Embedded Control Design with the NS32CG16

National Semiconductor
Application Note 632
Jeff Goldberg
July 1989

1.0 INTRODUCTION

Today's real-time embedded control applications require higher performance from their microprocessors than in the past. Control applications are becoming more complex, requiring more computational horsepower and addressing range than previous generations of 8- and 16-bit microprocessors could provide. At the same time, there is increased need to reduce overall system cost.

The NS32CG16 is the heart of a high performance, cost-effective system solution for embedded control applications. The NS32CG16 is a 32-bit microprocessor in National Semiconductor's Embedded System Processor family that provides special features for embedded control applications. With a true 32-bit arithmetic logic unit (ALU), 16 dedicated and general purpose 32-bit wide registers, and a 32-bit wide internal data path, the NS32CG16 has the raw power required for today's compute-intensive applications. With its 24-bit address bus, the NS32CG16 can address up to 16 Mbytes of memory in a linear address space without the performance degradations of segmented or paged memory architectures.

For intensive floating-point applications, the NS32CG16 supports a slave processor interface to an external floating point unit (FPU). The microprocessor and FPU connect without any glue logic and the high speed slave protocol supports operand transfer rates of 15 Mbytes/sec between the two devices.

The NS32CG16 provides two forms of direct memory access (DMA) support. A DMA controller (DMAC) can be easily implemented using inexpensive discrete logic since the NS32CG16 provides system timing and control signals during DMA cycles. The microprocessor also performs DMA-like cycles under software control without the aid of a DMAC.

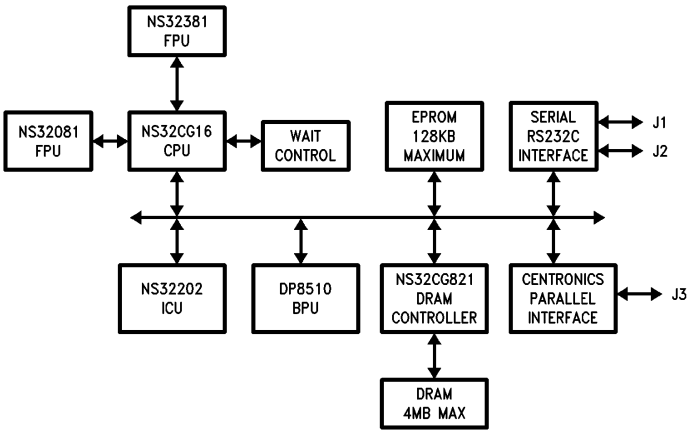
Unlike many other microprocessors, the NS32CG16 does not require separate bus controller and clock generator chips. The microprocessor generates all of the system timing and control signals. The 24-bit address and 16-bit data busses are multiplexed to reduce pin count and package size. The on-chip oscillator is designed to work with inexpensive third overtone crystals.

Since the NS32CG16's architecture is tuned for high level language (HLL) support, code size is typically 35% to 50% less than that required by other processors. Source code can be developed with a high level language such as C, thereby reducing development time. At the same time, fewer ROMs are needed, which reduces overall system cost.

This application note discusses design considerations for utilizing the NS32CG16 in real-time embedded control applications. It covers the following topics:

- An overview of the NS32CG16 memory organization
- An overview of the NS32CG16 bus operations
- System bus interface
- DRAM interface
- SRAM interface
- EPROM interface
- ICU interface
- Slave Processor interface
- BPU interface
- Serial port interface
- Parallel port interface

The NS32CG16 RT board shown in *Figure 1* is presented as a design example of an embedded control system, but is not targeted towards a specific application area. Instead a more general approach has been taken to show how the NS32CG16 is interfaced to a variety of memories and peripherals.



TL/EE/10465–1

FIGURE 1. NS32CG16 Functional Block Diagram

©1995 National Semiconductor Corporation TL/EE10465

RRD-B30M75/Printed in U. S. A.

The board is designed for 15 MHz operation and can be configured with up to 4 Mbytes of DRAM, 64 Kbytes of SRAM, and 128 Kbytes of EPROM. The board also features:

- The NS32381 and NS32081 FPUs for fast floating point execution.
- The NS32202 ICU for interrupt vectoring and program execution timing.
- The NS16552 DUART for dual channel serial communication.
- The DP8510 BPU for BITBLT acceleration.
- The NS32CG821 DRAM controller for easy, efficient DRAM interfacing.
- A Centronics compatible parallel printer port.

2.0 MEMORY ORGANIZATION

The 24-bit address bus of the NS32CG16 provides up to 16 Mbyte of memory in a uniform linear address space starting at zero and ending at 0xFFFFFF. Each memory location contains an eight-bit byte. Two contiguous bytes form a word. Two contiguous words form a doubleword. A word or doubleword can start at any address, since there are no memory alignment requirements with any of the processors in the Embedded Systems Processors family. Although addressable as bytes, memory is organized as words, where the address of the word is the address of its least significant byte.

While the NS32CG16 has no memory alignment requirement, alignment does affect the time to access a word or doubleword. Words (and doublewords) whose addresses are a multiple of two are accessed more quickly than those whose addresses are otherwise. It takes one bus cycle to access word-aligned words and two bus cycles to access word-aligned double words, whereas it takes two bus cycles to fetch non-word-aligned words and three bus cycles to fetch non-word-aligned doublewords.

The NS32CG16 supports memory mapping of I/O, peripheral devices, and slave processors. Such devices can be located anywhere in the address space. The following section describes the control signals for memory or I/O interfacing.

3.0 BUS OPERATIONS

The NS32CG16 performs six types of bus operations:

1. Instruction fetch
2. Memory read
3. Memory write
4. Interrupt acknowledge and return from an interrupt service routine.
5. EXTBLT cycle
6. Transfer information to and from a Slave Processor.

Cases 1 through 4 have identical bus timing characteristics and are discussed first. The only external difference among these cases is a 4-bit code placed on the bus status pins (ST0–ST3) during bus cycles for identifying which operation is occuring. Cases 5 and 6 have additional control signals and will be discussed later.

3.1 Memory and I/O Interface Signals

The following control signals provide the memory and I/O interface to the NS32CG16:

CTTL:	System clock. This signal should be used for timing reference and for synchronization of external devices with the microprocessor.
AD0–AD15:	These signals are a multiplexed address/data bus. AD0 is the least significant bit (LSB).
A16–A23:	These signals are the high-order bits of the address bus.
/HBE:	High-byte enable. This signal enables the most significant byte of an address during data transfers.
/ADS:	Address strobe. This signal controls address latches and provides the earliest indication that a bus cycle is in progress.
/DDIN:	Data direction in. This signal indicates the direction of the data transfer.
/DBE:	Data buffer enable. This signal is used to control the data buffers.
/TSO:	Timing state output. This signal identifies the beginning of state T2 and the end of state T3 of a bus cycle.
/RD:	Read strobe. This signal enables reading of data from memory or peripherals.
/WR:	Write strobe. This signal enables writing of data to memory or peripherals.
ST0–ST3:	Bus status. These four signals indicate the type of bus cycle currently in progress. If the processor is idle on the bus, these outputs indicate the reason.
/WAIT1–2:	Two-bit wait state inputs. These inputs are binary weighted to allow time for zero to three wait states to be inserted into a bus cycle.
/CWAIT:	Continuous wait. This signal inserts continuous wait states into a bus cycle as long as it is sampled low.

3.2 Read and Write Cycle Timings

Figure 2 is the timing diagram for a read cycle, and *Figure 3* for a write cycle. Both figures assume that the selected memory or peripheral is capable of communicating with the processor at full speed.

A full-speed bus access occurs during four cycles of CTTL, T1 through T4. During T1, the microprocessor applies an address on AD0–AD15 and A16–A23. The processor asserts /ADS to inform external circuitry that a bus cycle is beginning and to provide control to an external address latch for demultiplexing address bits 0–15 from AD0–AD15. Also during T1, the /DDIN and /HBE control signals become active to control the direction of data transfer through the data buffers, and to enable the high-order byte of memory, respectively.

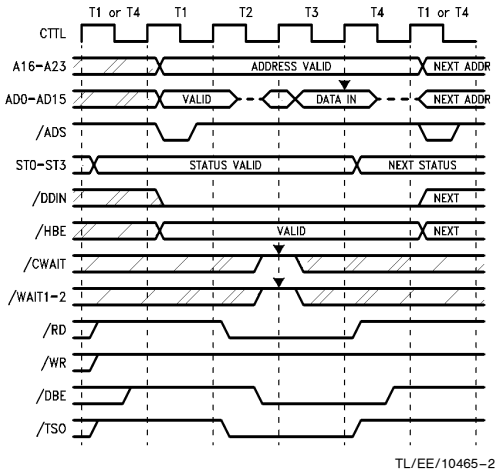


FIGURE 2. Read Cycle Timing

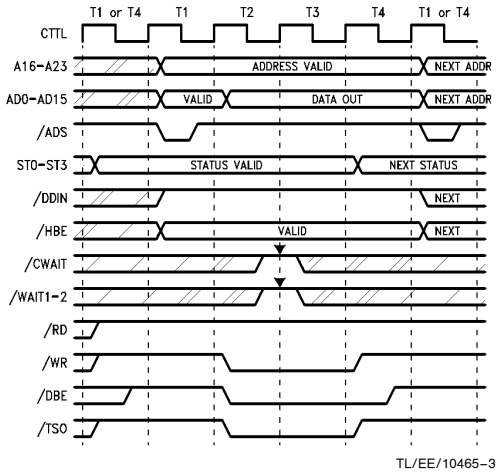


FIGURE 3. Write Cycle Timing

During T2, the microprocessor switches the data bus, AD0–AD15, to either accept or present data. The high-order bits of the address bus, A16–A23, remain valid and do not have to be latched. The signals /TSO, /DBE and either /RD or /WR are asserted at this time.

The T3 state provides for access time requirements and occurs at least once during each bus cycle. Since all bus control signals from the microprocessor are flat during T3, a bus cycle may be extended by causing T3 to be repeated. The wait state request lines /CWAIT and /WAIT1–2 are used for this purpose.

If the microprocessor is performing a read cycle, the data bus is sampled on the rising edge of CTTL at the end of T3. However, data must be held a little longer to meet the data hold requirements. The /RD signal is guaranteed to remain active before this time, so its rising edge can be used to disable the device providing the input data.

The T4 state finishes the bus cycle. At the beginning of T4, the /RD (or /WR) and /TSO signals become inactive. After providing for the necessary data hold time, /DBE will become inactive on the falling edge of CTTL in the middle of T4. During write cycles, data from the microprocessor remains valid throughout T4. The bus status lines (ST0–ST3) change at the beginning of T4, anticipating the bus cycles (if any) that will follow.

Both /CWAIT and /WAIT1–2 are sampled on the rising edge of CTTL at the end of T2. If any of these signals are active, the bus cycle will be extended by at least one clock cycle. Any combination of these signals may be activated at one time, however, the /WAIT1–2 inputs are only sampled by the microprocessor at the end of T2. These signals are ignored at all other times. *Figure 4* shows a bus cycle extended by three wait states, two of which are due to /WAIT2, and one due to /CWAIT.

3.3 EXTBLT Bus Operation Timing

The NS32CG16 can perform DMA-like bus cycles under software control. To accomplish this, the NS32CG16's general purpose registers should be loaded with the source and destination addresses, the count length, and other parameters associated with BITBLT operations (see the NS32CG16 data sheet for details). The EXTBLT bus operations will commence with the execution of the EXTBLT instruction.

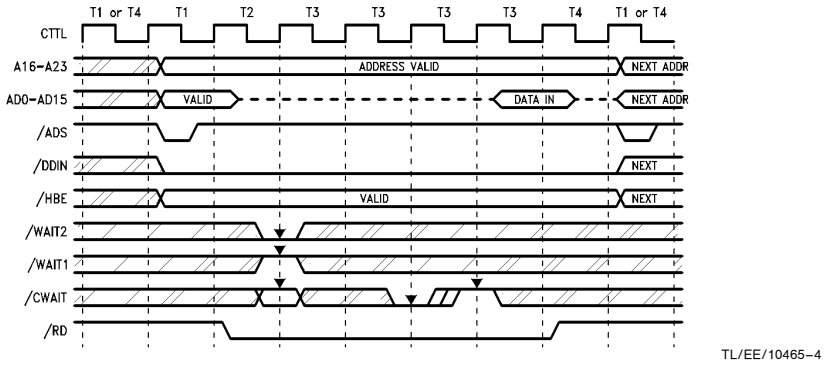
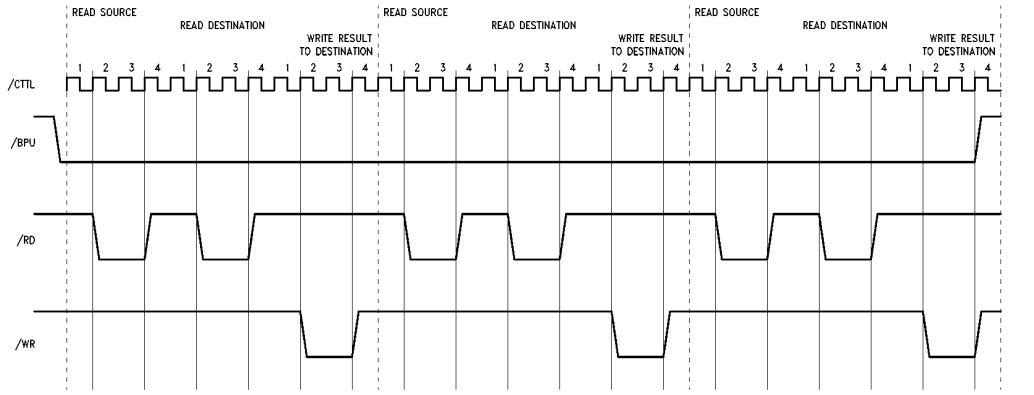


FIGURE 4. Cycle Extension of a Read Cycle



TL/EE/10465–5

FIGURE 5. EXTBLT Bus Operation

Figure 5 is the timing diagram for an EXTBLT bus operation. The operation consists of iterations of source and destination reads followed by destination writes. An external signal, /BPU, is used to indicate to external hardware that an EXTBLT bus operation is in progress.

4.0 SYSTEM BUS INTERFACE

Figure 6 shows a block diagram of the system bus interface. The intent of the system's bus architecture is to:

1. Limit the data buffers' capacitive loading to 50 pF or less.
2. Allow EXTBLT operations to be performed between EPROM and DRAM or SRAM.
3. Allow hardware to be added to a buffered wire-wrap area and perform EXTBLT operations between the wire-wrap area and DRAM or SRAM.

Since the board was designed for maximum user flexibility, the design described here allows for several memory configurations. Should the maximum amount of memory be installed on the board, a single pair of 74AS245 data buffers would be too heavily loaded to drive the full complement of memory and I/O. Therefore, three data buses are employed. The DRAM, SRAM, and BPU are located on one data bus with the EPROM, DUART, and PPI located on the second data bus. A third data bus is provided for the USER wire-wrap area.

In order to allow the BPU to access data from EPROM during an EXTBLT operation without having it first copied into RAM, buffers were installed between the BRD (Buffered RAM Data) and BED (Buffered EPROM Data) buses. Buffers were also installed between the USER and BRD buses to allow devices residing on the USER bus to access memory residing on the BRD bus during an EXTBLT operation.

Note: These considerations make system bus interface for the RT Board seem somewhat complicated. In order to achieve a simpler, more cost-effective design, the source and destination of an EXTBLT operation should reside on the same bus as the BPU.

Referring to sheets 1 and 2 of the schematics in Appendix D, the bus interface consists of:

- 74AS373 latches to demultiplex the 16-bit data bus.
- 74AS245 data buffers to float the microprocessor off the system bus and connect the EPROM and RAM buses (or USER and RAM buses) during an EXTBLT operation.
- B-series PAL address decode to generate chip selects.
- A-series PAL for data-buffer-enable control.

The address decoder is designed to map the SRAM (via three jumpered inputs) into a contiguous address space directly above the DRAM installed on the board (see Appendix A for state diagrams and PAL equations, and Appendix B for jumper options). By using a B-series PAL, decoder outputs /ROM, /SRAM, /DRAM, /USER0 and /USER1 are valid 45 ns after the start of the bus cycle. Decoder outputs /ICU, /IOSEL0 and /IOSEL1 are valid 6 ns later due to the propagation delay of address bits A15 and A8 through the address latch.

The microprocessor drives the data bus during the write bus cycle of an EXTBLT operation. The BPU must be located on the system side of the bus to avoid bus contention. Data buffer control logic disables buffers G11, F11, B6, D8, E19 and D19 to float the microprocessor off the system bus. The data-buffer-enable-control PAL uses the address decoder outputs along with the following control signals to qualify /DBE and enable the appropriate data buffer:

1. /SYNCBPUCYC: Synchronized BPU cycle. A BPU state machine interface output signal that is used to disable buffers G11, F11, B6, D8, E19, and D19 and enable buffers G10 and F10, or D20 and B20 during an EXTBLT operation.
2. B__ENBPU: Enable BPU. A BPU state machine interface output signal that is used to distinguish between an EXTBLT operation that uses the BPU located on the RAM bus, or an EXTBLT operation that uses a data processing device located on the USER bus.
3. /HLDA: Hold acknowledge. A microprocessor output signal that indicates when the microprocessor has relinquished control of the bus. This signal is provided for future expansion, but is not currently used in this design.

The outputs of the PAL are:

1. /BED: Buffered EPROM Data enable. This signal allows the microprocessor to access the memory and I/O residing on the BED bus when /DBE is active. /BED is disabled during an EXTBLT operation.

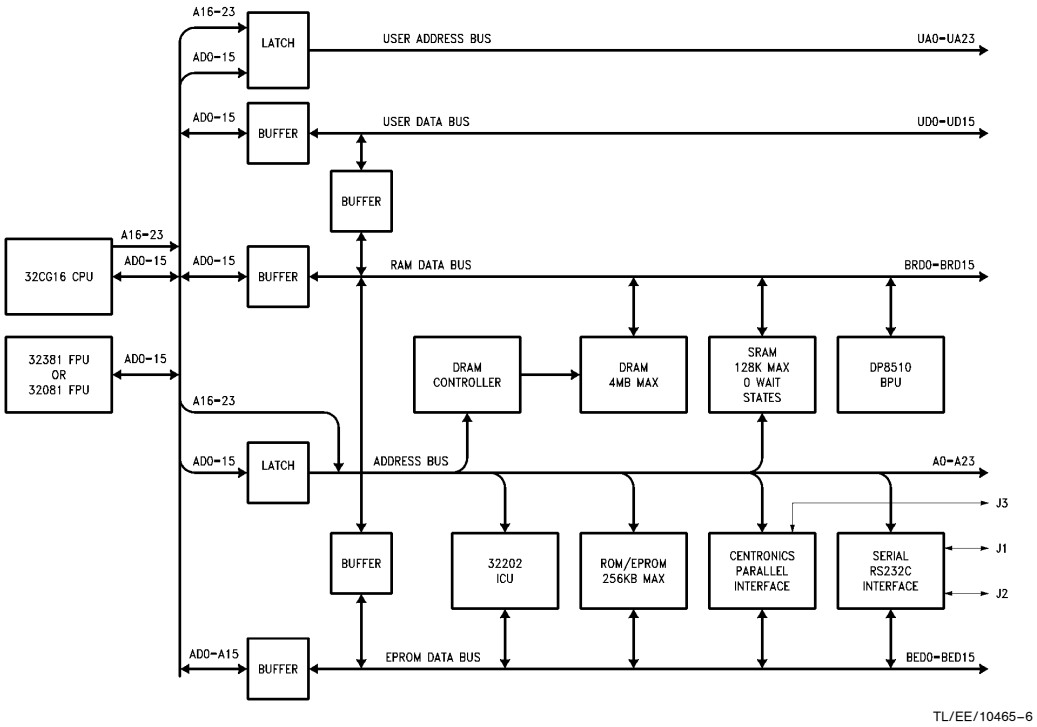


FIGURE 6. NS32CG16 RT Board System Bus Architecture

2. /BRD: Buffered RAM Data enable. This signal allows the microprocessor to access the memory residing on the BRD bus when the /DBE is active. /BRD is disabled during an EXTBLT operation.
3. /USER: USER bus enable. This signal allows the microprocessor to access the wire-wrapped components residing on the USER bus when /DBE is active. /USER is disabled during an EXTBLT operation.
4. /BRDBLT: Board Blit. This signal allows the BPU residing on the BRD bus to access EPROM during an EXTBLT operation when /DBE is active. /BRDBLT can be enabled only during an EXTBLT operation.
5. /USERBLT: USER Blit. This signal allows a data processing device residing on the USER bus to access the on board RAM during an EXTBLT operation when /DBE is active. This signal can be enabled only during an EXTBLT operation.

4.1 Data Buffer Control Timing

Figure 7 is a diagram of data buffer control timing during a read from EPROM, a write to SRAM, and followed by an EXTBLT bus operation in which the source is in EPROM and the destination is in SRAM. Note that /BPU may become active during T4 of a non-EXTBLT bus cycle and become inactive during T4 of an EXTBLT bus cycle. Since the data buffers are enabled until the middle of T4, /BPU must be qualified with ST0–ST3 and synchronized with CTTL to prevent the bus contention and data hold violations that would result from the simultaneous enabling and disabling of the board's data buffers.

The outputs of the PAL are fed into OR gates to qualify /DBE. If /DBE was qualified internally to the PAL, the propagation delay could cause the data buffers to continue driving the bus during T1 of the next bus cycle.

The bus interface design provides 132 ns from the time the memory receives a valid address until it must provide valid data. This allows enough time for microprocessor address valid delay, data setup time, address decoder delays, and data buffer propagation delays. The access time for the ICU and I/O is 6 ns less due to propagation delay of bits A15 and A8 through the address latch.

$$\begin{aligned}\text{Access Time} &= 3T - tAHv(CG16) - tDIs(CG16) - \\ &\quad tP(B\ PAL) - tP(AS245) \\ &= 3(66.6) - 30 - 15 - 15 - 7.5 \\ &= 132.3\ \text{ns}\end{aligned}$$

5.0 DRAM INTERFACE

This section presents the results of a timing analysis and describes the design of a DRAM interface optimized for speed, flexibility, and ease of design. The NS32CG821 DRAM controller is presented as a single chip solution for interfacing DRAM to the microprocessor. This interface allows the controller to drive 256 Kbit or 1 MBit DRAMs in an array of 512 Kbytes, 1 Mbyte, 2 Mbytes, or 4 Mbytes at 15 MHz with zero to two wait states. (Refer to sheet 3 of the schematics in Appendix D.)

Note: The following discussion assumes a familiarity with operational modes of the NS32CG821 or DP8421A.

The NS32CG821 generates all the required access control signal timing for DRAMs. An on-chip refresh request clock is used to automatically refresh the DRAM array. Refreshes and accesses are arbitrated on chip. If necessary, a /WAIT output inserts wait states into DRAM accesses in order to guarantee /RAS low time during accesses and /RAS precharge time after refreshes and back-to-back accesses.

5.1 DRAM Controller, Microprocessor, and DRAM Interface Signals

The following NS32CG821 control signals are used to interface the controller to the microprocessor and DRAM array.

Address, R/W, and Programming Signals:

1. R0–9, C0–9: Row address, Column address. These signals are driven by the address bus. They are used to specify the row and column addresses during an access to DRAM, and are used to program the chip when /ML is asserted.
2. B0,B1: Bank select. These signals are driven by address bits A19 or A21. Used to switch between the lower and upper banks of DRAM, and are also used to program the chip when /ML is asserted.
3. /ECAS0–ECAS3: Enable /CAS. These signals are driven by byte enable signals /HBE and A0. They are used to select the /CAS strobe to be enabled during an access to DRAM.
4. /WIN: Write enable in. This signal is driven by inverted /DDIN and identifies a write when active.
5. /ML: Mode load. When low, this signal enables the internal programming register that stores programming information.

DRAM Control Signals:

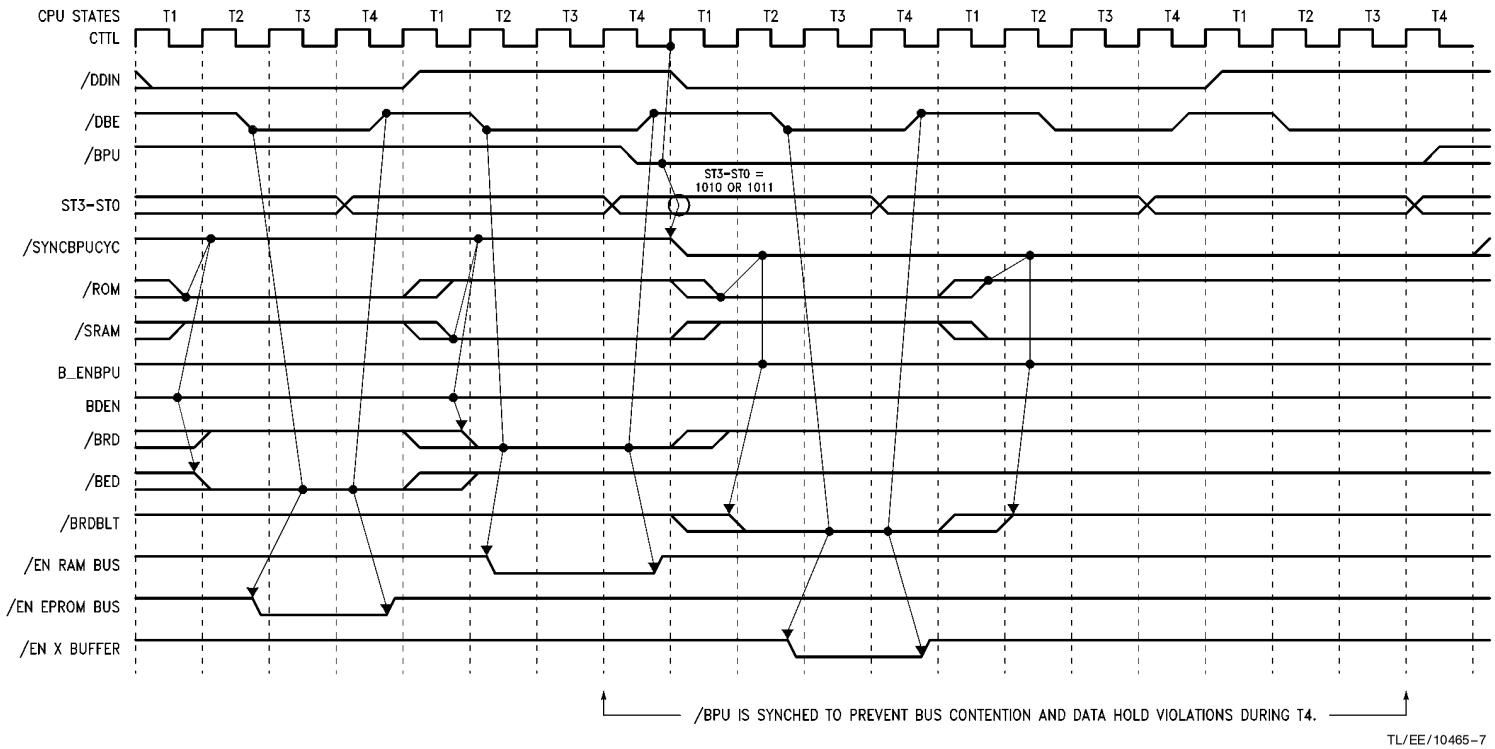
1. Q0–9: DRAM address. Multiplexed outputs of the R0–9, C0–9 inputs that form the DRAM address bus.
2. /RAS0–3: Row address strobes. These signals are used to latch the row address contained on Q0–9 into the DRAM.
3. /CAS0–3: Column address strobes. These signals are used to latch the column address contained on Q0–9 into the DRAM.
4. /WE: Write enable. This signal is controlled by /WIN and specifies the direction of data transfer to the DRAM.

Access Signals:

1. ALE: Address strobe. This signal is driven by an inverted microprocessor /ADS strobe, and indicates the start of bus cycle to DRAM controller.
2. /CS: Chip select. This signal is driven by the /DRAM address decoder output signal to indicate that a DRAM access is in progress.
3. /TSO: Timing state out. This signal is driven by the microprocessor /TSO output signal, and is used to deassert /RAS and /CAS when /TSO goes inactive at the end of T3.
4. /WAIT: Wait. This signal drives the microprocessor /CWAIT input signal to insert wait states during a DRAM access.
5. /WAITIN: Wait increase. This signal is used to insert one additional wait state into a bus access above the number of waits programmed into the DRAM controller during device initialization.

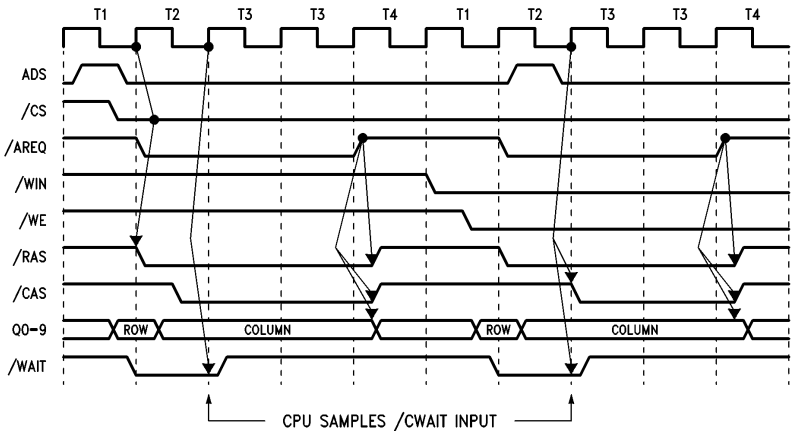
Clock Inputs:

1. CLK: System clock. This signal provides the clock input for the internal refresh/access arbitration state machine, and is used as the reference for /RAS precharge time, /RAS low time during refresh, and /WAIT signal extension.
2. DELCLK: Delay line clock. This signal is used to generate the delay between /RAS and /CAS, and is also used to create the internal refresh request clock.



TL/EE/10465–7

FIGURE 7. Data Buffer Control Timing



TL/EE/10465–8

FIGURE 8. DRAM Access Cycle Timing

5.2 DRAM Read and Write Cycles

Figure 8 is a timing diagram for DRAM read and write cycles that have one wait state. The access is initiated by pulsing ALE high and asserting /CS. If a DRAM refresh is not in progress, the pair of /RAS strobes (selected by the bank enabled signals B0 and B1) will be asserted on the rising edge of CTTL at the beginning of T2. /WAIT is asserted during T2 to insert one wait state into the bus access.

During a read cycle, the /CAS strobe(s) selected by the byte enable signals /ECAS0–3 will go active sometime during T2 or the early part of T3. During a write cycle /CAS is guaranteed to remain inactive until the beginning of T3. /WAIT will go inactive at the beginning of T3 after the CG16's /CWAIT hold time requirement is met. The access is terminated when /TSO goes inactive at the end of T3.

The DRAM controller is a software programmable device that must be programmed according to the DRAM's access time requirements, system clock frequency, memory architecture, and refresh scheme. In this design, the DRAM controller may be programmed for 10 or 15 MHz operation and to insert zero or one wait states during a bus access. If /WAITIN is pulled low, the controller will insert one additional wait state. The leading edge of /CAS is delayed during write cycles to guarantee the memory's data setup time. Since there is enough time during T4 and T1 of the next bus cycle for DRAM precharge, the controller is programmed for a non-interleaved access memory architecture. The controller is programmed to perform /RAS only refresh cycles when requested by the internal refresh request timer.

The controller is programmed according to the bits present on the address bus. There are several methods of programming the controller. One way is to assert /ML, present a programming selection on the address bus, then deassert /ML. By using one OR gate and a D F/F, the controller is programmed on the first microprocessor write cycle after power up. In effect, the device is programmed by writing to one of four addresses on the board, each ''address'' being the programming selection.

After programming, the NS32CG821 enters a 60 ms initialization period, during which time it performs dummy accesses to DRAM. The bootstrap program contains a delay loop to ensure that DRAM is not referenced during this time.

The routine used to program the controller according to the contents of the configuration dipswitch is as follows:

```
#LOAD TABLE WITH PROGRAMMING SELECTIONS
    .data

dramc: .double h'1245e6, h'13454c, h'1345b2, h'174518
#
#PROGRAM DRAM CONTROLLER
#
    .text
    .set switch,h'0fe8081  #POINT TO CFG SWITCH
    movb @switch,r6        #READ SWITCH
    andd $h'0c0,r6         #MASK ALL BUT SW8 and SW7
    ashd $-6,r6            #R6 BECOMES INDEX INTO DRAMC TABLE
    movd @dramc[r6:d],r6   #MOVE TABLE ENTRY INTO R6
    movb $0,0(r6)          #PERFORM WRITE ACCESS
#
#60 MS DELAY LOOP @15 MHZ
#
    movd $d'31000,r6       #LOOP 31K TIMES
delay: nop                 #EACH LOOP TAKES AT LEAST 30 CLOCKS
    acbd $-1,r6,delay      #WHICH IS 1980NS @15MHZ
```

The two clock inputs, CLK and DELCLK, may both be tied to the same clock input, or they may be separate clocks running at different frequencies, asynchronous to each other. The CLK input is driven by the system clock, CTTL. The DELCLK input may be driven by CTTL or by an independent clock source if the NS32CG16's power-saver mode is to be used.

When the microprocessor enters power-saver mode, the reduced CLK frequency will not affect the DRAM controller. The same cannot be said for DELCLK. The internal delay clock divisor is a programmable parameter dependent on the known operating frequency of DELCLK. If DELCLK is slowed down, the period between internal refresh requests will be longer, resulting in a loss of data integrity.

If it is known that the microprocessor will no longer require the data held in DRAM, or that DRAM will not be referenced while the microprocessor is in power-saver mode, DELCLK can be driven by CTTL. If data integrity is required, or if DRAM is to be referenced while in power-saver mode, the DRAM controller can still be driven by CTTL but must be first reprogrammed to operate at the lower clock frequency. An alternate solution is to provide an independent clock source for DELCLK.

The DRAM controller's maximum /RAS delay, referenced from the start of T2, is dependent on the capacitive load that /RAS is driving. Assuming a 100 pF load, the maximum delay is 26 ns.

$$t(CLK-RAS)@100\ pF = t(CLK-RAS)@50\ pF + (8\ ns/100\ pF)(100\ pF-50\ pF) = 22 + 4 = 26\ ns$$

The DRAM controller's maximum /CAS delay, referenced from the start of T2, is dependent on the capacitive load that /CAS is driving and the programmed parameters for tRAH and tASC. In order to allow the use of DRAMs from several different vendors, tRAH and tASC are programmed as shown:

tRAH = 15 ns, tASC = 0 ns @15 MHz, 0 Wait States
tRAH = 25 ns, tASC = 0 ns @15 MHz, 1 Wait State
tRAH = 25 ns, tASC = 0 ns @10 MHz, 0 Wait States
tRAH = 25 ns, tASC = 10 ns @10 MHz, 1 Wait State

Assuming a 100 pF load, the maximum /CAS delay is:

77 ns @15 MHz, 0 Wait States
87 ns @15 MHz, 1 Wait State
87 ns @10 MHz, 0 Wait States
97 ns @10 MHz, 1 Wait State

$$t(CLK-CAS)@100\ pF = t(CLK-CAS)@50\ pF + (9.33\ ns/100\ pF)(100\ pF-50\ pF) = 72 + 4.67 = 76.67\ ns\ @\ 15\ MHz,\ 0\ Wait\ States$$

Considering the DRAM controller's CLK high to /RAS asserted delay, CLK high to /CAS asserted delay and delay adjustment when DELCLK is driven at an "odd" frequency (15 MHz), data buffer propagation delay and microprocessor data setup time, the access times for tRAC and tCAC are:

tRAC = 84 ns, tCAC = 30 ns @15 MHz, 0 Wait States
tRAC = 150 ns, tCAC = 86 ns @15 MHz, 1 Wait State
tRAC = 151 ns, tCAC = 90 ns @10 MHz, 0 Wait States
tRAC = 251 ns, tCAC = 180 ns @10 MHz, 1 Wait State

$$tRAC = 2T - t(CLK-RAS) - tp(AS245) - tDIs = 133 - 26 - 7.5 - 15 = 84.5\ ns\ @15\ MHz,\ 0\ Wait\ States$$

$$tCAC = 2T - t(CLK-CAS) - (Delay\ Adjust) - tp(AS245) - tDIs = 133 - 77 - 3 - 7.5 - 15 = 30.5\ ns\ @15\ MHz,\ 0\ Wait\ States$$

6.0 SRAM INTERFACE

This section presents the results of a timing analysis and describes the design of an SRAM interface optimized for simplicity and speed. The interface is designed to provide 64 kbyte of zero-wait-state memory. It consists of two 32k*8 SRAMs and two ALS OR gates. (Refer to sheet 3 of the schematics in Appendix D.)

The SRAM's chip select lines are driven by the address decoding PAL's /SRAM output. By driving /CE directly with /SRAM, both the chip count and time to /CE valid is minimized.

Both the high and low bytes of SRAM are enabled during a read cycle. Since the NS32CG16's /RD strobe is guaranteed to be active long enough to satisfy the microprocessor's data hold requirement, there is no need to delay the trailing edge of the strobe. The SRAMs' /OE signal is driven directly by /RD.

Two ALS OR gates are used to qualify the microprocessor's /WR strobe with the byte enable signals /HBE and A0. This allows the microprocessor to selectively enable one or both bytes of SRAM during a write cycle.

Considering the memory access time provided by the bus interface, 120 ns SRAMs may be used at 15 MHz. If the clock frequency is reduced to 10 MHz, 200 ns SRAMs may be used. See the bus interface section for an example access time calculation.

7.0 EPROM INTERFACE

This section presents the results of a timing analysis and describes the design of an EPROM interface optimized for speed and simplicity. The interface is designed to provide either 64 kbytes or 128 kbytes of memory. The interface consists of two 32k*8 or 64k*8 EPROMs and one jumper to configure the board for the two EPROM sizes. (Refer to sheet 4 of the schematics in Appendix D.)

Both bytes of EPROM are selected and enabled during read cycles. The EPROM's chip select lines are driven by the address decoder PAL's /ROM output. By driving /CE directly with /ROM, both the chip count and time to /CE is minimized. Since the NS32CG16's /RD strobe is guaranteed to be active long enough to satisfy the microprocessor's data hold requirement, there is no need to delay the trailing edge of the strobe. The EPROMs' /OE is driven directly by /RD.

Considering the memory access time provided by the bus interface, the following EPROMs may be used:

120 ns @15 MHz, 200 ns @10 MHz, 0 Wait States
170 ns @15 MHz, 250 ns @10 MHz, 1 Wait State
250 ns @15 MHz, 350 ns @10 MHz, 2 Wait States
300 ns @15 MHz, 3 Wait States

8.0 INTERRUPT CONTROLLER INTERFACE

This section presents the results of a timing analysis and describes the design of an ICU interface optimized for speed and flexibility. (Refer to sheet 4 of the schematics in Appendix D.)

Note: The following discussion assumes a familiarity with operational modes of the NS32202 ICU and Series 32000 interrupt handling structure.

The NS32202 ICU is a support circuit that minimizes the software and real-time overhead required to handle multi-level, prioritized interrupts. It manages up to 16 interrupt sources, resolves interrupt priorities, and supplies a single byte interrupt vector to the microprocessor.

The ICU is operated in 8-bit bus mode to make available 16 hardware interrupt positions. Eight of those positions are available for software vectored interrupts. The ICU registers appear at even addresses (A0 = 0) since the ICU communicates with the least significant byte of the data bus. The ICU registers appear on doubleword boundaries to maintain compatibility with previous versions of software. The addresses of the ICU's internal registers are described in Appendix C. The base address of the ICU's internal registers is FFFE00.

Note: Address line A8 is a product term of the address decoder PAL's /ICU equation. This is done to prevent the microprocessor from reading the ICU's hardware vector register (HVCT) while acknowledging a non-maskable interrupt.

While acknowledging a non-maskable interrupt on its NMI input pin, the NS32CG16 performs a dummy read from address FFFF00 with the same status as an interrupt acknowledge cycle. If the HVCT is mistakenly read (by not decoding A8), there will be internal changes to the ICU. Any pending edge-triggered interrupts in position 15 will be cleared. If the auto-rotate mode is selected, the first priority register (FPRT) is cleared, thus preventing any interrupt from being acknowledged. In this case, a re-initialization of the FPRT register is required for the ICU to acknowledge interrupts again.

The ICU's software vector register (SVCT) is a copy of the HVCT. It allows the programmer to read the contents of the HVCT without initiating an interrupt acknowledge or return from interrupt cycle in the ICU. If the SVCT is read while ST1 equals 1, the ICU will provide the vector of the interrupt currently being serviced. If ST1 equals 0, the vector of the highest priority pending interrupt is provided instead.

ST1 equals 1 during a data transfer. If the ICU's ST1 is driven directly by the microprocessor, the programmer will only be able to read the vector of the interrupt currently being serviced. By qualifying ST1 with address line A7, the programmer will be able to read the current interrupt vector from address FFFE04 and the highest priority pending interrupt vector from address FFFE84.

By connecting ST1 in this fashion, the programmer can poll for interrupts coming from the master or cascaded ICUs by reading the master's SVCT. Once the cascaded ICU that is the source of the pending interrupt is identified, that ICU is polled to determine which interrupt needs servicing.

The maximum operating frequency of the ICU's internal counters is 10 MHz. Jumper W4.3 is used to configure the counters to run at one half the frequency of CTTL when the system clock exceeds 10 MHz.

8.1 ICU Bus Access Timing

Figure 9 is a timing diagram of an ICU bus access. The leading edge of the NS32CG16's /RD and /WR strobes is delayed to meet the ICU's status and chip select set up times. The access strobes /ICURD and /ICUWR are generated by qualifying /RD and WR with a /TSO strobe that is delayed one T-state by a D F/F.

Considering that the leading edges of the /RD and /WR strobes are delayed one T-state and the minimum strobe width is 160 ns, two wait states are required for an ICU access.

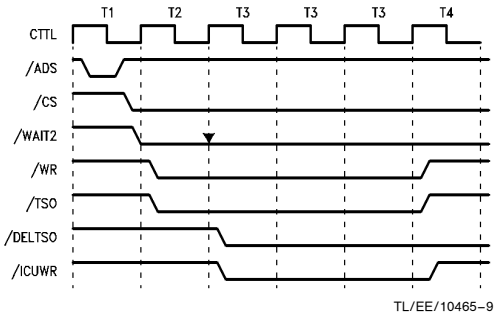


FIGURE 9. ICU Write Cycle

9.0 SLAVE PROCESSOR INTERFACE

The capabilities of the NS32CG16 can be expanded by using an external floating point unit (FPU) which interfaces to the NS32CG16 as a slave processor. National Semiconductor offers two FPUs that interface with the NS32CG16, the NS32081 (a low-cost solution), and the NS32381 (a high-performance solution).

Both FPUs are interfaced to the NS32CG16 in a tightly coupled configuration. The NS32CG16 and FPU communicate via the special Slave Protocol. (Refer to sheet 1 of the schematics in Appendix D.) The /SPC pin is used as the data strobe for slave processor transfers. In a Slave Processor bus cycle, data is transferred on the data bus and the status lines are monitored by the FPU to determine the type of transfer being performed.

9.1 Slave Processor Bus Cycle Timings

Figures 10 and *11* are timing diagrams of Slave Processor bus cycles. A Slave Processor bus cycle always takes two bus cycles, labled T1 and T4. During a read cycle, /SPC is active from the beginning of T1 until the beginning of T4. The FPU samples the status lines on the leading edge of /SPC. The NS32CG16 samples the data at the end of T1. During a write cycle, the microprocessor applies data and activates /SPC at T1, removing /SPC at T4. The FPU samples the status lines on the leading edge of /SPC and latches the data on the trailing edge.

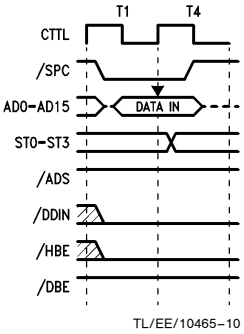


FIGURE 10. Slave Processor Read Cycle

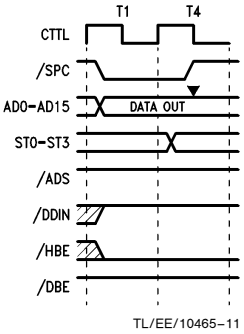


FIGURE 11. Slave Processor Write Cycle

The microprocessor does not pulse /ADS and no bus signals are generated. The direction of a transfer is determined by the sequence (slave protocol) established by the instruction under execution. The microprocessor does generate /DDIN to indicate the direction of transfer for debugging purposes.

10.0 BPU INTERFACE

Specialized hardware accelerators such as BITBLT processors and Multiplexer/Accumulators may be easily interfaced to the NS32CG16. The NS32CG16 RT board has provisions for interfacing the DP8510/11 BPUs to the BRD bus. A complete analysis of the DP8510 BPU interface is presented in the NS32CG16 Graphics Applications Note #2 ''Simple Embedded Control NS32CG16 System'' and is not repeated here.

A block diagram showing how to connect the interface to the board is shown on sheet 6 of the schematics in Appendix D. The connections to the address decoder have been highlighted to facilitate the mapping of the Control, Mask and Count register to their proper physical addresses. The Control and Count registers are connected to the BED bus while the Mask register is not directly connected to any bus.

The interface generates two output signals. /BPUCYC and B__ENBPU, used by the data buffer enable control logic to determine which set of data buffers are enabled during EXTBLT operations. /BPUCYC goes active to identify an EXTBLT bus operation. B__ENBPU is under software control and is set or cleared by writing to the B__ENBPU bit in the BPU interface's Control register. When B__ENBPU is active, the DP8510 BPU residing on the BRD bus is enabled. When B__ENBPU is inactive, EXTBLT operations will be performed using hardware on the USER bus.

11.0 SERIAL PORT INTERFACE

This section presents the results of a timing analysis and describes the design of a serial port interface optimized for speed and simplicity. The interface uses the NS16C552 DUART, CMOS drivers and receivers, and power generation circuitry to generate ±10V for the RS232 drivers (Refer to sheet 5 of the schematics in Appendix D.)

The NS16C552 is a dual version of the NS16550A UART. It is software compatible with the NS16450 and NS16550 and has improved performance that allows zero-wait-state communication with the NS32CG16.

The RT board uses CMOS drivers and receivers to reduce power consumption. Two LMC7660 switched capacitor voltage inverters are used to generate ±10V to power the NS14C88 RS232 driver. One LMC7660 is configured as a voltage doubler to supply +10V for the driver's +V and the second inverter's V_{IN} inputs. The second inverter supplies −10V for the driver's −V input.

12.0 PARALLEL PORT INTERFACE

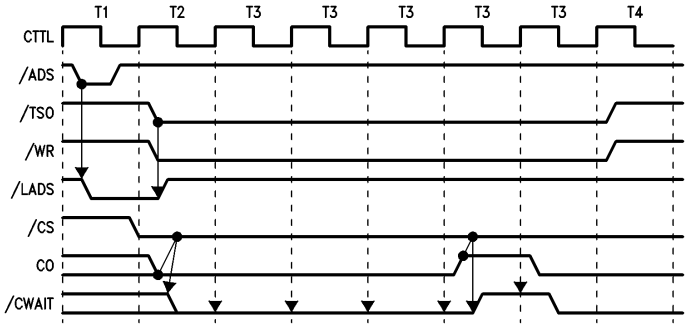
This section presents the results of a timing analysis and describes the design of a parallel port interface optimized for compatibility with Tiny Development System software.

The interface consists of an 8255 PPI, 7437 and ALS640 output buffers, and an ALS161 counter for wait state generation. (See sheet 5 of the schematics in Appendix D.) When the jumpers are configured as shown in Appendix B, the port is capable of driving an EPSON or compatible parallel printer.

Considering the minimum /WR pulse width required by the 8255, a parallel port access requires four wait states. Since the 8255 is an extremely slow peripheral, a counter is required to hold the NS32CG16's /CWAIT input low (the microprocessor's /WAIT1 and /WAIT2 inputs may request up to three wait states).

12.1 PARALLEL PORT ACCESS TIMING

Figure 12 is the timing diagram for a parallel port access. The microprocessor's /ADS is latched during T1 by an S/R flip flop. The latched strobe, /LADS, allows counter C15 to be loaded on the rising edge of CTTL at the beginning of T2, forcing its carry output low. CO drives /CWAIT whenever the PPI is selected. The /LADS latch is cleared by /TSO at the beginning of T2, allowing the counter to begin counting CTTL rising edges in T3. CO is asserted during the final wait state, allowing the cycle to terminate.



TL/EE/10465–12

FIGURE 12. Parallel Port Access Wait State Timing

APPENDIX A

PAL EQUATIONS

This appendix contains the logic equations for the address decode (D3) and data buffer enable control (D4) PALs. The BPU interface's MASK and CONTROL PALs are covered in the NS32CG16 Graphics Applications Note #2, ''Simple Embedded Control NS32CG16 System''.

```
PAL20L8B
ADDRESS DECODE PAL
ADDRESS DECODER FOR RT BOARD
NATIONAL SEMICONDUCTOR, SANTA CLARA, CA

A23 A22 A21 A20 A19 A18 A17 A16 A15 A8 R2 GND
R1 R0 IOSEL1 IOSEL0 ICU USER1 USER0 DRAM SRAM ROM OVER VCC

/ROM = A23*A22*A21*A20*/A19*/A18*/A17
       + /A23*OVER

/SRAM = /A23*/R2*/R1*/R0*/OVER
        + /A23*/A22*/A21*/A20*A19*/R2*/R1*R0*/OVER
        + /A23*/A22*/A21*A20*/R2*R1*/R0*/OVER
        + /A23*/A22*A21*/A20*/R2*R1*R0*/OVER
        + /A23*A22*/A21*/A20*R2*/R1*/R0*/OVER

/DRAM = /A23*/A22*/A21*/A20*/A19*/R2*/R1*R0*/OVER
        + /A23*/A22*/A21*/A20*/R2*R1*/R0*/OVER
        + /A23*/A22*/A21*/R2*R1*R0*/OVER
        + /A23*/A22*R2*/R1*/R0*/OVER

/USER0 = A23*/A22

/USER1 = A23*A22*/A21

/ICU = A23*A22*A21*A20*A19*A18*A17*A16*/A8

/IOSEL0 = A23*A22*A21*A20*A19*A18*A17*/A16*/A15

IOSEL1 = A23*A22*A21*A20*A19*A18*A17*/A16*A15


PAL16L8A
DECODE PAL
DATA BUFFER ENABLE DECODER FOR RT BOARD
NATIONAL SEMICONDUCTOR, SANTA CLARA, CA

NC ROM SRAM DRAM USER0 USER1 SYNCBPUCYC B_ENBPU HLDA GND
NC NC NC NC USERBLT BRDBLT USER BRD BED VCC

/BED = SRAM*DRAM*USER0*USER1*BDEN*SYNCBPUCYC

/BRD =        /SRAM*SYNCBPUCYC
            + /DRAM*SYNCBPUCYC

/USER =       /USER1*SYNCBPUCYC
            + /USER0*SYNCBPUCYC

/BRDBLT =     /ROM*B_ENBPU*/SYNCBPUCYC

/USERBLT =    /SRAM*B_ENBPU*/SYNCBPUCYC
            + /DRAM*B_ENBPU*BDEN*/SYNCBPUCYC
```

APPENDIX B

USER CONFIGURABLE OPTIONS

The following is a summary of the RT Board's jumpers organized in numerical sequence.

Jumper W1.1: Configures number of wait states inserted during ROM, USER0 and USER1 accesses. The ''X'' denotes that the jumper is installed.

TABLE B.1. ROM, USER0 and USER1 Wait State Options

Address Space	Waits	Jumpers				
		1, 2	3, 4	5, 6	7, 8	9, 10
ROM	0	—	—	—	—	—
	1	X	—	—	—	—
	2	—	X	—	—	—
	3	X	X	—	—	—
USER0	0	—	—	—	—	—
	1	—	—	X	—	—
	2	—	—	—	X	—
	3	—	—	X	X	—
USER1	0	—	—	—	—	—
	1	—	—	—	—	X

Jumpers W1.2, W1.3, W1.4: Maps the SRAM into a contiguous address space above DRAM. Install jumpers as indicated according to amount of DRAM installed on board. See also jumper W3.1.

TABLE B.2. SRAM Address Mapping

DRAM	Jumpers			SRAM Address Range
	W1.2	W1.3	W1.4	
0	(1, 2)	(1, 2)	(1, 2)	0–00FFFF
512k	(1, 2)	(1, 2)	(2, 3)	080000–08FFFF
1MEG	(1, 2)	(2, 3)	(1, 2)	100000–10FFFF
2MEG	(1, 2)	(2, 3)	(2, 3)	200000–20FFFF
4MEG	(2, 3)	(1, 2)	(1, 2)	400000–40FFFF

Jumper W3.1: Configures the bank select lines for the DRAM array. Configured according to the amount of DRAM installed on the board. The ''X'' denotes that the jumper is installed. See also jumpers W1.2, W1.3 and W1.4.

TABLE B.3. DRAM Bank Select Options

DRAM	Jumpers	
	1, 2	2, 3
0	—	—
512k	—	X
1MEG	—	X
2MEG	X	—
4MEG	X	—

TABLE B.4. Serial Port Configurations

DCE	DTE
(1, 3)	(1, 2)
(2, 4)	(3, 4)
(5, 7)	(5, 6)
(6, 8)	(7, 8)
(9, 10)	(9, 10)
(11, 12)	(11, 12)

Jumper W3.2: Configures the DRAM controller's DELCLK input. Install 1, 2 to drive DELCLK with CTTL (power-saver mode not used). Install 2, 3 to drive with a separate clock (power-saver mode used).

Jumper W3.3: Causes one additional wait state to be inserted during a DRAM access when installed 1, 2, otherwise should be installed 2, 3. Also see Configuration Switch.

Jumper W4.1: Configures the board for the amount of EPROM installed. Install 2, 3 for 64k using type 27256 EPROMs. Install 1, 2 for 128k using type 27512 EPROMs.

Jumper W4.2: Configures the ICU's CLK input. Install (1, 2) if CTTL $\leq$10 MHz. Install (2, 3) if CTTL $>$10 MHz.

Jumpers W6.1, W6.2, W6.3, W6.4: Configures the parallel port. Install jumpers W6.1 (2, 3), W6.2, W6.3 (1, 2) and W6.4 (1, 2) to configure as a printer port compatible with earlier versions of software.

Jumpers W6.5, W6.6: Configures PORT1 and PORT2 serial ports, respectively. Install jumpers as indicated to configure as DCE or DTE.

An eight position dipswitch is used to configure the serial ports' baud rate, the DRAM controller, and to select the monitor to be run. The ''X'' denotes that the switch is turned on.

TABLE B.5. Baud Rate Selection

Switch Setting				Baud Rate
S4	S3	S2	S1	
X	X	X	X	19200
X	X	X	—	9600
X	X	—	X	7200
X	X	—	—	4800
X	—	X	X	3600
X	—	X	—	2400
X	—	—	X	2000
X	—	—	—	1800
—	X	X	X	1200
—	X	X	—	600
—	X	—	X	300
—	X	—	—	150
—	—	X	X	134
—	—	X	—	110
—	—	—	X	75
—	—	—	—	50

TABLE B.6. Monitor Selection

Switch Setting		Monitor
SW6	SW5	
—	—	Reserved
—	X	Reserved
X	—	MONCG
X	X	RAMless

TABLE B.7. DRAM Wait State Options

Switch Setting		Frequency/Wait States
SW8	SW7	
—	—	15 MHz, 0 Waits
—	X	15 MHz, 1 Wait
X	—	10 MHz, 0 Waits
X	X	10 MHz, 1 Wait

APPENDIX C

MEMORY MAP AND DEVICE REGISTER ADDRESSES

TABLE C.1. RT Board Memory Map

Address Range (HEX)	Description
0–40FFFF	RAM (or ROM if OVER is set)
410000–7FFFFF	Reserved
800000–BFFFFF	USER0
C00000–DFFFFF	USER1
E00000–EFFFFF	Reserved
F00000–F1FFFF	ROM
F20000–FDFFFF	Reserved
FE0000–FE7FFF	IOSEL0
FE8000–FEFFFF	IOSEL1
FF0000–FF0060	BPU Interface Registers
FF0061–FFFDFF	Reserved
FFFE00–FFFFFF	ICU

TABLE C.2. ICU Address Assignments

Register Number and Address in HEX		Register Name
	R0(00)	HVCT
	R1(04)	SVCT
R3(0C)	R2(08)	ELTG
R5(14)	R4(10)	TPL
R7(1C)	R6(18)	IPND
R9(24)	R8(20)	ISRV
R11(2C)	R10(28)	IMSK
R13(34)	R12(30)	CSRC
R15(3C)	R14(38)	FPRT
	R16(40)	MCTL
	R17(44)	OCASN
	R18(48)	CIPTR
	R19(4C)	PDAT
	R20(50)	IPS
	R21(54)	PDIR
	R22(58)	CCTL
	R23(5C)	CICTL
R25(64)	R24(60)	LCSV
R27(6C)	R26(68)	HCSV
R29(74)	R28(70)	LCCV
R31(7C)	R30(78)	HCCV

Register Address = FFFEXX; XX denoted in parenthesis beside register number

TABLE C.3. DUART Address Assignments

Register Address		Register Name
PORT1	PORT2	
FE0080	FE0000	RBR, THR, DLL
FE0084	FE0004	IER, DLM
FE0088	FE0009	IRR, FCR, AFR
FE008C	FE000C	LCR
FE0090	FE0010	MCR
FE0094	FE0014	LSR
FE0098	FE0018	MSR
FE009C	FE001C	SCR

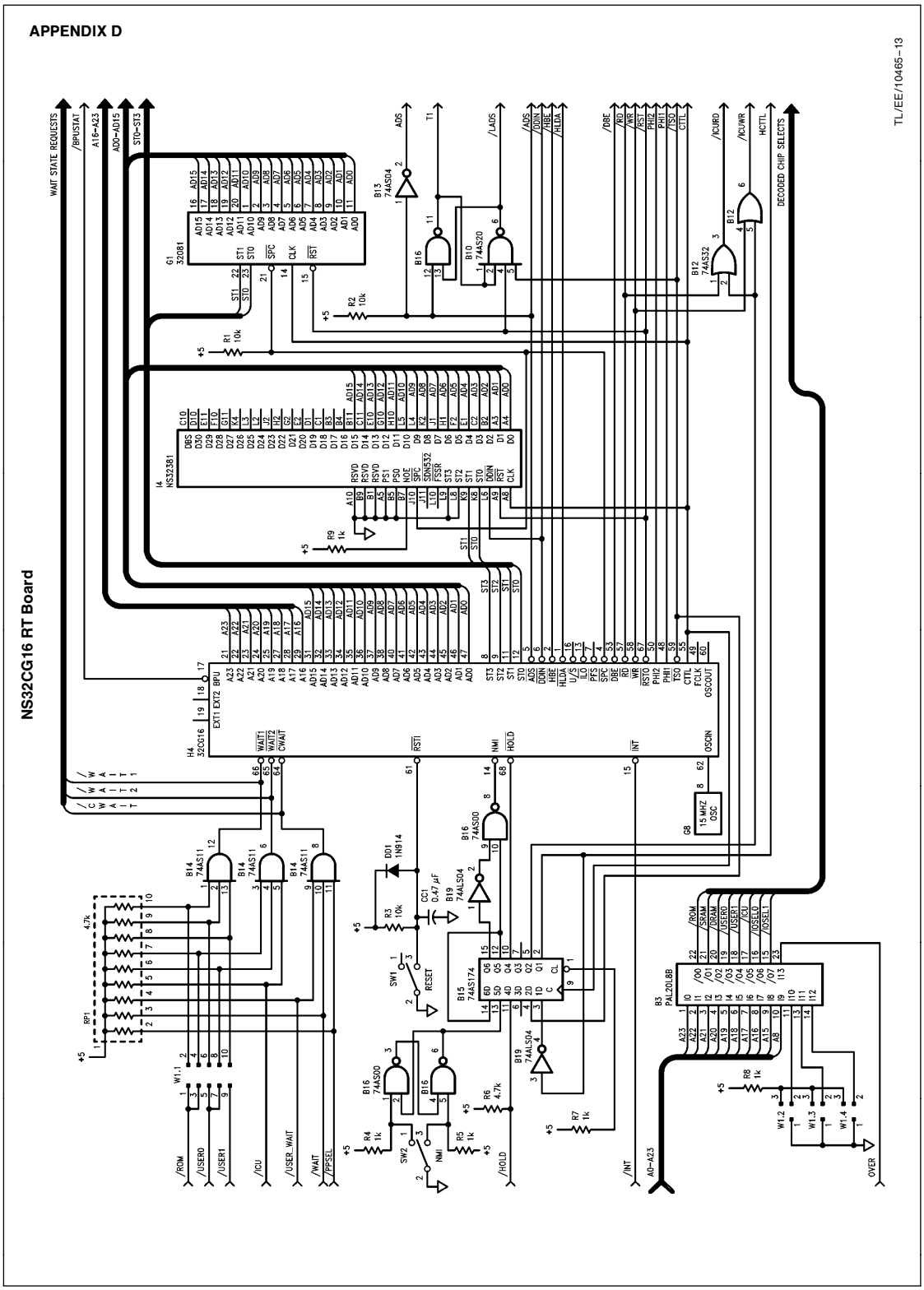
TABLE C.4. Parallel Port Address Assignments

Register Address	Register Name
FE8001	PORTA
FE8005	PORTB
FE8009	PORTC
FE800D	CONTROL

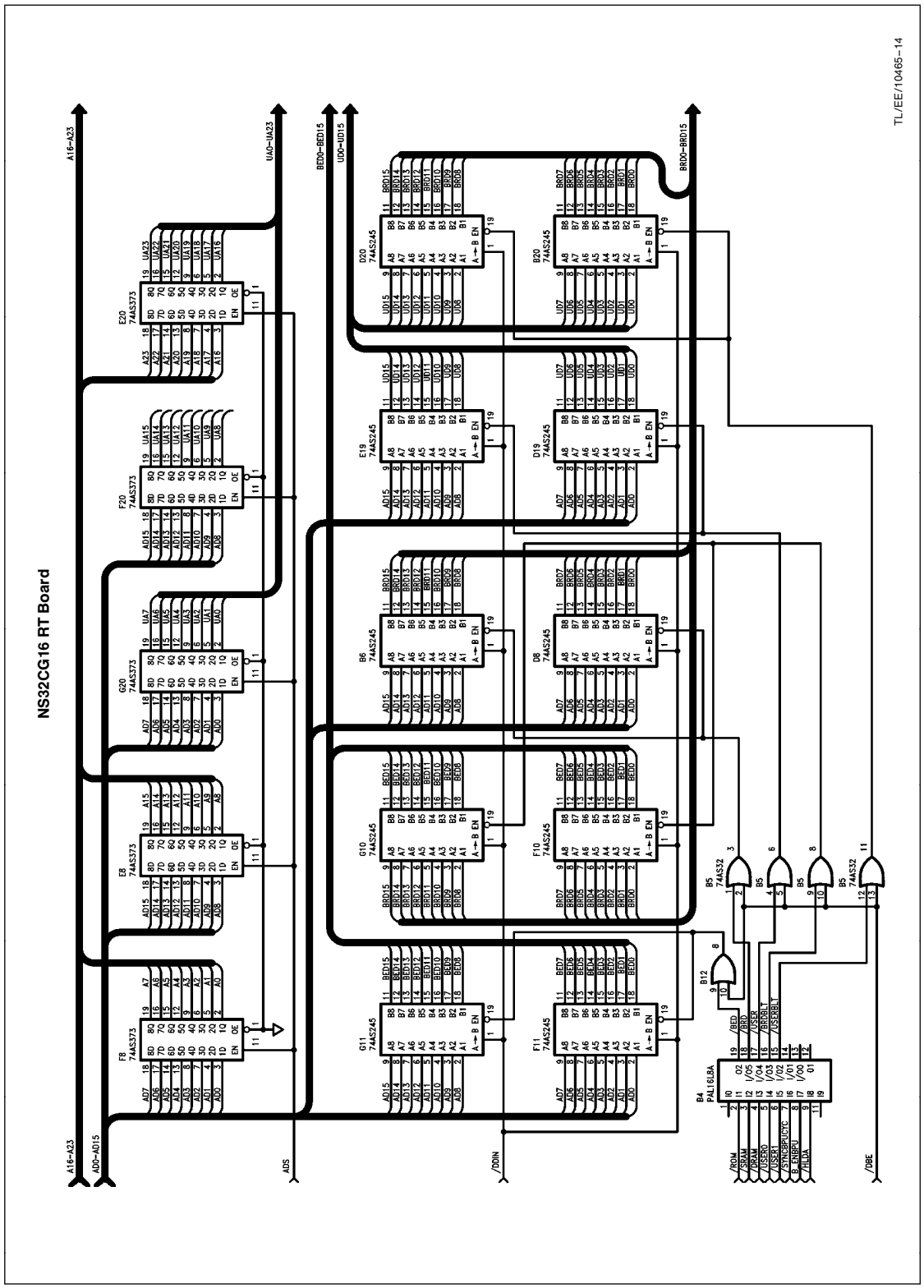
TABLE C.5. BPU Interface Register Address Assignments

Register Address	Register Name
FF0020	CONTROL
FF0040	MASK
FF0060	COUNT

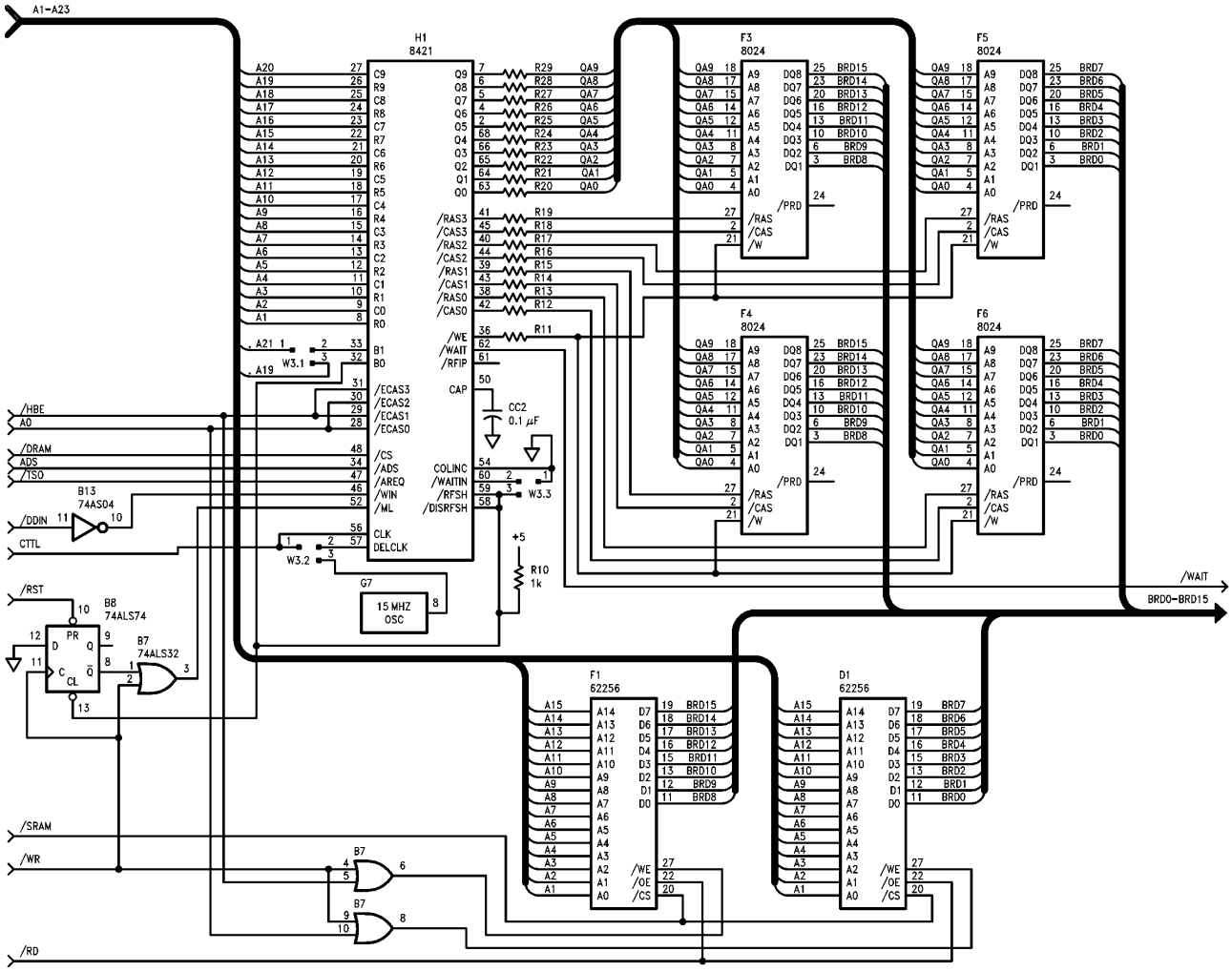
Configuration Switch Address: FE8081



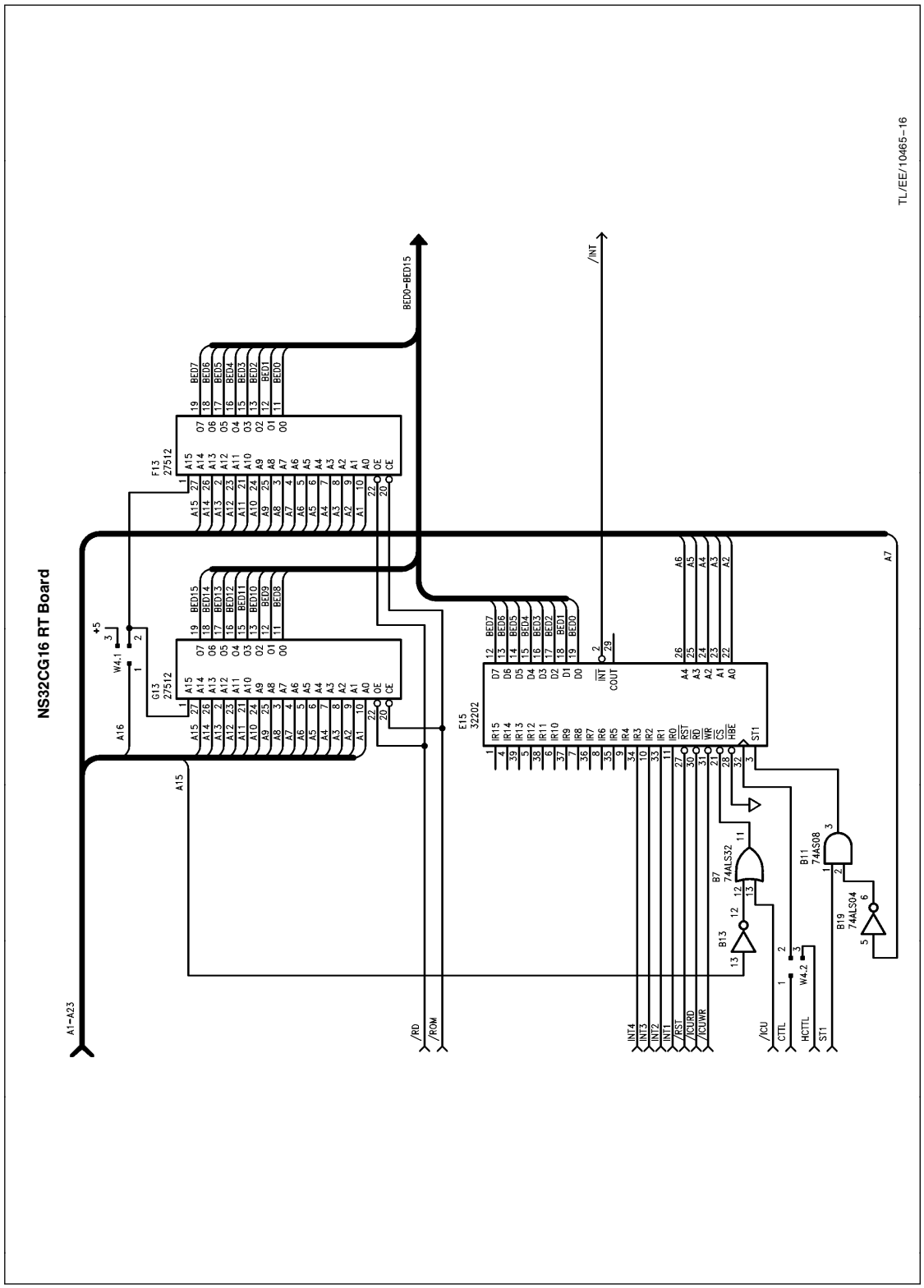
TL/EE/10465–13


NS32CG16 RT Board
A16–A23
AD0–AD15
UA0–UA23
BED0–BED15
UD0–UD15
BRD0–BRD15
ADS
/DDIN
/DBE
F8 74AS373
E8 74AS373
F20 74AS373
G20 74AS373
E20 74AS373
G11 74AS245
G10 74AS245
B6 74AS245
E19 74AS245
D20 74AS245
F11 74AS245
F10 74AS245
D6 74AS245
D19 74AS245
B20 74AS245
B12
B5 74AS32
B4 PAL16L8A
/ROM
/SRAM
/DRAM
/USER0
/USER1
/SYNCBPUCTC
B_ENBPU
/HLDA
/BED
/BRD
/USER
/BRDBLT
/USERBLT
TL/EE/10465–14

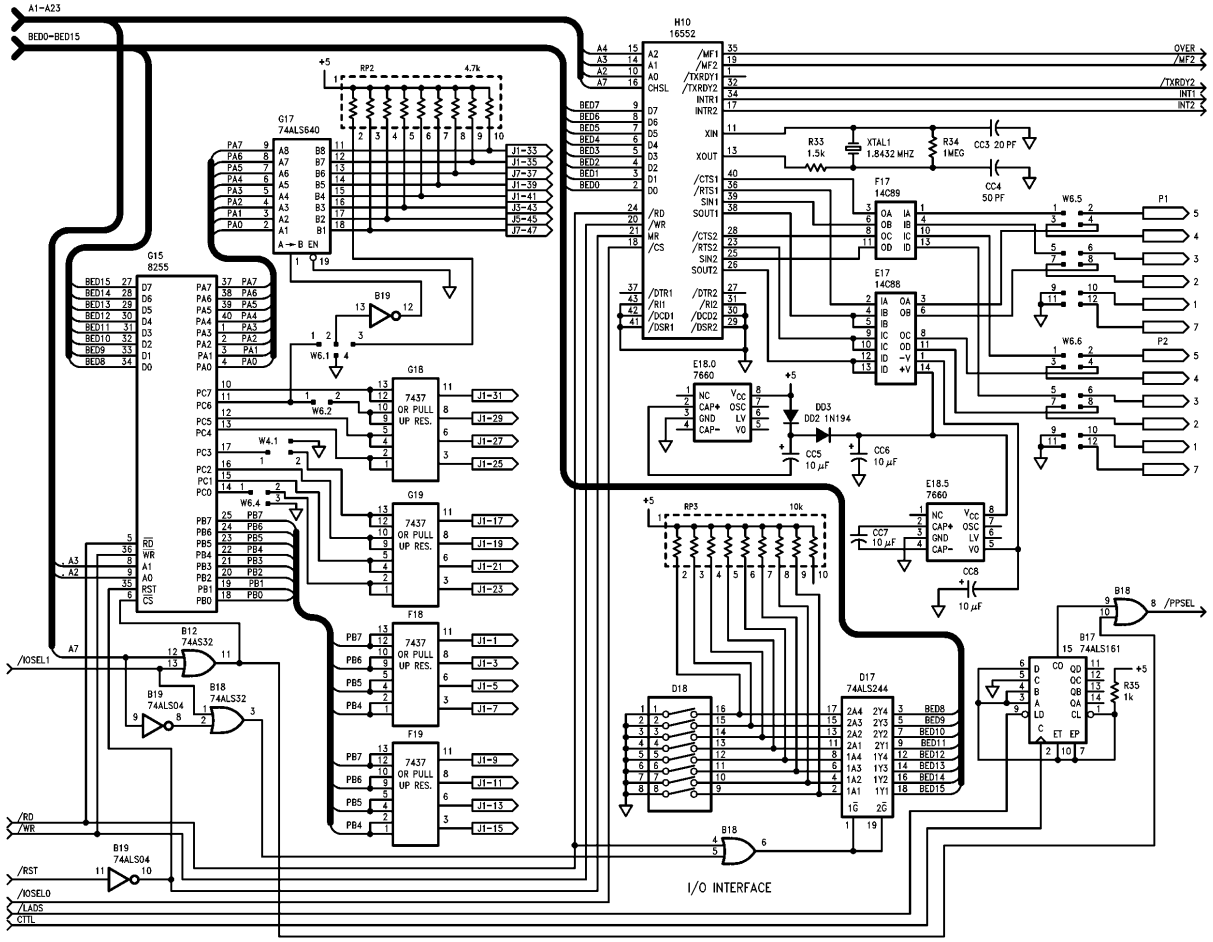
NS32CG16 RT Board



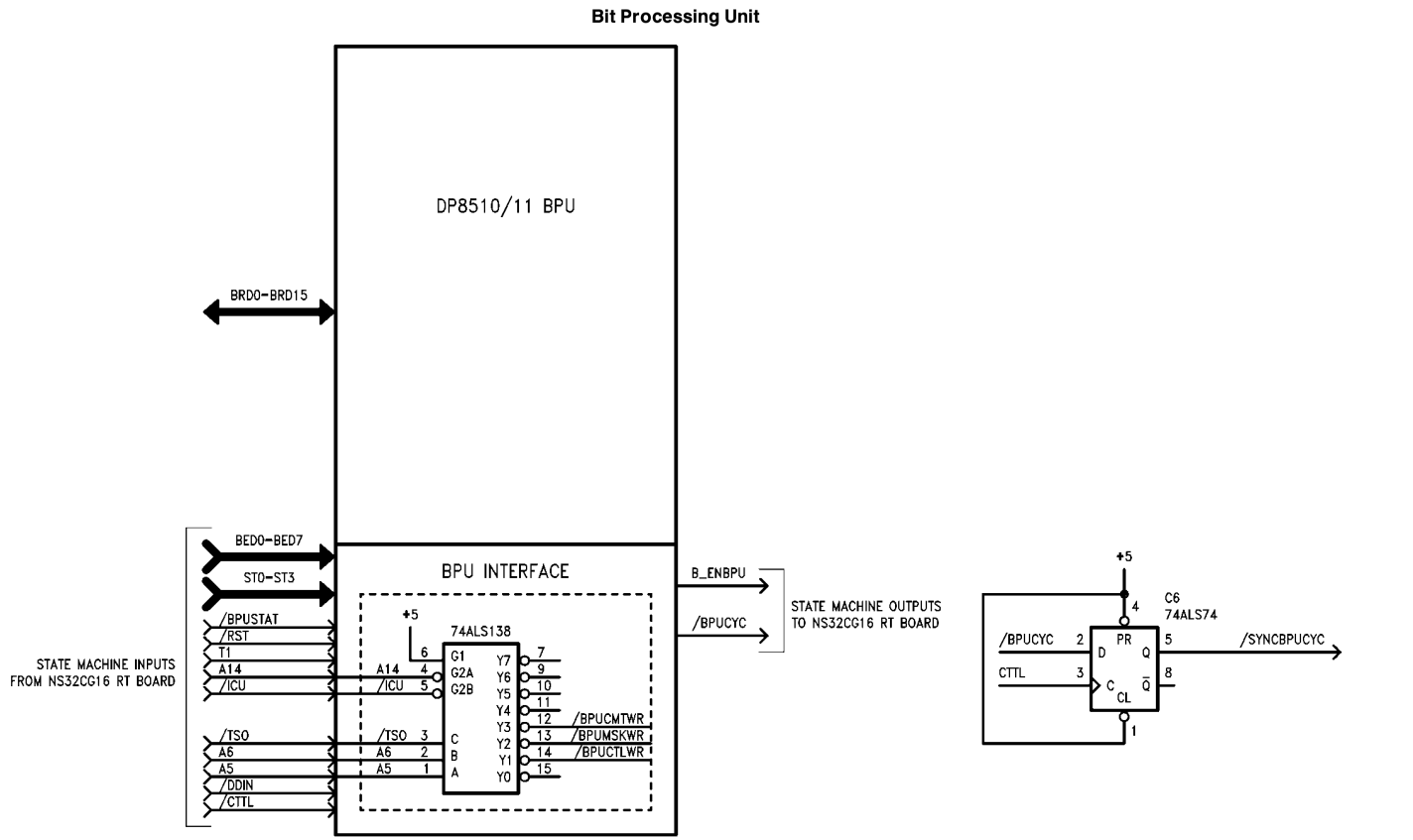
TL/EE/10465–15


NS32CG16 RT Board
A1-A23
A15
A16
A7
G13
27512
F13
27512
E15
32202
W4.1
W4.2
+5
BED0-BED15
/INT
/RD
/ROM
INT4
INT3
INT2
INT1
/RST
/ICURD
/ICUWR
/ICU
CTTL
HCTTL
ST1
B7
74ALS32
B11
74AS08
B13
B19
74ALS04
TL/EE/10465-16

NS32CG16 RT Board



TL/EE/10465–17



TL/EE/10465–18

Lit. # 100632

LIFE SUPPORT POLICY

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

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

National Semiconductor Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel: 1(800) 272-9959
TWX: (910) 339-9240

National Semiconductor GmbH
Livry-Gargan-Str. 10
D-82256 Fürstenfeldbruck
Germany
Tel: (81-41) 35-0
Telex: 527649
Fax: (81-41) 35-1

National Semiconductor Japan Ltd.
Sumitomo Chemical
Engineering Center
Bldg. 7F
1-7-1, Nakase, Mihama-Ku
Chiba-City,
Ciba Prefecture 261
Tel: (043) 299-2300
Fax: (043) 299-2500

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

National Semiconductores Do Brazil Ltda.
Rue Deputado Lacorda Franco
120-3A
Sao Paulo-SP
Brazil 05418-000
Tel: (55-11) 212-5066
Telex: 391-1131931 NSBR BR
Fax: (55-11) 212-1181

National Semiconductor (Australia) Pty, Ltd.
Building 16
Business Park Drive
Monash Business Park
Nottinghill, Melbourne
Victoria 3168 Australia
Tel: (3) 558-9999
Fax: (3) 558-9998

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and National reserves the right at any time without notice to change said circuitry and specifications.