

Samplingrates User Guide

Klaus von der Heide, DJ5HG

1. Introduction

Sampling an analog signal means to represent the continuous signal function by a set of values. The process of sampling is explained in figure 1.

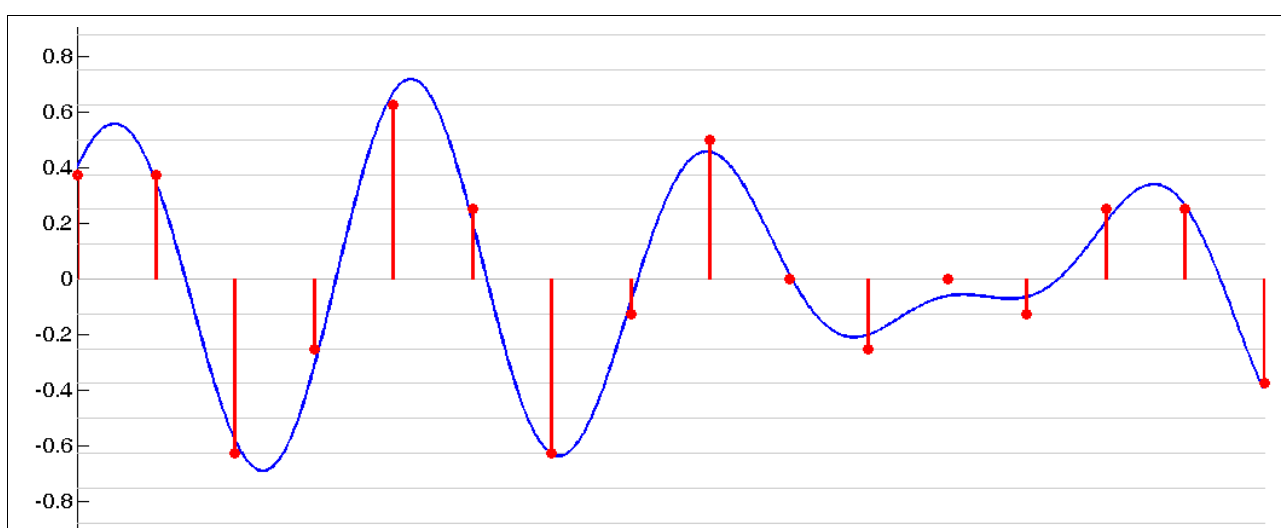


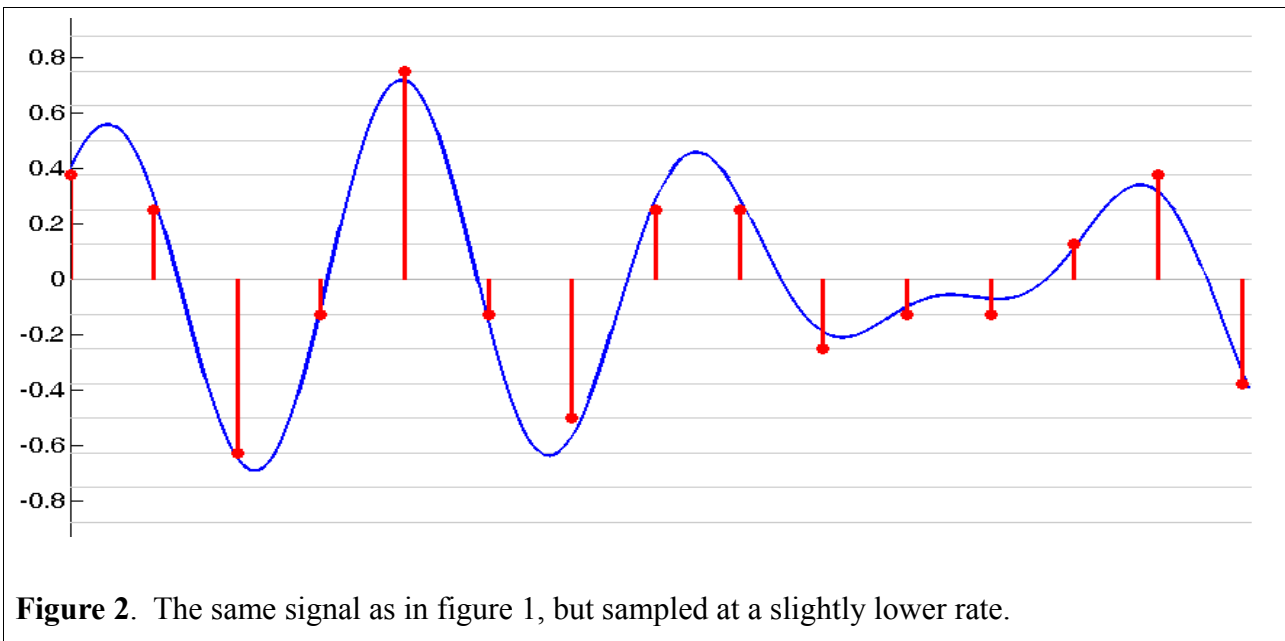
Figure 1. A signal is a real (or complex) function over continuous time. Sampling a signal means to discretize the time axis and to discretize the function values (red dots in the figure). The number of values taken per second is the *samplerate*. If the samplerate is larger than the highest frequency component of a complex signal or larger than twice the highest frequency component of a real signal, then the analog signal can be reconstructed from the digital values without loss.

If the number of possible function values (vertically) is n , then the values can be represented by binary numbers of $\log_2(n)$ bits. The digitization of the values is a compression. Usually the noise-floor of signals lies much above the compression loss of a digitization into 24 bits, but sometimes 16 bits are not enough (the example uses 16 values represented by binary numbers 0000 ... 1111).

Usually the samplerates 8000, 9600, 11025, 16000, 22050, 32000, 44100, 48000, and 96000 are supported by sound cards. Some of these often are not originally sampled at the nominal value, but computed from samples taken at a different rate. The consequences are more or less inaccurate rates. The resampling is done by the computer system, not in the sound card. Therefore the same external sound card can result in very different sample rates if used on different computers. An example is presented below where a cheap USB-soundstick works very well under Win7, but with WinXP the standard rate 8000 is wrong by more than 1%, which is unusable (the sampling clock then is wrong by more than 18 minutes per day).

2. Consequences of Differing Clocks

Figure 2 shows the same signal of figure 1 sampled at a slightly lower rate (15 samples instead of 16 within the time interval shown). The problem is not the lower rate, it is the fact that the digital signal processing application usually assumes a correct samplerate, i.e. it assumes that the spacing of the samples in time is that of figure 1. Therefore, the DSP-application sees the signal virtually squeezed or stretched in time. Since a frequency is the number of waves per unit of time, all frequencies are seen higher or lower, respectively.



If a signal is generated by a DSP-process for the virtual samplerate f_s , but the output samplerate is slightly higher, then the analog signal at the output of the digital-to-analog converter is squeezed in time. This has two consequences: (1) The frequency is higher, and (2) the duration of the signal is shorter.

Instruments in an orchestra must be tuned to an accuracy of about 0.1%. This accuracy therefore is a demand for the samplerates of soundcards in audio applications. Unfortunately, there exist combinations of sound cards and computer systems, which are worse by more than a factor 10.

If a sound card is used as the interface between the analog part and the digital part of the processing in a communication system (FSK441, JT65, PSK2k, etc.) then the inaccuracy of the frequencies due to differing samplerates often is tolerable. But squeezing or stretching the received signal in time also squeezes or stretches the digital symbols which may be critical. The demands on the accuracy of samplerates are as follows:

FSK441

A frame for example is **DJ5HG SM2CEW 27 27**, which with all spaces is 19 characters long. Each character is sent by a sequence of three tones. So the frame consists of 57 symbols. A time error of half a symbol over the entire frame would be the acceptable limit. So a relative error of $1/(2 \cdot 57) = 0.0088$ is tolerable.

JT65

A JT65-frame has 126 symbols. So a relative error of $1/(2 \cdot 126) = 0.004$ is tolerable.

PSK2k

A PSK2k-frame has 258 symbols. So a relative error of $1/(2*258) = 0.0019$ is tolerable.

Weak-signal applications demand for low-rate encoding, i.e. large frames. Therefore in future new communication modes are to be expected with even lower tolerance.

While a correction of inaccurate samplerates in music applications by resampling raises an enormous computational effort, it is very easy in applications of communication. PSK2k for example simply deletes samples in an equidistant time grid (or doubles samples) if the rate is too large (or too low), if the ratefactor is known. It is the purpose of `samplingrates` to determine the ratefactors of the input sampling and the output sampling.

3. The Program `samplingrates`

The user interface of `samplingrates` as shown in figure 3 has the following entries:

Clock Error: The error of the PC-clock in seconds per day. If this error is known, it should be entered here. In that case the measured samplerates are relative to UTC, otherwise relative to the PC-clock.

Input-ID: You can select the input device in a pull-down menu. The IDs are not the same as in other programs.

Output-ID: You can select the output device in a pull-down menu. To check the output channel you can use earphones at the selected output. After pushing the START button a continuous tone is outputted here. Be careful with the volume!

Samplerate: The output-samplerate to be measured.

Measurement Mode: Please select between two modes of generating the output sine wave, if you additionally want to measure it's frequency. If an *external frequency counter* is used to determine the frequency of the output signal, then the preset of the output frequency is 1000 Hz if the output samplerate is less than 32000, and 10000 Hz otherwise. If your choice is *harmonic*, then you can directly enter the frequency of a frequency normal, 60000 for example (the MSF). In this case an appropriate subharmonic is generated. You should distort the output sinewave to generate the harmonics and then feed this signal to your antenna. You now can compare both signals with a longwave receiver.

Operation: After having set all entries appropriately (the default settings are good for a first try) you can push the START button to start the sampling process. For a first try you should push the STOP button after about 10 seconds to check for possible errors. After this check you can start the measurement. It should last about 15 minutes or even much longer if higher precision is desired.

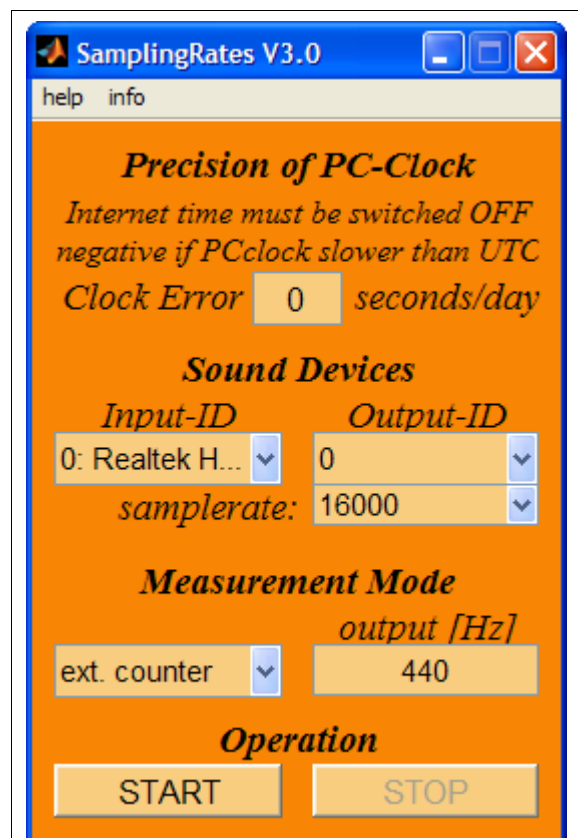


Figure 3. The Graphical User Interface of `samplingrates`.

4. The Output

The determined sample rates are printed into the black window. Additionally, there is a graphical output, which draws the differences between computer clock and sampling clocks as a function of time. Some results are presented in the following figures 4 - 8.

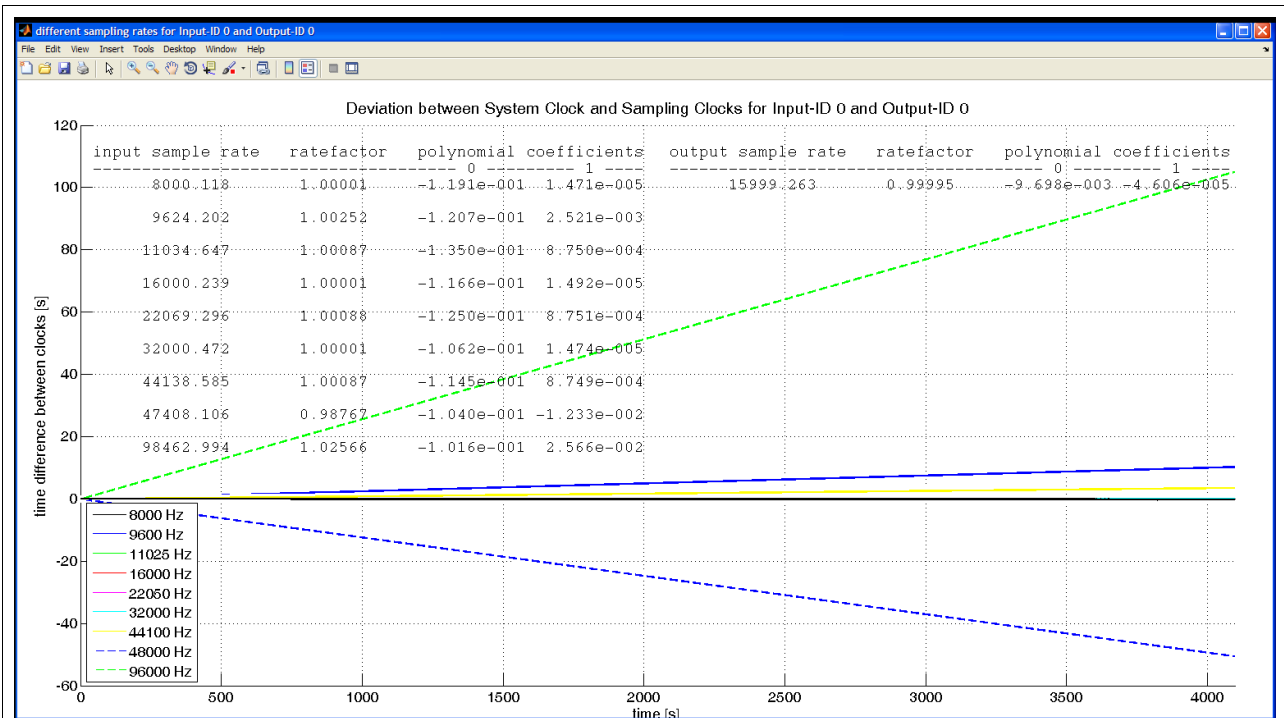


Figure 4. Result of an old integrated Intel audio device under WinXP. The rates 9600, 48000, and 96000 deviate significantly from the nominal values. All others are good.

What really is needed is the error of the samplerate clock compared to UTC. But the program compares the sample clock with the PC-clock. Therefore, it is recommended to check the PC-clock. The simplest way to do this is as follows:

1. Deactivate any synchronization with internet time services.
2. Determine the difference between the PC-clock and a precise UTC-clock ($dc1 = PCclock - UTC$)
3. Do the same as in 2. after a long time dt ($dc2 = PCclock - UTC$)
4. The error of the PC-clock in seconds per day is $86400 * (dc2 - dc1) / dt$, where $dc1$, $dc2$, dt should all be measured in seconds.

If the desired accuracy of the samplerate measurement is 0.1% and if the determination of the values $dc1$ and $dc2$ is precise to about 1 second, then a duration dt of 1800 seconds (ca. half an hour) suffices. For higher precision use a much larger dt . Usually the PC-clock is sufficiently precise to stay at the default value 0 for the PC-clock error.

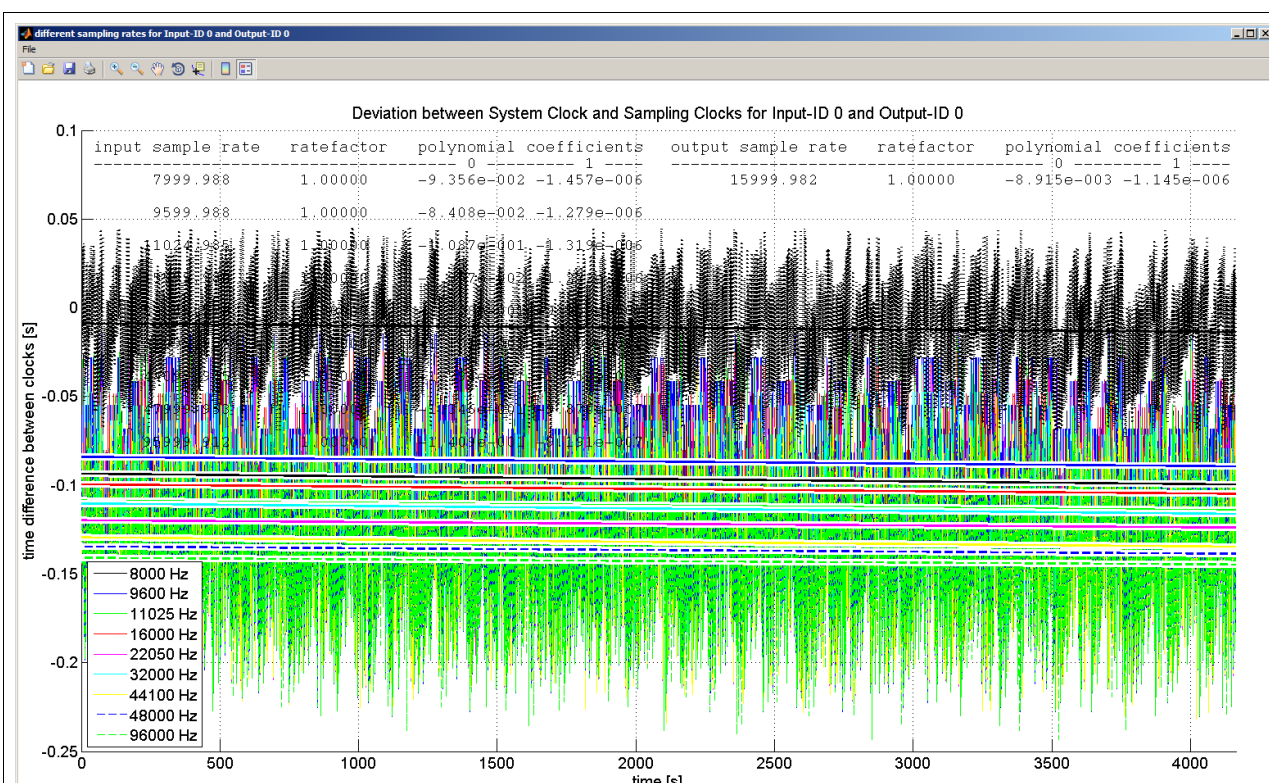


Figure 5. Result of an integrated audio device on a modern notebook under Win7. All sample rates are perfect (ratefactor 1.00000). Note that the tilt of the horizontal lines shows a drift between the sampling clocks and the PC-clock by only about 2 milliseconds per hour. The very noisy measurements here are only a result of the vertical scale of -0.25 ... +0.10 s compared to -60...+120 s in figure 4. The noise is the result of the inaccuracy of asking the system for the actual time in a running program, while other programs and the system itself are active.

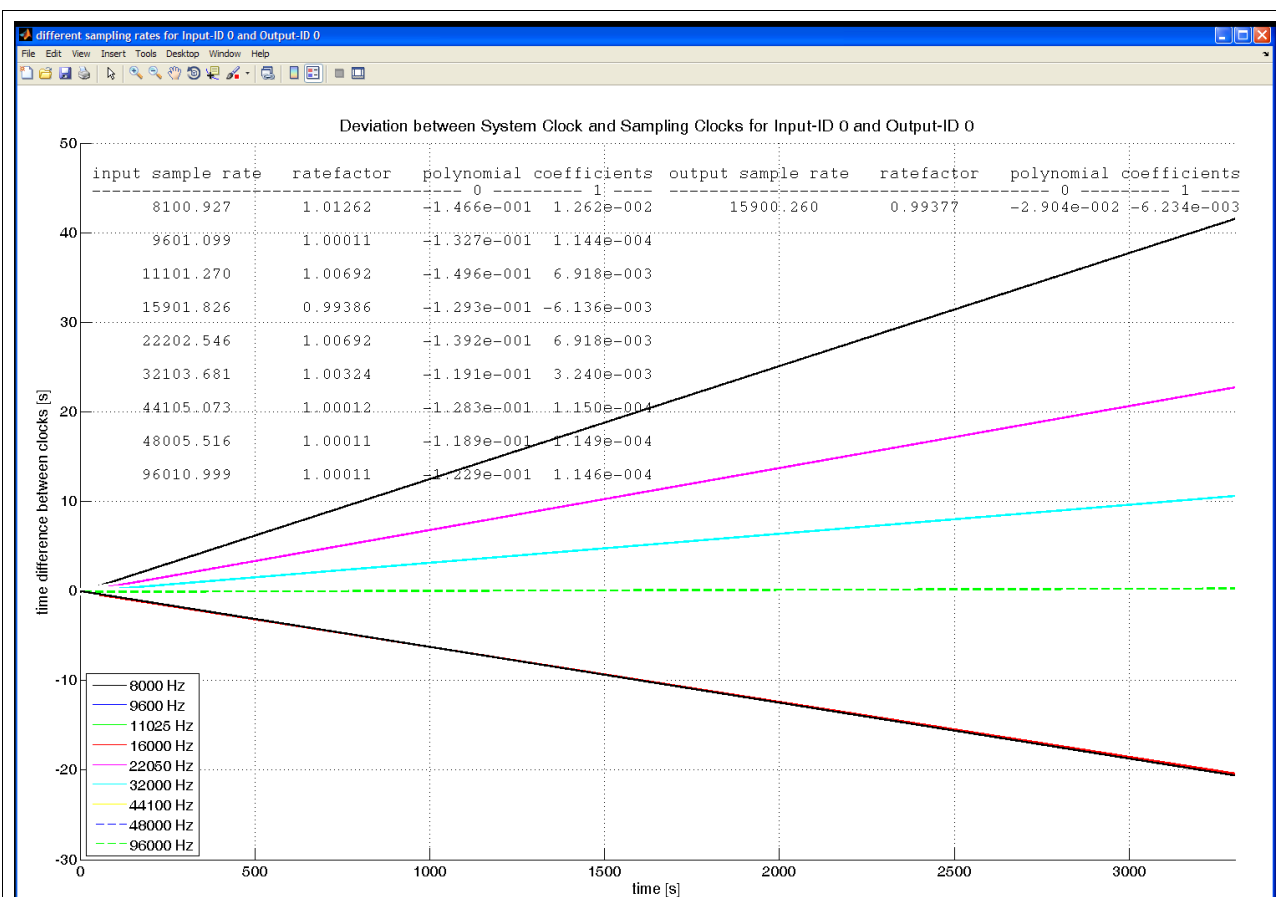


Figure 6. Result of a cheap external USB-sound card stick under WinXP. The rates for the nominal values 9600, 44100, 48000, and 96000 are good, 11025, 22050, and 32000 are bad, and 8000 and 16000 are unusable. Note that the difference between PC-clock and sampling clock for the rate 8000 is more than 40 seconds after less than an hour! This clock is worse compared to any old sandglass. The same soundcard stick used at the Win7-notebook used for figure 5 gives results nearly as perfect as those of figure 5.

5. Sampling Problems

If there are broken lines in the output diagram with breaks larger than 100 milliseconds then the computed samplerates are wrong, because the linear fit into the output data does not correspond to the linearly diverging clocks. In this case do the following:

1. Click the +zoom-button in the menu of the figure.
2. Select the largest area in the figure without breaks using the mouse.
3. Again click the +zoom-button to disable the zoom function.
4. Hit the key c or C on the keyboard (while the figure is in the foreground).

Now a new figure is drawn with the restricted time scale. The computed samplerates now are ok.

Figure 7 shows an example with cracked lines. The procedure 1. ... 4. yields the result of figure 8.

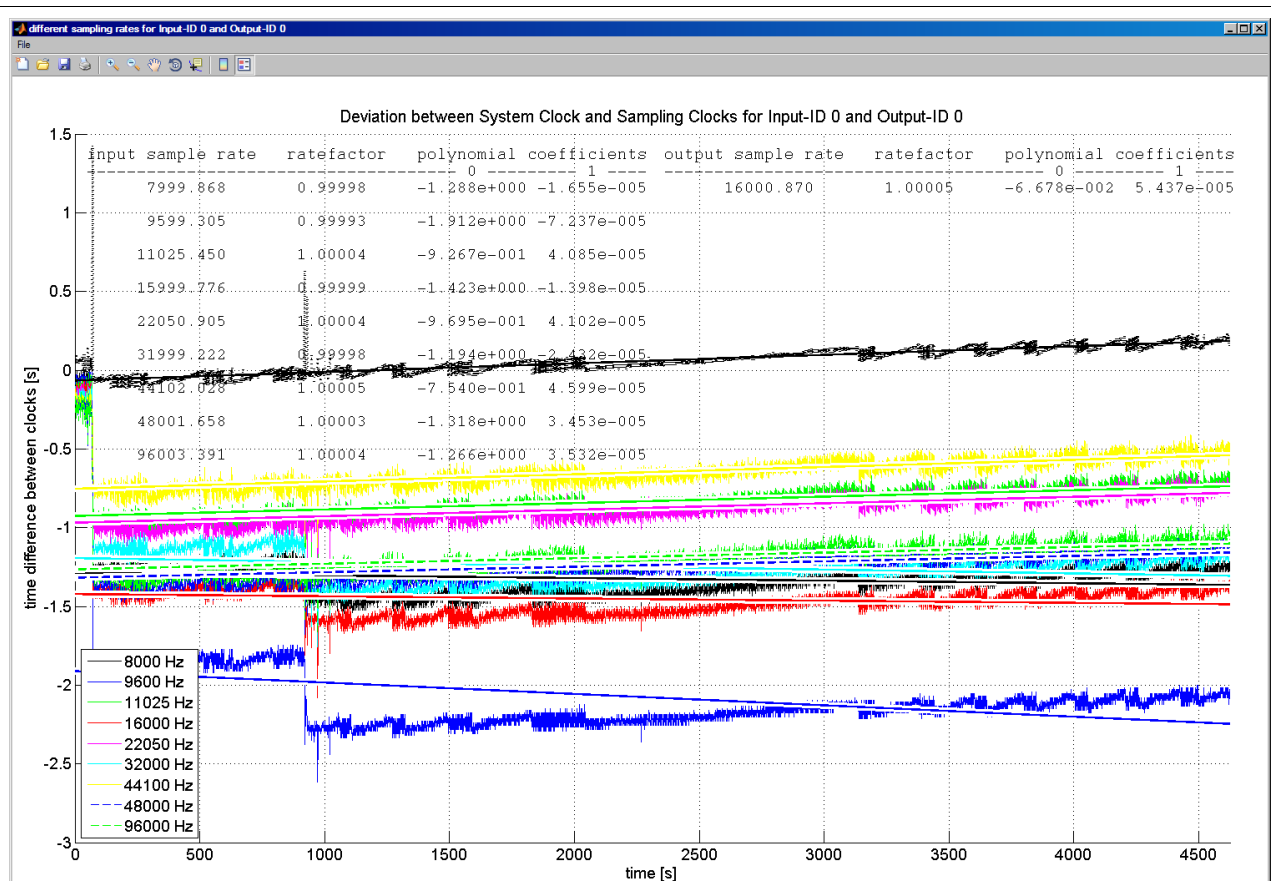


Figure 7. Cracks of the lines of more than a hundred milliseconds may be caused by different reasons. In this case the synchronization of the PC-clock with internet time was not deactivated.

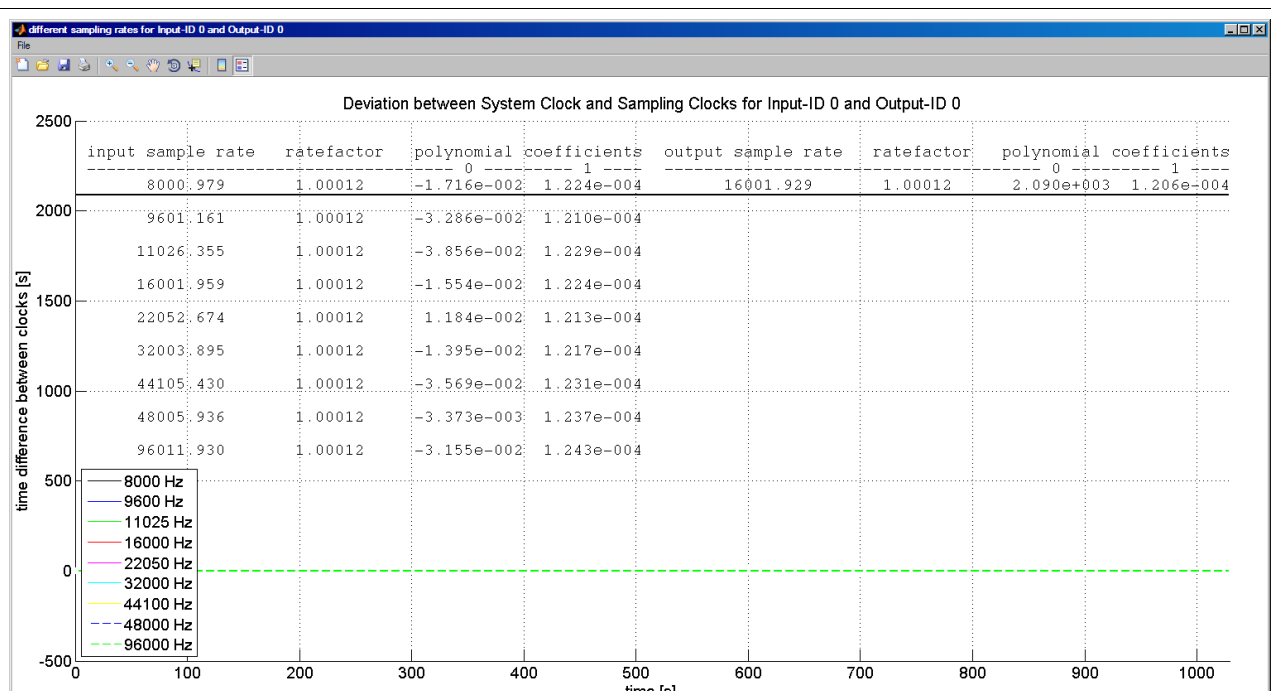


Figure 8. Following the described zooming-procedure the region between 2100 and 3100 s has been zoomed here resulting in perfect samplersates with identical ratefactor at all supported rates.