DP8400/8419 Error Correcting Dynamic RAM Memory System for the Series 32000®

National Semiconductor
Application Note 387
Webster (Rusty) Meier
December 1985

INTRODUCTION

Three PAL's® (Programmable Array Logic devices) were used in this application in order to interface between the NS32016, DP8419 and the DP8400 to produce an error correcting memory system for the Series 32000 microprocessor family. The PAL Interface Controller (hereafter referred to as P.I.C.) takes care of all interfacing logic, no extra control logic is needed.

FEATURES

- The P.I.C. controls the following types of cycles:
  - A) READ cycles with no errors detected, ALWAYS CORRECT MODE (1 WAIT state inserted).
  - B) READ cycles with single error detected, the correct data will be written back to memory and given to the CPU. One WAIT state is inserted into the READ cycle and one WAIT state is inserted into the next access cycle (and the access is delayed) if it immediately follows the READ cycle.
  - C) READ cycles with more then one error detected. In this case the processor is interrupted and appropriate action can be taken.
  - D) WRITE cycles (no WAIT states).
  - E) BYTE WRITE cycles, or READ MODIFY WRITE cycles (3 WAIT states inserted). If more then one error is detected in the READ portion of this cycle the processor will be interrupted so appropriate action can be taken.
  - F) DRAM REFRESH cycles (may cause a maximum of 5 WAIT states to be inserted into an access cycle if the access occurs while the refresh is taking place).
- All single bit errors are automatically corrected and rewritten back to memory.
- All double bit errors are detected and cause a system interrupt.
- Can directly drive up to 2M bytes of Dynamic RAM (4 banks of 22 256k DRAMS, each bank being 16 data bits plus 6 check bits).
- The P.I.C. allows full use of the DP8400 and all its modes of operation, including:
  - A) The DIAGNOSTIC modes (can do a diagnostic test of the DP8400 without needing to use external memory).
  - B) The COMPLEMENT modes (useful for doing the DOUBLE COMPLEMENT METHOD to try to correct 2 errors).
- The P.I.C. interfaces between the DP8409A or DP8419 Dynamic RAM controller, the DP8400 Expandable Error Checker and Corrector, the NS32016 processor, the NS32201 Timing Control Unit, and the NS32082 Memory Management Unit (if used in the system).
- Provides outputs to interrupt the CPU and to insert WAIT states if needed.
- This interface uses PAL's whose equations and timing are given, allowing the user to customize the interface to his own requirements (even a different processor family) if he so desires.
- Can work at 10 MHz (using the new DP8419, DP8400-2, and common 120 ns 64k DRAMs). Operation at higher frequencies is possible.

DESCRIPTION

The P.I.C. consists of 3 PAL's and one 74LS164 parallel output serial shift register (see P.I.C. logic diagram). If greater speed is needed for the shift register (CPU clock speed is over 6 MHz) one could use some similar type of shift register in a faster type of logic (''AS, ALS, F''), or could make one out of D flip-flops (74AS174).

If one is using a CPU other then the Series 32000 and does not have a fast clock (FCLK, twice system clock frequency) he could substitute a 5 or 10 tap delay line for the shift register.

The P.I.C. uses a shift register as an aid in determining the state of the CPU and where it is in an access cycle. When either of the two outputs, ''$\overline{\text{RASIN}}$'' or ''$\overline{\text{RFSH}}$'', go true the shift register is enabled and begins producing a series of delays. These delays, along with specific signals from the CPU, are used in the interface to determine the state of the CPU and create the appropriate control signals for the DP8400, the DP8409A/DP8419, and the processor. Other CPUs should be able to customize this interface to their requirements by adjusting the appropriate equations.

The logic in the upper right hand corner of the P.I.C. logic diagram may not be needed (74LS374's, 74LS244, 74LS240's LED's and several SSI gates). The logic allows the latching of the DRAM bank (BA17, BA18), the syndrome (S0–S7), and the error flags (AE, E0, E1) during an error condition. The latched data will be displayed on the LED's (until the I/O RESET signal is applied) and can be read from the data bus by the CPU. The address in error could also be latched by this same logic, if desired.

The 2 input AND gate (U5) in the upper left of the P.I.C. logic diagram holds $\overline{\text{CS}}$ low until after $\overline{\text{RASIN}}$ goes high on the DP8409A/19. This is particularly useful for READ cycles with one ERROR where $\overline{\text{RASIN}}$ is extended beyond the end of the current cycle, perhaps into another access cycle.

In this application double bit errors, in the dynamic RAM, generate an interrupt to the CPU. All single bit errors are automatically corrected and rewritten back to memory.

During a SYSTEM RESET the internal flip-flops of PAL #1 are set to a refresh state by making the RESET input look like a refresh request (External logic was used to ''NOR'' the DP8409A/19 RFI/O input with a system RESET input to produce the PAL #1 RFI/O input).

The P.I.C. performs HIDDEN REFRESHES (CPU not accessing the Dynamic RAM controlled by the DP8409A, indicated by ''/CS'' being high) assuming a 4 ''T'' state processor access cycle.

RRD-B30M105/Printed in U. S. A.

The P.I.C. allows the full use of the DP8400 and all its modes of operation. For example, the DP8400 has excellent diagnostic capabilities included in modes ''2'' and ''6''. These modes allow one to perform a complete diagnostic test of the DP8400 without using the external memory. This is possible using an I/O port to control ''M1 and M0'' of the DP8400, along with the diagnostic control signals ''DIAGCS and DIAGD'' as follows:

1) The user can set the I/O signals ''M1'' and ''DIAGCS'' both high and perform a mode 2 DIAGNOSTIC WRITE to the DP8400 with user generated CHECK bits on the high byte of the data bus. The CHECK bits will be latched into the DP8400 (CSLE held low) until the user sets the I/O signal ''DIAGCS'' low.
2) The user can then set the I/O signals ''M1'' low and ''DIAGD'' high and perform a mode 0 WRITE, latching the user generated data in the DP8400 input latches (DLE held low).
3) Next, the user can perform a normal mode 4 READ. This will in effect be a diagnostic READ of the user generated data and check bits without using the external memory. In this way the DP8400 can be completely checked out during system initialization.
4) The syndromes, check bits, and error flags can also be read, provided $\overline{\text{ODLE}}$, $\overline{\text{OB0}}$, and $\overline{\text{OB1}}$ are low, using mode 6A or by reading the latches.
5) When the diagnostics are completed the user can return the DP8400 to normal functioning by resetting the I/O port outputs to the original DP8400 operating mode values (''M0, M1, DIAGCS, DIAGD'' all low, and ''$\overline{\text{I/O RESET}}$'' high).

Using the I/O port signal ''M0'' the user could perform the DOUBLE COMPLEMENT METHOD to try to correct a DOUBLE bit error in the DRAM (see DP8400 data sheet for further information on the DOUBLE COMPLEMENT METHOD).

Another I/O port output, ''I/O $\overline{\text{RESET}}$'', allows the outputs ''$\overline{\text{DOUBLERROR}}$'' and ''$\overline{\text{ERROR}}$'' in PAL #3 to be reset. The signal ''$\overline{\text{ERRLAT}}$'' is used in this interface to latch the SYNDROME, DRAM bank, and ERROR flags during a CPU READ access with a single, double, or triple bit error. The CPU can READ these latched error signals by performing a memory READ from a specific memory location. (An OFF BOARD CHIP SELECT, ''$\overline{\text{CS–OFFB}}$''.) This READ will gate the latched error condition to the CPU data bus via the 74LS244 buffer and the signal $\overline{\text{SYNDROME-DATA}}$ (see the upper right hand corner of the P.I.C. controller logic diagram).

The PAL equations that follow are in the National Semiconductor PLAN™ format, which differs from the standard PALASM™ format.

EXAMPLE: PLAN FORMAT

''$\overline{\text{RASIN}}$:= RFSH * $\overline{\text{2D}}$ * ODLE''

This translates as, ''$\overline{\text{RASIN}}$'' is low after the rising edge of the input clock given that ''$\overline{\text{RFSH}}$'' was high and ''2D'' was low and ''$\overline{\text{ODLE}}$'' was high a setup time before the clock transitions high (here $\overline{\text{RASIN}}$, $\overline{\text{RFSH}}$, and $\overline{\text{ODLE}}$ are outputs of the PAL and 2D is an input).

EXAMPLE: PALASM FORMAT

''RASIN := $\overline{\text{RFSH}}$ * $\overline{\text{2D}}$ * $\overline{\text{ODLE}}$''

The above expression means the same as the PLAN format expression except it is written in PALASM format. In other words ''$\overline{\text{RASIN}}$'' will go low after the rising edge of the clock given that ''$\overline{\text{RFSH}}$'' was high, ''2D'' was low and ''$\overline{\text{ODLE}}$'' was high a setup time before the clock transitions high (here $\overline{\text{RASIN}}$, $\overline{\text{RFSH}}$, and $\overline{\text{ODLE}}$ are outputs and 2D is an input).

Depending on the Specific type of PAL's and logic used the user can calculate the speed requirements for the DRAM at the specified processor frequency as follows:

Here both ''t_{RAC}'' and ''t_{CAC}'' must be calculated and considered in determining what speed DRAM can be used in a particular system design. The DRAM chosen must meet both the ''t_{RAC}'' and ''t_{CAC}'' parameters calculated.

EXAMPLE SYSTEM, 10 MHz, DP8400-2, DP8419, FAST ''A'' PART PALs

#1) $\overline{\text{RASIN}}$ low = T1 − 2 ns (F_{CLK} − PHI1 skew) + 15 ns (''A'' PAL clocked output) = 100 − 2 + 15 = 113 ns maximum

#2) $\overline{\text{RASIN}}$ to $\overline{\text{RAS}}$ low = 20 ns maximum (DP8419)

#3) $\overline{\text{RASIN}}$ to $\overline{\text{CAS}}$ low = 80 ns (DP8419 $\overline{\text{RASIN}}$–$\overline{\text{CAS}}$ low maximum)

#4) 74F244 transceiver delay = 7 ns maximum

#5) DP8400-2 data setup time to ''CSLE, DLE'' = 10 ns maximum

#6) Minimum ''CSLE, DLE'' delay into ''T3'' = Minimum ''A'' PAL delay − minimum FCLK to PHI1 skew = 8 − 2 = 6 ns minimum

''t_{RAC}'' = T1 + T2 + TW − #1 − #2 − #4 − #5 + #6
= 100 + 100 + 100 − 113 − 20 − 7 − 10 + 6
= 156 ns

''t_{CAC}'' = T1 + T2 + TW − #1 − #3 − #4 − #5 + #6
= 100 + 100 + 100 − 113 − 80 − 7 − 10 + 6
= 96 ns

Therefore the DRAM chosen should have a ''t_{RAC}'' less than or equal to 156 ns and a ''t_{CAC}'' less than or equal to 96 ns. Standard 150 ns DRAMs meet this criteria.

Approximately 150 ns minimum RAS precharge time.

Approximately 200 ns minimum CAS precharge time.

Approximately 230 ns minimum RAS pulse width.

Approximately 180 ns minimum CAS pulse width.

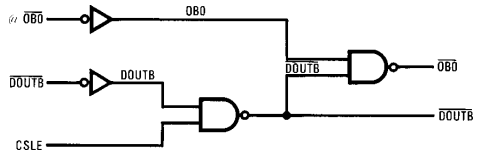
One must also consider the WRITE command to $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ lead times when choosing DRAMs for this system. During a READ access cycle, with a single bit error, a READ-MODIFY-WRITE access is performed. Here, the WRITE command to $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ lead times are one half period in length. This may present a problem to systems operating at frequencies of 10 MHz or greater. One can alleviate this problem by inserting an extra WAIT state into READ access cycles (see Use of P.I.C. at higher operating frequencies, #3) or by using external drivers from the PAL ''WE'' output to the DRAM ''WE'' input (thereby speeding up the $\overline{\text{WIN}}$ to $\overline{\text{WE}}$ delay and guaranteeing a greater $\overline{\text{WE}}$ to $\overline{\text{RAS}}$ and $\overline{\text{CAS}}$ lead time).

USE OF THE P.I.C. AT HIGHER FREQUENCIES

1) If one is using this interface above 4–6 MHz he should consider using the fast PAL's* (example ''PAL16R8A'' instead of ''PAL16R8''), a fast shift register (example 74F164), external fast logic (such as ''AS, ALS, or F'' type 74XX series) or the faster ''B'' type PALs to produce outputs ''$\overline{\text{DOUTB}}$, $\overline{\text{OB0}}$, $\overline{\text{OB1}}$'' to the DP8400, and the new DP8400-2 error correction chip. The fast PAL's* have an input to output maximum time of 25 ns, and 15 ns if it is a registered output. The slow PAL's* have an input to output maximum time of 35 ns, and 25 ns if it is a registered output.

One needs to produce ''$\overline{\text{DOUTB}}$, $\overline{\text{OB0}}$, $\overline{\text{OB1}}$'' faster at higher CPU speeds to guarantee that the CPU reads valid data during a READ access cycle. To do this he could use external fast logic as shown in the following figure.

Using the above example we can calculate (assuming a 10 MHz 32000 series processor) the time required to have valid data at the CPU data input pins.



TL/F/8400–1

@$\overline{\text{OB1}}$ would have the same configuration as $\overline{\text{OB0}}$

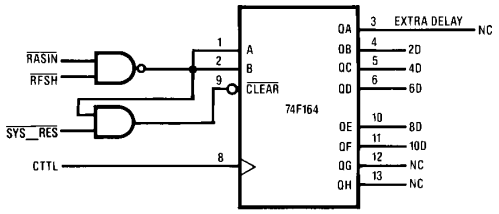
13 ns (maximum time of CSLE into state T3 assuming fast ''A'' PAL) + 9 ns (maximum 74ALS00 propagation delay) + 9 ns (max 74ALS00 prop delay) + 36 ns (maximum DP8400-2 ''$\overline{\text{OB0}}$, $\overline{\text{OB1}}$'' to output valid delay) + 7 ns (maximum 74F245 propagation delay) + 20 ns (data setup time required for the series 32000 with respect to the CTTL-clock) = 94 ns ***This value must not exceed 100 ns for a 10 MHz processor.

The delay of ''$\overline{\text{DOUTB}}$'' is to allow the DP8400 data, check bit and syndrome latches ''DLE, and CSLE'' to latch the data and check bits before turning off the DRAM output buffers.

The delay of ''$\overline{\text{OB0}}$ and $\overline{\text{OB1}}$'' allow the DRAM output buffers to turn off before the DP8400 starts driving the DP8400 memory data bus. In general the DRAM output buffers should turn off much faster then the DP8400 output buffers can turn on, so the user may want to allow ''$\overline{\text{OB0}}$, $\overline{\text{OB1}}$'' to become valid at the same time as ''$\overline{\text{DOUTB}}$'' transitions high.

2) In order to allow the use of slower DRAMs at higher CPU speeds one may want to slow down access cycles by adding an extra WAIT state.

To do this one could replace the 74LS164 IC with the following circuit:



TL/F/8400–2

Here ''CTTL'' was used instead of ''FCLK'' with a 74F164.

The ''$\overline{\text{RFSH}}$'' PAL equation must be adjusted to keep ''$\overline{\text{RFSH}}$'' 5 clock periods long, as follows:

$$\overline{\text{RFSH}}: = \overline{\text{RFIO}} * \text{INCY} * \overline{\text{2D}} + \overline{\text{RFSH}} * \overline{\text{RFIO}} + \overline{\text{RFSH}} * \overline{\text{6D}} + \overline{\text{RFSH}} * \text{CTTL}$$

If WAIT states are also wanted in WRITE access cycles the ''$\overline{\text{CWAIT}}$'' equations must include the following term:

$$+ \text{RFSH} * \overline{\text{INCY}} * \overline{\text{TSO}} * \text{DDIN} * \overline{\text{2D}}$$

If one wants to keep WRITE cycles without WAIT states inserted then the ''$\overline{\text{RASIN}}$'' equations must be modified for HIDDEN REFRESH and WRITE cycles as follows:

$$+ \text{RFSH} * \overline{\text{RASIN}} * \overline{\text{INCY}} * \overline{\text{2D}}$$

3) Another possibility for this interface at higher frequencies would be to adjust READ access cycles by adding another WAIT state to them, as well as adjusting BYTE WRITE cycles.

Using this method one would need another stage for the shift register or use a 74F164 and use CTTL as its clock instead of FCLK. If one looks at the above figure, using the 74F164, for reference the extra stage ''10D'' would be used. This would allow one to make the READ access cycle one ''T'' state longer by adjusting the READ and READ with error ''$\overline{\text{RASIN}}$'' equations.

To make the READ access cycle one ''T'' state longer another WAIT state would have to be added to READ cycles (making a total of 2 WAIT states) and the latch signals ''$\overline{\text{ODLE}}$'' and ''CSLE'' must be adjusted by delaying them back ½ ''T'' state (allowing a ½ cycle longer access time). This also has the advantage of allowing the other ½ cycle of time to get the data valid at the inputs of the Series 32000 CPU.

The BYTE WRITE access cycle could also be adjusted by delaying the signals ''$\overline{\text{ODLE}}$'' and ''CSLE'' by ½ cycle. No other equations need to be touched. This would allow an extra ½ cycle for access time during BYTE WRITE access cycles.

This would allow a standard 150 ns to possibly 200 ns DRAM in a 10 MHz system [80.5 ns + ½ ''T'' state (50 ns) = 130.5 ns column access time (t_{CAC})] but would sacrifice by having 2 WAIT states in READ access cycles.

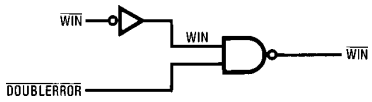
4) One also must be careful to make sure that $\overline{\text{CS}}$ is low, during an access, a minimum of 30 ns (DP8409A, 15 ns DP8419) before $\overline{\text{RASIN}}$ transitions low. If this is a problem one could tie $\overline{\text{CS}}$ permanently low (disabling hidden REFRESH) and use the system transceivers to select the memory system.

OTHER OPTIONS

If one is using the NS32082 Memory Management unit in a Series 32000 system he should connect the output ''$\overline{\text{PAV}}$'' (Physical Address Valid) to the P.I.C. instead of the address strobe output ''$\overline{\text{ADS}}$''.

An output for the BUS PARITY ERROR in a data transfer from the CPU to memory could also be detected, from the error flags and ''AE'' of the DP8400, and used to interrupt the CPU. However, the P.I.C. does not make use of that feature of the DP8400, though it would be very easy to add.

If one does not want to WRITE corrected data to memory in case of a DOUBLE BIT error, in READ access cycle, he could disable the WRITE signal, ''$\overline{\text{WIN}}$'', during a DOUBLE BIT error as follows:



TL/F/8400–3

NS32016, DP8400, DP8409A PALs Inputs and Outputs

PIN NUMBER OF THE PAL ON THE LEFT

PAL #1 Inputs

1)	''FCLK''	Fast Clock (twice ''CTTL'' frequency) from NS32201.
2)	''CTTL''	Output clock from NS32201.
3)	''$\overline{CS}$''	Chip Select for the Dynamic RAM controlled by the DP8409A and DP8400.
4)	''$\overline{DDIN}$''	Data Direction in, from NS32016, indicates the direction of the data transfer during a bus cycle.
5)	''RFIO''	Refresh request output from the DP8409A, also is used as a reset input to set PAL to a known state.
6)	''$\overline{INCY}$''	Output from PAL #2 indicating that the NS32016 is in an access cycle.
7)	''$\overline{AOHBE}$''	If address bit 0 and high byte enable (from NS32016) are both low this input is high. Used to determine when byte operations are in progress.
8)	''2D''	''RASIN'' or ''RFSH'' delayed by 2 periods of FCLK. This output is from the external shift register.
9)	''$\overline{ERRLAT}$''	Output from PAL #3 indicating that any error, ''AE'', was valid during a READ access cycle.
11)	''$\overline{OE}$''	Enables PAL outputs.
12)	''4D''	''2D'' delayed by 2 periods of RFCK, also an output of the external shift register.
18)	''6D''	''4D'' delayed by 2 periods of RGCK, also an output of the external shift register.
19)	''8D''	''6D'' delayed by 2 periods of RGCK, also an output of the external shift register.

PAL #1 Outputs

17)	''$\overline{RASIN}$''	Input to DP8409A
16)	''$\overline{RFSH}$''	Input to DP8409A, causes the DP8409A to enter mode 1 to do a refresh.
15)	''$\overline{WIN}$''	This output is used as an input to the DP8409A. It causes a WRITE to the DRAM.
14)	''$\overline{CYCLED}$''	This output is used in many other equasions and functions as a signal that the particular access cycle is midway to completion.

PAL #2 Inputs

1)	''$\overline{RFSH}$''	Output from PAL #1 that indicates whether the DRAMs are being refreshed.
2)	''$\overline{RASIN}$''	Output from PAL #1.
3)	''AO''	Output from NS32016, address bit 0.
4)	''$\overline{HBE}$''	Output from NS32016, high byte enable.
5)	''$\overline{DDIN}$''	Data Direction in, from NS32016.
6)	''$\overline{ADS}$''	Address strobe from NS32016.
7)	''$\overline{TSO}$''	Output from NS32016.
8)	''2D''	Output from the shift register.
9)	''$\overline{CS}$''	Chip select for the DRAM.
11)	''$\overline{CYCLED}$''	Output from PAL #1.
13)	''$\overline{ODLE}$''	Output Latch Enable to the DP8400 (Output from PAL #3).

PAL #2 Outputs

19)	''$\overline{PBUF1}$''	This signal enables the high byte of the processor, through the CPU transceiver, onto the DP8400/ Memory data bus.
18)	''$\overline{OB1}$''	Controls DP8400 output buffer for byte ''1''.
17)	''$\overline{OB0}$''	Controls DP8400 output buffer for byte ''0''.
16)	''$\overline{PBUFO}$''	This signal enables the low byte of the processor, through the CPU transceiver, onto the DP8400/ Memory data bus.
15)	''$\overline{DOUTB}$''	Controls memory buffers that interface between the DRAM and the DP8400 memory data bus.
14)	''$\overline{INCY}$''	Output indicating that the NS32016 is in an access cycle.
12)	''$\overline{CWAIT}$''	Output to NS32016 that causes WAIT states to be inserted into the NS32016 bus cycles.

PAL #3 Inputs

1)	''FCLK''	Fast clock from NS32201.
2)	''CTTL''	System clock from the NS32201.
3)	''DIAGCS''	Enable input from I/O port for diagnostics to enable ''CSLE'', check bit syndrome latch enable.
4)	''DIAGD''	Enable input from I/O port for diagnostics to enable ''DLE'', data latch enable.
5)	''$\overline{RESET}$''	Reset input from I/O port to reset PAL error latches.
6)	''CSRASIN''	Output from the PAL #1 logically ''NOR''ed with the DRAM Chip Select signal. This indicates the beginning of a selected DRAM access cycle.
7)	''AE''	Output from DP8400 indicating an error.
8)	''E01''	This is the ''E0'' and ''E1'' error flags, of the DP8400, logically ''NOR''ed together.
9)	''$\overline{DOUTB}$''	Controls memory buffers that interface between the DRAM and the DP8400/memory data base.
11)	''$\overline{OE}$''	Enables the PAL outputs.
12)	''$\overline{AOHBE}$''	If address bit 0 AND high byte enable (from NS32016) are both low this input is high. Used to determine when byte operations are in progress.
19)	''$\overline{DDIN}$''	Data Direction in, from NS32016.

PAL #3 Outputs

18)	''$\overline{ODLE}$''	Output that controls both the DP8400 Data latch and output latches. This output goes directly to both the ''DLE'' and $\overline{OLE}$ pin of the DP8400.
17)	''$\overline{CSLE}$''	Output that controls the DP8400 Check bit Syndrome latch. This output goes directly to the ''CSLE'' pin of the DP8400, it is only inverted so the PAL programmer will program it correctly.

NS32016, DP8400, DP8409A PALs Inputs and Outputs (Continued)

16) ''$\overline{\text{MODECC}}$'' Output that is used as an input to the DP8400. This signal controls whether the DP8400 is in READ or WRITE Mode.

15) ''$\overline{\text{DOUBLERR}}$'' Used to interrupt the system when a double bit error has been detected during a READ cycle.

14) ''$\overline{\text{ERRLAT}}$'' Used in the PAL controller to indicate that an error has occurred during a $\overline{\text{CS}}$ READ cycle or a $\overline{\text{CS}}$ BYTE WRITE cycle, as indicated by ''AE'' being valid. This signal can be used to latch the DRAM bank in error, the SYNDROME of the error, the ERROR flags, and the DRAM address (of the data in error) when a DRAM error occurs.

13) ''$\overline{\text{ERROR}}$'' This output is used to display the DRAM bank in error, the syndrome of the error, and the error flags of the DP8400 when a single, double, or triple bit error occurs. The preceding error condition is held in an external error register (74LS374's). The contents of the registers are displayed on LED's to help the user diagnose where a DRAM problem may reside in the memory system.

PAL NUMBER 1

```
PAL16R4A
FCLK CTTL /CS /DDIN RFIO /INCY /A0HBE 2D /ERRLAT GND
/OE 4D NC /CYCLED /WIN /RFSH /RASIN 6D 8D VCC
/RASIN : = RFSH*/INCY*/4D*/CTTL*ERRLAT          ;Start /RASIN
    +RFSH*/RASIN*/INCY*/4D                      ;WRITE or hidden RFSH
    +RFSH*/CS*/RASIN*/INCY*/DDIN*/6D            ;READ cycle
    +RFSH*/CS*/RASIN*/INCY*DDIN*/A0HBE*WIN      ;BYTE WRITE cycle
    +RFSH*/CS*/RASIN*/INCY*DDIN*/A0HBE*CTTL     ;Extend BYTE WRITE
    +RFSH*/CS*/RASIN*/DDIN*/ERRLAT*/8D          ;READ w/error

/RFSH : = /RFIO*INCY*RASIN                  ;RFSH in idle states or in long
       +/RFSH*/RFIO                         ; accesses of other devices or
       +/RFSH*/8D                           ; at the beginning of an access
       +/RFSH*CTTL

/WIN : =
     RFSH*/CS*/RASIN*/ERRLAT*6D*/CTTL*/DDIN     ;READ w/error
     +/WIN*RFSH*/RASIN*/ERRLAT*6D               ;READ w/error continue
     +RFSH*/CS*/RASIN*DDIN*2D*CTTL*A0HBE        ;WRITE
     +/WIN*RFSH*/CS*/RASIN*DDIN*2D*A0HBE        ;WRITE continue
     +RFSH*/CS*/RASIN*DDIN*/A0HBE*/CYCLED*/CTTL ;BYTE WRITE
     +/WIN*RFSH*/CS*/RASIN*DDIN*/A0HBE*6D       ;BYTE WRITE continue

/CYCLED : =
    RFSH*/RASIN*/CS*DDIN*4D*/A0HBE*/CTTL        :BYTE WRITE
    +RFSH*/RASIN*/CS*DDIN*/A0HBE*4D*/CYCLED     ;BYTE WRITE
    +RFSH*/RASIN*/CS*/DDIN*2D*/CTTL             :READ, READ w/error
    +RFSH*/RASIN*/CS*/DDIN*4D*/CYCLED           ;READ, READ w/error
    +RFSH*/RASIN*/CS*DDIN*2D*A0HBE              ;WRITE
    +RFSH*/RASIN*CS*2D*/CTTL                    ;HIDDEN REFRESH
    +RFSH*/CYCLED*/ERRLAT                       :Finish for READ w/error
    +RFSH*/CYCLED*CTTL                          ;Finish
```

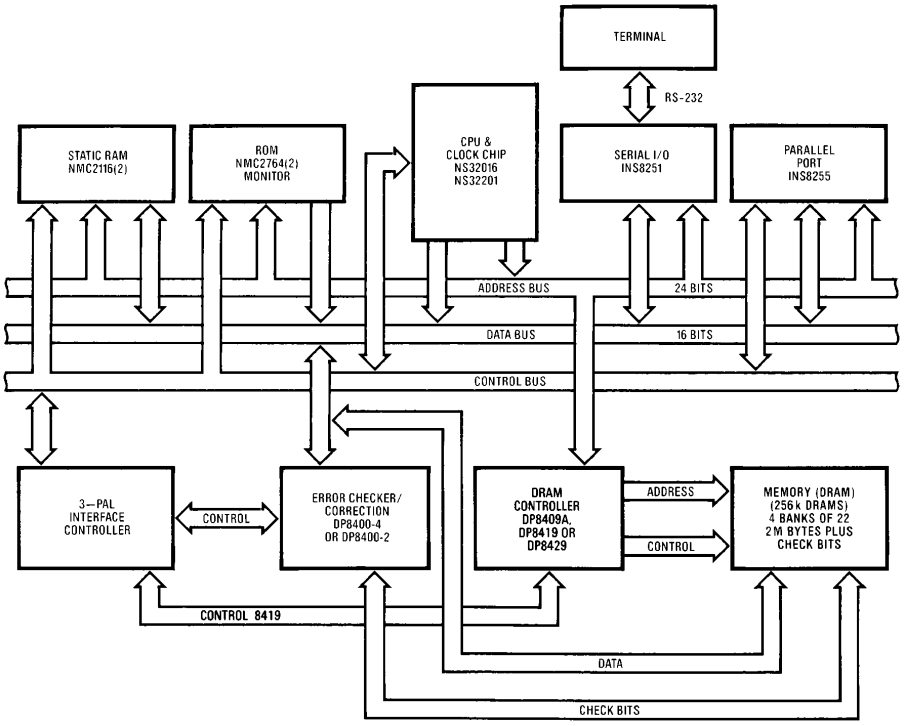
PAL NUMBER 2

```
PAL16L8A
/RFSH /RASIN AO /HBE /DDIN /ADS /TSO 2D /CS GND
/CYCLED /CWAIT /ODLE /INCY /DOUTB /PBUF0 /OB0 /OB1 /PBUF1 VCC

IF (VCC) /PBUF1 =
   RFSH*/CS*/INCY*/DDIN*2D*/HBE                  ;READ or READ w/error
   +RFSH*/CS*/INCY*DDIN*AO*/HBE*DOUTB*/ODLE*2D   ;BYTE WRITE high
   +RFSH*/CS*/INCY*DDIN*2D*/OB0*AO*/HBE          ;BYTE WRITE cont
   +RFSH*/CS*/INCY*DDIN*/AO*/HBE*DOUTB           ;word WRITE

IF (VCC) /OB1 =
   RFSH*/CS*/INCY*/DDIN*2D*/CYCLED*DOUTB         ;READ or READ w/error
   +RFSH*/CS*/INCY*DDIN*/AO*HBE*2D*/ODLE*DOUTB   ;BYTE WRITE low
   +RFSH*/CS*/INCY*DDIN*/AO*HBE*2D*/OB1*DOUTB    ;BYTE WRITE cont
   +RFSH*/OB1*DOUTB*2D                           ;READ w/error hold

IF (VCC) /OB0 =
   RFSH*/CS*/INCY*/DDIN*2D*/CYCLED*DOUTB       ;READ or READ w/error
   +RFSH*/CS*/INCY*DDIN*AO*/HBE*2D*/ODLE*DOUTB ;BYTE WRITE high
   +RFSH*/CS*/INCY*DDIN*AO*/HBE*2D*/OB0*DOUTB  ;BYTE WRITE cont
   +RFSH*/OB0*DOUTB*2D                         ;READ w/error hold

IF (VCC) /PBUF0 =
   RFSH*/CS*/INCY*/DDIN*2D*/AO                ;READ or READ w/error
  +RFSH*/CS*/INCY*DDIN*/AO*HBE*DOUTB*/ODLE*2D ;BYTE WRITE low
  +RFSH*/CS*/INCY*DDIN*2D*/AO*HBE*/OB1        ;BYTE WRITE cont
  +RFSH*/CS*/INCY*DDIN*/AO*/HBE*DOUTB         ;word WRITE

IF (VCC) /DOUTB =
    RFSH*/CS*/INCY*/DDIN*2D*CYCLED             ;READ or READ w/error
    +RFSH*/CS*/INCY*DDIN*/AO*HBE*2D*ODLE*OB1   ;BYTE WRITE low
    +RFSH*/CS*/INCY*DDIN*AO*/HBE*2D*ODLE*OB0   ;BYTE WRITE high

IF (VCC) /INCY = RFSH*/ADS*/2D*CYCLED      ;Start INCY
      +RFSH*/CS*/TSO*/2D                   ;Start INCY for access
                                           ; after forced refresh
                                           ; or READ w/error
      +RFSH*/INCY*CYCLED                   ;Continue
      +RFSH*/INCY*/TSO*/CS                 ;Continue for /CS access

IF (/CS) /CWAIT =
    /RFSH*/TSO                             ;Access in RFSH
    +RFSH*/TSO*RASIN                       ;Access after forced RFSH
    +RFSH*/INCY*/TSO*/DDIN*/2D             ;READ cycle
    +RFSH*/INCY*/TSO*DDIN*/AO*HBE*CYCLED   ;BYTE WRITE
    +RFSH*/INCY*/TSO*DDIN*AO*/HBE*CYCLED   ;BYTE WRITE
    +RFSH*/TSO*/CYCLED*/2D*RASIN           ;WAIT after READ w/error
```

PAL NUMBER 3

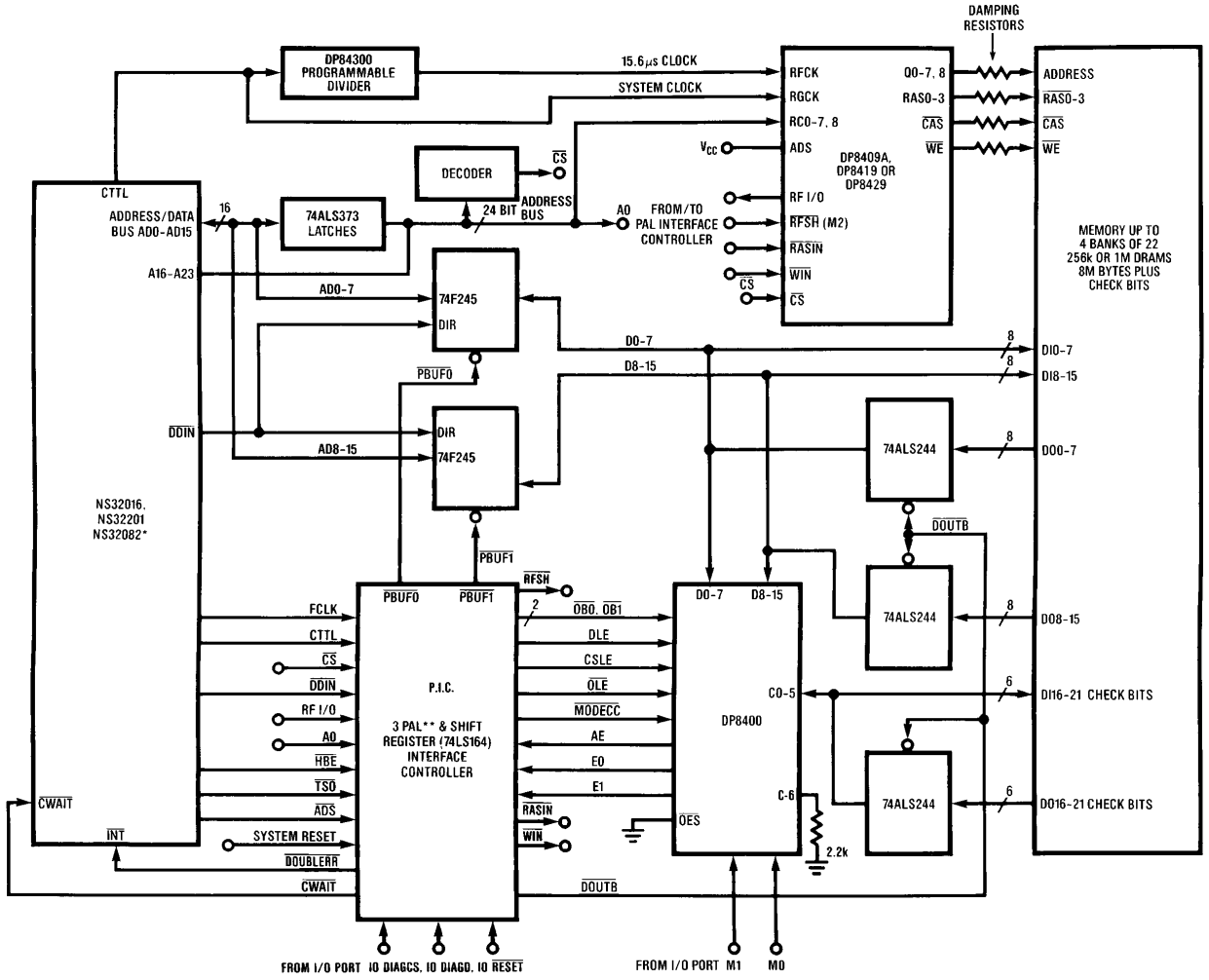
```
PALl6R6A
FCLK CTTL DIAGCS DIAGD /RESET CSRASIN AE EOl /DOUTB GND
/OE /AOHBE /ERROR /ERRLAT /DOUBLERR /MODECC /CSLE /ODLE /DDIN VCC

/ODLE : = CSRASIN*/DDIN*/DOUTB*/CTTL              ;Read
        +CSRASIN*/DDIN*/MODECC*CSLE*ODLE          ;Read with error
        +CSRASIN*DDIN*/AOHBE*/DOUTB*/CTTL         ;Byte Write
        +/ODLE*CSRASIN*DDIN*/AOHBE*CTTL           ;Continue during Byte Write
        +CSRASIN*DDIN*/AOHBE*/MODECC*CSLE*ODLE    ;Byte Write
        +CSRASIN*DDIN*AOHBE*/CTTL                 ;Word Write
        +/ODLE*CSRASIN*CTTL                       ;Hold "/ODLE"
        +/ODLE*DIAGD                              ;Hold "/ODLE" for
                                                  ;  diagnostics

/CSLE : = CSRASIN*/DDIN*/DOUTB*/CTTL              ;Read
        +CSRASIN*/DDIN*/MODECC                    ;Read with error
        +CSRASIN*DDIN*/AOHBE*/DOUTB*/CTTL         ;Byte Write
        +CSRASIN*DDIN*/AOHBE*/MODECC              ;Byte Write
        +CSRASIN*DDIN*AOHBE*/CTTL                 ;Word WRITE
        +/CSLE*CSRASIN*CTTL                       ;Hold "/CSLE"
        +/CSLE*DIAGCS                             ;Hold "/CSLE" for
                                                  ;  diagnostics

/MODECC :=
    CSRASIN*/ODLE*/DDIN*/CTTL         ;READ or Write w/error
   +CSRASIN*/ODLE*DDIN*/AOHBE*/CTTL   ;BYTE WRITE
   +CSRASIN*DDIN*AOHBE                ;WORD WRITE
   +/MODECC*CSRASIN                   ;Hold "/MODECC"

/DOUBLERR :=
  /DIAGCS*/DIAGD*RESET*CSRASIN*/ODLE*/CTTL*AE*EOl  ;Double bit error
                                                   ;  during READs
                                                   ;  or BYTE WRITEs
  +/DOUBLERR*RESET                                 ;Hold "/DOUBLERR"

/ERRLAT :=
    /DIAGCS*/DIAGD*CSRASIN*/ODLE*/CTTL*AE       ;Any Error during
                                                ;  READ or BYTE WRITE
    +/ERRLAT*CSRASIN                            ;Continue "/ERRLAT" during
                                                ;  READ or during BYTE WRITE

/ERROR  :=  /DIAGD*/DIAGCS*RESET*CSRASIN*/ERRLAT ;Store error syndrome
                                                 ;  and RAS bank and
                                                 ;  error flags
             +/ERROR*RESET                       ;Hold until RESET

;The output, "/CSLE", is shown inverted so the PAL will be
;programmed correctly, in other words, "/CSLE" goes low after the
;rising edge of FCLK given that one of its input equations was
;low a setup time before FCLK transitioned high. The output,
;"/CSLE", should go straight to the pin "CSLE" of the DP8400.
```

DP8400, DP8409A, NS32016 Error Correcting Dynamic RAM Computer System



TL/F/8400–4

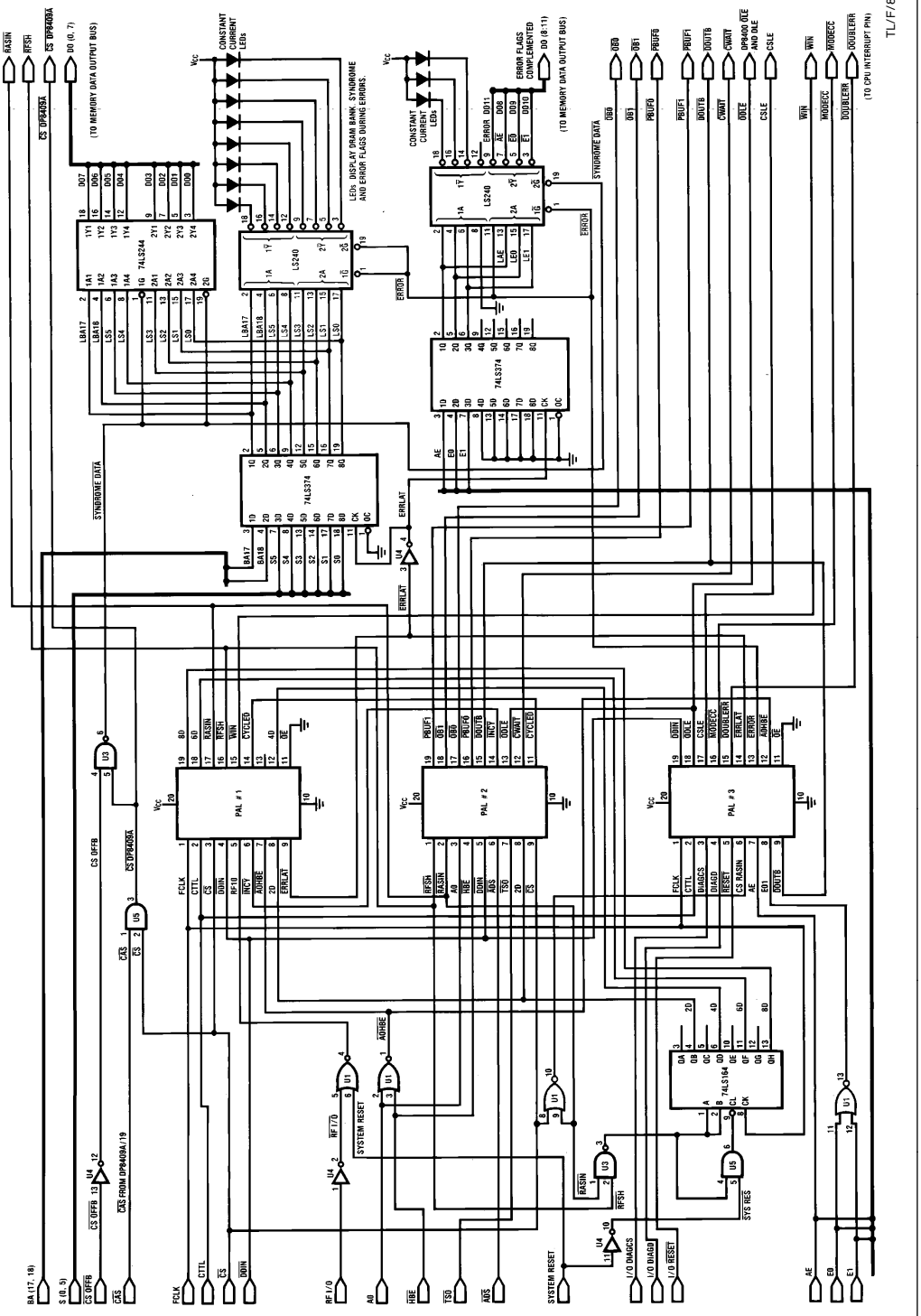
NS32016, DP8400, DP8409A or DP8418 Error Correcting Memory System



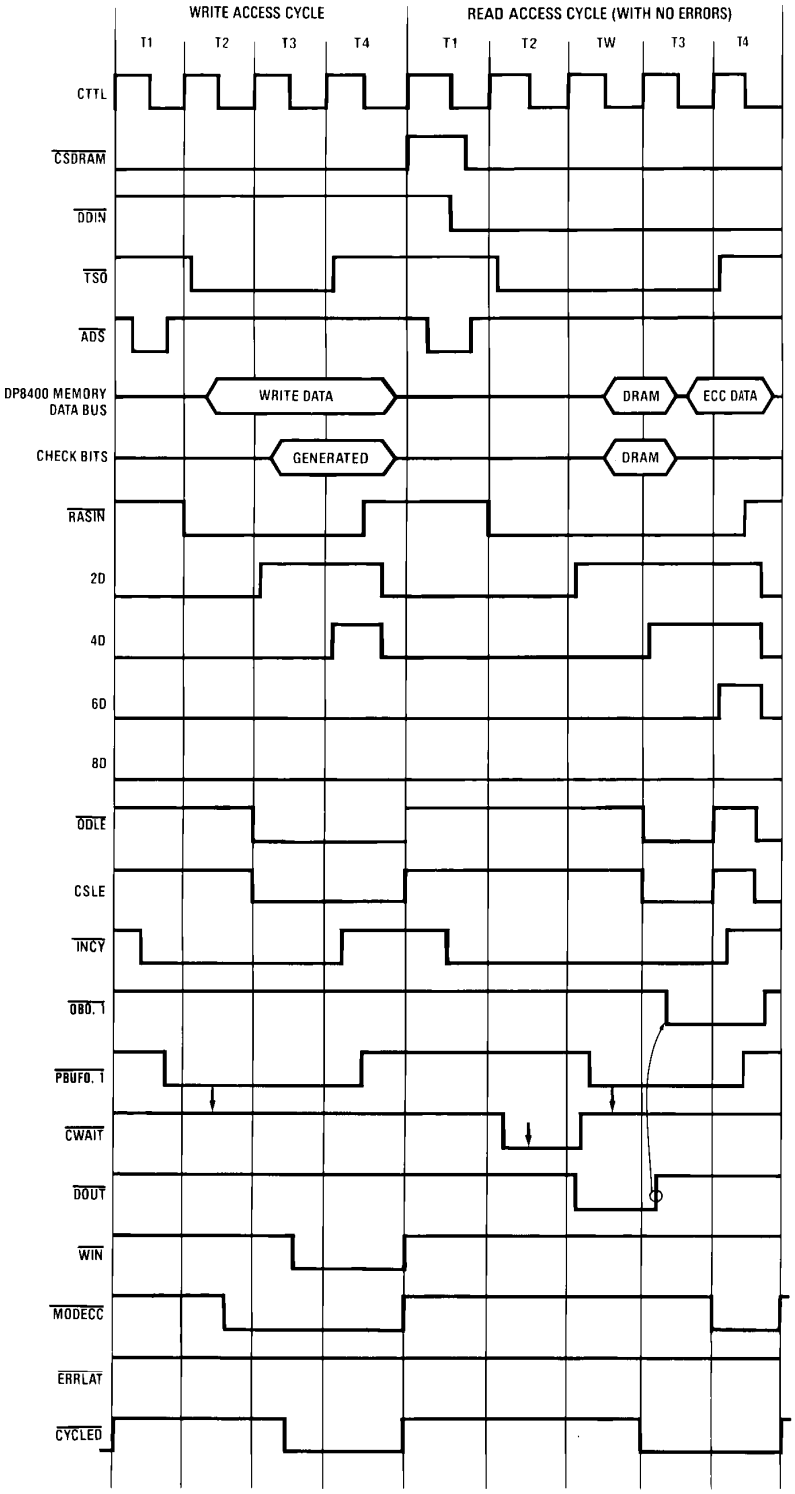
TL/F/8400–5

*IF system contains NS32082 MMU $\overline{PAV}$ should be used in place of $\overline{ADS}$

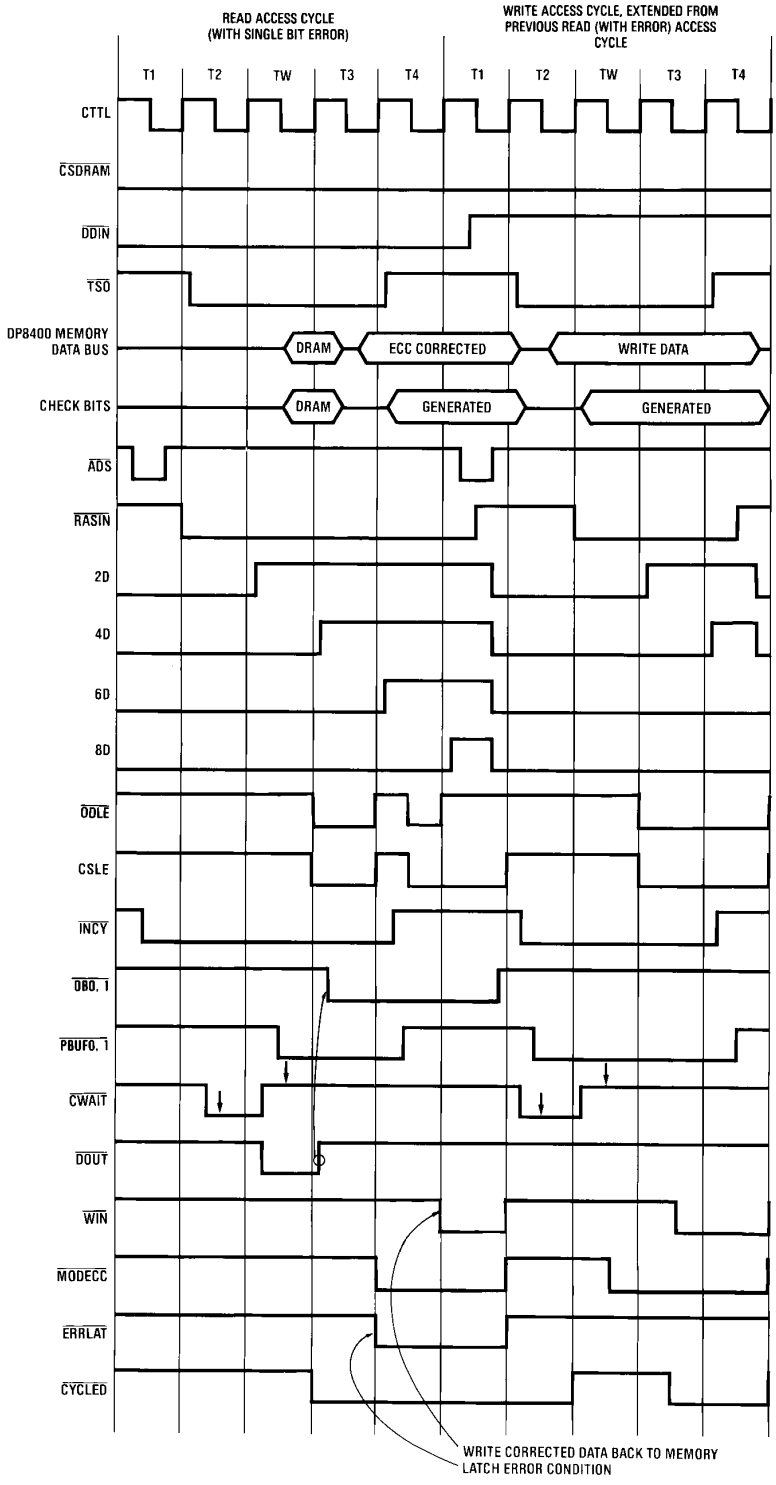
P.I.C. 3 PAL and Shift Register Interface Controller



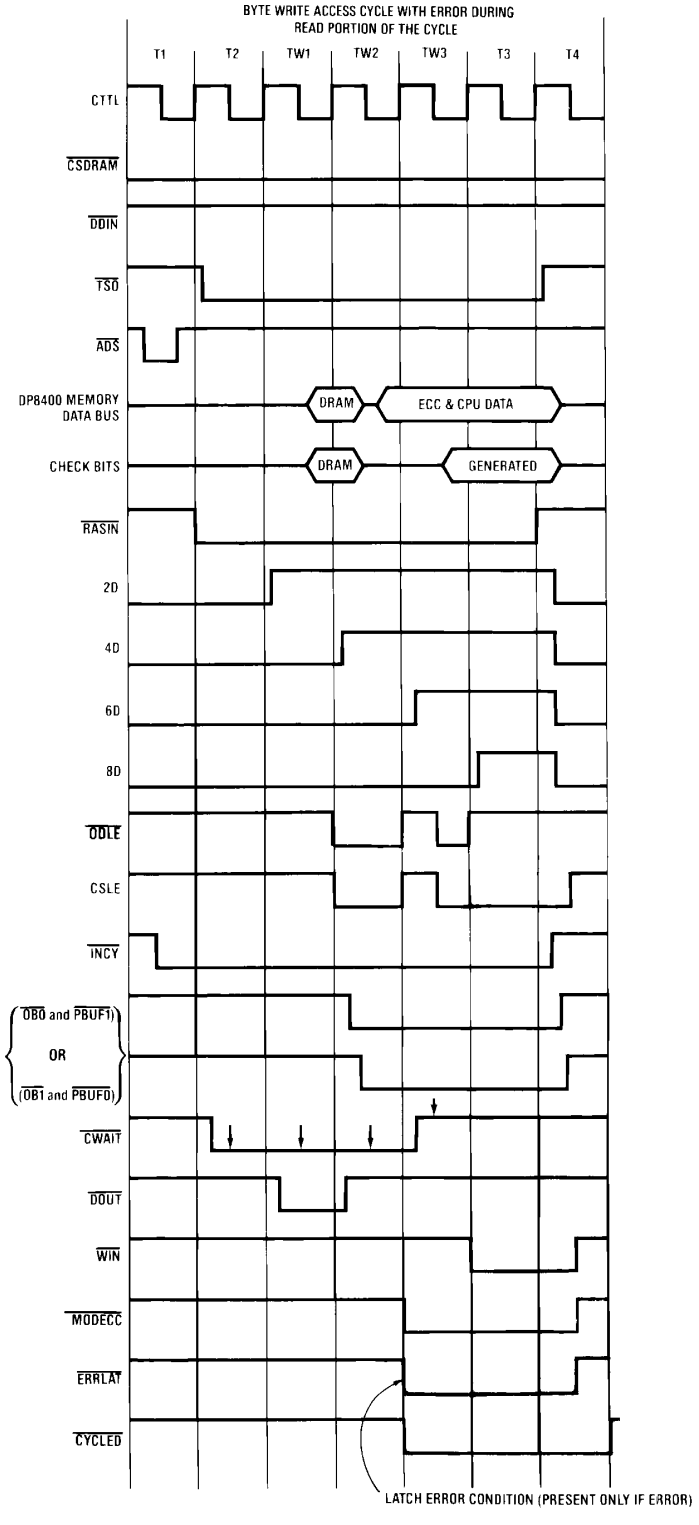
TL/F/8400–6


WRITE ACCESS CYCLE
READ ACCESS CYCLE (WITH NO ERRORS)
T1
T2
T3
T4
T1
T2
TW
T3
T4
CTTL
CSDRAM
DDIN
TSO
ADS
DP8400 MEMORY DATA BUS
WRITE DATA
DRAM
ECC DATA
CHECK BITS
GENERATED
DRAM
RASIN
2D
4D
6D
8D
ODLE
CSLE
INCY
OBO. 1
PBUF0. 1
CWAIT
DOUT
WIN
MODECC
ERRLAT
CYCLED

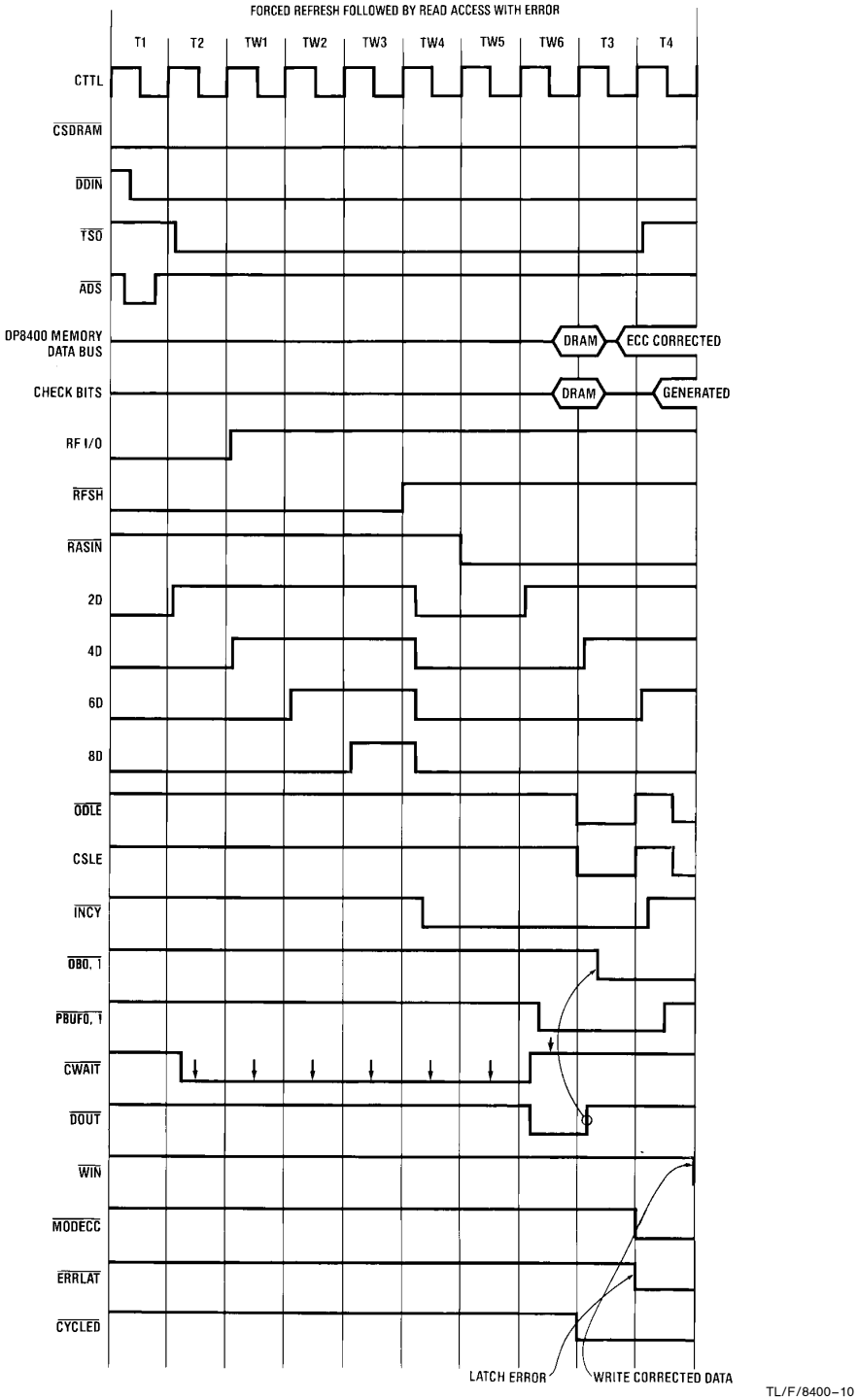
TL/F/8400–7


READ ACCESS CYCLE
(WITH SINGLE BIT ERROR)
WRITE ACCESS CYCLE, EXTENDED FROM
PREVIOUS READ (WITH ERROR) ACCESS
CYCLE
T1
T2
TW
T3
T4
T1
T2
TW
T3
T4
CTTL
CSDRAM
DDIN
TSO
DP8400 MEMORY
DATA BUS
DRAM
ECC CORRECTED
WRITE DATA
CHECK BITS
DRAM
GENERATED
GENERATED
ADS
RASIN
2D
4D
6D
8D
ODLE
CSLE
INCY
OBO, 1
PBUF0, 1
CWAIT
DOUT
WIN
MODECC
ERRLAT
CYCLED
WRITE CORRECTED DATA BACK TO MEMORY
LATCH ERROR CONDITION

TL/F/8400–8

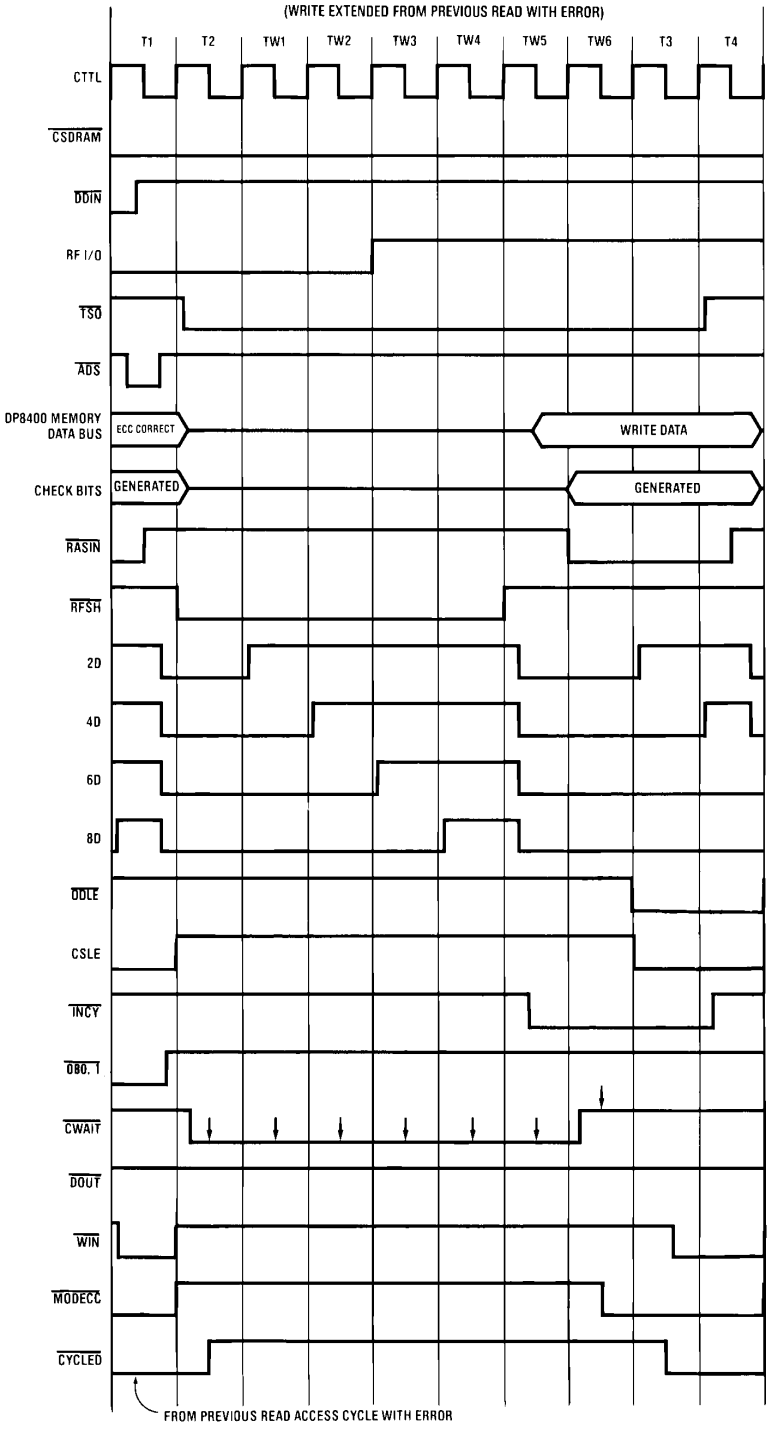

BYTE WRITE ACCESS CYCLE WITH ERROR DURING READ PORTION OF THE CYCLE
T1
T2
TW1
TW2
TW3
T3
T4
CTTL
CSDRAM
DDIN
TSO
ADS
DP8400 MEMORY DATA BUS
DRAM
ECC & CPU DATA
CHECK BITS
DRAM
GENERATED
RASIN
2D
4D
6D
8D
ODLE
CSLE
INCY
OB0 and PBUF1
OR
OB1 and PBUF0
CWAIT
DOUT
WIN
MODECC
ERRLAT
CYCLED
LATCH ERROR CONDITION (PRESENT ONLY IF ERROR)

TL/F/8400–9


FORCED REFRESH FOLLOWED BY READ ACCESS WITH ERROR
T1
T2
TW1
TW2
TW3
TW4
TW5
TW6
T3
T4
CTTL
CSDRAM
DDIN
TSO
ADS
DP8400 MEMORY DATA BUS
DRAM
ECC CORRECTED
CHECK BITS
DRAM
GENERATED
RF I/O
RFSH
RASIN
2D
4D
6D
8D
ODLE
CSLE
INCY
OBO, 1
PBUF0, 1
CWAIT
DOUT
WIN
MODECC
ERRLAT
CYCLED
LATCH ERROR
WRITE CORRECTED DATA

TL/F/8400–10

FORCED REFRESH WITH WRITE ACCESS AT THE SAME TIME
(WRITE EXTENDED FROM PREVIOUS READ WITH ERROR)



TL/F/8400–11

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
1111 West Bardin Road
Arlington, TX 76017
Tel: 1(800) 272-9959
Fax: 1(800) 737-7018

National Semiconductor Europe
Fax: (+49) 0-180-530 85 86
Email: cnjwge@tevm2.nsc.com
Deutsch Tel: (+49) 0-180-530 85 85
English Tel: (+49) 0-180-532 78 32
Français Tel: (+49) 0-180-532 93 58
Italiano Tel: (+49) 0-180-534 16 80

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductor Japan Ltd.
Tel: 81-043-299-2309
Fax: 81-043-299-2408