MICRO-PRODAC PULSETACH INTERFACE (PPI) BOARD

S#8571A40

TABLE OF CONTENTS

EFFECTIVE DATE: DECEMBER, 1982

FIGURES

TABLES

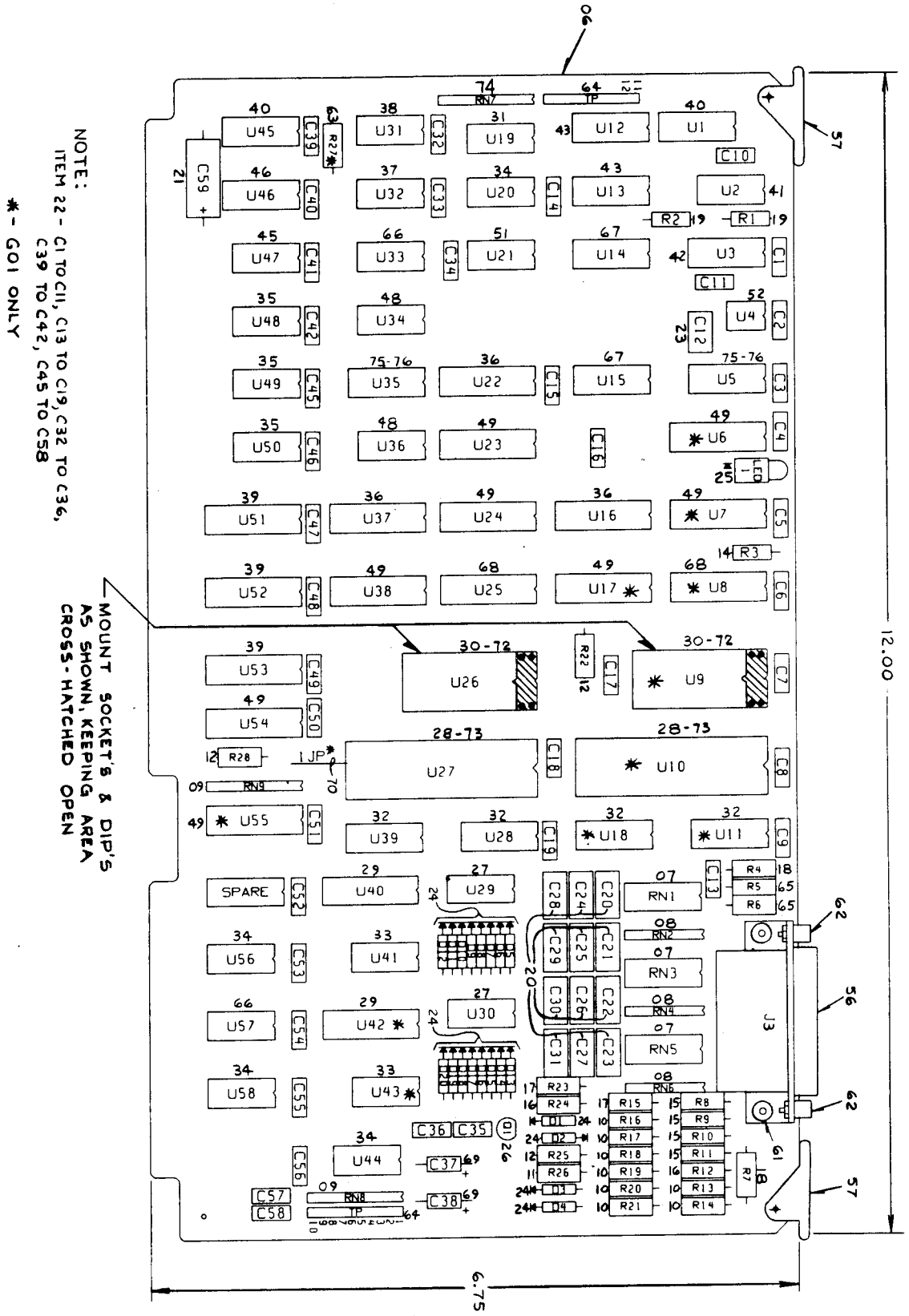
REFERENCES

1. PPI Bill of Materials.....8571A40
2. PPI Schematic.....4954C82
3. PPI Assembly View.....4941C08
4. Micro-Prodac Standards and Conventions.....8449A73
5. Micro-Prodac Product Description.....8450A03
6. Pulse Conditioner B/M.....8571A43
7. MCS-51 Family of Single Chip Microcomputers User's Manual, July 1981 (or later)...Intel 121517-001
8. Hardware Description of the PPI.....8571A86

1.0 INTRODUCTION

The Micro-Prodac Pulsetach Interface (PPI) Board is a part of the Micro-Prodac motor drive control and regulator system. The PPI board is the interface between that Multibus based system and any pulsetrain input. The PPI is primarily intended for tachometer inputs from sine wave reluctance, single ended square wave, or differential square wave tachs. The PPI is used with the Pulse Conditioner which contains a terminal block for tach wiring, and any required power supply and impedance matching networks. A standard cable connects the PPI to the conditioner.

The PPI has two identical input channels which work independently to measure speeds (frequencies) or accumulate pulses. Each channel is capable of generating a system interrupt. The channels allow diagnostic testing by the system processor and the on-board processors. An "OK" LED can be controlled by the system processor to indicate the result of such testing.


NOTE:
ITEM 22 - C1 TO C11, C13 TO C19, C32 TO C36, C39 TO C42, C45 TO C58
* - G01 ONLY
MOUNT SOCKET'S & DIP'S AS SHOWN, KEEPING AREA CROSS-HATCHED OPEN
12.00
6.75
J3
U1
U2
U3
U4
U5
U6
U7
U8
U9
U10
U11
U12
U13
U14
U15
U16
U17
U18
U19
U20
U21
U22
U23
U24
U25
U26
U27
U28
U29
U30
U31
U32
U33
U34
U35
U36
U37
U38
U39
U40
U41
U42
U43
U44
U45
U46
U47
U48
U49
U50
U51
U52
U53
U54
U55
U56
U57
U58
SPARE
RN1
RN2
RN3
RN4
RN5
RN6
RN7
RN8
RN9
LED
JP

FRONT VIEW

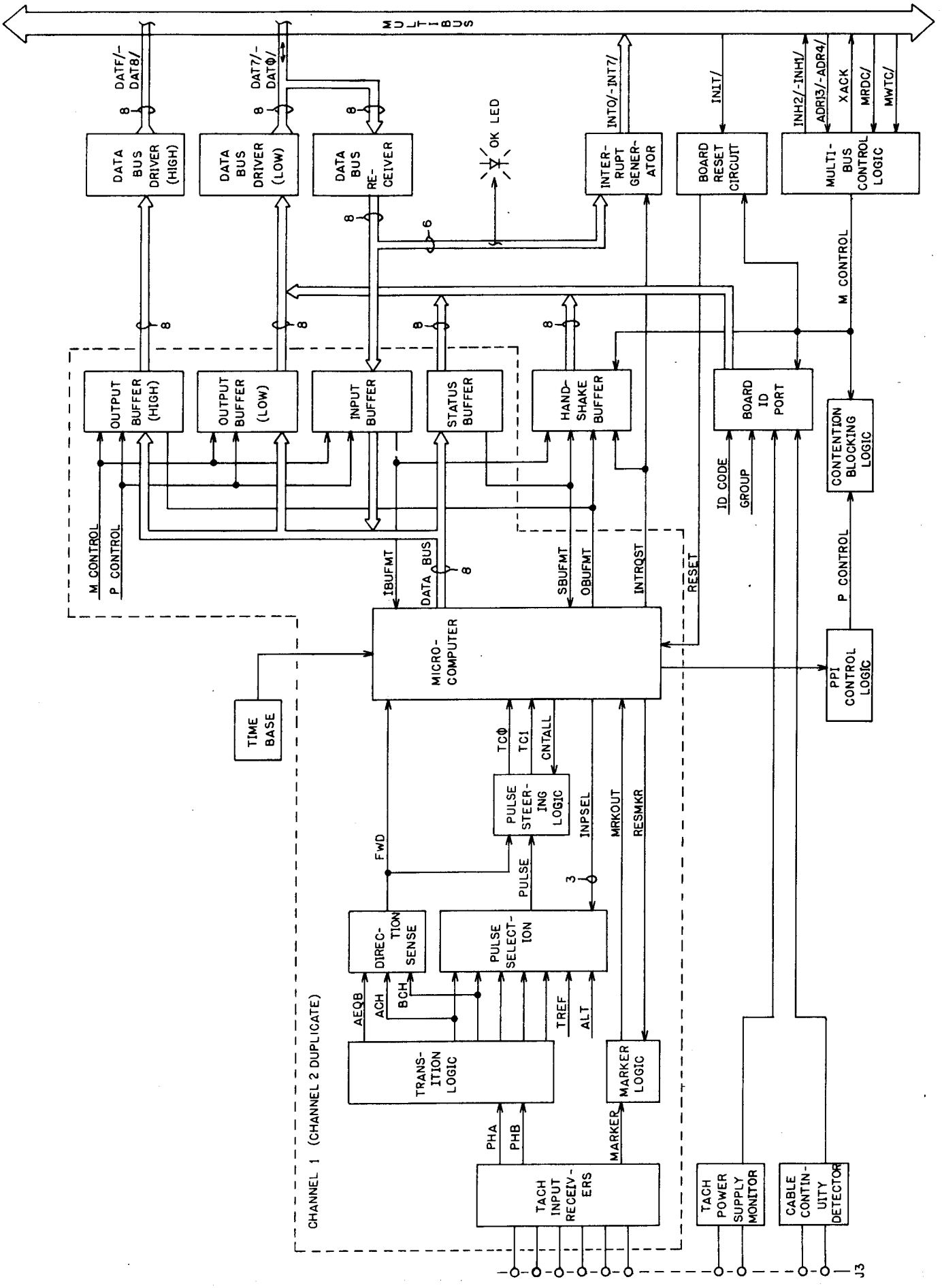


FIGURE 1.1 PPI BLOCK DIAGRAM

2.0 FUNCTIONAL DESCRIPTION

Since the PPI board contains microprocessors its operation depends on both hardware and firmware designs. This section describes primarily the hardware, with firmware described in the Application section. For a more thorough discussion of the hardware design see reference 8. Refer to the PPI block diagram (fig. 1), memory map (fig. 2), and schematic where needed. Schematics are referenced by page letter (ex: page A).

Definitions

All addresses are given in hexadecimal notation. "Hexadecimal" refers to a base-16 number system, and henceforth is abbreviated HEX, hex, or -H suffix (ex: 8000H).

Certain variables have data types deserving special mention. A "don't care" variable is defined such that the action of writing any data to it's address is sufficient to generate the intended control action regardless of the value of data. "Logical" data is byte data where FFH is true, 00H is false, and all other values are not allowed. "Integer" data is two bytes wide in two's complement format.

2.01 Bus Interface

Address Selection

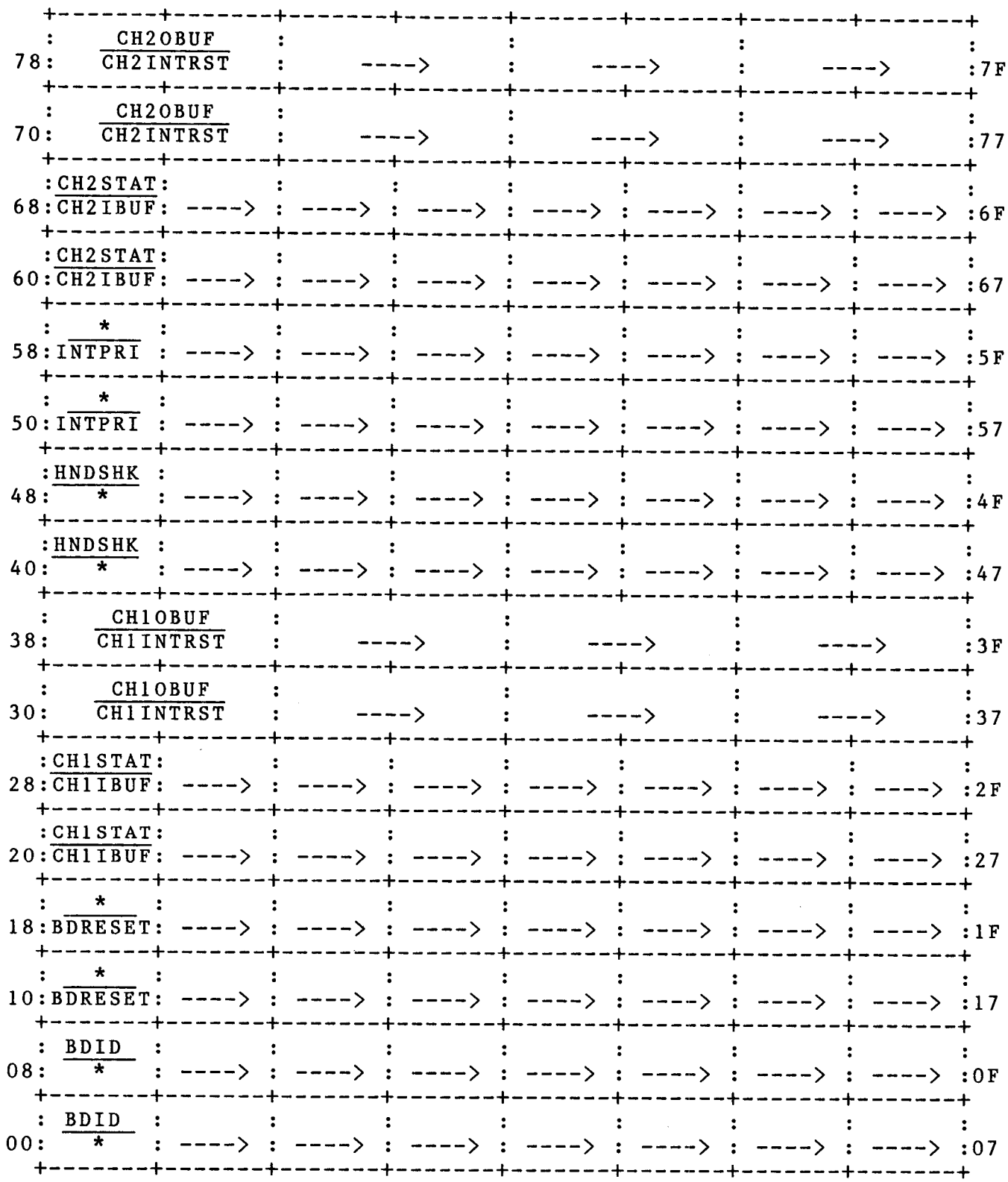
A full description of bus interfacing is contained in reference 4. The PPI board is memory mapped in a 20 bit address space. The board will respond to addresses in a 256 byte range (256 bytes = one "page") between 08000 and 08FFF hexadecimal. The exact page of the sixteen possible pages in that space is selected by four board select wires on the P2 connector mounted in the card cage. Schematic sheet A shows signals BDS8, BDS4, BDS2, and BDS1 on P2 pins 31, 33, 35, and 37 respectively. These should be wired so that the board select lines have a binary code equal to the cage slot number (zero to fifteen). The PPI has on-board pullup resistors so that jumpers must be added only to program logic zeroes. Henceforth, PPI addresses will be referred to as 08N00H to 08NFFH where N is the slot number (between 0 and F hex).

Inhibit Lines

The PPI board drives bus lines INH1/ and INH2/ which means that any system ROM or RAM on other boards will be inhibited when PPI boards are addressed. This allows memory boards with large address blocks to be shadowed by I/O boards like the PPI between 08000 and 08FFF hex.

Wait States

The PPI drives the transfer acknowledge handshake line XACK/ with wait states generated for all accesses. Wait state generator circuitry is shown on schematic sheet A. The typical wait time is 166 nanoseconds but on occaision that time may stretch to 8 microseconds due to the blocking logic. Whenever the 8031 microprocessors update the output or status buffers they block Multibus access until their transfer is complete. If the bus master accesses the board during that time it will be held with wait states. On rare occaisions the block may last longer than 8 microseconds if the 8031 is interrupted during the process. The block time is then dependent on the interrupt service routine time which may be different for each mode. This effect is only a problem if the bus master processor experiences a timeout while waiting for PPI access.



NOTES:

1. * Denotes open address; transferred data meaningless.
2. Read Var / Write Var Denotes different read and write data.
3. ----> Denotes memory image; do not use.
4. Addresses 80 to FF are open; do not use.

FIGURE 2.1.1 PPI MEMORY MAP

Figure 2.1.1 illustrates the PPI memory map. The map shows all board addresses, many of which are not actually used or are duplicates (images) of other addresses. Table 2.1.1 lists the preferred address and type of data for each variable. Some of the data is read-only or write-only. This is noted on the map.

VARIABLE	ADDRESS	RD/WRT	TYPE	DESCRIBED
BDID	8N00	R	BYTE	2.17
BDRESET	8N10	W	BYTE DON'T CARE	2.02
CH1_IBUF	8N20	W	BYTE	2.12
CH1_STAT	8N20	R	BYTE	2.14
CH1_OBUF	8N30	R	INTEGER	2.13
CH1_INTRST	8N30	W	BYTE DON'T CARE	2.16
HNDSHK	8N40	R	BYTE	2.15
INTPRI	8N50	W	BYTE	2.16
CH2_IBUF	8N60	W	BYTE	2.12
CH2_STAT	8N60	R	BYTE	2.14
CH2_OBUF	8N70	R	INTEGER	2.13
CH2_INTRST	8N70	W	BYTE DON'T CARE	2.16

TABLE 2.1.1 VARIABLE DEFINITIONS

2.02 Board Reset

The PPI board is held reset during power-on settling times by the bus signal INIT/ (see schematic sheet B) which generates the board reset signal RESET. The RESET signal can also be generated by the bus master at any time by writing to the board reset address (BDRESET - 8N10H). Resetting the PPI initializes the OK light, interrupt control register, interrupt request latches, and microcomputers. When the software reset is used, a pulse stretching circuit is required to meet specs for the micros. A digital counter is clocked at 0.75 megahertz to generate the 10 microsecond pulse.

After reset the PPI executes self diagnostics. These include checksum of EPROM, micro self test, and frequency check using TESTREF as input. The result of these diagnostics is reported to the bus master through CHxSTAT.

2.03 Tach Input Receivers

The tachometer input receivers are shown on sheet H of the schematic. These are six similar circuits used for differential buffering of phase A, phase B, and marker pulse inputs. All circuits are usable for square wave inputs from five to fifteen volts in amplitude. Only the phase B circuits should be used for sine wave tachs because these have symmetrical hysterisis which will not introduce duty cycle variation at low input amplitudes.

Each receiver contains an input filter with a five microsecond time constant. This limits the practical square wave input frequency to 50 kilohertz. The inputs are attenuated by a factor of ten which increases the common mode range to approximately 50 volts peak. Diode clamps are provided to protect the inputs from transient overvoltages up to +/- 300 volts.

2.04 Transition Detection Logic

Transition detection logic for channels one and two is shown on sheet E of the schematic. Input data from tach phases A and B are shifted through shift registers clocked at 0.75 megahertz. Whenever the outputs of stage one and two are different as determined by

"exclusive or" gates, a transition is occurring. The gate output CxACH or CxBCH will pulse high for all changes in level of phase A or B respectively. This is the times-two counting mode, since the pulse frequency is twice the incoming square wave frequency. For times-one mode (CxPHAX1 and CxPHBX1), the pulses are "anded" with the actual square wave so that the counting logic responds only to rising edges.

The times-four mode (CxPHABX4) is obtained by "exclusive or-ing" the times-two outputs of phase A and phase B. This results in a pulse being counted for every transition of both waveforms. Obviously the times-four mode is only useful with a tach having two quadrature (shifted 90 degrees) outputs. If counting is attempted in the times-four mode without quadrature outputs the actual count rate will be times-two.

At the time when a transition is occurring stage one is the new level and stage two is the old. This fact is used in the direction sensing logic (see section 2.06) and uses another "exclusive or" gate to determine if the new phase A level is the same as the new phase B level.

2.05 Pulse Selection

The pulse input selection logic is shown on sheet F of the schematic. An eight input multiplexer IC selects the desired input under control of the 8031 microcomputer. The eight input choices are: on-board test oscillator, off, phase A times-one, phase A times-two, phase B times-one, phase B times-two, times-four, and the alternate channel's input. This selection is made at the application level by downloading the initializing parameter INPSEL.

2.06 Direction Detection Logic

Tach direction sensing logic is shown on sheet F of the schematic. This logic is only usable with tachs having two quadrature (shifted 90 degrees) outputs. The logic computes direction based on all edges of the phase A and phase B waveforms and will not be fooled by vibrations at a single edge.

The algorithm for direction assumes that phase B leads phase A in the forward direction. If phase B is changing and its new level is different than phase A, or if phase A is changing and its new level is the same as phase B, then the direction is forward. If phase B is changing and its new level is the same as phase A, or if phase A is changing and its new level is different than phase B, then the direction is reverse. If both phase B and phase A are changing at the same time, that is an error resulting from noise and the direction logic ignores any such change.

The circuitry used to implement the direction logic is a sequential logic state machine. A two-to-four line decoder chip does the combinational logic and the present state (direction) is stored in a "D" flip-flop.

2.07 Pulse Steering Logic

Pulse steering logic divides the selected pulse train into the forward direction pulses CxIN and reverse direction pulses CxT1. These pulse trains feed into counters zero and one respectively allowing individual accumulation. If the microprocessor activates the line CxCNTALL/ then CxIN will contain all the pulses allowing agregate accumulation.

2.08 Marker Logic

Marker detection circuitry (schematic sheet E) is only used in pulse accumulation modes. The 8031 disables the circuit by holding a

marker reset line active (CxRESMKR/). Otherwise, the circuit looks for the presence of a marker pulse at every transition of the Phase B tach signal. If found, the marker is stored in a latch until the 8031 completes the proper marker activity (accumulator reset, preset, etc.). Afterwards the 8031 resets the latch.

2.09 Microcomputer

Microcomputer circuitry includes the 8031 uC, address latch, and EPROM as shown on sheets F and G of the schematic. See reference 7 (section 1.1) for a description of these standard circuits. The 8031 contains a central processing unit (CPU), 128 bytes of RAM, two 16 bit timer-counters, interrupt controller, a (USART) serial link, and several input-output pins.

Both timer-counters can count time using one microsecond pulses from the internal clock. In addition each has an input pin allowing it to count external pulses. Counter zero is used to count input pulses from the forward output of the steering logic. Counter one can count input pulses from the reverse direction output. Either or both counters may overflow into software counters by creating interrupts.

The interrupt controller has several internal and external sources that can be assigned high or low priority. External source zero is assigned to the pulse inputs and is given high priority so that pulse counting can be done in software counters.

The USART is used to pass data between the two 8031 channels on board. The serial link is full duplex, meaning each 8031 can send data to the other simultaneously.

The available I/O pins are used to control local functions such as input selection, bus interrupt generation, direction and marker logic, and data transfer with the bus. Initializing parameters downloaded from the bus are translated and sent to the 8031 output pins for actual control. This allows the 8031 to override those parameters during test modes.

2.10 Time Base

The time base is a 12.000 megahertz crystal oscillator as shown on schematic sheet F. The oscillator output MPCLK drives the microcomputers and the bus interface wait state generator. It also feeds a divider chain with a 0.75 MHz output which drives the reset pulse stretcher, transition logic shift registers, and direction logic latches.

MPCLK is divided down inside the micros to one megahertz which is the clock frequency for the internal timers. The accuracy of all clocks is dependent on the crystal which is very accurate and stable over the normal temperature range (see Appendix for specifications).

2.11 Test Reference

The test reference oscillator is shown on sheet B of the schemes. It is a 555 type circuit which runs at approximately 10.2 kilohertz. The test reference can be selected by the pulse selection logic to provide a confidence test in any mode of PPI operation. The frequency tolerance is +/- 20 percent.

2.12 Input Data Buffers

The input buffers for channels one and two are shown on sheets C and D of the schematic. Both buffers work in the same manner. Input data from the bus is written into an eight bit register. The same write pulse that latches data into the register also clears a flip-flop. The output of the FF is a flag CHxIBUFMT indicating whether the input buffer is empty or full. The 8031 micro always polls this flag

before accepting data from the input buffer. If the buffer is full the 8031 will read the data. The same read pulse that empties data out of the register also sets the flag.

The bus master has access to the CHxIBUFMT flag through the handshake register. The master must check that the buffer is empty before writing into it, or the original contents will be lost. The variables CH1IBUF and CH2IBUF are write-only byte input ports at addresses 08N20H and 08N60H respectively.

2.13 Output Data Buffers

The output data buffers for channels one and two are shown on sheets C and D of the schematic. Both buffers work in the same manner. Output data from the 8031 is written into two eight bit registers arranged so that one is the high byte and the other the low byte of a 16 bit register. The same write pulse that latches data into the register also clears a flip-flop. The output of the FF is a flag CHxOBUFMT indicating whether the output buffer is empty or full. The bus master may read the output buffer at any time. The same read pulse that empties data out of the register also sets the FF. If the bus master attempts to read the output buffer while the 8031 is writing new data into it then the bus master will be held up by wait states until the 8031 is done. The hold up is transparent to the bus master and lasts typically 10 microseconds. Afterwards the bus master completes its transfer with the new data.

The bus master can access the CHxOBUFMT flag through the handshake register to determine if the output buffer contains old or new data. If CHxOBUFMT is one then the buffer contains old data that has already been read. If CHxOBUFMT is zero then the buffer is full with new data.

Variables CH1OBUF and CH2OBUF are read-only 16 bit output buffers at addresses 08N30H and 08N70H respectively.

2.14 Output Status Buffers

The output status buffers for channels one and two are shown on sheets K and L of the schematic. Both buffers work in the same manner. Output status from the 8031 is written into an eight bit register. The same write pulse that latches status into the register also clears a flip-flop. The output of the FF is a flag CHxSBUFMT indicating whether the status buffer is empty or full. The bus master may read the status buffer at any time. The same read pulse that empties status out of the register also sets the FF. If the bus master attempts to read the status buffer while the 8031 is writing new status into it then the bus master will be held up by wait states until the 8031 is done. The hold up is transparent to the bus master and lasts typically 10 microseconds. Afterwards the bus master completes its transfer with the new status.

The bus master can access the CHxSBUFMT flag through the handshake register to determine if the status buffer contains old or new status. If CHxSBUFMT is one then the buffer contains old status that has already been read. If CHxSBUFMT is zero then the buffer is full with new status.

Variables CH1STAT and CH2STAT are read-only byte buffers at addresses 08N20H and 08N60H respectively.

2.15 Handshake Register

The handshake register HNDSHK is an eight bit port that can be read by the bus master as an aid in passing data to and from the 8031's. As shown on sheet B of the schematic, four bits apply to each of the two channels. A handshake mechanism is necessary because the

master and slave processors are not syncronized as to when they are transferring data or status information. The source processor writes its information into a buffer (input, output, or status) which automatically clears the corresponding "buffer empty" flag. The destination processor periodically inspects the flag via the handshake register. If it finds the buffer full it may empty the buffer which automatically sets the flag. The source processor is also able to inspect the flag to determine that the destination processor has taken the data.

The handshake register also contains bits that signal the state of the interrupt request latches. Note that the interrupt request bits are active low.

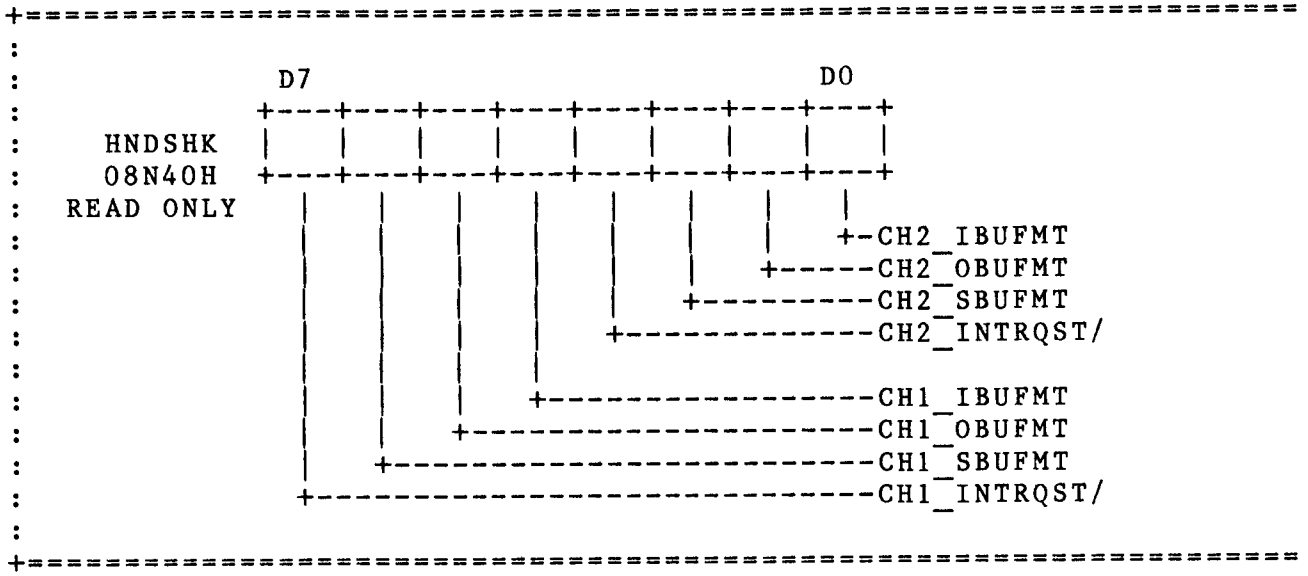


FIGURE 2.15.1 HANDSHAKE REGISTER BYTE (HNDSHK)

2.16 Interrupt Generation

Each channel has the capability to generate a bus interrupt. This feature may be of use in certain modes so that the bus master receives important status and/or data with minimal delay. The use of interrupts is optional. All interrupt circuitry is disabled after the PPI board is reset. When the PPI does generate interrupts, they are level active type, which means that they are latched on-board. Part of the bus master service routine must reset the interrupt request latch(es).

To use the interrupt system requires two levels of initialization. The PPI contains a hardware register which controls the priority of bus interrupts, and which allows selective control of enables for each channel. In addition a software parameter for each channel (INPSEL) specifies whether status outputs, data outputs, or both will create a bus interrupt. The generation of interrupts depends on the selected PPI mode (ie- speed, pulse accumulation, etc.). Refer to the appropriate initialization paragraphs in the Application Information.

Even though each channel can create an interrupt, the priority level must be the same for both. The bus master sets the level by writing to the interrupt priority register INTPRI at address 08N50H. The register also contains enable bits for each channel. Schematic sheet B shows the interrupt register and generator IC's . An eight line demultiplexer steers the channel one or two request to the desired bus interrupt line. The register is cleared to all zeroes after the PPI is reset.

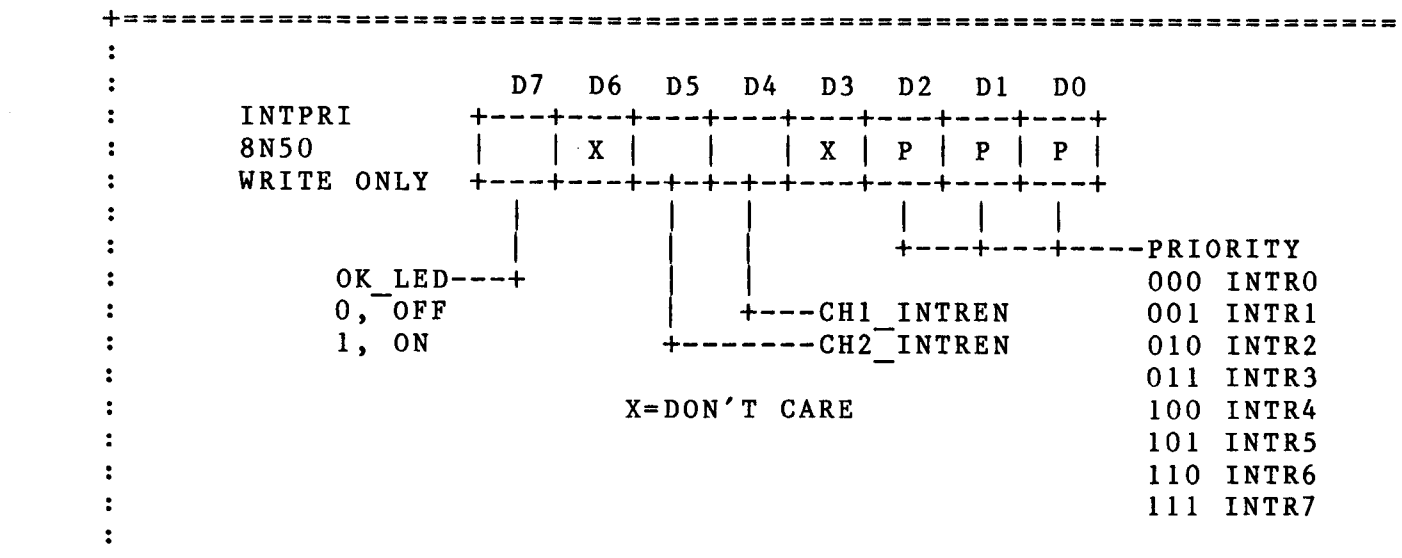


FIGURE 2.16.1 INTERRUPT PRIORITY REGISTER (INTPRI)

Each channel has an interrupt request latch as shown on schematic sheets C and D. The latch is set by a pulse from the 8031. The latch is cleared by three conditions: 1) an inactive enable line CHxINTREN, 2) an interrupt reset pulse from the bus master CHxINTRST, or 3) PPI board reset which clears the enables. The latch reset state overrides the set state. The enable lines CHxINTREN are controlled by the bus master through the priority register as described above. The interrupt reset pulse is a fast way for the bus master to clear the request latches by writing don't care data to CH1INTRST and/or CH2INTRST (addresses 8N30H and 8N70H respectively).

2.17 Board ID Register

A full description of the Board ID concept can be found in reference four. The PPI board ID is read as a byte at address 8N00H. Variable BDID as shown on the memory map has four sections: board type, group, tach power status, and connector continuity status. The board type is eight. Group is either zero or one. Group equal to zero indicates the -G01 version which has both channels of pulse conversion. Group equal to one indicates -G02 which contains only channel one circuitry. The tach power supply monitor and cable continuity status are described elsewhere in this document.

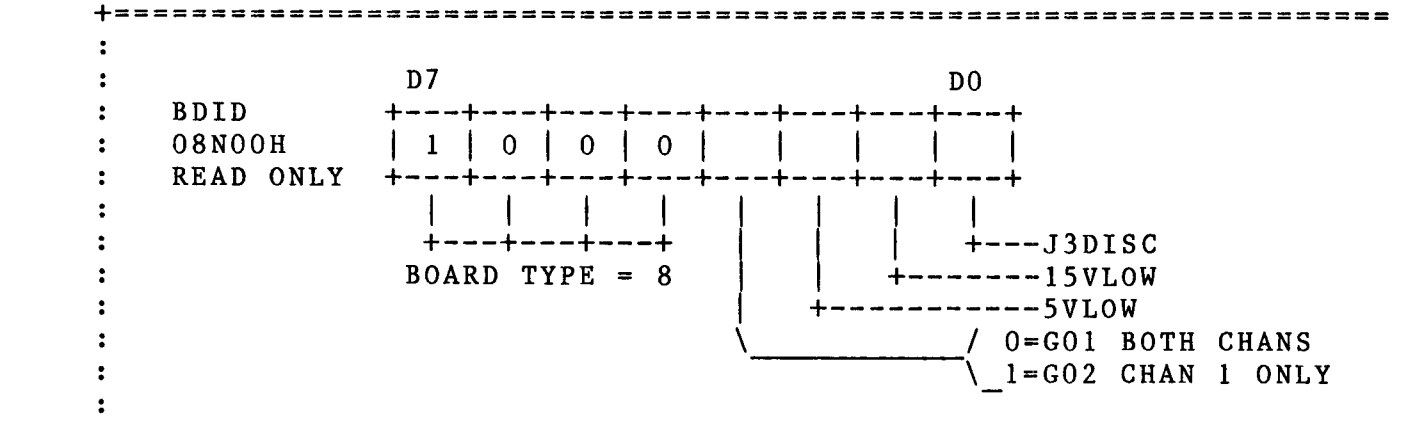


FIGURE 2.17.1 BOARD ID BYTE (BDID)

2.18 Tach Power Supply Monitors

The tach power supply monitor circuitry is shown on sheet J of the schematic. These comparators are similar to the tach differential input receivers. They can operate properly in the presence of up to 50 volts of common mode noise. The power supply voltage is applied to these circuits via the J3 front connector. If the voltage is below 4 volts, then the logic signals 5VLOW and 15VLOW will be true. For voltages between 4 and 13 volts 5VLOW will be false and 15VLOW will be true. Above 13 volts both will be false. Obviously the correct status of the two bits is determined by the power supply in use in a particular system.

5VLOW and 15VLOW can be read as bits D2 and D1 respectively at the BDID address 08N00H.

2.19 Cable Continuity Fault Detector

Schematic sheet B shows two signals (J3 pins 1 and 13) used to determine that the J3 front connector and cable have continuity. Provided they are tied together at the remote end of the cable then the continuity status bit J3DISC will be low. If the connection is broken anywhere along the loop then the status bit will go high to reflect the discontinuity. J3DISC can be read at the BDID address 08N00H Bit D0.

2.20 "OK" Light

The OK light is a green LED under control of the bus master processor. It should be turned on whenever the PPI board is judged to be operating correctly. It should be turned off only when the PPI card itself is diagnosed as faulty. Do not use the OK light to indicate system faults such as connector discontinuity, etc. This would confuse service personnel and violate the Micro-Prodac standards.

The OK LED control is bit 7 of the interrupt priority register (address 08N50H). Logic one turns the light on; zero turns it off. After board reset, the light is off.

3.0 APPLICATION INFORMATION

Getting Started

When initializing the PPI it is best to apply software reset to ensure that the sequence begins at a known point. The only exceptions are when the board is already on line and the user is changing modes or programming the second channel. After power-on reset or software board reset the PPI enters a diagnostic mode which includes checksum of EPROM, micro self test, and frequency check using TESTREF as input. The result of these diagnostics is reported to the bus master through CHxSTAT.

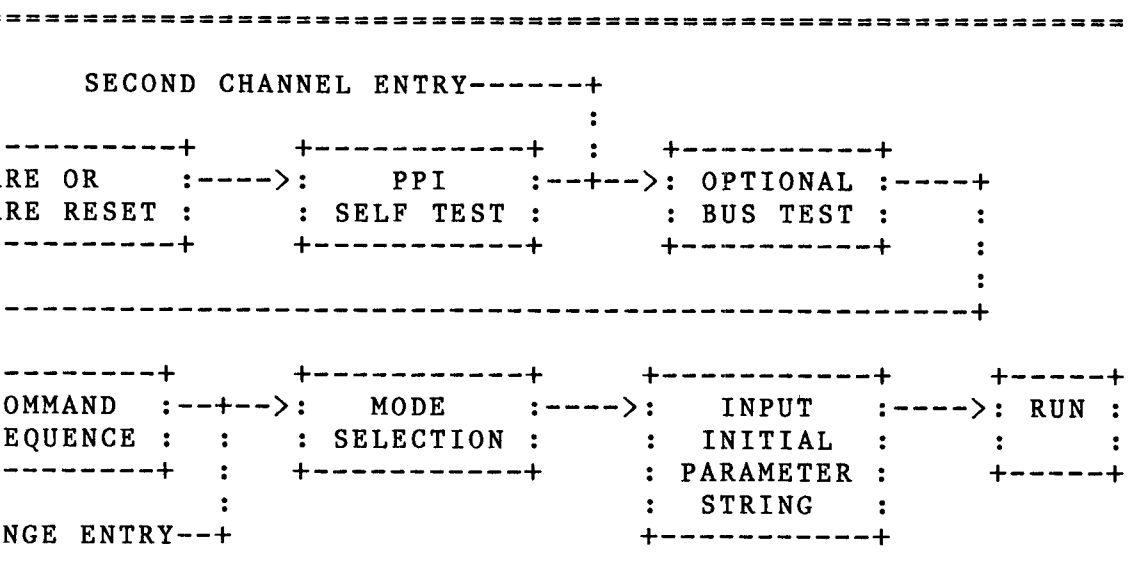


FIGURE 3.0.1 PPI INITIALIZATION FLOW DIAGRAM

Soon after reset the status buffer is filled with 01H indicating that the PPI is testing itself. The bus master should wait for the status handshake before reading status. Upon completion of self test the status buffer is loaded with test results and the TESTING bit is cleared. A one in a particular bit indicates failure of that test. An all zero buffer indicates self test has passed.

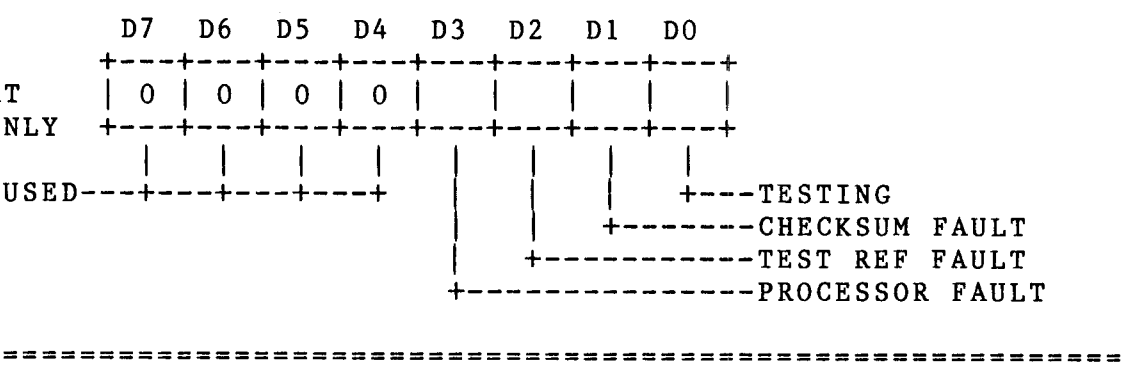


FIGURE 3.0.2 OUTPUT STATUS BUFFER (CHxSTAT) AFTER RESET

Bus Interface Check

Following successful self diagnosis the bus interface can be checked by cooperative effort of the PPI and the bus master. If this feature is not desired the master can proceed with the command

sequence. The master downloads a series of byte constants to CHxIBUF. The PPI will output the one's complement of the constant to the high byte of CHxOBUF and output the constant itself to the low byte of CHxOBUF and to CHxSTAT. The bus master can read these echoed values using the handshake lines and perform a comparison with the original. The series can continue until the master is satisfied the bus interface is operating correctly. At that time the "OK" LED can be turned on. Typical test sequences would be rotating one's or all the numbers from 0 to 255, but not the command sequence.

Command Sequence

After diagnostics the PPI firmware expects a command sequence of 3 bytes to pass through CHxIBUF. This sequence is labeled COMD1, COMD2, COMD3 (55H, AAH, 55H). The first two bytes are echoed back through CHxOBUF and CHxSTAT just as in the prior diagnostic sequence. The third byte COMD3 is acknowledged by 00H in CHxSTAT and 0000H in CHxOBUF.

Mode Select

After the command sequence has been correctly received and acknowledged the next byte in CHxIBUF is taken as a mode select byte. A check is performed to assure a valid mode. If the mode is invalid then FFH is loaded into CHxSTAT. The master must then reset and reinitialize the PPI. If the mode is valid the output and status buffers remain at zero. From this point on the contents of the status buffer represent fault or status information specific to the mode.

The PPI then expects a series of initializing parameters to be downloaded through CHxIBUF. Most parameters differ for each mode. Refer to the particular mode application section for further information.

Most PPI operating modes allow the the bus master to change to a new mode without going through the board reset sequence. When selecting a mode in this manner the contents of CHxOBUF and CHxSTAT are not guaranteed to be zero. User software should take this into account.

MODE	DESCRIPTION
00H	Invalid
01H	Version Identification Mode ***
02H	Speed Mode
03H	Linear Pulse Accumulation Mode
04H	Bidirectional Pulse Accumulation Mode ***
05H	Reserved
06H	Reserved
07H	Reserved

NOTES: 1. Modes 08H to FFH are invalid.
2. *** denotes preliminary; not available.

TABLE 3.0.1 MODE SELECT BYTE VALUES

Input Selection

One initializing parameter is the input selection byte INPSEL. This byte configures the transition detection logic and also establishes the interrupt structure for the channel. The figure below illustrates INPSEL and the variations it allows. The counting source can be selected from a test reference, phase A or B of the tach in the

times 1, 2, or 4 mode, or from the counting source of the alternate channel.

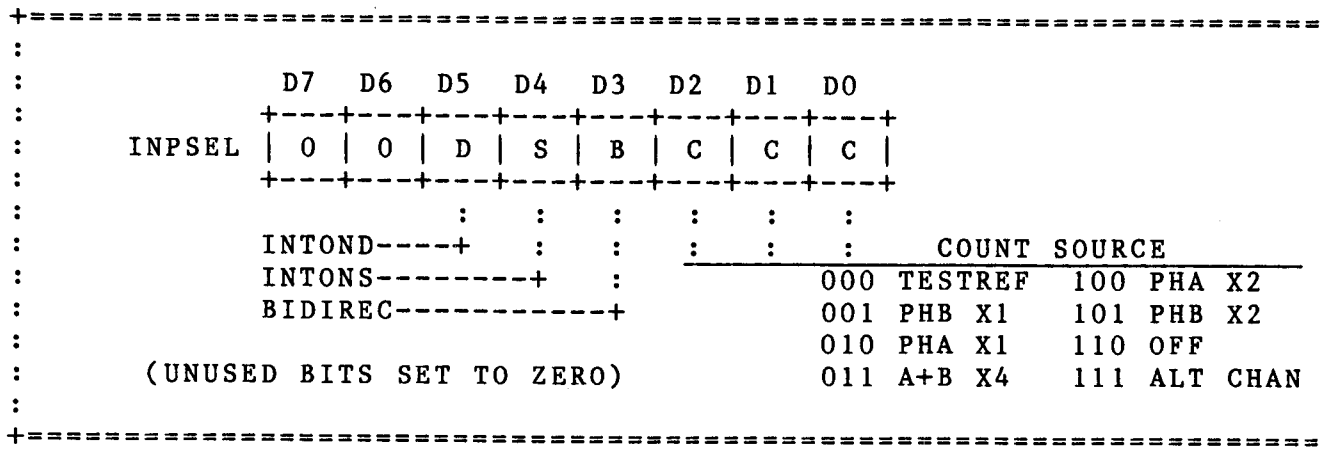


FIGURE 3.0.3 INPUT SELECT BYTE (INPSEL)

If INTOND is set then the PPI will interrupt the bus master whenever new unread data becomes available in CHxOBUF. If INTONS is set then the interrupt will occur whenever status information becomes available (such as fault codes). If both bits are set then interrupts will occur for both new data and new status information.

BIDIREC must be set to one if a bidirectional (quadrature) tach is used. This bit enables the direction detection logic. When the bit is set to zero the direction firmware is locked in the forward direction. If a single phase tach is used in the bidirectional count mode the logic will wrongly deduce that the direction is changing at every pulse edge.

3.1 VERSION IDENTIFICATION MODE **** PRELIMINARY ****

Theory

The version identification mode is used to determine the firmware version present in a particular PPI board. By using this mode a bus master processor can prevent operation where a mismatch exists between application software and PPI firmware.

Initialization

Initialization of the PPI is described in section 3.0 up to the point where an operating mode is selected. The value of the mode select byte is 01H for the Version Identification Mode. There is no parameter string to be passed in this mode such as in the other modes.

Operation

As soon as the mode select byte is received the PPI begins to output the version number to the status buffer. The version is updated continuously while in this mode. The contents of the output buffer are not disturbed. If this mode was entered from another operating mode then the status buffer should be emptied before reading the version number just in case it contained old unread status.

The version number is in hex between 00H and FEH. For example version 2.4 of the firmware would place 24H in the status buffer. All subsequent versions of firmware will have increasing release numbers. Version FFH is not allowed because this code represents the "INVALID MODE" status. Generally an increase in the least significant digit represents a minor firmware change whereas increasing the most significant digit is a major change.

On-Line Commands

The next byte received in CHxIBUF is taken to be a command to change to the mode represented by that value. For example 02H would change to the speed mode. No other commands are recognized. An invalid mode select byte will cause the same action as described in section 3.0.

3.2 SPEED MODE

Theory of Operation

Inherent in a speed (frequency) measurement system is the sensing of events per unit time. This system samples the number of incoming pulses in a known unit of time, and then mathematically divides the former by the latter to obtain a quantity representing speed. The 8031 microprocessor can provide this function simply and accurately, and has secondary benefits such as flexibility and self-test capability.

In a sampled data control system it is advantageous to have the sampling time fixed so that discrete time equations remain valid and so the processor can use its time most efficiently. This is difficult when measuring speed (pulse frequency) because the time between inputs varies widely from low to high speeds. It is desirable to measure many cycles when the frequency is high and few cycles when frequency is low in order to keep the measurement time reasonably constant. It is especially desirable to measure an integral number of pulses during the sample to prevent error due to the uncertainty in the fractional part of the number counted. At low frequencies there may be sample intervals in which no pulses are received. Special methods are needed to handle this condition.

In the PPI the length of the sample period TS is bounded by initializing parameters TS(min) and TS(max). TS is made up of two time intervals which are called the "open" time TO and "close" time TC. Interval TO normally begins at a transition of the incoming pulse train and always ends time TS(min) later. The second interval TC is variable length, beginning at the end of TO, and normally ending at the next transition of the pulse train. If no pulses arrive before TS(max) elapses the interval is ended. Normally the sample time corresponds to an integral number of pulses from the pulse train. The end of TC is also the beginning of the subsequent sample interval TS.

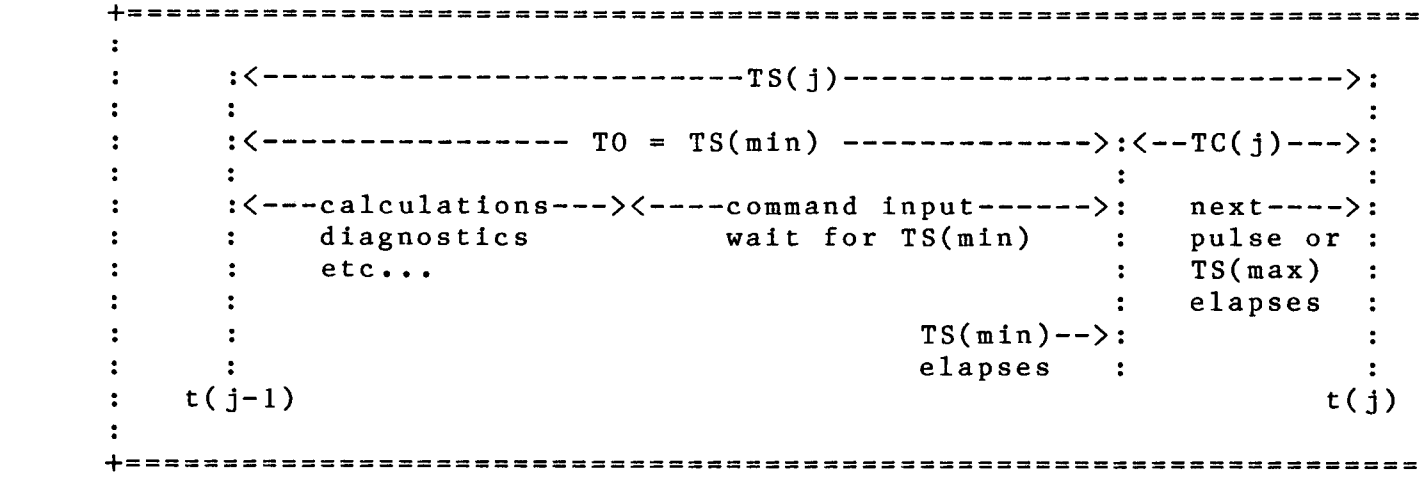


FIGURE 3.2.1 SAMPLE INTERVAL

During the sample interval the Pulse Accumulator counts all incoming pulses at frequency Fp, while the Time Accumulator counts clocks from a fixed time base Fx. Since the accumulators count continuously and are read on-the-fly, the number of pulses counted over the interval TS(j) are

(1) $$Np(j) = Fp * [t(j) - t(j-1)]$$

(2) $$Nt(j) = Fx * [t(j) - t(j-1)]$$

where Np is the number of incoming pulses, Nt is the number of time base clocks, and t(j-1) and t(j) are the times at the beginning and end of the sample interval respectively. Solving (2) for time and substituting into (1) yields

(3) $$Fp = Fx * Np(j) / Nt(j)$$

from which frequency Fp is known as a function of accumulator contents. In the PPI speed is calculated from the expression

(4) $$S(j) = Kc * Np(j) / Nt(j)$$

where S(j) is the average speed over sample j and Kc is a calibration constant.

Implementation

The pulse accumulator is a 16 bit hardware counter. The time accumulator is a 16 bit hardware counter with 8 bit software overflow. Both run continuously and are sampled on-the-fly at the end of each sample interval. Therefore the counts for a given interval are the difference between the present and previous sampled values. Math calculations are done in the "background" during interval TO. After interval TO the next incoming pulse interrupts the 8031 causing it to read the contents of the pulse and time accumulators. The time base clock Fx runs at one megahertz with high precision and stability.

In general the sample interval TS is not constant but will always be greater than TS(min) and less than TS(min)+T where T is the period of the incoming pulse. At low speeds where T is greater than [TS(max)-TS(min)] the sample interval is TS=TS(max). In this case a basic premise is violated that the sample interval corresponds to an integral number of pulses. The accuracy of individual samples will suffer although the average over many samples will still be very accurate.

Accuracy and Resolution

Accuracy within one sample interval depends on the crystal oscillator frequency, input pulse jitter, math round off error, and an uncertainty of one count of Nt due to the asyncronism between Fx and the input pulses. The crystal accuracy directly influences the absolute accuracy of the system as evidenced by equation (3) above. This is not a factor in systems that can be externally calibrated. In that case only the stability of the crystal can affect long term accuracy. Tolerance specifications are given in the appendix.

Jitter in the incoming pulse train causes system inaccuracy on a per-sample basis. This problem is particularly severe at low speeds where a measurement may be based on only one or two pulses. Over several samples the jitter and Nt uncertainty are reduced by averaging because the pulse and time accumulators never miss any counts, but are read on-the-fly. These problems can be reduced by using high precision tachs and long sample times.

Math error is never greater than 1/2 LSB (least significant bit) because the division operation is carried out to one extra bit and the result is rounded off to the required resolution. Subtraction and multiplication operations are carried out with no loss of significance or accuracy.

System resolution is approximately one part out of Nt parts. The worst resolution occurs when Nt is at its minimum value corresponding to short sample times. This value is known and fixed by the length of

interval TO.

The result is a speed measurement system having fast updates with moderate accuracy and resolution, but high average accuracy and resolution over several samples. This implies that the best performance can be obtained by averaging the speed output of the PPI, such as by using an integral type of speed controller. Generally speaking, long sample times yield higher accuracy and resolution over the speed range.

Speed Mode Initialization

Initialization of the PPI is described in section 3.0 up to the point where an operating mode is selected. If the Speed Mode is selected a parameter string must be downloaded through CHxIBUF as shown in the table. Parameters include input select byte, calibration constant, sample times, and speed settings for tach diagnostic calculations. Parameters are stored in internal memory of the 8031 microcomputer.

SEQUENCE		DESCRIPTION
1.	INPSEL	INPUT SELECT BYTE (SEE SECTION 3.0)
2.	SPARE	NOT USED - SET TO ZERO
3.	KCLO	CALIBRATION CONSTANT - LOW BYTE*
4.	KCMED	" " - MID BYTE
5.	KCHI	" " - HIGH BYTE
6.	SPARE	NOT USED - SET TO ZERO
7.	KTMINLO	MIN SAMPLE TIME - LOW BYTE**
8.	KTMINHI	" " " - HIGH BYTE
9.	KTMAXLO	MAX SAMPLE TIME - LOW BYTE**
10.	KTMAXHI	" " " - HIGH BYTE
11.	DELTALO	MAX SPEED CHANGE - LOW BYTE
12.	DELTAHI	" " " - HIGH BYTE
13.	NOISELO	NOISE THRESHOLD - LOW BYTE
14.	NOISEHI	" " - HIGH BYTE
15.	SPARE	NOT USED - SET TO ZERO
16.	SPARE	NOT USED - SET TO ZERO

* CALIBR. CONSTANT = $\dfrac{0.80 * 32{,}768 * 1{,}000{,}000}{\dfrac{\text{RPM (@100\%)}}{60} * \text{PPR} * \dfrac{\text{COUNTS}}{\text{PULSE}}}$

** KTMIN, KTMAX = $\dfrac{\text{TIME IN MICROSECONDS}}{256}$

TABLE 3.2.1 SPEED MODE PARAMETER STRING

The input select byte is described in section 3.0. The calibration constant is a 24 bit unsigned number loaded in 3 bytes starting with the low byte KCLO. The table shows the expression for generating the value of Kc. The factor 0.80 is from the desire to have 100 percent speed output as 80 percent of full scale to allow for overspeed. The factor one million results from the count frequency of the time accumulator of one megahertz. RPM is the tach speed to be considered 100 percent of speed (80 percent of full scale output). PPR is pulses per revolution of the tach. Counts per pulse is either 1, 2, or 4 depending on the counting source selected by the input select byte.

Sample time constants KTMIN and KTMAX are 16 bit unsigned numbers

corresponding to the desired time in microseconds divided by 256. The lower limit is 5,000 microseconds which would be a count of 20.

DELTA and NOISE are 16 bit unsigned numbers containing limit settings for the fault detectors. See the Diagnostic section.

Operation

After initialization normal speed mode operation begins. The PPI will output speed data after the first full sample is complete. If the interrupt system is initialized for data the PPI will generate a bus interrupt whenever new data is available. Fault information will be returned in the status buffer and will generate interrupts if enabled.

Speed data is in signed two's complement integer format. Full scale forward speed is +7FFFH, full scale reverse is -7FFFH (8000H), and zero speed is 0000H. When setting the value of the calibration constant it is recommended to equate system base speed with 80 percent of full scale to allow for possible overspeed operation.

When a sample interval contains a speed reversal the speed output will be zero. This is because reversals cause confusion as to the correct number of pulses occurring during the sample time. If noise causes a spurious input pulse it may be interpreted as a reversal which zeroes the output. This is a serious disturbance if the system is running at full speed but it will be detected by the noise fault detector (see diagnostic section).

On-Line Commands

The PPI will respond to 4 commands in the speed mode. Commands are input through CHxIBUF and are acted upon whenever calculations and diagnostics are not being run. A command may be entered any time the input buffer is empty and will be serviced at least once per sample interval. The PPI speed mode can only respond to one command at a time. Entering a command when the input buffer is full may result in a command being lost, a command error, or the execution of an unwanted command. Use the handshake. Command errors are indicated by the status buffer. The table lists all the valid commands and their corresponding action.

NAME	CMND	ACTION
NULL	00H	Null Command, no action taken
ICALB	01H	Increment Calibration Constant
DCALB	02H	Decrement Calibration Constant
---	04H	Reserved
---	08H	Reserved
NEWMOD	10H	Change Modes (next CHxIBUF taken as new MODE)
---	20H	Reserved
---	40H	Reserved
---	80H	Reserved

TABLE 3.2.2 SPEED MODE ON-LINE COMMANDS

The calibration constant can be modified on line. By writing the command ICALB to CHxIBUF the constant will be incremented by one count. The command DCALB will decrement the constant by one count. Changing the constant by several counts requires writing the proper command once for each count.

The operating mode of a channel can be changed on line. By writing the command NEWMOD to CHxIBUF the channel will stop operating

in the speed mode within one sample period. The next byte in CHxIBUF is taken as the new mode value. New mode initialization can then begin in an identical manner to normal initialization.

Diagnostics

In addition to the normal internal diagnostic checks performed after reset there are on-line checks related to the speed mode. These are tach power supply, cable continuity, delta speed calculation, overspeed, noise detection, calibration constant overflow, and command check.

Tach power supply voltage can be tested on line by observing the supply monitor outputs. Cable continuity can also be checked on line. Both detectors can be accessed through the board ID register.

The other speed mode diagnostic information is posted once per sample approximately 40 microseconds after the speed value. An interrupt will be generated if the interrupt on status bit is set in the input select byte INPSEL. New diagnostic outputs are valid when the status buffer is full. If the old diagnostic outputs have not been read they are logically "or-ed" with the new outputs and posted again.

Delta speed compares tach speed at every sample with the preceeding sample to determine if the difference is greater then normal. In a properly operating system the rate of change of speed is limited by inertia and available torque. Excess speed change is a sign of tach failure or noise. The allowable speed change is set by initialization parameter DELTA which is a 16 bit unsigned number. If the absolute value of the change in speed is greater than DELTA, the flag DELTA_SPEED is set in CHxSTAT indicating the fault. Delta speed will be disabled if DELTA is initialized at 2**16-1 (FFFFH). Care must be taken in the selection of DELTA. If DELTA is too small the fault will be indicated by the normal acceleration or deceleration of the motor, while a large DELTA may never enable the fault. The constant is downloaded in byte form, where DELTALO is the lower byte and DELTAHI is the upper byte.

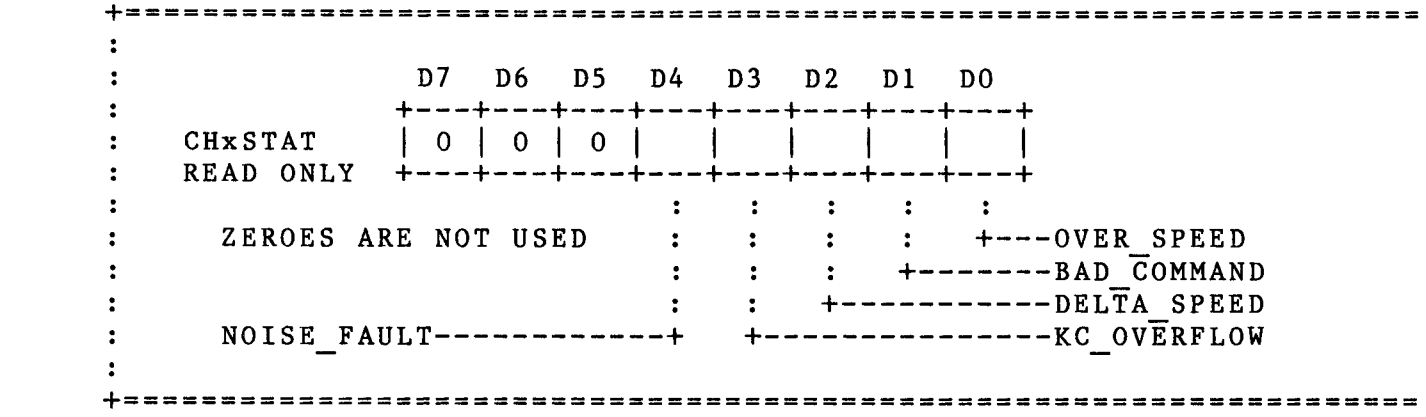


FIGURE 3.2.2 OUTPUT STATUS BUFFER (CHxSTAT) IN SPEED MODE

The OVER_SPEED flag will be set in CHxSTAT whenever the magnitude of calculated speed is greater than a 15 bit quantity. In that case the unsigned magnitude of the speed output is set to 2**15-1 (7FFFH). This condition may occur in 3 different cases:

Case 1: Np * Kc > 2**32-1
The numerator of the speed calculation is greater than a 32 bit quantity where Np is the number of tach pulses counted per sample interval and Kc is the calibration constant.

Case 2: (Np * Kc) / Nt > 2**15-1
The numerator divided by the denominator of the speed calculation is greater than a 15 bit quantity, where Np and Kc are given in case [1] and Nt is the number of time base clocks counted per sample interval.

Case 3
The quotient from case [2] was at 2**15-1 (7FFFH) and was to be rounded up based on the value of the remainder.

The noise detector prevents the speed output value from going to zero on a false reversal indication. False reversal indications can be caused by noise glitches on the tach pulses. The noise detector assumes a limit exists on the deceleration of the motor. A 16 bit speed threshold level NOISE is downloaded during the initialization sequence. If the speed from the previous sample interval was greater than the threshold then a reversal during the current sample interval is not physically possible and any reversal indication must be an error. The current speed is assigned the same direction as the previous speed and is calculated as though no reversal was indicated. The flag NOISE_FAULT in the CHxSTAT buffer is set to indicate the error condition. This flag will remain set for one sample period. If the bit remains set over several sample periods thee error is a recurring one. This function may be disabled by setting the threshold to full scale (7FFFH). Care must be used in selecting the threshold value so that actual reversals are not missed. If the threshold is set too low it is conceivable that a reversal could be missed and the speed value would never go to zero. The speed output would lock up in the wrong direction.

The calibration constant overflow flag KC_OVERFLOW in the status buffer is set in response to speed mode commands. An increment request when the constant equals FFFFFFH will be ignored and the flag will be set. A decrement request when the constant equals 000000H is handled the same way.

If an on-line command other than the valid commands is sent to the PPI in speed mode it will be ignored and the BAD_COMMAND flag will be set in CHxSTAT.

3.3 LINEAR PULSE ACCUMULATION MODE

Theory of Operation

The Linear Pulse Accumulator increments on all incoming pulses regardless of direction. Since the accumulator counts continuously the number of pulses counted over a time interval are

(1) $$Np = Fp * [t(j) - t(0)]$$

where Np is the number of incoming pulses (modulo 2**32), Fp is the average pulse frequency over the time interval, t(0) is the time at which the accumulator was last reset, and t(j) is the time at which the count is sampled.

Implementation

The pulse accumulator is a 16 bit hardware counter with 16 bit software overflow. The software counter is triggered by an 8031 interrupt from the overflow of the hardware counter. The counting rate is determined by the counting mode selected: times 1, times 2, or times 4 with respect to the input frequency (times 4 only with quadrature tach). The count is always in the up direction with a maximum count of 2**32-1 rolling over to a minimum count of 0. The maximum input frequency is limited by the tach input receivers at 50kHz. The counting rate can be 4 times that, or 200 kHz.

The accumulator can be reset, started, stopped, and read at the command of the bus master processor. Combinations of those commands are also possible. When the accumulator is read it is sampled on-the-fly so that no counts are missed.

Mode Initialization

Initialization of the PPI is described in section 3.0 up to the point where an operating mode is selected. The value of the mode select byte is 03H for the Linear Pulse Accumulation Mode. After this mode is selected a 16 byte parameter string must be downloaded through CHxIBUF as shown in the table. The only active parameter is the input select byte. This parameter is stored in internal memory of the 8031 microcomputer. The values in INPSEL should be set as follows:

BIDIREC = 0 (regardless of tach type)
INTONS = (desired value)
INTOND = 0
COUNT SELECT = (desired count).

After the parameters are downloaded the PPI goes through internal initialization for this mode which includes resetting and disabling the accumulator. The PPI begins operation by looping for a command input.

SEQUENCE	DESCRIPTION	
1.	INPSEL	INPUT SELECT BYTE (SEE SECTION 3.0)
2-16.	SPARE	NOT USED - SET TO ZERO

TABLE 3.3.1 LINEAR PULSE ACCUMULATION PARAMETER STRING

Operation

After initialization normal operation begins. The PPI will control the accumulator in response to on-line command inputs to the CHxIBUF. Fault information will be returned in the status buffer and will generate interrupts if enabled. Accumulator data is in unsigned integer (straight binary) format in two 16 bit words.

On-Line Commands

The PPI will respond to 5 commands in the linear pulse accumulation mode. Commands are input through CHxIBUF. A command may be entered any time the input buffer is empty. Entering a command when the input buffer is full may result in a command being lost, a command error, or the execution of an unwanted command. Use the handshake. Command errors are indicated by the status buffer. The figure shows all valid commands.

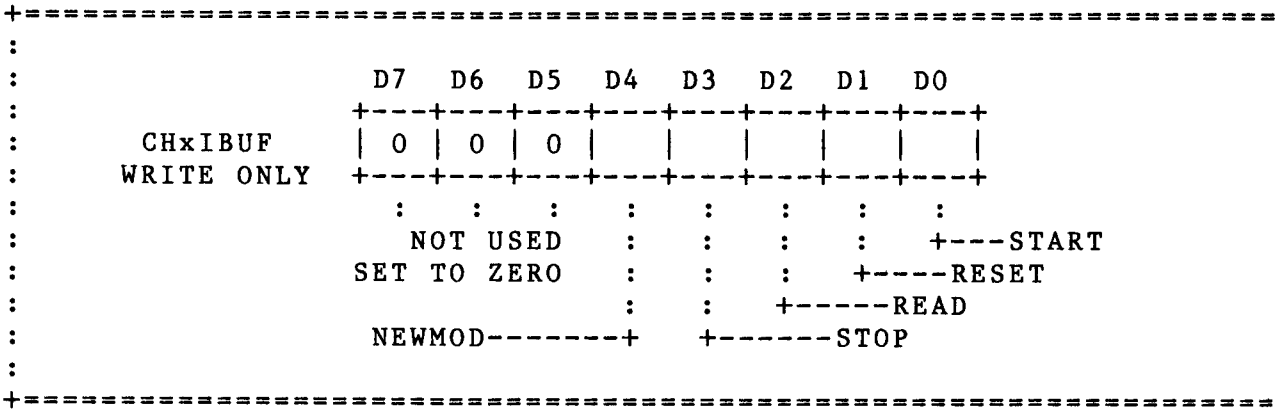


FIGURE 3.3.1 CHxIBUF IN LINEAR PULSE ACCUMULATION MODE

The PPI receives the command byte and scans it from left to right (bit D7 to D0) looking for any set bits. If a bit is set the corresponding command will be executed, then the scan will continue until all commands have been executed. Therefore it is possible to execute as many as 4 counter commands in 1 command byte. A command of 00H is ignored and the accumulator remains in its present state. The commands are:

NEWMOD: This bit will discontinue operation in the linear pulse accumulation mode. Any other set bits will be ignored. The next byte in CHxIBUF is taken as the new mode value. New mode initialization can then begin in an identical manner to normal initialization.

STOP: This bit disables the accumulator leaving it at its present count. If already stopped no action is taken.

READ: The accumulator can be read while stopped or read on-the-fly while it continues counting. Data is output to CHxOBUF with the upper word first followed by the lower word. The PPI waits for the bus master to read the upper word before it outputs the lower. The PPI will not execute any other commands until the master reads the upper and lower words. All transactions must use handshaking.

RESET: This bit resets the accumulator to zero while it is stopped or on-the-fly while it continues counting.

START: This bit enables the accumulator to start counting from its present value. If already started no action is taken.

The figure shows the timing for all commands. The total command response time depends on the combination of commands, and time spent waiting for the bus master to accept output data. All times are in microseconds. The timing of the READ subroutine can vary if the accumulator is sampled at a time when it is overflowing such that it has to be sampled twice. Also there is an uncertainty in all times due to the possibility of an 8031 interrupt occurrance. This happens whenever the accumulator overflows, and consumes up to 20 uS.

For example assume the bus master has just issued a command to reset the accumulator. Within 3 uS the command is accepted by the PPI. After 18 more uS the PPI has determined that the command is valid and that it should not change modes. It takes 3 uS each to check for STOP and READ commands, and then to verify that RESET is the desired command. It takes 6 uS to complete the reset action. Finally it takes 3 uS to return to the command input loop. In total it took 30 uS to decide what to do, 6 uS to do it, and 3 uS to return.

Diagnostics

In addition to the normal internal diagnostic checks performed after reset there are on-line checks related to this mode. These are tach power supply, cable continuity, and command check.

Tach power supply voltage can be tested on line by observing the supply monitor outputs. Cable continuity can also be checked on line. Both detectors can be accessed through the board ID register. If an on-line command other than the valid commands is sent to the PPI it will be ignored and the BAD_COMMAND flag will be set in CHxSTAT. If the interrupt-on-status flag INTONS was set in the input select byte then an interrupt will be sent to the bus master (provided the interrupt generator hardware is enabled).

CHxSTAT is updated after each command. The content of the status buffer is only valid when the buffer is full. If old diagnostic outputs have not been read they are logically "or-ed" with the new outputs and posted again.

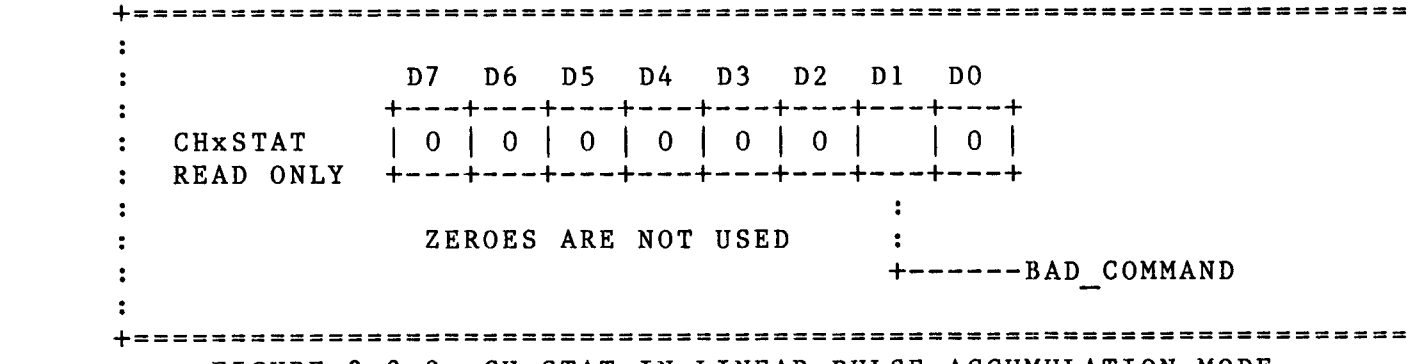


FIGURE 3.3.2 CHxSTAT IN LINEAR PULSE ACCUMULATION MODE

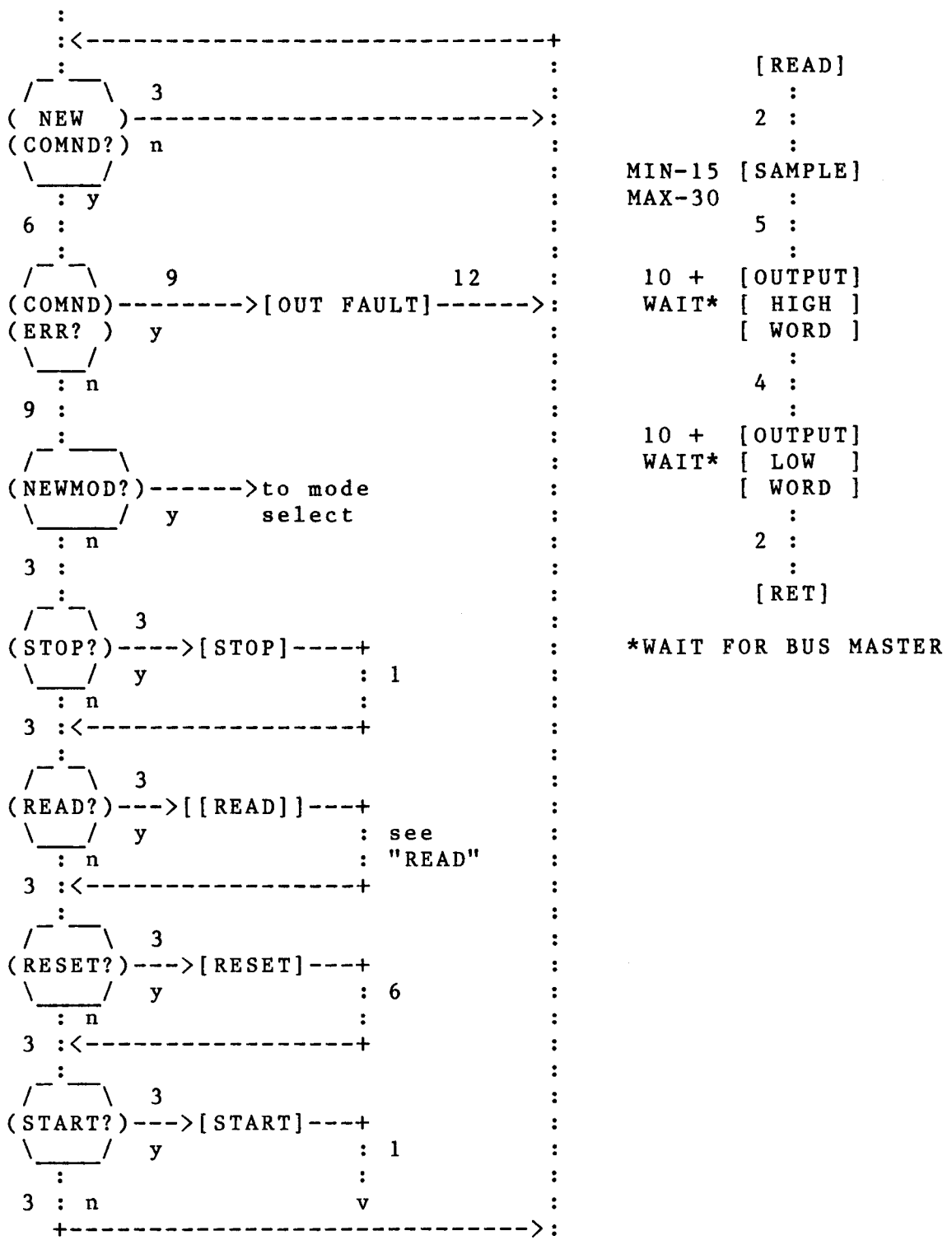


FIGURE 3.3.3 COMMAND RESPONSE TIMING (uS)

3.4 BIDIRECTIONAL PULSE ACCUMULATION MODE **** PRELIMINARY ****

Initialization

If the Pulse Accumulation Mode was selected (see section 3.0) more parameters must be downloaded through CHxIBUF. These parameters include the input select byte INPSEL, and a marker select byte MRKSEL.

The input select byte INPSEL configures the transition detection logic and also establishes the interrupt structure for the channel (see section 3.0).

The marker select byte MRKSEL is used to configure the marker circuitry. If the marker is enabled the accumulated pulse count will be zeroed each time a marker is received from the tach. Optionally, a fixed constant can be loaded instead of resetting the count. Optionally, a status flag MARK can be loaded into CHxSTAT, or a bus interrupt can be generated, or both.

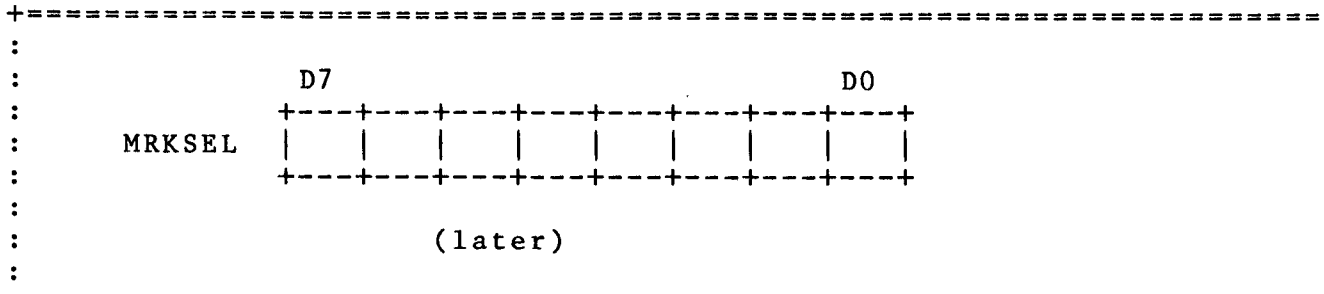


FIGURE 3.4.1 MARKER SELECT BYTE (MRKSEL)

Operation

After initialization normal pulse accumulation mode operation begins. The count is preset to zero. At the end of each sample interval the count is sent to CHxOBUF. Count data is in sixteen bit format where the most significant bit represents direction and the fifteen less significant bits are a straight binary number. The count will increment for forward direction and decrement for reverse. If the interrupt logic is so configured the PPI will interrupt the bus master at the end of every sample interval. Optionally the immediately current count can be requested once per sample period by writing the command CNTRQST to the CHxIBUF.

APPENDIX A1 PULSE CONDITIONER

Tachometers

The PPI and Pulse Conditioner were intended to operate with three types of tachometers: sine wave reluctance, single-ended square wave, and differential square wave. Generally the square wave tachs contain solid state optics and electronics and require a logic level power supply. The reluctance tachs are unpowered, but require a signal transformer for common mode isolation and impedance matching. These variations are accounted for by selecting groups on the Conditioner and by wiring modifications to the tach.

Reluctance tachs are low cost units that do not have low or zero speed (direction sensing) capability. The Pulse Conditioner must contain a matching transformer for each channel using a reluctance tach. Wire the tach as a differential circuit to the Phase B input only. Tach polarity is unimportant.

Square wave tachs require 5V or 15V isolated power supplies. The Pulse Conditioner must contain the proper supply for the particular tach in use. These tachs may have one output, two outputs (displaced 90 degrees), and/or a marker pulse output. Wiring is critical to ensure proper operation of the direction sensing logic and marker pulse logic on the PPI. Phase B leads Phase A by 90 degrees in the forward direction. The marker pulse must align with the Phase A pulse as shown below. Polarity must be correct for all outputs.

```
+-----------------------------------------------------------------+
:                                                                 :
:                      _________         _________                :
: PHASE A  __________/         \_________/         \_            :
:                                                     FORWARD     :
:              _________         _________                        :
: PHASE B  ____/         \_________/         \________ DIRECTION  :
:                                                       ----->    :
:                      _________                                  :
:  MARKER  __________/         \___________________              :
:                                                                 :
+-----------------------------------------------------------------+
```

FIGURE A.1.1 TACH TIMING CONVENTIONS

Differential output tachs wire directly to corresponding Pulse Conditioner inputs. They require line terminating resistors on each balanced twisted pair. These are supplied on certain groups of the Conditioner. Single ended tachs require a bias voltage (supplied on board) to be connected to the negative input, while the tach output connects to the positive input. Terminating resistors can not be used with these tachs.

Power Supply

Certain groups of the Pulse Conditioner contain a logic power supply capable of driving two square wave tachs. The input to the supply is always 110VAC and is fused on board. The output may be 5VDC or 15VDC depending on the group selected. The supply output is isolated from the AC line and from PSC.

Grounding

All wiring between the tachometers and the Pulse Conditioner must be shielded twisted pair cabling. Shields must be tied together and grounded to shield ground at one point only. The common (generally negative) side of the tach power supply should be connected to signal ground at one point in the system. Preferably this ground connection should be made at the Pulse Conditioner which requires a tachometer capable of floating above ground. If the ground connection must be

made at the tach, the PPI will operate properly in the presence of up to 55 volts peak common mode voltage. In no case should the connection to signal ground be made at both places as high ground current may be caused to flow in the tach signal wires.

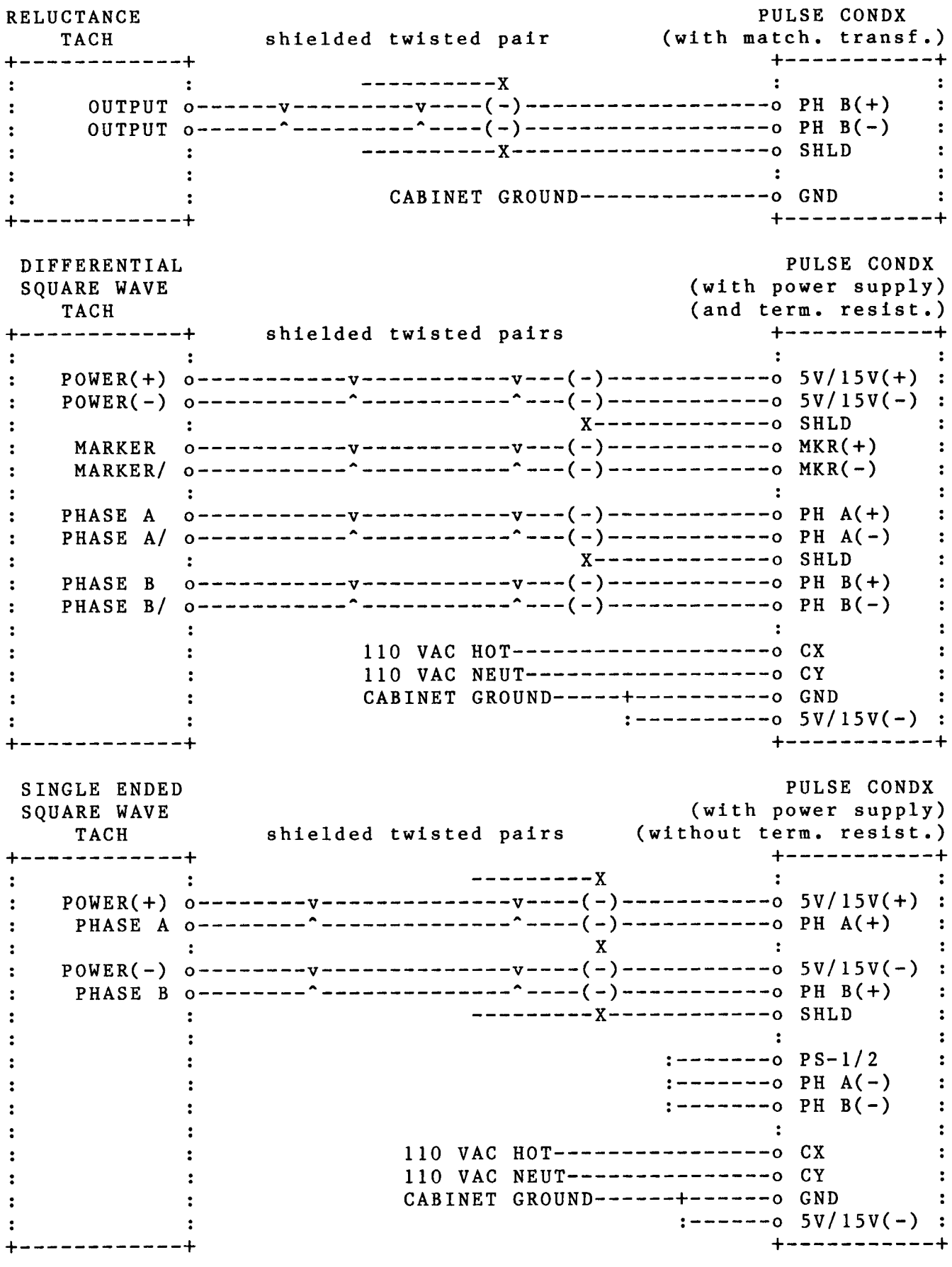


FIGURE A.1.2 RECOMMENDED TACH WIRING

APPENDIX A2 DATA BASE EXAMPLE

(later)

APPENDIX A3 PPI SPECIFICATIONS

PARAMETER	MIN	MAX	UNIT
1. Time Base			
Accuracy (12 MHz.)	-10	10	ppm
Stability (full temp range)	-25	25	ppm
2. Bus Access Time			
Normal Transfer	-	0.2	uS
Transfer with contention blocking logic	-	8.0	uS
3. Power Supply Requirements			
VCC supply voltage	4.75	5.25	VDC
VCC supply current (-G01 version)	-	2.50	ADC
VCC supply current (-G02 version)	-	1.80	ADC
Receiver supply voltage	11.40	28.00	VDC
Receiver supply current (at 12 volts)	-	5.0	mADC
Receiver supply current (at 24 volts)	-	10.0	mADC
4. Tach Input Receivers			
Input resistance	99k	109k	Ohms
Differential input (either polarity)	1.0	24.0	VDC
Hysteresis (typical)	1.0	1.0	VDC
Common mode input (12 V power supply)	-50	50	Vp
Common mode input (24 V power supply)	-80	80	Vp
Filter time constant	4	6	uS
Square wave input frequency	0	50	kHz
Overvoltage protection (transient)	-300	300	Vp
5. Environmental			
Temperature range	0	50	C
Humidity (non-condensing)	0	90	%
Shock (any axis)	-	2.0	G
Vibration (0 to 50 Hz, any axis)	-	1.5	G

6. Mechanical

Size 12.00H x 6.75W x 0.9T (inches)
Front connector 25 pin "D" type