# The Atari Page Printer Interface

The Atari Corporation

Sunnyvale, California

13 April 1988

Table of Contents

```
$Header: appi.me,v 1.1 88/04/13 01:26:56 art Rel $

$Source: /u/art/0_docs/RCS/appi.me,v $
$Author: art $

$Revision: 1.1 $
$Date: 88/04/13 01:26:56 $
$State: Rel $
$Locker: art $
$Log:   appi.me,v $
#  Revision 1.1  88/04/13  01:26:56  art
#  Initial revision
#
```

THE SCOPE OF THIS DOCUMENT is limited to a description of the Atari Page Printer Interface host adapter hardware and software system-level interface. This document does not provide information on the controller to printer engine device-level interface.

1. Introduction

The Atari Page Printer Controller functions as a translator between the parallel DMA bus interface (aka the Atari Computer System Interface) of the Atari ST and the serial video interface of the xerographic printer engine. The page printer controller is intelligent, providing device-independent command interpretation and control of printer operations. The following is a simplified block diagram of the Atari Page Printer Controller.

```
ATARI PAGE PRINTER CONTROLLER

      DMA            LPC                                DATA
      __           _______                              __
     |  |__________|       | <-- Horiz Sync ----      |  |
     |  |__________|       | --- Video Data --->      |  |
     |  |__________|       | <-- Video Clock -->      |  |
     |__|          |_______|                          |__|
                     |  |  |
                    _|__|__|_                          __
                   |        | <-- Print Ready ---     |  |
                   |        | --- Vert Sync ---->     |  |
                   |        | --- Command ------>     |  |
                   |________| <-- Status --------     |__|

                     MCU                               CONTROL
```

Bus communication follows a defined host-initiated dialog sequence consisting of a COMMAND PHASE, DATA OUT PHASE, and STATUS PHASE. Data transfers are asynchronous and follow a Data Request / Acknowledge handshake protocol, with one byte of data tranferred during each handshake.

Please note that the controller contains a two-byte FIFO which is loaded prior to the transfer of a new block of data.

The host can interrogate the controller for a list of configuration parameters which is received during an EXTENDED STATUS PHASE. The host can also send the controller a parameter list during an EXTENDED COMMAND PHASE.

## 2. Command Definitions

Commands are dispatched to page printer controller via a six-byte Command Descriptor Block. Please note that a delay is required between consecutive Command Descriptor Blocks to allow time for the controller to respond.

COMMAND DESCRIPTOR BLOCK

```
Byte 0      |oooooooo|
             | ||___|___ Operation Code
             |_|_________ Controller Number
Byte 1      |ooo-----|
             |_|_________ Device Number
Byte 2      |--------|
Byte 3      |--------|
Byte 4      |oooooooo|    Operation Length
Byte 5      |oo------|
             ||__________ Operation Modifiers
```

The Atari Page Printer Interface command set contains the following printer operations.

COMMAND SUMMARY TABLE

| OpCode | COMMAND |
|---|---|
| 0x03 | Request Sense |
| 0x0a | Print |
| 0x12 | Inquiry |
| 0x15 | Mode Select |
| 0x1a | Mode Sense |
| 0x1b | Stop Print |

## 2.1. 0x03 Request Sense

Upon receipt of this command, the controller performs an immediate printer status check and enters the STATUS PHASE to return the current status.

REQUEST SENSE

```
     Byte 0     |ooo00011|
                 | ||___|___ REQUEST SENSE OpCode (0x03)
                 |_|________ Controller Number
     Byte 1     |ooo-----|
                 |_|________ Device Number
     Byte 2     |--------|
     Byte 3     |--------|
     Byte 4     |--------|
     Byte 5     |--------|
```

Please refer to the STATUS DEFINITIONS section for a table of possible error codes.

## 2.2. 0x0a Print

Upon receipt of this command, the controller starts the printer and enters the DATA OUT PHASE to receive page images from the host.

```
PRINT

     Byte 0       |ooo01010|
                    | |  |___ PRINT OpCode (0x0a)
                    |_|______ Controller Number
     Byte 1       |ooo-----|
                    |_|______ Device Number
     Byte 2       |--------|
     Byte 3       |--------|
     Byte 4       |oooooooo|   Transfer Length
     Byte 5       |oo------|
                    ||_______ Preserve FIFO
                    |________ Generate H-Sync
```

Page images consist of bit-mapped data (photometric interpretation is always 0 = no mark, 1 = mark).

The Transfer Length specifies the definite number of pages to print. If the Transfer Length is zero, then one page is printed. If the Transfer Length is 255, then pages are printed indefinitely. The host must then terminate printing by sending a STOP PRINT command after receiving the next Status Byte. Please refer to the STOP PRINT command for more information.

Under normal multi-page operation, the two-byte FIFO in the controller is reset at the end of each page. However if the Preserve FIFO bit is set, then the FIFO is not reset until the end of the last page.

If the Generate H-Sync bit is set, then horizontal sync pulses are generated using the DMA Interrupt Request (_IRQ) signal. Each horizontal sync pulse is approximately 6 microseconds in duration.

Please note that Status Bytes are returned at the end of each page.

## 2.3. 0x12 Inquiry

Upon receipt of this command, the controller enters the EXTENDED STATUS PHASE and returns identification data consisting of device-specific information.

INQUIRY

```
     Byte 0     |ooo10010|
                 | ||___|___ INQUIRY OpCode (0x12)
                 |_|________ Controller Number
     Byte 1     |ooo-----|
                 |_|________ Device Number
     Byte 2     |--------|
     Byte 3     |--------|
     Byte 4     |--------|
     Byte 5     |1-------|
                 |__________ Extended Status
```

If byte 5 bit 7 is not set, then an Invalid Operation Code error is returned. Since the controller can not DMA data back to the host, the following Identification List is returned using the EXTENDED STATUS PHASE. Please note that a null Status Byte is returned before this operation.

IDENTIFICATION LIST

```
     Byte 0     |00000010|   Device Type (PRINTER)
     Byte 1     |--------|
     Byte 2     |--------|
     Byte 3     |--------|
     Byte 4     |oooooooo|   String Length
     Byte 5     |oooooooo|   Identification String
                     :
     Byte n     |oooooooo|
```

The Identification String contains ASCII characters describing the printer class, controller revision number, and controller manufacturer. The string is separated by colons and terminated by a space (for example "PAGE PRINTER:SLMC804v1.1:ATARI ").

## 2.4. 0x15 Mode Select

Upon receipt of this command, the controller enters the EXTENDED COMMAND PHASE and receives host-specified parameter data.

```
MODE SELECT

        Byte 0      |ooo10101|
                      | ||__|___ MODE SELECT OpCode (0x15)
                      |_|_______ Controller Number
        Byte 1      |ooo-----|
                      |_|_______ Device Number
        Byte 2      |--------|
        Byte 3      |--------|
        Byte 4      |--------|
        Byte 5      |o-------|
                      |_________ Reset Default
```

Please refer to the MODE SENSE command for a detailed description of the Parameter List. The host should perform a MODE SENSE before MODE SELECT in order to obtain the current controller state (except for Reset Default, where the current state is obliterated).

If the Reset Default bit is set, then the Parameter List bytes are not received but are internally reset to their power-up default values.

Please note that a Status Byte is returned following this operation.

## 2.5. 0x1a Mode Sense

Upon receipt of this command, the controller enters the EXTENDED STATUS PHASE and returns absolute, default, or current parameter data.

```
MODE SENSE

        Byte 0      |ooo11010|
                     | | |___|___ MODE SENSE OpCode (0x1a)
                     |_|_________ Controller Number
        Byte 1      |ooo-----|
                     |_|_________ Device Number
        Byte 2      |--------|
        Byte 3      |--------|
        Byte 4      |oooooooo|    List Length
        Byte 5      |o-------|
                     |___________ Return Absolute
```

An Absolute Parameter List contains the specifications of the printer, where each field contains its maximum value and each flag is set if its function is supported by the printer. A Default Parameter List is the first list returned after power up or following the execution of a MODE SELECT command with Reset Default. Any other list returned is a Current Parameter List and reflects the current internal operating state.

The List Length specifies the number of Parameter List bytes to return (not including the List Length byte). If the List Length is zero, then the entire Parameter List is returned.

If the Return Absolute bit is set, then the Parameter List contains the absolute printer specifications. The internal parameter data is not altered.

The following is returned using the EXTENDED STATUS PHASE. Please note that a Status Byte is returned before this operation.

PARAMETER LIST

```
    Byte 0      |oooooooo|   List Length

    Byte 1      |oooooooo|   Block Height (MSB)
    Byte 2      |oooooooo|   Block Height (LSB)

    Byte 3      |oooooooo|   Block Width (MSB)
    Byte 4      |oooooooo|   Block Width (LSB)

    Byte 5      |oooooooo|   Top Margin (MSB)
    Byte 6      |oooooooo|   Top Margin (LSB)

    Byte 7      |oooooooo|   Left Margin (MSB)
    Byte 8      |oooooooo|   Left Margin (LSB)

    Byte 9      |-ooooooo|
                  |||| ||___ Manual Feed
                  ||||_|____ Input Select
                  |||_______ Auto Select
                  ||________ Prefeed Paper
                  |_________ Thick Pixels

    Byte 10     |oooooooo|   Vertical Resolution (MSB)
    Byte 11     |oooooooo|   Vertical Resolution (LSB)

    Byte 12     |oooooooo|   Horizontal Resolution (MSB)
    Byte 13     |oooooooo|   Horizontal Resolution (LSB)

    Byte 14     |oooooooo|   System Timeout

    Byte 15     |oooooooo|   Scan Time (MSB)
    Byte 16     |oooooooo|   Scan Time (LSB)

    Byte 17     |oooooooo|   Page Count (MSB)
    Byte 18     |oooooooo|   Page Count (LSB)

    Byte 19     |oooooooo|   Input Capacity (MSB)
    Byte 20     |oooooooo|   Input Capacity (LSB)

    Byte 21     |oooooooo|   Output Capacity (MSB)
    Byte 22     |oooooooo|   Output Capacity (LSB)
    Byte 23     |--oooooo|
                   ||| ||___ Stagger Output
                   |||_|____ Output Select
                   ||_______ Duplex Print
                   |________ Color Separation

    Byte n      |--------|   Reserved
```

List Length

List Length specifies the number of Parameter List bytes to receive or return (not including the byte itself).

Block Height

Block Height specifies the vertical block extent in scan lines (this is changed automatically depending on the installed paper size except after a MODE SELECT command without Reset Default). On Paper Empty the default size is letter.

Block Width

Block Width specifies the horizontal block extent in pixels (this is changed automatically depending on the installed paper size except after a MODE SELECT command without Reset Default). On Paper Empty the default size is letter.

Top Margin

Top Margin specifies the top vertical block offset in scan lines (Top Margin is added to Block Height to obtain the bottom margin).

Left Margin

Left Margin specifies the left horizontal block offset in pixels (Left Margin is added to Block Width to obtain the right margin).

Manual Feed

If the Manual Feed bit is set, then paper is input through the manual paper feed (this bit is in effect "force manual feed", since the controller by default tries to feed paper manually whenever it detects a Paper Empty error).

Input Select

Input Select specifies which input to use for automatic paper feed (this is device dependent and should be set to zero for printers with one automatic paper feed).

Auto Select

If the Auto Select bit is set, then if the currently selected automatic paper feed is empty and an alternate paper feed containing the same paper size is available, then the alternate paper feed is automatically selected and no error is reported.

Prefeed Paper

If the Prefeed Paper bit is set, then paper is advanced and readied for immediate printing.

Thick Pixels
If the Thick Pixels bit is set, then the printer is a write-to-black (or equivalent) device.

Vertical Resolution
Vertical Resolution specifies the latitudinal resolution in dots per inch.

Horizontal Resolution
Horizontal Resolution specifies the longitudinal resolution in dots per inch.

System Timeout
System Timeout specifies the timeout in seconds for manual feed input and other wait-until conditions.

Scan Time
Scan Time specifies the horizontal scan interval in microseconds.

Page Count
Page Count contains the number of pages printed since power up or reset.

Input Capacity
Input Capacity specifies the page capacity of the currently selected automatic paper feed input.

Output Capacity
Output Capacity specifies the page capacity of the currently selected paper output tray.

Stagger Output
If the Stagger Output bit is set, then paper is output in an offset stack and can be toggled between print jobs.

Output Select
Output Select specifies which tray to use for paper output (this is device-dependent and should be set to zero for printers with one paper output tray).

Duplex Print
If the Duplex Print bit is set, then the printer produces double-sided pages.

Color Separation
If the Color Separation bit is set, then the printer produces color images using four-color separation (incredibly device dependent).

## 2.6. 0x1b Stop Print

Upon receipt of this command, the controller terminates a multi-page print operation at the end of the current page.

STOP PRINT :

```
      Byte 0      |ooo11011|
                   | || |___|____ STOP PRINT OpCode (0x1b)
                   |_|___________ Controller Number
      Byte 1      |ooo-----|
                   |_|___________ Device Number

      Byte 2      |--------|
      Byte 3      |--------|
      Byte 4      |--------|
      Byte 5      |--------|
```

This command is valid only after receiving a definite or indefinite multi-page PRINT command.

The print operation is terminated at the end of the current page because the controller must tell the printer to stop (or start) the next page while printing the current page. This can be a problem for printer drivers that indeterminately process a document on a byte-by-byte basis.

There is a 100 millisecond window after receiving the status in which a STOP PRINT command can be issued. An Invalid Operation Code error will be generated if this command is not issued during a multi-page PRINT operation.

Please note that one Status Byte is returned at the end of the last printed page.

## 3. Status Definitions

A Status Byte is returned following the successful or unsuccessful execution of a command on a selected device.

STATUS BYTE

```
Byte 0     |oooooooo|
            | ||___|___ Error Code
            |_|________ Device Number
```

The current printer status is checked at the end of command execution. In the case of a PRINT command, however, the current printer status is checked both before and after the command is executed. If an error is detected before PRINT execution, then the controller immediately aborts and returns a Status Byte. (The only exceptions are Toner Empty and Drum Empty, where PRINT is executed and the error is reported after execution, and Paper Empty, where the error is not reported in manual feed mode.)

The Status Byte will return the following error codes in low-to-high priority.

STATUS SUMMARY TABLE

| ErCode | STATUS | |
|---|---|---|
| 0x00 | No Error | |
| 0x02 | Ornery Printer | Printer |
| 0x03 | Toner Empty | |
| 0x04 | Warm Up | |
| 0x05 | Paper Empty | |
| 0x06 | Drum Empty | |
| 0x07 | Input Jam | |
| 0x08 | Through Jam | |
| 0x09 | Output Jam | |
| 0x0a | Cover Open | |
| 0x0b | Fuser Malfunction | |
| 0x0c | Imager Malfunction | |
| 0x0d | Motor Malfunction | |
| 0x0e | Video Malfunction | |
| 0x10 | System Timeout | Controller |
| 0x12 | Invalid Operation Code | Command |
| 0x15 | Invalid Device Number | |
| 0x1a | Invalid Parameter List | |

No Error

This code is returned when an error is not detected and marks the successful completion of command execution.

Ornery Printer

This is a catch-all code and is returned when the controller does not have an appropriate error code for a given printer error.

Toner Empty

This code is returned when the toner supply is exhausted (not fatal).

Warm Up

This code is returned while the fuser unit is warming up.

Paper Empty

This code is returned when the paper supply is exhausted in the currently selected automatic paper feed (not returned if manual feed).

Drum Empty
This code is returned when the drum surface is exhausted (not fatal).

Input Jam
This code is returned when an entry paper jam is detected.

Through Jam
This code is returned when a print paper jam is detected.

Output Jam
This code is returned when an exit paper jam is detected.

Cover Open
This code is returned while the printer cover is open.

Fuser Malfunction
This code is returned when an error is detected in the fuser unit (the fuser thermally fixes the toner to the paper).

Imager Malfunction
This code is returned when an error is detected in the imager unit (the imager generates and scans an image onto the drum).

Motor Malfunction
This code is returned when an error is detected in the motor unit (the motor drives the paper feed, drum, toner hopper, and fuser pulleys).

Video Malfunction
This code is returned when an error is detected in the video unit (the video controller provides the external "video" interface and controls the printer housekeeping functions).

System Timeout
This code is returned when the controller times out on a wait-until condition (eg manual feed input).

Invalid Operation Code
This code is returned when a given command is not implemented.

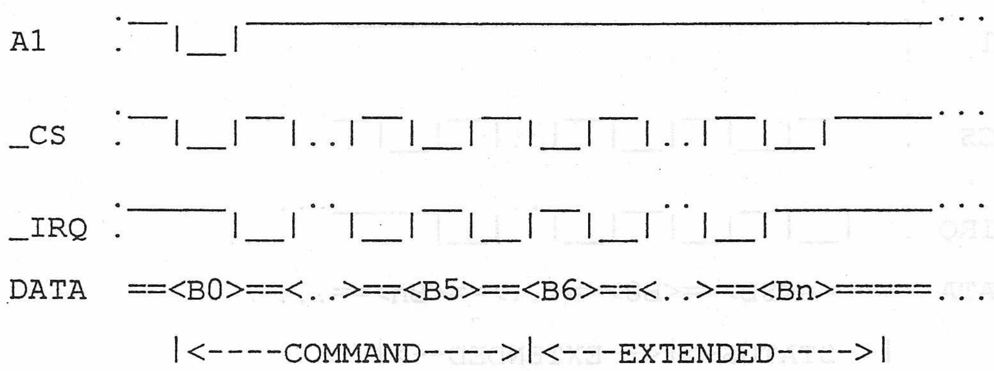
Invalid Device Number
This code is returned when a given device does not exist.

Invalid Parameter List
This code is returned when a given parameter is out of bounds.

## 4. Extended Command Phase

In the EXTENDED COMMAND PHASE each data byte is followed by a strobe of the DMA Interrupt Request (_IRQ) signal. This phase is used to transfer MODE SELECT data to the controller.

EXTENDED COMMAND PHASE


A1
_CS
_IRQ
DATA
==<B0>==<..>==<B5>==<B6>==<..>==<Bn>=====...
|<----COMMAND---->|<---EXTENDED---->|

Please note that after writing each command byte, the host must wait a minimum of 20 microseconds before testing the state of _IRQ (see Appendix). The controller does not raise _IRQ on the falling edge of _CS.

## 5. Extended Status Phase

In the EXTENDED STATUS PHASE each data byte is preceded by a strobe of the DMA Interrupt Request (_IRQ) signal. This phase is used to transfer MODE SENSE and INQUIRY data to the host.

```
EXTENDED STATUS PHASE

     ._____________________________...
A1   .

     ._____    __    __    __    __...
_CS  .     |__|  |__|  |..|  |__|

     .__    __    __    ..    _____...
_IRQ .  |__|  |__|  |__|  |__|

DATA  =====<SB>==<B0>==<..>==<Bn>==...

      |<-STATUS->|<--EXTENDED-->|
```

Please note that after reading each status byte, the host must wait a minimum of 20 microseconds before testing the state of _IRQ (see Appendix). The controller does not raise _IRQ on the falling edge of _CS.

Appendix -- Driver Example

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
                                                                          *
     PRINT PAGE                                                           *
     Print page of bit-mapped data to laser printer via DMA port.         *
                                                                          *
     (c) 1988 Atari Corporation                                           *
         All Rights Reserved.                                             *
                                                                          *
     Word address of letter-size page image in _page_image.  Total        *
     page image size is 954000 bytes (300 bytes by 3180 lines).           *
                                                                          *
     Physical width and height of image on letter-size paper is 8         *
     inches by 10.6 inches (20.32 cm by 26.924 cm).                       *
                                                                          *
     Laser printer controller number is 7.                                *
                                                                          *
     Inputs:          _page_image                                         *
                                                                          *
     Outputs:         returns -1 in d0 on error, else returns status      *
                                                                          *
     Modified:        d0, d1, d2, d3, d7, a0, a1                          *
                                                                          *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

     .globl   _print_page     ;
     .globl   _page_image     ;

     DEFINITIONS

madata equ    $ffff8604       ; /* dma control and status register */
mahigh equ    $ffff8609       ; /* dma counter high */
mamid  equ    $ffff860b       ; /* dma counter mid */
malow  equ    $ffff860d       ; /* dma counter low */
pip    equ    $fffffa01       ; /* mfp general purpose i/o */
lock   equ    $43e            ; /* dma lock semaphore */
hz_200 equ    $4ba            ; /* 200 hz timer */

     .text

print_page:
     lea      dmadata,a0      ;
     lea      dmahigh,a1      ;
     st       flock           ; /* lock dma channel */

  COMMAND PHASE

     move.w   #$88,2(a0)      ; /* assert command signal */
     move.l   #$ea008a,d0     ; /* PRINT command byte 0 */
```

```
        jsr     writcmnd          ; /* to controller number 7 */
        bne     prabrt            ;
        move.l  #$8a,d0           ; /* command byte 1 */
        jsr     writcmnd          ;
        bne     prabrt            ;
        move.l  #$8a,d0           ; /* command byte 2 */
        jsr     writcmnd          ;
        bne     prabrt            ;
        move.l  #$8a,d0           ; /* command byte 3 */
        jsr     writcmnd          ;
        bne     prabrt            ;
        move.l  #$8a,d0           ; /* command byte 4 */
        jsr     writcmnd          ;
        bne     prabrt            ;
        move.l  #$82,d0           ; /* command byte 5 */
        move.l  d0,(a0)           ; /* no acknowledge for byte 5 */
        moveq.l #2,d1             ; /* ~ 5 millisecond delay */
        add.l   _hz_200,d1        ; /* NB MINIMUM DELAY IS 20 MICROSECONDS */
codel:  cmp.l   _hz_200,d1        ;
        bge     codel             ;
```

DATA OUT PHASE

```
        move.l  _page_image,d0    ; /* load initial band base */
        move.l  d0,-(sp)          ;
        move.b  3(sp),dmalow      ; /* initialize dma base address */
        move.b  2(sp),dmamid      ;
        move.b  1(sp),dmahigh     ;
        addq.l  #4,sp             ;
        move.w  #$192,2(a0)       ; /* select sector count register */
        move.l  #$4c0112,(a0)     ; /* write sector count, start dma */
        clr.b   d3                ; /* clear two-byte FIFO adjust flag */
        moveq.l #23,d7            ; /* do 23 more bands */
bdloop: addi.l  #38400+32,d0      ; /* get final address + 32 */
        move.w  sr,d1             ; /* save status register */
        ori.w   #$700,sr          ; /* no interrupts, please */
bdwait: btst.b  #5,gpip           ; /* check for premature status phase */
        beq     stbyte            ; /* abort and get status byte */
        movep.w 2(a1),d2          ; /* get current DMA mid and low */
        cmp.w   d2,d0             ; /* compare to final address */
        bne     bdwait            ; /* not there yet? */
        subi.l  #32,d0            ; /* point to next band base */
        tas.b   d3                ; /* test two-byte FIFO adjust flag */
        bne     noadju            ; /* do not adjust next base? */
        addi.l  #2,d0             ; /* compensate for two-byte FIFO */
noadju: move.l  d0,-(sp)          ;
        move.b  3(sp),dmalow      ; /* reinitialize DMA base address */
        move.b  2(sp),dmamid      ;
        move.b  1(sp),dmahigh     ;
        addq.l  #4,sp             ;
        move.w  #$92,2(a0)        ; /* reset FIFO */
        move.w  #$192,2(a0)       ;
        move.l  #$4c0112,(a0)   ; /* reload sector count register */
        move.w  d1,sr           ; /* restore status register */
        dbra    d7,bdloop       ; /* more bands? */

*
*  STATUS PHASE
*
stwait: btst.b  #5,gpip         ; /* wait for status byte */
        bne     stwait          ;
stbyte: move.w  #$8a,2(a0)      ; /* select status register */
        move.w  (a0),d0         ; /* read status byte */
        moveq.l #2,d1           ; /* ~ 5 millisecond delay */
        add.l   _hz_200,d1      ; /* NB MINIMUM DELAY IS 20 MICROSECONDS */
stdel:  cmp.l   _hz_200,d1      ;
        bge     stdel           ;
        bra     prexit          ; /* return status byte */
prabrt: moveq.l #-1,d0          ; /* return error flag */
prexit: sf      flock           ; /* unlock dma channel */
        rts                     ;

*
*       WRITCMND
*       Write command byte to DMA controller.
*
*       Inputs:         d0.L = Data/control words
*                       a0   = Pointer to DMA controller (ff8604)
*       Outputs:        EQ = Successful command write
*                       NE = Error occurred
*       Modified:       d1
*
writcmnd:
        move.l  d0,(a0)         ; /* write command byte */
        moveq.l #2,d1           ; /* ~ 5 millisecond delay */
        add.l   _hz_200,d1      ; /* NB MINIMUM DELAY IS 20 MICROSECONDS */
wrdel:  cmp.l   _hz_200,d1      ;
        bge     wrdel           ;
        moveq.l #40,d1          ; /* 200 millisecond timeout */
        add.l   _hz_200,d1      ;
writlp: btst.b  #5,gpip         ;
        beq     writok          ; /* command byte acknowledged */
        cmp.l   _hz_200,d1      ;
        bge     writlp          ;
        moveq.l #-1,d1          ; /* timeout - set error flag */
writok: rts                     ;
```