Publication Number 4200078A

Integrated Microprocessor
IPC-16

PACE Technical Description

June 1975

PREFACE

This technical description defines and explains the utilization of the entire complement of hardware and software items that comprise and support the *single-chip* PACE (Processing and Control Element). PACE is a full-feature Control Processing Unit that is manufactured by the National Semiconductor Corporation.

The information contained herein is up-to-date at the time of publication but is subject to change without notice. Therefore, it is suggested that the National Semiconductor local sales office be contacted for the latest information pertaining to PACE.

CONTENTS

CONTENTS (Continued)

LIST OF TABLES

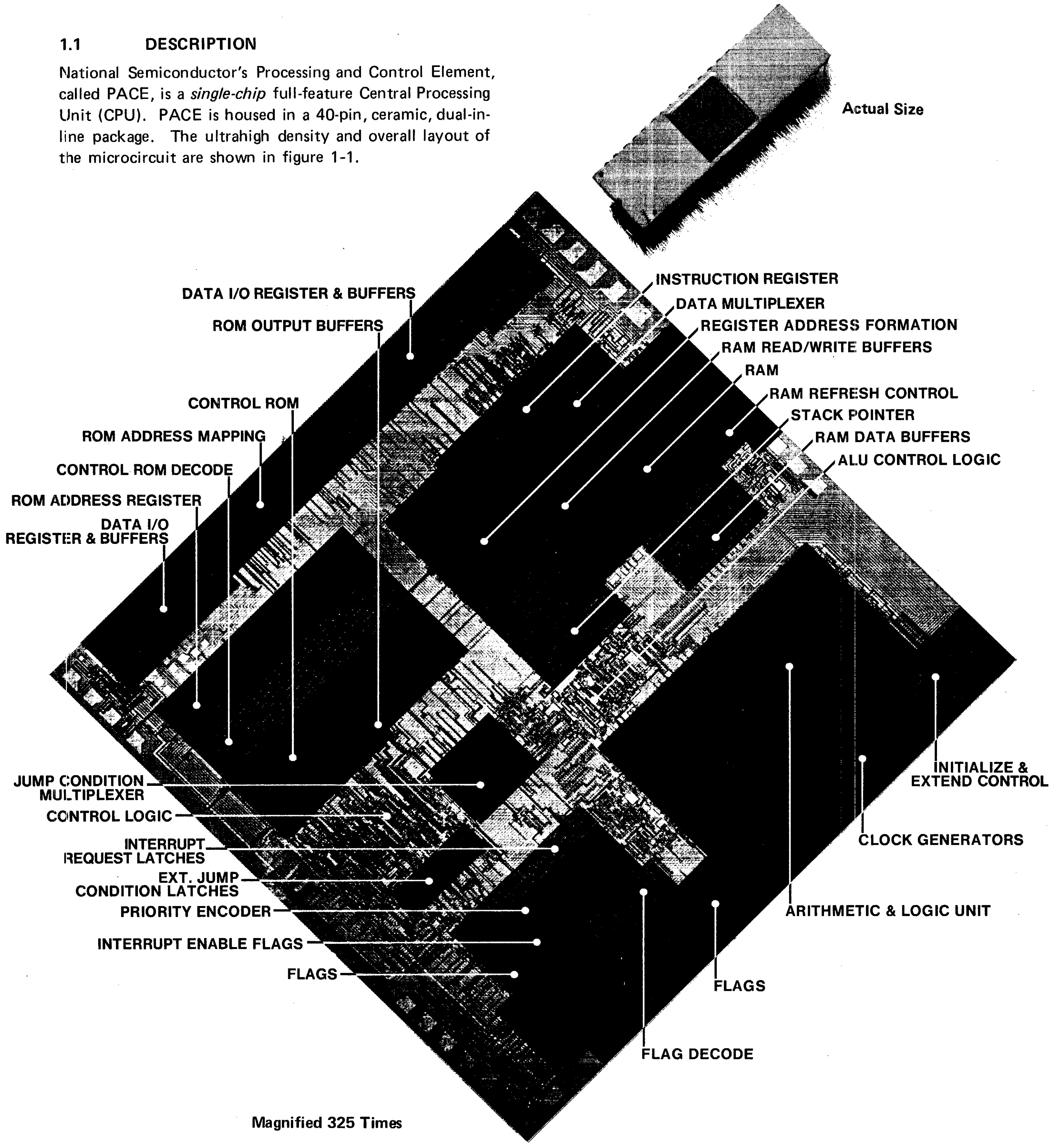
LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Continued)

CHAPTER 1
INTRODUCTION TO PACE

1.1 DESCRIPTION

National Semiconductor's Processing and Control Element, called PACE, is a *single-chip* full-feature Central Processing Unit (CPU). PACE is housed in a 40-pin, ceramic, dual-in-line package. The ultrahigh density and overall layout of the microcircuit are shown in figure 1-1.



NS10351

Figure 1-1. PACE Chip and Circuit Layout

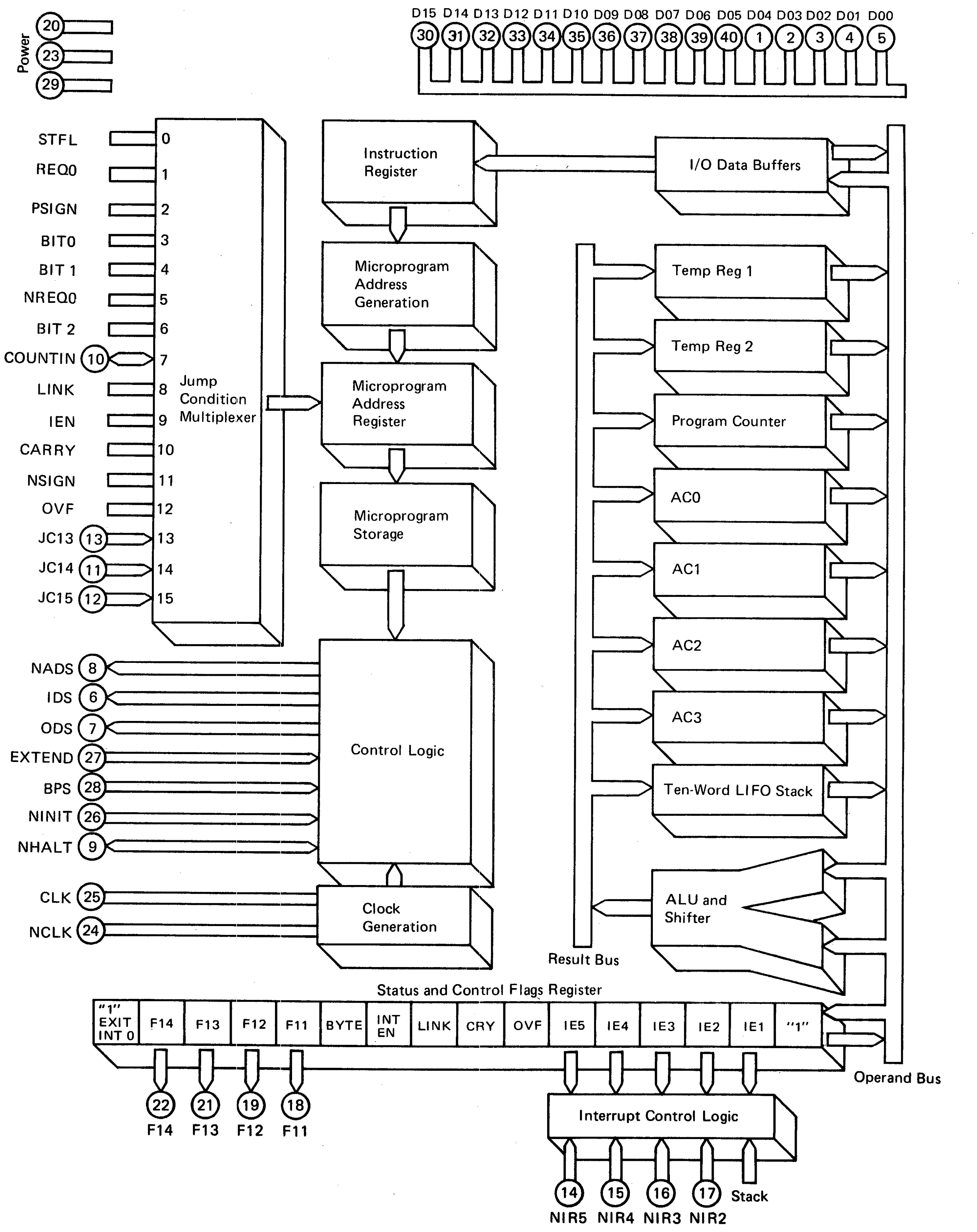


Figure 1-2. PACE Functional Block Diagram

PACE also is called a microprocessor, the prefix *micro* relating to the microscopic size of the physical circuit and components on the chip. An extraordinary amount of data-processing capability is provided in one component by the *single-chip* microconstruction.

Figure 1-2 reveals the CPU architecture and pinouts of the *single-chip* PACE — consisting of registers, control logic, an arithmetic unit, and the data buses. Some of the outstanding operational features of the PACE microprocessor are listed below.

Features

- 16-bit instruction word offers addressing flexibility and speed.
- 8- or 16-bit data word interfaces increase application flexibility.
- 45 instruction types provide efficient programming.
- Common memory and peripheral addressing means powerful I/O instructions.
- Shares instructions with National Semiconductor's IMP-16, allowing software compatibility.
- Four general-purpose accumulators reduce memory data transfers.
- 10-word Stack is utilized for interrupt processing/data storage.
- Six vectored priority-interrupt levels speed interrupt service and simplify hardware.
- Programmer-accessible status register may be preserved, tested, or modified.
- Typical 10-microsecond instruction execution guarantees high throughput.
- 1K-by-16 Read-Only Memory allows single-memory package systems.
- Single-phase true and complement clock minimizes external components.
- +5-volt and −12-volt standard supplies ensure minimum cost.

The PACE MOS/LSI chip is produced using silicon-gate, P-channel enhancement-mode standard-process technology. This means that the following very significant advantages are realized.

- Lower cost per function
- Lower component count
- Simplified design
- Higher reliability
- High noise immunity
- Low threshold voltage

Among some of the benefits of a single-chip device with the above-listed advantages are the following.

- LOWER COMPONENT COUNT --- Generally, this means lower procurement, incoming testing, inventory, handling, rework, and assembly cost — and higher reliability.
- SIMPLIFIED DESIGN --- LSI devices enable engineering design groups to take advantage of prepackaged circuits that are self-contained and perform a unified function. Also, a design group not strongly oriented towards digital design may make use of the latest techniques and devices without requiring expertise related to the design methods of interfacing the circuits internal to LSI devices. In summary, use of LSI devices requires considerably less engineering time to develop a product.
- HIGHER RELIABILITY --- The long history of field maintenance of all types of electronic systems clearly demonstrates the high reliability of LSI devices. System maintenance has shown that the reliability of low-power circuits is inversely proportional to the number of component lead connections in the system. This factor, coupled with the abundant functional capability of LSI, greatly increases the probability that an LSI-based system will function properly over extended periods of time.
- IMPROVED PERFORMANCE --- PACE offers higher throughput because of a powerful instruction set, a proven architecture, and 16-bit address generation and data handling.
- LOWER COST --- The reduction of cost is an aggregate savings resulting from the other advantages already enumerated. In the microprocessor field, the inherent functional superiority of high-density devices is seldom questioned. The superior performance of the single-chip PACE, coupled with reduced engineering and assembly cost, higher reliability, lower operating and maintenance costs, and smaller size of the microprocessor, definitely results in a much better performance-to-price ratio than heretofore possible. It makes the PACE microprocessor the most competitive processor on the market.

1.2 OUTSTANDING FEATURES OF PACE

The outstanding features of PACE are described in detail later. Nevertheless, to provide an overall view of the many favorable facets of PACE, these features are listed and briefly described below.

- 8- OR 16-BIT DATA HANDLING --- PACE is cost effective in applications dominated by 8-bit data interfaces. Efficient coding and address generation found only in 16-bit microprocessors are extended to 8-bit applications.
- INCREASED THROUGHPUT --- PACE minimizes data and program storage requirements, while increasing data-processing throughput.
- COMPATIBILITY WITH NATIONAL SEMICONDUCTOR'S IMP-16 MICROPROCESSORS --- May use Source Statement Translator to convert IMP-16 software to PACE software without incurring significant development costs. The IMP-16 and PACE may be used together in a digital hybrid system, allowing common software and peripheral interfaces.

- PACE BLUE CHIPS --- Support CPU. Include System Timing Element and 8-bit Bidirectional Transceiver Element.
- PACE GREEN CHIPS --- Peripheral interface chips, including Interface Latch Elements and Address Latch Elements. Provide storage and buffering between the PACE system TTL bus and user peripherals or memory. Available in 8-bit and 16-bit units, accommodating PACE's unique ability to operate on both 8-bit and 16-bit data.
- PACE MEMORY CHIPS --- Designed for optional use with PACE. Include 256-by-4-bit Random Access Memory (having on-chip address latches), 1024-by-16-bit Read-Only Memory (having on-chip address latches), and 512-by-8-bit erasable Programmable Read-Only Memory.
- PACE APPLICATION CARDS --- Intended for prototyping, preproduction and small production runs. Prefabricated and pretested CPU, Random Access Memory, Read-Only Memory/Programmable Read-Only Memory, input/output, and special functions are provided.
- PACE DEVELOPMENT SYSTEM --- May be used to evaluate the microprocessor and to develop and debug a myriad of application hardware and software. Includes a Control Panel and interfaces for Teletypewriter, Card Reader, and Tape Reader. Available options include Line Printer, dual-drive Floppy Disc, and Programmable Read-Only Memory Programmer.
- FIELD SUPPORT --- Microprocessor specialists (engineers) that understand hardware, software, and PACE applications. Available domestically and internationally and offer on-site technical assistance.
- BACK-UP APPLICATION SUPPORT --- National Semiconductor's home-base technical-support specialists support the microprocessor specialist in the field and, as required, provide direct support to users.
- FACTORY SERVICE --- Will repair any of our microprocessor products. This service is for both the OEM customer and the end user.
- TRAINING --- Offers elementary to advanced training, with locations in the West, Midwest, and East. Courses are in-depth and include hands-on instruction.
- USER GROUP --- Membership open to users and others interested in microprocessors. Provides a vehicle of communications between members and with National Semiconductor. Makes programs available from its User Group Software Library.
- SOFTWARE SUPPORT --- Includes Source Statement Editor, Assemblers, Loaders, Debug Routine, Utilities, and Diagnostics. (Also, as previously mentioned, a Source Statement Translator converts IMP-16 software to PACE software.)
- DOCUMENTATION --- Offers a *Product Description* that helps you to determine the suitability of a microprocessor to a particular application. A *Technical Description* gives a more in-depth understanding such that benchmarks may be established, programs written, and systems designed. A *PACE Users Manual* describes the use of PACE equipment and software.

1.3 PACE APPLICATIONS

Applications for PACE could very well be in the thousands. The suitability of PACE will, in many cases, be a matter of evaluation by potential users for their particular needs. A few applications are listed.

- Test system and instrument control
- Process controllers
- Machine tool control
- Terminal control
- Small business machines
- Traffic controllers
- Word-processing systems
- Peripheral device controllers
- Educational controllers
- Sophisticated games
- Distributed and multiprocessor systems
- Automotive controller

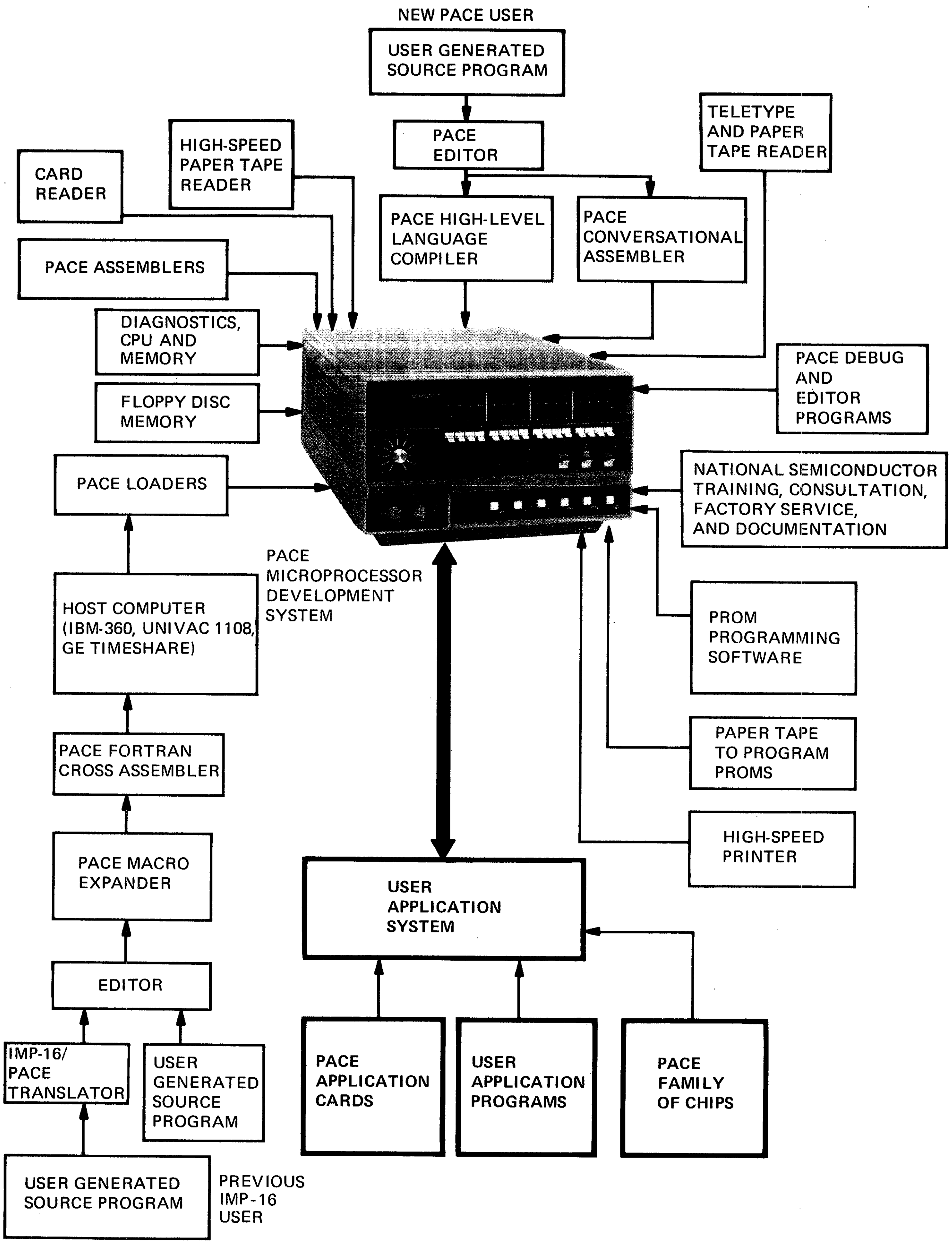
1.4 DYNAMICS OF PACE DEVELOPMENT

Developing a user's application system around the PACE chip can best be performed by utilizing the software and hardware support items provided by the PACE Microprocessor Development System. The *dynamics* of developing a user's application system are shown in figure 1-3. Obviously, *there's more to microprocessing than a microprocessor.*

Without this array of software and hardware support, a user would have to expend considerable manpower, time, and money to develop his application of a microprocessor-based system. But with National Semiconductor's total support development system, the user immediately can start work on his own application or system. In addition, the experience of National Semiconductor's staff engaged in design, applications, documentation, field support, back-up technical support, factory service, and training is readily available to help not only experienced designers but also newcomers throughout their development programs. *We do our best to make it easy to use PACE and our other microprocessor products.*

1.5 PACE FAMILY OF CHIPS

The PACE family of chips comprise three types: Blue, Green, and Memory devices. The chips facilitate system development and provide the main functional building blocks for interfacing, timing, and data/program storage. The chips are briefly described in the immediately following paragraphs, and their utilization in system applications is further described in chapter 2.


NEW PACE USER
USER GENERATED SOURCE PROGRAM
PACE EDITOR
TELETYPE AND PAPER TAPE READER
HIGH-SPEED PAPER TAPE READER
CARD READER
PACE HIGH-LEVEL LANGUAGE COMPILER
PACE CONVERSATIONAL ASSEMBLER
PACE ASSEMBLERS
DIAGNOSTICS, CPU AND MEMORY
FLOPPY DISC MEMORY
PACE DEBUG AND EDITOR PROGRAMS
PACE LOADERS
NATIONAL SEMICONDUCTOR TRAINING, CONSULTATION, FACTORY SERVICE, AND DOCUMENTATION
PACE MICROPROCESSOR DEVELOPMENT SYSTEM
HOST COMPUTER (IBM-360, UNIVAC 1108, GE TIMESHARE)
PROM PROGRAMMING SOFTWARE
PAPER TAPE TO PROGRAM PROMS
PACE FORTRAN CROSS ASSEMBLER
HIGH-SPEED PRINTER
PACE MACRO EXPANDER
USER APPLICATION SYSTEM
EDITOR
IMP-16/ PACE TRANSLATOR
USER GENERATED SOURCE PROGRAM
PACE APPLICATION CARDS
USER APPLICATION PROGRAMS
PACE FAMILY OF CHIPS
USER GENERATED SOURCE PROGRAM
PREVIOUS IMP-16 USER

NS10353

Figure 1-3. Dynamics of Utilization of PACE

1.5.1 PACE BLUE CHIPS

The PACE Blue Chips interface directly with the PACE microprocessor and include a System Timing Element (STE) and an 8-bit Bidirectional Transceiver Element (BTE). The STE requires application of +5 and -12 volts and the addition of only an external crystal (as shown in figure 1-4) to produce the MOS clock signals and substrate bias voltage required by the PACE microprocessor. In addition, the STE provides external TTL clock signals to accommodate user requirements.

NOTE

All signal names beginning with *N* or followed by an asterisk (✱) denote complemented signals that are asserted or activated by a logic '0'. Otherwise, signals are asserted by a logic '1'.

The BTE is an 8-bit transceiver that provides controlled translation of signals between the PACE microprocessor MOS buses and the system TTL buses. The BTE has a high-fanout TTL capability (up to 30 TTL loads). One BTE connected in a driver-only mode (WBD✱ control signal grounded) to the PACE microprocessor permits buffering for the System TTL Timing and Control Bus, thereby providing seven TTL control signals and flags for distribution as shown in figure 1-5. Two BTEs can be used to take advantage of the PACE capability to operate with a single time-multiplexed System TTL Address/Data Bus. Thus, system component count and interconnections are minimized. Time multiplexing of the System TTL Address/Data Bus for the Address, Data In, and Data Out Cycles is provided by PACE control signals that are routed over the System TTL Timing and Control Bus.

1.5.2 PACE GREEN CHIPS

The PACE Green Chips interface with the system TTL buses and memory or user peripherals. The PACE Green Chips include an Address Latch Element (ALE) and an Interface Latch Element (ILE). The Address and Interface Latch Elements are available in either 8-bit or 16-bit packages. The ALE provides memory address latching for single, time-multiplexed address/data bus systems using memory devices that do not have on-chip address latches. Figure 1-6 shows a portion of a typical system using a 16-bit ALE.

The ILE provides latched interfacing between the System TTL Address/Data Bus and the user peripherals. The user data and PACE data I/O controls can be connected to the ILE control inputs, as shown in figure 1-7, to provide either unidirectional or bidirectional peripheral data latching for either 8-bit or 16-bit peripheral data words. Timing and control signals from the PACE microprocessor are routed over the System TTL Timing and Control Bus to effect data input/output transfers between the ILE and the System TTL Address/Data Bus at the appropriate time. When data transfers are not effected, the ILE can be driven to a high-impedance state at the ILE-to-System TTL Address/Data Bus Interface and/or the ILE-to-peripheral Interface.

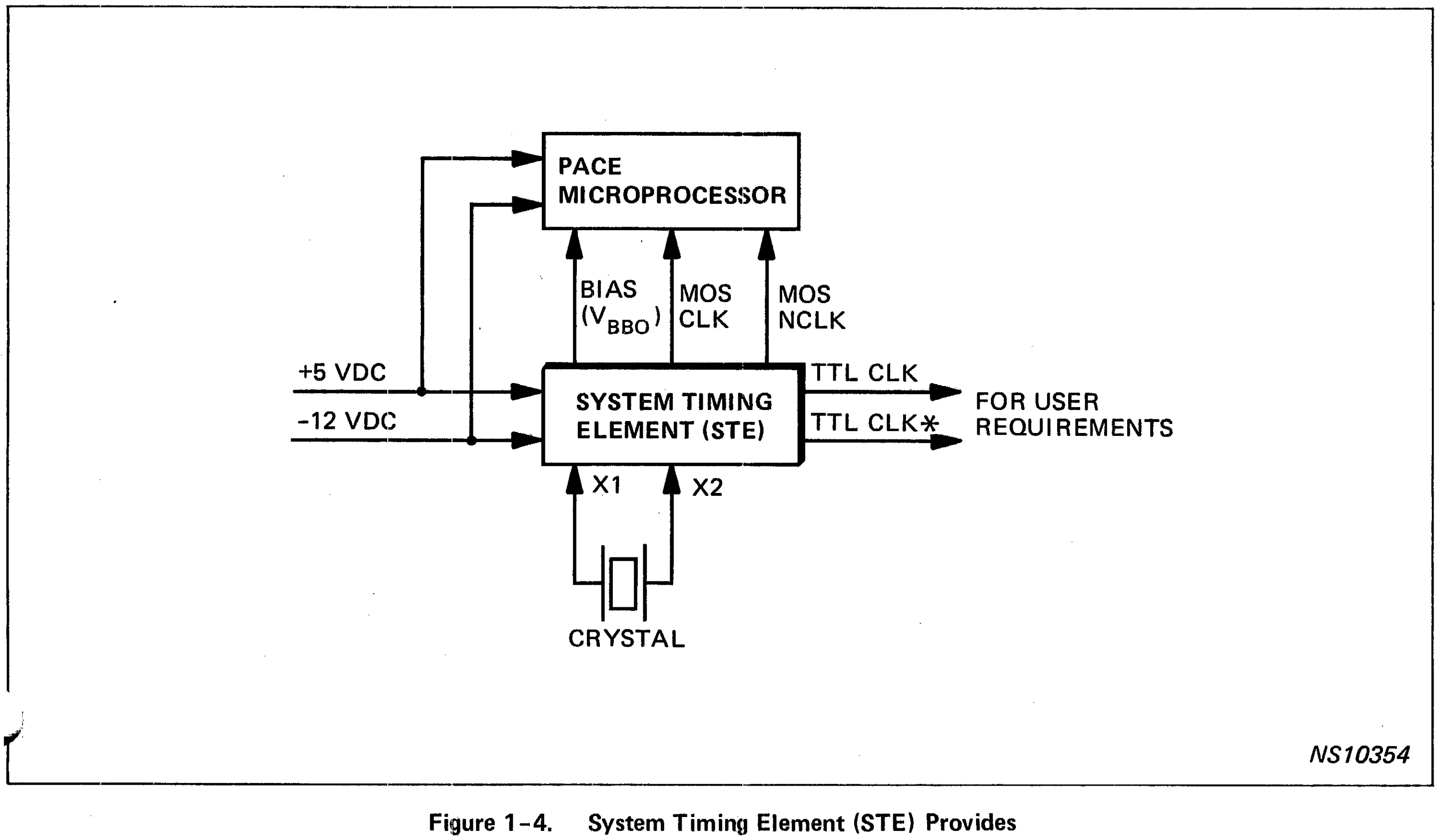


Figure 1-4. System Timing Element (STE) Provides All Timing Requirements

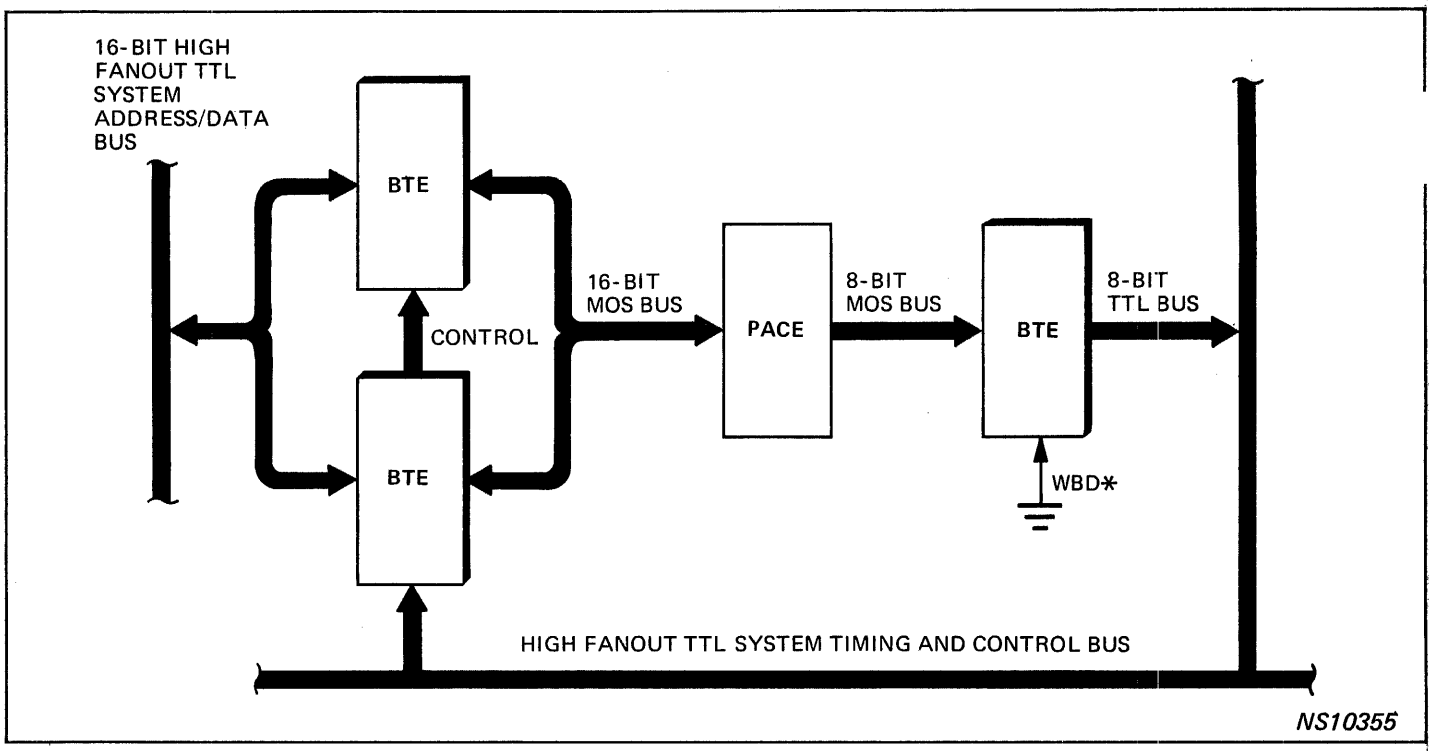

16-BIT HIGH FANOUT TTL SYSTEM ADDRESS/DATA BUS
BTE
BTE
CONTROL
16-BIT MOS BUS
PACE
8-BIT MOS BUS
BTE
8-BIT TTL BUS
WBD*
HIGH FANOUT TTL SYSTEM TIMING AND CONTROL BUS
NS10355

Figure 1-5. **Bidirectional Transceiver Element (BTE) Provides High-fanout TTL Buses**

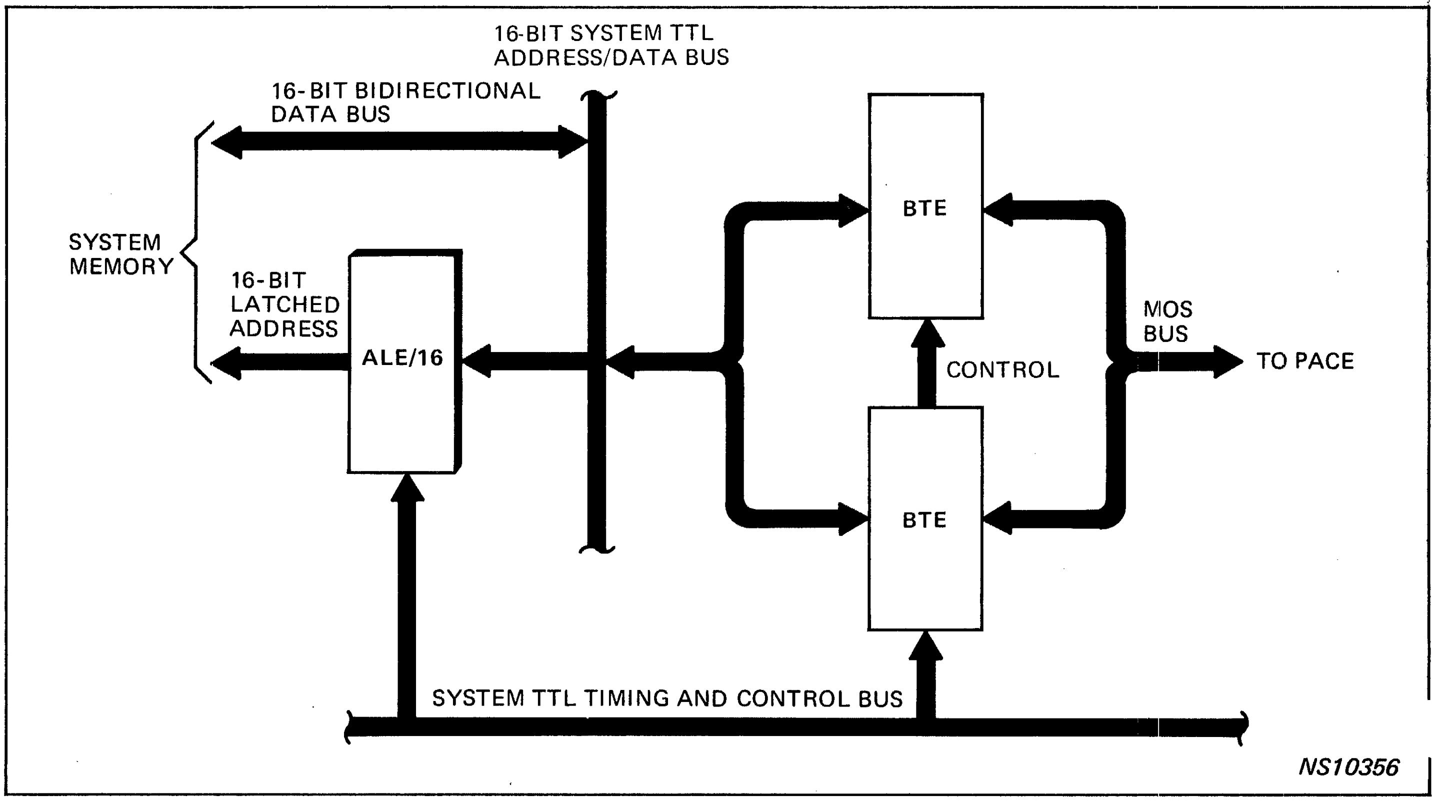

16-BIT SYSTEM TTL ADDRESS/DATA BUS
16-BIT BIDIRECTIONAL DATA BUS
SYSTEM MEMORY
16-BIT LATCHED ADDRESS
ALE/16
BTE
BTE
CONTROL
MOS BUS
TO PACE
SYSTEM TTL TIMING AND CONTROL BUS
NS10356

Figure 1-6. **Address Latch Element (ALE) Provides Memory Address Latching**

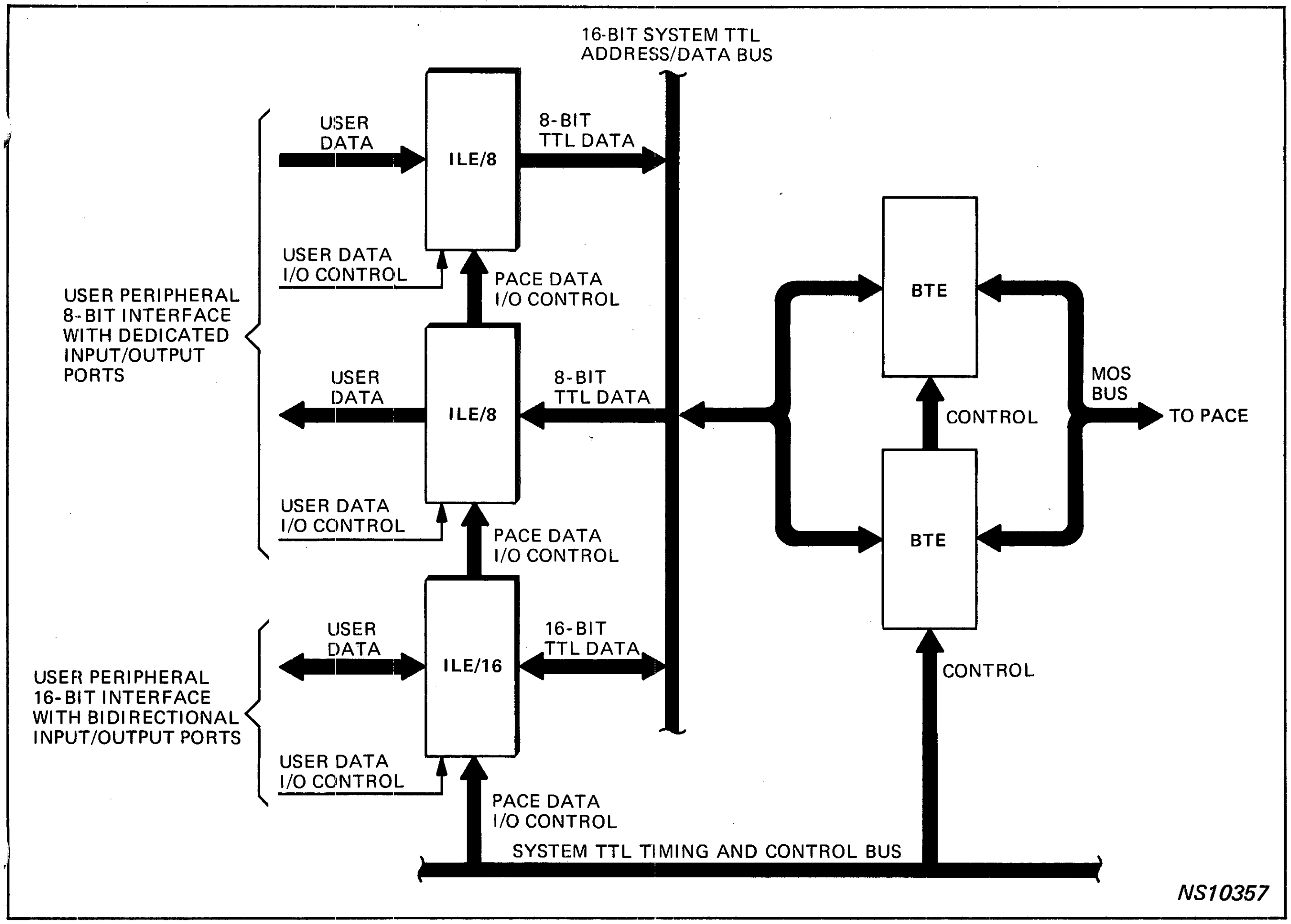


Figure 1-7. Interface Latch Element (ILE) Provides Interface Storage for Peripherals

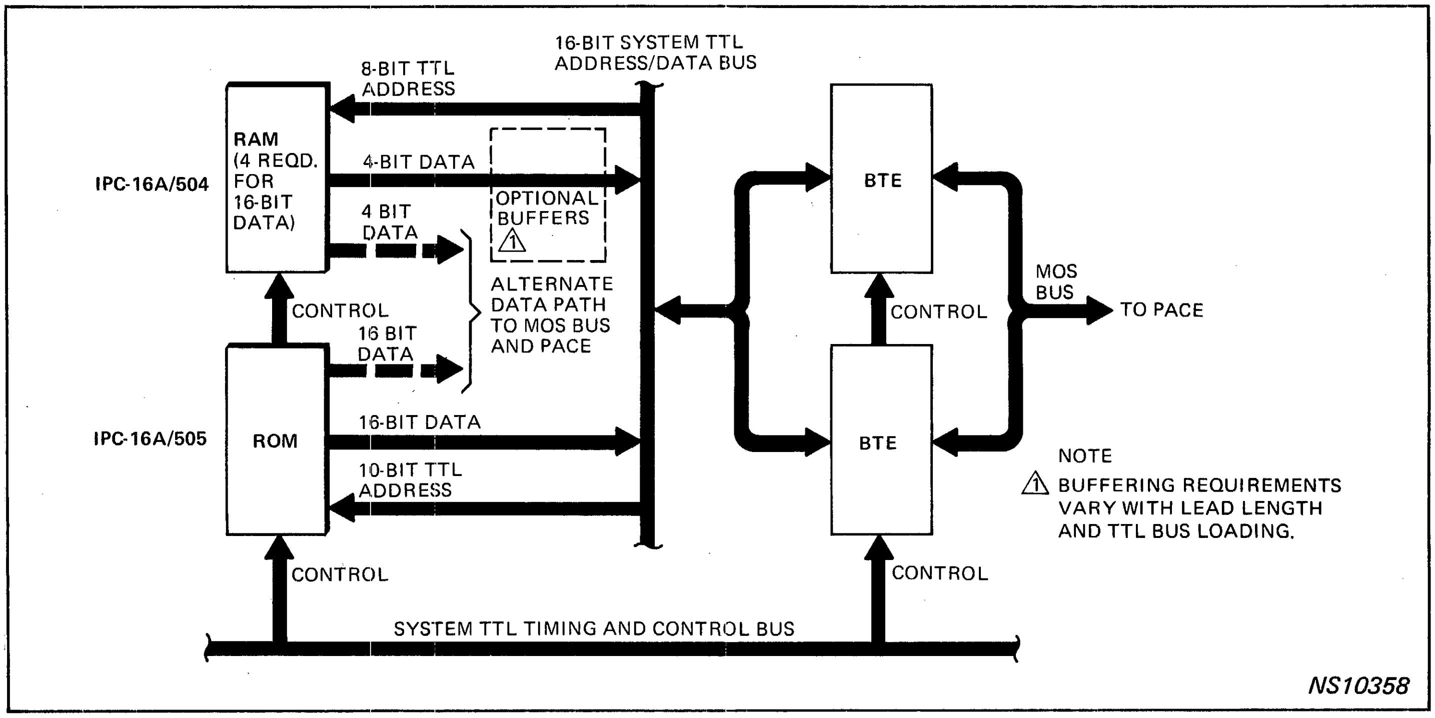

16-BIT SYSTEM TTL ADDRESS/DATA BUS
8-BIT TTL ADDRESS
RAM (4 REQD. FOR 16-BIT DATA)
IPC-16A/504
4-BIT DATA
OPTIONAL BUFFERS ⚠1
4 BIT DATA
BTE
ALTERNATE DATA PATH TO MOS BUS AND PACE
CONTROL
16 BIT DATA
MOS BUS
CONTROL
TO PACE
IPC-16A/505
ROM
16-BIT DATA
BTE
10-BIT TTL ADDRESS
NOTE
⚠1 BUFFERING REQUIREMENTS VARY WITH LEAD LENGTH AND TTL BUS LOADING.
CONTROL
CONTROL
SYSTEM TTL TIMING AND CONTROL BUS
NS10358

Figure 1-8. Implementation of RAM and ROM With On-chip Address Latches

1.5.3 PACE MEMORY CHIPS

The PACE Memory Chips include a read/write Random Access Memory (RAM), two Read-Only Memories (ROMs), and a Programmable Read-Only Memory (PROM). The RAM provides 256-by-4 bits of static read/write memory and also contains on-chip address latches and latched chip enables. Consequently, no need exists for ALEs between the System TTL Address/Data Bus and the RAM as shown in figure 1-8. Since the RAM contains static memory and utilizes only a +5-volt dc power source for operation, the hardware design is simplified for system applications using battery back-up or auto-fail/restart features that are often encountered in such applications as process control. Memory accessing is under control of the PACE microprocessor which provides control signals to the memory chips at the appropriate time by way of the System TTL Timing and Control Bus. The control signals permit the RAM to accept address information or to effect data I/O transfers.

The two available ROMs differ in memory density, presence of on-chip address latches, access time, and cost. The IPC-16A/505 ROM, shown in figure 1-8, is a high-performance, high-density (1024-by-16 bits) ROM that has on-chip address latches and requires only a +5-volt power supply. The 1024-by-16 bits of memory provide a control storage capacity that is sufficient for many applications. In addition, the on-chip address latches permit complete system implementation without supplemental address latches, and the bipolar technology eliminates the need for buffering in many cases. Thus, the IPC-16A/505 ROM is the ultimate solution to the low-cost storage requirements for high-volume systems.

Memory accessing of the ROM-stored data is accomplished by presenting address information on the System TTL Address/Data Bus at a time when the appropriate control signals are supplied by the PACE microprocessor over the System TTL Timing and Control Bus. In a similar manner, addressed data is read from memory onto the System TTL Address/Data Bus by the proper application of signals over the System TTL Timing and Control Bus. Since the CPU inputs are TTL compatible, the memory output can use the MOS bus as an alternate data path.

The IPC-16A/506 PROM contains 512-by-8 bits of ultraviolet erasable memory to provide a low-cost control storage solution for program development and for low-volume applications. Since the IPC-16A/506 PROM does not contain on-chip address latches, the ALE provides a convenient means to implement the IPC-16A/506 PROM, as shown in figure 1-9. Memory accessing is accomplished in a manner similar to the IPC-16A/505 ROM previously described, except the ALE is latched by the PACE microprocessor control signals when address information is present on the System TTL Address/Data Bus. Furthermore, the access time of the IPC-16A/506 requires the use of Cycle Extend as detailed in chapter 2.

Figure 1-9 also includes implementation of the IPC-16A/507 ROM. The ROM provides 512-by-8 bits of memory storage. The ROM is pin-for-pin compatible with the IPC-16A/506 PROM. Memory accessing is accomplished in the manner previously described for the IPC-16A/506 PROM.

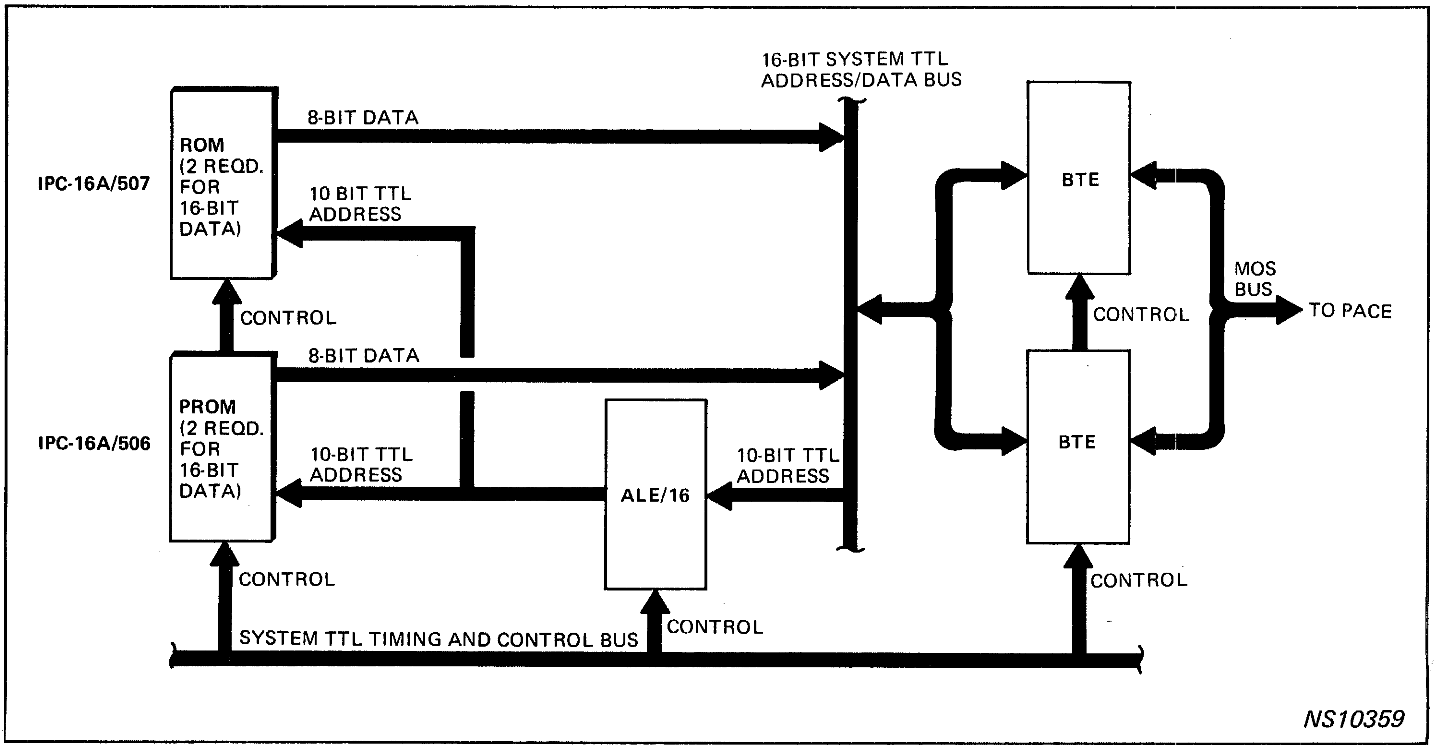


Figure 1-9. Implementation of ROM and PROM Without On-chip Address Latches

Figure 1-10 illustrates a typical one-board PACE application system containing 1024 16-bit words of ROM and 256 16-bit words of read/write RAM memory. The system shown in figure 1-10 is just one of many configurations that can be realized by using the PACE microprocessor and the associated family of chips. Using the PACE microprocessor and family of chips, the designer can put together an inexpensive but powerful microcomputer system in a minimum of time to meet individual requirements.

1.6 PACE APPLICATION CARDS

The following application cards support system design around the PACE microprocessor.

- PACE Application CPU Card
- PACE Application RAM Card
- PACE Application ROM/PROM Card

Future application cards will include:

- PACE Application Input/Output Interfacing Card
- PACE Application Multiply/Divide Card
- PACE Application DMA Controller Card

The application cards measure 4.375 inches by 4.862 inches and comprise circuits that implement the intended function of the card. The design-tested, proven application cards can be inserted into a universal card cage to immediately form the basis of a custom microcomputer system, thereby expediting hardware development. The application cards also can be used for end applications. The small card size particularly suits the use of application cards in physically confined situations such as portable equipment.

1.7 PACE MICROPROCESSOR DEVELOPMENT SYSTEM

The PACE Microprocessor Development System (IPC-16P), together with the available software and peripherals, provides a convenient means to expedite the hardware/software development stages of a system incorporating a PACE microprocessor. To facilitate software development, the the IPC-16P contains a Control Panel with various data indicators and switches that permit the user to examine and modify software during the development stage. In addition, the IPC-16P contains read/write memory (expandable to 32K words) as well as all required Card Reader/High-Speed Paper Tape Reader and Teletype®(TTY) interfacing and firmware (programs stored in ROM). The firmware consists of the routines necessary to permit peripherals (such as the TTY) to be used for loading, examining, or modifying software during development. Additional peripheral options include the Disc Operating System, Line Printer, and PROM Programmer.

The IPC-16P is similar to the National Semiconductor IMP-16P. The most important difference between the two systems is the PACE Development CPU Card contained in the IPC-16P. Consequently, the PACE Development CPU Card is separately available and, together with a field conversion kit, permits reconfiguration of an IMP-16P System for use as a PACE Microprocessor Development System. The PACE Development CPU Card is supplied with a cable that permits connection of the IPC-16P into an application system in place of a PACE Application CPU Card. Thus, the prototyping features inherent to the IPC-16P Control Panel, peripherals, and software can be used to develop a user application system containing PACE application cards.

NOTE

A hardware summary table of the PACE product line is located in appendix A, table A-1.

1.8 SOFTWARE SUPPORT

The importance of National Semiconductor-supplied support software cannot be overemphasized. The microprocessor design process is most efficient when the designer fully appreciates and uses the support software.

1.9 SPECIFYING HARDWARE BEHAVIOR WITH SOFTWARE

The microprocessor approach differs from older, discrete-logic controllers in only one important way, and all differences in approach stem from the following.

- In the random-logic approach, a set of logic is wired to handle each function, and all logic operations proceed in parallel.
- In the microprocessor approach, one central set of logic is provided inside the microprocessor. The central set of logic is rewired in real time, under control of the program, to handle each of the logic functions in serial.

Thus, the discrete-logic designer buys a set of functions and then wires the functions to perform a specific job. On the other hand, the microprocessor user buys a microprocessor and then must tell the microprocessor how to wire itself, from microsecond to microsecond, to perform different jobs.

The purpose of the system software is to aid the user in describing, designing, and debugging a microsecond-to-microsecond description of the microprocessor wiring. Such a description, when rendered in terms the microprocessor can understand, is called a program.

1.10 TYPES OF SOFTWARE

One of the most important steps in the programming process is the translation of the program description from commands that the programmer writes and understands into binary strings that the microprocessor uses to perform operations. Two types of commonly used translator programs are assemblers and compilers.

Utility programs facilitate the preparation of input code (using the Editor Program) and the debugging of the resultant object code (using the Debug Program) and, also, are used to enter the programs (using the Loader Program) into the Microprocessor Development System.

® Trademark of the Teletype Corporation

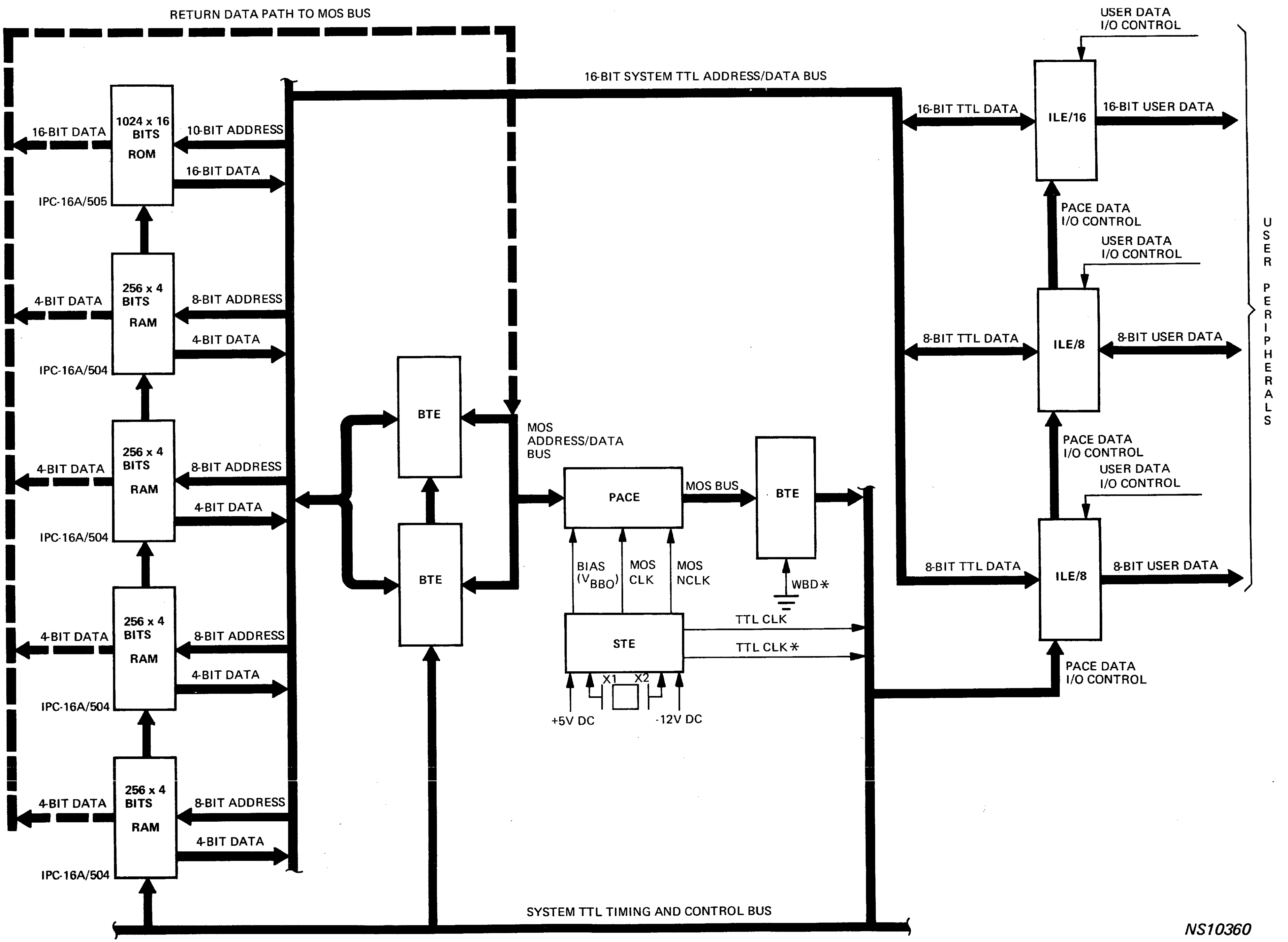


Figure 1 - 10. One-bo PACE Application System

The following paragraphs provide more detailed descriptions of assembler and compiler programs and three types of utility programs (Editor, Debug, and Loaders).

Figure 1-11 gives a birds-eye view of the PACE computer-program breakdown.

NOTE

A software summary table of the PACE software line is located in appendix A, table A-2.

1.10.1 PACE EDITOR

The PACE Editor enables the generation of new source statement text and the modification of existing source text in preparation for program assembly. The normal editing procedure is to input assembly-language source statements and comments, edit the text, and output the edited text, along with a punched paper tape suitable for input to the assembler.

1.10.2 PACE ASSEMBLERS

The user has the alternative of selecting among four PACE assemblers: the PACE Resident Assembler, the PACE IMP-16 Cross Assembler, the PACE Conversational Assembler, and the PACE FORTRAN Cross Assembler. All Assemblers are completely compatible in programs assembled and vary only in operating environments.

The PACE Resident Assembler runs on an IPC-16P. The PACE IMP-16 Cross Assembler runs on an IMP-16P or IMP-16L with a minimum of 4K words of memory and a TTY. The PACE Resident Assembler and PACE IMP-16 Cross Assembler accept *free-format* source statements from either the keyboard, paper tape, or a card reader and produce a Load Module (LM) on paper tape and an object listing on the TTY printer. The Resident and Cross Assemblers require three passes over the source program; however, if either the object listing or the LM is suppressed, only two passes are required.

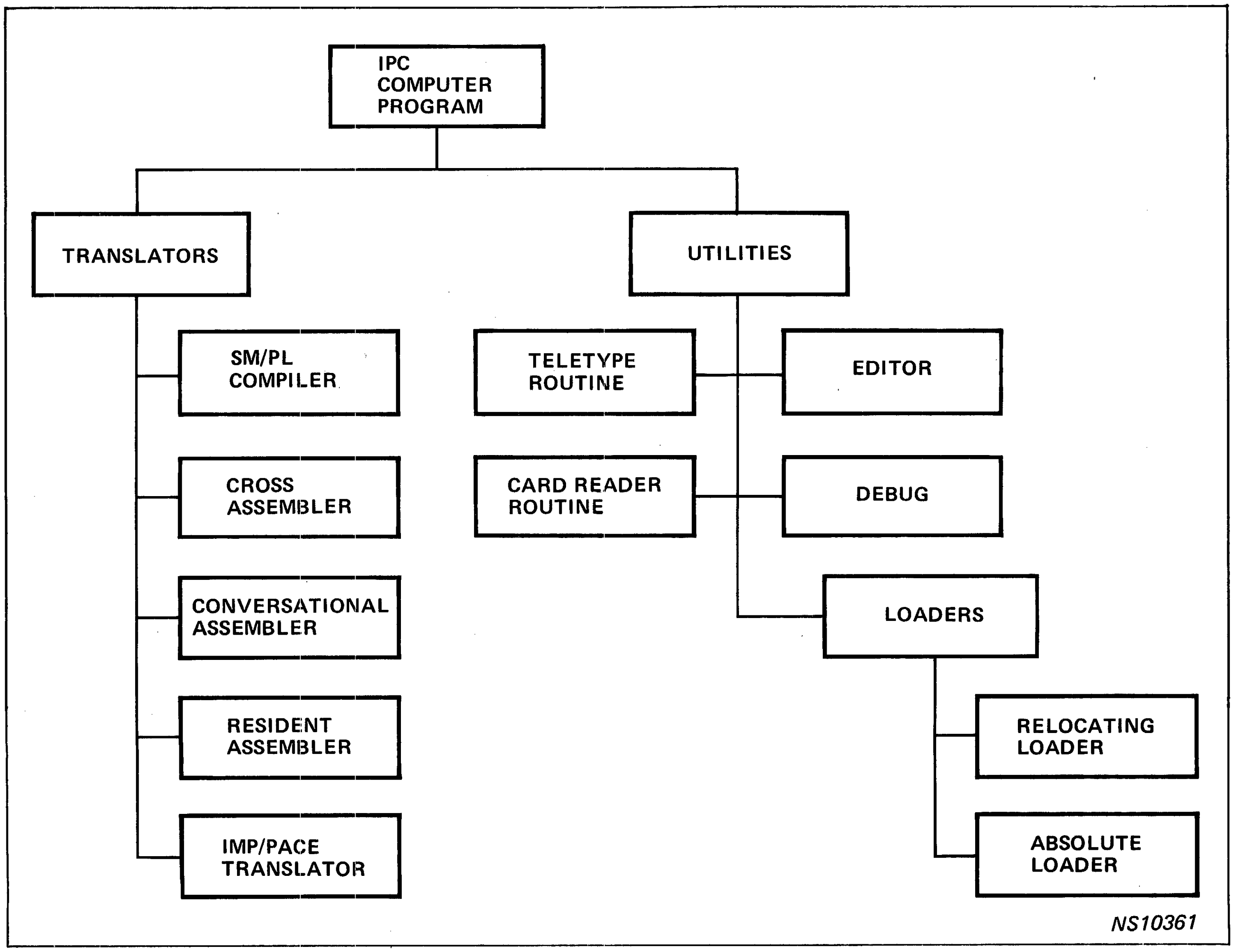


Figure 1-11. PACE Computer Programs

The PACE Conversational Assembler, which runs on an IPC-16P and combines the features of an editor and a resident assembler, simplifies the editing and assembly procedures by eliminating the need for multiple loadings of an editor, a resident assembler, and the user-generated program. The PACE Conversational Assembler requires 8K words of memory for operation.

The PACE FORTRAN Cross Assembler Program generates an object program from a source program on a host computer for subsequent execution by a PACE microprocessor. The assembler may be used on different host processors since the assembler is written in FORTRAN IV (USA Standard Language Subset). The assembler requires 100K bytes of memory and the following minimum hardware complement: processor input unit, scratch unit, list output unit, and binary output unit.

The PACE FORTRAN Cross Assembler accepts *free-format* source statements and, in two passes, produces an LM (object program) and a program listing.

1.10.3 PACE SM/PL COMPILER

The PACE SM/PL Compiler is a high-level computer program written in IPC assembly language. Comparable to high-level-language programming of the large-scale computers and minicomputers, the SM/PL Compiler considerably simplifies microcomputer programming. This results in fewer programming manhours and shortens leadtime – and, hence, reduces programming cost.

The SM/PL Compiler runs on an IPC-16P Microprocessor Development System, with a requirement of at least 12K memory words. The object code thus produced is highly efficient – in many cases comparable to the object code produced by programs written in the IPC assembly language. The object code is in standard Relocatable-Load-Module (RLM) format. All IPC peripherals are supported by the SM/PL Compiler.

A sequence of declarations and statements comprise the language of the SM/PL Compiler. Declarations control allocation of storage, define simple macros, and define procedures. Statements compute results and store them in a location defined by a variable name; statements also provide conditional tests and branching, iteration control, and procedure innovation.

Compiler procedures are in the form of subroutines that are defined by declarations and called by statements. Each subroutine may represent a program module, so a particular program may perform a number of tasks, each task being implemented by a subroutine. These subroutines are available to be used as procedures as part of other similar programs.

Figure 1-12 illustrates the software used to assemble or compile on PACE.

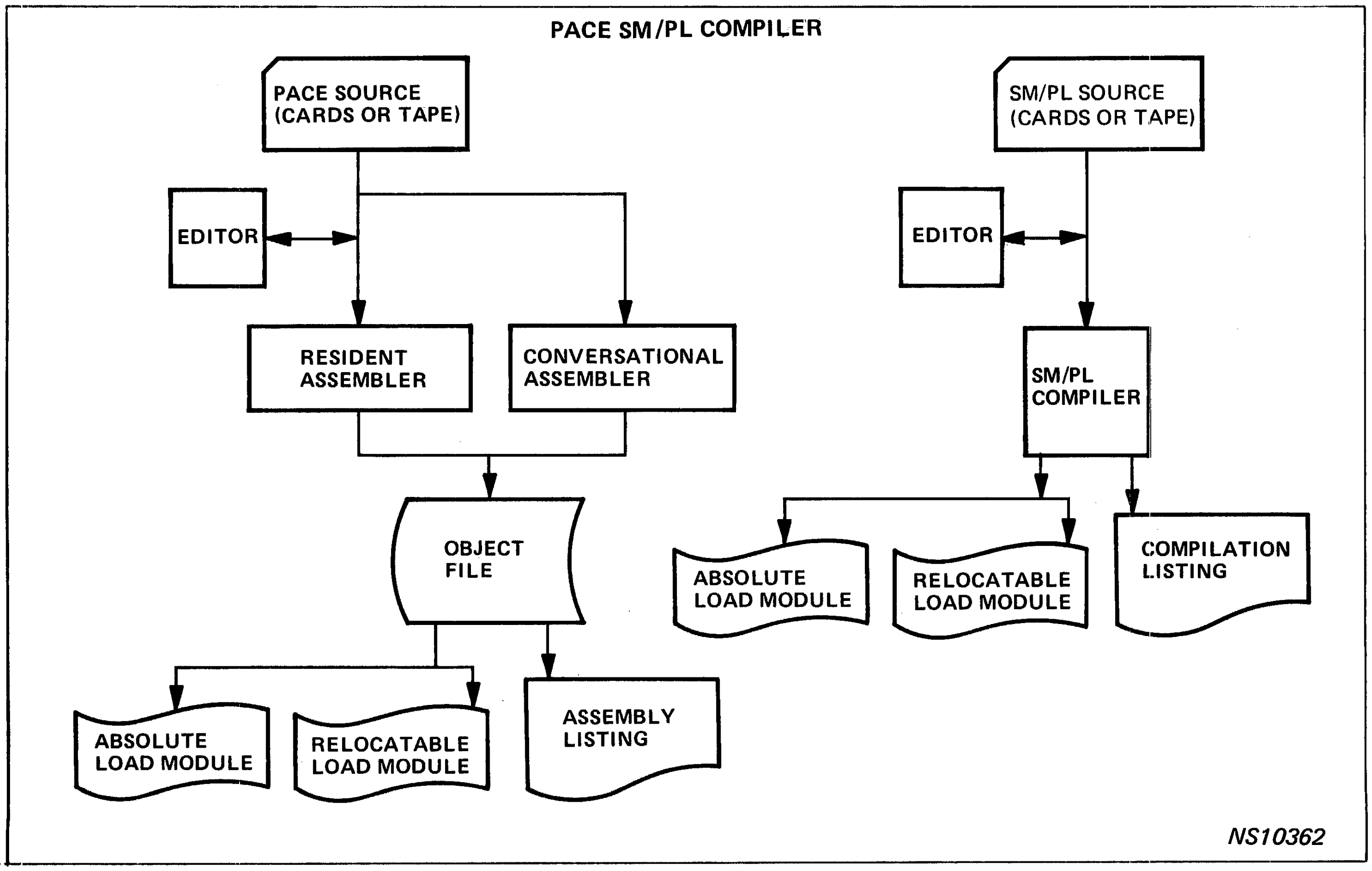


Figure 1-12. Software Used to Assemble or Compile on PACE

1.10.4 IMP-16/PACE TRANSLATOR

The IMP-16/PACE Translator is an ANSI FORTRAN program that transliterates source programs written in IMP-16 assembly language into PACE source programs that can be assembled and executed.

The IMP-16/PACE Translator is installed and available to users of the General Electric National Timesharing Service under the program name TRAN$$ to translate an existing IMP-16 assembly-language file into PACE assembly language so the new file may be assembled for execution on the PACE microprocessor. Figure 1-13 illustrates how PACE software is implemented on a host computer.

1.10.5 PACE LOADERS

The PACE loaders are programs that read and load one or more LMs, produced by a PACE assembler, into the main memory for execution.

The output by the PACE FORTRAN Cross Assembler is reformatted into an LM before loading into the PACE memory for execution. ANSI FORTRAN programs are available to reformat the output from the PACE FORTRAN Cross Assembler into an LM suitable to the loader and loading method employed.

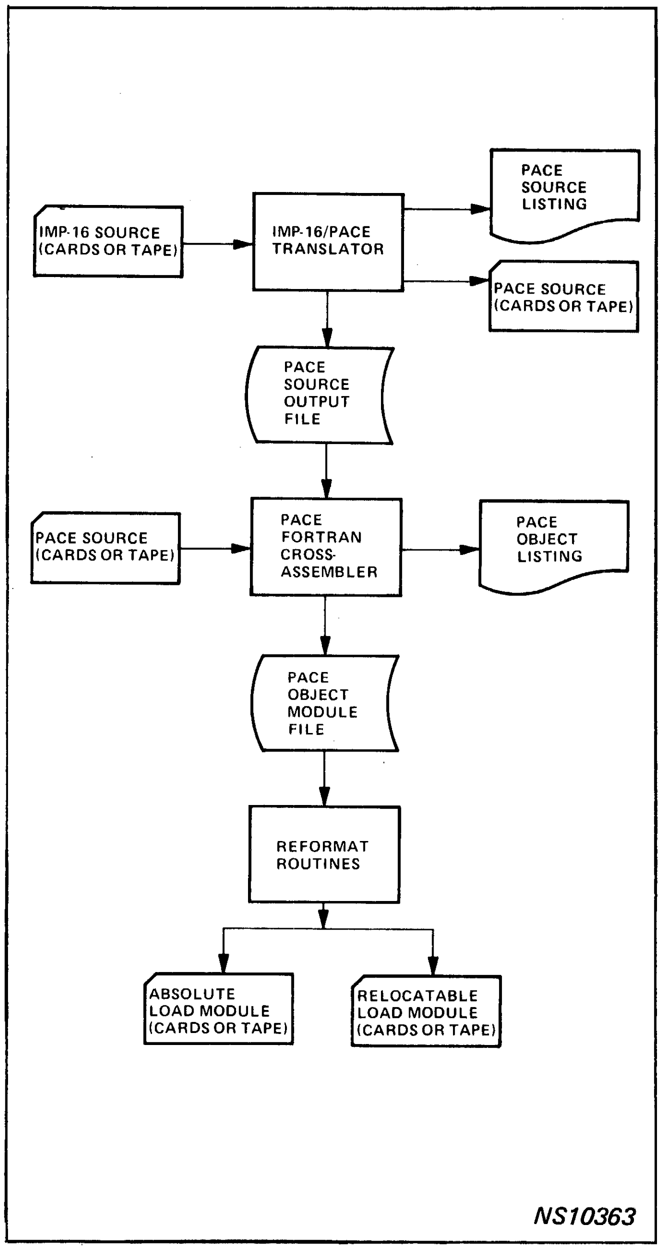
The outputs from the PACE Resident Assembler and the PACE IMP-16 Cross Assembler do not require reformatting. The LMs are output directly from the PACE Resident Assembler and PACE IMP-16 Cross Assembler onto paper tape.

Two methods are available for loading data into the main memory for execution: absolute and relocatable. Each loading method involves tradeoffs among the following considerations: the complexity of the loading process, the amount of work that must be performed by the user, and the flexibility available to the user at load time (versus assembly time).

Several PACE programs are available for loading correctly formatted LMs into the PACE memory for execution: PACE Relocating Loader (PACE General Loader), PACE Absolute Card Reader Loader, and PACE Absolute Paper Tape Loader. The loading methods and the loaders available for each method are described in the following paragraphs.

1.10.5.1 PACE Absolute Loaders

A PACE Absolute Loader, resident in the ROM of the IPC-16P, loads one or more programs into preallocated, fixed areas of memory. The exact memory areas to be occupied by each user-generated program must be determined by the user before assembly. Also, any linking of one program to another or to common, shared data must be accomplished at assembly time by assignment of common labels to fixed, absolute addresses in memory. The advantages of using a PACE Absolute Loader are that a small, simple loader may be used and no commands are required at load time.



IMP-16 SOURCE (CARDS OR TAPE)
IMP-16/PACE TRANSLATOR
PACE SOURCE LISTING
PACE SOURCE (CARDS OR TAPE)
PACE SOURCE OUTPUT FILE
PACE SOURCE (CARDS OR TAPE)
PACE FORTRAN CROSS-ASSEMBLER
PACE OBJECT LISTING
PACE OBJECT MODULE FILE
REFORMAT ROUTINES
ABSOLUTE LOAD MODULE (CARDS OR TAPE)
RELOCATABLE LOAD MODULE (CARDS OR TAPE)
NS10363

Figure 1-13. PACE Software Implemented on a Host Computer

1.10.5.2 PACE Relocating Loader

The PACE Relocating Loader (PACE General Loader) is a command-driven PACE program that reads one or more relocatable LMs from either the Card Reader or the Paper Tape Reader, relocates object code, and transfers control to the specified entry point. The PACE Relocating Loader provides the most flexible loading process. The PACE Relocating Loader process allows relocation of programs at LM load time rather than at assembly time. The PACE Relocating Loader follows either an inherent method for allocating programs to available memory or user-generated instructions that designate where each program should be loaded. Figure 1-14 illustrates the PACE loading sequence.

1.10.6 PACE INPUT/OUTPUT ROUTINES

The PACE Input/Output Routines are described in the following paragraphs.

1.10.6.1 PACE Teletype Routines

The PACE Teletype Routines reside in ROM on the TTY/Card Reader Interface Card of the IPC-16P System. The routines are used to send and receive information to and from the TTY or to receive data from the Paper Tape Reader. When both a High-speed Paper Tape Reader and a TTY Paper Tape Reader are used, the program verifies the tape reader that first supplies data and, subsequently, accepts input data from that tape reader.

1.10.6.2 PACE Card Reader Routine

The PACE Card Reader Routine resides in ROM on the TTY/Card Reader Interface Card of the IPC-16P System. The PACE Card Reader Routine accepts an absolute LM in Hollerith-coded card format and loads the data into main memory. There are no restrictions on loadable addresses; any read/write memory location can be used.

1.10.7 PACE DEBUG PROGRAM

The PACE Debug Program supervises the operation of a user program during checkout. This program provides the following facilities for testing computer programs:

- Printing selected areas of memory in hexadecimal or ASCII format
- Modifying the contents of selected areas in memory
- Modifying computer registers and stack
- Inserting instruction breakpoint halts
- Taking memory snapshots during execution of a user program
- Initiating execution at any point in a program
- Searching memory

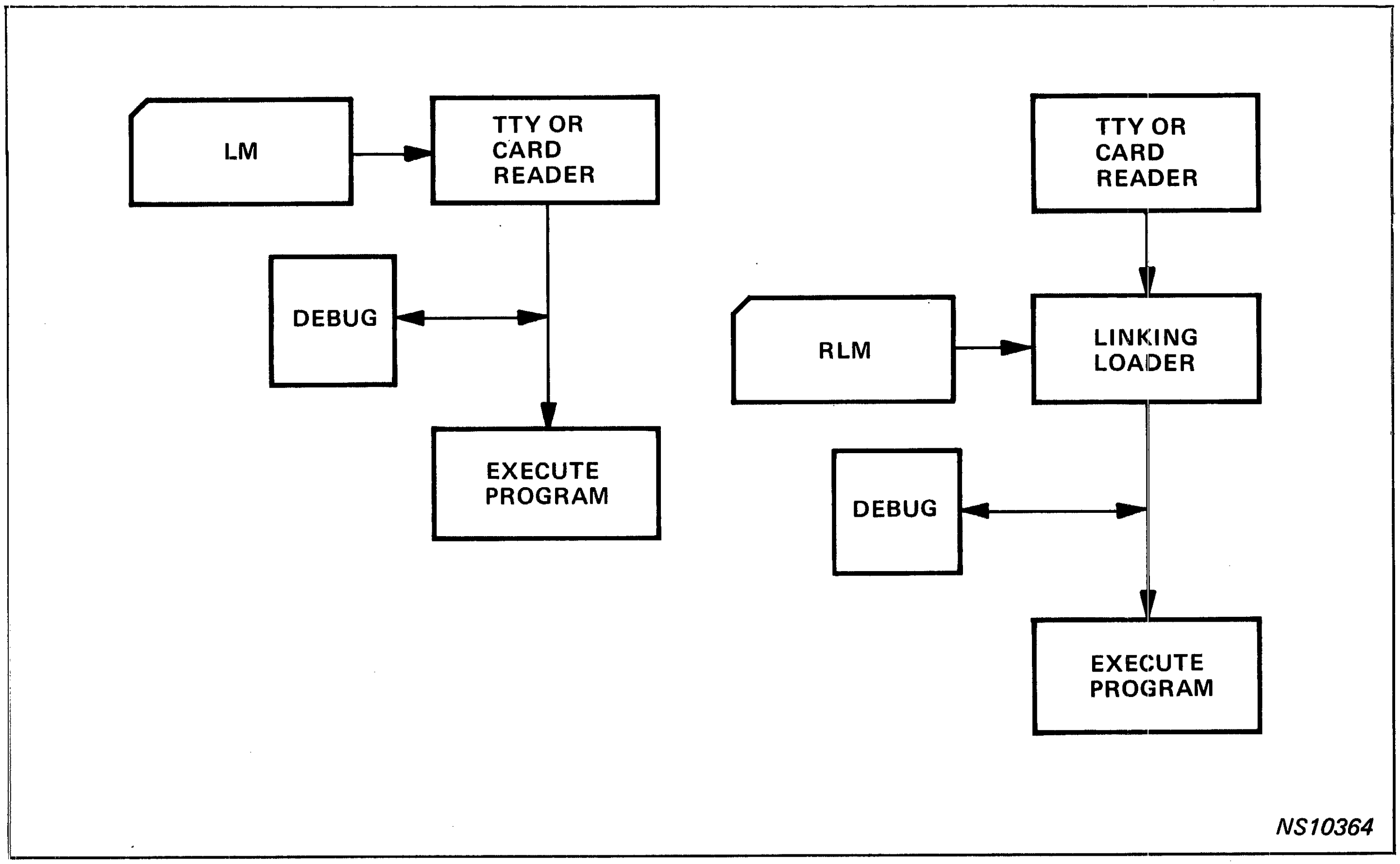


Figure 1-14 PACE Loading Sequence

CHAPTER 2
PACE AND FAMILY OF CHIPS

2.1 INTRODUCTION

The following paragraphs provide additional descriptive information and in-depth application data concerning the PACE microprocessor and family of chips. The PACE instruction set and addressing methods are detailed in appendix B. The instruction set description includes the instruction word bit configuration, assembler format, and instruction execution time formula for each instruction type. Applications data are provided for input/output control techniques, use of jump conditions and flags, interrupts, Cycle Extends, and DMA operation.

NOTE

Since this document was prepared during the final design phase of the PACE product line, some discrepancies may exist in the timing and electrical specifications presented with the following application information. For preliminary design purposes, refer to the latest data sheets to verify the timing and electrical parameters.

2.2 PACE MICROPROCESSOR

The PACE microprocessor provides the control and timing signals required for system or subsystem operation in addition to providing data manipulation and storage capabilities. The following paragraphs provide more information regarding the PACE microprocessor.

2.2.1 GENERAL DESCRIPTION

Data transfers between PACE (see figure 2-1) and memory or peripheral devices are effected over the 16-bit (D00-D15) parallel Input/Output Data Bus. The Input/Output Data Bus interfaces with the Instruction Register and the Operand Bus by way of the I/O Data Buffers. The Operand Bus also interfaces with seven registers (Temporary Registers 1 and 2, Program Counter, and AC0 through AC3) and a 10-word Stack. The seven registers and Stack are provided for data storage. Four of the registers (AC0 through AC3) are available to the programmer as general-purpose accumulators. The Program Counter contains the address of the next instruction. The contents of any selected register or the Stack are routed over the Operand Bus to the Arithmetic and Logic Unit (ALU) and Shifter. Resultant ALU and Shifter output is returned to the selected register or Stack, as appropriate, by way of the Result Bus. The ALU and Shifter, besides performing arithmetic operations, also sets status flags in accordance with the data length (8-bit or 16-bit) selected by the state of the BYTE Status Flag.

All status information is stored in the 16-bit Status and Control Flags Register. The Status and Control Flags Register contents can be loaded onto the Operand Bus for temporary storage on the Stack or in any accumulator for examination or modification of status information.

Instructions under execution by PACE are stored in the Instruction Register and are interpreted and executed by a microprogram stored in an on-chip ROM. Instruction execution time is determined by the instruction under execution, memory access time, and the external clock frequency.

2.2.2 EXTERNAL CLOCK REQUIREMENTS

The external clock signals (see figure 2-2) applied to PACE must consist of single-phase true and complement signals such as those produced by a System Timing Element (STE). PACE uses the external clock signals to generate internal multiphase clock signals that provide control timing for microprocessor operations.

2.2.3 DESCRIPTION OF HARDWIRED SIGNALS AND TIMING

PACE operations are controlled by software which *rewires* the PACE Control Logic in real time (at a speed determined by the external clock) to handle each microprocessor function in serial order. The clock and other signals that require hardwired connection to the PACE microprocessor are described in table 2-1. Pin assignments for the PACE CPU are shown in figure 2-3.

Instructions consist of four machine cycles or more, depending on the operations performed. The timing shown in figures 2-4, 2-5, and 2-6 represent the first machine cycle of the instruction being executed. The number of cycles for the instructions are given in appendix B.

NOTES

1. Positive logic convention is used throughout this manual. A logic '1' or high signal corresponds to a more-positive voltage level. A logic '0' or low signal corresponds to a more-negative voltage level. All signal names beginning with *'N'* or followed by an asterisk (✱) denote complemented signals that are asserted or activated by a logic '0'. Otherwise, signals are asserted by a logic '1'.
2. Bits are numbered from 00 to 15, right to left, with bit 00 representing the least significant bit.
3. The *X'* preceding a value denotes the hexadecimal numbering system.

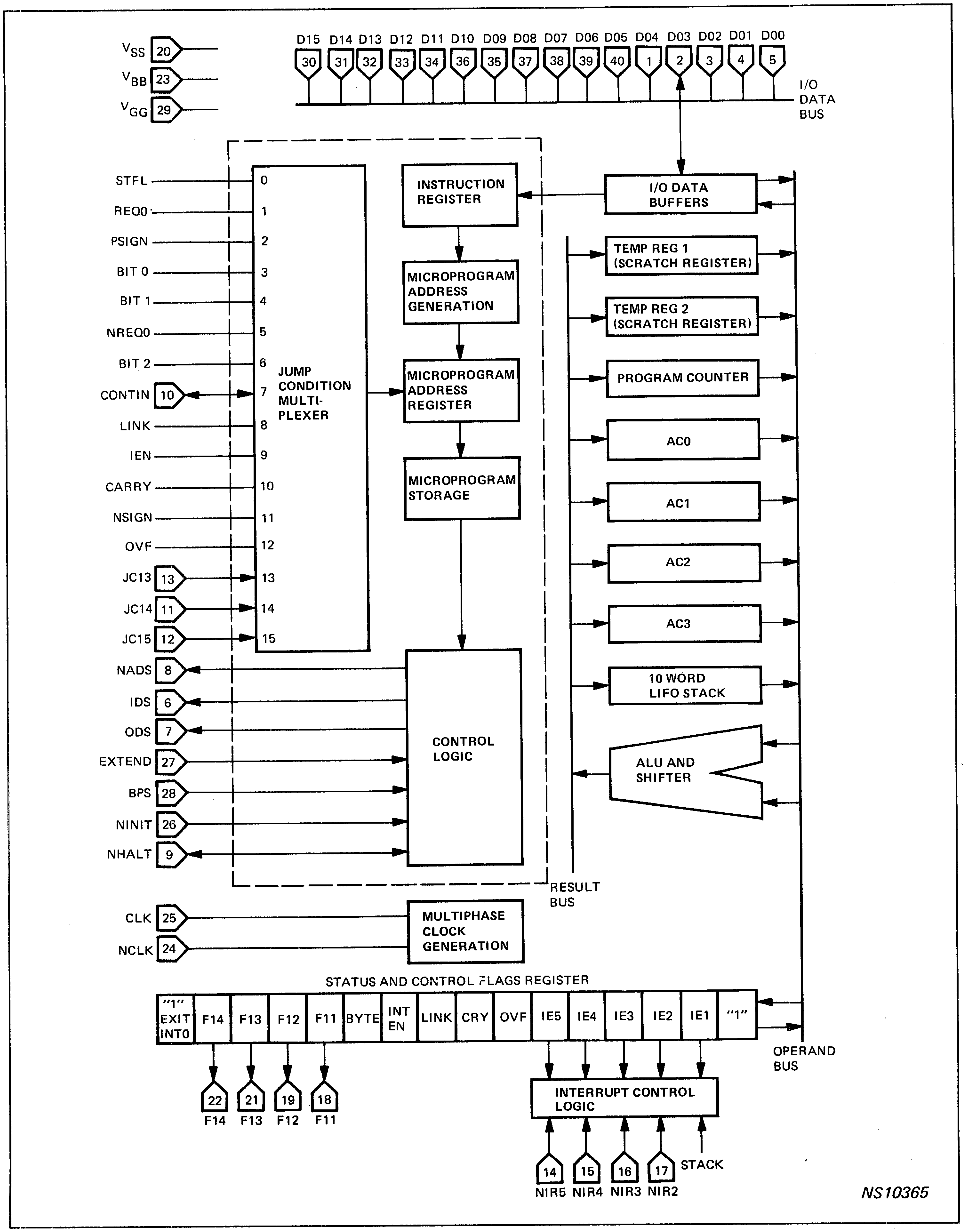
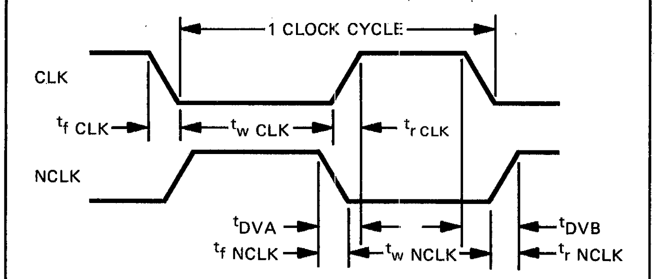


Figure 2-1. PACE Microprocessor Functional Block Diagram



Where:
t_r AND t_f = RISE AND FALL TIMES ≈ 25 ns, TYPICAL
t_{WCLK} AND t_{WNCLK} = CLOCK WIDTH = 210 ns, TYPICAL
t_{OVA} AND t_{ONB} = CLOCK OVERLAP = 25 ns, TYPICAL
t_p = CLOCK PERIOD = 0.5 μs, TYPICAL

NS10366

Figure 2-2. External Clock Timing Parameters

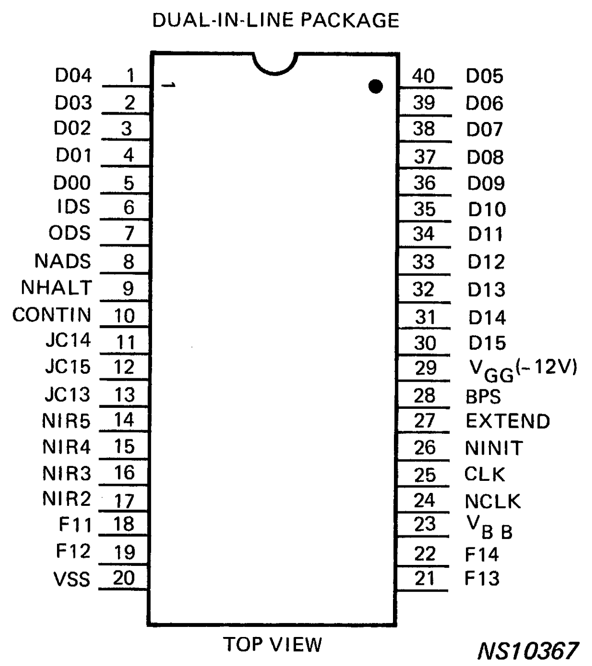


Figure 2-3. PACE Microprocessor Pin Assignments

Table 2-1. Descriptions of PACE Hardwired Signals

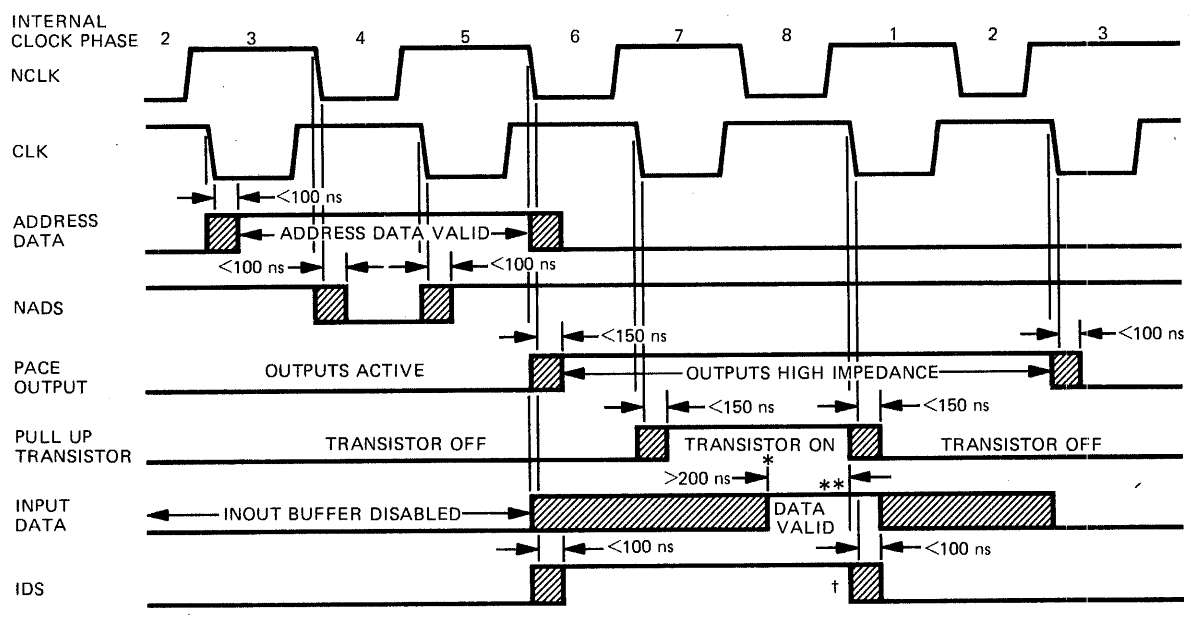
Signal Mnemonic/Name	Description
	NOTES 1. Some of the PACE microprocessor functions and buses referred to in the signal descriptions are illustrated in figure 2-1. 2. Refer to figures 2-4, 2-5, and 2-6 when signal descriptions discuss address output/data input timing, data output timing, and extending I/O signal timing, respectively. 3. Figure 2-7 illustrates user flag timing considerations.
CLK, NCLK/ True and complemented clock	External true and complemented clock inputs to PACE. Used in generation of PACE internal multiphase clock signals that provide timing control for internal PACE functions.
D00-D15/Data Bits 00-15	Input/output MOS Data Bus Lines.
IDS/Input Data Strobe	PACE output signal used to enable external devices so data can be placed on-line to PACE. IDS operation is as follows: 1. Following output of peripheral or memory address information from PACE (see figure 2-4), D00-D15 data line drivers (internal to PACE) assume high-impedance state and PACE Control Logic drives IDS Signal high. 2. IDS remains high for approximately 1.5 CLK periods. 3. Valid input data to PACE must be present on D00-D15 Input/Output Data Bus Lines when IDS is driven low again by Control Logic after approximately 1.5-CLK-period duration.

Table 2-1. Descriptions of PACE Hardwired Signals (Continued)

Signal Mnemonic/Name	Description
ODS/Output Data Strobe	PACE output signal used to enable external devices to accept data output from PACE. ODS operation is as follows: 1. Following output of peripheral or memory address information from PACE (see figure 2-5), data are placed on D00-D15 Input/Output Data Bus Lines by PACE. 2. At approximately the same time that data are placed on Input/Output Data Bus, ODS Signal is driven high by PACE Control Logic to signify that output data from PACE are available to memory or peripherals. 3. ODS remains high for approximately 1.5 CLK periods. 4. Output data remain on Input/Output Data Bus after ODS is driven low again by Control Logic after approximately 1.5-CLK-period duration. Thus, ODS trailing edge can be used to clock PACE output data into External Data Latch (ALE). ODS can also be used as read/write control signal for external RAM memory elements.
NADS/Address Data Strobe	PACE output signal used to clock address information from PACE into ALE. After address information (see figures 2-4 and 2-5) is placed on Input/Output Data Bus by PACE, NADS Signal is driven low for approximately 0.5 CLK period by PACE Control Logic. NADS is active in middle of approximately 1.5 CLK periods that address information is valid. Thus, either edge of NADS can be used to clock address information into ALE.
EXTEND/Extended Data Transfer	PACE input signal used to temporarily increase time duration of data input/output transfers to accommodate accessing of slow memories or peripherals without altering CLK frequency. EXTEND Signal must be driven high at beginning of ODS or IDS Signal (see figure 2-6). If EXTEND is held high as indicated in figure 2-6, data-transfer operation is extended by 1 CLK period. Holding EXTEND high for additional n clock periods increases data-transfer timing by n + 1 clock periods.
NINIT/Initialize	PACE input signal that initializes microprocessor functions. When NINIT is low, PACE operation is suspended and all PACE strobe signals (IDS, ODS, NADS, and so forth) are set to inactive state. After NINIT completes low-to-high transition, the following conditions are effected: 1. PACE Program Counter contents are set to zero. 2. Internal Stack Pointer (indicates last Stack level accessed) is cleared. 3. All flags and interrupt enables are set low except Level-0 Interrupt Enable which is set high. All other registers contain an arbitrary value.
NHALT/Control Panel Halt	PACE Control Logic input/output signal used for nonmaskable Level-0 Interrupt, microprocessor stall, and programmed HALT indicator output. When NHALT is applied as low input, microprocessor operation halts after completing execution of current instruction. When Halt Instruction is executed, NHALT Line is driven low by PACE Control Logic for a 7/8 duty cycle. Microprocessor can be stalled by using external open-collector driver to hold NHALT Line low for desired time duration, thereby overriding NHALT output buffer on PACE chip.

Table 2-1. Descriptions of PACE Hardwired Signals (Continued)

Signal Mnemonic/Name	Description
CONTIN/Continue Jump Condition	PACE Jump Condition Multiplexer input/output signal used to sense external signal through BOC Instruction. Also used to restore microprocessor operation from suspended state or cause subroutine branch to Level-0 Interrupt Service Routine (generally used to implement Control Panel functions). Driving CONTIN Input high for 4 CLK periods, minimum, causes halted microprocessor to resume operation. As output, CONTIN is driven low for approximately 3 clock periods by PACE Jump Condition Multiplexer to acknowledge that microprocessor operation is stalled. CONTIN Line must be pulsed to terminate Halt Instruction.
BPS/Base Page Select	Input signal to PACE Control Logic that enables one of two base-page addressing schemes to be selected. When BPS is low, first 256 words of memory constitute base page (page zero). When BPS is high, first 128 memory words and last 128 memory words constitute base page.
JC13, 14, 15/Jump Conditions 13, 14, and 15	User-specified branch-condition inputs to PACE Jump Condition Multiplexer. Some possible uses are testing system status and receiving serial data. When JC13, 14, or 15 is high, PACE Branch-On Condition Instruction effects program branch if Jump Condition Input is true.
F11, 12, 13, 14/Flags 11, 12, 13, and 14	PACE Status and Control Flags Register general-purpose control flag outputs. F11-14 may be used for direct control of system functions or serial data output. Individual flags may be set by PACE Set Flag Instruction and pulsed or reset by Pulse Flag Instruction (see figure 2-7). Push Flag and Pull Flag Instructions permit contents of Status and Control Flags Register to be saved on Stack during Interrupt Service Routine or subroutine execution, and then restored.
NIR2, 3, 4, 5/Interrupt Requests 2, 3, 4, and 5	Inputs to PACE Interrupt Control Logic. When NIR2, 3, 4, or 5 Input is low for 1 CLK period, minimum, corresponding internal Interrupt Request Latch is set. NOTE *Use of Interrupts* paragraph later in this chapter provides more comprehensive information concerning interrupt servicing.
V_{BB}	PACE input substrate voltage requirement derived from +5-volt and −12-volt supplies by STE.
V_{GG} (-12V)	PACE input power requirement.
V_{SS} (+5V)	PACE input power requirement.



Note: Signals are referenced to valid logic levels on clock inputs. All times are typical maximum or minimums. Internal clock phases are shown for reference only, they are not available externally.

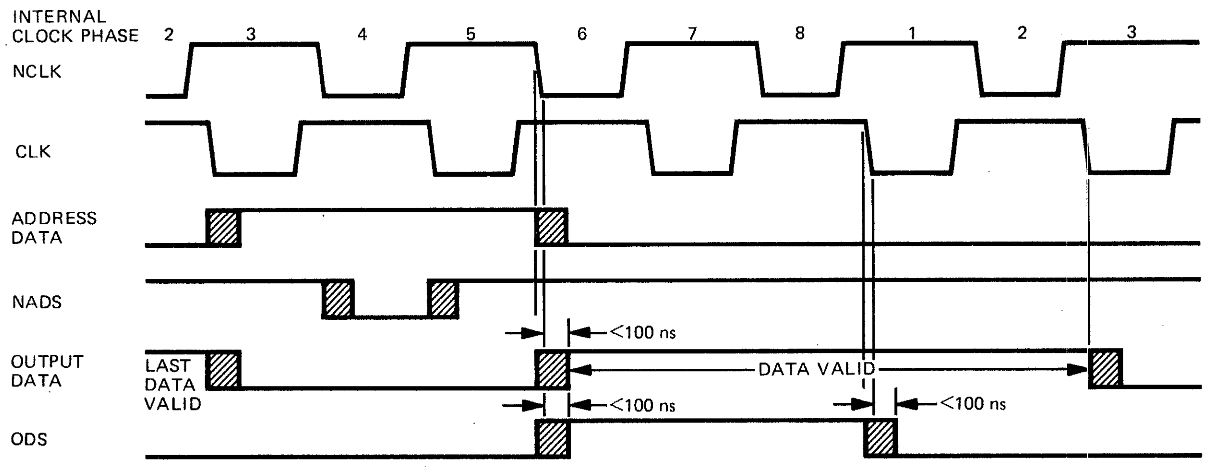
* V_{IN} must be > V_{SS} -2.35V at this time if logic "1" input.

** V_{IN} must be valid level (i.e., V_{SS} = 1) at this time (this timing allows for pull-up transistor constant).

†Data must be valid until trailing edge of IDS (i.e., data hold time = 0 ns).

NS10368

Figure 2-4. Address Output and Data Input Timing Diagram



NS10369

Figure 2-5. Data Output Timing Diagram

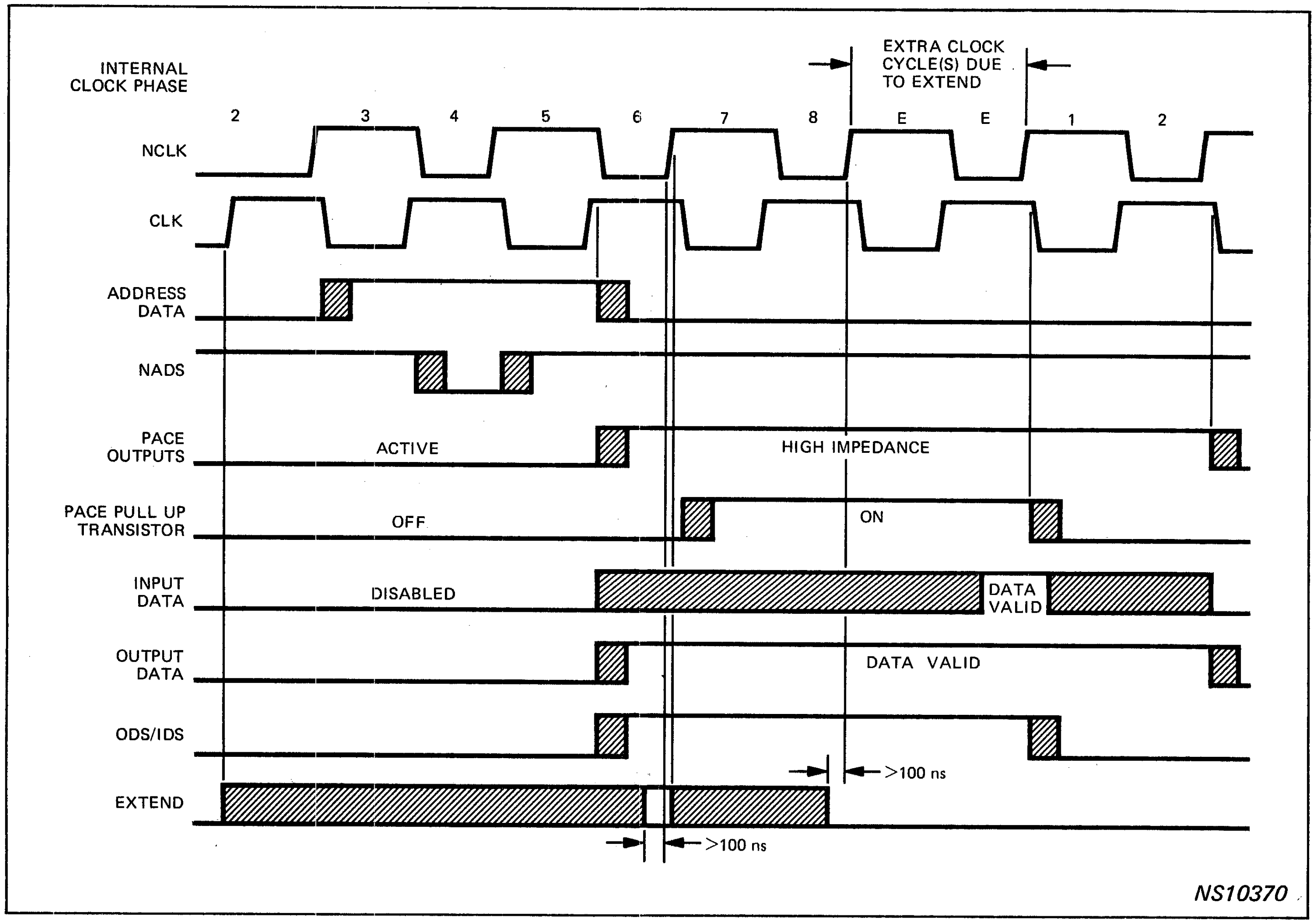


Figure 2-6. Extend I/O Signal Timing Diagram

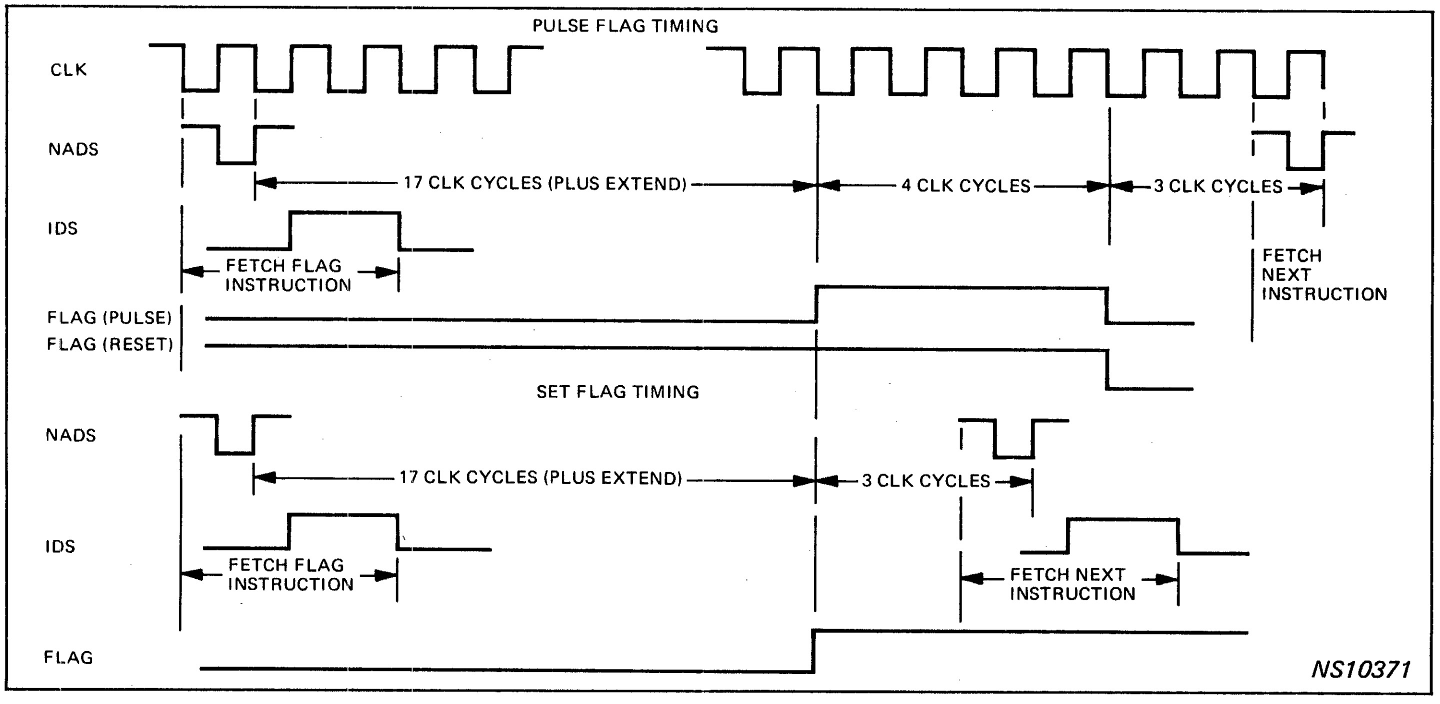


Figure 2-7. Pulse and Set Flag Timing Diagram

2.2.4 DESCRIPTIONS OF STATUS AND CONTROL FLAGS

Fourteen status and control flags are provided by the PACE microprocessor in the Status and Control Flags Register. The flags contained in the Status and Control Flags Register can be accessed or restored as a 16-bit data word by using the Copy Flags to Register or Copy Register to Flags Instructions, respectively. Also, the flags can be individually set, pulsed, or reset and the contents of the Status and Control Flags Register can be saved on the Stack or restored from the Stack by using the appropriate PACE instructions. Table 2-2 provides descriptions of the individual status and control flags.

2.2.5 INPUT/OUTPUT BUS STRUCTURE

The multiplexed I/O Data Bus pins on the PACE microprocessor permit implementation of a fully multiplexed bus structure, as shown in figure 2-8, to minimize wiring costs and reduce device counts. However, a variety of bus structures may be implemented. Two alternative bus structures are illustrated in figures 2-9 and 2-10. All input/output operations are controlled by the user-generated program under execution by PACE.

2.2.6 DATA REPRESENTATION

In the PACE microprocessor, data are represented in the twos-complement number system in which the negative of a number is formed by complementing each bit and, then, adding one to the complemented value of the number. The most significant bit position indicates the sign of the number: 0 for positive and 1 for negative. With a single 16-bit value, the greatest positive number is X'7FFF or $(32767)_{10}$ and the most negative number is X'8000 or $(32768)_{10}$. When the 8-bit data length is selected, the largest positive number is X'7F or $(127)_{10}$ and the most negative number is X'80 or $(128)_{10}$.

2.3 PACE INSTRUCTION SET

The PACE instruction set contains 45 instruction types that are capable of providing 337 individual instructions when flags, branch conditions, and other conditional signals or tests are considered. The 45 instruction types are divided into the eight following format groups:

- Branch Instructions
- Skip Instructions
- Memory Data-transfer Instructions (also serve as I/O instructions)
- Memory Data-operate Instructions
- Register Data-transfer Instructions
- Register Data-operate Instructions
- Shift and Rotate Instructions
- Miscellaneous Instructions

2.3.1 MEMORY ADDRESSING

Both direct and indirect memory addressing instructions are included in the PACE instruction set. The following paragraphs provide descriptions of direct and indirect memory addressing.

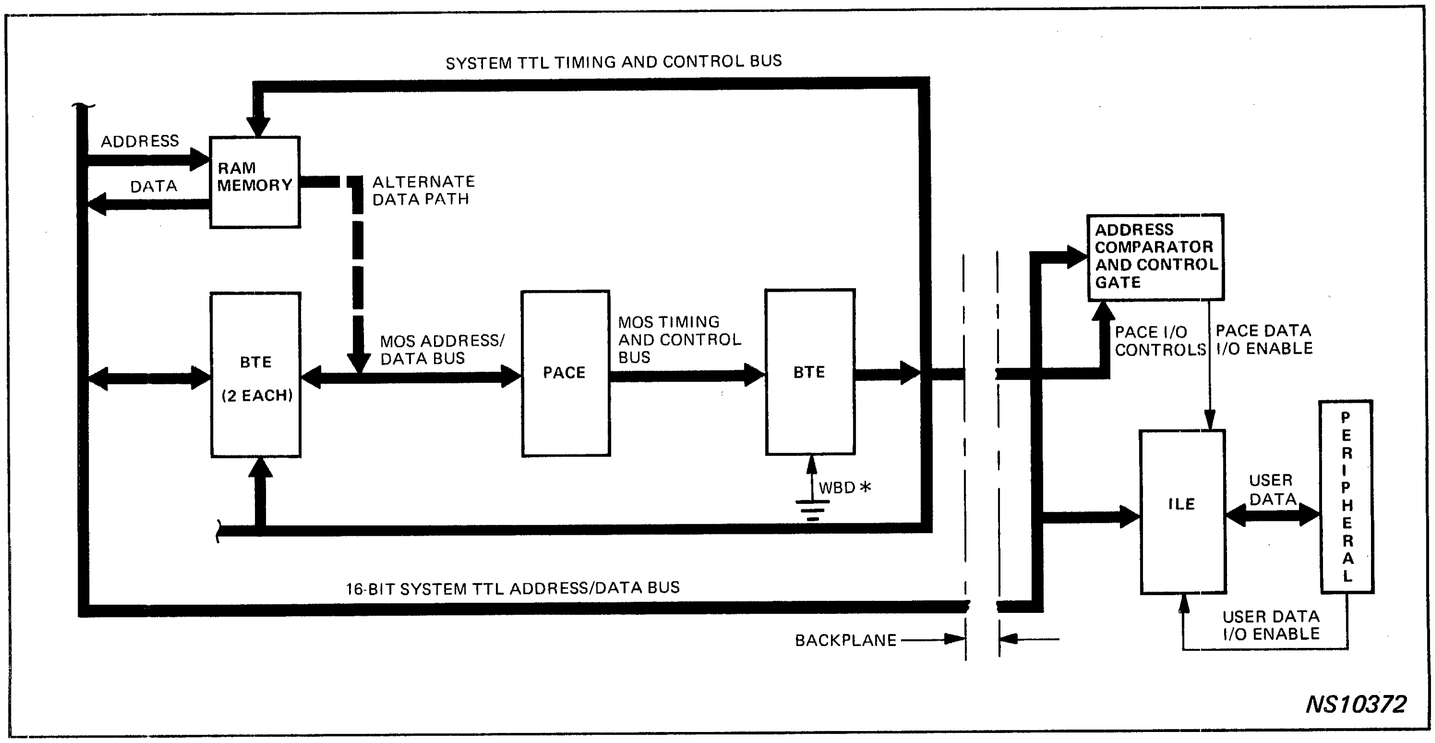


Figure 2-8. Fully Multiplexed Bus Structure

Table 2-2. Descriptions of Status and Control Flags

Register Bit	Flag Name	Description	Flag Code (fc)
0	High ('1')	Bit 0 is not used and is always in logic '1' state. Referencing bit 0 with SFLG or PFLG Instruction has no effect. (May be used as NOP Instruction.)	0000
1 2 3 4 5	IE1 IE2 IE3 IE4 IE5	Flags IE1 through IE5 serve as Interrupt Enable Flags for five of six PACE interrupt levels. If Interrupt Enable is high and associated Interrupt Request occurs, microprocessor executes Interrupt Service Routine. If Interrupt Enable is low, associated Interrupt Request is ignored.	0001 0010 0011 0100 0101
6	OVF	Overflow Flag is set to state of twos-complement arithmetic overflow by arithmetic instructions. Overflow Flag is set high if sign bits (most significant bit) of two operands are identical and sign bit of result is different from sign bit of operands. If A, B, and R are sign bits of operands and result, then Overflow Flag is set according to equation $\mathrm{OVF} = (\bar{A} \cdot \bar{B} \cdot R) + (A \cdot B \cdot \bar{R})$ Sign bit is most significant bit for data length selected; thus, if data length is 8 bits, then bit 7 is sign bit; if data length is 16, then bit 15 is sign bit. State of OVF Flag is affected by instructions ADD, DECA, SUBB, RADD, and RADC.	0110
7	CRY	Carry Flag is set to state of binary or decimal carry output of adder by arithmetic instructions. Carry output is derived from most significant bit for data length specified by BYTE Flag. State of CRY Flag is affected by instructions ADD, DECA, SUBB, RADD, and RADC.	0111
8	LINK	Link Flag is included in shift and rotate operations as specified by Shift and Rotate Instructions. Link Flag is unaffected if not selected.	1000
9	IEN	Master Interrupt Enable Flag simultaneously inhibits all five of lowest-priority interrupt levels. No Interrupt Request is serviced unless individual Interrupt Enable Flag for associated Interrupt Request and master Interrupt Enable Flag are high. IEN Flag is set low every time any interrupt (except Level-0) is serviced. IEN Flag is set high by execution of Return To Interrupt Instruction (RTI).	1001
10	BYTE	BYTE Flag selects 8-bit data length when high and 16-bit data length when low.	1010
11 12 13 14	F11 F12 F13 F14	Flags 11 through 14 are general-purpose control flags. Flags 11 through 14 drive PACE output pins and may be used to directly control system functions.	1011 1100 1101 1110
15	High ('1')	Bit 15 is not functional and is always in logic '1' state. Addressing bit 15 with SFLG or PFLG Instruction sets the Level-0 Interrupt Enable high. The Level-0 Interrupt is described in the *Use of Interrupts* paragraph later in this chapter.	1111

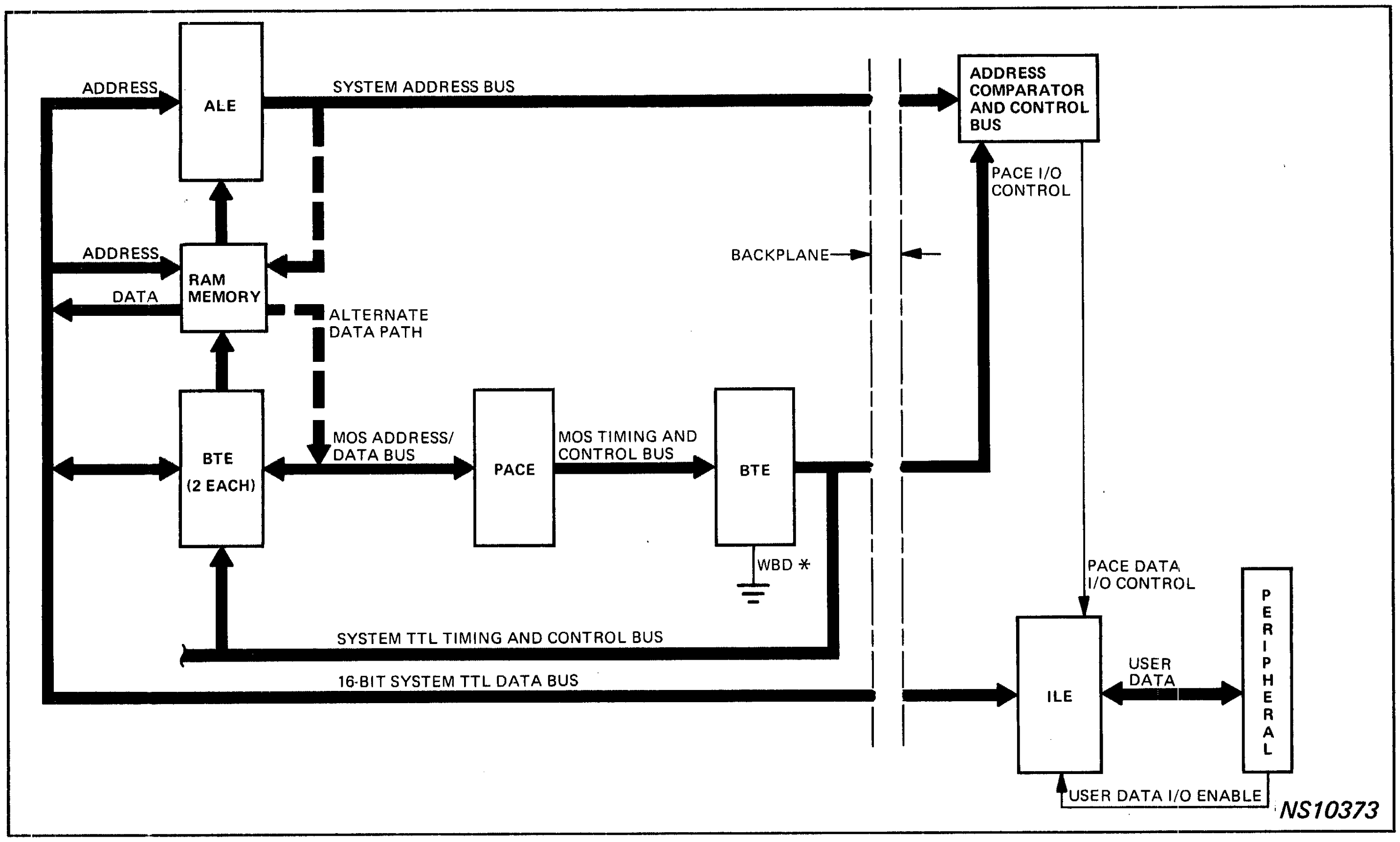


Figure 2-9. Separate Address/Multiplexed Data Bus Structure

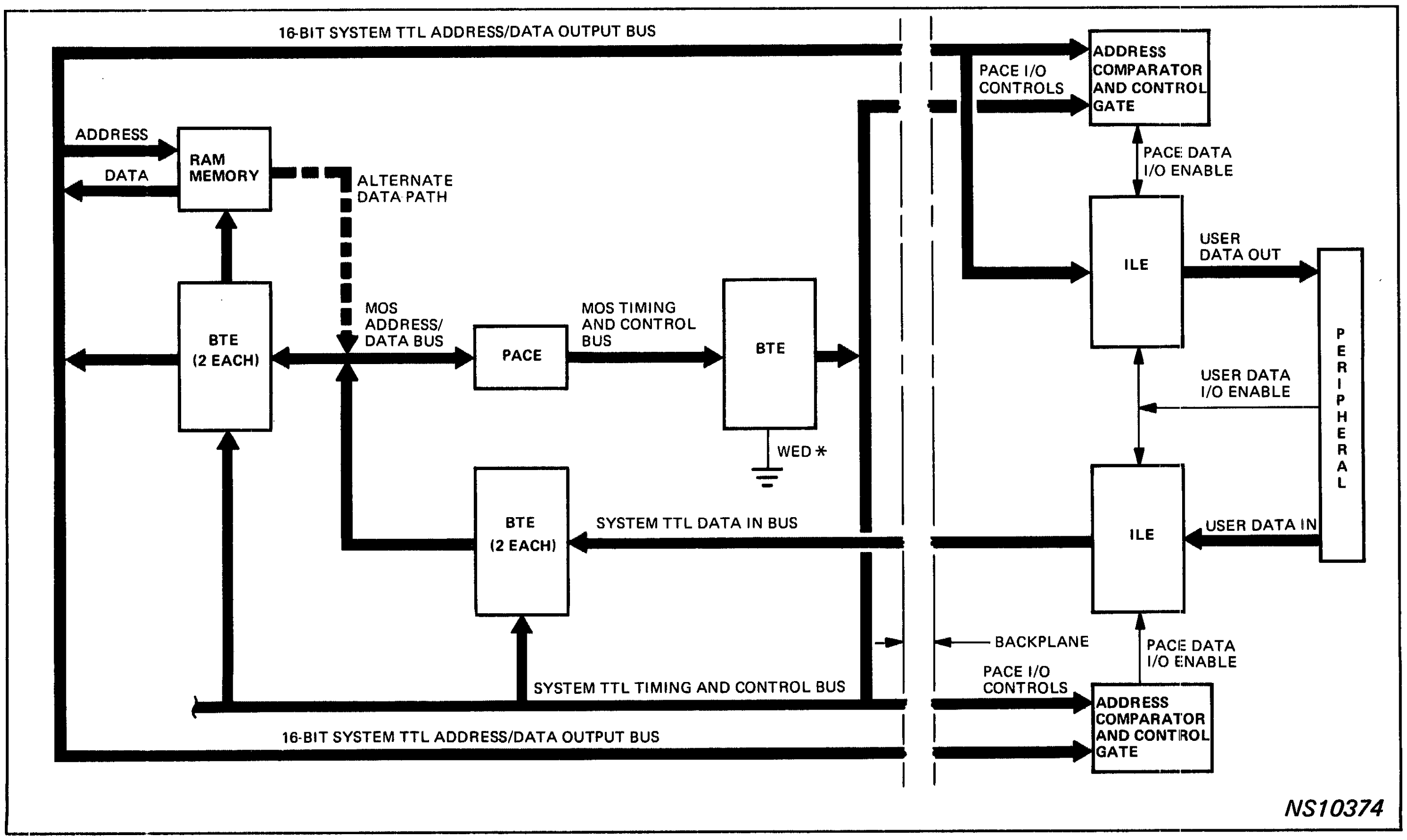


Figure 2-10. Separate Input/Output Bus Structure

2.3.1.1 Direct Addressing

Direct memory addressing has three available modes: base-page, Program-Counter-relative, and indexed. The addressing mode is specified by the *xr* field of the instruction as illustrated in figure 2-11. Figures 2-12 and 2-13 illustrate the three direct-addressing modes.

When the xr field is 00, base-page (page zero) addressing is specified. Two types of base-page addressing are available. The type of base-page addressing selected is determined by the state of the Base-Page Select Signal (BPS) input. When BPS is low (0), the 16-bit memory address is formed by setting bits 8 through 15 to zero and using the 8-bit displacement (disp) field for bits 0 through 7. Thus, the first 256 words of memory (locations 0 through 255) can be addressed. When BPS is high (1), the 16-bit memory address is formed by setting bits 8 through 15 equal to bit 7 of the disp field and using disp for bits 0 through 7. Thus, the first 128 words (0 through 127) and the last 128 words (X'FF80 through X'FFFF) of memory can be addressed. The latter technique is useful for splitting the base page between read/write and read-only memories or between memory and peripheral devices. Consequently, base-page addressing provides a convenient means of accessing data or peripherals.

When the xr field is 01, addressing relative to the Program Counter (PC) is specified. During the PC-relative addressing mode, the memory address is formed by adding the contents of PC to the value of the disp field, which is interpreted as a signed number. The 8-bit disp field is interpreted as a 16-bit value with the bit 7 value used for bits 8 through 15, thereby permitting representation of numbers from −128 through 127.

When the memory address is formed, the PC already is incremented and contains an address value that is one greater than the location of the current instruction. Thus, memory addresses that can be referenced range from 127 locations below through 128 locations above the address of the current instruction.

The indexed (or accumulator-relative) mode of addressing permits any location within the 65,536 word-address-space to be referenced. The disp field, as in PC-relative addressing, is interpreted as a signed value ranging from −128 through 127. The memory address is formed by adding disp to the contents of either Accumulator AC2 (when xr = 10) or Accumulator AC3 (when xr = 11). Table 2-3 presents a summary of the direct addressing modes.

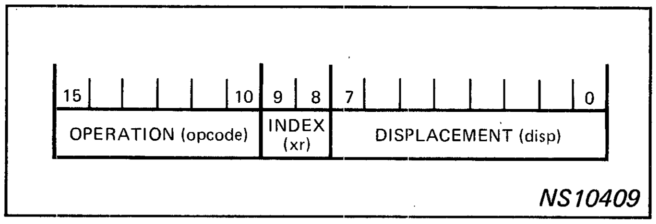


Figure 2-11. Memory-reference Instruction Format

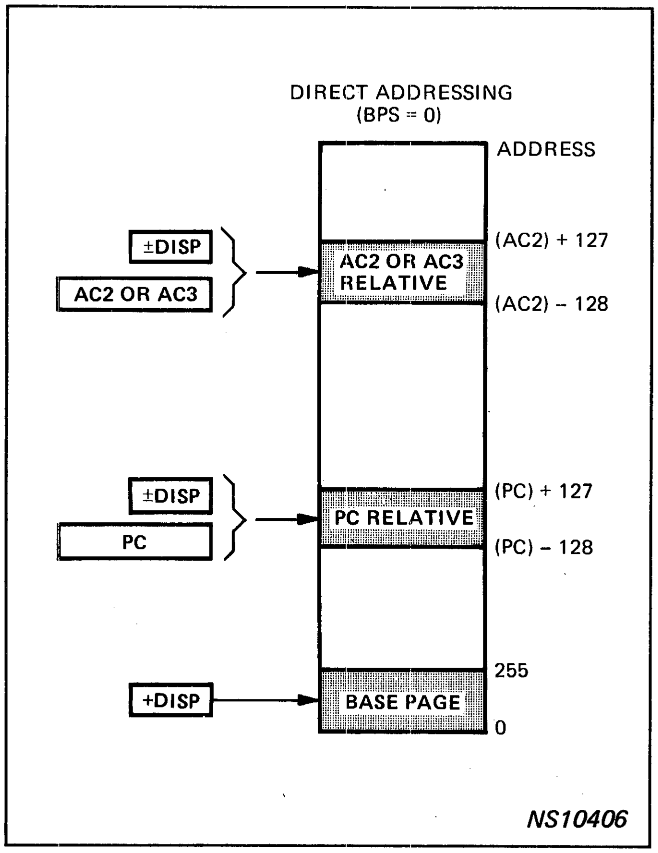


Figure 2-12. Direct Memory Addressing (BPS = 0)

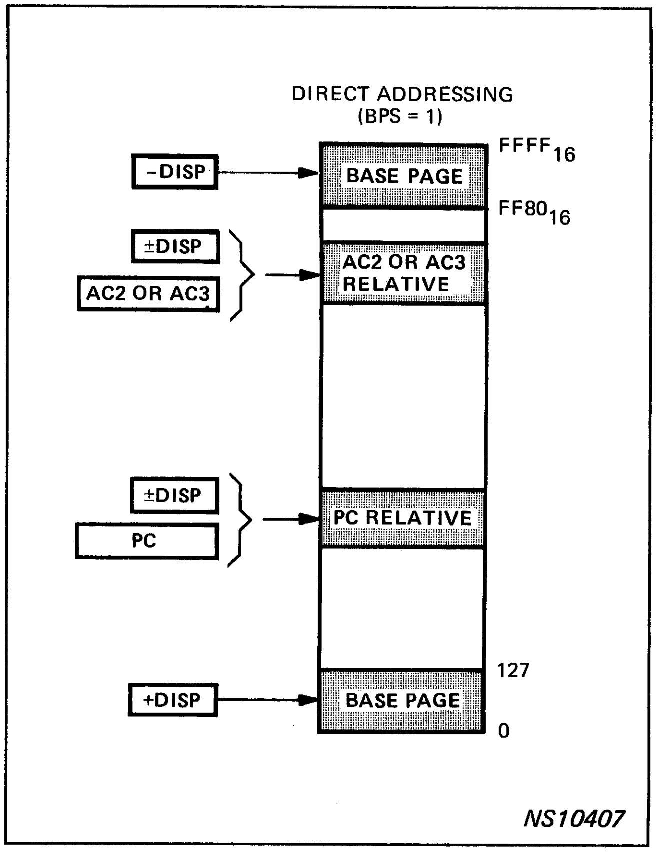


Figure 2-13. Direct Memory Addressing (BPS = 1)

Table 2-3. Summary of Direct Addressing Modes

xr Field	Addressing Mode	Effective Address
00	Base-page	EA = disp
01	Program-Counter-relative	EA = disp + (PC)
10	AC2-relative (indexed)	EA = disp + (AC2)
11	AC3-relative (indexed)	EA = disp + (AC3)
NOTES: 1. For base-page addressing, disp is positive and in range of 000 to 255 when BPS is low (0); or disp is signed number in range of -128 to +127 when BPS is high (1). 2. PC contains value one greater than address of current instruction. 3. For relative addressing, disp range is -128 to +127.		

2.3.1.2 Indirect Addressing

Indirect addressing consists of first establishing an address in the same manner as direct addressing (by either the base-page, PC-relative, or indexed mode). The contents of the memory location at the selected address then are used as the operand address. Figure 2-14 illustrates indirect addressing.

NOTE

The memory addressing modes also are used for peripheral I/O operations. Address space must be divided between memory and I/O devices.

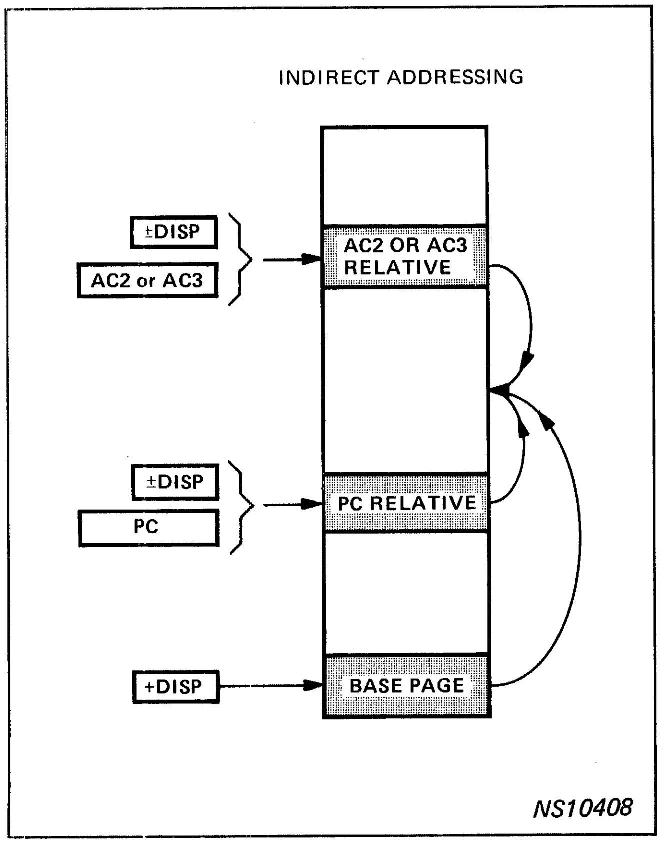


Figure 2-14. Indirect Memory Addressing

2.3.2 INSTRUCTION SUMMARY

The PACE instruction summary is contained in appendix B and provides the instruction mnemonic, meaning, a symbolic representation of the instruction, a verbal description of the operation, the instruction format, the assembler format and the execution time formula for the instruction. Also in appendix B, table B-1 defines the notation and symbols used for symbolic representation of each instruction contained in table B-2, the instruction summary.

2.4 PACE SUPPORT CHIPS

The following paragraphs provide more detailed information concerning the PACE family of support chips.

NOTE

Refer to the latest data sheets to verify the timing and electrical parameters.

2.4.1 SYSTEM TIMING ELEMENT (STE)

The STE is a member of the Blue Chip family of support elements intended specifically for application in PACE microprocessor systems. The STE, together with an external crystal and the application of +5 volts and -12 volts, fulfills the MOS clock signal and substrate bias voltage (V_{BB}) requirements of the PACE microprocessor. In addition, the STE produces TTL clock signals to accommodate user requirements. A functional block diagram of the STE is presented in figure 2-15.

The external series-resonant crystal (frequency = 4.0 mHz, tolerance = ±4.0 kHz) provides frequency control for the STE Oscillator shown in figure 2-12. The STE presents a capacitive load in the 3-pf to 10-pf range to the crystal. The Oscillator Output clocks the Divide-by-two Squaring Circuit, which, in turn, produces complementary timing pulses occurring at one half the crystal resonant frequency. The Divide-by-two Squaring Circuit also can be clocked by connecting an external frequency source to the EXTC Input. When an external frequency source is used, the X1 Input to the STE Oscillator must be grounded. Conversely, if an external crystal is used with the STE, the EXTC Input must be grounded.

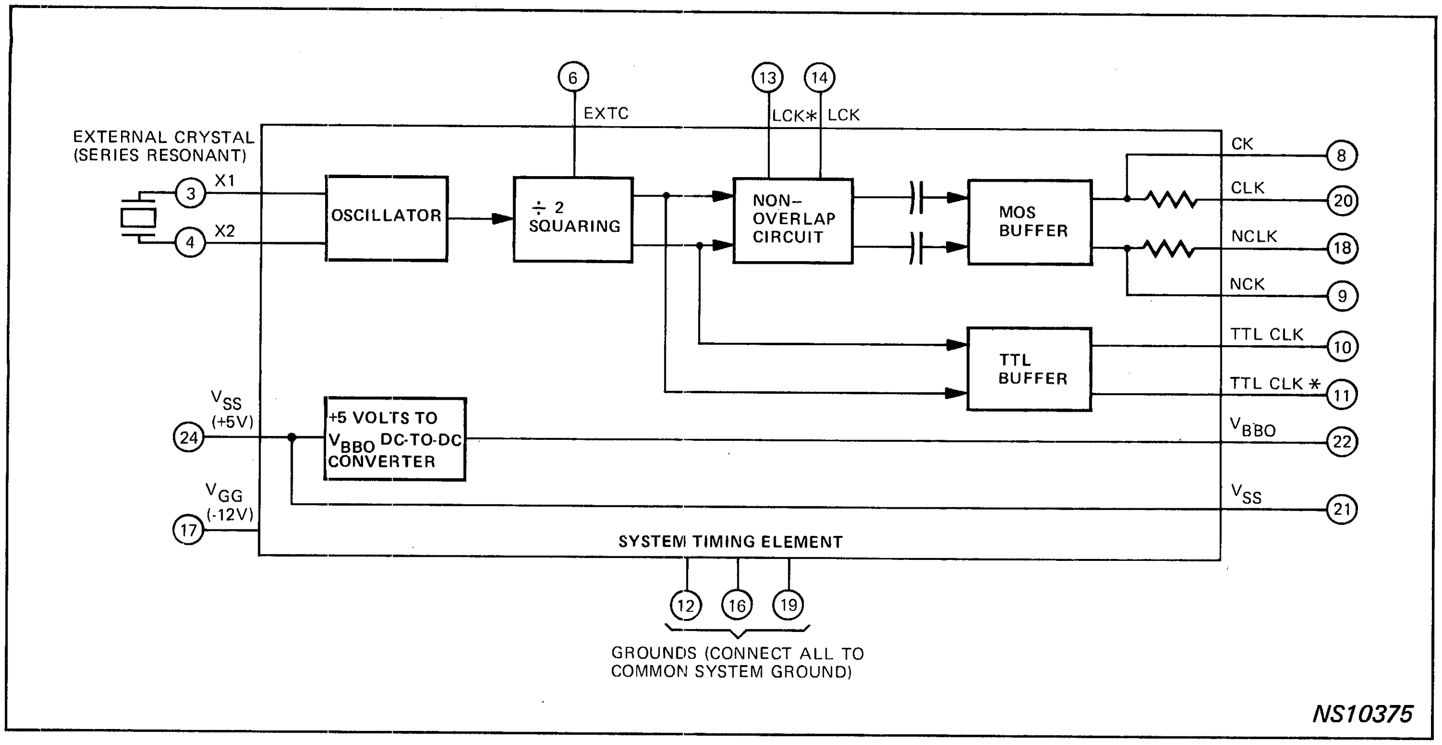


Figure 2-15. STE Functional Block Diagram

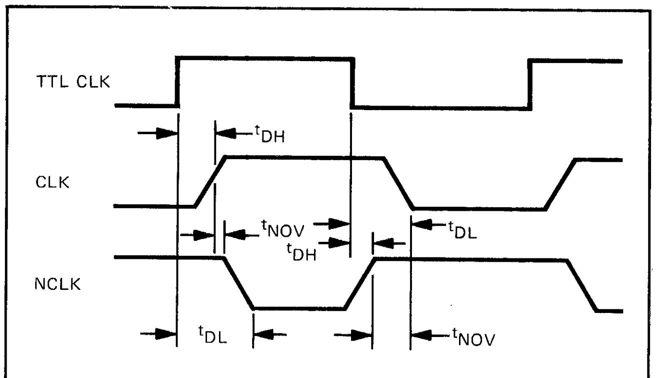
The outputs from the Divide-by-two Squaring Circuit simultaneously are applied to the Non-overlap Circuit and the TTL Buffer. The TTL-compatible TTLCLK* and TTLCLK Outputs from the TTL Buffer lead the MOS-compatible CLK and NCLK Outputs from the MOS Buffer as shown in figure 2-16.

The Non-overlap Circuit consists of a cross-coupled latch with a delay in the feedback path to ensure non-overlapping signals. The delay in the feedback path can be increased by externally connecting a capacitor between the LCK and LCK* Inputs to the STE. The Non-overlap Circuit output timing signals are level-shifted by capacitively coupling the signals to the MOS Buffer, which then translates the signal levels to the MOS levels required by PACE. The value of the coupling capacitors is chosen to optimize performance at 4.0 mHz.

If a crystal or external frequency source is used at a frequency lower than 4.0 mHz, the non-overlap interval increases. However, the MOS Output clock waveforms, with the increased non-overlap interval, still are acceptable to the PACE microprocessor.

The MOS Buffer provides two sets of level-translated clock outputs. One set of outputs (CLK and NCLK) is damped by on-chip series 43-ohm resistors. The 43-ohm damping resistance value is considered optimum for printed circuit card layouts containing clock interconnect lines that are no longer than 2 inches. The damped CLK and NCLK Outputs from the STE have adjacent ground and power lines which should be kept adjacent to clock lines exceeding 1 inch in length on the printed circuit card. Thus, the power and ground lines isolate the clock lines from other on-board signals, as well as from each other, by minimizing capacitive and inductive coupling.

The other set of MOS Buffer clock outputs (CK and NCK) is undamped to accommodate user requirements that may dictate some damping resistance value other than 43 ohms. A typical STE-to-PACE interconnection is illustrated in figure 2-17.



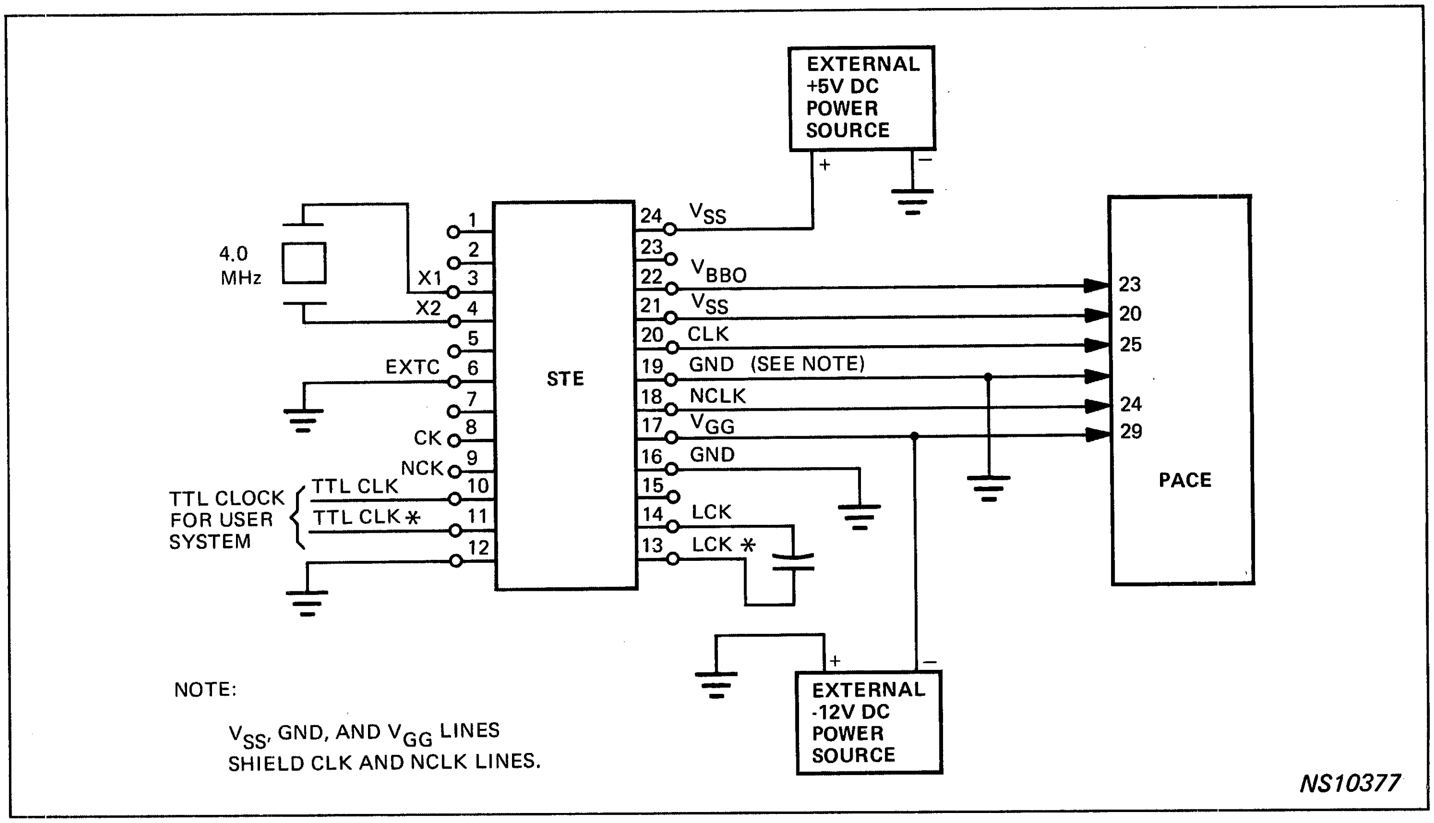
NOTE:
ALL WAVEFORMS REFERENCED TO 10% AND 90% AMPLITUDE POINTS.

WHERE:

t_{DH}	= DELAY TIME HIGH
t_{DL}	= DEALY TIME LOW
t_{NOV}	= NON-OVERLAP TIME

NS10376

Figure 2-16. Relative Timing of STE Output Waveforms



EXTERNAL +5V DC POWER SOURCE
4.0 MHz
X1
X2
EXTC
CK
NCK
TTL CLOCK FOR USER SYSTEM
TTL CLK
TTL CLK*
STE
VSS
VBBO
VSS
CLK
GND (SEE NOTE)
NCLK
VGG
GND
LCK
LCK *
PACE
EXTERNAL -12V DC POWER SOURCE
NOTE:
VSS, GND, AND VGG LINES SHIELD CLK AND NCLK LINES.
NS10377

Figure 2-17. STE to PACE Interconnection

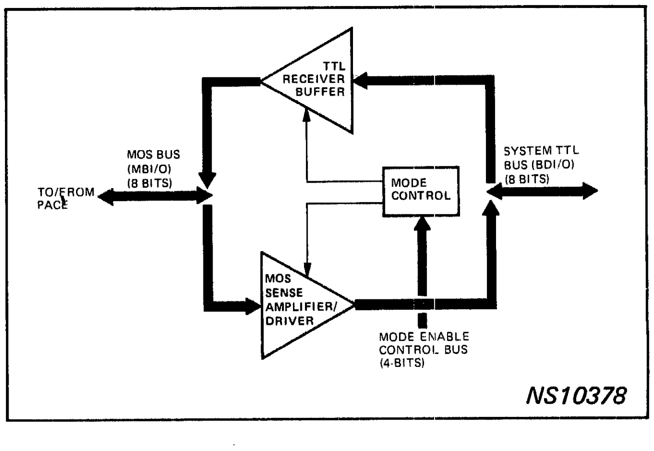
2.4.2 BIDIRECTIONAL TRANSCEIVER ELEMENT (BTE/8)

The BTE is a member of the Blue Chip family of support elements specifically intended for application in PACE microprocessor systems. The BTE provides input/output buffering between the PACE MOS input/output lines and TTL devices. The BTE has a high-fanout TTL capability of up to 30 TTL loads. A functional block diagram of the BTE is presented in figure 2-18.

MOS bus signals from PACE are applied through the Sense Amplifier/Driver to the System TTL Bus. Signals from the System TTL Bus are applied through the TTL Receiver/Buffer to the MOS Bus and, subsequently, to PACE. The BTE Mode Control function decodes input control signals and, in turn, produces signals that set the TTL Receiver/Buffer and the MOS Sense Amplifier/Driver in one of three available operational modes: drive, receive, or high-impedance.

The drive and receive modes are referenced to the TTL outputs (BDI/O). That is, in the drive mode, MOS signals (MBI/O) are accepted by the BTE and output as BDI/O Signals. In the receive mode, the BDI/O Signals are accepted by the BTE and output as MBI/O Signals. When in the high-impedance mode (TRI-STATE®), the BTE MBI/O and BDI/O Outputs assume the high-impedance state.

Figure 2-19 shows a typical system implementation of the BTE. The associated timing waveforms for the BTE control and data signals are shown in figure 2-20. In figure 2-19, one BTE is connected to operate only in the drive mode by grounding the Write Bus Data* pin (WBD*) to the BTE. No connections are made to the BTE Chip Enable Inputs (CE and CE*) or Strobe* Input (STR*) since, as indicated by the BTE truth table in figure 2-19, the CE, CE*, and STR* Input states are *don't care* when the WBD* Input is low. Thus, the BTE operating in the drive-only mode provides buffering for the MOS Timing and Control Bus Signals as shown in figure 2-19. Among the control signals are the Address Data Strobe (NADS), Input Data Strobe (IDS),



TTL RECEIVER BUFFER
MOS BUS (MBI/O) (8 BITS)
TO/FROM PACE
MODE CONTROL
SYSTEM TTL BUS (BDI/O) (8 BITS)
MOS SENSE AMPLIFIER/ DRIVER
MODE ENABLE CONTROL BUS (4-BITS)
NS10378

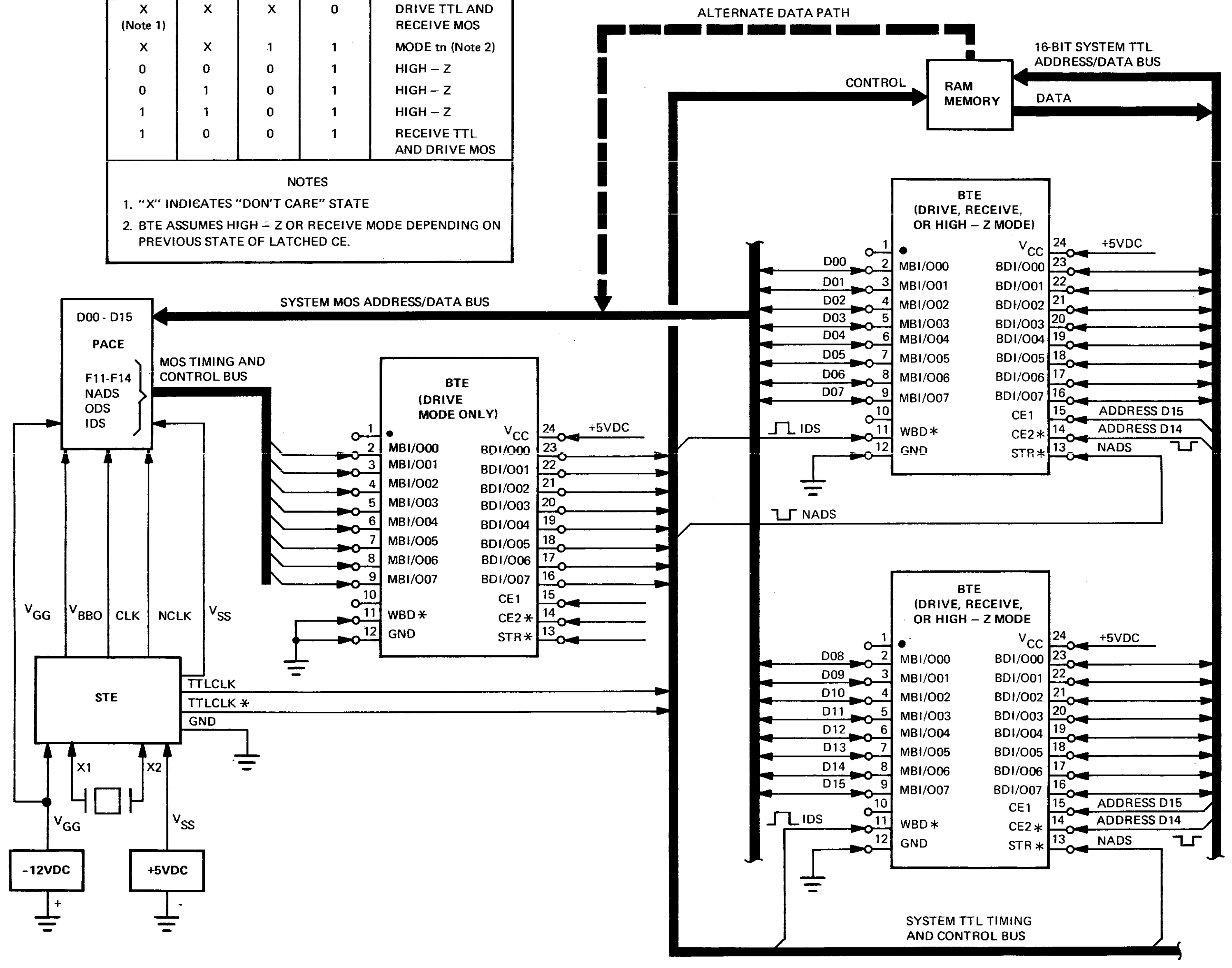
Figure 2-18. BTE Functional Block Diagram

®Trademark of National Semiconductor Corporation

BTE TRUTH TABLE

tn			tn + 1	
CE1	CE2∗	STR∗	WBD∗	BTE MODE
X (Note 1)	X	X	0	DRIVE TTL AND RECEIVE MOS
X	X	1	1	MODE tn (Note 2)
0	0	0	1	HIGH – Z
0	1	0	1	HIGH – Z
1	1	0	1	HIGH – Z
1	0	0	1	RECEIVE TTL AND DRIVE MOS

NOTES
1. "X" INDICATES "DON'T CARE" STATE
2. BTE ASSUMES HIGH – Z OR RECEIVE MODE DEPENDING ON PREVIOUS STATE OF LATCHED CE.



NS10379

Figure 2-19. BTE System Implementation

Output Data Strobe (ODS), and Flags 11 through 14 which are routed from PACE over the MOS Timing and Control Bus to the BTE. The resultant BTE outputs comprise the System TTL Timing and Control Bus Signals, which assume active states in accordance with the current user-generated program instruction executed by PACE.

Operational mode control of the two BTEs interfacing PACE to the System TTL Address/Data Bus is effected by the NADS and IDS Signals and two PACE address bits (Di and Dj). In this example, address bits D14 and D15 are arbitrarily chosen. BTE operational mode control is set during the time address information is present on the System TTL Address/Data Bus (see figure 2-20). When the NADS Signal goes low at the STR✱ Input, during address time, and address bits D14/D15 are applied to the CE1 and CE2✱ Inputs, an on-chip CE Latch either is set or reset in accordance with the states of the CE1 and CE2✱ Inputs. The output state of the CE Latch is used internally to enable the BTE to receive TTL data at the BDI/O pins or assume the high-impedance mode. Following address time, the BTE either drives or receives data onto or from the System TTL Address/Data Bus or assumes the high-impedance mode, as determined by the levels of the previously applied control signals. The BTE truth table in figure 2-19 shows the input control-signal levels required to activate each of the BTE modes.

The purpose of connecting address bits Di and Dj to the CE1 and CE2✱ Inputs in this example is to allow the RAM memory to drive the PACE chip directly over the Alternate Data Path shown in figure 2-19. This eliminates the need for memory buffers in the case of MOS memories. This approach is practical only when the memory and CPU are on the same circuit board, since the MOS bus lines should be kept short and away from high-speed lines. If TTL memory is used, or if buffers are provided, data are returned over the TTL bus and the BTE may be hardwired to the enable state.

2.4.3 INTERFACE LATCH ELEMENT (ILE/8, ILE/16)

The ILE is a member of the Green Chip family of support elements intended specifically for application in PACE microprocessor systems. The ILE is available in both 8-bit and 16-bit versions to provide bidirectional latched interfacing between the System TTL Address/Data Bus and user peripherals. Operation and control are identical for both the 8-bit and 16-bit ILE. The relationships among the various ILE internal functions are illustrated in figure 2-21.

Data directional-control signals from PACE or a user peripheral are applied to the Data Control function shown in figure 2-21. The logical states of the data directional-control signals between PACE and the ILE are determined by the current user-generated program instruction under execution. Control signals generated at the peripheral interface to the ILE determine the direction of data flow between the ILE and the peripheral. The data directional-control signals are interpreted logically by the Data Control function, which, in turn, provides control signal outputs that are applied to the UDI/O Driver, the BDI/O Driver, and the Multiplexer.

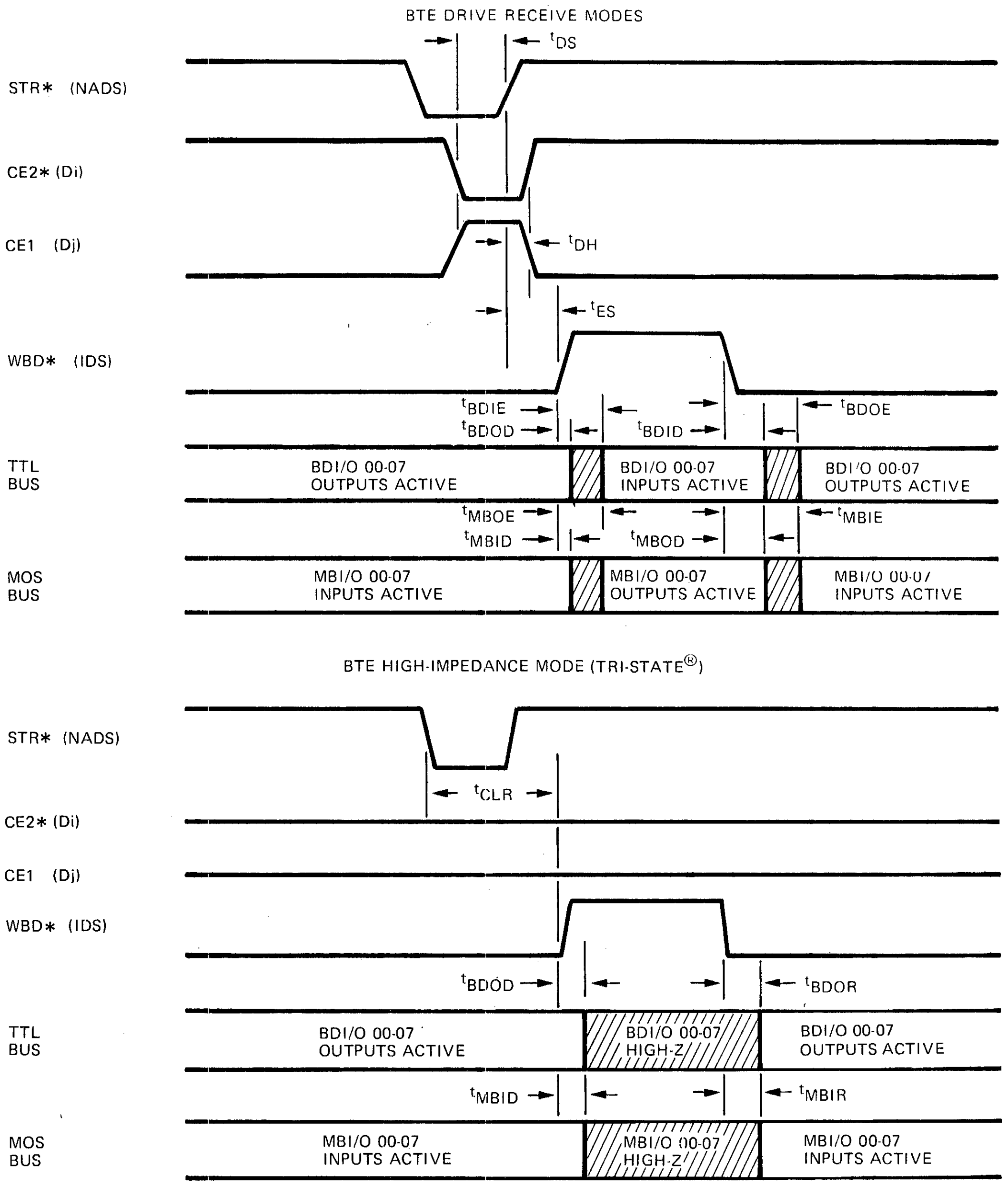
The control signals applied to the Multiplexer determine whether BDI/O or UDI/O data input or the D-type flip-flop output is used as the D-input to the flip-flop. The flip-flop output then follows the D-input upon application of the CLK Signal. Applying the flip-flop output as the D-input during CLK time allows a do-nothing state, thereby eliminating spurious output signals when data input does not change. The control signals applied to the Drivers determine whether the flip-flop output is applied as data to the BDI/O or UDI/O Line, or both. Also, the Driver control signals determine whether data input is accepted from either the BDI/O or UDI/O Line, or both, in accordance with the truth table in figure 2-22.

Figure 2-22 shows a typical system implementation of a 16-bit ILE operating as a bidirectional peripheral interface latch and an 8-bit ILE used only for input of peripheral data. The ILE truth table in figure 2-22 shows the input control signal states required to place the ILE in each of the eight available operational modes. The associated timing waveforms for the ILE control and data signals are shown in figure 2-23.

Data is transferred to/from the ILE BDI/O Lines over the System TTL Address/Data Bus from/to the BTE BDI/O Lines. Some of the same System TTL Timing and Control Bus signals (IDS and NADS) used for BTE operational control are also used for ILE operational control. The control signals and designs of the elements comprising the PACE concept permit compatible operation among the family of PACE support chips with a minimum of design effort. For example, when data are transferred from the ILE-16 through the BTE to PACE, the active PACE IDS Signal permits both the ILE-16 and BTE to be oriented operationally for data transfer in the desired direction (to PACE).

Prior to application of the IDS Signal, the BTE CE Latch must be set appropriately and the DM8136 Address Comparator Output must be driven high during address time. Address bits Di and Dj, applied to the BTE CE and CE✱ control inputs, are set high and low, respectively. When NADS goes low in the center of address time, the BTE on-chip CE Latch Output is set to permit the BTE to transfer data from the ILE to PACE during data time. Also, during address time, address information and NADS are applied to the DM8136 Address Comparator. If the address information compares to the fixed address applied to a second set of Address Comparator Inputs, the Address Comparator Output is latched high when NADS goes low. Thus, both BTE and ILE-16 now are ready to transfer data toward PACE when IDS occurs.

When IDS goes high during data-input time, the NANDed combination of IDS and the Address Comparator high output causes a BDOEN✱ Input to be applied to the ILE-16. At that time, BDINEN✱ is high and the user peripheral data-transfer control circuits (not shown in figure 2-22) must maintain high UDINEN✱ and UDOEN✱ control inputs to the ILE-16. As shown in the ILE truth table of figure 2-22, all ILE-16 control signal conditions therefore are met to permit data transfer from the ILE-16 over the



WHERE:

t_{DS}	= DATA STROBE TIME
t_{DH}	= DATA HOLD TIME
t_{ES}	= ENABLE STROBE TIME
t_{BDIE}	= TTL BUS DATA INPUT ENABLE TIME
t_{BDOD}	= TTL BUS DATA OUTPUT DISABLE TIME
t_{BDID}	= TTL BUS DATA INPUT DISABLE TIME
t_{BDOE}	= TTL BUS DATA OUTPUT ENABLE TIME
t_{MBOE}	= MOS BUS OUTPUT ENABLE TIME
t_{MBID}	= MOS BUS INPUT DEISABLE TIME
t_{MBOD}	= MOS BUS OUTPUT DISABLE TIME
t_{MBIE}	= MOS BUS INPUT ENABLE TIME
t_{CLR}	= CLEAR TIME
t_{BDOR}	= TTL BUS DATA OUTPUT RECOVERY TIME
t_{MBIR}	= MOS BUS INPUT RECOVERY TIME

NS10380

Figure 2-20. BTE Operational Modes Timing Diagram

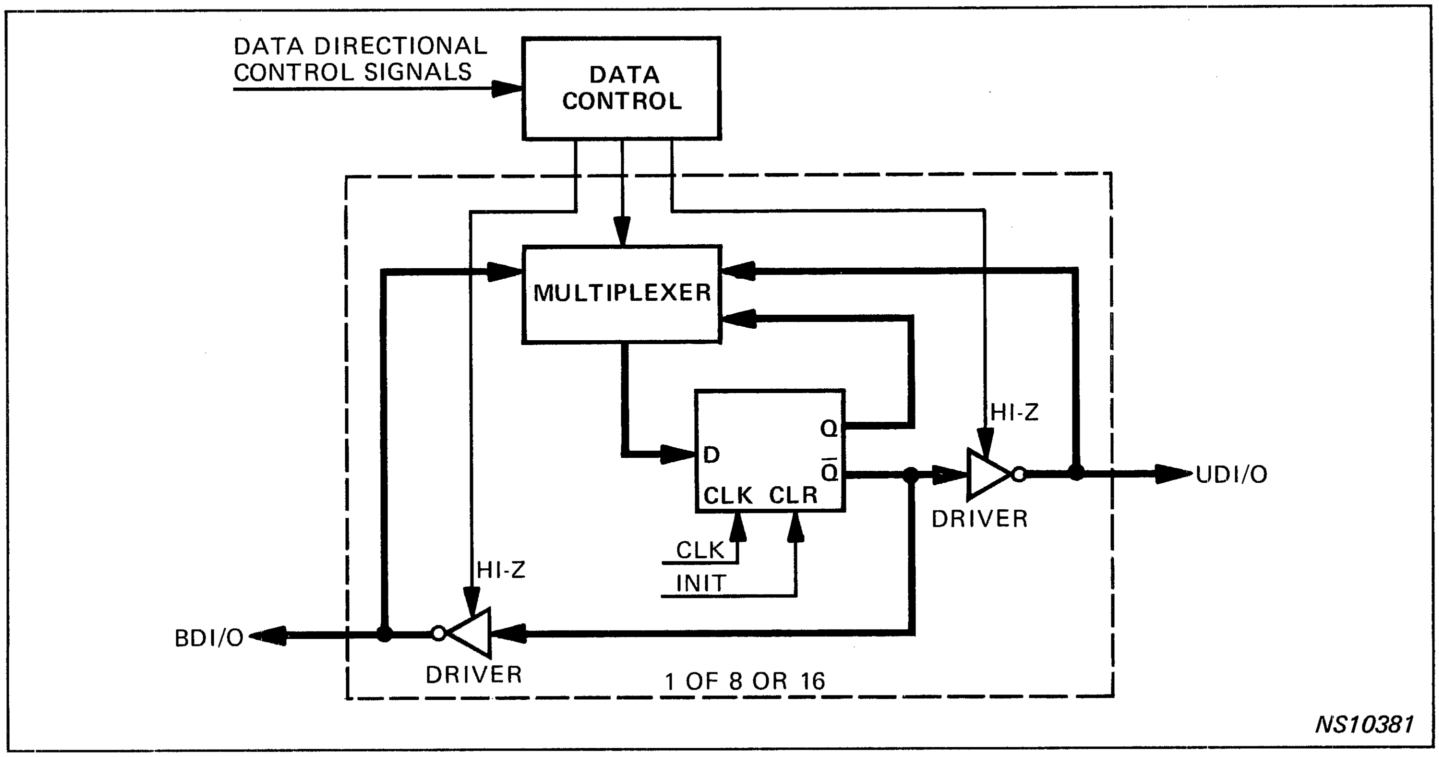


Figure 2-21. ILE Functional Block Diagram

BDI/O Lines to the BTE. At the same time, the IDS Signal applied to the BTE WBD* control input enables the BTE to accept and transfer the ILE-16 output data to PACE.

Data transfers from PACE through the BTE and, then, into the ILE-16 are accomplished by the PACE ODS Signal. During address time, the Address Comparator again compares the address on the System TTL Address/Data Bus to the fixed address input. If the address compares, the Address Comparator Output is latched high when NADS goes low. Then, during data time, the ODS Signal goes low (overriding all other BTE control inputs) and permits the BTE to transfer data from PACE onto the System TTL Address/Data Bus. At the same time, the high ODS Signal is NANDed with the Address Comparator high output to produce a low BDINEN* Input to the ILE-16. The low BDINEN* ILE-16 control input permits the ILE-16 to latch the data internally on the BDI/O Lines. When the next NADS Signal occurs during the following address time, the Address Comparator attempts to make an address comparison. If the address does not compare, the Address Comparator Output is latched low. Thus, the ILE-16 does not react to subsequent data or control signals from the PACE microprocessor until another successful address comparison occurs. Furthermore, without an address comparison, the BDI/O Inputs of the ILE are placed in a high-impedance state, thereby eliminating loading of the BDI/O Bus. When the user peripheral is ready to accept the data latched into the ILE-16, the user peripheral data-transfer control circuits must supply a low UDOEN* Signal to the ILE-16. Since control of data transfers between the ILE and user peripheral varies with the peripheral device and application, no attempt is made to detail the peripheral data-transfer control circuits.

Operation of the ILE-8 is identical to ILE-16 operation in the bidirectional mode. However, the ILE-8 in figure 2-22 is shown connected for input data transfers only. For operation of the ILE-8 shown in figure 2-22, user-generated software may use the Set Flag Instruction described in appendix B to set the PACE BYTE Flag high. Thus, the PACE microprocessor ignores the high-order bits (D08-D15) and operates on only the low-order bits (D00-D07) in accordance with the detailed instruction descriptions provided in appendix B. More information on 8-bit data processing is presented later in this chapter.

Since the ILE internally latches the most recent data transfer, usually during standard program execution, it is not necessary to use the ILE CLEAR Input. However, if system design dictates a necessity to clear the ILE, the circuit shown in figure 2-24 can be used to generate an INIT Signal that clears the ILEs during system power-up.

2.4.4 ADDRESS LATCH ELEMENT (ALE)

The ALE is a member of the Green Chip family of support elements specifically intended for application in PACE microprocessor systems. The ALE is available in either an 8-bit or 16-bit version to provide address latching for memory devices that do not have on-chip address latches. Figure 2-25 contains a functional block diagram of the ALE/16. Functionally, the ALE/16 is the same as the ALE/8. The only difference between the two devices is that the ALE/8 has an 8-bit capacity while the ALE/16 has a 16-bit capacity.

Input data (BD Lines) are applied to the ALE/16 BDI data-input lines (see figure 2-25). If either the BDIEN1* (Byte 0)

ILE TRUTH TABLE

CLEAR	BDOEN	UDOEN	BDIEN	UDIEN	BDI/O	UDI/O	COMMENTS
0	0	1	1	1	Q	HIGH – Z	OUTPUT BDI/O BITS FROM ILE
0	1	0	1	1	HIGH – Z	Q	OUTPUT UDI/O BITS FROM ILE
0	0	0	1	1	Q	Q	OUTPUT DATA TO BOTH BUSES
0	1 NOTE 1	1	1	1	HIGH – Z	HIGH – Z	STORE DATA WITH OUTPUTS IN HIGH – Z STATE
0	X	X	0	1	DATA	QN	ENTER BDI/O BITS TO ILE
0	X	X	1	0	QN	DATA	ENTER UDI/O BITS TO ILE
0	X	X	0	0	DATA	DATA	ENTER UDI/O AND BDI/O BITS
1	X	X	X	X	X	X	TO ILE (LOGIC HIGH AT EITHER INPUT DOMINATES)
1	X	X	X	X	X	X	CLEAR

NOTE
1. "X" INDICATES "DON'T CARE" STATE

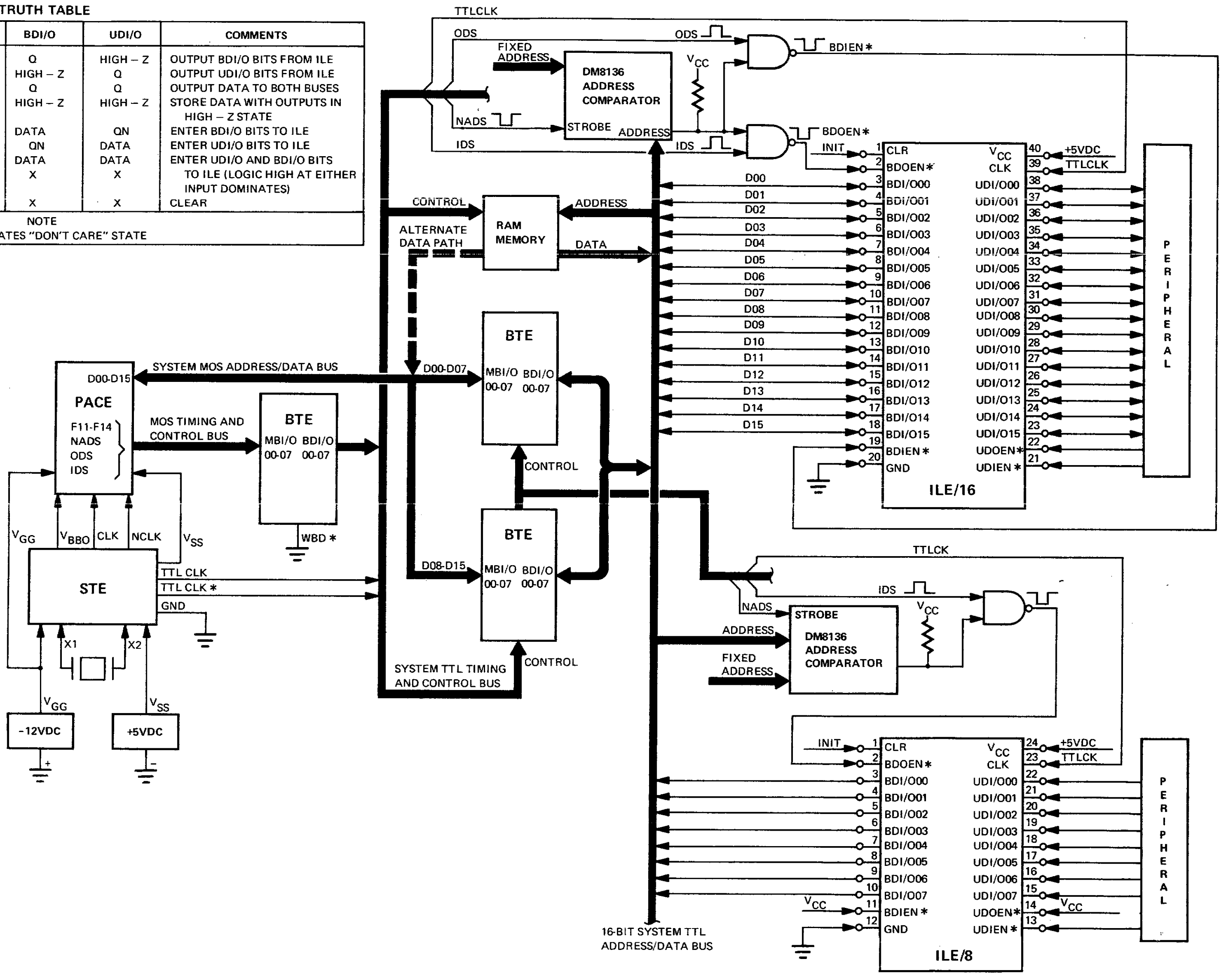


Figure 2-22. ILE System Implementation

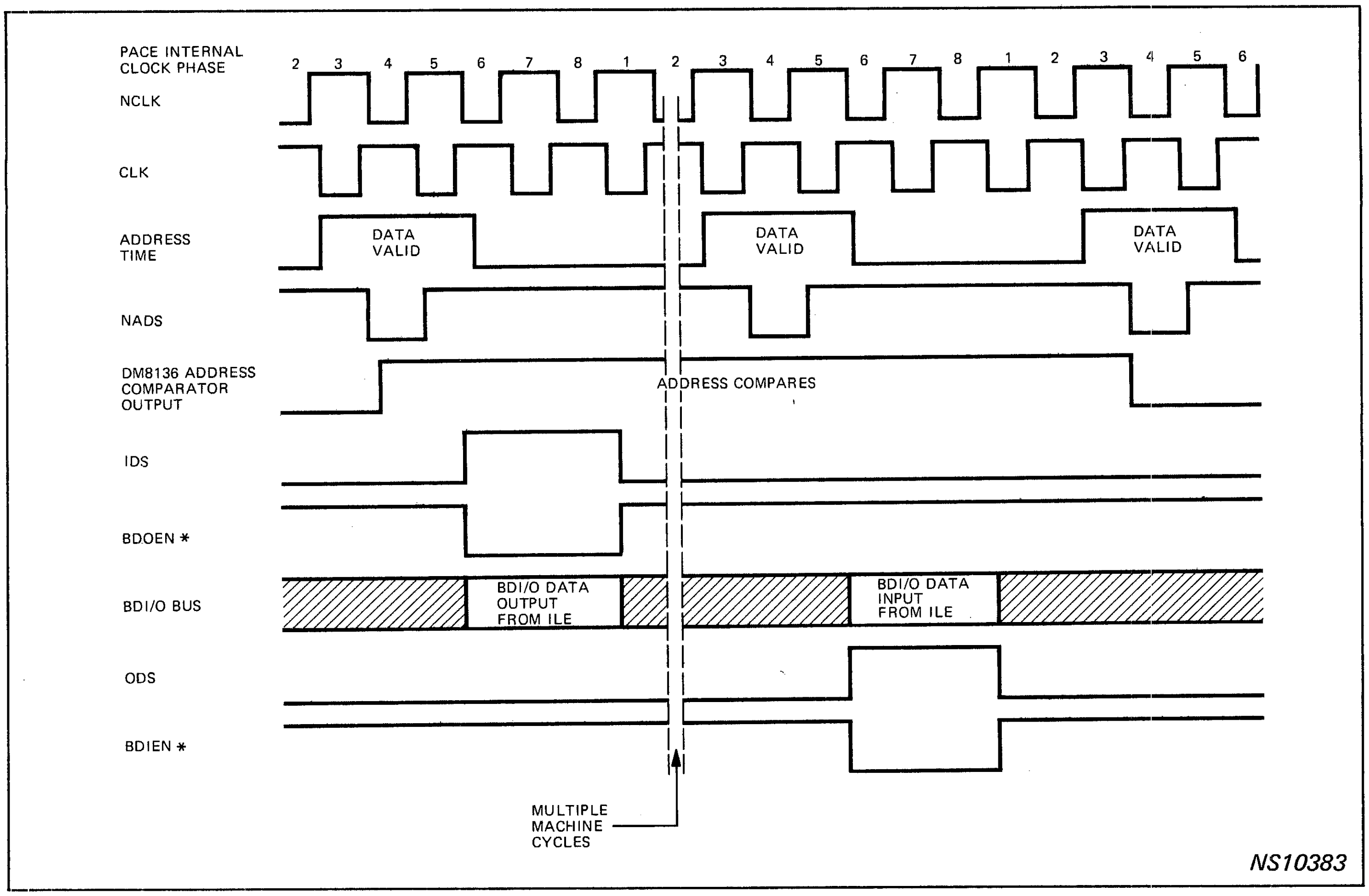


Figure 2-23. ILE Idealized BDI/O Timing Diagram

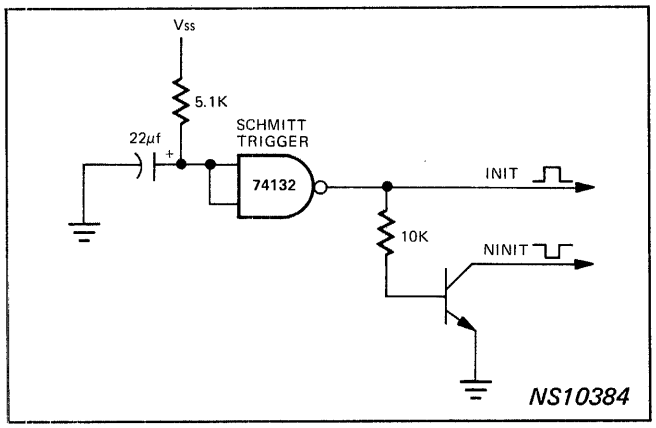


Figure 2-24. Circuit For Generating INIT

or BDIEN2✱ (Byte 1) Signal to the ALE/16 is low, the associated Data Control function passes the BDI data to the flip-flop D-input. If either the BDIEN1✱ or BDIEN2✱ Signal is high, the associated Data Control function passes the flip-flop output level to the D-input. Thus, when the data inputs are inhibited, the Data Control functions permit the D-flip-flops to maintain the status of the data previously input even though CLK Signals are continuously applied. The Data Control function eliminates the necessity to use clock gating circuits, thereby preventing the danger of false clocking inherent to clock gating circuits. The positive-going transition of the CLK Signal clocks the D-flip-flops. The resultant flip-flop outputs are applied through TRI-STATE® Output Buffers. When a high UDOEN✱ Signal is applied to the ALE/16, the Output Buffers are driven to the high-impedance state. Application of a high Clear Byte 0 or 1 Signal causes the outputs of the flip-flops in the associated byte to set the corresponding ALE address outputs (UDO) low. If clearing the ALE is necessary, the circuit shown in figure 2-24 can be used to generate an INIT Signal during system power-up.

Figure 2-26 shows a typical system implementation of an ALE/16 and an ALE/8. Since operation is the same for the ALE/8 and ALE/16, only ALE/16 operation is described.

Address information from PACE is passed through the BTEs onto the System TTL Address/Data Bus, as previously described, and applied to the ALE BDI00 through BDI15 Input Lines. In the middle of the time that address information is valid, PACE provides a low NADS Signal (see figure 2-27) that is routed to the ALE/16 BDIEN1✱ and BDIEN2✱ Inputs. The low BDIEN1✱ and BDIEN2✱ Inputs permit the ALE/16 to latch the address information (BDI/O00-15) present on the BDI00 through BDI15 Input Lines. When NADS terminates (before the end of address time), the resultant high signal prevents the ALE/16 from accepting any future data placed on the System TTL Address/Data Bus.

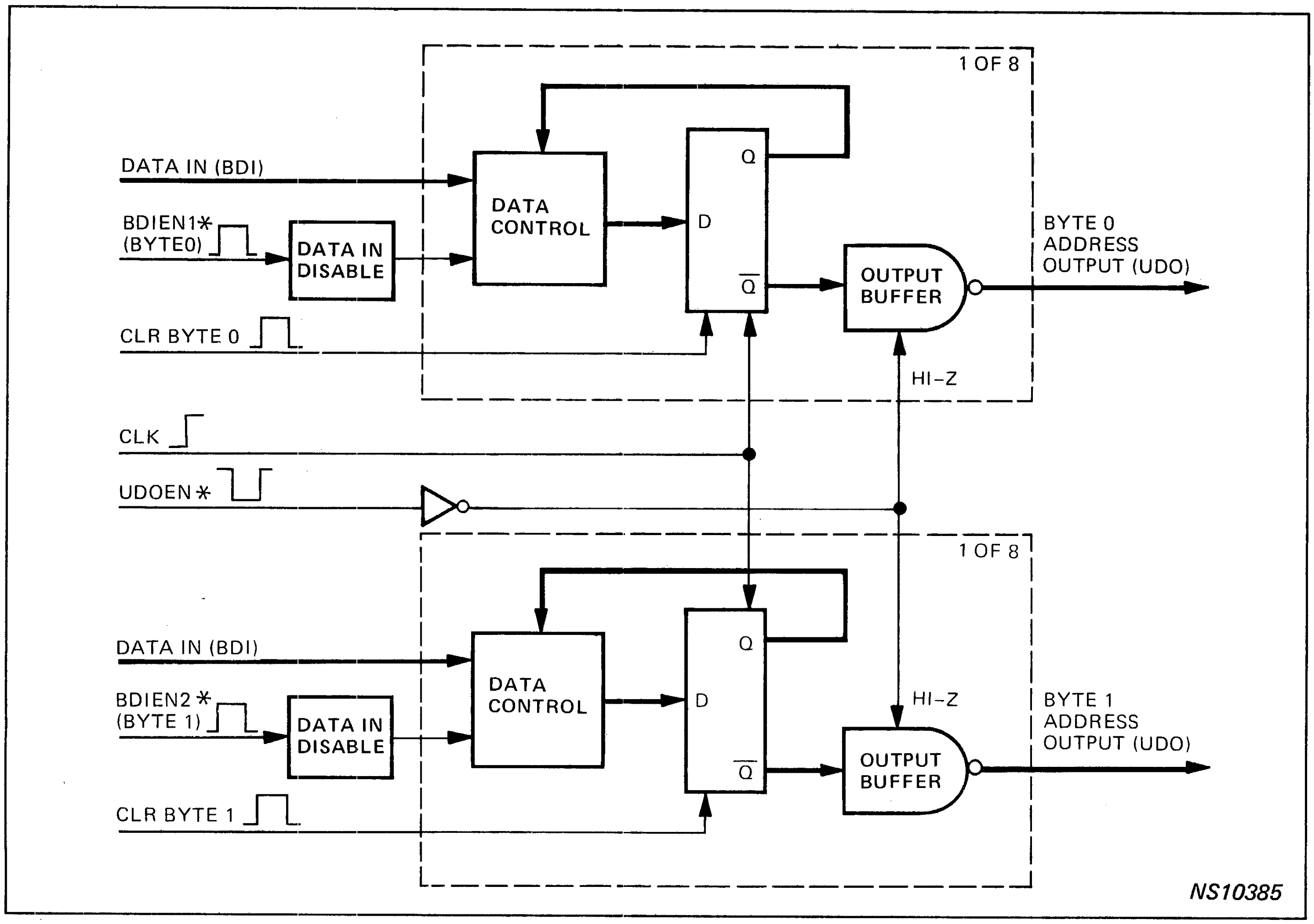


Figure 2-25. ALE/16 Functional Block Diagram

Ordinarily, the UDOEN* Input to the ALE/16 is connected to ground. Thus, during address time, the UDO00-15 Output Lines follow the address input data on the BDI00-15 Lines. However, for applications such as DMA, the ALE/16 UDOEN* Input can be used to drive the UDO00-15 Outputs to a high-impedance state.

After address time, the Memory Device then is enabled by the high IDS Signal on the System TTL Timing and Control Bus to pass the addressed data to the PACE Data Input Buffers in the manner already described for BTE operation.

2.4.5 RAM

The IPC-16A/504 RAM provides 256-by-4 bits of static NMOS read/write memory. In addition, the RAM chip contains address latches and chip-enable latches which permit the RAM to be used without an ALE. Figure 2-28 shows a functional block diagram of the RAM.

Address bits AD00 through AD04 (see figure 2-28) are applied to the Row Address Register while address bits AD05 through AD07 are applied to the Column Address Register. When a Latch* Signal is applied to the RAM, address bits 00 through 07 are latched into the Row and Column Address Registers. The address bits stored in the Row Address Register are applied to the Row Select function. The Row Select function then selects one of the 32 rows (each containing 32 bits) for input to or output from the 1024-bit Static RAM Array. The address bits stored in the Column Address Register are applied to the Column Select function. The Column Select function selects one of eight groups of 4 bits for input to or output from the 32-bit row selected by the Row Select function. The four selected bits then are passed by the Input/Output Circuits to the Output Drivers during a Memory Read Cycle. During a Memory Write Cycle, the four input data bits (DI01 through DI04) are written into the selected address by way of the Input/Output Circuits.

At the same time the address bits are latched, the Chip Enable* Input (CE*) is latched into the CE Register by the Latch* Signal. The CE and CE* Outputs from the CE Register are applied to the Input Data Control and Output Enable NOR gate, respectively. Once the address bits and CE* Signal are latched into the appropriate registers, the RAM is ready to execute either a Memory Read or Write Cycle. The Memory Read and Write Cycles are selected by the state of the Write Enable (WE) and Output Enable (OE) Signals.

ALE TRUTH TABLE

t_n				t_n+1
CLR	UDOEN*	BDIEN*	BDI DATA	UDO DATA
0	0	1	SEE NOTE X	Qn
0	0	0	1	1
0	0	0	0	0
0	1	X	X	HI-Z
1	0	X	X	0

NOTE: "X" INDICATES "DON'T CARE" STATE

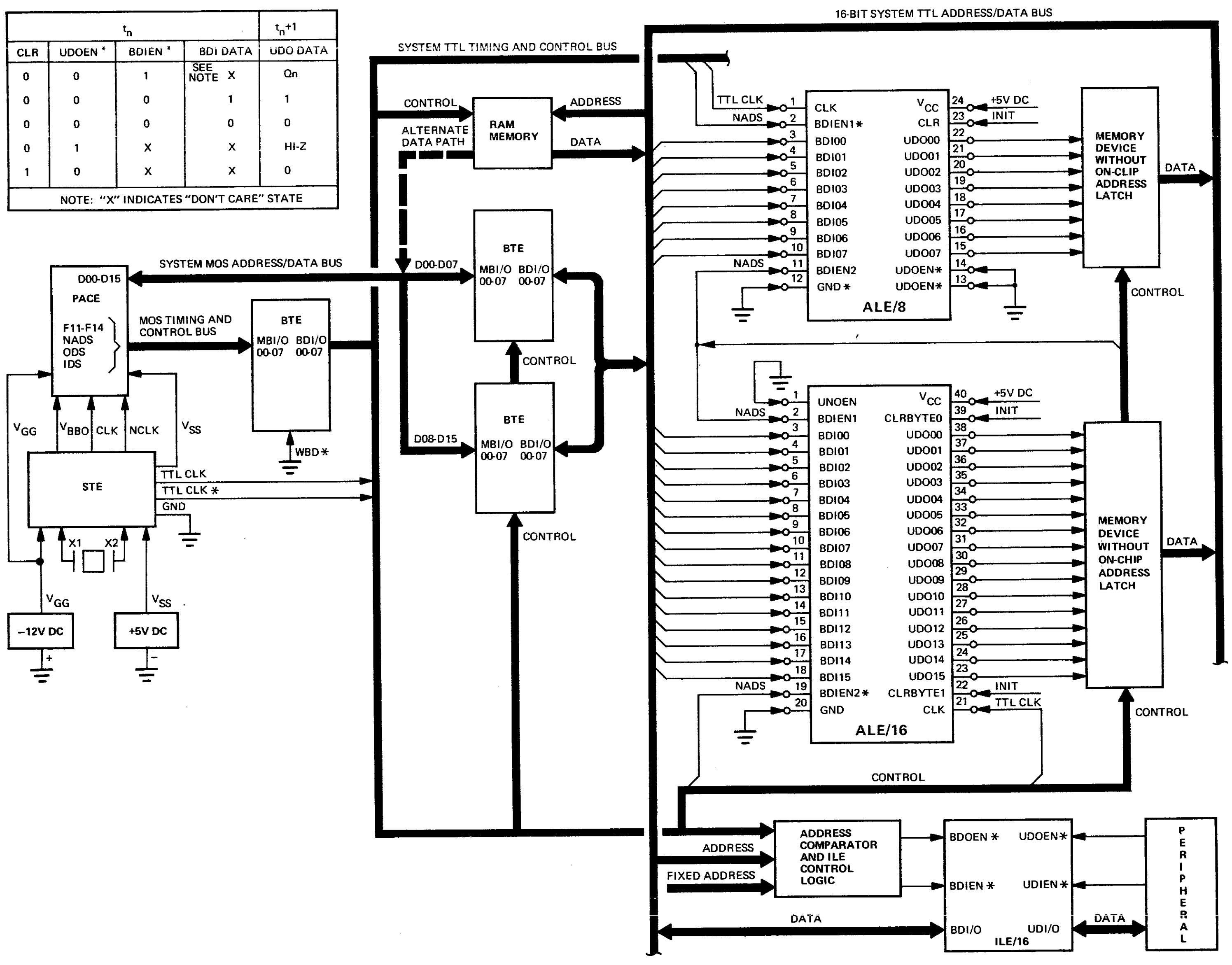


Figure 2-26. ALE System Implementation

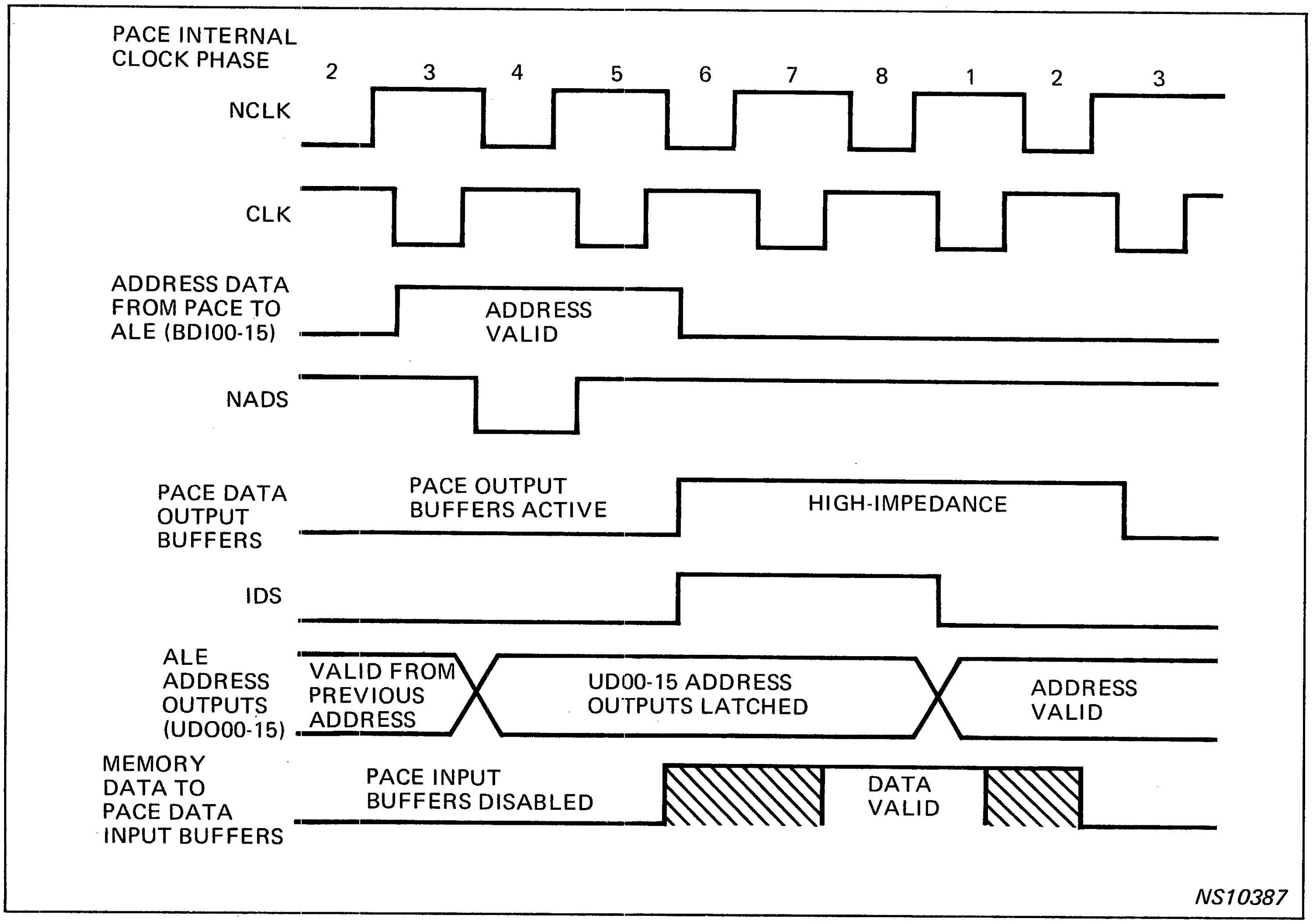

PACE INTERNAL CLOCK PHASE
2
3
4
5
6
7
8
1
2
3
NCLK
CLK
ADDRESS DATA FROM PACE TO ALE (BDI00-15)
ADDRESS VALID
NADS
PACE DATA OUTPUT BUFFERS
PACE OUTPUT BUFFERS ACTIVE
HIGH-IMPEDANCE
IDS
ALE ADDRESS OUTPUTS (UDO00-15)
VALID FROM PREVIOUS ADDRESS
UD00-15 ADDRESS OUTPUTS LATCHED
ADDRESS VALID
MEMORY DATA TO PACE DATA INPUT BUFFERS
PACE INPUT BUFFERS DISABLED
DATA VALID
NS10387

Figure 2-27. Idealized ALE Address and Memory Data Transfer Timing Diagram

For a Memory Read Cycle, the WE Input remains low and the OE Input goes high. The low WE Input inhibits the Input Data Control from accepting any data input over the DI01 through DI04 Lines. The high OE Signal input is inverted and the resultant low signal, together with the low CE✱ and WE Signals, causes the Output Enable NOR gate to provide a high enabling signal to the Output Drivers. The Output Drivers then pass the addressed data from the RAM Array onto the System TTL Address/Data Bus.

For a Memory Write Cycle, the WE Input goes high and the OE Input remains low. The low OE Signal input is inverted and the resultant high signal inhibits the Output Enable NOR gate. The resultant Output Enable NOR gate low-output signal causes the Output Drivers to assume a high-impedance output state. The high WE Signal and the high CE Signal output from the CE Register enable the Input Data Control to accept the DI01 through DI04 Input data bits. The input data bits then are transferred to the Input/Output Circuits. The Input/Output Circuits permit the 4 data bits to be written into the group of 4-bit addresses that are selected by the Row and Column Select functions as previously described. Figure 2-29 shows a typical system implementation of a RAM and a ROM. The two BTEs driving the System TTL Address/Data Bus are controlled by address bits D14 and D15 as described for BTE operation. The RAMs do not use the MOS Address/Data Bus as an alternate path but, by way of the optional Output Buffers, use the System TTL Address/Data Bus. The RAM and ROM both are shown in detail in figure 2-29 to illustrate a typical system implementation of high and low memory. The ROM is used for low memory and includes the base-page address range of X'0000 to X'00FF (000_{10} to 255_{10}). The RAM is used for high memory with an address range of X'FF00 to X'FFFF. If the PACE Base-Page Select Signal (BPS) input is driven high externally, the first 128 words of memory (addresses X'0000 to X'0080) and the last 128 words of memory (addresses X'FF7F to X'FFFF) comprise the base page. Thus, the memory base page of the system shown in figure 2-29 may reside entirely in ROM or be split between ROM and RAM.

As previously described, RAM operation is controlled by the state of the CE✱, WE, and CE Inputs. In figure 2-29, the CE✱ Inputs to the four RAMs are connected to the inverted D15 address bit from PACE. When a high memory address (X'FFXX) is placed on the System TTL Address/Data Bus by PACE, the D15 address bit is high (see figure 2-30). Consequently, when the RAMs are addressed (X'FFXX), the inverted version of D15 provides a low

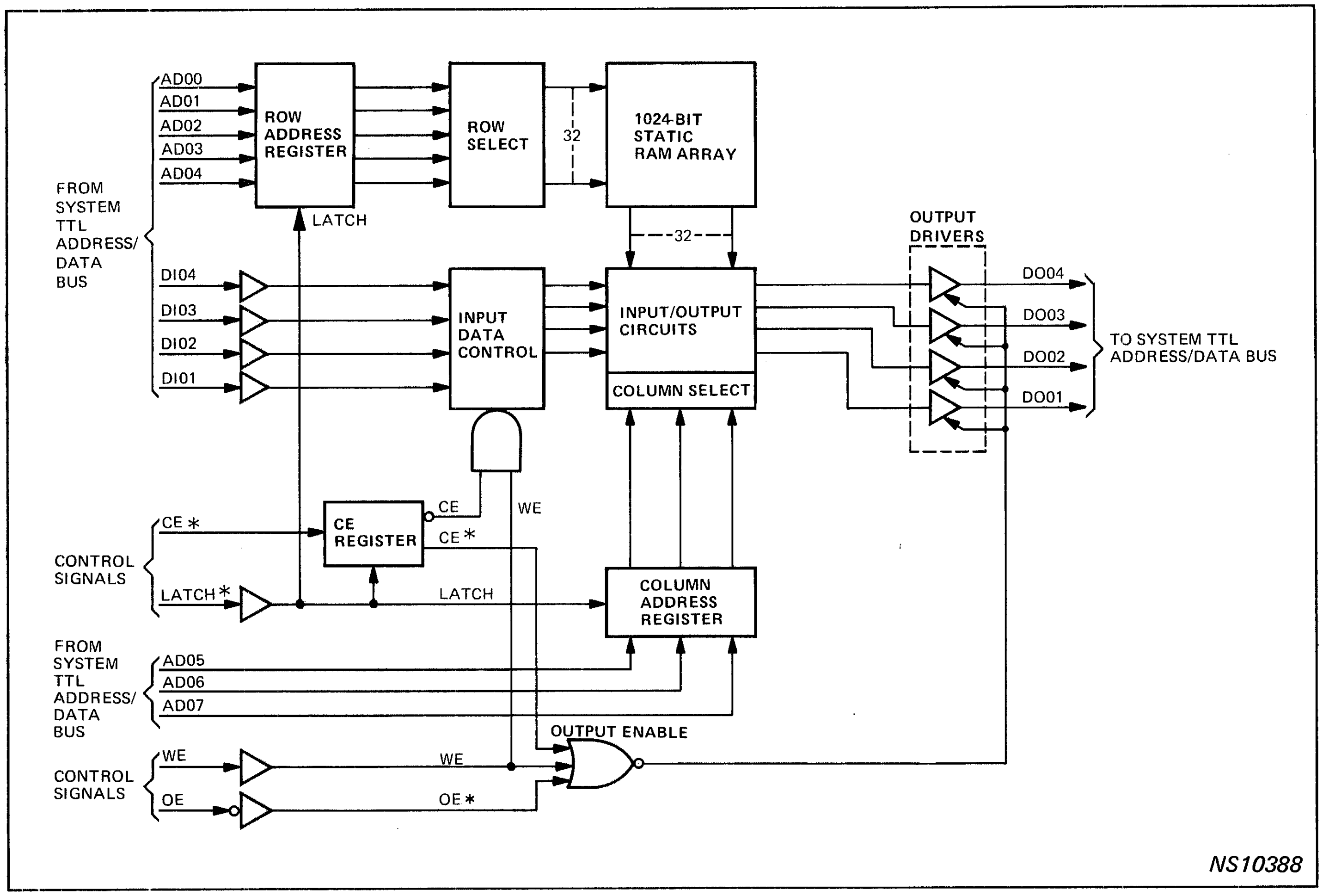


Figure 2-28. RAM Functional Block Diagram

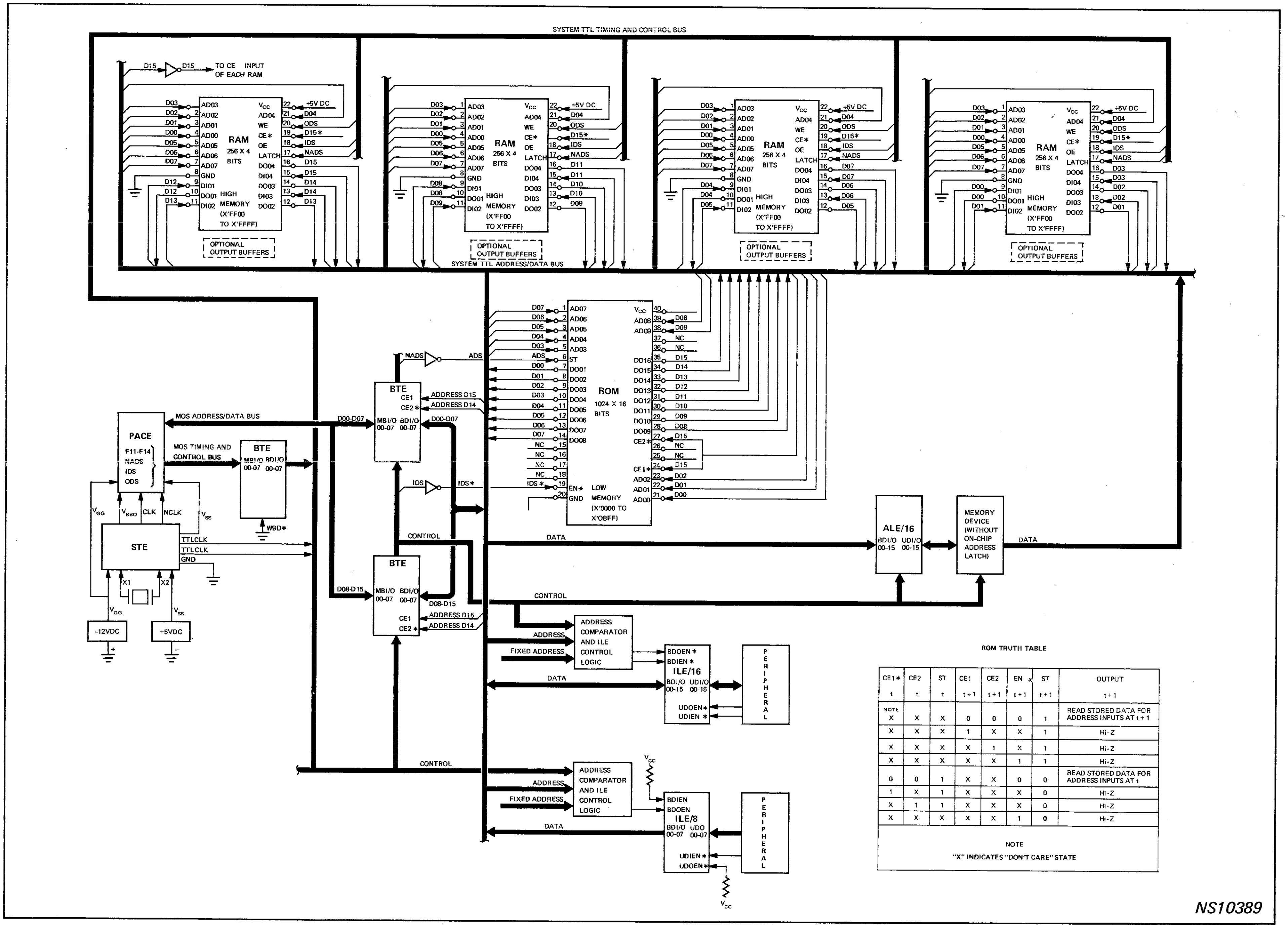
enabling signal to the CE✲ Inputs of the RAMs. Address bit D15 (without inversion) enables the ROM at the CE1✲ and CE2✲ Inputs when low memory is addressed (X'0XXX). Since the D15 address bit is low when the ROM is addressed, the RAM is inhibited (by D15 inversion to a high). Thus, address bit D15 provides a means of address differentiation between the low and high memories of the system shown in figure 2-29. Table 2-4 shows how the D14 and D15 address bits control BTE, RAM, and ROM operation.

When PACE addresses high memory, the address is driven by the BTEs onto the System TTL Address/Data Bus. The first eight (least significant) address bits (D00-D07) are applied to the AD00 to AD07 Inputs of each of the four RAMs. During address time (see figure 2-31), PACE supplies an NADS Signal that is routed over the System TTL Timing and Control Bus to the Latch Input of each RAM. The low NADS Signal at the Latch Inputs enables the RAMs to accept the address bits present at the AD00 to AD07 Inputs. When NADS goes high (before the end of address time), the address bits present at the AD00 to AD07 Inputs are stored in the address latches of the RAMs. During address time, the CE✲ Inputs to the RAMs are held low by the D15✲ version of the D15 address bit. The low CE✲ Inputs also are stored in latches in the RAMs when NADS goes high. The RAMs now are ready to execute a Memory Read or Write Cycle.

Table 2-4. Address Bit Control Configurations

Address Bit		Selects
D14	D15	
0	0	ROM
1	0	Spare
0	1	BTE
1	1	RAM

A Read Cycle is executed if PACE is performing a data-input operation. The PACE IDS Signal, connected to the OE Inputs of the RAMs, goes high after the address and CE✲ Signals are latched into the RAMs. When IDS goes high at the OE Inputs to the RAMs, the Output Drivers on the RAM chips permit the addressed data to be placed on the System TTL Address/Data Bus. The data outputs DO01 through DO04) from the RAM (on the right side of figure 2-29) are connected to the D00 through D03 (lease significant) bit lines of the System TTL Address/Data Bus. The data outputs from each of the remaining RAMs (from right to left) are connected to the next four successive most significant data bit lines of the System TTL Address Data Bus. So, in summation, the four RAMs are addressed simultaneously, enabled by the CE✲ Input (D15✲), directed by



ROM TRUTH TABLE

CE1* t	CE2 t	ST t	CE1 t+1	CE2 t+1	EN* t+1	ST t+1	OUTPUT t+1
NOTE X	X	X	0	0	0	1	READ STORED DATA FOR ADDRESS INPUTS AT t + 1
X	X	X	1	X	X	1	Hi-Z
X	X	X	X	1	X	1	Hi-Z
X	X	X	X	X	1	1	Hi-Z
0	0	1	X	X	0	0	READ STORED DATA FOR ADDRESS INPUTS AT t
1	X	1	X	X	X	0	Hi-Z
X	1	1	X	X	X	0	Hi-Z
X	X	X	X	X	1	0	Hi-Z

NOTE
"X" INDICATES "DON'T CARE" STATE

Figure 2-29. Typical System Implementation of RAM and ROM

the OE Input (IDS) to perform a Read Cycle, and then place 16 bits of data onto the System TTL Address/Data Bus to PACE.

A Write Cycle is executed if PACE is performing a data-output operation. The RAMs are addressed and enabled as described for a Read Cycle. The PACE ODS Signal is connected by way of the System TTL Timing and Control Bus to the WE Input of each RAM. When the ODS Signal goes high at the WE Inputs, 4 bits of the 16-bit data word placed on the System TTL Address/Data Bus by PACE are stored in the previously addressed memory location of each RAM. During the Memory Write Cycle, the DO01 through DO04 data output lines from each RAM are set in the high-impedance state by the ODS Signal applied to the WE Inputs of the RAMs. System data memory may be constructed in 8- and/or 16-bit lengths to accommodate system interfaces.

BIT D	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
	1	1	1	1	1	1	1	1	X	X	X	X	X	X	X	X
X'	F				F				X				X			

Figure 2-30. Address Bit Configurations Versus Hexadecimal Address

2.4.6 ROM

The IPC-16A/505 ROM provides 1024-by-16 bits of bipolar mask-coded Read-Only Memory. In addition, the ROM

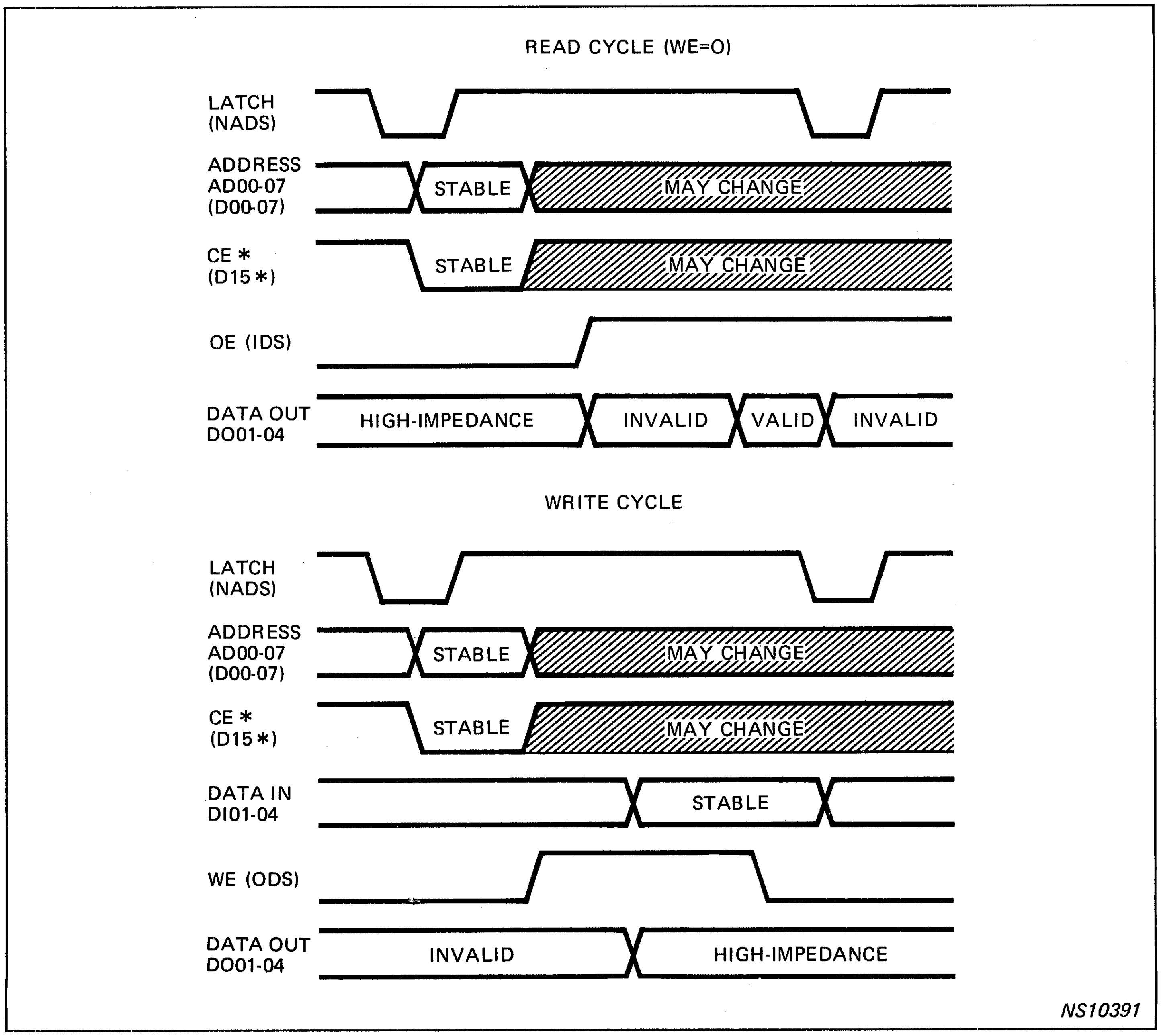


Figure 2-31. RAM Timing Diagram

chip contains address latches which permit the ROM to be used without an ALE. Latches also are provided for two of the three enabling inputs. Figure 2-32 shows a functional block diagram of the ROM.

Address bits AD00 through AD09 are applied to latches which accept the address information when the Strobe Signal Input is high. When the Strobe Signal goes low, the latches retain the latest address inputs and accept no new input information. The Chip Enable 1* and 2* (CE1*, CE2*) Inputs are ORed and the resultant signal is also latched by the Strobe Signal.

The AD03 through AD09 address bits are applied to a Buffer and 1-of-128 Decoder. The 1-of-128 Decoder uses the states of the AD03 through AD09 address bits to select one row containing 128 bits from the 128 rows in the ROM Array. The AD00 through AD02 address bits are applied through a Buffer to a 1-of-8 Decoder. The 1-of-8 Decoder uses the states of the AD00 through AD02 address bits to select one of eight groups of 16 bits contained in the row selected by the 1-of-128 Decoder. The 16 selected bits are applied to the inputs of the Output Drivers.

The logical OR state of the latched CE1* and CE2* Inputs is applied with the Enable* Signal (EN*) to the Output Enable OR gate. When all enabling inputs are low, the Output Enable OR gate provides a high signal to the Output Drivers state-control input. The high signal from the Output Enable OR gate permits the Output Drivers to pass the 16 selected data bits from the 1-of-8 Decoder to the System TTL Address/Data Bus.

When any of the three enabling inputs (CE1*, CE2*, or EN*) are high, the Output Drivers assume the high-impedance output state. The high-impedance output state permits memory expansion to a greater number of words without sacrificing speed, as is the case when open-collector outputs are used.

Figure 2-29 shows a typical system implementation of the ROM. As stated in the RAM description, the ROM is enabled when the address on the TTL Address/Data Bus is for low memory (X'0XXX). The low D15 and D14 bits, resulting from the low memory address, are applied to the ROM CE1* and CE2* Inputs during address time. The NADS Signal, supplied by PACE in the middle of address time, is inverted and the resultant ADS Signal is applied to the ROM Strobe Input. When the ADS Signal goes high (see figure 2-33), the D00 through D09, D14, and D15 address bits present at the ROM AD00 through AD09 and CE1*/CE2* Inputs, respectively, are accepted by latches on the ROM chip. When ADS goes low, the address bits are latched into the on-chip address latches and D15 is latched into the CE1*/CE2* Latch.

In order to access the ROM, PACE must perform a data-input operation (IDS active). As part of the data-input operation, PACE supplies an active IDS Signal after address time. The IDS Signal is routed over the System TTL Timing and Control Bus to an inverter. The inverter output (IDS*) is applied to the ROM EN* Input. When the EN* Input goes low, the Output Drivers on the ROM chip are enabled to pass the addressed data from the DO01 through DO16 Outputs onto bit lines BDI/O00 through BDI/O15 of the System TTL Address/Data Bus. When the IDS Signal terminates, IDS* goes high and the ROM Output Drivers assume a high-impedance output state to permit easy memory expansion

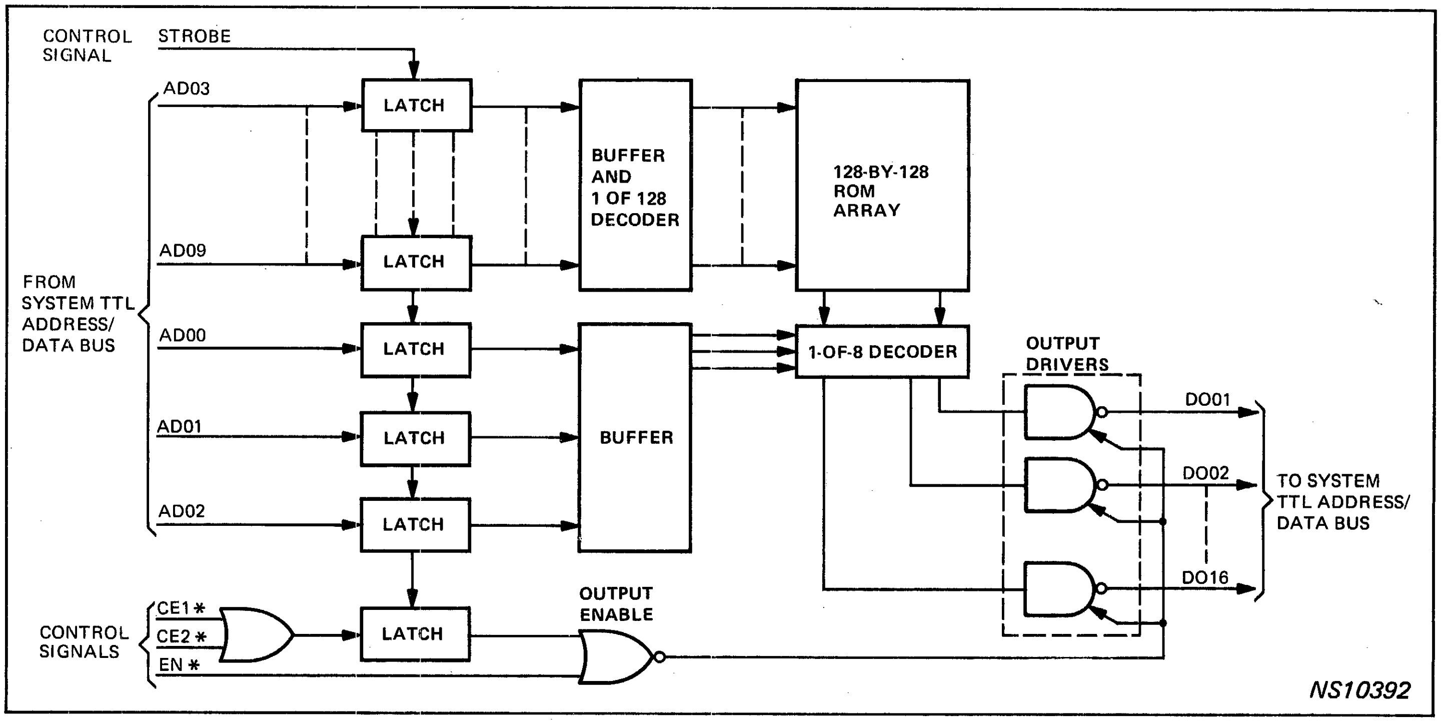


Figure 2-32. ROM Functional Block Diagram

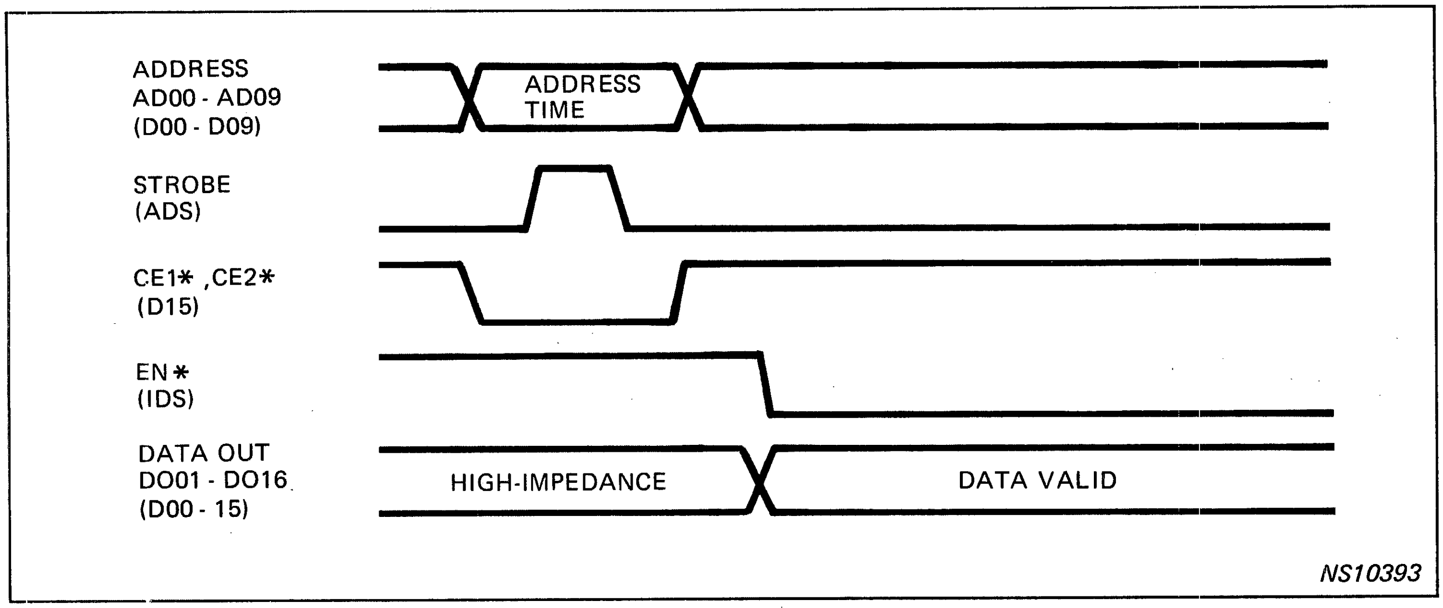


Figure 2-33. ROM Timing Diagram

2.5 PACE APPLICATION INFORMATION AND EXAMPLES

The following paragraphs provide additional information and examples concerning application of the PACE microprocessor. Some of the topics covered include the following:

- Minimum chip configuration
- 8-bit interfacing
- 16-bit interfacing
- BCD data handling
- Serial Input/Output
- Use of jump conditions and flags
- Use of interrupts
- Implementation of cycle extend/suspend
- Implementation of DMA

2.5.1 MINIMUM CHIP CONFIGURATION

The minimum chip configuration required for a system using a fully multiplexed 16-bit address/data bus is shown in figure 2-34. All system timing, control signal, and bus interfacing requirements are met with a PACE, an STE, and three BTEs. The memory size depends upon system application. Peripheral interfacing is not included, as requirements vary with application.

2.5.2 INPUT/OUTPUT CONTROL TECHNIQUES

The following paragraphs describe control techniques and special considerations for various types of PACE input/output conditions.

2.5.2.1 8-bit Interfacing

A system using an 8-bit data configuration for peripheral or memory addressing usually contains a 16-bit instruction memory (typically ROM), an 8-bit data memory (RAM), and 8-bit peripheral Device Interface Elements (ILE/8). The instruction memory is 16-bits wide because PACE operation always is controlled by 16-bit instructions regardless of the data length serviced. The hardware design incorporated into the PACE concept permits the microprocessor ALU, registers and Stack to manipulate 16-bit memory addresses and instructions while, at the same time, 8-bit data is manipulated in the 8-bit data mode. Thus, the PACE concept provides greater execution speeds and more powerful instructions for either 8-bit or 16-bit operations than can be achieved with conventional 8-bit microprocessors.

Selection of the 8-bit data-handling capability is accomplished by setting the microprocessor BYTE Status Flag high. When the BYTE Status Flag is set high, PACE execution of Shift and Rotate Instructions is modified and the operation of some of the status flags is changed. In systems servicing both 8-bit and 16-bit data, the user-generated software can incorporate the Set Flag and Pulse Flag Instructions to change the state of the BYTE Status Flag to accommodate the interfaced data length.

Eight-bit peripheral interfacing is described from the hardware viewpoint, along with the corresponding use of PACE control signals, by figure 2-22 and the associated text. The main hardware consideration for 8-bit memory or peripheral interfacing is to ascertain that the eight data lines from the peripheral interface or memory are connected to the eight low-order PACE data lines (D00-D07). However, aside from hardware and control signals, some special considerations should be observed for 8-bit memory or peripheral interfacing as regards memory addressing, the PACE status flags, and some of the PACE instructions.

Both the indexed and base-page memory addressing modes require consideration when 8-bit data is processed. Accessing both 16-bit (program) and 8-bit (data) words by using the base-page mode may be desirable. Since two different

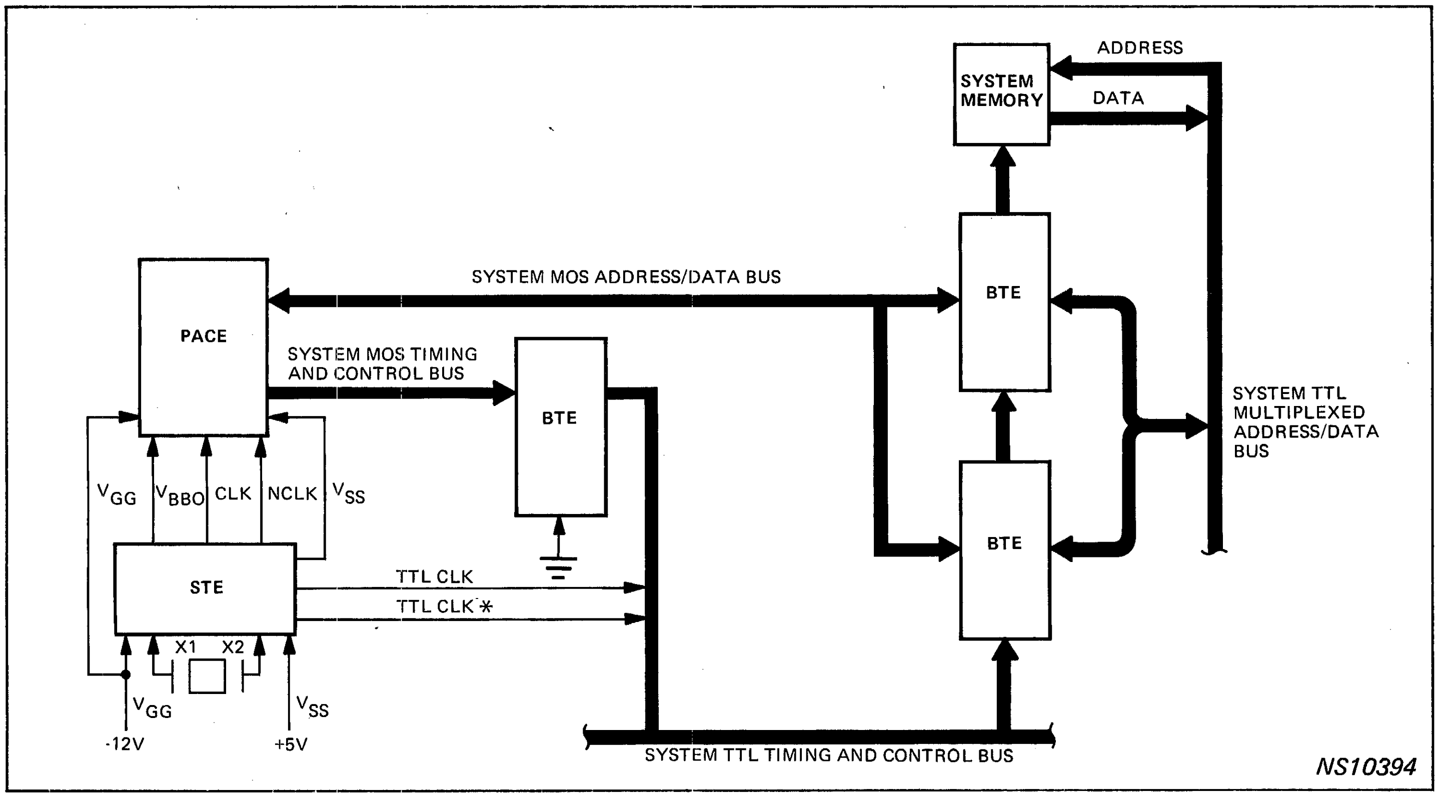

ADDRESS
SYSTEM MEMORY
DATA
SYSTEM MOS ADDRESS/DATA BUS
PACE
BTE
SYSTEM MOS TIMING AND CONTROL BUS
BTE
SYSTEM TTL MULTIPLEXED ADDRESS/DATA BUS
V_{GG}
V_{BBO}
CLK
NCLK
V_{SS}
BTE
STE
TTL CLK
TTL CLK*
X1
X2
V_{GG}
V_{SS}
-12V
+5V
SYSTEM TTL TIMING AND CONTROL BUS
NS10394

Figure 2-34. Minimum Chip Configuration

memories (ROM and RAM) are used, splitting the base page between the two memories may also be desirable. Figure 2-29 and the supporting text provide an example of how base-page splitting can be easily accomplished.

For indexed addressing, Accumulators AC2 and AC3 are used as 16-bit memory pointers. If Accumulators AC2 and AC3 are loaded from the 8-bit memory, the high-order 8 bits in the accumulators can be set equal to the sign of the low-order 8 bits by using the Load With Sign Extended Instruction (LSEX). Thus, a 16-bit twos-complement number results.

The Load With Sign Extended Instruction also can be used to set the state of the eight high-order data bits during 8-bit data transfers from peripherals. Alternatively, user-generated software can use Shift Instructions to set the eight high-order data bits to zero. The Shift and Rotate Instruction group (SHL, SHR, ROL, ROR) operates on the low-order 8 bits only and sets the high-order 8 bits to zero when the BYTE Status Flag is set for the 8-bit data-handling mode.

The Immediate Instructions (LI, CAI, AISZ) provide 16-bit, twos-complement data inputs. When working with 8-bit data, the high-order 8 bits usually can be ignored. If required, the high-order 8 bits can be cleared by using a Shift Instruction.

The Branch and Skip Instructions are modified to account for the 8-bit data length. Thus, the REQ0 and NREQ0 conditions are affected only by the low-order 8 bits. The PSIGN and NSIGN Signals indicate the sign of the low-order 8 bits. The Skip Instructions (SKNE, SKG, SKAZ, ISZ, DSZ) test only the low-order 8 bits. Thus, if a Skip Instruction compares 8-bit accumulator data with a 16-bit program memory word, the contents of the high-order 8 bits of both words are ignored. The Add Immediate, Skip if Zero Instruction (AISZ) is the only instruction that tests the entire 16-bit result when 8-bit data handling is selected. Therefore, the AISZ Instruction can be used to increment the index accumulators (AC2, AC3) without skipping every time the low-order 8 bits are zero. Consequently, the sign of 8-bit numbers must be extended by using the Load With Sign Extended Instruction to properly detect zero when using the AISZ Instruction for 8-bit data.

Since the Overflow and Carry Flags are modified by arithmetic instructions, the eight low-order data bits determine the state of the Overflow and Carry Flags when the 8-bit data length is selected. That is, the Carry Flag is set if a carry is generated by the low-order 8 bits and the Overflow Flag is set when an arithmetic overflow occurs in the low-order 8 bits.

The Link Flag is affected by Shift and Rotate Instructions. The Link Flag is set by data shifted out of the low-order 8 bits when the 8-bit data length is selected.

Working with 8-bit data and 16-bit instructions sometimes necessitates performing arithmetic operations by using a 16-bit operand from the program memory and an 8-bit operand from the data memory. If the result is to be treated as 8-bit data, no special considerations are required. However, if the result is to be treated as 16-bit data, the sign of the 8-bit operand first must be extended by using the Load With Sign Extended Instruction. Also, the carry, overflow, and conditional branch signals that are only a function of the low-order 8 bits should not be used. Alternatively, the

BYTE Flag temporarily may be set low for 16-bit data handling to accommodate the signals changed by the 8-bit data-handling mode.

The previously mentioned factors make the use of PACE in 8-bit applications convenient while still providing the advantages of a 16-bit instruction set. (Data lengths other than 8 bits or 16 bits also may be used when special external hardware is provided.)

2.5.2.2 16-bit Interfacing

No special considerations are necessary for 16-bit interfacing except to ascertain that the peripheral and memory data bit lines are connected to the appropriate PACE data bit lines (that is, D00 is the least significant bit and D15 is the most significant bit). Also, the BYTE Status Flag must be set low for proper 16-bit data operations. Figures 2-22, 2-26, and 2-29, together with the associated texts, provide information regarding 16-bit interfacing.

2.5.2.3 BCD Data

The PACE microprocessor is capable of adding four-digit-per-word BCD data with the Decimal Add Instruction (DECA) or two digits per word if BYTE equals 1. Consequently, no BCD-to-binary conversion is required. In appendix B, table B-4 provides a decimal addition program example that adds two 16-digit BCD strings using the DECA Instruction.

2.5.2.4 Serial Input

Serial interfaces to PACE can be provided by using a single-bit line of the address/data bus for the interface. Another method, which may be preferable for use in systems containing only a few peripherals, is to use a jump condition (JC13, JC14, or JC15) or interrupt input (NIR2 through NIR5) to service serial data. Using a jump condition or interrupt input avoids the need for address decoding and reduces the number of interface lines. The serial input data are applied to the selected interrupt or jump condition input. The state of the jump condition inputs then can be determined by instructions in the user-generated software.

2.5.2.5 Serial Output

As with serial input, a single-bit line of the address/data bus can be used for the interface. However, one of the user flag outputs (F11 through F14) may prove more effective for some system applications. The user flags can be set or cleared by using the Set Flag or Pulse Flag Instructions in the user-generated software. The use of jump condition or interrupt inputs and flag outputs is particularly well suited for asynchronous serial devices, such as a teletypewriter, since only one transmit and one receive line are involved.

2.5.3 USE OF JUMP CONDITIONS AND FLAGS

The PACE microprocessor contains a Jump Condition Multiplexer that samples the 16 jump conditions listed and described in appendix B, table B-3. The Branch-On Condition Instruction (BOC) tests the Jump Condition Multiplexer Output. If the condition for branching (selected by the condition code of the BOC Instruction) is active, a branch occurs; otherwise the next sequential instruction is executed.

The CONTIN Jump Condition is used by the HALT Instruction. If a Halt Instruction is executed, the microprocessor NHALT Output is driven low to indicate that microprocessor activity is suspended until the CONTIN Input is pulsed. While PACE operation is suspended, the NHALT Output Line has a 7/8 duty cycle; that is, every eighth clock phase, the NHALT Output goes high. The NHALT 7/8 duty cycle must be accounted for if the output is used as a logic signal but is of little concern if the output drives only a halt indicator. The NHALT Output goes high after the Halt Instruction is terminated by pulsing the CONTIN Input. The CONTIN Input must go high for four clock cycles, minimum, for PACE operation to resume.

The three unassigned jump condition inputs (JC13, JC14, and JC15) are for user purposes and may be implemented as required by the application.

The 14 status and control flags provided by PACE are listed and described in appendix B, table B-8. As previously described, the user flags (F11 through F14) and user jump conditions can be used for serial data input/output. In some cases, additional flags may be required for control purposes. The additional flags can be obtained conveniently by using a DM9334 8-bit addressable latch. An unused address bit or combination of bits may be used to enable the latch. Three bits can be used to address one of eight flags and another bit can specify set or reset as illustrated in figure 2-35. A Store Instruction may be used to output the address (data output is ignored).

In a similar manner, a multiplexer and latch can be used to expand user jump conditions. The latch is loaded from the address bus, if enabled by an unused address code, and selects a Multiplexer input to one of the user jump conditions.

2.5.4 USE OF INTERRUPTS

The PACE microprocessor provides a six-level priority interrupt structure. Each level is provided with an individual Interrupt Enable as shown in figure 2-36. A master Interrupt Enable (IEN) is provided for all five lower-priority levels at once. The master IEN is an input to the PACE Jump Condition Multiplexer. The state of Interrupt is tested by PACE during the Instruction Fetch Routine (internal to PACE) that is executed after completion of each instruction. Thus, if Interrupt is high, the interrupt is automatically serviced.

Negative-true Interrupt Request Inputs (NIR2 through NIR5) are provided to allow several interrupts to be wire-ORed to each input. When an Interrupt Request occurs, the associated Interrupt Request Latch (IR1 through IR5) is set if the corresponding Interrupt Enable Input is true. Since the Interrupt Request Latch can be set by any pulse exceeding one clock period, narrow timing or control pulses can be captured. If IEN is high, then an interrupt is generated and acknowledged after completing the current instruction.

During the interrupt sequence, an address is provided by the output from the priority encoder. The address is used

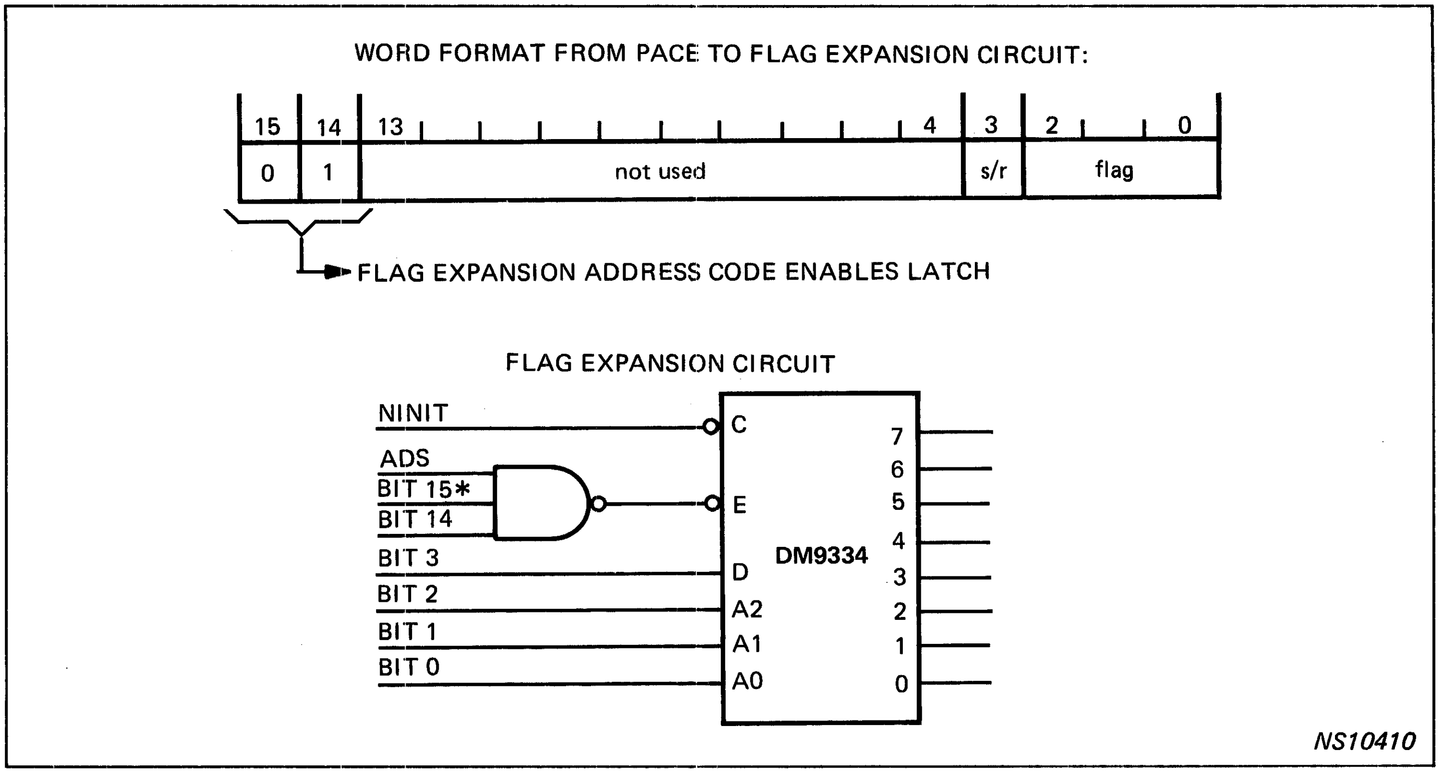


Figure 2-35. One Possible Circuit and Word Format for Obtaining Additional Flags

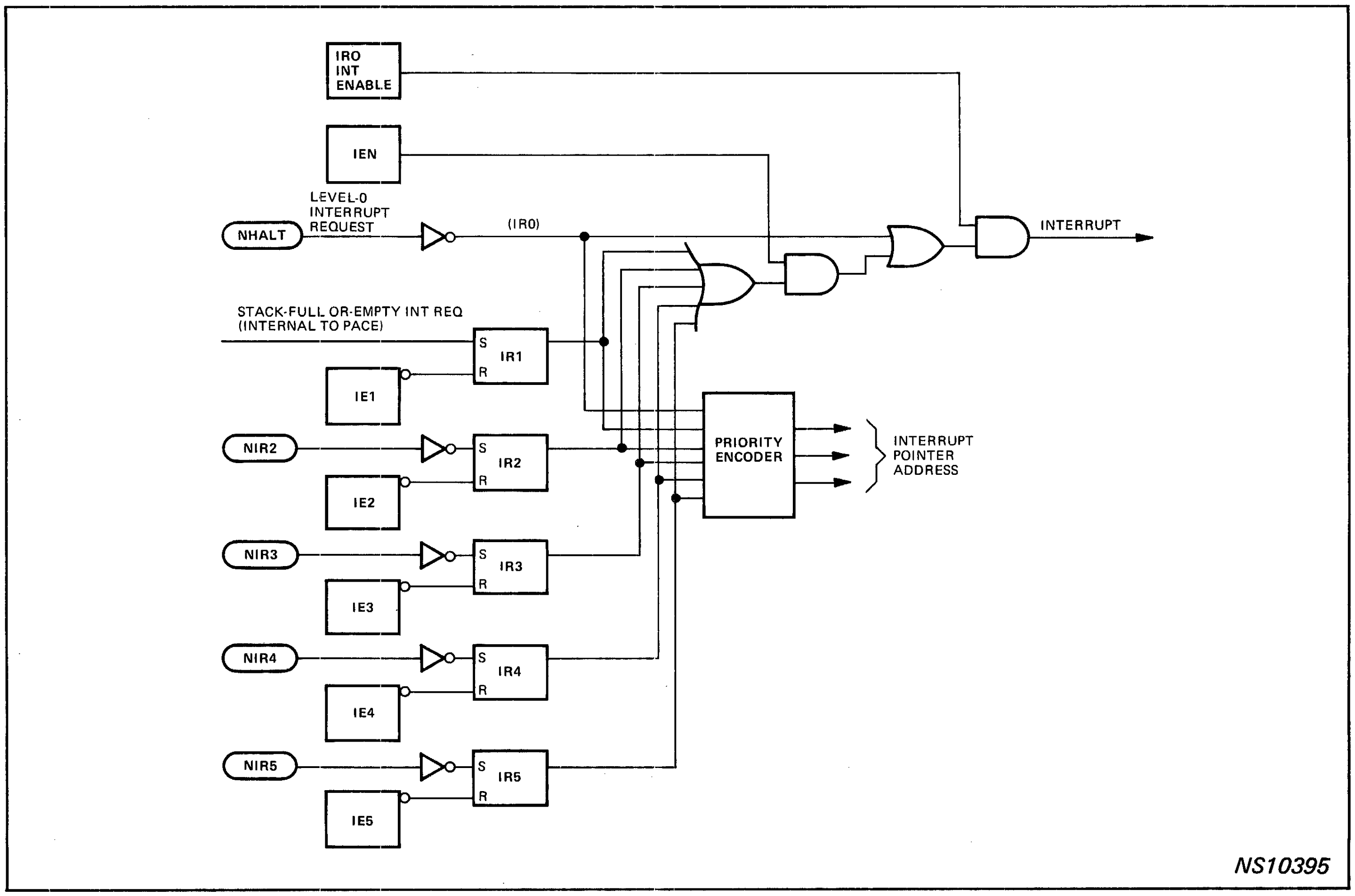


Figure 2-36. PACE Interrupt System

to access the Interrupt Pointer for the highest-priority Interrupt Request (IR0 is highest priority; IR5 is lowest priority). The Interrupt Pointers are stored in memory locations 2 through 8 (see table 2-5) for Interrupt Requests 1 through 5 and 0, respectively. The Interrupt Pointer specifies the starting address of the Interrupt Service Routine for the particular interrupt level, except in the case of the Level-0 Interrupt (IR0), which is used primarily for alarm interrupts and Control Panel implementation.

Table 2-5. Locations of Interrupt Pointers

Interrupt Pointer	Memory Location
Interrupt-0 Program	8
Interrupt-0 PC	7
Interrupt 5	6
Interrupt 4	5
Interrupt 3	4
Interrupt 2	3
Interrupt 1	2
Not Assigned	1
Initialization Instruction	0

Before Interrupt Service Routine execution, the Program Counter contents are pushed onto the Stack and IEN is set low (false). This interrupt handling requires 14 microseconds (28 clock cycles). The Interrupt Service Routine may set IEN high (true) after turning off the Interrupt Enable for the interrupt level currently being serviced (or resetting the Interrupt Request). The Interrupt Enable Flags can be set by the Set Flag (SFLG) and reset by the Pulse Flag (PFLG) Instructions. If an Interrupt Enable Flag is set or reset, one more instruction is executed before the interrupt is enabled or disabled. The Return From Interrupt Instruction (RTI) also may be used to set IEN true. In this case, there is no delay and a pending interrupt takes effect immediately after execution of RTI.

It should be recognized that the function of the individual Interrupt Enables IE1-IE5 is to *arm* or *disarm* the Interrupt Request Latch; whereas, the function of the Master Interrupt Enable (IEN) and Interrupt Enable IR0 is to *enable* or *disable* the latched Interrupt Request Lines.

Three types of external interrupts are likely to occur in PACE applications: short-duration (pulse) interrupts; long-duration resettable interrupts; and nonresettable interrupts. The short-duration interrupt exists for less than the interrupt response time and may be caused by a strobe pulse from a peripheral device or the occurrence of a high-speed transient condition. A short-duration interrupt must be latched to be recognized. Interrupts longer than the clock period are latched by the PACE Interrupt Request Latches. The Interrupt Service Routine must reset the Interrupt Request Latch by turning off the Interrupt Enable for the level being serviced. If the Interrupt Enable is left off, Interrupt Request pulses cannot set the Interrupt Request Latch.

Long-duration resettable interrupts last longer than the interrupt response time and may be reset by the Interrupt Service Routine. An example is a Buffer-full Interrupt by a peripheral device. The Interrupt Service Routine empties the buffer, removing the interrupt. A long-duration interrupt is ignored when Interrupt Enable is low but still generates an interrupt when Interrupt Enable is set true. In servicing long-duration interrupts, the Interrupt Request Latch must be cleared after the interrupt is reset by the Interrupt Service Routine.

Long-duration nonresettable interrupts last longer than the interrupt response time and are not reset by the Interrupt Service Routine. An example of a long-duration nonresettable interrupt is a photoelectric cell that detects the presence of an item on a conveyor. The signal produced by the photoelectric cell (or some other sensor) may last for a significant portion of a second. Setting the Interrupt Request Latch on the edge of the interrupt is desirable and may be accomplished using a simple RC circuit or single-shot to generate a pulse on the edge of the interrupt.

The interrupt response time for PACE is equal to the time to finish the current instruction at the time of the interrupt, plus the time to access the first instruction of the Interrupt Service Routine. Instruction execution times are given in appendix B, table B-2.

An example of an Interrupt Service Routine for Interrupt Level 3 is shown in table 2-6. Memory location 4 contains the address of the first instruction in the routine. When a Level-3 Interrupt occurs, the first instruction preserves the state of the flags on the Stack.

NOTE

IEN is set false by the interrupt prior to being saved on the Stack.

The flag data then are loaded into AC0 and all bits which are to be modified are masked out to zero. The desired bits then are set true by ORing with IESTAT. If the routine is interruptable, then IE3 is set to zero and IEN is set to one. The modified status word then is transferred from AC0 to the status register. The actual servicing of the interrupting device then takes place. At the end of the routine, the flags are restored and a Return Instruction is executed. If the interrupts are to be reenabled, the RTI Instruction must be used since RTI sets IEN true and restores the PC from the Stack.

A Stack Interrupt occurs when the Stack-empty or Stack-full condition exists. The Stack Interrupt consists of a pulse applied to the set input of Interrupt Request Latch 1 (see figure 2-36). The pulse sets the latch if the IEN1 Flag is true; otherwise, the pulse is ignored. The Stack is implemented with a RAM and a Pointer which can access RAM locations 0 to 9. A pulse occurs when the Stack Pointer is at 0 (one entry on Stack), and a Read-Stack Operation occurs to empty the Stack. A pulse also occurs when the Stack Pointer is equal to 7 (eight entries on Stack), and a

Write-Stack Operation occurs to fill the ninth word and leave one word empty to be used by the interrupt. When a Stack Interrupt occurs, the Stack condition can be determined by using the Stack-full Jump Condition (STFL).

With the interrupt scheme described, an interrupt does not occur at initialize but does occur every time the Stack becomes empty. If the Stack is to be extended into memory, a Stack-empty Interrupt is required but may be inhibited by turning off IEN1 in other cases. In order to prevent a Stack Interrupt when both hardware and software Stacks become empty, a dummy word may be pushed on the Stack by the Initialize Routine.

The Level-0 Interrupt is not maskable under program control. A Level-0 Interrupt may be used for alarm conditions such as a power failure or for implementing a software-based Control Panel such as that contained in the IPC-16P Microprocessor Development System. A Level-0 Interrupt can be generated by using the PACE NHALT and CONTIN Signal inputs. Figure 2-37 illustrates the relative timing for Level-0 Interrupt generation. As is shown in figure 2-37, the CONTIN Signal can be used as an interrupt acknowledge signal. For cases where an interrupt acknowledge is not required, or where the CONTIN Signal is used as a sense input to the program, the CONTIN Signal can be held continuously low. While holding the CONTIN Signal continuously low, the NHALT Signal must be driven low for the duration of the longest instruction execution time plus eleven clock cycles to guarantee that a Level-0 Interrupt occurs.

After the NHALT Signal returns to a high state, the Level-0 Interrupt is serviced. Servicing consists of first setting the Level-0 Interrupt Enable (IRO INT ENABLE in figure 2-36) low to lock out all other possible interrupts. Next, the PACE Program Counter contents are stored in the location specified by memory location 7 (see table 2-5). Then, the instruction at memory location 8 is executed. Storing the Program Counter contents in a memory location instead of on the Stack prevents generation of a Stack-full Interrupt.

To return from a Level-0 Interrupt, the PFLG15 or SFLG15 Instruction is executed to set the Level-0 Interrupt Enable Output high after execution of one additional instruction. The additional instruction is typically a JMP@ through memory location 7, which contains the Program Counter contents. Thus, a proper return to the interrupted program can be effected. During initialization, the Level-0 Interrupt Enable is set high.

In some applications, expansion of the user interrupts by providing several interrupts on a single input may be desirable. Several interrupts can be provided on a single input by using open-collector gates for a wired-OR input and polling all the devices on a given level to discover the origin of the interrupt. However, in some applications, the polling technique may take excessive time. In such cases, use of the DM9318 Priority Encoder is recommended to encode the number of the highest-priority interrupting device. A single instruction in the Interrupt Service Routine then can be used to read the number of the interrupting device over the data bus. The use of the DM9318 Priority Encoder is shown in figure 2-38. The use of a DM8131 Comparator with latched output to detect the appropriate peripheral address also is illustrated in figure 2-38.

NOTE

Status register masking is necessary only when Interrupt Enable status is to be modified to allow higher priority devices to interrupt. Pushing the status register onto the Stack is necessary only if the routine alters the contents of the status register.

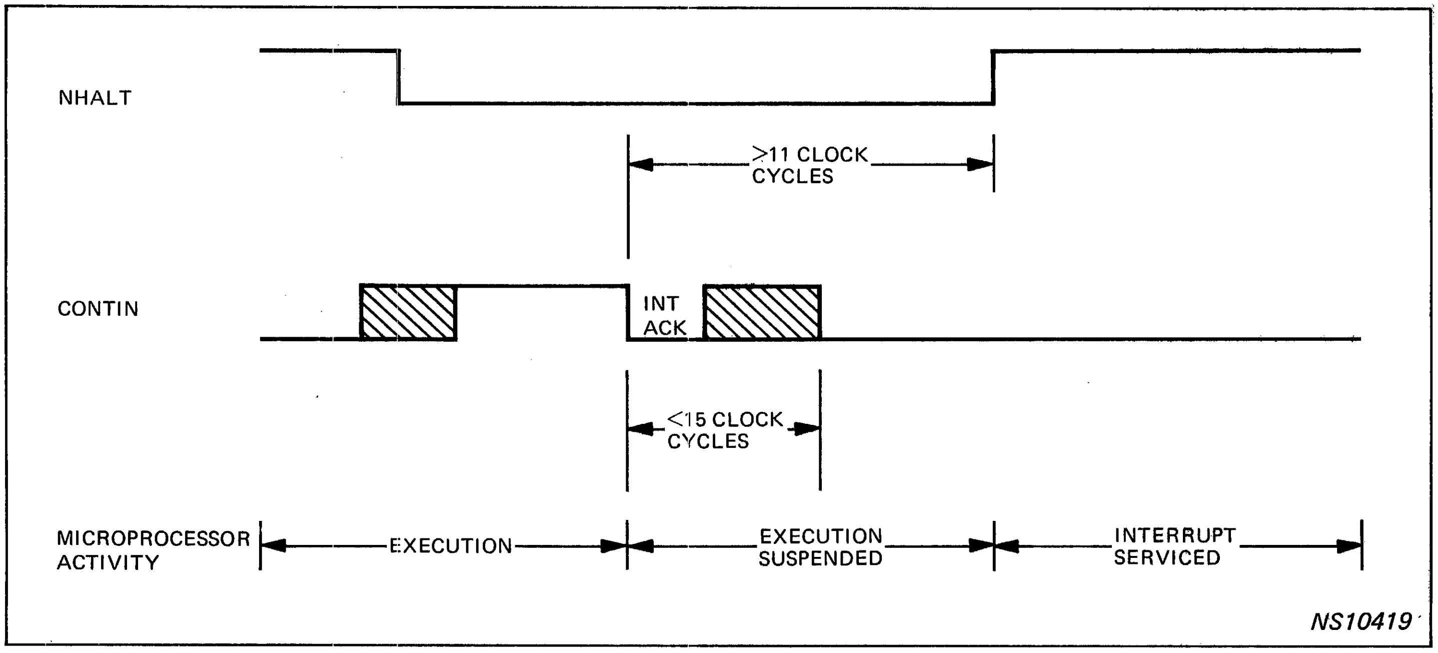


Figure 2-37. Relative Timing for Level-0 Interrupt Generation

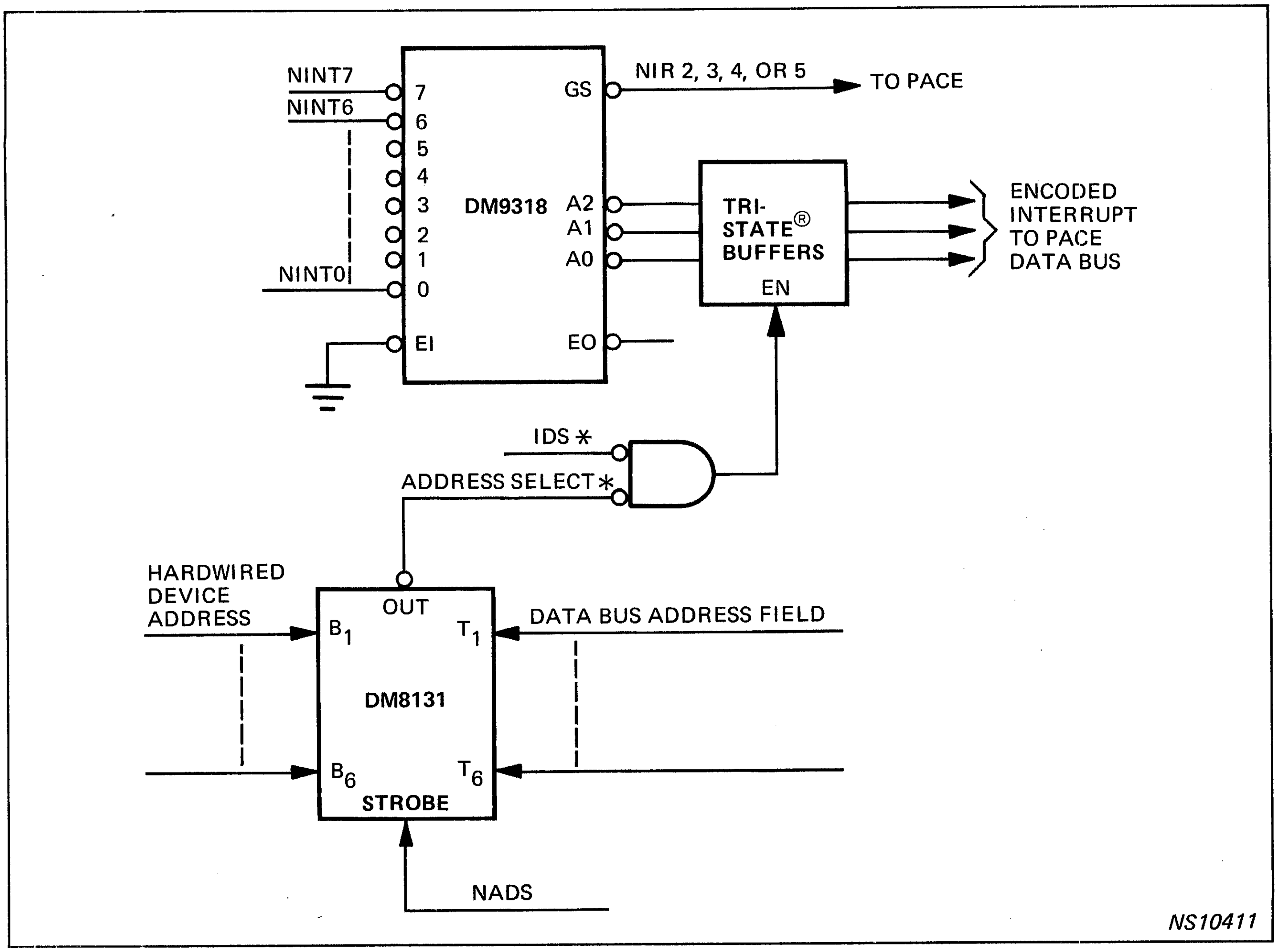


Figure 2-38. Use of DM9318 Priority Encoder and DM8131 Comparator for Interrupt Expansion and Detection

2.5.5 IMPLEMENTATION OF INITIALIZE AND CYCLE EXTEND SIGNALS

The following paragraphs describe the use of Initialize and Cycle Extend Signals. The Initialize Signal is used to initialize the PACE microprocessor and other system elements during the power-up condition or at any other desirable time. The Cycle Extend Signal can be used to increase the I/O cycle time by multiples of the clock period.

2.5.5.1 Initialize

The PACE Initialize Signal (NINIT) input may be used at any time to initialize the microprocessor and should always be used during system power-up. Application of a low NINIT Signal clears the Stack Pointer, sets the flags to zero, sets the Level-0 Interrupt Enable true, and sets the Program Counter contents to zero. The accumulators assume an arbitrary state. The NADS, IDS and ODS data strobes are set false. Thus, if system data are to be preserved during initialization, NINIT should be inhibited during data output cycles.

The PACE data strobes (NADS, ODS, and IDS) are inactive for 16 clock cycles after the trailing edge of the NINIT Signal occurs. After the 16 clock cycles, the first NADS Signal occurs and the first instruction is accessed from memory location X'0000, unless a Level-0 Interrupt (Control Panel Interrupt) is present. All other interrupt levels are disabled. Figure 2-24 shows a circuit that can be used for generating an INIT Signal during system power-up. The resultant output must be inverted to provide NINIT.

2.5.5.2 Cycle Extend

The PACE Extend Input can be used to increase I/O cycle times by multiples of the clock period (see figure 2-6). To extend I/O cycles, a circuit like that illustrated in figure 2-39 can be used. The NADS Signal is used to initiate the Extend pulse while the TTL CLK from the System TTL Timing and Control Bus is used to count out the desired number of clock cycles. The circuit shown in figure 2-39 provides an extend of one clock cycle for data-input operations, as might be required for an MOS ROM.

The Extend Input also can be used for suspending the microprocessor activity to provide a *cycle-steal* for Direct Memory Accessing (DMA). Refer to the *Implementation of DMA* paragraph for more information.

Table 2-6. **Interrupt Service Routine Example**

Assembly Code			Explanation
	. = 4		Set location counter equal to 4.
	.WORD	ISERV3	Pointer to service routine.
	. = 500		Set location counter equal to 500.
ISERV3:	PUSHF		Save flags on Stack.
	PUSH	AC0	Save AC0.
	CFR	AC0	Move flags to AC0.
	AND	AC0, MASK	Mask out old Interrupt Enable status.
	OR	AC0, IESTAT	OR in new Interrupt Enable status.
	CRF	AC0	Store in Flag Register.
	.		.
	.		.
	.		.
	.		Interrupt Service Routine.
	.		.
	.		.
	.		.
	.		.
	.		.
INTXIT:	PULL	AC0	Restore AC0.
	PULLF		Restore flags.
	RTI		Return to interrupted routine.
MASK:	.WORD	(mask data)	
IESTAT:	.WORD	(Interrupt Enable status data)	

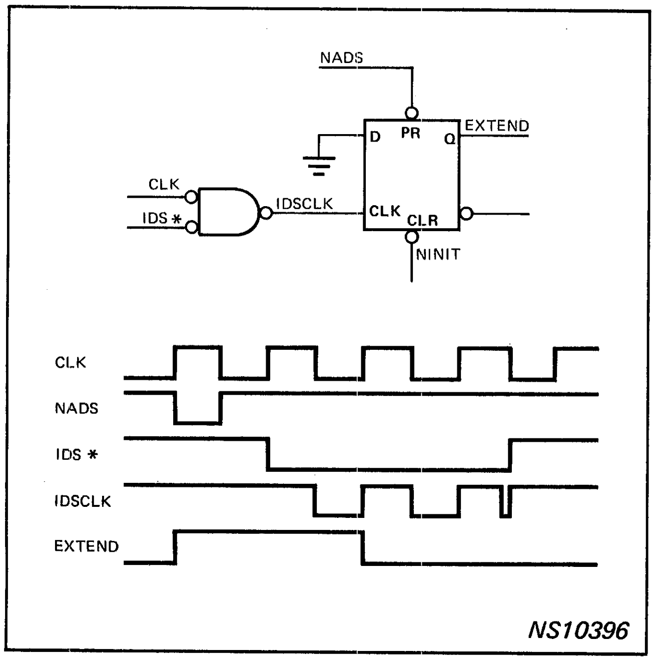


Figure 2-39. Circuit and Timing Diagram for One Clock Cycle Extend

2.5.6 IMPLEMENTATION OF DMA

The PACE family of chips permits systems to be configured that have a Direct Memory Access (DMA) capability. Figure 2-40 shows a typical system implementing DMA. The RAMs and ROM of figure 2-40 are connected exactly the same as the RAMs and ROM of figure 2-29. Therefore, the descriptive text for figure 2-29 is applicable also to RAM/ROM control in figure 2-40.

The DMA Bus Controller in figure 2-40 must be user-fabricated. The DMA Bus Controller should use the PACE IDS, ODS, and NADS Signals, the STE NCLK Signal, and the peripheral signals shown in figure 2-40 to produce Extend, BNADS, BIDS, BODS, and MBG control signals. The timing for the control signals produced by the DMA Bus Controller should be as shown in figure 2-41.

The PACE ODS and IDS Signals should be tested by the DMA Bus Controller for active states. If ODS and IDS are inactive and a priority bus request is present from a peripheral, then NADS should be tested for 1.5 clock cycles, minimum, while generating an Extend. If NADS is not active within 1.5 clock cycles, then the DMA Bus Controller

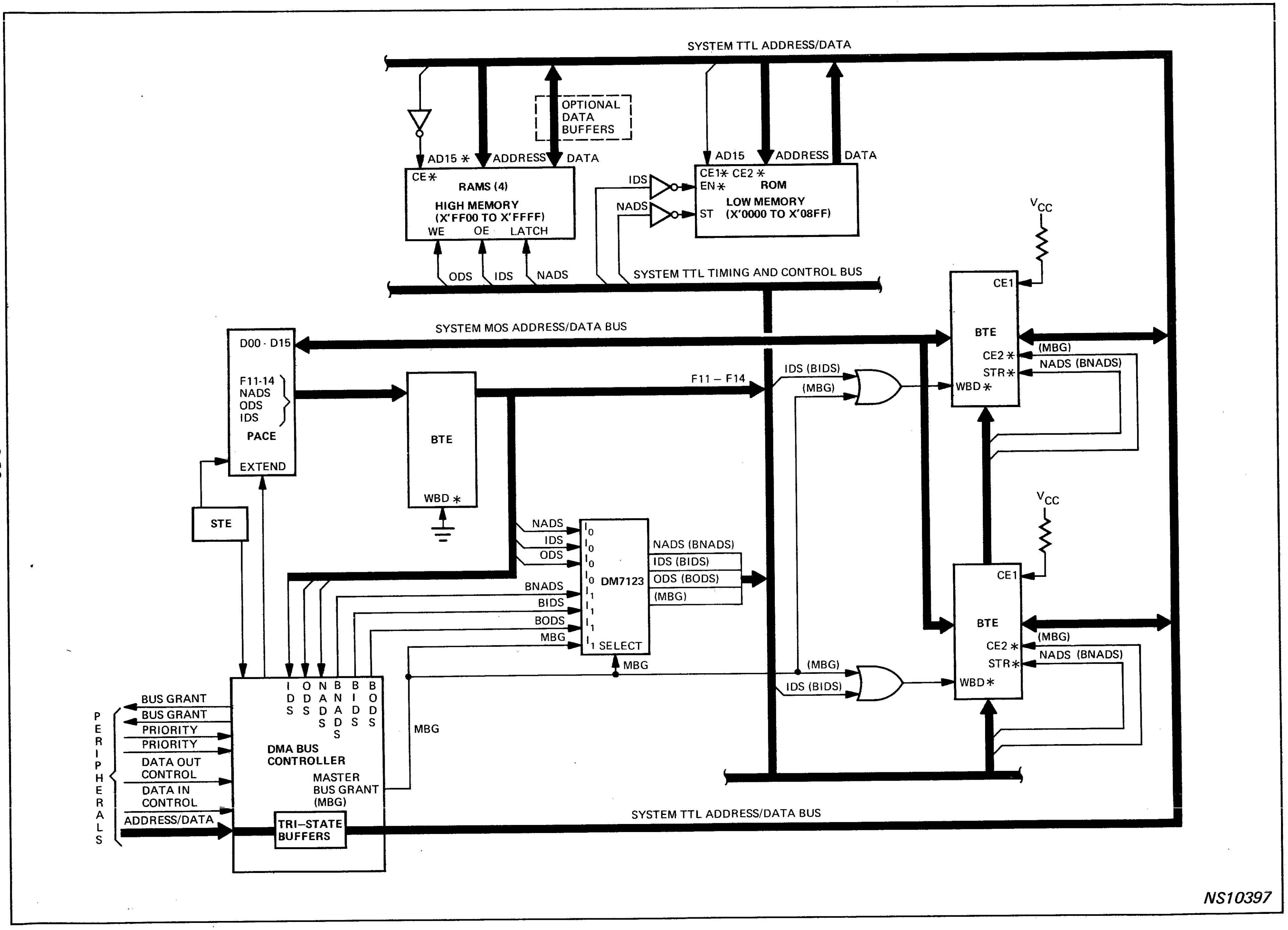


Figure 2-40. System Implementation of DMA Bus Controller

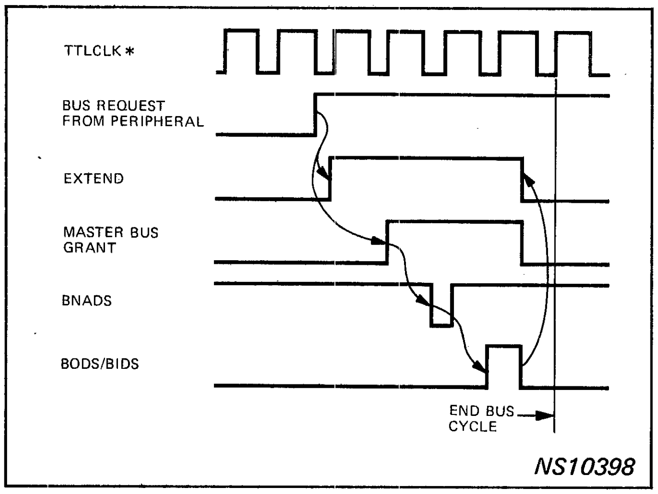


Figure 2-41. Timing Required from DMA Bus Controller

can generate a Master Bus Grant (MBG). If IDS, ODS, or NADS is active within 1.5 clock cycles, the Extend Signal should be removed until the PACE I/O cycle is complete. When generated, the Extend Signal suspends the PACE Microprocessor operation until the Extend Signal goes low. After the Extend Signal goes high and no NADS occurs for 1.5 clock cycles, the MBG Signal should go high. During the high MBG Signal, BNADS should go low. Then, the BODS/BIDS Signals should go high, as required for output/input operations between peripherals and memory. The cycle should be completed before the Extend and MBG Signals terminate. Approximately 0.5 clock cycle after BODS/BIDS terminates, the bus cycle is ended. Care should be exercised not to extend the bus cycle beyond refresh requirements of the PACE microprocessor (see data sheet).

As an alternate method, microprocessor operation may be suspended by driving the PACE NHALT Input low with an external gate. The external gate output overrides the PACE output buffer. Microprocessor operation then is suspended after execution of the current instruction. The suspension may last for an indefinite period of time without loss of CPU status and may be terminated by use of the PACE CONTIN Input (properly sequenced with removal of the NHALT Input). The timing sequence for the NHALT and CONTIN Signals is shown in figure 2-42. The NHALT and CONTIN method for suspending PACE operation can be useful for DMA block data transfers which require full bus-throughput capacity.

In the system of figure 2-40, a DM7123 quad two-input Multiplexer is used to select between the NADS, IDS, ODS, and AD14 Signals for standard system operation or the BNADS, BIDS, BODS, and MBG Signals for DMA operation. When the MBG Signal from the DMA Bus Controller is high, the Multiplexer outputs are BNADS, BIDS, BODS, and MBG. Whenever the DMA Bus Controller is inactive, the MBG and BIDS Signals are low and the Multiplexer outputs are NADS, IDS, ODS, and AD14. The Multiplexer outputs, together with the PACE F11 to F14 Flags, become the signals of the System TTL Timing and Control Bus.

The System TTL Address/Data Bus is under control of the two BTEs interfacing that bus during standard system operation. During DMA operation, the two BTEs assume a high-impedance output state and the DMA Bus Controller controls the System TTL Address/Data Bus. Memory accessing by the DMA Bus Controller is effected by the two system buses in exactly the same manner as previously described for figure 2-29.

The two BTEs are held in the high-impedance state during DMA operations in the following manner. The CE Input is connected high to V_{CC}. The high MBG Signal keeps the BTE WBD* Input high during DMA. The high MBG Input occurs before BNADS.

When BNADS occurs, the WBD*, CE1 and CE2* Inputs are high. Therefore, in accordance with the truth table of figure 2-19, when BNADS goes low at the BTE STR* Input, the BTE Outputs assume the high-impedance state.

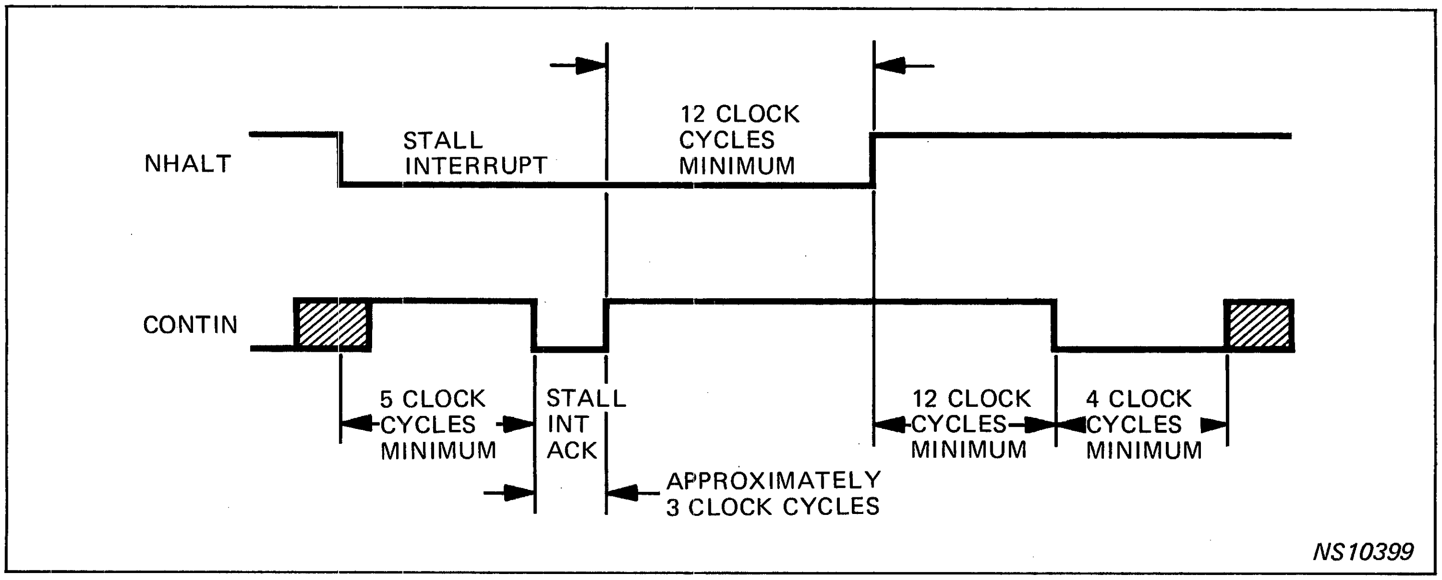


Figure 2-42. Timing Diagram for Externally Applied NHALT and CONTIN Signals

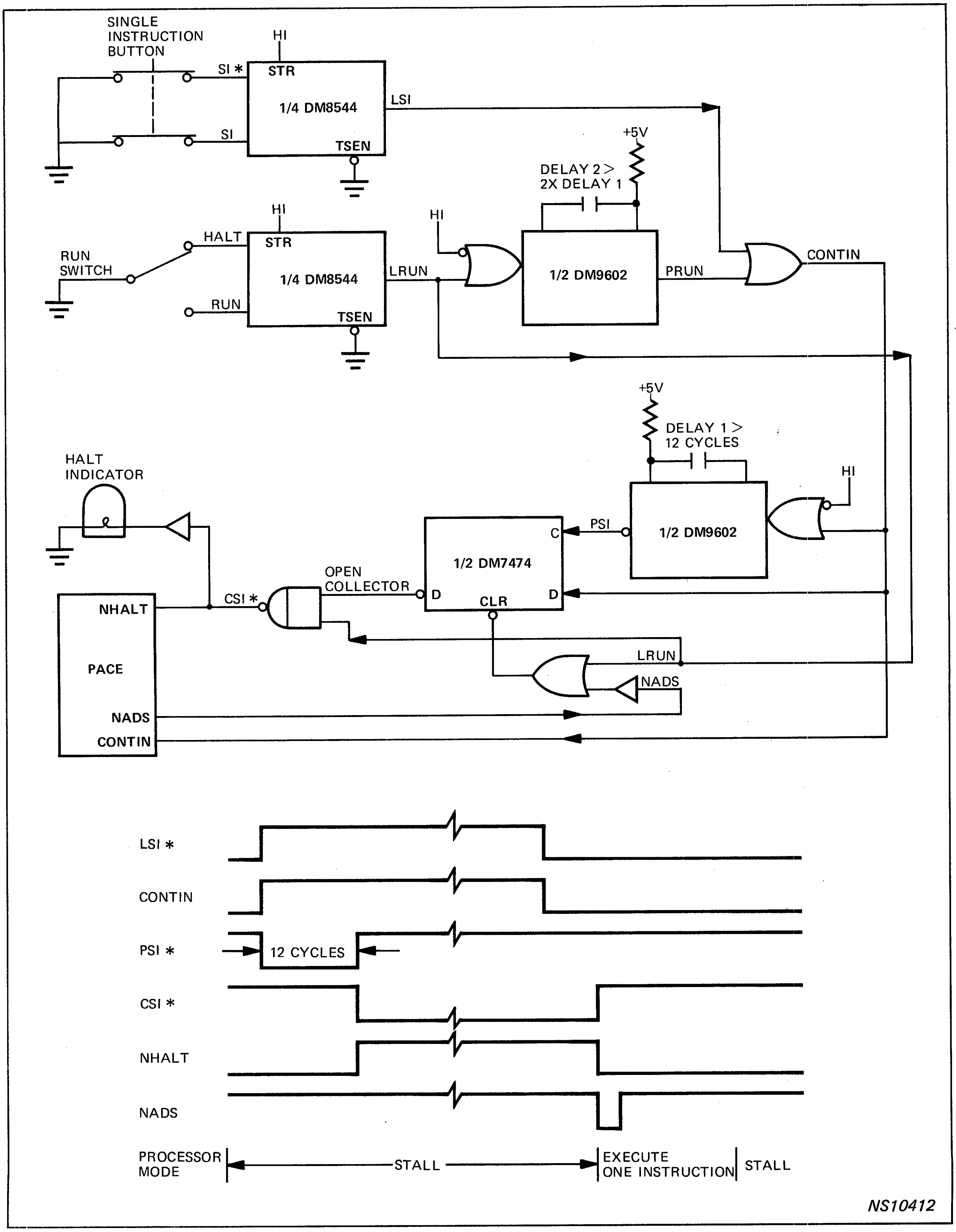


Figure 2-43. Control Panel Logic and Timing (Plus Time Line)

The high MBG Signal applied to the BTE WBD* Input keeps the BTE in the high-impedance mode until the data transfer is completed.

During standard system operation, the MBG Signal is low, so the BTE directional control is achieved through the state of IDS.

2.5.7 MINIMAL CONTROL PANEL

The previously described programmed halt and processor stall input allow the implementation of a simple Control Panel with a minimum of components. The Control Panel provides HALT, SINGLE INSTRUCTION, and RUN modes and displays data for a selected trap address or data and address for each single instruction executed.

The control logic and the associated timing are shown in figure 2-43. A one-shot is used in conjunction with the SINGLE INST Switch contact closure time (assumed greater than one-shot delay plus 12 clock cycles) to provide the proper single-instruction timing. The time between single-instruction closures is assumed to exceed the longest instruction execution time. The single-instruction sequence is terminated by generating a halt after the first NADS. (The halt must be generated in less than eight clock cycles after NADS.) The RUN mode is entered by generating the single-instruction timing and inhibiting the termination on ADS.

The data display logic is shown in figure 2-44. In the HALT mode, the address and data for each single instruction execution are displayed. In the RUN mode, switch-selected address and data are displayed each time the selected address occurs on the data bus.

While the capabilities of a simple Control Panel are limited and may not be suitable for the program development stage, a simple Control Panel is adequate in certain end applications. A very complete Control Panel facility is provided for program development purposes by the PACE Development System, IPC-16P, described in chapter 4.

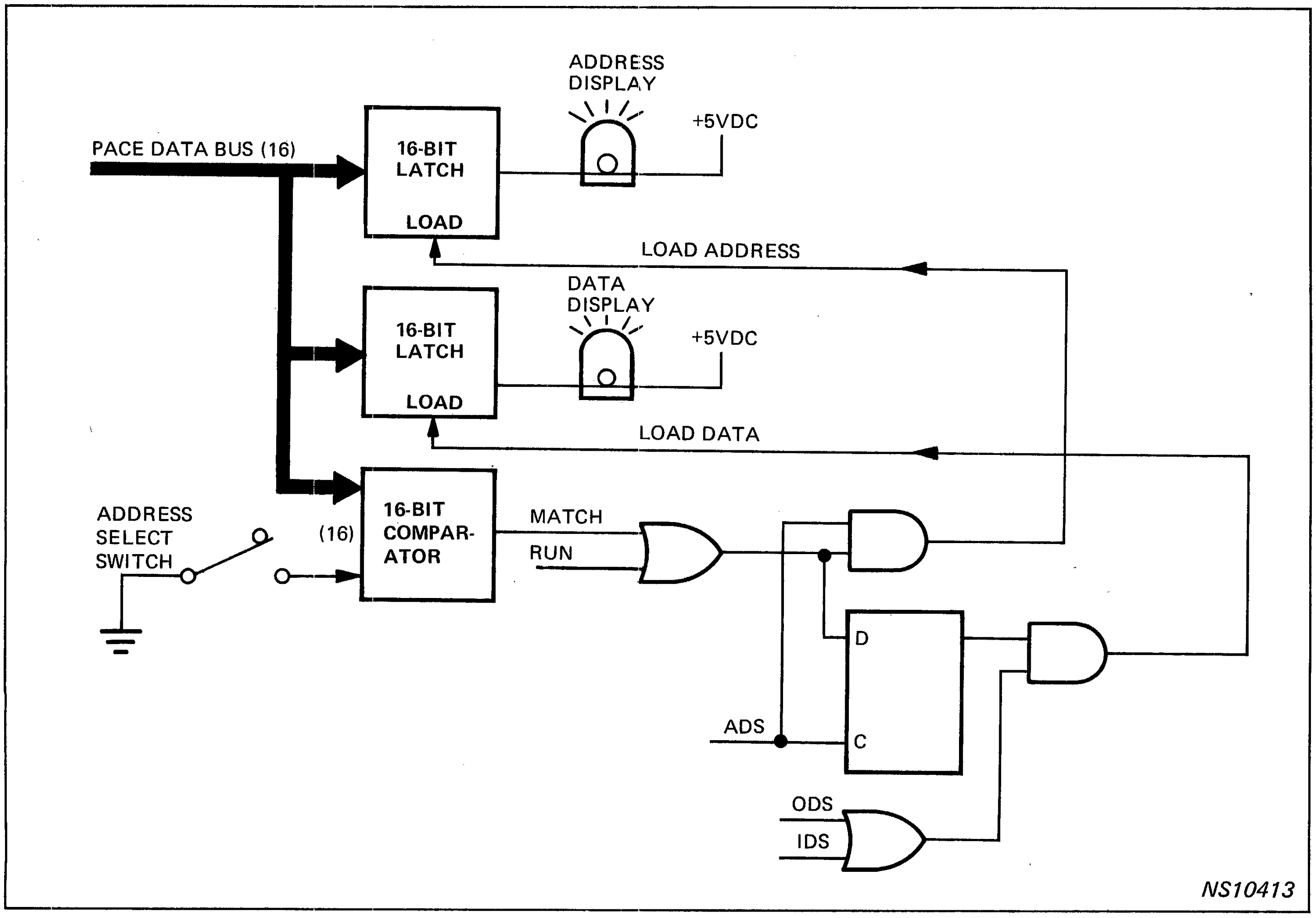


Figure 2-44. Panel Data Display Logic

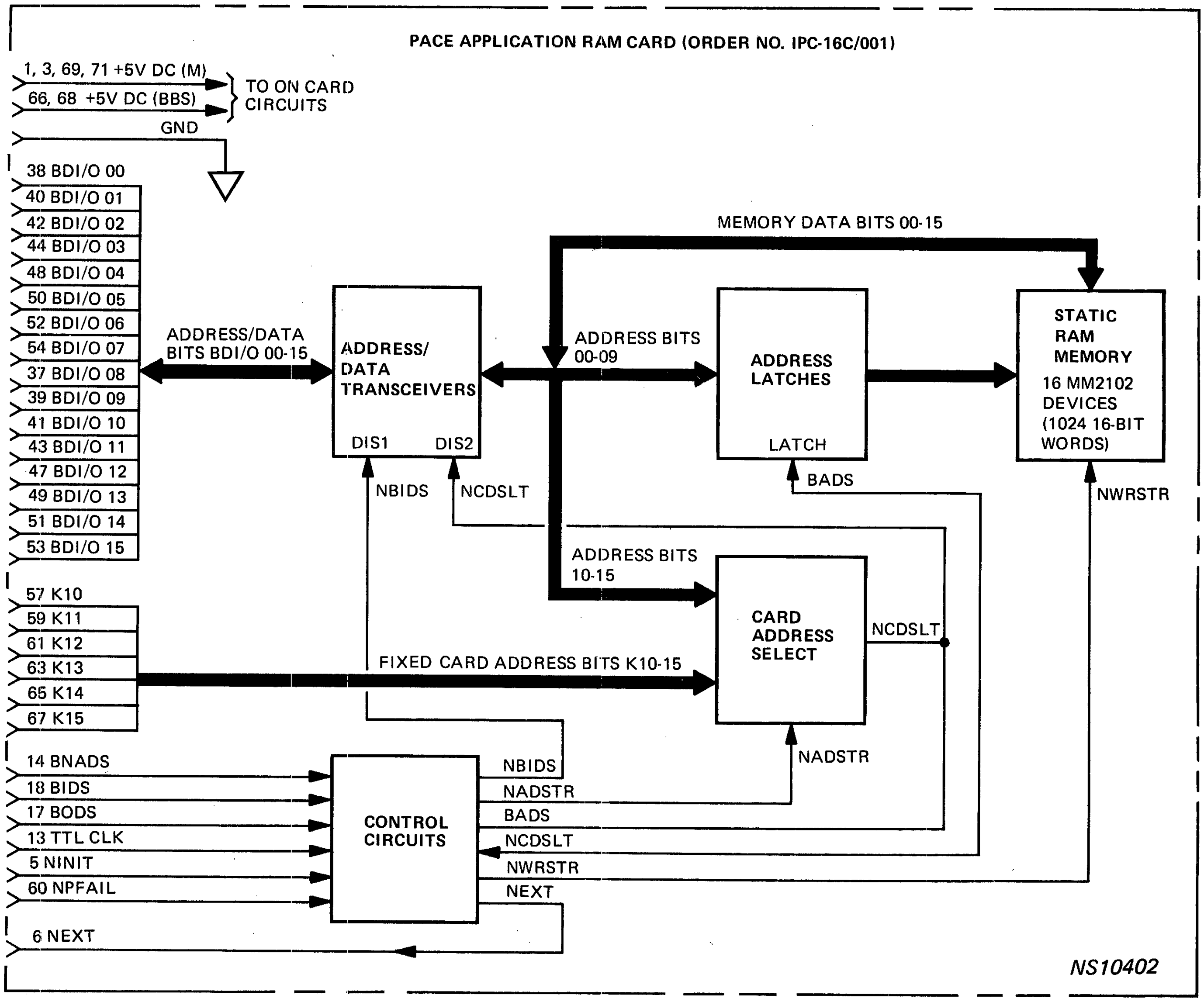


Figure 3-3. PACE Application RAM Card Functional Block Diagram

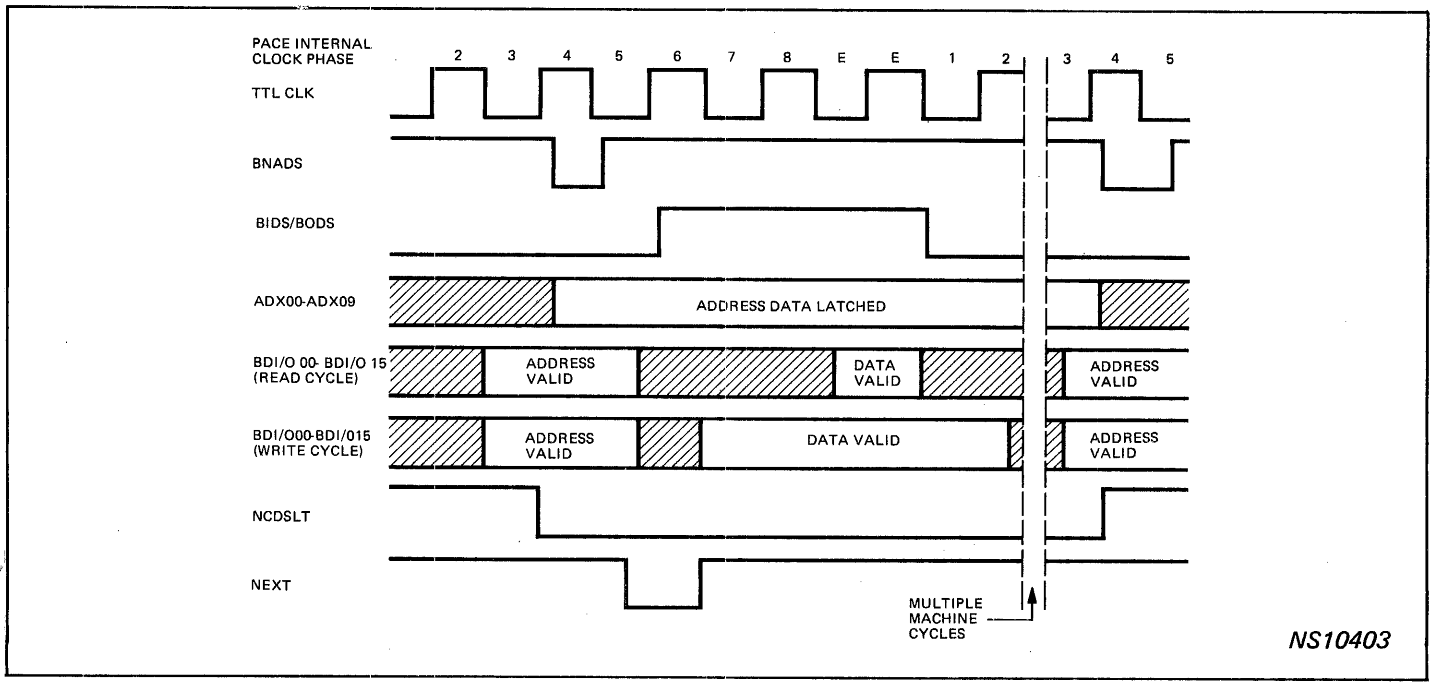


Figure 3-4. PACE Application RAM Card Timing Diagram

CHAPTER 3
PACE APPLICATION CARDS

3.1 INTRODUCTION

The following paragraphs provide information regarding the following three PACE application cards.

- PACE Application CPU Card (order number IPC-16C/100)
- PACE Application RAM Card (order number IPC-16C/001)
- PACE Application ROM/PROM Card (order numbers IPC-16C/001P and IPC-16C/001B)

The application cards are intended for small-volume end application use and prototyping. The three cards can be interconnected to form a system that, in turn, can be connected to the PACE Microprocessor Development System IPC-16P. The IPC-16P then can be used to develop software or firmware for the intended end application. Since the three cards can be used together to form the basis of a microcomputer system, engineering design time is reduced as compared to a system using integrated circuits as the basic component.

Physically each card is 4.37 inches by 4.862 inches (see figure 3-1). The small card size permits use in physically confined situations such as portable equipment applications. In addition, placing the microprocessor on one card, and the two types of memory on two individual cards, allows a modular configuration to be used in the system design.

Each circuit card is equipped with a 72-pin edge connector. Mating connectors, a line of compatible card cages, extender cards, and wire-wrap breadboard cards are available from a variety of sources.

Table 3-1 lists the sources of accessory equipment compatible with the PACE application cards.

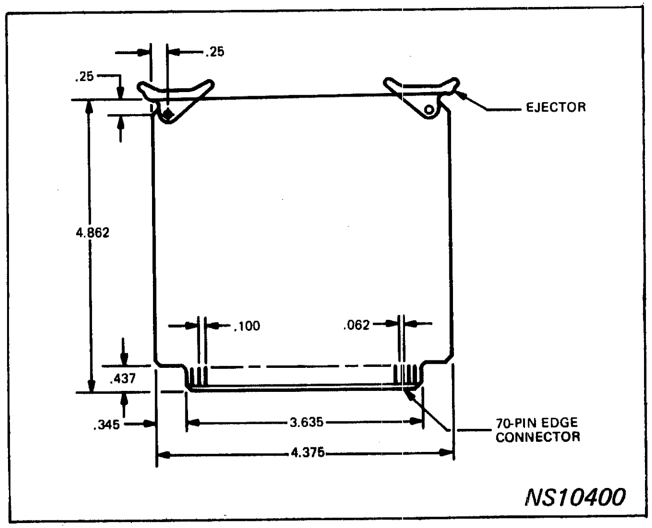


Figure 3-1. PACE Application Cards Dimensional Details

Table 3-1. Sources of Accessory Equipment for PACE Application Cards

Equipment	Source	Part Number
72-contact Edge Connector	Augat Robinson-Nugent Stanford Applied Eng. National Connector Cinch Winchester Elco Viking	14005-17P3 EC-721 CDP7000-72 900100-36 50-72C-30 HW36C0111 00-6307-072-309-001 3VH36/1JND5
13-connector Card Cage with Backplane	Augat Robinson-Nugent Scanbe	8170-MG1 MECA-1
Extender Card	Augat Robinson-Nugent	8136-MG13 EB-72
Universal w/w Card with Terminals	Augat Robinson-Nugent	8136-UMG1 UNI-24
High-density w/w Card with Terminals	Augat Robinson-Nugent	8136-MG15
Universal w/w Card without Terminals	Robinson-Nugent	(Special)

3.2 PACE APPLICATION CPU CARD

The PACE Application CPU Card contains all the circuits required to provide a CPU with TTL interface on one card. The PACE Application CPU Card contains a PACE, a System Timing Element (STE) with a 4.0-mHz crystal, three Bidirectional Transceiver Elements (BTEs), and miscellaneous circuits required for control-signal and initialization purposes. Figure 3-2 contains a functional block diagram of the PACE Application CPU Card with the card-edge connector pin assignments.

As illustrated in figure 3-2, the PACE Application CPU Card requires +5-volt dc and -12-volt dc power inputs. The substrate bias voltage (V_{BB}) and timing signals (CLK and NCLK) required by PACE are provided by the STE equipped with a 4.0-mHz crystal. The STE also provides a TTLCLK clock-reference output signal to a card-edge connector pin for system implementation, as required.

A single BTE is used as the Control Signals Buffer. The BTE is connected to provide unidirectional buffering for the PACE control signal outputs (NADS, IDS, ODS) and the user flags (F11 through F14). The control signals and user flags are applied from the Control Signals Buffer outputs to card-edge connector pins for system use.

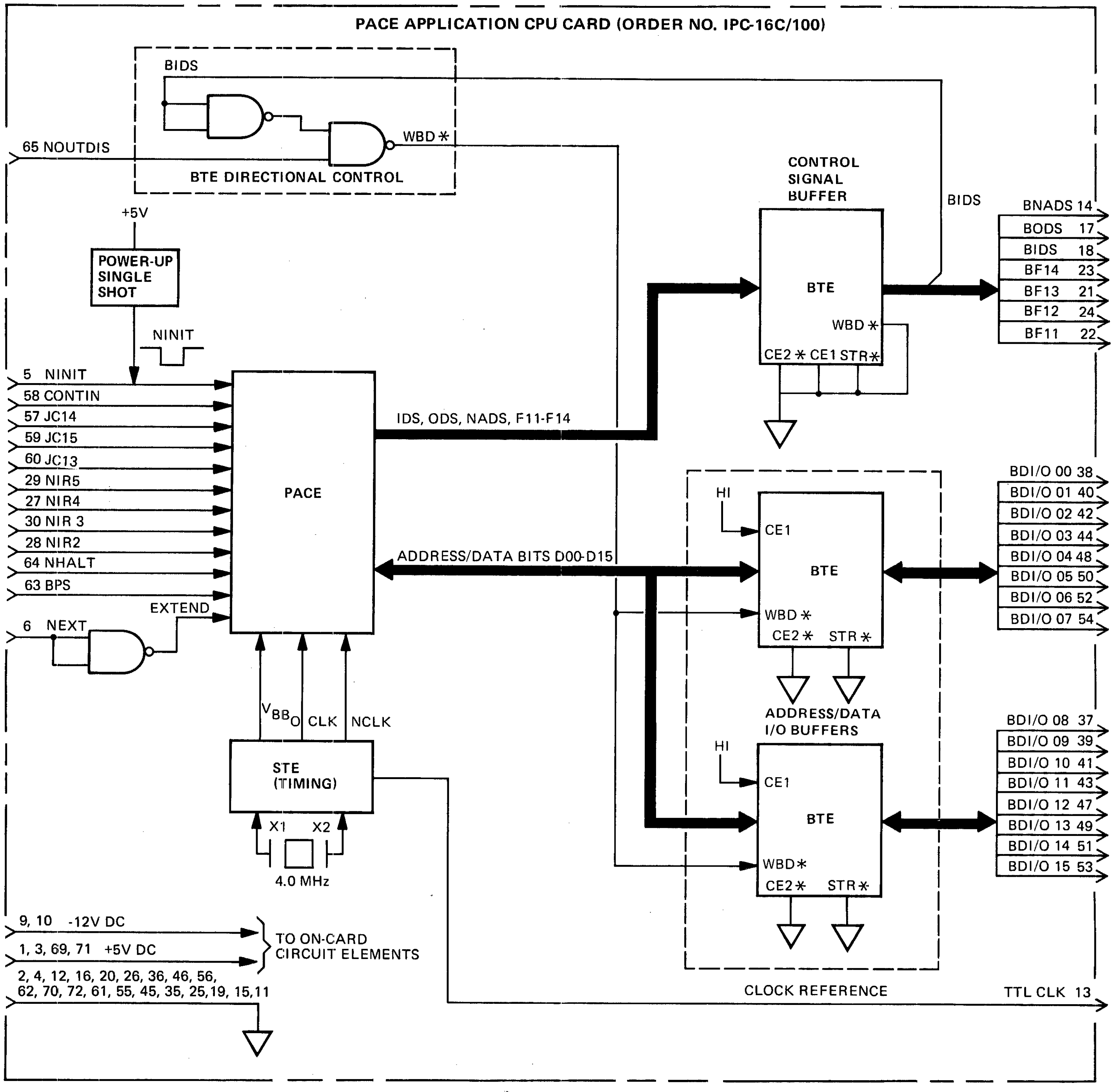


Figure 3-2. PACE Application CPU Card Functional Block Diagram

The PACE Application CPU Card uses a fully multiplexed address/data bus structure for communication between an external TTL bus and the PACE. Two BTEs function as the Address/Data I/O Buffers for the D00 through D15 System TTL Address/Data Bus Lines. The direction in which data is driven from the two BTEs is determined by the states of the input signals to two NAND gates comprising the BTE Directional Control function. When the PACE IDS Signal is high or the externally applied Output Disable✱ Signal (NOUTDIS) is low, the WBD✱ Signal from the BTE Directional Control function is high. The high WBD✱ Signal causes the BTEs comprising the Address/Data I/O Buffers to drive data from an external TTL bus to the PACE. When the WBD✱ Signal is low, data is driven from PACE to an external TTL bus.

A Power-up Single- shot is connected to the PACE NINIT Input. When power first is applied to the system, the Power-up Single- shot provides a negative-going pulse that initializes the PACE microprocessor. After the negative-going initialization pulse terminates, the PACE Application CPU Card is ready to commence operation.

Interrupt, jump condition, and control input signals for PACE are applied to card-edge connector pins and routed directly to PACE, with the exception of the Extend Signal. The Extend Signal Line includes an on-board inverter so a low NEXT Input is required to cause the PACE to extend microcycle timing for input/output operations.

CAUTION

> Maximum allowable input signal levels, specified in the PACE data sheet, must be carefully observed for all PACE input signals to the PACE Application CPU Card.

Most of the input signal lines are equipped with drilled pads so the user can add pull-up resistors, if desired. Also, the LCK✱ and LCK Inputs to the STE are equipped with drilled pads to accommodate user addition of an external feedback capacitor to vary the clock non-overlap interval. On-board test points are provided for monitoring the frequencies of the clock signals and the crystal.

3.3 PACE APPLICATION RAM CARD

The PACE Application RAM Card provides 1024 16-bit words of static read/write memory. The PACE Application RAM Card contains all the functions required for direct interfacing with the PACE Application CPU Card. A functional block diagram of the PACE Application RAM Card with the card-edge connector assignments is contained in figure 3-3. An associated timing diagram is presented in figure 3-4.

The Address/Data Transceivers (see figure 3-3) provide buffering for address information as well as buffering and directional control for the data transferred between the Static RAM Memory and the BDI/O00 to BDI/O15 bus lines. The Address/Data Transceivers contain drivers for outputting data from the PACE Application RAM Card onto the BDI/O00 to BDI/O15 bus lines. The driver outputs are in the high-impedance state unless the Control Circuits BIDS Signal and Card Select✱ Signal (NCDSLT) outputs both are low.

During address time (see figure 3-4), the PACE Application CPU Card supplies address information over the BDI/O00 to BDI/O15 bus lines and a low BNADS Signal, signifying that address information is available. The BDI/O00 to BDI/O09 address bits are passed by the Address/Data Transceivers to the Address Latches. The BDI/O10 to BDI/O15 address bits are passed to the Card Select function.

NOTE

> The Card Select function permits a unique address to be assigned to each of a number of memory cards that may be used in the same system.

The low BNADS Signal from the PACE Application CPU Card is applied to the Control Circuits which, in turn, produce a high BADS Signal output and a low NADSTR Signal output. The previously applied BDI/O10 to BDI/O15 address bit configuration is compared to the externally set K10 to K15 card address bit configuration by the Card Address Select function. If the address bit configurations are identical, the Card Address Select function NCDSLT Signal output is latched low when the NADSTR Signal goes low. The NCDSLT Signal remains low until an unsuccessful address comparison occurs while BADS is high. The high BADS Signal latches the BDI/O00 to BDI/O09 address bits into the Address Latches. The Address Latches apply the address to all 16 memory devices comprising the Static RAM Memory.

The low NCDSLT Signal simultaneously is applied to the DIS2 Input of the Address/Data Transceivers and to the Control Circuits. The Control Circuits then produce a low NEXT Signal output that is supplied to the PACE Application CPU Card. The low NEXT Signal causes the PACE microcycle time to be extended by one clock cycle for PACE data input/output operations with memory. No additional control signals are required at the user interface.

For an input operation, data is transferred from the PACE Application RAM Card to the PACE on the PACE Application CPU Card. The PACE Application CPU Card supplies a BIDS Signal to the PACE Application RAM Card Control Circuits. The Control Circuits then provide a low NBIDS Signal to the DIS1 Input of the Address/Data Transceivers and a high Write Strobe✱ Signal (NWRSTR) to the Static RAM Memory. The high NWRSTR Signal permits data to be read from the Static RAM Memory address (previously stored in the Address Latches) and transferred to the Address/Data Transceivers. The low NBIDS and NCDSLT Signals enable the Address/Data Transceivers internal drivers to drive the memory data onto the BDI/O00 to BDI/O15 bus lines to the PACE Application CPU Card.

During an output operation, data is output from the PACE Application CPU Card and written into the Static RAM Memory. The PACE Application CPU Card supplies a BODS Signal to the PACE Application RAM Card Control

Circuits. The Control Circuits NBIDS Signal output remains high and the NWRSTR Signal output goes low. Since NBIDS is high, the Address/Data Transceivers internal drivers outputs are in the high-impedance state. Thus, data cannot be driven from the PACE Application RAM Card onto the BDI/O00 to BDI/O15 bus lines, but data can be received by the Address/Data Transceivers. The BDI/O00 to BDI/O15 data received from the PACE Application CPU Card is passed by the Address/Data Transceivers to the Static RAM Memory. Since the NWRSTR Signal is low, the BDI/O00 to BDI/O15 data are written into the memory address previously stored in the Address Latches.

Application of the NINIT Signal clears the Control Circuits. The NPFAIL Input is supplied by user-developed circuits as an indication of a power failure. Should a power failure occur, the user-implemented +5-volt dc Battery Backup Supply (BBS) switches to the battery pack. The switch to the battery pack drives the NPFAIL Signal low at a time when a Write Cycle is not interrupted. The Control Circuits then preserve the current data in the Static RAM Memory by preventing any new data from being written into the Static RAM Memory despite the presence of spurious write signals. The devices comprising the Static RAM Memory and the required write inhibit logic in the Control Circuits are connected to the +5-volt dc BBS Line on the PACE Application RAM Card.

3.4 PACE APPLICATION ROM/PROM CARD

The PACE Application ROM/PROM Card is available in two configurations. One configuration, order number IPC-16C/001P, provides 1024 16-bit words of socketed static read-only memory. The memory is supplied as unprogrammed, ultraviolet erasable PROMS. The other configuration, order number IPC-16C/001B, contains all required circuits except for the memory devices. Sockets are provided on the board to accommodate either PROM or pin-compatible ROM devices. Both card configurations contain all the functions required for direct interfacing with the PACE Application CPU Card. A functional block diagram of the PACE Application ROM/PROM Card with card-edge connector assignments is contained in figure 3-5. An associated timing diagram is presented in figure 3-6.

Address information placed on the BDI/O00 to BDI/O15 bus lines (see figure 3-5) by the PACE Application CPU Card is applied to the Application ROM/PROM Card Address Latches and Card Address Select functions. The BDI/O00 through BDI/O09 address data bits are clocked into the Address Latches by the active BADS Signal (inverted BNADS Signal from the PACE Application CPU Card). The resultant ADX00 to ADX07 output address bits from the Address Latches are applied simultaneously to all eight devices comprising the Static ROM Memory. The ADX08 and ADX09 address bits are applied to the Memory Chip Select function. The BDI/O10 to BDI/O15 address bit configuration is compared by the Card Address Select function to the externally set K10 to K15 card address bit configuration. If the address bit configurations are identical, the Card Address Select function NCDSLT Signal output is latched low when the NADSTR Signal goes low. The NCDSLT Signal remains low until an unsuccessful address comparison occurs while BADS is high. The NADSTR Signal also is applied to the Control Circuits and, consequently, causes the Memory Cycle Signal (MEMCYC) output from the Control Circuits to go high. The high MEMCYC Signal is applied to the Memory Chip Select function.

The low NCDSLT Signal simultaneously is applied to the Control Circuits and the Memory Chip Select functions. The low NCDSLT Signal, together with the previously applied high MEMCYC Signal and ADX08/ADX09 address bits, cause one of the Memory Chip Select function NCS0 to NCS3 Outputs to go low. The Memory Chip Select function is a two-to-four demultiplexer that uses the configuration of the ADX08/ADX09 address bits to determine which NCS Output is driven low. The low NCS Line is applied to the Chip Select✱ Inputs (CS✱) of the two associated memory devices. Then, the two memory devices each output eight data bits (MD00-MD15) from the memory address contained in the Address Latches. The memory data output is applied to the Data Output Buffers.

When the PACE Application CPU Card provides a high BIDS Signal, the PACE Application ROM/PROM Card Control Circuits provide low NIOEXT and NDBOEN Signals to the Data Output Buffers. The Data Output Buffers internal drivers are in a high-impedance state when the NDBOEN Signal is high. When the NDBOEN Signal goes low, the Data Output Buffers internal drivers pass the MD00 to MD15 memory data onto the BD00 to BD15 bus lines. At the same time, the low NIOEXT Signal causes one internal driver of the Data Output Buffers to produce a low NEXT Signal. The low NEXT Signal is routed to PACE on the PACE Application CPU Card and causes the PACE microcycle timing to be extended one clock cycle for a data-input operation.

When the BIDS Signal from the PACE Application CPU Card terminates, the NIOEXT and NDBOEN Signals go high. Consequently, the Output Buffers internal drivers again assume a high-impedance output state. Application of a high BODS Signal or a low NINIT Signal to the PACE Application ROM/PROM Card clears the Control Circuits.

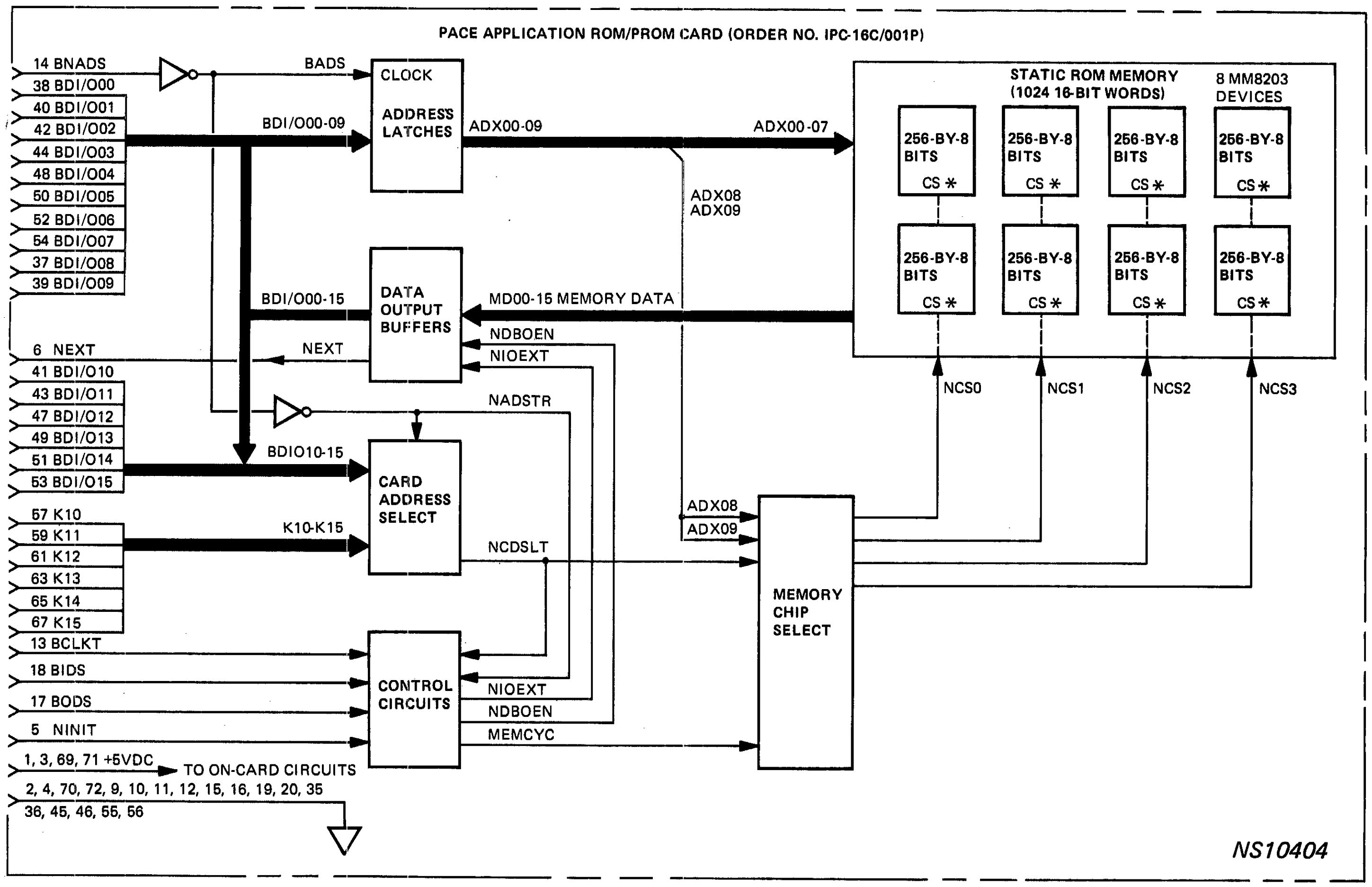


Figure 3-5. PACE Application ROM/PROM Card Functional Block Diagram

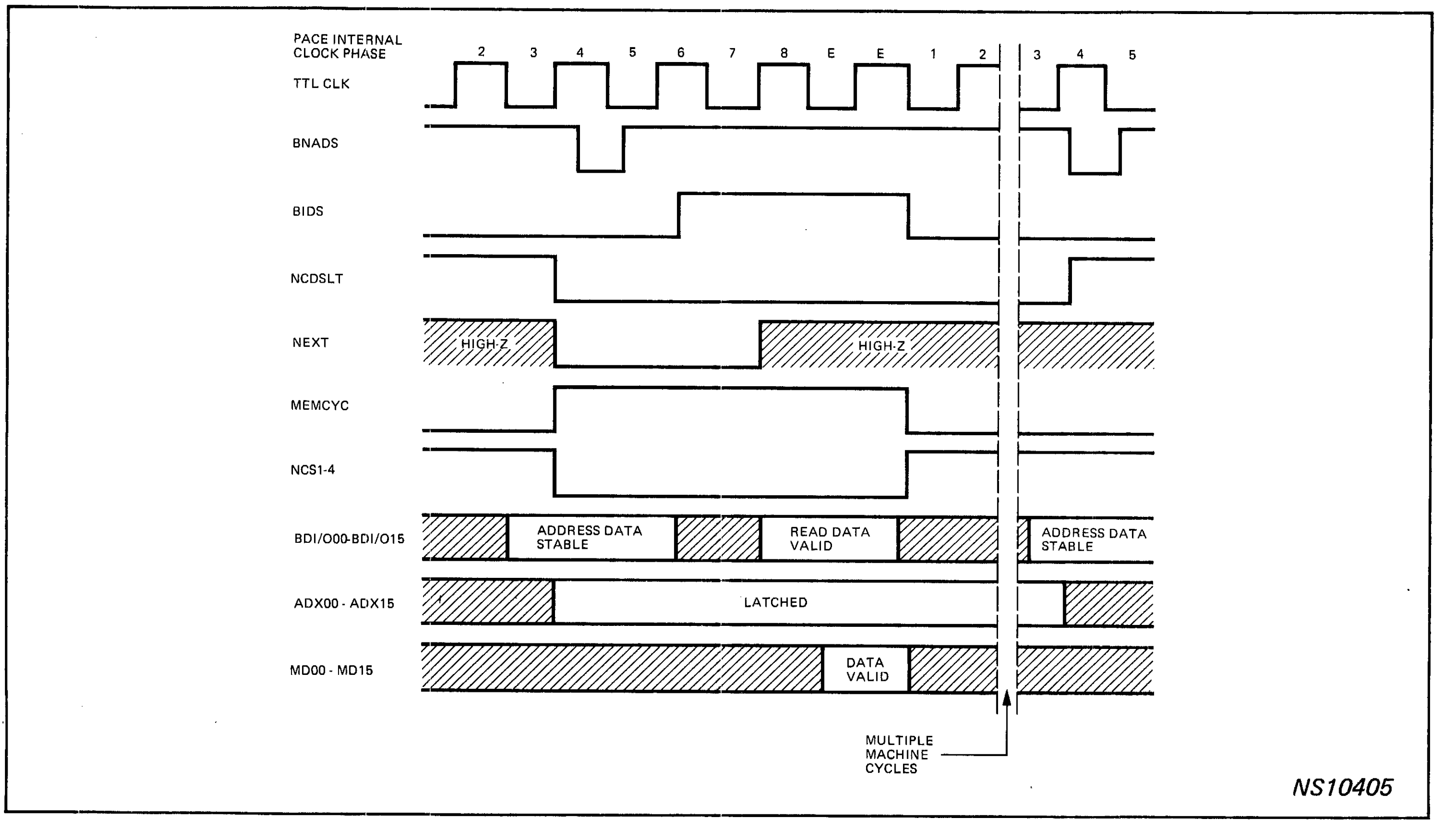


Figure 3-6 PACE Application ROM/PROM Card Timing Diagram

CHAPTER 4

PACE MICROPROCESSOR DEVELOPMENT SYSTEM

The PACE Microprocessor Development System IPC-16P is a prototyping tool designed to facilitate development of systems incorporating the PACE microprocessor. The IPC-16P (see figure 4-1) provides an economical and convenient means of expediting the development of software and hardware for the intended microprocessor system. The inherent features of the IPC-16P make the unit a flexible tool that is suitable for use in a wide variety of microprocessor system designs.

The IPC-16P is built around a PACE microprocessor that controls IPC-16P operations related to the development process. The IPC-16P, an optional teletypewriter, and the supporting PACE software package provide the user with everything required to develop, debug, and edit software or firmware to be used for an application system. With the addition of an optional PROM Programmer Card (that simply requires insertion into a prewired unoccupied IPC-16P card slot), the user can store even the developed software in PROMS for application system use.

The Programmers Control Panel (see figure 4-2) at the front of the IPC-16P contains control switches and two banks of 16 LED indicators that facilitate software development. The switches and indicators can be used to examine and modify, as desired, the contents of selected PACE CPU Registers or any memory storage location. In addition, operational control of the PACE microprocessor is effected at the Operators Control Panel (see figure 4-3). The following lists briefly describe the functions provided by the controls and indicators on the Programmers and Operators Control Panels.

Programmers Control Panel

- PROGRAM COUNTER/MEMORY ADDRESS (16 LED Indicators) --- Indicate current contents of PACE Program Counter, contents of 10 Stack levels or current memory address, depending upon position of Display Selector Rotary Switch.
- DATA DISPLAY (16 LED Indicators) --- Display information as selected by Display Selector Rotary Switch.
- DATA ENTRY (16 Two-position Switches) --- Set data to be loaded into PACE Registers or memory as specified by Display Selector Rotary Switch position.
- LOAD DATA (Momentary Switch) --- Loads data set on DATA ENTRY Switches into location selected by Display Selector Rotary Switch.
- SINGLE INST (Momentary Switch) --- Each time depressed, permits execution of only one instruction when PACE CPU is halted.
- INCR MEM ADDR (Momentary Switch) --- Each time depressed, current memory address is incremented by one, or each Stack level is displayed on DATA DISPLAY Indicators as determined by position of Display Selector Rotary Switch.
- DISPLAY SELECTOR ROTARY SWITCH (11-position Switch) --- Selects for display the PACE Registers AC0, AC1, AC2, AC3, Program Counter, Stack, and Flags, or the next instruction, current memory address, or memory data.

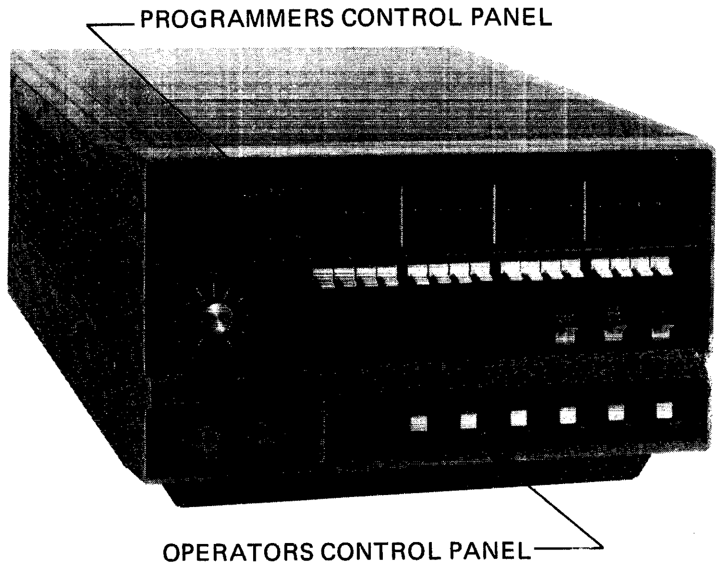


Figure 4-1. PACE Microprocessor Development System IPC-16P

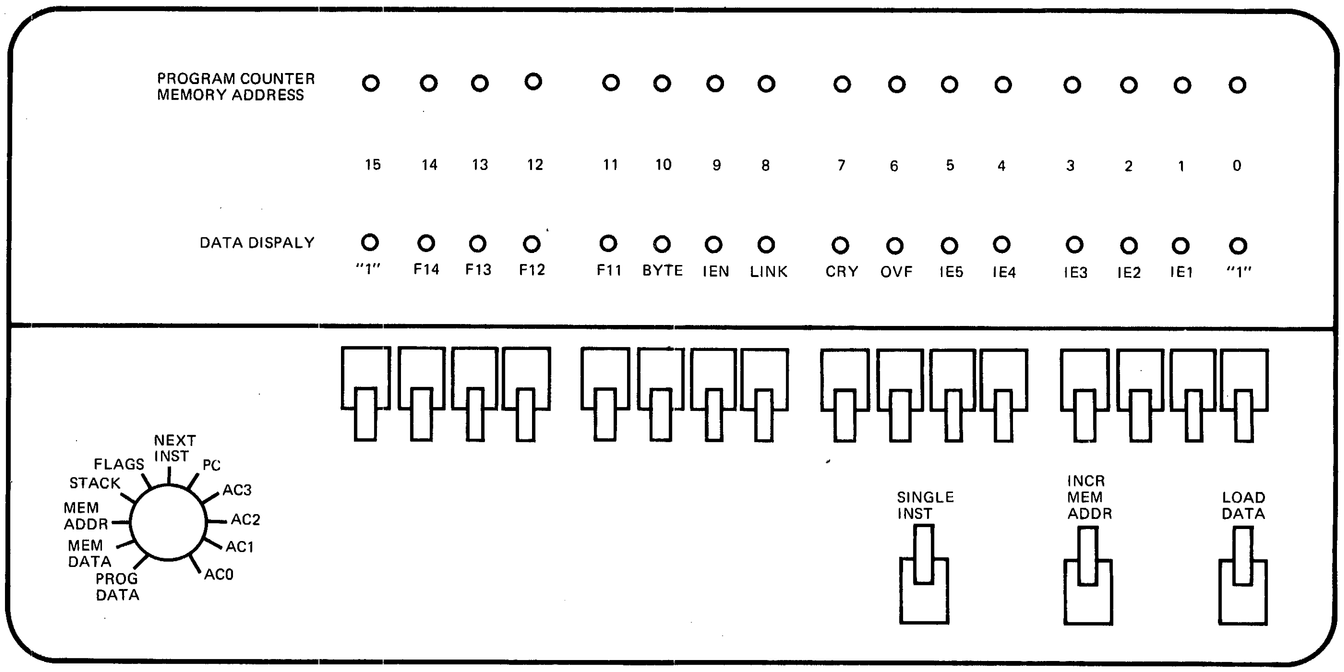
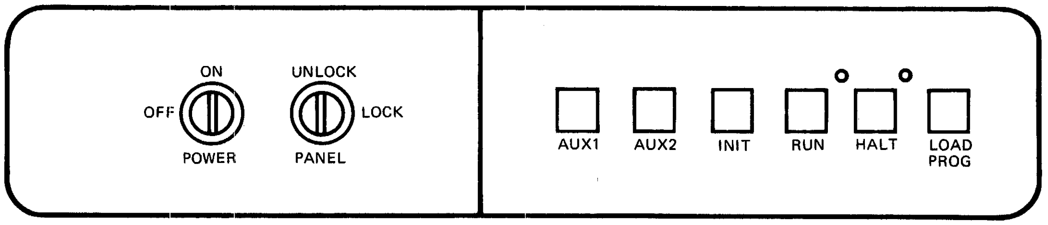


Figure 4-2. IPC-16P Programmers Control Panel

NS10415



NS10416

Figure 4-3. Operators Control Panel

Operators Control Panel

- POWER (Two-position Keyswitch) --- Prevents unauthorized application or removal of system power.
- PANEL (Two-position Keyswitch) --- Prevents unauthorized access to Programmers Control Panel that could change data under development.
- AUX1, AUX2 (SPDT Pushbutton Switches) --- For user implementation, as required.
- INIT (SPDT Pushbutton Switch) --- Initializes PACE microprocessor but does not alter memory contents.
- RUN (SPDT Pushbutton Switch) --- Causes program execution to commence at location specified by PACE Program Counter contents.
- RUN (Lamp Indicator) --- Lights when program is running.
- HALT (SPDT Pushbutton Switch) --- When pressed, program execution terminates at end of current instruction.
- HALT (Lamp Indicator) --- Lights when program is halted.
- LOAD PROG (SPDT Pushbutton Switch) --- When pressed, forces program branch to firmware that loads data from Paper Tape Reader.

The IPC-16P contains all required dc power supplies and a card cage that houses the cards used by the unit. The standard power supply for the IPC-16P contains +5-volt dc and -12-volt dc secondary sources. The +5-volt dc source is capable of delivering 18 amperes, maximum. The -12-volt dc source is capable of delivering 3.4 amperes, maximum. Table 4-1 lists the circuit card assemblies that can be used in the IPC-16P and shows the current requirements for each card. If expansion beyond the capabilities of the standard power supply is desired, an optional heavy-duty power supply is available. The optional heavy-duty power supply is capable of delivering 30 amperes, maximum, at +5 volts dc and 10 amperes, maximum, at -12 volts dc. Figure 4-4 shows the location and types of cards used in the IPC-16P. In appendix A, table A-2 lists the major units contained in the basic IPC-16P and provides a brief description of each item. In appendix A, table A-3 lists the available options for the IPC-16P and also provides a brief description of each item.

As can be seen by the description of the PACE Development CPU Card in table A-2 of appendix A, the IPC-16P can be connected to an application system during development of the required software. Figure 4-5 illustrates the bus connection to the application system as well as the IPC-16P internal bus structure.

Table 4-1. Power Requirements of Circuit Cards Usable in IPC-16P

Circuit Card Assembly	Power Supply Current Required (Amperes)	
	+5 volts	-12 volts
PACE Development CPU Card	2.8	0.2
PROM Programmer Card	1.2	1.5
Programmers Panel Card	2.0	–
Control Panel Interface Card	1.5	0.25
Memory Timing and Control Card	1.0	–
Memory Storage Card (4096 16-bit words)	1.4	0.9
TTY/Card Reader Interface Card	2.0	0.1
High-speed Printer Interface Card	1.0	–
Floppy-disc Interface Card	1.5	0.2

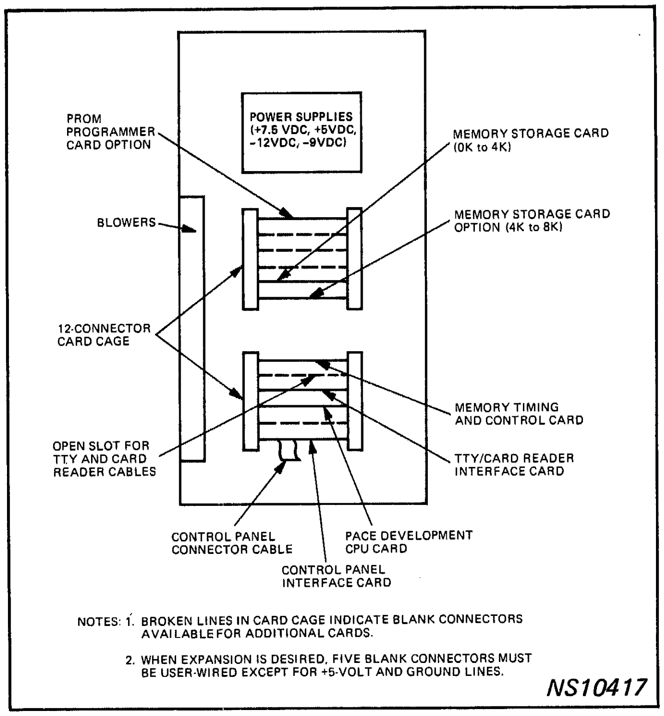

PROM PROGRAMMER CARD OPTION
POWER SUPPLIES (+7.5 VDC, +5VDC, -12VDC, -9VDC)
MEMORY STORAGE CARD (0K to 4K)
BLOWERS
MEMORY STORAGE CARD OPTION (4K to 8K)
12-CONNECTOR CARD CAGE
MEMORY TIMING AND CONTROL CARD
OPEN SLOT FOR TTY AND CARD READER CABLES
TTY/CARD READER INTERFACE CARD
CONTROL PANEL CONNECTOR CABLE
PACE DEVELOPMENT CPU CARD
CONTROL PANEL INTERFACE CARD
NOTES: 1. BROKEN LINES IN CARD CAGE INDICATE BLANK CONNECTORS AVAILABLE FOR ADDITIONAL CARDS.
2. WHEN EXPANSION IS DESIRED, FIVE BLANK CONNECTORS MUST BE USER-WIRED EXCEPT FOR +5-VOLT AND GROUND LINES.
NS10417

Figure 4-4. Top Internal View of IPC-16P

The IPC-16P provides two input/output interfaces by virtue of a multiplexer and the associated circuit elements contained on the PACE Development CPU Card. One interface is compatible with the PACE Application CPU Card previously described in chapter 3 of this document. The cable from the IPC-16P plugs into the socket on a paddle card which occupies the CPU card slot normally occupied by the PACE Application CPU Card. The other interface is used for the memory, Control Panel, and peripheral interfaces contained in the IPC-16P. The IPC-16P interface is not intended for interfacing the users application hardware to user peripherals or memory.

The IPC-16P bus structure at the interface between the Microprocessor Development System and the PACE Prototyping CPU Card (see figure 4-5) permits the expansion flexibility required to accommodate diverse Microprocessor Development System configurations. Four buses are available for use by peripherals, add-on memory, or special circuits. The Peripheral Data Bus is provided to route data from peripherals to the PACE microprocessor. The Memory Data Bus transfers data from memory to the CPU. The CPU Buffered Data Bus transfers CPU data to peripherals and memory. The Buffered Address Bus is used to address

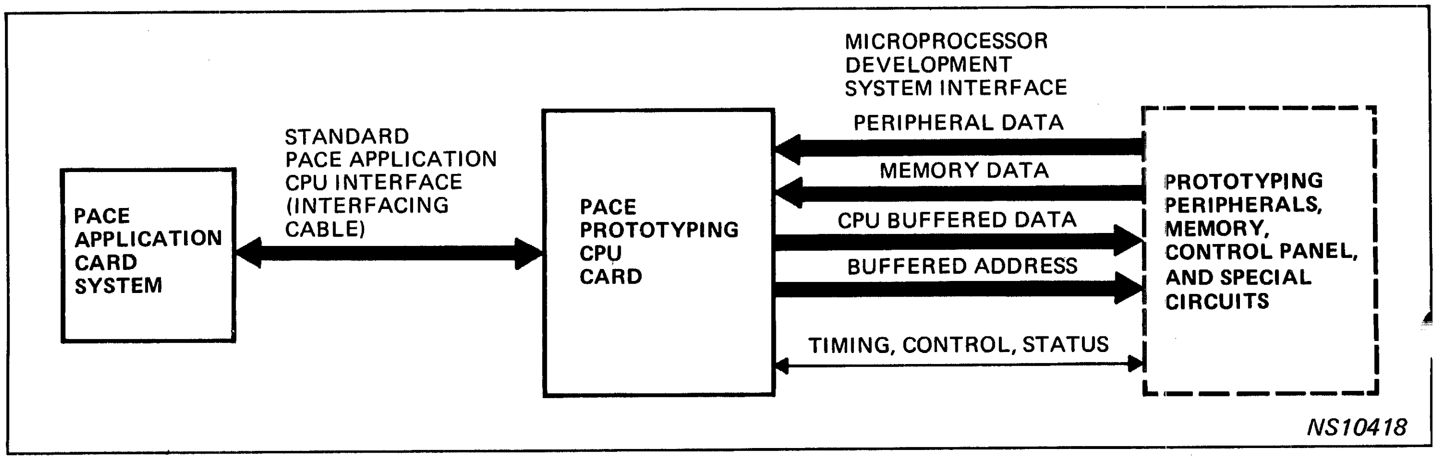

MICROPROCESSOR DEVELOPMENT SYSTEM INTERFACE
PACE APPLICATION CARD SYSTEM
STANDARD PACE APPLICATION CPU INTERFACE (INTERFACING CABLE)
PACE PROTOTYPING CPU CARD
PERIPHERAL DATA
MEMORY DATA
CPU BUFFERED DATA
BUFFERED ADDRESS
TIMING, CONTROL, STATUS
PROTOTYPING PERIPHERALS, MEMORY, CONTROL PANEL, AND SPECIAL CIRCUITS
NS10418

Figure 4-5. IPC-16P Generalized Bus Structure

peripherals and memory. Timing, control, and status flag signals are interchanged between the PACE microprocessor and peripherals to achieve proper software execution and peripheral device control.

The software that is available for use with the IPC-16P to support the development process is discussed generally in chapter 1 of this document. The specific programs are listed and briefly described in appendix A, tables A-2 and A-3.

The development process normally involves three different phases after the basic hardware design is completed and the software flowcharts are developed and coded on a first-pass basis. During the first phase, as much software as possible is debugged while using the IPC-16P on a stand-alone basis. The IPC-16P read/write memory is used for program storage, thereby permitting changes to be made rapidly as program errors are discovered.

The second phase of design begins after all possible software is debugged while using the IPC-16P on a stand-alone basis. The IPC-16P then is connected to the application system and the software portions that interact directly with the application hardware are debugged.

For the third and final phase of development, the interconnecting cable is disconnected from the application system. The application system then is checked out using the PACE Application CPU Card with the PACE microprocessor installed. If difficulties are encountered, the PACE microprocessor may be removed from the PACE Application CPU Card. The PACE paddle card then can be reconnected to the IPC-16P for further evaluation of the software design.

CHAPTER 5
PACE SUPPORT FUNCTIONS

5.1 TECHNICAL CONSULTATION

There is more to microprocessing than microprocessors is particularly applicable to the before-and-after sales support provided by National Semiconductor. Your initial inquiries regarding PACE are handled by a factory sales representative who, oddly enough, is schooled to solve problems in terms of money and in terms of utility. And, after all, that is what PACE with its supporting chips is all about – a powerful tool used to solve your problems, frequently, even those problems that formerly seemed to defy solution. Thus, in the first phase of the technical consultation, the question, *Can PACE solve my problem?* is asked by you and answered by us – usually, the answer is YES.

Once your microprocessing needs are defined, the next questions arise.

a. How can PACE solve my problem?

b. How soon can PACE solve my problem?

c. Is the PACE solution the one I want?

There are many ways PACE and its supporting chips can be used to solve a particular problem. The sales representative works continually with our applications engineers and other highly specialized technical personnel to solve your particular problem; that is, how PACE can best serve you. Your application may require lots of hardware and little software, little hardware and lots of software, or, generally, some in-between compromise. With a design capability, it may be convenient for you to design your own PACE system – input/output chips, control chips, memory chips, and other peripheral chips, all part of the PACE support family, are designed for this very purpose. If design is not your bag, then let us marry our technical expertise with your application to produce a PACE system that is uniquely qualified to do your job. Don't misinterpret *unique* to mean *expensive,* it doesn't; it means simply that ingenious hardware has been optimally combined with creative software to meet a particular need.

How long does all of this take? Well, PACE and most supporting chips are available now, so it is really a matter of *when to start,* and, of course, that is up to you. Last, but not least, *Is PACE the way to go?* Even our competitors would think twice before saying NO to this question and believe it – we can provide a lot of good reasons for saying YES.

Let us now look at specifics in the overall consultation/product-support chain that links National Semiconductor to you, the user.

5.1.1 MICROPROCESSOR SPECIALIST

The area sales representative who responds to your initial inquiry is ably assisted by a *microprocessor specialist.* This individual is equipped technically to help analyze your application, translate your needs into a viable hardware/software configuration, and then follow it through to system delivery. Even though the specialist knows microprocessing systems, in fact, knows them very well, he still must have the best of tools and these he has in PACE and its supporting family of chips – in addition to the broad line of semiconductor products manufactured by National Semiconductor.

Over and above the hardware tools, the field specialist has instant access to a very select assortment of people-tools; these include experts in applications, design, manufacturing and marketing. In short, the microprocessor specialist provides a flexible technical interface that can assist you, the user, in any part of the microprocessing spectrum – from very simple to very complex applications.

To best serve our customers in the U.S.A. and nearby territories, each of four general areas are divided into regions with each region being serviced by at least one microprocessor specialist. Likewise, our international customers have access to the same expertise (in their native language) from a number of points on the globe – Germany for European customers, Japan for Asian customers, Scotland for United Kingdom customers, Australia for customers in that part of the world, and our U.S. offices serve our good neighbors South of the border, while our Canadian neighbors are served either from Canadian offices or U.S. offices. As you can see, PACE and its supporting family of chips are multilingual and have no geographic barriers.

5.1.2 APPLICATIONS SUPPORT

So far we have met people in the field whose primary concerns are sales, system delivery, and operating integrity of the delivered equipment. Let us now examine the next link in the product-support chain – *how to use microprocessors.* It is not practical or economically feasible to have a factory consultant stay with each microporcessor we sell, so we do the next best thing. Our applications engineers have anticipated most of your problems and have generated application notes that, in most cases, will provide a solution. Not only is the application note a problem-solving device, it also is a functional tool that may open up areas of use not previously explored. The applications service is free and we urge you to use it – *think of it as free manpower because that is exactly what it is.*

5.2 TRAINING

He who trains is he who understands and to this purpose, National Semiconductor operates three training centers. The Eastern center is located in Miami, Florida; the Midwestern center in Dallas, Texas; and the Western center is located near San Francisco, in Santa Clara, California. Each training center is fully equipped and professionally staffed to provide students with a good mix of *hardware/software* theory and *hands-on* laboratory experience. Course curriculums vary in complexity from *Fundamentals of Microprocessors* for those technical personnel who have never worked with programmable systems to *Advanced Programming* for those who have microprocessor backgrounds.

Currently, the following courses are offered:

MICROPROCESSOR FUNDAMENTALS – This course is designed for the engineer, technician, or manager who is not familiar with programmable systems. It covers stored program concepts, number systems, logic, input/output control, use of standard software (assemblers, editors, loaders, debug, subroutine packages), simple programming concepts, and an overview of available microprocessors with a guideline of how to select a microprocessor for a specific application. There are no prerequisites for this course, but a knowledge of digital design techniques, binary numbers, hexadecimal numbers, and Boolean algebra will be helpful.

IMP-16/PACE APPLICATIONS – This is an in-depth course covering the IMP-16 and PACE microprocessors. Subjects covered are architecture, instruction set, input/output structure, interface design, applications design with chip sets, applications design with prepackaged cards, use of development systems, available peripherals, standard and optional software. Lab time is emphasized using experimental peripheral devices, development systems (including disc operating systems), microprocessor cards, and PROM programmers. Prerequisites: knowledge of basic microprocessor concepts, use of standard software such as assemblers and utilities, some exposure to assembly-language programming, and some knowledge of interfacing techniques. Anyone unversed in any of these subjects should attend the Microprocessor Fundamentals course before attending this course.

ADVANCED PROGRAMMING – Many engineers and programmers are finding that programming a microprocessor for a real-time application is considerably different from programming a minicomputer for a data-processing job. This course is designed for the engineer or programmer who must write complex applications software. Some of the subjects covered are real-time concepts, fixed-frequency events, time-of-day events, random external events, interrupt programming, real-time subroutines, program-called subroutines, programming complex math functions, hardware/software trade-offs, and system timing considerations. The IMP-16 and PACE microprocessors are used as the training machines in this course. Numerous examples are given and the student is required to solve numerous problems in the training center's lab. Prerequisites: thorough understanding of microprocessor fundamentals and characteristics; an understanding of assemblers, editors, debugs, loaders, and subroutine libraries; experience in programming at the assembly-language level.

NOTE

The Microprocessor Fundamentals course alone is not adequate preparation for this course. However, anyone who has the experience described but would like some refresher training before attending this course should attend the IMP-16/PACE Applications course.

All courses include four days (Monday through Thursday) of class schedules and a fifth day where the laboratory is opened to students for additional hands-on experience and for individual consultation with instructors.

Tuition for each course is $395.00, payable when enrollment is accepted. Enrollments should be made at least two weeks in advance of scheduled class start. Before remitting tuition, we suggest you check with the appropriate training center (listed below) to be sure space is available in the desired course. To ensure adequate facilities, class sizes are limited.

Training center addresses and telephone numbers are as follows:

Eastern Microprocessor Training Center
National Semiconductor Corporation
2721 Bayshore Drive South, Suite 121
Miami, Florida 33133
Telephone: (305) 446-8309

Central Microprocessor Training Center
National Semiconductor Corporation
13773 North Central Expressway, Suite 1132
Dallas, Texas 75231
Telephone: (214) 690-4552

Western Microprocessor Training Center/470
National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051
Telephone: (408) 732-5000, Extension 7183

5.3 FACTORY SERVICE

It would be nice to say that PACE and its supporting chips never fail and are never damaged; however, this would not be a credible statement. Nonetheless, failures are rare and the reasons are worth mentioning. In many companies, quality control occurs at the end of the design and manufacturing cycle; hence, they check overall performance of the chain without much regard for the individual links. At National Semiconductor, we believe in the old adage, *no chain is stronger than its weakest link* and react accordingly. Quality control is an integral part of each link in the chain – from concept through completion. Even after each link meets performance specifications, the end product is again checked for integrity of operation and, after a prolonged burn-in, it is further checked for reliability.

According to Murphy's Law, some failures will still occur in the field. If it happens to you, here is what you can expect from us. Upon receipt of the failed card or system, we strive for a turn-around time of five working days and, in most cases, we are successful. If the equipment is under warranty, there is no charge for repairs; you pay the freight one way, and we pay it the other. If equipment is not under warranty, you are charged at the rate of twenty-five dollars an hour plus parts and applicable tax; the minimum charge is fifty dollars. Freight costs are shared, just as they are for in-warranty returns.

Questions you may have should be addressed to personnel that will provide the fastest response: may we suggest the sales representative for anything relating to price and delivery and your microprocessor specialist as a first source for technical questions. Feel free to contact applications engineering at National Semiconductor for those technical questions the specialist cannot answer.

5.4 USER GROUP

National Semiconductor sponsors COMPUTE (Club of Microprocessor Programmers, Users, and Technical Experts). This user group is dedicated to the world-wide distribution of your ideas and techniques relating to the use of microprocessors. Members of COMPUTE communicate on a regular basis by way of *The Bit-Bucket* newsletter published by National Semiconductor. In the newsletter, you will find everything from soup to nuts — even a user-submitted software library. So get involved with PACE and COMPUTE; they make an excellent partnership. You can meet the former by calling your nearest sales representative and the latter by writing to:

COMPUTE/470
National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051
Telephone: (408) 732-5000, Extension 7183

Don't forget the */470* in the address. That's our mail stop, and your letter will be delayed (or worse yet, lost) without it.

APPENDIX A
PACE HARDWARE/SOFTWARE PRODUCT SUMMARIES

Table A-1. PACE Hardware Product Line

Product	Description	Order Number
PACE Microprocessor (PACE CPU)	MOS/LSI single-chip 16-bit microprocessor that uses 16-bit instructions and addressing to operate on 8- or 16-bit data. Also includes: 10-level Last-In/First-Out Stack for data or status storage; 45 instruction types; common memory and peripheral addressing; 4 general-purpose accumulators; 6 automatic-interrupt priority levels; and programmer-accessible status register.	IPC-16A/500
Bidirectional Transceiver Element (PACE BTE/8)	8-bit bidirectional bus transceiver in 24-pin DIP. Provides interfacing between PACE and system TTL bus.	IPC-16A/501
System Timing Element (PACE STE)	Provides (in single package) MOS system clocks and substrate voltage for PACE. Also, provides TTL system clock signal for user purposes. Requires only external crystal.	IPC-16A/502
Interface Latch Element (PACE ILE/8)	8-bit bidirectional buffered latch in 24-pin DIP. Provides latched interfacing between system TTL bus and peripherals.	IPC-16A/503
PACE RAM/1K	256-by-4-bit static read/write memory with on-chip address latches and bidirectional data lines in 24-pin DIP.	IPC-16A/504
PACE ROM/16K	1024-by-16-bit mask-coded Read-Only Memory with on-chip address latches in 40-pin DIP. Ultimate low-cost control store solution for medium- and high-volume systems.	IPC-16A/505
PACE PROM/4K	512-by-8-bit ultraviolet-erasable Programmable Read-Only Memory in 24-pin DIP. Low-cost control store solution for low-volume applications.	IPC-16A/506
PACE ROM/4K	512-by-8-bit mask-coded Read-Only Memory that is pin-for-pin compatible with IPC-16A/506. Provides low-cost control store solution for medium- and high-volume systems.	IPC-16A/507
Address Latch Element (PACE ALE/8)	8-bit address latch in 24-pin DIP for systems employing memory and/or peripheral devices requiring latched address.	IPC-16A/508
Interface Latch Element (PACE ILE/16)	16-bit bidirectional buffered latch in 40-pin DIP. Provides latched interfacing between system TTL bus and peripherals.	IPC-16A/513
Address Latch Element (PACE ALE/16)	16-bit address latch in 40-pin DIP for systems employing memory and/or peripherals requiring latched addresses.	IPC-16A/518
PACE Application CPU Card	4.375-inch by 4.862-inch printed circuit card containing PACE microprocessor, clock circuit, input/output buffering, and all required signal pullups and power-on initialize control. Designed for use with RAM, ROM/PROM, and Input/Output Interfacing Cards on OEM equipment or for system prototyping.	IPC-16C/100
PACE Application RAM Card	4.375-inch by 4.862-inch printed circuit card that provides 1024 16-bit words of static read/write memory. Contains all functions required to interface directly with PACE Application CPU Card.	IPC-16C/001
PACE Application ROM/PROM Card	4.375-inch by 4.862-inch printed circuit card that provides 1024 16-bit words of ROM or PROM storage. Contains all functions required to interface directly with PACE Application CPU Card.	IPC-16C/001P

Table A-1. PACE Hardware Product Line (Continued)

Product	Description	Order Number
PACE Application ROM/PROM Card	4.375-inch by 4.862-inch printed circuit card that contains sockets for eight 256-by-8-bit PROM or ROM elements. Contains all functions required to interface directly with PACE Application CPU Card.	IPC-16C/001B
PACE Application Input/Output Card	4.375-inch by 4.862-inch printed circuit card containing four 8-bit bidirectional input/output port elements. Input/Output ports are user configurable.	IPC-16C/800
PACE Development CPU Card with Interfacing Cable, Firmware, Software, and Documentation	8.5-inch by 11-inch printed circuit card containing: PACE microprocessor; clock circuit; input/output buffers; address latches; data multiplexer for memory and peripherals; and all required interfacing for Control Panel, clocks, memory refresh, and read/write flags. Designed for use in PACE Microprocessor Development System IPC-16P or, as part of field conversion kit, in IMP-16P.	IPC-16P/100
IMP-16P/PACE Conversion Kit	PACE Development CPU Card with interface cable, firmware, software, and documentation required to convert IMP-16P to IPC-16P.	IPC-16P/100KC (software on paper tape) - specify memory capacity. IPC-16P/100KQ (software on punched cards) - 8K memory, minimum, required.
PACE Microprocessor Development System	12-inch by 17-inch by 24-inch rack-mountable cabinet containing PACE Development CPU Card, Control Panel, Control Panel Interface Card, TTY/Card Reader Interface Card, Memory Timing and Control Card, 4096-word by 16-bit Memory Storage Card, and all required dc power supplies. Basic PACE software and firmware package is also included.	IPC-16P/1xx NOTE xx specifies memory capacity to maximum of 32K words in 4K increments. EXAMPLE IPC-16P/104 (4K of memory)

Table A-2. IPC-16P Major Units

Units and Contents	Description	Order Number
BASIC IPC-16P PACE MICROPROCESSOR DEVELOPMENT SYSTEM		IPC-16P/1xx NOTE IPC-16P is available with one or more 4K RAM memory modules, where *xx* in the order number indicates the approximate number of thousands of memory words included with system. Thus, an 8K system is designated IPC-16P/108.
Hardware (Chassis, wired for, and housing units as follows):		
• 1 Power Supply Assembly	Includes +7.5-, +5-, −12-, and −9-volt supplies to fulfill all secondary supply requirements for basic IPC-16P configuration. +5- and −12-volt supplies employ foldback current limiting and also are equipped with overvoltage protection.	
• 1 Card Cage, wired for and containing following:	Houses up to 12 8.5-inch by 11-inch printed circuit cards.	
• PACE Development CPU Card with Interfacing Cable	Contains PACE and electronics required to interface PACE with rest of IPC-16P. Also contains standard PACE Application CPU Interface at top card edge for connection through Interfacing Cable to user-designed equipment.	
• 1 Memory Timing and Control Card	Generates signals to control accessing of Memory Storage Cards. Contains timing electronics for Refresh Cycle periodically required by Dynamic Memory on Memory Storage Card. One Memory Timing and Control Card can service a maximum of eight Memory Storage Cards.	
• 1 Memory Storage Card	Provides 4096 16-bit words of Dynamic RAM storage. Thus, memory expansion is permitted in 4096 increments up to a maximum of 65,536 16-bit words.	
• 1 Control Panel Interface Card	Provides interfacing between Programmers and Operators Control Panels and PACE Development CPU Card. Included interfacing permits PACE Program Counter, Accumulators AC0 through AC3, Status and Control Flags Register, all 10 levels of Stack and any memory location to be displayed and/or modified at Programmers Control Panel.	
• 1 TTY/Card Reader/Tape Reader Interface Card	Provides interfacing between PACE Development CPU Card and TTY or Card Reader. ROMs containing PACE TTY and Card Reader programs are included on card.	
• 1 Control Panel containing:		
• 1 Programmers Control Panel	Contains data switches, function switches, and data and address indicators for examining or modifying contents of PACE Program Counter, Accumulators AC0 through AC3, Status and Control Flags Register, and 10 levels of Stack or Memory Storage Card locations.	
• 1 Operators Control Panel	Contains keylock POWER and Programmers Control Panel access switches as well as conventional switches to control run, halt, and initialization of IPC-16P. Also contains switch for loading data from Paper Tape Reader.	
NOTE Basic IPC-16P is supplied with Card Cage wired to accommodate two 4096-word Memory Storage Cards and PROM Programmer Cards.		

Table A-2. IPC-16P Major Units (Continued)

Units and Contents	Description	Order Number
Reference Manual: • PACE Users Manual Software: NOTE Software is supplied either as object paper tapes or object card decks to accommodate user requirements. Tapes are assumed; please specify when ordering.		
• PACE Debug Routine (PACDBG)	Relocatable PACE object program that supervises operation of user program during checkout. NOTE PACASM and PACASP must be run together.	
• PACE Resident Assembler (PACASM and PACASP)	Assembles object program from source program on IPC-16P.	
• PACE Conversational Assembler (CASM 16 and CEDT 16)	Combination editor and resident assembler that simplfies edit and assembly procedure by eliminating need for multiple loadings of editor, assembler, and user's programs. Requires 8K words of memory.	
• PACE Editor (PACEDT)	Permits editing of previously prepared source program (or any text) or generating and editing new text.	
• PACE Relocating Loader (PACE General Loader) (PACGLDR)	Loads correctly formatted LMs into memory for execution by PACE.	
• PACE CPU Diagnostic (PACEDI)	Exhaustively tests PACE microprocessor functions and instruction set.	
• PACE Memory Diagnostic (PACMDI)	Tests all bit positions of each address location of Memory Storage Cards to verify proper operation of memory.	
• PACE PROM Punch Program (PACPRMP)	Punches paper tape from which PROMs may be progıammed.	
Firmware:		
• PACE Control Panel Service Routine (PACEPNL)	Program that services Control Panel switches and indicators. Before servicing Control Panel, program saves current contents of PACE Program Counter, AC0 through AC3, Status and Control Flags Register, and all Stack levels.	
• PACE Teletype Routine (PACTTY)	Program to effect data-input/output operations between optional Teletype or High-speed Paper Tape Reader and IPC-16P memory.	
• PACE Card Reader Routine (PACCRD)	Program to effect data-input operations between optional Card Reader and IPC-16P memory.	

Table A-3. IPC-16P Options

Unit	Description	Order Number
Hardware:		
• Card Reader (Documation Model M300L)	Transfers data from card deck into IPC-16 memory at 300-card-per-minute maximum rate. Includes card reader, cable, and software.	IPC-16P/825
• Teletype (ASR Model 33)	Provides data input to IPC-16P memory by way of Paper Tape Reader or keyboard. Provides readout from IPC-16P on printer. Includes reader relay.	IMP-00/810
• High-speed Paper Tape Reader	Provides 120 CPS data input to IPC-16P memory from paper tape.	IMP-00/850
• High-speed Printer (Centronics Model 306)	Provides high-speed printout of IPC-16P memory data or PACE registers, accumulators, Stack and pointers.	IMP-00/812
• High-speed Printer Interface	Controls data-output operation of optional high-speed printer (Centronics).	IPC-16P/812W
• Memory Storage Card	Provides 4096 16-bit words of Dynamic RAM storage.	IPC-16P/004
• Memory Timing and Control Card	Generates signals to control data accessing of Memory Storage Cards. Contains timing electronics for Refresh Cycle periodically required by Dynamic Memory on Memory Storage Cards. One Memory Timing and Control Card can service a maximum of eight Memory Storage Cards.	IPC-16P/004T
• PROM Programmer Card	Permits programming of PROMs with user-generated programs. Includes software.	IPC-16P/805
Software:		
• PACE IMP-16 Cross Assembler — 4K version (PACECA and PACEIO)	Accepts free-format source statements from Teletype or Card Reader and produces LM on paper tape and object listing on Teletype. Program runs with 4096 words of memory, minimum, and Teletype.	IPC-16S/100C (paper tape) IPC-16S/100Q (cards)
• PACE IMP-16 Cross Assembler — 8K version (PACE8CA and PACE8IO)	Same as 4K version of PACE IMP-16 Cross Assembler except runs with 8192 words of memory, minimum.	IPC-16S/101C (paper tape) IPC-16S/101Q (cards)
• PACE FORTRAN Cross Assembler (PACEFORT)	Assembles object program from source program on host computer for subsequent execution by PACE microprocessor. Program may be assembled on IBM 360, Univac 1108, or GE Timeshare. Requires peripherals of processor input unit, scratch unit, list output unit, and binary output unit.	IPC-16S/102Q (cards)
• IMP/16 PACE Translator (TRAN$$)	Translates an existing IMP-16 assembly-language file into PACE assembly language.	Available from timeshare utility companies.
• IMP Punch RLM Program (IMPPRLM)	ANSI FORTRAN program that transcribes PACE FORTRAN Cross Assembler output LMs to punched cards in format compatible with PACCRD and PACGLDR.	Available from timeshare utility companies.

APPENDIX B

PACE INSTRUCTION RELATED SUMMARIES AND PROGRAM EXAMPLES

B.1 INTRODUCTION

Table B-1 defines the notation and symbols used for the symbolic representation of each instruction contained in table B-2, the instruction summary. The notations in table B-1 are presented in alphabetical order and, then, the symbols are listed. Upper-case mnemonics refer to fields in the instruction word. Lower-case mnemonics refer to the numerical value of the corresponding fields. In cases where both upper-case and lower-case mnemonics are composed of the same letters, only the lower-case mnemonic is given. The use of lower-case notation designates variables.

The formulas in table B-2 (the instruction summary) for computing the execution times of instructions are presented in terms of machine (microinstruction) cycles (M) and input/output data-transfer Cycle Extends (E_R for read and E_W for write). Each machine cycle (M) consists of four clock cycles. The following example shows the method to be employed for computing the execution times of instructions.

EXAMPLE

The formula (listed in table B-2) for the execution time of a RADD Instruction is $4M+E_R$. If the clock cycle (or period) is 500 nanoseconds and the Read Cycle Extend is 500 nanoseconds, then: $M=4(0.5\ \mu sec)=2\ \mu sec$

$E_R=0.5\ \mu sec$

therefore: $4M+E_R=4(2\ \mu sec)+0.5\ \mu sec=8.5\ \mu sec$. Thus, under the hypothetical clock cycle and Read Cycle Extend times used, the RADD Instruction requires 8.5 microseconds for execution.

Following the instruction summary, the branch conditions for the 16 condition codes used by the Branch-On Condition Instruction are described in table B-3.

Table B-1. Notations/Symbols Used in Instruction Descriptions

Notation/ Symbol	Meaning
ACr	Denotes specific working accumulator (AC0, AC1, AC2, or AC3), where *r* is number of accumulator referenced in instruction.
cc	Denotes 4-bit condition code value for conditional branch instructions.
CRY	Indicates Carry Flag is set if carry exists due to instruction (either addition or subtraction) or reset if no carry exists.
disp	Stands for displacement value and represents operand in non-memory-reference instruction or address field in memory-reference instruction. Disp is 8-bit, signed twos-complement number except when base page is referenced; in latter case, disp is unsigned if BPS=0.
dr	Denotes number of destination working accumulator specified in instruction-word field. Working accumulator is AC0, AC1, AC2, or AC3.
EA	Denotes effective address specified by instructions directly, indirectly, or by indexing. Effective address contents are used during execution of instruction.

Table B-1. Notations/Symbols Used in Instruction Descriptions (Continued)

Notation/ Symbol	Meaning
fc	Denotes number of referenced flag.
FR	Denotes Status and Control Flags Register.
IEN	Denotes Interrupt Enable Flag.
ℓ	Denotes inclusion of 1-bit Link Flag (LINK) in shift operations.
n	Unsigned number indicates number of bit positions to be shifted in Shift and Rotate Instructions.
OVF	Indicates Overflow Flag is set if overflow exists due to instruction (either addition or subtraction) or is reset if no overflow exists. Overflow occurs if signs of operands are alike and sign of result is different from operands.
PC	Denotes Program Counter. During address formation, PC is incremented by 1 to contain address 1 greater than that of instruction being executed.
r	Denotes number of working accumulator specified in instruction-word field. Working accumulator is AC0, AC1, AC2, or AC3.
STK	Denotes top word of 10-word Last-In/First-Out Stack.
sr	Denotes number of source working accumulator specified in instruction-word field. Working accumulator is AC0, AC1, AC2, or AC3.
xr	When not zero, xr value designates number of accumulator to be used in indexed and relative-memory addressing modes. When zero, base-page addressing is indicated.
( )	Denotes contents of item within parentheses. (ACr) is read as *contents of ACr.* (EA) is read as *contents of EA.*
[ ]	Denotes *result of.*
~	Indicates logical complement (ones complement) of value on right-hand side of ~.
$\rightarrow$	Means *replaces.*
$\leftarrow$	Means *is replaced by.*

Table B-1. Notations/Symbols Used in Instruction Descriptions (Continued)

Notation/ Symbol	Meaning
@	Appearing in operand field of instruction, denotes indirect addressing.
$+_{10}$	Modulo 10 addition.
$\wedge$	Denotes AND operation.
$\vee$	Denotes OR operation.
$\underline{\vee}$	Denotes EXCLUSIVE OR operation.

Table B-2. PACE Instruction Summary

Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
BRANCH INSTRUCTIONS			
Branch-On Condition BOC 15 12 | 11 08 | 07 00 0 1 0 0 | cc | disp	(PC)←(PC) + disp if cc true 16 possible condition codes (cc) exist. Condition codes are listed in table B-3. If condition for branching designated by cc is true, value of disp (sign extended from bit 7 through bit 15) is added to PC and sum is stored in PC.	BOC cc, disp	5M + E_R + 1M if branch
Jump JMP 15 10 | 09 08 | 07 00 0 0 0 1 1 0 | xr | disp	(PC)←EA Effective address EA replaces PC contents. Next instruction is fetched from location designated by new contents of PC.	JMP disp (xr)	4M + E_R
Jump Indirect JMP@ 15 10 | 09 08 | 07 00 1 0 0 1 1 0 | xr | disp	(PC)←(EA) Contents of effective address replace PC contents. Next instruction is fetched from location designated by new contents of PC.	JMP @disp (xr)	4M + 2E_R
Jump to Subroutine JSR 15 10 | 09 08 | 07 00 0 0 0 1 0 1 | xr | disp	(STK)←(PC), (PC)←EA Contents of PC are stored on top of Stack. Effective address replaces PC contents. Next instruction is fetched from location designated by new contents of PC.	JSR disp (xr)	5M + E_R
Jump to Subroutine Indirect JSR@ 15 10 | 09 08 | 07 00 1 0 0 1 0 1 | xr | disp	(STK)←(PC), (PC)←(EA) Contents of PC are stored on top of Stack. Contents of effective address replace PC contents. Next instruction is fetched from location designated by new contents of PC.	JSR @disp (xr)	5M + 2E_R
Return from Subroutine RTS 15 08 | 07 00 1 0 0 0 0 0 0 0 | disp	(PC)←(STK) + disp Contents of PC are replaced by sum of disp added to contents pulled from top of Stack. Program control is transferred to location specified by new contents of PC.	RTS disp	5M + E_R

Table B-2. PACE Instruction Summary (Continued)

Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
BRANCH INSTRUCTIONS (Continued)			
Return from Interrupt RTI 15 … 08 | 07 … 00 0 1 1 1 1 1 0 0 | disp	(PC)←(STK) + disp, IEN = 1 Interrupt Enable Flag (IEN) is set. PC contents are replaced by sum of disp and word pulled from top of Stack. Program control is transferred to location specified by new contents of PC.	RTI disp	$6M + E_R$
SKIP INSTRUCTIONS			
Skip if Not Equal SKNE 15 … 12 | 11 10 | 09 08 | 07 … 00 1 1 1 1 | r | xr | disp	If (ACr) ≠ (EA), (PC)←(PC) + 1 ACr contents and contents of effective memory location EA are compared. If contents of ACr and EA are not equal, next instruction in sequence is skipped. Contents of ACr and EA are unaltered. If 8-bit data length is selected, only lower 8 bits are compared.	SKNE r, disp (xr)	$5M + 2E_R$ + 1M if skip
Skip if Greater SKG 15 … 10 | 09 08 | 07 … 00 1 0 0 1 1 1 | xr | disp	If (AC0) > (EA), (PC)←(PC) + 1 AC0 contents and contents of effective memory location EA are compared as signed numbers. If contents of AC0 are greater (more positive) than contents of EA, next instruction in sequence is skipped. Contents of AC0 and EA are unaltered. If 8-bit data length is selected, only lower 8 bits are compared.	SKG 0, disp (xr)	$7M + 2E_R$ + 1M if skip
Skip if AND is Zero SKAZ 15 … 10 | 09 08 | 07 … 00 1 0 1 1 1 0 | xr | disp	If ((AC0) $\wedge$ (EA)) = 0, (PC)←(PC) + 1 AC0 contents and contents of effective memory location EA are ANDed. If result equals zero, next instruction in sequence is skipped. Contents of AC0 and EA are unaltered. If 8-bit data length is selected, only lower 8 bits are tested.	SKAZ 0, disp (xr)	$5M + 2E_R$ + 1M if skip

Table B-2. PACE Instruction Summary (Continued)

Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
SKIP INSTRUCTIONS (Continued)			
Increment and Skip if Zero ISZ 15 10 | 09 08 | 07 00 1 0 0 0 1 1 | xr | disp	(EA)←(EA) + 1, if (EA) = 0, (PC)←(PC) + 1 EA contents are incremented by 1. If new contents of EA equal zero, next instruction in sequence is skipped. If 8-bit data length is selected, only lower 8 bits are tested.	ISZ disp (xr)	$7M + 2E_R + E_W$ + 1M if skip
Decrement and Skip if Zero DSZ 15 10 | 09 08 | 07 00 1 0 1 0 1 1 | xr | disp	(EA)←(EA) - 1, if (EA) = 0, (PC)←(PC) + 1 EA contents are decremented by 1. If new contents of EA equal zero, next instruction in sequence is skipped. If 8-bit data length is selected, only lower 8 bits are tested.	DSZ disp (xr)	$7M + 2E_R + E_W$ + 1M if skip
Add Immediate, Skip if Zero AISZ 15 10 | 09 08 | 07 00 0 1 1 1 1 0 | r | disp	(ACr)←(ACr) + disp, if (ACr) = 0, (PC)←(PC) + 1 ACr contents are replaced by sum of contents of ACr and disp (sign bit 7 extended through bit 15). Initial contents of ACr are lost. If new contents of ACr equal zero, contents of PC are incremented by 1, thus skipping next instruction. AISZ Instruction always tests full 16-bit result independent of data length selected.	AISZ r, disp	$5M + E_R$ + 1M if skip
MEMORY DATA-TRANSFER INSTRUCTIONS			
Load LD 15 12 | 11 10 | 09 08 | 07 00 1 1 0 0 | r | xr | disp	(ACr)←(EA) ACr contents are replaced by EA contents. Initial contents of ACr are lost; contents of EA are unaltered.	LD r, disp (xr)	$4M + 2E_R$

Table B-2. PACE Instruction Summary (Continued)

Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
MEMORY DATA-TRANSFER INSTRUCTIONS (Continued)			
Load Indirect LD@	(AC0) ← ((EA))	LD 0, @disp (xr)	$5M + 3E_R$
15–10: 1 0 1 0 0 0 | 09 08: xr | 07–00: disp	AC0 contents are replaced indirectly by EA contents. Initial contents of AC0 are lost; contents of EA and location designating EA are unaltered.		
Store ST	(EA) ← (ACr)	ST r, disp (xr)	$4M + E_R + E_W$
15–12: 1 1 0 1, 11 10: r | 09 08: xr | 07–00: disp	EA contents are replaced by contents of ACr. Initial contents of EA are lost; contents of ACr are unaltered.		
Store Indirect ST@	((EA)) ← (AC0)	ST 0, @disp (xr)	$4M + 2E_R + E_W$
15–10: 1 0 1 1 0 0 | 09 08: xr | 07–00: disp	EA contents are replaced indirectly by AC0 contents. Initial contents of EA are lost; contents of AC0 and location designating EA are unaltered.		
Load with Sign Extended LSEX	(AC0) ← (EA) bit 7 extended	LSEX 0, disp (xr)	$4M + 2E_R$
15–10: 1 0 1 1 1 1 | 09 08: xr | 07–00: disp	AC0 contents are replaced by EA contents with bit 7 extended through bits 8-15. Initial contents of AC0 are lost; contents of EA are unaltered. LSEX permits 8-bit data loading from memory or peripheral to be operated on as 16-bit data.		
MEMORY DATA-OPERATE INSTRUCTIONS			
AND AND	(AC0) ← (AC0) $\wedge$ (EA)	AND 0, disp (xr)	$4M + 2E_R$
15–10: 1 0 1 0 1 0 | 09 08: xr | 07–00: disp	AC0 contents and EA contents are ANDed. Result is stored in AC0. Initial contents of AC0 are lost; contents of EA are unaltered.		

Table B-2. PACE Instruction Summary (Continued)

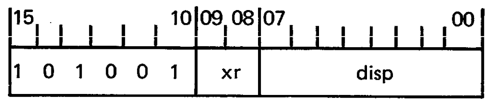
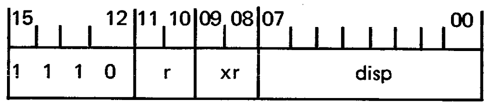
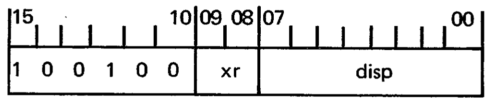
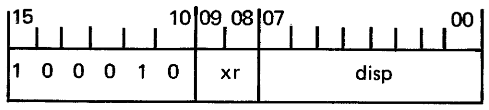
Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
MEMORY DATA-OPERATE INSTRUCTIONS (Continued)			
OR OR 	$(AC0) \leftarrow (AC0) \vee (EA)$ AC0 contents and EA contents are ORed inclusively. Result in stored in AC0. Initial contents of AC0 are lost; contents of EA are unaltered.	OR 0, disp (xr)	$4M + 2E_R$
Add ADD 	$(ACr) \leftarrow (ACr) + (EA), OV, CY$ ACr contents are added algebraically to EA contents. Sum is stored in ACr, and contents of EA are unaltered. Initial contents of ACr are lost. Overflow or Carry Flag is set if overflow or carry occurs, respectively; otherwise Overflow and Carry Flags are cleared.	ADD r, disp (xr)	$4M + 2E_R$
Subtract with Borrow SUBB 	$(AC0) \leftarrow (AC0) + \sim(EA) + (CY), OV, CY$ AC0 contents are added to complement of EA and carry. Result is stored in AC0 and contents of EA are unaltered. Initial contents of AC0 are lost. Carry and Overflow Flags are set according to result of operation.	SUBB 0, disp (xr)	$4M + 2E_R$
Decimal Add DECA 	$(AC0) \leftarrow (AC0) +_{10} (EA) +_{10} (CY), OV, CY$ AC0 contents are treated as 4-digit number and added modulo 10 (for each digit) to contents of EA (treated as 4-digit number) and carry. Initial contents of AC0 are lost; contents of EA are unaltered. Carry Flag is set based on decimal carry output. Overflow Flag is set to arbitrary state.	DECA 0, disp (xr)	$7M + 2E_R$

Table B-2. PACE Instruction Summary (Continued)

Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
REGISTER DATA-TRANSFER INSTRUCTIONS			
Load Immediate LI	(ACr)←disp	LI r, disp	4M + E_R
Bits 15–10: 0 1 0 1 0 0; 09–08: r; 07–00: disp	ACr contents are replaced by disp with sign bit 7 extended through bit 15. Initial contents of ACr are lost.		
Register Copy RCPY	(ACdr)←(ACsr)	RCPY sr, dr	4M + E_R
Bits 15–10: 0 1 0 1 1 1; 09–08: dr; 07–06: sr; 05–00: not used	Destination Register ACdr contents are replaced by contents of Source Register ACsr. Initial contents of ACdr are lost and initial contents of ACsr are unaltered.		
Register Exchange RXCH	(ACdr)←(ACsr), (ACsr)←(ACdr)	RXCH sr, dr	6M + E_R
Bits 15–10: 0 1 1 0 1 1; 09–08: dr; 07–06: sr; 05–00: not used	ACsr contents and ACdr contents are exchanged.		
Exchange Register and Stack XCHRS	(STK)←(ACr), (ACr)←(STK)	XCHRS r	6M + E_R
Bits 15–10: 0 0 0 1 1 1; 09–08: r; 07–00: not used	Contents of top of Stack and accumulator designated by ACr are exchanged.		
Copy Flags into Register CFR	(ACr)←(FR)	CFR r	4M + E_R
Bits 15–10: 0 0 0 0 0 1; 09–08: r; 07–00: not used	ACr contents are replaced by contents of FR. Initial contents of ACr are lost; contents of FR are unaltered.		

Table B-2. PACE Instruction Summary (Continued)

Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
REGISTER DATA-TRANSFER INSTRUCTIONS (Continued)			
Copy Register into Flags CRF 15 … 10: 0 0 0 0 1 0 | 09 08: r | 07 … 00: not used	(FR)←(ACr) FR contents are replaced by ACr contents. Initial contents of FR are lost; contents of ACr are unaltered.	CRF r	4M + E_R
Push Register onto Stack PUSH 15 … 10: 0 1 1 0 0 0 | 09 08: r | 07 … 00: not used	(STK)←(ACr) Stack is pushed by contents of accumulator designated by ACr. Thus, top of Stack holds ACr contents and Stack Pointer is incremented by 1. Initial contents of ACr are unaltered.	PUSH r	4M + E_R
Pull Stack into Register PULL 15 … 10: 0 1 1 0 0 1 | 09 08: r | 07 … 00: not used	(ACr)←(STK) Stack is pulled. Contents from top of Stack replace ACr contents. Initial contents of ACr are lost. Contents of Stack Pointer are decremented by 1.	PULL r	4M + E_R
Push Flags onto Stack PUSHF 15 … 10: 0 0 0 0 1 1 | 09 … 00: not used	(STK)←(FR) FR contents are pushed onto Stack. Contents of FR are unchanged.	PUSHF	4M + E_R
Pull Stack into Flags PULLF 15 … 10: 0 0 0 1 0 0 | 09 … 00: not used	(FR)←(STK) FR contents are replaced by contents pulled from top of Stack. Initial contents of FR are lost.	PULLF	4M + E_R

Table B-2. PACE Instruction Summary (Continued)

Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
REGISTER DATA-OPERATE INSTRUCTIONS			
Register Add RADD 15 … 10 | 09 08 | 07 06 | 05 … 00 0 1 1 0 1 0 | dr | sr | not used	(ACdr)←(ACdr) + (ACsr), OV, CY ACdr contents are replaced by sum of contents of ACdr and ACsr. Initial contents of ACdr are lost and contents of ACsr are unaltered. Overflow and Carry Flags are modified according to result.	RADD sr, dr	$4M + E_R$
Register Add with Carry RADC 15 … 10 | 09 08 | 07 06 | 05 … 00 0 1 1 1 0 1 | dr | sr | not used	(ACdr)←(ACdr) + (ACsr) + (CY), OV, CY ACdr contents are replaced by sum of ACdr and ACsr contents and carry. Initial ACdr contents are lost and ACsr contents are unaltered. Overflow and Carry Flags are modified according to result.	RADC sr, dr	$4M + E_R$
Register AND RAND 15 … 10 | 09 08 | 07 06 | 05 … 00 0 1 0 1 0 1 | dr | sr | not used	(ACdr)←(ACdr) $\wedge$ (ACsr) ACdr contents are replaced by result of ANDing ACdr and ACsr contents. Initial contents of ACdr are lost and initial contents of ACsr are unaltered.	RAND sr, dr	$4M + E_R$
Register EXCLUSIVE OR RXOR 15 … 10 | 09 08 | 07 06 | 05 … 00 0 1 0 1 1 0 | dr | sr | not used	(ACdr)←(ACdr) $\veebar$ (ACsr) ACdr contents are replaced by result of EXCLUSIVEly ORing ACdr contents and ACsr contents. Initial contents of ACdr are lost and initial contents of ACsr are unaltered.	RXOR sr, dr	$4M + E_R$
Complement and Add Immediate CAI 15 … 10 | 09 08 | 07 … 00 0 1 1 1 0 0 | r | not used	(ACr)←~(ACr) + disp ACr contents are replaced by sum of complement of ACr and disp (sign bit 7 extended through bit 15). Initial contents of ACr are lost. Values of 0 and 1 in disp field produce ones and twos, complement, respectively, of (ACr).	CAI r, disp	$5M + E_R$

Table B-2. PACE Instruction Summary (Continued)

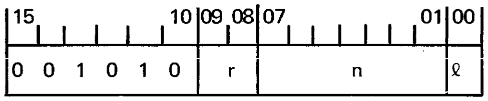
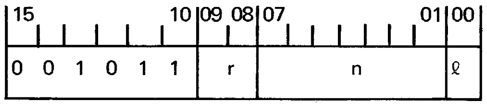
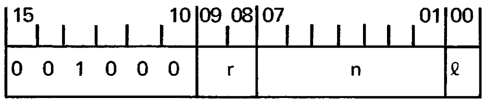
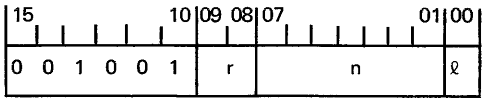
Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
SHIFT AND ROTATE INSTRUCTIONS			
Shift Left SHL 	(ACr)←(ACr) shifted left n places, w/wo link ACr contents are shifted left n (n = 0-127) bit positions. If selected data length is 8 bits, then bits 8-15 are set to zero. Data shifted out of most significant bit for specified data length are lost if $\ell = 0$ and are loaded into LINK if $\ell = 1$.	SHL r, n, ℓ	$(5 + 3n)\ M + E_R$, n = 1-127; $6M + E_R$, n = 0
Shift Right SHR 	(ACr)←(ACr) shifted right n places, w/wo link ACr contents are shifted right n (n = 0-127) bit positions. If selected data length is 8 bits, then bits 8-15 are set to zero. Zeros are shifted into most significant bit for specified data length if $\ell = 0$. Contents of LINK are shifted in if $\ell = 1$, and contents of LINK are unchanged. Data shifted out of least significant bit are lost.	SHR r, n, ℓ	$(5 + 3n)\ M + E_R$, n = 1-127; $6M + E_R$, n = 0
Rotate Left ROL 	(ACr)←(ACr) rotated left n places, w/wo link ACr contents are rotated left n (n = 0-127) bit positions. If selected data length is 8 bits, then bits 8-15 are set to zero. Data shifted out of most significant bit position for specified data length are shifted into least significant bit if $\ell = 0$, and into LINK if $\ell = 1$, in which case least significant bit is loaded from LINK.	ROL r, n, ℓ	$(5 + 3n)\ M + E_R$, n = 1-127; $6M + E_R$, n = 0
Rotate Right ROR 	(ACr)←(ACr) rotated right n places, w/wo link ACr contents are rotated right n (n = 0-127) bit positions. If selected data length is 8 bits, then bits 8-15 are set to zero. Data shifted out of least significant bit are shifted into most significant bit for specified data length if $\ell = 0$, and into LINK if $\ell = 1$, in which case most significant bit is loaded from LINK.	ROR r, n, ℓ	$(5 + 3n)\ M + E_R$, n = 1-127; $6M + E_R$, n = 0

Table B-2. PACE Instruction Summary (Continued)

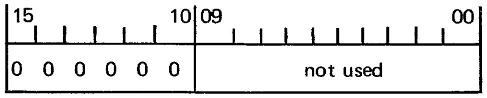
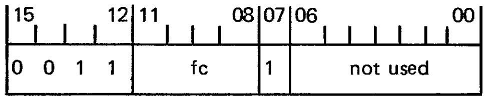
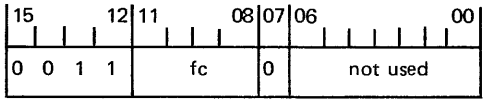
Instruction/Mnemonic	Operation/Description	Assembler Format	Execution Time/Cycles (M)
MISCELLANEOUS INSTRUCTIONS			
Halt HALT 	Halt Microprocessor halts and remains halted until CONTIN Input to Jump Condition Multiplexer makes transition from logic '1' to logic '0'.	HALT	- - - - - -
Set Flag SFLG 	$(FR)_{fc} \leftarrow 1$ Flag, or bit of FR, specified by flag code fc is set true. All other bits of FR are unaltered.	SFLG fc	$5M + E_R$
Pulse Flag PFLG 	$(FR)_{fc} \leftarrow 1$, $(FR)_{fc} \leftarrow 0$ Flag (bit fc of FR) is first set true and then set false (after four clock periods), causing pulsing or resetting of flag, depending on initial state of flag. All other bits of FR are unaffected.	PFLG fc	$6M + E_R$

Table B-3. Branch Conditions

Condition Code (cc)	Mnemonic	Condition
0000	STFL	Stack Full (contains nine or more words).
0001	REQ0	(AC0) equal to zero (see note 1).
0010	PSIGN	(AC0) has positive sign (see note 2).
0011	BIT0	Bit 0 of AC0 true.
0100	BIT1	Bit 1 of AC0 true.
0101	NREQ0	(AC0) is nonzero (see note 1).
0110	BIT2	Bit 2 of AC0 is true.
0111	CONTIN	CONTIN (continue) Input is true.
1000	LINK	LINK is true.
1001	IEN	IEN is true.
1010	CARRY	CARRY is true.
1011	NSIGN	(AC0) has negative sign (see note 2).
1100	OVF	OVF is true.
1101	JC13	JC13 Input is true (see note 3).
1110	JC14	JC14 Input is true.
1111	JC15	JC15 Input is true.

NOTES:
1. If selected data length is 8 bits, only bits 0 through 7 of AC0 are tested.
2. Bit 7 is sign bit (instead of bit 15) if selected data length is 8 bits.
3. JC13 is used by PACE Microprocessor Development System and is not accessible during prototyping.

B.2 PROGRAMMING EXAMPLES

The following paragraphs provide typical programming examples.

B.2.1 DECIMAL ADDITION

The decimal addition program (see table B-4) adds two 16-digit BCD strings that are packed 4 digits per word. The two strings to be added are stored in memory starting at locations STR1 and STR2. The resulting digit string is stored in memory starting at location STR2.

B.2.2 TENS COMPLEMENT

Representation of negative decimal numbers in tens-complement form may be desirable for many PACE applications, since the Decimal-Add Instruction then can be used directly for signed number additions. The tens-complement program converts an unsigned BCD number to a tens-complement negative number representation.

The sign of a tens-complement number can be tested by using the BOC Instruction with the PSIGN Jump Condition to test the most significant word of the decimal number.

NOTE

Negative numbers have leading nines while positive numbers have leading zeros.

The tens-complement program presented in table B-5 converts a 16-digit number packed in 4 words of memory beginning at location NUM.

B.2.3 DECIMAL SUBTRACTION

The decimal subtraction program listed in table B-6 performs a decimal subtract by forming the tens complement and using the Decimal-Add Instruction. The 16-digit string, starting at location STR2, is subtracted from the string starting at location STR1.

B.2.4 BINARY MULTIPLICATION

Two binary-multiplication program examples are provided in table B-7. The first program example multiplies the 16-bit value in AC2 by the 16-bit value in AC0 and provides a 32-bit result in AC1 (high order) and AC0 (low order).

NOTE

Positive numbers of 16-bit magnitude are assumed (that is, most significant bit is zero).

The second program multiplies the 16-bit value in AC2 by the 16-bit value in AC0 and provides a 32-bit result in AC0 (high order) and AC1 (low order).

NOTE

16-bit magnitude only is assumed.

Table B-4. Decimal Addition Program Example

```
                        .TITLE   DECADD, ' DECIMAL ADDITION'
                        ;
      ØØØØ              .ASECT
      Ø1ØØ              .=X'1ØØ
                        ;
      ØØØØ     RØ       =        Ø
      ØØØ1     R1       =        1
      ØØØ2     R2       =        2
      ØØØ3     R3       =        3
      ØØØ7     CRY      =        7
                        ;
Ø1ØØ  Ø2ØØ  A  STR1:    .WORD    X'2ØØ
Ø1Ø1  Ø25Ø  A  STR2:    .WORD    X'25Ø
Ø1Ø2  Ø1ØØ  A  ADDR1:   .WORD    STR1           ;ADDRESS OF ADDEND STRING
Ø1Ø3  Ø1Ø1  A  ADDR2:   .WORD    STR2           ;ADDRESS OF AUGEND/RESULT ST
                        ;
                        ;
Ø1Ø4  51Ø4  A  START:   LI       R1,4           ;NUMBER DIGITS/4 TO AC1 (LOO
Ø1Ø5  C9FC  A           LD       R2,ADDR1       ;LOAD INDEX REGISTERS WITH
Ø1Ø6  CDFC  A           LD       R3,ADDR2       ;  ARGUMENT ADDRESSES
Ø1Ø7  37ØØ  A           PFLG     CRY            ;CLEAR CARRY FLAG
Ø1Ø8  C2ØØ  A  LOOP:    LD       RØ,(R2)        ;ADDEND TO ACØ
Ø1Ø9  8BØØ  A           DECA     RØ,(R3)        ;DECIMAL ADD WITH AUGEND
Ø1ØA  D3ØØ  A           ST       RØ,(R3)        ;STORE RESULT
Ø1ØB  7AØ1  A           AISZ     R2,1           ;INCREMENT INDEX
Ø1ØC  7BØ1  A           AISZ     R3,1           ;  REGISTERS
Ø1ØD  79FF  A           AISZ     R1,-1          ;DECREMENT LOOP COUNT
Ø1ØE  19F9  A           JMP      LOOP           ;ADD NEXT WORD
                        ;
      Ø1Ø4              .END     START
```

Table B-5. Tens Complement Program Example

```
                           .TITLE  TENCOM, ' TENS-COMPLEMENT '
                           ;
          0000             .ASECT
          0100             .=X'100
                           ;
          0000    R0       =       0
          0001    R1       =       1
          0002    R2       =       2
          0007    CRY      =       7
                           ;
    0100  00C8 A  NUM:     .WORD   200
    0101  0100 A  ADDR:    .WORD   NUM
    0102  999A A  CONST:   .WORD   X'999A
                           ;
    0103  5104 A  START:   LI      R1,4              ;LOOP COUNT TO AC1
    0104  C9FC A           LD      R2,ADDR           ;ADDRESS TO AC2 INDEX REGIST
    0105  3780 A           SFLG    CRY               ;SET CARRY FLAG FOR FIRST LO
    0106  C1FB A  LOOP:    LD      R0,CONST          ;CONSTANT TO AC0
    0107  9200 A           SUBB    R0,(R2)           ;COMPLEMENT AND ADD DECIMAL
                                                     ;  NUMBER PLUS CARRY
    0108  D200 A           ST      R0,(R2)           ;STORE RESULT
    0109  3700 A           PFLG    CRY               ;CLEAR CARRY FOR SUBSEQUENT
    010A  7A01 A           AISZ    R2,1              ;INCREMENT POINTER
    010B  79FF A           AISZ    R1,-1             ;DECREMENT LOOP COUNT
    010C  19F9 A           JMP     LOOP              ;REPEAT LOOP
                           ;
          0103             .END    START
```

Table B-6. Decimal Subtraction Program Example

```
                            .TITLE  DECSUB, ' DECIMAL SUBTRACTION '
                            ;
                            ;          CALLING SEQUENCE
                            ;
                            ;             JSR     DECS
                            ;
         0000    R0         =       0
         000B    SIGN       =       11
         000A    CRY        =       10
         0007    CARRY      =       7
                            ;
    0000 9999 A  H9999:     .WORD   X'9999
    0001 0200 A  OP1:       .WORD   X'200              ;OPERAND 1 IN LOCATION 200
    0002 0201 A  OP2:       .WORD   X'201              ;OPERAND 2 IN LOCATION 201
                            ;
    0003 3700 A  DECS:      PFLG    CARRY              ;CLEAR CARRY FLAG
    0004 3B00 A             PFLG    SIGN               ;CLEAR SIGN (USER) FLAG
    0005 C1FA T             LD      R0,H9999           ;TAKE 9'S COMPLEMENT OF OP1
    0006 91FA T             SUBB    R0,OP1
    0007 89FA T             DECA    R0,OP2
    0008 4A04 A             BOC     CRY,CTRUE          ;BRANCH ON CARRY TRUE
                            ;
                            ; CARRY = 0 INDICATES NEGATIVE RESULT
                            ;
    0009 3B80 A             SFLG    SIGN               ;SET SIGN FLAG
    000A 7001 A             CAI     R0,1
    000B E1F4 T             ADD     R0,H9999           ;TAKE 9'S COMPLEMENT
    000C 8000 A             RTS     0                  ;RETURN
                            ;
                            ; CARRY = 1 INDICATES POSITIVE RESULT
                            ;
    000D 7801 A  CTRUE:     AISZ    R0,1               ;ADD END AROUND CARRY
    000E 8000 A             RTS     0                  ;RETURN IF RESULT NEQ 0
    000F 8000 A             RTS     0                  ;RETURN IF RESULT EQ 0
                            ;
         0003               .END    DECS
```

Table B–7. Binary Multiplication Examples

```
                            .TITLE   BIMULT, ' BINARY MULTIPLICATION '
                            ;
          0000              .ASECT
          0100              .=X'100
                            ;
          0000    R0        =        0
          0001    R1        =        1
          0002    R2        =        2
          0003    R3        =        3
          000A    CARRY     =        10
                            ;
     0100 FFFF A  CONST:    .WORD    X'FFFF          ;CONSTANT FOR DOUBLE-PRECISI
                                                     ;  ADDITION
     0101 5100 A  START:    LI       R1,0            ;CLEAR RESULT REGISTER
     0102 5310 A            LI       R3,16           ;LOOP COUNT TO AC3
     0103 7000 A            CAI      R0,0
     0104 6940 A  LOOP:     RADD     R1,R1           ;SHIFT RESULT LEFT INTO CARR
     0105 7400 A            RADC     R0,R0           ;SHIFT CARRY INTO MULTIPLIER
                                                     ;  AND MULTIPLIER INTO CARRY
     0106 4A02 A            BOC      CARRY,TEST      ;TEST FOR ADD
     0107 6980 A            RADD     R2,R1           ;ADD MULTIPLICAND TO RESULT
     0108 91F7 A            SUBB     R0,CONST        ;ADD CARRY TO HIGH-ORDER RES
     0109 7BFF A  TEST:     AISZ     R3,-1           ;DECREMENT LOOP COUNT
     010A 19F9 A            JMP      LOOP            ;REPEAT LOOP
                            ;
          0101              .END     START
                            .TITLE   BIMULT,  ' BINARY MULTIPLY'
                            ;
          0000              .ASECT
          0100              .=X'100
                            ;
          0000    R0        =        0
          0001    R1        =        1
          0002    R2        =        2
          0003    R3        =        3
          0003    BIT0      =        3
          0008    LINK      =        8
                            ;
                            ;
     0100 5100 A  START:    LI       R1,0            ;CLEAR RESULT REGISTER
     0101 5310 A            LI       R3,16           ;LOOP COUNT IN AC3
     0102 7000 A            CAI      R0,0            ;COMPLEMENT MULTIPLIER
     0103 4301 A  LOOP:     BOC      BIT0,SHIFT      ;TEST BIT 0
     0104 6980 A            RADD     R2,R1           ;ADD MULTIPLICAND TO RESULT
     0105 3800 A  SHIFT:    PFLG     LINK            ;CLEAR LINK
     0106 2503 A            ROR      R1,1,1          ;SHIFT AC1 INTO LINK
     0107 2C03 A            SHR      R0,1,1          ;SHIFT LINK INTO AC0
     0108 7BFF A            AISZ     R3,-1           ;DECREMENT LOOP COUNT
     0109 19F9 A            JMP      LOOP            ;REPEAT LOOP
                            ;
          0100              .END     START
```

Table B-8. Descriptions of Status and Control Flags

Register Bit	Flag Name	Description	Flag Code (fc)
0	High ('1')	Bit 0 is not used and is always in logic '1' state. Referencing bit 0 with SFLG or PFLG Instruction has no effect. (May be used as NOP Instruction.)	0000
1 2 3 4 5	IE1 IE2 IE3 IE4 IE5	Flags IE1 through IE5 serve as Interrupt Enable Flags for five of six PACE Interrupt levels. If Interrupt Enable is high and associated Interrupt Request occurs, microprocessor executes Interrupt Service Routine. If Interrupt Enable is low, associated Interrupt Request is ignored.	0001 0010 0011 0100 0101
6	OVF	Overflow Flag is set to state of twos-complement arithmetic overflow by arithmetic instructions. Overflow Flag is set high if sign bits (most significant bit) of two operands are identical and sign bit of result is different from sign bit of operands. If A, B, and R are sign bits of operands and result, then Overflow Flag is set according to equation $\text{OVF} = (\overline{A}, \overline{B}, R) + (A, B, \overline{R})$ Sign bit is most significant bit for data length selected; thus, if data length is 8 bits, then bit 7 is sign bit; if data length is 16, then bit 15 is sign bit. State of OVF Flag is affected by instructions ADD, DECA, SUBB, RADD, and RADC.	0110
7	CRY	Carry Flag is set to state of binary or decimal carry output of adder by arithmetic instructions. Carry output is derived from most significant bit for data length specified by BYTE Flag. State of CRY Flag is affected by instructions ADD, DECA, SUBB, RADD, and RADC.	0111
8	LINK	Link Flag is included in shift and rotate operations as specified by Shift and Rotate Instructions. Link Flag is unaffected if not selected.	1000
9	IEN	Master Interrupt Enable Flag simultaneously inhibits all five of lowest priority interrupt levels. No Interrupt Request is serviced unless individual Interrupt Enable Flag for associated Interrupt Request and master Interrupt Enable Flag are high. IEN Flag is set low every time any interrupt (except Level 0) is serviced. IEN Flag is set high by execution of Return-To-Interrupt Instruction (RTI).	1001
10	BYTE	BYTE Flag selects 8-bit data length when high and 16-bit data length when low.	1010
11 12 13 14	F11 F12 F13 F14	Flags 11 through 14 are general-purpose control flags. Flags 11 through 14 drive PACE output pins and may be used to directly control system functions.	1011 1100 1101 1110
15	High ('1')	Bit 15 is not functional and is always in logic '1' state. Addressing bit 15 with SFLG or PFLG Instruction sets the Level-0 Interrupt Enable high.	1111