Integrated Device Technology, Inc.

IDT79R3051™ SYSTEM DESIGN EXAMPLE

APPLICATION NOTE AN-86

by Andrew Ng

INTRODUCTION

This application note describes a memory evaluation board that is an example of many of the design considerations for systems based on an IDT79R3051™ RISController™ family CPU.

The memory board, illustrated in Figure 1, consists of:

- An R3051 CPU
- Reset circuitry
- An address de-multiplexer
- A data transceiver
- Wait-state and memory control logic
- 128K bytes of SRAM
- 128K bytes of EPROM
- A dual channel UART
- A real time counter
- An interrupt controller

In addition, an expansion connector supplies all the CPU signals for the addition of external modules such as DRAM memory systems or other application specific I/O systems. The memory and I/O system on the example board are compatible with the IDT7RS382 R3000 Evaluation Board. Thus 7RS382 software such as the IDT/sim PROM Debug Monitor can run on the example board. The board is typical of an embedded controller core such as for LAN adapters, laser printers, facsimiles, and avionics applications. The differences would appear in which peripherals are used and memory type, size, and speed requirements.

The board was designed as a generic example of the construction of a system using the IDT79R3051 RISController with both low parts count and cost sensitive requirements. However, since many generalities were taken into consideration, many systems can reduce both parts count and cost even further. Although the board is not populated with parts that have the highest performance achievable, its design can be easily modified to do so. In addition, PAL™ support for further experiments with optimizations and trade-offs can be done to accommodate different kinds and speeds of memory and I/O. While the board is designed with SRAM for the simplicity of a design example, the extension to a DRAM system with CAS before RAS refresh is only slightly more complex.

THE R3051 RISCONTROLLER CPU

The IDT79R3051 family is a series of high-performance 32-bit microprocessor RISControllers designed to bring the high-performance inherent in the MIPS™ RISC architecture into low cost, simplified, and power sensitive applications.

The instruction set is compatible with the 79R3000A and 79R3001 RISC CPUs. Features of the R3051 family include:

- 4kB (R3051) to 8kB (R3052) of Instruction Cache on-chip
- 2kB of Data Cache on-chip
- Clocked from a single, double-frequency clock input
- On-chip 4 deep read and write buffer
- On-chip DMA arbiter
- Flexible burst/simple block bus interface
- Multiplexed address and data bus for low cost packaging, simplicity of use
- Base versions use fixed address translation to simplify software
- Extended architecture versions use 64-entry, fully associative Translation Lookaside Buffer (TLB) to support page mapping and virtual memory

The R3051 RISController combines a similarly featured R3000A CPU system consisting of over 50 LSI/MSI parts into a single integrated chip.

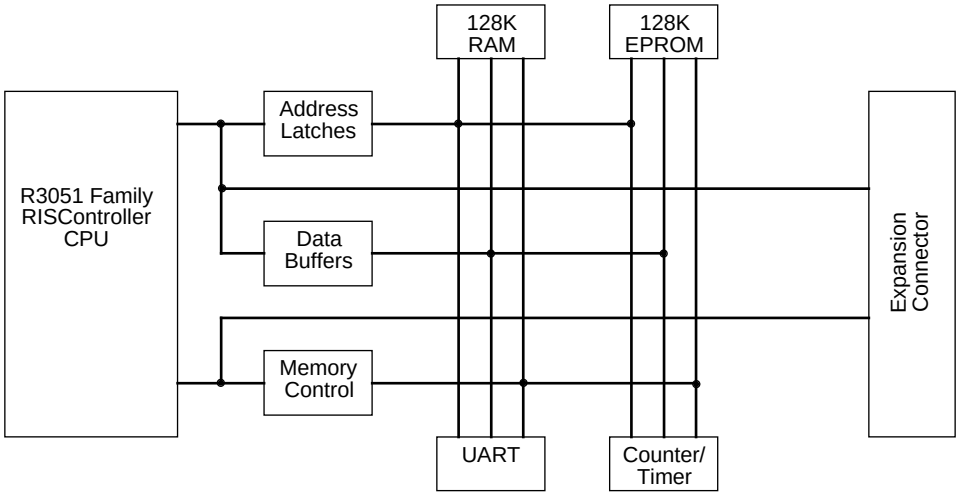


Figure 1. System Block Diagram

DETAILED DESIGN REVIEW

The following sections give a detailed review of how each functional block relates specifically to designing with the R3051 RISController. Particular attention is focused on alternative design strategies that could reduce parts count and improve performance as well as on a description of the original design. The subsystem block designs include:

- Analog reset logic
- A PAL-based memory controller (3x PALs)
- Address de-multiplexer (4x IDT74FCT373T)
- Data transceiver (4x IDT74FCT623T)
- 128kB of SRAM (4x IDT71256 32kx8 45ns SRAM)
- 128kB of EPROM (4x 27256 32kx8 125ns EPROM)
- 68681 DUART
- 8254 Timer
- Interrupt controller (1x PAL)
- Off-card connector

Reset, Reset Vector, and Clock Buffer Circuitry

The $\overline{\text{Reset}}$ signal is based on a linear integrated circuit, a TI TL7705A supply voltage supervisor with a Power-On Reset Generator. A 1 µF capacitor is used to program the reset generator for a 13 ms Reset period.

Note that because the R3051 synchronizes the $\overline{\text{Reset}}$ input signal internally, an RC circuit can be used instead. An example is to pull $\overline{\text{Reset}}$ high with a resistor of about 10K Ohms, tie $\overline{\text{Reset}}$ to a 22 µF capacitor which is tied to ground, and tie $\overline{\text{Reset}}$ to a push button switch that is tied to ground. The example board can be reprogrammed and populated to experiment with $\overline{\text{Reset}}$.

Certain configuration options (the reset vector) are selected in the R3051 by using the interrupt pins at the rising edge of $\overline{\text{Reset}}$. On the example board, the interrupt pins are simply pulled up (or down) since $\overline{\text{SInt}}$(2:0) are not used in this system (software can permanently mask these interrupt inputs in the Status Register). However, if they are used (via the expansion connector) they would need to be multiplexed with the reset function. There are a number of techniques to perform this multiplexing: for example, if the interrupting agent is not capable of tri-stating its interrupt during $\overline{\text{Reset}}$, an external multiplexer such as an IDT74FCT257T can be used, with the enable always tied active and the select tied to $\overline{\text{Reset}}$. If the interrupting agent tri-states its interrupt during Reset, then using simple pull-ups or pull-downs will still operate properly.

The clocks on the board are buffered by an IDT74FCT240C(T) inverting tri-state buffer. This buffer was selected partially to provide a board testability path for injecting a test clock, as well as to buffer the signals to increase their drive. The primary reason for the buffer, however, is to invert $\overline{\text{SysClk}}$ to form SysClk, the signal that is used to clock the state machines on this board. Buffer output pins closest to the ground pin (pins with the lowest pin inductance) were used first to help lessen potential noise and ground bounce problems. The Clk2xIn oscillator is socketed, so that the board may be populated with different speed parts.

In this design, the FCT240C(T) enables are pulled down to be active all of the time. Since $\overline{\text{SysClk}}$ does not tri-state when $\overline{\text{Tri-State}}$ ($\overline{\text{SInt}}$(1)) is active during the reset vector, it is helpful to an ATE programmer to be able to tri-state the inverter.

Memory Controller

The example board's Memory Controller consists of three 22V10 PALs. The first PAL is used for address decoding, the second for wait state and cycle counting, and the third for byte enables. The PALs are functionally described in the following paragraphs. The PAL equations are included in the appendices. The PALs are all placed in sockets, and thus can easily be reprogrammed for various experiments.

Address Decoder

The Address Decoder PAL, MEMDEC.JED, uses Address(31:17) to generate chip selects. The chip selects are decoded according to the 7RS382 address map as described in the 7RS382 Hardware User's Guide. Three spare I/O pins are provided, which could be used to decode additional chip selects. These spare outputs are in place of the 'USER CS1X*' chip selects provided for on the 7RS382 board, but not explicitly supplied by this example board.

The address decoder does not wait for ALE to begin generating the chip-select outputs. It does this so that maximum performance may be achieved, since the Chip Select outputs will be generated earlier in the transfer. However, as a result, the CS outputs may tend to "glitch" as a valid address is driven. Thus, the Read Enable and Write Enable seen in the memory system must be synchronized so that they are valid only within the time that the CPU is attempting a read or write transfer. This combination allows maximum performance: address and chip enables are seen early in the transfer, but the Read and Write signals are generated synchronously to insure proper system operation.

One of the extra I/O pins can be used as a test enable input to tri-state the outputs for board level ATE. Some systems will not need to decode as many address bits or may have a fixed map, and thus may able to use FCT138's or 16V8's to do the address decoding instead of the relatively expensive 22V10 part.

Memory Cycle Controller

The purpose of the Memory Cycle Controller is to provide a wait-state generator which stalls the R3051's Bus Interface Unit, so that various types and speeds of memory can be used. The Memory Cycle Controller is implemented with a 22V10 PAL called MEMCONT.JED. Note that this PAL was selected in order to make the PAL equations more readable. A lower cost solution may implement the state machine in two 16R8 PALs.

The Memory Cycle Controller allows various speeds of memory devices to be used, by using the throttled read supported by the R3051 bus interface. Other kinds of transactions are treated as simplified cases of the throttled read.

The basic state machine looks for the start of a read or write transaction by looking for an asserting edge of $\overline{\text{Rd}}$ or $\overline{\text{Wr}}$. When a transaction is begun, the state machine starts a 5-bit binary up counter, C(4:0). C(4:0) then increments on each SysClk rising edge. C(4:0) is used as the basic timing master for all

of the other control signals generated in the state machine.

In the memory scheme used here, rather than search for the negating edge of Rd or Wr at the end of the transaction, a CycEnd synchronous decoder is used to tell the C counter when the end of the memory cycle occurs. This type of strategy is used because the de-asserting edges of Rd and Wr occur within the setup and hold times of a buffered/inverted (FCT240C(T)) SysClk. Typically, the de-asserting edge of Rd, Wr, and Burst should not be used to control a SysClk based state machine. Similarly, the rapid negation of ALE by the processor makes it difficult to synchronously sample ALE when using a state machine driven by a buffered clock.

CycEnd serves to synchronously reset the state machine when a de-asserting Rd or Wr edge is expected, whether or not the Rd or Wr de-asserting edge meets the setup and hold times of the state machine. Another output, EnStart is used to start the byte enables by waiting a number of cycles before asserting. The amount of time the transfer waits is used to allow drivers used in the previous transfer to tri-state, and may be necessary in systems which employ devices whose output disable time is long relative to the system clock frequency.

Other outputs from the Memory Cycle Controller PAL include the R3051 transfer termination inputs RdCEn, Ack, and BusError. On a read transfer, Burst and one of the Chip Enable inputs from the Address Decoder are used to determine the timing and quantity of RdCEn signals to be asserted for this transfer (according to the requested transfer size and the memory device speed).

Ack is asserted at the end of a write cycle to indicate completion of the transfer, and optionally towards the end of a Quad Word (Burst) read cycle. A description of the various kinds and options of read and write cycles is thoroughly explained in the R3051 Family Hardware User's Guide. The number of cycles before and between the assertion of Ack and RdCEn is programmable, allowing flexibility for various types of memories.

Finally, the BusError output is used to end an undecoded memory cycle. In the R3051, Rd is negated one-half cycle after the BusError input is asserted.

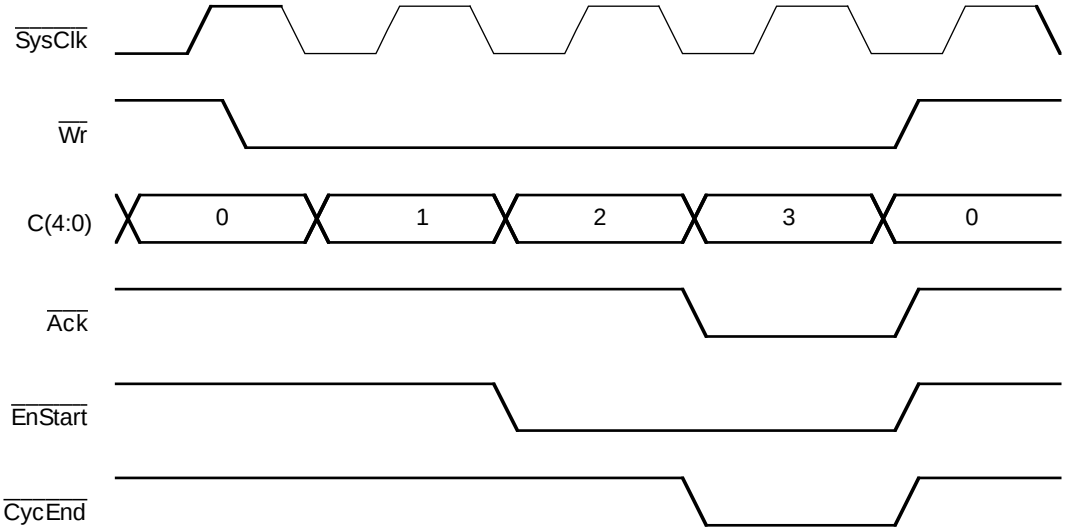


Figure 2. Timing of CycEnd

Other Approaches

Of course, alternative methods and techniques to memory interfacing with an R3051 family CPU exist. Four approaches easily implemented in discrete components include:

- using a SysClk based CycEnd counter (as used in this example)
- using asynchronously resettable registers for the counter
- using interlocking SysClk and SysClk registers
- using an unbuffered SysClk

All of these methods can be used to design for the clocking scheme of the R3051 Family, which uses both the rising and falling edges to control its outputs. The use of both edges of the clock allows the R3051 to mitigate the 1 clock inter-transaction latency that is associated with most other CPUs that need the extra clock to fixup and start new memory cycles. However,because the R3051 Family asserts and de-asserts its edges the same way on both Rd and Wr cycles, specific methods can be employed so that the memory system is always clocked from one edge of SysClk. An example of this is the CycEnd method used on this board, which ignores the edges that are not synchronized with the state machine. Although traditional high-performance CPUs require complex state machines to operate efficiently, the beauty of the R3051 family is the simplicity of its interface. Memory control state machines for the R3051 family are really only minor variations on traditional wait-state machines, and can also easily take advantage of the 1/2 clock inter-transaction savings provided by the CPU interface.

Each of the four approaches has advantages as well as drawbacks relative to each other. The following paragraphs will give a brief description of each technique. Each of the methods could be used by themselves or combined with one

or more of the other methods, to achieve the optimal price/performance/parts count for a given application. Systems employing dedicated interface chips (such as the IDT R372x family, or customer specific ASIC or Gate Array devices), may choose to make different trade-offs than those using discrete component based solutions.

Using SysClk and generating a Cycle End indicator

The SysClk based CycEnd approach as described above is straightforward because of its similarity to traditional wait-state machines. As mentioned above, it does not require the terminating edge of Rd or Wr to complete a transaction.

The system implemented in this design example is limited in speed by:

tclk/2 >= t240 + tpalco + t3051setup + tcap + twire

which works out to 28 MHz for a 10 nsec 16V8, over 40 MHz for a 5 nsec 16R8 PAL, and 33 MHz for a 10 nsec 22V10 PAL.

Using Asynchronous Reset to terminate the Cycle Counter

The second potential method, which uses an asynchronous reset to terminate the cycle, requires AND'ing together Rd and Wr into the the reset line of the counter C(4:0) and can be demonstrated by reprogramming the PAL on the example board. The reset-to-valid output, reset width, and the reset recovery time to clock are among the speed limiting paths in this approach when implemented in PALs. Unfortunately, the reset-to-output delay of a PAL is usually less optimized and relatively slow.

tasyncreset <= tclk/2 - trdn - tcap - twire

For example, a 20 MHz system would require a reset-to-output delay of 17ns, which can be found in a 10 nsec 22V10 PAL (with a 15 nsec reset to valid output data time).

Using interlocking PALs clocked on opposite edges

The third potential approach uses a SysClk based register to detect asserting edges and a SysClk based register to detect de-asserting edges. The outputs of each of the PALs interlock by controlling the outputs of the other PALs. This allows the flexibility of seeing all edges and being able to control outputs optimally by using any 1/2 clock edge (such as output enables). Such an approach obviously requires more PALs, and is somewhat speed limited by:

tclk/2 >= t240 + tpalco + tpalsetup + tcap + twire

which works out to 20 MHz for a 10 nsec 16V8 PAL.

In systems using chips designed specifically to interface to the R3051 family (such as the IDT R3721 DRAM controller), this approach is simpler to implement and leads to the highest levels of performance.

Using an unbuffered SysClk

The fourth potential approach uses an unbuffered SysClk based state machine. This leads to the requirement of having 0 hold time on the registers as well as a 2 nsec minimum propagation delay time to meet the R3051 timing requirements (note that using a buffered SysClk instead of the unbuffered version would require negative hold time on the registers). Despite these restrictions, some PALs can be found that meet all of these requirements. This approach leads to a one cycle latency in reacting to R3051 output assertions. An asserting Rd or Wr would be seen a clock too late to bring RdCEn or Ack low during their first possible sampling clock. Using an unbuffered SysClk has a speed advantage over the other techniques:

tclk >= tpalco + t3051setup + tcap + twire

tclk/2 >= t3051prop + tpalsetup + tcap + twire

which can support designs of 35 MHz for a 10 nsec 16V8 PAL and well over 40 MHz with a 7.5 nsec 16R8 PAL.

An additional consideration relative to using an unbuffered SysClk is the amount of loading placed on the clock, and the impact of additional loading on R3051 AC parameters. Of course, when using a single chip memory controller such as the IDT R3721 or a customer designed ASIC, these loading considerations are minimal.

In summary, the R3051 Family uses both edges of the clock to assert control signals in order to reduce inter-transaction delay between external bus cycles. However, by using one or a combination of the above techniques in a design, a traditional wait-state machine can still be used with the addition of only minor variations.

Read and Write Enables

The Read and Write Enables PAL, MEMEN.JED, uses EnStart and CycEnd to control the initiation and length of the output enable and write enable assertions. Rd and Wr are used to select between read and write cycles. Note that it would have been possible to combine individual bank selects with the address decoder PAL, rather than use a distinct PAL to control the timing of the assertion of Write and Read Byte Strobes.

On read cycles, RdEn is asserted as the system's primary output enable signal. RdDataEn is used to enable the FCT623T data transceiver bank. RdDataEn in most systems would simply be 'DataEn' as supplied straight from the processor. This system provides RdDataEn in case other transceiver banks are added to the system.

The byte enables are used to support partial word writes which are used during byte, halfword, and tri-byte operations. Write cycles combine the byte enables, BE(3:0), with Wr, EnStart, and CycEnd to form the write enable outputs WrEn(D:A) which are attached to the byte banks within the memory system. Whether or not the system is Little or Big Endian, WrEn(A) is always attached to the LSB. WrEn(D:A) can also be implemented using an FCT257T multiplexer. WrDataEn is used to control the FCT623T data transceiver bank and must be held extra long to provide memory data hold time.

Finally, the Byte Enable PAL also has a synchronized PowReset output called Reset and a "guarded" GUARTCS. The guarded chip select, GUARTCS is an example of interfacing R3051 signals to a Motorola-type I/O Device as opposed to an Intel-type I/O Device.

Motorola-type devices multiplex their read/write input pin and expect a data strobe pin to validate the data out or to latch the data in, while Intel-type devices have separate read and write strobes. Since the MC68681 DUART is a Motorola device, the data strobe must start late and end early, so that read/write is held throughout that period. Additionally, the MC68681 uses its chip select pin as a data strobe. As a data strobe, it is important not to have decoder glitches on the chip select since reads in I/O devices are often used to update

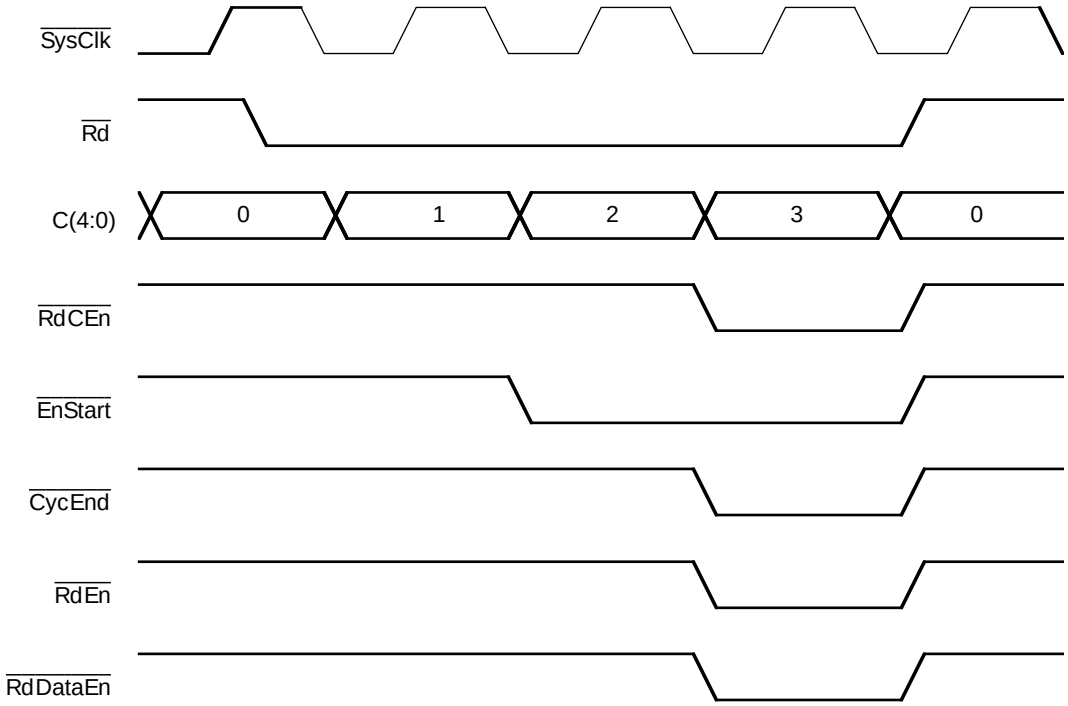


Figure 3. Timing Diagram of RdEn

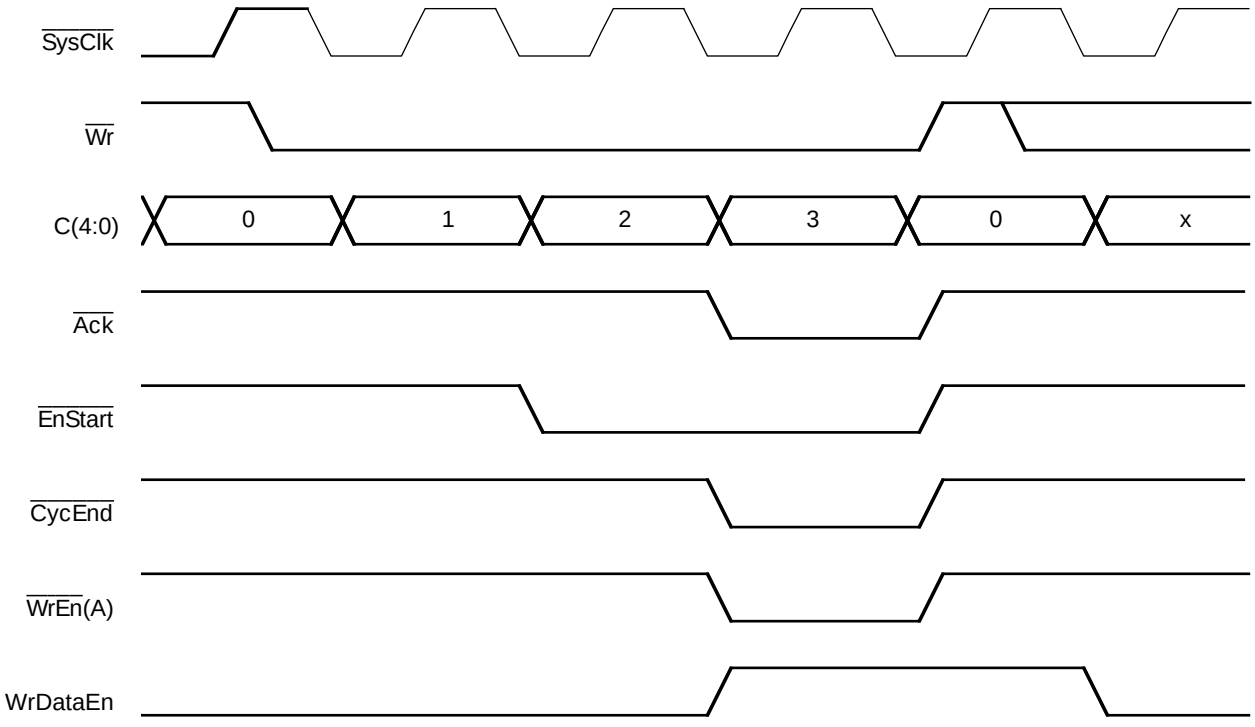


Figure 4. Timing Diagram of WrEn(A)

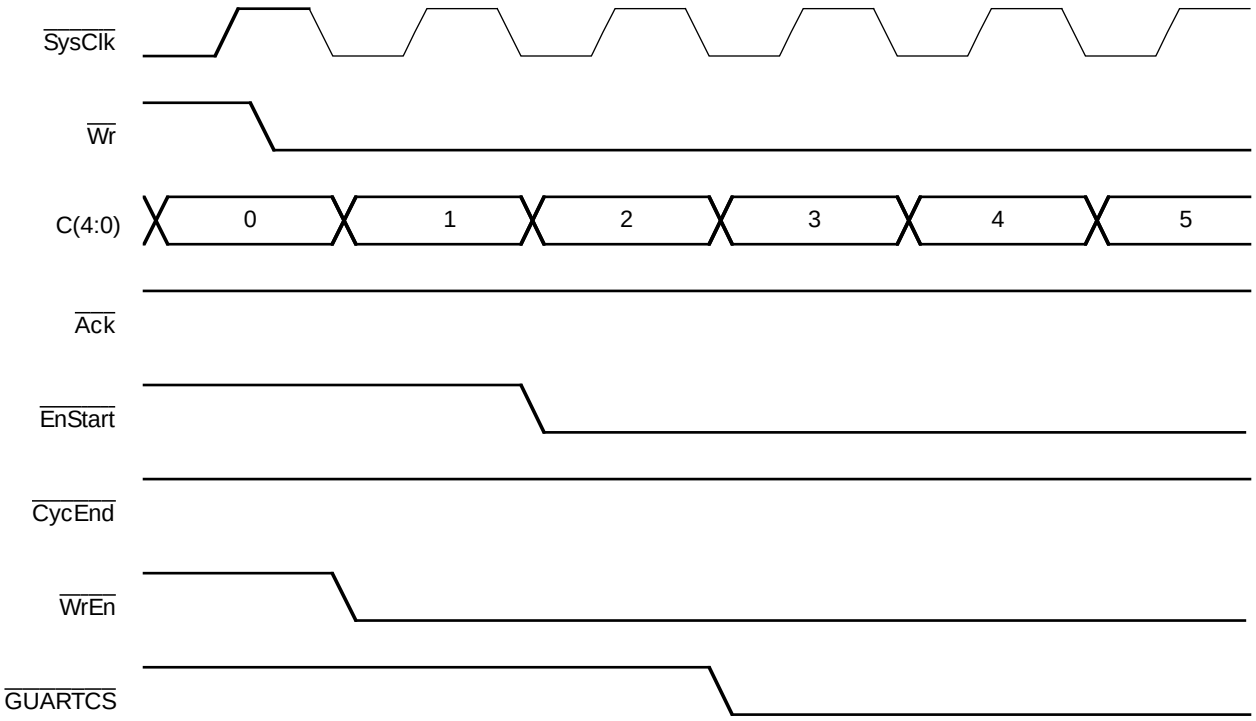


Figure 5. Timing Diagram of Start of GUARTCS

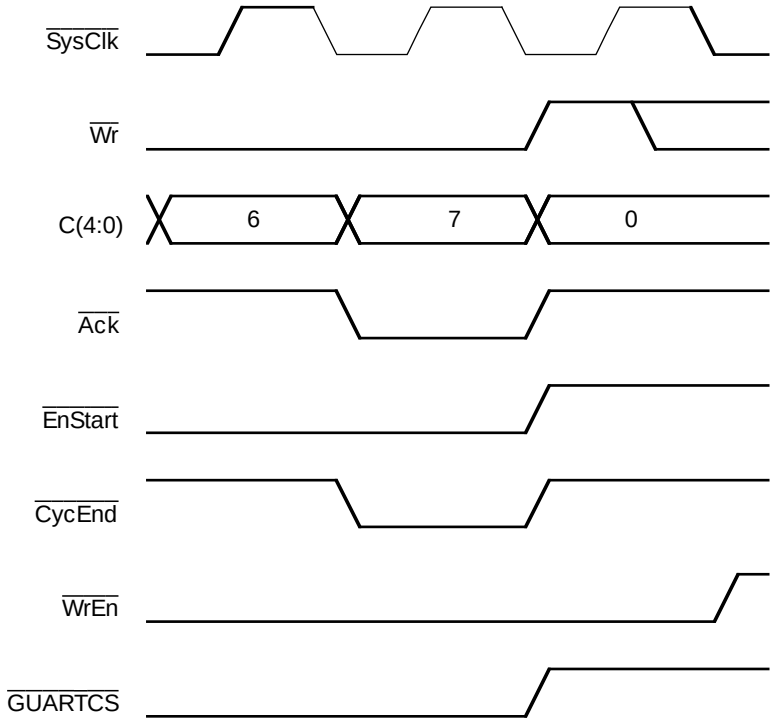


Figure 6. Timing Diagram of End of GUARTCS

FIFO pointers. Thus, the guarded GUARTCS uses EnStart and CycEnd to shorten up UARTCS. Finally, WrEn is provided to extend Wr to allow additional data hold time at the end of the write cycle. WrEn could easily be inserted with another OR term into WrEn(A).

Address Latch and Transceiver De-multiplexer

The address latch bank consists of four FCT373T 8-bit transparent latches. ALE is used for the latch enable on the FCT373T's. The transparent phase allows extra address decoding time during the time that ALE is high; the outputs of the latches are fed directly to the address decode PAL and to the memory devices. In order to insure that address hold time to the latches are met, it is important to take care with the use of the ALE signal. The number and length of the ALE traces is critical and should be kept to a minimum.

Rather than use FCT373's, DRAM systems may want to use FCT821's or FCT823's, which are wider latches. RAS/CAS address multiplexing can be performed by sequencing the output enables of the latches and having the outputs of the latches tied together and driving the DRAM address bus.

The data transceiver bank on the example board uses four FCT623T 8-bit transceivers. FCT623T's were chosen over the similar 10-bit FCT861's and 9-bit FCT863's simply to reduce pin count. The FCT861/3's provide a more conventional interface, since both output enables are active low, instead of one enable active high, and the other active low as in the FCT623T's. However, since this system uses PALs to control the transceivers, the use of FCT623's poses no additional complexity to the design.

FCT623T's were selected instead of FCT245's because of the ease of interfacing to dual output enable pins instead of a direction and enable pins as in the FCT245. Interfacing with FCT245 controls would ideally require that the direction control only be changed when the output enable is disabled. This requires extending a combined (latched) Rd and Wr based signal for an extra cycle at the end of a memory transaction, which may be the beginning of the next memory cycle. Unless the direction pin is controlled with a SysClk based state machine, a signal like EnStart would be necessary to keep the enable pin de-asserted in the subsequent cycle until the direction pin control becomes valid. Some systems with high noise tolerance, e.g., IBM-PC adapter boards, forgo the extra cycle ideal and simply bus contend for a very short time (a few ns) into its memory system by having the read strobe directly control the direction. DataEn, output from the CPU, can be used in such systems to simplify control signal generation.

When there are no pending DMA, read, or write requests, the R3051 tri-states the A/D(31:0) bus during these non-bus clock cycles to reduce power consumption. One can optionally add external pullup or pulldown resistors so that the A/D(31:0) bus is always defined for board level ATE and so that the input pins of the latches and transceivers are stabilized.

Finally, systems that can output disable (oe to Z-state) all memory readable devices within:

tdisable < tclk/2 - t3051dataenn + taddr - tcap - twire

might not require the transceiver bank and thus could reduce the parts count by 4.

EPROM and Static RAM Memory

The memory on the example board is populated with 125 nsec Erasable PROMs (EPROMs) and 45 nsec Static RAMs (SRAMs). Four 27C256 32Kx8 EPROMs are used to form 128K bytes of ROM. The EPROMs are placed in sockets and thus can easily be removed for reprogramming or replacement; alternative designs may wish to add circuitry to allow in-board programming of the EPROMs (e.g. Flash Erase EPROMs).

The EPROMs have a relatively long output disable time (oe to z-state), typical of ROMs and thus require data buffers to prevent contention on the multiplexed AD(31:0) bus, since the following equation is not met:

tclk/2 >= tdisablecontrol + tdisable - taddr + tcap + twire

In addition, the disable time for these EPROMs is long enough that, except for relatively slow systems (under 20 MHz), extra clocks need to be added to the next bus cycle to prevent bus contention with other memory banks. This is determined by:

tclk >= tdisablecontrol + tdisable - tdata + tcap + twire

The SRAM bank is formed using four IDT71256 32Kx8 SRAMs for a total of 128K bytes. The RAM chips have common data I/O pins, separate read and write strobes, and chip selects. RAMs without a separate read strobe (output enable pin) may require more complex address decoding when used in a multiple bank configuration.

DUART, Timer, and Interrupt Controller

An MC68681 DUART and an MAX235 RS232 transceiver are used to form two RS232 serial communication links. The DUART control registers are word addressed, but only D(7:0) are used. The MC68681 is an example of a Motorola-type I/O interface as explained above.

An iP8254 timer/counter chip is used for a real-time clock or timer. The iP8254 is an example of an Intel-type I/O interface. The iP8254's need for separate read and write strobes matches up well with the R3051.

Software control of these chips is best described by their respective data sheets. Typically, most software programs for the 7RS382 have used the DUART in a polling mode and the timer in a square wave mode. Interrupts Int(5:3) are controlled by UARTIntOC, Timer OutB, and Timer OutA respectively from MSB to LSB. The 16R8 PAL, called MEMINT.JED, is used to control these interrupts latches in the assertion transition of the original interrupt lines.

The controller holds the interrupt line to the processor for Timer A and Timer B until they are acknowledged (as required by the R3051). Acknowledgement is indicated by reading the interrupt controller at Virtual Address BF800010 and BF800014 (Physical Address 1F800010 and 1F800014) respectively. This action incidentally reads extraneous data from the Timer chip itself on D(7:0). The DUART interrupt must be acknowledged by using the DUART control registers.

The output disable to data in z-state time for these I/O peripherals is relatively long, as is typical for I/O devices. This forms the critical timing path for the placement of $\overline{\text{EnStart}}$ in the Memory Controller and Memory Enable PALs.

Expansion Connector

Two 50-pin connectors are provided which bring out the R3051 RISController pins to allow off-board expansion. The $\overline{\text{BusReq}}$ and $\overline{\text{BusGnt}}$ pins are not presently used on this board. If DMA is to be used, the R3051 control outputs $\overline{\text{Rd}}$, $\overline{\text{Wr}}$, $\overline{\text{Burst}}$, $\overline{\text{DataEn}}$, and ALE are pulled high or low so that they remain inactive when tri-stated.

SCHEMATICS AND PAL EQUATIONS

Appendices include the System Design Example Board Schematics and the PAL equations.

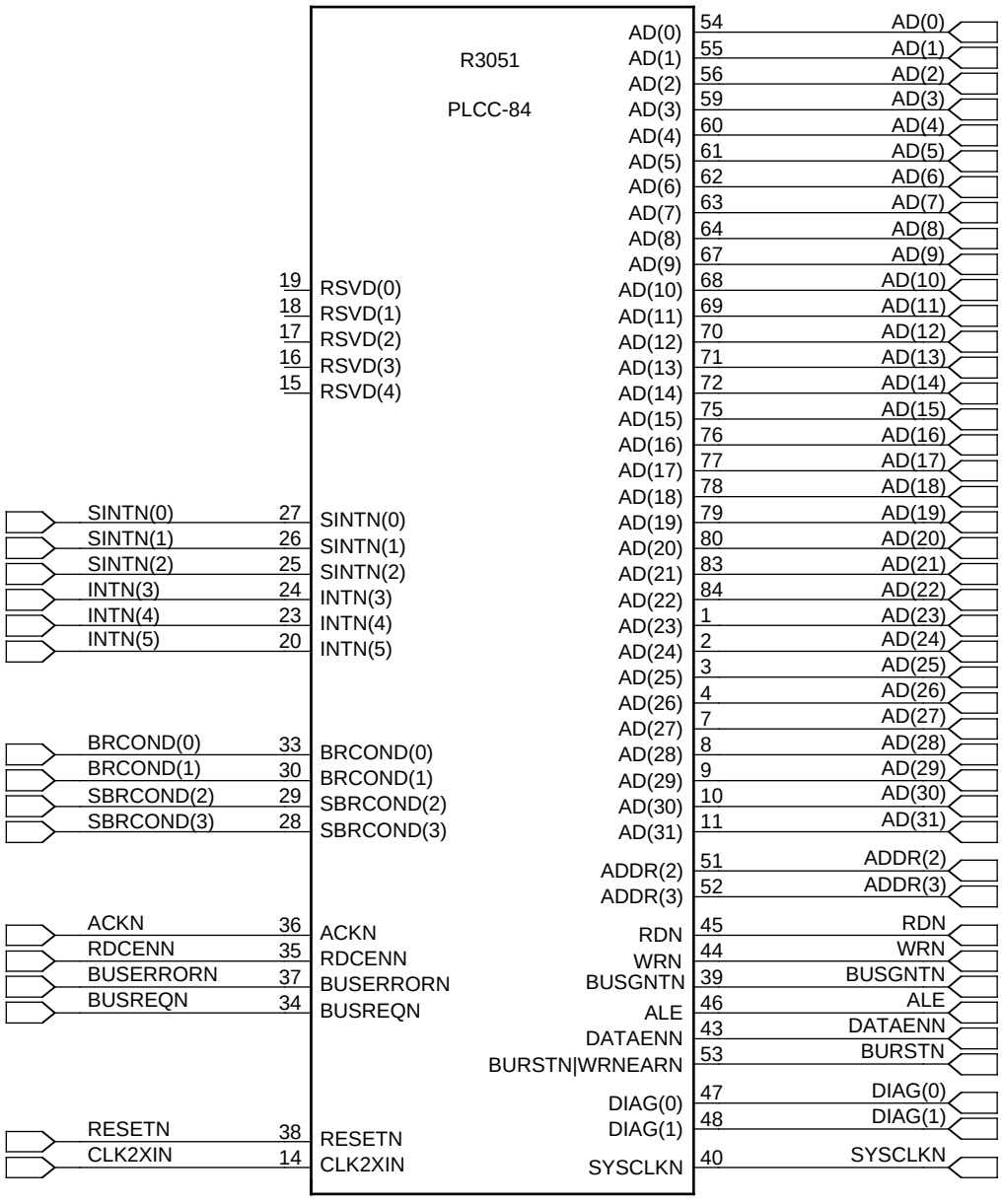


Figure 7. R3051 RISController

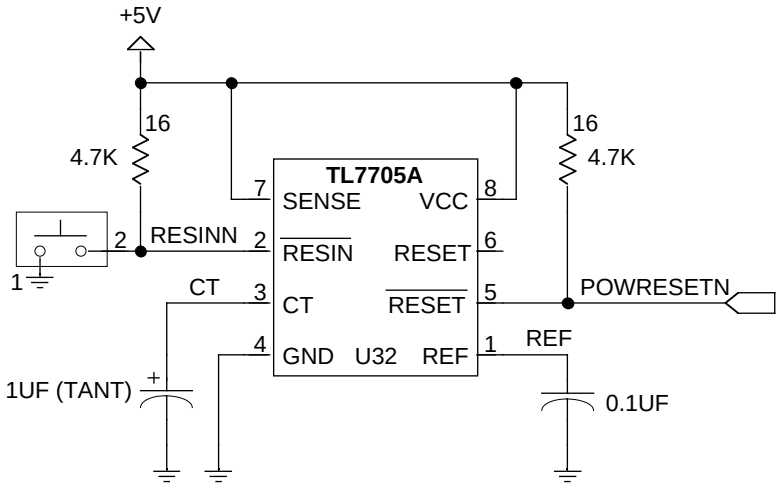


Figure 8. Reset Logic

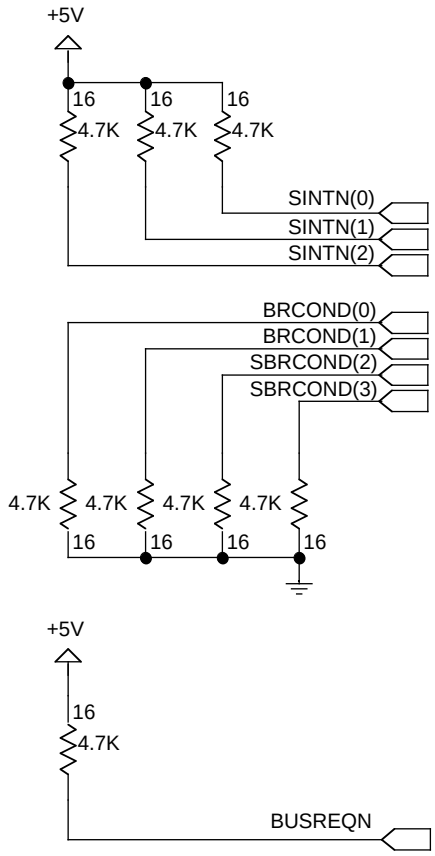


Figure 9. Unused Inputs

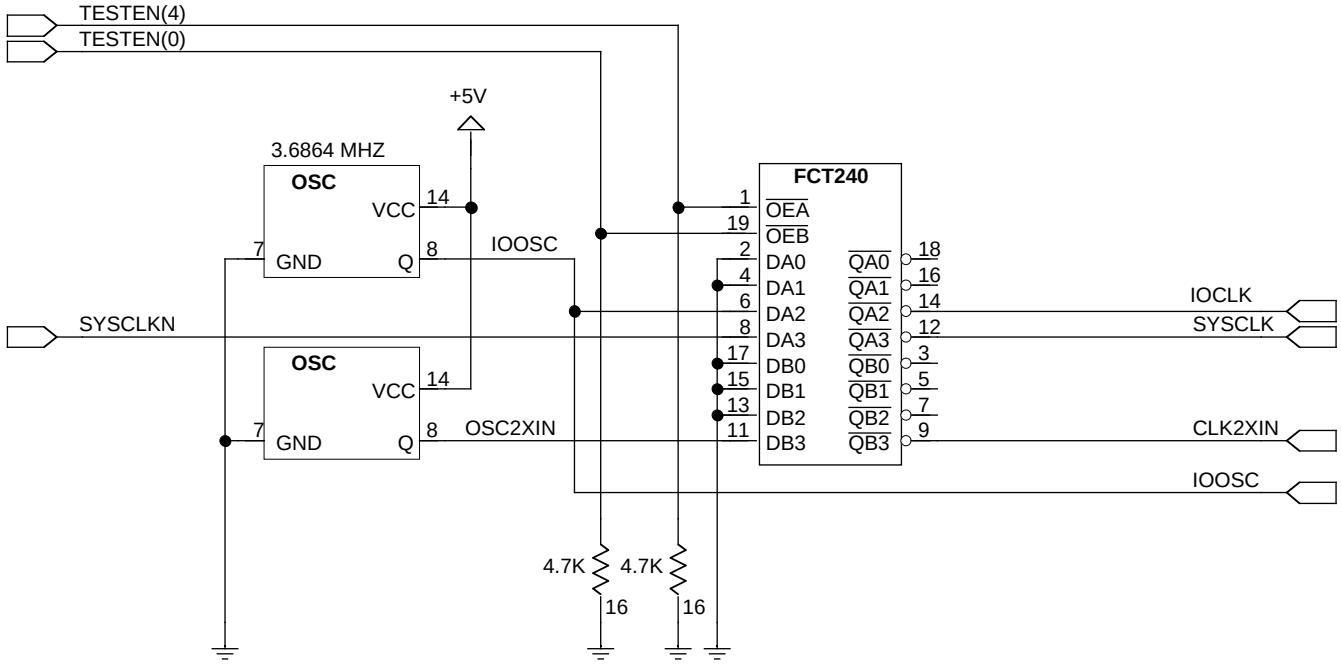


Figure 10. Clock Logic

NOTES:
MEMSPARE0 -- CARDCSN | XCSN0
MEMSPARE1 -- C4 | WRLASTN | WORLDBOOTN
MEMSPARE2 -- TESTEN | SHADOW RAM | DATAENN | XCSN1

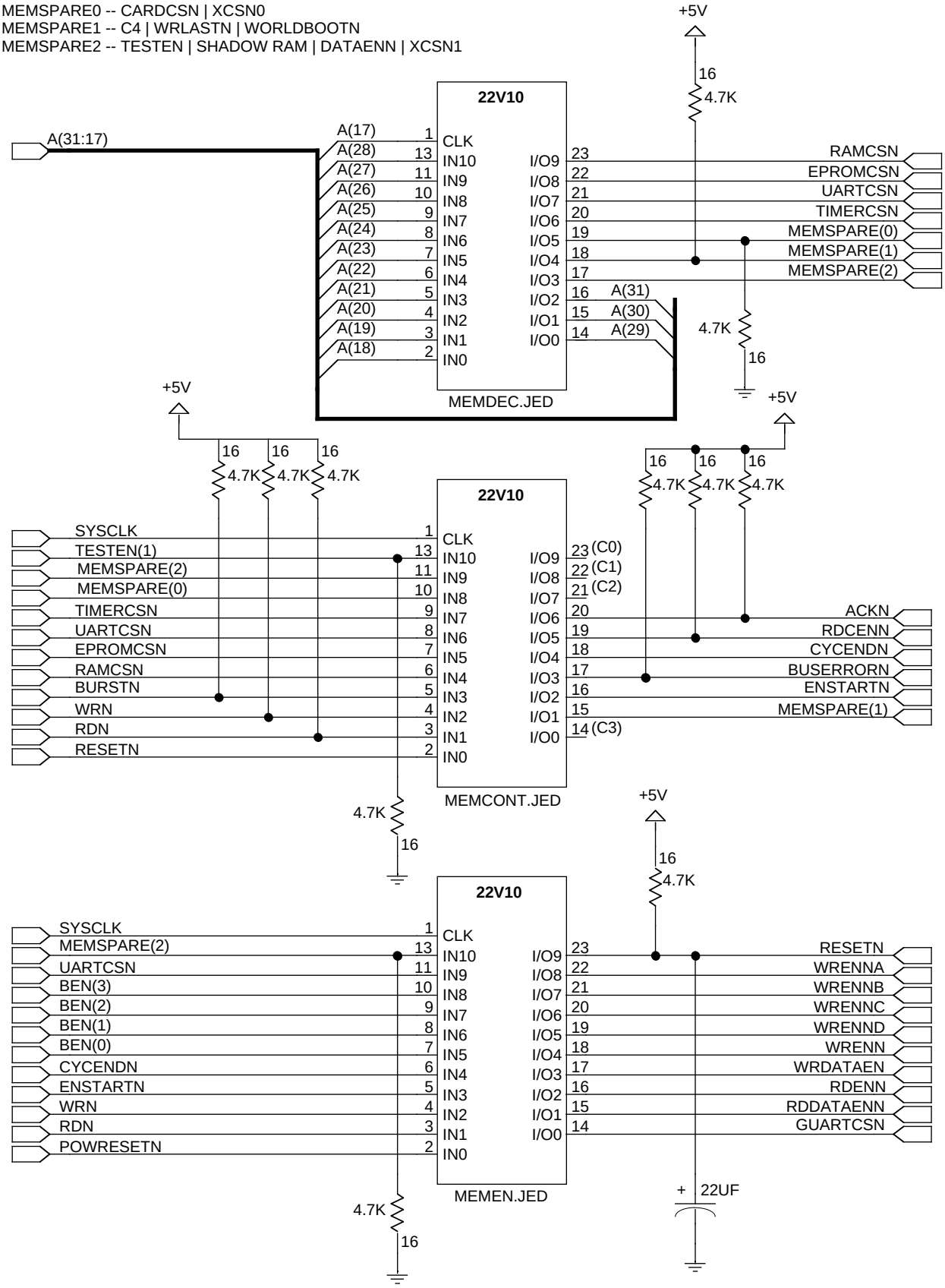


Figure 11. Memory Controller

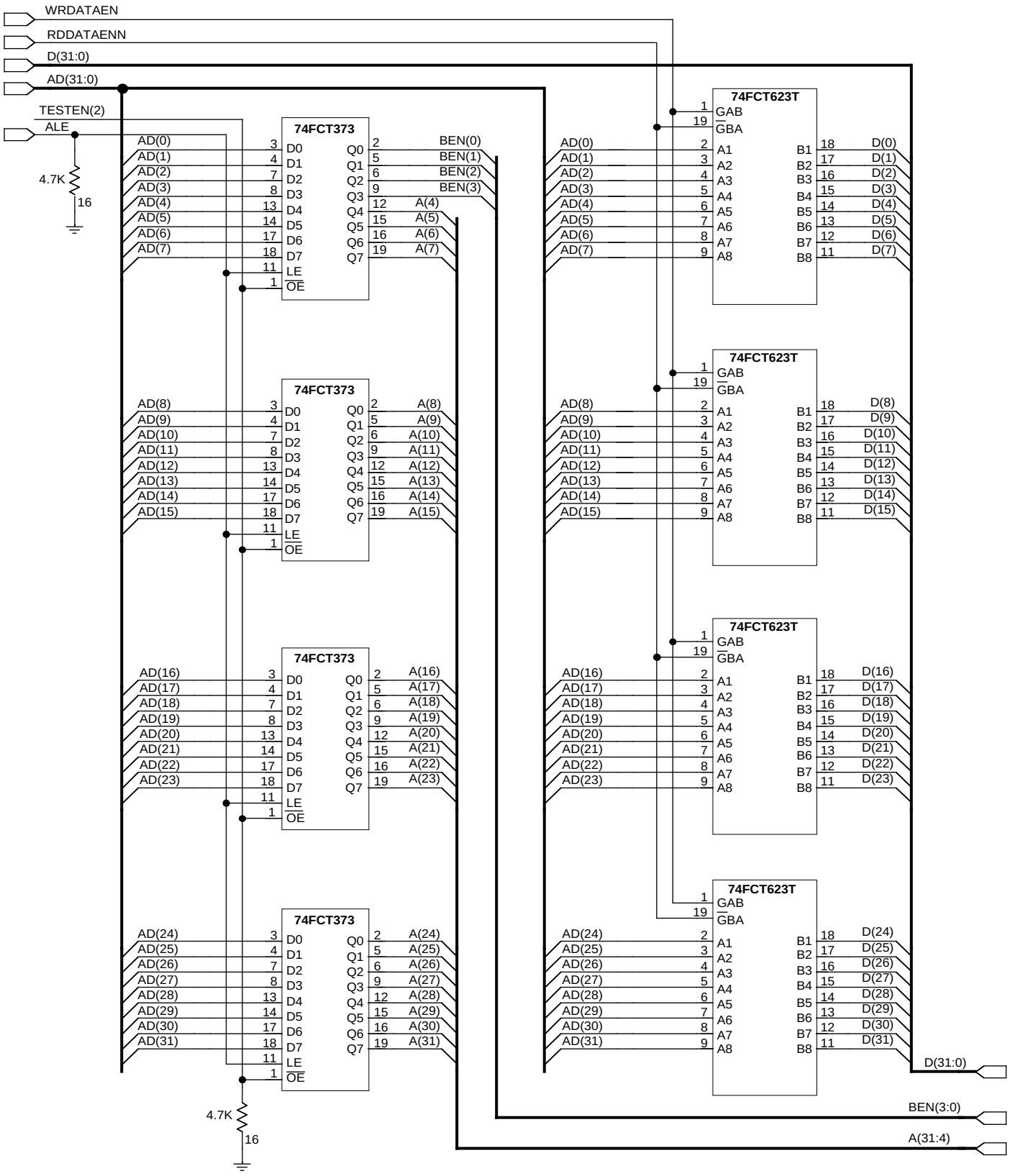
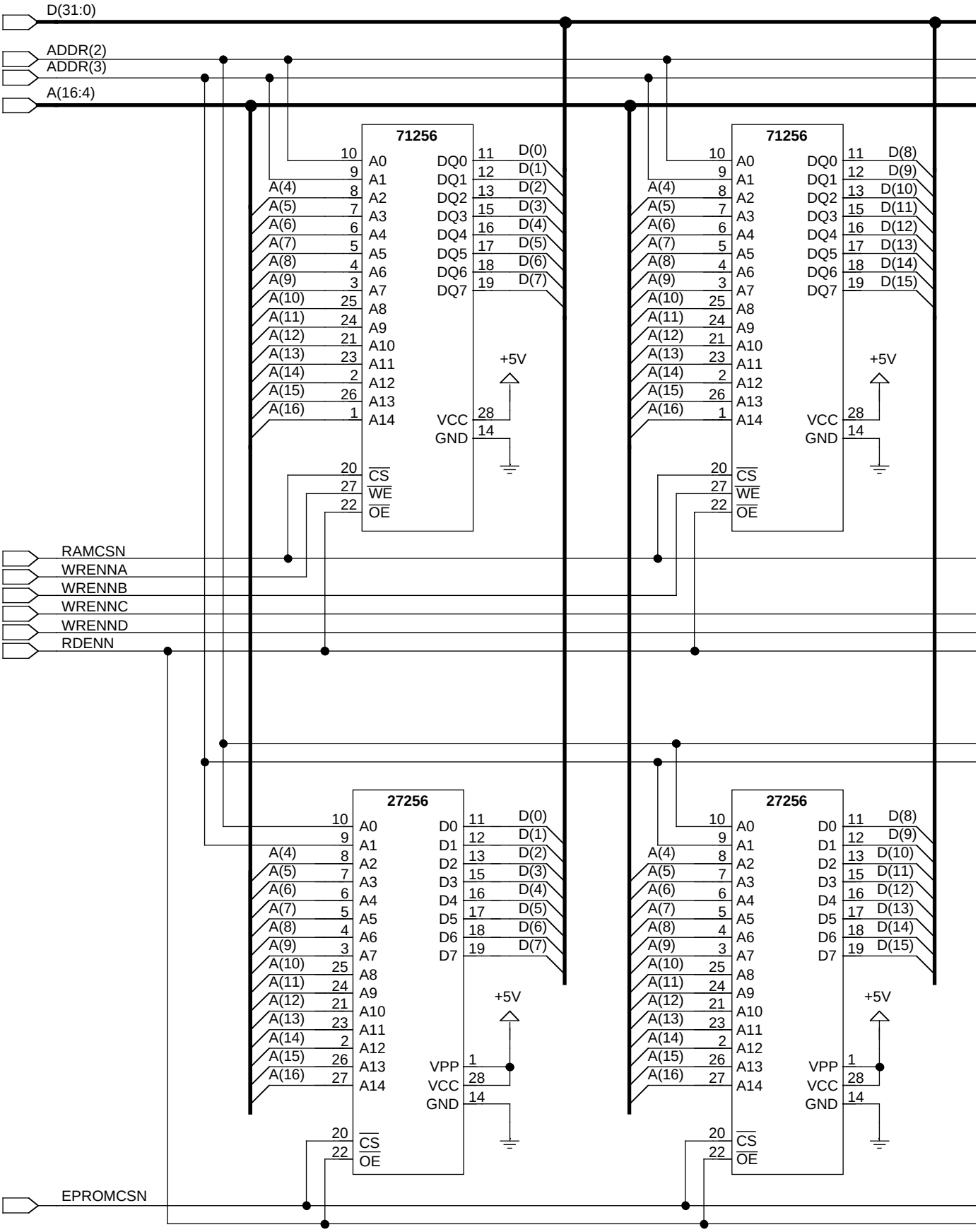


Figure 12. Address Latch Data Transceiver Demultiplexer



NOTE: BANK A - - LITTLE ENDIAN LSB BYTE 0
- - BIG ENDIAN LSB BYTE 3

Figure 13. ROM and Static RAM Memory

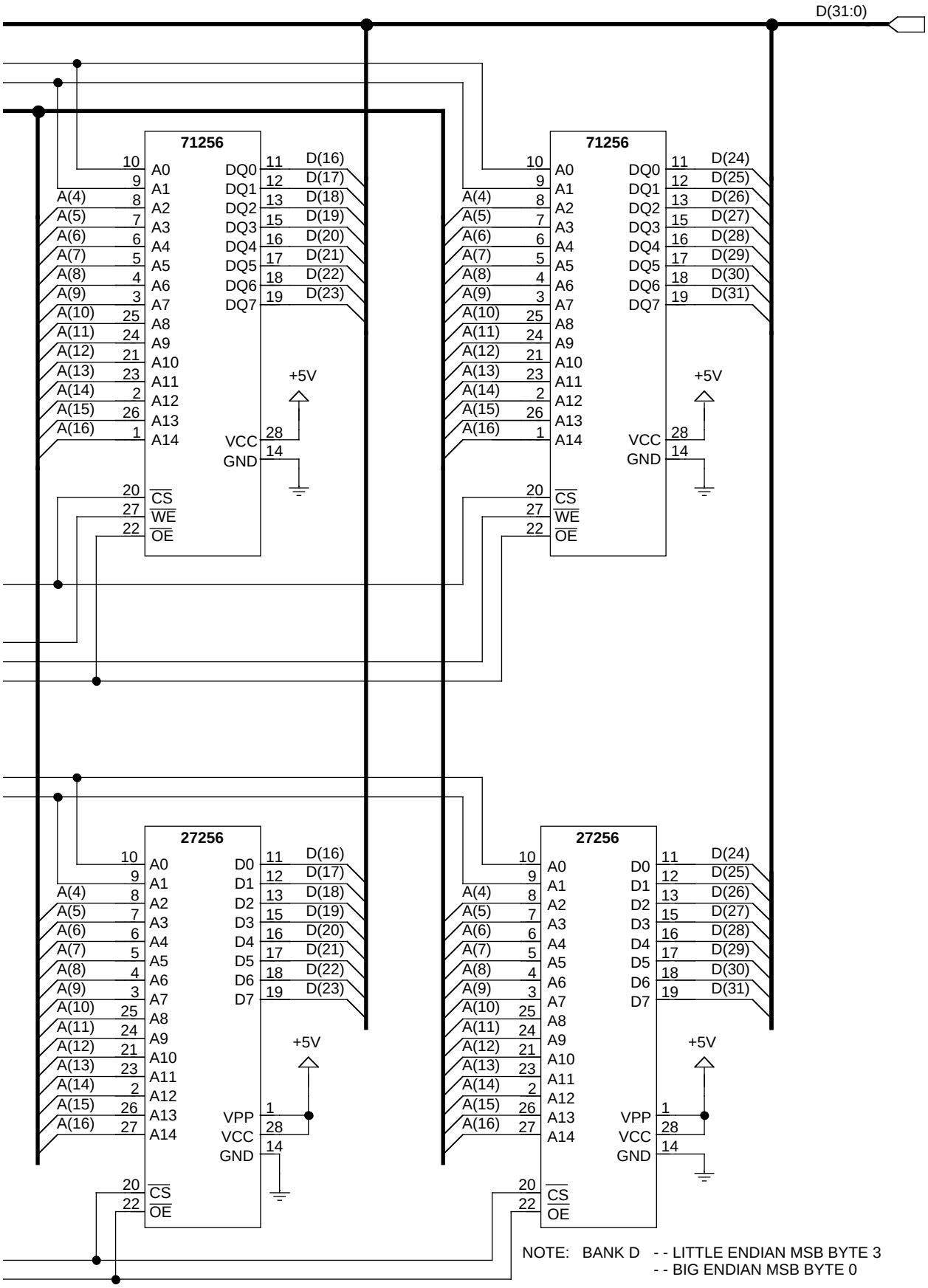


Figure 13. ROM and Static RAM Memory

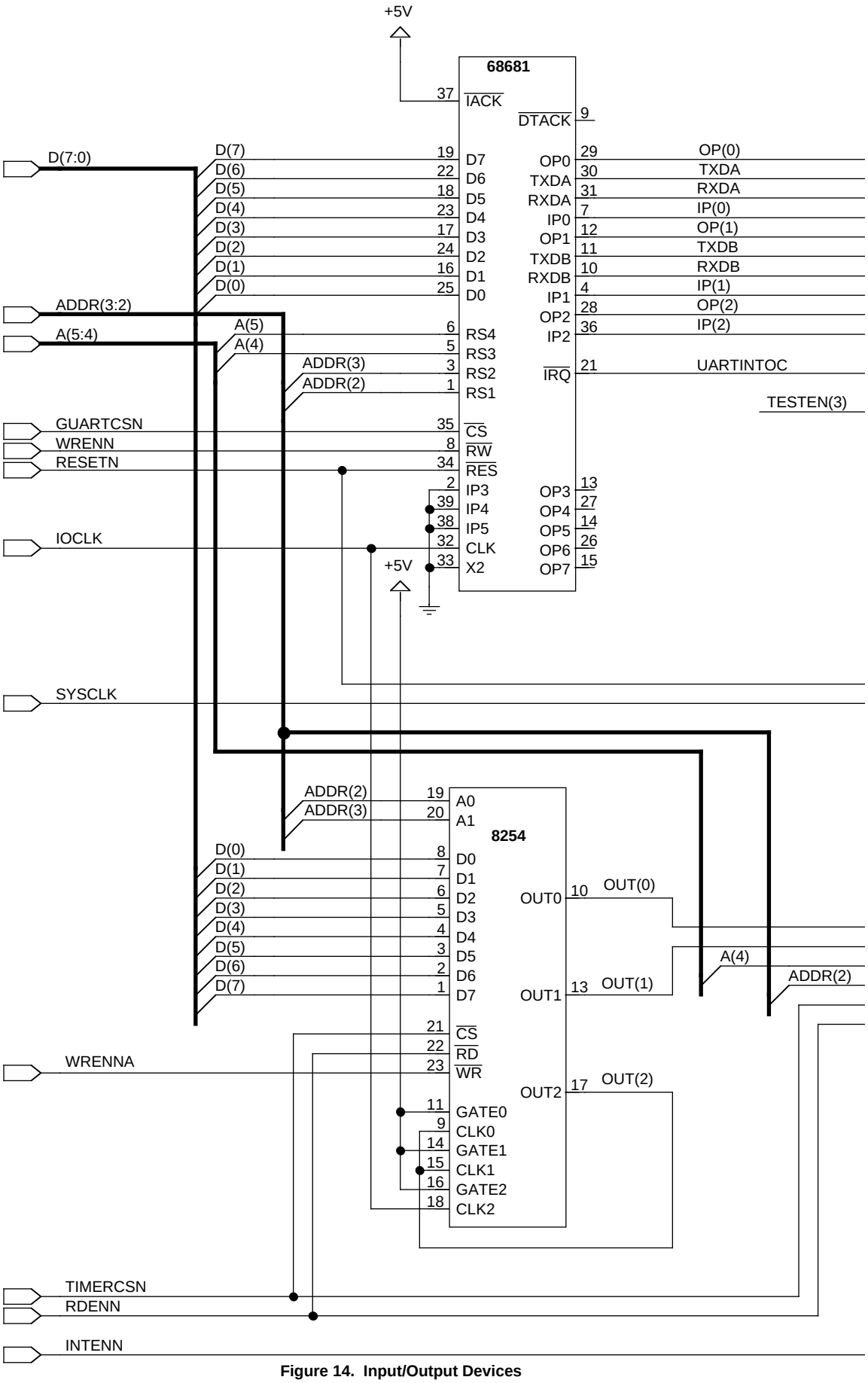


Figure 14. Input/Output Devices

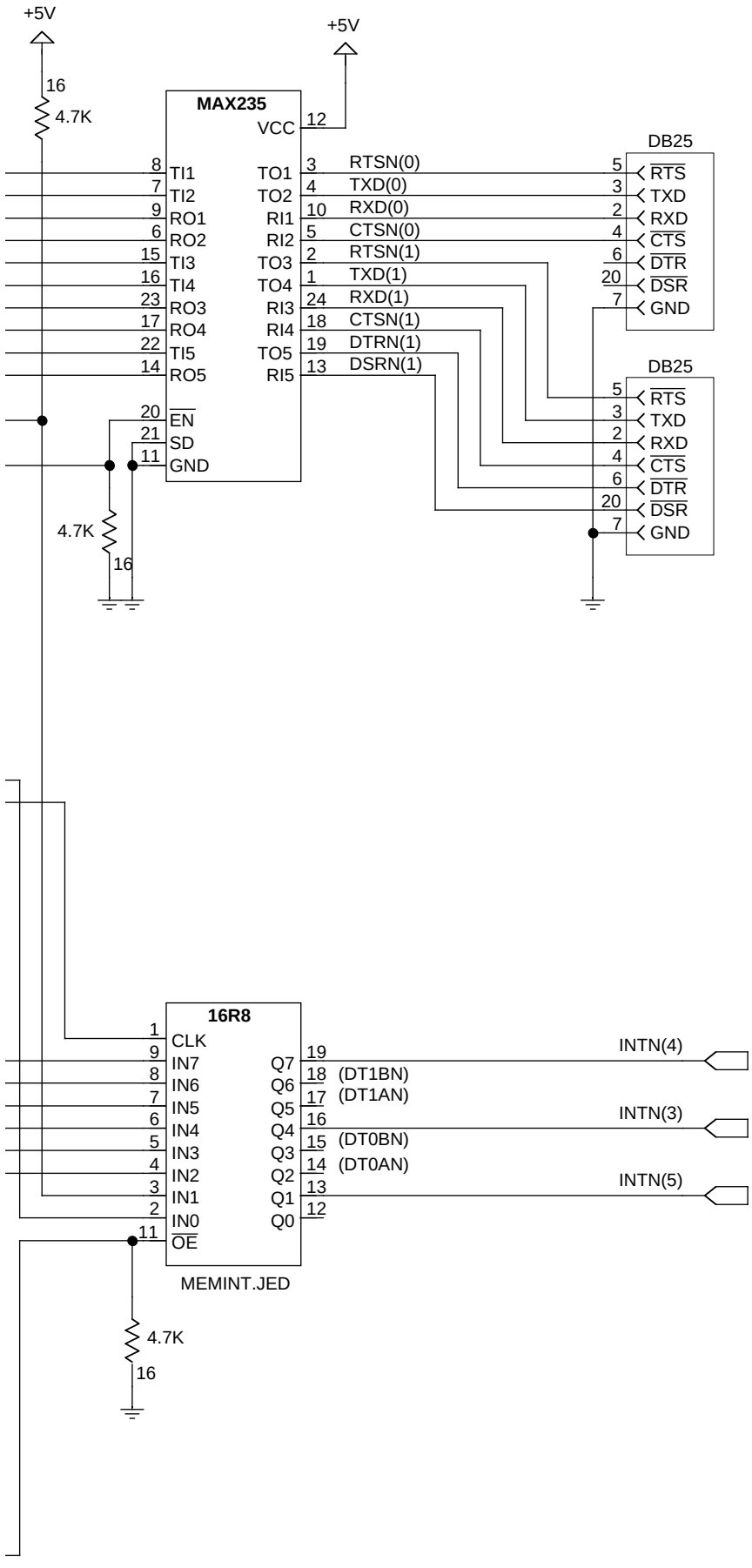


Figure 14. Input/Output Devices

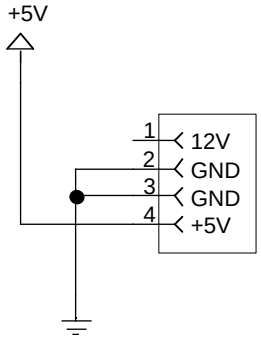


Figure 15. Power Connector

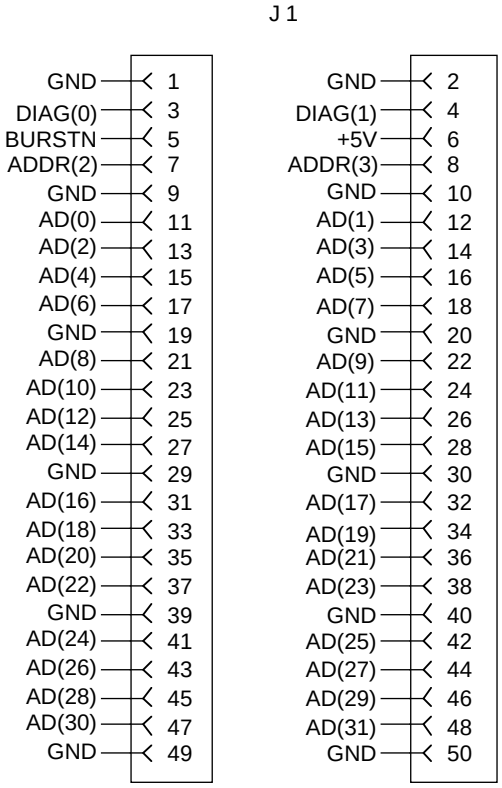


Figure 16. 50-Pin Connector

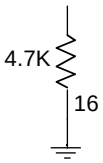


Figure 17. Spares

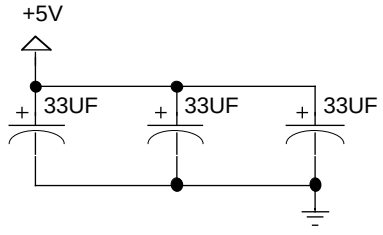


Figure 18. Primary Power Decoupling Capacitors

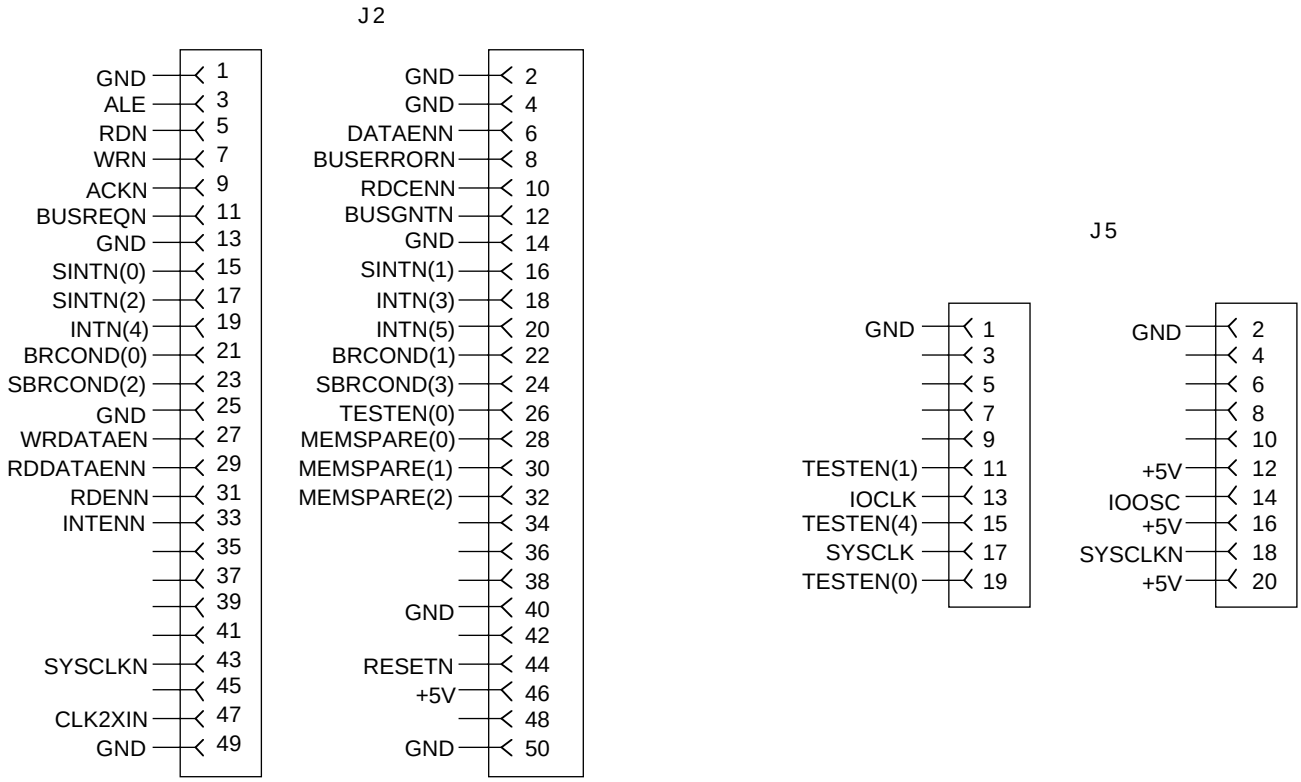


Figure 19. 50-Pin Connector

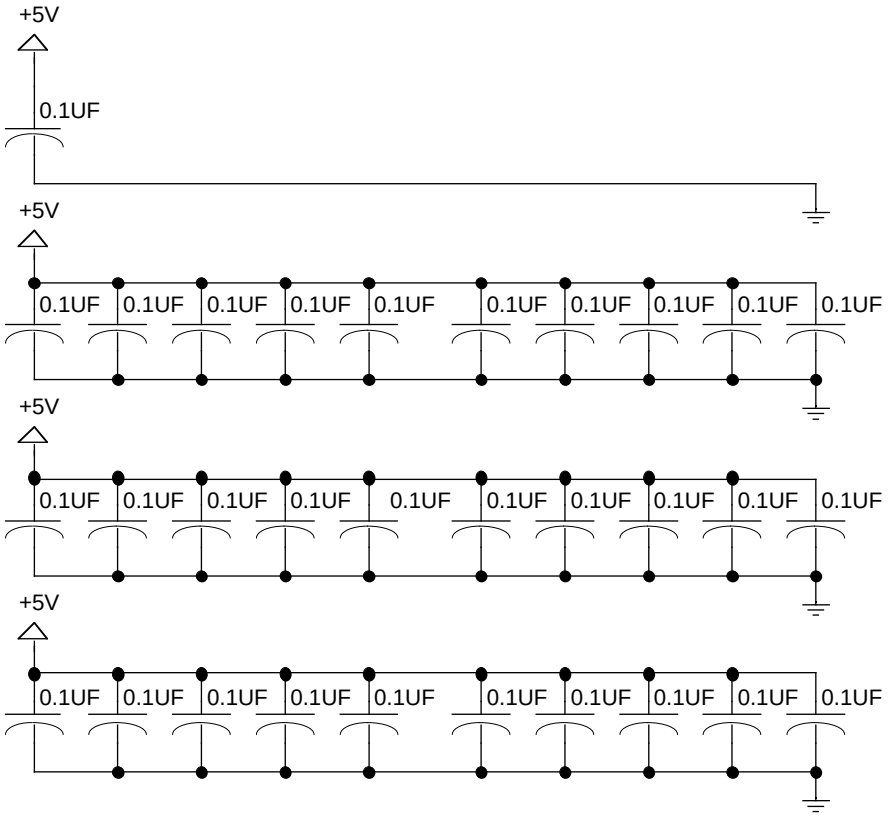


Figure 21. Decoupling Capacitors

```
{ TITLE   : MEMDEC.LPLC
          UPAL1 MEMORY AND I/O ADDRESS DECODER PAL FOR THE R305X
          BEHAVIORAL BUS EMULATOR MEMORY EVALUATION BOARD
  PURPOSE : DECODES DEMULTIPLEXED ADDRESS TO GENERATE CHIP SELECTS.
  LANG    : LPLC — TM OF CAPILANO COMPUTING SYSTEMS
  AUTHOR  : ANDY NG, IDT INC.
  UPDATES : C2503 03-18-91 AP NOTE FIRST RELEASE
}

MODULE UPAL1     ;
TITLE  UPAL1     ;
TYPE   AMD 22V10 ;

INPUTS ;
      {  DEMULTIPLEXED MEMORY ADDRESS LINES }
      A17          NODE[PIN1]  ;         { MSB ADDRESS LINES 31-17      }
      A18          NODE[PIN2]  ;
      A19          NODE[PIN3]  ;
      A20          NODE[PIN4]  ;
      A21          NODE[PIN5]  ;
      A22          NODE[PIN6]  ;
      A23          NODE[PIN7]  ;
      A24          NODE[PIN8]  ;
      A25          NODE[PIN9]  ;
      A26          NODE[PIN10] ;
      A27          NODE[PIN11] ;
      A28          NODE[PIN13] ;


      { OUTPUT FEEDBACK NODES (NEEDED FOR LPLC'ISM) }
      A29          NODE[PIN16] ;
      A30          NODE[PIN15] ;
      A31          NODE[PIN14] ;
      MEMSPARE0      NODE[PIN19] ;
      MEMSPARE1      NODE[PIN18] ;
      MEMSPARE2      NODE[PIN17] ;

OUTPUTS ; { ATTRIBUTES C — COMBINATIONAL, R — REGISTERED, H — HIGH, L — LOW }

      { CHIP SELECTS }
      RAMCSN         NODE[PIN23] ATTR[CL] ;  { STATIC RAM CHIP SELECT       }
      EPROMCSN       NODE[PIN22] ATTR[CL] ;  { EPROM CHIP SELECT            }
      UARTCSN        NODE[PIN21] ATTR[CL] ;  { UNGATED UART CHIP SELECT     }
      TIMERCSN       NODE[PIN20] ATTR[CL] ;  { TIMER CHIP SELECT            }

      { I/O PINS USED AS INPUTS }
      A29          NODE[PIN14] ATTR[CL] ;  { MSB ADDRESS LINES 31-17      }
      A30          NODE[PIN15] ATTR[CL] ;
      A31          NODE[PIN16] ATTR[CL] ;
      MEMSPARE0      NODE[PIN19] ATTR[CL] ;
      MEMSPARE1      NODE[PIN18] ATTR[CL] ;
      MEMSPARE2      NODE[PIN17] ATTR[CL] ;

      { OUTPUT ENABLES }
      RAMCSNEN       NODE[PIN23EN] ;
      EPROMCSNEN     NODE[PIN22EN] ;
```

```
UARTCSNEN       NODE[PIN21EN] ;
TIMERCSNEN      NODE[PIN20EN] ;
A29EN           NODE[PIN14EN] ;
A30EN           NODE[PIN15EN] ;
A31EN           NODE[PIN16EN] ;
MEMSPARE0EN     NODE[PIN19EN] ;
MEMSPARE1EN     NODE[PIN18EN] ;
MEMSPARE2EN     NODE[PIN17EN] ;

{ ASYNCHRONOUS RESET AND SYNCHRONOUS PRESET NODES }
RESETEN         NODE[RESET]   ;
PRESETEN        NODE[PRESET]  ;


{ 7RS382 COMPATIBLE PHYSICAL ADDRESS DECODE MAP }
{   RAM      00000000H — 0001FFFFH    32K    }
{   EPROM    1FC00000H — 1FC1FFFFH    32K    }
{   UART     1FE00000H — 1FE0003FH           }
{   TIMER    1F800000H — 1F80002CH           }


TERMS ; { LPLC “TABLE” ALGORITHM TAKES TOO LONG TO COMPILE }

{ NOTES: MEMSPARE0 IS BEING USED FOR A BOARD CHIP SELECT
         DRIVABLE BY ANOTHER MEMORY SYSTEM.  WITHOUT IT
         ASSERTED LOW, THIS BOARD WILL NOT ISSUE ANY MEMORY
         SIGNALS NOR OUTPUT ENABLE SHARED CONTROL PINS.        }
{ NOTES: MEMSPARE1 IS NOT BEING USED.  IT COULD BE USED AS AN
         OUTPUT IF IT OR THE UPAL2 OUTPUT IT IS CONNECTED TO IS
         TRISTATED.                                            }
{ NOTES: MEMSPARE2 IS BEING USED AS A TESTEN INPUT PIN TO
         TRISTATE THE OUTPUTS DURING BOARD TESTING.  ANOTHER
         USE WOULD BE FOR A BOARD CHIP SELECT — MEMCSN.
         MEMSPARE2 IS CONNECTED TO A UPAL3 INPUT PIN.          }

{ I/O PINS USED ONLY AS INPUTS }
A29EN           = 0 ;
A30EN           = 0 ;
A31EN           = 0 ;
MEMSPARE0EN     = 0 ;
MEMSPARE1EN     = 0 ;
MEMSPARE2EN     = 0 ;
A29         NOT = 0 ;
A30         NOT = 0 ;
A31         NOT = 0 ;
MEMSPARE0   NOT = 0 ;
MEMSPARE1   NOT = 0 ;
MEMSPARE2   NOT = 0 ;

{ RESET AND PRESET ARE NOT USED IN THIS PAL. }
RESETEN  = 0 ;
PRESETEN = 0 ;


RAMCSNEN        = !MEMSPARE2 ;
RAMCSN NOT      = !MEMSPARE0 AND
                  !A31 AND !A30 AND !A29 AND !A28
                AND !A27 AND !A26 AND !A25 AND !A24
                AND !A23 AND !A22 AND !A21 AND !A20
```

```
                AND !A19 AND !A18 AND !A17
    ;


    EPROMCSNEN      = !MEMSPARE2 ;
    EPROMCSN NOT    = !MEMSPARE0 AND
                      !A31 AND !A30 AND !A29 AND  A28
                    AND  A27 AND  A26 AND  A25 AND  A24
                    AND  A23 AND  A22 AND !A21 AND !A20
                    AND !A19 AND !A18 AND !A17
    ;


    UARTCSNEN       = !MEMSPARE2 ;
    UARTCSN NOT     = !MEMSPARE0 AND
                      !A31 AND !A30 AND !A29 AND  A28
                    AND  A27 AND  A26 AND  A25 AND  A24
                    AND  A23 AND  A22 AND  A21 AND !A20
                    AND !A19 AND !A18 AND !A17
    ;


    TIMERCSNEN      = !MEMSPARE2 ;
    TIMERCSN NOT    = !MEMSPARE0 AND
                      !A31 AND !A30 AND !A29 AND  A28
                    AND  A27 AND  A26 AND  A25 AND  A24
                    AND  A23 AND !A22 AND !A21 AND !A20
                    AND !A19 AND !A18 AND !A17
    ;


END;
END UPAL1.
```

```
{ TITLE  : MEMCONT.LPLC
        UPAL2 MEMORY CONTROLLER PAL FOR THE R305X BEHAVIORAL BUS EMULATOR
        MEMORY EVALUATION BOARD
  PURPOSE: PRODUCES READ, WRITE, AND BUS ERROR ACKNOWLEDGE CONTROLS (RDCENN,
        ACKN, BUSERRORN) BASED ON A 4 OR 5 BIT COUNTER AND CYCLE END
        STALL CYCLE (WAIT STATE) EQUATIONS.
  LANG   : LPLC — TM OF CAPILANO COMPUTING SYSTEMS
  AUTHOR : ANDY NG, IDT INC.
  UPDATES: C4B28 03-18-91 AP NOTE FIRST RELEASE
}

MODULE UPAL2     ;
TITLE  UPAL2     ;
TYPE   AMD 22V10 ;

INPUTS ;
      { REGULAR INPUT PINS }
      SYSCLK         NODE[PIN1]  ;        { UN-INVERTED SYSTEM CLOCK     }
      RESETN         NODE[PIN2]  ;        { MASTER RESET                 }
      RDN            NODE[PIN3]  ;        { READ                         }
      WRN            NODE[PIN4]  ;        { WRITE                        }
      BURSTN         NODE[PIN5]  ;        { BURST READ | WRITE NEAR      }
      RAMCSN         NODE[PIN6]  ;        { RAM CHIP SELECT              }
      EPROMCSN       NODE[PIN7]  ;        { EPROM CHIP SELECT            }
      UARTCSN        NODE[PIN8]  ;        { UART CHIP SELECT             }
      TIMERCSN       NODE[PIN9]  ;        { TIMER CHIP SELECT            }
      MEMSPARE0      NODE[PIN10] ;        {                              }
      MEMSPARE2      NODE[PIN11] ;        {                              }
      TESTEN         NODE[PIN13] ;        { TEST PIN TO Z-STATE OUTPUTS  }

      { REGISTER FEEDBACK PINS }
      C WIDTH[5]     NODE[PIN15,PIN14,PIN21,PIN22,PIN23] ;
      ENSTARTN       NODE[PIN16] ;
      CYCENDN        NODE[PIN18] ;
      RDCENN         NODE[PIN19] ;
      ACKN           NODE[PIN20] ;
      BUSERRORN      NODE[PIN17] ;


OUTPUTS ; { ATTRIBUTES C — COMBINATIONAL, R — REGISTERED, H — HIGH, L — LOW  }

      { REGISTERED OUTPUT PINS }
      { BINARY UP COUNTER INPUTS MSB TO LSB C4, C3, C2, C1, C0 }
      C WIDTH[5]     NODE[PIN15,PIN14,PIN21,PIN22,PIN23] ATTR[RL] ;
      ENSTARTN       NODE[PIN16] ATTR[RL] ;  { READ/WRITE OUTPUT ENABLE START }
      CYCENDN        NODE[PIN18] ATTR[RL] ;  { CYCLE END (COMPOSITE ACK)     }
      RDCENN         NODE[PIN19] ATTR[RL] ;  { R305X READ BUFFER CLOCK ENABLE }
      ACKN           NODE[PIN20] ATTR[RL] ;  { R3050X ACKNOWLEDGE           }
      BUSERRORN      NODE[PIN17] ATTR[RL] ;  { R305X BUS ERROR              }

      { OUTPUT ENABLES }
      CEN WIDTH[5]   NODE[PIN15EN,PIN14EN,PIN21EN,PIN22EN,PIN23EN] ;
      ENSTARTNEN     NODE[PIN16EN] ;
      CYCENDNEN      NODE[PIN18EN] ;
      RDCENNEN       NODE[PIN19EN] ;
      ACKNEN         NODE[PIN20EN] ;
      BUSERRORNEN    NODE[PIN17EN] ;
```

```
        { ASYNCHRONOUS RESET AND SYNCHRONOUS PRESET NODES }
        RESETEN        NODE[RESET]  ;
        PRESETEN       NODE[PRESET] ;


    TABLE ;

        { RESET AND PRESET ARE NOT BEING USED.                          }
        RESETEN  = 0 ;
        PRESETEN = 0 ;

        { PURPOSE: PROVIDES REGISTERED VERSION OF RDN AND WRN.

          NOTE:    QRDN AND QWRN ARE KEPT LOW ONE EXTRA CLOCK BY CYCENDN.
                   THIS IS BECAUSE THE RISING EDGE OF RDN OR WRN MAY NOT
                   HAVE ENOUGH HOLD TIME FROM THE RISING EDGE OF
                   (BUFFERRED) SYSCLK.
          NOTE:    QRDN AND QWRN DO NOT NECESSARILY TRANSITION BACK HIGH
                   BETWEEN CONSECUTIVE MEMORY CYCLES, E.G., WRITE FOLLOWED
                   BY A WRITE.                                    }

        { QRDN NOT     :=   RESETN AND (!RDN OR (!QRDN AND !CYCENDN)) ; }
        { QWRN NOT     :=   RESETN AND (!WRN OR (!QWRN AND !CYCENDN)) ; }


        { PURPOSE: C[4]-C[0] PROVIDES A 5-BIT BINARY UP COUNTER.  IT IS RESET
                   ANYTIME RESETN IS ASSERTED AND AT THE END
                   OF EVERY MEMORY CYCLE AFTER CYCENDN IS ASSERTED.
                   IT BEGINS COUNTING UP WHEN A READ OR WRITE CYCLE IS
                   INITIATED.
          NOTE:    CYCENDN IS ASSUMED TO ASSERT WITH THE LAST RDCENN
                   ON READS AND WITH ACKN ON WRITES.  THUS CYCENDN WILL CLEAR
                   THE COUNTER WHETHER OR NOT RDN OR WRN HIGH TRANSITION
                   MEETS THE REGISTER SETUP AND HOLD TIME REQUIREMENTS.        }

        { NOTE:    TO ADD A GENERAL PURPOSE READY (A.K.A. BUSYN AND WAITN)
                   INPUT, CHANGE EACH OF THE COUNTER C[4:0] EQUATIONS SO
                   THAT THEIR VALUE CAN BE HELD WITH AN ADDITIONAL TERM, E.G.:
                   C[0]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                                AND (    (C[0] XOR 1)
                                     OR (C[0] AND !READY) ) ;
                   A READY INPUT CAN BE USED FOR DUAL-PORT MEMORY INTERFACING,
                   EEPROM WRITE INTERFACING, ETC.
        }

        CEN[0]   =   !TESTEN ;
        CEN[1]   =   !TESTEN ;
        CEN[2]   =   !TESTEN ;
        CEN[3]   =   !TESTEN ;
        CEN[4]   =   !TESTEN ;

        C[0]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                     AND (C[0] XOR 1) ;
        C[1]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                     AND (C[1] XOR C[0]) ;
        C[2]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                     AND (C[2] XOR (C[1] AND C[0])) ;
        C[3]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
```

```
            AND (C[3] XOR (C[2] AND C[1] AND C[0])) ;
C[4]    :=  RESETN AND CYCENDN AND (!RDN OR !WRN)
            AND (C[4] XOR (C[3] AND C[2] AND C[1] AND C[0])) ;


{ PURPOSE: ENSTARTN OUTPUT PROVIDES THE TIMING FOR THE LEADING
       EDGE OF OEN AND WEN STROBES SO THAT 1. THE ADDRESS LINES HAVE
       TIME TO BE DECODED AND 2. OE/DATA PINS HAVE TIME TO Z-STATE
       FROM READS ON THE PRECEDING CYCLE.  THE CYCENDN TERM IS
       NEEDED TO HOLD OFF A CONSECUTIVE MEMORY CYCLE, E.G., WHEN
       WRITE DEASSERTS AND REASSERTS WITHIN THE SAME CLOCK.
       ENSTARTN SHOULD NOT BE USED TO END WRITE TRANSCEIVER
       ENABLES AS IT DEASSERTS WITH THE WRITE LINE INSTEAD OF
       HOLDING FOR ONE MORE 1/2 CLOCK.                    }

ENSTARTNEN    = !TESTEN ;
ENSTARTN  NOT := !MEMSPARE0 AND RESETN AND (C >= 1) AND CYCENDN ;


{ PURPOSE: CYCLE END GOES LOW (SYNCHRONOUSLY) DURING THE LAST RDCENN ON
       READS AND DURING ACKN ON WRITES.  IT RETURNS HIGH
       SYNCHRONOUSLY BY INTERLOCKING ON THE COUNTER OUTPUTS
       WHICH COUNT ONE GREATER THAN THE ASKED FOR VALUE BEFORE
       RESETTING BACK TO ZERO (VIA CYCENDN).  THUS CYCENDN WILL
       DEASSERT ON THE SAME CLOCK AS THE RDN, WRN, OR BURSTN RISING
       EDGES REGARDLESS OF WHETHER OR NOT THOSE RISING EDGES MEET
       THE REGISTER'S SETUP AND HOLD TIMES.                }
{ NOTE:   TO FIT CYCENDN INTO A 16V8, TWO OUTPUTS MAY BE NEEDED.    }

CYCENDNEN    = !TESTEN ;
CYCENDN NOT := RESETN AND CYCENDN AND (
            (!RAMCSN   AND (C == 02H) AND !RDN AND  BURSTN)
          OR (!RAMCSN   AND (C == 08H) AND !RDN AND !BURSTN)
          OR (!RAMCSN   AND (C == 03H) AND !WRN          )
          OR (!EPROMCSN AND (C == 03H) AND !RDN AND  BURSTN)
          OR (!EPROMCSN AND (C == 0CH) AND !RDN AND !BURSTN)
          OR (!UARTCSN  AND (C == 06H) AND          BURSTN)
          OR (!TIMERCSN AND (C == 06H) AND          BURSTN)
          OR ( {!BUSERRORN} (C == 1FH)                )
);


{ NOTE: IN THIS EXPERIMENT MEMSPARE0 IS PULLED LOW AND CAN BE
      USED TO DISABLE THIS CONTROLLER'S RDCENN, ACKN, AND BUSERRORN.
      SINCE MEMSPARE0 IS ATTACHED TO THE MEMDEC.LPLC PAL, THE
      MEMDEC PAL COULD COMBINE THE CSN'S SO THAT THESE SIGNALS
      ARE ONLY DRIVEN WHEN NEEDED.                        }

{ NOTE: ANOTHER POSSIBILITY IS TO USE MEMSPARE0 AS AN EXTRA CHIP
      SELECT.                                    }


{ PURPOSE: READ BUFFER CLOCK ENABLE IS USED BY THE R305X TO STROBE
       DATA INTO ITS INTERNAL READ BUFFERS.                }
{ NOTE:   IT IS ASSUMED THAT THE UART AND TIMER ARE
       IN UNCACHABLE MEMORY SPACE AND WILL NOT BE BURST READ.
       IF THEY ARE BURST READ, THE STATE MACHINE LOOPS 4 TIMES.    }

RDCENNEN    = !MEMSPARE0 ;
```

```
RDCENN NOT  := RESETN AND CYCENDN AND (
          (!RAMCSN AND !RDN
            AND (          (C == 02H)
                 OR (!BURSTN AND (C == 04H))
                 OR (!BURSTN AND (C == 06H))
                 OR (!BURSTN AND (C == 08H))
                )
          )
        OR (!EPROMCSN AND !RDN
            AND (          (C == 03H)
                 OR (!BURSTN AND (C == 06H))
                 OR (!BURSTN AND (C == 09H))
                 OR (!BURSTN AND (C == 0CH))
                )
          )
        OR (!UARTCSN AND !RDN
            AND (          (C == 06H)
                )
          )
        OR (!TIMERCSN AND !RDN
            AND (          (C == 06H)
                )
          )
);


{ PURPOSE: ACKNOWLEDGE IS PRIMARILY USED TO END WRITE CYCLES.  IT
        SHOULD BE PULSED ONE (HALF) CLOCK CYCLE BEFORE THE WRITE
        STROBE IS NEEDED.  ON READ CYCLES, ACKNOWLEDGE WILL
        IMPLICITLY BE GENERATED BY THE R305X, HOWEVER, IF OPTIMAL
        TIMING IS DESIRED, ACK SHOULD BE DRIVEN NO SOONER THAN 1
        CLOCK BEFORE THE END OF A SINGLE READ AND FOR BURSTS NO
        SOONER THAN 4 CLOCKS BEFORE THE END OF THE LAST READ.       }

ACKNEN      = !MEMSPARE0 ;
ACKN NOT    := RESETN AND CYCENDN AND (
            (!RAMCSN AND !WRN                    { WRITE CYCLE }
              AND (          (C == 03H)
                  )
            )
          OR (!RAMCSN AND !RDN AND !BURSTN          { READ  CYCLE }
              AND (          (C == 05H)
                  )
            )
          OR (!EPROMCSN AND !RDN AND !BURSTN         { READ  CYCLE }
              AND (          (C == 09H)
                  )
            )
          OR (!UARTCSN AND !WRN AND BURSTN          { WRITE CYCLE }
              AND (          (C == 06H)
                  )
            )
          OR (!TIMERCSN AND !WRN                   { WRITE CYCLE }
              AND (          (C == 06H)
                  )
            )
);
```

```
    { PURPOSE: BUSERRORN SIMPLY ENDS A WAYWARD UNDECODED BUS CYCLE.  ON
            READS IT CAUSES AN EXCEPTION.  ON WRITES IT DOES NOT CAUSE
            AN EXCEPTION CONDITION FOR THE PROCESSOR.  TO DO THAT, LATCH
            BUSERRORN AND FEED IT TO AN INTERRUPT PIN OR A BRANCH
            CONDITION PIN.                                  }

    BUSERRORNEN    = !MEMSPARE0 ;
    BUSERRORN NOT := RESETN AND CYCENDN AND (
                                (C == 1FH)

    ) ;


END ;
END UPAL2.
```

```
{ TITLE   : MEMEN.LPLC
          UPAL3 MEMORY READ AND WRITE ENABLE PAL FOR THE R305X BEHAVIORAL BUS
          EMULATOR MEMORY EVALUATION BOARD
  PURPOSE : GENERATES READ AND WRITE ENABLES FOR MEMORY CONTROLS.
  LANG    : LPLC — TM OF CAPILANO COMPUTING SYSTEMS
  AUTHOR  : ANDY NG, IDT INC.
  UPDATES : C7C4F 03-18-91 AP NOTE FIRST RELEASE
}

MODULE UPAL3     ;
TITLE  UPAL3     ;
TYPE   AMD 22V10 ;

INPUTS ;
      {  DEMULTIPLEXED MEMORY ADDRESS LINES }
      SYSCLK         NODE[PIN1]  ;         { INVERTED SYSCLKN            }
      POWRESETN      NODE[PIN2]  ;         { POWER UP RESET              }
      RDN            NODE[PIN3]  ;         { READ LINE                   }
      WRN            NODE[PIN4]  ;         { WRITE LINE                  }
      ENSTARTN       NODE[PIN5]  ;         { ENABLE START                }
      CYCENDN        NODE[PIN6]  ;         { CYCLE END                   }
      BEN0           NODE[PIN7]  ;         { BYTE ENABLE 0               }
      BEN1           NODE[PIN8]  ;         { BYTE ENABLE 1               }
      BEN2           NODE[PIN9]  ;         { BYTE ENABLE 2               }
      BEN3           NODE[PIN10] ;         { BYTE ENABLE 3               }
      UARTCSN        NODE[PIN11] ;         { UART CHIP SELECT            }
      MEMSPARE2      NODE[PIN13] ;         { SPARE INPUT                 }

      { OUTPUT FEEDBACK NODES (NEEDED FOR LPLC'ISM) }
      RESETN         NODE[PIN23] ;
      WRENN          NODE[PIN18] ;
      WRDATAEN       NODE[PIN17] ;

OUTPUTS ; { ATTRIBUTES C — COMBINATIONAL, R — REGISTERED, H — HIGH, L — LOW }

      { WRITE ENABLES }
      WRENNA         NODE[PIN22] ATTR[RL] ;  { WRITE ENABLE FOR BYTE 0      }
      WRENNB         NODE[PIN21] ATTR[RL] ;  { WRITE ENABLE FOR BYTE 1      }
      WRENNC         NODE[PIN20] ATTR[RL] ;  { WRITE ENABLE FOR BYTE 2      }
      WRENND         NODE[PIN19] ATTR[RL] ;  { WRITE ENABLE FOR BYTE 3      }
      WRENN          NODE[PIN18] ATTR[RL] ;  { WRITE ENABLE MOTO-TYPE I/O   }
      WRDATAEN       NODE[PIN17] ATTR[RL] ;  { WRITE DATA XCEIVER ENABLE    }

      { READ ENABLES }
      RDENN          NODE[PIN16] ATTR[RL] ;  { READ OUTPUT ENABLE (FOR WORDS)}
      RDDATAENN      NODE[PIN15] ATTR[RL] ;  { READ DATA XCEIVER ENABLE     }

      { MISCELLANEOUS CONTROLS }
      RESETN         NODE[PIN23] ATTR[RL] ;  { SYNCHRONIZED RESET          }
      GUARTCSN       NODE[PIN14] ATTR[RL] ;  { GATED/GUARDED UART CHIP SELECT}

      { I/O PINS USED AS INPUTS }
      { NONE }

      { OUTPUT ENABLES }
      WRENNAEN       NODE[PIN22EN] ;
      WRENNBEN       NODE[PIN21EN] ;
      WRENNCEN       NODE[PIN20EN] ;
```

```
        WRENNDEN        NODE[PIN19EN] ;
        WRENNEN         NODE[PIN18EN] ;
        WRDATAENEN      NODE[PIN17EN] ;
        RDENNEN         NODE[PIN16EN] ;
        RDDATAENNEN     NODE[PIN15EN] ;
        RESETNEN        NODE[PIN23EN] ;
        GUARTCSNEN      NODE[PIN14EN] ;

        { ASYNCHRONOUS RESET AND SYNCHRONOUS PRESET NODES }
        RESETEN         NODE[RESET]   ;
        PRESETEN        NODE[PRESET]  ;

TABLE ;

        { RESET AND PRESET ARE NOT USED IN THIS PAL. }
        RESETEN  = 0 ;
        PRESETEN = 0 ;


        { PURPOSE: WRITE BYTE ENABLES AND WRITE WORD ENABLE ALLOW
                SUFFICIENT TIME FOR THE ADDRESS TO DECODE AND
                FOR A VALID CHIP SELECT BEFORE ENABLING THE
                WRITE STROBE FOR A SPECIFIC BYTE BANK.
          NOTE:    BANK A IS THE BIG ENDIAN'S LSB BYTE3 OR THE LITTLE
                ENDIAN'S LSB BYTE0.  IT ALWAYS HOLDS D(7:0).
                BANK D IS THE BIG ENDIAN'S MSB BYTE0 OR THE BIG
                ENDIAN'S MSB BYTE3.  IT ALWAYS HOLDS D(31:23).
                                                }

        WRENNAEN        = !MEMSPARE2 ;
        WRENNA      NOT := RESETN AND (
                        !WRN AND !BEN0 AND !ENSTARTN AND CYCENDN
        );

        WRENNBEN        = !MEMSPARE2 ;
        WRENNB      NOT := RESETN AND (
                        !WRN AND !BEN1 AND !ENSTARTN AND CYCENDN
        );

        WRENNCEN        = !MEMSPARE2 ;
        WRENNC      NOT := RESETN AND (
                        !WRN AND !BEN2 AND !ENSTARTN AND CYCENDN
        );

        WRENNDEN        = !MEMSPARE2 ;
        WRENND      NOT := RESETN AND (
                        !WRN AND !BEN3 AND !ENSTARTN AND CYCENDN
        );

        { PURPOSE: WRENN IS USED TO PROVIDE A WRITE LINE THAT HOLDS
                LOW FOR AN EXTRA CYCLE, SO THAT IT CAN BE USED FOR
                MOTOROLA-TYPE I/O DEVICES ON THEIR MULTIPLEXED
                READ/WRITE LINE.                        }

        WRENNEN         = !MEMSPARE2 ;
        WRENN       NOT := RESETN AND (
                         (!WRN AND CYCENDN)
                       OR (!WRENN AND !CYCENDN)
```

```
);

{ PURPOSE: WRDATAEN AND RDDATAENN DRIVE THE OUTPUT ENABLE
        CONTROLS ON A FCT623T TRANSCEIVER BANK FOR THE
        DATA BUS.  THE CONTROLS CAN BE USED FOR ANY
        DUAL-OUTPUT ENABLE TRANSCEIVER (1 FOR EACH
        DIRECTION.  OUTPUT ENABLE/DIRECTION CONTROLLED
        TRANSCEIVERS (FCT245) REQUIRE MORE INTERFACING
        IF OUTPUT CONTENTION IS TO BE AVOIDED BY
        ONLY CHANGING THE DIRECTION WHEN THE OUTPUTS ARE
        DISABLED.                                   }

{ NOTE:    WRITE DATA ENABLE DEASSERTS ONE CLOCK AFTER
        WRN DOES TO PROVIDE SUFFICIENT HOLD TIME FOR THE
        WRITE DATA INTO THE MEMORY (SEE UPAL2 QWRN FOR A
        MORE DETAILED EXPLANATION).
  NOTE:    WRDATAEN IS ACTIVE HIGH FOR THE FCT623T OUTPUT ENABLE
        CONTROL.  FOR THE FCT861 OUTPUT ENABLES, USE ACTIVE
        LOW.
  NOTES:   THE FIRST OR-TERM ASSERTS WRDATAEN WHILE THE SECOND
        OR-TERM DEASSERTS WRDATAEN.                 }

WRDATAENEN      = !MEMSPARE2 ;
WRDATAEN        := RESETN AND (
                  (!WRN AND !ENSTARTN)
                OR (WRDATAEN AND (!ENSTARTN OR !CYCENDN))
);

RDENNEN         = !MEMSPARE2 ;
RDENN       NOT := RESETN AND (
                  !RDN AND !ENSTARTN AND CYCENDN
);

{ PURPOSE: RDDATAENN IS CONNECTED TO THE MEMORY BOARD'S
        DATA TRANSCEIVER OUTPUT ENABLE (FCT623T OR FCT861)
        AND ONLY ENABLES FOR THIS BOARD'S CHIP SELECTS.
        IF THE MEMORY CONTROLLER IS USED FOR ANOTHER
        BOARD'S MEMORY, THEN THE TRANSCEIVER OUTPUT ENABLE
        SHOULD BE DISABLED FOR THOSE CHIP SELECTS (VIA
        MEMSPARE2.                                  }

{ NOTE:    IN MOST SYSTEMS, R305X'S DATAENN OUTPUT CAN BE
        CONNECTED DIRECTLY TO THE TRANSCEIVER ENABLE PIN
        INSTEAD OF USING A SYNTHESIZED RDDATAENN.          }

RDDATAENNEN     = !MEMSPARE2 ;
RDDATAENN   NOT := RESETN AND (
                  !RDN AND !ENSTARTN AND CYCENDN
);


{ PURPOSE: RESET SYNCHRONIZES THE POWER UP RESET FOR THE
        MEMORY CONTROLLER STATE MACHINES AND FOR THE R305X.  }

RESETNEN        = !MEMSPARE2 ;
RESETN      NOT := !POWRESETN ;


{ PURPOSE: GUARDED/GATED UART CHIP SELECT, GUARTCSN GATES
```

```
            UARTCSN BECAUSE THE UART BEING USED HAS A MOTOROLA-
            TYPE I/O DEVICE INTERFACE WHICH MULTIPLEXES ITS
            READ/WRITE INPUT PIN SUCH THAT THE CHIP SELECT MUST
            STROBE IN OR OUT DATA.  THIS IS IN CONTRAST TO AN
            INTEL-TYPE I/O DEVICE INTERFACE WHICH WOULD HAVE A
            SEPARATE READ STROBE AND WRITE STROBE AS WELL AS A
            CHIP SELECT.  IT IS IMPORTANT NOT TO HAVE A
            GLITCH (FROM ADDRESS DECODING THE CHIP SELECT) ON
            READS IN ORDER TO ALLOW THE I/O DEVICE TO UPDATE
            FIFO POINTERS, ETC.  THUS GUARTCSN STARTS LATE AND
            ENDS EARLY, SO THAT READ/WRITE IS HELD VALID
            THROUGHOUT THE CHIP SELECT.                    }

    GUARTCSNEN      = !MEMSPARE2 ;
    GUARTCSN    NOT := RESETN AND (
                    !UARTCSN AND !ENSTARTN AND CYCENDN
    );

END;
END UPAL3.
```

```
{ TITLE  : MEMINT.LPLC
         UPAL4 MEMORY I/O INTERRUPT CONTROLLER PAL FOR THE R305X BEHAVIORAL
         BUS EMULATOR MEMORY EVALUATION BOARD
  PURPOSE: REPLICATES THE TIMER/UART INTERRUPT CONTROLLER ON THE 7RS382 BOARD.
         ADDITIONAL FUSE BITS ADDED FOR 16V8 COMPATIBILITY.
  LANG   : LPLC — TM OF CAPILANO COMPUTING SYSTEMS
  AUTHOR : IDT INC.
  UPDATES: C3F98 01-04-91 16V8 PCB VERSION FIRST RELEASE A.N.
}


{ U24A_382 INTERRUPT PAL}
{ 1-2-90,12-14-89 }
{JEDEC file's CHECKSUM = 379E } { NOTE: 01-04-91 — NOT APPLICABLE TO 16V8 }

{    CONTROL PAL FOR 8254 TIMER'S AND UART INTERRUPT
     USED FOR EVALUATION BOARD 382 }

MODULE U24A_382;
TITLE U24A_382;
TYPE MMI 16R8;

{ FUSE BITS FOR 16V8 FAMILY ATTRIBUTES USED AS A 16R8 }
FUSE 2048..2079 00000000000000000000000000000000 ;
FUSE 2080..2111 00000000000000000000000000000000 ;
FUSE 2112..2143 00000000000000001111111111111111 ;
FUSE 2144..2175 11111111111111111111111111111111 ;
FUSE 2176..2193 111111111111111101              ;

INPUTS;

MRES/       NODE[PIN2];
UARTINT/    NODE[PIN3];
PMRD/       NODE[PIN4];
CSTIM/      NODE[PIN5];
EA02        NODE[PIN6];
EA04        NODE[PIN7];
OUT1        NODE[PIN8]; {input from Timer output OUT1}
OUT0        NODE[PIN9]; {input from Timer output OUT0}

DT0A/       NODE[PIN14]; {feedback}
DT0B/       NODE[PIN15]; {feedback}
T0INT/      NODE[PIN16]; {feedback}

DT1A/       NODE[PIN17]; {feedback}
DT1B/       NODE[PIN18]; {feedback}
T1INT/      NODE[PIN19]; {feedback}

OUTPUTS;

UINT5/      NODE[PIN13];
DT0A/       NODE[PIN14];
DT0B/       NODE[PIN15];
T0INT/      NODE[PIN16]; { goes to R3000's UINT3}

DT1A/       NODE[PIN17];
DT1B/       NODE[PIN18];
T1INT/      NODE[PIN19]; { goes to R3000's UINT4}
```

```
TABLE;
{   8254 TIMER generates 2 square-wave outputs OUT0 and OUT1.
    When OUT0 goes from high to low, this PAL asserts interrupt
    TOINT/, which will interrupt R3000 through UINT3.
    Same scheme applies to OUT1, T1INT/ and UINT4.
    Reading physical addresses 1F80 0010 and 1F80 0014 (which are
    virtual addresses BF80 0010 and BF80 0014 in this 382 board)
    will clear interrupt UINT3 and UINT4, respectively.

    This PAL also synchronizes UART interrupt signal }


DT0A/     :=    OUT0;      {delay TIMER's OUT0 through a register}
DT0B/     :=    DT0A/;     {delay again}
T0INT/ NOT :=   MRES/ AND
              ((NOT DT0A/ AND DT0B/) OR
              (NOT T0INT/ AND (NOT EA04 OR EA02 OR CSTIM/ OR PMRD/)));

DT1A/     :=    OUT1;
DT1B/     :=    DT1A/;
T1INT/ NOT :=   MRES/ AND
              ((NOT DT1A/ AND DT1B/) OR
              (NOT T1INT/ AND (NOT EA04 OR NOT EA02 OR CSTIM/ OR PMRD/)));

UINT5/    :=    UARTINT/ OR NOT MRES/ ;
              {put UART's interrupt through a register to synchronize
              it with R3000 clock }
END;
END U24A_382.
```

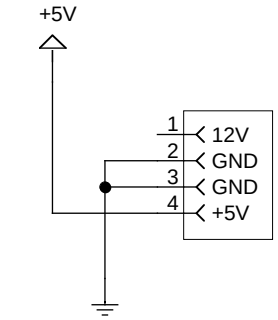


Figure 15. Power Connector

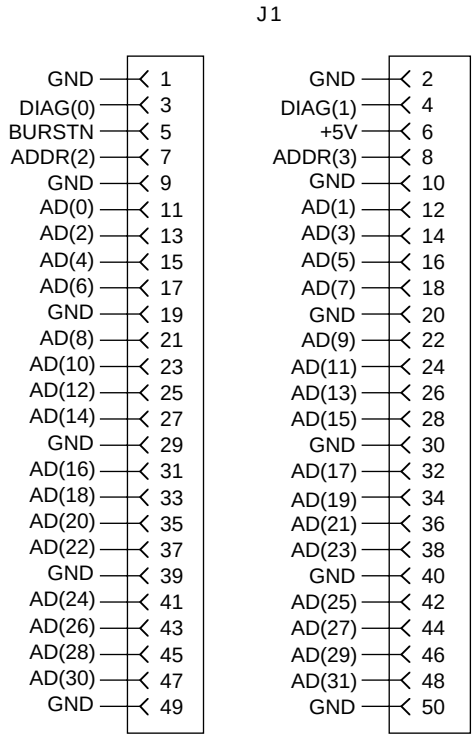


Figure 16. 50-Pin Connector

Figure 17. Spares

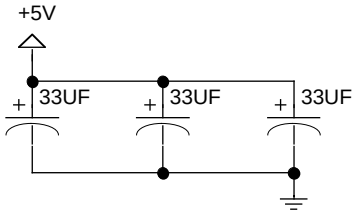


Figure 18. Primary Power Decoupling Capacitors

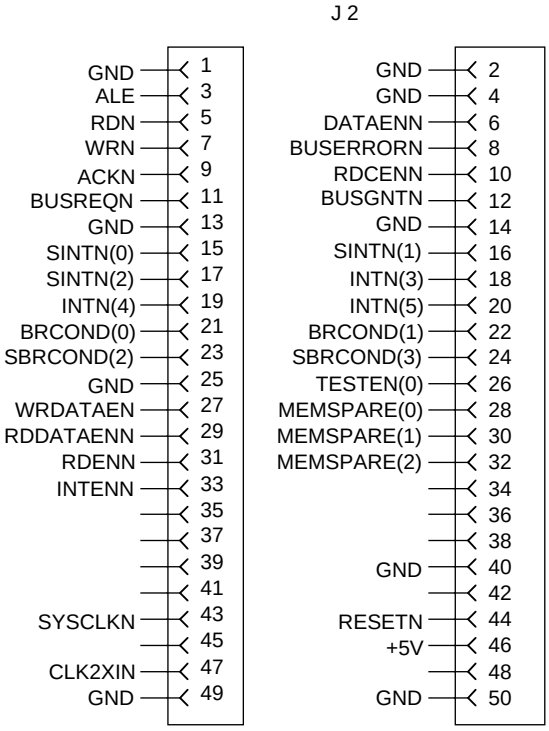


Figure 19. 50-Pin Connector

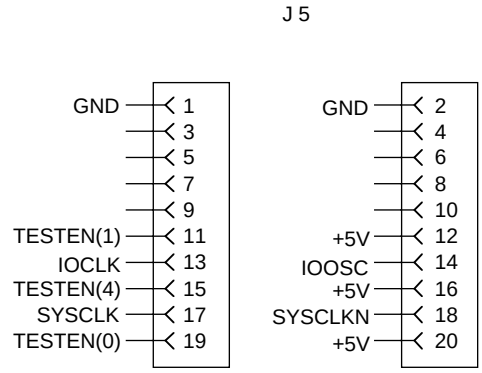


Figure 20. 100 Mil Jumper Headers

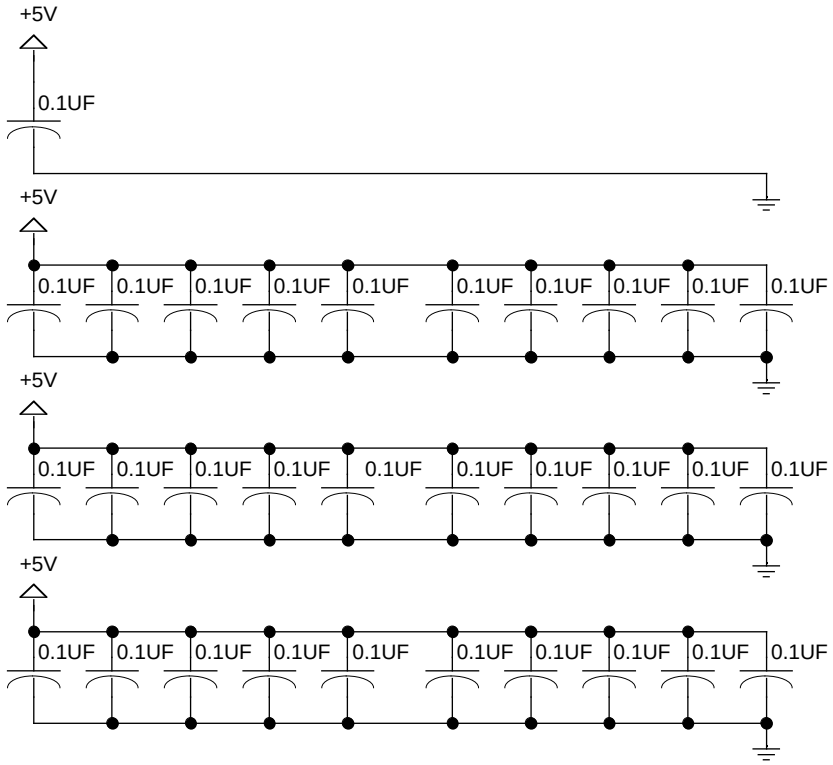


Figure 21. Decoupling Capacitors

```
{ TITLE   : MEMDEC.LPLC
            UPAL1 MEMORY AND I/O ADDRESS DECODER PAL FOR THE R305X
            BEHAVIORAL BUS EMULATOR MEMORY EVALUATION BOARD
  PURPOSE : DECODES DEMULTIPLEXED ADDRESS TO GENERATE CHIP SELECTS.
  LANG    : LPLC – TM OF CAPILANO COMPUTING SYSTEMS
  AUTHOR  : ANDY NG, IDT INC.
  UPDATES : C2503 03-18-91 AP NOTE FIRST RELEASE
}

MODULE UPAL1     ;
TITLE  UPAL1     ;
TYPE   AMD 22V10 ;

INPUTS ;
        {  DEMULTIPLEXED MEMORY ADDRESS LINES }
        A17             NODE[PIN1]  ;           { MSB ADDRESS LINES 31-17       }
        A18             NODE[PIN2]  ;
        A19             NODE[PIN3]  ;
        A20             NODE[PIN4]  ;
        A21             NODE[PIN5]  ;
        A22             NODE[PIN6]  ;
        A23             NODE[PIN7]  ;
        A24             NODE[PIN8]  ;
        A25             NODE[PIN9]  ;
        A26             NODE[PIN10] ;
        A27             NODE[PIN11] ;
        A28             NODE[PIN13] ;


        { OUTPUT FEEDBACK NODES (NEEDED FOR LPLC'ISM) }
        A29             NODE[PIN16] ;
        A30             NODE[PIN15] ;
        A31             NODE[PIN14] ;
        MEMSPARE0       NODE[PIN19] ;
        MEMSPARE1       NODE[PIN18] ;
        MEMSPARE2       NODE[PIN17] ;

OUTPUTS ; { ATTRIBUTES C – COMBINATIONAL, R – REGISTERED, H – HIGH, L – LOW }

        { CHIP SELECTS }
        RAMCSN          NODE[PIN23] ATTR[CL] ;  { STATIC RAM CHIP SELECT        }
        EPROMCSN        NODE[PIN22] ATTR[CL] ;  { EPROM CHIP SELECT             }
        UARTCSN         NODE[PIN21] ATTR[CL] ;  { UNGATED UART CHIP SELECT      }
        TIMERCSN        NODE[PIN20] ATTR[CL] ;  { TIMER CHIP SELECT             }

        { I/O PINS USED AS INPUTS }
        A29             NODE[PIN14] ATTR[CL] ;  { MSB ADDRESS LINES 31-17       }
        A30             NODE[PIN15] ATTR[CL] ;
        A31             NODE[PIN16] ATTR[CL] ;
        MEMSPARE0       NODE[PIN19] ATTR[CL] ;
        MEMSPARE1       NODE[PIN18] ATTR[CL] ;
        MEMSPARE2       NODE[PIN17] ATTR[CL] ;

        { OUTPUT ENABLES }
        RAMCSNEN        NODE[PIN23EN] ;
        EPROMCSNEN      NODE[PIN22EN] ;
        UARTCSNEN       NODE[PIN21EN] ;
```

```
        TIMERCSNEN      NODE[PIN20EN] ;
        A29EN           NODE[PIN14EN] ;
        A30EN           NODE[PIN15EN] ;
        A31EN           NODE[PIN16EN] ;
        MEMSPARE0EN     NODE[PIN19EN] ;
        MEMSPARE1EN     NODE[PIN18EN] ;
        MEMSPARE2EN     NODE[PIN17EN] ;

        { ASYNCHRONOUS RESET AND SYNCHRONOUS PRESET NODES }
        RESETEN         NODE[RESET]   ;
        PRESETEN        NODE[PRESET]  ;


        { 7RS382 COMPATIBLE PHYSICAL ADDRESS DECODE MAP }
        {    RAM       00000000H – 0001FFFFH    32K    }
        {    EPROM     1FC00000H – 1FC1FFFFH    32K    }
        {    UART      1FE00000H – 1FE0003FH           }
        {    TIMER     1F800000H – 1F80002CH           }


TERMS ; { LPLC “TABLE” ALGORITHM TAKES TOO LONG TO COMPILE }

        { NOTES: MEMSPARE0 IS BEING USED FOR A BOARD CHIP SELECT
                 DRIVABLE BY ANOTHER MEMORY SYSTEM.  WITHOUT IT
                 ASSERTED LOW, THIS BOARD WILL NOT ISSUE ANY MEMORY
                 SIGNALS NOR OUTPUT ENABLE SHARED CONTROL PINS.         }
        { NOTES: MEMSPARE1 IS NOT BEING USED.  IT COULD BE USED AS AN
                 OUTPUT IF IT OR THE UPAL2 OUTPUT IT IS CONNECTED TO IS
                 TRISTATED.                                             }
        { NOTES: MEMSPARE2 IS BEING USED AS A TESTEN INPUT PIN TO
                 TRISTATE THE OUTPUTS DURING BOARD TESTING.  ANOTHER
                 USE WOULD BE FOR A BOARD CHIP SELECT – MEMCSN.
                 MEMSPARE2 IS CONNECTED TO A UPAL3 INPUT PIN.           }

        { I/O PINS USED ONLY AS INPUTS }
        A29EN            = 0 ;
        A30EN            = 0 ;
        A31EN            = 0 ;
        MEMSPARE0EN      = 0 ;
        MEMSPARE1EN      = 0 ;
        MEMSPARE2EN      = 0 ;
        A29          NOT = 0 ;
        A30          NOT = 0 ;
        A31          NOT = 0 ;
        MEMSPARE0    NOT = 0 ;
        MEMSPARE1    NOT = 0 ;
        MEMSPARE2    NOT = 0 ;

        { RESET AND PRESET ARE NOT USED IN THIS PAL. }
        RESETEN  = 0 ;
        PRESETEN = 0 ;


        RAMCSNEN         = !MEMSPARE2 ;
        RAMCSN NOT       = !MEMSPARE0 AND
                                !A31 AND !A30 AND !A29 AND !A28
                             AND !A27 AND !A26 AND !A25 AND !A24
```

```
                          AND !A23 AND !A22 AND !A21 AND !A20
                          AND !A19 AND !A18 AND !A17
        ;


        EPROMCSNEN        = !MEMSPARE2 ;
        EPROMCSN NOT      = !MEMSPARE0 AND
                               !A31 AND !A30 AND !A29 AND  A28
                          AND  A27 AND  A26 AND  A25 AND  A24
                          AND  A23 AND  A22 AND !A21 AND !A20
                          AND !A19 AND !A18 AND !A17
        ;


        UARTCSNEN         = !MEMSPARE2 ;
        UARTCSN NOT       = !MEMSPARE0 AND
                               !A31 AND !A30 AND !A29 AND  A28
                          AND  A27 AND  A26 AND  A25 AND  A24
                          AND  A23 AND  A22 AND  A21 AND !A20
                          AND !A19 AND !A18 AND !A17
        ;


        TIMERCSNEN        = !MEMSPARE2 ;
        TIMERCSN NOT      = !MEMSPARE0 AND
                               !A31 AND !A30 AND !A29 AND  A28
                          AND  A27 AND  A26 AND  A25 AND  A24
                          AND  A23 AND !A22 AND !A21 AND !A20
                          AND !A19 AND !A18 AND !A17
        ;


END;
END UPAL1.
```

```
{ TITLE  : MEMCONT.LPLC
           UPAL2 MEMORY CONTROLLER PAL FOR THE R305X BEHAVIORAL BUS EMULATOR
           MEMORY EVALUATION BOARD
  PURPOSE: PRODUCES READ, WRITE, AND BUS ERROR ACKNOWLEDGE CONTROLS (RDCENN,
           ACKN, BUSERRORN) BASED ON A 4 OR 5 BIT COUNTER AND CYCLE END
           STALL CYCLE (WAIT STATE) EQUATIONS.
  LANG   : LPLC — TM OF CAPILANO COMPUTING SYSTEMS
  AUTHOR : ANDY NG, IDT INC.
  UPDATES: C4B28 03-18-91 AP NOTE FIRST RELEASE
}

MODULE UPAL2     ;
TITLE  UPAL2     ;
TYPE   AMD 22V10 ;

INPUTS ;
        { REGULAR INPUT PINS }
        SYSCLK          NODE[PIN1]  ;          { UN-INVERTED SYSTEM CLOCK      }
        RESETN          NODE[PIN2]  ;          { MASTER RESET                  }
        RDN             NODE[PIN3]  ;          { READ                          }
        WRN             NODE[PIN4]  ;          { WRITE                         }
        BURSTN          NODE[PIN5]  ;          { BURST READ | WRITE NEAR       }
        RAMCSN          NODE[PIN6]  ;          { RAM CHIP SELECT               }
        EPROMCSN        NODE[PIN7]  ;          { EPROM CHIP SELECT             }
        UARTCSN         NODE[PIN8]  ;          { UART CHIP SELECT              }
        TIMERCSN        NODE[PIN9]  ;          { TIMER CHIP SELECT             }
        MEMSPARE0       NODE[PIN10] ;          {                               }
        MEMSPARE2       NODE[PIN11] ;          {                               }
        TESTEN          NODE[PIN13] ;          { TEST PIN TO Z-STATE OUTPUTS   }

        { REGISTER FEEDBACK PINS }
        C WIDTH[5]      NODE[PIN15,PIN14,PIN21,PIN22,PIN23] ;
        ENSTARTN        NODE[PIN16] ;
        CYCENDN         NODE[PIN18] ;
        RDCENN          NODE[PIN19] ;
        ACKN            NODE[PIN20] ;
        BUSERRORN       NODE[PIN17] ;


OUTPUTS ; { ATTRIBUTES C — COMBINATIONAL, R — REGISTERED, H — HIGH, L — LOW  }

        { REGISTERED OUTPUT PINS }
        { BINARY UP COUNTER INPUTS MSB TO LSB C4, C3, C2, C1, C0 }
        C WIDTH[5]      NODE[PIN15,PIN14,PIN21,PIN22,PIN23] ATTR[RL] ;
        ENSTARTN        NODE[PIN16] ATTR[RL] ;  { READ/WRITE OUTPUT ENABLE START }
        CYCENDN         NODE[PIN18] ATTR[RL] ;  { CYCLE END (COMPOSITE ACK)      }
        RDCENN          NODE[PIN19] ATTR[RL] ;  { R305X READ BUFFER CLOCK ENABLE }
        ACKN            NODE[PIN20] ATTR[RL] ;  { R3050X ACKNOWLEDGE             }
        BUSERRORN       NODE[PIN17] ATTR[RL] ;  { R305X BUS ERROR                }

        { OUTPUT ENABLES }
        CEN WIDTH[5]    NODE[PIN15EN,PIN14EN,PIN21EN,PIN22EN,PIN23EN] ;
        ENSTARTNEN      NODE[PIN16EN] ;
        CYCENDNEN       NODE[PIN18EN] ;
        RDCENNEN        NODE[PIN19EN] ;
        ACKNEN          NODE[PIN20EN] ;
        BUSERRORNEN     NODE[PIN17EN] ;
```

```
        { ASYNCHRONOUS RESET AND SYNCHRONOUS PRESET NODES }
        RESETEN         NODE[RESET]  ;
        PRESETEN        NODE[PRESET] ;


TABLE ;

        { RESET AND PRESET ARE NOT BEING USED.                                  }
        RESETEN  = 0 ;
        PRESETEN = 0 ;

        { PURPOSE: PROVIDES REGISTERED VERSION OF RDN AND WRN.

          NOTE:    QRDN AND QWRN ARE KEPT LOW ONE EXTRA CLOCK BY CYCENDN.
                   THIS IS BECAUSE THE RISING EDGE OF RDN OR WRN MAY NOT
                   HAVE ENOUGH HOLD TIME FROM THE RISING EDGE OF
                   (BUFFERRED) SYSCLK.
          NOTE:    QRDN AND QWRN DO NOT NECESSARILY TRANSITION BACK HIGH
                   BETWEEN CONSECUTIVE MEMORY CYCLES, E.G., WRITE FOLLOWED
                   BY A WRITE.                                                  }

        { QRDN NOT      :=   RESETN AND (!RDN OR (!QRDN AND !CYCENDN)) ; }
        { QWRN NOT      :=   RESETN AND (!WRN OR (!QWRN AND !CYCENDN)) ; }


        { PURPOSE: C[4]-C[0] PROVIDES A 5-BIT BINARY UP COUNTER.  IT IS RESET
                   ANYTIME RESETN IS ASSERTED AND AT THE END
                   OF EVERY MEMORY CYCLE AFTER CYCENDN IS ASSERTED.
                   IT BEGINS COUNTING UP WHEN A READ OR WRITE CYCLE IS
                   INITIATED.
          NOTE:    CYCENDN IS ASSUMED TO ASSERT WITH THE LAST RDCENN
                   ON READS AND WITH ACKN ON WRITES.  THUS CYCENDN WILL CLEAR
                   THE COUNTER WHETHER OR NOT RDN OR WRN HIGH TRANSITION
                   MEETS THE REGISTER SETUP AND HOLD TIME REQUIREMENTS.         }

        { NOTE:    TO ADD A GENERAL PURPOSE READY (A.K.A. BUSYN AND WAITN)
                   INPUT, CHANGE EACH OF THE COUNTER C[4:0] EQUATIONS SO
                   THAT THEIR VALUE CAN BE HELD WITH AN ADDITIONAL TERM, E.G.:
                   C[0]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                                  AND (    (C[0] XOR 1)
                                        OR (C[0] AND !READY) ) ;
                   A READY INPUT CAN BE USED FOR DUAL-PORT MEMORY INTERFACING,
                   EEPROM WRITE INTERFACING, ETC.
        }

        CEN[0]   =   !TESTEN ;
        CEN[1]   =   !TESTEN ;
        CEN[2]   =   !TESTEN ;
        CEN[3]   =   !TESTEN ;
        CEN[4]   =   !TESTEN ;

        C[0]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                        AND (C[0] XOR 1) ;
        C[1]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                        AND (C[1] XOR C[0]) ;
        C[2]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                        AND (C[2] XOR (C[1] AND C[0])) ;
```

```
C[3]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                AND (C[3] XOR (C[2] AND C[1] AND C[0])) ;
C[4]    :=   RESETN AND CYCENDN AND (!RDN OR !WRN)
                AND (C[4] XOR (C[3] AND C[2] AND C[1] AND C[0])) ;


{ PURPOSE: ENSTARTN OUTPUT PROVIDES THE TIMING FOR THE LEADING
           EDGE OF OEN AND WEN STROBES SO THAT 1. THE ADDRESS LINES HAVE
           TIME TO BE DECODED AND 2. OE/DATA PINS HAVE TIME TO Z-STATE
           FROM READS ON THE PRECEDING CYCLE.  THE CYCENDN TERM IS
           NEEDED TO HOLD OFF A CONSECUTIVE MEMORY CYCLE, E.G., WHEN
           WRITE DEASSERTS AND REASSERTS WITHIN THE SAME CLOCK.
           ENSTARTN SHOULD NOT BE USED TO END WRITE TRANSCEIVER
           ENABLES AS IT DEASSERTS WITH THE WRITE LINE INSTEAD OF
           HOLDING FOR ONE MORE 1/2 CLOCK.                              }

ENSTARTNEN     = !TESTEN ;
ENSTARTN  NOT := !MEMSPARE0 AND RESETN AND (C >= 1) AND CYCENDN ;


{ PURPOSE: CYCLE END GOES LOW (SYNCHRONOUSLY) DURING THE LAST RDCENN ON
           READS AND DURING ACKN ON WRITES.  IT RETURNS HIGH
           SYNCHRONOUSLY BY INTERLOCKING ON THE COUNTER OUTPUTS
           WHICH COUNT ONE GREATER THAN THE ASKED FOR VALUE BEFORE
           RESETTING BACK TO ZERO (VIA CYCENDN).  THUS CYCENDN WILL
           DEASSERT ON THE SAME CLOCK AS THE RDN, WRN, OR BURSTN RISING
           EDGES REGARDLESS OF WHETHER OR NOT THOSE RISING EDGES MEET
           THE REGISTER'S SETUP AND HOLD TIMES.                         }
{ NOTE:    TO FIT CYCENDN INTO A 16V8, TWO OUTPUTS MAY BE NEEDED.       }

CYCENDNEN    = !TESTEN ;
CYCENDN NOT := RESETN AND CYCENDN AND (
                  (!RAMCSN   AND (C == 02H) AND !RDN AND  BURSTN)
               OR (!RAMCSN   AND (C == 08H) AND !RDN AND !BURSTN)
               OR (!RAMCSN   AND (C == 03H) AND !WRN            )
               OR (!EPROMCSN AND (C == 03H) AND !RDN AND  BURSTN)
               OR (!EPROMCSN AND (C == 0CH) AND !RDN AND !BURSTN)
               OR (!UARTCSN  AND (C == 06H) AND           BURSTN)
               OR (!TIMERCSN AND (C == 06H) AND           BURSTN)
               OR ( {!BUSERRORN} (C == 1FH)                     )
);


{ NOTE: IN THIS EXPERIMENT MEMSPARE0 IS PULLED LOW AND CAN BE
        USED TO DISABLE THIS CONTROLLER'S RDCENN, ACKN, AND BUSERRORN.
        SINCE MEMSPARE0 IS ATTACHED TO THE MEMDEC.LPLC PAL, THE
        MEMDEC PAL COULD COMBINE THE CSN'S SO THAT THESE SIGNALS
        ARE ONLY DRIVEN WHEN NEEDED.                                    }

{ NOTE: ANOTHER POSSIBILITY IS TO USE MEMSPARE0 AS AN EXTRA CHIP
        SELECT.                                                         }


{ PURPOSE: READ BUFFER CLOCK ENABLE IS USED BY THE R305X TO STROBE
           DATA INTO ITS INTERNAL READ BUFFERS.                         }
{ NOTE:    IT IS ASSUMED THAT THE UART AND TIMER ARE
           IN UNCACHABLE MEMORY SPACE AND WILL NOT BE BURST READ.
           IF THEY ARE BURST READ, THE STATE MACHINE LOOPS 4 TIMES.     }
```

```
RDCENNEN     = !MEMSPARE0 ;
RDCENN NOT  := RESETN AND CYCENDN AND (
               (!RAMCSN AND !RDN
                  AND (                 (C == 02H)
                        OR (!BURSTN AND (C == 04H))
                        OR (!BURSTN AND (C == 06H))
                        OR (!BURSTN AND (C == 08H))
                       )
               )
            OR (!EPROMCSN AND !RDN
                  AND (                 (C == 03H)
                        OR (!BURSTN AND (C == 06H))
                        OR (!BURSTN AND (C == 09H))
                        OR (!BURSTN AND (C == 0CH))
                       )
               )
            OR (!UARTCSN AND !RDN
                  AND (                 (C == 06H)
                       )
               )
            OR (!TIMERCSN AND !RDN
                  AND (                 (C == 06H)
                       )
               )
);


{ PURPOSE: ACKNOWLEDGE IS PRIMARILY USED TO END WRITE CYCLES.  IT
           SHOULD BE PULSED ONE (HALF) CLOCK CYCLE BEFORE THE WRITE
           STROBE IS NEEDED.  ON READ CYCLES, ACKNOWLEDGE WILL
           IMPLICITLY BE GENERATED BY THE R305X, HOWEVER, IF OPTIMAL
           TIMING IS DESIRED, ACK SHOULD BE DRIVEN NO SOONER THAN 1
           CLOCK BEFORE THE END OF A SINGLE READ AND FOR BURSTS NO
           SOONER THAN 4 CLOCKS BEFORE THE END OF THE LAST READ.        }

ACKNEN       = !MEMSPARE0 ;
ACKN NOT    := RESETN AND CYCENDN AND (
                (!RAMCSN AND !WRN                          { WRITE CYCLE }
                   AND (                (C == 03H)
                        )
               )
               OR (!RAMCSN AND !RDN AND !BURSTN            { READ  CYCLE }
                     AND (                (C == 05H)
                          )
               )
               OR (!EPROMCSN AND !RDN AND !BURSTN          { READ  CYCLE }
                     AND (                (C == 09H)
                          )
               )
               OR (!UARTCSN AND !WRN AND BURSTN            { WRITE CYCLE }
                     AND (                (C == 06H)
                          )
               )
```

```
                    OR (!TIMERCSN AND !WRN                          { WRITE CYCLE }
                          AND (                 (C == 06H)
                              )
                       )
        );


        { PURPOSE: BUSERRORN SIMPLY ENDS A WAYWARD UNDECODED BUS CYCLE.  ON
                   READS IT CAUSES AN EXCEPTION.  ON WRITES IT DOES NOT CAUSE
                   AN EXCEPTION CONDITION FOR THE PROCESSOR.  TO DO THAT, LATCH
                   BUSERRORN AND FEED IT TO AN INTERRUPT PIN OR A BRANCH
                   CONDITION PIN.                                           }

        BUSERRORNEN    = !MEMSPARE0 ;
        BUSERRORN NOT := RESETN AND CYCENDN AND (
                                               (C == 1FH)

        ) ;


END ;
END UPAL2.
```

```
{ TITLE   : MEMEN.LPLC
            UPAL3 MEMORY READ AND WRITE ENABLE PAL FOR THE R305X BEHAVIORAL BUS
            EMULATOR MEMORY EVALUATION BOARD
  PURPOSE : GENERATES READ AND WRITE ENABLES FOR MEMORY CONTROLS.
  LANG    : LPLC – TM OF CAPILANO COMPUTING SYSTEMS
  AUTHOR  : ANDY NG, IDT INC.
  UPDATES : C7C4F 03-18-91 AP NOTE FIRST RELEASE
}

MODULE UPAL3     ;
TITLE  UPAL3     ;
TYPE   AMD 22V10 ;

INPUTS ;
        {  DEMULTIPLEXED MEMORY ADDRESS LINES }
        SYSCLK          NODE[PIN1]  ;          { INVERTED SYSCLKN             }
        POWRESETN       NODE[PIN2]  ;          { POWER UP RESET               }
        RDN             NODE[PIN3]  ;          { READ LINE                    }
        WRN             NODE[PIN4]  ;          { WRITE LINE                   }
        ENSTARTN        NODE[PIN5]  ;          { ENABLE START                 }
        CYCENDN         NODE[PIN6]  ;          { CYCLE END                    }
        BEN0            NODE[PIN7]  ;          { BYTE ENABLE 0                }
        BEN1            NODE[PIN8]  ;          { BYTE ENABLE 1                }
        BEN2            NODE[PIN9]  ;          { BYTE ENABLE 2                }
        BEN3            NODE[PIN10] ;          { BYTE ENABLE 3                }
        UARTCSN         NODE[PIN11] ;          { UART CHIP SELECT             }
        MEMSPARE2       NODE[PIN13] ;          { SPARE INPUT                  }

        { OUTPUT FEEDBACK NODES (NEEDED FOR LPLC'ISM) }
        RESETN          NODE[PIN23] ;
        WRENN           NODE[PIN18] ;
        WRDATAEN        NODE[PIN17] ;

OUTPUTS ; { ATTRIBUTES C – COMBINATIONAL, R – REGISTERED, H – HIGH, L – LOW }

        { WRITE ENABLES }
        WRENNA          NODE[PIN22] ATTR[RL] ;  { WRITE ENABLE FOR BYTE 0      }
        WRENNB          NODE[PIN21] ATTR[RL] ;  { WRITE ENABLE FOR BYTE 1      }
        WRENNC          NODE[PIN20] ATTR[RL] ;  { WRITE ENABLE FOR BYTE 2      }
        WRENND          NODE[PIN19] ATTR[RL] ;  { WRITE ENABLE FOR BYTE 3      }
        WRENN           NODE[PIN18] ATTR[RL] ;  { WRITE ENABLE MOTO-TYPE I/O   }
        WRDATAEN        NODE[PIN17] ATTR[RL] ;  { WRITE DATA XCEIVER ENABLE    }

        { READ ENABLES }
        RDENN           NODE[PIN16] ATTR[RL] ;  { READ OUTPUT ENABLE (FOR WORDS)}
        RDDATAENN       NODE[PIN15] ATTR[RL] ;  { READ DATA XCEIVER ENABLE     }

        { MISCELLANEOUS CONTROLS }
        RESETN          NODE[PIN23] ATTR[RL] ;  { SYNCHRONIZED RESET           }
        GUARTCSN        NODE[PIN14] ATTR[RL] ;  { GATED/GUARDED UART CHIP SELECT}

        { I/O PINS USED AS INPUTS }
        { NONE }

        { OUTPUT ENABLES }
        WRENNAEN        NODE[PIN22EN] ;
        WRENNBEN        NODE[PIN21EN] ;
```

```
        WRENNCEN        NODE[PIN20EN] ;
        WRENNDEN        NODE[PIN19EN] ;
        WRENNEN         NODE[PIN18EN] ;
        WRDATAENEN      NODE[PIN17EN] ;
        RDENNEN         NODE[PIN16EN] ;
        RDDATAENNEN     NODE[PIN15EN] ;
        RESETNEN        NODE[PIN23EN] ;
        GUARTCSNEN      NODE[PIN14EN] ;

        { ASYNCHRONOUS RESET AND SYNCHRONOUS PRESET NODES }
        RESETEN         NODE[RESET]   ;
        PRESETEN        NODE[PRESET]  ;

TABLE ;

        { RESET AND PRESET ARE NOT USED IN THIS PAL. }
        RESETEN  = 0 ;
        PRESETEN = 0 ;


        { PURPOSE: WRITE BYTE ENABLES AND WRITE WORD ENABLE ALLOW
                   SUFFICIENT TIME FOR THE ADDRESS TO DECODE AND
                   FOR A VALID CHIP SELECT BEFORE ENABLING THE
                   WRITE STROBE FOR A SPECIFIC BYTE BANK.
          NOTE:    BANK A IS THE BIG ENDIAN'S LSB BYTE3 OR THE LITTLE
                   ENDIAN'S LSB BYTE0.  IT ALWAYS HOLDS D(7:0).
                   BANK D IS THE BIG ENDIAN'S MSB BYTE0 OR THE BIG
                   ENDIAN'S MSB BYTE3.  IT ALWAYS HOLDS D(31:23).
                                                                    }

        WRENNAEN         = !MEMSPARE2 ;
        WRENNA      NOT := RESETN AND (
                           !WRN AND !BEN0 AND !ENSTARTN AND CYCENDN
        );

        WRENNBEN         = !MEMSPARE2 ;
        WRENNB      NOT := RESETN AND (
                           !WRN AND !BEN1 AND !ENSTARTN AND CYCENDN
        );

        WRENNCEN         = !MEMSPARE2 ;
        WRENNC      NOT := RESETN AND (
                           !WRN AND !BEN2 AND !ENSTARTN AND CYCENDN
        );

        WRENNDEN         = !MEMSPARE2 ;
        WRENND      NOT := RESETN AND (
                           !WRN AND !BEN3 AND !ENSTARTN AND CYCENDN
        );

        { PURPOSE: WRENN IS USED TO PROVIDE A WRITE LINE THAT HOLDS
                   LOW FOR AN EXTRA CYCLE, SO THAT IT CAN BE USED FOR
                   MOTOROLA-TYPE I/O DEVICES ON THEIR MULTIPLEXED
                   READ/WRITE LINE.                                 }

        WRENNEN          = !MEMSPARE2 ;
        WRENN       NOT := RESETN AND (
```

```
                    (!WRN AND CYCENDN)
                  OR (!WRENN AND !CYCENDN)
);

{ PURPOSE: WRDATAEN AND RDDATAENN DRIVE THE OUTPUT ENABLE
           CONTROLS ON A FCT623T TRANSCEIVER BANK FOR THE
           DATA BUS.  THE CONTROLS CAN BE USED FOR ANY
           DUAL-OUTPUT ENABLE TRANSCEIVER (1 FOR EACH
           DIRECTION.  OUTPUT ENABLE/DIRECTION CONTROLLED
           TRANSCEIVERS (FCT245) REQUIRE MORE INTERFACING
           IF OUTPUT CONTENTION IS TO BE AVOIDED BY
           ONLY CHANGING THE DIRECTION WHEN THE OUTPUTS ARE
           DISABLED.                                          }

{ NOTE:    WRITE DATA ENABLE DEASSERTS ONE CLOCK AFTER
           WRN DOES TO PROVIDE SUFFICIENT HOLD TIME FOR THE
           WRITE DATA INTO THE MEMORY (SEE UPAL2 QWRN FOR A
           MORE DETAILED EXPLANATION).
  NOTE:    WRDATAEN IS ACTIVE HIGH FOR THE FCT623T OUTPUT ENABLE
           CONTROL.  FOR THE FCT861 OUTPUT ENABLES, USE ACTIVE
           LOW.
  NOTES:   THE FIRST OR-TERM ASSERTS WRDATAEN WHILE THE SECOND
           OR-TERM DEASSERTS WRDATAEN.                        }

WRDATAENEN       = !MEMSPARE2 ;
WRDATAEN        := RESETN AND (
                    (!WRN AND !ENSTARTN)
                  OR (WRDATAEN AND (!ENSTARTN OR !CYCENDN))
);

RDENNEN          = !MEMSPARE2 ;
RDENN       NOT := RESETN AND (
                    !RDN AND !ENSTARTN AND CYCENDN
);

{ PURPOSE: RDDATAENN IS CONNECTED TO THE MEMORY BOARD'S
           DATA TRANSCEIVER OUTPUT ENABLE (FCT623T OR FCT861)
           AND ONLY ENABLES FOR THIS BOARD'S CHIP SELECTS.
           IF THE MEMORY CONTROLLER IS USED FOR ANOTHER
           BOARD'S MEMORY, THEN THE TRANSCEIVER OUTPUT ENABLE
           SHOULD BE DISABLED FOR THOSE CHIP SELECTS (VIA
           MEMSPARE2.                                         }

{ NOTE:    IN MOST SYSTEMS, R305X'S DATAENN OUTPUT CAN BE
           CONNECTED DIRECTLY TO THE TRANSCEIVER ENABLE PIN
           INSTEAD OF USING A SYNTHESIZED RDDATAENN.          }

RDDATAENNEN      = !MEMSPARE2 ;
RDDATAENN   NOT := RESETN AND (
                    !RDN AND !ENSTARTN AND CYCENDN
);


{ PURPOSE: RESET SYNCHRONIZES THE POWER UP RESET FOR THE
           MEMORY CONTROLLER STATE MACHINES AND FOR THE R305X.  }

RESETNEN         = !MEMSPARE2 ;
RESETN      NOT := !POWRESETN ;
```

```
        { PURPOSE: GUARDED/GATED UART CHIP SELECT, GUARTCSN GATES
                   UARTCSN BECAUSE THE UART BEING USED HAS A MOTOROLA-
                   TYPE I/O DEVICE INTERFACE WHICH MULTIPLEXES ITS
                   READ/WRITE INPUT PIN SUCH THAT THE CHIP SELECT MUST
                   STROBE IN OR OUT DATA.  THIS IS IN CONTRAST TO AN
                   INTEL-TYPE I/O DEVICE INTERFACE WHICH WOULD HAVE A
                   SEPARATE READ STROBE AND WRITE STROBE AS WELL AS A
                   CHIP SELECT.  IT IS IMPORTANT NOT TO HAVE A
                   GLITCH (FROM ADDRESS DECODING THE CHIP SELECT) ON
                   READS IN ORDER TO ALLOW THE I/O DEVICE TO UPDATE
                   FIFO POINTERS, ETC.  THUS GUARTCSN STARTS LATE AND
                   ENDS EARLY, SO THAT READ/WRITE IS HELD VALID
                   THROUGHOUT THE CHIP SELECT.                          }

        GUARTCSNEN       = !MEMSPARE2 ;
        GUARTCSN    NOT := RESETN AND (
                              !UARTCSN AND !ENSTARTN AND CYCENDN
        );

END;
END UPAL3.
```

```
{ TITLE  : MEMINT.LPLC
           UPAL4 MEMORY I/O INTERRUPT CONTROLLER PAL FOR THE R305X BEHAVIORAL
           BUS EMULATOR MEMORY EVALUATION BOARD
  PURPOSE: REPLICATES THE TIMER/UART INTERRUPT CONTROLLER ON THE 7RS382 BOARD.
           ADDITIONAL FUSE BITS ADDED FOR 16V8 COMPATIBILITY.
  LANG   : LPLC — TM OF CAPILANO COMPUTING SYSTEMS
  AUTHOR : IDT INC.
  UPDATES: C3F98 01-04-91 16V8 PCB VERSION FIRST RELEASE A.N.
}


{ U24A_382 INTERRUPT PAL}
{ 1-2-90,12-14-89 }
{JEDEC file's CHECKSUM = 379E } { NOTE: 01-04-91 — NOT APPLICABLE TO 16V8 }

{     CONTROL PAL FOR 8254 TIMER'S AND UART INTERRUPT
      USED FOR EVALUATION BOARD 382 }

MODULE U24A_382;
TITLE U24A_382;
TYPE MMI 16R8;

{ FUSE BITS FOR 16V8 FAMILY ATTRIBUTES USED AS A 16R8 }
FUSE 2048..2079 00000000000000000000000000000000 ;
FUSE 2080..2111 00000000000000000000000000000000 ;
FUSE 2112..2143 00000000000000000111111111111111 ;
FUSE 2144..2175 11111111111111111111111111111111 ;
FUSE 2176..2193 111111111111111101               ;

INPUTS;

MRES/       NODE[PIN2];
UARTINT/    NODE[PIN3];
PMRD/       NODE[PIN4];
CSTIM/      NODE[PIN5];
EA02        NODE[PIN6];
EA04        NODE[PIN7];
OUT1        NODE[PIN8]; {input from Timer output OUT1}
OUT0        NODE[PIN9]; {input from Timer output OUT0}

DT0A/       NODE[PIN14]; {feedback}
DT0B/       NODE[PIN15]; {feedback}
T0INT/      NODE[PIN16]; {feedback}

DT1A/       NODE[PIN17]; {feedback}
DT1B/       NODE[PIN18]; {feedback}
T1INT/      NODE[PIN19]; {feedback}

OUTPUTS;

UINT5/      NODE[PIN13];
DT0A/       NODE[PIN14];
DT0B/       NODE[PIN15];
T0INT/      NODE[PIN16]; { goes to R3000's UINT3}

DT1A/       NODE[PIN17];
DT1B/       NODE[PIN18];
T1INT/      NODE[PIN19]; { goes to R3000's UINT4}
```

```
TABLE;
{     8254 TIMER generates 2 square-wave outputs OUT0 and OUT1.
      When OUT0 goes from high to low, this PAL asserts interrupt
      TOINT/, which will interrupt R3000 through UINT3.
      Same scheme applies to OUT1, T1INT/ and UINT4.
      Reading physical addresses 1F80 0010 and 1F80 0014 (which are
      virtual addresses BF80 0010 and BF80 0014 in this 382 board)
      will clear interrupt UINT3 and UINT4, respectively.

      This PAL also synchronizes UART interrupt signal }


DT0A/      :=     OUT0;       {delay TIMER's OUT0 through a register}
DT0B/      :=     DT0A/;      {delay again}
T0INT/ NOT :=     MRES/ AND
                  ((NOT DT0A/ AND DT0B/) OR
                  (NOT T0INT/ AND (NOT EA04 OR EA02 OR CSTIM/ OR PMRD/)));

DT1A/      :=     OUT1;
DT1B/      :=     DT1A/;
T1INT/ NOT :=     MRES/ AND
                  ((NOT DT1A/ AND DT1B/) OR
                  (NOT T1INT/ AND (NOT EA04 OR NOT EA02 OR CSTIM/ OR PMRD/)));

UINT5/     :=     UARTINT/ OR NOT MRES/ ;
                  {put UART's interrupt through a register to synchronize
                  it with R3000 clock }
END;
END U24A_382.
```