intel®

AP-442

APPLICATION NOTE

33 MHz 386 System Design Considerations

SHAHZAD BAQAI
KIYOSHI NISHIDE

May 1990

Order Number: 240725-001

Information in this document is provided in connection with Intel products. Intel assumes no liability whatsoever, including infringement of any patent or copyright, for sale and use of Intel products except as provided in Intel's Terms and Conditions of Sale for such products.

Intel retains the right to make changes to these specifications at any time, without notice. Microcomputer Products may have minor variations to this specification known as errata.

*Other brands and names are the property of their respective owners.

†Since publication of documents referenced in this document, registration of the Pentium, OverDrive and iCOMP trademarks has been issued to Intel Corporation.

Contact your local Intel sales office or your distributor to obtain the latest specifications before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

33 MHz 386 SYSTEM DESIGN CONSIDERATIONS

CONTENTS

PAGE

CONTENTS

PAGE

RELATED DOCUMENTATION

This Application Note should be used in conjunction with the 386 DX microprocessor Data Sheet (Order Number 231630-011) and the 386 DX Hardware Reference Manual (Order Number 231732-004). A list of related references is provided in the appendix for getting more information on high speed design issues.

SECTION I. INTRODUCTION

The 386 DX Microprocessor is an advanced 32-bit microprocessor designed using Intel's CHMOS IV process for applications which require very high performance. It is optimized for multitasking operating systems. The 32-bit register and data paths support 32-bit address and data types allowing up to four gigabytes of physical memory and 64 terabytes of virtual memory to be addressed. The integrated memory management and protection architecture includes address translation registers, advanced multitasking hardware and a protection mechanism to support operating systems. In addition, the 386 DX microprocessor allows the simultaneous running of DOS with other operating systems. Instruction pipelining, on chip address translation and high bus bandwidth ensure short average instruction execution times and high system throughput. To facilitate high performance system hardware designs, the 386 DX microprocessor bus interface offers address pipelining, dynamic data bus sizing and direct byte enable signals for each byte of the data bus.

This Application Note is intended to show how to complete a successful design of a 'Core' system using the 386 DX-33, the 33 MHz clock version. A Core system is a minimum system configuration, in this case comprising the CPU, the 82385 32-bit Cache controller, Dynamic and Static RAM and an I/O mechanism with which to communicate with the CPU.

The Application Note examines the design techniques necessary when executing a design at this frequency. Many of the methods used at lower frequencies, such as 16 MHz and 20 MHz, are no longer valid at this higher frequency. Phenomena, whose effects are negligible at the lower frequencies, must be taken into account in the design. The physical positioning of components relative to each other plays a significant part in the success of the design, since transmission line effects (reflection, radiation) are no longer negligible.

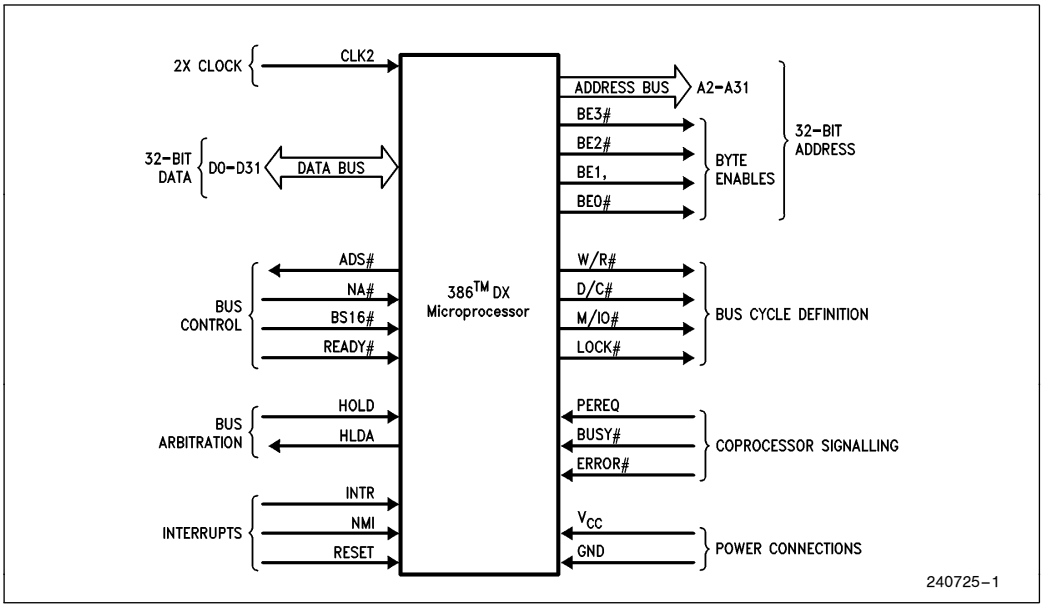


Figure 1-1. Functional Signal Groups

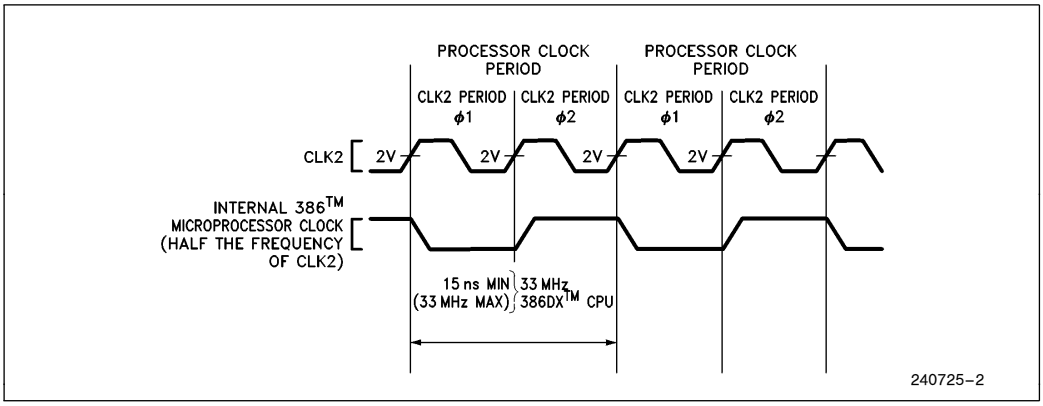


Figure 1-2. CLK2 Signal and Internal Processor Clock

SECTION II. HIGH SPEED SYSTEM DESIGN CONSIDERATIONS

2.1 Overview Of High Speed Effects

This section is included as a brief overview of general issues that are applicable to both higher and lower frequencies of circuit design.

The CHMOS IV 386 DX CPU differs from previous HMOS microprocessors in that its power dissipation is primarily capacitive; there is almost no DC power dissipation. Power dissipation depends mostly on frequency. This fact is used in designs where power consumption is critical.

Power dissipation can be distinguished as either internal (logic) power or I/O (bus) power. Internal power varies with operating frequency and to some extent with wait states and software. Internal power increases with supply voltage also. Process variations in manufacturing affect internal power, although to a lesser extent than with NMOS processes.

I/O power, which accounts for roughly one-fifth of the total power dissipation, varies with frequency and voltage. It also depends on capacitive bus load. Capacitive bus loadings for all output pins are specified in the 386 DX CPU data sheet. The 386 DX CPU output valid delays will increase if these loadings are exceeded. The addressing pattern of the software can affect I/O power by changing the effective frequency at the address pins. The variation in frequency at the data pins tends to be smaller; thus varying data patterns should not cause a significant change in power dissipation.

POWER AND GROUND PLANES

Power and ground planes must be used in 386 DX CPU systems to minimize noise. Power and ground lines have inherent inductance and capacitance, therefore an impedance $z = (L/C)^{*1/2}$. The total characteristic impedance for the power supply can be reduced by adding more lines. This effect is illustrated in 2.1 which shows that two lines in parallel have half the impedance of one. To reduce the impedance even further, the user should add more lines. In the limit, an infinite number of parallel lines, or a plane, results in the lowest impedance. Planes also provide the best distribution of power and ground.

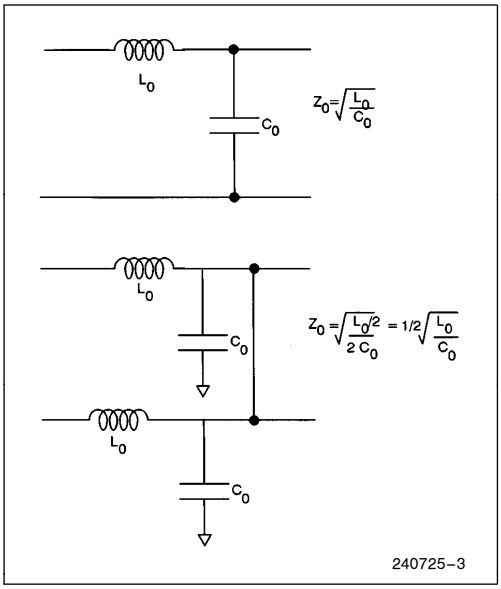


Figure 2-1. Reducing Characteristic Impedance

The 386 DX CPU has 20 V_{CC} pins and 21 V_{SS} (ground) pins. All power and ground pins must be connected to a plane. Ideally, the 386 DX CPU is located at the center of the board, to take full advantage of these planes. Although the 386 DX CPU generally demands less power than the 80286, the possibility of power surges is increased due to higher frequency and pin count. Peak-to-peak noise on V_{CC} relative to V_{SS} should be maintained at no more than 400 mV, and preferably to no more than 200 mV.

DECOUPLING CAPACITORS

The switching activity of one device can propagate to other devices through the power supply. For example, in the TTL NAND gate of Figure 2.2, both Q3 and Q4 transistors are on for a short time when the output is switching. This increased load causes a negative spike on V_{CC} and a positive spike on ground.

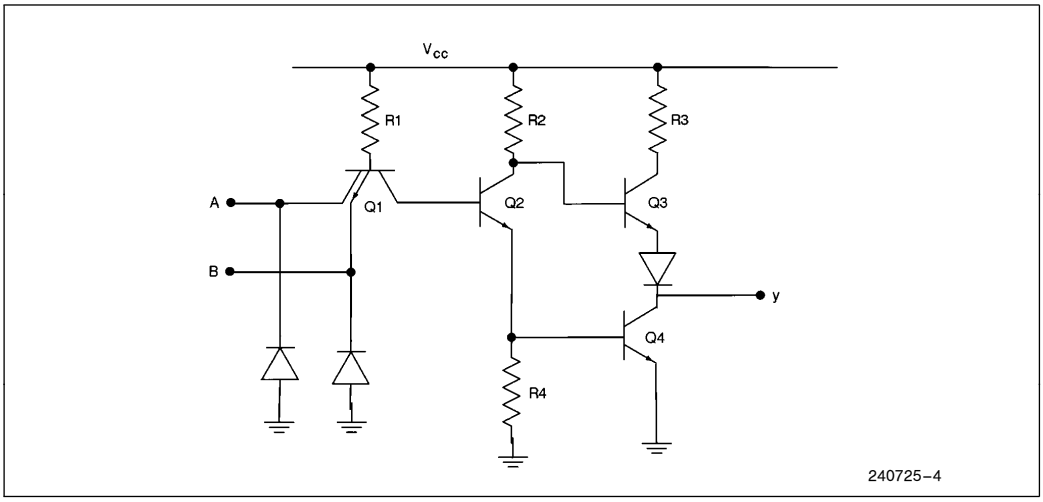


Figure 2-2. Circuit without Decoupling

In synchronous systems in which many gates switch simultaneously, the result is signifcant noise on the power and ground lines.

Decoupling capacitors placed across the device between Vcc and ground reduce Voltage spikes by supplying the extra current needed during switching. These capacitors should be placed close to their devices because the inductance or connection lines negates their effect.

When selecting decoupling capacitors, the user should provide 0.01 microfarads for each device and 0. 1 microfarads for every 20 gates. Radio-frequency capacitors must be used; they should be distributed evenly over the board to be most effective. In addition, the board should be decoupled from the external supply line with a 2.2 microfarad capacitor.

Chip capacitors (surface-mount) are preferable because they exhibit lower inductance and require less total board space. They should be connected as in Figure 2.3. Leaded capacitors can also be used if the leads are kept as short as possible. Six leaded capacitors are required to match the effectiveness of one chip capacitor, but because only a limited number can fit around the 386 DX, the configuration in Figure 2.4 results.

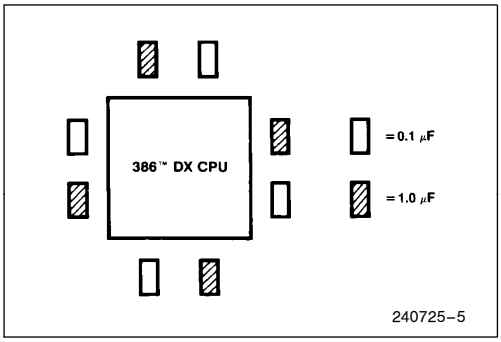


Figure 2-3. Decoupling Chip Capacitors

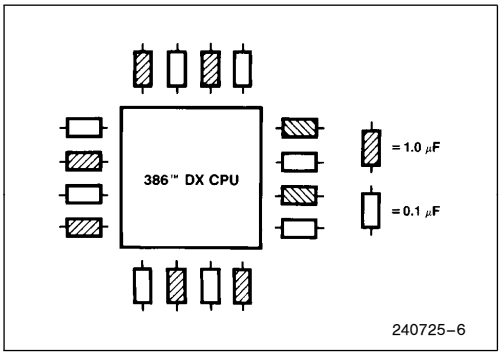


Figure 2-4. Decoupling Leaded Capacitors

HIGH FREQUENCY DESIGN CONSIDERATIONS

At high signal frequencies, the transmission line properties of signal paths in a circuit must be considered. Reflections, interference, and noise become significant in comparison to the high-frequency signals. They can cause false signal transitions, data errors, and input voltage level violations. These errors can be transient and therefore difficult to debug. In this section, some high-frequency design issues are discussed. Their effects and ways to minimize will be introduced in the next section.

REFLECTION AND LINE TERMINATION

Input voltage level violations are usually due to voltage spikes that raise input voltage levels above the maximum limit (overshoot) and below the minimum limit (undershoot). These voltage levels can cause excess current on input gates that results in permanent damage to the device. Even if no damage occurs, most devices are not guaranteed to function as specified if input voltage levels are exceeded.

Signal lines are terminated to minimize signal reflections and prevent overshoot and undershoot. If the round-trip signal path delay is greater than the rise time or fall time of the signal, terminate the line. If the line is not terminated, the signal reaches its high or low level before reflections have time to dissipate, and overshoot and undershoot occur. There are a few termination techniques that are used in different applications, these will be discussed in the next section.

INTERFERENCE

Interference is the result of electrical activity in one conductor causing transient voltages to appear in another conductor. It increases with frequency and closeness of the two conductors.

There are two types of interference to consider in high frequency circuits: electromagnetic interference (EMI) and electrostatic interference (ESI).

EMI (also called crosstalk) is caused by the magnetic field that exists around any current carrying conductor. The magnetic flux from one conductor can induce current in another conductor, resulting in transient voltage. Several precautions can minimize EMI.

Running a ground line between two adjacent lines wherever they traverse a long section of the circuit board. The ground line should be grounded at both ends.

Running ground line between the lines of an address bus or a data bus if either of the following conditions exist.

— The bus is on an external layer of the board.

— The bus is on an internal layer but not sandwiched between power and ground planes that are at most 10 mils away.

Avoiding closed loops in signal paths (see Figure 2.5). Closed loops cause excessive current and create inductive noise, especially in the circuitry enclosed by a loop.

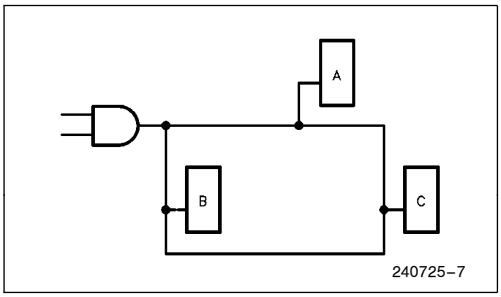


Figure 2-5. Avoid Closed-Loop Signal Paths

ESI is caused by the capacitive coupling of two adjacent conductors. The conductors act as the plates of a capacitor; a charge built up on one induces the opposite charge on the other.

The following steps reduce ESI:

Separating signal lines so that capacitive coupling becomes negligible.

Running a ground line between two lines to cancel the electrostatic fields.

LATCHUP

Latchup is a condition in a CMOS circuit in which V_{CC} becomes shorted to Vss. Intel's CHMOS IV process is immune to latchup under normal operating conditions. Latchup can be triggered when the voltage limits on I/O pins are exceeded, causing internal PN junctions to become forward biased. The following guidelines help prevent latchup:

Observing the maximum rating for input voltage on I/O pins.

Never applying power to an 386 DX CPU pin or a device connected to an 386 DX CPU pin before applying power to the 386 DX CPU itself.

Preventing overshoot and undershoot on I/O pins by adding line termination and by designing to reduce noise and reflection on signal lines.

THERMAL CHARACTERISTICS

The thermal specification for the 386 DX CPU defines the maximum case temperature. This section describes how to ensure that an 386 DX CPU system meets this specification.

Thermal specifications for the 386 DX CPU are designed to guarantee a tolerable temperature at the surface of the 386 DX CPU chip. This temperature (called the junction temperature) can be determined from external measurements using the known thermal characteristics of the package. Two equations for calculating junction temperature are as follows:

$T_j = T_a + (@j_a * PD)$ and

$T_j = T_c + (@j_c * PD)$

where:

T_j = Junction Temperature
$@j_a$ = Junction to ambient temperature coeff.
T_c = Case Temperature
T_a = Ambient Temperature
$@j_c$ = Junction to Case
PD = Power Dissapation temperature coeff.

Case temperature calculations offer several advantages over ambient temperature calculations.

Case temperature is easier to measure accurately than ambient temperature because the measurement is localized to a single point (top center of the package).

The worst-case junction temperature (T_j) is lower when calculated with case temperature for the following reasons:

— The junction-to-case thermal coefficient ($@j_c$) is lower than the junction-to-ambient thermal coefficient ($@j_a$); therefore, calculated junction temperature varies less with power dissipation (PD).

— $@j_c$ is not affected by airflow in the system; $@j_a$ varies with air flow.

With the case-temperature specification, the designer can either set the ambient temperature or use fans to control case temperature. Finned heat sinks or conductive cooling may also be used in environments where the use of fans is precluded. To approximate the case temperature for various environments, the two equations above should be combined by setting the junction temperature equal for both, resulting in this equation:

$$T_a = T_c - ((@j_a - @j_c) * PD)$$

The current data sheet should be consulted to determine the values of @ja (for the system's air flow) and ambient temperature that will yield the desired case temperature. Whatever the conditions are, the case temperature is easy to verify.

2.2 Transmission Line Effects

As a general rule, any interconnection is considered a transmission line when the time required for the signal to travel the length of the interconnection is greater than one-eighth of the signal rise time. (True K. M. , "Reflection: Computations and Waveforms, The Interface Handbook", Fairchild Corp, Mountain View, CA, 1975, Ch. 3). As frequencies increase, designers must account for the negative effects associated with transmission lines. The section that follows will attempt to describe these effects and provide some suggestions for minimizing their negative effect on the system.

Before describing each effect, it is important to know how to characterize a trace on different types of transmision lines. This includes knowing the characteristic impedance of a trace, Z_o, and the propagation delay for a given trace, t_{pd}. These parameters will be used in determining what effects must be accounted for and to select component values used in minimizing the effects.

TRANSMISSION LINES TYPES

Although many types of transmission lines (conductors) exist, those most commonly used on the printed circuit boards are microstrip lines, strip lines, printed circuit traces, side-by-side conductors and flat conductors.

MICRO STRIP LINES

The micro strip trace consists of a signal plane that is seperated from a ground plane by a dielectric as shown in Figure 2.6. G-10 fiber-glass epoxy, which is most common, has an $e_r = 5$ where e_r is the dielctric constant of the insulation. Let:

w = the width of the signal line (inches)
t = the thickness of copper
h = the height of dielectric for controlled impedance (inches)

The characteristic impedance Z_o, is a function of dielectric constant and the geometry of the board. This is given by:

$$Z_o = (87/(e_r + 1.41)^{1/2} \ln (5.98/0.8\ w + t)\ \Omega$$

where e_r is the relative dielectric constant of the board material.

The propagation delay (t_{pd}) associated with the trace is a function of the dielectric only.

$$t_{pd} = 1.017\ (0.475e_r + 0.67)^{1/2}\ \text{ns/ft}$$

STRIP LINES

A strip line is a strip conductor centered in a dielectric medium between two voltage planes. The characteristic impedance is given by:

$$Z_o = 60/(e_r)^{1/2} \ln (5.98b/(0.8W + t))\ \Omega$$

where b = distance between the planes for the controlled impedance as shown in Figure 2.10

The propagation delay is given by:

$$t_{pd} = 1.017\ (e_r)^{1/2}\ \text{ns/ft}$$

Typical values of the characteristic impedance and propagation delay of these types of lines are:

$$Z_O = 50\Omega$$

$$t_{pd} = 2\ \text{ns/ft (or 6 in/ns)}$$

2.3 Reflection

The first effect is reflection. As the name indicates it is the reflection of a signal as it propagates down the trace. The reflection results from a mismatch in impedance. The impedance of a transmission line is a function of the geometry of the line, its distance from the ground plane, and the loads long the line. Any discontinuity in the impedance will cause reflections.

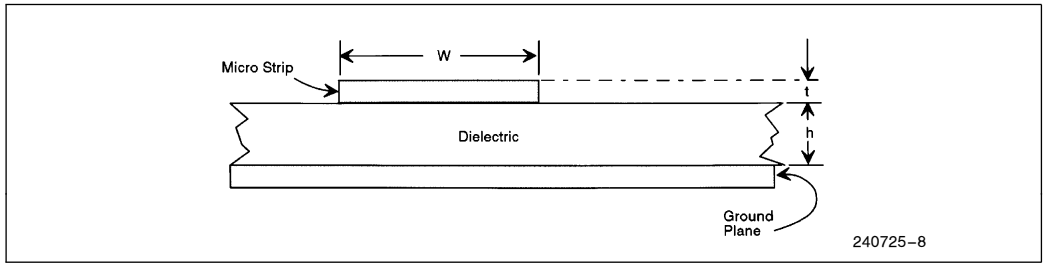


Figure 2-6. Micro Strip Lines

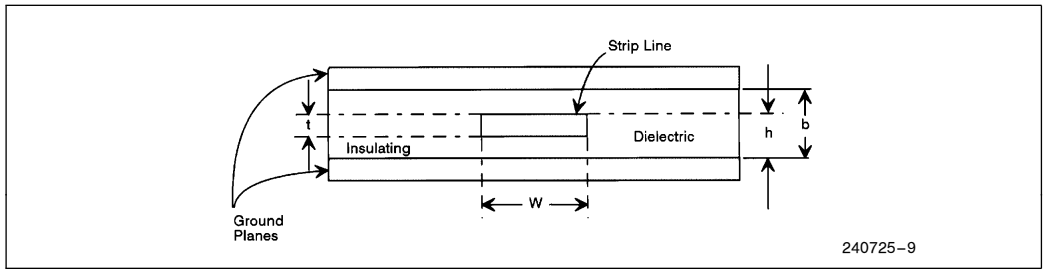


Figure 2-7. Strip Lines

Impedance mismatch occurs between the transmission line characteristic impedance and the input or output impedance of the devices that are connected to the line. The result is that the signals are reflected back and forth on the line. These reflections can attentuate or reinforce the signal depending upon the phase relationships. The results of these reflections include overshoot, undershoot, ringing and other undesirable effects.

At lower edge rates, the effects of these reflections are not severe. However at higher rates, the rise time of the signal is short with respect to the propagation delay. Thus it can cause problems as shown in Figure 2-8.

Overshoot occurs when the voltage level exceeds the maximum (upper) limit of the output voltage, while undershoot occurs when the level passes below the minimum (lower) limit. These conditions can cause excess current on the input gates which results in permanent damage to the device.

The amount of reflection voltage can be easily calculated. Figure 2-9 shows a system exhibiting reflections.

The magnitude of a reflection is usually represented in terms of a reflection coefficient. This is illustrated in the following equations:

$$T = v_r/v_i = \text{Reflected voltage/Incident voltage}$$

$$T_{load} = (Z_{load} - Z_O)/(Z_{load} + Z_O)$$

$$T_{source} = (Z_{source} - Z_O)/(Z_{source} + Z_O)$$

Reflections voltage V_r is given by V_i, the voltage incident at the point of the reflections, and the reflection coefficient.

The model transmission line can now be completed. In Figure 2-9, the voltage seen at point A is given by the following equation:

$$V_a = V_s * Z_O/(Z_O + Z_s)$$

This voltage V_a enters the transmission line at "A" and appears at "B" delayed by t_{pd}.

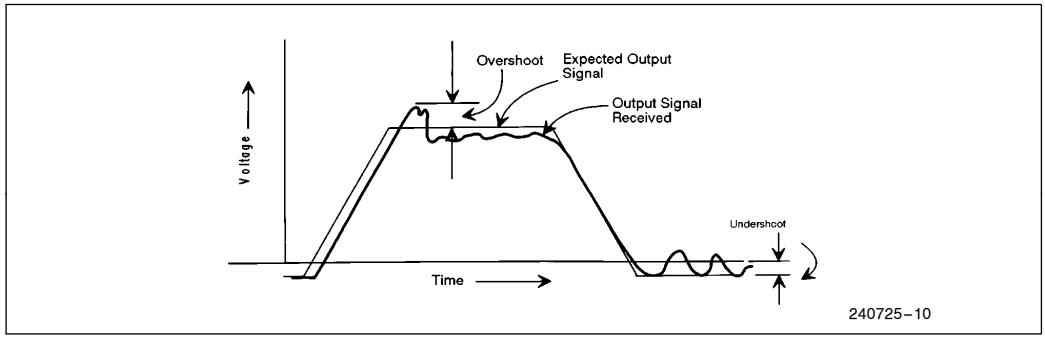


Figure 2-8. Overshoot and Undershoot Effects

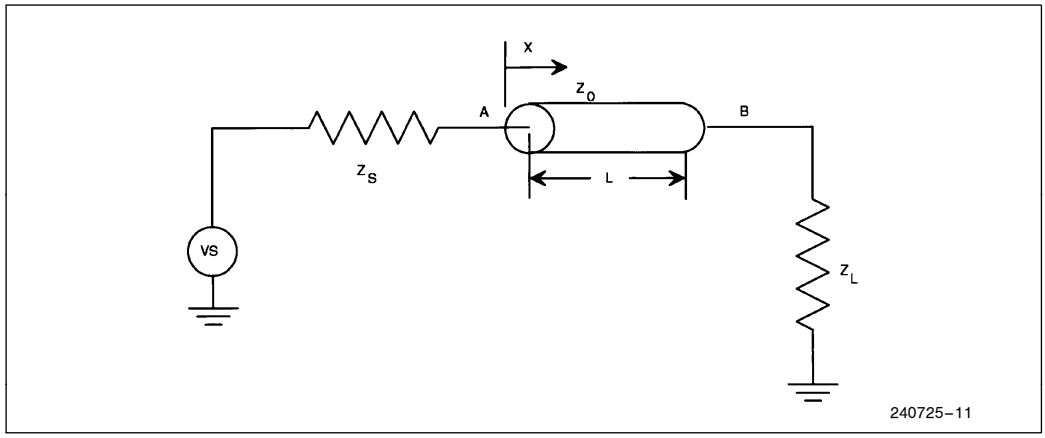


Figure 2-9. Loaded Transmission Line

$$V_b(t - x/t)\ H(t - x/v)$$

where x = distance along the transmission line from point "A" and H(t) is the unit step function. The waveform encounters the loads ZL, and this may cause reflection. The reflected wave enters the transmission line at "B" and appears at point "A" after time delay (t_{pd}):

$$V_{r1} = T_{load} * V_b$$

This phenomenon continues infinitely, but it is negligible after 3 or 4 reflections. Hence:

$$V_{r2} = T_{source} * V_{r1}$$

Each reflected waveform is treated as a seperate source that is independent of the reflection coefficient at that point and the incident waveform. Thus the waveform from any point and on the transmission line and at any given time is as follows:

$$\begin{aligned} V(x,t) = {} & (Z_O/(Z_O+Z_s))\ \{V_s(t-(x/v))\ H(t-(x/v)) + \\ & T_1\ [V_s(t-((2L-x)/v)\ H(t-(t-((2L-x)/v)))] + \\ & T_1T_s\ [V_s(t-((2L+x)/v)\ H(t-(t-((2L+x)/v)))] + \\ & T_{12}T_s\ [V_s(t-((4L-x)/v)\ H(t-(t-((4L-x)/v)))] + \\ & T_{12}T_s^2\ [V_s(t-((4L+x)/v)\ H(t-(t-((4L+x)/v)))] \\ & + \ldots\} \end{aligned}$$

Each reflection is added to the total voltage through the unit step function H(t). The above equation can be rewritten as follows:

$$\begin{aligned} V(x,t) = {} & (Z_O/(Z_O+Z_s))\ \{V_s(t-(t-t_{pd}x)\ H(t-t_{pd}x) + \\ & T_1\ [V_s(t-t_{pd}(2L-x))\ H(t-t_{pd}(2L-x))] + \\ & T_1T_s\ [V_s(t-t_{pd}(2L+x))\ H(t-t_{pd}(2L+x))] + \ldots\} \end{aligned}$$

Impedance discontinuity problems are managed by imposing limits and control during the routing phase of the design. Design rules must be observed to control trace geometry, including specification of the trace width and spacing for each layer. This is very important because it ensures the traces are smooth and constant without sharp turns.

HOW TO MINIMIZE

There are several techniques which can be employed to further minimize the effects caused by an impedance mismatch during the layout process:

1. Impedance Matching
2. Daisy Chaining
3. Avoid 90° Corners
4. Minimize the Number of Vias

IMPEDANCE MATCHING

Impedance matching is the process of matching the impedance of the the source or load to the impedance of the trace. This matching is accomplished using a technique called termination. Termination makes the effective source or load impedance, seen by the trace, to be approximately equal to the characteristic impedance of the trace. Before terminating a line one must determine if termination is required. This is done by a simple calculation. If the propagation delay down a trace from source to destination is greater than or equal to one-third the signals rise time, termination is needed. (i. e. $T_{pd} \geq \frac{1}{3}\ t_r$). The rise time is the 0%-100% rise time specified for the source. If this value is specified for 10%-90% or 20%-80%, it must be scaled by multiplying the specified value by 1.25 or 1.67, respectively. The propagation delay is caculated by multiplying the trace propagation delay, t_{pd}, descibed earlier by the trace length.

Once it is determined that termination is needed, use the equation described earlier to calculate the trace's characteristic impedance. The specification sheets for the load can be consulted to determine the load impedance, Z_L. These values are needed to select the component values used to terminate.

The next chore is selecting the type of termination to use. In this section we will examine 4 different techniques and point out the advantages and disadvantages. Figure 2.10 shows the four types of termination and the corresponding component values.

Parallel termination, shown in Figure 2-10(a), is a good technique to maintain the waveform. The waveform at the load is a perfect image of the waveform at the source. In addition there is no added propagation delay associated with this technique. The disadvantage of this technique is that it requires a fair amount of additional power and it is not suggested for characteristic impedances of less than 100 ohms because of the large d.c. current required.

Thevenin termination, shown in Figure 2-10(b), is another option. This technique also requires a large amount of power, but does not have the restrictions for characteristic impedance. This technique is very good at removing overshoot and undershoot while not adding any additional delay. Another advantage is that the trace can be biased toward Vcc or GND by simpling selecting the appropriate resistor values. This can help maintain fast edges on important signal transitions.

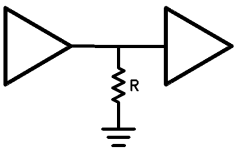
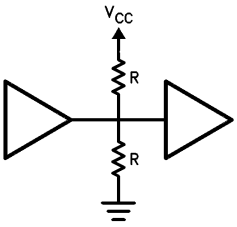
Name	Circuitry	Advantages	Disadvantages
Parallel	 $R = Z_O$	Waveform at receiver is almost perfect image of input Bipolar/Advanced CMOS No added T_{PD}	High power dissipation $Z_O \geq 100\Omega$, else D.C. current limit
Thevenin	 $R = 2\,Z_O$	Good overshoot and undershoot suppression Bipolar or Bipolar/CMOS systems No added T_{PD}	High power dissipation

Figure 2-10(a). Termination Techniques

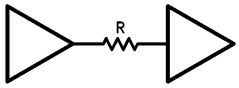
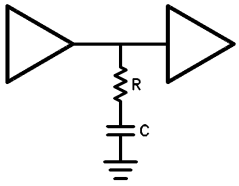
Name	Circuitry	Advantages	Disadvantages
Series	 $R = Z_O - Z_{OUT}$	Low power consumption CMOS—CMOS Systems Easy to adjust signal amplitude to match switching threshold	Added T_{PD}
A.C.	 $R = Z_O$, C = 200 pF–500 pF	Low—medium power dissipation (capacitor blocks D.C. coupling of signal) No added delays High-speed CMOS families	Two added components

Figure 2-10(b). Termination Techniques

Series termination, shown in Figure 2-10(b), is a very easy technique of matching impedance. It only requires on resistor and very little additional power is required. In addition the resistor value can be selected to provide constructive or destructive reflections and thus alter the signal amplitude to match the switching threshold. The major disadvantage of this technique is the added delay it introduces.

The fourth technique is A.C. termination, shown in Figure 2-10(b). It requires a small amount of additional power, this is decreased over parallel termination by the introduction of the capacitor, and adds no extra delay to the path. The major disadvantage is that it requires two extra components.

After examing the systems needs and selecting a termination technique, the impedance values determined earlier, Z_O and Z_L, can be used to determine the component values to implement the termination. These values should be seen as a starting point and may be altered to remove a specific problem experienced on a signal or to bias signals in an appropriate fashion.

DAISY CHAINING

Another technique of minimizing reflections is to daisy-chain signals, shown in Figure 2-11. This means to run a single trace from a source and to distribute the loads along this trace. The alternative is to run multiple traces from the source to each load. Each trace will have reflections of its own and these will be transmitted down the other traces once they have returned to the source. To manage such a system separate termination would be required for each branch. To eliminate these multiple terminators from T-connections, high frequency designs are routed as daisy chains.

Because each gate provides its own impedance load along the chain, it is necessary to distribute these loads evenly along the length of the chain. Hence, the impedance along the chain will change in a series of steps and is easier to match. The overall speed of this line is faster and predictable. Also all loads should be placed at equal distances (regular intervals).

90 DEGREE ANGLES

Eliminating 90° angles also minimizes reflections. It is much more desirable to use 45° or 135° angles as shown in Figure 2-12.

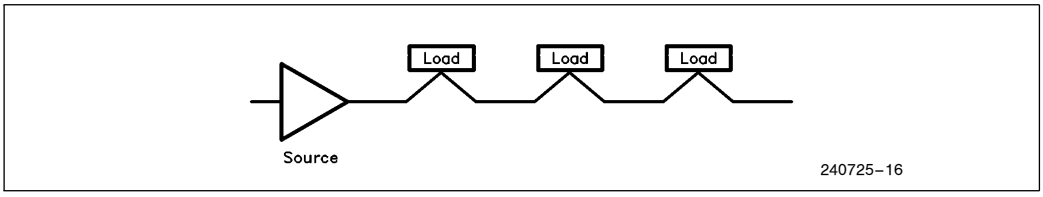


Figure 2-11. Daisy Chaining

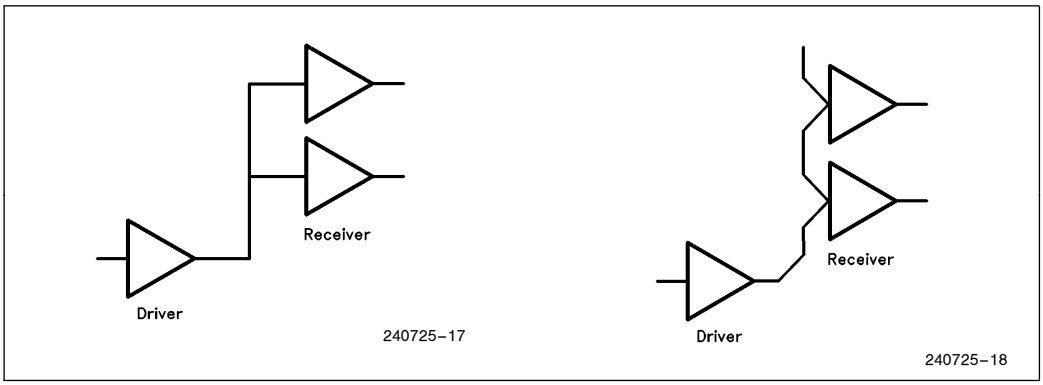


Figure 2-12. Avoiding 90 Degree Angles

VIAS (FEED THROUGH CONNECTIONS)

Another impedance source that degrades high frequency circuit performance is the via. Expert layout techniques can reduce vias to avoid reflection sites on PCBs.

Following these guidelines will not guarantee elimination of all reflections, but they will minimize the number and size.

2.4 Cross Talk

Cross talk is another negative effect of transmission lines. It is a problem at high frequencies because, as operating frequency increase, the signal wavelength become comparable to the length of the interconnections on the PC board. In general, interference such as cross talk, occurs when electrical activity in one conductor causes a transient voltage to appear in another conductor. Main factors that increase interference in any circuit are:

1. Variation of current and voltage in the lines causes frequency interference. This interence increases with increase in frequency.
2. Coupling occurs when conductors are in close proximity.

Cross talk is the phenomenom of a signal in one trace producing a similar signal in an adjacent trace. It may not be a carbon copy of the original signal. It may only be occasional noise that corrupts the integrity of the second signal. The easiest way to minimize crosstalk is to eliminate or at least minimize the number of parallel traces. Parallel traces can be on a single layer or on adjacent signal layers.

There are three ways that parallel traces can couple and thereby produce a signal or at least influence the signal on a second trace. These methods of coupling are inductive, radiative, and capacitive. Inductive coupling is where the two traces act as inductors. The field produced by a signal in one trace induces a current in the second trace. Radiative coupling occurs when the two parallel traces act as a dipole, an antenna. One radiates a signal and the other receives it, thus corupting the signal already present on the trace. The final method is capacitive coupling. Two parallel traces separated by a dielectric act as a capacitor. If both traces are in a high state and one transitions to a low. The capacitor will try to maintain the high and thus cause a slow transition time on the second trace. These effects can be minimized by reducing the number of parallel traces.

HOW TO MINIMIZE

When laying out a board for an high speed 386 DX based system, several guidelines should be followed to minimize crosstalk. Some of them are as follows:

1. To reduce crosstalk, it is necesary to minimize the common impedance paths.
2. Run a ground line between two adjacent lines. The lines should be grounded at both ends.
3. Seperate the address and data busses by a ground line. This technique may however be expensive due to large number of address and data lines.
4. Remove closed loop signal paths which create inductive noise.
5. Capacitive coupling can be reduced by reducing the number of parellel traces. Parallel traces can be minimized by insuring that signals on adjacent signal layers run orthogonal, perpendicular. Ground planes or traces can be inserted to provide shielding. A ground plane between signal layers eliminates any coupling that could occur. On a single trace, a ground trace can be run between traces to prevent coupling.

In some instances it is necessary to run traces parallel to each other. In these cases try to make the distance as short as possible and choose signals in which the transition time is not as critical so that the coupling effects do not produce problems. In addition the coupling can be minimized by increasing the spacing between parallel traces.

2.5 Skew

Skew is another effect of transmission lines. This is very important in a synchronous system. Long traces add propagation delay. A longer trace or a load placed further down a trace will experience more delay than a short trace or loads very close to the source. This must be taken into account when doing the worst case timing analysis. In a system where events must occur synchronous to a clock signal, it is important to make sure the signal is available to all input a sufficient amount of time prior to the corresponding clock edge. When performing the component placement this is one of the considerations that must be accounted for.

These guidelines have always been recommended for board design; however, they are much more important at higher frequencies. At the slower frequencies designers could ignore these practices occassionally and not experience difficulties. This is not the case at higher frequencies.

2.6 DC Loading

To maintain proper logic levels, all digital signal outputs have a maximum load, they are capable of driving. DC loading is the constant current required by an input in either the high or the low state. It limits the ability of a device driving the bus to maintain proper logic levels. For a 386 DX based system, a careful analysis must be performed to ensure that in a worst case situation no loading limits are exceeded. Even if a bus is loaded slightly beyond its worst case limit, it might cause problems if a batch of parts whose input loading is close to maximum is encountered. Proper logic level will then fail to be maintained and unreliable operation may result. Marginal loading problems are particularly insidious, since the effect is often erratic operation and non repetitive errors that are extremely difficult to track down. For both the high and low logic levels, the sum of the currents required by all the inputs and the leakage currents of all outputs (drivers) on the bus must be added together. This sum must be less than the output capability of the weakest driver. Since the 386 DX is a CHMOS device having negligible dc loading, the main contributors to dc loading will be the TTL devices.

2.7 AC Loading

The AC or capacitive loading is caused by the input capacitance of each device and limits the speed at which a device driving a bus signal can change the state from high to low or low to high. Designers of microprocessor systems have traditionally calculated load capacitance of their systems by determining the number of devices and their individual capacitance loading attached to a signal plus the amount of trace capacitance. Typically, the trace capacitance was a set "lumped" number of pf (i.e. 2 pf to 3 pf per inch) when it is thought of at all. This lumped method is a general rule-of-thumb which generates a good first pass approximation. For low frequency designs, the lumped method works since system and component margins are large enough to cover any minor differences due to the approximation.

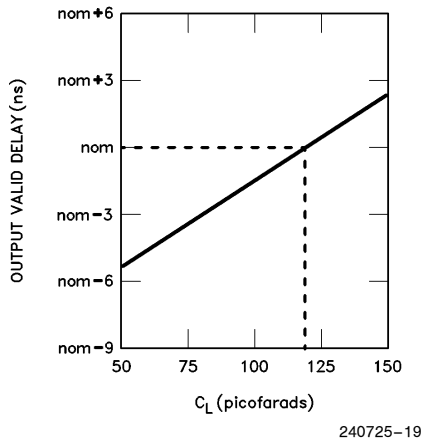
For high frequency designs, the component and system margins are no longer available to the designer. With less than 1 ns of margin, even the amount of trace capacitance can make a circuit path critical.

A more accurate calculation of capacitive loading can be derived by modeling the device loads and system traces as a series of Transmission Lines Theory. Transmission Line Theory provides a more accurate picture of system loading in high frequency systems. In addition, it allows new factors such as inductance and the effect of reflections upon the quality of the signal waveform to be factored into consideration.

2.8 Derating Curve and Its Effects:

A derating curve is a graph that plots the output buffer against the capacitive load. The curve is used to analyze a signal delay without necessitating a simulation every time the processor's loading changes. This graph assumes the lumped capacitance model to calculate the total capacitance. The delay in the graph should be added to the specified AC timing value for the device that is driving the load. The derating curve is different for different devices because each device has different output buffers.

A derating curve is generated by tying the chip's output buffers to a range of capacitors. The voltage and resistance values chosen for the output buffers are at the highest specified temperature and are rising (worst case) values. The value of the capacitors centres around the AC timing values for the chip. For 33 MHz and above, this is 50 pF. Since the AC timing specifications are measured for a signal reaching 1.5 V. A curve is then drawn from kthe range of time and capacitance values, with 50 pF representing the average and with nominal or zero derating. These curves are valid only for 50 pF–150 pF load range. Beyond this range the output buffers are not characterized. The the derating curve for the 386 DX are shown in 2-13. These curves use the lumped capacitance model for circuit capacitance measurements and must be modified slightly when doing worst-case calculations that involve transmission line effects.



NOTE:
This graph will not be linear outside of the C_L range shown.

Figure 2-13. Typical Output Valid Delay Versus Load Capacitance at Maximum Operating Temperature (C_L = 120 pF)

2.9 High Speed Clock Circuits

For performance at high frequencies, the clock signal (CLK2) for the 386 DX CPU must be free of noise and within the specifcations listed in the 386 DX CPU data sheet. Achieving the proper clock routing around a 33 MHz printed circuit board is delicate because a myriad of problems, some of them subtle, can arise design guidelines are not followed. For example, fast clock edges cause reflections from high impedance terminations. These reflections can cause significant signal degradation in systems operating at 33 MHz clock rates. This section covers some design guidelines which should be observed to properly lay out the clock lines for efficient 386 DX operation.

- Since the rise/fall time of the clock signal is typically in the range of 2-4 ns, the reflections at this speed could result in undesirable noise and unacceptable signal degradation. The degree of reflections depends on the impedance of the traces of the clock connections. These reflections can be optimized by terminating the CLK2 output with proper terminations and by keeping length of the traces as short as possible. The preferred method is to connect all of the loads via a single trace as shown in Figure 2-14, thus avoiding the extra stubs associated with each load. The loads should be as close to one another as possible. Multiple clock sources should be for distributed loads.
- A less desirable method is the star connection layout in which the clock traces branch to the load as closely as posssible (Figure 2-15). In this layout, the stubs should be kept as short as possible. The maximum allowable length of the traces depends upon the frequency and the total fanout, but the length of all the traces in the star connection should be equal. Lengths of less than one inch are recommended. In this method the CLK2 signal is terminated by a series resistor. The resistor value is calculated by measuring the total capacitive load on the CLK2 signaI and referring to Figure 2-16. If the total capacitive load is less than 80 pF, the user should add capacitors to make up the diference. Because of the high frequency of CLK2, the terminating resistor must have low inductance; carbon resistors are recommended.
- Use an oscilloscope to compare the CLK2 waveform with those in Figure 2-17.

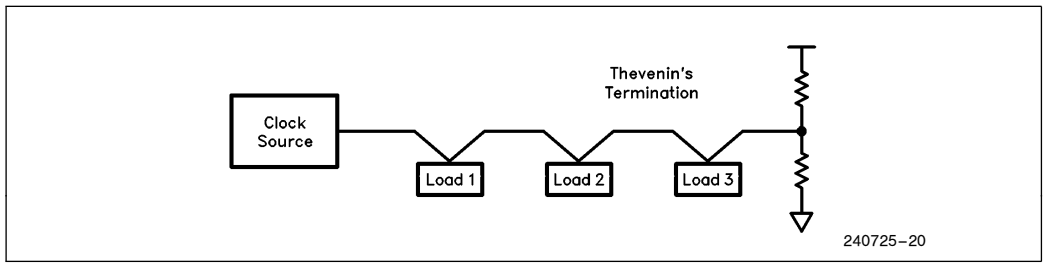


Figure 2-14. Clock Routing

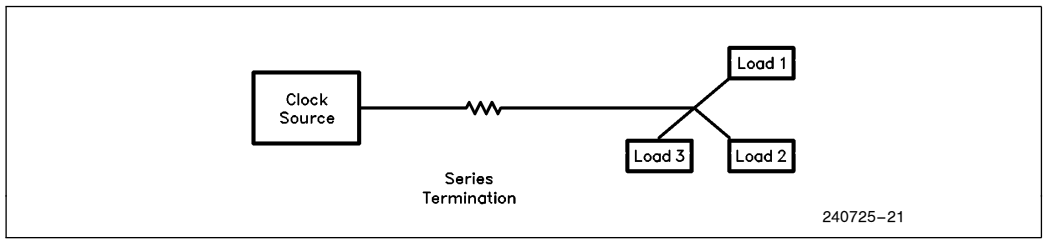
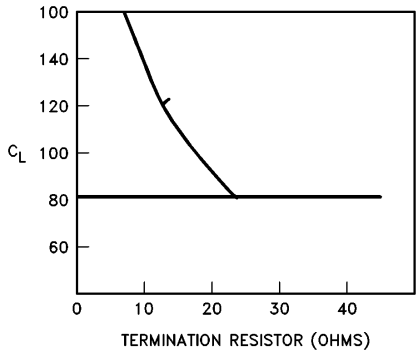


Figure 2-15. Star Connection



- $C_L = C_{IN}$ (386) + C_{IN} (387) + C_{IN} (PALs) + . . . + C_{BOARD}.
  C_{BOARD} is calculated from layout and board parameters; thickness, dielectric constant, distance to ground/V_{CC} planes.
- Termination resistor must be low inductance type. Recommend carbon filled type.

Figure 2-16. CLK2 Series Termination

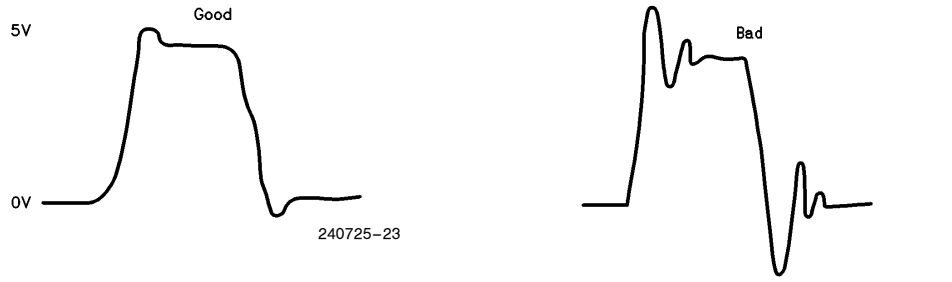


Figure 2-17. CLK2 Waveforms

SECTION III. DESIGN EXAMPLE

At higher processor speeds the window of time available to perform specific tasks become very small. This window can be equated to multiples of the CLK2 period. Within this time signals must be supplied from a source and reach a destination in time to meet any setup requirements. At 16 MHz the CLK2 period is 31 ns. At 33 MHz it shrinks to half this value, 15 ns. The longer time allowed the use of slower logic families and the delays associated with longer traces. As the window decreases system designers have to practice more care in the selection of logic families and in the choices made for component placement and signal routing on PCBs. This section attempts to list the signal paths whose worst case timing analysis results in very small margins and therefore require closer attention from designers to guarantee that all a. c. timing specifications are met.

This section also includes a sample design based on 33 MHz version of the 386 DX. It should not be taken as a recommended design. The circuit is used only to highlight the design considerations for high speed systems.

3.1 System Architecture

Figure 3.1 shows the system block diagram. It has four major subsystems.

1) CPU subsystem

2) DRAM subsystem

3) Cache subsystem

4) ROM and I/O subsystem

The system has 1 megabytes of Page-Mode DRAMS (60 ns RAS access time), 128 kilobytes of EPROMS (200 ns access time), an 8259A-2, and an 82510. The cache subsystem is optional. Schematics and PAL codes are given in appendix A and B respectively.

3.2 CPU Subsystem

The CPU subsystem consists of the 386 DX microprocessor, a clock and reset circuitry, and bus control logic. Clean and proper clock is very important in the designs at high frequencies.

RESET STATE MACHINE

This state machine is used to generate three control signals, namely RESET, REFREQ and CLK. The CLK signal is half of the CPU clock, CLK2 and is used mainly in I/O and EPROM subsystem.

RESET is generated through the input from RESET triggering circuitry (as shown in the CPU schematic). The min RESET Setup and Hold time for operation at 33 MHz are 5 ns and 2 ns respectively.

A 61.44 KHz clock is used to produce a synchronous refresh request (REFREQ) signal for the DRAM controller, which employ a transparent, distributed, DRAM refresh technique that allows the processor and cache to run while the refresh cycle is in progress.

3.3 DRAM Subsystem

An non-interleaved DRAM system is used in the sample board, which simplifies the design. Since the board provide caching, the performance of DRAM subsystem is outweighed by the simplicity and economy of the design. It employs a transparent, distributed, DRAM refresh technique which allows the processor and cache to run while the refresh cycle is in progress. It uses the 3-state capability of the 16R8-7 and the 74ACT258 to multiplex the refresh address. A further consideration is the choice of DRAM devices. If one uses a memory device such as the AAA2801 (which supports a CAS# before RAS# refresh and provides an internal refresh counter) further simplifications can be made in both the circuitry and the control logic.

DRAM CONTROL STATE MACHINE

The state machine is implemented with three 16R8-type E-speed PALs (see page 4 of the schematics). E-speed PALs must be used since the CLK2 frequency, 66.67 MHz, is higher than the maximum clock frequency of the D-speed PALs.

In order to generate DRAM control signals with smallest delay from the CLK2 edges, all state machines are implemented as Moore machines. The state machines flip-flops generate most of the DRAM control signals directly. This is an expensive design approach in terms of hardware but allows signal timings and skews to be fine tuned.

DRAM CYCLES—NO CACHE CONFIGURATION

Pages C-1 through C-4 show examples of DRAM cycles. In order to hide the DRAM page hit-or-miss decision time, the DRAM controller always tries to put the 386 DX in pipelined mode. The first read cycle requires only two wait states since RAS# has been precharged (see page C-1). The second cycle takes only two clock cycles. The second cycle is a pipelined, page-hit read cycle, which is the best case. The third cycle is a pipelined, page-hit write cycle. This cycle requires one wait state. DRAMs capture data at the falling edge of CAS# during Early Write cycles. The 386 DX drives

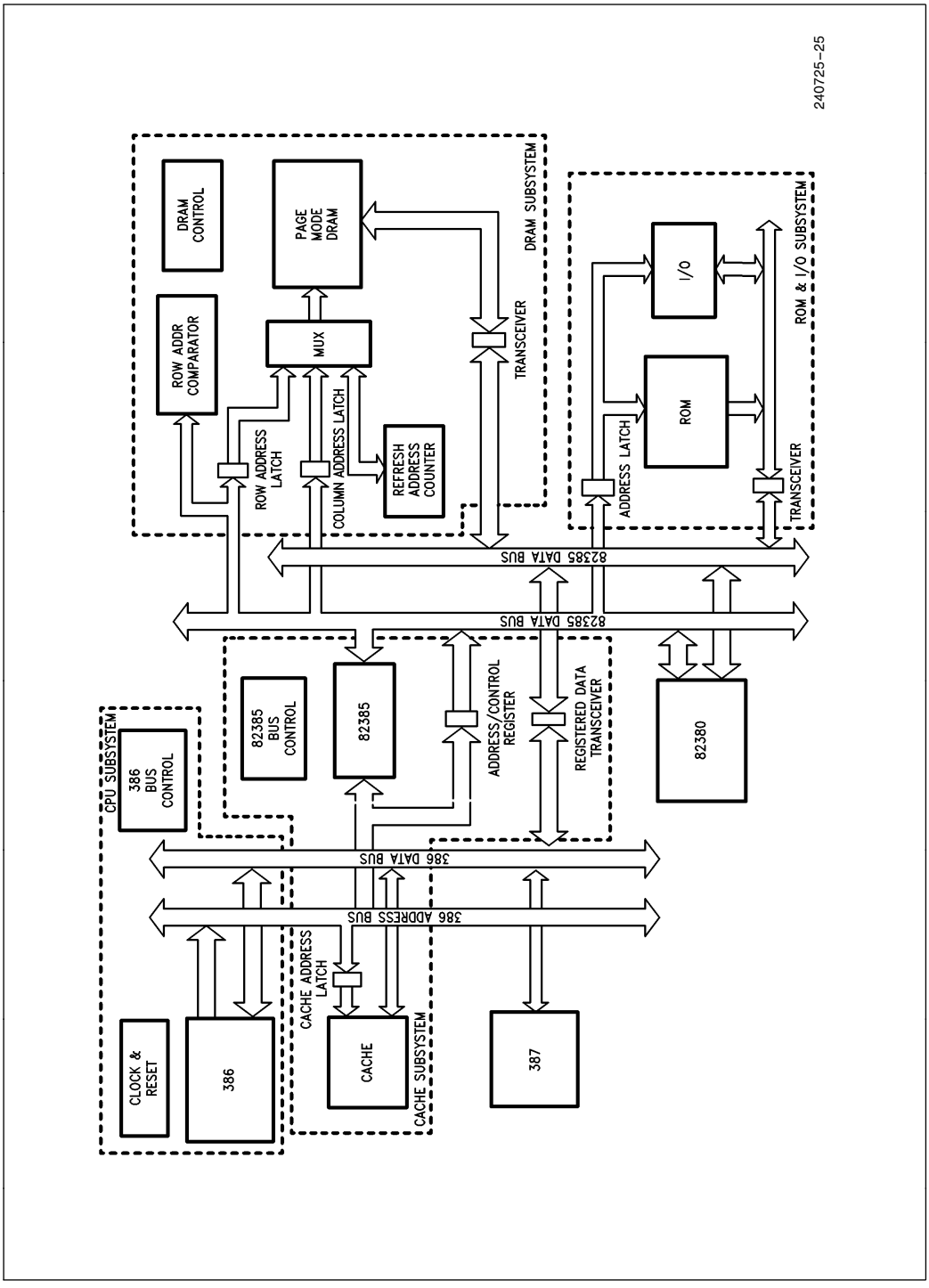


Figure 3-1. Block Diagram

valid write data at the rising edge in the middle of Tip (edge C) with a max prop delay of 24 ns (T12 max). This means that the CAS# is generated after the rising edge in the middle of the second T2p (edge A). CAS# is, therefore, generated at the end of RAS# hold time with respect to CAS# (if the next cycle is a page miss, RAS# will go inactive at the end of the current write cycle), and so on.

The fifth cycle is a page miss, which is actually detected at the end of the fourth cycle (page C-2). Since the DRAM controller must wait for minimum RAS# precharge time, the fifth cycle requires three wait states. The sixth cycle is also a page miss. This cycle, however, requires only two wait states because the miss was detected early enough in the previous cycle to have RAS# precharged by the end of the T1p. If the seventh cycle is another page miss, it will require three wait states.

The eighth cycle is ended with T2i. Consequently, the ninth cycle must wait for minimum RAS# precharge time and requires three wait states.

A DRAM refresh cycle is shown on page C-4. The DRAM address multiplexer output is disabled, and the refresh address counter output is enabled. The cycle does a RAS# only refresh cycle where only RAS# is asserted with a proper refresh address. After the refresh cycle is completed, a read cycle which has been suspended due to the refresh is resumed.

STATE DIAGRAMS

Pages B-1 through B-11 show state diagrams of the DRAM controller. The precharge state machine on page B-2 measures the required RAS# precharge time and CAS#-to-RAS# precharge time. The CAS#-READY# state machine on page B-2 implements a pin strap option of having or not having the 82385. For no cache configuration, the Cache variable must be forced low.

TIMING CALCULATIONS

Timing equations are described on pages D-1 and D-2. Their corresponding results are given on pages D-3 through D-7.

Capacitive load on the 386 DX address bus was assumed to be less than 85 pF. Capacitive load on the DRAM address bus was calculated to be less than 22 pF.

3.4 CACHE Subsystem

At 33 MHz DRAM speeds are not fast enough to design zero wait state memory systems. A cache can be used to take advantage of the higher performance available from the higher speed 386 DX microprocessors. The cache takes advantage of the faster SRAM while keeping system costs down by using the cheaper but slower DRAMs.

Details of the cache subsystem are shown on Figure 3.2 and 3.3. The 82385 address and data busses are interfaced to the 386 DX address and data busses via 74AS574s and 74AS646s. Static RAMs (20 ns access time) are used for the cache memory.

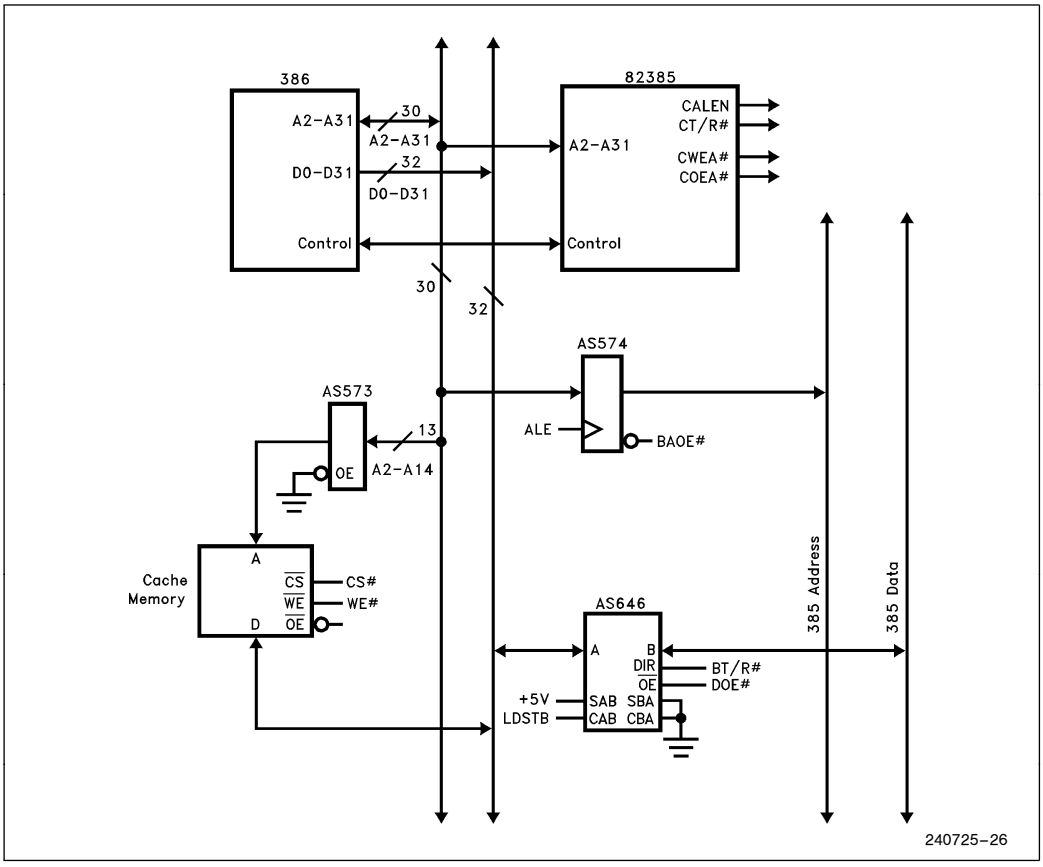


Figure 3-2. Block Diagram of Cache Subsystem

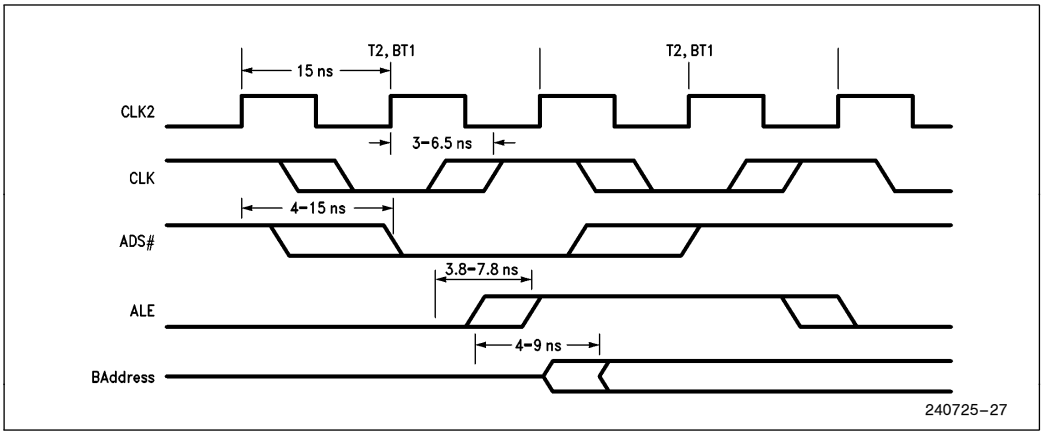


Figure 3-3. Address Valid Delay for Cache Subsystem

In selecting SRAM there are several types one can choose to use. Some SRAM require a latch for the address and a transceiver for the data. Others have an OE#, output enable, signal and incorporate the transceiver on chip. The third type is called integrated SRAM and these contain both the latch and the transceiver on chip. However, there are two timing paths that dictate the speed selection within each type. Figure 3.4 shows a typical system configuration using each type.

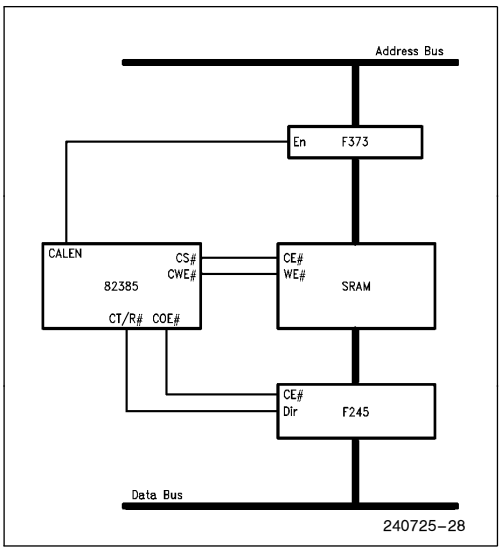


Figure 3.4(a) SRAM w/o OE#

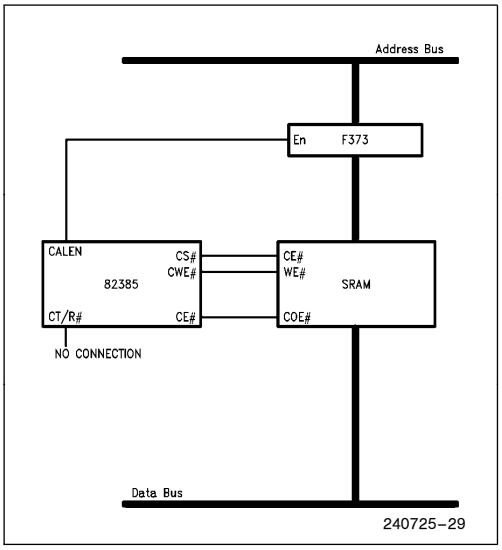


Figure 3.4(b) SRAM with OE# Control

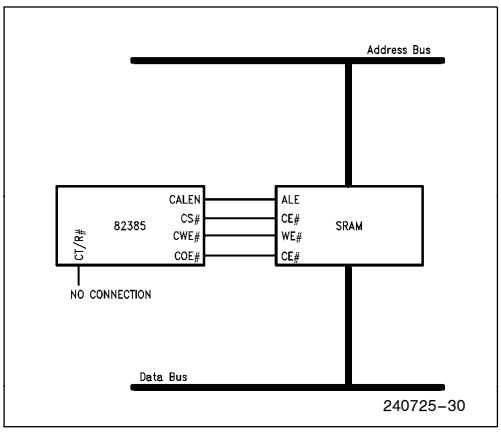


Figure 3-4. (c) Integrated SRAM

The critical times for the SRAM are the SRAM OE# to data delay and the SRAM address to data delay. The following analysis applies to SRAMs with an OE# signal as shown in Figure 3.4b. First examine the path of OE# to data. This path must be completed within 2 CLK periods. The COE# signal from the 385 Cache Controller must be valid and the SRAM must drive data onto the data bus so that the data setup time of the 386 DX CPU is met.

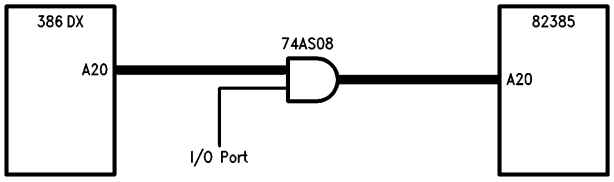
2 X CLK2 period - t_{25b} 82385 COE# valid delay (max) - SRAM access time (OE# to data) - t_{21} 386 DX data setup $\geq$ 0

Using the specified values from the data sheets reveals that the SRAM must have an OE# to data delay of 10ns or less. The other path is for the address to become available and data to reach the 386 DX CPU. This path has 4 CLK2 periods. The 385 Cache Controller must supply the CALEN signal to pass the address to the SRAM and then the SRAM must drive the data on the data bus so that the data setup time is met on the 386 DX CPU.

4 X CLK2 period - t_{21b} 82385 CALEN valid delay (max) - t_{pd} (x373 latch) - SRAM access time (address to data) - t_{21} 386 DX data setup $\geq$ 0

Once again using the data sheet the access time can be determined. Depending on the type of transparent latch the SRAM needs an address to data access time of 20ns or 25ns. If an F series 373 is used the faster 20ns SRAM must be used, but if an FCT373a or PCT373a is used the 25ns SRAM is sufficient.

The A_{20} path is another path with a small margin. The reason is the AND gate that many designers insert to provide 1MB wraparound of address in real mode. Figure 3.5 shows the circuit block diagram. A_{20} must leave the 386 DX and reach the 385 Cache Controller within 2 CLK2 periods.

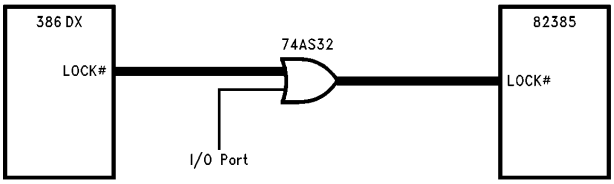


2 X CLK2 period - t_6 386 DX address valid delay (max) - t_{tp} AND prop. delay - t_{7d} 82385 address setup $\geq$ 0

Figure 3-5. Critical Timing A20

To meet this timing the propagation delay of the AND gate must be less than 6ns. This dictates the use of a 74AS08 gate or faster device.

Analysis of the LOCK# path also shows a small margin. The reason is the OR gate that many designers insert to disable the LOCK# signal to the 385 Cache Controller. This allows locked accesses to be cached. Figure 3.6 shows the circuit block diagram. LOCK# must leave the 386 DX and reach the 385 Cache Controller within 2 CLK2 periods.



2 X CLK2 period - t_8 386 DX LOCK# valid delay (max) - t_{tp} OR prop. delay - t_{7b} 82385 LOCK# setup $\geq$ 0

Figure 3-6. Critical Timing Lock#

To meet this timing the propagation delay of the OR gate must be less than 6ns. This dictates the use of a 74AS32 gate or faster device.

The final path examined here is the NA# path. Recently designers have selected to use an I/O port and an OR gate to disable pipelining selectively. Figure 3.7 shows the circuit block diagram. NA# must leave the 386 DX and reach the 385 Cache Controller within 2 CLK2 periods.

Using the specified values in the appropriate data sheets results in the need for the propagation delay of the OR gate must be no greater than 5.8ns. This dictates the use of a 74AS32 gate or faster device.

This list is not meant to be exhaustive. It is merely meant to highlight a few of the critical timings. Each designer should perform a thorough timing analysis of the system they are designing to verify that all timing requirements are met.

In addition to the specified timing parameters in the data sheets, designers should account for propagation delays introduced by the trace and by capacitive loading. The propagation delay added by the trace is explained in the section on transmission line effects and supplies an equation to determine the amount of delay.

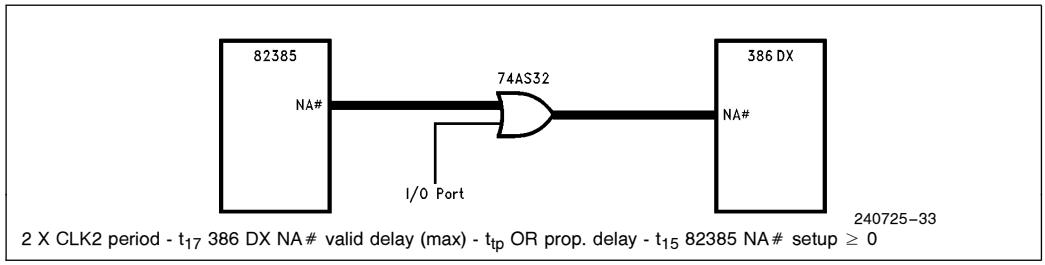


Figure 3-7. Critical Timing NA#

Another factor that becomes more important at higher frequencies is loading. DC loading and especially capacitive loading must be considered during the design stage. If the board is to be assembled and tested in stages, then the DC loads should be considered for all configurations of the board. Most termination techniques require additional current. If a board has a marginal loading situation, one is limited in one's choices of termination techniques. If a capacitive loading problem exists, the timing situations can become extremely difficult at higher frequencies. If timing is critical, do not overload the capacitance at which a device was tested. If a device is overloaded, derating must be taken into consideration.

Capacitive loading also introduces a delay on signals. Many components including the 386 DX include a capacitive derating curve in the data sheet. To use the curve in the 386 DX data sheet, the capacitive load must be calculated. This is done by summing the input capacitances of all devices driven by a given output from the 386 Microprocessor. Find this value on the X-axis of the derating curve in the data sheet and move up till the derating curve is intersected. Then move at a right angle to the left until intersecting the Y-axis. A value of nom+ or nom− something is found. This is the nominal value plus or minus some amount. The nominal value is the value found in the data sheet. Add the offset from the curve to this nominal value to get the resulting delay corresponding to the capacitive loading in the system. Note: The trace capacitance was not included in this calculation. It is accounted for in the trace propagation delay mentioned earlier.

DRAM CYCLES WITH 82385 ENABLED

When the 82385 is enabled (the CACHE variable of the state machine on page B-2 is forced High), the DRAM controller inserts one extra wait state in all read cycles. This extra time is needed to allow a cache update cycle to occur after each cache read miss cycle. During a cache update cycle, the read data from DRAMs must propagate through the 74AS646 and the 74F245 (optional) and must be ready for a SRAM write cycle with enough setup time.

Timing diagrams on pages C-5 through C-9 show cache and DRAM cycles.

TIMING CALCULATIONS

Timing equations are found on pages D-8 and D-9. Only tCAS, tRAC, tCAC, tAA, tPC, and tCAP are different in this configuration. Actual values for DRAM timings are found on page D-10.

3.5 I/O - EPROM Subsystem

A block diagram of the I/O-EPROM subsystem is shown on Figure 3.8. This subsystem has separate address and data busses. The address bus is 14 bits wide, and the data bus is 16 bits wide.

The bus controller is designed with B-speed PALs which are clocked by the CLK# signal (Figure 3.8). There are a few unique design issues in this scheme.

As shown on Figure 3.10, ADS# is now an asynchronous signal for the state machine. It is impossible for the state machine to capture valid ADS# without re-synchronization of the signal. To guarantee recognition of valid ADS#, two D flip-flop is clocked by CLK# and provides a synchronous ADS# (or Latched ADS#) which is in phase with the state machine.

The second issue is its asynchronous nature of the state machine output signal. With the state machine running almost asynchronously to CLK2 (B PALs also have a long clock-to-output propagation delay), signals generated by the state machine must be re-synchronized before they are returned to the 386 DX. Signals that go to I/O devices and EPROMs need no re-synchronization since these devices are asynchronous. Signals which require re-synchronization are BS16# and DEN#. Each rising edge of DEN# is synchronized to CLK2 by a J-K flip-flop as shown on Figure 3.9. This is important to avoid bus contention after an I/O or EPROM read-cycle. BS16# is synchronized to CLK2 by D flip-flops.

EPROM and I/O cycle timings are shown on pages C-10 through C-13. The worst case is a write cycle to the 82510 and may require as many as 14 wait states.

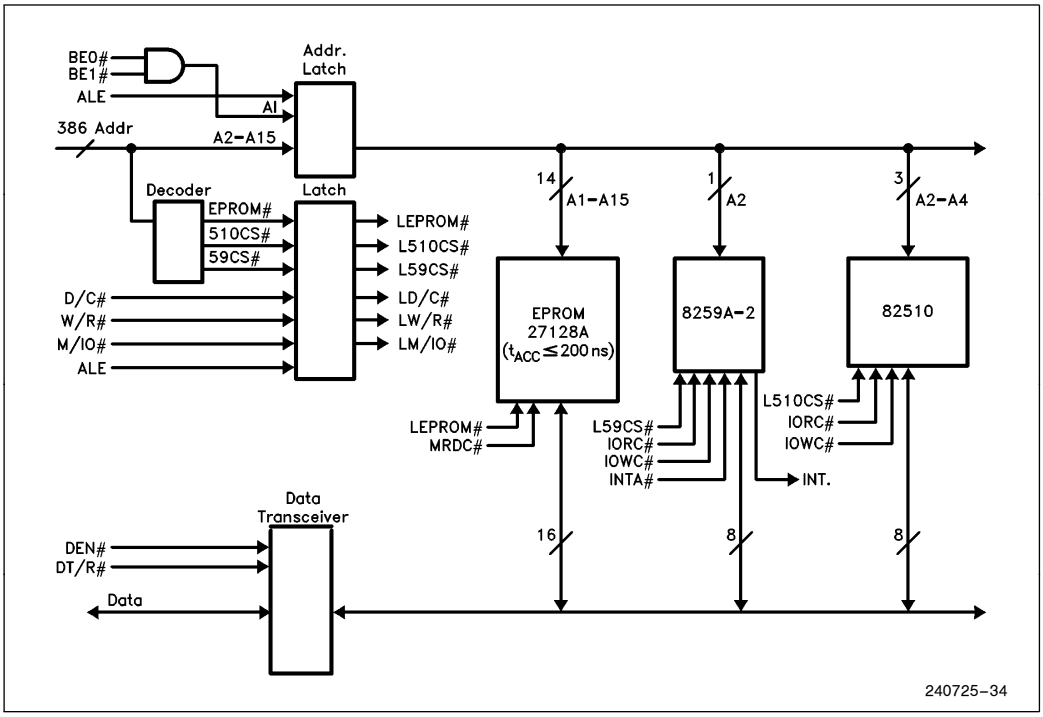
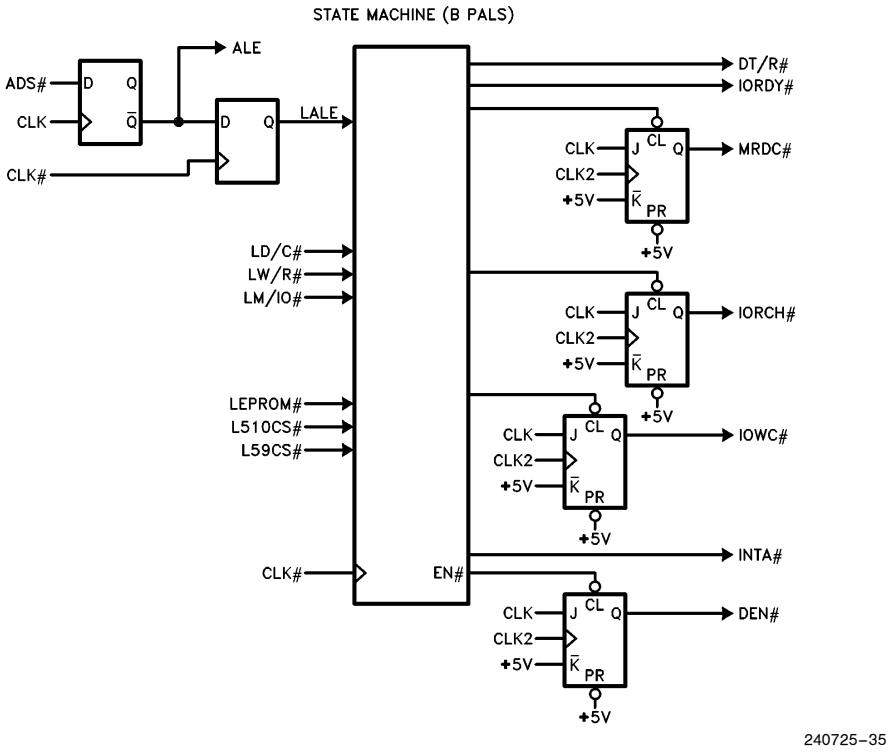


Figure 3-8. Block Diagram of I/O, EPROM Subsystem



NOTE:
Create BS16# for 386 Using DEN# and EPROM# (Synch to CLK)

Figure 3-9. Control Logic for I/O, EPROM Subsystem

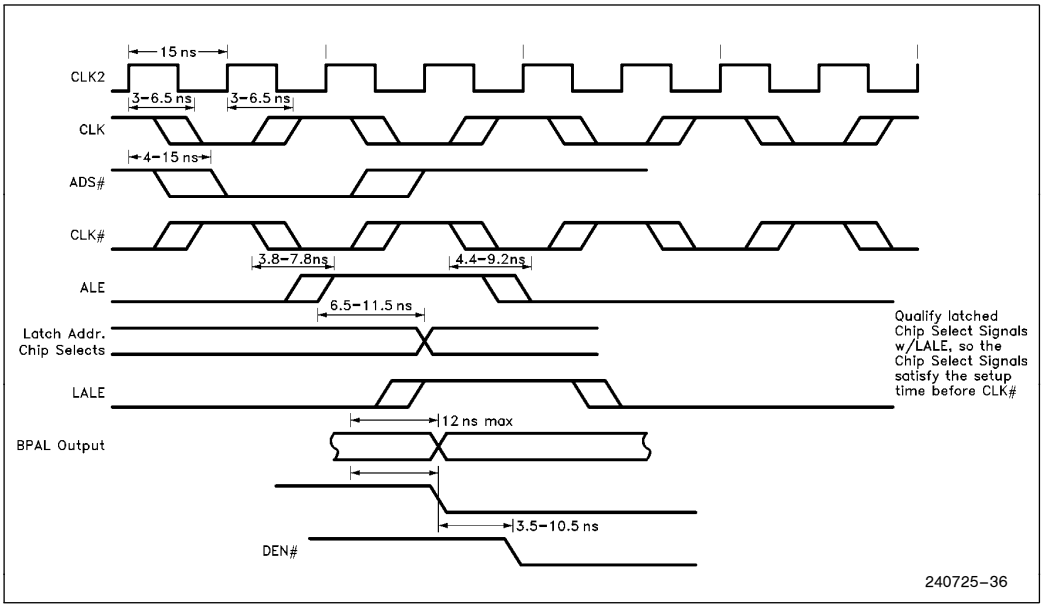
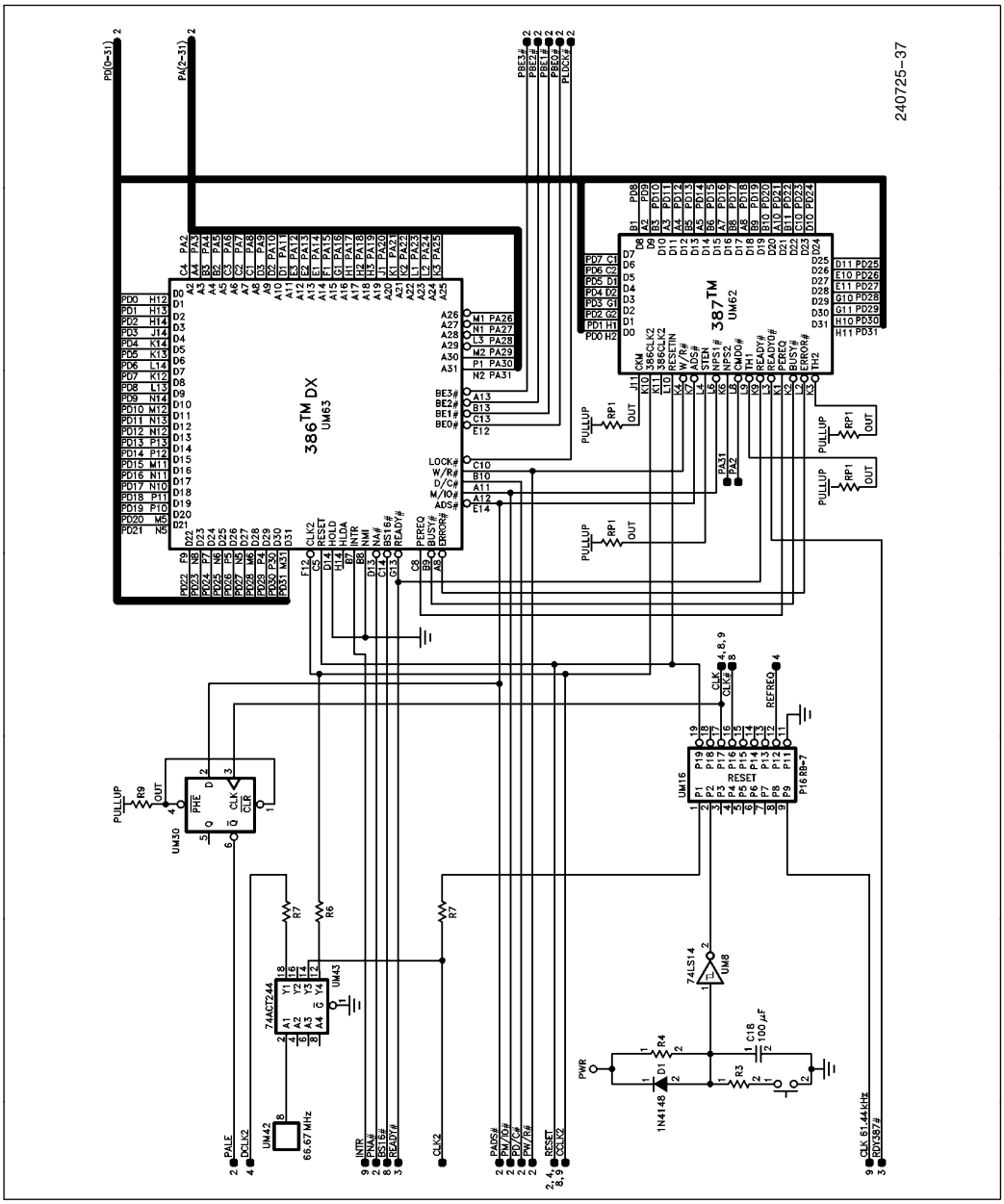
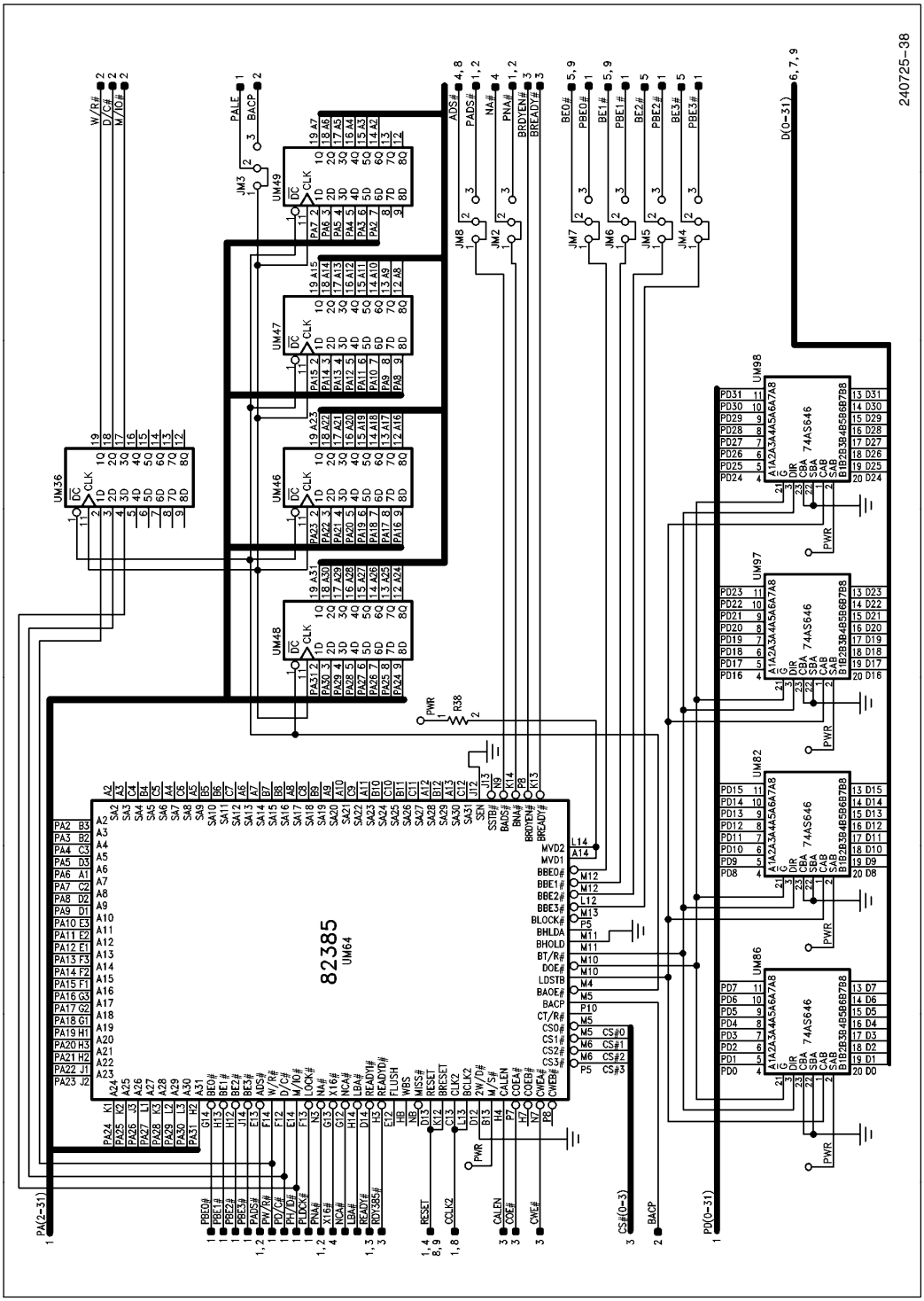
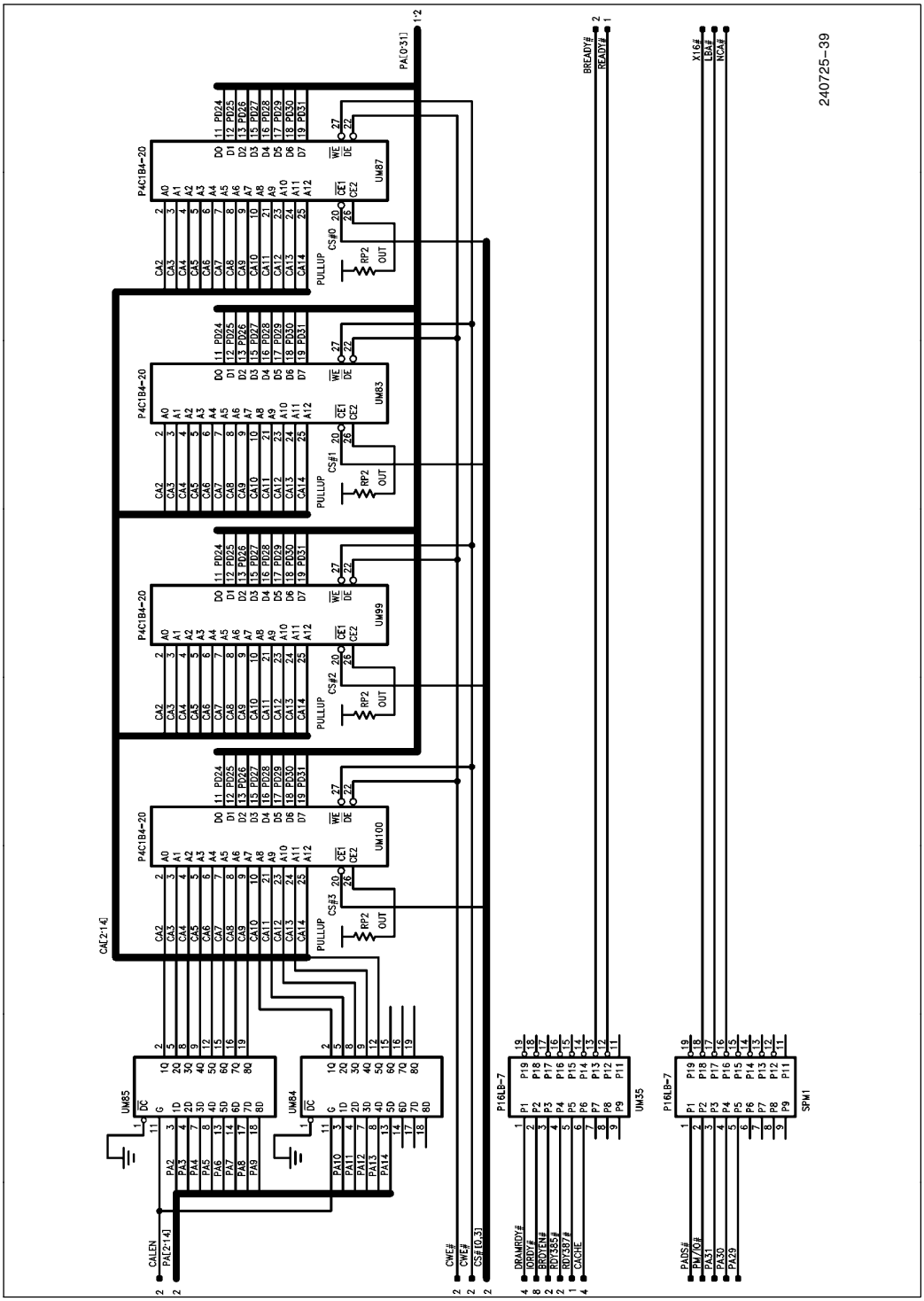


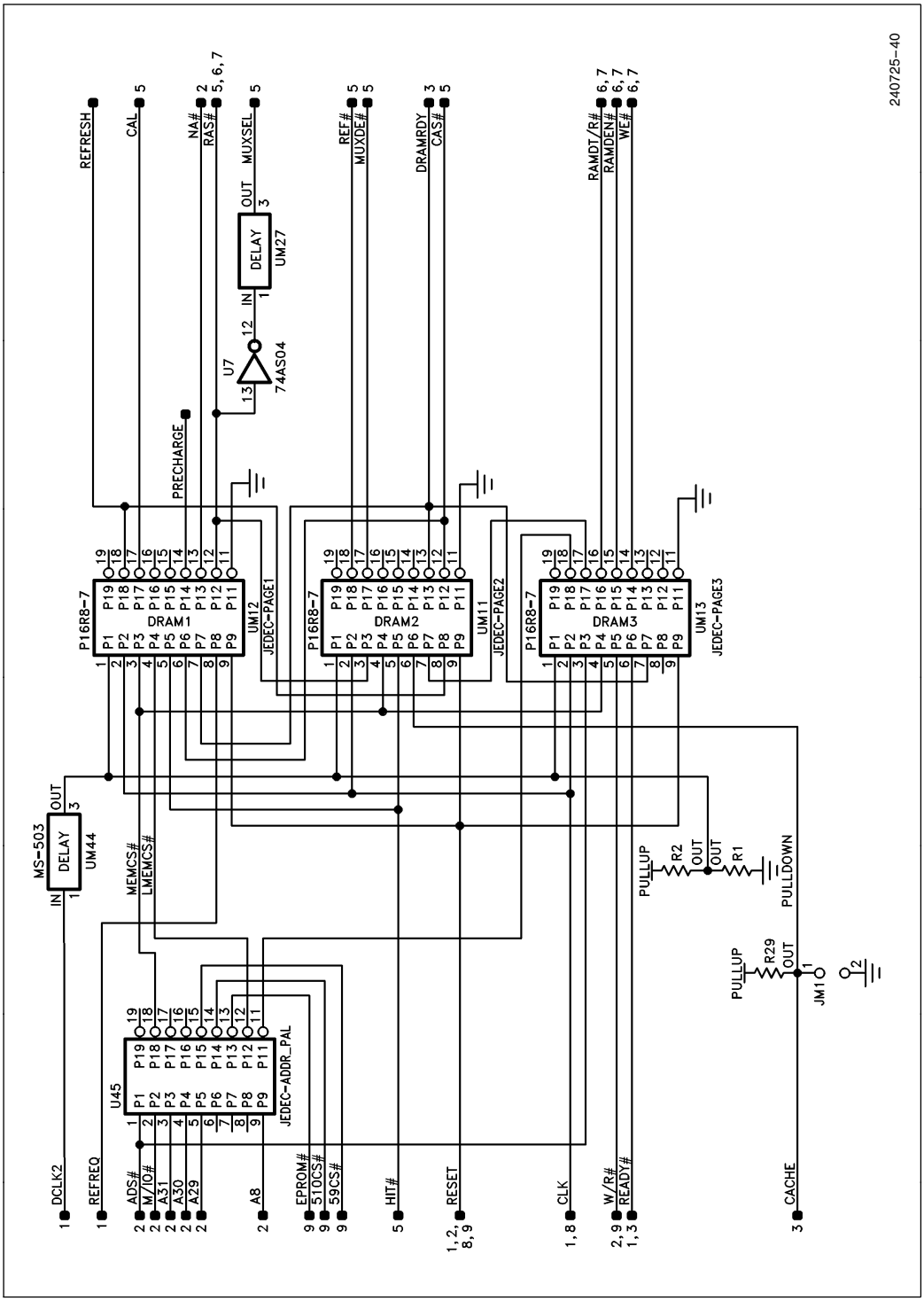
Figure 3-10. ADS# Should Be Synchronized to Guarantee Recognition

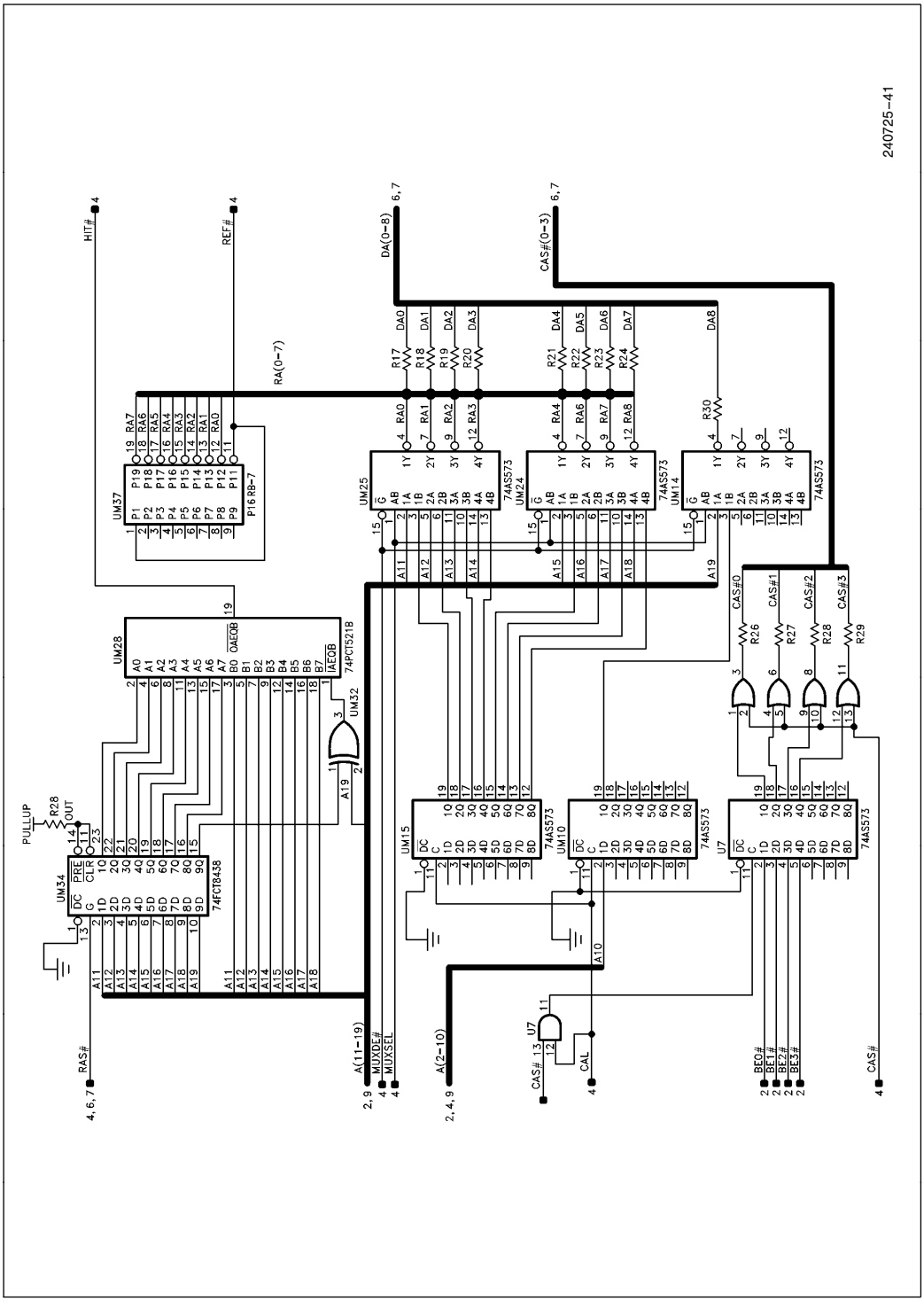
APPENDIX A
SCHEMATICS

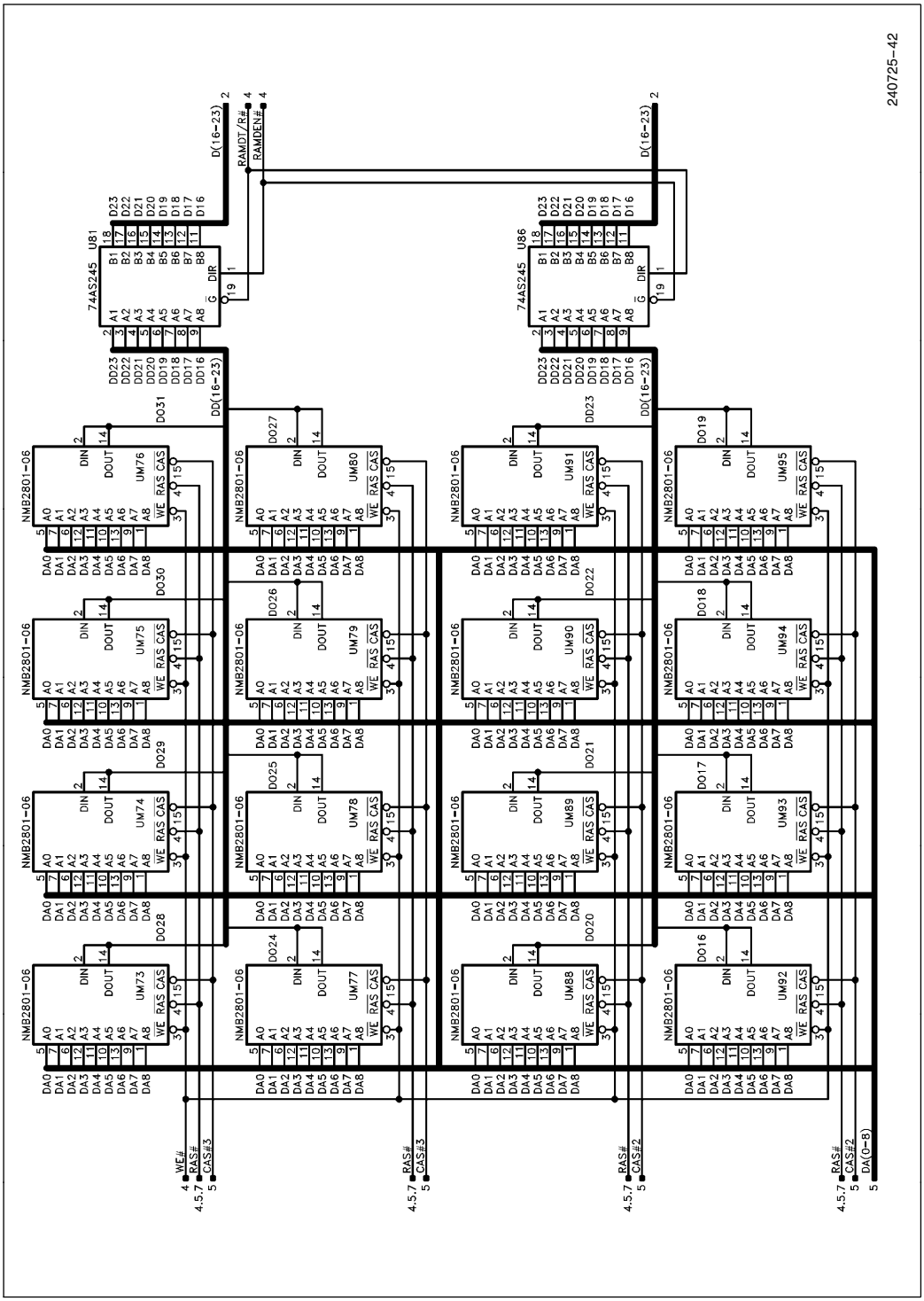


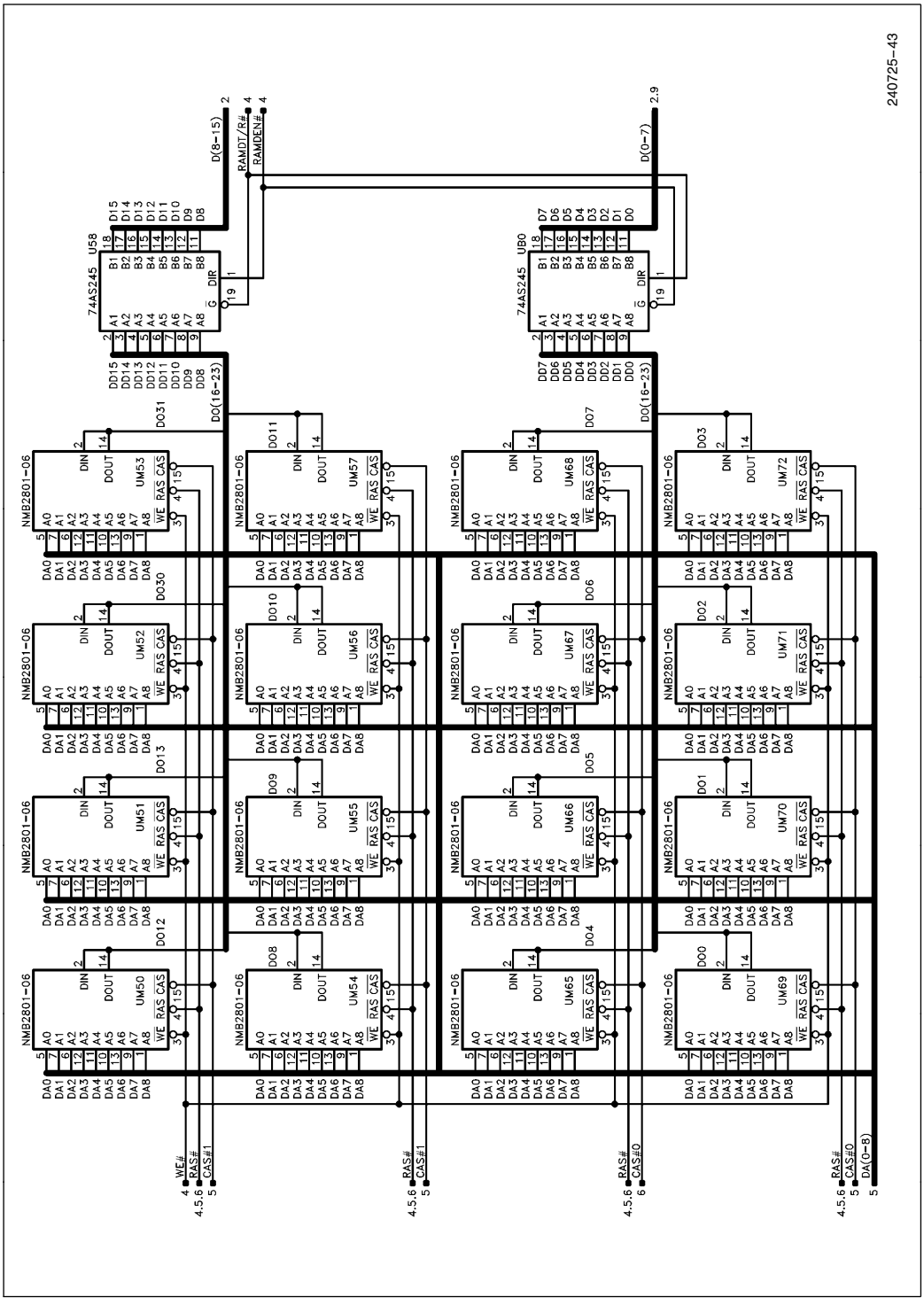

240725-38
82385
UM64
UM36
UM46
UM48
UM447
UM449
UM82
UM86
UM97
UM98
74AS646
PA(2-31)
PD(0-31)
D(0-31)
CS#(0-3)
BACP
PWR
RESET
CLK2
BCLK2
CALEN
COE#
CWE#
W/R#
D/C#
M/IO#
PALE
ADS#
NA#
PNA#
BRDYEN#
BREADY#
PBE0#
PBE1#
PBE2#
PBE3#
BE0#
BE1#
BE2#
BE3#
CS#0
CS#1
CS#2
CS#3

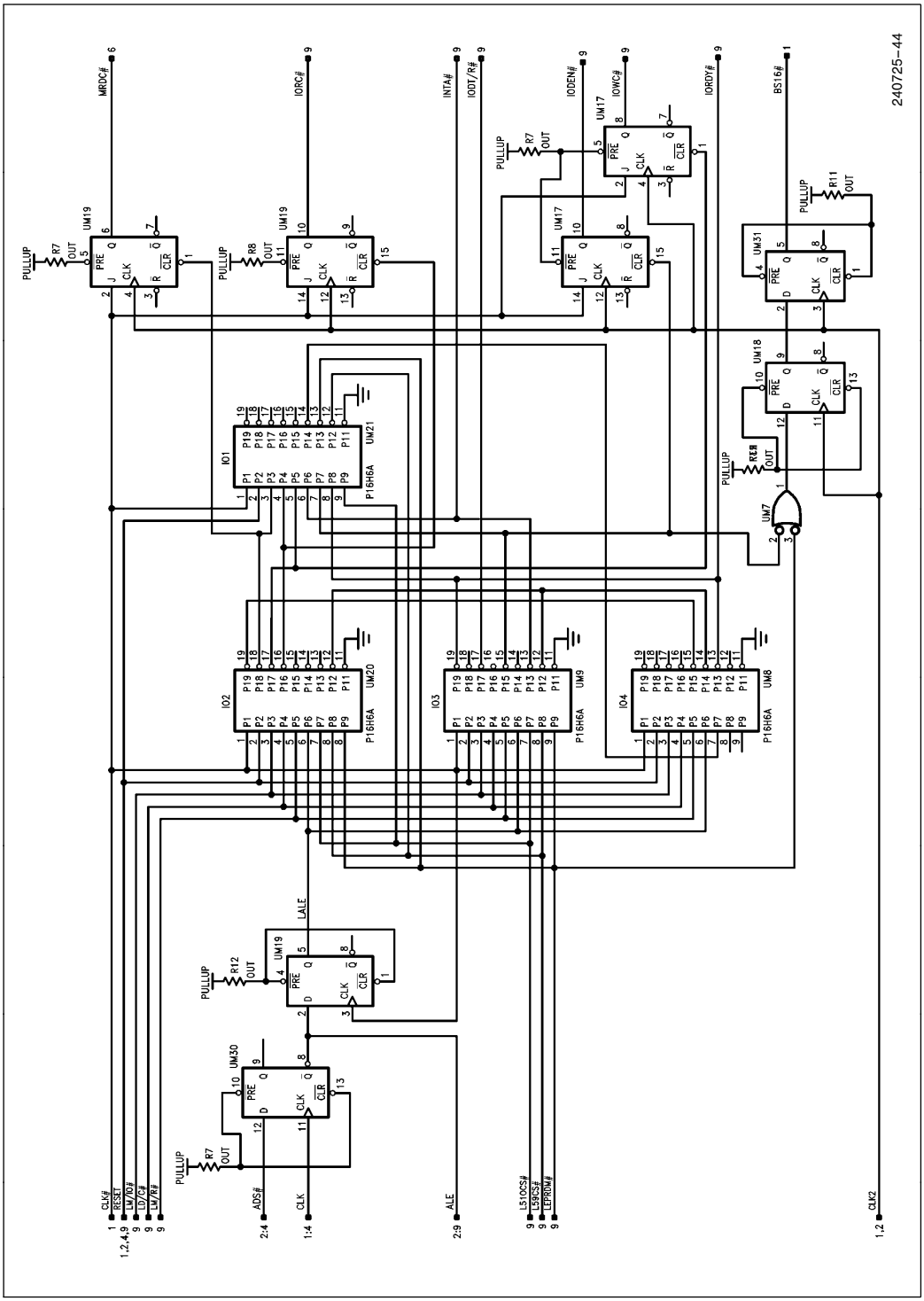

240725-39

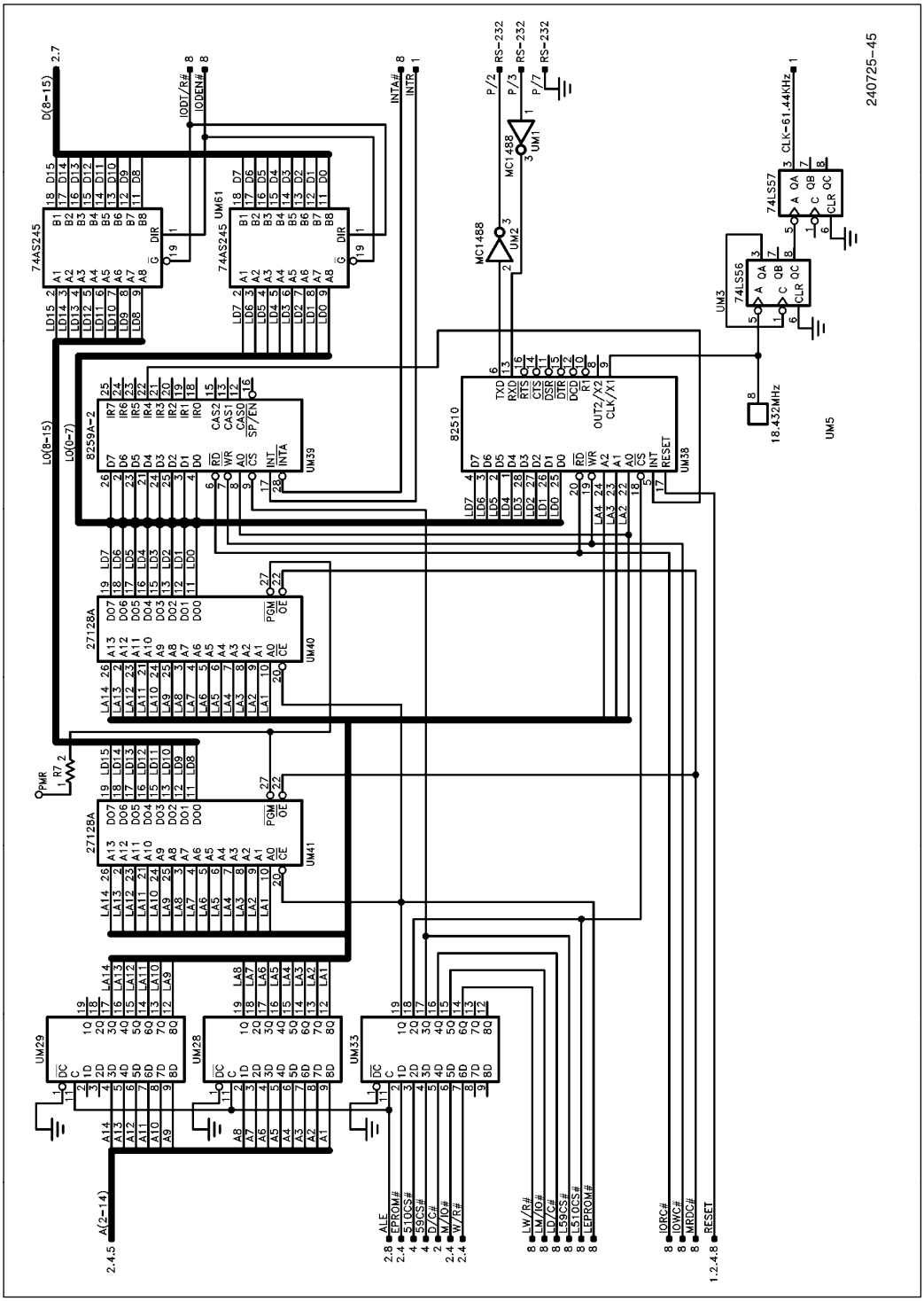

240725-40
REFRESH
CAL
NA#
RAS#
MUXSEL
REF#
MUXDE#
DRAMRDY
CAS#
RAMDT/R#
RAMDEN#
WE#
PRECHARGE
U7
74AS04
DELAY
UM27
P16R8-7
DRAM1
DRAM2
DRAM3
UM12
UM11
UM13
JEDEC-PAGE1
JEDEC-PAGE2
JEDEC-PAGE3
MS-503
UM44
MEMCS#
LMEMCS#
U45
JEDEC-ADDR_PAL
PULLUP
R2
R1
PULLDOWN
R29
JM10
DCLK2
REFREQ
ADS#
M/IO#
A31
A30
A29
A8
EPROM#
510CS#
59CS#
HIT#
RESET
CLK
W/R#
READY#
CACHE

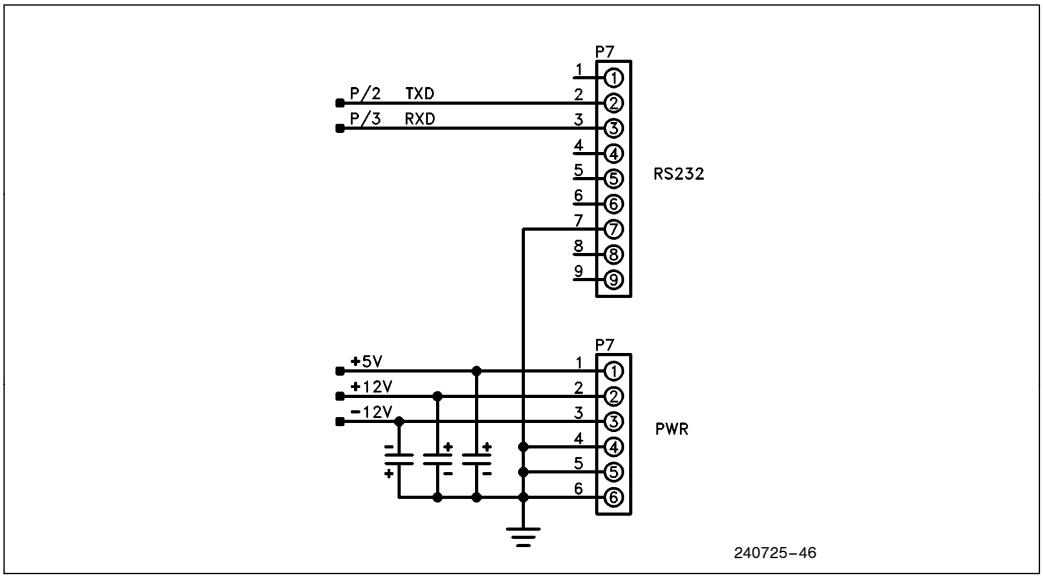

240725-41
HIT#
REF#
DA(0-8)
CAS#(0-3)
RA(0-7)
PULLUP
R28
UM34
74FCT8438
UM28
74PCT5218
UM32
UM37
P16 RB-7
UM25
UM24
UM14
74AS573
R17
R18
R19
R20
R21
R22
R23
R24
R30
DA0
DA1
DA2
DA3
DA4
DA5
DA6
DA7
DA8
CAS#0
CAS#1
CAS#2
CAS#3
R26
R27
R28
R29
UM15
UM10
U7
A(11-19)
A(2-10)
MUXDE#
MUXSEL
RAS#
CAS#
CAL
BE0#
BE1#
BE2#
BE3#

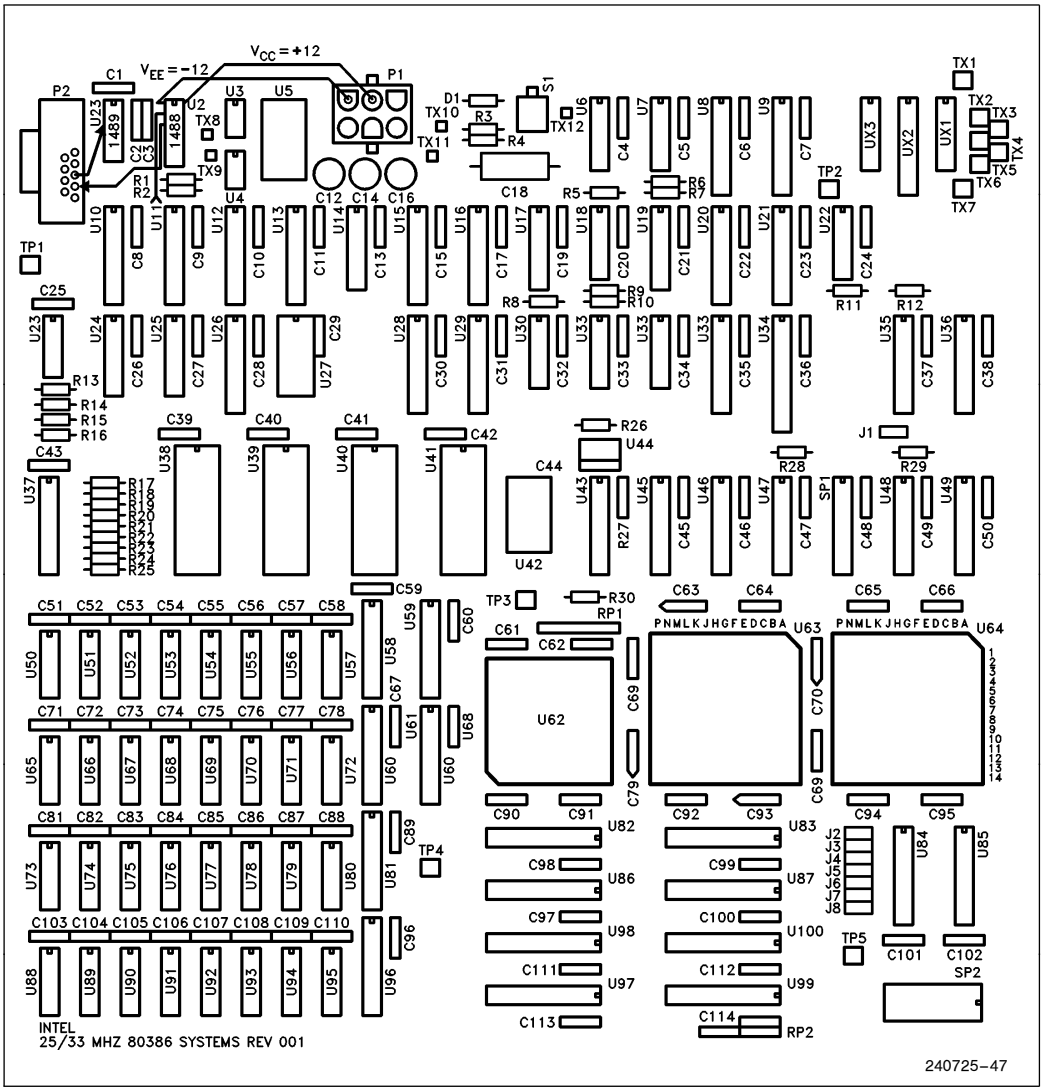

240725-42
74AS245 U81
74AS245 U86
RAMDT/R#
RAMDEN#
D(16-23)
DD(16-23)
NMB2801-06
UM73
UM74
UM75
UM76
UM77
UM78
UM79
UM80
UM88
UM89
UM90
UM91
UM92
UM93
UM94
UM95
WE#
RAS#
CAS#3
CAS#2
DA(0-8)


240725-43
74AS245 U58
74AS245 UB0
DIR
G
D(8-15)
D(0-7)
RAMDT/R#
RAMDEN#
DO(16-23)
NMB2801-06
DIN
DOUT
RAS
CAS
WE
UM50
UM51
UM52
UM53
UM54
UM55
UM56
UM57
UM65
UM66
UM67
UM68
UM69
UM70
UM71
UM72
WE#
RAS#
CAS#1
CAS#0
DA(0-8)


240725-44
MRDC#
IORC#
INTA#
IODT/R#
IODEN#
IOWC#
IORDY#
BS16#
PULLUP
R7
R8
R11
R12
R13
OUT
UM19
UM17
UM31
UM18
UM7
UM30
UM20
UM9
UM8
UM21
P16H6A
IO1
IO2
IO3
IO4
LALE
CLK#
RESET
LW/IO#
LD7:0#
LW/R#
ADS#
CLK
ALE
L510CS#
L58CS#
LEPROM#
CLK2

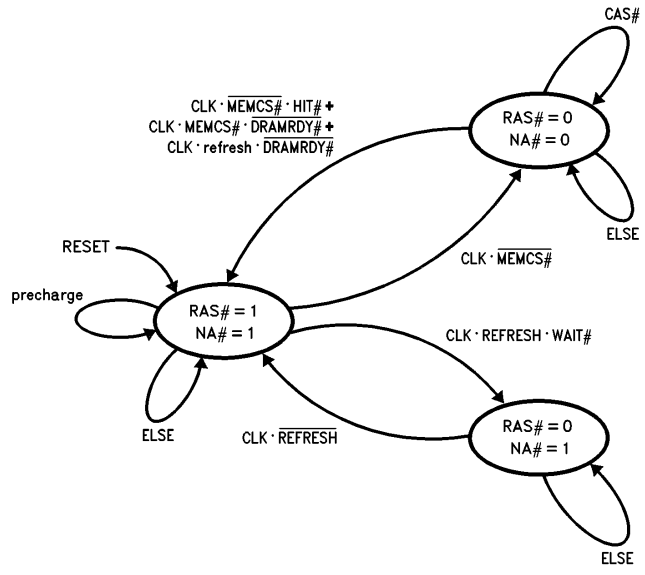

240725-45
D(8-15)
IODT/R#
IODEN#
INTA#
INTR
P/2
P/3
P/7
RS-232
74AS245
UM61
MC1488
UM1
UM2
74LS57
CLK=61.44KHz
UM3
74LS56
LD(8-15)
LD(0-7)
8259A-2
UM39
82510
UM38
18.432MHz
UM5
27128A
UM40
UM41
PWR
R7
UM29
UM28
UM33
A(2-14)
ALE
EPROM#
510CS#
59CS#
D/C#
W/IO#
W/R#
LW/R#
LW/IO#
LD/C#
LEPROM#
IORC#
IOWC#
MRDC#
RESET


P7
P/2 TXD
P/3 RXD
RS232
P7
+5V
+12V
−12V
PWR
240725−46


V_{CC} = +12
V_{EE} = −12
P1
P2
C1
U1
U2
U3
U4
U5
TX8
TX9
D1
TX10
TX11
R3
R4
S1
TX12
C18
R5
R6
R7
TX1
TX2
TX3
TX4
TX5
TX6
TX7
UX1
UX2
UX3
TP1
TP2
R8
R9
R10
R11
R12
C25
R13
R14
R15
R16
C39
C40
C41
C42
C43
C44
R26
U44
R28
R29
J1
SP1
U42
C59
TP3
R30
RP1
C63
C64
C65
C66
U62
U63
U64
C69
C70
C79
C90
C91
C92
C93
C94
C95
U82
U83
U84
U85
C98
C99
U86
U87
C97
C100
U98
U100
C111
C112
U97
U99
C113
C114
RP2
TP4
TP5
C101
C102
SP2
INTEL
25/33 MHZ 80386 SYSTEMS REV 001
240725-47

APPENDIX B
STATE DIAGRAMS AND PALCODES

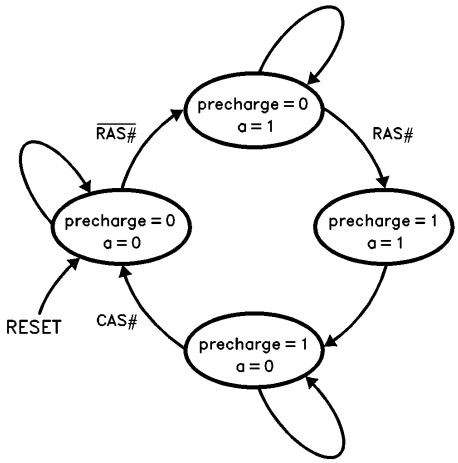
RAS# Generator — DRAM 1—ABL



240725–51

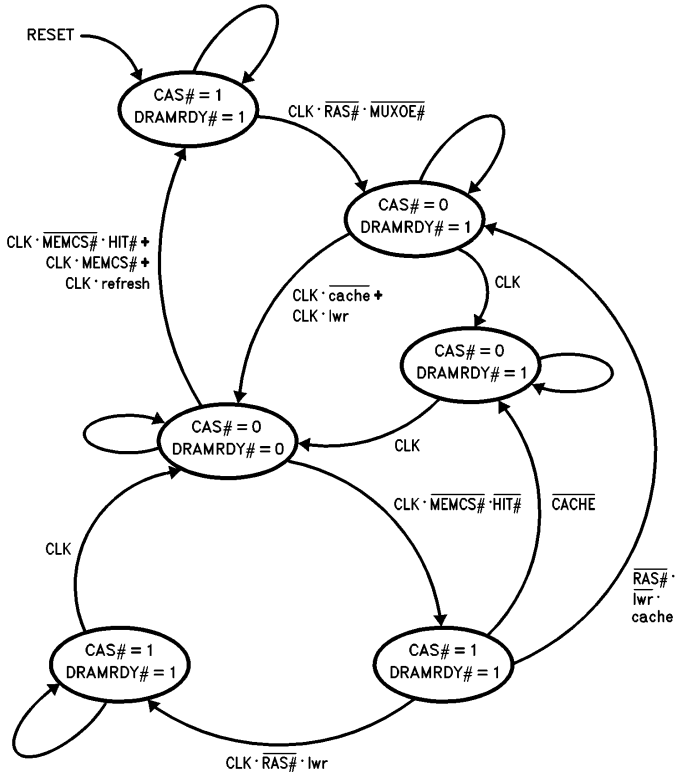
$\overline{MEMCS\#}$ = M/IO# • $\overline{ADS\#}$ • [A31-0E{1FFFFFFF..00000000}]
$\overline{LMEMCS\#}$ = MEMCS# + mreq

Precharge



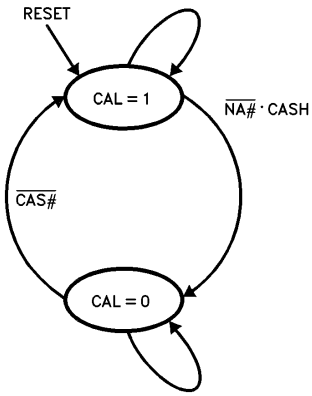
240725–52

CAS#, READY#



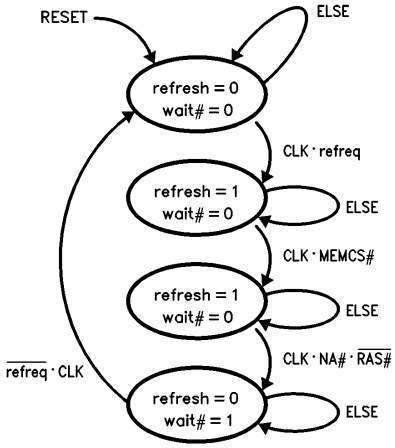
240725–53

CAL Generator



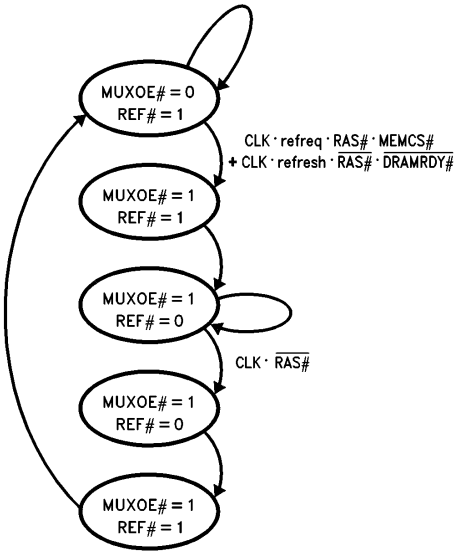
240725–54

Refresh



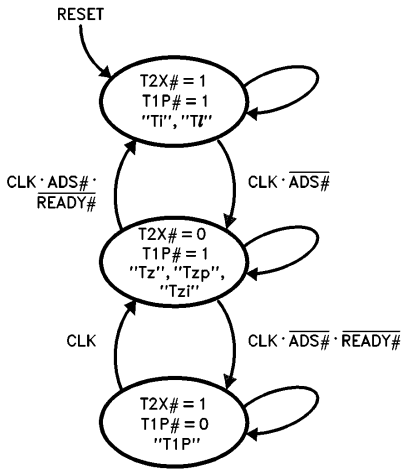
240725–55

MUXOE#, REF# Generator



240725–56

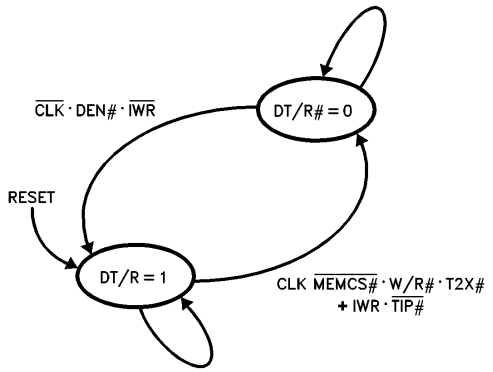
(DRAM2)



240725–57

(DRAM3)

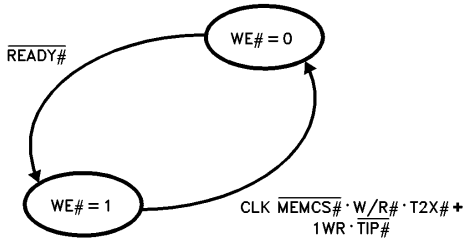
DT/R #



240725–58

(DRAM3)

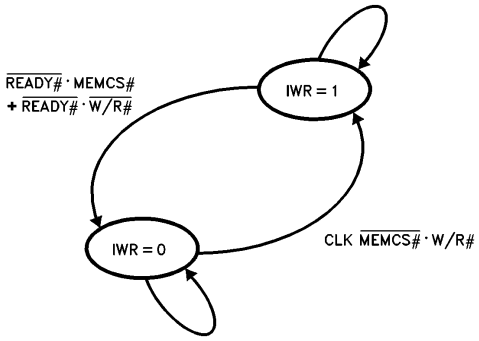
WE #



240725–59

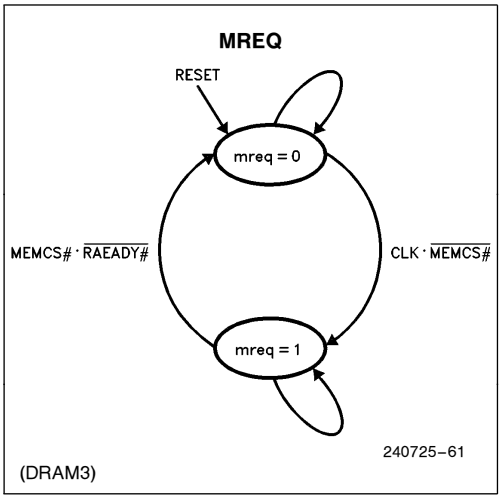
(DRAM3)

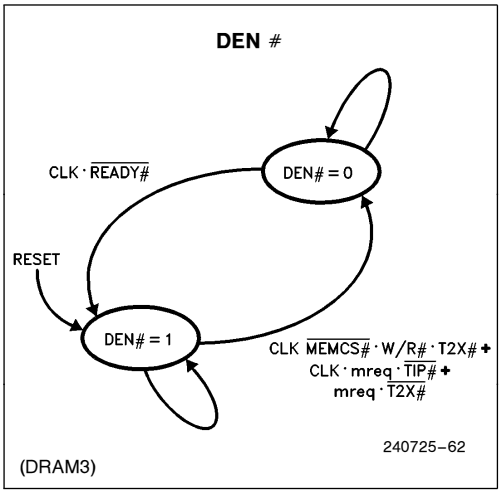
LWR



240725–60

(DRAM3)


MREQ
RESET
mreq = 0
MEMCS# · RAEADY#
CLK · MEMCS#
mreq = 1
240725–61
(DRAM3)


DEN #
CLK · READY#
DEN# = 0
RESET
DEN# = 1
CLK MEMCS# · W/R# · T2X# +
CLK · mreq · TIP# +
mreq · T2X#
240725–62
(DRAM3)

```
module         RESET_GEN flag '-r3'

title   'RESET_GENERATION_LOGIC - INTEL CORPORATION'

    RESET_PAL     device       'P16R8';

    x = .X.;           "ABEL don't care symbol
    c = .C.;           "ABEL clocking input sybol

" Inputs

    CLK2    pin   1;   "CLK2
    RESTRIG pin   2;   "signal from reset circuitry
    CLK_61  pin   9;   "61.44KHz clock

" Outputs

    REFREQ  pin  12;   "REFREQ, sync 61.44KHz clock
    RFQTMP  pin  13;   "temporary stage in sync of 61.44MHz clk
    CLK-    pin  16;   "CLK#
    CLK     pin  17;   "CLK = CLK2 / 2
    RESTMP  pin  18;   "temporary stage in generating RESET
    RESET   pin  19;   "RESET

equations

    CLK := (!CLK # (!RESTMP & RESET));
    CLK- := CLK;
    RESTMP := RESTRIG;
    RESET := RESTMP;
    RFQTMP := CLK_61;
    REFREQ := RFQTMP;


 test_vectors

    ([CLK2, CLK_61, RESTRIG, CLK, CLK-, RESTMP, RESET, RFQTMP, REFREQ] ->
     [CLK, CLK-, RESTMP, RESET, RFQTMP, REFREQ])

"   C  C  R  C  C  R  R  R  R      C  C  R  R  R  R
"   L  L  E  L  L  E  E  F  E      L  L  E  E  F  E
"   K  K  S  K  K  S  S  Q  F      K  K  S  S  Q  F
"   2     T     ~  T  E  T  R         ~  T  E  T  R
"      6  R        M  T  M  E            M  T  M  E
"      1  I        P     P  Q            P     P  Q
"         G

    [c, x, 1, x, x, x, x, x] -> [x, x, 1, x, x, x];
    [c, x, 1, x, x, 1, x, x] -> [x, x, 1, 1, x, x];
    [c, x, 0, x, x, 1, x, x] -> [x, x, 0, 1, x, x];

    [c, x, x, x, x, 0, 1, x] -> [1, x, x, x, x, x];  " clk generation
    [c, x, x, 1, x, x, 0, x] -> [0, 1, x, x, x, x];
    [c, x, x, 0, x, x, x, x] -> [1, 0, x, x, x, x];
```

240725-48

PAL Codes: RESET

```
    [c, x, x, 1, x, 1, x, x, x] -> [0, 1, x, x, x, x];

    [c, x, 0, x, x, x, x, x, x] -> [x, x, 0, x, x, x];  " restmp gen
    [c, x, x, x, x, 0, x, x, x] -> [x, x, x, 0, x, x];  " reset gen
    [c, x, 1, x, x, x, x, x, x] -> [x, x, 1, x, x, x];

    [c, x, x, x, x, 1, x, x, x] -> [x, x, x, 1, x, x];

    [c, 0, x, x, x, x, x, x, x] -> [x, x, x, x, 0, x];  " 61.44KHz clk
    [c, x, x, x, x, x, x, 0, x] -> [x, x, x, x, x, 0];
    [c, 1, x, x, x, x, x, x, x] -> [x, x, x, x, 1, x];
    [c, x, x, x, x, x, x, 1, x] -> [x, x, x, x, x, 1];

end RESET_GEN;
```

240725-49

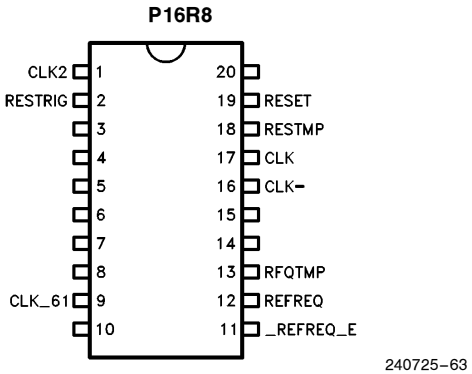
```
ABEL(tm) 3.10  -  Document Generator            14-Feb-90 09:53 AM
RESET_GENERATION_LOGIC - INTEL CORPORATION
Equations for Module RESET_GEN

Device RESET_PAL


- Reduced Equations:

    !CLK := (CLK & !RESET # CLK & RESTMP);

    !CLK~ := (!CLK);

    !RESTMP := (!RESTRIG);

    !RESET := (!RESTMP);

    !RFQTMP := (!CLK_61);

    !REFREQ := (!RFQTMP);
```

240725-D4

PAL Codes: RESET (Continued)

ABEL 3.10—Document Generator 14-Feb-90 09:53 AM
RESET__GENERATION__LOGIC—INTEL CORPORATION
Chip diagram for Module RESET__GEN

Device RESET__PAL



PAL Codes: RESET (Continued)

```
module         ADDR_DEC flag '-r3'

title   'ADDRESS_DECODE_LOGIC - INTEL CORPORATION'

    ADDR_PAL     device  'P16L8';

    x = .X.;            "ABEL don't care symbol
    c = .C.;            "ABEL clocking input sybol

" Inputs

    ADS~     pin   1;   "ADS#
    M_IO~    pin   2;   "M/IO#
    A31      pin   3;   "Addr bit 31
    A30            pin   4;   "Addr bit 30
    A29      pin   5;   "Addr bit 29
    A6       pin   9;   "Addr bit 6
    mreq     pin  11;   "Latched memory chip select

" Outputs

    MEMCS~   pin  18;   "Memory chip select
    _59CS~   pin  15;   "8259A chip select
    _510CS~  pin  14;   "82510 chip select
    EPRDM~   pin  13;   "EPROM chip select
    LMEMCS~  pin  12;   "Latched/unlatched memory chip select

equations

    !MEMCS~ = !ADS~ & M_IO~ & !A31 & !A30 & !A29;
    !LMEMCS~ = (!ADS~ & M_IO~ & !A31 & !A30 & !A29) # mreq;
    !_59CS~ = !M_IO~ & !A6;
    !_510CS~ = !M_IO~ & A6;
    !EPRDM~ =  M_IO~ & A31 & A30 & A29;


 test_vectors

    ([ADS~, M_IO~, A31, A30, A29, A6, mreq, MEMCS~] ->
     [MEMCS~, LMEMCS~, _59CS~, _510CS~, EPRDM~])

"   A  M  A  A  A  A  m  M      M  L  5  5  E
"   D     3  3  2  6  r  E      E  M  9  1  P
"   S  I  1  0  9     e  M      M  E  C  0  R
"   ~  O              q  C      C  M  S  C  D
"      ~                 S      S  C  ~  S  M
"                        ~      ~  S     ~  ~
"                                  ~
"

    [1, x, x, x, x, 0, 1] -> [1, 1, x, x, x];

    [1, x, x, x, x, 1, 1] -> [1, 0, x, x, x];  "LMEMCS~
    [0, 1, 0, 0, 0, x, x, x] -> [0, x, 1, 1, 1];
```

240725-92

```
    [0, 1, 0, 0, 0, x, 0, 0] -> [0, 0, 1, 1, 1];
    [0, 1, 0, 0, 0, x, x, x] -> [0, x, 1, 1, 1];
    [1, x, x, x, x, x, 1, 0] -> [1, 0, x, x, x];

    [1, x, x, x, x, x, x, x] -> [1, x, x, x, x];  "---CS~
    [x, 1, x, x, x, x, x, x] -> [x, x, 1, 1, x];
    [x, 0, x, x, x, x, x, x] -> [1, x, x, x, 1];
    [x, 1, 0, x, x, x, x, x] -> [x, x, 1, 1, 1];
    [x, 1, x, 0, x, x, x, x] -> [x, x, 1, 1, 1];
    [x, 1, x, x, 0, x, x, x] -> [x, x, 1, 1, 1];
    [x, 1, 1, 0, 0, x, x, x] -> [1, x, 1, 1, 1];
    [x, 1, 0, 1, 0, x, x, x] -> [1, x, 1, 1, 1];
    [x, 1, 0, 0, 1, x, x, x] -> [1, x, 1, 1, 1];
    [x, 0, x, x, x, 0, x, x] -> [1, x, x, 1, 1];
    [x, 0, x, x, x, 1, x, x] -> [1, x, 1, x, 1];

    [x, 1, 1, 1, 1, x, x, x] -> [1, x, 1, 1, 0];
    [0, 1, 0, 0, 0, x, x, x] -> [0, x, 1, 1, 1];
    [1, 1, 0, 0, 0, x, x, x] -> [1, x, 1, 1, 1];
    [0, 0, x, x, x, 0, x, x] -> [1, x, 0, 1, 1];
    [0, 0, x, x, x, 1, x, x] -> [1, x, 1, 0, 1];

end ADDR_DEC;
```

240725-93

PAL Codes: Address Decoder

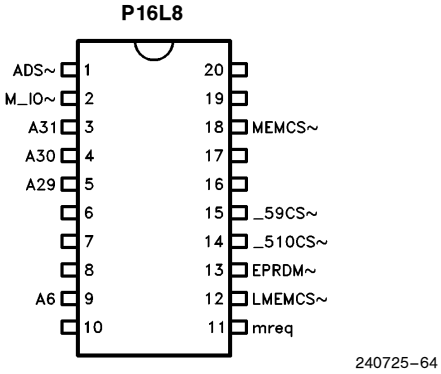
```
ABEL(tm) 3.10  -  Document Generator            14-Feb-90 09:50 AM
ADDRESS_DECODE_LOGIC - INTEL CORPORATION
Equations for Module ADDR_DEC

Device ADDR_PAL


- Reduced Equations:

    !MEMCS~ = (!A29 & !A30 & !A31 & !ADS~ & M_IO~);

    !LMEMCS~ = (mreq # !A29 & !A30 & !A31 & !ADS~ & M_IO~);

    !_59CS~ = (!A6 & !M_IO~);

    !_510CS~ = (A6 & !M_IO~);

    !EPRDM~ = (A29 & A30 & A31 & M_IO~);
```

240725-D5

PAL Codes: Address Decoder (Continued)

ABEL 3.10—Document Generator 14-Feb-90 09:50 AM
ADDRESS_DECODE_LOGIC—INTEL CORPORATION
Chip diagram for Module ADDR_DEC

Device ADDR_PAL



PAL Codes: Address Decoder (Continued)

```
module         PAGE_MODE_DRAM_CTRL_1  flag '-r3'

title   'PAGE MODE DRAM CONTROLLER - PAL 1, INTEL CORPORATION'

    PAGE1    device      'P16R8';

    x       =     .X.;        " ABEL 'don't care' symbol
    c       =     .C.;        " ABEL 'clocking input' symbol

" Inputs

    CLK2    pin   1;   "80386 CLK2
    CLK     pin   2;   "Processor Clock
    MEMCS~  pin   3;   "Memory Chip Select
    LMEMCS~ pin   4;   "Latched/Unlatched Memory Chip Select
    HIT~    pin   5;   "DRAM Page Hit Signal
    CAS~    pin   6;   "Column Address Strobe
    DRAMRDY~      pin   7;   "DRAM Ready Signal
    refreq  pin   8;   "Refresh Request Signal
    RESET   pin   9;   "System Reset

" Outputs

    RAS~    pin  12;    "Row Address Strobe
    NA~     pin  13;    "Next Address Signal
    precharge     pin  14;    "RAS Precharge Signal
    a       pin  15;
    wait~   pin  16;   "delays RAS~ until refresh adress is valid
    CAL     pin  17;   "Column Address Latch
    refresh pin  18;   "Refresh Signal (active once refresh is acknowledged.)

    unused  pin  19;   "

state_diagram [RAS~, NA~]

    state [1, 1]:    if precharge then [1, 1] else
                 if (CLK & refresh & wait~) then [0, 1] else
                 if (CLK & !LMEMCS~& !refresh) then [0, 0] else [1, 1];
    state [0, 0]:    if RESET then [1, 1] else
                 if CAS~ then [0, 0] else
                 if (CLK & !MEMCS~ & HIT~ #
                   CLK & MEMCS~ & !DRAMRDY~ #
                   CLK & refresh & !DRAMRDY~) then [1, 1] else [0, 0];
    state [0, 1]:    if RESET then [1, 1] else
                 if (CLK & !refresh) then [1, 1] else [0, 1];
    state [1, 0]:    goto [1, 1];

state_diagram [precharge, a]

    state [0, 0]:    if (!RAS~) then [0, 1] else [0, 0];
    state [0, 1]:    if (RESET) then [0, 0] else
                 if (RAS~) then [1, 1] else [0, 1];
    state [1, 1]:    goto [1, 0];
    state [1, 0]:    if (CAS~) then [0, 0] else [1, 0];
```

240725–94

PAL Codes: DRAM 1

```
state_diagram [CAL]

    state [1]:      if (!NA~ & CAS~) then [0] else [1];
    state [0]:      if (RESET) then [1] else
              if (!CAS~) then [1] else [0];

state_diagram [refresh, wait~]

    state[0, 0]:    if (CLK & refreq) then [1, 0] else [0, 0];
    state[1, 0]:    if (RESET) then [0,0] else
                        if (CLK & MEMCS~) then [1, 1] else [1, 0];
    state[1, 1]:    if (RESET) then [0,0] else
                        if (CLK & NA~ & !RAS~) then [0, 1] else [1, 1];
    state[0, 1]:    if (RESET) then [0,0] else
                        if (CLK & !refreq) then [0, 0] else [0, 1];

test_vectors

    ([CLK2,CLK,MEMCS~,LMEMCS~,HIT~,CAS~,DRAMRDY~,refreq,RESET] ->
     [RAS~,NA~,precharge,CAL,refresh])

"    C  C  M  L  H  C  D  r  R      R  N  p  C  r
"    L  L  E  M  I  A  R  e  E      A  A  r  A  e
"    K  K  M  E  T  S  A  f  S      S  ~  e  L  f
"    2     C  M  ~  ~  M  r  E      ~     c     r
"          S  C        R  e  T            h     e
"          ~  S        D  q               a     s
"             ~        Y                  r     h
"                      ~                  g
"                                         e

    [c, x, x, x, x, x, 1, x, 1] -> [1, 1, x, 1, 0];
    [c, x, x, x, x, x, 1, x, 1] -> [1, 1, x, 1, 0];
    [c, 1, 1, 1, x, 1, 1, 0, 0] -> [1, 1, x, 1, 0]; "Ti, phase 1
    [c, 0, 1, 1, x, 1, 1, 0, 0] -> [1, 1, x, 1, 0]; "    phase 2
    [c, 1, 1, 1, x, 1, 1, 0, 0] -> [1, 1, x, 1, 0]; "T1, Read, Non-Pipelined
    [c, 0, 0, 0, x, 1, 1, 0, 0] -> [1, 1, 0, 1, 0];
    [c, 1, 0, 0, x, 1, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2
    [c, 0, 1, 0, x, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
    [c, 1, 1, 0, x, 1, 1, 0, 0] -> [0, 0, 0, 0, 0]; "T2P
    [c, 0, 0, 0, x, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "     Page Hit
    [c, 1, 0, 0, 0, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2P
    [c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
    [c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "T1P, Read, Pipelined
    [c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
    [c, 1, 1, 0, 0, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2P
    [c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
    [c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "T1P, Write
    [c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
    [c, 1, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0]; "T2P
    [c, 0, 0, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
    [c, 1, 0, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0]; "T2P
    [c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
```

240725–95

PAL Codes: DRAM 1 (Continued)

```
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "T1P
[c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
[c, 1, 1, 0, 0, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2P
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "    Page Miss
[c, 1, 0, 0, 1, 0, 0, 0, 0] -> [1, 1, 0, 1, 0]; "T1P
[c, 0, 1, 0, 1, 1, 1, 0, 0] -> [1, 1, 1, 1, 0];
[c, 1, 1, 0, 1, 1, 1, 0, 0] -> [1, 1, 1, 1, 0]; "T2
[c, 0, 1, 0, 1, 1, 1, 0, 0] -> [1, 1, 0, 1, 0];
[c, 1, 1, 0, 1, 1, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2
[c, 0, 1, 0, 1, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
[c, 1, 1, 0, 1, 1, 1, 0, 0] -> [0, 0, 0, 0, 0]; "T2P
[c, 0, 0, 0, x, 0, 1, 0, 0] -> [0, 0, 0, 1, 0];
[c, 1, 0, 0, 1, 0, 1, 0, 0] -> [1, 1, 0, 1, 0]; "T2P
[c, 0, 0, 0, 1, 0, 0, 0, 0] -> [1, 1, 1, 1, 0];
[c, 1, 0, 0, 1, 0, 0, 0, 0] -> [1, 1, 1, 1, 0]; "T1P
[c, 0, 1, 0, 1, 1, 1, 0, 0] -> [1, 1, 0, 1, 0];
[c, 1, 1, 0, 1, 1, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2
[c, 0, 1, 0, 1, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
[c, 1, 1, 0, 1, 1, 1, 0, 0] -> [0, 0, 0, 0, 0]; "T2P
[c, 0, 0, 0, 1, 0, 1, 0, 0] -> [0, 0, 0, 1, 0];
[c, 1, 0, 0, 0, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2P
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "T1P
[c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
[c, 1, 1, 0, 0, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2P
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "T1P
[c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
[c, 1, 1, 0, 0, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2i
[c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
[c, 1, 1, 1, 0, 0, 0, 0, 0] -> [1, 1, 0, 1, 0]; "T1
[c, 0, 0, 0, x, 1, 1, 0, 0] -> [1, 1, 1, 1, 0];
[c, 1, 0, 0, x, 1, 1, 0, 0] -> [1, 1, 1, 1, 0]; "T2
[c, 0, 1, 0, x, 1, 1, 0, 0] -> [1, 1, 0, 1, 0];
[c, 1, 1, 0, x, 1, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2
[c, 0, 1, 0, x, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
[c, 1, 1, 0, x, 1, 1, 0, 0] -> [0, 0, 0, 0, 0]; "T2P
[c, 0, 0, 0, x, 0, 1, 0, 0] -> [0, 0, 0, 1, 0];
[c, 1, 0, 0, 0, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2P
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "T1P
[c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
[c, 1, 1, 0, 0, 0, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2P
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "T1P
[c, 0, 1, 0, 0, 1, 1, 1, 0] -> [0, 0, 0, 0, 0];
[c, 1, 1, 0, 0, 0, 1, 1, 0] -> [0, 0, 0, 1, 1]; "T2P
[c, 0, 0, 0, 0, 0, 0, 1, 0] -> [0, 0, 0, 1, 1];
[c, 1, 0, 0, 0, 0, 0, 1, 0] -> [1, 1, 0, 1, 1]; "T1P, Refresh
[c, 0, 1, 0, 0, 1, 1, 1, 0] -> [1, 1, 1, 1, 1];
[c, 1, 1, 0, 0, 1, 1, 1, 0] -> [1, 1, 1, 1, 1]; "T2
[c, 0, 1, 0, 0, 1, 1, 1, 0] -> [1, 1, 0, 1, 1];
[c, 1, 1, 0, 0, 1, 1, 1, 0] -> [0, 1, 0, 1, 1]; "T2
[c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 1, 0, 1, 1];
```

240725-96

```
    [c, 1, 1, 0, 0, 1, 1, 0, 0] -> [0, 1, 0, 1, 0]; "T2
    [c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 1, 0, 1, 0];
    [c, 1, 1, 0, 0, 1, 1, 0, 0] -> [1, 1, 0, 1, 0]; "T2, Pending Read
    [c, 0, 1, 0, 0, 1, 1, 0, 0] -> [1, 1, 1, 1, 0];
    [c, 1, 1, 0, 0, 1, 1, 0, 0] -> [1, 1, 1, 1, 0]; "T2
    [c, 0, 1, 0, 0, 1, 1, 0, 0] -> [1, 1, 0, 1, 0];
    [c, 1, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 1, 0]; "T2
    [c, 0, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0];
    [c, 1, 1, 0, 0, 1, 1, 0, 0] -> [0, 0, 0, 0, 0]; "T2P
    [c, 0, 0, 0, 0, 0, 0, 1, 0] -> [0, 0, 0, 1, 0];
    [c, 1, 0, 0, 0, 0, 0, 1, 0] -> [0, 0, 0, 1, 0]; "T2P
    [c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0];
    [c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1, 0]; "T1P

end PAGE_MODE_DRAM_CTRL_1;
```

240725-97

PAL Codes: DRAM 1 (Continued)

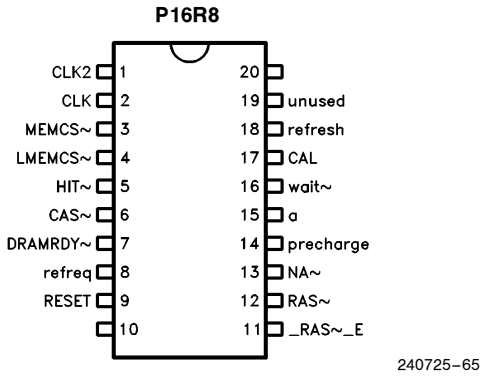
```
ABEL(tm) 3.10  -  Document Generator            15-Feb-90 05:47 PM
PAGE MODE DRAM CONTROLLER - PAL 1, INTEL CORPORATION
Equations for Module PAGE_MODE_DRAM_CTRL_1

Device PAGE1


- Reduced Equations:

    !RAS~ := (NA~ & !RAS~ & !RESET & refresh
              # DRAMRDY~ & !HIT~ & !NA~ & !RAS~ & !RESET
              # DRAMRDY~ & MEMCS~ & !NA~ & !RAS~ & !RESET
              # !HIT~ & !MEMCS~ & !NA~ & !RAS~ & !RESET & !refresh
              # !CLK & !RAS~ & !RESET
              # CAS~ & !NA~ & !RAS~ & !RESET
              # CLK & !LMEMCS~ & NA~ & RAS~ & !precharge & !refresh
              # CLK & NA~ & RAS~ & !precharge & refresh & wait~);


    !NA~ := (DRAMRDY~ & !HIT~ & !NA~ & !RAS~ & !RESET
             # DRAMRDY~ & MEMCS~ & !NA~ & !RAS~ & !RESET
             # !HIT~ & !MEMCS~ & !NA~ & !RAS~ & !RESET & !refresh
             # !CLK & !NA~ & !RAS~ & !RESET
             # CAS~ & !NA~ & !RAS~ & !RESET
             # CLK & !LMEMCS~ & NA~ & RAS~ & !precharge & !refresh);


    !precharge := (CAS~ & !a
                   # !RAS~ & !precharge
                   # RESET & !precharge
                   # !a & !precharge);


    !a := (precharge # RESET & a # RAS~ & !a);

    !CAL := (!CAL & CAS~ & !RESET # CAL & CAS~ & !NA~);

    !refresh := (!refresh & wait~
                 # CLK & NA~ & !RAS~ & wait~
                 # RESET & refresh
                 # !refreq & !refresh
                 # !CLK & !refresh);


    !wait~ := (CLK & !refreq & !refresh
               # !MEMCS~ & !wait~
               # !CLK & !wait~
               # RESET
               # !refresh & !wait~);
```

240725–50

PAL Codes: DRAM 1 (Continued)

ABEL 3.10—Document Generator 15-Feb-90 05:47 PM
PAGE MODE DRAM CONTROLLER—PAL 1, INTEL CORPORATION
Chip diagram for Module PAGE__MODE__CTRL__1

Device PAGE1



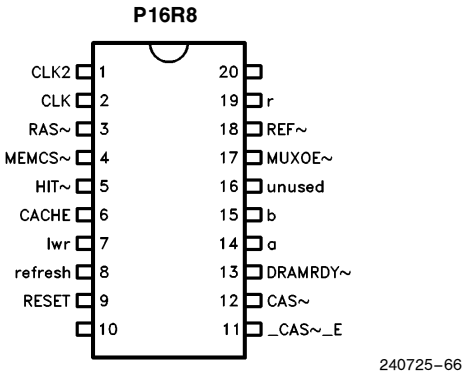
PAL Codes: DRAM 1 (Continued)

```
ABEL(tm) 3.10  -  Document Generator            15-Feb-90 06:16 PM
PAGE MODE DRAM CONTROLLER - PAL 2, INTEL CORPORATION
Equations for Module PAGE_MODE_DRAM_CTRL_2

Device PAGE2


- Reduced Equations:

    !CAS~ := (CAS~ & CLK & DRAMRDY~ & !RESET & !a & !b
            # !CACHE & DRAMRDY~ & !RESET & a & !b & !lwr
            # DRAMRDY~ & !RAS~ & !RESET & a & !b & !lwr
            # !CAS~ & !CLK & !RESET & a & b
            # !CAS~ & DRAMRDY~ & !RESET & a
            # CAS~ & CLK & DRAMRDY~ & !MUXOE~ & !RAS~ & a & b);


    !DRAMRDY~ := (CAS~ & CLK & DRAMRDY~ & !RESET & !a & !b
                # !CAS~ & !CLK & !DRAMRDY~ & !RESET & a & b
                # !CAS~ & CLK & DRAMRDY~ & !RESET & a & !b
                # !CAS~ & CLK & DRAMRDY~ & !RESET & a & lwr
                # !CACHE & !CAS~ & CLK & DRAMRDY~ & !RESET & a);


    !a := (CAS~ & !CLK & DRAMRDY~ & !RESET & !a & !b
          # CAS~ & CLK & DRAMRDY~ & !RAS~ & !RESET & a & !b & lwr);


    !b := (CAS~ & !CLK & DRAMRDY~ & !RESET & !a & !b
          # CAS~ & DRAMRDY~ & RAS~ & !RESET & a & !b
          # !CACHE & CAS~ & DRAMRDY~ & !RESET & a & !b
          # CAS~ & DRAMRDY~ & !RESET & a & !b & lwr
          # !CAS~ & CLK & !DRAMRDY~ & !MEMCS~ & !RAS~ & !RESET & a & b &
        !refresh
          # !CAS~ & !CLK & DRAMRDY~ & !RESET & a & !b
          # CACHE & !CAS~ & CLK & DRAMRDY~ & !RESET & a & b & !lwr);


    !MUXOE~ := (!MUXOE~ & !REF~
                # REF~ & !r
                # MUXOE~ & RESET
                # DRAMRDY~ & !MUXOE~ & !RAS~
                # !MEMCS~ & !MUXOE~ & RAS~
                # !MUXOE~ & !refresh
                # !CLK & !MUXOE~);


    !REF~ := (MUXOE~ & !RESET & r);

    !r := (MUXOE~ & !REF~ & !RESET & !r
          # CLK & MUXOE~ & !RAS~ & !REF~ & !RESET);
                                                                                    240725-98
```

PAL Codes: DRAM 2

ABEL 3.10—Document Generator 15-Feb-90 06:16 PM
PAGE MODE DRAM CONTROLLER—PAL 2, INTEL CORPORATION
Chip diagram for Module PAGE__MODE__DRAM__CTRL__2

Device PAGE2



PAL Codes: DRAM 2 (Continued)

```
module         PAGE_MODE_DRAM_CTRL_2  flag '-r3'

title   'PAGE MODE DRAM CONTROLLER - PAL 2, INTEL CORPORATION'

    PAGE2    device      'P16R8';

    x       =     .X.;        " ABEL 'don't care' symbol
    c       =     .C.;        " ABEL 'clocking input' symbol

" Inputs

    CLK2    pin   1;   "80386 CLK2
    CLK     pin   2;   "Processor Clock
    RAS~    pin   3;   "Row Address Strobe
    MEMCS~  pin   4;   "Memory Chip Select
    HIT~    pin   5;   "DRAM Page Hit Signal (unused)
    CACHE   pin   6;   "Hi when 385 is used; otherwise, Low
    lwr     pin   7;   "Latched Write/Read
    refresh pin   8;   "Refresh Signal
    RESET   pin   9;   "System Reset

" Outputs

    CAS~    pin  12;    "Column Address Strobe
    DRAMRDY~      pin  13;    "DRAM Ready
    a       pin  14;    "
    b       pin  15;    "
    unused  pin  16;   "
    MUXOE~  pin  17;   "DRAM Address Multiplexer Output Enable
    REF~    pin  18;   "Enables refresh counter instead of MUX
    r       pin  19;

    cstate  =  [CAS~, DRAMRDY~,a, b];
    idle    =  [  1 ,   1     ,1, 1];  "Idle
    start   =  [  0 ,   1     ,1, 1];  "CAS~ Active
    wait    =  [  0,    1     ,1, 0];  "CAS~ Active, Wait State
    active  =  [  0 ,   0     ,1, 1];  "CAS~ and DRAMRDY~ Active
    inactive_1    =  [  1 ,   1     ,1,  0];   "Page  Hit,  CAS~  and  DRAMRDY~
Inactive
    inactive_2    =  [  1 ,   1     ,0,  0];   "Page  Hit,  CAS~  and  DRAMRDY~
Inactive
    illegal_a     =  [0,0,0,0];
    illegal_b     =  [0,0,0,1];
    illegal_c     =  [0,0,1,0];
    illegal_d     =  [0,1,0,0];
    illegal_e     =  [0,1,0,1];
    illegal_f     =  [1,0,0,1];
    illegal_g     =  [1,0,1,0];
    illegal_h     =  [1,0,1,1];
    illegal_i     =  [1,1,0,1];
    illegal_j     =  [1,0,0,0];

    muxstate      =  [MUXOE~, REF~, r];
    enabled =  [  0   ,  1    , 1];  "Multiplexer Outputs Enabled
```

240725-99

PAL Codes: DRAM 2 (Continued)

```
    disabled_1    =  [  1   ,  1    , 1];  "Multiplexer Outputs Disabled
    disabled_2    =  [  1   ,  0    , 1];  "Refresh Address Enabled
    disabled_3    =  [  1   ,  0    , 0];  "Wait for RAS#
    disabled_4    =  [  1   ,  1    , 0];  "Refresh Address Disabled
    illegal_z     =  [0,0,0];
    illegal_y     =  [0,0,1];
    illegal_x     =  [0,1,0];

state_diagram cstate

    state idle:       if (CLK & !RAS~ & !MUXOE~) then start else idle;
    state start:      if RESET then idle else
                  if (CLK & !CACHE # CLK & !wr) then active else
                  if CLK then wait else start;
    state wait:       if RESET then idle else
                  if CLK then active else wait;
    state active:     if RESET then idle else
                  if (CLK & !MEMCS~ & RAS~ #
                    CLK & MEMCS~ #
                    CLK & refresh) then idle else
                  if (CLK & !MEMCS~ & !RAS~) then inactive_1
                  else active;
    state inactive_1: if RESET then idle else
                  if (CLK & !RAS~ & !wr) then inactive_2 else
                  if (!RAS~ & !!wr & CACHE) then start else
                  if (!!wr & !CACHE) then wait else
                  inactive_1;
    state inactive_2: if RESET then idle else
                  if CLK then active else inactive_2;
    state illegal_a:  goto idle;
    state illegal_b:  goto idle;
    state illegal_c:  goto idle;
    state illegal_d:  goto idle;
    state illegal_e:  goto idle;
    state illegal_f:  goto idle;
    state illegal_g:  goto idle;
    state illegal_h:  goto idle;
    state illegal_i:  goto idle;
    state illegal_j:  goto idle;

state_diagram muxstate

    state enabled:    if (CLK & refresh & RAS~ & MEMCS~ #
                    CLK & refresh & !RAS~ & !DRAMRDY~) then
                    disabled_1 else enabled;
    state disabled_1: if (RESET) then enabled else disabled_2;
    state disabled_2: if (RESET) then enabled else
                  if (CLK & !RAS~) then disabled_3 else disabled_2;
    state disabled_3: if (RESET) then enabled else disabled_4;
    state disabled_4: goto enabled;
    state illegal_z:  goto enabled;
    state illegal_y:  goto enabled;
    state illegal_x:  goto enabled;
```

240725–A0

PAL Codes: DRAM 2 (Continued)

```
test_vectors

    ([CLK2,CLK,MEMCS~,lwr,HIT~,RAS~,refresh,RESET,CACHE] ->
     [CAS~,DRAMRDY~,MUXOE~,REF~])


"    C  C  M  l  H  R  r  R  C      C  D  M  R
"    L  L  E  w  I  A  e  E  A      A  R  U  E
"    K  K  M  r  T  S  f  S  C      S  A  X  F
"    2     C     ~  ~  r  E  H      ~  M  O  ~
"          S           e  T  E         R  E
"          ~           s               D  ~
"                      h            Y
"                            ~


    [c, x, x, 0, x, x, x, 1, 0] -> [1, 1, 0, 1]; "Cache disabled
    [c, x, x, 0, x, x, x, 1, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 0, x, 1, 0, 0, 0] -> [1, 1, 0, 1]; "Ti
    [c, 1, 1, 0, x, 1, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 0, 0, x, 1, 0, 0, 0] -> [1, 1, 0, 1]; "T1
    [c, 1, 0, 0, x, 1, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 0, x, 0, 0, 0, 0] -> [1, 1, 0, 1]; "T2
    [c, 1, 1, 0, x, 0, 0, 0, 0] -> [0, 1, 0, 1];
    [c, 0, 0, 0, x, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T2P
    [c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
    [c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
    [c, 1, 0, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T1P
    [c, 1, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
    [c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
    [c, 1, 0, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 1, 0, 0, 0, 0, 0] -> [1, 1, 0, 1]; "T1P
    [c, 1, 1, 1, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 0, 1, 0, 0, 0, 0, 0] -> [1, 1, 0, 1]; "T2P
    [c, 1, 0, 1, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
    [c, 0, 0, 1, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2p
    [c, 1, 0, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T1P
    [c, 1, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
    [c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
    [c, 1, 0, 0, 1, 1, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 0, 1, 1, 0, 0, 0] -> [1, 1, 0, 1]; "T1P
    [c, 1, 1, 0, 1, 1, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 0, 1, 1, 0, 0, 0] -> [1, 1, 0, 1]; "T2
    [c, 1, 1, 0, 1, 1, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 0, 1, 0, 0, 0, 0] -> [1, 1, 0, 1]; "T2
    [c, 1, 1, 0, 1, 0, 0, 0, 0] -> [0, 1, 0, 1];
    [c, 0, 0, 0, x, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T2P
    [c, 1, 0, 0, 1, 0, 0, 0, 0] -> [0, 0, 0, 1];
    [c, 0, 0, 1, 1, 1, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
    [c, 1, 0, 0, 1, 1, 0, 0, 0] -> [1, 1, 0, 1];
    [c, 0, 1, 1, 1, 1, 0, 0, 0] -> [1, 1, 0, 1]; "T1P
    [c, 1, 1, 1, 1, 1, 0, 0, 0] -> [1, 1, 0, 1];
                                                                                    240725–A1
```

PAL Codes: DRAM 2 (Continued)

```
[c, 0, 1, 1, 1, 0, 0, 0, 0] -> [1, 1, 0, 1]; "T2
[c, 1, 1, 1, 1, 0, 0, 0, 0] -> [0, 1, 0, 1];
[c, 0, 0, 1, 1, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T2P
[c, 1, 0, 1, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
[c, 0, 0, 1, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T1P
[c, 1, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T1P
[c, 1, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
[c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2i
[c, 1, 1, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 0, 0, x, 1, 0, 0, 0] -> [1, 1, 0, 1]; "T1
[c, 1, 0, 0, x, 1, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 1, x, 1, 0, 0, 0] -> [1, 1, 0, 1]; "T2
[c, 1, 1, 1, x, 1, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 1, x, 0, 0, 0, 0] -> [1, 1, 0, 1]; "T2
[c, 1, 1, 1, x, 0, 0, 0, 0] -> [0, 1, 0, 1];
[c, 0, 0, 1, x, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T2P
[c, 1, 0, 1, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
[c, 0, 0, 1, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T1P
[c, 1, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T1P
[c, 1, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 1, 0, 0] -> [1, 1, 1, 1];
[c, 0, 1, 0, 0, 1, 1, 0, 0] -> [1, 1, 1, 0]; "T1P
[c, 1, 1, 0, 0, 1, 1, 0, 0] -> [1, 1, 1, 0];
[c, 0, 1, 0, 0, 1, 1, 0, 0] -> [1, 1, 1, 0]; "T2
[c, 1, 1, 0, 0, 1, 1, 0, 0] -> [1, 1, 1, 0];
[c, 0, 1, 0, 0, 0, 1, 0, 0] -> [1, 1, 1, 0]; "T2
[c, 1, 1, 0, 0, 0, 1, 0, 0] -> [1, 1, 1, 0];
[c, 0, 1, 0, 0, 0, 1, 0, 0] -> [1, 1, 1, 1]; "T2
[c, 1, 1, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 0, 0, 1, 0, 0, 0] -> [1, 1, 0, 1]; "T2
[c, 1, 1, 0, 0, 1, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 0, 0, 1, 0, 0, 0] -> [1, 1, 0, 1]; "T2
[c, 1, 1, 0, 0, 1, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1]; "T2
[c, 1, 1, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1];
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0, 0, 0] -> [1, 1, 0, 1];
[c, 0, 1, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 1]; "T1P
[c, 1, 1, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1];
[c, 0, 0, 0, 0, 0, 0, 0, 0] -> [0, 0, 0, 1]; "T2P
[c, x, x, 0, x, x, x, 1, 1] -> [1, 1, 0, 1]; "Cache eanbled
```

240725–A2

PAL Codes: DRAM 2 (Continued)

```
      [c, x, x, 0, x, x, x, 1, 1] -> [1, 1, 0, 1];
      [c, 0, 1, 0, x, 1, 0, 0, 1] -> [1, 1, 0, 1]; "Ti
      [c, 1, 1, 0, x, 1, 0, 0, 1] -> [1, 1, 0, 1];
      [c, 0, 0, 0, x, 1, 0, 0, 1] -> [1, 1, 0, 1]; "T1, Read
      [c, 1, 0, 0, x, 1, 0, 0, 1] -> [1, 1, 0, 1];
      [c, 0, 1, 0, x, 0, 0, 0, 1] -> [1, 1, 0, 1]; "T2
      [c, 1, 1, 0, x, 0, 0, 0, 1] -> [0, 1, 0, 1];
      [c, 0, 0, 0, x, 0, 0, 0, 1] -> [0, 1, 0, 1]; "T2P
      [c, 1, 0, 0, 0, 0, 0, 0, 1] -> [0, 1, 0, 1];
      [c, 0, 0, 0, 0, 0, 0, 0, 1] -> [0, 1, 0, 1]; "T2P
      [c, 1, 0, 0, 0, 0, 0, 0, 1] -> [0, 0, 0, 1];
      [c, 0, 0, 0, 0, 0, 0, 0, 1] -> [0, 0, 0, 1]; "T2P
      [c, 1, 0, 0, 0, 0, 0, 0, 1] -> [1, 1, 0, 1];
      [c, 0, 1, 0, 0, 0, 0, 0, 1] -> [0, 1, 0, 1]; "T1P, Read
      [c, 1, 1, 0, 0, 0, 0, 0, 1] -> [0, 1, 0, 1];
      [c, 0, 0, 0, 0, 0, 0, 0, 1] -> [0, 1, 0, 1]; "T2P
      [c, 1, 0, 0, 0, 0, 0, 0, 1] -> [0, 0, 0, 1];
      [c, 0, 0, 0, 0, 0, 0, 0, 1] -> [0, 0, 0, 1]; "T2P
      [c, 1, 0, 0, 0, 0, 0, 0, 1] -> [1, 1, 0, 1];
      [c, 0, 1, 1, 0, 0, 0, 0, 1] -> [1, 1, 0, 1]; "T1P, Write
      [c, 1, 1, 1, 0, 0, 0, 0, 1] -> [1, 1, 0, 1];
      [c, 0, 0, 1, 0, 0, 0, 0, 1] -> [1, 1, 0, 1]; "T2P
      [c, 1, 0, 1, 0, 0, 0, 0, 1] -> [0, 0, 0, 1];
      [c, 0, 0, 1, 0, 0, 0, 0, 1] -> [0, 0, 0, 1]; "T2p
      [c, 1, 0, 0, 0, 0, 0, 0, 1] -> [1, 1, 0, 1];

end PAGE_MODE_DRAM_CTRL_2;
^Z
```

240725–A3

PAL Codes: DRAM 2 (Continued)

```
module         PAGE_MODE_DRAM_CTRL_3  flag '-r3'

title   'PAGE MODE DRAM CONTROLLER - PAL 3, INTEL CORPORATION'

    PAGE3    device      'P16R8';

    x       =     .X.;        " ABEL 'don't care' symbol
    c       =     .C.;        " ABEL 'clocking input' symbol

" Inputs

    CLK2    pin   1;   "80386 CLK2
    CLK     pin   2;   "Processor Clock
    ADS~    pin   3;   "Address Strobe
    MEMCS~  pin   4;   "Memory Chip Select
    WR            pin   5;   "Write/Read
    READY~  pin   6;   "System Ready
    DRMRDY~ pin   7;   "DRAM Ready
    unused1 pin   8;
    RESET   pin   9;   "System Reset

" Outputs

    T2X~    pin  12;    "active during T2, T2p, and T2i
    T1P~    pin  13;    "active during T1p
    WE~     pin  14;    "DRAM Write Enable
    DEN~    pin  15;    "DRAM Data Bus Transceiver Enable
    DTR     pin  16;    "DRAM Data Bus Transceiver R/W# Direction signal
    lwr     pin  17;    "Latched Write/Read
    mreq    pin  18;    "Latched Memory Chip Select
    unused2 pin  19;    "

state_diagram [T2X~, T1P~]

    state [1, 1]:    if (CLK & !ADS~) then [0, 1] else [1, 1];
    state [0, 1]:    if RESET then [1, 1] else
                  if (CLK & !ADS~ & !READY~) then [1, 0] else
                  if (CLK & ADS~ & !READY~) then [1, 1] else [0, 1];
    state [1, 0]:    if RESET then [1, 1] else
                  if (CLK) then [0, 1] else [1, 0];
    state [0, 0]:    goto [1, 1];

state_diagram [WE~]

    state [1]:      if (CLK & !MEMCS~ & WR & T2X~ #
                   lwr & !T1P~) then [0] else [1];
    state [0]:      if (RESET) then [1] else
              if (CLK & !READY~) then [1] else [0];

state_diagram [DEN~]

    state [1]:      if (CLK & !MEMCS~ & !WR & T2X~ #
                   mreq & !T2X~ #
                   CLK & mreq & !T1P~) then [0] else [1];
```

240725-A4

PAL Codes: DRAM 3

```
    state [0]:      if RESET then [1] else
              if (CLK & !READY~) then [1] else [0];

state_diagram [DTR]

    state [1]:      if (CLK & !MEMCS~ & WR & T2X~ #
                   lwr & !T1P~) then [0] else [1];
    state [0]:      if (RESET) then [1] else
              if (!CLK & DEN~ & !lwr) then [1] else [0];

state_diagram [lwr]

    state [0]:      if (CLK & !MEMCS~ & WR) then [1] else [0];
    state [1]:      if (RESET) then [0] else
              if (!READY~ & MEMCS~ #
                  !READY~ & !WR) then [0] else [1];

state_diagram [mreq]

    state [0]:      if (CLK & !MEMCS~) then [1] else [0];
    state [1]:      if RESET then [0] else
              if (!READY~ & MEMCS~) then [0] else [1];

test_vectors

    ([CLK2,CLK,ADS~,WR,MEMCS~,READY~,RESET] ->
     [T2X~,T1P~,DEN~,lwr,WE~,DTR, mreq])

"    C  C  A  W  M  R  R      T  T  D  l  W  D  m
"    L  L  D  R  E  E  E      2  1  E  w  E  T  r
"    K  K  S     M  A  S      X  P  N  r  ~  R  e
"    2     ~     C  D  E      ~  ~  ~           q
"                S  Y  T
"                ~  ~

    [c, x, x, x, x, x, 1] -> [1, 1, 1, 0, 1, 1, x];
    [c, x, x, x, x, x, 1] -> [1, 1, 1, 0, 1, 1, 0];
    [c, 1, 1, x, 1, 1, 0] -> [1, 1, 1, 0, 1, 1, 0];
    [c, 0, 1, x, 1, 1, 0] -> [1, 1, 1, 0, 1, 1, 0]; "Ti
    [c, 1, 1, x, 1, 1, 0] -> [1, 1, 1, 0, 1, 1, 0];
    [c, 0, 0, 0, 0, 1, 0] -> [1, 1, 1, 0, 1, 1, 0]; "T1
    [c, 1, 0, 0, 0, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 0, 0, 0, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
    [c, 1, 0, 0, 0, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
    [c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 1, 1];
    [c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1P
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 0, 1, 0, 0, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
    [c, 1, 0, 1, 0, 0, 0] -> [1, 0, 1, 1, 1, 1, 1];
    [c, 0, 1, 1, 1, 1, 0] -> [1, 0, 1, 1, 0, 0, 1]; "T1P
    [c, 1, 1, 1, 1, 1, 0] -> [0, 1, 0, 1, 0, 0, 1];
```

240725–A5

PAL Codes: DRAM 3 (Continued)

```
[c, 0, 0, 0, 0, 1, 0] -> [0, 1, 0, 1, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 1, 0] -> [0, 1, 0, 1, 0, 0, 1];
[c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 0, 1];
[c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1P
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
[c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 1, 1];
[c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1p
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
[c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
[c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
[c, 0, 0, 1, 0, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
[c, 1, 0, 1, 0, 1, 0] -> [0, 1, 0, 1, 1, 1, 1];
[c, 0, 0, 1, 0, 0, 0] -> [0, 1, 0, 1, 1, 1, 1]; "T2p
[c, 1, 0, 1, 0, 0, 0] -> [1, 0, 1, 1, 1, 1, 1];
[c, 0, 1, 1, 1, 1, 0] -> [1, 0, 1, 1, 0, 0, 1]; "T1P
[c, 1, 1, 1, 1, 1, 0] -> [0, 1, 0, 1, 0, 0, 1];
[c, 0, 1, 1, 1, 1, 0] -> [0, 1, 0, 1, 0, 0, 1]; "T2
[c, 1, 1, 1, 1, 1, 0] -> [0, 1, 0, 1, 0, 0, 1];
[c, 0, 0, 0, 0, 1, 0] -> [0, 1, 0, 1, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 1, 0] -> [0, 1, 0, 1, 0, 0, 1];
[c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 0, 1];
[c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1P
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
[c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 1, 1];
[c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1P
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
[c, 0, 1, 0, 1, 0, 0] -> [0, 1, 0, 0, 1, 1, 0]; "T2i
[c, 1, 1, 0, 1, 0, 0] -> [1, 1, 1, 0, 1, 1, 0];
[c, 0, 0, 1, 0, 1, 0] -> [1, 1, 1, 0, 1, 1, 0]; "T1
[c, 1, 0, 1, 0, 1, 0] -> [0, 1, 1, 1, 0, 0, 1];
[c, 0, 1, 1, 1, 1, 0] -> [0, 1, 0, 1, 0, 0, 1]; "T2
[c, 1, 1, 1, 1, 1, 0] -> [0, 1, 0, 1, 0, 0, 1];
[c, 0, 1, 1, 1, 1, 0] -> [0, 1, 0, 1, 0, 0, 1]; "T2
[c, 1, 1, 1, 1, 1, 0] -> [0, 1, 0, 1, 0, 0, 1];
[c, 0, 0, 0, 0, 1, 0] -> [0, 1, 0, 1, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 1, 0] -> [0, 1, 0, 1, 0, 0, 1];
[c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 0, 0, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 0, 1];
[c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1P
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
[c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 1, 1];
[c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1P
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
[c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
[c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 1, 1];
[c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1P
[c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
```

240725–A6

```
    [c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 0, 0, 0, 1, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
    [c, 1, 0, 0, 0, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P
    [c, 1, 0, 0, 0, 0, 0] -> [1, 0, 1, 0, 1, 1, 1];
    [c, 0, 1, 0, 1, 1, 0] -> [1, 0, 1, 0, 1, 1, 1]; "T1P
    [c, 1, 1, 0, 1, 1, 0] -> [0, 1, 0, 0, 1, 1, 1];
    [c, 0, 0, 0, 0, 0, 0] -> [0, 1, 0, 0, 1, 1, 1]; "T2P

end PAGE_MODE_DRAM_CTRL_3;
```

240725–A7

PAL Codes: DRAM 3 (Continued)

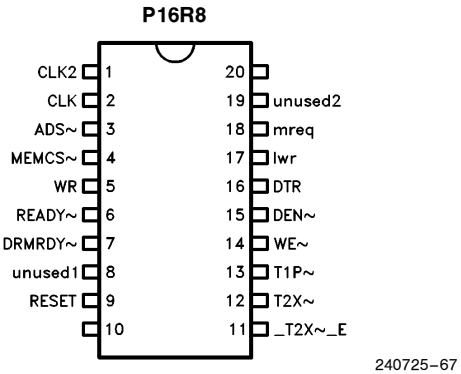
```
ABEL(tm) 3.10  -  Document Generator            14-Feb-90 09:54 AM
PAGE MODE DRAM CONTROLLER - PAL 3, INTEL CORPORATION
Equations for Module PAGE_MODE_DRAM_CTRL_3

Device PAGE3


- Reduced Equations:

    !T2X~ := (CLK & !RESET & !T1P~ & T2X~
              # READY~ & !RESET & T1P~ & !T2X~
              # !CLK & !RESET & T1P~ & !T2X~
              # !ADS~ & CLK & T1P~ & T2X~);


    !T1P~ := (!CLK & !RESET & !T1P~ & T2X~
              # !ADS~ & CLK & !READY~ & !RESET & T1P~ & !T2X~);


    !WE~ := (READY~ & !RESET & !WE~
             # !CLK & !RESET & !WE~
             # !T1P~ & WE~ & !wr
             # CLK & !MEMCS~ & T2X~ & WE~ & WR);


    !DEN~ := (!DEN~ & READY~ & !RESET
              # !CLK & !DEN~ & !RESET
              # CLK & DEN~ & !T1P~ & mreq
              # DEN~ & !T2X~ & mreq
              # CLK & DEN~ & !MEMCS~ & T2X~ & !WR);


    !DTR := (!DTR & !RESET & !wr
             # !DEN~ & !DTR & !RESET
             # CLK & !DTR & !RESET
             # DTR & !T1P~ & !wr
             # CLK & DTR & !MEMCS~ & T2X~ & WR);


    !iwr := (!READY~ & !WR
             # MEMCS~ & !READY~
             # RESET & !wr
             # !WR & !!wr
             # MEMCS~ & !!wr
             # !CLK & !!wr);


    !mreq := (MEMCS~ & !READY~
              # RESET & mreq
              # MEMCS~ & !mreq
              # !CLK & !mreq);
```

240725–A8

PAL Codes: DRAM 3 (Continued)

ABEL 3.10—Document Generator 14-Feb-90 09:54 AM
PAGE MODE DRAM CONTROLLER—PAL 3, INTEL CORPORATION
Chip diagram for Module PAGE__MODE__DRAM__CTRL__3

Device PAGE3



PAL Codes: DRAM 3 (Continued)

```
module         PAGE_MODE_DRAM_CTRL_4  flag '-r3'

title   'PAGE MODE DRAM CONTROLLER - PAL 4, INTEL CORPORATION'

    PAGE4    device      'P16R8';

    x       =     .X.;        " ABEL 'don't care' symbol
    c       =     .C.;        " ABEL 'clocking input' symbol

" Inputs

    CLOCK   pin   1;
    D0      pin   2;
    D1      pin   3;
    D2      pin   4;
    D3      pin   5;
    D4      pin   6;
    D5      pin   7;
    D6      pin   8;
    D7      pin   9;
    OE      pin   11;

" Outputs

    A0      pin  12;
    A1      pin  13;
    A2      pin  14;
    A3      pin  15;
    A4      pin  16;
    A5      pin  17;
    A6      pin  18;
    A7      pin  19;

addr = [A7..A0];

equations

    addr := addr + 1;

end PAGE_MODE_DRAM_CTRL_4;
```

240725-A9

PAL Codes: DRAM 4

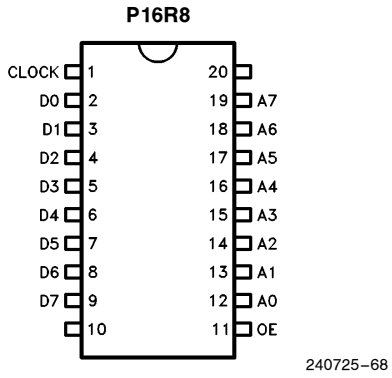
```
ABEL(tm) 3.10  -  Document Generator            14-Feb-90 09:54 AM
PAGE MODE DRAM CONTROLLER - PAL 4, INTEL CORPORATION
Equations for Module PAGE_MODE_DRAM_CTRL_4

Device PAGE4


- Reduced Equations:

    !A7 := (A0 & A1 & A2 & A3 & A4 & A5 & A6 & A7
            # !A0 & !A7
            # !A1 & !A7
            # !A2 & !A7
            # !A3 & !A7
            # !A4 & !A7
            # !A5 & !A7
            # !A6 & !A7);


    !A6 := (A0 & A1 & A2 & A3 & A4 & A5 & A6
            # !A0 & !A6
            # !A1 & !A6
            # !A2 & !A6
            # !A3 & !A6
            # !A4 & !A6
            # !A5 & !A6);


    !A5 := (A0 & A1 & A2 & A3 & A4 & A5
            # !A0 & !A5
            # !A1 & !A5
            # !A2 & !A5
            # !A3 & !A5
            # !A4 & !A5);


    !A4 := (A0 & A1 & A2 & A3 & A4
            # !A0 & !A4
            # !A1 & !A4
            # !A2 & !A4
            # !A3 & !A4);


    !A3 := (A0 & A1 & A2 & A3 # !A0 & !A3 # !A1 & !A3 # !A2 & !A3);

    !A2 := (A0 & A1 & A2 # !A0 & !A2 # !A1 & !A2);

    !A1 := (A0 & A1 # !A0 & !A1);

    !A0 := (A0);
```

240725–B0

PAL Codes: DRAM 4 (Continued)

ABEL 3.10—Document Generator 14-Feb-90 09:54 AM
PAGE MODE DRAM CONTROLLER—PAL 4, INTEL CORPORATION
Chip diagram for Module PAGE__MODE__DRAM__CTRL__4

Device PAGE4



end of module PAGE__MODE__DRAM__CTRL__4

PAL Codes: DRAM 4 (Continued)

```
module     IO_CTRL_1  flag '-r3'

title     'IO BUS CONTROLLER - PAL 1, INTEL CORPORATION'

    IO1    device      'P16R4';

    x           =     .X.;              " ABEL 'don't care' symbol
    c           =     .C.;              " ABEL 'clocking input' symbol

" Inputs

    CLK         pin   1;   "Processor Clock
    RESET       pin   2;   "System Reset
    MRDC~       pin   3;   "Memory (EPROM) Read Command
    IORC~       pin   4;   "I/O Read Command
    IOWC~       pin   5;   "I/O Write Command
    INTA~       pin   6;   "Interrupt Acknowledge
    DEN~        pin   7;   "I/O Bus Data Transceiver Enable
    IORDY~      pin   8;   "I/O-EPROM Ready
    L51OCS~     pin   9;   "8251O Chip Select
    OEN~        pin  11;   "PAL output Enable
    L59CS~      pin  12;   "8259A-2 Chip Select
    LEPROM~     pin  13;   "EPROM Chip Select
    unused_0    pin  18;   "
    unused_1    pin  19;   "

" Outputs

    delay       pin  14;   "
    s2          pin  15;   "
    s1          pin  16;   "
    s0          pin  17;   "

    dstate      =  [delay, s2, s1, s0];
    idle        =  [  1  , 1 , 1 , 1 ];
    start       =  [  1  , 1 , 1 , 0 ];
    wait_14     =  [  1  , 0 , 1 , 0 ];
    wait_13     =  [  1  , 0 , 1 , 1 ];
    wait_12     =  [  1  , 0 , 0 , 0 ];
    wait_11     =  [  1  , 1 , 0 , 0 ];
    wait_10     =  [  1  , 1 , 0 , 1 ];
    active      =  [  0  , 1 , 1 , 1 ];


state_diagram dstate

    state idle:    if (!DEN~ & !MRDC~ # !DEN~ & !IORC~ #
                       !DEN~ & !IOWC~ # !DEN~ & !INTA~) then start
                   else idle;
    state start:   if (!L51OCS~ & !IOWC~) then wait_14 else
                   if (!L51OCS~ & !IORC~) then wait_13 else
                   if (!L59CS~ & !IOWC~) then wait_11 else
                   if (!LEPROM~ # !L59CS~ & !IORC~ # !INTA~) then wait_10;
    state wait_14: goto wait_13;
```

240725–B1

```
    state wait_13: goto wait_12;
    state wait_12: goto wait_11;
    state wait_11: goto wait_10;
    state wait_10: goto active;
    state active:  if !IORDY~ then idle else active;

end IO_CTRL_1;
^Z
```

240725–B2

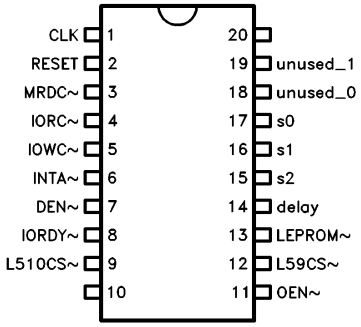
PAL Codes: IO-1

```
ABEL(tm) 3.10  -  Document Generator            15-Feb-90 06:40 PM
IO BUS CONTROLLER - PAL 1, INTEL CORPORATION
Equations for Module IO_CTRL_1

Device IO1


- Reduced Equations:

    !delay := (IORDY~ & !delay & s0 & s1 & s2 # delay & s0 & !s1 & s2);

    !s2 := (delay & s1 & !s2
            # !IORC~ & !L51OCS~ & delay & !s0 & s1
            # !IOWC~ & !L51OCS~ & delay & !s0 & s1);


    !s1 := (delay & !s0 & !s1
            # delay & s0 & s1 & !s2
            # !INTA~ & IORC~ & IOWC~ & delay & !s0 & s2
            # IORC~ & IOWC~ & !LEPROM~ & delay & !s0 & s2
            # !IORC~ & L51OCS~ & !L59CS~ & delay & !s0 & s2
            # !INTA~ & L51OCS~ & delay & !s0 & s2
            # L51OCS~ & !LEPROM~ & delay & !s0 & s2
            # !IOWC~ & L51OCS~ & !L59CS~ & delay & !s0 & s2);


    !s0 := (delay & !s0 & !s1 & !s2
            # delay & s0 & s1 & !s2
            # !IOWC~ & !L59CS~ & delay & !s0 & s1 & s2
            # !IOWC~ & !L51OCS~ & delay & !s0 & s1 & s2
            # !DEN~ & !INTA~ & delay & s0 & s1
            # !DEN~ & !IOWC~ & delay & s0 & s1
            # !DEN~ & !IORC~ & delay & s0 & s1
            # !DEN~ & !MRDC~ & delay & s0 & s1);
                                                                                240725–B3
```

PAL Codes: IO-1 (Continued)

ABEL 3.10—Document Generator 15-Feb-90 06:40 PM
IO BUS CONTROLLER—PAL 1, INTEL CORPORATION
Chip diagram for Module IO__CTRL__1

Device IO1



240725-69

end of module IO__CTRL__1

PAL Codes: IO-1 (Continued)

```
module     IO_CTRL_2  flag '-r3'

title     'IO BUS CONTROLLER - PAL 2, INTEL CORPORATION'

    IO2    device      'P16R6';

    x           =     .X.;              " ABEL 'don't care' symbol
    c           =     .C.;              " ABEL 'clocking input' symbol

" Inputs

    CLK         pin   1;   "Processor Clock
    RESET       pin   2;   "System Reset
    LMIO        pin   3;   "Latched M/IO#
    LDC         pin   4;   "Latched D/C#
    LWR         pin   5;   "Latched W/R#
    LALE        pin   6;   "Latched ALE
    L510CS~     pin   7;   "82510 Chip Select
    L59CS~      pin   8;   "8259A-2 Chip Select
    LEPROM~     pin   9;   "EPROM Chip Select
    OEN~        pin  11;   "PAL Output Enable
    rdy~        pin  12;   "I/O-EPROM Ready (n-1)
    rdy510~     pin  19;   "I/O-EPROM Ready (n-2)

" Outputs

    recovery    pin  13;   "I/O Recovery Time
    s1          pin  14;   "
    s0          pin  15;   "
    IORC~       pin  16;   "I/O Read Command
    IOWC~       pin  17;   "I/O Write Command
    MRDC~       pin  18;   "Memory (EPROM) Read Command

    rstate      =  [recovery, s1, s0];
    idle        =  [  0     , 1 , 0 ];
    active      =  [  0     , 1 , 1 ];
    inactive_0  =  [  1     , 1 , 1 ];
    inactive_1  =  [  1     , 0 , 1 ];
    inactive_2  =  [  1     , 0 , 0 ];
    inactive_3  =  [  1     , 1 , 0 ];
    illegal_a   =  [  0     , 0 , 0 ];
    illegal_b   =  [  0     , 0 , 1 ];


state_diagram rstate

    state idle:         if (!IORC~ # !IOWC~) then active else idle;
    state active:       if (IORC~ # IOWC~) then inactive_0 else active;
    state inactive_0:   goto inactive_1;
    state inactive_1:   goto inactive_2;
    state inactive_2:   goto inactive_3;
    state inactive_3:   goto idle;
    state illegal_a:    goto idle;
    state illegal_b:    goto idle;
```

240725–B4

```
state_diagram [IOWC~]

    state [1]: if (!recovery & !LMIO & LDC & LWR & (!L510CS~ # !L59CS~))
               then [0] else [1];
    state [0]: if RESET then [1] else
               if (!L510CS~ & !rdy510~ # !rdy~) then [1] else [0];

state_diagram [IORC~]

    state [1]: if (!recovery & !LMIO & LDC & !LWR & (!L510CS~ # !L59CS~))
               then [0] else [1];
    state [0]: if RESET then [1] else
               if !rdy~ then [1] else [0];

state_diagram [MRDC~]

    state [1]: if (LALE & LMIO & !LWR & !LEPROM~) then [0] else [1];
    state [0]: if RESET then [1] else
               if !rdy~ then [1] else [0];

end IO_CTRL_2;

^Z
```

240725–B5

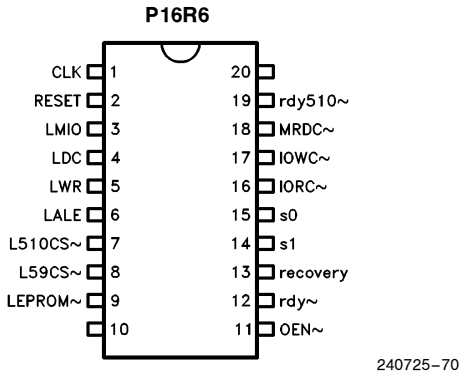
PAL Codes: IO-2

```
ABEL(tm) 3.10  -  Document Generator            14-Feb-90 09:34 AM
IO BUS CONTROLLER - PAL 2, INTEL CORPORATION
Equations for Module IO_CTRL_2

Device IO2


- Reduced Equations:

    !recovery := (!recovery & !s1 # !IORC~ & !IOWC~ & !recovery # !s0 & s1);

    !s1 := (recovery & s0);

    !s0 := (recovery & !s0 # !s1 # IORC~ & IOWC~ & !s0);

    !IOWC~ := (!IOWC~ & !RESET & rdy510~ & rdy~
               # !IOWC~ & L510CS~ & !RESET & rdy~
               # IOWC~ & !L59CS~ & LDC & !LMIO & LWR & !recovery
               # IOWC~ & !L510CS~ & LDC & !LMIO & LWR & !recovery);


    !IORC~ := (!IORC~ & !RESET & rdy~
               # IORC~ & !L59CS~ & LDC & !LMIO & !LWR & !recovery
               # IORC~ & !L510CS~ & LDC & !LMIO & !LWR & !recovery);


    !MRDC~ := (!MRDC~ & !RESET & rdy~
               # LALE & !LEPROM~ & LMIO & !LWR & MRDC~);
                                                                          240725-B6
```

PAL Codes: IO-2 (Continued)

ABEL 3.10—Document Generator 14-Feb-90 09:34 AM
IO BUS CONTROLLER—PAL 2, INTEL CORPORATION
Chip diagram for Module IO__CTRL__2

Device IO2



end of module IO__CTRL__2

PAL Codes: IO-2 (Continued)

```
module     IO_CTRL_3  flag '-r3'

title     'IO BUS CONTROLLER - PAL 2, INTEL CORPORATION'

    IO3    device      'P16R6';

    x           =     .X.;              " ABEL 'don't care' symbol
    c           =     .C.;              " ABEL 'clocking input' symbol

" Inputs

    CLK         pin   1;   "Processor Clock
    RESET       pin   2;   "System Reset
    LMIO        pin   3;   "Latched M/IO#
    LDC         pin   4;   "Latched D/C#
    LWR         pin   5;   "Latched W/R#
    LALE        pin   6;   "Latched ALE
    L510CS~     pin   7;   "82510 Chip Select
    L59CS~      pin   8;   "8259A-2 Chip Select
    LEPROM~     pin   9;   "EPROM Chip Select
    OEN~        pin  11;   "PAL Output Enable
    rdy~        pin  12;   "I/O-EPROM Ready (n-1)
    IORDY~      pin  19;   "I/O-EPROM Ready

" Outputs

    INTA~       pin  13;   "Interrupt Acknowledge
    st0         pin  14;   "
    DEN~        pin  15;   "I/O Bus Transceiver Enable
    st1         pin  16;   "
    DTR         pin  17;   "I/O Bus Transceiver Direction
    st2         pin  18;   "

state_diagram [INTA~, st0]

    state [1, 1]:  if (!LMIO & !LDC & !LWR & LALE) then [1, 0] else [1, 1];
    state [1, 0]:  if RESET then [1, 1] else
                   if !LALE then [0, 0] else [1, 0];
    state [0, 0]:  if RESET then [1, 1] else
                   if !rdy~ then [1, 1] else [0, 0];
    state [0, 1]:  goto [1, 1];

state_diagram [DEN~, st1]

    state [1, 1]:  if LALE & (!LEPROM~ # !L510CS~ # !L59CS~) then [1, 0] else

                   if !INTA~ then [0, 0] else [1, 1];
    state [1, 0]:  if RESET then [1, 1] else
                   if !LALE then [0, 0] else [1, 0];
    state [0, 0]:  if RESET then [1, 1] else
                   if !rdy~ then [1, 1] else [0, 0];
    state [0, 1]:  goto [1, 1];

state_diagram [DTR, st2]
```

240725–B7

```
    state [1, 1]:  if LALE & (!LEPROM~ # !L510CS~ # !L59CS~) & LWR then [0, 1]

                   else [1, 1];
    state [0, 1]:  if RESET then [1, 1] else
                   if !IORDY~ then [0, 0] else [0, 1];
    state [0, 0]:  goto [1, 1];
    state [1, 0]:  goto [1, 1];

end IO_CTRL_3;
^Z
```

240725–B8

PAL Codes: IO-3

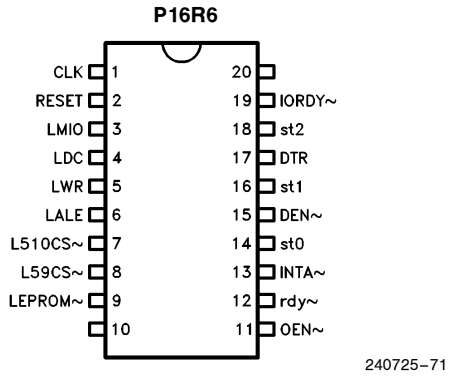
```
ABEL(tm) 3.10  -  Document Generator            15-Feb-90 06:45 PM
IO BUS CONTROLLER - PAL 2, INTEL CORPORATION
Equations for Module IO_CTRL_3

Device IO3


- Reduced Equations:

    !INTA~ := (!INTA~ & !RESET & rdy~ & !st0
               # INTA~ & !LALE & !RESET & !st0);


    !st0 := (!RESET & rdy~ & !st0
             # INTA~ & !RESET & !st0
             # INTA~ & LALE & !LDC & !LMIO & !LWR & st0);


    !DEN~ := (!DEN~ & !RESET & rdy~ & !st1
              # DEN~ & !LALE & !RESET & !st1
              # DEN~ & !INTA~ & L51OCS~ & L59CS~ & LEPROM~ & st1
              # DEN~ & !INTA~ & !LALE & st1);


    !st1 := (!RESET & rdy~ & !st1
             # DEN~ & !RESET & !st1
             # DEN~ & !INTA~ & st1
             # DEN~ & !L59CS~ & LALE & st1
             # DEN~ & !L51OCS~ & LALE & st1
             # DEN~ & LALE & !LEPROM~ & st1);


    !DTR := (!DTR & !RESET & st2
             # DTR & !L59CS~ & LALE & LWR & st2
             # DTR & !L51OCS~ & LALE & LWR & st2
             # DTR & LALE & !LEPROM~ & LWR & st2);


    !st2 := (!DTR & !IORDY~ & !RESET & st2);
```

240725–B9

PAL Codes: IO-3 (Continued)

ABEL 3.10—Document Generator 15-Feb-90 06:45 PM
IO BUS CONTROLLER—PAL 2, INTEL CORPORATION
Chip diagram for Module IO__CTRL__3

Device IO3



end of module IO__CTRL__3

PAL Codes: IO-3 (Continued)

```
module     IO_CTRL_4  flag '-r3'

title     'IO BUS CONTROLLER - PAL 2, INTEL CORPORATION'

    IO4    device      'P16R6';

    x           =     .X.;              " ABEL 'don't care' symbol
    c           =     .C.;              " ABEL 'clocking input' symbol

" Inputs

    CLK         pin   1;   "Processor Clock
    RESET       pin   2;   "System Reset
    LMIO        pin   3;   "Latched M/IO#
    LDC         pin   4;   "Latched D/C#
    LWR         pin   5;   "Latched W/R#
    LALE        pin   6;   "Latched ALE
    delay       pin   7;   "Delay Signal for Wait State Generation
    unused_0    pin   8;   "
    unused_1    pin   9;   "
    OEN~        pin  11;   "PAL Output Enable
    unused_3    pin  12;   "
    unused_4    pin  19;   "

" Outputs

    IORDY~      pin  13;   "I/O-EPROM Ready
    rdy~        pin  14;   "I/O-EPROM Ready (n-1)
    rdy5IO~     pin  15;   "I/O-EPROM Ready (n-2)
    nc_0        pin  16;   "
    nc_1        pin  17;   "
    nc_2        pin  18;   "

    rstate      =  [IORDY~, rdy~, rdy5IO~];
    idle        =  [   1  ,  1  ,    1   ];
    rdy2        =  [   1  ,  1  ,    0   ];
    rdy1        =  [   1  ,  0  ,    1   ];
    rdy0        =  [   0  ,  1  ,    1   ];
    illegal_a   =  [   1  ,  0  ,    0   ];
    illegal_b   =  [   0  ,  0  ,    0   ];
    illegal_c   =  [   0  ,  0  ,    1   ];
    illegal_d   =  [   0  ,  1  ,    0   ];

state_diagram rstate

    state idle:         if (LMIO & !LDC & LWR & LALE) then rdy1 else
                        if !delay then rdy2 else idle;
    state rdy2:         if RESET then idle else rdy1;
    state rdy1:         if RESET then idle else
                        if !LALE then rdy0 else rdy1;
    state rdy0:         goto idle;
    state illegal_a:    goto idle;
    state illegal_b:    goto idle;
    state illegal_c:    goto idle;
```

240725-C0

PAL Codes: IO-4

```
    state illegal_d:    goto idle;

end IO_CTRL_4;
^Z
```

240725-C1

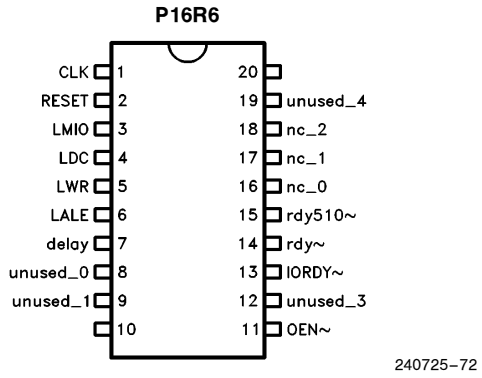
```
ABEL(tm) 3.10  -  Document Generator            15-Feb-90 06:55 PM
IO BUS CONTROLLER - PAL 2, INTEL CORPORATION
Equations for Module IO_CTRL_4

Device IO4


- Reduced Equations:

    !IORDY~ := (IORDY~ & !LALE & !RESET & rdy510~ & !rdy~);

    !rdy~ := (IORDY~ & LALE & !RESET & rdy510~ & !rdy~
              # IORDY~ & !RESET & !rdy510~ & rdy~
              # IORDY~ & LALE & !LDC & LMIO & LWR & rdy510~ & rdy~);


    !rdy510~ := (IORDY~ & !LALE & !delay & rdy510~ & rdy~
                 # IORDY~ & !LWR & !delay & rdy510~ & rdy~
                 # IORDY~ & LDC & !delay & rdy510~ & rdy~
                 # IORDY~ & !LMIO & !delay & rdy510~ & rdy~);
```

240725-C2

PAL Codes: IO-4 (Continued)

ABEL 3.10—Document Generator 15-Feb-90 06:55 PM
IO BUS CONTROLLER—PAL 2, INTEL CORPORATION
Chip diagram for Module IO__CTRL__4

Device IO4



end of module IO__CTRL__4

PAL Codes: IO-4 (Continued)

```
module         LADDR_DEC flag '-r3'

title   'LOCAL_DECODE_LOGIC - INTEL CORPORATION'

    LADDR_PAL     device       'P16L8';

    x = .X.;           "ABEL don't care symbol
    c = .C.;           "ABEL clocking input symbol
    h = 1;             "logic 1
    l = 0;             "logic 0

" Inputs

    ADS~    pin   1;   "ADS#
    M_IO~   pin   2;   "M/IO#
    A31     pin   3;   "Addr bit 31
    A30           pin   4;   "Addr bit 30
    A29     pin   5;   "Addr bit 29

" Outputs

    X16~    pin  18;   "indicates a 16-bit access
    LBA~    pin  17;   "local bus access
    NCA~    pin  16;   "non-cache access

equations

    !X16~ =  !ADS~ & M_IO~ & A31 & A30 & A29;
    LBA~  =  h;
    NCA~  =  h;


end LADDR_DEC;
```

240725–C3

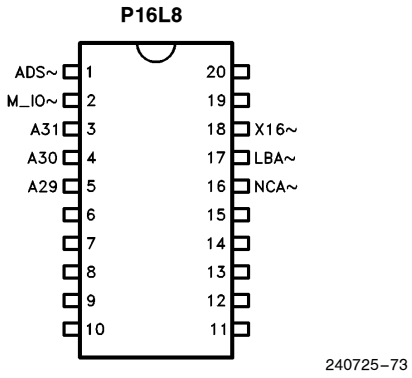
```
ABEL(tm) 3.10  -  Document Generator            14-Feb-90 09:51 AM
LOCAL_DECODE_LOGIC - INTEL CORPORATION
Equations for Module LADDR_DEC

Device LADDR_PAL


- Reduced Equations:

    !X16~ = (A29 & A30 & A31 & !ADS~ & M_IO~);

    !LBA~ = (0);

    !NCA~ = (0);
```

240725–C4

PAL Codes: Local Decoder

ABEL 3.10—Document Generator 14-Feb-90 09:51 AM
LOCAL__DECODE__LOGIC—INTEL CORPORATION
Chip diagram for Module LADDR__DEC

Device LADDR__PAL



end of module LADDR__DEC

PAL Codes: Local Decoder (Continued)

```
module         READY flag '-r3'

title   'READY_LOGIC - INTEL CORPORATION'

    RDY     device       'P16L8';

" Inputs

    DRAMRDY~       pin   1;   "DRAM READY#
    IORDY~   pin   2;   "IO/EPROM READY#
    RDYEN~   pin   3;   "RDYEN# of 82385
    RDY385~  pin   4;   "READYO# OF 82385
    RDY387~  pin   5;   "READYO# OF 82387
    CACHE    pin   6;   "High if cache exits; otherwise, Low

" Outputs

    READY~  pin  12;   "READY# for 80386
    BREADY~ pin  13;   "BREADY# for 82385

equations

    !BREADY~ = !DRAMRDY~ # !IORDY~;
    !READY~ = (CACHE & !RDY385~) # !RDY387~ #
              (CACHE & !RDYEN~ & (!DRAMRDY~ # !IORDY~) #
              !CACHE & (!DRAMRDY~ # !IORDY~));

end READY;
```

240725-C5

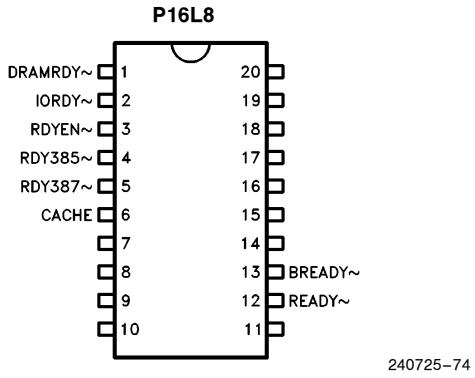
```
ABEL(tm) 3.10  -  Document Generator            15-Feb-90 07:02 PM
READY_LOGIC - INTEL CORPORATION
Equations for Module READY

Device RDY


- Reduced Equations:

    !BREADY~ = (!IORDY~ # !DRAMRDY~);

    !READY~ = (!CACHE & !IORDY~
               # !CACHE & !DRAMRDY~
               # !IORDY~ & !RDYEN~
               # !DRAMRDY~ & !RDYEN~
               # !RDY387~
               # CACHE & !RDY385~);
```

240725-C6

PAL Codes: Ready

ABEL 3.10—Document Generator 15-Feb-90 07:02 PM
READY__LOGIC—INTEL CORPORATION
Chip diagram for Module READY

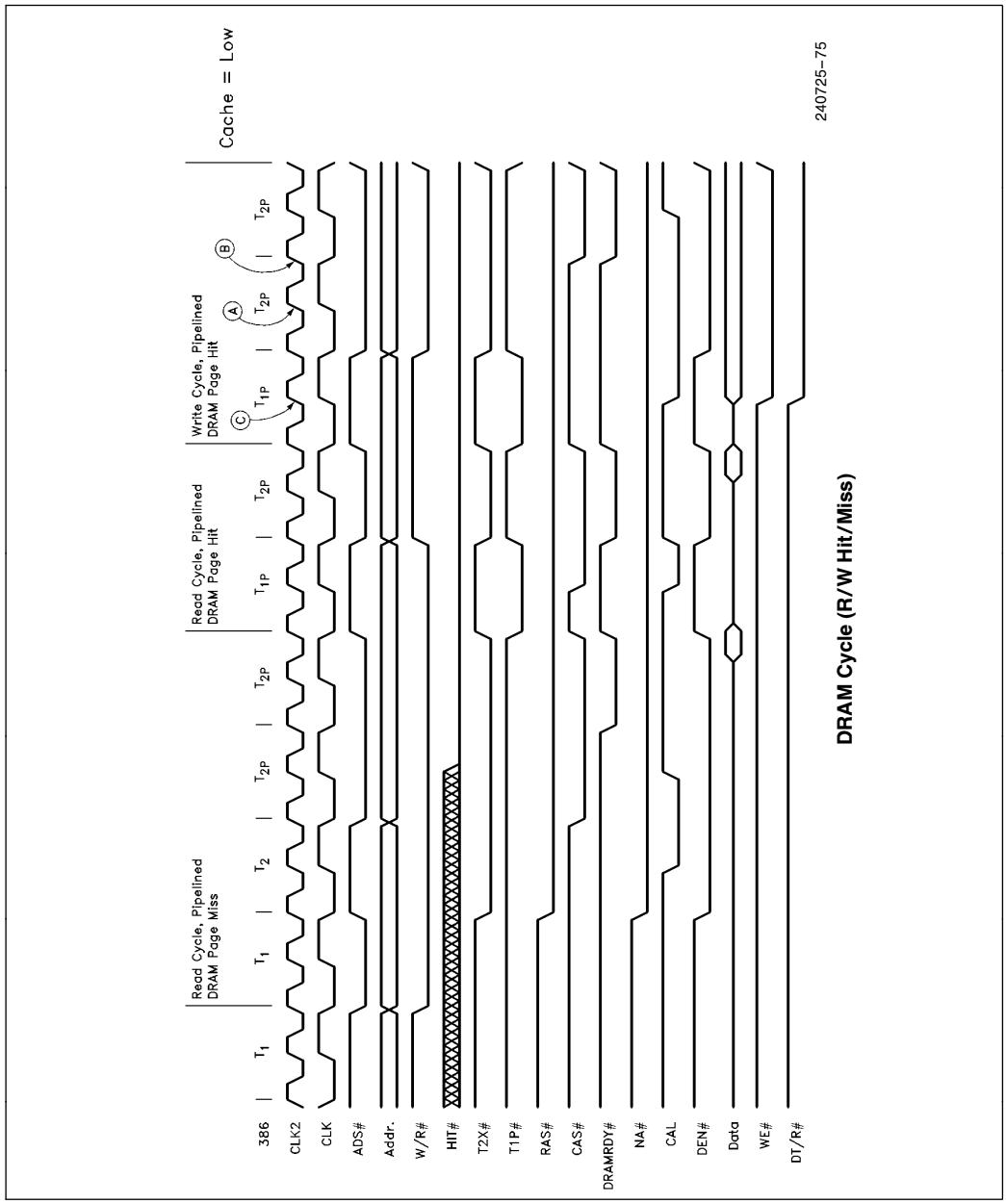
Device RDY



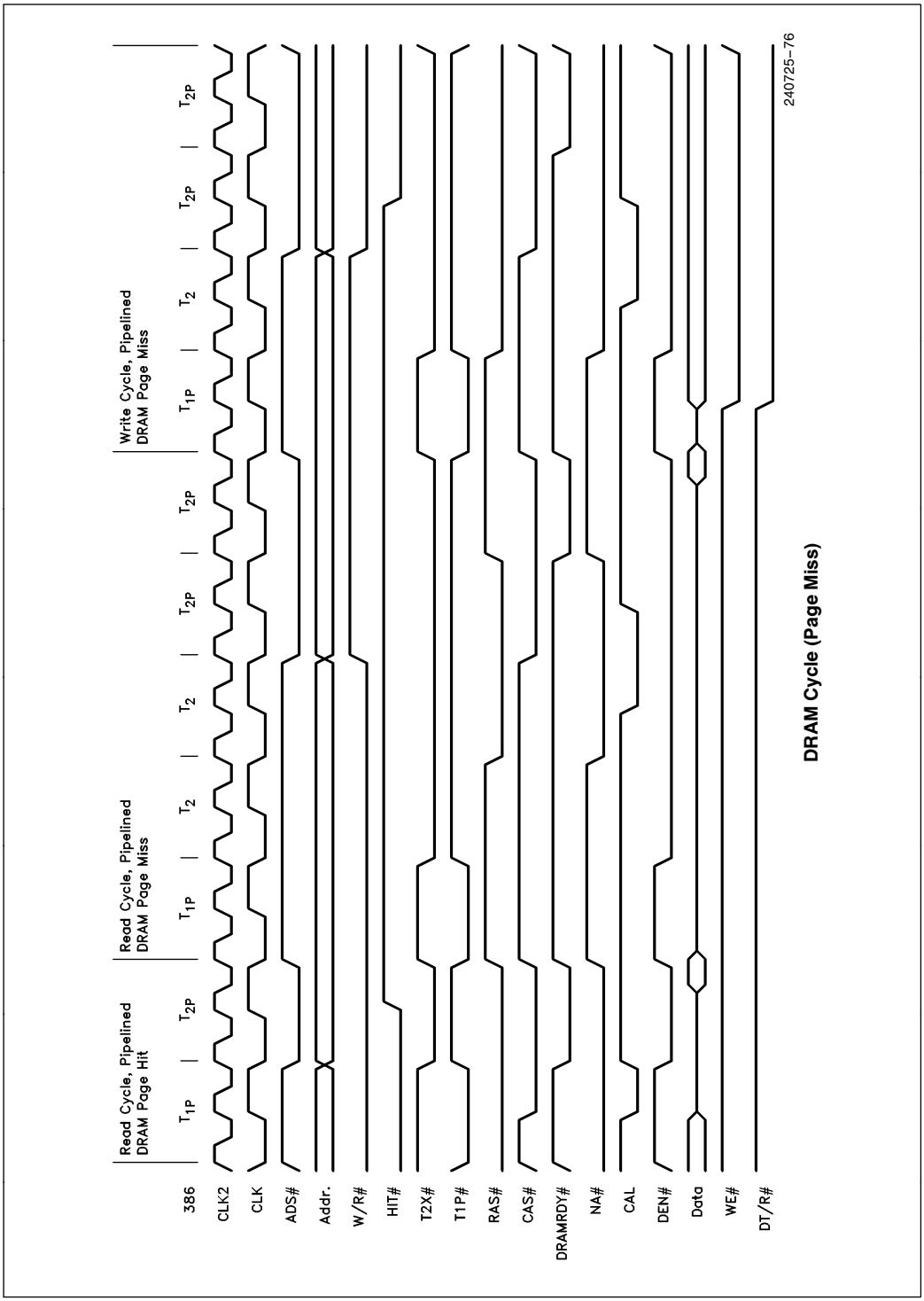
end of module READY

PAL Codes: Ready (Continued)

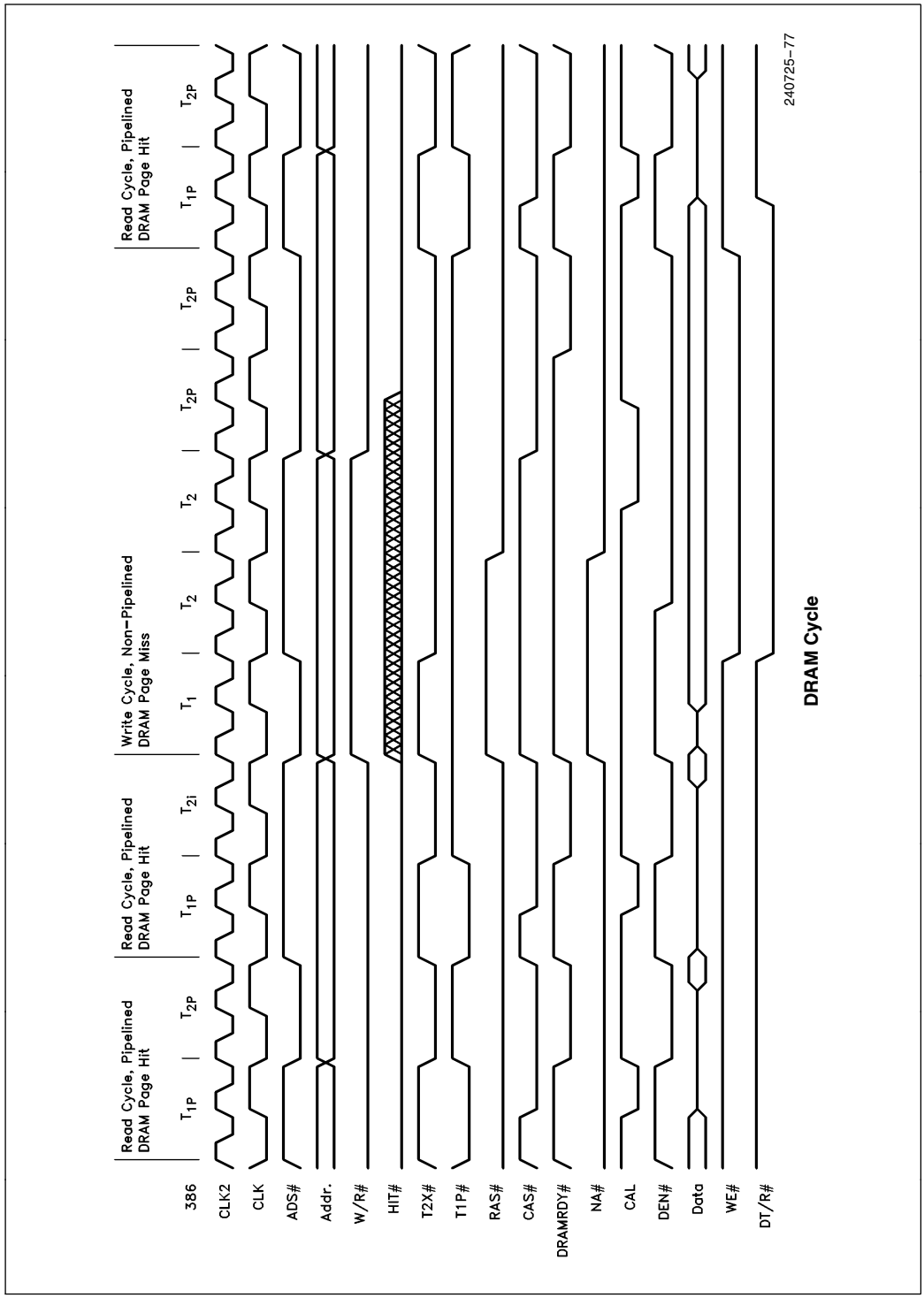
APPENDIX C
TIMING EQUATIONS



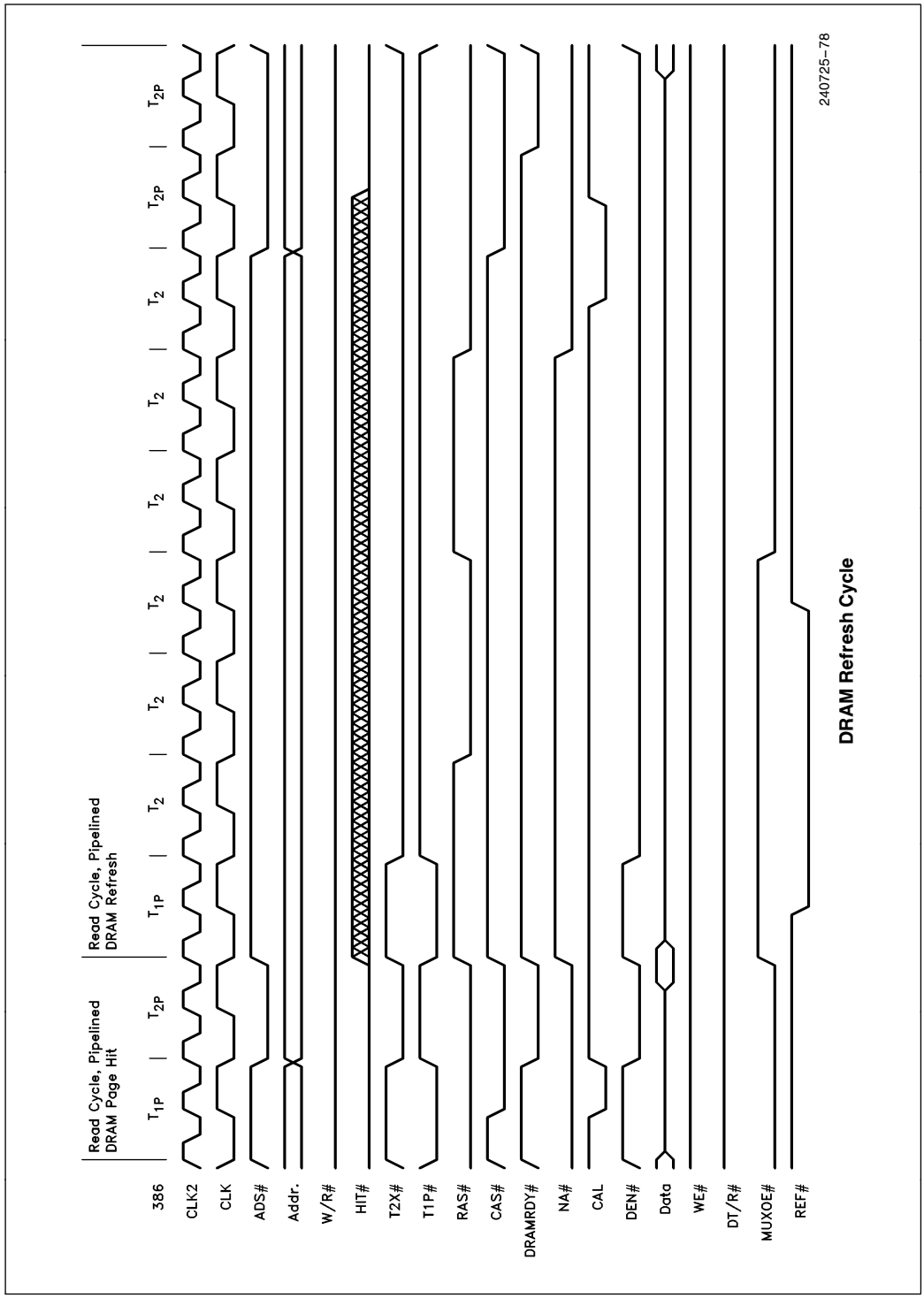
DRAM Cycle (R/W Hit/Miss)



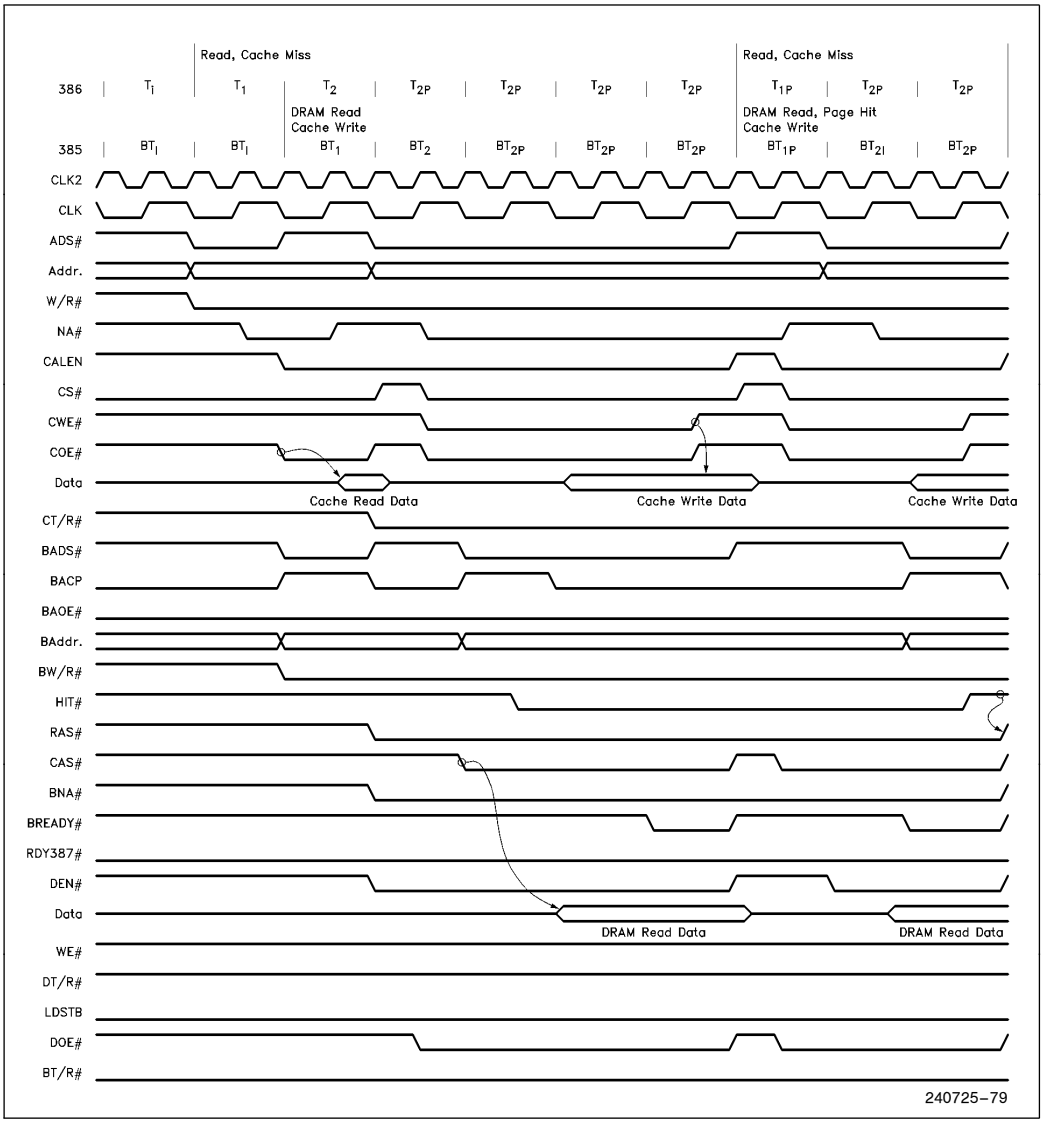
DRAM Cycle (Page Miss)


Read Cycle, Pipelined DRAM Page Hit
Read Cycle, Pipelined DRAM Page Hit
Write Cycle, Non-Pipelined DRAM Page Miss
Read Cycle, Pipelined DRAM Page Hit
386
CLK2
CLK
ADS#
Addr.
W/R#
HIT#
T2X#
T1P#
RAS#
CAS#
DRAMRDY#
NA#
CAL
DEN#
Data
WE#
DT/R#
240725-77

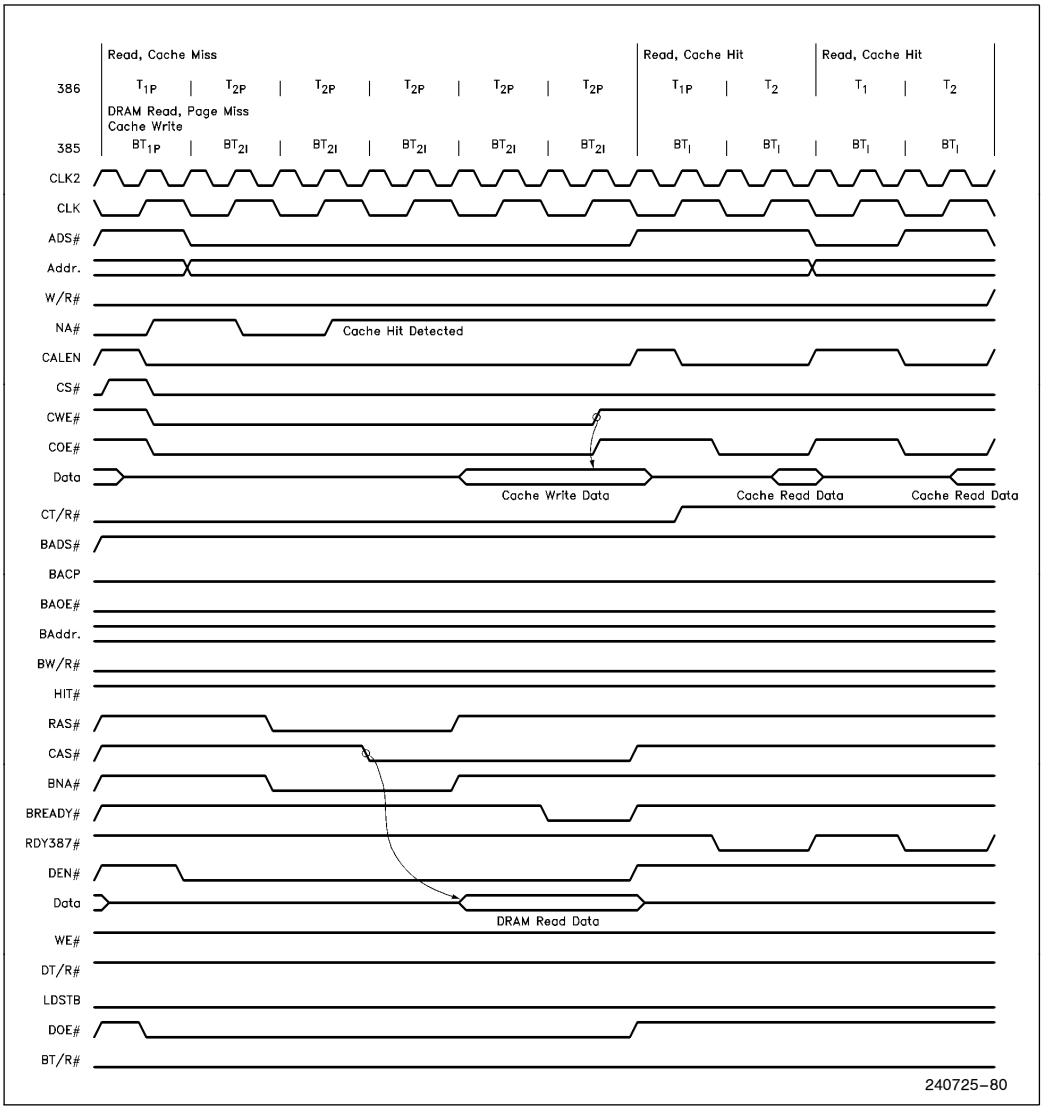
DRAM Cycle



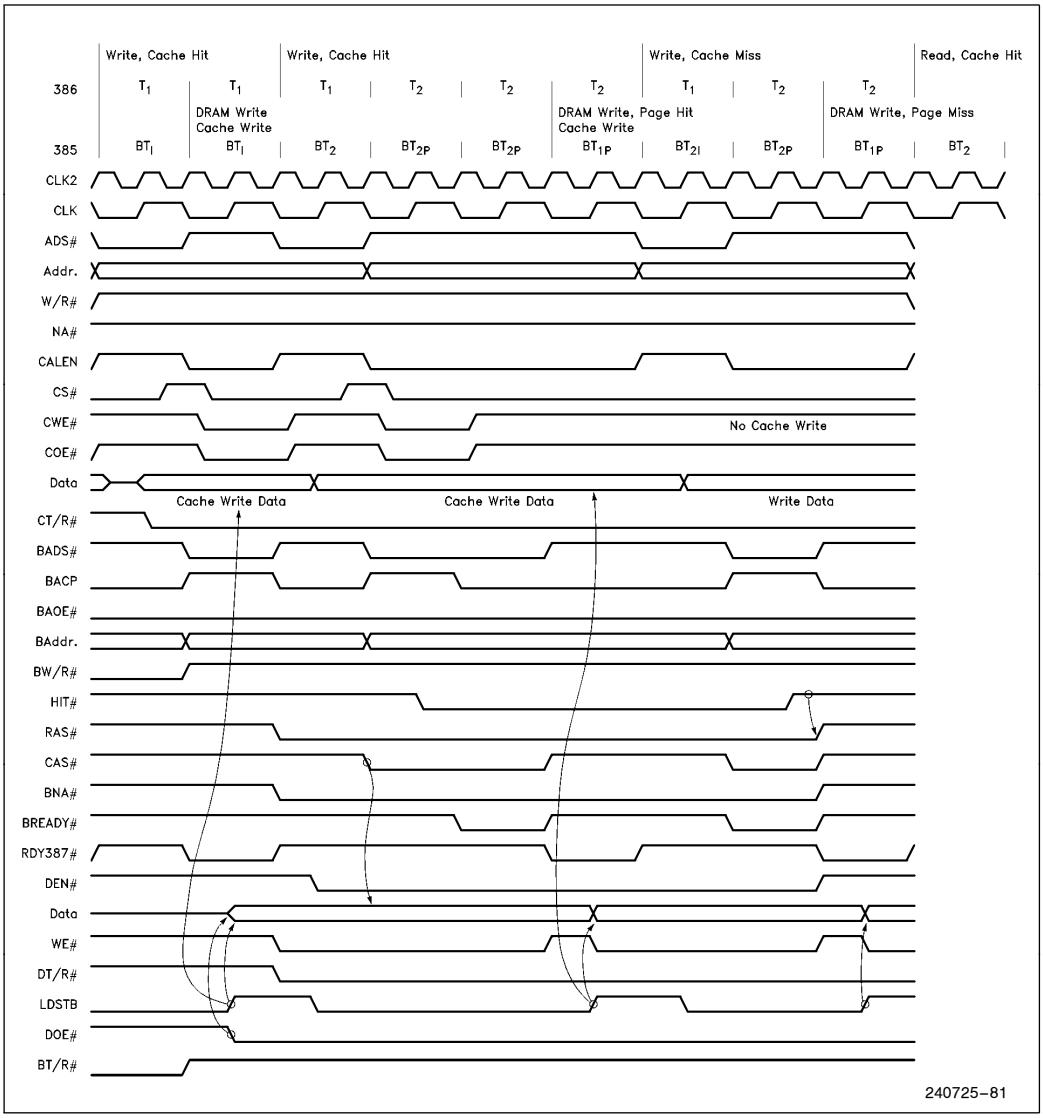
DRAM Refresh Cycle


Read, Cache Miss
Read, Cache Miss
386
T_I
T_1
T_2
T_{2P}
T_{2P}
T_{2P}
T_{2P}
T_{1P}
T_{2P}
T_{2P}
DRAM Read
Cache Write
DRAM Read, Page Hit
Cache Write
385
BT_I
BT_I
BT_1
BT_2
BT_{2P}
BT_{2P}
BT_{2P}
BT_{1P}
BT_{2I}
BT_{2P}
CLK2
CLK
ADS#
Addr.
W/R#
NA#
CALEN
CS#
CWE#
COE#
Data
Cache Read Data
Cache Write Data
Cache Write Data
CT/R#
BADS#
BACP
BAOE#
BAddr.
BW/R#
HIT#
RAS#
CAS#
BNA#
BREADY#
RDY387#
DEN#
Data
DRAM Read Data
DRAM Read Data
WE#
DT/R#
LDSTB
DOE#
BT/R#
240725-79

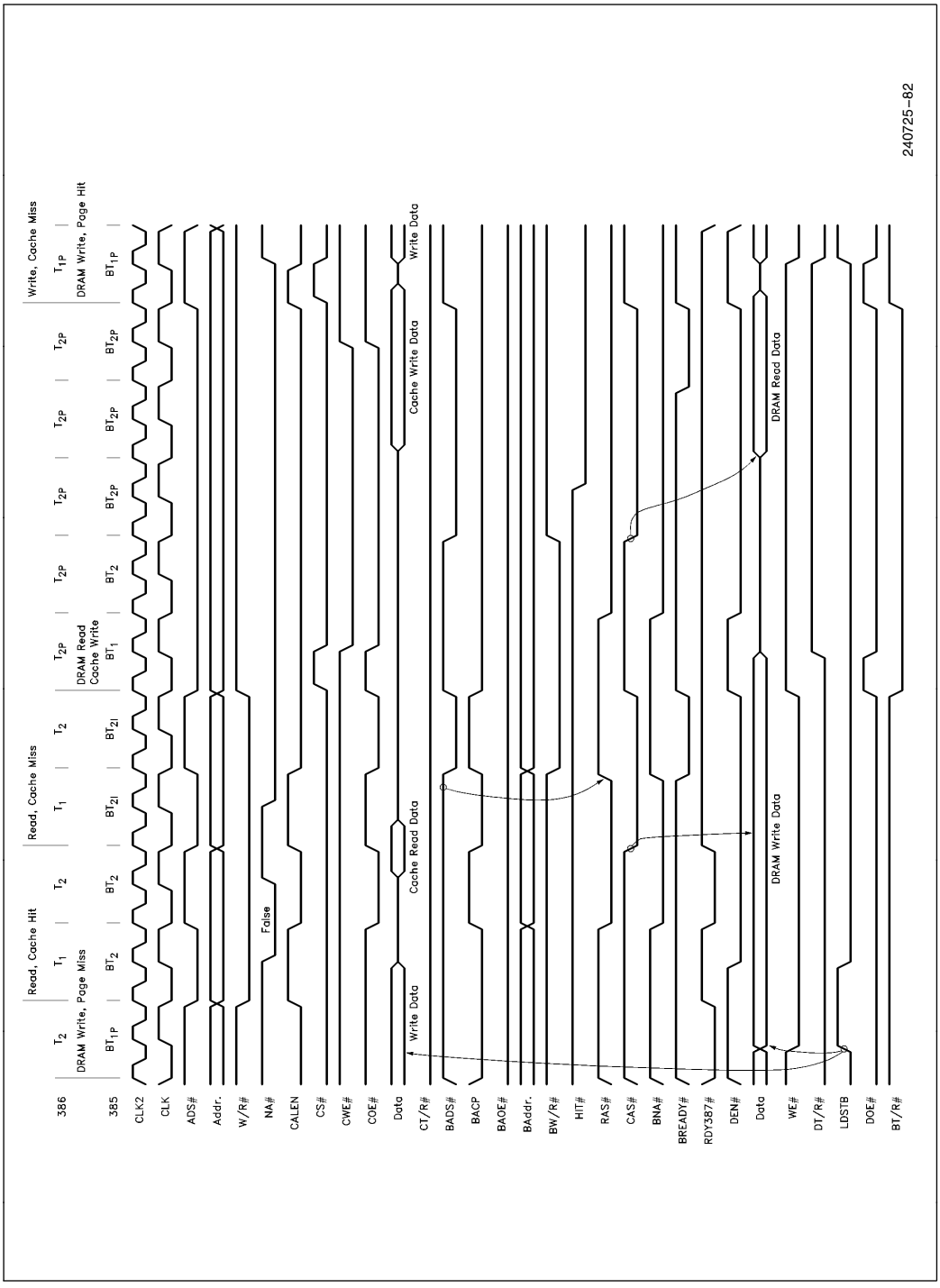
Cache Cycle



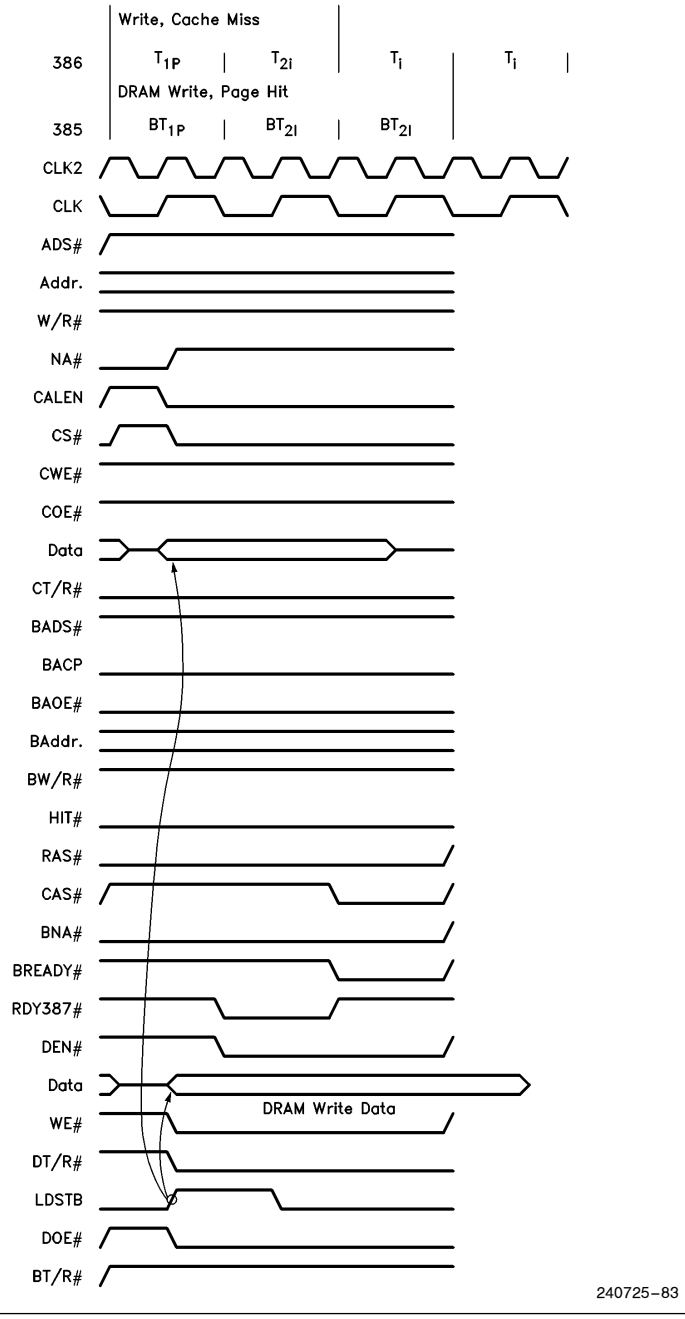
Cache Cycle (Continued)



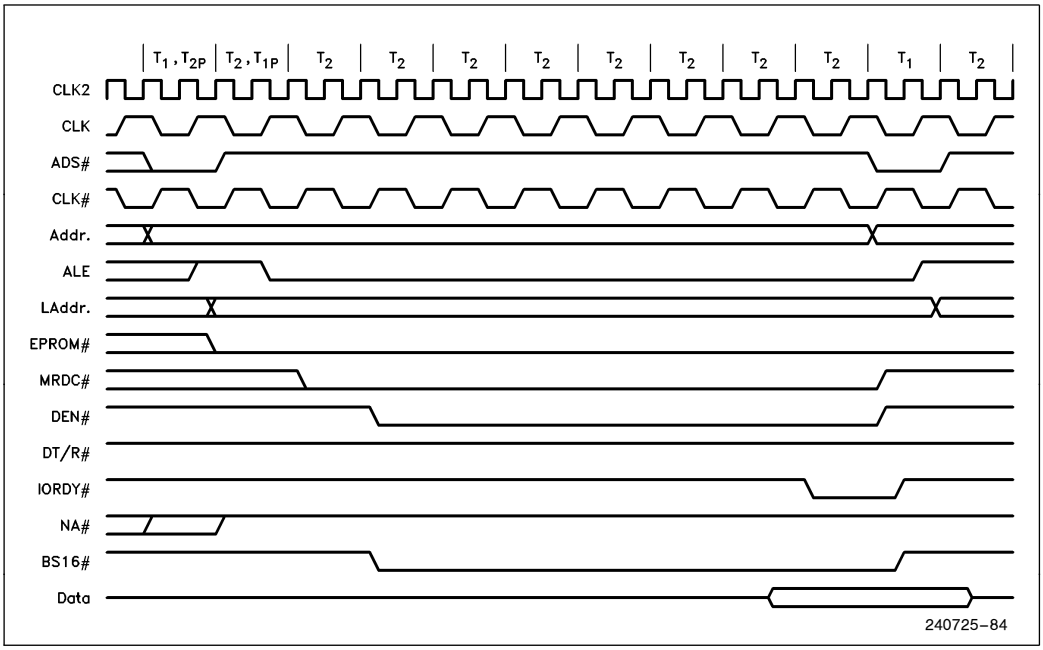
Cache Cycle (Continued)

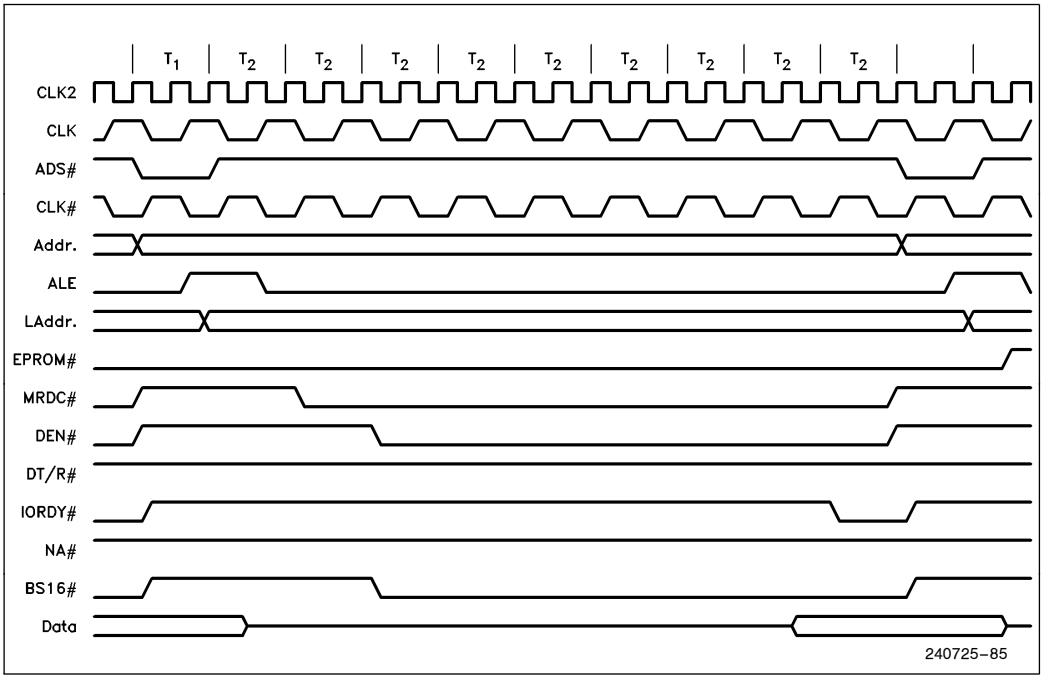


Cache Cycle (Continued)

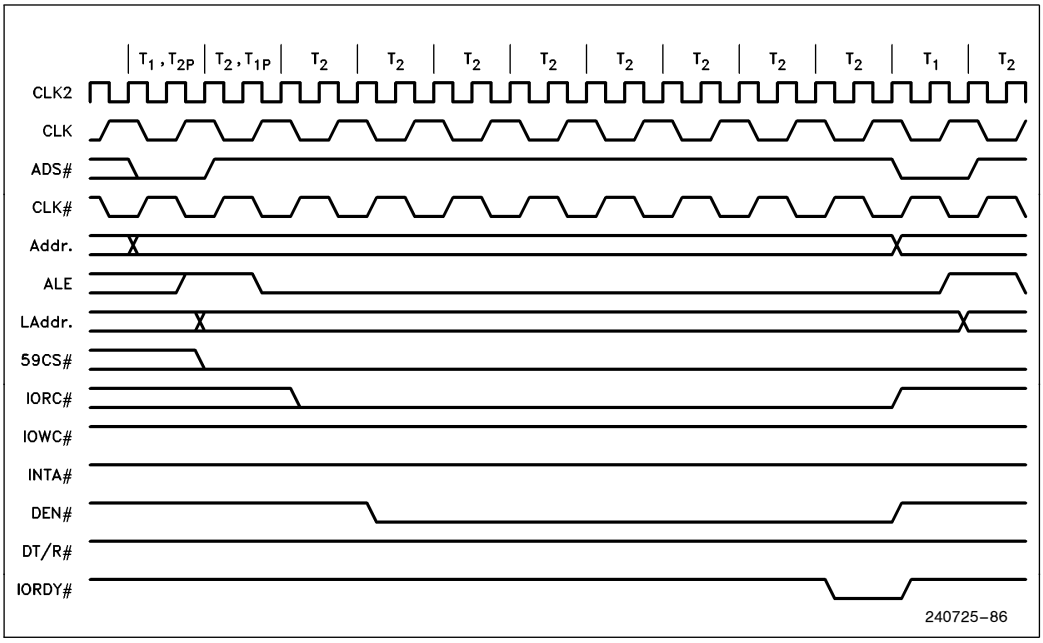


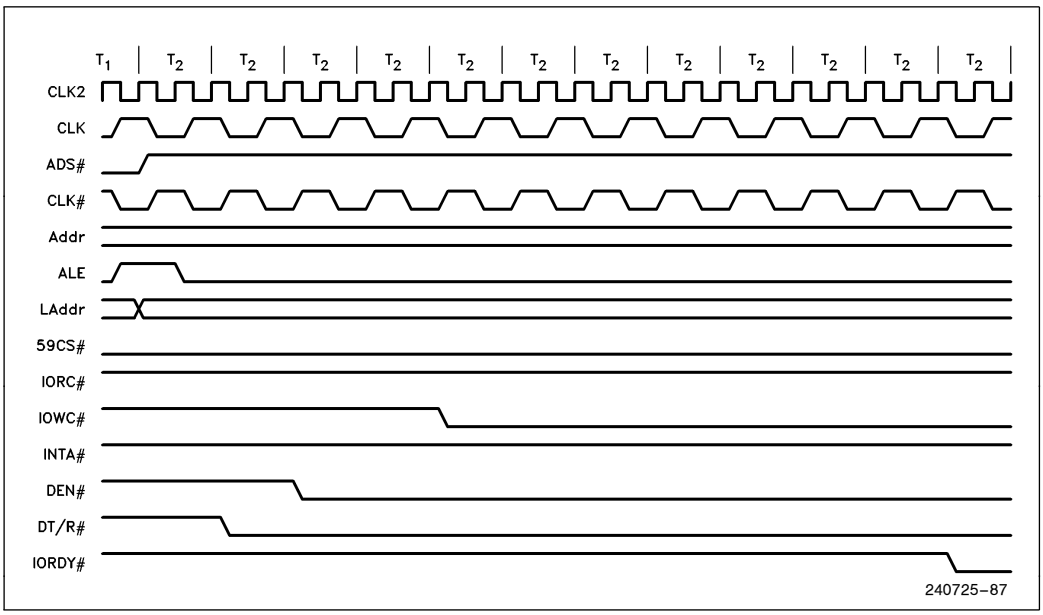
Cache Cycle (Continued)



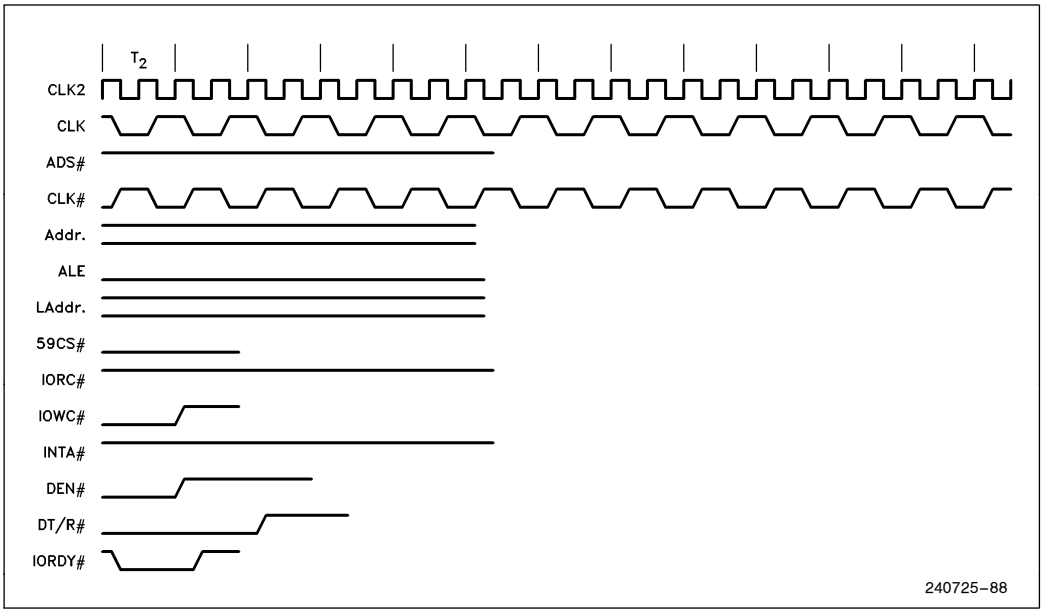


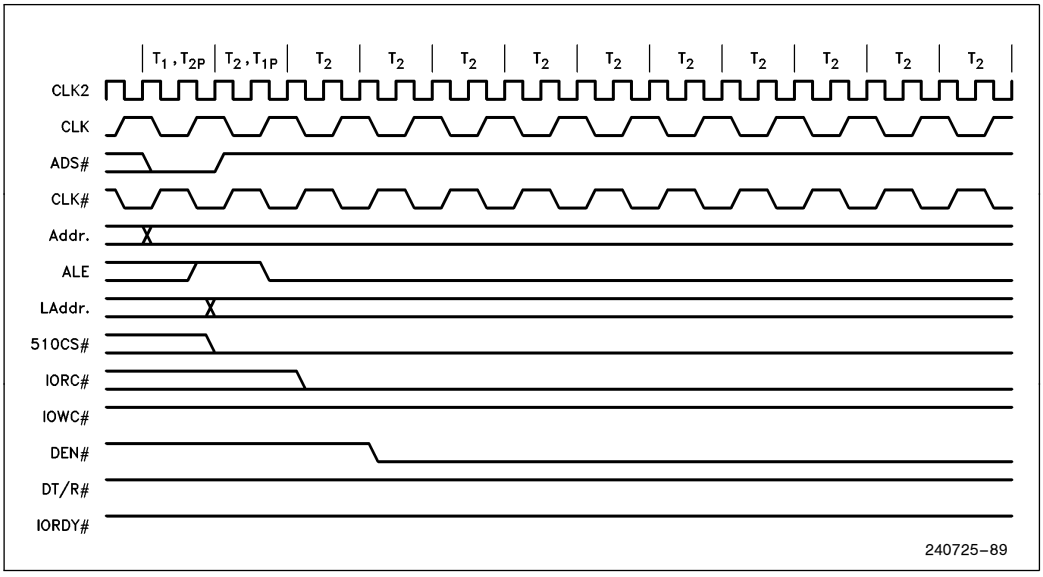
EPROM and I/O Cycles



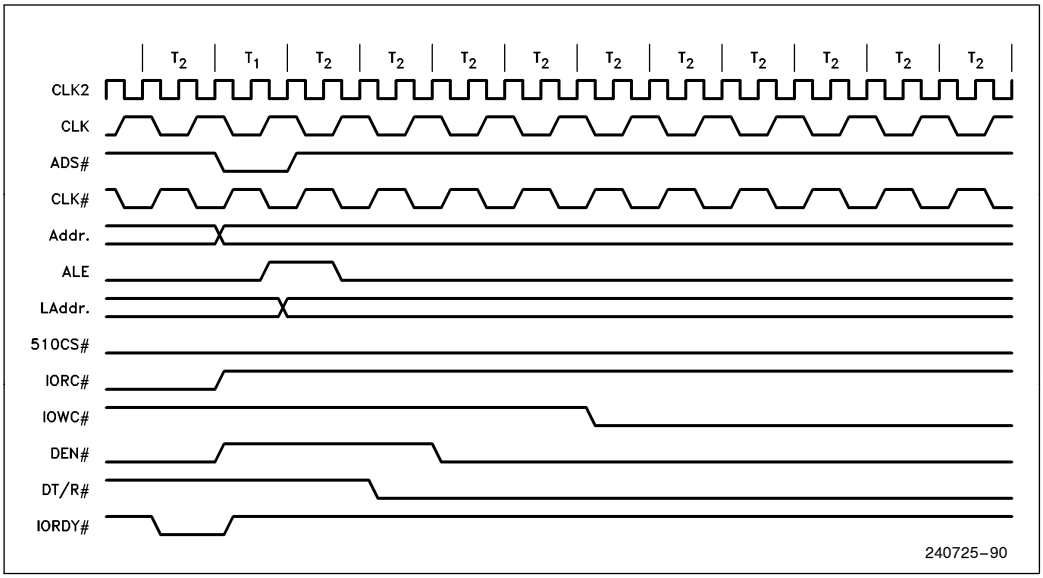


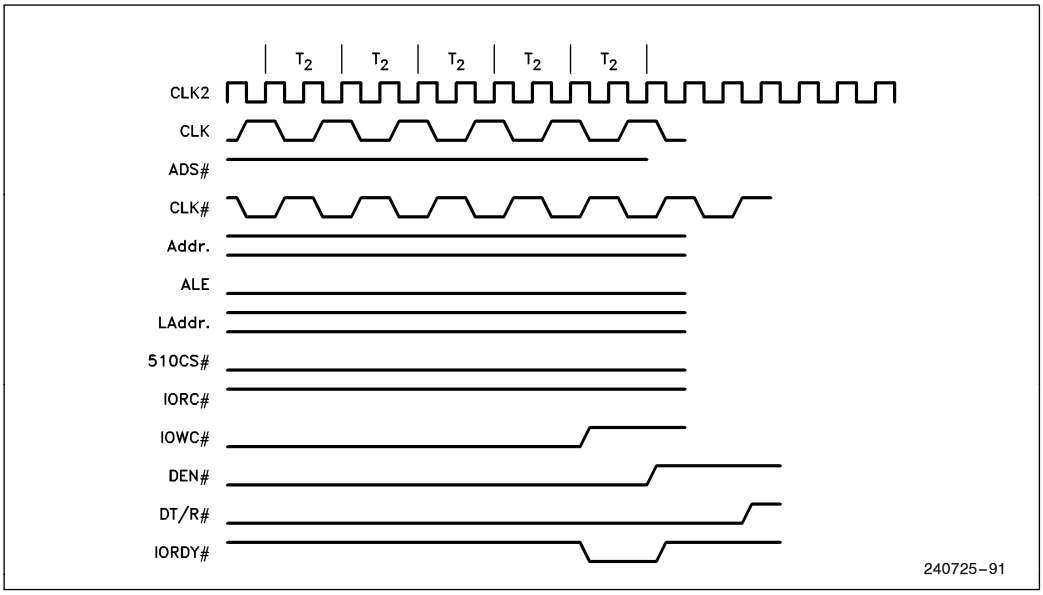
EPROM and I/O Cycles (Continued)





EPROM and I/O Cycles (Continued)





EPROM and I/O Cycles (Continued)

APPENDIX D
TIMING EQUATIONS

EQUATIONS FOR DRAM TIMINGS (NO CACHE CONFIGURATION):

Read and Write Cycles (Common Parameters):

tRC: Random Read or Write Cycle Time

CLK2 × 10

tRP: RAS# Precharge Time

CLK2 × 4

tRAS: RAS# Pulse Width

CLK2 × 4

A random DRAM cycle may have a RAS# pulse which is only four CLK2 periods wide. This is the case if the cycle is followed by Idle cycles (DRAMs not selected or Ti's) or a DRAM page miss.

tCAS (Read): CAS# Pulse Width

CLK2 × 3

CAS# pulses can be as narrow as three CLK2 cycles during Page Mode read cycles.

tCAS (Write): CAS# Pulse Width

CLK2 × 2

CAS# pulses can be as narrow as two CLK2 cycles during Page Mode write cycles.

tASC: Column Address Setup Time

min (CLK2 × 2 + AS32.tphl.min − Delay.max − ACT258.StoZ.tpl.max − ACT258.Cap.Derating, CLK2 × 3 + AS32.tphl.min − t6.max − 386.Cap.Derating − AS373.DtoO.tpd.max − ACT258.ItoZ.tpl.max − ACT258.Cap.Derating)

The Column Address becomes valid as RAS# switches from High to Low or as the 386 address becomes valid while RAS# is already Low (i.e., Page Mode, Pipelined cycles)

tCAH: Column Address Hold Time

CLK2 + AS373.GtoO.tpd.min + ACT258.ItoZ.tpl.min − AS32.tphl.max

The CAL (Column Address Latch) signal is activated one CLK2 period after the active-going edge of CAS#.

tAR: Column Address Hold Time to RAS#

CLK2 × 3 + AS373.GtoO.tpd.min + ACT258.ItoZ.tpl.min − RAS.Delay.max

tRCD: RAS# to CAS# Delay Time

CLK2 × 2 + AS32.tphl.min − RAS.Delay.max

tRAD: RAS# to Column Address Delay Time

(min) ACT258.StoZ.tphl.min + Delay.min − RAS.Delay.max

(max) ACT258.StoZ.tphl.max + Delay.max + ACT258.Cap.Derating − RAS.Delay.min

tRSH: RAS# Hold Time

CLK2 × 2 − AS32.tphl.max + RAS.Delay.min

The worst case occurs when a DRAM Page miss or Idle is detected at the end of the current DRAM Page miss cycle.

tCSH: CAS# Hold Time

CLK2 × 6 + AS32.tplh.min − RAS.Delay.max

tCRP: CAS# to RAS# Precharge Time

CLK2 × 2 + RAS.Delay.min − AS32.tplh.max

This is guaranteed by the DRAM control state machine.

tASR: Row Address Setup Time

CLK2 × 2 − t6.max − 386.Cap.Derating − ACT258.ItoZ.max − ACT258.Cap.Derating + H124.tpd.min + H125.tpd.min + PAL.tco.min + RAS.Delay.min

tRAH: Row Address Hold Time

ACT258.StoZ.tphl.min + Delay.min − RAS.Delay.max

tT: Transition Time (Rise and Fall)

tREF: Refresh Period

tREF2: Refresh Period

Read Cycles:

tRAC: Access Time

CLK2 × 6 − H124.tpd.max − H125.tpd.max − PAL.tco.max − t21.min − F245.max − RAS.Delay.max

tCAC: Access Time from CAS#

CLK2 × 3 − H124.tpd.max − H125.tpd.max − PAL.tco.max − AS32.tphl.max − t21.min − F245.max

tAA: Access Time from Address

CLK2 × 6 − t6.max − 386.Cap.Derating − AS373.DtoO.max − ACT258.ItoZ.tp.max − ACT258.Cap.Derating − t21.min − F245.max

tRCS: Read Command Setup Time

CLK2 + AS32.tphl.min

tRCH: Read Command Hold Time to CAS#

CLK2 − AS32.tplh.max

tRRH: Read Command Hold Time to RAS#

CLK2 − RAS.Delay.max

tOFF: Output Buffer Turn-off Time

CLK2 × 2 + F245.tzh.min

Write Cycles:

tWCS: Write Command Setup Time

CLK2 × 3 + AS32.tphl.min

tWCH: Write Command Hold Time

CLK2 × 2 − AS32.tplh.max

tWCR: Write Command Hold Time to RAS#

CLK2 × 6 − RAS.Delay.max

tWP: Write Command Pulse Width

CLK2 × 5

tRWL: Write Command to RAS# Lead Time

CLK2 × 5 + RAS.Delay.min

tCWL: Write Command to CAS# Lead Time

CLK2 × 5

tDS: Data-in Setup Time

CLK2 × 3 + H124.tp.min + H125.tp.min + AS32.tphl.min − T12.max − F245.tp.max

tDH: Data-in Hold Time

CLK2 × 2 + F245.tpz.min − AS32.tphl.max

tDHR: Data-in Hold Time to RAS#

CLK2 × 6 + F245.tpz.max + RAS.Delay.min

Page Mode Cycles:

tPC: Page Mode Cycle Time

CLK2 × 4

tRAPC: Page Mode RAS# Pulse Width

CLK2 × 4

tRSW: RAS# to Second WE# Delay Time

CLK2 × 7 − RAS.Delay.max

tCP: CAS# Precharge Time

CLK2

tWI: Write Invalid Time

CLK2

tCAP: Access Time from Column Precharge Time

CLK2 × 4 − H124.tp.max − H125.tp.max − PAL.tco.max − t21.min − F245.max

80386 A.C. SPECIFICATIONS

Symbol	Parameter	80386-33 Minimum	Maximum
	Operating Frequency	8.00	33.33
t1	CLK2 Period	15.00	62.50
t2a	CLK2 High Time	6.25	
t2b	CLK2 High Time	4.50	
t3a	CLK2 Low Time	6.25	
t3b	CLK2 Low Time	4.50	
t4	CLK2 Fall Time		4.00
t5	CLK2 Rise Time		4.00
t6	A2-A31 Valid Delay	4.00	15.00
t7	A2-A31 Float Delay	4.00	20.00
t8	BE0#-BE3#, LOCK# Valid Delay	4.00	15.00
t9	BE0#-BE3#, LOCK# Float Delay	4.00	20.00
t10	W/R#, M/IO#, D/C#, ADS# Valid Delay	4.00	15.00
t11	W/R#, M/IO#, D/C#, ADS# Float Delay	4.00	25.00
t12	D0-D31 Write Data Valid Delay	5.00	24.00
t13	D0-D31 Float Delay	4.00	17.00
t14	HLDA Valid Delay	4.00	20.00
t15	NA# Setup Time	5.00	
t16	NA# Hold Time	3.00	
t17	BS16# Setup Time	5.00	
t18	BS16# Hold Time	3.00	
t19	Ready# Setup Time	7.00	
t20	Ready# Hold Time	4.00	
t21	D0-D31 Read Setup Time	5.00	
t22	D0-D31 Read Hold Time	3.00	
t23	HOLD Setup Time	11.00	
t24	HOLD Hold Time	3.00	
t25	RESET Setup Time	8.00	
t26	RESET Hold Time	3.00	
t27	NMI, INTR Setup Time	5.00	
t28	NMI, INTR Hold Time	5.00	
t29	PEREQ, ERROR#, BUSY# Setup Time	5.00	
t30	PEREQ, ERROR#, BUSY# Hold Time	4.00	

PAL SPECIFICATIONS

Symbol	Parameter	Minimum	Maximum
ts	Input or Feedback Setup Time	7.00	
tco	Clock to Output	3.00	6.50

ROW ADDRESS LATCH SPECIFICATIONS
74FCT843B (IDT)

Symbol	Parameter	50 pF Minimum	Maximum
tplh	Dn to On Propagation Delay	3.00	6.50
tphl		3.00	6.50
tplh	G to On Propagation Delay	6.00	8.00
tphl		4.00	8.00
ts	Setup Time	2.00	
th	Hold Time	3.00	

240725–C7

Timings for No Cache Configuration

ROW ADDRESS COMPARATOR SPECIFICATIONS
74PCT521B (Performance)

Symbol	Parameter	Minimum	Maximum
tplh	An or Bn to Q Propagation Delay	1.50	5.50
tphl		1.50	5.50
tplh	I to Q Propagation Delay	1.50	4.60
tphl		1.50	4.60

DRAM ADDRESS MULTIPLEXER SPECIFICATIONS
74ACT258

Symbol	Parameter	Minimum	Maximum
tplh	S to Zn Propagation Delay	1.00	11.50
tphl		1.00	11.00
tplh	E# to Zn Propagation Delay	1.00	9.50
tphl		1.00	9.50
tplh	In to Zn Propagation Delay	1.00	9.50
tphl		1.00	8.00

DATA TRANSCEIVER SPECIFICATIONS
74F245

Symbol	Parameter	Minimum	Maximum
tplh	An to Bn or Bn to An Propagation Delay	2.50	7.00
tphl		2.50	7.00
tzh	Output Enable Time	3.00	8.00
tzl		3.50	9.00
tphz	Output Disable Time	3.00	7.50
tplz		2.00	7.50

COLUMN ADDRESS LATCH SPECIFICATIONS
74AS573

Symbol	Parameter	Minimum	Maximum
tplh	Dn to On Propagation Delay	3.00	6.00
tphl		3.00	6.00
tplh	G to On Propagation Delay	6.00	11.50
tphl		4.00	7.50
ts	Setup Time	2.00	
th	Hold Time	3.00	

RAS# DELAY

Symbol	Parameter	Minimum	Maximum
tp	Propagation Delay	0.00	0.00

240725–C8

Timings for No Cache Configuration (Continued)

OR SPECIFICATIONS
74AS32

Symbol	Parameter	Minimum	Maximum
tplh	Propagation Delay	1.00	5.80
tphl		1.00	5.80

DRAM TIMING REQUIREMENTS

Symbol	Parameter	For 80386-33 Minimum	For 80386-33 Maximum	Timing Margin (NMB 2801-06) Minimum	Timing Margin (NMB 2801-06) Maximum
Read and Write Cycles (Common Parameters):					
tRC	Random Read or Write Cycle Time	150.00		29.00	
tRP	RAS# Precharge Time	60.00		5.00	
tRAS	RAS# Pulse Width	60.00		0.00	
tCAS	CAS# Pulse Width (Read)	45.00		34.00	
tCAS	CAS# Pulse Width (Write)	30.00		25.00	
tASC	Column Address Setup Time	9.70		9.70	
tCAH	Column Address Hold Time	14.20		8.20	
tAR	Column Address Hold Time to RAS#	50.00		10.00	
tRCD	RAS# to CAS# Delay Time	31.00		25.00	14.00
tRAD	RAS# to Column Address Delay Time	5.00	21.30	1.00	6.70
tRSH	RAS# Hold Time	24.20		9.20	
tCSH	CAS# Hold Time	91.00		51.00	
tCRP	CAS# to RAS# Precharge Time	24.20		21.20	
tASR	Row Address Setup Time	5.45		3.45	
tRAH	Row Address Hold Time	5.00		3.00	
tT	Transition Time (Rise and Fall)				
tREF	Refresh Period				
tREF2	Refresh Period				
Read Cycles:					
tRAC	Access Time	68.25		8.25	
tCAC	Access Time from CAS#	17.45		6.45	
tAA	Access Time from Address	41.20		9.20	
tRCS	Read Command Setup Time	16.00		16.00	
tRCH	Read Command Hold Time to CAS#	9.20		9.20	
tRRH	Read Command Hold Time to RAS#	15.00		15.00	
tOFF	Output Buffer Turn-off Time		33.00		16.00
Write Cycles:					
tWCS	Write Command Setup Time	46.00		46.00	
tWCH	Write Command Hold Time	24.20		19.20	
tWCR	Write Command Hold Time to RAS#	90.00		50.00	
tWP	Write Command Pulse Width	75.00		70.00	
tRWL	Write Command to RAS# Lead Time	75.00		62.00	
tCWL	Write Command to CAS# Lead Time	75.00		70.00	
tDS	Data-in Setup Time	17.75		17.75	
tDH	Data-in Hold Time	26.20		21.20	
tDHR	Data-in Hold Time to RAS#	97.50		57.50	
Page Mode Cycles:					
tPC	Page Mode Cycle Time	60.00		23.00	
tRAPC	Page Mode RAS# Pulse Width	60.00			
tRSW	RAS# to Second WE# Delay Time	105.00			
tCP	CAS# Precharge Time	15.00		10.00	
tWI	Write Invalid Time	15.00			
tCAP	Access Time from Column Precharge Time		38.25		4.25

240725–C9

Timings for No Cache Configuration (Continued)

ADDRESS DECODER REQUIREMENTS

Symbol	Parameter	For 80386-33 Minimum	Maximum
tpd	Available Propagation Delay		8.75

ROW ADDRESS COMPARATOR REQUIREMENTS

Symbol	Parameter	For 80386-33 Minimum	Maximum
tpd	Available Propagation Delay		8.75

NA# SETUP TIME

Symbol	Parameter	Minimum	Maximum
tNA#	Available NA# Setup Time	5.25	

QUAD TTL TO 10KH-ECL TRANSLATOR
MC10H124

Symbol	Parameter	Minimum	Maximum
tpd	Propagation Delay	2.75	3.25

QUAD 10KH-ECL to TTL TRANSLATOR
MC10H125

Symbol	Parameter	Minimum	Maximum
tpd	Propagation Delay	0.00	0.00

DELAY ELEMENT

Symbol	Parameter	Minimum	Maximum
tpd	Propagation Delay	4.00	6.00

240725-D0

Timings for No Cache Configuration (Continued)

DRAM SPECIFICATIONS

Symbol	NMB 2801-06 Minimum	NMB 2801-06 Maximum	VITELIC V53C256 (70 ns) Minimum	VITELIC V53C256 (70 ns) Maximum
tRC	121.00		130.00	
tRP	55.00		50.00	
tRAS	60.00	100000	70.00	75000.00
tCAS	11.00		15, 20	75000.00
tCAS	5.00			
tASC	0.00		0.00	
tCAH	6.00		15.00	
tAR	40.00		55.00	
tRCD	6.00	45.00	25.00	55.00
tRAD	4.00	28.00	20.00	35.00
tRSH	15.00		15, 25	
tCSH	40.00		70.00	
tCRP	3.00		15.00	
tASR	2.00		0.00	
tRAH	2.00		15.00	
tT			3.00	25.00
tREF				
tREF2				
tRAC		60.00		70.00
tCAC		11.00		15.00
tAA		32.00		35.00
tRCS	0.00		0.00	
tRCH	0.00		5.00	
tRRH	0.00		5.00	
tOFF		17.00	0.00	15.00
tWCS	0.00		0.00	
tWCH	5.00		15.00	
tWCR	40.00		55.00	
tWP	5.00		15.00	
tRWL	13.00		20.00	
tCWL	5.00		20.00	
tDS	0.00		0.00	
tDH	5.00		15.00	
tDHR	40.00		55.00	
tPC	37.00		50.00	
tRAPC				
tRSW				
tCP	5.00		15.00	
tWI				
tCAP		34.00		45.00

240725–D1

CAPACITIVE LOAD TIMING DERATING FOR 74ACT258

Load Capacitance (pF)	Additional Propagation Delay (ns)
60.00	0.26 (p = 0.02625q - 1.3125)
80.00	0.79
100.00	0.89 (p = 0.022q - 1.3125)
120.00	1.33
140.00	1.77
160.00	2.21
180.00	2.65
200.00	3.09
220.00	3.83 (p = 0.01666q + 0.1666)
240.00	4.17
260.00	4.50
280.00	4.83
300.00	5.17

DRAM ADDRESS BUS TIMING DERATING

Reason	Capacitive Load (pF)	Additional Propagation Delay (ns)
DRAM Address Inputs	160.00	
F258 Output		
Microstrip/Strip Lines	60.00	
TOTAL	220.00 ===>	3.80

240725–D2

Timings for No Cache Configuration (Continued)

EQUATIONS FOR DRAM TIMINGS (82385 Active):

Read and Write Cycles (Common Parameters):

tRC: Random Read or Write Cycle Time

CLK2 × 10

tRP: RAS# Precharge Time

CLK2 × 4

tRAS: RAS# Pulse Width

CLK2 × 4

A random DRAM cycle may have a RAS# pulse which is only four CLK2 periods wide. This is the case if the cycle is followed by Idle cycles (DRAMs not selected or Ti's) or a DRAM page miss.

tCAS (Read): CAS# Pulse Width

CLK2 × 5

CAS# pulses can be as narrow as five CLK2 cycles during Page Mode read cycles.

tCAS (Write): CAS# Pulse Width

CLK2 × 2

CAS# pulses can be as narrow as two CLK2 cycles during Page Mode write cycles.

tASC: Column Address Setup Time

min (CLK2 × 2 + AS32.tphl.min − Delay.max − ACT258.StoZ.tpl.max − ACT258.Cap.Derating, CLK2 × 3 + AS32.tphl.min − t6.max − 386.Cap.Derating − AS373.DtoO.tpd.max − ACT258.ltoZ.tpl.max − ACT258.Cap.Derating)

The Column Address becomes valid as RAS# switches from High to Low or as the 386 address becomes valid while RAS# is already Low (i.e., Page Mode, Pipelined cycles)

tCAH: Column Address Hold Time

CLK2 + AS373.GtoO.tpd.min + ACT258.ltoZ.tpl.min − AS32.tphl.max

The CAL (Column Address Latch) signal is activated one CLK2 period after the active-going edge of CAS#.

tAR: Column Address Hold Time to RAS#

CLK2 × 3 + AS373.GtoO.tpd.min + ACT258.ltoZ.tpl.min − RAS.Delay.max

tRCD: RAS# to CAS# Delay Time

CLK2 × 2 + AS32.tphl.min − RAS.Delay.max

tRAD: RAS# to Column Address Delay Time

(min) ACT258.StoZ.tphl.min + Delay.min − RAS.Delay.max

(max) ACT258.StoZ.tphl.max + Delay.max + ACT258.Cap.Derating − RAS.Delay.min

tRSH: RAS# Hold Time

CLK2 × 2 − AS32.tphl.max + RAS.Delay.min

The worst case occurs when a DRAM Page miss or Idle is detected at the end of the current DRAM Page miss cycle.

tCSH: CAS# Hold Time

CLK2 × 6 + AS32.tphl.min − RAS.Delay.max

tCRP: CAS# to RAS# Precharge Time

CLK2 × 2 + RAS.Delay.min − AS32.tplh.max

This is guaranteed by the DRAM control state machine.

tASR: Row Address Setup Time

CLK2 × 2 − t6.max − 386.Cap.Derating − ACT258.ltoZ.max − ACT258.Cap.Derating + H124.tpd.min + H125.tpd.min + PAL.tco.min + RAS.Delay.min

tRAH: Row Address Hold Time

ACT258.StoZ.tphl.min + Delay.min − RAS.Delay.max

tT: Transition Time (Rise and Fall)

tREF: Refresh Period

tREF2: Refresh Period

Read Cycles:

tRAC: Access Time

CLK2 × 8 − H124.tpd.max − H125.tpd.max − PAL.tco.max − − F245.max − AS646.tpd.max − F245.max − RAS.Delay.max − SRAM.tDW − CLK2 + 385.t22a.min

tCAC: Access Time from CAS#

CLK2 × 5 − H124.tpd.max − H125.tpd.max − PAL.tco.max − AS32.tphl.max − F245.max − AS646.tpd.max − F245.max − SRAM.tDW − CLK2 + 385.t22a.min

tAA: Access Time from Address

CLK2 × 8 − t6.max − 386.Cap.Derating − AS373.DtoO.max − ACT258.ItoZ.tp.max − ACT258.Cap.Derating − F245.max − AS646.tpd.max − F245.max − SRAM.tDW − CLK2 + 385.t22a.min

tRCS: Read Command Setup Time

CLK2 + AS32.tphl.min

tRCH: Read Command Hold Time to CAS#

CLK2 − AS32.tplh.max

tRRH: Read Command Hold Time to RAS#

CLK2 − RAS.Delay.max

tOFF: Output Buffer Turn-off Time

CLK2 × 2 + F245.tzh.min

Write Cycles:

tWCS: Write Command Setup Time

CLK2 × 3 + AS32.tphl.min

tWCH: Write Command Hold Time

CLK2 × 2 − AS32.tplh.max

tWCR: Write Command Hold Time to RAS#

CLK2 × 6 − RAS.Delay.max

tWP: Write Command Pulse Width

CLK2 × 5

tRWL: Write Command to RAS# Lead Time

CLK2 × 5 + RAS.Delay.min

tCWL: Write Command to CAS# Lead Time

CLK2 × 5

tDS: Data-in Setup Time

CLK2 × 3 + H124.tp.min + H125.tp.min + AS32.tphl.min − − 385.t43c.max − AS646.GotO.tp.max − F245.tp.max

tDH: Data-in Hold Time

CLK2 × 2 + F245.tpz.min − AS32.tphl.max

tDHR: Data-in Hold Time to RAS#

CLK2 × 6 + F245.tpz.max + RAS.Delay.min

Page Mode Cycles:

tPC: Page Mode Cycle Time

CLK2 × 6

tRAPC: Page Mode RAS# Pulse Width

CLK2 × 4

tRSW: RAS# to Second WE# Delay Time

CLK2 × 7 − RAS.Delay.max

tCP: CAS# Precharge Time

CLK2

tWI: Write Invalid Time

CLK2

tCAP: Access Time from Column Precharge Time

CLK2 × 6 − H124.tp.max − H125.tp.max − PAL.tco.max − − F245.max − AS646.tpd.max − F245.max − SRAM.tDW − CLK2 + 385.t22a.min

DRAM TIMING REQUIREMENTS

Symbol	Parameter	For 80386-33 Minimum	For 80386-33 Maximum	Timing Margin (NMB 2801-06) Minimum	Timing Margin (NMB 2801-06) Maximum
Read and Write Cycles (Common Parameters):					
tRC	Random Read or Write Cycle Time	150.00		29.00	
tRP	RAS# Precharge Time	60.00		5.00	
tRAS	RAS# Pulse Width	60.00		0.00	
tCAS	CAS# Pulse Width (Read)	75.00		64.00	
tCAS	CAS# Pulse Width (Write)	30.00		25.00	
tASC	Column Address Setup Time	9.70		9.70	
tCAH	Column Address Hold Time	14.20		8.20	
tAR	Column Address Hold Time to RAS#	50.00		10.00	
tRCD	RAS# to CAS# Delay Time	31.00		25.00	14.00
tRAD	RAS# to Column Address Delay Time	5.00	21.30	1.00	6.70
tRSH	RAS# Hold Time	24.20		9.20	
tCSH	CAS# Hold Time	91.00		51.00	
tCRP	CAS# to RAS# Precharge Time	24.20		21.20	
tASR	Row Address Setup Time	6.20		4.20	
tRAH	Row Address Hold Time	5.00		3.00	
tT	Transition Time (Rise and Fall)				
tREF	Refresh Period				
tREF2	Refresh Period				
Read Cycles:					
tRAC	Access Time	67.50		7.50	
tCAC	Access Time from CAS#	16.70		5.70	
tAA	Access Time from Address	37.70		5.70	
tRCS	Read Command Setup Time	20.80		20.80	
tRCH	Read Command Hold Time to CAS#	9.20		9.20	
tRRH	Read Command Hold Time to RAS#	15.00		15.00	
tOFF	Output Buffer Turn-off Time		33.00		16.00
Write Cycles:					
tWCS	Write Command Setup Time	46.00		46.00	
tWCH	Write Command Hold Time	24.20		19.20	
tWCR	Write Command Hold Time to RAS#	90.00		50.00	
tWP	Write Command Pulse Width	75.00		70.00	
tRWL	Write Command to RAS# Lead Time	75.00		62.00	
tCWL	Write Command to CAS# Lead Time	75.00		70.00	
tDS	Data-in Setup Time	9.00		9.00	
tDH	Data-in Hold Time	31.70		26.70	
tDHR	Data-in Hold Time to RAS#	97.50		57.50	
Page Mode Cycles:					
tPC	Page Mode Cycle Time	90.00		53.00	
tRAPC	Page Mode RAS# Pulse Width	60.00			
tRSW	RAS# to Second WE# Delay Time	105.00			
tCP	CAS# Precharge Time	15.00		10.00	
tWI	Write Invalid Time	15.00			
tCAP	Access Time from Column Precharge Time		37.50		3.50

APPENDIX E
REFERENCES

REFERENCES

Advanced CMOS Logic Designer's Handbook, Texas Instruments Inc., 1988.

Blood W., *MECL System Design Handbook*, Motorola Corp., 1983.

Keeler R., "High Speed Digital Printed Circuit Boards," *Electronic Packaging & Production*, pp. 140-145, Jan. 1986.

Tomlinson J., "Avoid The Pitfalls of High Speed Logic Design," *Electronic Design*, pp. 75-84, Nov. 9, 1989.

Pace C., "Terminate Bus Lines to Avoid Overshoot and Ringing," *EDN*, pp. 227-234, Sept. 17, 1987.

Royle D., "Rules Tell Whether Interconnections Act Like Transmission Lines," *EDN*, pp. 131-136, June 23, 1988.

Royle D., "Correct Signal Faults by Implementing Line-Analysis Theory," *EDN*, pp. 143-148, June 23, 1988.

Winchester E., "Guidelines Help You Design High-Speed PC Boards," *EDN*, pp. 221-226, Nov. 28, 1985.

Yeargan J. R., Day R. L., and Nguyen T., "Effects of Printed Circuit Board Transmission Lines an Loading on Gate Performance," *IEEE Transactions on Industrial Electronics*, Vol. IE-34, no. 3, pp. 399-405, Aug. 1987.