Tuesday, April 08, 2003

TNG-4 Command Mode:

Command Mode is a separate code module from streaming mode. The argument is that anyone doing streaming wouldn't be doing Command Mode and *vice versa*. Commands may be set in groups. In general, when a command returns data, the device issuing the command should wait for the expected number of bytes to be received before issuing the subsequent commands. The TNG-4 transmit buffer is 78 bytes deep.

Normally, TNG-4 is set for 19.2k Baud, 1 stop bit, and no parity.

TNG-4 commands are ASCII bytes (To send the "9D" command, transmit a single byte equal to that value, not the ASCII characters "9" and "D"). Commands bytes values are in hexadecimal. The following commands are supported:

1. **9D:** Return ID/version. This command returns 10 bytes: an 8-byte ID number (fixed for now as "MindTel"), and a two-byte software revision (major, minor), "C1".

2. **B8:** Set 8-bit ADC conversions. No bytes are returned. This is the default mode.

3. **BE:** Set extended, 10-bit or 12-bit conversions. This command establishes the return of 10 or 12-bit conversion data until receipt of a B8 command in chips supporting the extended ADC resolution. No bytes are returned.

4. **EF:** Set ADC data left justified. This is the default mode for extended conversions where the data is sent MSB, LSB and the least significant bits of the LSB are zero. No bytes are returned.

5. **E0:** Set ADC data right justified. This command right-justifies the bits of an extended resolution A/D conversion in the returned MSB, LSB bytes. No bytes are returned.

6. **CA:** Read all ADC channels. This command returns the data for all 8 ADC channels (8 bytes, or 16 bytes in extended resolution). The data for each channel is sent most significant byte (MSB) first followed by the least significant data byte (LSB) for each ADC channel starting with channel 1.

7. **C0:** Read the first N ADC channels where N ranges from 1 to 8. The number of consecutive ADC channels to read is sent in a byte immediately following the command. Depending on the ADC resolution and the number of channels, this command returns 1-16 bytes of data.

8. **C1:** Read ADC channel 1. This command returns 1 or 2 bytes of ADC data for ADC channel 1, depending on the ADC resolution.

9. **C2:** Read ADC channel 2. This command returns 1 or 2 bytes of ADC data for ADC channel 2, depending on the ADC resolution.

10. **C3:** Read ADC channel 3. This command returns 1 or 2 bytes of ADC data for ADC channel 3, depending on the ADC resolution.

11. **C4:** Read ADC channel 4. This command returns 1 or 2 bytes of ADC data for ADC channel 4, depending on the ADC resolution.

12. **C5:** Read ADC channel 5. This command returns 1 or 2 bytes of ADC data for ADC channel 5, depending on the ADC resolution.

13. **C6:** Read ADC channel 6. This command returns 1 or 2 bytes of ADC data for ADC channel 6, depending on the ADC resolution.

14. **C7:** Read ADC channel 7. This command returns 1 or 2 bytes of ADC data for ADC channel 7, depending on the ADC resolution.

15. **C8:** Read ADC channel 8. This command returns 1 or 2 bytes of ADC data for ADC channel 8, depending on the ADC resolution.

16. **DA:** Write all DAC values. This command takes the next four received bytes as the values for DAC channels 1-4 and updates the DAC.

17. **D1:** Write DAC channel 1. This command takes the next received byte as the data to update DAC channel 1.

18. **D2:** Write DAC channel 2. This command takes the next received byte as the data to update DAC channel 2.

19. **D3:** Write DAC channel 3. This command takes the next received byte as the data to update DAC channel 3.

20. **D4:** Write DAC channel 4. This command takes the next received byte as the data to update DAC channel 4.

21. **BC:** Set Port-B configuration. This command takes the next byte received as the Port-B configuration data byte. Each bit of the configuration byte corresponds to a port bit. Set configuration bits to 1 to enable input, 0 for output. This is similar for Ports C and D.

22. **CC:** Set Port-C configuration. This command takes the next byte received as the Port-C configuration data byte. Bits 0,6, and 7 are masked (can't be reconfigured).

    Port C bit 0 = test switch (input)
    Port C bit 1 = DAC enable line (output)
    Port C bit 3 = SPI clock (output)
    Port C bit 4 = SPI data in (input)
    Port C bit 5 = SPI data out (output)
    Port C bit 6 = RS-232 data out (output)
    Port C bit 7 = RS-232 data in (input)

23. **DC:** Set Port-D configuration. This command takes the next byte received as the Port-D configuration data byte.

24. **AC:** Set configuration of all three digital ports. The next 3 bytes received are used as the configuration bytes for Ports B, C, and D, respectively.

25. **BD:** Write Port-B data (0BDh). This command takes the next received byte as data to write to Port-B.

26. **CD:** Write Port-C data. This command takes the next received byte as data to write to Port-C.

27. **DD:** Write Port-D data. This command takes the next received byte as data to write to Port-D.

28. **FB:** Read Port-B data. This command sends one byte read from Port-B.

29. **FC:** Read Port-C data. This command sends one byte read from Port-C.

30. **FD:** Read Port-D data. This command sends one byte read from Port-D.

31. **FA:** Read all digital I/O ports. This command returns 3 bytes—Port-B, Port-C, and Port-D.

32. **AD:** Write all digital I/O ports. This command accepts the next three bytes sent as data for Ports B, C, and D, respectively.

99. SPI write/read. This command requires a subsequent SPI flag byte and 1-7 data bytes. The SPI flag byte is detailed below. If the read bit is set, TNG-4 will return from 1 to 7 bytes, depending on the number of bytes sent. SPI linked-list chains are not supported.

SPI Flag Byte: = 0 if no SPI data follows; otherwise:

7	6	5	4	3	2	1	0
R/W	S2	S1	S0	CM	D2	D1	D0

R/W: This bit = 0 when data is output only.
This bit = 1 when reading data.

S2-S0: These bits specify which SPI enable line to use.
000 = Port C bit 2
001 = Port D bit 7
010 = Port D bit 6
011 = Port D bit 5
100 = Port D bit 4
101 = Port D bit 3
110 = Port D bit 2
111 = Port D bit 1

CM: This bit=ON causes a reinterpretation of the SPI flag byte. No SPI data is sent.

With CM=1 The SPI flag byte is interpreted as:

7	6	5	4	3	2	1	0
SMP	CKE	---	CKP	1	---	SSPM1	SSPM0

SMP = SSPSTAT:SMP bit for PIC with same meaning.
0 = Input sampled at end of output (default).
1 = Input sampled in middle of output bit.

CKE = SSPSTAT:CKE bit for PIC with same meaning.
0 = Data output on leading edge of clock.
1 = Data output on trailing edge of clock (default).

CKP = SSPCON:CKP bit for PIC with same meaning.
0 = Clock normally low (default).
1 = Clock normally high.

SSPM1 and SSPM0 set the clock rate.
00=FOSC/4 (1MHz, default)
01=FOSC/16 (250kHz).
10=FOSC/64 (62.5kHz).
11=undefined (don’t do it!).

This command redefines SPI operations until the next SPI configuration byte is received or TNG-4 is power-cycled.

D2-D0: These bits specify the number of data bytes to send/receive (1-6). When D2-D0 equals 7, the next byte sent will be interpreted as the number of subsequent bytes in the message. If this next byte is zero, no bytes are sent or received.

Examples:

Sending the hex bytes 99 1A will change the SPI protocol to an inverted clock at 62.5kHz with data changing on each clock's leading edge. No bytes are output.

Sending the hex bytes 99 48 will reset the SPI protocol to its default state.

Sending 99 85 80 42 00 00 00 sends 5 bytes using Port C bit 2 as the enable line. The first byte sent is 80. Five bytes are also read and retransmitted to the host computer.

Sending 99 97 09 80 42 00 00 00 00 00 00 00 sends and reads 9 bytes using Port D bit 7 as the enable line. The first byte transmitted is 80.

This protocol will support MAX3100/3110 operations if you do the initialization yourself, and send two bytes where the first byte is always 080h.