2015 IEEE Conference on Control Applications (CCA)
Part of 2015 IEEE Multi-Conference on Systems and Control
September 21-23, 2015. Sydney, Australia

# Built-In Stepped-Sine Measurements for Digital Control Systems

Daniel Y. Abramovitch*

***Abstract*— Mechatronic systems are a class of systems in which the mechanics and electronics are closely intertwined. Examples range from small scale mechanics such as hard disk drives, optical disk drives, and atomic force microscopes (AFMs) [1] to large scale mechanisms such as wind turbines [2]. Mechatronic systems often include multiple lightly damped resonances and/or anti-resonances. The control of these systems involves either restricting the bandwidth to being roughly a quarter of the frequency of the first system resonance or identifying these dynamics and finding a way to mitigate them with the control algorithm [3], [4]. This paper describes a stepped-sine or sine dwell algorithm (also known as swept-sine in the industrial literature) which is built into the real-time controller for precise identification of highly resonant mechatronic systems.**

## I. Introduction

System ID methods are categorized into parametric and non-parametric methods [5], [6]. Parametric methods usually are operated directly on time domain measurements of the system. They have advantages in that they are often simple, involve low levels of computation at each time step, and can be applied directly to control design. Almost always, these are discrete-time models of the system that work well when the underlying system is well damped, but suffer with systems that have lightly damped dynamics.

Non-parametric methods generally involve stimulating the system with some auxiliary signal and measuring response. Often these measurements involve generating an estimate of the Frequency Response Function (FRF) [7], also called the Empirical Transfer Function Estimate (ETFE) [5], [6], which denotes a set of ordered pairs where one value is a frequency and the other is the response of some part of the system at that frequency. A Transfer Function (TF) denotes a parametric model of the system with response to some frequency variable. Going from a TF to a FRF involves evaluating the TF at a set of frequencies and recording its complex response. Going from a FRF to a TF is significantly more difficult, as it involves some sort of optimized fit of the FRF to a set of parameters over some frequency range. This step is highly dependent upon the quality of the FRF [8].

This paper contains no new theory. Instead, it presents a practical method of implementing very fast, highly accurate stepped-sine measurements as a part of a digital controller. It makes the case that such a built-in algorithm is far more desirable than the same algorithm on an external instrument. The author believes that the former is a key step in automating the control design of highly oscillatory mechatronic systems [9].

*Daniel Y. Abramovitch is a system architect in the Mass Spectrometry Division, Agilent Technologies, 5301 Stevens Creek Blvd., M/S: 3U-DG, Santa Clara, CA 95051 USA, danny@agilent.com

## II. FFT versus Stepped-Sine Measurements

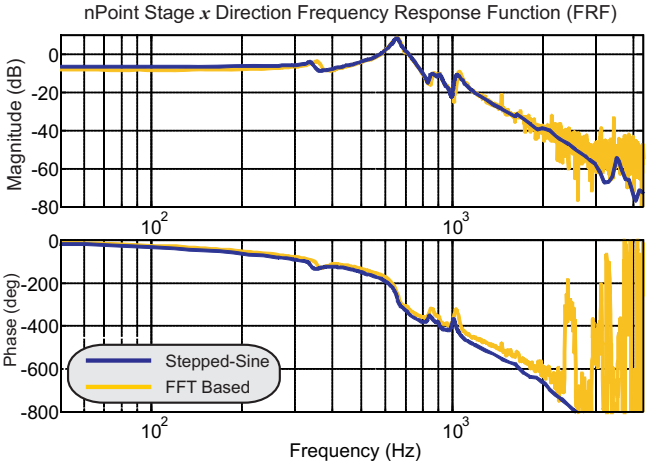


Fig. 1. A comparison of FFT and stepped-sine based FRFs on an nPoint NPXY100 [10] stage. (Courtesy: Jeff Butterworth.)

Mechatronic systems are often characterized by system flexibility, which manifests itself as a large number of high Q resonances and anti-resonances in the physical system FRF. Such responses are typically hard to measure using time domain methods because the amount of signal concentrated near any particular feature is small. In other words, identifying large numbers of high Q features during normal operation is difficult if not impossible, unless normal operation continuously stimulates all of those features.

Likewise Fast Fourier Transform (FFT) based methods do not focus signal in any one feature area, instead relying on broadband excitation (pseudo-random, noise-like signals) or on a chirped sine signal. FFTs are computationally fast, but this speed comes at a price. Besides the lack of frequency isolation on the input, the frequency bins are fixed for a given sample rate and number of samples.

In contrast stepped-sine or sine dwell [7] uses a single sinusoid injected into the system. The input is continued until the system goes to steady state and then the output is measured. The LTI system response that sinusoid will be another sinusoid at a different magnitude and phase, but at the same frequency. By repeating measurements, as series of frequency points can be identified and these will delineate the frequency response function of the system. Because the stimulus is a single sinusoid, the algorithm uses coherent demodulation to extract the system response.

978-1-4799-7786-4/15/$31.00 ©2015 IEEE

The focus on one frequency at a time has a clear SNR advantage over broadband methods. While both methods will include components from the normal operation of the loop in the measured signal, FFT methods cannot isolate on a single stimulus frequency. The loop signals may show up in the stepped-sine measurements, but these will largely be not coherent with the stimulus and therefore suppressed by the coherent demodulation. Furthermore, the measurement degrades gracefully in the presence of nonlinearities [11], [12]. For this reason, stepped-sine responses typically produce much "cleaner" measurements, especially at higher frequencies where the mechatronic system response is low as clearly seen in the example of Figure 1. This has also been analyzed in the context of settling time and signal to noise ratios in earlier work [13].

The cost of doing this is a far more complicated software algorithm combined with significantly longer measurement times, as the measurements are repeated with each frequency step [13]. For this reason, stepped-sine measurements have largely been restricted to external and expensive instruments. As will be discussed in Section III, this is a severe limitation in modern digital control systems. The contribution of this paper is to reformulate the algorithm so it is simple enough to implement in an FPGA. In doing so, it enables high precision FRF measurements to be added to any digital controller.

## III. Frequency Response Function Measurements

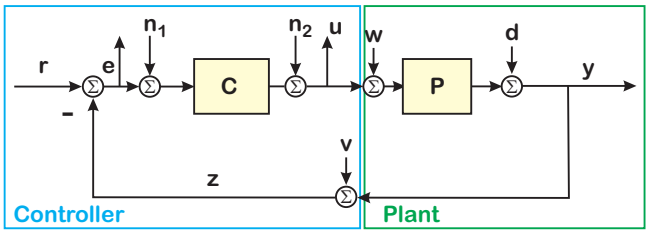


Fig. 2. A basic block diagram for frequency response function measurements with noise added in. The process noise and measurement noise are signified by $w$ and $v$, respectively, while an unmeasured disturbance that affects the output is called $d$.

A basic single input, single output (SISO) linear system is shown in Figure 2. The system has the typical inputs and outputs. One of the traditional stimulus places is at $n_1$ although the system setup may require stimulus at access point $n_2$. The error signal, $e$, and the control input, $u$, represent two measurement points, as does the measured output, $z$. The reference signal may not be accessible in an output feedback configuration, but the other injection points are all included in the controller. Ignoring noise and disturbances ($w = v = d = 0$), we have closed-loop measurements:

$$T_{cl} = \frac{Y}{R} = \frac{Y}{N_1} = \frac{PC}{1+PC}. \quad (1)$$

$$S_{cl} = \frac{E}{R} = \frac{E}{N_1} = \frac{1}{1+PC}. \quad (2)$$

Also, the compensator, $C$, can be measured from available signals $E$ and $U$, as $C = U/E$, and the open loop FRF can be obtained from measurements of closed loop quantities by means of

$$PC = \frac{T_{cl}}{1-T_{cl}} \quad \text{or} \quad PC = \frac{1}{S_{cl}} - 1. \quad (3)$$

From an "opened" closed-loop measurement as in (3) and a measurement or model of $C$, we can get our plant response, $P$. Clearly, many combinations of measurements are possible and as long as there is sufficient signal to noise in the measurement, loop manipulations can be done to extract needed responses [14], [15].

Looking at just the plant to measurement of Figure 2 with the loop opened and letting $d = 0$, we can measure:

$$Z(f) = P(f)U(f) + P(f)W(f) + V(f). \quad (4)$$

Extracting $P(f)$ from $Z(f)/U(f)$ would be corrupted by the noise. However, if we use cross spectra and take the expectation [16], then

$$E\{Z(f)U^*(f)\} = P(f)E\{U(f)U^*(f)\}. \quad (5)$$

since the noise, $w$ and $v$ is typically uncorrelated with $u$ (with the loop open). Practically, the single sided auto or cross spectrum is used [16], i.e.,

$$G_{zu}(f) = \begin{cases} 2E\{Z(f)U^*(f)\} & f \geq 0 \\ 0 & f < 0 \end{cases}. \quad (6)$$

Note that this depends on expectation which means that we need to be careful about how we average our measured signals. This will be discussed in Section VII. In actual measurements, these transforms will be replaced by time-dependent estimates of the Fourier transforms [16].

Likewise the coherence function is given by

$$\gamma^2(f) = \frac{G_{zu}(f)G_{uz}(f)}{G_{uu}(f)G_{zz}(f)} \quad \text{where} \quad G_{zu}(f) = G^*_{uz}(f) \quad (7)$$

and gives an indication of how much of the output is generated from the input [16]. The coherence function is an excellent figure of merit, and is limited below by 0 and above by 1. If the output, $y$, is entirely caused by the input then $\gamma^2(f) = 1$. The presence of noise or nonlinearities will cause the coherence to be less than 1. For any FRF measurement, $\gamma^2(f)$ tells us how much we can trust the measurement.

## IV. The Case for Built-In Stepped-Sine

A generic control system is shown in Figure 3. Identifying the many component blocks in the system involves injecting and extracting signals at multiple locations, and the majority of access points are buried in the digital controller.

Using an instrument through analog test points involves not only creating those test points with circuitry, but also forcing many signals that were digital to be converted to analog signals before being measured with our external instrument. Perhaps more critical is the fact that much of the controller is inaccessible to the instrument. This can be remedied by modifying the instrument to have digital interfaces, as was done with the HP 3563A [17], but connecting

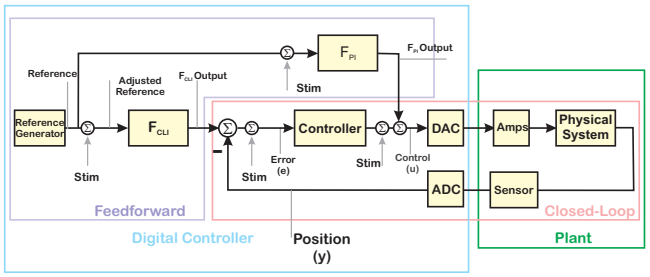


Fig. 3. Block diagram for controlled system showing stimulus points and response measurement points.

such an instrument involved coming up with a digital bus protocol between the instrument and the control system [14], [9]. Given this level of work, one might as well build the instrument right inside the controller.

Returning to Figure 3, we see that not only do we want multiple digital access points, but reconfiguration of the controller would require modifying those access points. This reconfigurability is exactly what software allows us to do. In a single CPU or DSP based system, operations for computing the stepped-sine stimulus and integration would take away from the processing time available for real time control. This evaporates with the parallel processing capability of Field Programmable Gate Arrays (FPGAs). While difficult to program, FPGAs allow algorithms to multiplex in chip space instead of processing time. Very simply, this means that in an FPGA, we do not burden our controller by the addition of this measurement algorithm. This allows for measurements of systems with extremely high sample frequencies. In the examples of Section VIII, the atomic force microscope (AFM) system in question was sampled at 2 MHz. The FPGA firmware would have allowed measurements on dynamic systems sampled at 40 MHz. In contrast, the HP 3562A and HP 3563A were external instruments with sample rates limited to 250 MHz [17].

## V. The Stepped-Sine Integral

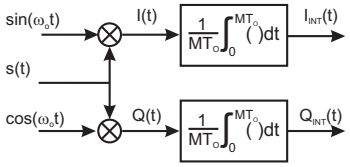


Fig. 4. The stepped-sine demodulation replaces the low pass filter of the lock-in amplifier [18] with an integration over an integer number of periods of the sinusoid.

The solution in this paper involves breaking the computation up into two blocks, those that need to be done in real time and those that can be treated as pre and post processing. The real-time computations involve stimulating the loop at various points, extracting responses at other points, and computing the stepped-sine integral, will be described in Section VI. The pre and post processing, in which the measurement parameters are set and integrated response is tabulated and turned into magnitude and phase will be described in Section VII.

The stepped-sine method would have one of the loop stimulus signals set to a sinusoid at a desired frequency, $\omega_0 = 2\pi f_0 = \frac{2\pi}{T_0}$. The other stimulus inputs would be set to 0. A given output signal, s(t), can be demodulated using a stepped-sine demodulator. Referring back to Figure 3 we can use Fourier series to decompose the signal, s(t), as

$$s(t) = A_0 + \sum_{k=1}^{\infty} \left(A_k \sin(k\omega_0 t) + B_k \cos(k\omega_0 t)\right). \quad (8)$$

We can expect that if the stimulus signal is single sinusoid, then $s(t)$ will have a strong first Fourier component:

$$\begin{aligned} s(t) &\approx A_1 \sin(\omega_0 t) + B_1 \cos(\omega_0 t) + n(t) && (9)\\ &= C_1 \sin(\omega_0 t + \phi_1) + n(t), && (10)\end{aligned}$$

where

$$C_1 = \sqrt{A_1^2 + B_1^2} \quad \text{and} \quad \phi_1 = arctan\frac{A_1}{B_1}. \quad (11)$$

Mixing with in-phase and quadrature signals as shown in Figure 4 yields

$$\begin{aligned} \int I(t)dt &= \int s(t)\sin(\omega_0 t)dt \\ &\approx \int C_1 \sin(\omega_0 t + \phi_1)\sin(\omega_0 t)dt \\ &\quad + \int n(t)\sin(\omega_0 t)dt \end{aligned} \quad (12)$$

and

$$\begin{aligned} \int Q(t)dt &= \int s(t)\cos(\omega_0 t)dt \\ &\approx \int C_1 \cos(\omega_0 t + \phi_1)\cos(\omega_0 t)dt \\ &\quad + \int n(t)\cos(\omega_0 t)dt. \end{aligned} \quad (13)$$

As mentioned above, in a stepped-sine demodulator, we will want to integrate over an integer, M, number of periods of the frequency that we wish to demodulate. Making the integrals definite and using well known trigonometric identities, yields:

$$\begin{aligned} \tfrac{1}{MT_0}\int_0^{MT_0} I(t)dt &= \tfrac{C_1}{2}\Big(\cos\phi_1 \tfrac{1}{MT_0}\int_0^{MT_0} dt \\ &\quad - \tfrac{1}{MT_0}\int_0^{MT_0} \cos(2\omega_0 t + \phi_1)dt \\ &\quad + \tfrac{1}{MT_0}\int_0^{MT_0} n(t)\sin(\omega_0 t + \phi_1)dt\Big) \text{ and} \end{aligned} \quad (14)$$

$$\begin{aligned} \tfrac{1}{MT_0}\int_0^{MT_0} Q(t)dt &= \tfrac{C_1}{2}\Big(\sin\phi_1 \tfrac{1}{MT_0}\int_0^{MT_0} dt \\ &\quad - \tfrac{1}{MT_0}\int_0^{MT_0} \sin(2\omega_0 t + \phi_1)dt \\ &\quad + \tfrac{1}{MT_0}\int_0^{MT_0} n(t)\cos(\omega_0 t + \phi_1)dt\Big). \end{aligned} \quad (15)$$

Equations 14 and 15 both have the properties that the second term on the right hand side goes to 0 for all positive $M$. The third term goes to 0 for increasing $MT_0$ as long as $n(t)$ is uncorrelated with the mixing sinusoids.

Such precise control of the integration period is difficult in an analog circuit but straightforward in a digital operation. As $MT_0$ gets large the contribution of $n(t)$ goes to 0, yielding the familiar relationships

$$I_{int} = \frac{1}{MT_0}\int_0^{MT_0} I(t)dt \approx \frac{C_1}{2}\cos(\phi_1) \quad (16)$$

and

$$Q_{int} = \frac{1}{MT_0}\int_0^{MT_0} Q(t)dt \approx \frac{C_1}{2}\sin(\phi_1). \quad (17)$$

By assembling the two integrals into one complex number we get the first Fourier component of $s(t)$ at $f_0$, $S(f_0)$. We can do this for any number of signals around the loop and any desired set of frequencies and using Equation 5, compute the complex FRF, $H(f)$, between those two measurement points.

There are several issues with standard methods of demodulation. The first is that imperfections in the integration approximation and noise in the signal require that $MT_0$ be large, relative to the period of the frequency at which demodulation is to take place, $T_0$, so $M$ must be large.

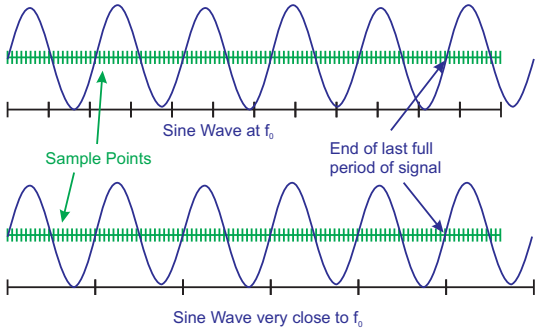


Fig. 5. The top drawing shows a sine wave which doesn't end up on an integer number of sample points. Adjusting $f_0$ slightly allows an integer number of periods to line up with an integer number of samples, as shown in the bottom drawing.

The second is that with a digital controller, we have to be careful if we want to honor our desire to integrate over an integer number of periods of oscillation. We want

$$NT_S = MT_0, \quad (18)$$

where $N$ are are the number of samples in the integration, $T_S$ is the sample period, $M$ is the number of periods of oscillation, and $T_0$ is the period of oscillation. As illustrated in Figure 5 the data sample rate is rarely an integral multiple of the oscillation frequency, so it is difficult to make Equation 18 hold. Most digital systems are run at a fixed sample rate,$f_S = \frac{1}{T_S}$. The oscillation frequency, $f_0 = \frac{1}{T_0}$, comes from the frequencies at which we want to measure the FRF. That means $f_0$ will vary but $f_S$ will not. A solution is that for any desired $f_0$ and $M$, we can pick $N$ such that:

$$NT_S \leq MT_0 = N_{Real}T_S \leq (N+1)T_S. \quad (19)$$

We then round $N_{Real}$ to the nearest integer. We don't want to change $T_S$ or $M$, so we are left with adjusting $T_0$ so that

$$\tilde{N}T_S = M\tilde{T}_0, \quad (20)$$

Here $\tilde{N}$ is either $N$ or $N+1$ and $\tilde{T}_0$ is the adjusted period of oscillation which makes equality hold. The adjustments to that make equality hold can be kept small if $N$ and $M$ are made large. The computational hardware must have enough bits in the register that holds the intermediate integral approximation so as to allow for long integrals over many sample points. This is in contrast to the algorithm the author presented in [19] and [20] in which the oscillatory frequency was much closer to the sample frequency and there was a strong desire to minimize latency by keeping $M$ small.

The steps involved in generating a FRF with stepped-sines are:

- Select a set of oscillation frequencies, $\{f_0\}$, at which to compute the FRF.
- For each oscillation frequency, $f_0$, and desired number of oscillations, $M_0$, there will be a number of data samples, $N$, such that

$$NT_S \leq M_0T_0 \leq (N+1)T_S. \quad (21)$$

- Adjust each $T_0 = \frac{1}{f_0}$ so that equality holds for one side of Equation 21. For $M_0$ sufficiently large, this adjustment will be small.
- Set up a measurement period of $NT_S = M_0T_{0,adj}$.
- Set oscillator in the system controller to generate sinusoidal stimulus at frequency, $f_{0,adj} = \frac{1}{T_{0,adj}}$.
- At each sample time step, inject the sinusoidal stimulus signal into the chosen input in the control loop and measure the response at the chosen measurement points of the loop, and compute the next step in the Fourier integral.
- Store the partial integrals in memory.
- After $N$ samples, finalize the Fourier integrals by dividing by $MT_{0,adj} = NT_S$ to get $\frac{C_1}{2}\cos(\phi_1)$ and $\frac{C_1}{2}\sin(\phi_1)$ at each frequency, in real time.
- In post processing, use the integral values to compute the desired auto and cross spectra. If averaging is on, repeat the measurement and integral $N_{avg}$ times.
- In post processing, compute the desired FRFs between sets of these signals using the averaged cross and auto spectra.

The choice of measurement frequencies, the adjustments, and the setup of the integral can be done in a host computer. Generation of the stimulus, measurement of the signal responses, and calculation of the integral need to be done on the real-time system. (One could argue that one could simply store the measured data, pass it to the host computer, and do the calculations there. However, for even modest sample rates, the data transfer becomes huge and impractical. 10 periods of a 100 Hz signal sampled at 1 MHz results in $10^5$ points per channel to store transfer. That is a lot of memory for a real-time system.) Finally, Fourier coefficients are passed back to the host computer for post processing and another stimulus/integration run is set up.

## VI. Stepped-Sine Stimulus and Integration for FPGAs

The key to approximating the integral in real time is to turn it into a convenient sum form, that is:

$$\begin{aligned}\frac{1}{MT_0}\int_0^{MT_0} I(t)dt &= \frac{1}{NT_S}\int_0^{NT_s} I(t)dt \\ &\approx \frac{1}{NT_S}\sum_{k=0}^{N-1} I(kT_S)T_S = \frac{1}{N}\sum_{k=0}^{N-1} I(kT_S).\end{aligned} \quad (22)$$

Likewise,

$$\frac{1}{MT_0}\int_0^{MT_0} Q(t)dt = \frac{1}{NT_S}\int_0^{NT_s} Q(t)dt \approx \frac{1}{NT_S}\sum_{k=0}^{N-1} Q(kT_S)T_S = \frac{1}{N}\sum_{k=0}^{N-1} Q(kT_S). \quad (23)$$

We are using a rectangular rule approximation, unlike the fifth order polynomial of [7] or the trapezoidal rule of [19], since our relatively high sample rate and large number of samples minimizes the integration error caused by the simple approximation. We still need to normalize the sums by $\frac{1}{N}$ and for a long integration with a fast sample frequency, $N$ can be huge. For example on a Xilinx FPGA using a DSP48E block [21], has multiplies of 25 bit by 18 bit numbers, where the numbers are in twos compliment form. If we have 25 bit data values multiplied times 18 bit sine/cosine values, the resulting product has 43 bits, of which the top 2 are redundant. We want to sum a lot of these values, so we need a fairly large accumulator. (The DSP48E has a 48 bit P (product) register.)

So, if we have up to 20 million sums, that's enough for 100 cycles of 10 Hz signals sampled at 2 MHz. For 20e6 sums, we need log2(20e6) = 24.25 bits or just under 25 bits. We have bits 43-48 free (6 bits) and then we need 25 bits of accumulation total, we will need a 67 bit register. Choosing a 68 bit accumulation register, we still need to multiply by 1/N, but doing a multiply on a 68 bit quantity would involve a slow process with multiple DSP48Es. However, there is a different way.

To do a 1/N average, we do the computation $N = L(2^K)$. The $1/2^K$ will be done by the right shift by K bits and the last part of the normalization will be done by a multiply by $1/L$. Now, $1 < L < 2$ so $0.5 < 1/L < 1$, and we can represent $1/L$ as s1.17 two's complement number where the number looks like 0.1XXXXXXXXXXXXXXXX, where the "X" bits can be either 0 or 1. Our host computer can calculate $K$ and $1/L$ for each frequency and download it to the real-time system, which can normalize a large sum by first right shifting by $K$ bits and then by multiplying the resulting value by our 18 bit representation of $1/L$. Our stepped-sine integral has been turned into a single multiply and accumulate at each time step, followed by the fairly easy $1/N$ normalization just described.

## VII. Software Pre and Post Processing

The choice and tweaking of measurement frequencies to tile into integer numbers of sample periods is all done in a host computer, as is the calculation of $1/L$ and $K$ for the normalization. Once the integrals are collected on the real-time system, cross and auto spectra are computed and the entire process is repeated for the requisite number of averages. Thus, we estimate Equation 5 with

$$E\{Z(f)U^*(f)\} \approx \frac{1}{N_{avg}}\sum_{1}^{N_{avg}} Z_i(f)U_i^*(f), \quad (24)$$

for all signals of interest. Typically, we need $N_{avg} \geq 3$ for the coherence to have any relevance, but the stepped-sine is clean enough that $N_{avg} \leq 10$ usually works. This is in sharp contrast to FFT methods, where $N_{avg}$ can often be in the hundreds to try to restore SNR. While this is being done, a new frequency is measured. Once all frequencies have been measured in this way, the FRFs can be extracted from the averaged cross and auto spectra.

## VIII. Simulation and Measurement Results

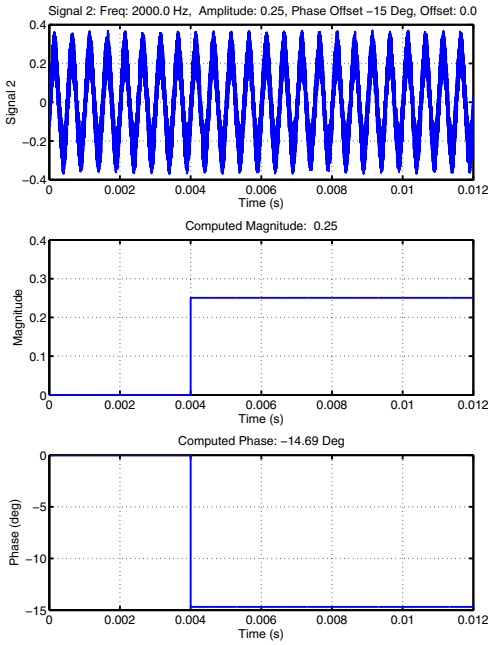


Fig. 6. Bit accurate simulation of FPGA stepped-sine integral. $f_0 = 2000Hz$, Amplitude = 0.25, and Phase Offset = $-15°$. Integration done for 8 periods of oscillation, after which result is extracted.

The FPGA portion of the algorithm was simulated in ModelSim. Two results are shown in Figure 6 and 7. The sample rate for both of these is 2 MHz, and the simulated amplitude is 0.25 while the phase is at $-15°$. The sines, at 2 kHz and 101 kHz, respectively, were corrupted by AGWN. The stepped-sine integral was run for 8 periods before faithfully returning the magnitude and phase. (These were extracted in the ModelSim testbench.)

Following the successful tests of the FPGA portion, it was integrated into a real-time digital controller while the software portion was implemented on a host computer. A closed-loop measurement of an nPoint N-XY30 stage is shown in Figure 8, while the extracted plant response is shown in Figure 9. Note that the NPXY30 has a much stiffer response than the NPXY100A of Figure 1. However, the main observation is the cleanliness of the FRF measurement.

## IX. Conclusions

This paper has demonstrated a stepped-sine algorithm that can be built into a real-time digital controller allowing large numbers of frequency response functions to be rapidly measured on real-time systems. The use of a built in measurement allows a digital patch panel between different measurement points, while the use of the controller processor matches the measured responses to what the controller will actually see. The algorithm is made so efficient that it can

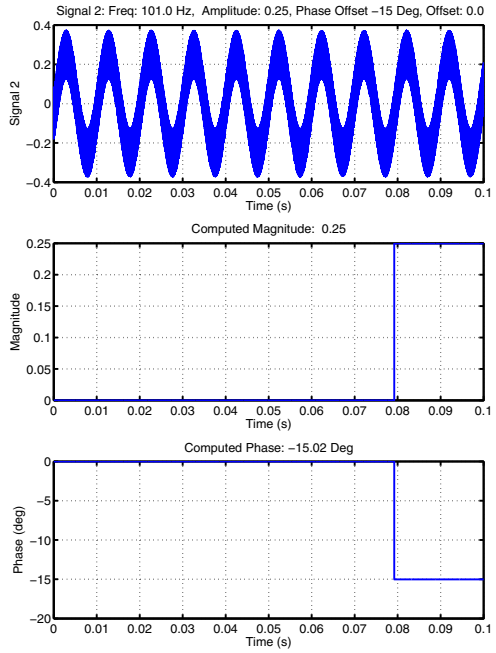


Fig. 7. Bit accurate simulation of FPGA stepped-sine integral. $f_0 = 101Hz$, Amplitude = 0.25, and Phase Offset = $-15°$. Integration done for 8 periods of oscillation, after which result is extracted.

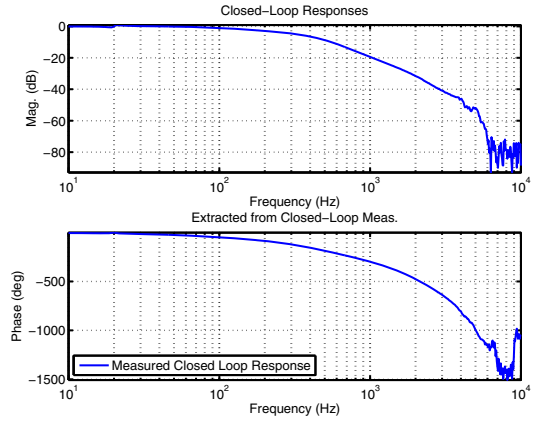


Fig. 8. Built in stepped-sine measurement of nPoint NPXY30 [10] x actuator in closed-loop.

easily run in parallel to the control action and yet provide high fidelity measurements. It represents a key component in doing system identification on systems with complicated dynamics, such as mechatronic systems [9].

## REFERENCES


[1] D. Y. Abramovitch, "A tale of three actuators: How mechanics, business models and position sensing affect different mechatronic servo problems," in *Proceedings of the 2009 American Control Conference*, (St. Louis, MO), pp. 3357–3371, AACC, IEEE, June 10-12 2009.

[2] L. Y. Pao and K. E. Johnson, "Control of wind turbines," *IEEE Control Systems Magazine*, vol. 31, pp. 44–62, April 2011.

[3] D. Y. Abramovitch, "The Multinotch, Part I: A low latency, high numerical fidelity filter for mechatronic control systems," in *Proceedings of the 2015 American Control Conference*, (Chicago, IL), pp. 2161–2166, AACC, IEEE, July 2015.

[4] D. Y. Abramovitch, "The Multinotch, Part II: Extra precision via $\Delta$ coefficients," in *Proceedings of the 2015 American Control Conference*, (Chicago, IL), pp. 4137–4142, AACC, IEEE, July 2015.


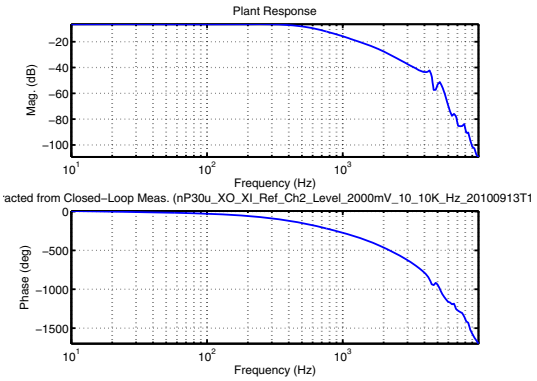


Fig. 9. nPoint NPXY30 x axis plant response extracted from measurement of Figure 8. DSA.


[5] L. Ljung, *System Identification: Theory for the User*. Prentice-Hall Information and System Sciences Series, Englewood Cliffs, New Jersey 07632: Prentice-Hall, 1987.

[6] L. Ljung and T. Glad, *Modeling of Dynamic Systems*. Upper Saddle River, NJ: Prentice Hall, 1994.

[7] R. C. Blackham, J. A. Vasil, E. S. Atkinson, and R. W. Potter, "Measurement modes and digital demodulation for a low-frequency analyzer," *Hewlett-Packard Journal*, vol. 38, pp. 17–25, January 1987.

[8] D. Y. Abramovitch, S. Hoen, and R. Workman, "Semi-automatic tuning of PID gains for atomic force microscopes," in *Proceedings of the 2008 American Control Conference*, (Seattle, WA), AACC, IEEE, June 11–13 2008.

[9] D. Y. Abramovitch, "Trying to keep it real: 25 years of trying to get the stuff I learned in grad school to work on mechatronic systems," in *Proceedings of the 2015 Multi-Conference on Systems and Control*, (Sydney, Australia), IEEE, IEEE, September 2015.

[10] nPoint, "Multi-axis stages," 2015. [Online; accessed February 5, 2015].

[11] F. Wang, D. Abramovitch, and G. Franklin, "A method for verifying measurements and models of linear and nonlinear systems," in *Proceedings of the 1993 American Control Conference*, (San Francisco, CA), pp. 93–97, AACC, IEEE, June 1993.

[12] D. Abramovitch, F. Wang, and G. Franklin, "Disk drive pivot nonlinearity modeling Part I: Frequency Domain," in *Proceedings of the 1994 American Control Conference*, (Baltimore, MD), pp. 2600–2603, AACC, IEEE, June 1994.

[13] J. Schoukens, R. M. Pintelon, and Y. J. Rolain, "Broadband versus stepped sine FRF measurements," *IEEE Transactions on Instrumentation and Measurement*, vol. 49, pp. 275–278, April 2000.

[14] D. Abramovitch, "The Banshee Multivariable Workstation: A tool for disk drive servo research," in *Proceedings of the ASME Winter Annual Meeting*, (Anaheim, CA), ASME, ASME, November 1992.

[15] D. Y. Abramovitch and C. P. Taussig, "Determination of open loop responses from closed loop measurements," United States Patent 5,446,648, Hewlett-Packard, Palo Alto, CA USA, August 1995.

[16] J. S. Bendat and A. G. Piersol, *Random Data: Analysis and Measurement Procedures*. New York, NY: John Wiley & Sons, second ed., 1986.

[17] Hewlett-Packard, *HP 3563A Control Systems Analyzer*, 1990.

[18] Wikipedia, "Lock-in amplifier," 2012. [Online; accessed December 30, 2014].

[19] D. Y. Abramovitch, "Low latency demodulation for atomic force microscopes, Part I: Efficient real-time integration," in *Proceedings of the 2011 American Control Conference*, (San Francisco, CA), AACC, IEEE, June 29–July 1 2011.

[20] D. Y. Abramovitch, "Low latency demodulation for atomic force microscopes, Part II: Efficient calculation of magnitude and phase," in *Proceedings of the IFAC 18th World Congress*, (Milan, Italy), IFAC, IFAC, August 28–September 2 2011.

[21] Xilinx, *7 Series DSP48E1 Slice Users Guide*, ug479 (v1.6) ed., August 7 2013.