

## L5202/L5209 Link Module Frequency Counter Input:

### Hardware Description:

The L5202 and L5209 modules provide in hardware the capability to count events and measure frequency on channel 1 (terminal 1). Channel 2 provides a high speed interface to a module's CPU for direction sensing. The *hardware* characteristics of the frequency counter input are as follows:

- Single ended 0V to 5V input.
- Max. input frequency is designed to be 65.535kHz.
- Single direction counting on channel 1 (terminal 1) only. Counts occur on both rising and falling edges.
- Digital i/o (L5202) or input (L5209) channels 3 - 12 are available for ordinary logic functions.

### Software Description:

A default Link configuration is provided to support frequency counting. The configurable frequency counting parameters reside in the Handler2 function block. The software provides the following features:

- Calculation of channel 1 input frequency.
- Max. input frequency configurable from 14 Hz to 65,535 Hz.
- Pseudo-bidirectional counting with direction sensed at ~1ms intervals.
- Count direction may be based upon a logic level or a quadrature signal.
- A divide by N square wave is available for Link software function blocks. The frequency divider is configurable from 1 to 32500. A logic level specifying the direction of the signal is also provided for Link function blocks.
- The update period for frequency calculations is configurable from 8 ms to 35 ms.

## Application Recommendations:

Applications recommended for L5202/L5209 frequency counter:

- Speed reference input to a Link based control system.
- Input to a Link system for operator display parameters.
- Medium precision counting, *eg.* feet of product produced, etc.
- Speed feedback for systems with mechanical BW less than 400Hz.
- Low speed (less than 20 Hz) single event counting may be implemented on *any* digital i/o channel.

Applications *not recommended* for L5202/L5209 frequency counter:

- Absolute precision position counting.
- Position control.
- Very high performance speed control applications.

## Limitations of the Frequency Counter:

Several characteristics of the frequency counter must be evaluated when considering the L5202/L5209 frequency counter for an application. The first considerations are obviously the frequency range, accuracy, and electrical interface. The frequency range of the counter is adequate for most industrial applications. While providing high accuracy, the L5202/L5209 frequency counter has some unusual characteristics which affect accuracy for some applications. A discussion of the accuracy of the counter is provided in the last section of this document.

The *electrical interface* has a significant impact on the usefulness of the L5202 or L5209 in a given application. The inputs are single ended rather than differential. Practically, this severely limits the distance between the module and the electrical signal source (encoder or repeater). At present, no specification exists for the distance to signal source other than to recommend as short a distance as possible.

A single ended square wave of any voltage from 5V to 24V may be safely used with this frequency counter. However, to provide the maximum noise immunity, it is recommended that higher amplitude signals be attenuated to 5V with resistors. For the L5209, series resistors may be used. For the L5202, which has 6mA active pullups on each channel, a resistor between channel terminal and ground is also necessary.

$$R \approx 0.7 * (V_{max} - 5) \text{ k}\Omega \quad \text{Series attenuation resistor for L5209}$$

$$R_g \approx 5 / (6 + I_{in}) \text{ k}\Omega \quad \text{Ground shunt resistor for L5202.}$$

$$R_s \approx (V_{max} - 5) / I_{in} \text{ k}\Omega \quad \text{Series attenuation resistor for L5202.}$$

The active pullup channels of the L5202 may cause problems with low current signal sources. Oscillation of signal source drivers at the wrong frequency have been observed with low current CMOS drivers. The L5202 has been successfully tested with SSD's 5702/2 terminal repeater. Most encoders and repeaters designed for industrial environments should provide adequate current drive capability to interface with the L5202. The L5209 has no internal pullups, and therefore does not experience this problem.

One significant electrical interface factor affecting the use of the L5202 or L5209 as a frequency counter is power supply. The Link modules require +24V unregulated supplies, while many frequency sources require +12V or +5V supplies. Due to its single ended inputs, the Link module must be located near the frequency source. If the Link

module is located with an encoder, it is likely that two power supplies must be provided at that potentially remote location. Furthermore, it is unlikely that the remaining logic i/o channels will be useful at the remote position. Locating the Link module within an enclosure with drives or other Link modules is desirable, and power is likely to be available. Unfortunately, the enclosure is not necessarily located near enough to an encoder for single ended signal transmission. Thus the use of a signal converter is likely, adding to hardware cost.

A paramount issue with the L5202/L5209 frequency counter is that *the counter is not a true bi-directional counter*. Rather, the direction is sampled at ~1ms intervals. This characteristic has two potential effects. First, upon direction changes, the frequency value is defined to be zero. Any counts occurring within the 1ms interval of a direction change are lost. Although a divide by n signal is available to software for counting, the L5202 and L5209 *cannot be used to measure position precisely in applications where direction changes or repeated stopping occur*. Furthermore, *under no circumstances should the frequency counter be used in closed loop position control*. The frequency counter is very good at measuring frequency; it is not intended as a position counter. Note that applying the frequency counter to measuring length of run for a continuously produced web is acceptable.

Another potential effect of sampled direction sensing is aliasing. Aliasing is likely if the input frequency is modulated at or above 500Hz. In most drive applications, the mechanical bandwidth will not approach 500Hz. Therefore, this frequency counter may be safely used as a speed feedback device in many cases. In very high performance applications such as servo systems, however, aliasing is a real possibility. Use of this frequency counter in very high performance speed control applications is not recommended.

Other direction sensing issues should be considered when using quadrature signals. Dedicated quadrature direction sensing circuitry does not exist in the L5202/L5209. Due to limitations in the hardware, quadrature direction sensing may fail at very high frequencies with low slew rate inputs. The symptom is a markedly noisy frequency measurement at higher frequencies. Quadrature direction sensing has been tested successfully at frequencies beyond 66kHz with a 200ns rise time square wave. Problems have been observed above 42kHz with a 2 $\mu$ s rise time square wave. Use of the SSD 5702/2 terminal repeater unit is regarded as safe for quadrature signals up to 65,535 Hz.

The use of quadrature signals has another significant effect. If the quadrature signal is lost on channel 2, the frequency counter is unable to determine the correct direction. Because the software cannot continuously track the second phase of the signal, it cannot detect loss of the quadrature phase. The result is that the frequency measurement

degenerates into noise with variance rising with increasing signal frequency on channel 1.

## Software Configuration Notes:

The software function block supporting the frequency input has been designed to permit maximum flexibility. *It is possible for the user to create a hardware / software configuration which will be unable to operate due to cpu performance limitations.* This section describes how to avoid this problem.

The user may select the *frequency measurement update time*. This represents the real rate at which frequency samples are sent by the Handler2 (digital i/o) function block to other software function blocks. The first thing to note is that accumulating counts over a longer interval provides a larger count value, ie. more resolution. *Reducing the sample interval (speeding up sampling) reduces the resolution, and therefore accuracy, of frequency measurements.* For example, 65.535 kHz sampled at 35ms yields resolution of  $\pm 0.0218\%$ , while reducing the sample time to 10ms yields resolution of  $\pm 0.0763\%$ . (Accuracy and resolution will be discussed in the last section).

The second consideration when selecting sample time is possible *cpu overload*. Each software function block requires time to execute. In addition, each message sent between function blocks requires processing time. The cpu can handle only a finite number of function blocks and messages per second. Ordinarily, a configuration will contain a control loop, and messages are propagated around the loop only as fast as the cpu can manage. In these cases, the loop will run quickly or slowly depending upon the necessary cpu overhead. If for some reason, a configuration of function blocks (within a single Link module or a system) generates messages faster than an individual module's cpu can process them, then that module will cease to operate (crash).

The use of frequency sampling can potentially cause a Link module to crash. A loop configuration is inherently self-regulating, because messages which cause or "spawn" other messages are themselves spawned again only after the loop processing is complete. Frequency samples are not subject to this self regulating process: they occur at a fixed rate regardless of whether the cpu is keeping up. If the update rate is too fast, then the cpu will fall behind in processing until it finally crashes.

*Thus there are two incentives to configure long frequency sample times: high resolution and reduced likelihood of crashing a Link module.* Obviously, control loop performance will be enhanced by a shorter sample time, provided that the cpu can keep up.

The risk of crashing a module may be reduced in two ways: increase the frequency sample time and severely limit the number of destinations (and hence the spawned

messages) for the fast frequency output. For example, send the frequency sample directly to a control loop, but send it through a Sampler function block running at a slower rate to update an operator display.

The user may also select a *frequency divider value for the "divide by n" output* of the Handler2 (digital i/o) software function block. It is very easy to crash a module by failing to properly limit the value of the frequency divider (n). For example, specifying a divide by 1 output with a maximum input frequency of 50kHz will force the cpu to try to handle at least 100,000 messages per second (one per edge) in addition to its other processing. The module will certainly crash -- immediately. For this reason, simple guidelines are provided for choosing n:

**n > 20 \* max freq. (kHz)**      Absolute limit on n (~20ms per cycle).

**n > 50 \* max freq. (kHz)**      Recommended (~50ms per cycle).

It is the cpu speed which actually limits resolution when using counter function blocks in software to count high frequency events. For example, if a 30kHz input is expected from a 1000ppr encoder, the minimum value of n is 1500. This means that a counter function block will increment every 1.5 revolutions or 3000 encoder edges. Such a counter would be useful for recording meters or even feet of product on some drive systems, but the measurement resolution will not extend to many (if any) decimal places. *Decreasing the input signal frequency decreases the count resolution of software counter function blocks used with the divide by n feature.*

Note that if the real frequency of an input signal is less than 0.020 kHz (20 Hz), then a divide by one signal may be configured (n = 1). In this case, full resolution counting may be implemented easily. In fact, any of the ordinary input channels could be used in this case.

## Accuracy, Etc.

The algorithm used to compute frequency is as follows:

$$\text{frequency} = \frac{\text{actual freq.} * \text{scale}}{\text{full scale freq.}} = \frac{\text{actual counts} * \text{nominal time} * \text{scale}}{\text{actual time} * \text{full scale counts}}$$

Note that this calculation involves two division operations. The remainder when dividing the count measurement is retained. Thus no input counts are lost by the calculation itself. The remainder when dividing the nominal time is used to round the result.

The result is scaled. In the software, the scaling factor may be changed to alter what is considered full scale. Full scale frequency may be selected between 14 and 65,535 Hz. Some effects causing inaccuracy are independent of the maximum frequency selected, while others are not. This is discussed below.

Characteristics of the frequency calculation and measurement which are not considered in the accuracy specifications are as follows:

- Direction is sampled. When a direction change is encountered, the sampled count value is defined to be zero. Thus *counts may be lost on direction change*. Although this effect is not considered in the accuracy figure, *awareness of this effect is critical* when evaluating this frequency counter for applications
- Due to the sampling of direction, frequency modulation (*ie. rate of change*) must be limited to less than ~400Hz to prevent aliasing. Although this effect is not considered in the accuracy figure, *awareness of this effect is critical* when evaluating this frequency counter for applications.
- The measurement of a fixed frequency will *fluctuate* due to retention of the count remainder. This actually *improves* the accuracy of the frequency measurement averaged over time. In the absence of other sources of error, the *average* frequency measurement would actually be perfect over a long time. A maximum ±0.4% fluctuation in calculated frequency is expected at 65.535 kHz. This effect is not considered in the accuracy figure.

Characteristics of the frequency calculation and measurement which affect accuracy are as follows:

- Clock crystal tolerance is  $\pm 0.02778\%$  with a  $\pm 0.05556\%$  variation over full temperature range ( $0^{\circ}\text{C}$  to  $+50^{\circ}\text{C}$ ). Thus the crystal will introduce an extreme worst case inaccuracy of  $\pm 0.08334\%$  into the frequency calculation. This figure is independent of what is considered the full scale frequency.
- Maximum roundoff error at full scale for a single sample is  $\pm 0.00153\%$ . This figure is independent of what is considered the full scale frequency.
- The minimum frequency which can be measured is 14.3 Hz (one count per 35ms) yielding a maximum *resolution* of 0.022% at 65.535 kHz. The resolution is affected by the selected full scale frequency. The quantization error in percentage may be computed by

$$\frac{1}{2} * \frac{1}{\text{(sample interval)}} * \frac{1}{\text{(full scale frequency)}} * 100\%$$

where time is measured in milliseconds, and frequency in kHz.

The greatest source of inaccuracy in the algorithm is the resolution limit. The greatest source of inaccuracy overall is the clock crystal. Thus the *maximum error in a single frequency calculation* (assume 65.535 kHz sampled at 35ms) may be computed as follows:

	0.00153%	Max. roundoff error in one sample.
+	0.02180%	Resolution error.
+	0.02778%	Clock crystal tolerance.
+	0.05556%	Clock crystal variation over temp. range.
=	0.10667%	Max. error including worst case crystal variation.

---

Thus, at maximum 65.535 kHz sampled at 35 ms, frequency measurement accuracies are given as follows:

- *Excluding* clock crystal variation:  $\pm 0.023\%$
- *Including* clock crystal tolerance at fixed temp.:  $\pm 0.051\%$
- *Including worst case* clock crystal variation:  $\pm 0.107\%$

Using the figures and equations provided, frequency measurement accuracies at another maximum frequency, 30.0 kHz sampled at 35ms, are given as follows:

- *Excluding* clock crystal variation:  $\pm 0.049\%$
- *Including* clock crystal tolerance at fixed temp.:  $\pm 0.077\%$
- *Including worst case* clock crystal variation:  $\pm 0.132\%$

The variation of the clock crystal is not considered in many applications. This is because the clock crystal will introduce an *essentially constant error* in all frequency measurements, if it is assumed that the temperature variation is small or slow. In contrast, the roundoff and resolution errors vary randomly with each frequency measurement. It is important to note that if absolute frequency measurement is critical, or if it is desired to compare frequency measurements from two different Link modules, then the crystal variation must be considered. These considerations are necessary in any system in which frequency is measured digitally.

Theoretically, a fixed frequency could be established which would cause a constant roundoff error in measurement. In practice, however, there is jitter in the  $\sim 1$  ms scan time due to varying interrupt latency in the microprocessor, and the roundoff error is reasonably assumed to be a random variable with a zero mean.