From: Mike Barall 04/30/84
Subject: RAM available to parallel device handlers.

1. CARD SLOTS

The parallel I/O system can support a maximum of 8 parallel devices. Each device occupies one of eight card slots. The card slots are numbered from 0 to 7, with slot 0 being the highest priority device and slot 7 being the lowest priority device.

All parallel device handlers are bank selected over floating-point ROM ($D800-$DFFF). Thus, the device handler code does not need to be relocatable in order for it to be able to function in any slot. A handler can determine which slot it is in by examining OS variable SHPDVS ([$248,1]); bit 0 is set if slot 0 is selected, bit 1 is set if slot 1 is selected, and so on.

2. Page $D6xx and $D7xx RAM

Memory locations $D600-$D7FF are reserved for use by parallel device handlers. This RAM is not used at all by the OS; in particular, it is not cleared during coldstart or warmstart.

Each card slot has a portion of this RAM allocated to it, according to the following scheme:

$D600 - $D61F	Slot 0 RAM
$D620 - $D63F	Reserved for use by modem devices
$D640 - $D67F	Slot 1 RAM
$D680 - $D6BF	Slot 2 RAM
$D6C0 - $D6FF	Slot 3 RAM
$D700 - $D73F	Slot 4 RAM
$D740 - $D77F	Slot 5 RAM
$D780 - $D7BF	Slot 6 RAM
$D7C0 - $D7FF	Slot 7 RAM

As the table indicates, slots 1-7 each own 64 bytes of RAM, while slot 0 owns 32 bytes of RAM. There are 32 bytes reserved for use by modem devices because 64 bytes is not enough RAM to support a modem.

Obviously, a parallel device handler which uses more than 32 bytes of page $D6xx and $D7xx RAM cannot be placed in slot 0. With this one exception, parallel device handlers should be designed so that they will function properly in any slot. In particular, before accessing page $D6xx and $D7xx RAM, the handler must determine which slot it is in (by examining SHPDVS) and do an address calculation.

3. ZEROPAGE RAM

There are 11 bytes on page zero which can be used by parallel device handlers (except during IRQ processing):

STATUS	[$30,1]
CHKSUM	[$31,1]
BUFRLO	[$32,1]
BUFRHI	[$33,1]
BFENLO	[$34,1]
BFENHI	[$35,1]
BUFRFL	[$38,1]
RECVDN	[$39,1]
XMTDON	[$3A,1]
CHKSNT	[$3B,1]
NOCKSM	[$3C,1]

These 11 bytes are normally reserved for use by SIO. Since it is not possible for SIO to be active at the time that a parallel device handler is called (except during IRQ processing), parallel device handlers may use these locations freely; their original values do not have to be saved and restored.

In addition, there are 4 bytes on page zero that are reserved for use by parallel device IRQ routines:

ABUFPT	[$1C,4]

These 4 bytes are not used at all by the OS. Parallel device IRQ routines need not save and restore the original values of these memory locations.

Parallel device IRQ routines which need more than 4 bytes of zeropage RAM should use the 11 SIO bytes listed above; naturally, the IRQ routine must save and restore the original values of these bytes. Parallel device IRQ routines needing more than 15 bytes of zeropage RAM, and parallel device non-IRQ routines needing more than 11 bytes of zeropage RAM, can get more by saving and restoring other locations on page zero. The safest spot to use for this purpose is probably the zeropage IOCB ([$20,$10]).

4. OTHER RAM

The following memory locations may be used as scratch storage by non-IRQ parallel device handler routines. They are normally reserved for use by SIO.

```
      CDEVIC      [$23A,1]
      CCOMND      [$23B,1]
      CAUX1 [$23C,1]
      CAUX2 [$23D,1]
      TEMP  [$23E,1]
      ERRFLG      [$23F,1]
      CRETRY      [$29C,1]
      DRETRY      [$2BD,1]
      TIMFLG      [$317,1]
      STACKP      [$318,1]
      TSTAT [$319,1]
```

5. THE DATA CONTROL BLOCK

Parallel device handler routines which are entered through the low-level entry point ($D805) are expected to read the DCB ([$300,$C]) to determine what command they are to execute. These routines may NOT modify the DCB, except for DUNIT ([$301,1]). The original value of DUNIT is saved by PIO before the parallel device handler is called, and it will be restored by PIO after the parallel device handler returns. (Note: The restoration takes place only at the very end of PIO; thus, if one parallel device handler changes DUNIT and then returns indicating that it did not handle the call, subsequent parallel device handlers and SIO will see the modified value of DUNIT.)

By the way, the parallel deivce handler should add DDEVIC ([$300,1]) and DUNIT to determine whether or not is being addressed; simply examining DDEVIC is not enough.

6. VECTORS

Parallel device handlers should provide, in page $D6xx and $D7xx RAM, whatever vectors are necessary to replace the parallel device handler with a RAM-resident handler. Prime candidates for vectoring are the IRQ entry point (through $D808) and the low-level entry point (through $D805).

There is no uniform scheme for vectoring due to the fact that any program which intercepts one of these vectors is going to be extremely device-specific anyway.

7. A BIZARRE WARNING

Page $D5xx is used for hardware locations within cartridges. Therefore, when parallel device drivers access their RAM on pages $D6xx and $D7xx, they must be careful not to generate any references to page $D5xx.

Due to a quirk in the 6502 microprocessor, this is not as easy as it sounds. Whenever indexed addressing is used to cross a page boundary, the 6502 will generate an extra memory cycle during which it references the RAM location one page below the desired address. For example, consider:

```
      LDX   #$FE
      LDA   $D510,X
```

During the execution of the LDA instruction, the 6502 will generate a reference to memory location $D50E, as well as to the desired address $D60E.

This applies not only to LDA instructions, but to all instructions (including stores). It also applies to indirect indexed addressing, as in LDA (ZPAGE),Y.