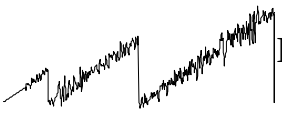
CU-NEES-06-5



NEES at CU Boulder
01000110 01001000 01010100
The George E Brown, Jr. Network for Earthquake Engineering Simulation

A Study of the Effects of Fixed-Point Arithmetic on Hybrid Simulation Accuracy

by

Dr. Gary Haussmann

Software Engineer

October 2006

Department of Civil Environmental and Architectural Engineering
University of Colorado
UCB 428
Boulder, Colorado 80309-0428

Abstract

As part of ongoing research at the CU-Boulder Fast Hybrid Testing laboratory, new methods are examined which can improve the hybrid testing process. One such method involves the use of a Field-Programmable Gate-Array (FPGA) which is capable of performing simulation computations on hundreds of degrees-of-freedom (DOFs), enabling the simulation of extremely complicated structures. While the current PC-based simulation platform used at CU-Boulder for FHT tests is capable of analyzing a structure with dozens of degrees-of-freedom at a real-time rate of 1024Hz, the FPGA can analyze many more degrees-of-freedom (hundreds) at a much higher rate (40MHz). The key issue with performing simulations on the FPGA is the lack of floating-point arithmetic on the FPGA, which has the potential to severely reduce the accuracy of a Fast Hybrid Test.

This report examines the degradation of accuracy that occurs when a hybrid simulation performs its measurement and computation using fixed-point arithmetic instead of the more common floating-point arithmetic typically used in simulation computation. The implementation of a fixed-point simulation is discussed, and the subsequent reduction in precision that occurs due to fixed-point rounding errors is quantified and compared against the more precise floating-point simulation. A calibration process is described for minimizing fixed-point error, and a comparison of this error with other errors in the hybrid simulation is provided.

Contents

1 Background and Motivation

This report examines the impact of fixed-point computation when use in a hybrid structural dynamics simulation. While the use of a Field-Programmable Gate-Array (FPGA) holds the potential for modeling much larger and complicated structures, the lack of floating-point computation on the FPGA needs to be addressed. Errors are introduced when the normal floating-point computations are replaced with related fixed-point arithmetic. This study aims to quantify fixed-point error, determine what effect various models and measurement methods have on the fixed-point error, and possibly determine what techniques can be used to mitigate the errors introduced by the introduction of fixed-point arithmetic. Specifically, we will determine what calibration process is required for the FPGA to produce the highest fidelity model for structural simulation, and fast hybrid simulation in particular. Also, the fixed-point error is compared to other sources of error in the hybrid simulation to aid in determining viable targets for improving accuracy.

1.1 Definitions and Concepts

Hybrid Simulation A hybrid simulation combines both numerical computation (a computer model) and a physical specimen. During a hybrid simulation, data from the computer model is used to influence the physical test specimen and vice-versa. Typically the computer is used to model parts of a structure that are impractical to build, while the physical specimen is used to represent components that are difficult to model correctly on a computer.

Field Programmable Gate-Array (FPGA) An FPGA is an integrated circuit device (chip) with internal circuitry that can be re-routed and re-programmed. In the same way that modern computer chips can run different software programs, the FPGA can be programmed with different digital configurations in order to mimic the behavior of various kinds of circuits. An FPGA programmed to perform structural simulation can perform computations many times faster than an equivalent computer CPU, stemming from the hardware-based nature of the FPGA. An FPGA limitation relevant to this report is that computations on an FPGA are performed using fixed-point arithmetic instead of floating-point.

Fixed-point Arithmetic Numerical computation on modern computing devices use *floating-point* arithmetic, which is able to accurately represent a large range of values, from the very large to the very small. However, floating-point computations use a relatively complicated data representation and require extra work for re-normalization after each computation. In contrast, a fixed-point computation is much simpler and faster than floating-point, but cannot represent as large of a range of values.

1.2 Motivation and Goals

A key advantage to using the FPGA for simulation computation is speed; while the FPGA clock speed is only 40MHz, it can perform a large number of simultaneous computations on each clock tick. In addition, the FPGA does not need to transfer data off-chip to RAM memory, thus avoiding the latency slowdown that a traditional CPU would see when modifying or updating digital data.

The primary disadvantage to using the FPGA is the loss in accuracy inherent in the use of fixed-point data quantities. The analog I/O on the FPGA board is limited to signed 16-bit quantities, and internal FPGA computation is typically done in fixed-point arithmetic. Nevertheless, with proper techniques an FPGA-based computation can possibly approach the accuracy of a traditional CPU-based computation.

In addition, in a hybrid simulation the interface to the physical specimen is also a key source of error. Data transferred between the computation model and the physical specimen suffers from errors due to both measurement quantization and nonideal actuator control. It

is important to quantify the error due to fixed-point computation so it can be compared to other measurement errors present in the hybrid simulation.

1.3 Process and Deliverables

The process of examining the effects of fixed-point math involves using an FPGA to perform the fixed-point calculations. The process of quantifying these effects proceeds in three steps, or phases:

- Phase One (Initial Testing and Validation) consists of verifying that basic functionality of the FPGA and the computer system that hosts and programs the FPGA. Functionality for this phase consists of properly accessing data on the SCRAMNet network and manipulating/displaying it via the FPGA. The SCRAMNet card is a deterministic network that enables real-time transfer of data between the simulation computer(s) and the instrumentation system that manipulates/measures the physical specimen. The SCRAMNet network, with its high bandwidth and guaranteed data delivery, is a vital part of fast-hybrid testing at CU-Boulder.
- Phase Two (Single DOF Simulation Model) is intended to provide a simple framework upon which various precision and accuracy tests can be performed. To this end, the FPGA will be programmed to properly model a single degree-of-freedom structural dynamics problem, similar to those used in the CU-Boulder Fast Hybrid Test (FHT) production system. The fixed-point simulation model will be examined for general correctness, but no quantifying of the error is done in this phase.
- Phase Three, the final phase, will consist of quantifying the precision problems inherent in fixed-point, and how they may be mitigated with normalization. Also, the overall simulation correctness will be quantified by performing and comparing two hybrid simulations, one with fixed-point and one with floating-point computations, using the same single degree-of-freedom model developed in Phase Two.

2 Initial Testing and Validation

The initial testing had three goals:

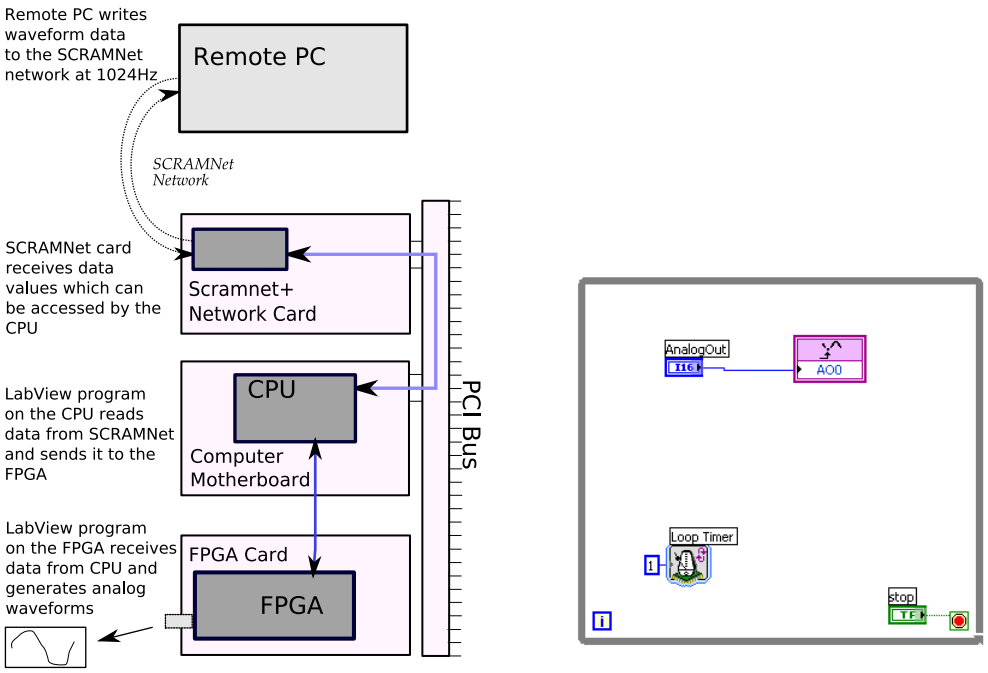
- Verifying the FPGA can be controlled and reprogrammed remotely from a computer running Microsoft Windows
- Verifying several simple LabView “programs” by running them on the FPGA and using them to produce analog outputs on the FPGA I/O lines.
- Verify that the FPGA can access data values on the ScramNET network, by controlling an external voltage based on values from the ScramNET network.

The deliverable of this inital (Phase One) test was a completed system that used the FPGA to generate analog waveforms based on data read from the SCRAMNet network–see figure 1. The waveform is generated on a separate computer connected to the SCRAMNet network, and the FPGA analog output is verified for correctness via oscilloscope.

In this case, the FPGA is programmed with minimal logic. The FPGA is configured to accept a fixed-point value from the CPU and generate an analog voltage output that can be measured (figure 1(b)).

2.1 Results

After implementing the described components, a test run was performed by generating a sinusoid waveform on the remote PC platform and writing waveform values to the SCRAMNet. Waveform samples were written at a rate of 1024 samples/second, corresponding too the standard sample rate of data produced by the MTS measurement system. The writing



(a) Data Flow for Phase One

(b) FPGA Logic for Phase One

Figure 1: Data flow and logic for Phase One Electronics

of data generates an interrupt on the SCRAMNet network which is picked up by the CPU, triggering it to read the SCRAMNet data and transfer that data to the FPGA.

The CPU/FPGA combination was observed to correctly read data from the SCRAMNet and generated a sinusoidal voltage waveform on the FPGA analog output, as observed by an oscilloscope. Further stress-testing of the data rate was then done by programming another computer on the SCRAMNet network to inject data at a rate faster than 1024Hz. Rates up to 1MHz were observed to perform correctly; faster sample rates could not be verified since the analog outputs from the FPGA card cannot run any faster than 1 million samples/second.

In the interest of performance testing, further tests were done to determine how fast data values can be transferred across the SCRAMNet network and relayed to the FPGA. This was done by increasing the rate of the waveform samples written to SCRAMNet, and increasing processing intervals in appropriate parts of the CPU and FPGA logic. As a result, we succeeded in transferring up to 20,000 data samples per second from the remote PC system, which the FPGA used to synthesize waveforms. The limiting factor at this point was the speed at which the remote PC could generate and write data values to the SCRAMNet network. Theoretical peak performance of the FPGA is processing one sample per clock tick, or 40 million samples/second; however, once more complicated logic is programmed into the FPGA it will conceivably require more than one clock tick to process and analyze the data.

3 Single DOF Computation Model

The second phase consisted of constructing a single degree-of-freedom computational model. This model will be used as the example simulation which will be normalized and reformulated to demonstrate improved accuracy. The model used is that of a single degree-of-freemdom spring/mass/damper system, illustrated in figure 2. The computational sim-

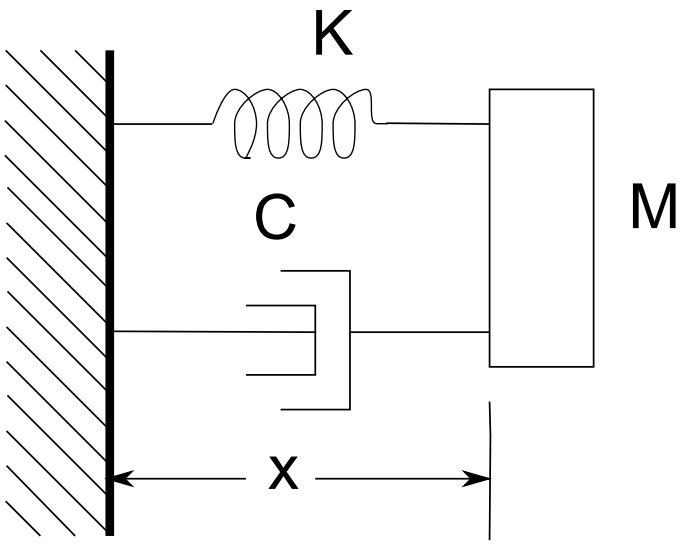


Figure 2: Single Degree-of-freedom Model Used in Phase Two Computations

ulation is performed purely using fixed-point mathematics; the fractional portion of the fixed-point representation is varied to observe its effect on the simulation accuracy.

The equation of motion for this system is

$$M\ddot{x} + C\dot{x} + Kx = f(t) \tag{1}$$

where M, K, and C are constants and $f(t)$ is set to zero. The initial conditions are $x(0) = 1, \dot{x}(0) = 0$. Solving for x, the closed-form solution is that of a damped oscillator, with the form of the solution depending on the value of the damping ratio $\zeta = \frac{C}{2\sqrt{KM}}$:

$$\begin{aligned} x(t) = &\quad e^{-\gamma t}(c_1 cos(\omega t) + c_2 sin(\omega t)) && \text{for } \zeta < 1 \\ x(t) = &\quad c_1 e^{-\gamma t} + c_2 t e^{-\gamma t} && \text{for } \zeta = 1 \\ x(t) = &\quad c_1 e^{-\gamma_1 t} + c_2 e^{-\gamma_2 t} && \text{for } \zeta > 1 \end{aligned}$$

where we define the constants

$$\begin{aligned} \gamma &= \frac{C}{2M} \\ \omega_0 &= \sqrt{K/M} \\ \omega &= \sqrt{\omega_0^2 - \gamma^2} \\ \gamma_1 &= \gamma + \omega \\ \gamma_2 &= \gamma - \omega \end{aligned}$$

A primary source of error with the use of fixed-point is the limited dynamic range; the allowed ratio of highest to lowest values is on the order of 10^-3. For example, if 10 is the largest possible value represented in a three-digit fixed-point representation, then any value smaller than 0.01 is considered to be zero in fixed-point. By contrast, a floating-point representation could easily handle values much smaller, down to at least 10^{-5} and possibly

smaller values. The limited dynamic range poses a problem since explicit integration methods typically need a very small value of Δt in order to remain stable. As an example, the update equation for the central-difference scheme:

$$d_{i+1} = \frac{\Delta t^2 (f_i - r_i) + 2M d_i - \left(M + \frac{\Delta t}{2} C\right) d_{i-1}}{M + \frac{\delta t}{2} C} \tag{2}$$

contains a Δt^2 term applied to the acceleration. For a value of $\Delta t = 0.01$, the size of the acceleration is thus scaled by $\Delta t^2 = 0.0001$, which requires at least 10 fractional bits in a fixed-point representation. If the value of $f - Kx$ is less than one–a fairly common occurrence–then the acceleration quantity will be rounded down to zero, and thus any acceleration that would have occurred on that timestep is lost.

A possible solution to the rounding problem for small Δt is to store different state variables during the iterative solution procedure. In the above formulation (equation 2) the state variables are d_{i+1}, d_i, and d_{i-1}; another possible method uses different state variables, namely d_i, d_{i-1}, and v_i. This "velocity-based" formulation has the advantage of avoiding the use of Δt^2 which is the primary source of rounding error in the displacement-based method shown in equation 2. A velocity-based formulation computes both displacement and velocity each timestep:

$$v_i = \frac{\frac{d_i - d_{i-1}}{\Delta t} + \frac{\Delta t}{2M}(f_i - r_i)}{1 + \frac{\Delta t C}{2}} \tag{3}$$

$$d_{i+1} = d_{i-1} + 2\ \Delta t\ v_i \tag{4}$$

This velocity-based formulation apparently suffers from fewer rounding problems: with equivalent M, K, and C values the velocity-based version will produce a proper oscillation where the displacement-based simulation produces only zeros (figure 3). Due to the improved results produced by the velocity-based formulation, tests in phase three use the velocity-based formulations for quantifying simulation error.

4 Quantifying Hybrid Simulation Errors

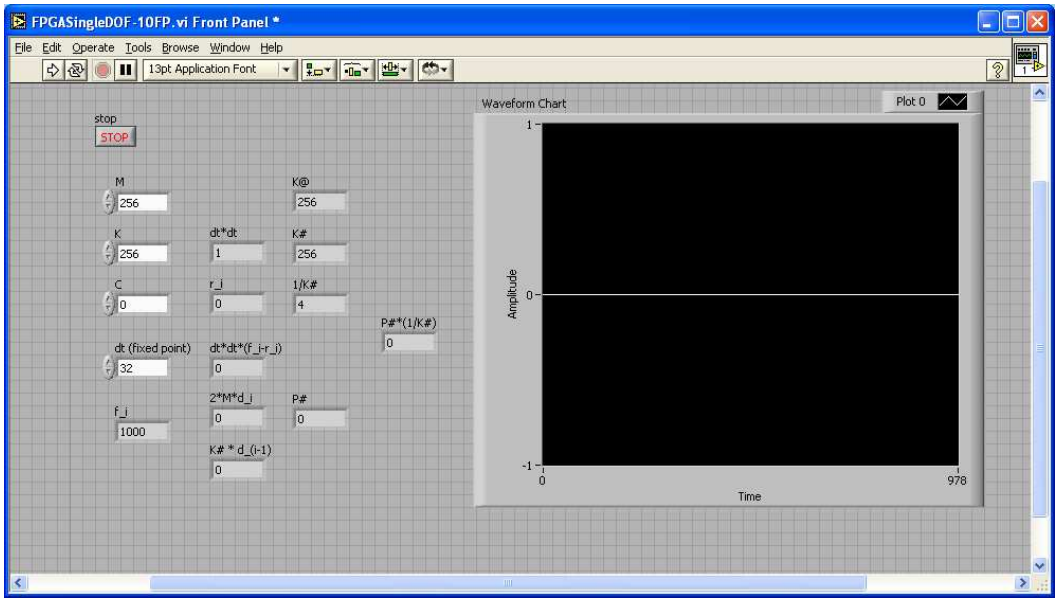
In an effort to measure the errors generated due to fixed-point error, a single degree-of-freedom simulation was performed and compared to the ideal analytical solution. The system used for this comparison is the mass/spring/damper system show in figure 2.

Graphs comparing the simulation results and floating-point results are shown in figures 4, 5, and 6. In all cases the fixed-point simulation shows good accuracy, capturing both the overall behavior and small-scale details that are seen in the equivalent floating-point simulation. As the model displacement and velocity damp down to values near zero, rounding errors begin to effect the fixed-point simulation causing it to diverge from the floating-point simulation.

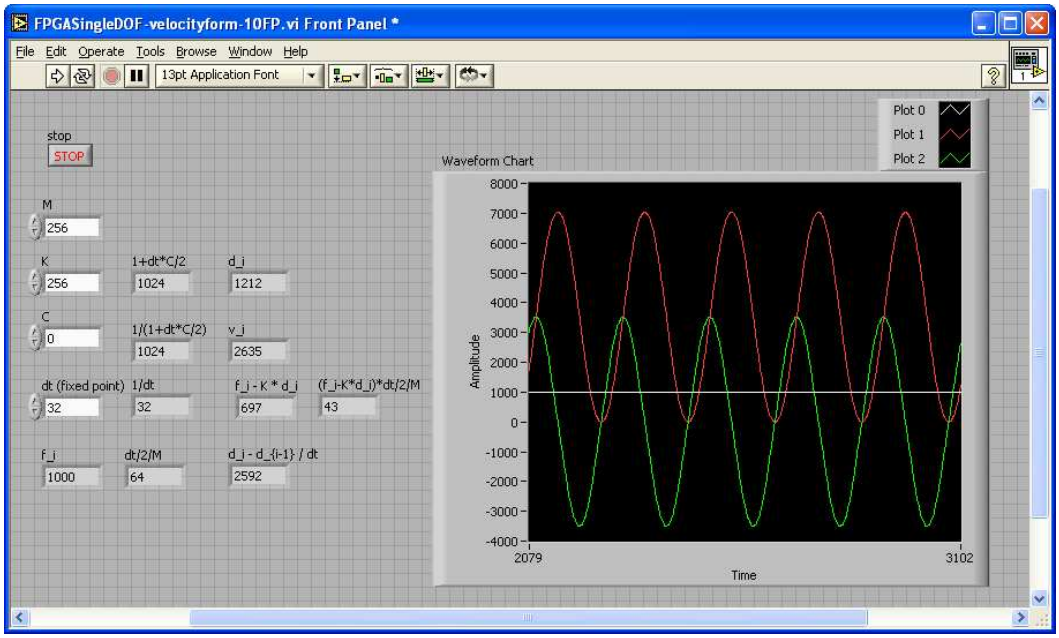
Finally, figure 7 shows the error of the fixed-point simulation by considering the floating-point solution as the "exact" solution. The average error of the fixed-point simulation for the various system types is illustrated in table 1. In general, the average error is of the same order as the rounding error of single-precision floating-point numbers, although the maximum error is much larger than this quantity. Also, the error in later parts of the simulation appear systemic, converging toward a single quantity (-0.0005) for all system types, which can be traced and corrected.

5 Calibration Process for Accurate Hybrid Simulation

The calibration process is intended to reduce error introduced in to the hybrid simulation due to the inaccuracy of fixed-point computation. Based on the above computation results,



(a) Displacement-based simulation



(b) Equivalent Velocity-based simulation

Figure 3: Comparison of displacement- and velocity-based fixed-point simulations

System Type	M	K	C	Average Error
Underdamped	1	1	0.5	2.150×10^{-6}
Critically damped	1	1	2	1.80×10^{-6}
Overdamped	1	1	4	1.67×10^{-6}

Table 1: Average Error For Different Oscillator System Types

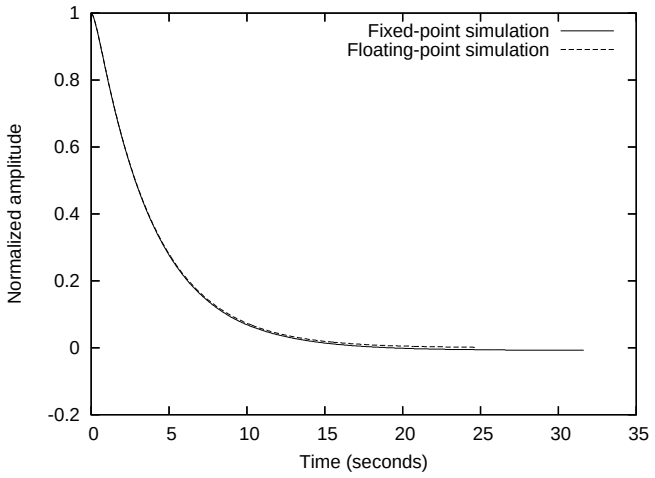


Figure 4: Overdamped system: fixed-point computation vs. floating-point solution

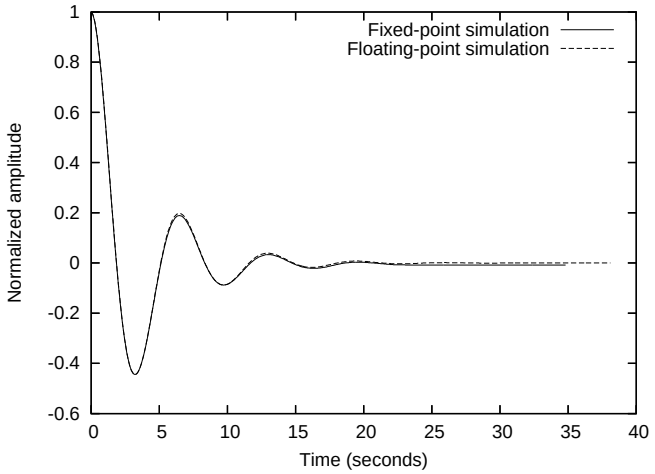


Figure 5: Underdamped system: fixed-point computation vs. floating-point solution

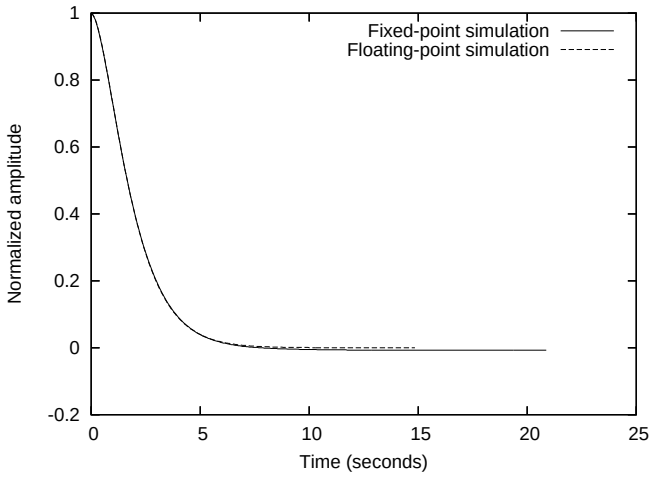


Figure 6: Critically damped system: fixed-point computation vs. floating-point solution

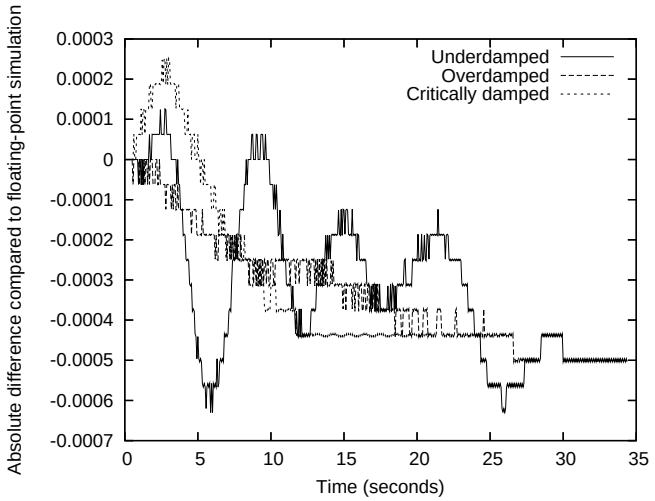


Figure 7: Error values vs. time for fixed-point simulation of different system types

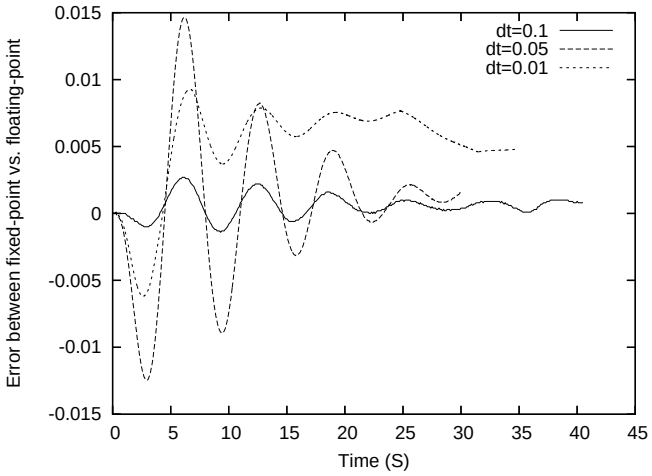


Figure 8: Fixed-point simulation error values vs. time for various values of Δt

the calibration process i dependent on the model and physical specimen used in the hybrid simulation.

5.1 Choice of Time Step

The value of Δt has an impact because it limits the dynamic range of values in the simulation, due to its presence as both a multiplier and divisor in equation 2. The dynamic range is reduced as Δt becomes smaller or larger than one, thus Δt should be chosen as close to one as possible. However, the value of Δt is predefined in many hybrid simulations, and is typically tied to the timing constraints of instrumentation used. Thus, the limit imposed by choice Δt cannot typically be mitigated due to physical constraints. Figure 8 shows the errors produced for the single-DOF simulation when different values of Δt are used; the error as the value of Δt decreases.

5.2 Choice of constants K and M

The other primary determinant are the values of K and M; like Δt, the values of K and M reduce dynamic range when they are much smaller or larger than unity. The range reduction can be mitigated by scaling the displacements with ω_0; in this case, the quantities $(d_i \cdot \omega_0)$ are computed and stored instead of d_i. As a result, the dynamic range is limited by $\sqrt{K/M}$ instead of K and M; however, even in this case the dynamic range will still be reduced. The renormalization of K and M to make them closer to one by appropriate choice of units will still produce a dynamic range larger than simply scaling by ω_0. An example of unnormalized K and M is shown in figure 9 where the values of K and M are made progressively larger and eventually reach the fixed-point range limit.

5.3 Choice of fixed-point fractional bits

In any fixed-point simulation, the representation of numbers requires bits to represent both the fractional and integer portions. In any representation, using more bits for the fractional part will result in less bits being available for the integer part, and vice-versa. In order to allow for the most bits to be available to the fractional part, the allowed integer range should be made as small as possible. The number of bits needed for the integer part is related to the maximum integer by

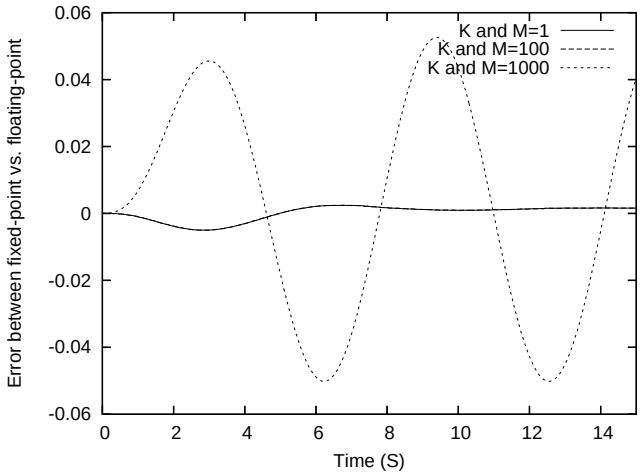


Figure 9: Fixed-point simulation error values vs. time for various values of K and M

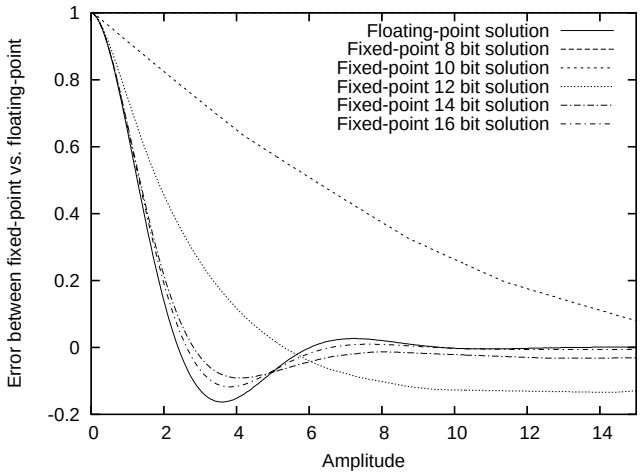


Figure 10: Fixed-point simulation compared to floating-point for various fractional bit sizes

$$bits = \log_2(max\ int) \tag{5}$$

Thus, the maximum required integer value must be predicted via simulation or other method and use to determine the number of bits required for the integer portion of the fixed-point representation. Figure 10 shows the change in simulation results as the number of fractional bits is changed.

6 Computation Speed and Limits

As part of the FPGA investigate research the performance limits of the FPGA were examined to determine their capabilities. The capabilities here are examined in two parts: computation speed and model size.

6.1 Computation Speed

For the FPGA, computational speed is primarily dependent on signal delay. As data from one operation (an add, subtract, or multiply) is produced, it takes some non-zero amount of time for that data to travel to the next piece of logic which then performs some other operation on the data. The delay is dependent on several factors, such as the interconnect used and the distance that the signal must travel on the chip.

For the hybrid model performed here, the relevant delay is the delay between when a new state d_{i+1}, v_{i+1} is computed from the previous state d_i, v_i. The "compiler" that generates logic configurations for the FPGA can estimate the worst-case delay and report it. For a single-DOF fixed-point system, the delay for the main loop is 23.740nS, which allows for a maximum theoretical update rate of 40.42MHz. However, this maximum speed is reduced by instrumentation I/O; the on-board I/O pins have a settle time of 1uS allowing for maximum update rate of 1MHz, and the inherent delays in SCRAMnet are in the hundreds of nS, limiting the update rate to the hundreds of kHz.

Also, is is important to note that modal analysis of linear elastic systems results in models where the modes are fully decoupled. In such an analysis with fully decoupled modes, each mode is totally independent and the system is fully parallelizable. This full parallelization means that adding more modes to the model neither increases the delay nor slows down the maximum update rate.

6.2 Logic Real Estate

Although speed is not a large constraint on the size of models that can be represented, the amount of available chip space is certainly a constraint. Each additional degree of freedom requires logic gates and wiring on the FPGA, which consumes space or "real estate" on the FPGA. Thus, the limit to larger models occurs when there is no more space available on the FPGA. Possible solutions include using a larger FPGA, breaking the model into smaller parts, and updating multiple DOFs with each operator instead of just one.

To quantify this restriction somewhat, several models were built with varying numbers of DOFs, and the percentage of chip space used was recorded. These numbers are shown in table 2; projections suggest that the XC2V3000 FPGA can update several hundred degrees-of-freedom 40 million times per second.

DOFs	Logic Blocks	Percent of Chip
4	2340	16
16	2908	20
24	3036	21
32	3224	22
48	3378	23
64	4136	28
Projected sizes:		
200	7614	53
400	12884	90
450	14201	99

Table 2: FPGA Chip Space Used For Various Simulation Sizes (Degrees-of-Freedom). The larger DOF sizes are extrapolated from smaller sizes. The FPGA used here is the XC2V3000, a Xilinx Virtex 2 with 3M available logic blocks

It should be noted that these numbers are for a basic simulation model where the various DOFs are fully decoupled and all responses are linear, which is typical for modal analysis of

linear elastic models. The amount of chip space used by each DOF is greater if either inter-DOF coupling or nonlinear responses are added, which would reduce the overall number of DOFs the the FPGA could support at any one time.

7 Conclusion

This report has examined the effect of using fixed-point arithmetic for the simulation portion of a Fast Hybrid Test. It is clearly evident that the use of an FPGA using fixed-point arithmetic provides a dramatic increase in computation speed and size compared to that of a standard CPU.

The impact of fixed-point computation on simulation accuracy is shown to be no worse than standard floating-point if a proper number of fractional bits is used, and the numeric quantities are kept near unity. However, for particularly large or small number values the fixed-point representation produces a large error amount that can cripple overall test accuracy if not corrected or mitigated somehow. Typical methods of correcting this error involve re-scaling the simulation constants so that simulation values are closer to one.

7.1 Acknowledgement

This metrology effort is supported by the NEES Consortium, Inc. and National Science Foundation.

Appendix: Parameters and Configuration Used For Simulation

For the simulation comparing the effect of changing Δt the model was a single-DOF dynamics system with the following parameters:

$$\begin{aligned} \Delta t &= 0.1,\ 0.05,\ 0.01 \\ K &= 1 \\ M &= 1 \\ C &= 0.3 \\ f(t) &= 0 \\ x(0) &= 1 \\ \dot{x}(0) &= 0 \\ \text{bits} &= 16 \end{aligned}$$

The simulation comparing the effect of changing K and M used a model of a single-DOF dynamics system with the following parameters:

$$\begin{aligned} K &= 1,\ 100,\ 1000 \\ M &= 1,\ 100,\ 1000 \\ C &= 1,\ 100,\ 1000 \\ f(t) &= 0 \\ x(0) &= 1 \\ \dot{x}(0) &= 0 \\ \Delta t &= 0.05 \\ \text{bits} &= 16 \end{aligned}$$

For the simulation comparing the effect of changing the number of fractional bits, the model was a single-DOF dynamics system with the following parameters:

$$\begin{aligned} \text{bits} &= 8,\ 10,\ 12,\ 14,\ 16 \\ K &= 1 \\ M &= 1 \\ C &= 1 \\ f(t) &= 0 \\ x(0) &= 1 \\ \dot{x}(0) &= 0 \\ \Delta t &= 0.05 \end{aligned}$$