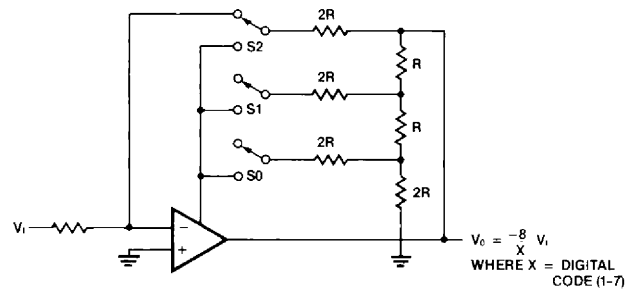


Figure 8) A CMOS DAC used in the amplifier feedback loop to adjust the amplifier gain.

The ladder network provides the feedback to the amplifier while the internal feedback resistor is used as the input resistor. If one goes through the equations, the result for Figure 8 is VO = VI x 8/X, where X is the digital code for the switch settings from 1 to 7. The circuit gain runs from a low of $^8/_7$ for the switches in the indicated position, to a gain of 8 when S0 is high and the other switches are low. Table 1 shows a listing of the gains obtainable.

S2	S1	S0	GAIN
L	L	H	8
L	H	L	4
L	H	H	$2^2/_3$
H	L	L	2
H	L	H	$1^3/_5$
H	H	L	$1^1/_3$
H	H	H	$1^1/_7$

Table 1) Gains of the circuit in Figure 5 as a function of the switch settings.

Analog Circuit Examples

This section will present some actual circuit examples for using a NOVRAM memory combined with a DAC. The circuits that appear in this section have been built and tested. The concepts presented may be useful to stimulate ideas which will help to solve the reader's system problems and may even be of immediate use. Figure 9 shows how the NOVRAM memory/DAC combination provides a voltage or a current for the application.

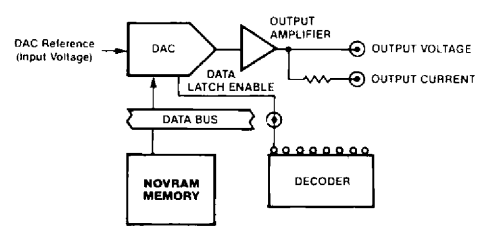


Figure 9) A NOVRAM memory DAC combination provides a voltage or a current to correct analog circuits.

It is, of course, possible for a single NOVRAM memory to provide the address setting for multiple DACs. The DAC size used is selected for the user's application, depending on the accuracy and resolution required. There are even multiple DACs available in a single package such as the SAB 3013 from Philips for more cost sensitive applications.

Tuneable Crystal Oscillator

The first application example of a NOVRAM memory used in combination with a DAC is that of a quartz crystal oscillator. These circuits find application in many areas, including aviation and nautical navigation, as well as time measuring due to high stability. The oscillator is normally trimmed with a small padding capacitor in shunt or series with the crystal. This trim is used to 'pull' the resonance point of the crystal by a few parts per million (PPM) to set the operating frequency of the circuit. The capacitor may have to be adjusted in the field to retrim for the aging effects of the crystal and its associated circuitry.

The circuit in Figure 10 uses a NOVRAM memory/ DAC combination to provide the trim voltage for a varactor.

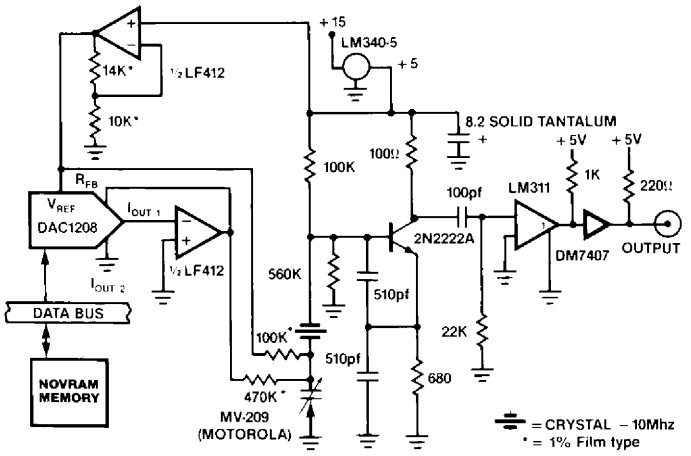


Figure 10) Tuneable crystal oscillator.

A varactor is a diode whose capacitance is a function of the applied voltage. This varactor in series with the crystal provides the actual trim function. The fixed operating point for the varactor is supplied through the 100K resistor. Variable bias for the diode is supplied by the DAC through the 470K resistor. Figure 11 shows that a 50 PPM frequency trim range is achievable with the 12-bit DAC used.

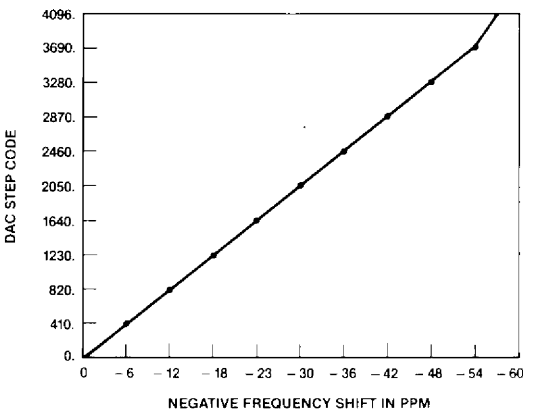


Figure 11) Tuning range of the tuneable crystal oscillator of Figure 10.

The frequency shift is down so it is recommended to specify the crystal approximately 25 PPM higher than the desired frequency. Initial trimming and re-trimming is easily accomplished by changing the DAC address settings as stored in the NOVRAM memory.

Software Programmable Voltage Reference

Many systems (such as DVMs, test equipment, data acquisition systems and most forms of measurement and control apparatus) require a voltage reference that places a limit on total system performance. Figure 12 shows how a NOVRAM memory/DAC combination can provide a means of adjusting the ouput of a precision 10 volt reference.

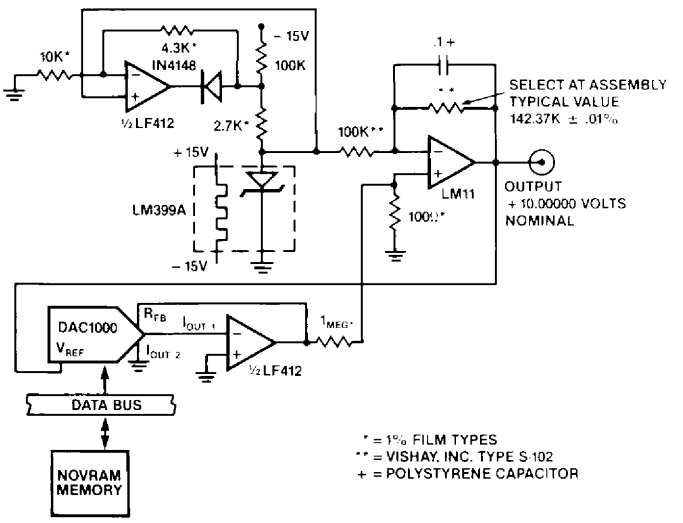


Figure 12) Software programmable voltage reference.

An LM399A 6.95 volt reference is used in a bootstrap configuration to supply bias to the LF412 amplifier which in turn drives the LM399A. The 100K resistor insures start-up. The reference supplies bias to the LM11 amplifier, which supplies the circuit's output.

The NOVRAM memory/DAC-1000 combination supplies an offset voltage for the LM11 of 1 millivolt full scale in 1 microvolt increments. The 0.1 µF capacitor insures dynamic stability and low noise at the LM11 output. To calibrate the output to within 1 microvolt, one sets the DAC to half scale and selects the feedback resistor of the LM11 until the output is within a few hundred microvolts of the desired value. Then the RAM portion of the NOVRAM memory is exercised, providing new inputs for the DAC until the desired value is achieved. This setting is then stored in the EEPROM portion of the NOVRAM memory. If a wider trim range is desired, the 1 megohm resistor can be reduced, but this degrades the setpoint resolution appropriately.

Self-Calibrating, Interchangeable Probe Thermometer

A standard industrial temperature sensor with high linearity and long term stability is obtained using platinum resistance temperature detectors (RTDs). The RTD is specified in terms of its resistance at 0° centigrade (as this is a function of the manufacturing process), while the gain slope is relatively constant from unit to unit. A constant gain amplifier with an offset to compensate for the changing impedance at 0° centigrade is shown in Figure 13.

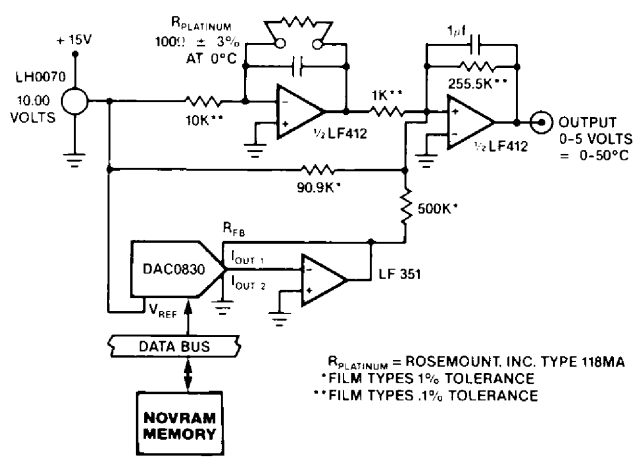


Figure 13) Self-calibrating, interchangeable probe thermometer.

The NOVRAM memory/DAC combination is used to modify the offset voltage of the amplifier to allow full interchangeability of probes in the field.

The plantinum RTD shown has a ±3% tolerance at 0°C (±7.5°C) and is driven with a 1 milliampere constant current source by placing it in the feedback loop of the LF412 amplifier. The constant current is provided by the 10 volt reference IC. The amplifier output will be a linear function of the sensed temperature at the RTD. The 1 µF capacitor limits noise pick-up and also insures that the RTD, a wirewound device with parasitic inductance, does not cause amplifier oscillations. The second half of the LF412 provides a fixed gain to the signal.

The 90.9K resistor provides a current to the summing junction of the amplifier to move beyond the correction for the worst case sensor. The NOVRAM memory/DAC pair then pull enough current from the summing junction to correct for the inserted RTD. Over a 0°C to 55°C range, this circuit is accurate to within ±0.25°C while allowing the use of probes with a ±7.5°C tolerance specification.

Automatic Scale Calibration

The scale normally does not worry about a zero-level reading from its sensor, since it may have a wide variety of items on the platform such as wrapping paper or containers. An algorithm is usually required to automatically zero the scale before loading the material into the container. The transducer used in scale applications has a large variation in gain slope which must be corrected before shipping the scale or when changing the load cell. Figure 14 shows a circuit for providing these necessary corrections for the gain slope of the sensor.

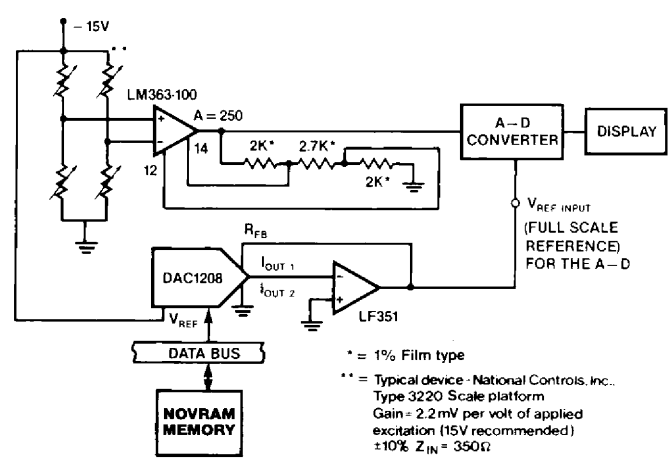


Figure 14) Automatic scale calibration.

The transducer shown, combined with the fixed gain amplifier, can produce outputs of 7.424 volts to 9.075 volts for full load depending on the transducer selected. To bring this result to the required value, the NOVRAM memory/DAC combination is used to vary the reference of an analog-to-digital converter. Since the DAC and the platform bridge are driven from the same supply, the measurement is ratiometric and no stable voltages are necessary. As the −15 volt supply changes, the readout on the display will not vary. To calibrate a new platform, the scale is first zeroed out using the internal algorithm and then a fixed known weight is added to the platform. Then the NOVRAM memory/DAC unit is exercised until the correct readout is obtained. This calibration can be called from the scale's keyboard. Security for this adjustment can be in a software access code which is also stored in the NOVRAM memory.

Gain Trimming For Photomultiplier Tube

The last example handles gain variations in a slightly different manner. The gain of a photomultiplier tube varies over time, temperature and power supply for a given input level. The output is a current from a high impedance source. A circuit to trim the changing gain is shown in Figure 15.

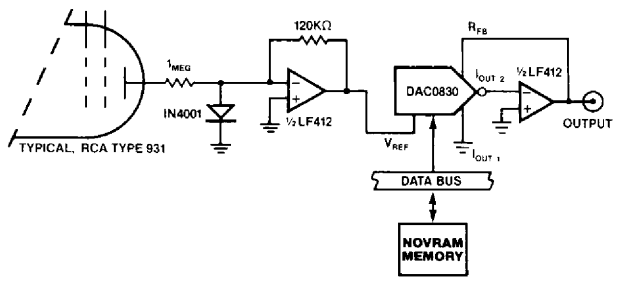


Figure 15) Gain trimming for photomultiplier tubes.

This current is converted to a voltage by the amplifier on the left side of the figure. For a full scale current of 100 μA, the output voltage of this amplifier is 12 volts. This voltage output is used as the reference input for a NOVRAM memory/8-bit DAC combination which amplifies the reference from $^1/_{256}$ through 1 depending on DAC setting. This gain can be varied in steps of $^1/_{256}$. The currents out of the photomultiplier tube are normally accurate to only 1% once the calibration is complete. Some applications, however, may require smaller steps in resolution. If this is the case, one could use a 10-bit DAC. Another method of obtaining fine resolution is to use an 8-bit DAC connected as shown in Figure 8 in place of the DAC arrangement of Figure 15. If the feedback resistor of the left-hand amplifier is changed to 50K, very fine tuning (around a voltage gain of 2) is possible. This gain of 2 is established when the DAC is set for midrange of the digital value which gives a fine tuning range.

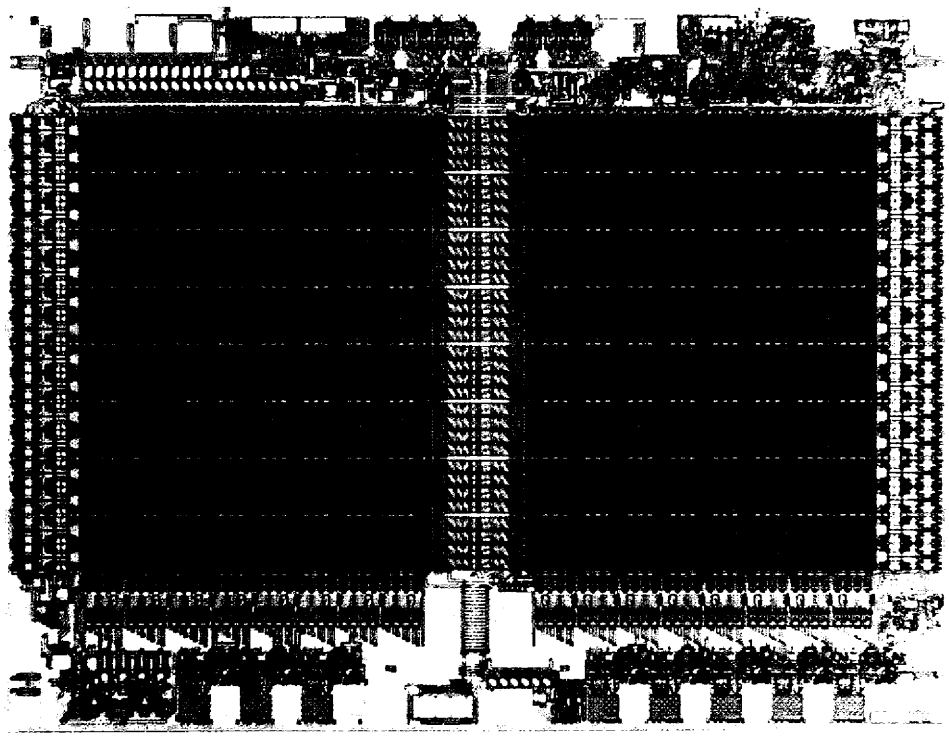
Conclusion

This application note has shown how the NOVRAM memory can be used to replace commonly used mechanical components such as trimmers and DIP switches. This replacement improves reliability and reduces service costs for recalibration and resetting. In addition, actual equipment costs can be reduced.

Once a single NOVRAM memory is in the system, it is easy to include additional features in the unused portions. For example, a system designer could use a NOVRAM memory to replace DIP switches used for configuration data and then place calibration data for DACs in the unused memory. If even more unused space exists, storage of other desired data such as ID numbers or a service log would be possible.

The possible uses of NOVRAM memories are limitless. The designer is encouraged to build upon the ideas presented by this application note.

NOVRAM is a trademark of Xicor, Inc., for its nonvolatile RAM devices.



THE WORLD'S EASIEST-TO-USE EEPROMS ARE HERE

BY GEORGE LANDERS

Introduction

The EEPROM has been available for several years and performs the very useful function of storing data or programs on a nonvolatile basis, while still allowing alteration of that information in the system. Many designers have wanted to use EEPROMs, but have been limited by the support circuitry needed by currently available products. With the announcement of 'The World's Easiest-To-Use EEPROM', Xicor has removed these limitations. This new product family operates with the simplicity of a standard static RAM.

Much has been said about the ease-of-use of EEPROMs. Most EEPROMs are indeed very easy to use during the read operation but, when it comes to the write operation, the term 'easy-to-use' applies only loosely. All EEPROMs introduced prior to the Xicor components described in this application note require one or more of the following during the write operation:

1. One or more high voltage power supplies may be required in addition to the +5 volt power supply.
2. A specially shaped high voltage pulse may be needed.
3. Addresses and data may be required for the entire write time.
4. It may be necessary for data stored in an address location to be preconditioned before writing new information.
5. Additional timing components such as capacitors may be necessary.

The Xicor EEPROMs require none of these for operation, and exhibit the following features:

1. Only a single +5 volt power supply is required for any operation including the write operation.
2. Only TTL level signals are required to control the part.
3. The write operation requires that addresses and data be stable for less than 200nsec to initiate the self controlled 10msec internal write cycle.
4. The write operation accepts random data to be changed to random data with no preconditioning.
5. An optional mode is available that allows the Xicor parts to plug into an existing socket for some of the older EEPROMs requiring shaped high voltage pulses.

The Xicor EEPROM Family

The Xicor family of EEPROMs presently consists of two pin-compatible members: the X2816A, which is organized 2048 x 8 and the X2804A, which is organized 512 x 8. Figure 1 shows the pinouts of these two components.

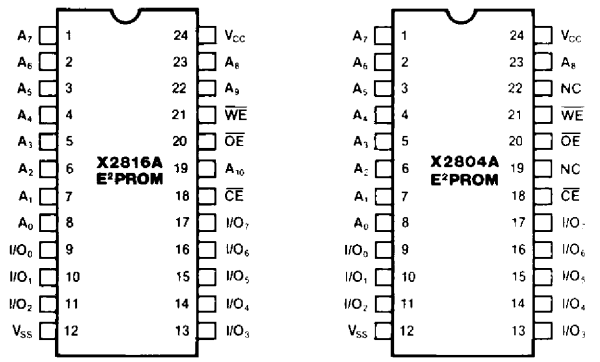


Figure 1) Pinouts of the Xicor family of EEPROMs.

Parts conform to the JEDEC standard pinouts for byte-wide memories. These two different chips use the same design rules. The smaller chip was designed in response to many requests for a smaller, more cost effective part for those applications requiring fewer bytes of storage.

The X2816A and the X2804A are manufactured using the same proven process used in Xicor's popular 5 volt programmable NOVRAM™ memories. The process is a reliable n-channel floating gate MOS technology using triple polysilicon. The method of data storage utilizes charge trapped on a floating gate, similar to the popular 2716 UV-EPROM. The charge is added and removed from the floating gate utilizing Fowler-Nordheim tunneling, enhanced by textured polysilicon surfaces. This enhancement allows fabrication with thicker oxides than other EEPROMs which don't employ textured emission surfaces. A more thorough treatment of positive reliability and data retention implications of textured surfaces is available in Xicor Reliability Report RR501.

Operation of the Xicor EEPROMs

The X2816A will be used to demonstrate the use and features of the Xicor EEPROM family. The X2804A works identically to the X2816A and fits all those applications that require smaller segments of memory. The X2816A uses a three line control structure to ease interface requirements. These three lines are:

$\overline{CE}$: Chip Enable is activated (LOW) whenever it is desired to access the part for a read or a write operation. When $\overline{CE}$ is HIGH the power is reduced to a standby level.

$\overline{WE}$: Write Enable is activated (LOW) whenever it is desired to write new data into a byte. $\overline{OE}$ must be HIGH to allow a write to start.

$\overline{OE}$: Output Enable is activated (LOW) whenever it is desired to read data from a byte. This eliminates all chances of data bus contention.

The mode selection chart in Figure 2 shows the required setting of the control signals to select the various modes of operation.

$\overline{CE}$	$\overline{OE}$	$\overline{WE}$	MODE	I/O	POWER
H	X	X	Standby	High Z	Standby
L	L	H	Read	D_{OUT}	Active
L	H	L	Byte Write	D_{IN}	Active
L	H	H	Read and Write Inhibit	High Z	Active

Figure 2) Mode selection chart for the Xicor EEPROM family.

The one possible state of the inputs not shown (all three control lines LOW) is also an inhibit mode.

Read Cycle

The read cycle for the X2816A is shown in Figure 3.

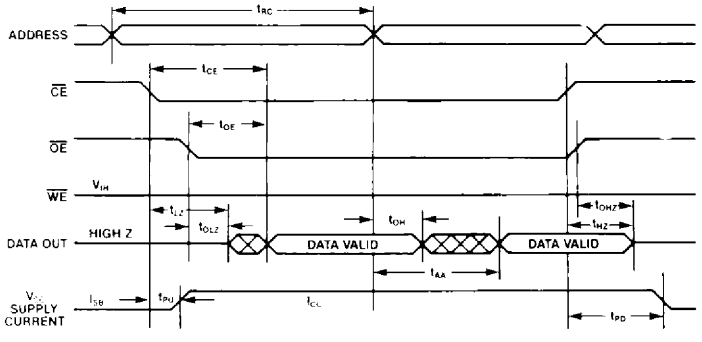


Figure 3) Timing diagram for the Xicor EEPROM read cycle.

This cycle is totally static. It is level sensitive and requires no clocking. Addresses must be stable for the entire read cycle. If $\overline{CE}$ and $\overline{OE}$ are both LOW, a change of address produces new data on the output buffers. The $\overline{CE}$ pin also controls the power level. When $\overline{CE}$ is HIGH, the power dissipation is at a standby level increasing to operating level only during the time that the $\overline{CE}$ pin is LOW. Most EEPROMs on the market operate in a similar manner.

Write Cycle

The byte write cycle on the Xicor X2816A gives new meaning to the term 'easy-to-use' regarding EEPROMs. A standard microprocessor write cycle, a single +5 volt power supply and TTL level signals are the only requirements to write new data to a selected byte. No other hardware support is necessary. Most systems will not require extra wait states for this operation, because times required are short. All timing can be completed within 200nsec. The addresses, data and controls can be removed entirely as the chip latches addresses, data and the write command at the end of the 200nsec. The X2816A takes 10msec to complete the internal byte write cycle but, once initiated, is totally self timed and does not use the data bus. This latching feature allows the microprocessor to use the bus for other purposes during these self timed write cycles. During the internal write cycle, power consumption is at the operating level.

All byte write cycle timing is referenced to an internal write pulse, which is generated by the simultaneous LOW of both $\overline{CE}$ and $\overline{WE}$. Figure 4 shows the internal write cycle timing for all possible combinations of $\overline{CE}$ and $\overline{WE}$.

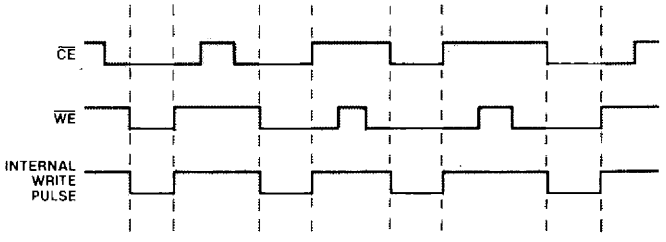


Figure 4) The internal write pulse is generated by the simultaneous LOW of both $\overline{CE}$ and $\overline{WE}$.

The internal write pulse is initiated by the last edge of $\overline{CE}$ or $\overline{WE}$ to go LOW and is terminated by the first of those two edges to go HIGH. The specific relationship of each of the two symbols is not important. Timing depends on the last edge down and the first edge up.

Figure 5 shows the byte write cycle for the X2816A with the internal write pulse used as the timing reference.

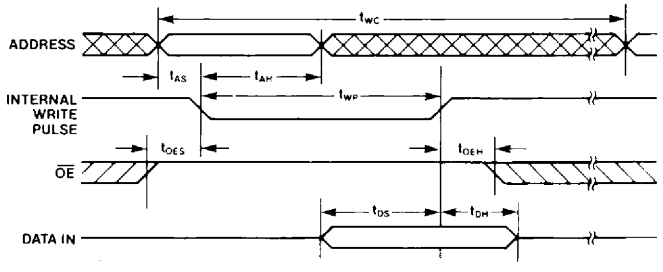


Figure 5) Timing diagram for the Xicor EEPROM byte write cycle.

The addresses are latched by the falling edge of the internal write pulse, while data is latched by the rising

edge. Once the addresses and data are latched on-chip, all further tasks are self timed. The internal byte write cycle, averaging 4 to 6msec, takes a maximum of 10msec to complete. Any attempt to read the X2816A during the internal write cycle will result in open state outputs. The last internal write cycle action releases control of the output buffers to the three control lines. It is necessary that the power supply remain within specification during the entire internal write cycle time. Once initiated, the internal byte write cycle cannot be terminated by any signal and will complete under its own control.

The write cycle is inhibited by holding either the $\overline{OE}$ pin LOW or $\overline{CE}$ HIGH. Write protection can be achieved during power-up and power-down by holding $\overline{OE}$ pin LOW. If $\overline{OE}$ goes LOW after the write has started, the write cycle will complete.

Unlike most other EEPROMs, it is not necessary to precondition the byte prior to writing new data. The byte write cycle allows the change from random data directly to random data. Separate byte erase and program cycles are unnecessary. To change data, one writes as in a static RAM.

It is necessary to refer to the data sheet for the latest timings; however, Table 1 shows the byte write cycle timing for the 300nsec part.

Symbol	Parameter	Limits		Units
		Min.	Max.	
t_{WC}	Write Cycle Time	10		ms
t_{AS}	Address Set-Up Time	10		ns
t_{AH}	Address Hold Time	120		ns
t_{WP}	Write Pulse Width	150		ns
t_{OES}	Output Enable Set-Up Time	10		ns
t_{OEH}	Output Enable Hold Time	10		ns
t_{DV}	Data Valid Time		1	μs
t_{DS}	Data Set-Up Time	50		ns
t_{DH}	Data Hold Time	10		ns

Table 1) Timing requirements for the 300 nsec X2816A.

Optional High Voltage Write Cycle

Although high voltage power supplies and/or pulses are not necessary for the X2816A, Xicor has added an additional mode for customer convenience. This is the optional high voltage write mode. The Xicor X2816A option provides a plug-in replacement for the Intel 2816, NMC 2816, NMC 9716 and similar devices, with no extra interfacing required. The Xicor X2816A eliminates complicated timing required by similar devices as shown in Figure 6.

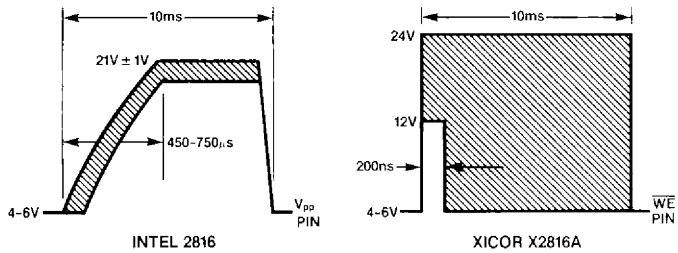


Figure 6) Comparison of timing and level requirements of the high voltage pulse between the Intel 2816 and the Xicor X2816A. The shaded portion indicates the operating range of each part.

The X2816A uses no current from the high voltage pulse. Bringing pin 21 above 12 volts is sufficient to trigger the internal byte write cycle. All actual timing is executed by the part itself. Since the X2816A can write directly from random data to random data without an intervening erase, the user can make software changes to shorten the code by removing the preceding byte erase cycle, although this is not necessary if complete compatibility is desired.

Optional Chip Erase Cycle

Another mode available on the X2816A as a user option is the chip erase cycle, during which all bytes are simultaneously changed to 1's or HIGH. Since the part does not require that data in the byte be preconditioned, the only use for a chip erase is for setting the unused bytes to a known HIGH state. This operation requires that high voltage pulses be applied to $\overline{OE}$ and $\overline{WE}$. The high voltage provides a third state on the $\overline{OE}$ pin, enabling an extra function without adding pins. This function is provided tc make the X2816A compatible to all modes of operation on the high voltage 2816. It is not a required mode for the majority of applications. No time is saved by using the chip erase mode. It takes 20sec total to write every byte on the Xicor X2816A, regardless of whether the chip is totally erased or not. On the other hand, the Intel or similar 2816 requires 20 sec to completely write an erased chip and up to 40 sec to overwrite a chip with random data.

Implications of the X2816A Improvements

In the introduction of this application note there were five major features listed for the Xicor X2816A. These were:

1. 5 volts only
2. TTL level signals
3. Short signals

4. Random data to random data
5. Optional high voltage mode

The implications of each one will be discussed in this section. One of the major goals of Xicor is to design as much support requirement into the chip as practical.

5 Volts Only

The only power supply requirement for the Xicor X2816A is a single +5 volt supply. No other power supply is required, either by the chip or any support circuitry that may be used. This makes the board design considerably easier and less costly. Many EEPROMs require one or more high voltage supplies to provide necessary signal levels for writing. This high voltage is applied either as dc voltage directly to the chip or as the source of the levels for an external pulse. Xicor has chosen to make the designer's task easier by including the necessary high voltage levels for tunneling on-chip. This high voltage, generated on command during the internal byte write cycle, is totally transparent to the user.

High voltage is present on the X2816A only during the time of an actual internal byte write cycle. This considerably reduces the quiescent field strengths, thereby decreasing the stress on internal nodes and oxides. Figure 7 shows a comparison of the on-chip voltages using internal or external high voltage pulses as opposed to a high voltage power supply.

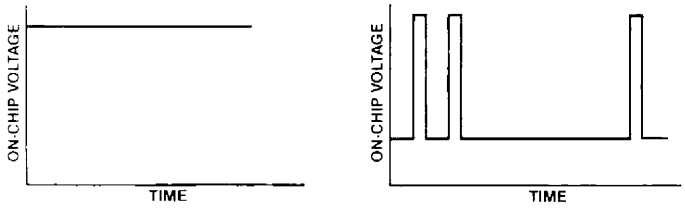


Figure 7) Comparison of on-chip voltages between an EEPROM using a) high voltage power supplies and b) internal or external high voltage pulses

Since constant high voltage on the chip has deleterious reliability implications, an EEPROM with a high voltage pulse generator on-chip offers two benefits. These being the elimination of boardspace and design time for external components, and reduced stress-time levels on the chip to avoid a potential reliability hazard. In addition, parts requiring multiple power supplies can lead to power supply sequencing problems. If the high voltage supply is present without the lower voltage supply the part may be destroyed.

TTL Level Signals

Some EEPROMs require specially shaped high voltage pulses. These pulses usually last the entire period of the write cycle. The Xicor X2816A provides on-chip all necessary high voltage functions totally transparent to the user. When an external high voltage pulse is required by an EEPROM, it may require controlled rise and fall times (which may be different, as well as requiring unusual signal levels). Taking this external pulse requirement away saves design complexity, costs and reliability problems.

Short Signals

Some EEPROMs require that address and data be held valid during the entire write cycle. This may require the use of wait states or, more usually, external latches to capture the addresses and data as well as the control signals. The Xicor X2816A requires only a 200nsec signal to initiate the internal byte write cycle. Once started it will complete by itself. The elimination of external latches saves considerable cost and boardspace.

These first three feature advantages can be seen in the evolution of EEPROMs shown in Figure 8.

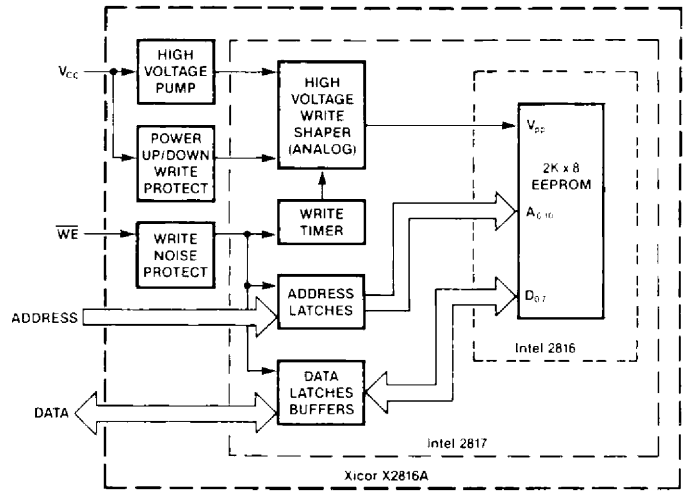


Figure 8) Evolution of easy-to-use EEPROMs from the Intel 2816 and 2817 to the Xicor X2816A.

From requiring an external high voltage pulse and latches, to a high voltage power supply, this evolution shows finally, no external support requirement (other than possible optional buffering in layer systems).

Internal latches also allow tremendous write time savings for large systems. Different data may be programmed into multiple parts. Multiple parts may be programmed in parallel instead of sequentially. In a system using 100 parts without latches, 2000sec would be required to write all locations, as opposed to 20sec for a system using the X2816A, because, while the first part is on its internal write cycle, the other parts may be written simultaneously.

Random Data To Random Data

Some EEPROMs require that the byte be preconditioned to a specific state before writing new data into the byte. Again, this is not the case for the Xicor X2816A. All byte erase requirements are performed transparently as part of the internal byte write cycle. When data must be preconditioned, a full write cycle to the required starting data must be performed. This leads to extra software to reduce as much time as possible. An example of possible extra software is shown in Figure 9.

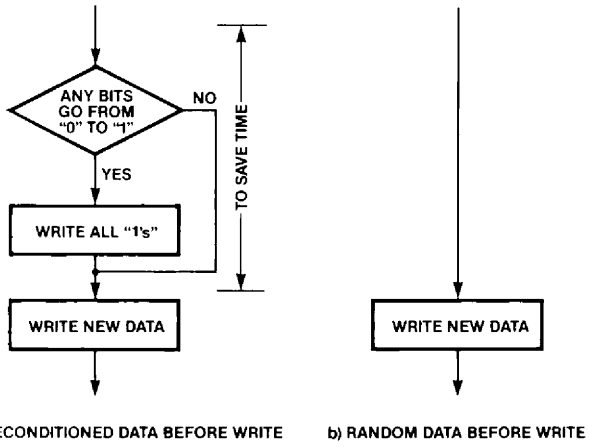


Figure 9) Comparison of software requirements for minimum write time systems between a) preconditioned data and b) random data prior to the write cycle.

Optional High Voltage Mode

In addition to all aforementioned benefits, the Xicor X2816A can also substitute directly for the Intel or similar 2816 as shown in Figure 10.

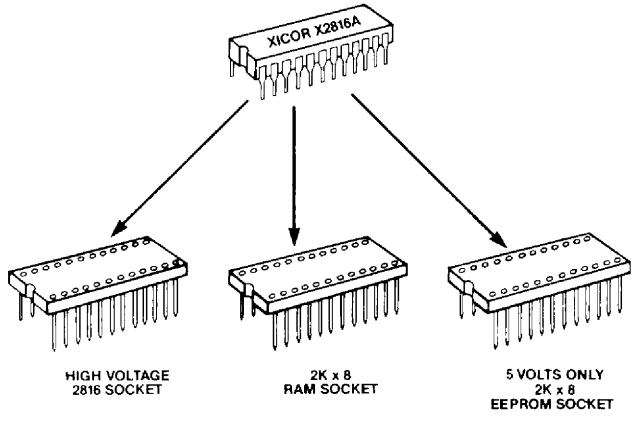


Figure 10) The Xicor X2816A works equally well in three environments.

Designers have the option of using the Xicor X2816A in a standard low voltage RAM socket, or a high voltage EEPROM socket, designed for the high voltage 2816. No other EEPROM offers such flexibility. It is important to keep pin 21 above 2.4 volts in this mode. Otherwise, a second false write can be triggered at the completion of the high voltage pulse.

The X2816A Can Be Used Like A RAM

The only EEPROM currently able to plug into a standard 2K x 8 RAM socket is the Xicor X2816A. The X2816A will operate with all signals provided by that conventional socket. The device becomes fully functional with a simple read or write operation. The only restriction being to wait at least 10msec, or until the internal byte write cycle is completed, whichever comes first, before starting another EEPROM cycle. It is possible, however, to use the bus for other tasks during this wait.

Programming Optimization

Several methods can be used to determine when one can command another EEPROM cycle. The system designer can use either a software or hardware time-out or use an interrogation scheme. The time-out can be derived from a software loop, a 16.7msec AC marker or some other timed interrupt.

It is possible to speed up writing blocks of data into the X2816A by using the interrogation method. Since the average part completes its internal byte write cycle in 4 to 6msec, it is quicker to initiate the next write cycle as soon as control of the output buffers is turned back to the $\overline{CE}$ and $\overline{OE}$ pins. The output buffers go to the high impedance mode during write and can return to the low impedance mode, under control of $\overline{CE}$ and $\overline{OE}$, only after the internal write cycle completes. In this manner, the part can signal the system that a write cycle has finished.

Interfacing To Microprocessors

Since the Xicor X2816A will operate in a 2K x 8 RAM socket, interface to a microprocessor is simple. The connections to an 8085 microprocessor are shown in Figure 11.

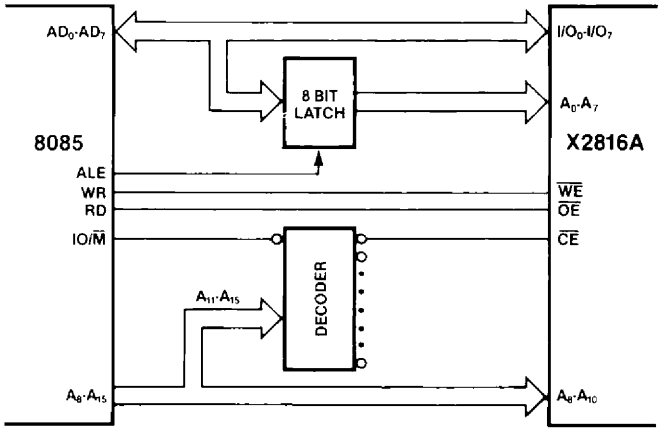


Figure 11) Interfacing the X2816A to the 8085 microprocessor.

The 8 bit latch captures the low order addresses from the combined address/data bus. The latch and a decoder are the only additional circuitry required for this interface, normally found in any system using standard RAM or ROM. These are the same connections required by a fully static RAM.

Conclusion

Each improvement made in EEPROMs provides extended potential applications. By eliminating the need for any hardware support, the Xicor family of EEPROMs makes it as easy to alter nonvolatile data as it is to write to a static RAM. Clearly, this unprecedented ease-of-use widens standard EEPROM applications through lower costs and reduced boardspace. A 5 volt power supply requirement improves reliability and lowers the damage risk to other components, unlike EEPROMs that require a fixed high voltage power supply. In addition to download type applications, where a remote system is reprogrammed over a phone line, these smarter EEPROMs will eventually be used in machines that adjust themselves to their particular environment. The Xicor EEPROM family allows designers freedom from former limitations.

Xicor Application Brief

Vol. 1, No. 1 July 8, 1983

NONVOLATILE DATA INTEGRITY: Inadvertent Write/Store Elimination

By Reggie Huff

Xicor's exciting nonvolatile memory products are backed up by designed-in protection features which ensure data integrity. These include:

- Onboard V_{CC} Sensor
  All operations inhibited when $V_{CC}<3.0V$.
- Noise Filter
  The E^2PROM has an additional feature which blocks noise spikes on control lines.
- Orderly Power Transition
  The device will not self-generate inadvertent write/store operations.
- Write/Store Inhibited Control Pins
  Multi-pin write/store command signal requirements provide both data security and design flexibility.

With Xicor nonvolatile memories:

Data is maintained through power-on, power-off, power-up, power-down, system crash, and the entire range of system conditions when some simple design rules are observed. Often nonvolatile system designers are frustrated by inadvertent system command signals during power-up and power-down operations. Being generally nonperiodic in nature, these elusive culprits can lead the designers to the false conclusion that the memory device is malfunctioning. This, however, is rarely the case. The system is more often sending an unintended write/store command. This problem is easily resolved as shown in this application brief.

GIGO[1] Going to Sleep

Just as a person falling asleep at the wheel can inadvertently command his vehicle into an undesirable situation, digital systems transitioning from normal operation to a power-off state or vice versa can distribute random data, addresses, and control signals along the way.

Since Xicor nonvolatile memories accurately and reliably store data as instructed, garbage stored at power-down will be impeccably retained and available upon power-up. (That's nonvolatile GIGO.)

[1]GIGO is an acronym popular in the computer world for "Garbarge In Garbage Out".

Protection-Conscious Design

Data integrity is a major criterion with Xicor products and several superb features were designed into Xicor's memories to ensure it.

- V_{CC} Sensor
  An onboard sensor establishes a threshold supply voltage of 3.0V below which the write operation on E^2PROMs and the store operation on NOVRAMs are blocked. Above this voltage, write and storage operations are available and therefore must be protected from unplanned instructions.

- Orderly Supply Transitions
  As a system powers up or down, the possiblility of unintentional, internally generated control signals increases dramatically. The Xicor nonvolatile memory family has designed-in protection to eliminate self-generated write/store commands.

- Noise Filter
  An additional feature designed into Xicor's E^2PROM family is a noise filter to prevent glitches on the $\overline{WE}$ line from initiating a write cycle. This feature ignores pulses of less than 20ns duration so that noise spikes are not misconstrued as write commands.

- Write/Store Inhibition Control Pins
  The entire Xicor nonvolatile memory line requires combinational control pin conditions in order to execute a write/store command. The NOVRAM has two essential pins and the E^2PROM has three essential pins involved in the store/write cycle initiation. By disallowing any one of the required pin conditions, the user can prevent unplanned nonvolatile data changes so that data integrity is maintained.

Write/Store Pin Conditions

$\overline{ARRAY\ RECALL}$[2]	$\overline{STORE}$	
X	H	Store Operation Disabled
L	X	Array Recall Blocks Store Initiation (See Note Below)
H	L	Store Operation Executed[3]

NOVRAM Family

$\overline{CE}$	$\overline{OE}$	$\overline{WE}$	
X	X	H	($\overline{WE}$) Write Inhibit
X	L	X	($\overline{OE}$) Write Inhibit
H	X	X	($\overline{CE}$) Write Inhibit
L	H	L	Write Operation Executed[3]

E^2PROM Family

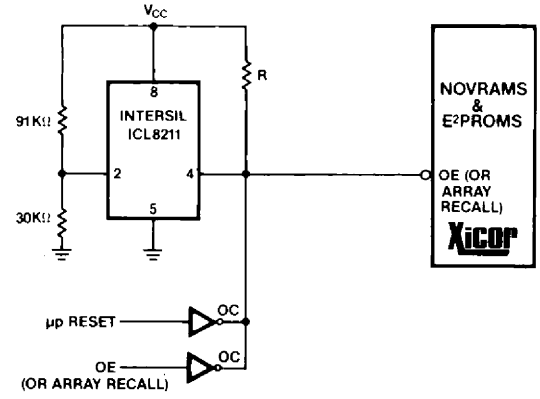
[2]ARRAY RECALL blocks all control inputs but does not halt a store in process.
[3]These are the only conditions allowing nonvolatile data change.

Solution I- "Hold-Low" Protection

The simplest solution is to pull the $\overline{OE}$ (or ARRAY RECALL) to a logic "0" whenever the supply voltage is below the (5.0-10%) system threshold.

The Intersil[4] ICL 8211 programmable voltage reference is an inexpensive 8-pin mini DIP which will sense a selected voltage threshold and output a logic "0" when the supply is below that threshold. Conversely, as the sensed voltage rises above the selected threshold, the ICL 8211 outputs a logic "1" following its supply voltage level.

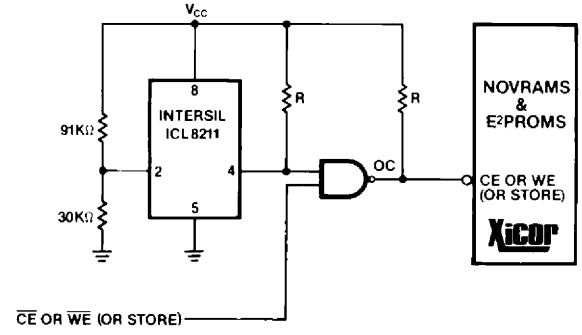
"Hold Low" Protection

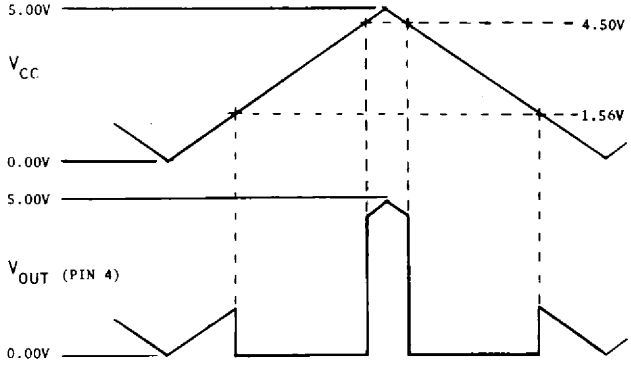


Solution II - "Hold-High" Protection

The second method of data protection during power supply transitions is to keep the NOVRAM $\overline{STORE}$ pin (or the $\overline{WE}$ and/or $\overline{CE}$ pins in the E^2PROM family) near the power supply voltage. By preventing the low condition of these pins which is necessary for a write or store operation, inadvertent stores will be eliminated.

"Hold High" Protection





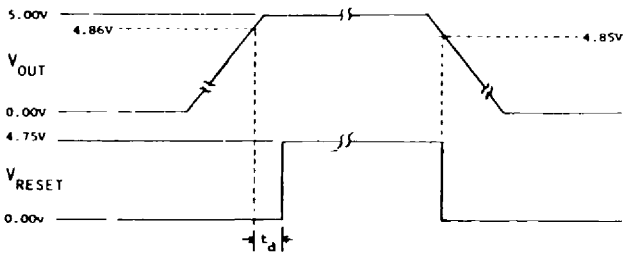
INTERSIL ICL 8211 PROGRAMMABLE VOLTAGE REFERENCE SUPPLY AND OUTPUT WAVEFORMS

Figure 1

The graph in Figure 1 shows the performance of the Intersil ICL 8211. The top plot is a sawtooth which is connected to "5V supply" as shown in the Solution I and Solution II schematic diagrams. The bottom plot is the output of the ICL 8211. Note that when the supply is above 4.50V, the ICL 8211 output tracks it at logic "1". When the supply sawtooth is below 1.56V, the ICL 8211 output tracks the power supply. However, since the Xicor memory family has internal protection inhibiting write/store operations when V_{CC} is below 3V, no inadvertent write/stores will occur in this range. In the critical range between 3V, where internal protection stops, and 4.5V, where normal operation begins, the ICL 8211 insures a 0V output.

[4]See Intersil ICL 8211, ICL 8212 Programmable voltage reference data sheet in Intersil Data Book.

As an alternate approach to the ICL 8211, some designers may prefer to incorporate the SGS L487. This device is a 500 mA precision 5V voltage regulator which includes an open collector power-on, power-off reset output pin, which can protect the nonvolatile memories just as the ICL 8211 does. The timing diagram in Figure 2 shows the voltage on this reset output pin as the supply voltage transitions through power-up and power-down.



SGS L487 PRECISION VOLTAGE REGULATOR
OUTPUT AND RESET WAVEFORMS

Figure 2

Nonvolatile Data Integrity Intact

The potential problem of nonvolatile data changes during power up/down operations is addressed in this application brief. The problem is easily eliminated. A unique multiple-control-pin combination is required for the initiation of nonvolatile data alterations in each Xicor E^2PROM memory. By ensuring that during periods when the operating supply voltage is below the specified minimum, this unique control pin combination is not asserted, and inadvertent nonvolatile data changes are eliminated. Two possible solutions are described in the Solution I and Solution II sections using two different parts (Intersil 8211 and SGS L487) to implement the solutions.

These solutions may not be appropriate in all applications and certainly do not constitute an exhaustive list of effective solutions. The Xicor applications engineers are happy to discuss your situation and to assist you with any application questions you may have.

Xicor Application Brief

Vol.1, No.2 July 23, 1983

AUTOMATIC PAGE WRITE *** $\overline{\text{DATA}}$ POLLING

Two New and Innovative Features on the 64K E^2PROM

By Reggie Huff

Xicor, the industry leader in nonvolatile memory technology, introduces two trend-setting features which are sure to establish the standards for future nonvolatile memory products. These evolutionary additions enhance the ease-of-use nature of Xicor's E^2PROM even beyond that of the industry standard X2816A (16K E^2 PROM) introduced by Xicor in 1982.

Automatic Page Write

Allows the user to write up to 16 bytes at a time using standard microprocessor write cycles. After a 200μs "write-window", the I/O buffers go into a high impedance mode so the bus is freed for other tasks. Assuming the typical write period of 5ms, the effective write time is thereby reduced to (5ms/16 bytes =) 312.5μ seconds per byte. Clearly, a quantum leap in E^2PROM write speed performance.

$\overline{\text{Data}}$ Polling

Provides a simple software technique for testing for the completion of a nonvolatile write cycle. This method requires no extra pins and no external hardware which is consistent with Xicor's user oriented design strategies. $\overline{\text{Data}}$ Polling allows the user to take advantage of the typical write times without sacrificing compatability with other byte-wide memories. This method also allows the last two unconnected pins in the standard 28 pin package (pins 1 and 26) to be reserved for higher order address lines in the next generation E^2PROM.

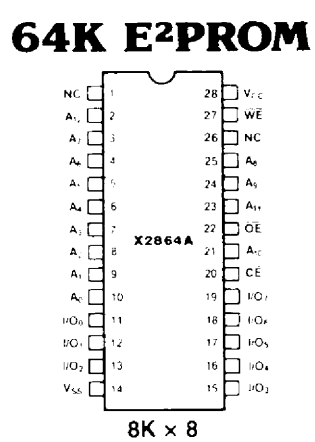


Figure 1

The X2864A package shown in Figure 1 reveals the two unconnected pins on the 28 pin package. This allows for board designs which accommodate the 64K E^2PROM to upgrade to 256K with the only change being the addition of high order address signals. The same 28 pin footprint can also house a lower-justified 16K E^2PROM - X2816A - with the simple addition of two jumpers (connecting pins 23 to 27 and 26 to 28). See Sales Brief, Vol.1, No. 1.

Automatic Page Write — Specifics

The Automatic Page Write feature designed into the X2864A brings the E^2PROM to yet another level in its rapid evolution. Xicor's popular on-board address and data latch has been expanded to an array of 16 bytes, so that the user may write up to a full page (16 bytes) in each 5ms nonvolatile write-cycle. No special signals are required to set up the operation, and the procedure is executed using standard microprocessor write-cycles.

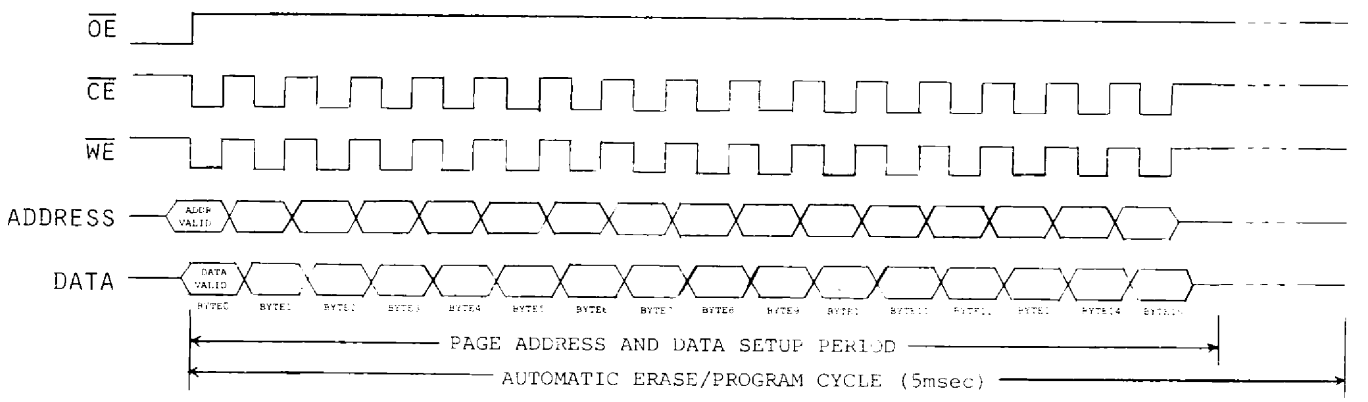


Figure 2

Figure 2 shows the timing diagram documenting this feature. The Automatic Erase/Program Cycle is initiated on the last falling edge of either $\overline{CE}$ or $\overline{WE}$ providing that $\overline{OE}$ is high. At this time, the address bits are locked in defining the page to which the subsequent 15 bytes may be written.

Note: A page is specified by the eight most significant address bits. Therefore, the addresses of all bytes written in a single write-cycle may differ only in the four least significant bits.

A 200μs window is nested within the Automatic Erase/Program Cycle and is designated as the page address and data set-up period in Figure 2. This window defines the time during which the X2864A may be written and as many as 16 standard microprocessor write cycles may be executed. Of course, between write cycles, the microprocessor may access other devices in order to fetch the data to be stored in the E^2PROM. The use of this feature is optional. Any number of bytes between one and sixteen may be written and those may be in any order.

At the end of this 200μs window, the X2864A begins its internal write sequence by freeing the busses for other system use. From this point until the completion of the nonvolatile write cycle, the device will ignore control pin inputs. This allows the system to continue task performance while the X2864A busies itself internally performing the Automatic Erase procedure and copying the data from the latches into the appropriate locations in the E^2PROM array.

Several questions are naturally raised at this point:

1. What happens to the bytes not addressed on a page which is being written to?
2. What happens if the page address is changed during a Write Cycle (i.e., bytes are written to different pages)?
3. What if 200μs is too short to fetch and page write all 16 bytes?

A1:The bytes which are located on the page being written to and which are not actually modified will contain the same data before and after the nonvolatile write cycle. The user may write any number of bytes up to 16 in a given cycle, and the bytes need not be sequential since within a given page, they may be written in any order. If a particular location is written to more than once on the same page, then the last data written to that location is the data which will be copied into the E^2PROM.

A2:The requirement that all bytes be contained within the address variation of the four least significant address bits is a strict one. If any of the higher order address bits are changed during the page address and data set-up period, the resultant E^2PROM content is not predictable. That situation must simply be avoided in the software.

A3:The 200μs (guaranteed minimum) timeframe was selected after careful consideration of the wide variety of microprocessors available. This window was chosen as optimal and allows the vast majority of microprocessors to utilize the Automatic Page Write feature completely. Particularly slow microprocessor systems can enjoy the benefits of this feature by minimizing machine cycle overhead in the E^2PROM write routine or by writing 4 or 8 bytes in each cycle instead of 16.

The Automatic Page Write feature provides the user with a simple method of increasing the effective write speed by a factor of 16. In order to further utilize the write speed potential of the X2864A, the user must have some way of sensing the actual completion of the Nonvolatile Write Cycle. The sensing technique is the feature called $\overline{\text{Data}}$ Polling.

$\overline{\text{Data}}$ Polling — Specifics

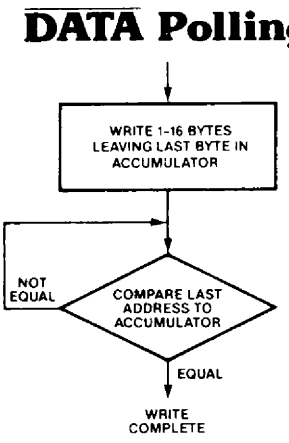
Due to the unparalleled convenience of on-chip address latches, on-chip data latches, on-chip cycle timer, and the Automatic Page Write feature, the microprocessor writing to Xicor's X2864A E^2PROM is free to handle other tasks while the memory sequences through its internal write cycle unattended. Upon completion of latching (1 to 16) byte(s), the microprocessor can direct its attention elsewhere until the maximum specified write time has elapsed. At this point, the memory may be accessed for read or subsequent write operations.

The X2864A devices normally complete the nonvolatile write cycle substantially faster than the maximum time specified on the data sheet. In order to take advantage of this faster write potential, Xicor engineers created $\overline{\text{Data}}$ Polling.

$\overline{\text{Data}}$ Polling Operation

$\overline{\text{Data}}$ Polling uses the most significant bit (MSB) of the last byte written to indicate to the microprocessor when the write operation is complete. During the nonvolatile write cycle, I/O7 is the complement of the last D7 written. Thus, the microprocessor can simply read the last byte written and compare the result with the actual data written (which was retained in the accumulator). When the values are equal, the write cycle is complete.

The flowchart below shows the minimal software necessary to "Poll" for "Data", thereby testing for nonvolatile write cycle completion.



During write cycle

- I/O_7 is the complement of bit written
- I/O_0 through I/O_6 are floating
- A compare with byte written fails

After write cycle is completed

- All bits are correct
- A compare with byte written passes

Since, prior to nonvolatile write cycle completion, the MSB is inverted, the comparison test will fail and the loop will continue. When the write cycle is complete, the comparison test will succeed and control will fall out of the loop. The user may then access the memory for the next read or write operation.

Data Polling Subroutine Examples

```
8080/8085 Microprocessor Instructions
Retain last byte written in the accumulator and the last
address written to in HL

XXX CMP M              COMPARE MEMORY TO ACCUMULATOR
    JNZ XXX            JUMP IF NO MATCH
    RET                RETURN FROM SUBROUTINE
```

```
6502 Microprocessor Instructions
Retain the last byte written in the accumulator

XXX CMP (LAST), Y      COMPARE MEMORY TO ACCUMULATOR
    BNE XXX            BRANCH IF NO MATCH
    RTS                RETURN FROM SUBROUTINE
```

Another Industry Breakthrough

The X2864A Automatic Page Write and Data Polling features provide the user with versatile application options never before possible. Xicor steps out again with on-chip features designed to meet the needs of pace-setting systems engineers. Effective E^2PROM byte-write times below 300μs using no extra hardware, no extra pins, and only two lines of code define the leading edge.

Xicor Application Brief

X2443/X2444 NONVOLATILE RAMS

The Industry's First 5V-Only NOVRAM Family Branches into Serial Interface Microcomputer Systems

By Reggie Huff

Vol.1,No.3 August 30, 1983

Xicor, the trend setting pioneer in nonvolatile memory technology, leaps ahead again with the first NOVRAM designed for port-oriented, single chip microcontrollers and microcomputers. The X2443/X2444 are highly cost-effective additions to the ever expanding writable nonvolatile memory family created at Xicor.

- **Low Cost**
- **5 Volt Only**
- **Three Pin Control**
- **Unlimited RAM Data Changes**
- **Minimal Support Circuitry Requirements**
- **Low Power**
- **Fully Static**
- **Compact 8-Pin Dip**
- **10 Year Data Retention**
- **High Security Data Protection**

The X2443/X2444 are 256 bit (16x16) serial NOVRAMs providing both static RAM accessability and E^2PROM nonvolatility packaged in a cost-effective 8-pin DIP. The devices use Xicor's proven NMOS triple-poly floating-gate thick-oxide technology to provide long term reliability. Low power Standby and Sleep modes, provide the ideal solution for remote and portable system needs. The memory can be completely controlled through the use of as few as three control signals allowing conservation of single chip microcomputer I/O lines.

The only control pins required for interface are CE, SK, DI, and DO. Three control line operation can be realized by tying DI and DO together to create a single port serial-data I/O line. The pin configuration on page 1 of the data sheet shows that all required control pins are on the same side of the space saving package. This simplifies board layout and minimizes real estate requirements.

Device operation is primarily controlled through instructions which are clocked into the DI pin as eight bit words. The instruction register is initialized upon the rising edge of CE and thereafter waits for a 'marker bit' (which is simply a '1' on DI upon the rising edge of SK). The next four data bits clocked in are interpreted as address bits identifying which of the 16 memory words will be operated upon. The last three bits clocked in are the op-code specifying which operation the device is to perform. (Please refer to diagrams 1-5 on the data sheet.)

The following seven 8-bit instructions control all the features of the X2443 and X2444 memory systems.

	Marker Bit	Address Bits	Instruction Bits
RAM READ	1	AAAA	11X
RAM WRITE	1	AAAA	011
RECALL	1	XXXX	101
STORE	1	XXXX	001
SLEEP	1	XXXX	010
WRITE ENABLE	1	XXXX	100
WRITE DISABLE	1	XXXX	000

X2443/X2444 Instruction Set

Note: There are two types of instructions; instructions having a specific address field and those with nonspecifc address fields (XXXX). The former always preceed a RAM data operation, the latter are array oriented operations affecting all address locations simultaneously.

The SK pin is a static clock input active on the rising edge. This static nature allows the clock sequence to be interrupted at any point and to then be continued at the user's convenience. If the SK pin stops toggling during an instruction sequence, the device simply halts where it is and waits indefinitely. The user, therefore, is able to clock out four or eight bits of a word, for instance, and then leave the NOVRAM waiting, unattended, until more of the word is desired. While the SK pin is inactive and CE is high, the output buffer will remain in a High-Z state. The only exception to this is the output data phase of a RAM Read instruction. In this case, the output buffer will maintain the last data bit clocked out until the subsequent rising edge of SK. This static nature allows the implementation of the serial interface with a software routine which controls an I/O port. If desired, the SK pin may be controlled by a free running clock at a rate up to 1MHz on the X2444 (0.9MHz on the X2443).

The CE pin must remain high for active operation. When CE is brought low, the instruction register is reset and the low power standby mode is selected (see I_{SB} on data sheet). This allows early termination of instructions in progress which may be useful, for example, in the event of a system crash. CE must be brought low before a new instruction will be recognized. If CE remains high, RAM Read and Write operations will continue with the clock by "wrapping around". That is to say that a word will continue to be read (or written) again and again (bits 1-16, 1-16 ...) until CE is brought low resetting the instruction register. When CE resets the instruction register, the internal bit pointer is also reset. This means that any subsequent RAM access will always start with D_O of the word addressed independent of the preceding instruction.

The RAM Read instruction consists of two parts, instruction and data. The first eight bits of the instruction specify the Read op-code and identify the word to be read with the address field. The last bit of the op-code is a "Don't Care" in order to give the controller a free cycle in which to change the port from an output to an input pin. This is sometimes necessary in combined I/O line applications. The next sixteen clock pulses each correspond to subsequent data bits. Note that the output buffer remains in a High-Z state except when delivering data in response to a Read instruction.

The RAM Write instruction follows a similar pattern as the Read. The first high data bit which is clocked in is taken to be the marker bit. The next four bits address the 16-bit word to be written, and the last three bits are the instruction op code 0 1 1. The next sixteen clock cycles write sixteen bits into the RAM at the location addressed.

The Store instruction is clocked in starting with the marker bit and continuing through four "Don't Care" bits as shown in the instruction set. Since the entire RAM array is transferred to the E^2PROM with this single command, the effect is global so the address bits are nonspecific. The X2443 and X2444 are entirely self-timed devices eliminating the need for any external timing apparatus. The on-board logic sequences through the finite state machine, automatically generating the high voltage necessary for tunneling, and places the output buffer in a High-Z state. The RAM data array is then copied to the E^2PROM and the device returned to a ready state . . . **ALL WITHIN 10ms!**

The Recall instruction is clocked in just as the Store instruction is (with nonessential address bits). The effect of this instruction is to copy the entire E^2PROM array contents to the static RAM in a single operation.

The Sleep instruction causes the static RAM to power down losing its data. The quiescent current is substantially reduced from its normal 15mA range (see I_{SL} on the data sheet), and the E^2PROM data, of course, remains secure. The only way to awaken the X2444 from Sleep Mode is through the execution of a Recall procedure. This copies the E^2PROM into the RAM and returns the memory system to its normal Standby condition.

Data Security Mechanisms

The integrity of NOVRAM memory data is essential, and Xicor engineers designed in several features to ensure it. First, after each power up, the NOVRAM must execute at least one Recall operation before it will respond to a Store command. This safety mechanism guarantees that the device will not perform an inadvertent Store while V_{cc} is stabilizing. An internal write flag is another protection factor allowing the user to lock both arrays of data against unintentional changes. The write-flag must be enabled for a RAM Write or Store operation to be executed.

The Write Enable/Disable instructions set and reset the write-flag respectively. The flag is automatically disabled upon power up, so that a Write Enable instruction is required before the device will respond to Write or Store instructions. This feature allows the user to lock and unlock the data change mechanism for added security. For maximum data protection, the flag should be set just prior to a RAM Write or Store instruction and reset immediately thereafter. (Note: The Store procedure automatically resets the flag.) Alternately, this safety feature may be bypassed to achieve software minimization by simply setting the flag as an initialization after power up and immediately after each Store operation.

Hardware Control

The last data security feature involves the $\overline{\text{Store}}$ and $\overline{\text{Recall}}$ pins which allow hardware initiation of the Store and Recall operations. The result of these operations is identical whether initiated through hardware or software control. An internal Schmitt trigger buffer on the $\overline{\text{Recall}}$ input pin makes its operation threshold level sensitive, ideal for automating a Recall-upon-power up mechanism with a simple RC circuit (see Figures 1 and 2 under 'Typical Serial Interfaces' on the X2444 data sheet). The hardware $\overline{\text{Store}}$ pin is protected with a noise filter which prevents glitches and noise spikes from activating an unintentional Store cycle (see t_{STD} on data sheet). The $\overline{\text{Store}}$ and $\overline{\text{Recall}}$ control pins initiate their respective operations independently of the CE, SK, DI, and DO pin conditions.

They can be used to automate the Store (for crash protection) and/or Recall (for Recall upon power up) operations making them transparent to the software system (see Figures 1 and 2 under 'Typical Serial Interfaces'on the X2444 data sheet). They may otherwise be tied to the supply rail effectively removing the hardware feature from consideration.

The potential conflict between hardware and software instructions is resolved by the built-in priority system:

1. **$\overline{\text{RECALL}}$**: Independent of all other pin conditions; will interrupt any operation (except a Store) in progress initiating a Recall operation. While $\overline{\text{Recall}}$ is low, all other input signals are ignored.
2. **$\overline{\text{STORE}}$**:
   a. The Recall-since-power up and set-write flag conditions must have been met in order to initiate this procedure.
   b. The $\overline{\text{Recall}}$ pin must be high.
   c. The $\overline{\text{Store}}$ pin overrides everything except a RAM Write in progress. In order to ensure the integrity of the captured data, a RAM Write instruction will not be interrupted during its data phase. Instead, the Store operation will be inhibited until the RAM Write instruction is terminated by bringing CE low. If CE is brought low during the data phase of the RAM Write, those bits of the addressed word already overwritten will be stored as new data and those bits not yet overwritten will be stored without change. Please refer to Application Brief Vol.1, No.4 for further information.

This Store hierarchy allows for a quick and orderly Store operation upon the recognition of a pending power failure (refer to Applications Note 102: Storing Data in NOVRAM Memories During Power Failure). Since the hardware $\overline{\text{Store}}$ control is level sensitive, it may simply be held low through the completion (or termination) of the RAM Write instruction in which case the Store cycle will begin when CE falls low. Since $\overline{\text{Store}}$ is not latched, it may be returned high before CE falls and the Store procedure will not be executed.

The last point of clarification is in the area of cycle-ability. The distinct advantage of NOVRAMs over E^2PROMs is the ability to change data as often as one wishes. The RAM portion of the memory can be written to, and read from, indefinitely. The only time it would be necessary to update the E^2PROM portion of the NOVRAM is before a power down. Storing upon the least frequent of either data changes or power down will maximize the life of the device generally placing that far beyond the actual system needs.

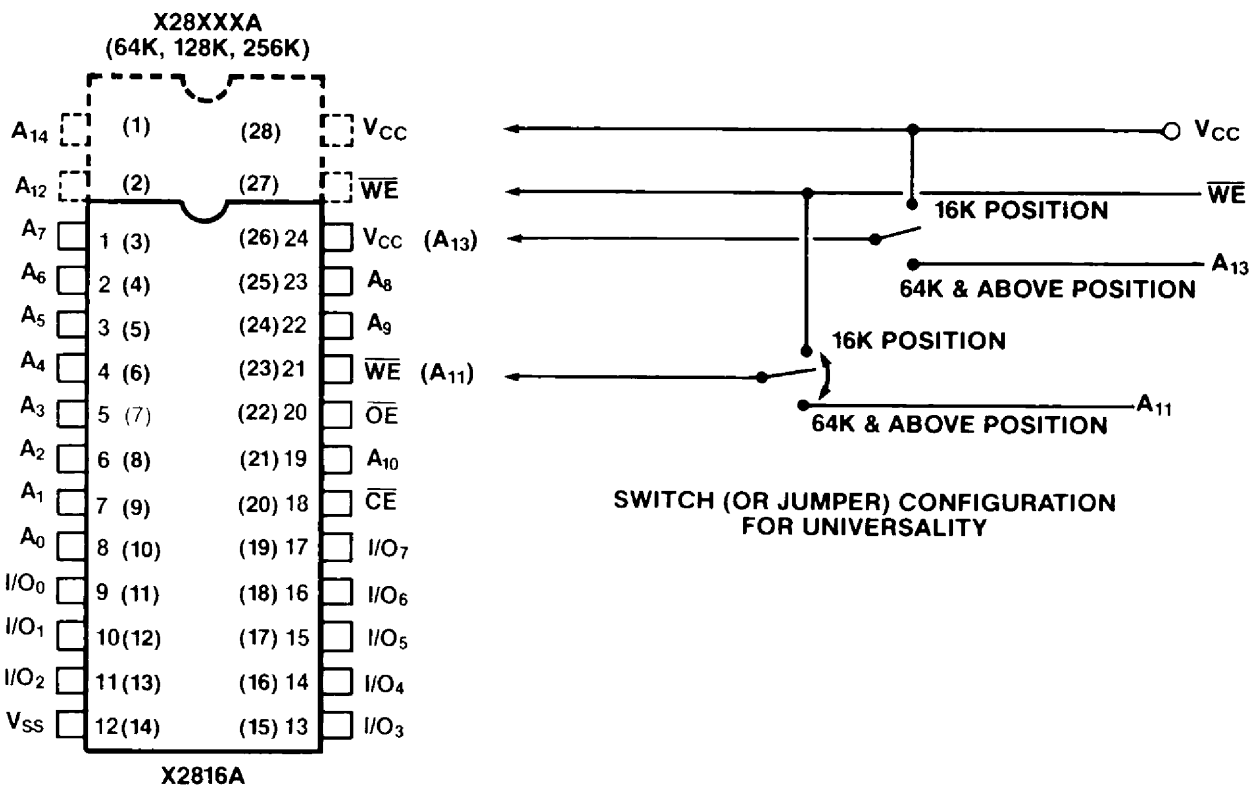
The combination of Xicor's renowned user oriented features with a little creative engineering make the Serial NOVRAM family the most versatile memory devices available to single chip microcomputer systems. A dual array memory system incorporating serial synchronous operations coupled with fast acting hardware inputs, data security options, low power operations, 5 volt nonvolatility, three pin control, and very low cost takes the system designer to the industry's leading edge.

X2816A Design Allows Upgrade to X2864A and Beyond

By Rick Orlando

Vol.1,No.5 August 23, 1983

Since the announcement of the X2864A, many customers have expressed a desire to design an X2816A socket that can be upgraded to accept the X2864A. The 28-pin footprint of the X2864A allows eventual migration up to the 256K bit level. If a board design is done properly, the socket can accept the entire product family.



First, let us look at the "universal" socket.

One can see that the socket will accept Xicor E^2PROMs from the 16K level through the 256K level with essentially two switches determining which footprint the socket is to be.

The next question is how can one minimize the address decoding logic for a board populated with the above "universal" sockets? We will assume that we have a board with eight "universal" sockets. We would like to design the address decoding logic to allow the eight X2816A's to be replaced by two X2864A's once they become cost effective.

First, we look at the addresses that each chip will be mapped into:

	Chip #	Address Range
X2816A	(1)	0000 - 07FF
	(2)	0800 - 0FFF
	(3)	1000 - 17FF
	(4)	1800 - 1FFF
	(5)	2000 - 27FF
	(6)	2800 - 2FFF
	(7)	3000 - 37FF
	(8)	3800 - 3FFF

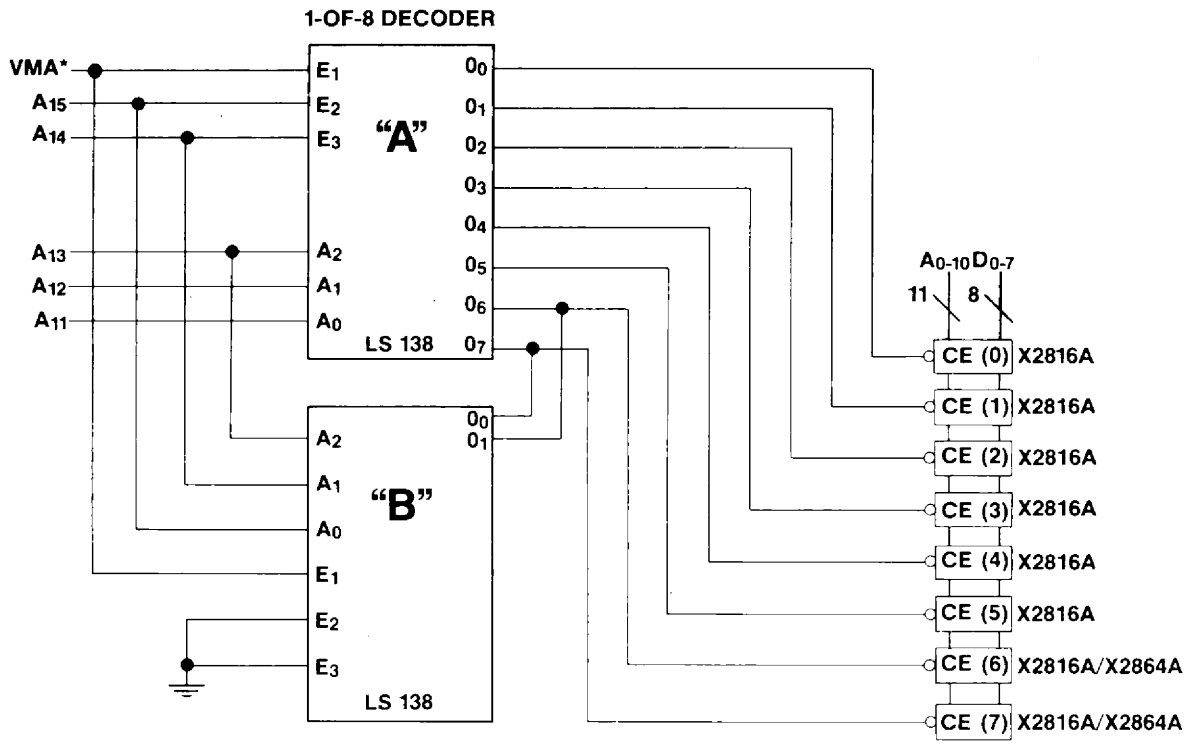
	Chip #	Address Range
X2864A	(1)	0000 - 1FFF
	(2)	2000 - 3FFF

The breakdown of the decoding for such a scheme is listed as follows:

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	**X2816A Chip Selects*** 1	2	3	4	5	6	7	8	**X2864A Chip Selects*** 1	2
0	0	0	0	0	0	1	1	1	1	1	1	1	0	1
0	0	0	0	1	1	0	1	1	1	1	1	1	0	1
0	0	0	1	0	1	1	0	1	1	1	1	1	0	1
0	0	0	1	1	1	1	1	0	1	1	1	1	0	1
0	0	1	0	0	1	1	1	1	0	1	1	1	1	0
0	0	1	0	1	1	1	1	1	1	0	1	1	1	0
0	0	1	1	0	1	1	1	1	1	1	0	1	1	0
0	0	1	1	1	1	1	1	1	1	1	1	0	1	0

*Chip Select is active low.

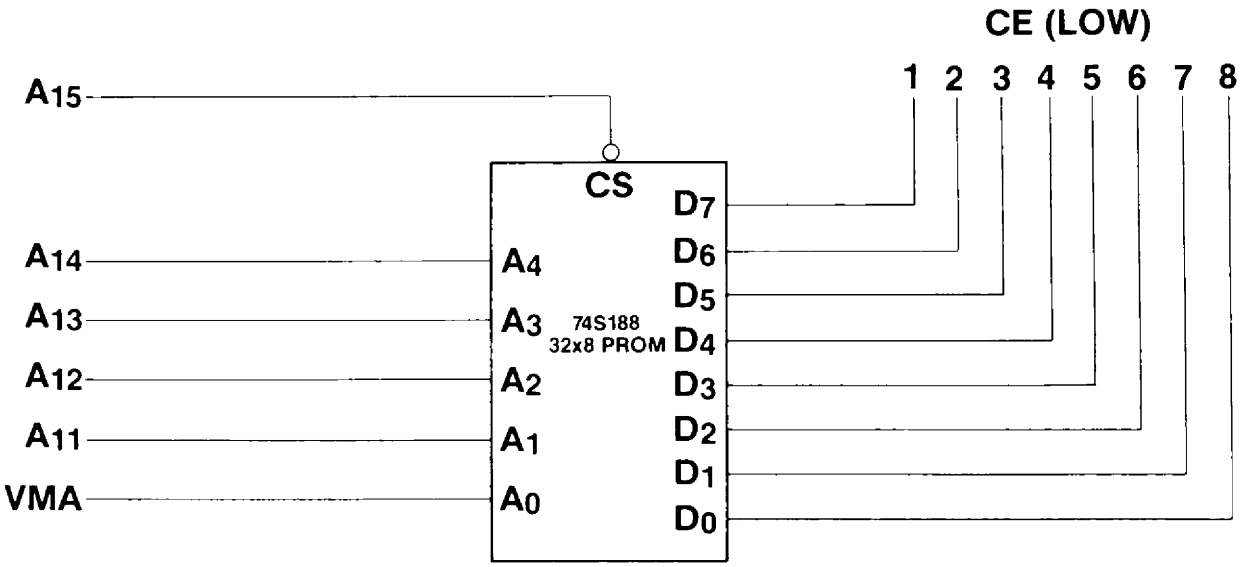
One possible solution is to use two 74LS138's with one being wired for the X2816A and one for the X2864A. If X2816A's are used in the board, LS138-A is installed and LS138-B is omitted. Conversely, if X2864A's are being used, LS138-B is installed and LS138-A is omitted.



*VMA = VALID MEMORY ADDRESS.

The two X2864A's are placed in sockets 6 and 7.

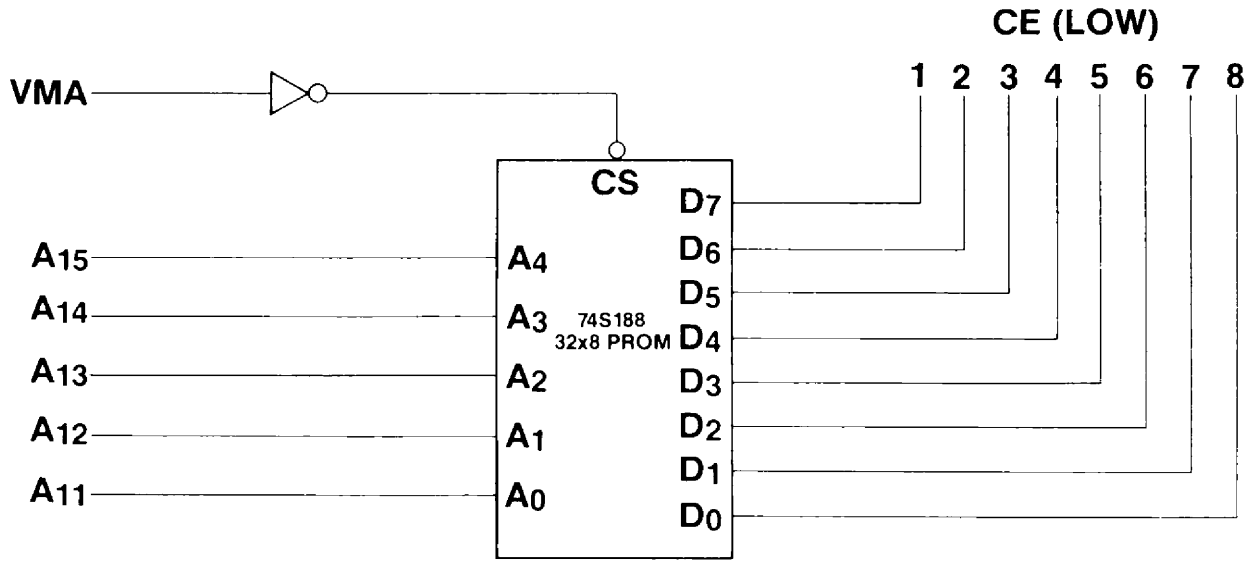
Another solution for the address mapping question is to use a 32x8 bipolar PROM for address decoding.



One simply changes the PROM for replacing the X2816A's with X2864A's.

Another related case is where the designer would like to increase the memory capacity of the board as the X2864A becomes available. In this case, we need to replace the eight X2816A's (128K bits/board) with eight X2864A's (512K bits/board).

Using the "universal" sockets, this is relatively easy. We use the following circuit, again using a 32x8 bipolar PROM.



One simply changes the program in the PROM to go from using X2816A's to X2864A's. The listing on the following page shows the PROM program for the X2816A's and the X2864A's.

					16K Bytes **X2816A** Program 0000 - 3FFF								64K Bytes **X2864A** Program 0000 - FFFF							
A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
1	0	0	0	0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1
1	1	0	0	0	1	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1
1	1	0	0	0	1	1	1	0	1	1	1	1	0	1	1	1	1	1	1	1
0	0	1	0	0	1	1	1	1	0	1	1	1	1	0	1	1	1	1	1	1
1	0	1	0	0	1	1	1	1	1	0	1	1	1	0	1	1	1	1	1	1
0	1	1	0	0	1	1	1	1	1	1	0	1	1	0	1	1	1	1	1	1
1	1	1	0	0	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1
0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

The PROM decode method allows a mixed combination of X2816A's and X2864A's to be used on the same board. It also allows different types of memory to be used such as X2816A's and 64K CMOS RAMs on the board. If the board is to be used with either 16K or larger devices, then only two jumpers are needed per board to configure all of the "universal" sockets. Just insure that the V_{CC} jumper will handle all of the current required by the memories.

The address decoding can be extended even further to incorporate expansion to the 256K bit E^2PROM. A larger PROM would be all that is required, but this topic is beyond the scope of this brief.

One can see that through using foresight in design, a modern designer can implement a memory socket that is truly universal in allowing the use of the most cost-effective E^2PROM densities available.

Replacing DIP Switches with Nonvolatile Technology

By Rick Orlando

Vol.1,No.7 September 26, 1983

One of the most prevalent applications for small nonvolatile memories is that of replacing DIP switches. The advantages of the nonvolatile memories is clear. They take up less room, are easier to use, and lend themselves to automated board assembly. 256 bits of information, or the equivalent of 32, 8-bit DIP switches can be implemented in a single package.

Xicor's new X2443/X2444 Serial NOVRAMs add yet another feature -- low cost. When coupled with the serial devices' minimal interface requirements, the X2400 series takes on DIP switches head on, and is obviously the cost/performance leader. The purpose of this brief is to describe how easy it is to replace a DIP switch with an X2400 series NOVRAM.

DIP Switch Interface

There are two common types of DIP switch interfaces.

I/O Port

The first uses an I/O port with internal pull-up resistors. Figure 1a shows a typical circuit that could be used either with a single chip microcomputer or with an I/O port on a microprocessor bus. In either case, the internal pull-ups present a logic "1" to the input as long as the DIP switch is open. To use an X2400 series Serial NOVRAM in the DIP switch socket, one only needs to tie pins 14, 15, and 16 of the 16-pin socket to V_{CC}.

One then plugs an X2400 series part in the uppermost half of the socket, and the circuit becomes that shown in Figure 1b. V_{CC}, STORE, and RECALL are tied hard to 5 volts, so that all nonvolatile operations are controlled through software. The four interface lines from the X2400 series are connected to the four least significant I/O lines of the port.

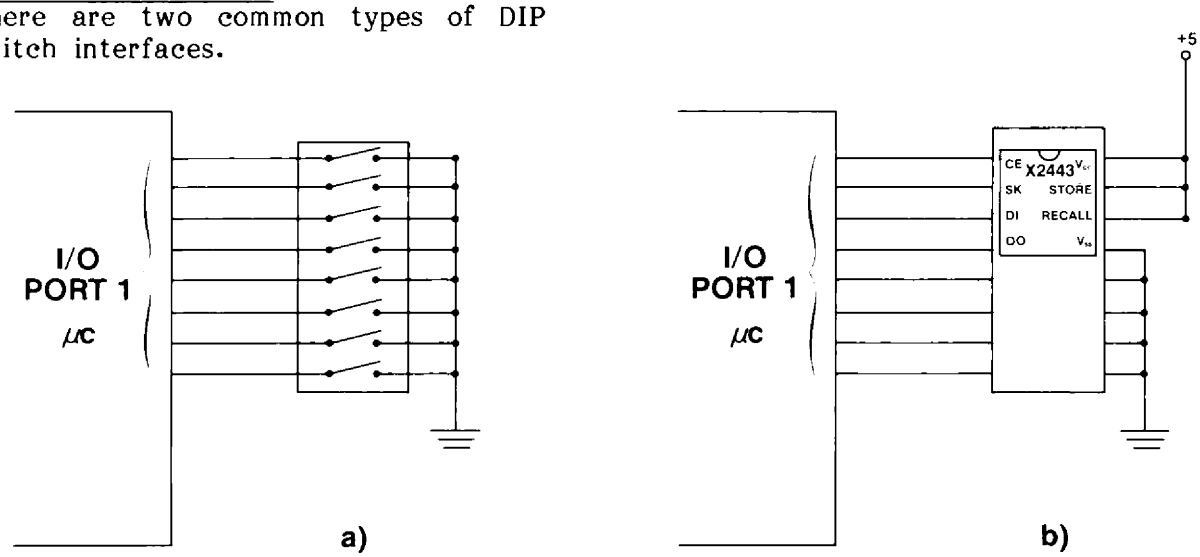


Figure 1. Microcomputer with Internal Pullups

No Internal Pullups

The second type of interface uses ports which do not have internal pullups. In this instance, the X2400 series part can be plugged into the top section of the pullup resistor socket, with a jumper from pin 13 of the 16-pin site to ground, for the V_{SS} on the serial part. Again, V_{CC}, STORE, and RECALL are tied to +5V through the connections used for the resistor pack. The DIP switch socket simply remains empty. See Figure 2.

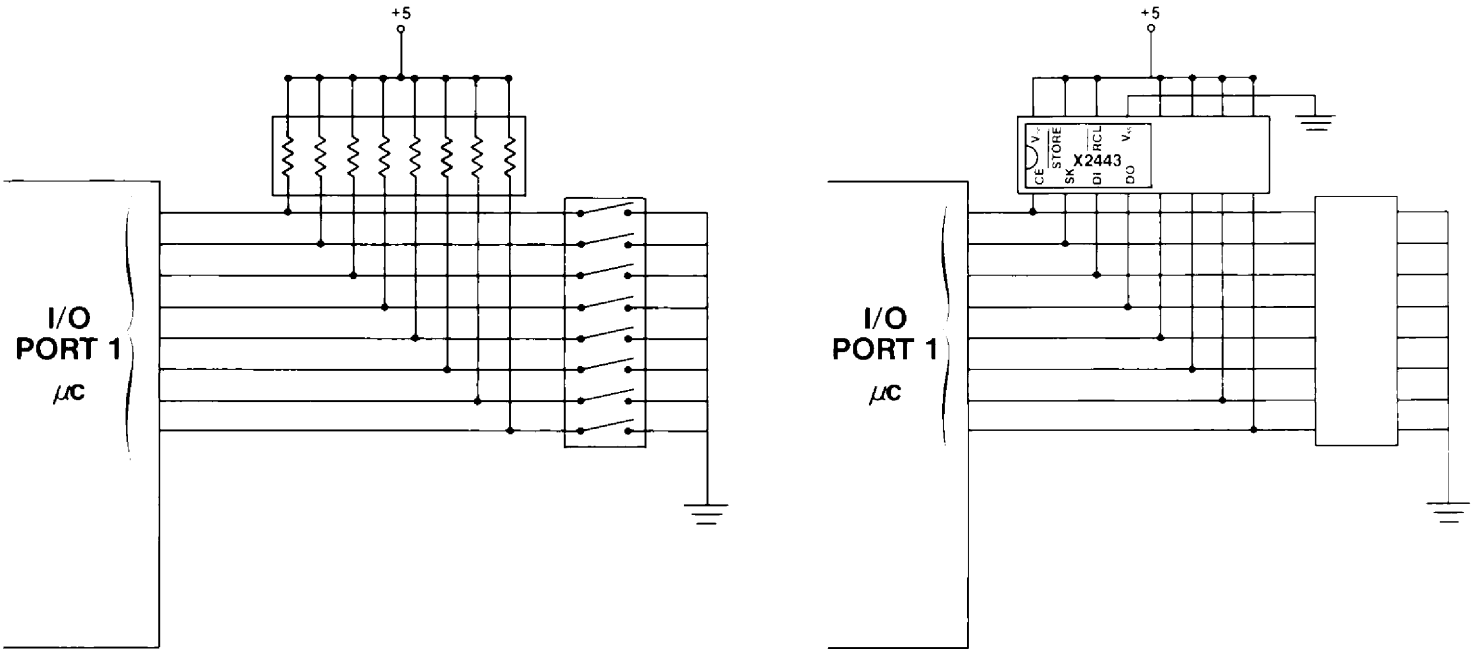


Figure 2. Microcomputer without Internal Pullups

Both of these implementations free up four more I/O lines to be used elsewhere. They also require the same software to drive the X2400 series parts.

Assume that the processor is a 6801 with the X2444 replacing a DIP switch. The Procedure "INIT" initializes the port (see Section 1, "X2444 Driver Program for 6801").

Serial output is accomplished by loading the data to be output into the A Accumulator. A loop routine then shifts a bit into the carry, sets the serial data out (Data in for X2444) to either a "1" or "0" depending upon the state of the carry and toggles the clock. See Section 2 Procedure "SHIFT1" of X2444 Driver Program for further details.

The serial input is performed by a loop which examines the state of the serial data in, (Data out for X2444) sets the carry accordingly, shifts the carry into the accumulator and toggles the clock. See Section 3 Procedure "SHIFTIN" of X2444 Driver Program for further details.

The complete software is as follows, and it occupies about 100 bytes of code. As one can see, the X2400 parts are indeed a great replacement for DIP switches!!

```
;*************************************************************************
;                          6801 X2444 DRIVER
;       ASSUME THAT PORT1 IS USED AS THE 2444 INTERFACE
;       PORT1'S REGISTERS ARE LOCATED AS FOLLOWS
;                DATA DIRECTION          HEX  0000
;                PORT                    HEX  0002
;                                PORT1              X2444
;       I/O 0                    SERIAL CLOCK       SERIAL CLOCK
;       I/O 1                    SERIAL OUT         SERIAL IN
;       I/O 2                    SERIAL IN          SERIAL OUT
;       I/O 3                    2444 SELECT        CHIP SELECT
;
;       COMMANDS ARE PASSED TO THE X2444 ROUTINE BY A PARAMETER IN THE
;       A ACCUMULATOR, WHILE THE ADDRESS IF NEEDED IS PASSED ON THE STACK
;       SERIAL DATA IN OR OUT USES THE TEMPORARY LOCATION TEMP1,
;       WHICH IS A SIXTEEN BIT WORSD.  THE X2444 COMMANDS ARE ENCRYPTED AS
;       FOLLOWS.
;       COMMAND CODE                INSTRUCTION          OPCODE
;       0                           READ                 1AAAA11X
;       1                           WRITE                1AAAA011
;       2                           RESET WRITE ENABLE   11111000
;       3                           STORE                11111001
;       4                           SLEEP                11111010
;       5                           SET WRITE ENABLE     11111100
;       6                           RECALL               11111101
; "1"s ARE USED INSTEAD OF DON'T CARE TO DISTINGUISH BETWEEN DATA AND
;  NON DATA OPERATIONS.
;*************************************************************************
DIRECTION1          .EQU        00.
PORT1               .EQU        02.
TEMP1               .EQU        080H    ;RAM STORAGE FOR DATA
COUNT               .EQU        082H    ;COUNTER VARIABLE
DATUM               .EQU        084H    ;DATA STORAGE
ADDRESS             .EQU        086H    ;ADDRESS STORAGE
ERRORDATA           .EQU        088H     ;ERROR DATA
;*************************************************************************
;                   PROCEDURE INIT
;       THIS PROCEDURE INITIALIZES THE X2444 INTERFACE
;*************************************************************************
INIT    LDAA        #1BH            ; B=1011, I/O 0,1 AND 3 OUTPUTS, 2 INPUT
        STAA        DIRECTION1      ; WRITE TO DATA DIRECTION REGISTER
        CLRA                        ;SET CE TO 0(INACTIVE), DOUT AND SK TO 0
        STAA        PORT1           ;AND STORE IN DATA PORT
        RTS                         ;

;*************************************************************************
;       SHIFTER ROUTINE- SHIFT1
; THIS ROUTINE TAKES THE DATA IN THE A ACCUMULATOR AND CLOCKS IT MOST
; SIGNIFICANT BIT FIRST INTO THE X2444.  THE FLOW IS SHIFT A BIT, TOGGLE
; THE SERIAL OUTPUT(6801) ACCORDING TO STATE, AND TOGGLE SERIAL CLOCK
;*************************************************************************

SHIFTOUT LDAB       #08.            ;LOAD THE BIT COUNT WITH 8
        STAB        COUNT           ;STORE IN COUNTER
SHIFT1  ROLA                        ;SHIFT BIT INTO CARRY BIT
        LDAB        #14H            ;WE SET DATA OUT TO ZERO, WHILE SETTING CHIP
                                    ;ENABLE.  SERIAL CLOCK IS LOW.
        BCC         TRANS           ;IF BIT IS A ZERO, THEN TRANSMIT
        ORAB        #02H            ;IF IT IS A ONE, THEN SET DATA OUT
TRANS   STAB        PORT1           ;STORE THE DATA INTO THE PORT
        ORAB        #01H            ;AND SET THE CLOCK FOR A TRANSITION
        STAB        PORT1           ;BY WRITING A 1 TO SERIAL CLOCK
        ANDB        #1AH            ;KEEP THE DATA VALID, BUT SET SK TO ZERO
        STAB        PORT1           ;AND STORE IN THE PORT
        LDAB        #14H            ;TOGGLE CLOCK DOWN, SET DOUT TO 0, BUT KEEP
        STAB        PORT1           ;X2444 SELECTED
        DEC         COUNT           ; DECREMENT THE BIT COUNTER
        BNE         SHIFT1          ;IF COUNT IS NOT ZERO, TRANSMIT NEXT BIT
        ROLA                        ;ONE MORE ROTATE TO PRESERVE INSTRUCTION
        RTS                         ;RETURN FROM SUBROUTINE
;*************************************************************************
;                   SHIFTIN ROUTINE
; THIS SUBROUTINE SHIFTS IN 8 BITS OF DATA INTO THE A ACCUMULATOR FROM THE
; X2444. THE METHOD IS TO ENTER WITH THE CLOCK LOW, TOGGLE THE SERIAL CLOCK,
; EXAMINE THE INPUT DATA, AND SHIFT IT INTO THE A ACCUMULATOR.  THIS IS DONE
; 8 TIMES.  THE ROUTINE IS EXITED WITH THE CHIP DESELECTED, AND THE BYTE
; READ FROM THE CHIP IN THE A ACCUMULATOR
;*************************************************************************
SHIFTIN LDAB        #8.             ;LOAD THE BIT COUNT
        STAB        COUNT           ;AND STORE IT IN THE COUNTER
                                    ;AT THIS POINT THE X2444 SHOULD BE SELECTED
                                    ;THEREFORE, WE DO NOT NEED TO SELECT CHIP
NEXT    LDAB        #04             ;MASK BIT FOR I/O 2 OF PORT
        BITB        PORT1           ;CHECK TO SEE IF INPUT IS A ONE OR ZERO
        CLC                         ;CLEAR THE CARRY
        BEQ         CLOCK           ;IF IT IS A ZERO, LEAVE CARRY AT 0
        SEC                         ;OTHERWISE SET CARRY TO LOAD INTO A
CLOCK   LDAB        #15H            ;SEND A CLOCK TO X2444,BUT KEEP CHIP SELECT HIGH
        STAB        PORT1           ;BY WRITING A 1 TO SERIAL CLOCK OUTPUT
SHIFT   LDAB        #14H            ;SET UP TO CLEAR CLOCK, BUT KEEP X2444 SELECTED
        STAB        PORT1           ;AND STORE
        ROLA                        ;ROTATE CARRY INTO LSB OF ACCUMULATOR A
        DEC         COUNT           ;DECREMENT COUNTER
        BNE         NEXT            ;IF NOT ZERO, THEN WE ARE NOT DONE, GET NEXT
        RTS                         ;AND RETURN FROM SUBROUTINE
```

Section 1

Section 2

Section 3

```
;*************************************************************************
;         X2444 DRIVER ROUTINE
; IT IS ASSUMED THAT THE INSTRUCTION IS PASSED IN THE A ACCUMULATOR.  AN
; ADDRESS, IF NEEDED, IS PASSED ON THE STACK(CURRENT SP-2)
; DATA TO BE READ OR WRITTEN WILL BE HELD IN TEMP1
;*************************************************************************
DRIVE    CMPA     #0F8H          ;CHECK TO SEE IF IT IS READ OR WRITE
         BGE      NONDATA        ;IF NOT, THEN BRANCH AROUND
         TSX                     ;TRANSFER STACK TO INDEX REGISTER
         ORAA     2,X            ;THE ADDRESS SHOULD BE SP+2
         JSR      SHIFTOUT       ;OUTPUT THE INSTRUCTION
         ANDA     #04H           ;CHECK TO SEE IF IT IS A READ OR WRITE
         BNE      RD             ;IF AC3=1, IT IS A READ

WRT      LDAA     TEMP1          ;IF IT IS A WRITE, GET THE FIRST BYTE
         JSR      SHIFTOUT       ;WRITE THE FIRST BYTE
         LDAA     TEMP1+1        ;GET THE SECOND BYTE
         JSR      SHIFTOUT       ;WRITE THE SECOND BYTE
         BRA      DONE           ;WRITE INSTRUCTION COMPLETE
RD       JSR      SHIFTIN        ;GET THE FIRST BYTE
         STAA     TEMP1          ;STORE IN TEMP1
         JSR      SHIFTIN        ;GET THE SECOND BYTE
         STAA     TEMP1+1        ;STORE IN TEMP1+1
         BRA      DONE           ;READ COMPLETE
NONDATA  JSR      SHIFTOUT       ;OUTPUT THE INSTRUCTION
DONE     CLRA
         STAA     PORT1          ;DESELECT THE X2444 BY MAKING CS 0
         RTS                     ;RETURN FROM SUBROUTINE
;*************************************************************************
;                  MAIN INSTRUCTION ROUTINES- COULD BE MACROS
;*************************************************************************
;*************************************************************************
;         READ ROUTINE
; ASSUMES THAT THE ADDRESS IS IN THE A ACCUMULATOR, DATA IS LEFT IN X REGISTER
;*************************************************************************
READ     ASLA                     ;SHIFT THE ADDRESS 3 TIMES TO LINE IT
         ASLA                     ;UP WITH THE INSTRUCTION FIELD
         ASLA
         PSHA                     ;PUSH ADDRESS ON THE STACK
         LDAA     #087H           ;LOAD INSTRUCTION INTO A ACCUMULATOR
         JSR      DRIVE           ;PERFORM INSTRUCTION
         LDX      TEMP1           ;GET THE RESULT IN THE INDEX REGISTER
         PULA                     ;CLEAN UP THE STACK
         RTS                      ;AND RETURN
;*************************************************************************
;                                 WRITE ROUTINE
; ASSUMES THAT THE ADDRESS IS IN THE A ACCUMULATOR, DATA TO WRITE IS IN THE
; X REGISTER.
; ALSO CALLS SET WRITE ENABLE LATCH ROUTINE(SWREN) TO ENABLE THE WRITE OPERATION
;*************************************************************************
WRITE    ASLA                     ;SHIFT THE ADDRESS 3 TIMES
         ASLA
         ASLA
         PSHA                     ;PUSH ADDRESS ONTO THE STACK
         JSR      SWREN           ;SET THE WRITE ENABLE LATCH
         LDAA     #083H           ;LOAD WRITE INSTRUCTION
         STX      TEMP1           ;STORE DATA IN TEMP1
         JSR      DRIVE           ;PERFORM INSTRUCTION
         PULA                     ;CLEAN UP STACK
         RTS
;*************************************************************************
RWREN    LDAA     #0F8H           ;LOAD THE INSTRUCTION
         JSR      DRIVE           ;AND EXECUTE
         RTS
;*************************************************************************
STORE    LDAA     #0F9H           ;LOAD THE INSTRUCTION
         JSR      DRIVE           ;PERFROM OPERATION
         RTS                      ;AND RETURN
;*************************************************************************
SLEEP    LDAA     #0FAH           ;LOAD THE INSTRUCTION
```

READY/$\overline{\text{BUSY}}$
A Solution or a Limitation ?

By Rick Orlando

Vol.1, No.11 December 14, 1983

The Ready/$\overline{\text{Busy}}$ signal was introduced in 1982 by Intel on their 2817 E^2PROM. It provided a means of determining the completion of the E^2PROM write cycle. Since the Intel 2817 was one of the first self-timed E^2PROMs, a method of determining when the relatively long write cycle was completed was desired, both from the end users point as well as the manufacturers. The problem was a simple one. Although the 2817 required only RAM write signals to initiate a write, the device took a relatively long period of time to complete the write cycle to allow subsequent access to the chip. Xicor, of course, has opted for the $\overline{\text{Data}}$ Polling approach which has been accepted by the industry as the most versatile and universal approach. Even Intel has hinted that the newer devices in the E^2PROM area should support $\overline{\text{Data}}$ Polling.

The Intel 2817 had a serious handicap in the fact that its typical write cycle time was listed as 10 msec, while its maximum was 75 msec. This represents a difference in chip re-write time of over 130 seconds!! This presented a serious problem to the system designer who wanted to design his system for "worst-case" operation, since he had to assume the 75 msec maximum. This large delta was the major reason for the implementation of some type of write completion notification mechanism.

It should be pointed out that if the variation in the typical write times for an E^2PROM are very close and if the actual times are close to the maxim specified value, the utility of a write notification technique becomes minimal. As the maximum approaches the typical interval, the increase in overall write speed for the chip which can be gained from taking advantage of the typical value becomes smaller. This increase approaches zero in the case of equivalent specifications. Even the newer Intel 2817A has a wide difference between the specified typical and maximum write cycles. Write cycle time is very precisely controlled, and therefore, there is not a large range of actual write cycle values. This minimal variation allows the Xicor E^2PROMs to be specified with short maximum write cycles. Because of this, write cycle termination notification is not as important. Both devices do feature write cycle notification in the form of High-Impedance Interrogation on the X2816A and $\overline{\text{Data}}$ Polling on the X2864A.

Ready/$\overline{\text{Busy}}$ is not an ideal method for write cycle termination notification. It is a hardware signal which resides on pin 1 of the 2817A's universal 28-pin socket. Unfortunately, the fact that it is an output rather than the expected address input immediately makes the socket non-standard in terms of inter-device type (RAM, EPROM, etc.) interchangability. This was the whole purpose for the JEDEC 28-pin standard for byte-wide memories. The presence of Ready/$\overline{\text{Busy}}$ also limits the expansion of the byte-wide E^2PROM socket to the 128K bit level, and will not allow compatability through the 256K bit level. Xicor is currently sampling the 64K E^2PROM. The 256K bit is the next logical generation, and is well within the limits of the currently existing technologies. Xicor is keeping pin 1 left open on all E^2PROM devices with densities lower than 256K bits to allow full expansion of the 28-pin universal byte-wide socket. This, of course, leads one to a software oriented approach to write cycle notification such as $\overline{\text{Data}}$ Polling.

Ready/$\overline{\text{Busy}}$ is also a hardware interface method. Aside from the fact that it occupies a potential address line on the socket, it also requires some means of providing its state back to the processor which is writing to the E^2PROM. This can be accomplished through either inputting the signal into a port on the processor's bus, or using it to generate an interrupt. While this method might seem attractive in systems which have only one E^2PROM in them, it becomes a serious problem in systems which have more than one E^2PROM. One could tie all of the Ready/$\overline{\text{Busy}}$ signals together, but that would mean that each write cycle would have to wait for the previous cycle to complete, even if it was on a different chip than the one currently needed. Use of multiple ports or interrupt channels (one for each E^2PROM) has been proposed, but unfortunately, this increases the support circuitry requirements for the E^2PROMs and makes them less and less "RAM like". Since the overall user demands for E^2PROMs have been to eliminate the support circuitry and make the devices more "RAM like", it appears that Ready/$\overline{\text{Busy}}$ is more of a step backwards in this respect rather than a step forward.

Another potential problem with Ready/$\overline{\text{Busy}}$ lies in the way it is used in the Intel devices. The reason for the long maximum write cycle of the 2817 and 2817A is that the devices will check to see if the particular byte written is programmed "hard" enough to maintain its data. If the device determines that this is not the case, then it will try to program the byte again and again until the byte is verified. If a particular byte does not program due to its endurance limit being exceeded, the chip will continue to try to write to the byte indefinitely, and Ready/$\overline{\text{Busy}}$ will never toggle. The chip will then hang up the system until the power is turned off and then on again. In order to design around this "feature", one must put in a default timing mechanism to determine if the byte has taken too long to write. The system must then figure out a way to "wake up" the E^2PROM, an operational mode not documented in the literature.

The proponents of Ready/$\overline{\text{Busy}}$ say that it eases the implementation of an interrupt driven method to write to the E^2PROM, relieving the processor from keeping track of the chip timing. In most applications, the processor does not really have much to do during the time that the particular byte is being programmed, and a software timing loop is more than adequate. This software timing method works best with those chips whose typical write time are close to their maximum write times for the reasons described earlier. The only exceptions to this scenario are those applications involving real-time programs which are trying to load the E^2PROM as fast as it will take the data, such as in the downloading of a program. Unfortunately, even in this application, the designer cannot improve the baud rate of the down-load based upon typical write times of the E^2PROMs.

In summary, the Ready/$\overline{\text{Busy}}$ approach has limited utility when one considers all of the factors involved in its implementation. User preference has shown that if features are to be added to memory devices, they should be implemented so that they are controlled exclusively from software rather than hardware. It is towards this end that Xicor has developed $\overline{\text{Data}}$ Polling on its new X2864A and expects that this method will become the industry standard for write cycle completion notification. This standardization process can already be seen in recent product announcements including those from Intel and others which have stated that their next generation E^2PROMs (the 64K in this instance) will support the $\overline{\text{Data}}$ Polling approach.

Xicor Application Brief

Using $\overline{\text{DATA}}$ POLLING in an Interrupt Driven Environment

By Rick Orlando

Vol.2, No.12 March 20, 1984

The use of interrupt driven system design has become increasingly popular in many applications. Interrupt driven systems usually can achieve higher performance and improved user friendliness. An interrupt driven system can perform a variety of tasks while waiting for a certain condition to occur, rather than constantly looping and waiting for the occurence. Writing to E^2PROMs is no exception. Since the devices take a relatively long period to complete a write cycle, the system could perform a variety of tasks in the meantime.

$\overline{\text{Data}}$ Polling was introduced on the X2864A to allow notification to the processor of write cycle completion. The manner in which it works is quite simple. The processor first writes a byte of data into the E^2PROM. Any subsequent reads to any location of the chip will produce the complement of the data last written (hence, the name $\overline{\text{Data}}$ Polling) until the E^2PROM's internal write cycle is complete. At this point, reads to any location in the E^2PROM will result in the valid data at that location. It can be seen that one can simply write a byte, and then perform frequent compares of the data in the location just written. The data will not be correct until the chip has completed its internal write cycle, and the $\overline{\text{Data}}$ circuitry is disabled.

In applications where the processor does not have anything to do during the write cycle, the software can simply perform compare loops until the write cycle is complete, and then write the next byte. In applications which are more processing time limited, a test loop can be placed in the main program loop, which will check the status of a previous write cycle on each pass through the main or outermost software loop. Almost all microprocessor applications software has such a top level loop. $\overline{\text{Data}}$ Polling is obviously adequate in these environments.

The interrupt intensive applications may not have a main control loop nor can they usually afford the processing time for the processor to sit and loop until the write cycle is complete. In these applications, it would be ideal if the write cycle completion notification could be interrupt controlled. Although it is not obvious, $\overline{\text{Data}}$ Polling can be used in these applications as well.

It should be noted that the whole reason for write cycle notification is because the typical write times for the E^2PROMs are substantially shorter than the specified maximums. The magnitude of the delta between the typical and the maximum values determines the importance of the write cycle notification. One can easily see that if the maximum write cycle time and the typical were equal, one would only have to time a fixed interval for each write cycle, either from a software loop or a hardware timer. The hardware timer would generate an interrupt 10 msec after the write cycle was initiated, and the next byte could be written. Keep in mind that the discussion of write times for E^2PROMs are in terms of msec rather than the usec in which the processor executes instructions. A few usecs here or there are not important when compared to the write cycle time of about 10 msec.

$\overline{\text{Data}}$ Polling does not require any additional hardware interface in order to be used. It is an exclusively software oriented method for determining write cycle completion. Even in an interrupt environment, no additional circuitry is required, since all of the interface to the chip occurs through the data and address bus.

In order to use $\overline{\text{Data}}$ Polling in an interrupt driven system one only needs a time-based interrupt generator. This could be a programmable timer or even something as simple as an AC frequency interrupt. The key is that the processor does not check to see if the device has completed the write cycle until the interrupt occurs. The interrupt routine simply compares the data last written to the E^2PROM to the data coming from the E^2PROM. If the two match, the device can be written again. If not, the processor simply returns from the interrupt routine to where it was and continues processing until the next interrupt. The interrupt source is maskable which prevents the overhead of servicing the periodic interrupt during the intervals when a write has not been performed.

A programmable timer or counter is the most elegant solution. Figure 1 shows the hardware configuration of a typical system with the E^2PROM and the programmable timer on the bus. It should be noted that no unusual circuitry is needed from the E^2PROM socket, which preserves its usefulness as a truly universal socket. The timer interrupt output drives one of the processor's interrupt lines. Many systems already have such a timer on the bus, and as a result require no additional hardware changes to implement this method.

The software implementation is rather simple. Figure 2 shows an example of how it might be done using a 6800 microprocessor and a simple timer. The timer control and data registers are mapped into the memory locations described in the initial header along with the temporary RAM variables, which are used to store the last data written and its address.

The write routine (WEEPROM) initially checks to see if the E^2PROM is ready to perform a write. If not, it simply exits with an error code to show that the write has not taken place. If the write is performed, the timer is loaded with the initial count for 4 msec, and the timer interrupt is enabled. The processor then goes off and performs its normal duties until the interrupt takes place. At that point, the interrupt routine (CKEEP) is entered. It first checks for the proper data that was written to see if the write is complete, using $\overline{\text{Data}}$ Polling. If it is, the ready flag is set, and the routine is exited. If not, the counter is loaded with a smaller increment, such as 500 usec until the chip's write cycle is completed. This essentially allows the majority of the write time to pass (4 msec) before the processor checks at the more frequent interval of every 500 usec.

500 usec has been chosen in this example for the interrupt granularity. The value used for a particular application should be chosen based upon the actual system requirements.

One can see that this implementation is rather easy and can be performed with hardware that already may exist in the system. By using the periodic interface approach, the system has the advantage of using an interrupt driven write algorithm, while maintaining only a software interface to the E^2PROM. Most of the "bookkeeping" sections of the example code are the same as one would use with any method of write termination notification. The end result of using $\overline{\text{Data}}$ Polling in an interrupt environment is optimization of the write cycle period as well as preservation of the pinout of the universal 28-pin socket for expansion through the 256K bit level for E^2PROMs.

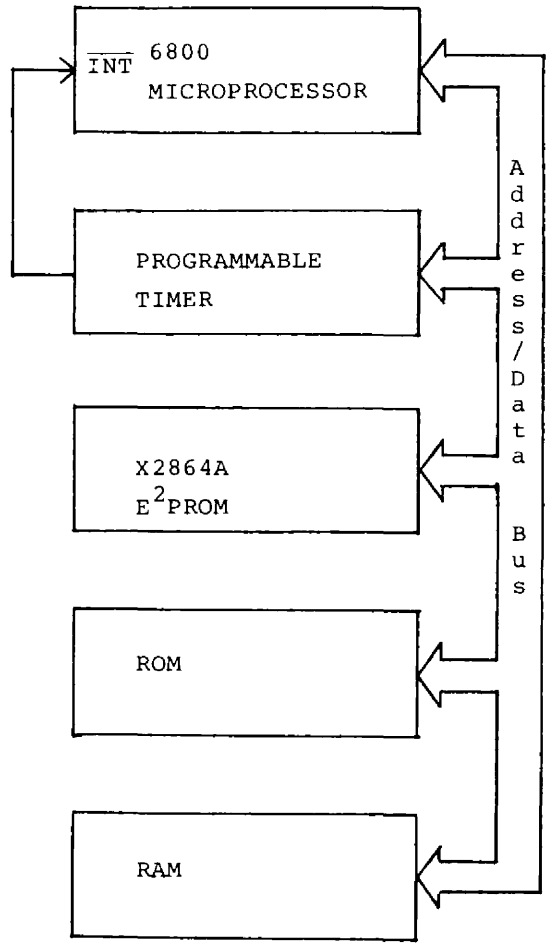


Figure 1

```
PAGE -   1  DATA_COD
File: :DATA_CODE.TEXT                CODE FOR DATA POLLING IN INTERRUPT ENVIRONMENT, 6800 VERSION

0000:                                        .PROC DATA_CODE
Current memory available:    4380
0000:                        ;*************************************************************************
0000:                        ;        SAMPLE CODE FOR USING DATA POLLING IN INTERRUPT ENVIRONMENT
0000:                        ;    THIS CODE SHOWS AN EXAMPLE OF HOW TO USE DATA POLLING IN A
0000:                        ; INTERRUPT DRIVEN WRITE MODE
0000:                        ;*************************************************************************
0000:                        ;                    MEMORY LOCATIONS
0000: 0100                   TIMER    .EQU     0100H    ;LOCATION OF TIMER DATA REGISTER(COUNT DOWN VALUE)
0000:                                                   ;16 BIT VALUE(2 BYTES)
0000: 0102                   CTIMER   .EQU     0102H    ;TIMER CONTROL REGISTER
0000: 0103                   LASTA    .EQU     0103H    ;RAM LOCATION FOR LAST WRITTEN DATA
0000: 0104                   TEMP     .EQU     0104H    ;RAM LOCATION FOR LAST ADDRESS WRITTEN
0000: 0106                   ERROR    .EQU     0106H    ;ERROR FLAG FOR WRITE
0000: 0107                   READY    .EQU     0107H    ;MEMORY READY FOR NEXT WRITE FLAG
0000: 0108                   CONFIG   .EQU     0108H    ;TIMER CONFIGURATION BYTE
0000:
0000:                                    .ORG 0F000H
F000:                        ;*************************************************************************
F000:                        ;                    EEPROM WRITE ROUTINE
F000:                        ;    THIS ROUTINE WRITES A BYTE OF DATA PASSED IN THE A ACCUMULATOR INTO
F000:                        ; THE EEPROM AT LOCATION POINTED TO BY THE VALUE IN THE X INDEX REGISTER.
F000:                        ; THE ROUTINE THEN INITIALIZES THE DATA IN THE COUNTER TO 4096, SINCE THE TIMER
F000:                        ; COUNTS AT A 1 MHZ FREQUENCY.  THIS WILL TIME OUT THE INITIAL 4 MSEC OF THE WRI
F000:                        ; WRITE CYCLE.  THE ROUTINE THEN ENABLES THE INTERRUPTS AN RETURNS CONTROL TO
F000:                        ; THE CALLING ROUTINE
F000:                        ;*************************************************************************
```

Figure 2

```
F000: C6 00                 WEEPROM LDAB    #00H    ;COMPARE TO SEE IF THE MEMORY IS READY FOR WRITE
F002: F1 0107                       CMPB    READY   ;CHECK MEMORY LOCATION READY
F005: 27 **                         BEQ     WRITE   ;IF READY THEN WRITE
F007: C6 00                         LDAB    #00H    ;OTHERWISE SET THE ERROR FLAG
F009: F7 0106                       STAB    ERROR   ;AND STORE IT IN ERROR
F00C: 20 **                         BRA     EXIT    ;AND EXIT
F00E: A7 00                 WRITE   STAA    0,X     ;WRITE THE DATA IN A TO LOCATION X
F010: B7 0103                       STAA    LASTA   ;STORE DATA IN RAM AT LASTA
F013: FF 0104                       STX     TEMP    ;AND STORE LOCATION IN EEPROM AT TEMP
F016: C6 10                         LDAB    #10H    ;LOAD THE FIRST BYTE OF 4096 HEX INTO B ACCUMULATOR
F018: F7 0100                       STAB    TIMER   ;WRITE TO TIMER DATA REGISTER
F01B: C6 00                         LDAB    #00H    ;LOAD THE SECOND BYTE OF 4096 HEX
F01D: F7 0101                       STAB    TIMER+1 ;STORE IN THE LSB OF TIMER REGISTER
F020: F6 0108                       LDAB    CONFIG  ;GET THE TIMER INITIALIZATION CODE
F023: F7 0102                       STAB    CTIMER  ;STORE IT IN THE TIMER CONTROL REGISTER
F026: 0E                            CLI             ;CLEAR INTERRUPT MASK TO LET TIMER INTERRUPT
F027: 39                    EXIT    RTS             ;RETURN FROM SUBROUTINE
F028:                       ;*******************************************************************
F028:                       ;           CHECK EEPROM ROUTINE
F028:                       ;   THIS ROUTINE CHECKS TO SEE IF THE EEPROM IS DONE.  IT IS ENTERRED
F028:                       ; EVERY TIME THAT THE INTERRUPT IS GENERATED FROM THE TIMER.  IT COMPARES
F028:                       ; THE DATA AT LOCATION TEMP IN THE EEPROM WITH THE DATA THAT IS STORED IN
F028:                       ; LASTA LOCATION IN RAM.  IF THE COMPARISON FAILS, THEN THE EEPROM IS NOT
F028:                       ; DONE  YET , AND THE TIMER IS RESET TO TIME 1/2 MSEC(OR 512 USEC).  THE
F028:                       ; ROUTINE IS THEN EXITED.
F028:                       ;*******************************************************************
F028: B6 0103               CKEEP   LDAA    LASTA   ;GET THE LAST DATA WRITTEN
F02B: FE 0104                       LDX     TEMP    ;GET THE ADDRESS FOR THE BYTE IN EEPROM
F02E: A1 00                         CMPA    0,X     ;COMPARE TO SEE IF THE WRITE IS COMPLETE
```

```
PAGE -   2  DATA_COD
File: :DATA_CODE.TEXT                CODE FOR DATA POLLING IN INTERRUPT ENVIRONMENT, 6800 VERSION

F030: 26 **                         BNE     REINIT  ;IF NOT EQUAL REINITIALIZE THE TIMER
F032: 86 00                         LDAA    #00     ;SET THE READY FLAG
F034: B7 0107                       STAA    READY   ;AND STORE IT IN READY
F037: 20 **                         BRA     RET     ;AND RETURN FROM SUBROUTINE
F039: 86 02                 REINIT  LDAA    #02H    ;LOAD THE FIRST BYTE FOR COUNTER
F03B: B7 0100                       STAA    TIMER   ;WRITE TO TIMER DATA REGISTER
F03E: 86 00                         LDAA    #00H    ;LOAD THE SECOND BYTE
F040: B7 0101                       STAA    TIMER+1 ;WRITE TO LSB OF TIMER DATA REGISTER
F043: 3B                    RET     RTI             ;RETURN FROM INTERRUPT
F044:                       ;*******************************************************************
F044:                           .END
```

```
SYMBOL TABLE DUMP
        CODE FOR DATA POLLING IN INTERRUPT ENVIRONMENT, 6800 VERSION

AB - Absolute       LB - Label      UD - Undefined      MC - Macro
RF - Ref            DF - Def        PR - Proc           FC - Func
PB - Public         PV - Private    CS - Consts         RS - Residents

CKEEP     LB F028:  CONFIG    AB 0108:  CTIMER
LASTA     AB 0103:  READY     AB 0107:  REINIT
WEEPROM   LB F000:  WRITE     LB F00E:

AB 0102:  DATA_COD  PR ------:  ERROR     AB 0106:  EXIT      LB F027
LB F039:  RET       LB F043:  TEMP      AB 0104:  TIMER     AB 0100
```

Figure 2 cont.

Reprinted from

A McGRAW-HILL PUBLICATION

Electronics®

5-volt-only EE-PROM mimics static-RAM timing

by George Landers, *Xicor Inc., Milpitas, Calif.*

5-volt-only EE-PROM mimics static-RAM timing

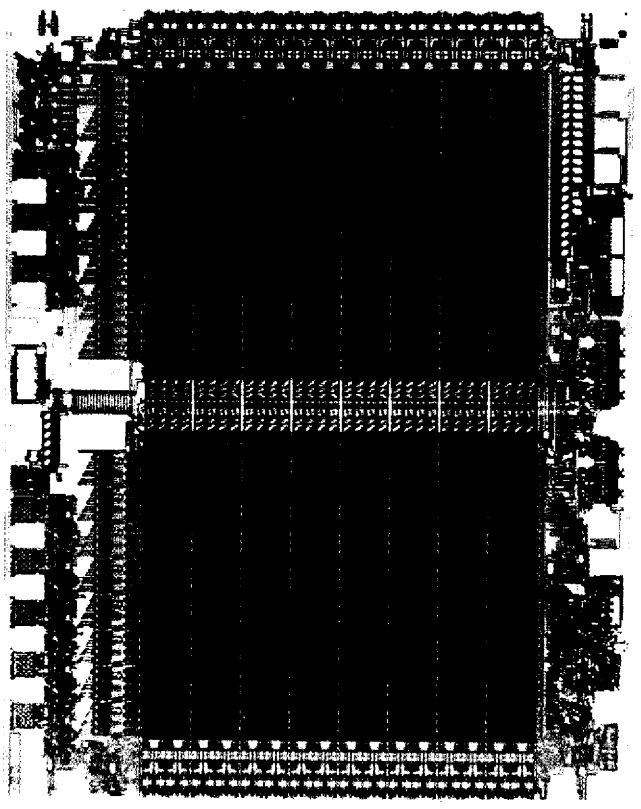
On-chip charge pump, interface latches simplify designs; textured polysilicon enhances tunneling through thick oxides

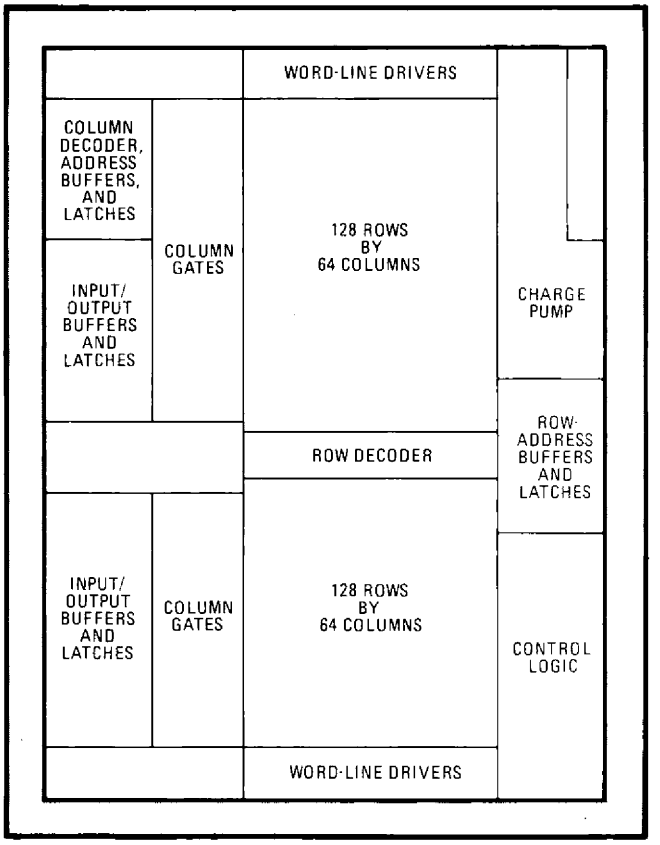
by George Landers, *Xicor Inc., Milpitas, Calif.*

☐ The approaching mastery in fabricating electrically erasable programmable read-only memory conjures up dramatically different system designs. On the most mundane—but perhaps most immediately valuable—level, alterable nonvolatile semiconductor memory will soon banish routine service calls by allowing remote changing of system software. Not far away, if still somewhat tinged with the aura of science fiction, is the vision of self-programmable systems that adapt themselves to a changing operating environment. The catalyst for these advances is an EE-PROM that is simple to incorporate in microprocessor-based systems.

Now being launched by Xicor is a family of EE-PROMs that is the first to do away completely with external supporting hardware (Fig. 1). The chips contain the charge pump that generates a high programming voltage from a 5-volt supply. Further, latches on chip hold the data, address, and control signals during alteration of the cells, which typically takes 5 milliseconds per byte and is timed internally. The part marks the debut of an EE-PROM that can simply be dropped into a standard 24-pin static random-access-memory socket.

The 5-micrometer n-channel MOS technology applied to produce the X2816A 2-K-by-8-bit EE-PROM and the X2804A, a 512-by-8-bit version, is the same as that being used to build the devices in the Novram line of static RAMs with nonvolatile backup arrays [*Electronics*, Oct. 11, 1979, p. 111]. Recently, however, theoretical





1. All aboard. This 16-K electrically erasable programmable read-only memory integrates all its support circuits. A charge pump generates the programming voltage from a 5-V supply; latches hold addresses and data during the internally timed write cycle.

Shedding light on electron tunneling

The present generation of electrically erasable programmable read-only memories using floating-gate structures draws on a reservoir of process development, circuit design, and basic physics. The floating-gate process has been in production for many years and forms the building block of EE-PROMs. The next step toward a practical 5-volt programmable EE-PROM centers on removing the high currents typically used to alter data by avalanche or hot-electron injection. A high voltage can be generated on chip as long as only minute currents are required, as with the new circuit designs.

As far as the underlying physics, the tunneling processes found in most floating-gate EE-PROM devices are described by a theory introduced in the 1920s by Fowler and Nordheim. As shown in the theory, if the emitting surface is flat, very thin oxides of around 100 angstroms are necessary for significant tunneling currents at reasonable voltages of 15 to 20 volts. To the continued puzzlement of researchers, experimental data has fit the theory roughly, but not especially closely.

Xicor purposely fabricates textured emitting surfaces that are covered with low-lying bumps or hills formed during the oxidation of the polysilicon surface. Recently, tunneling theory has been extended to describe this textured-surface geometry with the result that conventional devices are now better understood as well.

The low-lying hills, which serve as the electron emitters, are less than 150 Å high and more than 500 Å across their base. The figure on the right shows the triple-polysilicon tunneling structure in cross section, a scanning-electron-microscope photograph of a typical textured tunneling surface, and the geometry of a typical bump on the polysilicon surface.

Because the oxidation is a well-controlled step, the properties of the emitters are exceptionally regular. The shape of the emitters tends to increase the electric field at the crest of the hills, enhancing the emission of electrons substantially, which allows the use of thick oxide layers of approximately 800 Å. As indicated in the figure, increasing the voltage not only increases the emission, but enlarges the area from which it occurs. This effect explains the discrepancies between experiments and the earlier tunneling theory. The thick oxides have important practical advantages: they are easier to manufacture and lead to increased retention of data.

Until recently, the theoretical work on Fowler-Nordheim tunneling had solved only the limited case of perfectly flat plates. Roger Ellis and H. A. R. Wegener of Xicor recently presented measurements and calculations that agree over a range of eight orders of magnitude in the current. With the aid of the methods of differential geometry, the tunneling characteristics of a textured surface were calculated for the first time. As the scale of the texturing is reduced, the solution naturally reduces to the familiar flat-plate case. The figure on the far right compares the tunneling currents for flat and textured surfaces.

The two structures were designed for the same operating point—a current density of 10^{-4} amperes per square centimeter at 17 V. At low fields, such as are applied to read data, the thick oxide used with the textured surface has only about a thousandth the current of a flat surface. As a result, data retention would be expected to be far longer.

The current from a flat emitter in fact can be modeled much more closely by considering some texturing of its surface. Even single-crystal polished silicon wafers have surface features on the order of 5 Å, and normally processed polysilicon has even larger variations. Thus, Xicor's tunneling structures accentuate features that are always present in floating-gate devices.

work has significantly added to the understanding of the tunneling of electrons from textured polysilicon, the mechanism exploited in all these products (see "Shedding light on electron tunneling," above).

This work explains how a textured surface emits more electrons than a smooth one for a given voltage and oxide thickness. (Scanning-electron-microscope studies of the polysilicon surface show that the texturing consists of low-lying bumps about 150 angstroms high and 500 Å across.) This enhanced emission allows the use of typically 800-Å-thick oxides, instead of very thin, 100-Å layers that are much harder to produce reliably.

Besides being easier to manufacture, thicker oxides lead to increased retention of data. What's more, a 16-K EE-PROM with 5-μm linewidths and 800-Å-thick oxides promises to be more readily scaled down for denser memory arrays than one with, for example, 3-μm lines and 100-Å-thick oxides.

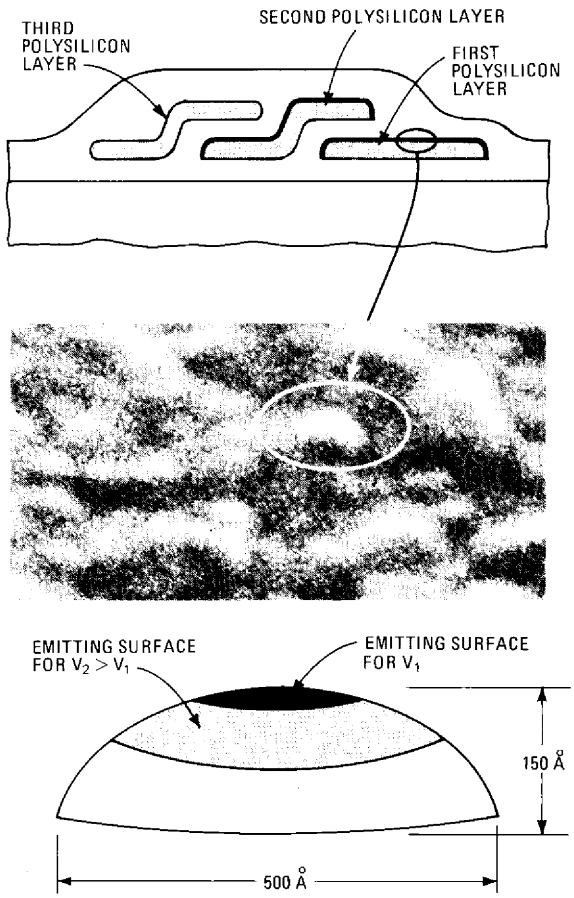
Floating-gate technology along with the architectural features making the parts simple to use present the state of the art in EE-PROMs after a decade of development. Metal-nitride-oxide-semiconductor structures yielded the first nonvolatile memories that were electronically alterable. These devices store data by trapping electrons within the nitride and oxide dielectrics. Besides the problems these devices encounter—data disturbance during the read operation and the loss of data over time—they require multiple power supplies, one of which is often negative, and signal swings beyond TTL levels. All this complicates their incorporation within microprocessor-based systems that work with a single 5-V power supply and TTL levels.

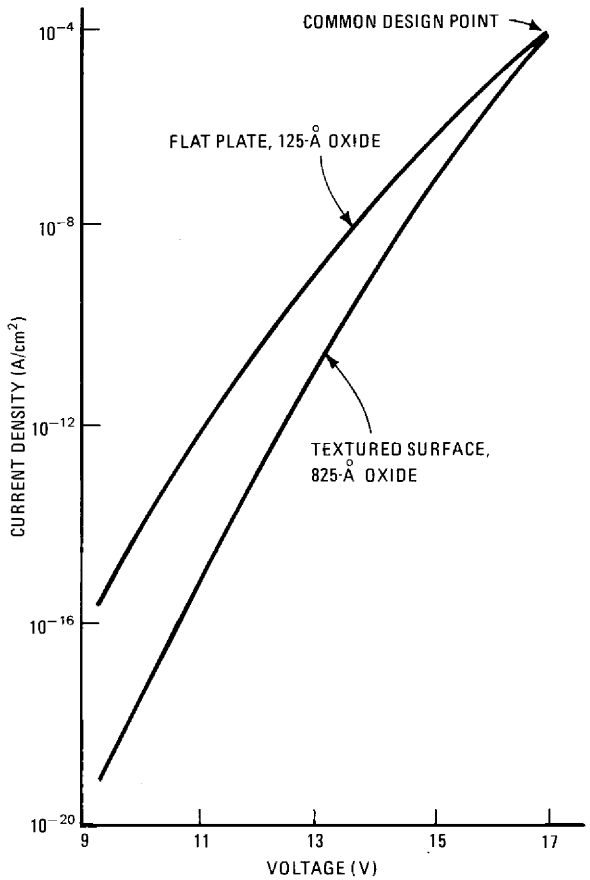
Further complicating their use is the fact that the addresses and data must be stable for the entire write cycle, lasting up to 40 ms. It takes extra hardware to capture these signals and to time the write interval in order to free the processor for other tasks.

Comparing EE-PROMs

The second generation in EE-PROMs was ushered in by the 2816 from Intel Corp. of Santa Clara, Calif. This part stores data by trapping charge on floating polysilicon gates, as is done in ultraviolet-light–erasable PROMs, or E-PROMs, and improves the data integrity compared with MNOS parts. Although the 2816 has a standard pin configuration and uses TTL signal levels, it still requires an externally generated high-voltage pulse for altering data, not to mention latches for holding the address and data signals.

Measured against the 2816, recently introduced third-





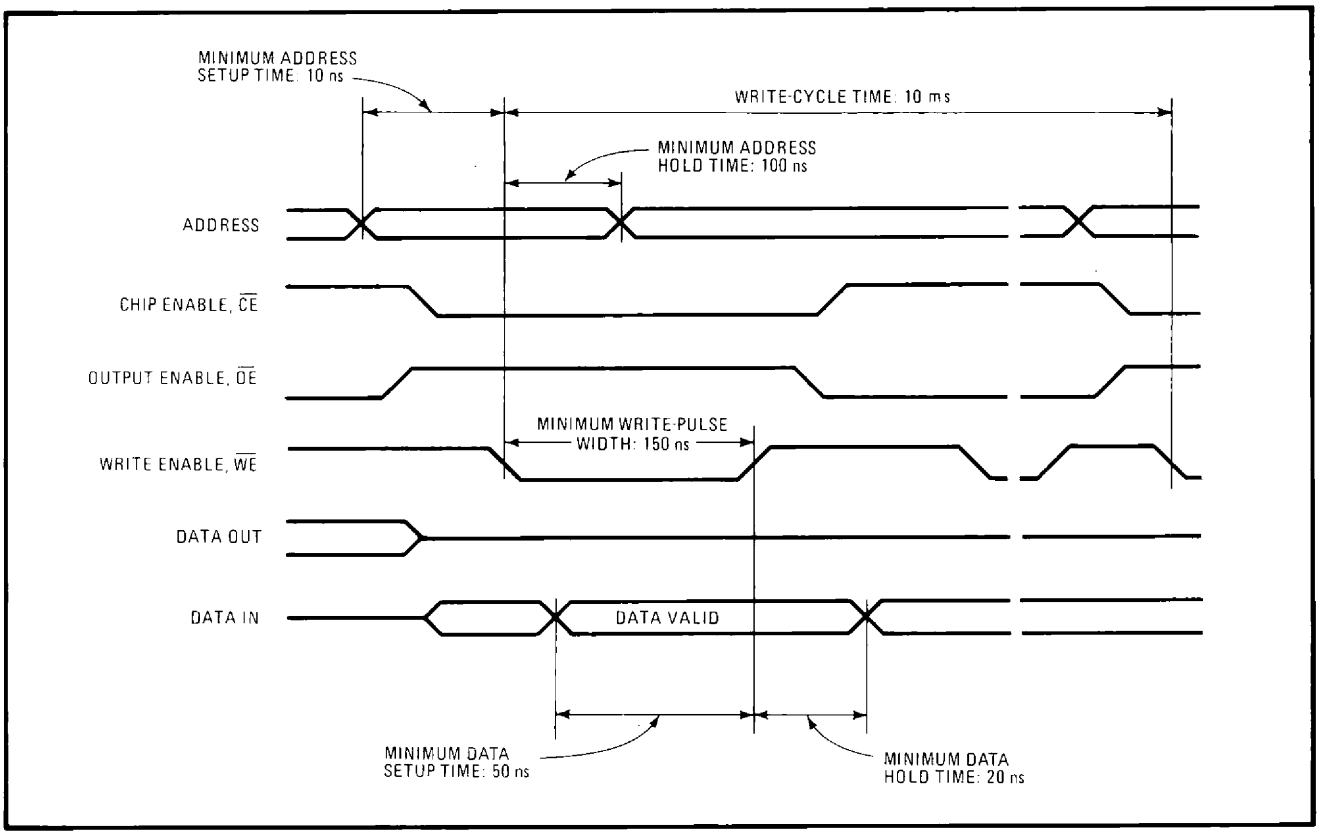
generation parts incorporate some or all of the required supporting hardware on chip (see table). The 2817 from Intel moves the external high-voltage pulse generator onto the chip, so that a fixed 21-V supply is all the user must provide. Though it does include the necessary interface latches, it still requires an external capacitor to time the write cycle. More recently, the 5213 from Seeq Technology Inc., San Jose, Calif., operates from a single 5-V power supply, but still needs latches and external timing. Only the X2816A completely eliminates the external components.

COMPARISON OF RECENT ELECTRICALLY ERASABLE PROGRAMMABLE READ-ONLY MEMORIES

Part	Intel 2816	Intel 2817	Seeq 5213	Xicor X2816A
On-chip charge pump				
Address and data latches				
Automatic erase				
Internal timing of write cycle				
Internal control of write-pulse shape		needs external capacitor		
Maximum erase-write cycle time (ms)	20	75	20	10

All the X2816A's input and output signals are TTL-compatible and the addresses and data are latched so that they need be stable for only 200 nanoseconds to initiate the 10-ms write cycle. Once the write cycle starts, the part self-times the remainder of the operation, freeing the microprocessor and the data bus for other tasks. Freedom from an external timing capacitor or other hardware leads to considerable savings in component and assembly expense as well as in board space. In addition, the cost of design is lower because the part is far simpler to operate.

As can be seen in Fig. 2, the timing of a write cycle for the X2816A is as simple as that for a static RAM. The latches are active only during a write cycle, when they hold the addresses and data to allow the microprocessor to use the bus for other tasks. A write cycle is activated by both chip-enable and write-enable lines going low while output-enable is high. The addresses are latched on the last low-going edge of either the chip-enable or write-enable signal. The data inputs are latched by the



2. Like a static RAM. The write-cycle timing for the X2816A EE-PROM looks much like that for a static random-access memory. Although the write cycle takes a maximum of 10 ms, latches hold the address and data signals, freeing the processor for other tasks.

first of those two signals to return to the high level.

Unlike with most EE-PROMs, there is no need to precondition the data at the desired address before the write cycle, for the X2816A automatically performs an erase function immediately after the cycle starts. Both the erase and write of the data occur during the 10-ms write cycle. The condition requiring the output-enable signal to be high to initiate the write cycle ensures that the part will not be mistakenly programmed when the power is switched on or off.

A compatible part

Conveniently, a socket designed for one of the earlier EE-PROMs can accept an X2816A as well. An internal detector on the write-enable pin senses a signal above 12 V and initiates the internal write cycle (and thus the part may second-source the 2816). This high-voltage signal is used only to detect the system's request to write data—otherwise, the part draws virtually no current from the high-voltage supply.

Further, because of their internal control over the write cycle, the Xicor EE-PROMs can plug into the standard sockets of 2-K-by-8-bit static RAMs. They will operate with the signals normally applied to a RAM, with the only restriction being the delay of 10 ms after starting a write cycle before accessing data. As mentioned already, the X2816A is not on the bus and requires no servicing or supervision during this 10-ms wait. Since the parts time their own write cycle, other EE-PROMs may be updated while a write cycle is continuing on the first unit.

The 10 ms quoted is the maximum delay for writing—the typical delay is only half that. By polling the part during its write cycle, a user can usually reduce the waiting time. One method is to place a particular byte of data at some address and then ask for data from that address during the write cycle. If the data that is retrieved checks against the data written, the part has finished its cycle.

With the cost of a single service call to modify a system in the field mounting toward $200, no doubt the system that can be serviced from afar will be an early development goal. With an EE-PROM plus a modem or other communication method, a telephone call suffices to download the program and configuration data pertaining to all or some of the systems tied together in a network.

A prime application for this technique would be a system of point-of-sale terminals for a market chain. Pricing for items could be dumped to all terminals in the system by calling each store and modifying the price look-up table in each terminal. Similarly, gasoline prices at computer-controlled service-station pumps may be remotely updated.

Indeed, from here it is only a simple step to imagine writing programs that learn as they go. A terminal might analyze the way it was being used and adjust itself for optimum performance in a particular application. By the same token, the next wave of automated manufacturing systems may calibrate themselves, hold information about the steps that have been completed, then interrogate themselves to determine their point in the manufacturing process. □

Reprinted from **ELECTRONICS**, June 30 1982 copyright 1982 by McGraw-Hill, Inc with all rights reserved

Publisher's letter

Though nonvolatile semiconductor memories have been commercially available for a decade or so, the early versions were no joy to work with—multiple power supplies, high voltages, and slow writing discouraged potential users. The push to create a simpler part is yielding noteworthy results, particularly the 16-K electrically erasable programmable read-only memory that Xicor Inc. describes on page 2.

Solid state editor Rod Beresford first heard about the company's new chip back in January. "At the time," he recalls, "we had just published our annual markets forecast, in which we were projecting that consumption of EE-PROMs would nearly quadruple by 1985, to over $330 million. Many of those parts will be going into microprocessor-based systems, where 5-volt power supplies and TTL signal levels are the only way of life."

Xicor's X2816A gets high marks for ease of use. In fact, it's not only microprocessor-compatible, it's a truly self-supporting EE-PROM that's as simple to use as a static random-access memory. Beyond those features, though, our editor was struck by the research at Xicor on the electron tunneling that provides the storage mechanism in EE-PROMs. "I studied tunneling in school," notes Rod, "and can appreciate what the Xicor researchers were up against in trying to get a better fit between theory and experiments. I think they succeed admirably."

Xicor

Understand your application in choosing NOVRAM, EEPROM

Richard Orlando, Xicor Inc., Milpitas, CA

As appeared in EDN Magazine

Understand your application in choosing NOVRAM, EEPROM

Examining how NOVRAMs and EEPROMs serve various applications illustrates the memory devices' capabilities and simplifies device selection.

Richard Orlando, Xicor Inc

If your system design calls for electrically erasable nonvolatile data storage, you can simplify the selection of semiconductor memory for that task by choosing from among four basic types—NOVRAM, EEPROM, EAROM and battery-backed CMOS RAM. Assuming that you've examined the system-level tradeoffs among these memory types (EDN, April 14, pg 135) and have narrowed your choice to the first two, use the information presented here to understand the detailed tradeoffs and design considerations underlying NOVRAM and EEPROM use. In some application classes, either memory type functions adequately; in others, you have a clearcut choice. And in still others, consider taking advantage of both—an approach that often results in cost reductions and enhanced features.

NOVRAMs use multiple technologies

First, however, understand how each memory type works. Nonvolatile static RAM (NOVRAM) combines two memory technologies on one monolithic chip. In **Fig 1**, the NOVRAM shown contains 1k bits of static RAM and 1k bits of electrically erasable PROM (EEPROM). The device comprises cells that in turn each contain one cell of each memory type, rather than housing two separate memory arrays (see **box**, "Anatomy of a NOVRAM cell").

In this NOVRAM, data gets read and written exactly as in a standard static RAM. In addition, the $\overline{\text{Store}}$ signal transfers each RAM cell's data into a shadowing EEPROM cell; EEPROM-stored data gets reloaded into the RAM via the $\overline{\text{Recall}}$ signal. Note that the EEPROM-cell portion is accessible only through the RAM portion.

One of this device type's most powerful features is its ability to transfer the entire RAM contents into nonvolatile storage in one operation, initiated by bringing the TTL-compatible $\overline{\text{Store}}$ LOW. The operation takes less than 10 msec, and once data is stored in this manner, only another store operation can alter it—even if the chip loses power.

Generating $\overline{\text{Store}}$ in the event of a power failure therefore saves the RAM contents, subject only to power remaining on the chip for the next 10 msec. RAM data can also be changed without disturbing the shadowing EEPROM, allowing the system to manipulate two separate groups of data.

EEPROMs offer greater density, fewer features

EEPROM, your other major memory choice, resembles UV-erasable EPROM. Unlike EPROM, however, it can be written electrically in circuit; it needs no prior erasure by exposure to ultraviolet radiation.

First-generation EEPROMs are merely electrically

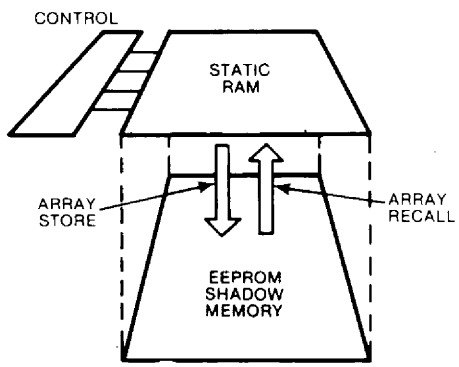


Fig 1—Nonvolatile static RAM (NOVRAM) *is organized so that each static-RAM bit is overlaid on a bit of nonvolatile electrically erasable PROM (EEPROM).*

Careful analysis simplifies the EEPROM vs NOVRAM choice

alterable ROMs (EAROMs). They're reprogrammable only after an entire memory array (or at least one page) is electrically erased. Similarly, second-generation devices require erasure of individual bytes before programming. Third-generation EEPROMs, however, automatically and internally erase a to-be-written byte as part of the write cycle; they also contain much of the required voltage-generating and pulse-shaping functions on chip.

Two examples of third-generation EEPROMs currently in production are the Intel 2817 and the Seeq 5213. The 2817 latches the data to be written and eliminates the need for prewrite erasure. However, it requires an external high-voltage supply as well as a timing capacitor for deriving internal timing signals.

TABLE 1—EEPROM/NOVRAM COMPARISONS

	NOVRAM (X2212)	EEPROM (X2816A)
Density (bits)	1024	16,384
Price (1k level)	$9.00	$23.00
Cost/bit	$0.0088	$0.0014

The 5213 generates the high voltage on chip but requires external latches that hold the data and address valid during erase and write operations.

Fourth-generation EEPROMs are characterized by

Anatomy of a NOVRAM cell

NOVRAM-cell operation depends on a phenomenon termed Fowler-Nordheim tunneling. In the NOVRAM, a layer of oxide isolates a gate from an underlying section of polysilicon. Applying a large positive voltage to this floating gate while holding the underlying polysilicon near ground programs the gate.

Specifically, electrons attracted to the floating gate's significantly higher potential tunnel across the separating oxide. As a result, the floating gate acquires a net negative charge from the tunneled electrons.

The cell is erased in a similar manner: The floating gate is held at a low potential while the potential of the top polysilicon layer is raised; the electrons then tunnel across the oxide from the floating gate to the neighboring polysilicon sandwich.

The EEPROM technology employed in Xicor's NOVRAM uses a 3-layer polysilicon sandwich that, when coupled with a 6-transistor static-RAM cell, results in the NOVRAM circuit shown in **Fig A.** The state of the static-RAM cell determines whether the EEPROM cell is programmed or erased during a store cycle.

Capacitance ratios are the key to the data transfer from RAM to EEPROM. If node N_1 is LOW, Q_7 is turned off, allowing the junction between capacitors C_3 and C_4 to float. Because the combined capacitance of C_3 and C_4 is larger than C_P, the floating gate follows the Store-node voltage. When the voltage on the floating gate is sufficiently high, electrons tunnel from $POLY_1$ to $POLY_2$, and the

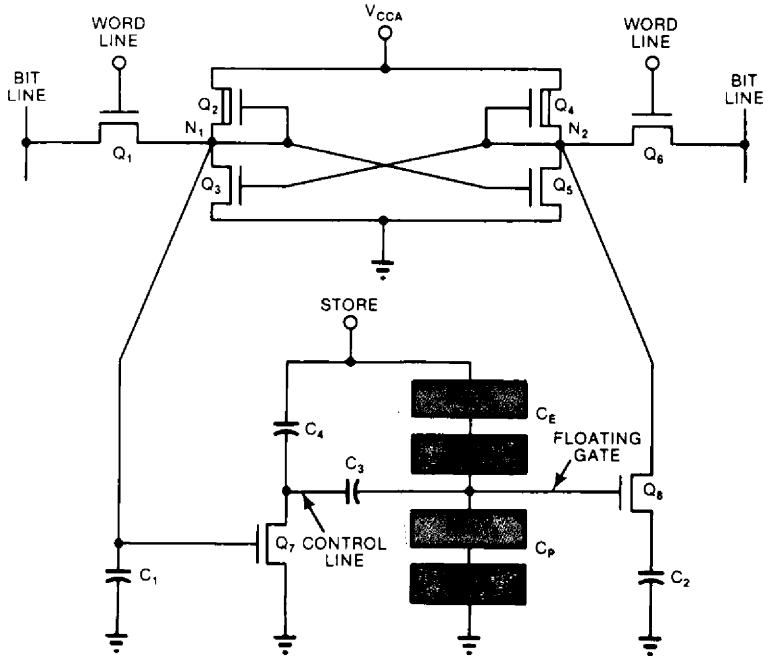


Fig A—A NOVRAM cell *consists of two sections: a 6-transistor RAM and a shadowing 2-transistor EEPROM.*

Article Reprints

on-chip generation of all high-voltage and wave-shaping functions in addition to their use of on-chip latches and self-timing features. Their byte-write requirements are identical to those of static RAM except that the EEPROM write cycle, once initiated by normal static-RAM timings, takes as long as 10 msec. Once a byte-write operation begins, the EEPROM is self supporting, freeing the processor and all external circuitry for other tasks. Read timing to the EEPROM is identical to that of a standard EPROM, RAM or ROM.

An important feature of a fourth-generation EEPROM is its compatibility with currently used RAM, EPROM and ROM. An EPROM- or ROM-based system needs only an additional Write Enable line to each socket to provide retrofitting for EEPROM. This control line allows the changing of data tables and program store without removing the component from the system, as required with EPROMs.

Choosing between NOVRAM and EEPROMs

Many application requirements can be satisfied by either of the two memory types. However, note that although NOVRAM is the most versatile in terms of features and capabilities, the price you pay for its greater intelligence is increased cell size.

Specifically, a fourth-generation EEPROM's cell is small and simple, allowing much higher density storage than in a NOVRAM. The EEPROM is also more efficient as memory-array area increases, thanks to the

gate becomes negatively charged.

If node N_1 is HIGH, Q_7 turns on, grounding the junction between C_3 and C_4. C_3, larger than C_E, holds the floating gate near ground when the Store node gets pulled HIGH. This action creates a sufficiently large field between $POLY_2$ and $POLY_3$ to tunnel electrons away from the floating gate, leaving it with a positive charge.

The recall operation also depends on capacitance ratios. C_2 is larger than C_1. When the cell receives the external $\overline{\text{Recall}}$ command, the internal power supply (V_{CCA}) first goes LOW to equalize the voltages on N_1 and N_2. When V_{CCA} is allowed to rise, the node with the lighter capacitive load rises more rapidly. The flip flop's gain causes the lightly loaded node to latch HIGH and the opposite side to latch LOW. If the floating gate has a positive charge, C_2 is connected to N_2 through Q_8, and N_2 latches LOW. If the floating gate has a negative charge, Q_8 gets turned off and N_1 experiences the heavier loading.

A major task in the development of the NOVRAM was to reduce the amplitude and simplify the waveform of external voltages needed for programming or erasure. Earlier devices required carefully shaped pulses with amplitudes exceeding 20V.

The first step in the cell design was to reduce the internal voltage level presented to the cell to initiate electron tunneling. The voltage magnitude required for programming a floating gate is related to the intensity of the electric field generated at the oxide-polysilicon interface by that voltage.

Electric-field strength at the oxide-polysilicon interface can be increased by using an extremely thin oxide, on the order of 100Å. A second technique uses textured polysilicon to locally enhance the field at the surface and achieve Fowler-Nordheim tunneling. It achieves better data retention.

Once the internal voltage-level requirement was reduced, a key achievement in device design was the on-chip generation of the high-voltage pulses needed to program or erase an individual cell. A Store-voltage generator **(Fig B)** provides the solution; it uses a 32-stage capacitor/transistor charge pump.

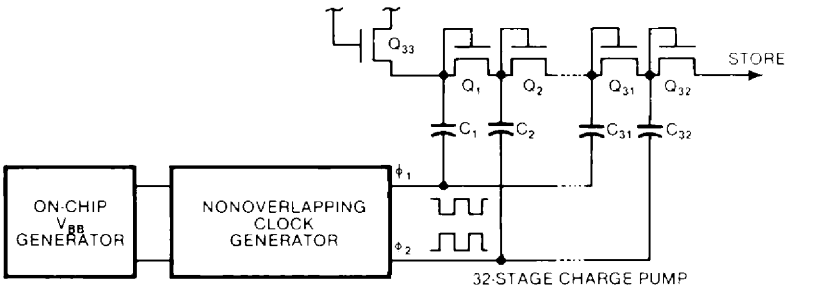


Fig B—A 32-stage charge pump *internally generates a NOVRAM's high Store voltage, allowing the device's NOVRAM and EEPROM sections to operate from 5V.*

Each NOVRAM cell combines RAM and EEPROM

TABLE 2—EEPROM/NOVRAM DATA-CAPTURE SPEEDS

	NOVRAM (X2212)	EEPROM (X2816A)
Byte-Write Time	256 × 1 μsec	256 × 10 msec
Store Time	10 msec	0
Total Time	10.26 msec	2.56 sec

decrease in the relative proportion of support-circuitry area required. Therefore, EEPROMs are more likely to be the device of choice if your application needs large amounts of memory.

The larger cell size and more extensive on-chip support that gives NOVRAM its added capabilities also results in a higher cost per bit, which might not be justified in applications that don't require all of a NOVRAM's features. Consider, for example, the cost-per-bit comparison between the X2212 256×4-bit NOVRAM and the X2816A 2k×8-bit EEPROM (**Table 1**): NOVRAM cost per bit is more than six times greater than that of EEPROM.

However, cost-per-bit ratios can be deceiving for systems requiring a minimum amount of nonvolatile memory. Lower density nonvolatile memories often are more cost effective in a NOVRAM configuration. The smallest NOVRAM currently available, the 64×4 X2210, is also the least expensive 5V device.

Another selection factor to consider is the required write time. An EEPROM requires a relatively long write time (10 msec/byte max), while NOVRAM write time is that of a typical static RAM. Therefore, NOVRAMs are more suited for applications requiring frequent memory-data changes, while EEPROMs most suit applications calling for infrequent memory writes.

A NOVRAM is also better suited to data-capture applications. **Table 2** compares two 256×4-bit NOVRAMs organized in a byte-wide configuration with a 2k×8 EEPROM in terms of the time needed to store 256 bytes of information. These times assume a 1-μsec/byte max processor write-cycle time. You can see that the NOVRAM's single-store operation makes it much faster. A NOVRAM system can update and store 10,000 bytes of data in the time needed to store two bytes of EEPROM information.

Another important NOVRAM feature is the device's ability to initiate and complete a nonvolatile store of data under external-signal control. This feature can be a key decision criterion in real-time applications such as power-fail re-entrant systems.

Both types serve power-fail-tolerant controllers

As noted, however, many applications can profitably use either device type. One common application in this class centers on retaining important system information in the event of a power loss. In most systems, power failures require reinitialization of the entire system, necessitating the temporary loss of system operation. In real-time control applications, this loss of control can cause expensive and sometimes dangerous failures of the process or equipment being controlled.

Such an application's main requirement is therefore some type of nonvolatile storage upon power failure. A prime consideration in this type of environment is the storage of a fixed amount of data upon receipt of an

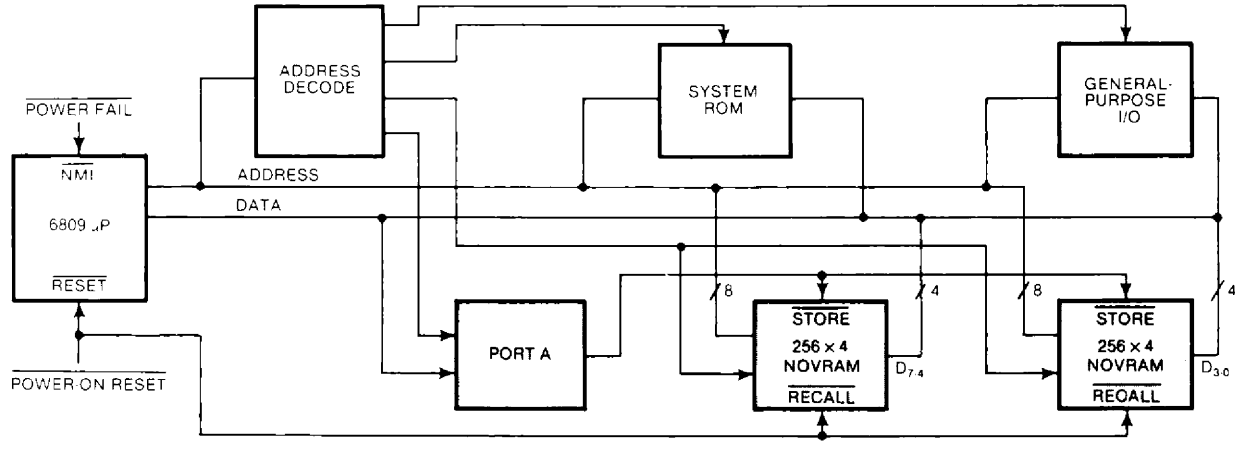


Fig 2—In this controller, *a NOVRAM retains the μP state in the event of a power failure.*

```
;*****************************************************************************
; AN EXAMPLE OF THE POWER-DOWN AND POWER-ON CODE FOR THE 6809 USING
; THE NOVRAM FOR PROCESSOR STATUS STORAGE.
;*****************************************************************************
; LOCATION DEFINITIONS
NVRAMLO         .EQU    0000H
NVRAMHI         .EQU    NVRAMLO + 256.
; THE NOVRAM IS LOCATED AT THE BOTTOM OF THE 6809 MEMORY MAP
DIRTY           .EQU    NVRAMHI ;TEST BYTE FOR POWER FAIL CONDITION
OLDSTK          .EQU    NVRAMHI-6 ;STACK POINTER NONVOLATILE LOCATION
STACK           .EQU    NVRAMHI-7 ;PROCESSOR STACK BEGINNING LOCATION
TEMP1           .EQU    NVRAMHI-1 ;IMPORTANT PROCESS PARAMETERS
TEMP2           .EQU    NVRAMHI-2
TEMP3           .EQU    NVRAMHI-3
PORTA           .EQU    8000H   ;LOCATION OF DATA REGISTER FOR PORT
                .ORG 0F800H
;*****************************************************************************
;                       POWER FAILURE ROUTINE
;*****************************************************************************
PFAIL           STS     OLDSTK  ;WRITE CURRENT STACK POINTER INTO NOVRAM
; AT THIS POINT, THE POWER FAIL INTERRUPT HAS PUSHED ALL OF THE
; CURRENT VALUES OF THE PROCESSOR REGISTERS ONTO THE STACK. THE
; STACK POINTER POINTS TO THESE VALUES.
                LDA     #0A5H   ;LOAD ACCUMULATOR WITH POWER FAIL FLAG
                STA     DIRTY   ;STORE FLAG IN NOVRAM
                LDA     #00.    ;WRITING A 0 TO THE PORT GENERATES
                STA     PORTA   ;A STORE SIGNAL TO THE NOVRAM
LOOP            BRA     LOOP    ;SIT AND WAIT UNTIL POWER DISAPPEARS
;*****************************************************************************
;                       POWER-ON RESET ROUTINE
;*****************************************************************************
RESET           LDA     #0FFH   ;SET ALL OUTPUTS TO A "1"
                STA     PORTA   ;WRITE TO PORT TO KEEP STORE HIGH
                LDA     DIRTY   ;LOAD FLAG TO SEE IF POWER FAILED
                CMPA    #0A5H   ;IT WILL BE A 0A5H IF IT DID
                BNE     INIT    ;IF NOT DO NORMAL INITIALIZATION
                LDA     #00H    ;CLEAR THE POWER FAILURE FLAG
                STA     DIRTY   ;IN THE NOVRAM
                LDA     #00H    ;AND GIVE A STORE SIGNAL
                STA     PORTA   ;TO STORE THE NEW FLAG
                LDA     #0FFH   ;RESET PORT.
                LDS     OLDSTK  ;LOAD OLD STACK VALUES
                RTI             ;RETURN FROM POWER FAIL INTERRUPT
INIT  ;NORMAL INITIALIZATION CODE
;                       .

;                       .

;                       .

                .ORG 0FFFCH
                .WORD   PFAIL   ;POWER FAILURE INTERRUPT VECTOR
                .WORD   RESET   ;POWER-ON RESET INTERRUPT VECTOR
                .END
```

Fig 3—A power-failure-tolerant controller's 6809-μP assembly-language routines *handle both failure store and recovery. NOVRAM handles the stack and other temporary storage.*

NOVRAM doubles as bootstrap and global memory

external $\overline{\text{Power Fail}}$ signal. You can use an EEPROM for this purpose if the processor has sufficient time to recognize the power failure and respond by writing the data into memory. Otherwise, a NOVRAM is the device of choice because it captures data in one nonvolatile store operation.

Fig 2 shows a simple controller that uses a NOVRAM to retain the state of a μP in the event of a power failure. The $\overline{\text{Power Fail}}$ signal generates a μP interrupt, and the NOVRAM stores the contents of all RAM including the μP stack.

Upon interrupt acknowledgement, the μP registers are pushed onto the stack as program control branches to the interrupt routine (**Fig 3**). The routine writes the current stack pointer and a test byte to the NOVRAM, signifying that a power failure has occurred, and then generates a $\overline{\text{Store}}$ signal. The power supply is designed to ensure that the V_C level remains above 4.5V for 10 msec after it generates the $\overline{\text{Power Fail}}$ signal.

Once power is restored, the $\overline{\text{Power-On Reset}}$ signal generates a $\overline{\text{Recall}}$ signal to the NOVRAM. The power-on routine in the μP checks the state of the test byte to see if a process was interrupted by a power failure. If so, the stack pointer gets loaded with the address of the saved processor state, a return from interrupt is executed, and the process resumes.

NOVRAM stores terminal configurations

An application in which NOVRAM is the device of choice lies in the storage of terminal-configuration information, consisting of such parameters as baud rate, data format and parity method. The conventional approach to this task (**Fig 4**) stores data in DIP switches on a pc board somewhere in the terminal; the user must have a terminal manual handy for decoding switch settings to change any of the preset features.

One alternative uses menu-driven configuration modes to set the terminal and a NOVRAM to store the terminal-configuration parameters. The user can easily change the configuration information for specific tasks and retain this data until the terminal loses power.

Upon power-up, a set of predefined default parameters stored in the NOVRAM's EEPROM section goes to RAM, and the terminal is configured. The NOVRAM also allows the user to change default parameters for subsequent sessions by transferring the modified RAM data to EEPROM—in either a general or privileged user environment. The NOVRAM's ability to manipulate two sets of data proves important here because the terminal software operates on the data in the NOVRAM's RAM section, regardless of whether the terminal is in the default configuration or a user-entered one.

In **Fig 4**'s conventional approach, an 8-section DIP switch holds the configuration information. If a switch position is open, the pull-up resistor causes a ONE to appear at the buffer input; a closed switch denotes a ZERO. Decoding the buffer's address and reading the data provides the switch information. If the system needs more than eight bits, the design requires additional switches, resistors, buffers and logic.

If a block of memory addresses is reserved for configuration information, the granularity of the address decoding increases with the number of DIP

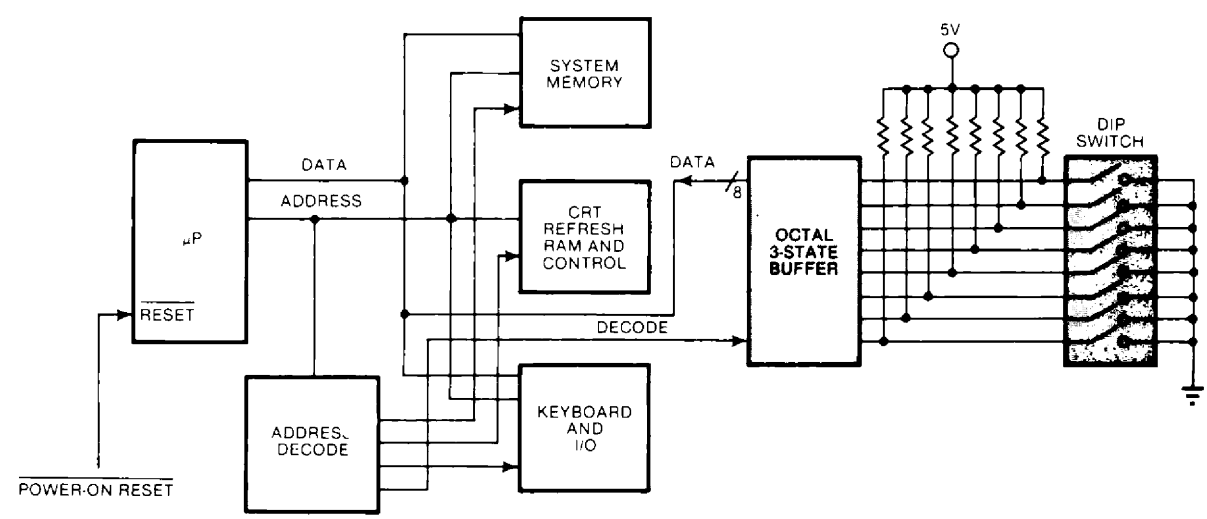


Fig 4—A conventional terminal configuration *uses an 8-section DIP switch to program the terminal's operating parameters.*

In-system data modifications make EEPROMs more versatile than EPROMs

switches required. And you can change the default data only by altering individual switch positions.

The NOVRAM implementation of this system (**Fig 5**) permits the storage of 1k bits of configuration information in one 18-pin X2212. If you reserve an 8k memory-address block for configuration storage, the NOVRAM requires only a single chip-select decode. The only restriction in this arrangement is that four parameter bits get read simultaneously, rather than eight. Note that storing the same amount of information using the conventional approach calls for 128 DIP switches and octal buffers, 1024 resistors and sufficient address decoding to provide 128 separate locations within the 8k field—an address granularity of 64.

A terminal user employs the keyboard to enter operational parameters into the NOVRAM. The user enters a configuration mode when the terminal is in the off-line or local mode. A menu display shows the current terminal configuration; the user moves the cursor and/or strikes a control key to alter the current values. Once the configuration is established, the user exits the configuration mode, and the terminal operates according to the new parameters. The user can also change the default parameters by entering a control signal that places the new configuration mode in the NOVRAM's EEPROM section.

In this application, very few terminals would ever require the NOVRAM's full storage capacity for configuration information. You could therefore employ the unused portion to store other operational and maintenance parameters.

NOVRAM loader provides reusable memory

A system that employs a bootstrap loader during initialization is another prime NOVRAM application candidate. Examples of such applications include single-chip μCs operating in external-memory modes and full-blown systems requiring the maximum allowable memory space. A common approach to this requirement stores the bootstrap program in ROM or EPROM. However, the program occupies memory space that might be used for other purposes during system operation. Because most initialization routines use a relatively small amount of memory space, this approach can be particularly wasteful in space-limited systems.

As an alternative, you can preprogram the bootstrap into the EEPROM section of a NOVRAM. Upon reset, the system generates a $\overline{\text{Recall}}$ signal to the NOVRAM, loading the bootstrap into RAM. The bootstrap program executes, and the NOVRAM RAM section then becomes free for other uses. This design feature even allows bootstrap-program alteration via external control for servicing or software updates.

Fig 6 shows a simple disk-oriented system that uses NOVRAM as a bootstrap memory. After booting, the NOVRAM becomes a global RAM. The device—and

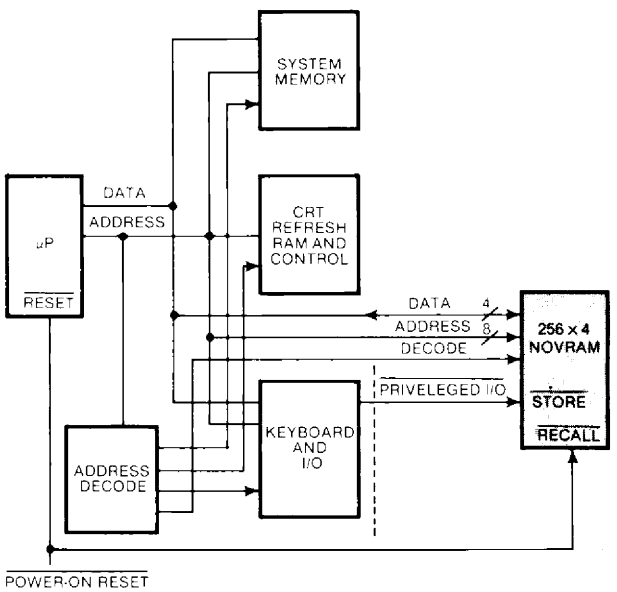


Fig 5—Replacing a DIP switch with a NOVRAM *results in greater versatility for programming terminal operating parameters.*

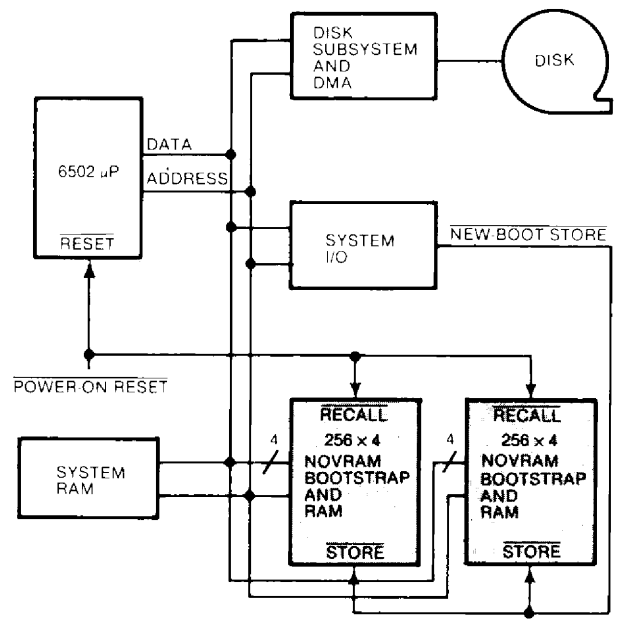


Fig 6—This disk-oriented system uses two NOVRAMs *as bootstrap memory. Once loading is complete, the NOVRAM functions as global RAM.*

```
;*****************************************************************************
; THIS PROGRAM SEGMENT DEMONSTRATES THE OPERATION OF A SYSTEM WHICH
; USES A BOOTSTRAP PROGRAM PRELOADED INTO THE NOVRAM'S EEPROM. UPON
; POWER-ON RESET, THIS BOOT IS RECALLED INTO THE EEPROM'S RAM. THE
; BOOTSTRAP PORGRAM CONFIGURES THE SYSTEM, AND THEN USES THE NOVRAM'S
; RAM AS REGULAR RAM. THE NOVRAM IS LOCATED AT THE BOTTOM OF THE
; 6502'S MEMORY MAP SO THAT THE HIGHEST LOCATIONS CAN CONTAIN THE
; PROCESSOR'S INTERRUPT VECTORS. ACTUAL LOADING OF THE OPERATING
; SYSTEM INTO MAIN MEMORY IS ACCOMPLISHED BY A DMA CONTROLLER, WHICH
; LOADS THE PROGRAM FROM THE DISK.
;*****************************************************************************
DMADATA         .EQU    8000H   ;DMA DATA REGISTER
DMACTRL         .EQU    8001H   ;DMA CONTROL REGISTER
PROGRAM         .EQU    0200H   ;START OF PROGRAM MEMORY

                .ORG    0FF00H
; THIS PROGRAM SECTION IS LOADED INTO THE EEPROM SECTION OF THE
; NOVRAM, AND IS RECALLED UPON POWER-UP.
BOOT            LDX     #0FFH   ;LOAD THE X INDEX REGISTER WITH FF
                TXS             ;TRANSFER TO STACK POINTER AT 01FF
CONFIGUR        ; THIS CODE SECTION CONFIGURES THE DMA CONTROLLER,
                ; AND SETS THE INITIAL LOAD LOCATION AT 0200H, SO
                ; THAT PROGRAM DATA WILL NOT OVERWRITE THE STACK.
SEEK            ; THIS CODE SECTION TELLS THE DMA CONTROLLER WHAT
                ; DISK SECTION IT SHOULD GET THE PROGRAM FROM, AS
                ; WELL AS HOW MUCH DATA TO LOAD.
GO              LDA     #01.    ; 01 IN THE DMA CONTROL REGISTER
                                ; WILL INDICATE DISK DATA LOADING
                STA     DMADATA ; STORE IN THE DMA CONTROL REGISTER
                LDA     #0A5H   ; WRITE TO TEST BYTE TO SIGNIFY BOOT
                STA     TEST    ; IF WE SEE A #0A5H,WE ARE IN BOOT
LOOP            JMP     LOOP    ; LOOP UNTIL DMA CONTROLLER INTERRUPTS
;*****************************************************************************
; NMI FROM DMA INTERRUPT WILL VECTOR HERE. THIS ROUTINE CHECKS THE
; STATUS FROM THE DISK SUBSYSTEM AND DMA CONTROLLER TO INSURE THAT
; THE REQUESTED DATA HAS BEEN LOADED. ONCE A SUCCESSFUL BOOT HAS
; TAKEN PLACE, THE SYSTEM CAN CHANGE THE NMI VECTOR SINCE THE NOVRAM
; NOW FUNCTIONS AS A REGULAR RAM.
;*****************************************************************************
INTERRUPT       LDA     DMACTRL ; CHECK TO SEE IF THE DESIRED PROGRAM
                                ; HAS BEEN SUCCESSFULLY LOADED. IF SO,
                                ; ALL WILL BE ZEROES EXCEPT D0 WHICH
                                ; WILL BE SET TO SIGNIFY THAT AN
                                ; INTERRUPT WAS GIVEN.
                CMP     #01     ; ARE WE LOADED?
                BNE     BOOT    ; IF NOT, WE HAVE AN ERROR, RE-BOOT.
                LDA     TEST    ; GET TEST BYTE TO SEE IF IN A BOOT.
                CMP     #0A5H   ; IF SO, TEST BYTE WILL BE A5 HEX.
                BEQ     PROGRAM ; AND JUMP TO PROGRAM BEGINNING.
ERROR           ;OTHERWISE WE HAVE AN ERROR.
TEST            .BYTE   00.     ; A ZERO IS STORED IN TEST INITIALLY.
                .ORG    0FFFAH
NMI             .WORD   INTERRUPT; NMI WILL GO TO SYSTEM START ROUTINE.
RESET           .WORD   BOOT    ; POWER-ON RESET WILL CAUSE AUTO BOOT.
```

Fig 7—A 6502-μP assembly-language boot routine *is located temporarily in NOVRAM, which forms the highest 256 bytes of memory in this system.*

Use EEPROM if data changes are byte size and infrequent

hence the bootstrap routine—is in the highest memory segment so it can hold all the interrupt vectors. μPs such as the 6502 and 6800 use these locations for reset and interrupt pointers.

In the bootstrap program (**Fig 7**), the reset vector for the 6502 μP points to the boot routine. **Fig 6**'s two NOVRAMs reside in the highest 256 bytes of the address map. Upon power-up, the NOVRAM's EEPROM section gets loaded into the device's RAM section. The μP then initializes the stack pointer, and the DMA controller begins a data transfer from the disk. A test byte gets set to show that a boot process is under way.

Once the DMA transfer begins, the μP loops until an interrupt signifies that the operation is complete. The μP vectors to the interrupt-handling routine, which determines if a valid DMA has occurred. If an error has occurred, the program causes a jump to location Program, where the first byte of the loaded program resides. The NOVRAM RAM is then free for general use. Note that you must take care not to accidentally overwrite the interrupt and reset vectors, located in the highest memory locations.

EEPROM stores controller parameters

Turn now to some applications in which an EEPROM is the device of choice. One such task is the storage of coefficients in PID (proportional integral-differential) controllers.

Modern control applications such as the PID algorithm are characterized by two basic qualities. First, they are computationally intense. Second, their ability to precisely control a set condition is based on their knowledge of the effects of their outputs. This knowledge results from deriving the various controller coefficients via calculations: Each controller output must be calculated with reference to the previously defined term.

If a PID system loses power, it must resynthesize all data before it approaches the level of performance exhibited before the power loss. The data tables for each control task are fairly large and require a substantial amount of memory. Therefore, a controller might use EEPROM for algorithm-coefficient storage.

Note also that most PID-controller deviations result from the sensitivity of the system's sensors as well as the response time and accuracy of the control outputs. These variables might change in a particular unit but are usually the same when power returns to the controller; they need only be updated occasionally as the system runs. An EEPROM's slow write time and fast read time make it ideally suited for this infrequent-write application.

Finally, note that parameters stored in EEPROM are available to the system whenever it's running—whether programmed into the system during initialization or resulting from previous system operation. An EEPROM implementation of such a system thus results in shorter system-interrupt recovery time as well as self-recalibration upon component replacement.

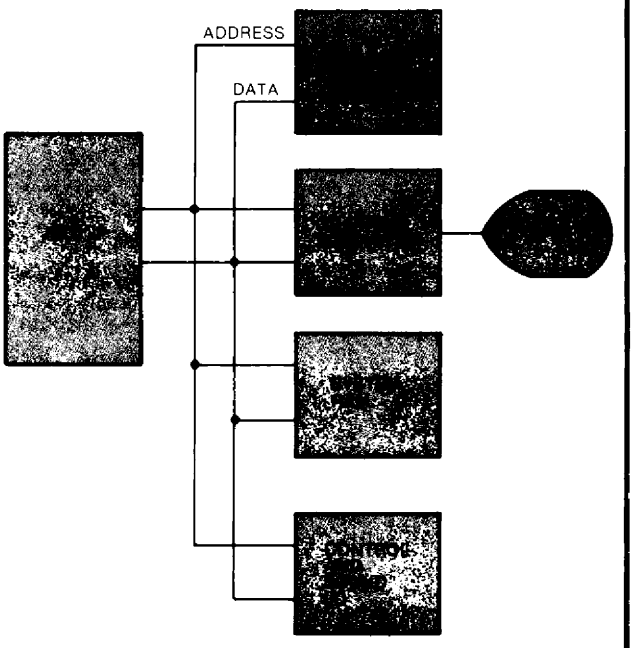


Fig 8—A self-learning video game uses EEPROM *to change difficulty levels after play is completed.*

Self-learning video games use EEPROMs

Another potential EEPROM application centers on the storage of self-teaching or self-modifying code, through which a process or algorithm can tailor itself based on the results of previous executions. Such applications are characterized by updates to program storage, which usually occur relatively infrequently. This high read-to-write ratio of memory access, as well as the densities required in the program store, generally dictate an EEPROM implementation.

An example of this application category is a self-learning video game (**Fig 8**). Such a game's success depends largely on its ability to keep a player interested by continually increasing the level of challenge after repeated plays.

At the end of a certain period (**Fig 9**), the game analyzes the scores and modifies its program (including timing loops and difficulty factors) to present a more complex play to the next group of players. The learning algorithm also makes the game easier to play under

certain conditions, preventing unwarranted increases in difficulty.

The initial game code includes several routines that make the game progressively more difficult to play. These routines get bypassed in the initial program execution by always-executable branch instructions. At

```
;*****************************************************************
; THIS IS AN EXAMPLE OF 6809 GAME CODE WITH BRANCH NEVERS
;*****************************************************************
PLAY            ;THIS SECTION OF CODE IS THE EASIEST
;                       .
;                       .
SECT1           BRA     SECT2   ;WE START OUT BY BYPASSING THIS SECTION
                ;THIS SECTION OF CODE IS MORE DIFFICULT
SECT2           BRA     SECTN   ;WE ALSO BYPASS THIS SECTION
;                       .
SECTN           BRA     ENPLAY  ;WE HAVE N DIFFICULT ROUTINES
;                       .
ENPLAY          LDA     TEST    ;CHECK THE END OF GAME FLAG
                CMPA    #0FFH   ;IT WILL BE AN FF IF GAME IS OVER
                BNE     PLAY    ;IF GAME NOT OVER, GO BACK TO MAIN LOOP
                BRA     FINISH  ;IF IT IS, GO TO FINISH SECTION

FINISH          ;THIS IS THE AREA FOR THE END OF THE GAME CODE. IT MAY
                ;CALL PROCEDURES TO MAKE THE GAME HARDER IF THE
                ;ANALYSIS OF PAST SCORES WARRANTS.
;*****************************************************************
; THIS SECTION OF CODE INCREASES THE DIFFICULTY LEVEL OF THE GAME BY
; REPLACING ONE OF THE BRANCHES AROUND THE MOST DIFFICULT ROUTINES
; WITH A DUMMY BRANCH OR BRANCH NEVER INSTRUCTION. THIS ROUTINE ALSO
; INCREASES A GLOBAL DIFFICULTY FACTOR, ON WHICH MANY OF THE GAME
; PLAYING ALGORITHMS ARE COMPUTED, BY ONE.
;*****************************************************************
DIFCULT         .BYTE   00      ;DIFFICULTY FACTOR
;               TABLE OF BRANCH LOCATIONS
TABLE           .WORD   SECT1
                .WORD   SECT2
;                       .
;                       .
TABLEND         .WORD   SECTN

MAKEHARD        LDX     #TABLE  ;START LOOKING THROUGH TABLE AT THE TOP
LOOK            CMPX    #TABLEND  ;ARE WE AT THE END OF THE TABLE?
                BEQ     NEXT    ;IF SO, GAME CAN'T BE MADE HARDER!
                LDA     [,X++]  ;LOOK AT THE OPCODE AT THE BRANCH, AND
                                ;INCREMENT INDEX REGISTER FOR NEXT LOOK
                                ;(AUTO INCREMENT BY 2 INDEXED)
                CMPA    #21H    ;CHECK TO SEE IF IT IS A BRANCH NEVER
                BNE     LOOK    ;IF NOT, THEN CHECK THE NEXT ONE
                STA     [0,X]   ;OTHERWISE MAKE THE GAME HARDER BY
                                ;INCLUDING A ROUTINE BY CHANGING A BRANCH
                                ;NEVER (INDEXED INDIRECT)
                INC     DIFCULT ;INCREMENT DIFFICULTY FACTOR
ATTRACT         ;ENTER THE ATTRACT MODE FOR THE GAME
;                       .
;                       .
NEXT            ;WE CAN'T MAKE THE GAME HARDER, SO WHAT DO WE DO??
                .END
```

Fig 9—Written in 6809-μP assembly language, *this self-learning video-game program changes branch instructions based on previously obtained scores.*

the end of each play, the system determines from the score whether to make the algorithm more difficult. If so, it eliminates some of the branches around difficult parts of the game software. A simple table stores all of these branches. Other features, including speed parameters and energy levels, can also be stored to make the game more difficult as scores improve. Storing them in EEPROM provides the additional advantage of easy updates and changes in the basic table.

EEPROM and NOVRAM team up

As a final example, consider how you might combine EEPROM and NOVRAM in an automobile navigational system that could direct a driver to a location within a specific city or area. Proponents of this approach envision beacons located throughout an area, notifying each in-car computer of the car's current location. Provided with this information, a local electronic map and the desired destination, the computer would direct the driver along the most efficient route.

Data-storage requirements would be extensive, implying the use of EEPROM. After all, the system must not only be programmed with a map of the area roads but must also be able to select between many possible alternatives based upon continuously changing factors such as time of day and known construction areas. Using EEPROM would allow the car's driver to load the navigational computer upon entering a location such as a filling station.

A NOVRAM would also prove critical to this application. It would contain rapidly changing current information, which would get transferred to the NOVRAM's EEPROM section upon reaching a destination. The approach allows power removal from the system while the car is parked, eliminating battery drain. Restarting the vehicle would transfer the current data from the NOVRAM's EEPROM section back to its RAM section.

Fig 10 shows how the hardware could be implemented. Map information, stored in EEPROM, gets changed as necessary via the map-download controller, a serial interface over which the data is transmitted. The transmission rate is low because the map data is written into EEPROM, which specs a slow write cycle.

The system has two main interfaces—to the driver and to the vehicle. The former consists of a keyboard for input and a CRT for display of the map and other information. The latter receives data such as mileage and speed so that the system can monitor the driver's progress along a given route. **EDN**

Author's biography

Richard Orlando is product marketing manager at Xicor (Milpitas, CA), where his duties include product development. He is a member of the IEEE Computer Society, the ACM, Tau Beta Pi and Eta Kappa Nu. Rick holds a BS degree in computer-systems engineering from the University of Massachusetts at Amherst. His interests include research in the areas of distributed processing, reconfigurable processor architectures and μP applications.

Article Interest Quotient (Circle One)
High **476** ***Medium*** **477** ***Low*** **478**

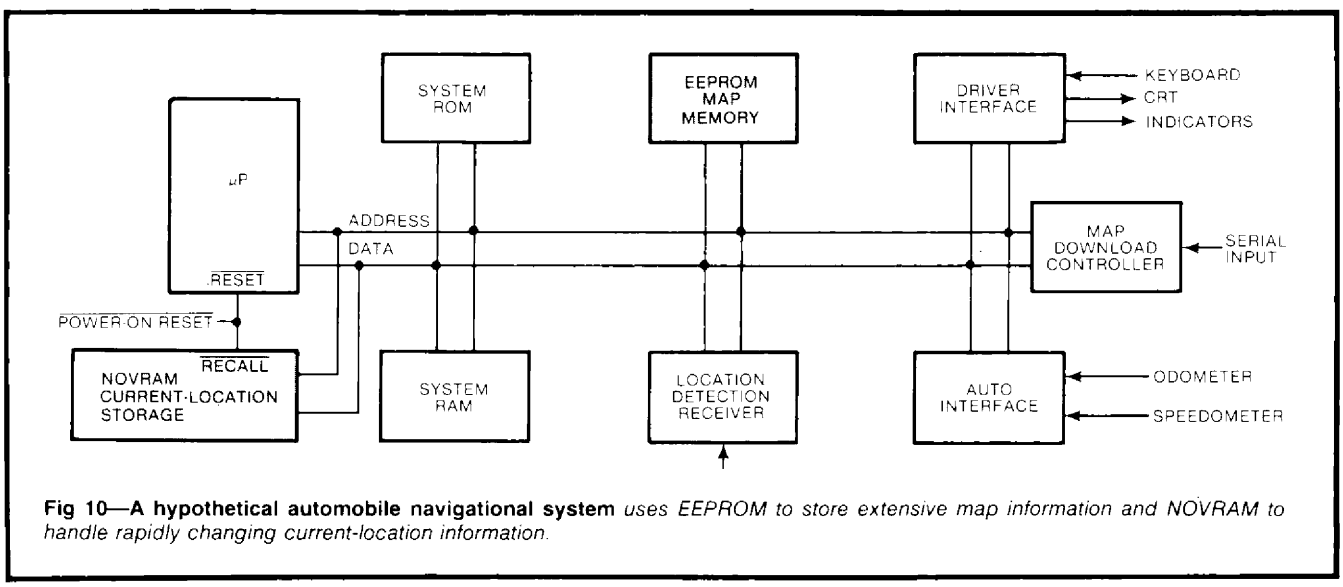


Fig 10—A hypothetical automobile navigational system *uses EEPROM to store extensive map information and NOVRAM to handle rapidly changing current-location information.*

Non-volatile memories keep appliances out of the dark

Richard Orlando, Xicor Inc., Milpitas, CA

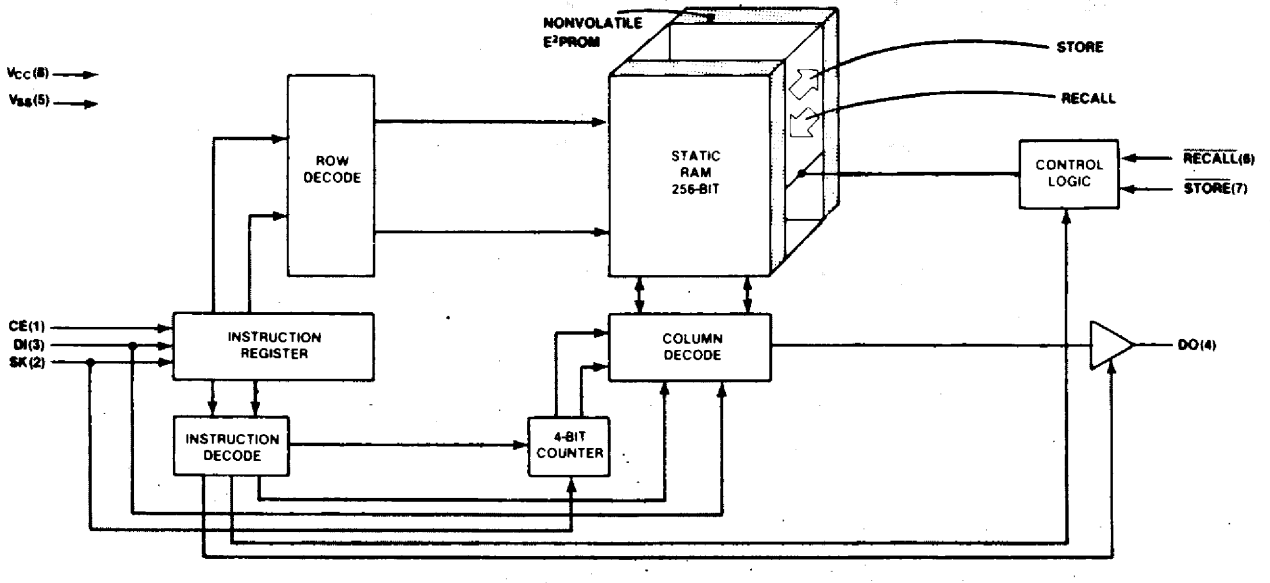
Appliance design has undergone a revolution in recent years. The advent of the low-cost, single-chip microcomputer has opened many applications for these small computers in the appliance market. Initial applications were based upon new types of appliances where digital control was a necessity. Today one sees even the venerable "white goods" using single-chip microcomputers to add features and capabilities to the end products. With this migration to digital control, a need for non-volatile memory has developed, and many new non-volatile memory devices have been made available to the designer.

Appliance control applications have gone through an orderly evolution. The design methods of the past used electromechanical devices, such as switches, relays, mechanical timers and, of course, wafer switches. The requirements of older appliances could be easily satisfied by these devices. Washing machines, for example, using multiplane wafer switches driven by a simple timer could initiate, time and terminate the different cycles of the laundry washing process. And the electronic range allowed simple electromechanical timing of a cooking cycle.

Since the appliance industry has been subject to the whims and attitudes of the consumer, the desired capabilities of appliances have grown as a function of added features. A simple example is the evolution of the home stove controller: first, accurate control over cooking temperature, then the ability to turn off the oven after a programmed time, and, finally, the complete programmable oven that not only turns itself off after a programmed time has elapsed, but also initiates the cooking cycle at a certain time of day.

The increased capabilities of the appliances coupled with the availability of low-cost, single-chip micro-

Fig. 1
FUNCTIONAL DIAGRAM × 2444 (16 × 16)



computers has led to the final step in the evolution, that of full digital control. The use of the microcomputer as a control mechanism allows the designer increased flexibility, reliability and precision in the control process, not easily attainable with the older design methods. Decreased development costs are also possible since a flexible digital controller can be used in a variety of different products, or models of the same product.

Microcomputer designs were not free of their own unique problems, however. The microcomputer interface required to perform the actual control functions was somewhat complex. New issues had to be addressed in terms of product reliability, since the semiconductor devices introduced different failure modes than those exhibited by electromechanical devices. The microcomputer also had a major disadvantage over prior design techniques due to its inherent volatile nature: when the power was removed from the appliance, the microcomputer not only stopped functioning, it lost any data it had maintained based upon the current state of the system.

One advantage that the older electromechanical timers possessed was that if the power went off to the appliance, the control system would maintain the state it was in when the power was interrupted. When the power was restored to the appliance, it would continue from where it left off. One can easily appreciate the irritation of a homemaker who, having left a roast in the oven, returns from errands to find that it had not resumed cooking after a blackout.

Emergence of non-volatility

With many appliance designs, there is a definite need to prevent such untoward situations. Some type of non-volatility is a necessity in appliance design. Since the cost of the appliance is a great concern, this non-volatility must be cost-effective. One of the earlier approaches was that of battery backup on the microcomputer itself, or on a separate CMOS memory in the system. The disadvantages of this approach are based simply on the limitations of batteries and the cost of implementation. Unfortunately, there were not many alternatives until now.

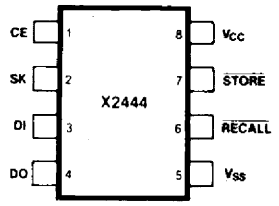
The past five years have seen a remarkable evolution in the emergence of semiconductor non-volatile memories. Unlike the battery backup of the data in either on-board or external RAM, these devices were able to retain data without the external power, in a manner similar to that of an EPROM. The main difference between these devices and the EPROM was their ability to be "rewritter in-circuit, as opposed to being removed from the circuit, erased, and then "reprogrammed" before they were put back into the circuit.

Unfortunately, these early devices were expensive and difficult to use. They required multiple "programming" voltages, extensive support circuitry, and were quite unreliable. These devices, for the most part, were organized for microprocessor "bus" applications, and as such required too many I/O lines for efficient interfacing to single-chip microcomputers, where I/O lines are a precious commodity.

The development of 5v floating-gate, NMOS non-volatile memories eliminated many of the disadvantages of semiconductor non-volatile devices. These devices not only decreased the support circuitry required for their use, but increased the reliability of the devices. Unfortunately, these devices were also designed for "bus" applications and were relatively expensive due to their large densities (>1K bits). A need was recognized in the appliance and other industries for an inexpensive and reliable non-volatile memory designed exclusively for interfacing to single-chip microcomputers.

The Xicor X2444 answers this need. The device is a low-cost 16 × 16 non-volatile static RAM (NOVRAM for short) which features serial interface designed for interfacing to a single-chip microcomputer with a minimum requirement for both I/O

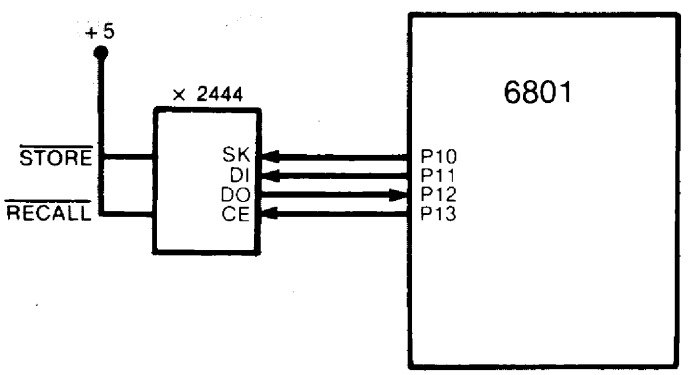
Fig. 2
PIN CONFIGURATION
8-PIN DIP. 300"



PIN NAMES

CE	CHIP ENABLE
SK	SERIAL CLOCK
DI	SERIAL DATA IN
DO	SERIAL DATA OUT
$\overline{\text{RECALL}}$	RECALL
$\overline{\text{STORE}}$	STORE
V_{CC}	+5V
V_{SS}	GROUND

Fig. 3
× 2444-6801 INTERFACE



lines and software. Housed in an eight-pin mini-DIP, the X2444 provides inexpensive non-volatile data storage for both operational and configuration parameters. Its low cost (less than $4.00 in unit quantities) makes it the least expensive non-volatile storage on the market, even rivalling the DIP switch in unit cost, while providing the equivalent of 32 DIP switches in terms of data capacity.

The NOVRAM concept

The NOVRAM idea is not new. Xicor invented this type of memory more than three years ago. The concept is quite simple. Figure 1 shows a block diagram of the X2444. It consists of a 256-bit (16 × 16) static RAM with a 256-bit 5V E^2PROM array overlaid bit for bit in a "shadow" type manner. Two signals, $\overline{\text{STORE}}$ and $\overline{\text{RECALL}}$, control the transfer of data between the E^2PROM array and the static RAM. The STORE function replicates the data which is currently in the RAM into the non-volatile E^2PROM array. In a similar manner, the RECALL function transfers the non-volatile data in the E^2PROM array into the RAM. One can see that by simply performing a STORE during power failure, the data is then retained in the non-volatile E^2PROM , and can be restored to the RAM using the RECALL once power is returned to the system.

The X2444's serial interface method is ideal for microcomputer applications. Figure 2 shows the pinout and signal designation for the X2444. The four-line serial interface consists of a Chip Select (CS), a Serial Clock (SC), a Data In (DI) line, and a Data Out (DO) line. The Data In and Data Out timings are designed to allow the implementation of a single Serial Data line by typing both Data In and Data Out to a single I/O line from the microcomputer, reducing the I/O lines to three. All data transfer to and from the X2444 are performed over this serial interface by either synchronous 8-bit instructions or 16-bit data operations. The X2444 has two external pins, $\overline{\text{STORE}}$ and $\overline{\text{RECALL}}$, for performing the non-volatile operations via hardware control in the event of power failure. The X2444 also includes distinct STORE and RECALL instructions over the serial interface to allow only software control over the non-volatile operations.

The serial interface is accomplished using discrete "bit-banging" from the single-chip micro. An instruction is performed by loading an accumulator with the proper bit pattern, and shifting it out through an I/O line while toggling the serial clock low and then high again between each bit.

The software for this interface is simple, and an example of a 6801 implementation is shown in Figure 3. The software assumes that the X2444 is connected to bits 0, 1, 2 and 3 of the 6801 I/O Port 1. The interconnect between the 6801 and the X2444 is shown in Figure 4. The three main parts of the software segment are three subroutines, SHIFTIN, SHIFTOUT and DRIVE. The SHIFTOUT routine takes the eight bits of data in the A accumulator, and shifts it out through Bit 1 of Port 1. Between each data bit output the clock is toggled. This routine is used for either instruction or data output to the X2444.

The SHIFTIN subroutine gives the X2444 eight clock cycles, and shifts the data from the X2444 into the A accumulator. This routine is used only in the READ instruction. The DRIVE subroutine actually provides the driver to interpret the desired operation and issue the proper sequence of commands to the X2444. It should be noted that this sample interface uses the software-controlled STORE and RECALL commands and leaves the X2444 STORE and RECALL inputs tied to Vcc.

E.g. . . . microwave oven controller

One of the newest appliances in the consumer environment is the microwave oven. This also proves to be an ideal example for the application of a non-volatile memory.

The microwave oven started with a control mechanism which was no more than the simple electromechanical timer borrowed from electric ranges. Since the microwave oven cooks in times which are orders-of-magnitude faster than a conventional stove, it became apparent that an accurate and precise control mechanism was needed. The microwave was one of the first appliances to embrace full digital control using a single-chip microcomputer.

Figure 5 shows a typical microwave oven control system based upon the 6801 microcomputer. The user interface includes a keyboard, alarm and display, while the oven interface includes the magnetron control, door interlock, and an optional temperature probe. Non-volatile memory has been added to the system design through the use of an X2444. The interface method to the 6801 and the driving software are similar to that above. The key difference to note is the addition of an external signal to drive the $\overline{\text{STORE}}$ input. This allows the controller to automatically store the data in the RAM into the E^2PROM upon Power-Failure. The circuitry in the power supply senses a loss of power by monitoring either the ac or unregulated dc levels. Once a power failure has been detected, the power supply circuitry pulls the $\overline{\text{STORE}}$ input low. The power supply circuitry need only ensure that Vcc is held valid to the X2444 for 10 msec, and all of the data in the RAM will be stored into the E^2PROM array. Upon Power-on-Reset, the 6801 issues a RECALL command to the X2444, and all of the data is restored.

The remainder of the microwave control circuitry is fairly standard. A 4 × 4 keyboard provides an input mechanism for the user, while the status indicators and display provide visual feedback. A two-line magnetron control allows the use of variable power levels in the cooking process. Timing is performed using the 6801's internal 16-bit timer which is driven off the 60-Hz reference from the power supply. Standard features include a safety door

interlock and alarm. Optional features are provisions for a temperature probe for magnetron control or temperature-based cooking algorithms. The a-d converter used for temperature sensing has a serial interface similar to that of the X2444, and is placed on the same serial bus. Distinct chip selects enable the X2444 or the a-d converter to be accessed. Many such devices are currently on the market including the new TLC540 from Texas Instruments.

The X2444's non-volatile memory serves many functions in this application. Frequently used recipes or cooking sequences can be stored so that the microwave will sequence through a complex cooking algorithm automatically. The X2444's ability to store the data currently in the RAM into the E²PROM is very useful here. As the cooking process takes place, the 6801 keeps a copy of the preset time and power setting in the X2444's RAM. As cooking time elapses, a location in the X2444 is updated to show the elapsed time. In the event of a power failure, the current values of these variables are automatically stored into the E²PROM section of the X2444. Once power is restored to the microwave, the data in the E²PROM is loaded into the RAM section of the X2444, and cooking continues from where it was interrupted. Intelligence can be added to the control algorithm to compensate for the continued cooking (due to retained heat) that occurs after the power outage.

The X2444's unique NOVRAM architecture makes such an application feasible. Since current E²PROM technology has limitations on the number of times that the non-volatile data can be changed, one would not want to change the contents of the E²PROM each time the timer was incremented. If one were to use a typical E²PROM with a write limitation of 10,000 writes, the device would be worn out in a relatively short period of time at a write rate of one per second. Instead, the X2444 allows the system to update the E²PROM section of the chip only in the event of a power failure while using the unlimited RAM write capability of the X2444 every time the counter value changes.

The X2444's can also be used for a variety of other purposes in microwave design. As was mentioned earlier, one can save development time and money if a universal controller is designed. Many different models could use the same controller simply by adding circuitry

Fig. 5
6801 MICROWAVE CONTROLLER

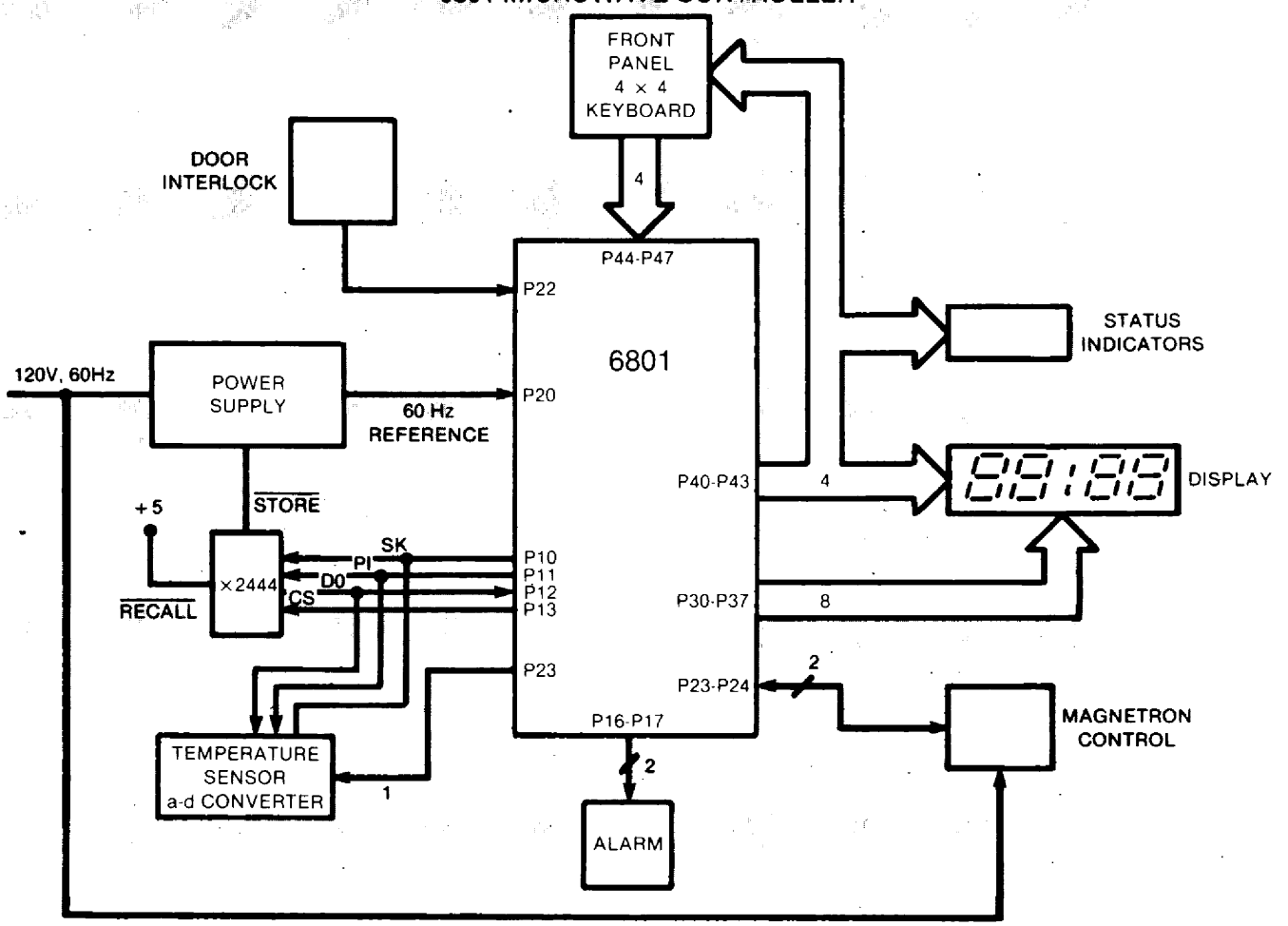


Fig. 4
X2444 DRIVER PROGRAM FOR 6801

```
PAGE -   1                                                    X2444 DRIVER PROGRAM FOR 6801
File: X2444              CODE AS OF AUGUST 22, 1983   Minimal Driver for X2444, 6801 Version 3.0

0000:                                .TITLE "X2444 DRIVER PROGRAM FOR 6801"
0000:                                .ABSOLUTE
2 blocks for procedure code  7440 words left
0000:                                .PROC X2444
Current memory available:    7992
0000:                                .ORG  1000H
1000:                           ;*************************************************************************
1000:                           ;                              6801 X2444 DRIVER
1000:                           ;       ASSUME THAT PORT1 IS USED AS THE 2444 INTERFACE
1000:                           ;       PORT1'S REGISTERS ARE LOCATED AS FOLLOWS
1000:                           ;                 DATA DIRECTION              HEX  0000
1000:                           ;                 PORT                        HEX  0002
1000:                           ;                                  PORT1              X2444
1000:                           ;       I/O 0                      SERIAL CLOCK       SERIAL CLOCK
1000:                           ;       I/O 1                      SERIAL OUT         SERIAL IN
1000:                           ;       I/O 2                      SERIAL IN          SERIAL OUT
1000:                           ;       I/O 3                      2444 SELECT        CHIP SELECT
1000:                           ;
1000:                           ;       COMMANDS ARE PASSED TO THE X2444 ROUTINE BY A PARAMETER IN THE
1000:                           ;       A ACCUMULATOR, WHILE THE ADDRESS IF NEEDED IS PASSED ON THE STACK
1000:                           ;       SERIAL DATA IN OR OUT USES THE TEMPORARY LOCATION TEMP1,
1000:                           ;       WHICH IS A SIXTEEN BIT WORSD.  THE X2444 COMMANDS ARE ENCRYPTED AS
1000:                           ;       FOLLOWS.
1000:                           ;       COMMAND CODE                INSTRUCTION          OPCODE
1000:                           ;       0                           READ                 1AAAA11X
1000:                           ;       1                           WRITE                1AAAA011
1000:                           ;       2                           RESET WRITE ENABLE   11111000
1000:                           ;       3                           STORE                11111001
1000:                           ;       4                           SLEEP                11111010
1000:                           ;       5                           SET WRITE ENABLE     11111100
1000:                           ;       6                           RECALL               11111101
1000:                           ; '1's ARE USED INSTEAD OF DON'T CARE TO DISTINGUISH BETWEEN DATA AND
1000:                           ;  NON DATA OPERATIONS.
1000:                           ;*************************************************************************
1000: 0000                      DIRECTION1          .EQU          00.
1000: 0002                      PORT1               .EQU          02.
1000: 0080                      TEMP1               .EQU          080H    ;RAM STORAGE FOR DATA
1000: 0082                      COUNT               .EQU          082H    ;COUNTER VARIABLE
1000: 0084                      DATUM               .EQU          084H    ;DATA STORAGE
1000: 0086                      ADDRESS             .EQU          086H    ;ADDRESS STORAGE
1000: 0088                      ERRORDATA           .EQU          088H     ;ERROR DATA
1000:                           ;*************************************************************************
1000:                           ;                   PROCEDURE INIT
1000:                           ;       THIS PROCEDURE INITIALIZES THE X2444 INTERFACE
1000:                           ;*************************************************************************
1000: 86 1B                     INIT        LDAA        #1BH            ; B=1011, I/O 0,1 AND 3 OUTPUTS, 2 INPUT
1002: 97 00                                 STAA        DIRECTION1      ; WRITE TO DATA DIRECTION REGISTER
1004: 4F                                    CLRA                        ;SET CE TO 0(INACTIVE), DOUT AND SK TO 0
1005: 97 02                                 STAA        PORT1           ;AND STORE IN DATA PORT
1007: 39                                    RTS                         ;
1008:
1008:                           ;*************************************************************************
1008:                           ;       SHIFTER ROUTINE- SHIFT1
1008:                           ; THIS ROUTINE TAKES THE DATA IN THE A ACCUMULATOR AND CLOCKS IT MOST
1008:                           ; SIGNIFICANT BIT FIRST INTO THE X2444.  THE FLOW IS SHIFT A BIT, TOGGLE
1008:                           ; THE SERIAL OUTPUT(6801) ACCORDING TO STATE, AND TOGGLE SERIAL CLOCK
1008:                           ;*************************************************************************
PAGE -   2  X2444               X2444 DRIVER PROGRAM FOR 6801
File: X2444              CODE AS OF AUGUST 22, 1983   Minimal Driver for X2444, 6801 Version 3.0

1008: C6 08                     SHIFTOUT LDAB       #08.            ;LOAD THE BIT COUNT WITH 8
100A: D7 82                              STAB       COUNT           ;STORE IN COUNTER
100C: 49                        SHIFT1   ROLA                       ;SHIFT BIT INTO CARRY BIT
100D: C6 14                              LDAB       #14H            ;WE SET DATA OUT TO ZERO, WHILE SETTING CHIP
100F:                                                               ;ENABLE.  SERIAL CLOCK IS LOW.
100F: 24 xx                              BCC        TRANS           ;IF BIT IS A ZERO, THEN TRANSMIT
1011: CA 02                              ORAB       #02H            ;IF IT IS A ONE, THEN SET DATA OUT
1013: D7 02                     TRANS    STAB       PORT1           ;STORE THE DATA INTO THE PORT
1015: CA 01                              ORAB       #01H            ;AND SET THE CLOCK FOR A TRANSITION
1017: D7 02                              STAB       PORT1           ;BY WRITING A 1 TO SERIAL CLOCK
1019: C4 1A                              ANDB       #1AH            ;KEEP THE DATA VALID, BUT SET SK TO ZERO
101B: D7 02                              STAB       PORT1           ;AND STORE IN THE PORT
101D: C6 14                              LDAB       #14H            ;TOGGLE CLOCK DOWN, SET DOUT TO 0, BUT KEEP
101F: D7 02                              STAB       PORT1           ;X2444 SELECTED
1021: 7A 0082                            DEC        COUNT           ; DECREMENT THE BIT COUNTER
1024: 26 E6                              BNE        SHIFT1          ;IF COUNT IS NOT ZERO, TRANSMIT NEXT BIT
1026: 49                                 ROLA                       ;ONE MORE ROTATE TO PRESERVE INSTRUCTION
1027: 39                                 RTS                        ;RETURN FROM SUBROUTINE
1028:                           ;*************************************************************************
1028:                           ;                   SHIFTIN ROUTINE
1028:                           ; THIS SUBROUTINE SHIFTS IN 8 BITS OF DATA INTO THE A ACCUMULATOR FROM THE
1028:                                    TS TO ENTER WITH THE CLOCK (ON
                                                 OUTPUT IT INTO
```

external to the 6801 microcomputer. Configuration information can be stored in the X2444 at time of manufacture which the 6801 can then determine upon Power-on-Reset to control the features and functions of its particular microwave. Additional X2444s can be added on the serial bus as user or model options. These optional X2444s require only an additional chip select, and can be used for such features as increased recipe storage or operational modes. The X2444's non-volatile memory also can be used for calibrating the temperature probe and storing the response time of the magnetron to allow quick calibrations or more complex and precise temperature-control algorithms.

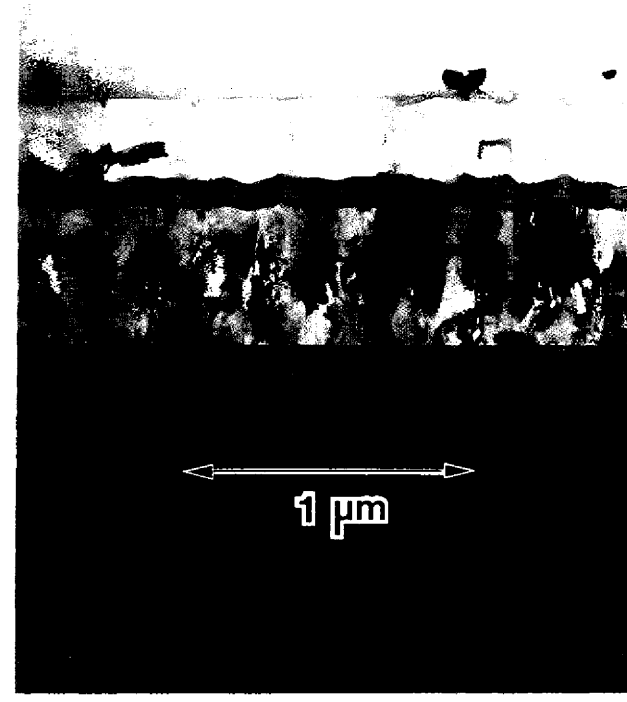
General applications

There are many other areas in the appliance field which are natural applications for the X2444. Since most electronic appliance controllers utilize the single-chip microcomputer, the X2444's serial bus is the ideal solution their non-volatile storage needs.

"User-programmable" parameters such as favorite stations, cooking algorithms or preset time-of-day events all make the appliances more "user-friendly" especially if these parameters are retained in the event of power loss. System configuration parameters can be stored in the X2444 to allow the design of appliances in a modular fashion, substantially reducing development costs while increasing the reliability of each new product. System status saved in the X2444 in the event of a power failure is restored upon Power-On so that the system can complete interrupted tasks as well as ensure that the appliance is left in a safe and stable state.

The availability of the new 5V non-volatile memories allows the appliance designer to add more features and capabilities for a minimum cost. Whether it be used for power-failure data storage, or as user set-up information, the X2444 will make appliance designs less complex, more cost-effective, more fault-tolerant, and easier to use. □

Reprinted from **DESIGN NEWS**, October 1983
© 1983 by CAHNERS PUBLISHING COMPANY



Reliability Reports

NOVRAM* RELIABILITY REPORT

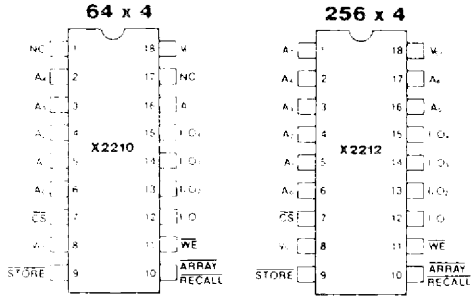
BY BILLY KWONG
& DR. JOHN CAYWOOD

*NOVRAM is Xicor's designation for its nonvolatile RAM memory

This report is based on data collected through July, 1983.

INTRODUCTION

This report covers the Xicor X2210 and X2212 NOVRAM memories. In these memories, each memory bit integrates one bit of static RAM and one bit of electrically erasable-programmable ROM (E^2PROM) into one cell. The controls STORE and RECALL cause the data to be transferred in parallel from all RAM bits into the associated E^2PROM bits and back again. These devices which exemplify Xicor's innovative technology require only a 5V power supply and TTL level signals for all operations, including STORE and RECALL. These two devices employ the same design and processes and are organized 64 x 4 and 256 x 4 for the X2210 and X2212, respectively. Figure 1 shows the pinout for the two parts:



PIN NAMES

A_0—A_8	ADDRESS INPUTS
I/O	DATA INPUT/OUTPUT
$\overline{WE}$	WRITE ENABLE
$\overline{CS}$	CHIP SELECT
$\overline{ARRAY\ RECALL}$	ARRAY RECALL
$\overline{STORE}$	STORE
V_{CC}	+5V
V_{SS}	GROUND
NC	NO CONNECT

Figure 1. X2210 and X2212 pin assignment drawings

Figure 2 shows the functional diagram for the 2212 (256 x 4);

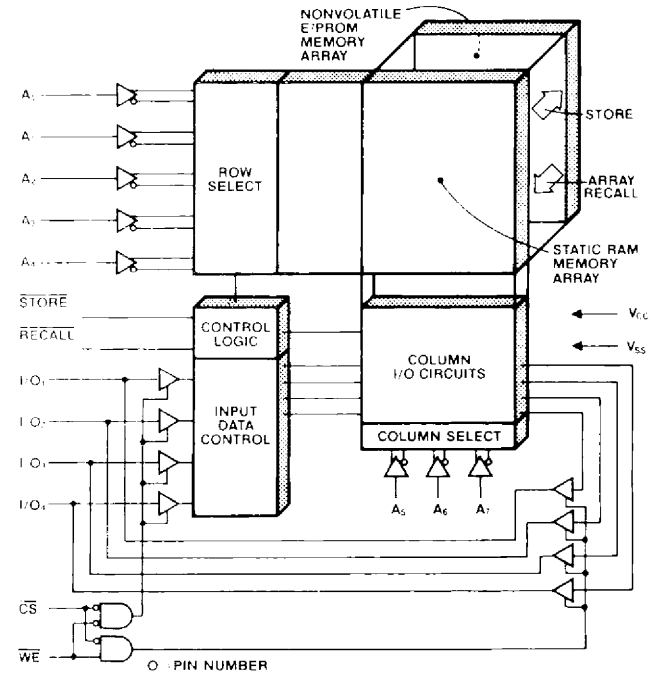


Figure 2. Functional diagram of X2212 memory

Figure 3 shows the package dimensions for the package which is common to both devices.

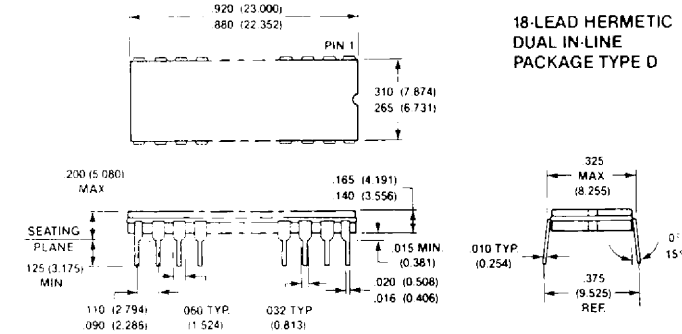


Figure 3. Package outline drawing for memories in cerdip

The bit maps for the three devices shown in Figure 4 and 5 illustrate the physical location of the various address bits.

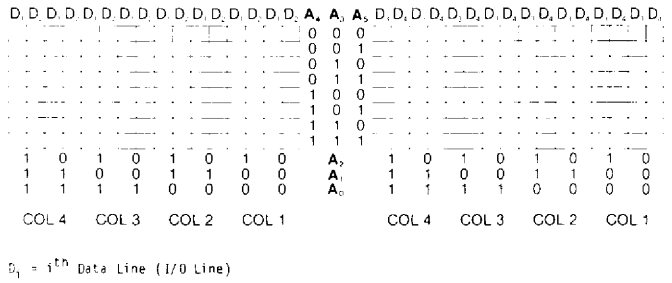


Figure 4. X2210 physical address map

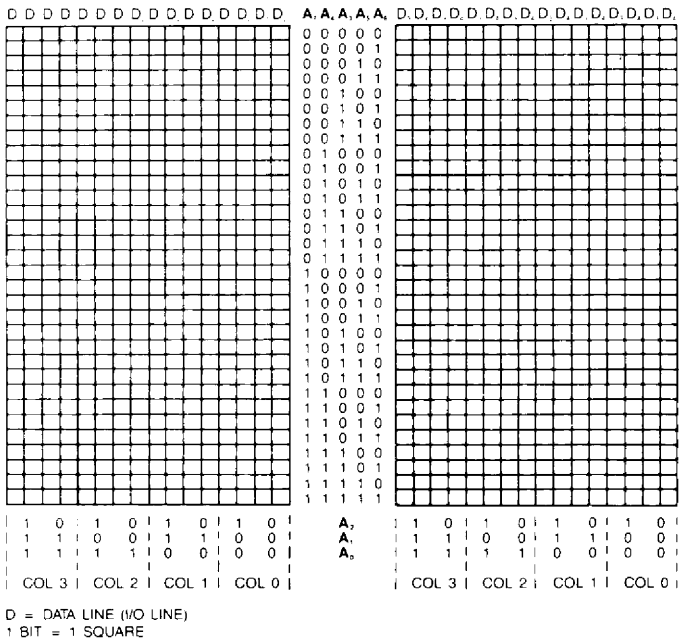


Figure 5. X2212 physical address map

TECHNOLOGY

Xicor NOVRAM memories store their nonvolatile data on electrically isolated polysilicon gates. These gates are islands of polysilicon surrounded by about 800 Å of SiO_2, one of the best insulators known. This is similar to the structure used in UV light erasable EPROM's.

Electrons once trapped on the floating (isolated) gates will remain there unless they receive a large energy input from an outside source (e.g. the ultraviolet photon in the case of UV erasable EPROM's), or until a sufficiently high electric field is applied to distort the energy bands sufficiently to allow Fowler-Nordheim tunneling to occur.[1]

Fowler-Nordheim tunneling, which will be discussed in more detail later in this report, is the mechanism employed to charge and discharge the floating storage gates of Xicor's NOVRAM memories. The storage gates are formed in the second of three layers of polysilicon as illustrated in Figure 6.

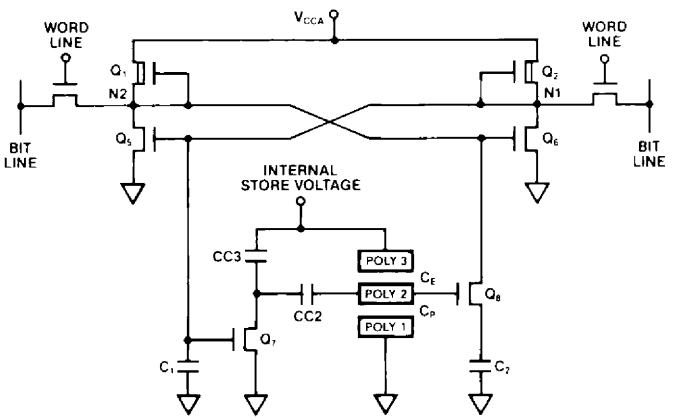


Figure 6. Schematic diagram of a NOVRAM memory cell showing how a standard six-transistor static RAM cell is merged with a floating gate E^2PROM cell.

Electrons move to the floating gate by tunneling from POLY 1 to POLY 2. When the high fields which are used to cause the desired tunneling are not present, the electrons remain trapped on POLY 2.

As Figure 6 illustrates, the NOVRAM memory cell is a conventional six transistor static RAM cell to which a floating gate E^2PROM cell containing two transistors has been added. During the normal READ/WRITE operations, V_{CCA} is fixed at the positive supply level (nominally 5V) and the Internal Store Voltage is fixed at ground. Only the six transistors of the static RAM cell are effective and the operation is exactly that of a conventional six transistor static RAM.

The RECALL operation depends on capacitance ratios. The value of C_2 in Figure 6 is larger than that of C_1. When the external RECALL command is received, the memory array power supply, V_{CCA}, is initially pulled low to equalize the voltages on nodes N_1 and N_2. These nodes equalize quickly to V_{CCA} through the depletion transistors, Q_1 and Q_2. When V_{CCA} is then allowed to rise, the node with the lighter capacitive loading will rise more quickly and turn on the pull down transistor on the opposing side, which will keep the more slowly rising node clamped low. If the floating gate is charged positively, Q_8 is turned on, which connects C_2 to N_2. Thus N_1, which is loaded by the smaller capacitor C_1, rises more rapidly, causing the latch to set with N_1 high and N_2 low. If the floating gate is charged negatively, Q_8 is turned off, which isolates C_2 from N_2 and allows N_2 to rise more rapidly than N_1. Thus the latch is set with N_2 high and N_1 low. During the RECALL operation the Internal Store Voltage remains at ground.

The STORE operation also utilizes capacitance differences to transfer data from RAM to E^2PROM. When node N_1 is low, transistor Q_7 is turned off. This allows the junction between capacitors CC_2 and CC_3 to float. Since the combined capacitance of CC_2 and CC_3 is larger then that of C_p, the capacitor between POLY 1 and POLY 2, the floating gate follows the potential of the Internal Store Voltage. Thus when the Internal Store Voltage becomes high (several times V_{CC}), a sufficient field exists between POLY 1 and POLY 2 to cause electron tunneling and the floating gate is charged negatively.

When node N_1 is high, transistor Q_7 is turned on which grounds the junction between CC_2 and CC_3. Since the capacitance of CC_2 is larger than that of C_E, the capacitor formed between POLY 2 and POLY 3, CC_2 holds the floating gate near ground when the Internal Store Voltage goes high. In this case the high field exists between POLY 2 and POLY 3 and electrons tunnel from POLY 2 to POLY 3, which discharges the floating gate.

This description shows that the Internal Store Voltage is at ground except during the STORE operation. Moreover, during the STORE operation all bits are stored simultaneously, which is possible because the RAM bit associated with each E^2PROM bit acts as a data-latch. These two statements obviate the possibility of any disturb conditions such as those sometimes found in other nonvolatile memories because external voltages are applied to the nonvolatile memory element only while it is being stored.

TUNNELING PHYSICS

Because the innovative aspects of the NOVRAM memory revolve around the nonvolatile storage procedure, it seems appropriate to discuss the storage phenomenon in more detail. As was mentioned above, the storage occurs via a tunneling mechanism first described by Fowler and Nordheim in 1928 and subsequently named after them.[1] The basic idea is illustrated in Figure 7.

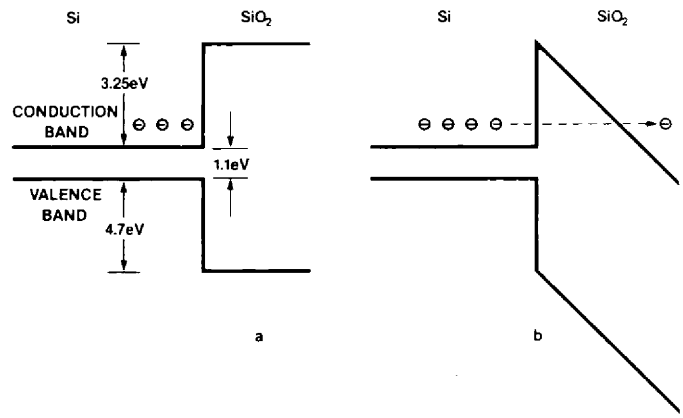


Figure 7. Energy band diagrams of the Si/SiO₂ system in the neutral state (a), and during the store operation (b).

The energy difference between the conduction and valence bands in Si is about 1.1 eV; the energy difference between those bands in SiO_2 is approximately 9 eV. When the two materials are joined, the conduction band in SiO_2 is 3.25 eV above that in Si. The differences in valence band energies is even larger (approximately 4eV). Since the thermal energy of an electron averages only 0.025 eV at room temperature, the chances of an electron in silicon gaining enough thermal energy to surmount the barrier and enter the conduction band in SiO_2 is exceedingly small. This case is illustrated in Figure 7.

Fowler and Nordheim pointed out that in the presence of a high electric field, the energy bands will be distorted as illustrated in Figure 7b. Under these conditions there is a small but finite probability that an electron in the conduction band in the silicon will tunnel through the energy barrier and emerge in the conduction band of the SiO_2 as is illustrated in Figure 7a.

Fowler-Nordheim emission was observed early in this century for the case of electrons being emitted from metals into vacuum, and in 1969 Lenzlinger and Snow observed this phenomenon for the Si-SiO_2 system.[2] The Fowler-Nordheim current increases exponentially with applied field and becomes readily observable (i.e. $J \sim 10^{-6}$ A/cm^2) for the Si-SiO_2 system for fields on the order of 10 MV/cm for the case in which the Si surface is smooth.

It has been known for some time that "enhanced" electron emission currents could be observed for Si-SiO_2 systems for which the Si surface has a texture.[3-5] (Texture in this context means that the Si surface has features of the order of a few hundred angstroms.) These enhanced currents occur at applied fields with values smaller than one quarter of those necessary for the same current from a smooth surface. It has been thought that the enhanced emission occurs because of locally enhanced fields near the top regions of the features on the surface.

Lewis attempted to model Fowler-Nordheim tunneling from a textured surface by calculating current from a number of hemispheres set in a plane.[6] Attempts to quantitatively fit experimentally observed currents with this model have not been very successful. Hu, et al, for example, found that to fit their data to existing theory it was necessary to assume an energy barrier of approximately 1eV between conduction bands in Si and SiO_2.[7] Even with this assumption, the fit was poor because the measured current increased more rapidly than the calculated values, as is illustrated in Figure 8.

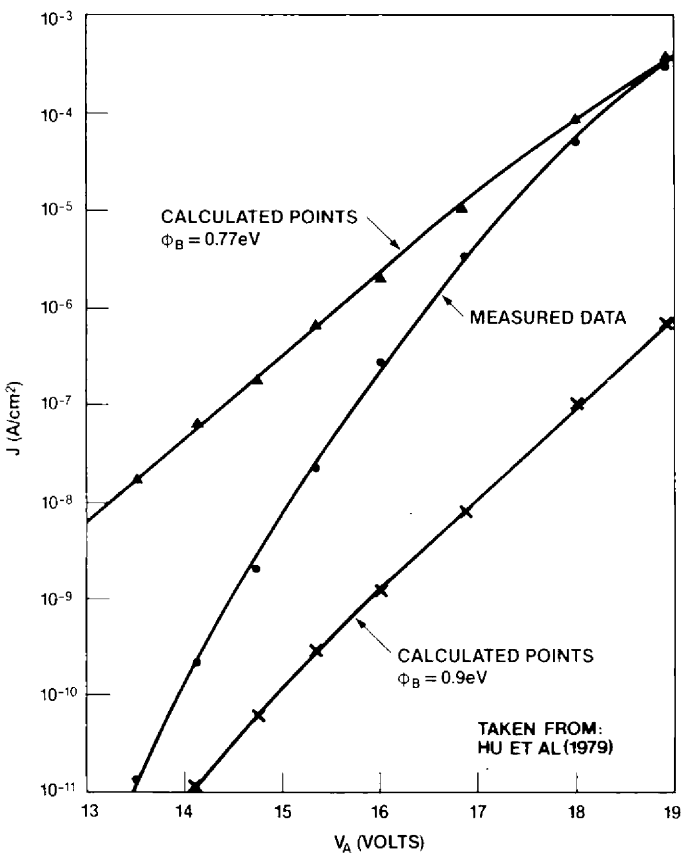


Figure 8. Current vs. applied voltage for current from textured poly surface emitted through 1760 Å of thermal SiO_2 compared with calculations based on previously available theory.

Because of the importance of tunneling to the operation of its products, Xicor has felt it important to adequately characterize and model tunnel emission from textured polycrystalline silicon (poly) surfaces. One avenue of exploration was that of the characterization of the physical topology of the tunneling structure. Figure 9 is a scanning electron micrograph of the top (emitting) surface of a layer of polysilicon from which the oxide has been removed for clarity.

Figure 9. Scanning electron microscope (S.E.M.) photograph of top emitting textured poly surface with oxide removed. The 0.5 μ-long bar gives the scale

As can be seen from the micrograph, this surface is composed of a densely packed array of features reminiscent of a cobblestone street. A count of these features on SEM photos of material from several lots determined that there is an average of about 50 features per square micron.

A transmission electron microscope (T.E.M.) cross-section of a tunneling structure is shown in Figure 10.

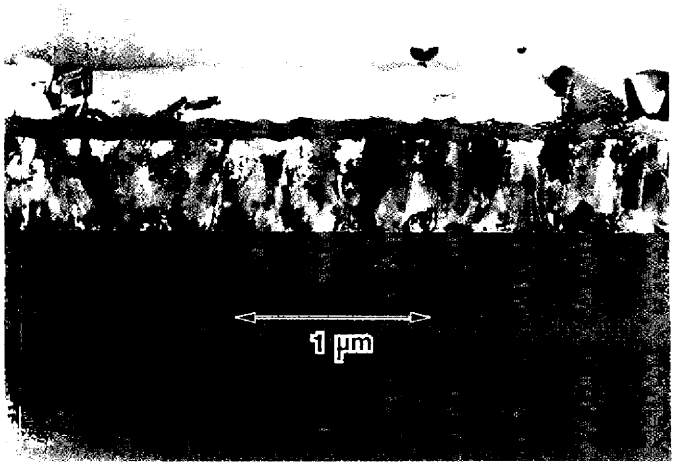


Figure 10. Transmission electron microscope photograph of a cross-section through a textured tunneling structure. The one micron-long arrow shows the scale.

This photo demonstrates the conformal nature of the structure. The top surface of the prior deposition of polysilicon is formed into a series of hillocks 200-300 Å high and 1000-1500 Å across the base. The free surface of the oxide grown on this silicon replicates the silicon surface. Thus, polysilicon deposited atop the prior polysilicon layer subsequent to oxidation has dimples on its undersurface occurring over the already existing hillocks.

The topology of the polysilicon surfaces causes the electric field lines to no longer be parallel as in the case of parallel emitting and collecting surfaces, but rather to diverge and converge in response to the local topology as is illustrated in Figure 11.

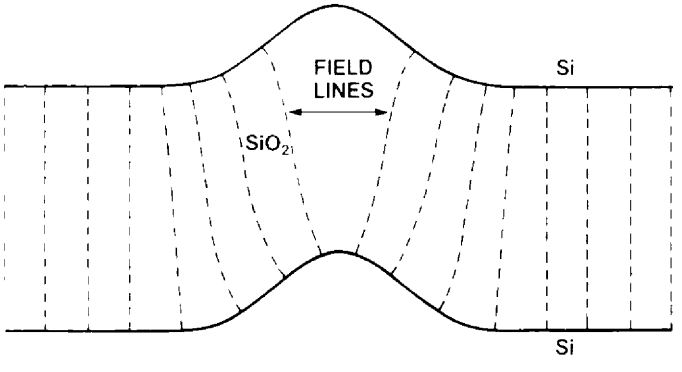


Figure 11. Sketch illustrating the manner in which field lines converge near convex feature and diverge near concave features.

As is shown, the field lines converge near surfaces of positive curvature (i.e., bumps) and diverge near surfaces of negative curvature (i.e., dimples). Since Fowler-Nordheim tunneling depends exponentially on the electric field at the surface of the polysilicon, a good model of the electron emission requires an accurate knowlege of the field over the complete surface.

Until recently, this field problem was one of the unsolved problems of mathematical physics. (It is classically known as the "lighting rod problem"). Roger Ellis, a member of the Xicor staff, has recently solved this problem.[8] The technique used was to transform by differential geometry from Euclidean space into a space in which the field lines were parallel. The electrostatics problem was solved in this space and the solution transformed back into the original space. The tunnel current was found to be described, in a spherical coordinate system, by the equation

$$J_{collecting} = \frac{\left[\int_S \frac{q^3E^2}{8\pi h\phi_B} \exp\left\{\frac{-4(2m)^{1/2}\phi_B^{3/2}}{3\hbar qE}\right\} ds\right]_{emitting}}{\left[\int_S ds\right]_{collecting}} \quad (1)$$

where the electric field, E, is given by

$$E = \frac{V_A}{\xi(s_2) - \xi(s_1)} \left[\left(\frac{d\Theta^2}{dr}\right) + \frac{1}{r^2}\right]^{1/2} \frac{d\xi}{d\Theta} \quad (2)$$

and

$$\xi = \frac{1}{\left|\frac{d\vec{r}}{d\Theta}\right|} \frac{de_t}{d\Theta}(\vec{e}_t)$$

is the curvature of the surface at which the field is evaluated.[9]

This expression has several interesting properties. One is that the field depends on the difference of the curvatures of the emitting and collecting surfaces. Another is that the electric field also depends on the derivative of the curvature of the emitting surface.

To verify the accuracy of this expression, the current-voltage characteristics of a textured polysilicon tunnel structure were experimentally determined and compared with those predicted by equation 1. Because of the field enhancement on the bumps, a higher current is expected at a given applied voltage polariaty which causes electronic emission from the bumps than for that polarity which causes emission from the dimples. Thus, by analogy which a diode, the polarity with the bumps negative (higher current emitted) is called the forward bias direction and the other polarity is called the reverse bias direction.

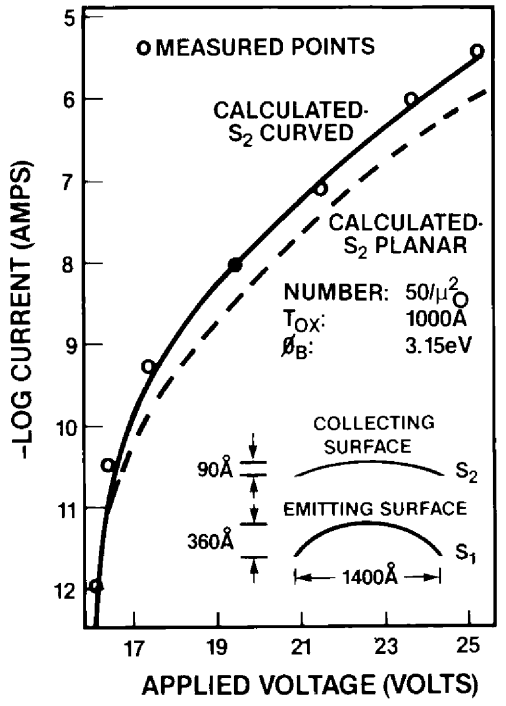


Figure 12. Forward tunneling characteristic comparing the measured data with curves calculated for a concave and a planar collecting surface.

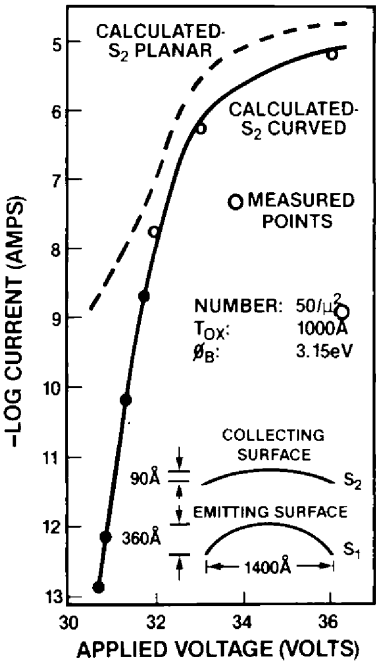


Figure 13. Reverse tunneling characteristic comparing the measured data with curves calculated for a concave and planar emitting surface.

Figure 12 shows the results of comparing experiment and calculation for the forward bias condition. As can be seen, theory and experiment agree within about 20% over seven orders of magnitude in current. The parameters used in calculating the predicted currents were not arbitrarily chosen. The value of the oxide/silicon conduction band barrier, Φ_B, was taken to be 3.15 eV as measured by Weinberg.[10] The feature density, $50/\mu^2$, and the bump base and height are typical of those seen on SEM's and TEM's such as Figures 9 and 10. The dashed line in Figure 12 is the current which would be predicted if the top (i.e., collecting) electrode were flat rather than dimpled. This illustrates the effect of curvature of the collecting surface on the field at the emitting surface.

To further verify equation 1, the reverse bias current was measured on the same device as was used for the forward bias case. The results are compared with the predicted current in Figure 13. This shows about 20% agreement over eight orders of magnitude in current with the same parameters used as in calculating the forward bias case. Clearly, the model does an excellent job of predicting the measured current.

Several points of interest can be observed in Figure 14. One point is that at a voltage at which the forward current is saturating, 25V, the reverse current is unobservable (extrapolation of the measured current predicts $\sim 10^{-27}$ A). Another point is that the current emission predicted from a flat top surface is greater than that predicted from a dimpled top surface. Moreover, the current emitted from a flat top surface decreases more slowly with decreasing voltage than that from a dimpled flat surface. This latter fact is important for data retention as will be discussed below.

In summary, Xicor is able to accurately model the tunnel current emitted from textured poly surfaces through SiO_2 layers. The magnitude of the tunnel current as well as its voltage dependence are dependent upon the surface topology. Thus, we have the means to optimize the emission characteristics. Lastly, the asymmetric nature of the tunnel emission makes possible cell designs which are better adapted to particular requirements than is possible with symmetric tunnel characteristics—just as a diode offers more design possibilities than does a resistor.

DATA RETENTION

As was suggested above, textured poly tunneling structures have a significant advantage in data retention, in comparision with those employing flat surfaces and thin oxides. One basis for this advantage is illustrated in Figure 14.

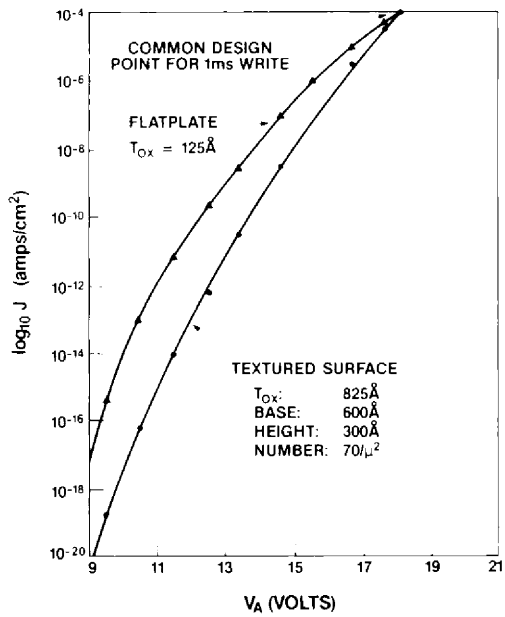


Figure 14. Comparison of calculated tunneling J-V curves for emission from a planar and a textured structure. The devices were designed to have the same emission in the high current regime where programming takes place.

in which the current voltage characteristics of a tunneling device which employs a thin oxide between planar surfaces and a tunneling device which employs a thick oxide between textured silicon surfaces are compared. For this comparison we match the currents in the high current regime, a reasonable criterion since most memories are designed to program in about the same time period (~10 msec.) As can be seen, the same current can be obtained from a smooth surface with an 825 Å thick tunnel oxide. Note however, that at lower values of applied voltage typical of read and storage conditions, the current emitted from a textured surface is approximately four orders of magnitude lower than that from a smooth surface. This implies that a memory which utlizes tunneling from a textured surface should have better data retention and be less susceptible to disturb conditions than one which utilizes tunneling from a smooth surface.

These differences may become even more significant as devices are scaled. It is clear that, in order to scale the memory properly, lower programming voltages are needed so that the isolation widths and device channel lengths can be reduced both in the memory array and in the peripheral circuitry. However, for a typical part which stores data in 3 msec. and must retain it for 10 years, the tunneling current under storage and reading conditions must be at least 10^{11} times smaller than under programming conditions because the retention time is 10^{11} times longer than the storage time. Actually, for margin, one would design for a difference in currents of 10^{13} -10^{14}. For planar surface tunneling structures, this may be a difficult design constraint because the slope of the J-V curve is fixed, which means that the maximum allowed read voltage drops with the programming voltage on a volt-for-volt basis, not proportionately. On the other hand, textured surface tunneling structures, in their current manifestation have a steeper J-V characteristic than planar ones and the J-V characteristic of a textured structure can be tailored to yield a steeper curve if desired. This means that for a given maximum read voltage, a textured structure requires a lower programming voltage which leads to better scaling.

To verify the excellent data retention expected to Xicor NOVRAMs, a study was carried out to measure data loss as a function of temperature. Figure 15 shows log cumulative data loss vs. log time for samples of 100 X2210's at each of three temperatures. Data loss is defined as occurring when the first bit in the array loses data. As is clear in Figure 15, quite high temperatures were required to get appreciable data loss in experimentally useful times. Note that even at 300° C, 2000 hours (~3 months) are required to begin to see data loss.

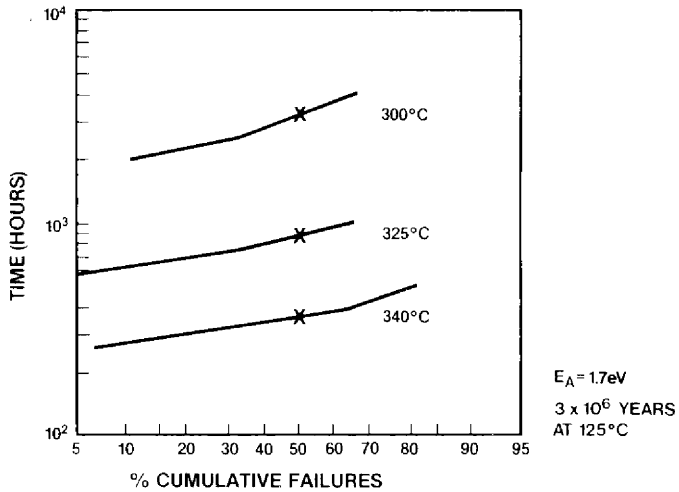


Figure 15. Log cumulative data loss vs. log time for three storage temperatures on samples of 100 X2210's. Data loss is defined to occur when the first bit in an array loses data.

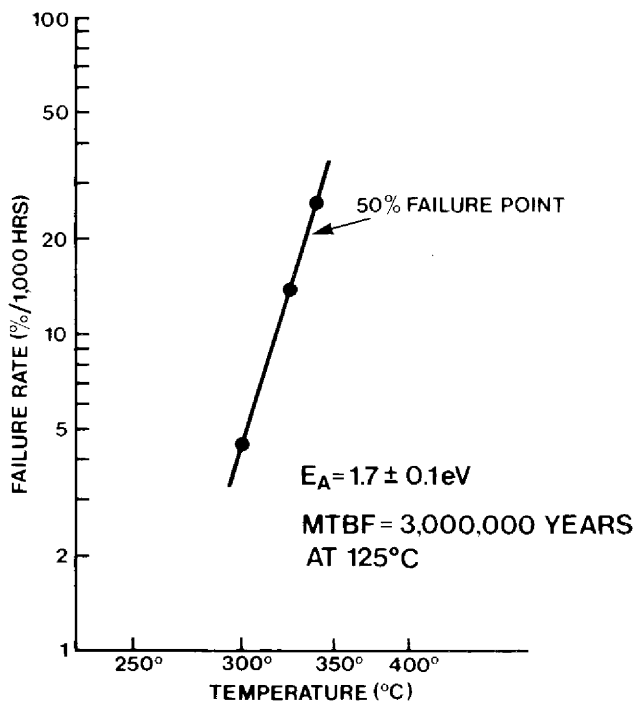


Figure 16. Log data loss rate vs. inverse temperature for X2210's.

Figure 16 shows the result of calculating failure rates based on these results and plotting vs. inverse temperature. Since the rates fall on a straight line, we can extract an activation energy and extrapolate to lower temperatures (see the next section for a discussion of activation energies). The result is that the experimental value of the activation energy is 1.7 eV and the mean time for data loss for this mechanism (which we believe to be the fundamental loss mechanism of this technology) is 3 million years for retention at 125°C.

BASIC RELIABILITY CONCEPTS

There are a couple of simple concepts basic to most reliability work. One is the long established observation that failure rates follow the bathtub-shaped curve illustrated in Figure 17.

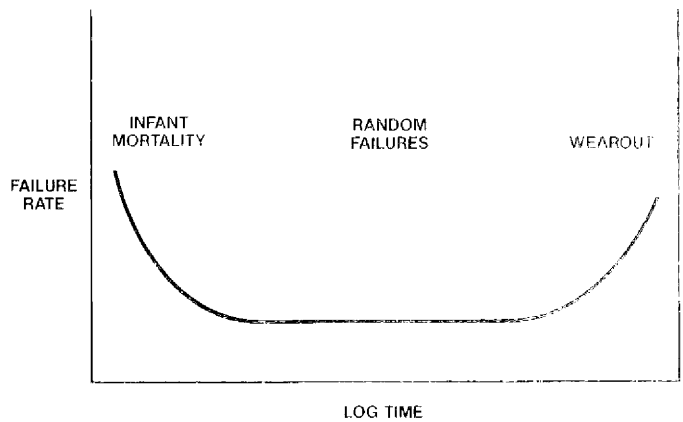


Figure 17. Illustration of bathtub curve of failure rates showing regions in which infant mortality, random failures, and wearout mechanisms dominate the failure rate.

There is an infant mortality region characterized by a rapidly declining failure rate as the "weak" parts are eliminated from the population, a random failure characterized by an invariant or slowly declining failure rate, and a wearout region characterized by an increasing failure rate as the units reach the end of life.

Each region of the failure rate curve has certain specific failure modes which predominate. For example, the infant mortality region is dominated by failures which arise out of manufacturing defects. Table I gives a summary of the common failure mechanisms and stresses which may be used to accelerate the failure rates of the various mechanisms which have been culled from the literature.[11-14]

The classic parameter which is used to accelerate failure rates is temperature. It is known that a very broad class of failure mechanisms have a temperature dependence proportional to $\exp(-E_a/kT)$ where E_a is called the activation energy, k is Boltzmann's constant, and T is the absolute temperature. This is true because a number of basic physical phenomena such as diffusion rates and chemical reaction rates have this dependence. The significance of this is that if the activation energy is known for the failure mechanisms in question, then the failure rates arising from these mechanisms can be measured at elevated temperature where they are high enough to

be conveniently measured and extrapolated back to lower operating temperatures where the failure rates may be so low as to require an inconveniently large number of device hours to measure. The relationship which allows one to translate failure rates from one temperature to another is known as the Arrhenius relation. Figure 18 illustrates this relation for a number of common values of activation energy.

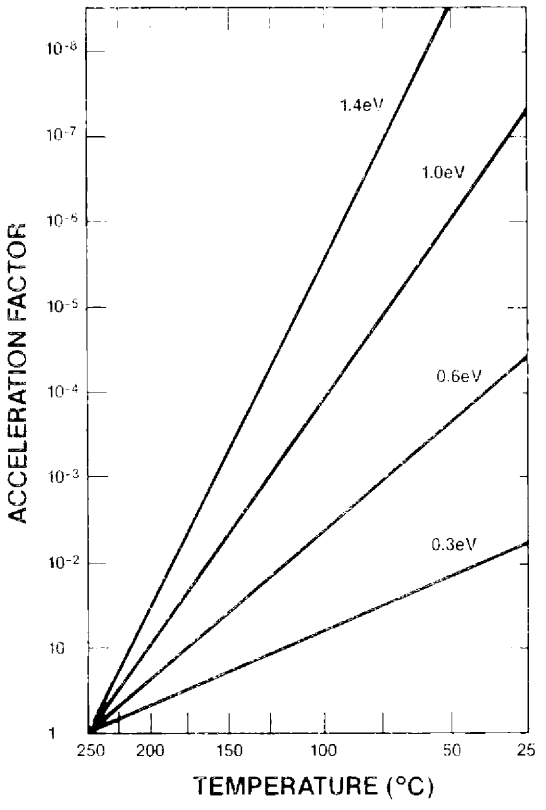


Figure 18. Acceleration factor vs. temperature calculated for various activation energies from Arrhenius relation.

RELIABILITY TESTING

Four types of tests were conducted to establish the reliability of the parts:

1. High temperature dynamic lifetest
2. Data retention bake
3. High temperature reverse bias
4. Environmental

These tests will be discussed in turn.

Failure Mechanism	Type	Activation Energy	Detection Method
Ionic Contamination	Infant/ Random/ Wearout	1.0 eV	High Temp. Bias
Surface Charge	Wearout	0.5-1.0 eV	High Temp. Bias
Polarization	Wearout	1.0 eV	High Temp. Bias
Electro-migration	Wearout	1.0 eV	High Temp. Operating Life
Microcracks	Random	---	Temperature Cycling
Oxide Rupture	Infant/ Random	0.3 eV	High Temp. Operating Life
Silicon Defects	Infant/ Random	0.3 eV	High Temp. Bias
Oxide Defect Leakage	Infant/ Random	0.6 eV	High Temp. Operating Life
Electron Trapping In Oxide	Wearout	---	Low Temp. High Voltage Operating Life

Table I. MOS Failure Mechanisms

High Temperature Dynamic Lifetest

This is the usual data from which failure rate predictions are made. For this to be a valid predictor of failure rate, the parts must really function in the manner in which they would in operation. Thus overly elevated temperatures at which the unit does not function internally are to be avoided, since this may lead to overly optimistic predictions.

Xicor gathers this data at 125° C ambient which is within the known operating range of the units under test. The stimulus pattern consists of recalling a known pattern, writing a checkboard pattern over the recalled pattern bit by bit, reading the checkboard pattern, writing its complement, reading the complement and then beginning over again.

The units in lifetest are tested at 168, 500, 1000, and 2000 hours to determine that they are still within specification. The first step of the readout tests is to recall the information stored in the nonvolatile section of memory to determine if the previously stored pattern is still retained. If so, the memory is completely exercised over voltage including verification of the STORE function. Finally, the predetermined pattern is restored to memory for the next lifetest period.

Data Retention Bake

This test is sometimes referred to as a storage bake, but we prefer the term "data retention bake" because this better describes the principal function it serves in the case of electrically programmable nonvolatile memories.

In this test, a pattern is stored in the memories and the memory is baked at 250° C with no bias applied. At intervals the memory is removed from bake, and the nonvolatile data recalled and checked for accuracy. The data is not restored at readouts in order to ascertain the worst case retention. Since Xicor warrants cycling endurance of various values, this test was performed on parts specified for 1000 cycle endurance which had performed 1000 complete data alterations, as well as on parts which had not received this treatment.

HTRB

HTRB stands for High Temperature Reverse Bias, a term which originated with bipolar circuits, in which case the test reverse biased the junctions of all of the input stages. For MOS circuits, a better term would be high temperature static bias. In this stress, which Xicor carries out at 150° C, Vss is grounded and a static positive voltage is applied to all of the input and outputs as well as to Vcc. This has the effect of applying a static bias equal to the power supply across the gate oxides of the circuit transistors. This stress is intended to expose failures which might occur as a result of drift of mobile ionic contaminants or latent defects in the gate oxides.

It is known that many defects are accelerated by voltage as well as temperature. For example, it has been shown by Crook that the failure rate of oxide defects increases 10^7 times per MV/cm increase in the electric field.[15] For this reason, HTRB stresses were conducted with both 5.5V and 7.5V bias applied to the units under test.

Package Environmental Tests

The package environmental tests were done in accordance with MIL STD 38510 and MIL STD 883/B Group C and D. Table II lists the tests, the test conditions and the acceptance criteria. The results column represents the summary of data from 3 lots. All lots passed the 883/B qualification criteria.

Test	Method	Conditions	LTPD	Accept #	Results
Temp. Cycling	1005	Test Condition C (10 cycle —65 to 125° C)	15	1/25	0/80
Constant Acceleration	2001	Test Condition E (30,000 g Y1 axis only)			0/80
Seal	1014	Test Condition B			
—Fine		(5×10^{-8} cc/min)			2/80
—Gross		Test Condition C			0/78
Lead Integrity	2004	Test Condition B2 (Lead Fatigue)	15	1/25	0/78
Seal	1014				
—Fine		Test Condition B			0/78
—Gross		Test Condition C			1/78
Thermal Shock	1011	Test Condition B (15 cycles —55°C to 125°C)	15	1/25	0/76
Temp. Cycling	1010	Test Condition C (100 cycles)			0/76
Moisture Resistance	1004				0/76
Seal	1014				
—Fine		Test Condition B			1/76
—Gross		Test Condition C			0/75
Internal Water Vapor Content	1018	5,000 ppm max. water content at 100° C	---	0/3 or 1/5	0/17
Mechanical Shock	2002	Test Condition B (1500 g peak 3 axis)	15	1/25	0/75
Vibration Variable Frequency	2007	Test Condition A (20 g peak 3 axis)		1/25	0/75
Constant Acceleration	2001	Test Condition E		1/25	0/75
Seal	1014				
—Fine		Test Condition B		1/25	2/75
—Gross		Test Condition C		1/25	0/73
Salt Atmosphere	1009	Test Condition A	15	1/25	0/73
Seal	1014				
—Fine		Test Condition B			0/73
—Gross		Test Condition C			0/73
Adhesion of Lead Finish	2025		15	1/25 (# of leads from 3 devices)	0/25
Lid Torque	2024		15	1/25	0/25

Table II. ENVIRONMENTAL TEST FOR 18 LEAD CERDIP PACKAGE MIL—STD—883B. Method 5005.7 Group D.

RESULTS

Table III exhibits the results of dynamic life test on 2300 units of X2210 and X2212. The data show a total of 4 failures in over 1.7×10^6 device hours. The causes of the failures were determined through failure analysis and are also listed in Table III.

Tables IV and V show the results of 150°C static life test at 5.5V and 7.5V, respectively. No failures were seen out of 280 units tested with 5.5V applied bias, whereas one unit failed out of the 175 units tested with 7.5V applied bias.

Tables VI and VII show the results of 250°C retention bakes before and after putting 1000 program-erase cycles on units rated as being for use in 1000-cycle applications. The data on uncycled material shows the loss of 4 bits from 350 units baked for 1000 hours. The data on cycled material shows the loss of 4 bits from 175 units baked. These results are not statistically significantly different in overall failure rate, but examination of the form of the data shows a tendency for the failures in the cycled samples to be concentrated at early readouts, as might be expected for latent defects which had been accelerated by the high fields present during the store process. The data loss fraction through 1000 hours at 250°C (which is equivalent to 1.08×10^6 hours at 70°C for the 0.6 eV failures observed) is seen to be in the range of 1-2% which is about half that reported previously for EPROM's.[16] Results reported for other E^2PROM technologies would indicate that the present results are greatly superior to retention observed on the other technologies.[17]

	48 Hr.		168 Hr.		500 Hr.		1000 Hr.		2000 Hr.		
	#Fail	#In	#Fail	#In	#Fail	#In	#Fail	#In	#Fail	#In	Total Hours
Lot #1 (2212)	0	105	0	105	0	105	0	105	0	105	$2.0 \cdot 10^6$
Lot #2 (2212)	0	25	0	24	0	24	0	24			22848
Lot #3 (2212)	0	25	0	25	0	25	0	25			23800
Lot #4 (2212)	0	25	0	21	0	16	0	16			15832
Lot #5 (2212)	0	25	0	25	0	25	0	25			23800
Lot #6 (2212)	0	25	0	25	0	25	0	25			23800
Lot #7 (2210)	0	25	0	25	0	25	0	25			23800
Lot #8 (2210)	0	25	0	24	0	24	0	24			22848
TOTALS	0	280	0	274	0	269	0	269	0	105	$3.7 \cdot 10^6$

Table IV. STRESS 5.5V HTRB AT 150°C

	48 Hr.		168 Hr.		500 Hr.		1000 Hr.		
	#Fail	#In	#Fail	#In	#Fail	#In	#Fail	#In	Total Hours
Lot #2 (2212)	0	25	0	23	0	23	0	23	21896
Lot #3 (2212)	1[a]	25	0	23	0	23	0	23	21896
Lot #4 (2212)	0	25	0	23	0	23	0	23	21896
Lot #5 (2212)	0	25	0	25	0	25	0	25	23800
Lot #6 (2212)	0	25	0	25	0	25	0	25	23800
Lot #7 (2212)	0	25	0	24	0	24	0	24	22848
Lot #8 (2210)	0	25	0	25	0	24	0	24	22968
TOTALS	1	150	0	144	0	143	0	143	1.3×10^5

a = single bit retention failure, leaky oxide -0.6 eV

Table V. STRESS 7.5V HTRB AT 150°C

	168 Hr.		500 Hr.		1000 Hr.		2000 Hr.		
	#Fail	#In	#Fail	#In	#Fail	#In	#Fail	#In	Total Hours
Lot #1 (2212)	2[a]	353	0	93	0	93	0	93	2.1×10^5
Lot #2 (2212)	0	385	0	98	0	98	0	98	2.3×10^5
Lot #3 (2212)	0	99	0	99	0	99	0	99	1.9×10^5
Lot #4 (2212)	0	279	0	99	0	99	0	99	2.1×10^5
Lot #5 (2212)	0	322	0	99	0	97	0	97	2.2×10^5
Lot #6 (2212)	0	283	0	99	0	99	0	99	2.2×10^5
Lot #7 (2210)	1[b]	287	1[c]	99	0	98	0	98	2.1×10^5
Lot #8 (2210)	0	292	0	99	0	99	0	99	2.2×10^5
TOTALS	3	2300	1	785	0	782	0	782	1.7×10^6

a = 2 units — single bit retention failure, oxide leakage, 0.6 eV
b = 1 unit — single bit oxide breakdown, 0.3 eV
c = 1 unit — single bit oxide breakdown, 0.3 eV

Table III. STRESS DYNAMIC LIFE TEST

	48 Hr.		168 Hr.		500 Hr.		1000 Hr.		2000 Hr.	
	#Fail	#In	#Fail	#In	#Fail	#In	#Fail	#In	#Fail	#In
Lot #2 (2212)	0	51	0	51	0	51	0	51	0	51
Lot #3 (2212)	0	49	0	49	0	49	2[b]	49	0	47
Lot #4 (2212)	0	50	1[a]	50	0	49	0	49		
Lot #5 (2212)	0	50	0	50	0	50	1[c]	50		
Lot #6 (2212)	0	50	0	50	0	50	0	50		
Lot #7 (2212)	0	50	0	47	0	47	0	47		
Lot #8 (2210)	0	50	0	50	0	50	0	50		
TOTALS	0	350	1	347	0	346	3	346	0	98

a = 1 failure — single bit oxide leakage, 0.6 eV
b = 2 failures — single bit oxide leakage, 0.6 eV
c = single bit oxide leakage, 0.6 eV

Table VI. STRESS 250°C BAKE "NO CYCLE"

	48 Hr.		168 Hr.		500 Hr.		1000 Hr.	
	#Fail	#In	#Fail	#In	#Fail	#In	#Fail	#In
Lot #2 (2212)	1[a]	25	1[b]	24	0	23	0	23
Lot #3 (2212)	0	25	0	25	0	25	0	25
Lot #4 (2212)	2[c]	25	0	23	0	23	0	23
Lot #5 (2212)	0	25	0	25	0	25	0	25
Lot #6 (2212)	0	25	0	25	0	25	0	25
Lot #7 (2210)	0	25	0	25	0	25	0	25
Lot #8 (2210)	0	25	0	25	0	25	0	25
TOTALS	3	175	1	172	0	171	0	171

a = single bit failure oxide leakage, 0.6 eV
b = single bit failure oxide leakage, 0.6 eV
c = 2 single bit failures oxide leakage, 0.6 eV

Table VII. STRESS 1K CYCLES BAKE AT 250°C

CALCULATION OF PREDICTED FAILURE RATE

There is no simple, one-step formula for inferring a predicted failure rate from the experimental data. Instead, the failures of each individual activation energy must be treated differently. The first step is to calculate the equivalent device hours at the ambient temperature of interest, utilizing the Arrhenius relationship discussed earlier. This calculation should be carried out for every mechanism observed or expected. Thus, for example, the calculation for the 0.3 eV activation energy oxide rupture mechanism should be carried out whether this failure mechnaims is observed or not, since this mechanism is always anticipated in MOS integrated circuits. The extrapolation should be carried out utilizing the junction temperature at the ambient temperature of interest and not the ambient temperature itself. The upper confidence limit is then calculated for the failure rate for each activation energy. The upper confidence limits for the various activation energies are then summed for a total failure rate prediction. (The meaning of the "upper confidence level" is that with a certainty, or probability, of a certain level we can say that the true value is less than the stated value. Thus, the confidence level rate calculated is non-zero even for the case where no failures are observed, because we can't be sure that there will be none.)

The results are tabulated in Table VIII based on the data tabulated in Tables III-V. As is seen, these data lead to a predicted failure rate of 0.045%/1000 hours at 70°C ambient with a 60% UCL and 0.028%/1000 hours at 55°C ambient with a 60% UCL. For comparison with other vendors who may use expected values, the expected values of the failure rates are calculated to be .030%/1000 hours (300 FIT) at 70°C and 0.019%/1000 hours (190 FIT) at 55°C.

In reducing these data, we made the extremely conservative assumption of applying no acceleration for the elevated voltage of the 7.5 V HTRB data. Had a voltage acceleration been applied, the predicted failure rates would have been lowered by another two orders of magnitude. Although this technique of data enhancement has been employed by some companies, the more conservative estimate has been used here. These results show that Xicor has attained on its NOVRAM products failure rates comparable to those reported by major suppliers on standard volatile products.[18-20]

Activation Energy (eV)	Hours at 125°C	Hours at 150°C	Number of Failures	Equivalent Hours at 70°C	Expected Value of Failure Rate at 70°C (F.I.T.)	60% UCL Failure Rate (F.I.T.)	Equivalent Hours at 55°C	Expected Value of Failure Rate at 55°C (F.I.T.)	60% UCL Failure Rate (F.I.T.)
0.3	$1.71 \cdot 10^6$	$5.43 \cdot 10^5$	2	$8.99 \cdot 10^8$	220	340	$1.35 \cdot 10^7$	150	230
0.6	$1.71 \cdot 10^6$	$5.43 \cdot 10^5$	3	$3.73 \cdot 10^7$	80	110	$8.42 \cdot 10^7$	40	50
1.0	$1.71 \cdot 10^6$	$5.43 \cdot 10^5$	0	$2.69 \cdot 10^8$	—	—	$1.05 \cdot 10^9$	—	—
TOTAL					300	450		190	280

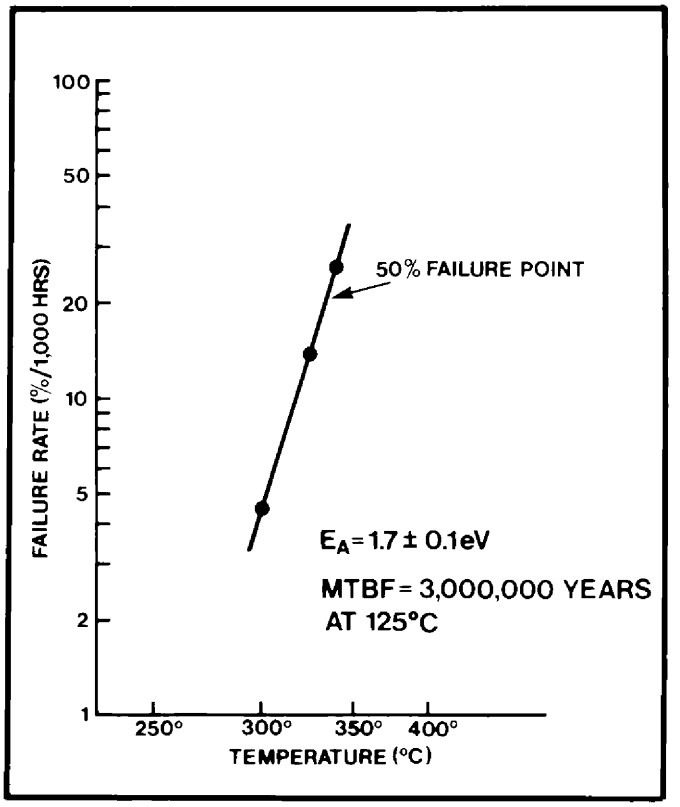
Table VIII. RESULTS FOR TABLES III-V

SUMMARY

The data presented in this reliability report show that the data retention of Xicor's NOVRAM technology is excellent. Even at as high a temperature as 300°C, only about 2% lose data in 1000 hours. Theoretical grounds for expecting this result are discussed. The predicted expected failure rate at 55°C is 190 FIT (.019%/1000 hours) for a value of 280 FIT (.028%/1000 hours) at a 60% UCL. Finally, the cerdip packaging employed is shown to be capable of passing the Group D qualification requirements of MIL-STD 883B, Method 5005.7.

REFERENCES

1. R. H. Fowler and L. Nordheim, Proceedings of the Royal Society of London, *A119*, 173-81 (1928).
2. M. Lenzlinger and E. H. Snow, Journal of Applied Physics, *40*, 278-83 (1969).
3. D. J. Di Maria and D. R. Kerr, Applied Physics Letters, *27*, 505-7 (1975).
4. R. M. Anderson and D. R. Kerr, Journal of Applied Physics, *48*, 4834-6 (1977).
5. H. R. Huff et al, Journal of the Electrochemical Society, *127*, 2483-8 (1980).
6. T. J. Lewis, Journal of Applied Physics, *26*, 1405-10 (1955).
7. C. Hu et al, Applied Phusics Letters, *35*, 189-91 (1979).
8. R. K. Ellis, IEEE Electron Device Letters, *EDL-13*, 330-3 (1982).
9. R. K. Ellis, H. A. R. Wegener, & J. M. Caywood, International Electron Devices Meeting Technical Digest, 749-52 (1982).
10. Z. Weinberg, Solid State Electronics, *20*, 11-18 (1977).
11. G. L. Schnable and R. S. Keen, Jr., IEEE Trans. on Electron Devices, *ED16*, 322-32 (1969).
12. S. R. Hofstein, Solid State Electronics, *10*, 657 (1967).
13 J. R. Black, Proc. 6th Reliability Physics Symposium, 148-53 (1967).
14. R. E. Shiner et al, Proc. 18th Reliability Physics Symposium, 238-43 (1980).
15. Dwight L. Crook, 17th Annual Reliability Physics Symposium, 1-7 (1979).
16. B. Euzent et al, Proc. 19th Annual Reliability Physics Symposium, 11-16 (1981).
17. R. E. Shiner et al, Proc. 21st Annual Reliability Physics Symposium, 248-56 (1983).
18. Bruce Euzent, Intel Reliability Report RR-14, "2115/2125 N-Channel Silicon Gate MOS 1K Static RAMs" (1976).
19. Chieh Lin Ping, National Semiconductor "Reliability of N-Channel Metal Gate MOS/LSI Microcircuits" (1982).
20. Bruce Euzent & Stuart Rosenberg, Intel Reliability Report RR-18, "HMOS Reliability" (1978).



X2816A/04A PRELIMINARY RELIABILITY SUMMARY

By Bruce Prickett
& Dr. John Caywood

Cover Photo: Figure 7 Data retention vs. temperature for memory arrays fabricated with Xicor's technology.

This report is based on data collected through September, 1983.

Table of Contents

X2816A/04A PRELIMINARY RELIABILITY SUMMARY

by Bruce Prickett &
John Caywood

Introduction

The X2804A and X2816A are 4K and 16K electrically erasable programmable read only memories (E^2PROMs) organized as 512 x 8 and 2K x 8. These memories operate on a single 5 volt power supply for all operations. Figures 1 and 2 show the pin configurations of the X2816A and X2804A and the functional diagram of the X2816A. Figure 3 is the package outline drawing for these parts. Figure 4 and 5 are bit maps which illustrate the physical location of the memory bits of the X2804A and X2816A respectively.

Although Xicor introduced its X2816A and X2804A E^2PROMs only recently, Xicor is an experienced manufacturer of E^2PROMs. This is because the family of NOVRAMs (X2201A,X2210 and X2212) which Xicor introduced earlier all contained E^2PROMs integrated together with static RAM on a bit-for-bit basis. The process technology and the 5 volt design techniques which had been developed for the NOVRAMs were carried over to the E^2PROM products. This experience enabled Xicor to introduce these 5-Volt only E^2 PROMs with a self timed write cycle that includes latched address and data buffers. By producing parts with these features in production quantities, Xicor has set the industry standard for E^2 PROMs.

Technology

These E^2PROMs employ the same triple poly n-channel process which Xicor developed previously for NOVRAMs. With this technology, data is stored as the presence or absence of charge on a piece of second-level polysilicon which acts as a gate for a readout transistor. This piece of polysilicon is completely surrounded by ~ 1000 Å of thermally grown SiO_2, one of the best electrical insulators known. Charge is transferred onto and off of the storage gate by means of a quantum mechanical phenomenon, called Fowler-Nordheim tunneling. This phenomenon has been described in detail in recent publications.[1-3]

The operation of the cell can be explained in terms of the schematic diagram shown in Figure 6. First consider the left-hand read section. When the word select line (which is fabricated in third level polysilicon) is de-selected (i.e., low), the READ transistor is off. When the word select line is selected (i.e., high), conduction through the READ transistor depends on the charge on the floating storage gate. If the floating gate is charged negatively, the channel beneath it is in accumulation and no conduction occurs. This is read as a "0". If the floating gate is charged positively, the channel beneath it is in depletion and the channel conducts. This is read as a "1". The floating gate is programmed (i.e., charged negatively) by taking the Word Select line and source lines high while the poly 1 WRITE BIT line is held low. The floating gate is erased (i.e., charged positively) by taking the word select line high while holding both the source and WRITE BIT lines low.

As was explained in an earlier publication, tunnel current emitted from a textured surface inherently decreases more rapidly with decreasing voltage than does emission through thin oxide grown on a planar surface.[3] This results in lower leakage currents from the floating gate during storage and during the read operation of the part. The excellent data retention which users may expect from Xicor E^2PROMs is shown in Figure 7 where the log of the measured data loss rate is plotted vs. inverse temperature. The slope of a straight line passing through these data gives the thermal activation energy for intrinsic data loss in Xicor's technology: 1.7 eV. Extrapolation of this line to lower temperatures suggests that the mean time before data loss should be 3,000,000 years at 125°C. Such extrapolations cannot be taken literally but clearly show that data will be retained as long as anyone cares for temperatures within the specified storage conditions.

Xicor's textured poly technology also yields E^2PROMs whose ability to be re-programmed (usually called "endurance") is good. Figure 8 shows the current conducted by a typical cell as a function of the number of times that cell had been programmed to a "0" or erased to a "1". As can be seen, the current conducted in the erased state begins declining noticeably at greater than a million cycles of data change, but after ten million cycles, the cell current in the erased state is still about 70 μA while the current in the programmed state is zero. These values provide good margin to the 50 μA value which is the nominal value at which the sense amplifier circuit distinguishes between "1" and "0".

Reliability Study and Results

As was mentioned above, Xicor's E^2PROMs use the same process technology and design techniques as the Xicor family of NOVRAMs. Additionally, these products are fabricated in the same wafer fab facility and assembled in the same assembly area with the same assembly technology and controls as the NOVRAMs. For these reasons, the E^2PROMs should be expected to exhibit the same excellent long term reliability already demonstrated by Xicor's NOVRAMs. The reliability studies reported here were designed to corroborate this assumption.

One further note is that the X2804A is a smaller version of the X2816A, created by designing a mask set which is just like the X2816A except that 3/4 of the array and two address buffers are removed (i.e., the X2804A is not a fallout die from the X2816A production). Thus, the reliability study focused on the X2816A as the more sensitive reliability indicator since it has four times as many memory bits and about twice the active silicon area.

Since the study illustrated in Figure 7 showed a strong acceleration in data loss at elevated temperatures, data retention was measured at 250°C. (For an activation energy of 1.7 eV, data loss at 250°C is accelerated 1.4×10^5 times with respect to 125°C.) Table 1 shows the results of data retention testing on two lots of material which was not intentionally cycled and on two lots after 10,000 program/erase cycles. Although the sample is small, in neither case was any data loss observed, which corroborates the results of Figure 7.

Two other types of stress tests which were conducted on the X2816A were high temperature reverse bias (HTRB) at 150°C and dynamic life test at

results for the HTRB and dynamic life tests, respectively. As these results show, no failures were observed in these tests for HTRB, and three units failed in 890,000 device hours of 125°C dynamic life test.

The failure rate of the 2816 was estimated based on the data in Tables II and III. The method used is that discussed in some detail in RR502. The results, shown in Table IV, predict an expected failure rate of 15 F.I.T. (.0015%/1000 hr.) at 70°C and 3 F.I.T. (.00032%/1000 hr.) at 55°C. The extrapolated value for the 60% UCL of the failure rates are 260 F.I.T. (.026%/1000 hr.) and 140 F.I.T. (.014%/1000 hr) at 70°C and 55°C ambient, respectively. The large difference between expected failure rate and 60% confidence level for the failure rate are an indication of the enormous number of device hours necessary to confirm the low failure rates observed.

Summary

The technology used in producing the Xicor X2816A and X2804A are reviewed. The reasons for expecting excellent data retention and good endurance are discussed. Finally, data are shown which confirm that these E^2PROMs share the excellent long-term reliability already demonstrated by Xicor's NOVRAM family.

References

1. R.K. Ellis, IEEE Electron Device Letters EDL-13, 330-3 (1982).

2. R.K. Ellis, H.A.R. Wegener, & J.M. Caywood, International Electron Devices Meeting Technical Digest, 749-52 (1982).

3. Billy Kwong & John Caywood, NOVRAM Reliability Report, XICOR pub. RR502 (1983).

Table I

250°C RETENTION BAKE

	48 HOUR		168 HOUR		500 HOUR		1000 HOUR		2000 HOUR		TOTAL HOURS
	#FAIL	#IN	#FAIL	#IN	#FAIL	#IN	#FAIL	#IN	#FAIL	#IN	
Lot #1	0	50	0	50	0	50	0	50	0	50	97600
Lot #2 (Cycled)	0	24	0	24	0	24	0	24	0	24	46848
Lot #3	0	20	0	20	0	20	0	20	0	20	39040
Lot #4 (Cycled)	0	55	0	55	0	55	0	55	0	55	1.1×10^5
TOTALS	0	149	0	149	0	149	0	149	0	149	2.9×10^5

Table II

STRESS 5.5V HTRB AT 150°C

	48 HOUR		168 HOUR		500 HOUR		1000 HOUR		TOTAL HOURS
	#FAIL	#IN	#FAIL	#IN	#FAIL	#IN	#FAIL	#IN	
Lot #1	0	25	0	25	0	25	0	25	23800
Lot #2	0	25	0	25	0	25	0	25	23800
Lot #3	0	105	0	105	0	105	0	104	99460
Lot #4	0	25	0	25	0	25	0	25	23800
TOTALS	0	180	0	180	0	180	0	179	1.7×10^5

Table III

DYNAMIC LIFE TEST AT 125°C

	168 HOUR		500 HOUR		1000 HOUR		2000 HOUR		TOTAL HOURS
	#FAIL	#IN	#FAIL	#IN	#FAIL	#IN	#FAIL	#IN	
Lot #1	0	159	0	84	0	84	0	84	1.7×10^5
Lot #2	0	152	0	93	0	93	1[a]	93	1.9×10^5
Lot #3	0	75	0	75	0	75	0	75	1.5×10^5
Lot #4	0	388	0	99	1[c]	99	0	98	1.3×10^5
Lot #5	1[b]	387	0	126	0	126	-	-	1.5×10^5
TOTALS	1	1161	0	477	1	477	1	350	8.9×10^5

a = column failure; ionic contamination;1 eV
b = stuck row; ionic contamination; 1 eV
c = single bit retention failure; non-repeatable ionic contamination; 1 eV

Table IV

Activation Energy (eV)	Hours at 125°C	Hours at 150°C	Number of Failures	Equivalent Hours at 70°C	Expected Value of Failure Rate at 70°C	60% UCL Failure Rate at 70°C	Equivalent Hours at 55°C	Expected Value of Failure Rate at 55°C	60% UCL Failure Rate at 55°C
.3	8.9×10^5	1.7×10^5	0	4.78×10^6	0%/1000	.0195%/1000	7.60×10^6	0%/1000hr	.012/1000hr
.6	8.9×10^5	1.7×10^5	0	2.26×10^7	0%/1000	.0041%/1000	5.73×10^7	0%/1000	.0016/1000hr
1.0	8.9×10^5	1.7×10^5	3	1.97×10^8	.0015%/1000	.0021%/1000	9.29×10^8	.00032%/1000	.00044/1000hr
TOTAL			3		.0015%/1000	.0257%1000hr		.00032%/1000	.014%/1000hr

Figure 1

E^2PROM pin assignments

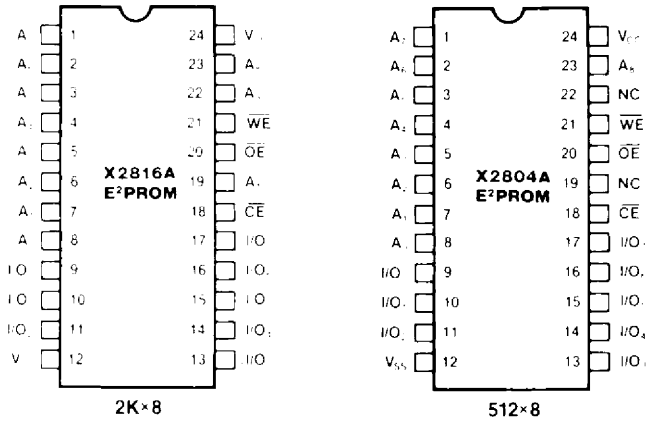


Figure 2

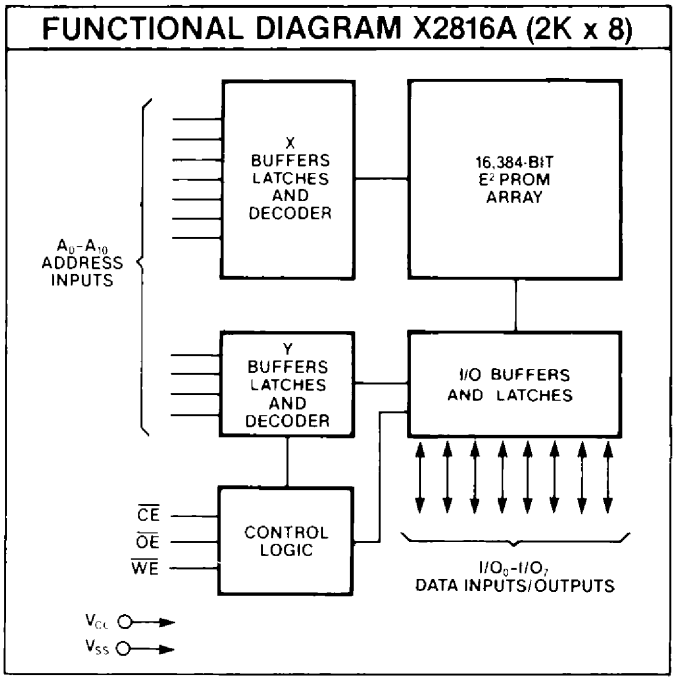
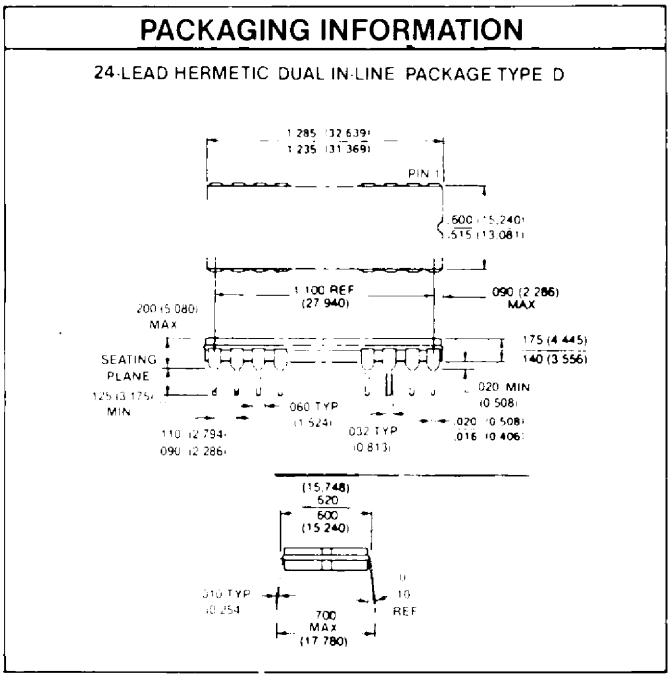


Figure 3



Reliability Reports

Figure 4: X2804A physical bit map. The arrangement of bits within each byte are displayed for column Ø and suppressed for the other columns in which the arrangement of column Ø is duplicated.

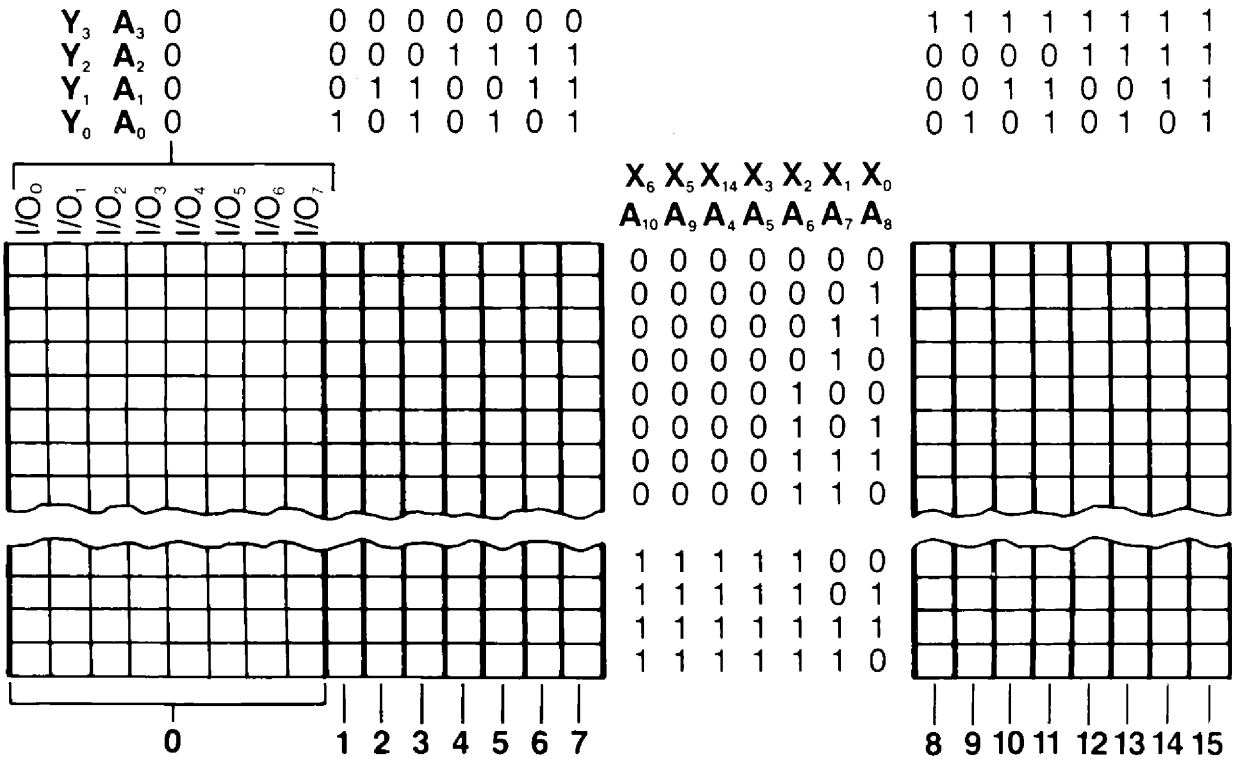


Figure 5: X2816A physical bit map. The arrangement of bits within each byte are displayed for column Ø and suppressed for the other columns.

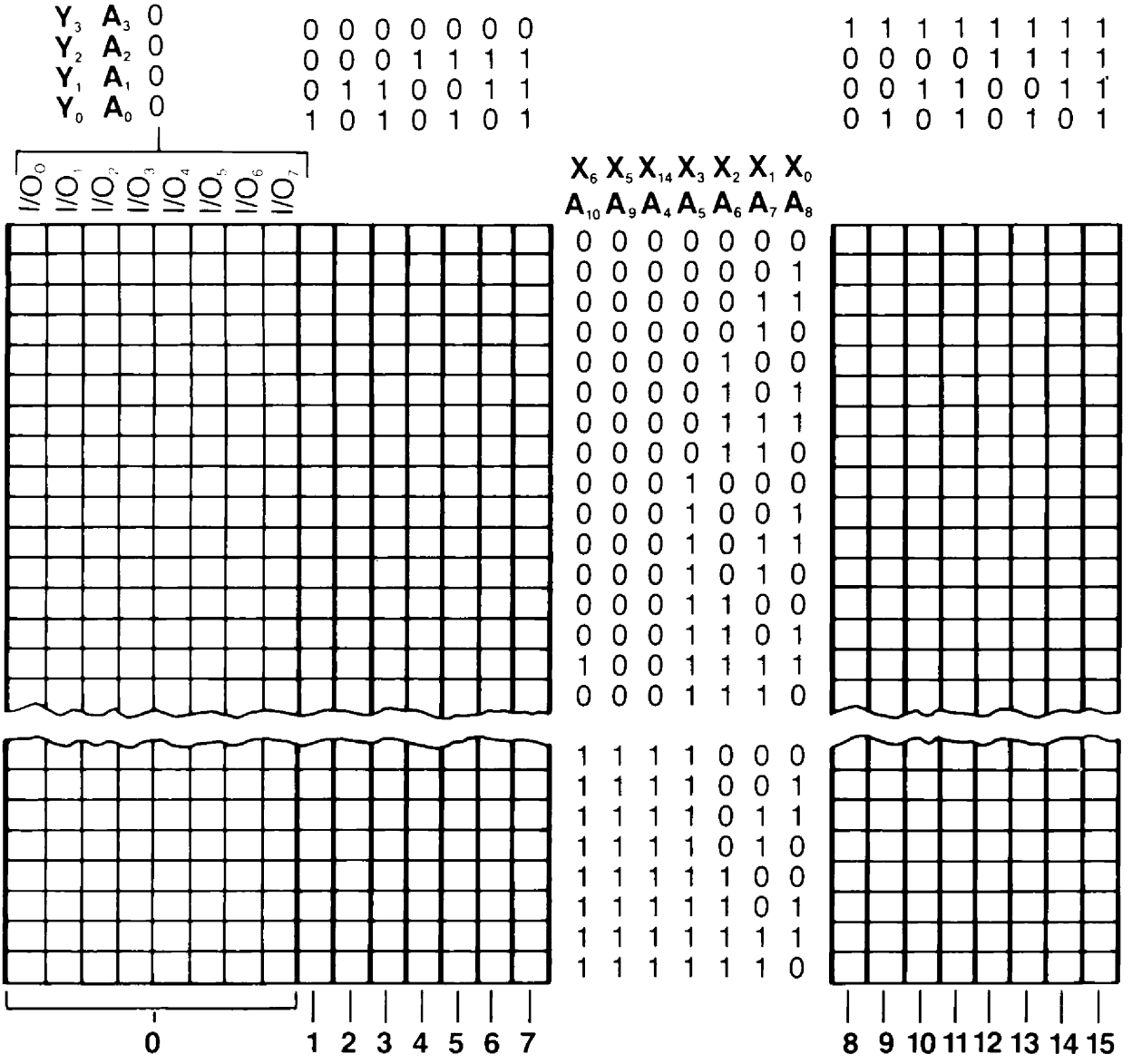


Figure 6: E^2 cell schematic

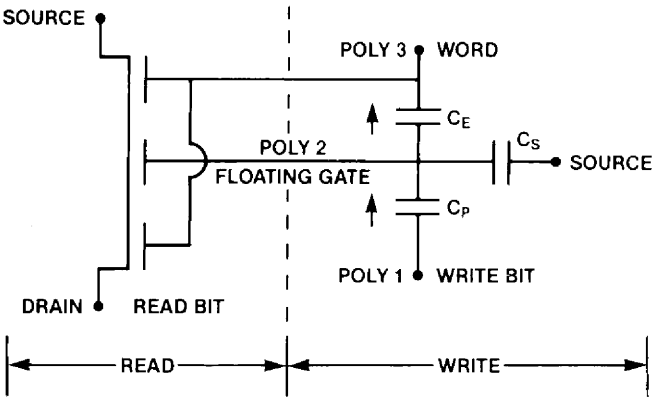


Figure 7: Data retention vs. temperature for memory arrays fabricated with XICOR's technology

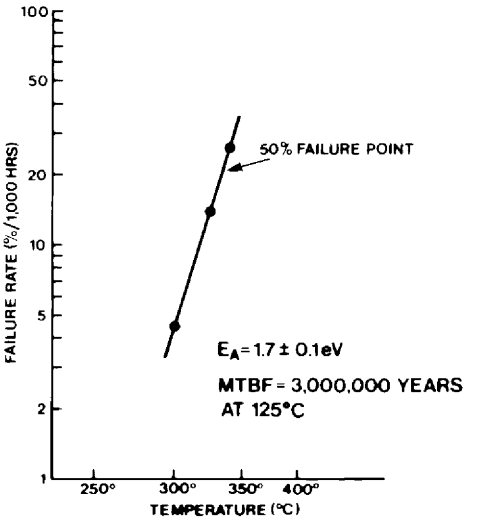


Figure 8: E^2 cell endurance data for a typical cell

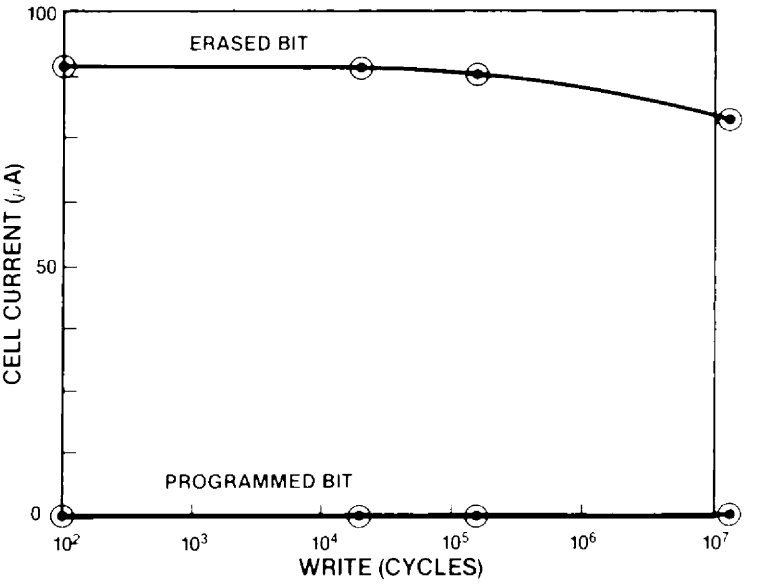
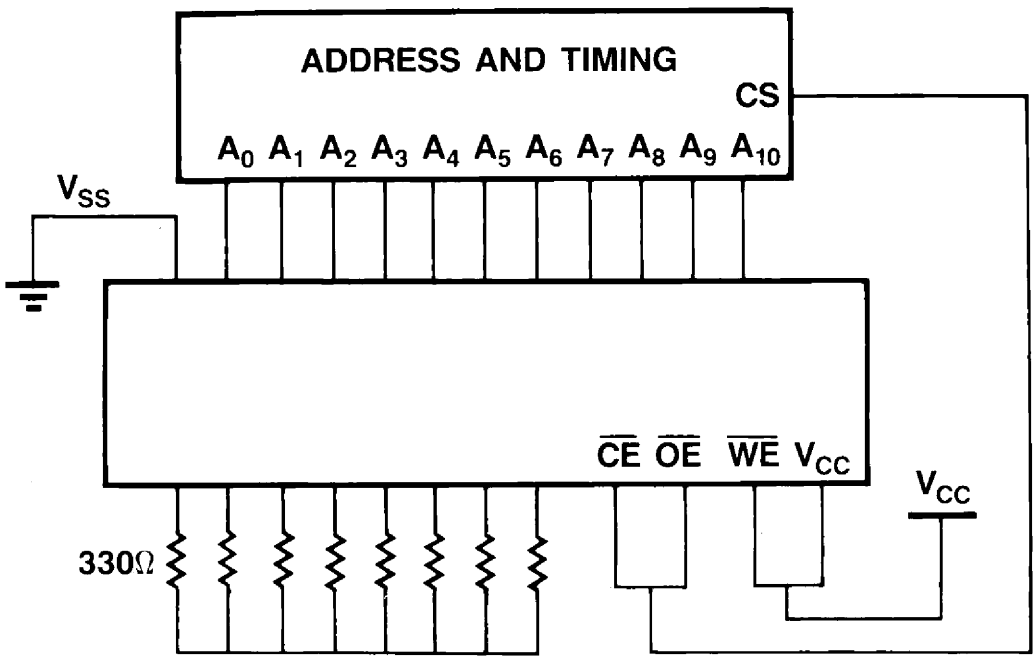


Figure 9: Dynamic life test set-up showing the circuit configuration (a) and the timing (b). The clock frequencies of higher order address signals are divided down in binary sequence from A_0.

a.



b.

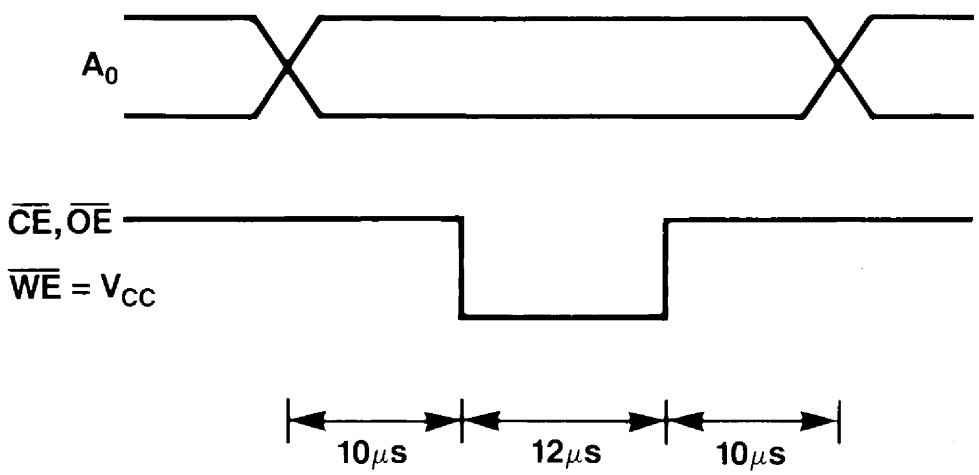


Figure 10: Static life test bias diagram