Operation of the reciprocal frequency counter Pico-Fmeter2

The functions and routines for frequency measurement with the Pico-Fmeter2 can be found in the file "fm2-RP2040-3.c". The comments on the program sections explain some details in the execution. This description is intended to describe how the frequency measurement works in order to make the how and why clear.

Measuring method:

The frequency of a signal is defined as the number of oscillations per second. In the simplest case, the oscillations are counted in a counter for exactly one second and the result is directly obtained in Hertz (Hz).
In relation to a 10 MHz signal with a counting uncertainty of ± 1, counts of 9999999, 10000000 and 10000001 can result, which corresponds to a 7-digit resolution. At higher frequencies, the resolution increases, at lower frequencies it decreases and becomes significantly worse when only a few Hz have to be measured.
When measuring the 50 Hz mains frequency, even with a highly precise measurement time of 1 s, the result of 49, 50 or 51 with this measurement method is only accurate to two digits with 2%.
This is the "classic" frequency counter measurement method that was still common without the support of µCs.

If you need higher resolution at low frequencies, it is therefore necessary to no longer count the events/time, but to measure the time for a single oscillation (period) and convert this to the frequency by forming the reciprocal value.
Based on a typical reference frequency (Fref) of 10 MHz and a period (t) of 20 ms of the 50 Hz signal, the time measurement results in a count of 200,000. The frequency is obtained from the conversion: f = 1/t * Fref.
Since the resolution of the time measurement here is ± 1 too, the calculated frequency has a resolution of 5 valid digits. This is a significant improvement over the original 2 digit resolution measurement.

With the 1st measuring method you can measure very high frequencies with high resolution and with the 2nd very low ones. Even if you are willing to switch the measuring method depending on the input frequency, you only get a resolution of around 0.03% with both methods at approx. 3.162 kHz ($\sqrt{1} * 10e7$), which is a poor value.

Reciprocal frequency measurement:

In order to obtain a high resolution independent of the respective input frequency, both methods can be combined in such a way that on the one hand the previously required 1 s measurement time is used and on the other hand not only a single period of the signal but as many periods are counted until 1 s measurement duration has been reached or exceeded.

Two counters are now required for this, one of which counts the number of periods (N) of the signal and a second counter the exact time (T) of one or more periods. It is very important that the counter for the time measurement is started synchronously at the beginning of a period and stopped synchronously at the end of the last period. After a measurement duration of one second, the resolution of the measurement is now >= 1*10e7 and the result has 7 digits with the assumed 10 MHz as a reference for the time measurement. The number of periods counted is always an integer, which does not adversely affect the resolution.

If you use fast counters for the number of periods and the measuring time over all periods, the measuring range no longer has to be set.
The measurement result is given as: f = N/T * Fref.

In the simplest case, a measurement sequence can look something like this:

1. Reset the counters for N and T
2. Wait for the beginning of a period and let both counters count synchronously
3. Allow counter to count until >= 1 s measurement time has elapsed: T >= 1*10e7
4. Wait for the beginning of a period and stop both counters synchronously

The period that led to the measurement being stopped is still counted, but its period duration is no longer taken into account. This is correct because this counter was still set to '0' for the 1st period. The number of periods (N) and the total measurement time (T) therefore match and can be evaluated.
A new measurement with the same procedure is then started.

Now you can imagine an electronic circuit for the process described, which works with separate counters, gates and flip-flops. So far so good, there is a disadvantage in the process described: at least one period between two measurements is omitted when the meter readings are read out is not taken into account. This may not matter at higher frequencies, but even when adjusting a second pulse for a clock, there is only a new result every two seconds.

Since a microcontroller is required for frequency calculation, you can omit separate counters for frequency measurements down to the lower MHz range by cleverly using the already integrated µC counters. These counters usually have a 'capture' unit that can save the current counter reading synchronously with an external or internal event. You can use this function to record and evaluate all periods of a signal without gaps. Instead of resetting counters, the last counter reading for a 'capture' event is regarded as a '0' reference and the counter values of a new measurement are calculated from 'new value' - 'old value'. The last 'new value' becomes the 'old value' for the next measurement.
If you calculate with unsigned binary values (32 bit), overflows of the counters do not have to be taken into account. Once started, the counters simply continue to run.

So much for the general requirements for a reciprocal frequency measurement.

Reciprocal frequency measurement on the RP2040:

The RP2040 controller used on the Pico board is fast, has a powerful M0+ core (double even, of which only one is used here) with lots of memory but not a single counter with a 'capture' unit, which is mandatory for a reciprocal frequency measurement. This is bad at first!

However, the RP2040 has two fast micro-programmable PIO units that can be used to emulate reasonably fast counters with capture functionality. As previously described, a counter for period duration and a counter for number of periods are required.

Per PIO there are 4 x SM units (state machines), one of which is required here in order to obtain a counter with a capture function. Operating at maximum frequency, a SM unit detects positive and negative edges of the input signal, using a continuously incrementing x-register to obtain the exact times of the edge changes.
With each falling edge, the point in time is written to an output register for further processing and a time stamp is thus generated for each completed period. Optimally programmed, the x-register is increased with 1/4 of the clock frequency of the SM and the measured frequency can theoretically be 1/8 of the SM clock frequency with two edge changes per period. The practical value is just below that, so that, for example, about 15 MHz can be measured with a 133 MHz processor clock and 33.25 MHz of the SM unit.
These time stamps now provide the exact times for each individual input period.

A counter for the number of periods is required too. The RP2040 offers DMA units for this, one of which is required to continuously write the time stamp from the PIO output register to a fixed address in the RAM and thus make it available for subsequent evaluation. With each DMA transfer, the transfer counter of the DMA channel is decremented. This counter is 32 bits wide, is set once to the maximum value of 0xffffffff and returns the number of periods counted with the correct sign when comparing “old value” – “new value”. In order to read the current time stamp and the current value of the transfer counter, the DMA channel must be stopped. This is the only way to ensure that both values belong to each other.
If the DMA channel were stopped for too long, time stamps would be lost and the entire measurement result would become unusable.

The output register of a PIO-SM channel provides a FIFO memory with a depth of eight values. Delayed reading when the DMA channel is stopped therefore does not slow down the acquisition of new time stamps.
With a maximum input frequency of 15 MHz and 8-fold buffering, the DMA channel can therefore be stopped for a maximum time of 0.53 µs, which corresponds to 70 processor cycles at a clock frequency of 133 MHz. With <= 10 processor cycles, which are required for reading out the DMA transfer counter and time stamp from the RAM, there is enough security to read out the measured values without errors.
Up to an input frequency of 15 MHz, the exact number as well as the exact point in time are available for each input period.

A DMA channel stops automatically when its transfer counter is decremented to 0. With an interrupt, it is reset to the start value with a brief interruption. As previously described, this short stop is not a problem due to the FIFO buffering.

As already mentioned, the resolution of the time base is ¼ processor clock and, to stay with the numerical example above, at 1/33.25 MHz around 30 ns. If you want to get continously readings with three measurements/s, the resolution for each measurement is:
0.3 s / 3*10e-7 s = 1*10e7. Each measurement result thus provides 7 valid digits (or in other words 0.1 ppm), regardless of whether 1 Hz or 10 MHz are measured.

A PIO unit with its 4 x SM units in conjunction with 4 x DMA channels could simultaneously perform 4 independent frequency measurements. If higher input frequencies are required, a second measuring channel can be programmed, which is operated with a fixed prescaler and also delivers 7-digit results / 0.3 s.
This describes the reciprocal measuring function of the Pico-Fmeter.

Improved measurement resolution at higher input frequencies:

The description of the measuring method above makes it clear: when measuring low frequencies of, for example, 1 Hz, an increased resolution of the result is only possible through an increased resolution of the time measurement. This would require additional hardware, which is not provided for here and is therefore not explained further here.

A higher resolution is often desired or required for measuring a local frequency standard of 10 MHz. With a slightly increased input signal of 0.1 Hz at 10 MHz, measurements at intervals of 1 s deliver about 9 x 10.000000 MHz and then 1 x 10.000001 MHz. Both values are “correct” because they are within the ± 1 resolution.

As described above, at 10 MHz, 1*10e7 individual time stamps are available for each period within one second, but are not considered in the case of evaluation every second, since only the measuring points start value and end value are taken into account here.

With the linear regression, statistics offers a method to lay a straight line through the measuring points for many individual values, which does not have to correspond exactly to the start and end point of the individual points. The slope of the straight line also takes into account fluctuating ± 1 intermediate values and provides a higher resolution of the measurements by the factor $\sqrt{N}$where N is the number of measured values taken into account.

With 100 individual values, a factor of 10 results in one digit more resolution, with 10,000 already a factor of 100 corresponding to two digits. It makes sense to calculate individual intermediate measured values from the available time stamps and to evaluate them statistically. In order to include as many individual measured values as possible in the evaluation, a µC that is as fast as possible is required. The RP2040 is also a good choice in this regard.

In order to obtain as many support points as possible for the linear regression, the aim is to evaluate 100000 time stamps/s (10 µs grid). Below 100 kHz each time stamp is used and above 100 kHz the last interval of around 10 µs (synchronized with the time stamps).
While a measurement is running, the individual time stamps are prepared for the final evaluation. Additions and multiplications of the individual values must be carried out with high resolution in order not to allow overflows to occur in the calculations. 64-bit variables are used here, which require a certain amount of computing time and limit the evaluation rate to 100 kHz. The final measured value is calculated after the specified measuring time (typically 1 s) has elapsed.

At a signal frequency of 1 Hz, there is only a start and end value, as usual. It stays with the 7-digit resolution. At 10 MHz, with a measurement duration of 1 s, the resolution theoretically increases by a factor of 316 ($\sqrt{100000}$) to a 10-digit resolution:
1 / (33.25*1e6 * 316) = 0.95*1e-10
It will continue to be measured without gaps and each period of the input signal will be considered!

With this theoretical resolution one could show off perfectly, if there weren't some barriers set in practice. The PIO units within the RP2040 interfere slightly, which only becomes apparent at this high resolution.
This increased resolution only makes any sense with a highly accurate external 10 MHz reference signal. In addition, the internal evaluation of ext. Fref has to be switched off.
There are other limitations that are not discussed further here. It is therefore always advisable to ensure the resolution achieved by regarding long-term stability.

The regression calculation is permanently active on the Pico-Fmeter2.

Higher input frequencies:

So far, the basic function of the measurement has been described, which does not require a prescaler up to about 15 MHz. An additional prescaler must be used for higher input frequencies. The Pico-Fmeter2 uses a 4:1 divider consisting of 2 x D-FFs (74AUP2G80), which according to the data sheet works up to 600 MHz. In order to save additional hardware, the 4:1 divided signal is divided again by an internal counter (PWM channel) by 8:1, so that the total divider ratio is 32:1. With the prescaler switched on, the result must be scaled appropriately, resolution or accuracy are not reduced in the process!

The input frequency of a PWM channel used as a prescaler is: µC clock frequency / 2.
With a typical 133 MHz µC clock, it is around 63 MHz in real terms. With the upstream 4:1 divider, the maximum input frequency increases to around 250 MHz.

In the first program versions, the input frequency was measured with a PIO and 2 x SM units; directly on the one hand and that behind the prescaler on the other in parallel, and the result of the channel with prescaler was displayed at higher input frequency. The advantage was no switching delay in the display. With a 7-digit resolution, this method can be used without any problems.

After the addition of the regression calculation, disturbances became visible in the now higher-resolution results, which meant that only one SM unit could be used and the input pin of the SM unit was switched accordingly for the evaluation of the predivided one. Switching to the channel with a prescaler takes place from >= 8 MHz and switching back at <= 4 MHz. These frequencies were selected so high that an input frequency of > 4 MHz with an active prescaler before the regression calculation can still deliver ~100,000 time stamps and thus maximum resolution.

In order to achieve a fast switching time when the input frequency is increased to the upper MHz range, a classic rough frequency count (see explanation above) with a fixed gate time of 1 ms is carried out parallel to the normal measurement and is permanently monitored.

There are two ways of dealing with the measured value when the prescaler is switched over. You can either discard the last result completely and start a new measurement or you can also discard a few of these 100 kHz time stamps in the permanently active regression calculation in order not to reduce the measuring rate when switching. The latter method is set on the Pico-Fmeter2. If the frequency changes very slowly around the switching threshold, a disturbance can be observed in the 8th – 9th digit with a high resolution setting. If, on the other hand, the complete measurement is discarded, there is a delay for a new measured value, which is completely free of interference.
.

even faster:
To measure even higher input frequencies, the processor clock can be doubled as an option: no malfunction was observed.
This option is tested at power on.
If required, it can be used, otherwise the working frequency is left at the 133 MHz specified in the data sheet.