CERN-ACC-2015-0136

Marc.Magrans.De.Abril@cern.ch

The Phase-locked loop Algorithm of the Function Generation Controller

Marc Magrans De Abril, Quentin King, Raul Murillo-Garcia, CERN, Geneva, Switzerland,

Keywords:

Abstract

This paper describes the phase-locked loop algorithms that are used by the real-time power converter controllers at CERN. The algorithms allow the recovery of the machine time and events received by an embedded controller through WorldFIP or Ethernet-based fieldbuses. During normal operation, the algorithm provides less than 10 _s of time precision and 0.5 _s of clock jitter for the WorldFIP case, and less than 2.5 _s of time precision and 40 ns of clock jitter for the Ethernet case.

Presented at ICALEPCS, 17-23 October 2015, Melbourne, Australia

Geneva, Switzerland
October, 2015

CERN-ACC-2015-0136
03/11/2015

THE PHASE-LOCKED LOOP ALGORITHM OF THE FUNCTION GENERATION/CONTROLLER

M. Magrans de Abril, Q. King, R. Murillo Garcia, CERN, Geneva, Switzerland

Abstract

This paper describes the phase-locked loop algorithms that are used by the real-time power converter controllers at CERN. The algorithms allow the recovery of the machine time and events received by an embedded controller through WorldFIP or Ethernet-based fieldbuses. During normal operation, the algorithm provides less than 10 μs of time precision and 0.5 μs of clock jitter for the WorldFIP case, and less than 2.5 μs of time precision and 40 ns of clock jitter for the Ethernet case.

INTRODUCTION

It is widely known that the existence of measurement and computation delays and jitter affect the stability and performance of the control algorithms [1–3]. It is also well known that as the control system becomes larger (i.e. spatial extension or number of jointly controlled elements), the complexity of the control algorithm increases and its performance and stability degrades [4]. This is precisely the situation of the control of power converters at CERN. For example, the LHC project required the joint control of over 1700 converters across the 27 km of underground tunnels with a tracking error of less than five part per million (ppm). The large spatial extension requires a correspondingly large communication network to transport the information, which increases the delay, jitter, and packet loss. If there are different subsystems to be jointly controlled (e.g. main dipole and quadrupole magnets, simultaneous setting of the real-time orbit corrections, synchronisation of injection and ejection converters, etc.), then the control algorithm should also consider the relative delays between them.

In order to minimise the above effects, the devices that require real-time synchronisation at CERN used to employ a special purpose communication network that transports timing information with small jitter and known delay [5, 6]. The information necessary to control a device is therefore sent using two different networks. An Ethernet network (known as the technical network) transports the soft real-time data. A second network (known as timing network) transports the hard real-time data containing such things as the machine time, accelerator state and events.

Since the year 2000, the power converter controllers known as Function Generation Controllers (FGC) have deviated from this general architecture [7]. As shown in Fig. 1 the FGC uses the fieldbus as the mechanism to receive both soft and hard real-time data from the general purpose Linux computer (known as the front-end). That is, the control and the timing networks converge on a front-end, instead of being connected directly to the embedded controller. This change reduces the installation cost and the number of connection related incidents. In the FGC case, further reduction of development, installation, and maintenance costs have been achieved by using commercial off-the-shelf components and protocols for the fieldbuses. That is, WorldFIP on the FGC version 2 (FGC2), and FGC_Ether on the FGC version 3 (FGC3) [8].

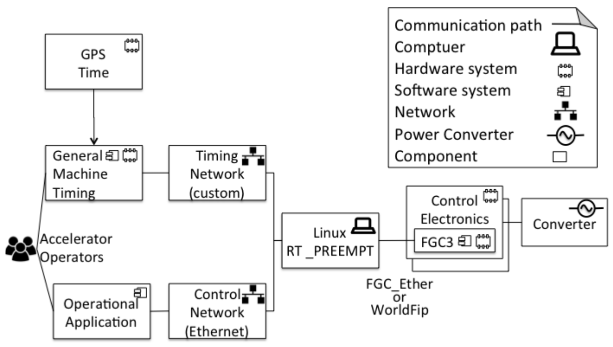

Figure 1: Transport of timing and control information from the GPS signal and the particle accelerator operator to the power converter using one fieldbus to transport timing, control, and monitoring information.

Removing the direct connection between the timing network and the FGC implies that the UTC time and its phase have to be recovered. The time with 20 ms precision is recovered by a periodic broadcast signal from the general purpose computer to the FGC, while the 20 millisecond phase is recovered using a Phase-Locked Loop (PLL) running on the FGC. This paper describes the requirements, design, and performance of the PLL algorithms used for both the FGC2 and FGC3.

SYNCHRONISATION OF THE LHC CONVERTERS

Requirements and Design

Between the year 2000 and 2003 the FGC2 was developed for the LHC. On one hand, the correct operation of the LHC required a radiation tolerant fieldbus and the synchronised application of the converter current across the accelerator with an accuracy of less than 5 ppm. This requires a time precision of 1 ms to achieve the synchronised operation of the converters [9]. It also required a jitter of less than 10 μs to have less than 1 ppm jitter induced noise from the sigma-delta ADC. On the other hand, it was not of primary concern the control and monitoring bandwidth between the operator and the controller because the circuits and the ramps in current are very slow. These requirements were fulfilled by a 2.5 Mbit/s WorldFIP fieldbus cycling at 50 Hz. Given the low jitter of this real-time fieldbus (i.e. $\sigma_j \approx 0.5\mu$s), it has been possible to reconstruct the 20 ms phase with respect to the timing network using a periodic broadcast packet containing the UTC time and the accelerator events.

Figure 2 shows the block diagram of the software PLL algorithm. The implementation relies on a free running 16-bit 2 MHz counter provided by the Motorola M68HC16 CPU. First, the fieldbus synchronisation signal $f(t)$ latches the free running counter $c[m]$, and generates the sync reference $r[n]$. Every 20 ms the reference is compared against the expected sync time $y[n]$ to compute the phase error $e[n]$. The error is then used to adjust the expected millisecond period in terms of the CPU 2 MHz tics $q[n]$ using a Proportional Integral (PI) controller with gain scheduling (the PI is the simplest controller that can follow the phase ramp generated by the fieldbus time signal). The millisecond Interrupt Service Routine (ISR) uses the current value of the free running counter $c[m]$ and the expected millisecond duration to generate the next millisecond interrupt $m(t)$. As the CPU counter comparator only handles integers, the fractional part $q*[n] - floor(q*[n])$ is carried over to the next iteration to minimise the quantisation error.

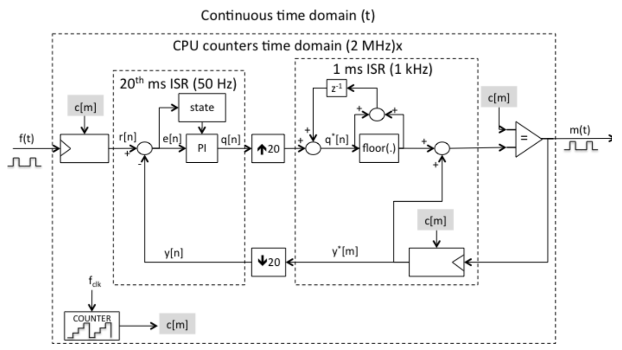

Figure 2: FGC2 PLL Algorithm.

Model

Figure 2 can be analysed using a linear module using the following assumptions: the output counter generating the millisecond interrupt can handle real numbers, the quantisation noise is much smaller than the fieldbus jitter, and finally, the FGC2 clock drift rate is small compared to the fieldbus jitter. In this case, the z-transform model of the PLL is:

$$\begin{aligned} E(z) &= \frac{1}{1+L(z)}R(z) \\ Y(z) &= \frac{L(z)}{1+L(z)}R(z) \\ L(z) &= \frac{20(k_i+k_p)}{(z-1)^2}\left(z-\frac{k_p}{k_i+k_p}\right) \end{aligned} \tag{1}$$

Where $L(z)$ is the open loop transfer function, k_i and k_p are the integral and proportional gains of the PI controller, $E(z)$ is the z-transform of the phase error, $R(z)$ is the reference input (i.e. a ramp plus white noise representing the jitter), and finally, $Y(z)$ is the z-transform of the expected arrival time.

The relation between the input jitter and the phase error is straightforward [10]:

$$\begin{aligned} S_{ee}(f) &= |H_{je}(e^{j2\pi f})|^2 S_{jj}(f) \\ &= |\frac{1}{1+L(e^{j2\pi f})}|^2 \sigma_j^2 \\ \sigma_e^2 &= \sigma_j^2 \int_0^1 |\frac{1}{1+L(e^{j2\pi f})}|^2 df \end{aligned} \tag{2}$$

Where S_{ee} and S_{jj} are the power spectrum of the jitter and phase error in the discrete frequency domain, $H_{je}(z)$ is equal to the z-transform transfer function between the reference and the phase error, and finally, σ_e and σ_j are the respective phase error and jitter standard deviations measured in 2 MHz tics.

Note that the diagram in Fig. 2 obviates the propagation of the fieldbus boradcast packet. The PLL design assumes a fixed propagation delay of 450 μs. Therefore, given that the maximum difference in cable length across the LHC is less than 2 km, the time accuracy should be better than 10 μs.

Stability and Performance

There are three basic criteria that should be met in the design of a PLL: stability, lock time, and jitter rejection. A PLL is stable if the magnitude of the closed loop poles is less than 1. As shown in Fig. 3 the proportional and integral gains should be well below 1/10. The lock time is determined by the decay of the transient response, which in turn is determined by the magnitude of the open loop poles. That is, smaller pole magnitude implies faster lock time and also a more stable PLL. Finally, jitter rejection can be calculated using eq. 2. Figure 4 shows the logarithm of the jitter rejection as a function of the proportional and integral gains. A minimum jitter requires the smallest possible integral gain, and a proportional gain about 0.005. So, stability and jitter rejection are at odds with each other. In order to bring together both requirements and have a lock time below 10 seconds, a gain scheduling algorithm was implemented. The PLL state machine is composed of three states:

- *Fast Slew* ($|e[n]| > 1$ ms). Instead of a PI algorithm the millisecond duration is fixed to $q[n] = 2100$ ticks (1.05 ms). This state allows the phase error to reach a close initial condition in less than half a second without the risk of overrunning the millisecond ISR.
- *Capture* ($50\mu s > |e[n]| \geq 1$ ms). A PI algorithm is used with $k_i = 2^{-12}$ and $k_p = 2^{-8}$.
- *Lock* ($|e[n]| < 50\mu s$ for more than 1.5 seconds). A PI algorithm was used with $k_i = 2^{-16}$ and $k_p = 2^{-9}$.

Simulation and Measurements

Figure 5 shows the simulated and measured phase errors in μs after a power cycle of the FGC2. The picture shows three differentiated phases corresponding to the three different states of the PLL. Note that the simulation shows a longer period in the Fast Slew state due to a difference in the initial values of the free running counters. Note that the PLL locks

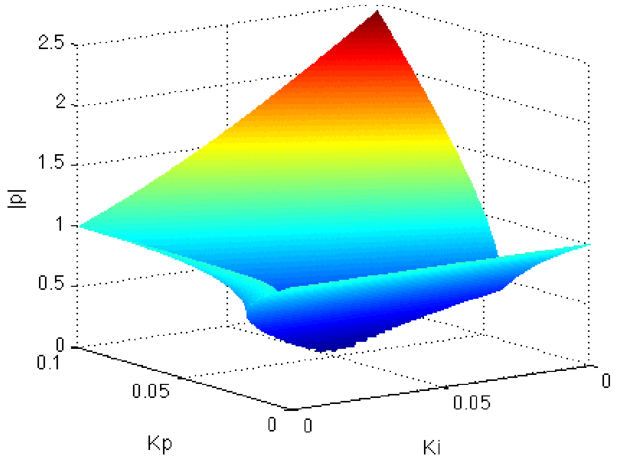

Figure 3: Magnitude of the closed loop poles of the FGC2 PLL algorithm.

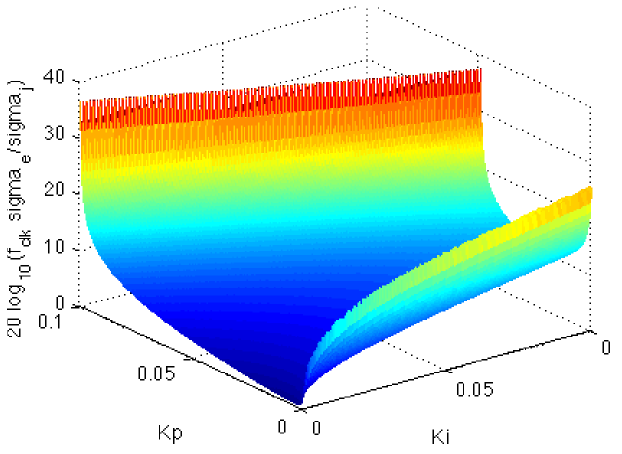

Figure 4: Jitter rejection logarithm of the FGC2 PLL algorithm.

in less than 3 seconds with a phase error jitter with $\sigma_e = 0.5\mu$s.

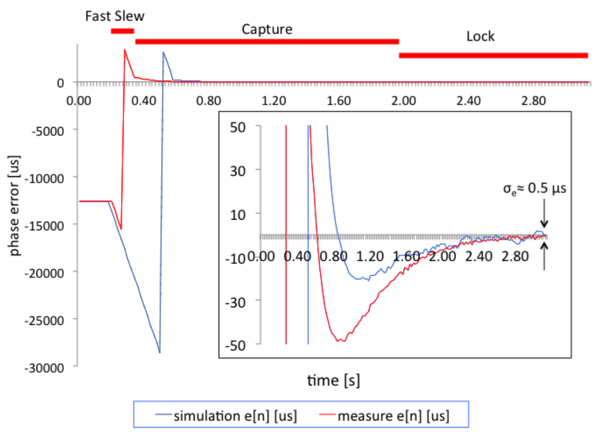

Figure 5: FGC2 PLL phase error after a power cycle.

SYNCHRONISATION OF THE INJECTOR CONVERTERS

Requirements and Design

The FGC3 controller was developed between the years 2007 and 2011 for the Proton Synchroton Booster (PSB) and Linac 4 accelerators. The FGC3 will be also used for the new projects at ISOLDE, AD, ELENA, and the renovation of the power converter controls across the LHC injection chain. During the early phases of the project it was clearly appreciated that the bandwidth and timing requirements of the injector chain were more demanding than the ones for the LHC [7]. A time precision of 1 μs and a jitter of 100 ns was required.This allowed a higher regulation regulation frequency (10 kHz), the short (1-10 ms) and high precision ramps (100-1000 ppm) on some of the injector magnets, the reliable propagation of timing information from the FGC3 to the rest of the converter control electronics (e.g. voltage control loop), and finally, the synchronised setting of the current between converters.

For this purpose, the FGC_Ether fieldbus was developed, allowing not just a 100 Mbps bandwidth between the general purpose computer and the FGC3, but also the distribution of a low jitter ($\sigma_j \approx$ 10 ns) 50 Hz signal from the timing network to the FGC3 using the same Ethernet cable [8]. The 50 Hz phase was then recovered with a hybrid PLL using an IQD 25 MHz VCXO to clock the free running counter on the FPGAmk. An FPGA was used as phase comparator. Finally, a Renesas RX610 CPU runs the 20 ms ISR to perform the filtering and control calculations of the PLL algorithm.

Figure 6 shows the FGC3 PLL algorithm. The reference $r[n]$ is calculated internally every 20 ms by adding 800,000 tics (20 ms) to the last reference value. This reference is compared with the 25 MHz counter latched by the arrival of the FGC_Ether 50 Hz signal $ext(t)$ (or time packet in case the 50Hz sync pulses are temporarily missing). The phase error $e_{ext}[n]$ is smoothed using an averaging filter of length 10, and the reference is fed to the PI algorithm every tenth iteration. The PI output $q[n]$ is limited by the state machine to the maximum VCXO pullability of ±100 ppm, and sent to the FPGA to set the frequency offset. The FPGA upsamples the PI output to 625 KHz to recover its fractional part, and to increase the theoretical resolution of the 14-bit DAC by 8 additional bits. The VCXO frequency is then used as a clock for the free running counter $c[m]$, and also as a clock for the FGC3 CPU and DSP.

The linear analysis of the algorithm follows exactly the one for the FGC2 PLL by assuming that the DAC has an infinite resolution, and taking into account that the down-sampled and filtered noise, $e[n]$, is uncorrelated. The open loop transfer function is:

$$L(z) = \frac{K_{vcxo}(k_i + k_p)}{(z-1)^2}(z - \frac{k_p}{k_i + k_p}) \qquad (3)$$

Where $K_{vcxo} \approx 0.001$, the regulation frequency is 5 Hz, $k_i = 0.003$, and $k_p = 0.05$. The integral and proportional

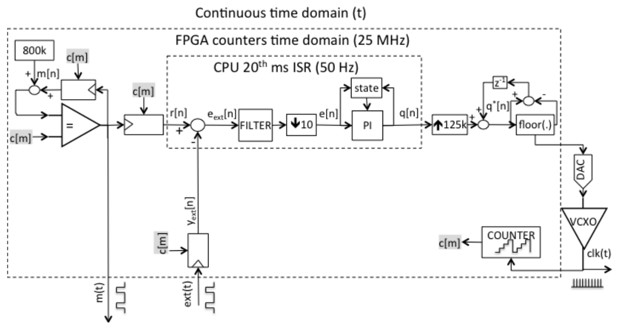

Figure 6: FGC3 PLL Algorithm.

gains have been chosen to minimise the jitter while keeping the lock time under 2-3 seconds. Note that the loop gain is much lower than the FGC2, which corresponds to the slower lock time constant, and the higher accuracy.

It is also worth mentioning that as in the FGC2 case, we have obviated the compensation of the Ethernet and 50 Hz signal propagation delay over the FGC_Ether fieldbus. In the FGC3 case, there is no compensation on the 50 Hz signal, and the propagation delay of the Ethernet packet is compensated using the average offset with respect to the 50 Hz signal. Given that the distance between the computer and the FGC3 in the injectors is smaller than 500m, the time precision is better than 2.5 μs, and adjusting the cable length it is easy to achieve 100 ns.

Simulation and Measurements

The FGC3 PLL algorithm achieves a phase error jitter with $\sigma_e < 40$ ns when the 50 Hz signal is used, and $\sigma_e \approx 0.5\mu$s when the time of arrival of the Ethernet packet is used (degraded mode). In order to achieve a small and stationary jitter on the Ethernet arrival time, careful attention should be put on the real time thread sending the packet on the Linux front-end: the power saving features of the CPU and operating system should be disabled, the load of the computer should be as low as possible, the thread handling the synchronous send of the Ethernet time packet should minimise the computational load and be as deterministic as possible, the real-time thread should also have the highest priority possible, and finally, the CPU affinity should be fixed. Otherwise, the jitter won't only be higher but it could be non-stationary.

The better FGC3 phase error jitter is achieved by the lower jitter of the 50 Hz signal ($\sigma_j \approx 10$ ns), and the higher resolution of the 25 MHz VCXO with respect to the 2 MHz counter used on the FGC2. On the other hand, the pullability of the VCXO ($\pm$100 ppm) is much smaller than the equivalent pullability of the FGC2. This is the price paid for the additional resolution of the actuator. In order to minimise its impact and improve the lock time, two measures were taken. First, the 20 millisecond reference counter, $m[n]$, is initialised to the value of the free running counter at the arrival of the first Ethernet time packet. And second, the VCXO frequency, $q[n]$, is initialised to the previously held value before a reset.

As in the case of the FGC2, the algorithm performance was both simulated and measured using the same source code. Given that the free running counter is initialised by the Ethernet time packet, the time to lock after a power cycle is always the same (< 2 s). Figure 7 shows the simulated and measured phase errors for the worst case scenario where the FGC_Ether fieldbus has been reconnected after being disconnected for several hours. In this case, the phase error has drifted with a speed of below 1 μs/s due to the inaccuracy of $q[n]$ and the relative drifts of the FGC3 and timing network clocks. For this worst case scenario the lock time is around 24 seconds. Although the FGC3 PLL didn't need a PI with gain scheduling, the PLL states were kept for informative purposes. On the plot the Fast Slew phase corresponds to the VCXO being clipped to its maximum pullability, while Lock corresponds to a phase error below 16 μs for more than 2 s.

Note that plot shows two differentiated phases, one where the VCXO input is saturated, and then once the phase difference is small enough, the linear regulator takes over.

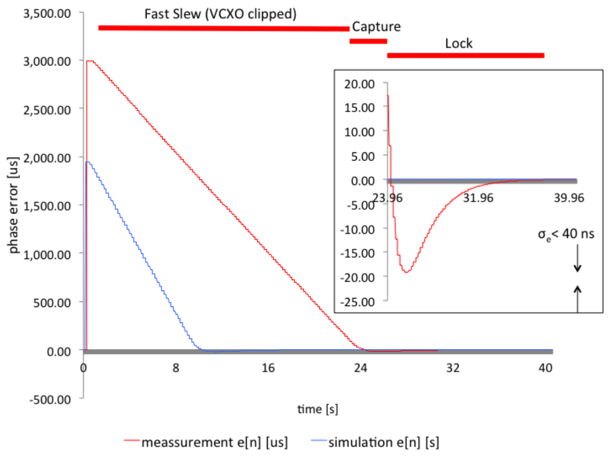

Figure 7: FGC3 PLL phase error after loosing the timing signal for two hours.

CONCLUSION

The PLL algorithms for the FGC2 and FGC3 have been described and their performance exceed the operational requirements of the LHC and its injectors. Although both algorithms use the same control algorithm, the FGC3 PLL departed from the pure software solution chosen for the FGC2. The FGC3 uses an VCXO and a higher frequency counter clock to achieve a more precise lock of the phase. This additional precision can be exploited thanks to the lower jitter of the 50 Hz signal coming from the FGC_Ether interface ($\sigma_j \approx 10$ ns). Instead on the FGC2 the input jitter depends on the time of arrival of the WorldFIP periodic packet ($\sigma_j \approx 0.5\mu$s). This additional precision on the FGC3 does not affect the lock time thanks to the proper initialisation of the FGC3 free running counters when the first FGC_Ether packet is received.

REFERENCES

[1] C. Azeredo-Leme,“Clock Jitter Effects on Sampling: A Tutorial”, IEEE Circuits and Systems Magazine, Aug 2011.

[2] Kao, Chung-Yao, et al., “Simple stability criteria for systems with time-varying delays”, Automatica 40.8, 2004.

[3] K. J. Astrom, R. M. Murray, “Feedback Systems”, Princeton University Press, 2008.

[4] V.D. Blondel and J.N Tsitsiklis, “A survey of computational complexity results in systems and control”, Automatica 36, 2000.

[5] J. C. Bau et al., “Managing the real-time behaviour of a particle beam factory: the CERN Proton Synchrotron complex and its timing system principles”, IEEE Transactions on Nuclear Science 45, 1998.

[6] J. Lewis et al., “The CERN LHC Central timing, a vertical slice”, ICALEPCS, 2007.

[7] D. Calcoen et al., “Evolution of the CERN power converter function generator/controller for operation in fast cycling accelerators”, ICALEPCS, 2011.

[8] S. Page et al., “Migration from WorldFIP to a Low-Cost Ethernet Fieldbus for Power Converter Control at CERN”, ICALEPCS, 2013.

[9] T. Wijnands, “Functional specification for LHC Power Converter control”, LHC Project Note 183, 1999.

[10] J. G. Proakis et al., “Digital Signal Processing”, Prentice-Hall, 1992.