IPC Commands and Communication Protocol

Nomenclature:

Nibble:	*4 bits*	
Byte:	*8 bits*	*(two nibbles)*
Word:	*16 bits*	*(four nibbles)*

IPC Commands

0 Reset — **resets the IPC**

1 Status — **report input status**

Returns: 1 byte

Bits of which are:

0 set if data available in keyboard buffer, or key held
1 set if sound is still being generated
2 set if keyboard shift setting has changed, with key held
3 set if key held down
4 set if input is pending from RS232 channel 1
5 set if input is pending from RS232 channel 2
6 return state of p26, currently not connected
7 set if serial transfer was cleared, now zero

2 OPEN SER1 — **open RS232 channel 1**
3 OPEN SER2 — **open RS232 channel 2**
4 CLOSE SER1 — **close RS232 channel 1**
5 CLOSE SER2 — **close RS232 channel 2**

These open/close RS232 channels, affecting handshaking.

6 READ SER1 — **read RS232 channel 1**
7 READ SER2 — **read RS232 channel 2**

Return: 1 byte

Upper 2 bits	Status information
Lower 6 bits	byte count to follow.

Maximum number of data bytes: 25

8 READ KEYBOARD — **read keyboard**

Returns:

nibble:

ms bit:	set if final last keydef is still held
3 bits:	count of keydefs to follow.

then, for each of the 0..7 keydefs:

nibble:
bits:
3 Lost keys
2 Shift
1 Ctrl
0 Alt
byte:
top two bits: ignore
three bits: Column
three bits: Row

As per key row table.

There is a version of the IPC used on the thor that will also return keydef values for a keypad.

9 READ KEYROW keyboard direct read

Input: 1 nibble Keyboard column, encoded in lower three bits.
Return: 1 byte Keys held down, as defined in the key row table

10 SOUND initiate sound

Input: (All bytes/words are big-endian.)
Byte: pitch 1
Byte: pitch 2
Word: interval between steps
Word: duration (0=forever)
Nibble: signed step in pitch
Nibble: wrap
Nibble: randomness (none unless msb is set)
Nibble: fuzziness (none unless msb is set)

11 STOP SOUND immediately stop sound generation

12 MDV REDUCE SENSE Microdrive reduced sensitivity

Input: 1 nibble Least significant bit only used.

This was intended for allowing marginal tapes to be handled, but had a strange side effect of causing permanent level 2 interrupts!

The QView capsled kit uses this bit of h/w (on leg of the chip) to drive the led.

Hermes uses a parameter value of 9 to switch on its clever handling, when the reset command becomes an extended functionally command and the test command will return the complemented byte value.

13 SET BAUD RATE change baud rate

Input: 1 nibble (Three least significant bits.)

Baud rate bit values:

75	0111
300	0110
600	0101
1200	0100
2400	0011
4800	0010
9600	0001
19200	0000

The actual clock rate is supplied from the ZX8302 to the IPC, but this command is also needed in the IPC for timing out transfers.

14 RANDOM NUMBER random number generator

Return: 1 word 16 bit random number

15 SELF-TEST LOOPBACK test

Input: 1 byte Test data
Return: 1 byte Test data

IPC Communication Protocol

Communication with the IPC is done using the following bit serial protocol via the ZX8302 registers.

Commands and data are sent in big-endian format.

The data is then clocked out one bit at a time by writing a byte containing %000011x0 to the address ZX8302base+0x23, where the "x" is current data bit. The ZX8302+0x20 is then polled until the acknowledge bit (6) goes to zero.

Data is received by clocking the data in by writing %00001110 to ZX8302base+0x23 for each bit of the data, and then polling ZX8302base+0x20 until the acknowledge bit (6) is zero, at which point bit 7 holds the valid data bit.

Note: ZX8302base+0x22 is where full bytes are sent for serial output, with an interrupt being generated when the data has completed transmission.

This document is based upon “ipccmd” from the Minerva source code distribution.