A METHOD OF MEASURING NOISE AND DETECTING GLITCHES IN MAGNET POWER SUPPLIES AT THE APS STORAGE RING*

T. Fors†, J. Carwardine, A. Hillman, J. Maclean, and S. Sarkar#
Advanced Photon Source, Argonne National Laboratory, Argonne, IL

Abstract

Goals for Advanced Photon Source reliability and orbit stability make it important to quickly identify and replace individual converters that may glitch or have relatively noisy output. The many converters involved and the limitations of the standard EPICS control interface makes it difficult to detect these converters. This paper presents our approach for quantifying noise and identifying glitches in individual converters. Implementation of the algorithms in the local power supply controller is addressed with concern for resource limitations such as bandwidth, sampling rate, and processing power.

1 INTRODUCTION

1.1 Motivation

Operational goals for the magnet power supply system at the Advanced Photon Source (APS) are to meet 99% availability with a mean time between failures (MTBF), defined as an unscheduled beam loss, of 200 h [1]. Last fiscal year approximately 10 beam losses were attributed to power supply glitches. With nearly 1400 magnet power supplies in the storage ring, it is very difficult to determine which supply caused a beam dump. Normally, it is not until a supply becomes a repeat offender that it can be identified and removed from operation.

In addition to these reliability goals, orbit stability requirements are becoming more demanding, with the goal being to reach submicron stability in the near future. A noisy supply can easily prevent us from reaching this goal.

Work is underway at the APS to provide ways of easily tracking down problematic power supplies so they can be quickly replaced. One method being developed uses the beam position monitors to detect the signature imposed on the beam orbit by a noisy supply [2]. While this method can easily identify which sector a noise source is in, and can often locate the girder where it is located, identifying the individual power supply is still difficult. This paper discusses another tool developed to aid us in meeting these reliability and stability requirements.

1.2 Objective

Our objective is to implement a method of detecting these suspect power supplies quickly and easily. Because of the magnet time constant, we need to be able to detect glitches in the 10s to 100s of milliseconds range. We chose to instrument each supply such that its noise could be measured and output glitches detected using digital signal processing techniques [3]. While designing new hardware for this task would give us great flexibility and computational power to implement our algorithm, we felt we could develop a system that would meet the objectives using the existing data acquisition and control system and avoid the cost of upgrading the control system hardware for 1400 supplies.

1.3 Challenges

The APS power supply control network is structured as shown in Fig. 1.

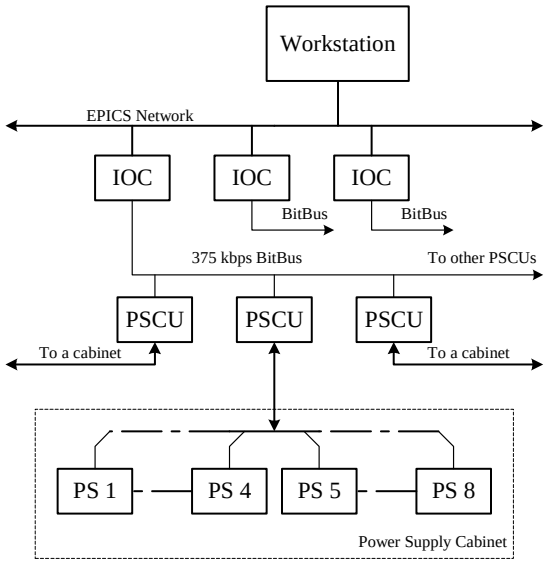

Figure 1: Power supply control network.

VME-based I/O controllers (IOCs) running the EPICS software [4] form the controls network. A Motorola 68000-based power supply control unit (PSCU) [5] handles the control and monitoring functions for a single cabinet that can contain up to eight supplies. Communications between the PSCU and IOC are via a 375 kbps master/slave BitBus network. The interface between the PSCU and individual power supplies provides a direct-wired connection to the 16-bit analog to digital converter (ADC) in the power supply. The present PSCU design only samples data from the power supply on demand from the IOC.

*Work supported by U.S. Department of Energy, Office of Basic Energy Sciences, under Contract No. W-31-109-ENG-38.
† tfors@aps.anl.gov
# Present address: Northeast Proton Therapy Center, Boston, MA

To detect glitches of the duration we're concerned with, we need to sample output current faster than 100 Hz. Due to a data acquisition bottleneck with the BitBus network, we chose to implement our digital algorithm in the PSCU. This decision presented us with our most significant challenge: speed.

The PSCU contains a 68000 processor running at a clock frequency of 16 MHz. It will be sampling eight power supplies at a 100-Hz rate and performing our calculations on those samples. Therefore, our algorithm must execute as quickly as possible. We are also limited by the PSCU to fixed-point math.

2 BLOCK DIAGRAM

Figure 2 shows a block diagram of our noise measurement and glitch detection algorithm.

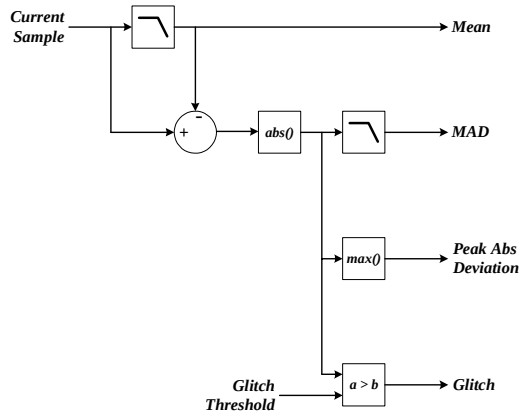

Figure 2: Algorithm block diagram.

2.1 Current Sample

We start with discrete time sampling of the power supply output current. The existing control system already provides samples of the output current at 0.5 Hz, as requested by the IOC. For our application, we will be sampling this current at a much faster rate, which will be driven by the PSCU instead of the IOC.

2.2 Mean

Applying a digital low-pass filter to the current samples, we get the mean value of the output current. The mean value represents the DC component of the power supply output current and is used to extract the AC component.

2.3 Mean Absolute Deviation (MAD)

Subtracting the mean value from the current samples gives us the AC component of the power supply output or instantaneous deviations from the mean. The most common method of expressing this AC component is to use the standard deviation or root mean square (rms) of these instantaneous values. The rms calculation can be time-consuming to perform in embedded hardware, and since we would like to sample as quickly as we can with our existing hardware, we need an alternative.

Another method of measuring deviation from the mean is to use the mean absolute deviation. This is calculated simply by applying another low-pass filter to the absolute value of our instantaneous deviations. This method is more suited to our embedded application.

2.4 Glitch Detection

The mean absolute deviation gives us a measure of the noise, but a fast glitch on the output of a power supply will go largely undetected. By applying a threshold comparator to the instantaneous absolute deviations, we can detect glitches that would otherwise be missed.

2.5 Peak Absolute Deviation

A peak detector is also added to give us an indication of the magnitude of glitches.

3 IMPLEMENTATION

3.1 Low-Pass Filter

At the heart of our algorithm is the low-pass filter. We chose an infinite impulse response (IIR) filter for our low-pass because it requires less memory and fewer computational cycles per discrete time sample than the finite impulse response (FIR) filter. The IIR filter's difference equation is

$$y_n = (1-\alpha)\cdot x_n + \alpha \cdot y_{n-1} \tag{1}$$

$$\alpha = \frac{2^{15} - 2^b}{2^{15}} \tag{2}$$

By forcing the coefficient, α, to be defined by equation (2), where b is an integer number, we can further reduce the computational burden on the microprocessor because the multiplication functions in equation (1) can now be performed with simple bit-shift operations. These bit-shift operations take significantly fewer clock cycles to execute than the multiply operation, as shown in Table 1.

Table 1: 68000 Processor Execution Cycles

Instruction	Clock Cycles
Add	6
Subtract	6
Shift	8+2b
Multiply	70
Divide	158

The C code to implement this filter is shown below.

```
y=((y<<15)-(y<<b)+(x<<b))>>15;
```

By leaving the value b adjustable, we can vary our filter coefficient according to equation (2), which in turn varies the response of our filter. The relationship between the filter coefficient, α, and the time constant, τ, of an equivalent RC filter can be approximated by equation (3), where T is the sample period of the IIR filter.

$$\alpha \approx \frac{\tau}{T+\tau} \tag{3}$$

3.2 Fixed-Point Math

The problem with the C code shown above is that at each iteration of the filter, the fixed-point result y is scaled down by 2^{15}. Changing this code to eliminate the scaling at each filter iteration improves the resolution of our result. We can then perform a true division operation in the IOC, where we can do floating-point math and dramatically improve our results.

3.3 "Look-Away"

Set-point changes in the magnet power supplies happen as a course of normal operation. For example, during magnet conditioning, new set points are sent to the supplies every few seconds. To avoid detecting these sudden changes in output current as glitches, a simple "look-away" counter is used. When a new set point is received by the PSCU, it loads a counter that decrements with each sample. While this counter is non-zero, it suspends the noise monitor algorithm to allow for the output current to stabilize at its new value.

A similar counter is implemented in the glitch detector. Once a glitch is detected, the counter is initialized and prevents further glitches from being detected until it reaches zero again. This method prevents counting a single event as multiple glitches.

3.4 Mutual Exclusion and Priority Inversion

The previous PSCU code consisted of one task, the BitBus task. This task received BitBus messages from the IOC. It decoded and dispatched the messages to the proper routine, then sent the results back to the IOC. With only one task using the PSCU hardware, there was never any contention.

We added a second task to the PSCU that implements our algorithm. It performs the periodic sampling of the output current, the IIR filters, and the remaining calculations for the eight power supplies. It consumes all of the otherwise-idle time of the processor in order to achieve a sample rate for each supply of approximately 160 Hz.

Both tasks access the same PSCU hardware that interfaces to the power supplies. To synchronize them, we implemented a simple semaphore [6] with the TAS (Test and Set) instruction of the 68000 processor [7]. This ensures that only one task can access the hardware at a time. Giving a higher priority to the BitBus does not ensure it will always execute when it needs to because the semaphore will be claimed by the noise monitor task the majority of the time. To solve this problem of priority inversion, the BitBus task sets a flag when it wants to run. Before the noise monitor task claims the semaphore, it will yield to the BitBus task until the flag is cleared. Then it will attempt to claim the semaphore.

4 CONCLUSIONS

The new PSCU code has been installed in the APS storage ring and has been running for several months now. Initial results indicate we need to improve the integrity of our current sampling. Frequently, current readings are in error because of marginal design and quality control issues, including shorted traces on interface cards, incorrect capacitor values, and circuit cards in supplies that had not been updated to the latest revision. The noise monitor tool has been instrumental in systematically finding and correcting these problems in the 1400 power supplies that make up the storage ring. Work is progressing to correct these problems. When this work is completed, we believe we will have a system that meets our initial goals. Our system provides simultaneous detection of output current glitches and a measure of output current stability for individual power supplies. This will allow us to quickly and efficiently identify faulty supplies so they can be removed from operation.

5 ACKNOWLEDGMENTS

The authors wish to thank D. Donkers for his assistance in testing new PSCU firmware and installing the new EPROMs.

6 REFERENCES

[1] A. Hillman et al., "Magnet Power Supply Reliability at the Advanced Photon Source," these proceedings (2001).

[2] C. Schwartz et al., "A Localization Algorithm for Strong Sources of Orbit Motion in the APS Storage Ring," these proceedings (2001).

[3] J. Carwardine, F. Lenkszuz, and R. Merl, "Fundamentals of Digital Signal Processing with Applications to Accelerators," *USPAS '99.*

[4] http://www.aps.anl.gov/epics/

[5] O. Despe and D. McGhee, "Control Units for APS Power Supplies," *PAC 1993 Proceedings* (1993).

[6] Andrew S. Tanenbaum, *Operating Systems Design and Implementation*, Prentice-Hall (1987).

[7] *Motorola M68000 Family Programmer's Reference Manual* (1992).