CHAPTER 6

Designing with the programmable three-cycle memory interface

The programmable three-cycle memory interface found on T4xx and T8xx series transputers is described in detail in Chapter 5. This interface provides a very flexible arrangement for interfacing with external memory and I/O devices. The number of possibilities is almost limitless so this chapter sets out to illustrate the use of the external memory interface through examples of common designs such as static and dynamic memory interfacing. This requires a thorough understanding of the memory interface strobes and the reader is recommended to review Chapter 5 if necessary.

Probably the most common system design will consist of a 32 bit transputer with 1 or 2 Mbytes of dynamic RAM, suitable for use as a general purpose processing module and capable of running the INMOS TDS or Toolset development system software. The programmable memory interface can run at zero wait states using 70 ns DRAM and designs will generally take advantage of relatively cheap DRAM rather than the more expensive SRAM.

Figure 6.1 shows a summary of the programmable strobe timings for reference.

The following designs are based on a 20 MHz transputer so each period `Tm` has a duration of 25 ns.

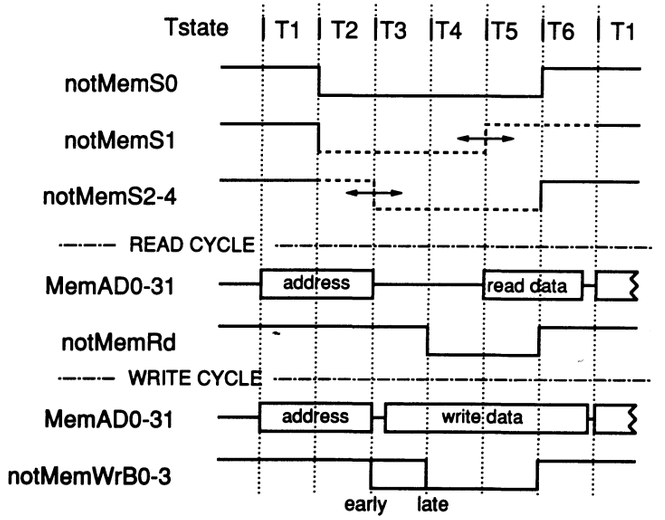

Figure 6.1 Summary of the programmable memory strobes.

6.1 Fast SRAM interface

Figure 6.2 shows internal configuration 8 which is suitable for use with fast SRAM such as INMOS 1820 64K × 4 devices. These have relatively fast access times and short output disable times, allowing them to be used in the minimum length external memory cycle of six Tm periods or three processor cycles.

To demultiplex the address/data bus it is necessary to provide address latches and a suitable control strobe, commonly referred to as address latch enable ALE. Of the five transputer strobes, notMemS0 has fixed timing and notMemS4 is conventionally reserved for wait state generation. In this example notMemS0 is used to drive the address/data bus demultiplexer latches and notMemS3 provides the chip enable. The address is held by the bus demultiplexer latches when notMemS0 falls at the end of T1 and is not released until the end of T5. Any of strobes notMemS2–4 could be used for chip enable, with suitable configuration, but choosing notMemS3 allows internal configuration 8 (SRAM in three processor clock cycles) to be selected.

Read cycle: notMemS3 is programmed to go low at the beginning of T4 and returns high at the end of T5. Data is read on the rising edge of notMemRd at the end of T5. The access time required for the memory devices is given by:

$$t_{acc}(\text{zero wait states}) = T4 + T5 - Tm = 25\,\text{ns}$$

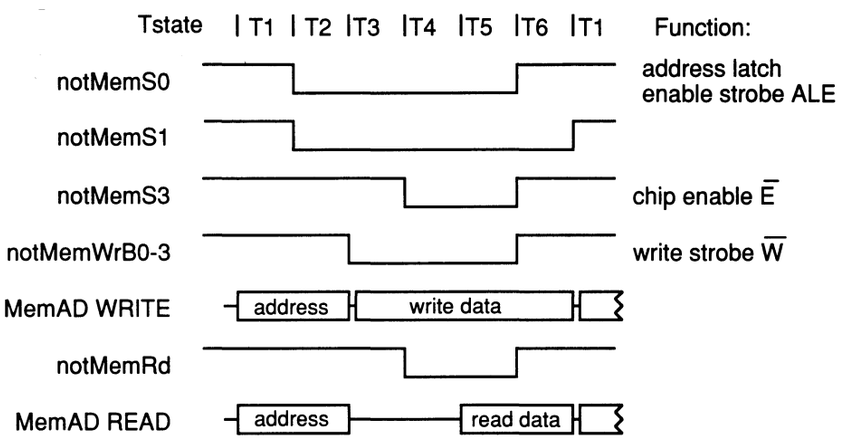

Figure 6.2 Internal configuration 8 for fast SRAM.

where a data set-up time of 1 Tm has been allowed. Each wait state, which extends the duration of T4, will add one Tm (25 ns) to this figure.

Write cycle: The write strobes notMemWrB0–3 are programmed for early write to take them low as close to the start of the cycle as possible. The chip enable, provided by notMemS3, is taken low one Tstate later to ensure that the RAM does not try to output data during the early stages of a write cycle causing a clash on the bus. This timing configuration delays chip enable until relatively late into the cycle and requires fast access RAM to function with zero wait states. If RAM devices with separate output enables and chip enables are used, it is possible to advance the chip enable strobe reducing the demands on RAM performance as described in Section 6.1.1.

The control strobe for the bus demultiplexer latches ALE is provided by notMemS0 which returns high at the end of T5, releasing the address latches. For RAM devices which require write recovery time (address hold after write/chip enable high), notMemS1 must be used for ALE to hold the address until the end of T6.

A connection diagram for this configuration is shown in Figure 6.3.

6.1.1 SRAM with output enable

A modified timing configuration may be used with SRAM which has an output enable strobe, reducing the demands on access time. Figure 6.4 shows internal configuration 8 with notMemS2 used to provide chip enable.

The read strobe notMemRd is connected to the output enable to ensure that the RAM outputs remain active throughout the write cycle. This allows the active falling

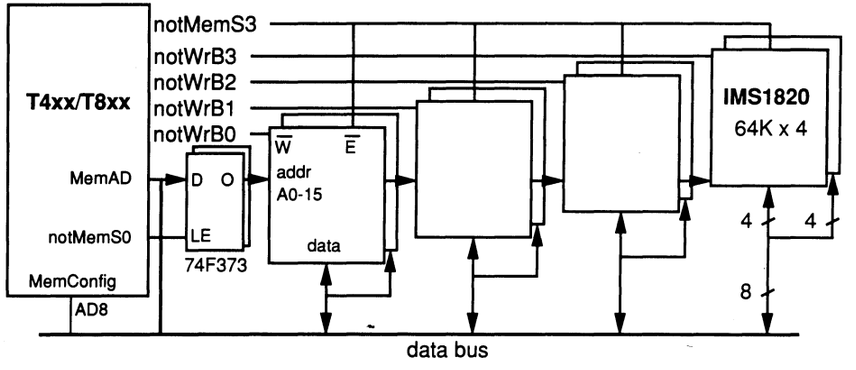

Figure 6.3 T4xx/T8xx with 256 kbytes of fast SRAM.

edge of chip enable notMemS2 to be advanced to the end of T2 which increases the access time to:

$$t_{acc}\,(\text{zero wait states}) = T3 + T4 + T5 - Tm = 50\,ns$$

Figure 6.5 shows a 32K memory system using this configuration. It will run at zero wait states using INMOS 1630 8K × 8 devices with an access time of 45 ns.

6.2 Slow SRAM interface

Slow SRAM devices require a slower memory cycle to allow extra access time and it is usually found that problems are caused by their slow output disable times. If bus contention following a read cycle is to be avoided, the output must be disabled before the end of T6. The minimum duration for T6 is 1 Tm allowing a maximum disable time of 25 ns which is often unsuitable for slower RAM chips. The 62256 85 ns SRAM for example has an output disable time of 30 ns.

Figure 6.6 shows a timing configuration suitable for this device where T5 and T6 have been increased to allow extra access time and extra bus release time. The width of the read strobe notMemRd is extended through increasing the duration of T5; this allows extra enable time for the slow output drivers.

RAM chip enable is driven by the fixed address strobe notMemS0 which goes low as early as possible in the cycle maximising the available access time. The RAM chip enable access time required for zero wait state operation is given by:

$$t_{acc}\,(\text{zero wait states}) = T2 + T3 + T4 + T5 - Tm$$
$$= 100\ ns$$

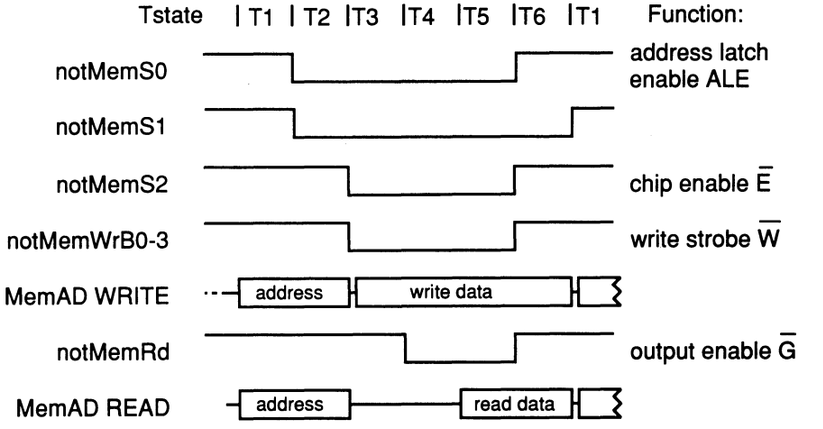

Figure 6.4 Modified strobe selection for SRAM with output enable.

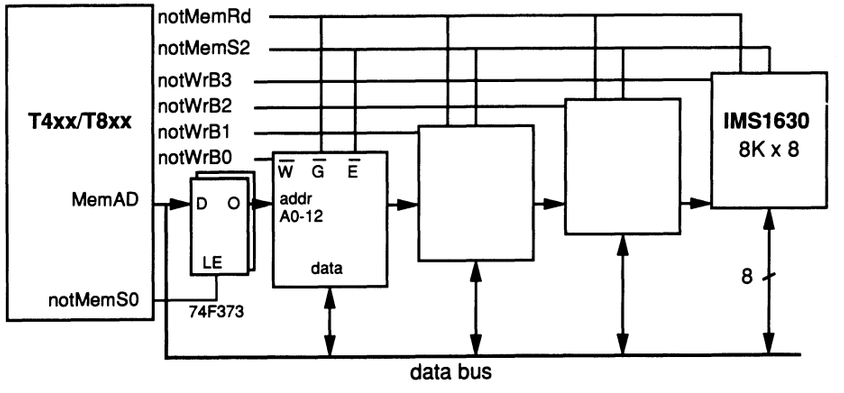

Figure 6.5 Connection of SRAM with output enable.

The output enable access time required is given by:

$$t_{oe} = T4 + T5 - Tm = 50\,ns$$

Early or late write can be used but in the case of early write, data will only be valid on the rising edge of the write strobe. Additional wait states may be added if necessary by the use of MemWait which extends the duration of T4 under external control.

If the RAM devices are the fastest external devices in the system, which is usually the case, the memory interface strobes should be configured to suit the RAM speed, leaving MemWait free for use with other, slower devices.

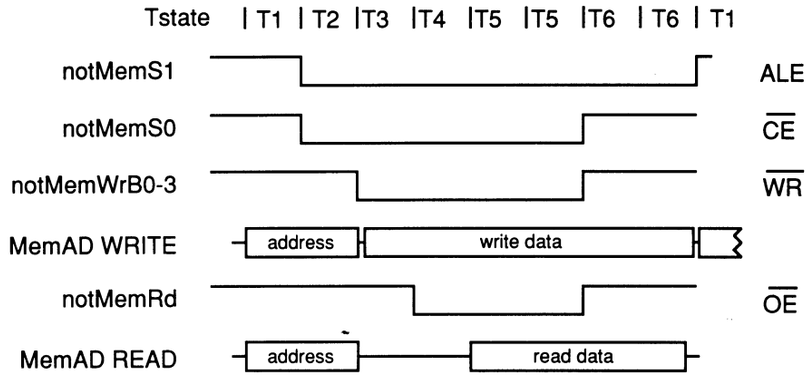

Figure 6.6 Timing configuration suitable for slower SRAM.

6.3 DRAM interface

The following two examples look at a simple low-cost interface with 1–2 Mbytes of dynamic memory. The transputer provides direct support for DRAM in the form of control strobes and automatic refresh cycles, making the interface relatively straightforward. The simplest configuration requires only the addition of a column address latch and a row/column address multiplexer.

The procedure adopted is to select the memory devices and appropriate timing strobes, design a suitable memory configuration and finally check timing carefully, including the effect of buffers if necessary. It is assumed here that the DRAM is the fastest memory in the system so the memory configuration is chosen to suit the access time of the DRAM devices. Any slower devices may be catered for by inserting wait states using `MemWait` in conjunction with strobe `notMemS4`.

The following design is based on a 20 MHz transputer so all parameters quoted are based on 1 `Tm` = 25 ns.

6.3.1 Selection of memory devices

1 Mbyte or 2 Mbytes of DRAM can be implemented economically using 256K × 4 devices. In this case the 44256 has been chosen. Table 6.1 shows a summary of important parameters for various speed parts.

6.3.2 Choice of strobes/configuration

The first of the designs considered here uses the fastest external memory configuration (three processor clock cycles) where Tstates `T1`–6 are of duration 1 `Tm`. Figure 6.7 shows a timing diagram for internal configuration 4.

Table 6.1 44256 DRAM timing requirements.

Device	Cycle time (ns min)	RAS access time (ns min)	RAS precharge time (ns min)
44256-7	140	70	60
44256-8	160	80	70
44256-10	190	100	80
44256-12	220	120	90

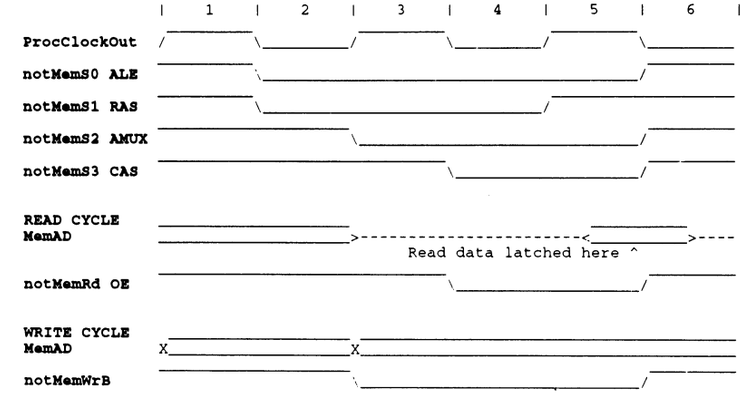

Figure 6.7 Timing diagram for DRAM running at zero wait states – configuration 4.

In order to drive DRAM correctly, several control signals are required which are provided by the programmable strobes. The selection of strobes is now discussed.

ALE Address latches are required to demultiplex the address/data bus. They are driven by a control strobe, commonly referred to as *address latch enable* ALE. The fixed strobe `notMemS0` has timing suitable for this function; its falling edge occurs at the end of `T1` which is a suitable point to hold address information. Note that dynamic RAM devices contain internal address latches for the first stage address (row address) so it is only necessary to latch the second stage (column address) externally.

RAS The row address is set up on the DRAM address lines and RAS (Row Address Select) is taken low to latch this address. RAS must go low as early as possible when the address is stable to allow maximum RAS access time (data valid from RAS low). The optimum point for this is at the end of `T1`. The falling edge of strobe `notMemS1`

is fixed at the end of T1 and its rising edge is programmable which makes it suitable for generating RAS.

To ensure sufficient RAS precharge time (RAS high pulse width), RAS must return high as early as possible. RAS terminates at the end of T4 in Figure 6.7 which allows maximum precharge time.

AMUX After the address has been latched by the falling edge of RAS, the row address must be held for the *row address hold time*. The row/column address multiplexer can then be switched to route the column address to the DRAM address lines. The earliest point for this switch is at the end of T2 allowing 1 Tm row address hold time. The falling edge of strobe notMemS2 is programmable past the end of T2 and it returns high at the end of T5. This is a suitable point to switch the multiplexer back to row address in time for the next cycle.

With some fast DRAMs the row address hold time is very short and AMUX (Address Multiplexer) may be derived from a slightly delayed RAS signal. An example of this is documented in *The Transputer Applications Notebook – Systems and Performance* INMOS (1989).

CAS The falling edge of CAS (Column Address Select) latches the column address. To ensure that the address is stable, CAS is taken low at the end of T3, 1 Tm after switching the multiplexer.

$\overline{\text{WR}}$ The write strobes notMemWrB0–3 drive the DRAM write enable lines. Early or late write can be used as the memory devices have output enable control which prevents a clash of data on the bus during write cycles. With an early write, data is strobed into memory on the falling edge of CAS and must be valid at this point. With late write, data is strobed into memory on the falling edge of the write strobe.

$\overline{\text{OE}}$ The DRAM output enables are driven by the read strobe notMemRd. During write cycles the outputs remain disabled throughout the whole cycle.

The fastest configuration for the external memory interface, where T1–6 are of duration 1 Tm, has a cycle time of 6 × 25 ns = 150 ns. Referring to Table 6.1, this is only suitable for 70 ns parts. The natural division of cycle time is 75 ns RAS access time and 75 ns RAS precharge time as shown in the timing diagram of Figure 6.8. The early termination of RAS at the end of T4 allows an overlap of RAS precharge with RAS access time, extending the access time available to:

$$t_{acc} = T2 + T3 + T4 + T5 - Tm = 75\,ns$$

where a data set-up time of 1 Tm has been allowed.

The CAS access time allowed is T4 + T5 − Tm = 25 ns. This should cause no problems with 70 ns DRAM which has a CAS access time of 18 ns maximum. Internal configuration 4 shown in Figure 6.7 therefore looks suitable for use with 70 ns DRAM running at zero wait states.

At the end of a read cycle, the output drivers are disabled when notMemRd returns

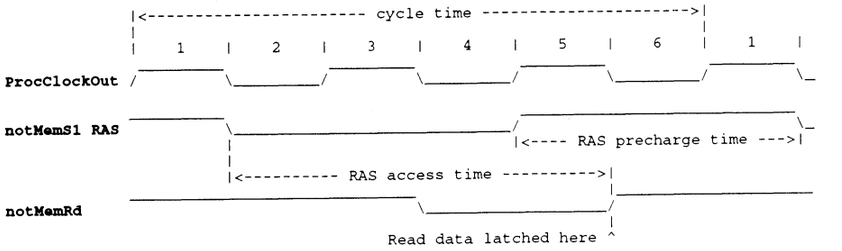

Figure 6.8 RAS access time and precharge time.

high. The data bus must be free 25 ns later, at the start of T1 of the next cycle. The output disable time quoted for the 70 ns DRAM is 18 ns maximum so bus contention will not occur.

6.3.3 Address decoding and refresh circuitry

Simple RAM-only systems do not require address decoding so RAS and CAS can be driven directly by transputer strobes. Refresh will take place automatically if selected.

To perform refresh of dynamic RAM it is only necessary to provide a row address and assert RAS which causes an entire row to be refreshed; this is called *RAS-only refresh*. If RAS (notMemS1) is directly connected to all memory devices, any DRAM access will automatically perform a RAS-only refresh.

The transputer contains an internal 10 bit refresh counter which can refresh devices of up to 1M × n bits. During a refresh cycle, the 10 bit refresh or row address, which increments at regular intervals, is placed on MemAD2–11. For 256K × 4 DRAMs, only nine address bits MemAD2–10 are used which cycle through counts from 0 to 511. A complete refresh cycle will therefore take (2^9 × refresh interval) where the refresh interval can be set at 18, 36, 54 or 72 periods of ClockIn.

The 44256 requires a complete refresh every 8 ms which is satisfied by the slowest refresh interval of 72 periods of ClockIn:

$$\text{complete refresh cycle} = 2^9 \times 72 \times 1/(5\ \text{MHz}) = 7.37\ \text{ms}$$

Where address decoding is required it can be applied to either RAS or CAS. During a RAS-only refresh cycle, RAS must be enabled and CAS must be disabled. This can be achieved by gating one of the memory refresh strobes notMemRf or MemnotRfD1 with CAS. Remember that MemnotRfD1 requires latching to demultiplex it from data bit D1.

The memory is organised as banks of eight devices giving 1 Mbyte per bank. Allowing RAM to repeat throughout the lower half of the memory space, decoding can be based on address lines A31 and A20 as shown in Table 6.2.

Table 6.2 Decoding for 1 Mbyte banks of memory.

A31	A20	
1	0	bank 1 (0 – 1 Mbyte)
1	1	bank 2 (1 – 2 Mbyte)

Table 6.3 Address decoding applied to the CAS strobes.

notMemS3	A31	A20	CAS1	CAS2	Notes
1	X	X	1	1	RAM disabled
0	1	0	0	1	RAM bank 1
0	1	1	1	0	RAM bank 2
0	0	X	1	1	Refresh cycle

The simplest logic design is obtained by applying decoding to CAS. Expansion of the scheme of Table 6.2 to include the CAS strobe notMemS3 is shown in Table 6.3. Two strobes CAS1 and CAS2 are derived, one for each 1 Mbyte bank of DRAM.

When notMemS3 is low and A31 is high, a memory access is taking place in the lower half of the memory space and one bank of DRAM is enabled, depending on the value of A20. During a refresh cycle, A31 stays low and can be used as a refresh strobe for a simple memory system – if A31 is low, then either access is being made to the upper half of the memory space, or a refresh is taking place. In either case CAS1 and CAS2 are disabled and a refresh occurs. The use of A31 makes it unnecessary to include one of the refresh strobes in the decoding.

This scheme can be implemented using NAND or NOR logic but the NOR implementation shown in Figure 6.9 is preferable as it presents the minimum propagation delay on notMemS3 (CAS). The inverters at the NOR gate outputs are bus drivers suitable for driving capacitive loads (see Section 6.3.5).

An example implementation of this logic using a PAL is given in Appendix C.

6.3.4 Address/data bus demultiplexer and row/column address multiplexer

The address/data bus can be demultiplexed in the conventional manner by using a transparent latch driven by ALE (notMemS0 in this case). It is only necessary to latch the column address lines. A driver can be used to buffer the row address providing row/column address switching as shown in Figure 6.10.

At the beginning of an external memory cycle AMUX is high and the row address is enabled through the 74F827 buffer. The column address is held in the 74F841 transparent latch when the ALE strobe goes low. When AMUX goes low the latched

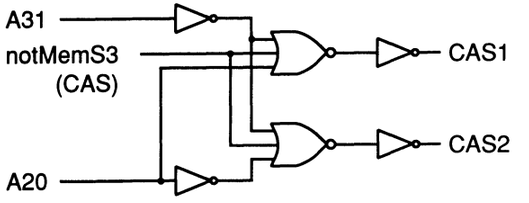

Figure 6.9 Column address strobe decoding logic.

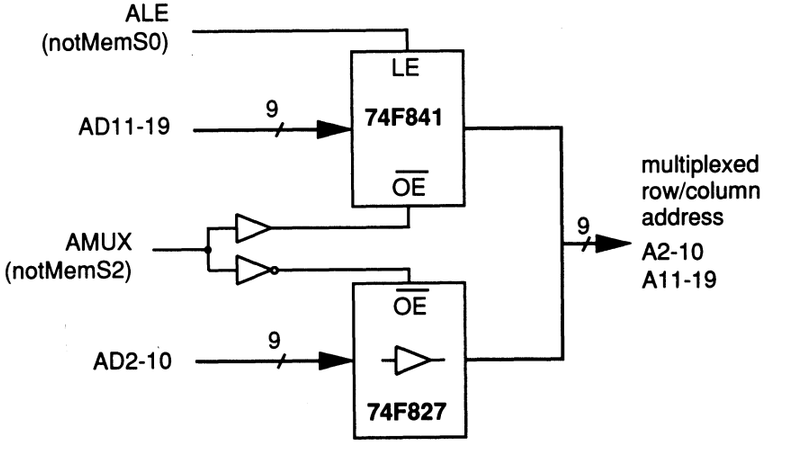

Figure 6.10 Row/column address multiplexer and address latch logic.

column address is output on to the bus. The propagation delay between AMUX and the chip output enables being asserted/de-asserted is made roughly equal by including a buffer to minimise overlap of the output active periods during switchover.

In addition it will be necessary to latch A20 and A31 for the decoding. If other devices are placed in the upper part of the address space, e.g. EPROM, then the relevant address lines will also need to be latched.

6.3.5 Buffering

The maximum load capacitance on any transputer memory interface output pin is 60 pF. Where several memory devices are driven by an output, it is important to consider the cumulative loading effect.

The row address strobe RAS is generated directly by strobe notMemS1, which must drive all 16 memory devices, a load of:

16 devices × 7 pF = 112 pF + layout capacitance

where the DRAM inputs present a maximum load capacitance of 7 pF. Either a high-power bus driver must be employed, or the signal can be split and two drivers used to drive eight devices each. The same problem exists with the read strobe notMemRd which must also drive all 16 memory devices.

The write strobes notMemWrB0–3 drive four devices each, a load of:

4 devices × 7 pF = 28 pF + layout capacitance

This is well below the 60 pF limit so buffering is not required provided that a good layout is used.

The transputer data lines drive only two devices each (one memory device and the row/column address multiplexer), so buffering is not strictly necessary.

The column address strobes CAS1 and CAS2 each drive eight DRAM devices which present a load of:

8 devices × 7 pF = 56 pF + layout capacitance

A standard bus driver is quite suitable for this task.

The address lines are already buffered by the row/column address multiplexer logic and must drive a load of:

16 devices × 5 pF = 80 pF + bus capacitance

where the DRAM address inputs present a maximum load capacitance of 5 pF.

To prevent ringing, all buffered signal lines should be series terminated. The value of the termination resistor will vary from system to system but will usually lie in the range 20 to 50 Ω.

It is now necessary to check carefully all memory timing parameters for both read and write cycles, taking into account the delays introduced by decoding and buffering logic.

6.3.6 Complete circuit

Figure 6.11 shows the circuit diagram for a system containing 1 Mbyte of DRAM The decoding for CAS1 is not shown for clarity (see Figure 6.9). A second 1 Mbyte bank can be connected in an identical manner with all CAS lines driven by CAS2.

6.3.7 Wait states

By adding wait states, the memory cycle may be extended to suit DRAM with

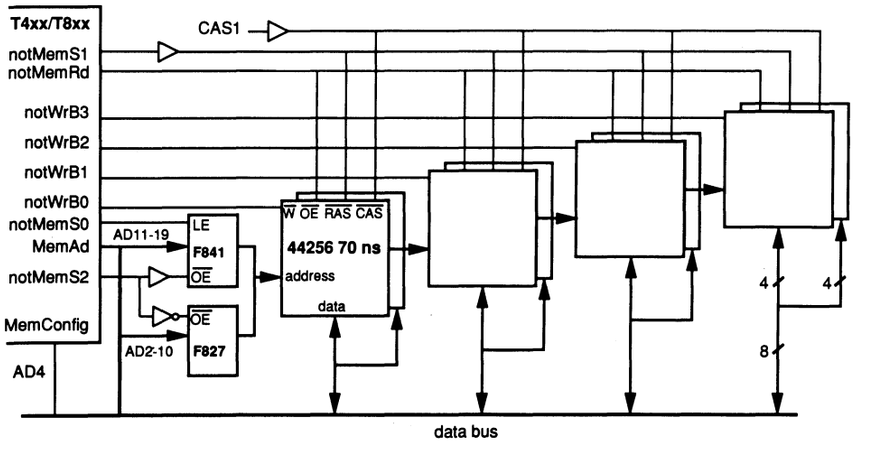

Figure 6.11 Memory connection diagram (only one bank shown).

slightly longer access times. Wait states may be added simply by connecting MemWait to notMemS4 and programming notMemS4 to go low at a suitable point (see Chapter 5 for further details).

The active edges of strobes notMemS0 and notMemS2–4 are not affected by the addition of wait states, only their duration. notMemS1, however, will be caused to terminate earlier than expected when wait states are inserted which will certainly cause problems with the design of Section 6.3.6. It is not possible therefore to make use of faster or slower memory with this design, simply by connecting MemWait to strobe notMemS4 for slow memory, or to 0 V for fast memory.

6.3.8 Configuration for slower DRAM

Where it is required to make use of slower memory devices, the addition of wait states alone is sometimes unsatisfactory, for example where the memory has a relatively long output disable time. In this case, a slower external memory cycle must be used.

The next fastest external memory configuration takes four processor cycles (200 ns), as each external memory cycle must take an even number of periods Tm. This gives a cycle time suitable for use with 80 and 100 ns memory devices.

With a 200 ns cycle time, a division of 100 ns precharge and an access time of 125 ns (neglecting data set-up time) is the logical choice; the apparent gain of 25 ns is due to the early termination of RAS. The actual access time is given by:

$$t_{acc} = T2 + T3 + T4 + T5 - Tm = 100\,ns$$

This will suit 80 ns devices but leaves no margin with the slower 100 ns devices.

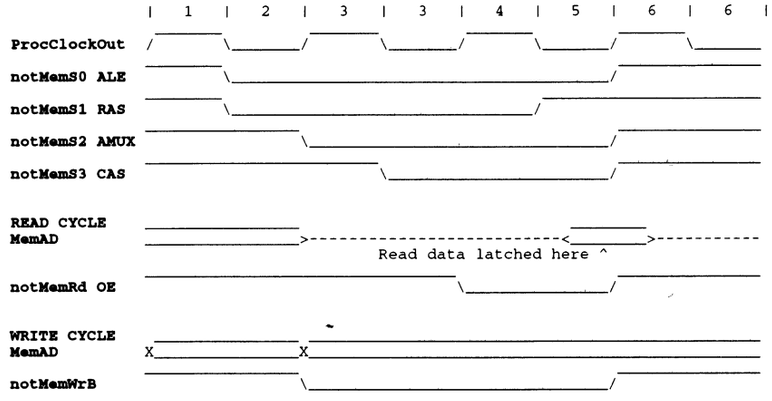

Figure 6.12 Four processor cycle memory configuration for 80 ns DRAM.

Increasing the duration of T1 or T6 will increase the RAS precharge time. T6 is increased in this example to allow extra bus turn-around time for slower devices. Increasing the duration of T2, T3, T4 or T5 will increase the RAS access time but T5 will also affect the ratio between access and precharge time. The modified timing configuration is shown in Figure 6.12.

6.4 EPROM

A common requirement for a processor board is the ability to boot from ROM at power up. This requires a quantity of ROM, usually EPROM, containing boot code, to be located in the upper part of the memory space. This example describes how to add a quantity of EPROM to the system described in Section 6.3.6.

EPROMs are inherently very slow devices with typical access times in the range 120 to 250 ns which will certainly require wait states to be inserted during ROM accesses. The more serious problem, however, is the slow output disable time which will cause bus contention if the EPROM outputs are connected directly to the data bus. There are two solutions to this problem. The available output disable time can be increased by extending the duration of T6 up to a maximum of four periods Tm (100 ns for a 20 MHz transputer), which should be sufficient for most EPROMs. The drawback with this method is that it will considerably slow down accesses to other devices, notably RAM, which is undesirable. The second method is to buffer the EPROM output on to the data bus using fast buffers which will disable within one period Tm.

The programmable memory interface cannot perform byte access so the EPROM data path must be 32 bits wide. This requires a minimum of four (single) byte-wide

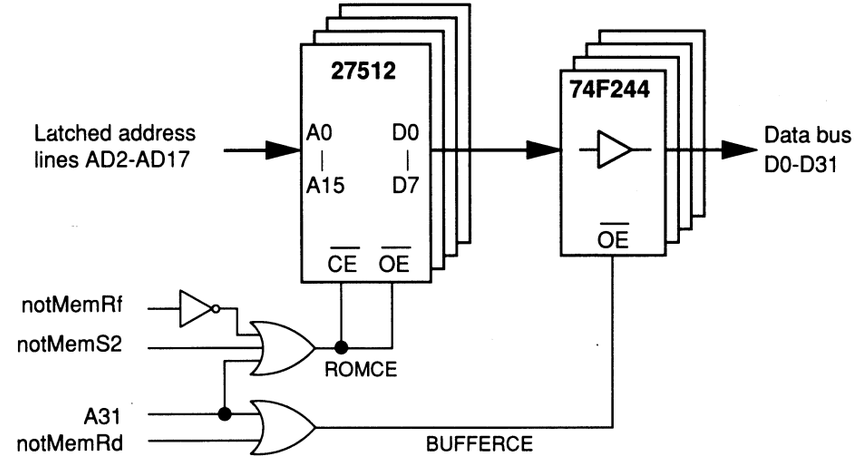

Figure 6.13 EPROM connection diagram showing output buffering.

devices. The design described here uses four 27512 devices giving a total of 256 Kbytes of EPROM, though this could easily be extended. It is important to consider whether the EPROM is required to contain memory configuration information; if so the design must function correctly with both the user configuration and the default boot configuration 31. This case has already been considered in Chapter 5, so for this design it is assumed that the EPROM holds only user program code.

Figure 6.13 shows a basic layout designed to work with configuration 4 (shown in Figure 6.7). The EPROM outputs are buffered by 74F244 drivers which are enabled by the read strobe and address line A31. This allows EPROM to repeat throughout the upper half of the memory space. During a refresh cycle the buffers are held disabled by the read strobe which stays high over this period.

The EPROMs are enabled by the decoded address signal, programmable strobe notMemS2, and the refresh strobe notMemRf, which prevents them from being enabled during a refresh cycle. Strobe notMemS2 is chosen as the main chip enable because it enables the EPROMs at the end of T2 and disables them at the end of T5 which are the optimum points.

The implementation of this logic in a PAL is given in Appendix C.