Port-Powered Frequency Counter
Model 232FC
Document No. 232FC2295

B&B Electronics Mfg. Co. Inc.
P.O. Box 1040 -- Ottawa, IL 61350
PH (815) 433-5100 -- FAX (815) 434-7094

Internet:
http://www.bb-elec.com
sales@bb-elec.com
techsupt@bb.elec.com

© 1994 B&B Electronics -- July 1994

***   Warning   ***

Important Usage Information

In order to prevent the 232FC's microcontroller from "locking up" and potentially damaging operation, always apply power to the 232FC **before** connecting the frequency source to be measured. Power can be applied either by raising the RTS and DTR handshakes, or with an external power supply.

NOTE: When using an external supply, the supply should be connected only to specifically labeled power inputs (power jack, terminal block, etc.). Connecting an external power supply to the handshake lines may damage the unit. Contact technical support for more information on connecting an external power supply to the handshake lines.

If you find that the microcontroller has locked up (won't respond to commands), remove the frequency source and power down the 232FC. Wait several seconds before re-applying power, then reconnect the frequency source.

Chapter 1. General Information

Introduction

B&B Electronics Mfg. Co. 232FC allows fr equency measurements to be made with any RS-232 port. The unit can be powered from the port handshake lines, making it a convenient, low-cost data acquisition tool.

FC.EXE is a software program provided with the 232FC that allows sampling, data logging, and many plotting features. The file FC.ASC is the manual for the software. It can be found on the diskette shipped with the 232FC.

Packing List

The following items should be shipped in the carton:

232FC Frequency Counter
User Manual
3.5" Floppy Disk

Contact the shipper immediately if any of the items above is missing or has damage.

Specifications

Communications:
RS-232, 9600 baud no parity, 8 data bits, 1 stop bit
Measurement Response Time:
One full period after the command is issued
Frequency Input Range: 6 Hz - 2 MHz

Maximum Input Voltage: 20 V

Signal Input Current: $V_{in} \leq 5V$ $I_{in} = 4~\mu A$

$V_{in} \geq 5V$ $I_{in} = (V_{in} - 5V) / 220~\Omega$

Power Requirements:
Port powered: 15 mA maximum @ 5V
External power: 40 mA maximum @ 125V

Chapter 2. Operation

Overview

The 232FC uses a counter to measure the pulse width of the input signal. The counter checks the state of the input signal approximately once every 1.3 μs. This sampling period is referred to as a "timer tick". The values returned by the 232FC are actually counts of how many timer ticks passed during each portion of the input signal. These values must be converted to "real time." Both the widths of the high and low states of the input signal are measured in units of timer ticks. Figure 1 is an illustration of an input signal.
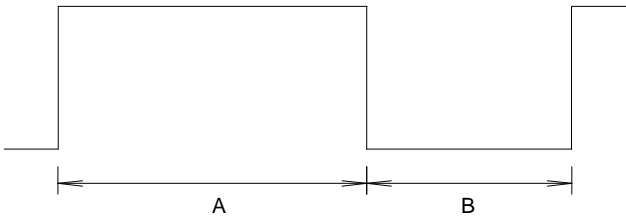

Figure 1

The 232FC uses four bytes to represent the signal. The first two bytes are the length of pulse A, measured in timer ticks. The second pair of bytes are the length of pulse B measured in timer ticks. All values are transmitted in hexadecimal format, lower byte first.

With this type of fixed resolution measurement, error increases with input frequency. In order to extend the maximum frequency range and decrease measurement error, the 232FC uses two modes of operation: Direct mode and Prescale mode.

Direct mode uses the 1.3 μs resolution to make frequency *and* duty cycle measurements at frequencies from 6 Hz - 50 KHz.

Prescale mode divides the input signal by a programmable divisor from 2 - 256. This greatly extends the frequency range, but duty cycle measurements are no longer possible as the signal is "squared up" by the divider. Measurements in this mode can be made from 24 Hz - 2 MHz.

Connecting the 232FC

When making connections to the 232FC, it is important to follow the connection procedure listed below.
1. Connect the RS-232 connector.
2. Power the 232FC, either with an external supply, or by raising both the RTS and DTR handshake lines.
3. Connect the frequency source.

If the frequency source is connected before the 232FC is powered, input current is drawn from the frequency source which may create enough voltage for the 232FC microcontroller to be marginally powered. This typically causes the microcontroller to appear "locked up" when power is applied. If this occurs, simply disconnect the frequency source, remove power from the 232FC, wait several seconds and follow the proper connection procedure.

Making Measurements

The 232FC uses single character commands to initiate all measurements. The commands are listed in Table 1. Before making measurements, follow the connection procedure listed in this chapter.

Table 1. 232FC Commands

Command	Mode	Divisor D	Min freq, Hz	5% error freq, Hz
$	Direct	1/2	6-	38,400
0	Prescale	2	23-	73,100
1	Prescale	4	46-	146,200
2	Prescale	8	94	292,500
3	Prescale	16	188	585,100
4	Prescale	32	375	1,173,000
5	Prescale	64	750	*2,340,500
6	Prescale	128	1500	*4,681,000
7	Prescale	256	3000	*3,930,000

* Note that the maximum frequency of the 232FC is approximately 2 MHz. Higher frequencies are distorted by the input capacitance of the unit and may not give valid readings.

To make a measurement, send the appropriate command for the input frequency range. The 232FC will reply with a five byte string. The response will be returned in after one period of the input signal has passed. The first byte is an echo of the command received. The next four bytes represent the pulse widths of the signal being measured. The format of the reply is as follows:

$	Command Echo
A-Lower	Lower byte of A in timer ticks
A-Upper	Upper byte of A in timer ticks
B-Lower	Lower byte of B in timer ticks
B-Upper	Upper byte of B in timer ticks

"A" represents the width of the high pulse and "B" represents the width of the low pulse. The values of A an B are measured in units of timer ticks of the 232FC microcontroller. To convert from timer ticks to time, both A and B must be multiplied by the constant C, where C = 1.30208 µs/timer tick.

To find the frequency of the input signal, use the following equation:

$$f = 2D / C (A + B)$$

where, A is length of pulse A in timer ticks
B is length of pulse B in timer ticks
C = 1.30208 µs / timer tick
D = the appropriate divisor, found in Table 1

Direct Mode Measurements

In direct mode, the input signal is presented directly to the microcontroller, which measures the high and low pulses to the nearest 1.3 µs. This mode allows frequency *and* duty cycle measurements to be made in the range from 6 Hz to 50 kHz. The following is an example of a direct mode measurement of a 1000Hz, 60% duty cycle signal.

Example: Direct Mode Measurement

1 KHz, 60% Duty Cycle

| 0.60 ms | 0.40 ms |
| A | B |
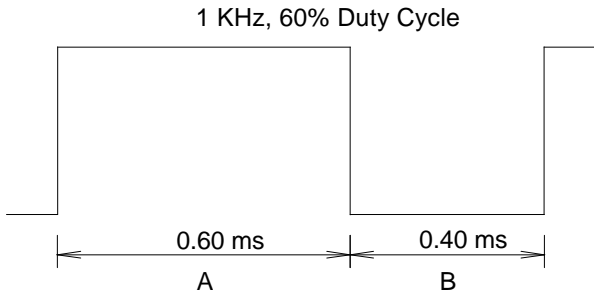
Figure 2

Command sent: $

Reply: $CD013301

Decoded reply:

$	CD	01	33	01
Command Echo	Lower Byte of Pulse A	Upper Byte of Pulse A	Lower Byte of Pulse B	Upper Byte of Pulse B

A = 01CDh = 461d timer ticks

B = 0133h = 307d timer ticks

C = 1.30208 μs/timer tick

D = 0.5 in Direct Mode (taken from Table 1)

$$f = 2D / C (A + B)$$

$$f = 2(0.5) / [1.30208 \times 10^{-6} (461 + 307)] = 1.0 \text{ kHz}$$

Duty Cycle = 100 A / (A + B) = 60%

Prescale Mode Measurements

In prescale mode, a programmable divider or prescaler is used to divide the input signal by a known divisor, from 2 to 256. This extends the frequency range of input signals, and reduces measurement error. When using the 232FC in prescaled mode, duty cycle measurements are no longer possible as the signal is "squared up" by the divider. The following is an example a prescale mode measurement of a 20 KHz signal.

Example: Prescale Mode Measurement

20 KHz Input Signal

50 us

Divide by 8 Prescaler Output

400 us	400 us
A	B
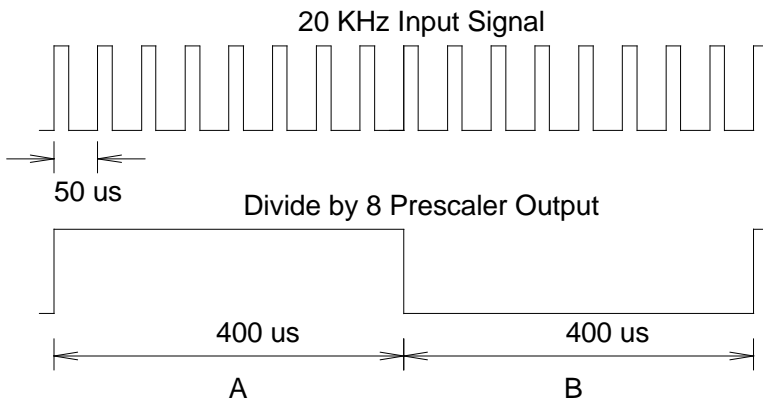
Figure 3

Command sent: 2
Reply: 233013301

Decoded reply:

2	33	01	33	01
command echo	LSB of pulse A	MSB of pulse A	LSB of pulse B	MSB of pulse B

Conversion:
      A = 0133h = 307d timer ticks
      B = 0133h = 307d timer ticks
      C = 1.30208 μs/timer tick
      D = 8 (Taken from Table 1)

$$f = 2D / C (A + B)$$

$$f = 2(8) / [1.30208 \times 10^{-6} (307 + 307)] = 20.0 \text{ kHz}$$

Measurement Error

The measurement error or uncertainty of the 232FC is based on the sampling rate, or timer tick interval. The input signal is sampled every 1.3 μs. The maximum error possible is then equal to the sampling rate, or 1.3 μs. This is true regardless of input frequency or measurement mode. As the input frequency increases, 1.3 μs becomes a more significant fragment of the desired signal. So while the measurement error remains constant, the percent error increases as the input frequency increases. Although this is true in both direct and prescale mode. Prescale mode offers a large advantage in that the input frequency is reduced by the divisor (2 - 256). The following example illustrates the improvement in accuracy using prescale mode.

Example: Error calculation

Input frequency, f = 20.0 KHz
Period, T = 1/f = 50.0 μs

Direct mode:
Error = 50.0 μs ± 1.30 μs
This gives a period measurement in the range:
48.70 μs < T < 51.30 μs
And a corresponding frequency measurement of:
19.49 KHz < f < 20.53 KHz
Error = 2.6%

Prescale mode, Divisor = 256:

Now the input frequency is 20 KHz / 256 = 78.125 Hz
Period, T = 1/f = 0.0128 s
Error = 0.0128 s ± 1.30 μs
Now we have a period measurement in the range:
0.0127961 s < T < 0.0127987 s
And a corresponding frequency measurement of:
78.133 < f < 78.149
Error = 0.0102%

Note that error is decreased by the value of the divisor. In this example, by a factor of 256. In order to minimize error, always select the highest divisor possible for the input frequency being measured.

Chapter 3. Programming Considerations

Sample Programs

Sample programs are provided with the 232FC demonstrating the operation of the unit in QuickBASIC, Pascal, and C.

Converting the 232FC measurements

The 232FC reply is sent in the form of four hexadecimal bytes, two bytes for each portion of the input signal. To convert this to meaningful information, each pair of bytes must be converted into one hexadecimal word. If you choose to work in hexadecimal, this can be done by multiplying the upper byte by 100h to shift it to the upper position. Or, if you prefer to work in decimal, multiply the decimal equivalent of the upper byte by 256. Add this result to the hex or decimal lower byte. This result is number of timer ticks in that portion of the input signal. The following is an example of the two-byte to word conversion.

232FC Reply: $8D107F10
Decoded reply:

$	8D	10	7F	10
Command Echo	Lower Byte of Pulse A	Upper Byte of Pulse A	Lower Byte of Pulse B	Upper Byte of Pulse B

Pulse "A" Conversion in decimal
Upper Byte: 10h = 16d
Lower Byte: 8Dh = 141d
16 x 256 = 4096
4096 + 141 = <u>4237 timer ticks in pulse "A"</u>

Pulse "B" Conversion in hex
Upper Byte: 10h x 100h = 1000h
1000h + 7Fh = <u>107Fh timer ticks in pulse "B"</u>

Out of Range Measurements

Measurements that are out of range of the 232FC can be identified by a response of FFFFFFFF following the command echo. This response indicates that the frequency being measured is too low for the range requested by the command. This response will also be seen when there is no signal connected to the 232FC.

Appendix A: 232FC Schematic