Controller Fixed-Point Optimization with Genetic Algorithms

Heiko Wolfram
Wolfram Control Engineering & Consulting
Gera, Germany
<heiko.wolfram@gmx.de>

Abstract – **This work describes a way to optimize the controller fixed-point representation in programmable logic devices (eg. FPGA) with genetic algorithms. The optimization uses the error between floating-point and fixed-point representation as well as a quantization noise error model. Thus, both terms allow weighting between the to be expected theoretical and actually occurred simulation error. This task could be automated easily due to the script features of the simulation system.**

Fixed-Point; Simulation; Quantization; Optimization; Genetic Algorithm; FPGA

I. Introduction

The controller design and simulation is typically based on floating point representation (infinite precision). The implementation of the control algorithm in the "real world" target hardware usually requires a fixed point representation with finite precision. However, this representation is bounded due to the limited resources in programmable devices. Or the word length is fixed by the arithmetic logic unit (ALU) in digital signal processors (DSP).

The accuracy within DSP implementations is determined by the maximum required integer number range of the signals. An automated conversion into fixed-point arithmetic is therefore easily possible [1, 2]. The word length for FPGA applications is not fixed unlike DSP implementations. There cannot be a compromise between the maximum number representation and accuracy. Thus, a trade-off is necessary between system noise and the word length, which represents the FPGA load.

Digital filters applied with test signals in open-loop operation have been studied in several publications for example in [3–8]. Analysis methods based on affine arithmetic and interval arithmetic were used in [9, 10] to examine range and precision in DSP applications, whereas [11] combines both, analysis methods and simulations to reduce overestimation of word length and therefore reduce implementation costs.

Regulators are considered for fixed-point optimization in this paper. They usually have an integral part and cannot be easily analyzed in open-loop configuration. Furthermore, the complete loop including plant has an impact on the resulting system performance. Thus, the regulator as a kind of digital filter cannot be analyzed as a stand-alone unit. The analysis must therefore be made in closed loop, but this makes the signals statistically dependent.

The proposed optimization in this paper consists of two parts to minimize these effects. These parts shall be determined by two independent simulation runs. The first step determines the error between floating point and fixed-point simulation independent of the occurring signal correlation. Resulting quantization effects might have rigorous effects for the whole system (e. g. oscillations or equilibrium). A simple noise model is used in the second step to avoid statistical dependence. The result of this model should therefore address an additional weighted term in the optimization.

An evolutionary algorithm, which is inspired by natural selection, is particularly suitable due to the non-convex nature of this optimization problem.

II. Some Basics

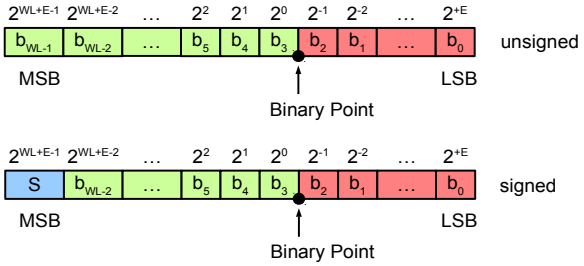
A. Number Representation

The fixed-point representation consists of a 3-tuple word length WL, fraction length E and the sign S

$$\langle S, WL, E \rangle \quad , \tag{1}$$

where S of the signed representation enters the word length and thus reduces the available number range (Figure 1).

Figure 1. Fixed Point Representation (Binary Point shown for E<0)



The number range for signed and unsigned representation results to

$$\begin{aligned} &\text{unsigned:} \quad 0 \ldots (2^{WL}-1)2^{E} \\ &\text{signed:} \quad -(2^{WL-1})2^{E} \ldots (2^{WL-1}-1)2^{E} \end{aligned} \quad . \tag{2}$$

The word length of the integer portion $WL_i = WL+E$ can be calculated very easily from their maximum value. For an unsigned number, this is

ISBN 978-80-261-0386-8, © University of West Bohemia, 2015

$$WL_i^{us} = \lceil \log_2(x+1) \rceil \,, \quad x \geq 0 \quad , \tag{3}$$

whereas $\lfloor\,\rfloor$ represents the floor and $\lceil\,\rceil$ the ceil operator. Similarly, the integer word length for a signed number is obtained to

$$WL_i^s = \begin{cases} \lceil \log_2(|x|+1)+1 \rceil & \forall\, x \geq 0 \\ \lceil \log_2|x|+1 \rceil & \forall\, x < 0 \end{cases} \quad , \tag{4}$$

where the bit for the sign must be added and the correction term for negative numbers is omitted.

The reciprocal value can be used for numbers less than one

$$WL + E - c = -\left\lfloor \log_2 \left| \frac{1}{x} \right| \right\rfloor \,, \quad |x| \leq 1$$
$$\text{unsigned: } c = 1 \;;\; \text{signed: } c = 2 \tag{5}$$

to make a statement about the relationship between WL and E. Only the word length decides about accuracy and the resulting number range.

B. *Evolutionary System Optimization*

Evolutionary algorithms (EA) are a class of optimization methods, which are inspired by the natural evolution of living beings. Solution sets are developed artificially across generations for a particular problem based on natural selection. They run through similar processes as in the real world:

- *Recombination:* Distribution and rearrangement of DNA
- *Mutation:* Random variation of the genome
- *Selection:* Selection of the population with the best "fitness"

Advantage of this natural like optimization method is a satisfactory solution to very complex search spaces. Disadvantage is the slow convergence behavior and resulting calculation time because the presented optimization is based on system simulations.

III. ALGORITHM

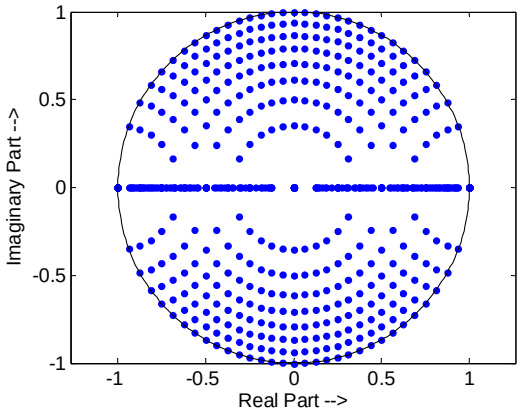
The conversion of a floating-point into a fixed-point model is very complex and should be done in three steps. Controller parameters have no share on system noise and are only responsible for the performance and system stability. A large class of signals, but at least the interfaces to the analog world, are predetermined and others can be further derived. Integer signal widths are obtained from the maximal value range. Optimization is necessary only for the rest of signals. Thus, the fixed-point transformation shall be divided into:

- parameter quantization;
- specify from predetermined signals the
  - integer signal width, sign and
  - fraction length, if possible;
- system optimization of the remaining signals.

A. *Parameter Quantization*

Regulators are generated by any of the direct form filter structures in the simplest case. The resulting pole/zero (P/Z) locations can have significant parameter quantization errors as shown in Figure 2 by the roots of a 2nd-order polynomial.

Figure 2: Second Order Filter Roots with Binary Point at -3



The sensitivity may be found very easily from polar coordinate transformation

$$\begin{aligned} P(z) &= 1 + a_1 z^{-1} + a_2 z^{-2} \\ &= (1 - p_1 z^{-1})(1 - p_2 z^{-1}) \\ &= 1 - 2r\cos\Theta\, z^{-1} + r^2 z^{-2} \end{aligned} \tag{6}$$

and their solutions

$$\begin{aligned} p_{1/2} &= r e^{\pm j\Theta} = r\cos\Theta \pm \mathrm{j} r \sin\Theta \\ &= \alpha \pm \mathrm{j}\beta = -\frac{a_1}{2} \pm \mathrm{j}\sqrt{a_2 - \frac{a_1^2}{4}} \quad \forall \quad a_2 > \frac{a_1^2}{4} \end{aligned} \tag{7}$$

using the identity

$$\beta^2 = a_2 - \frac{a_1^2}{4} = r^2 - \alpha^2 \quad . \tag{8}$$

The sensitivity of P/Z-locations derives to

$$\Delta p_i = \sum_{k=1}^{2} \frac{\partial p_i}{\partial a_k} \Delta a_k \;;\; i = 1,2$$
$$\text{with} \quad \frac{\partial p_i}{\partial a_1} = -\frac{1}{2} \mp \mathrm{j}\frac{a_1}{4\beta} \;;\; \frac{\partial p_i}{\partial a_2} = \pm \mathrm{j}\frac{1}{2\beta} \tag{9}$$

Thus for a 2nd order polynomial can be stated that

- a linear error dependency for the real part,

- a 1/β-dependence of the imaginary error,
- or a 1/r-dependence directly on the imaginary axis

exists. The linear error dependence of the real part is only determined by Δa_1. This results to a lowest root error location, which is close to the unit circle and close to the imaginary axis.

However, the sensitivity of the P/Z-locations with respect to parameter changes is less important for usage in regulators, but rather the limit to the instability, i. e. the distance to the unit circle.

Therefore, a fraction length is proposed for initial parameter conversion, which adds at least another node upward directed to the unit circle

$$E \geq \left\lfloor \log_2 \frac{1}{1-r} + 1 \right\rfloor \quad , \tag{10}$$

i. e. the smaller the distance between roots and unit circle, the smaller the parameter quantization has to be. This proposal is limited only to stable poles and zeros.

Not content of this article is a possible filter structure transformation and/or filter splitting into second order stages (SOS). Here it will be referred to relevant literature, e. g. [12, 13].

B. Determination of Integer Signal Width and Fraction Length

The determination of integer signal width WL_i is very easy using the determined min/max values from simulation results. This was already shown in section II.A. Simulation results should be multiplied by a correction factor to obtain sufficient margin to the number range limit.

Input and output signals to the "real world" are usually defined in technical systems. Restrictions can be found from the specification. Examples are the bit widths of A/D and D/A converters or PWM stages. Technical restrictions are for example limits in loads and moments, pressures, maximum travel ranges, power or current consumptions.

Therefore, a significant number of internal signals can be solely described by the specified interfaces. Other word and fraction lengths can be determined by forward and/or back propagation. For instance in case of separate controller terms (P, I, D) simple guidelines can be applied:

- The output quantization and proportional gain determine the input quantization, or vice versa.
- The input quantization and integrator gain determine the quantization of the accumulator.
- The input quantization and differential gain determine the sensitivity of the differentiator.

Unfortunately, not all signals can be determined by such simple considerations. The system optimization shall be used for the rest of it.

C. System Optimization

The system optimization shall consist of two steps. The error between floating point and fixed-point is determined in the first step. The error of the noise model is derived in the second step.

The weighting function J to be minimized defines the sum of all fraction lengths E. The error of both simulations **e** must fit inequality constraints (IEC) and has to lie within an error bound **ε**

$$\begin{aligned} J &= \min \sum -E \\ \text{IEC:}\ \ \boldsymbol{\varepsilon} &> \boldsymbol{W}\,|\boldsymbol{e}| \end{aligned} \quad . \tag{11}$$

The weighting matrix **W** can be used to prioritize or sum the error.

1) Error Models

a) Truncation

The truncation model determines the calculation error between floating-point and fixed-point representation. However, the signals can be quite different at various input values due to the correlation of the quantization error signals. Therefore the system should be constantly stimulated for possible reduction of these dependencies.

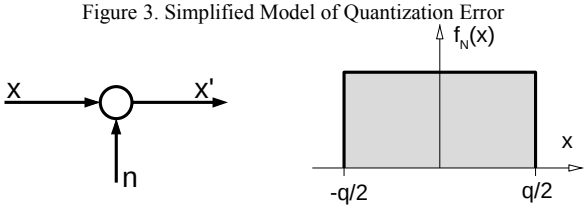
b) Noise Model

A noise model is used in a second step to prevent this correlation. The additive quantization error n in Figure 3 is assumed to be a statistically independent, bias-free, uniformly distributed white noise

$$n \in \left(-\frac{q}{2}, \frac{q}{2} \right]; \ \bar{n} = 0; \ \bar{n^2} = \frac{q^2}{12} \quad . \tag{12}$$

This simplified model is independent of the operational area and should achieve identical solutions in each operating point.

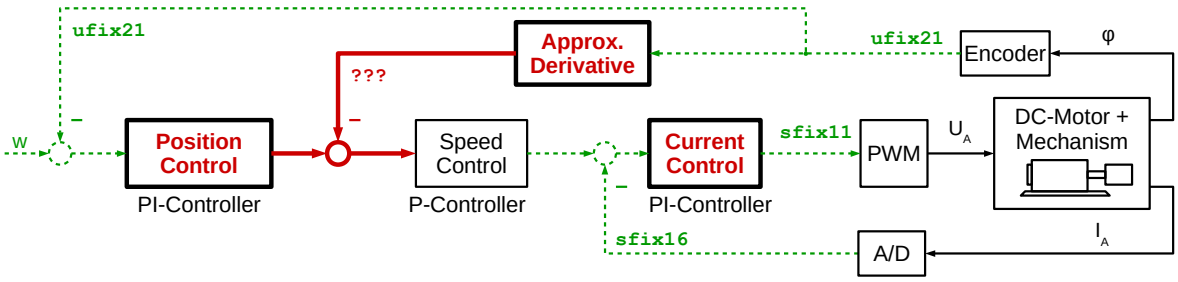
Figure 3. Simplified Model of Quantization Error



The error e is defined as the standard deviation of all relevant (output) signals.

A solution to this problem is analytically possible for linear time-invariant (LTI) systems. All relevant transfer functions must be defined, but this requires a wide knowledge of control engineering and is very time consuming by design. In contrast to – System simulation needs beforehand very less amount of time, but is time-consuming for the actual optimization.

Figure 4. Example



IV. EXAMPLE

The applicability of the evolutionary system optimization shall be investigated by an example of a cascaded position control for an electromechanical system shown in Figure 4. Cascade structures are usually used for motor control in industrial applications, e. g. in machine tools. The special PI-P-PI cascade is employed to track a ramp function w without position slack, used for rotational high precision continuous raster scanning. The unknown bit widths are marked in thick and known signals in dashed lines. The EA has to determine the accumulator fraction length of the position and current regulator and of the approximate derivative block as well as the fraction length of the speed signal.

Known signals are:

- the PWM input of signed 11 bit,
- the encoder bit width of 21 bit,
- and the A/D current signal of signed 16 bit

which induce the same related lengths on set signal w, position control input, approximate derivative input, speed control output and current control input.

The simulation is based on double precision. Corresponding floor blocks for truncation or uniform noise blocks are integrated to induce the related quantization noise in each step. Additionally, the time is monitored in order to respond to singularities during the simulation. It is not task of the optimization to define the word length and quantize the parameters.

The inequality constraint only includes the position error in the quasi-static simulation area without weighting. The following limits are set:

Truncation model: $\max|e| \le 1\,\mathrm{LSB}$

Noise model: $\mathrm{std}\, e \le 1/3\,\mathrm{LSB}$

Both models are not coupled to each other in order to make the results comparable.

A. Results

Each simulation lasted approximately about 20 seconds on an Intel Core™ Duo processor T2400 (2 MB Cache, 1.83 GHz, 667 MHz FSB). A few hundred of them were used for optimization.

The EA with inequality constraints (11) has shown to be inefficient for system optimization. Instead, the constraints were added as additive penalty terms into the fitness function

$$J = \min \sum -\boldsymbol{E} + k_1 \mathrm{IEC} + k_2 \mathrm{ERR} \quad \text{where}$$
$$\mathrm{IEC} = \left\{ \sum \mathbf{f}(\boldsymbol{W}\,\boldsymbol{e}) : \boldsymbol{W}|\boldsymbol{e}| \geq \boldsymbol{\varepsilon} \right\} \qquad (13)$$
$$\mathrm{ERR} = \{1 : \text{Simulation failed}\}$$

Factor k_1 adds the inequality constraints and k_2 a non-successful simulation.

The optimization results are shown in Table 1 and Figure 5. It can be seen that the values for both models lie in similar ranges.

TABLE 1: OPTIMIZATION RESULTS

		Truncation	Noise
Fitness		5	2
Inequality Constraints		1	0.28665
-E Accumulator [bit]	***Position***	9	6
	Speed	—	—
	Current	1	2
	Differentiator	0	0
-E Speed signal [bit]		-5	-6

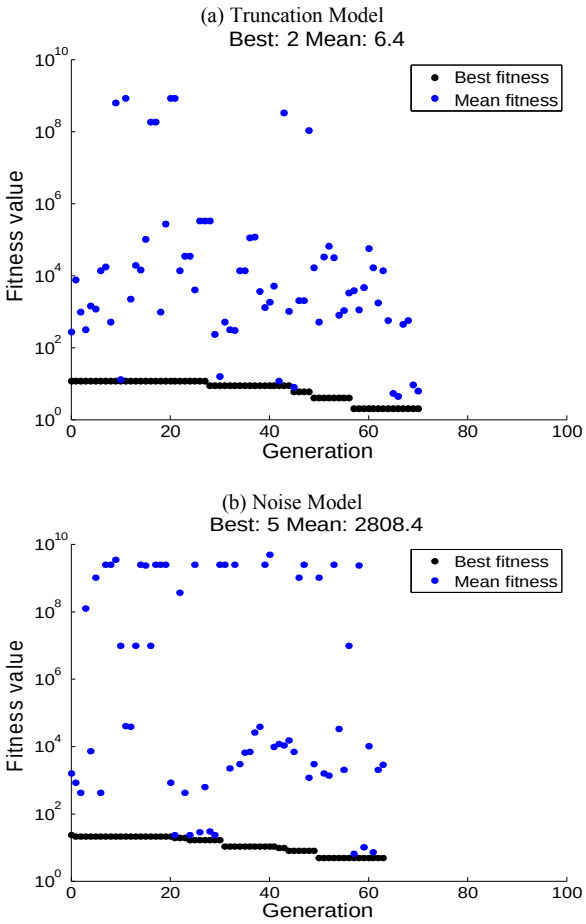
A cross comparison of the results is shown in Table 2. It turns out that the result of the noise model does not reach the desired fitness in the truncation model. The proposed linear combination of the two error models prevents a unilateral solution – Both models must fulfill their respective inequality constraints.

TABLE 2: CROSS COMPARISON OPTIMIZATION RESULTS

Model	Fitness	Inequality Constraints
Noise result into truncation	3002	3
Truncation result into noise model	5	0.22981

The gotten optimization result of the velocity signal is very interesting. It shows a large quantization interval of 32 LSB, but one would only expect a fraction length of $E \le 1$ for the velocity controller P-part of 0.635 by back-propagation.

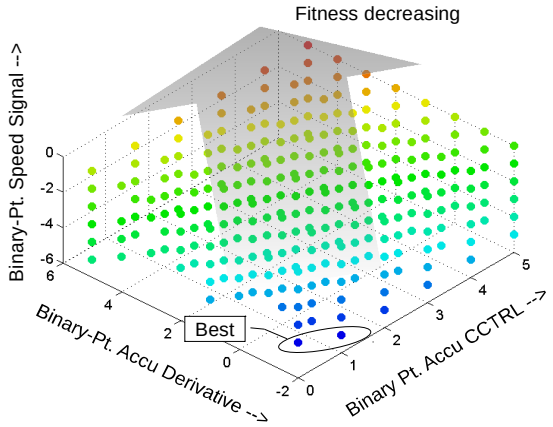
Figure 5. Simulation Results



Unfortunately both models show dependencies within different operating points. This has been confirmed by further simulations. The reason for this could not be finally clarified, possibly due to the short simulation time or simulation step size. Therefore, the simulation should cover different operating points as much as possible and extend for a longer simulation time.

The search space of the optimization without the position controller is shown in Figure 6. It can be seen, that the search area is indeed non-convex.

Figure 6. Search Space *(Best Fit=blue; Worst Fit=red)*



V. SUMMARY

The potential of the proposed algorithm was verified by an example. It turned out that good results could be achieved within relatively few simulation steps. For this purpose, the fitness function with inequality constraints was transformed to a fitness with additive penalty terms. A generic, on floating point numbers based EA framework had been used for the tests. Significantly faster convergence could be achieved with a custom algorithm on integer numbers. Simulation results should be kept in a ring buffer, since simulations were performed several times with the same parameters.

The broad applicability of the algorithm to various problems of fixed-point transformation as well as providing a fully automated solution for the end user is subject of further research.

REFERENCES

[1] H. Keding, M. Willems, M. Coors and H. Meyr, "FRIDGE: A Fixed-Point Design and Simulation Environment", in Proc. Design Automation and Test in Europe, IEEE, 1998, pp. 429–435.

[2] The MathWorks, Simulink Fixed Point User's Guide, The MathWorks, Inc., 2011

[3] W. Sung and K.-I. Kum, "Simulation-based Word-length Optimization Method for Fixed-Point Digital Signal Processing Systems", in IEEE Transactions on Signal Processing, IEEE, Dec. 1995, pp. 3087–3090.

[4] S. Kim, K.-I. Kum and W. Sung, "Fixed-point optimization utility for C and C++ based digital signal processing programs", in IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, IEEE, Nov. 1998, pp. 1455–1464.

[5] C. Shi and R.W. Brodersen, "Automated fixed-point data-type optimization tool for signal processing and communication systems", in 41st Proceedings on Design Automation Conference, IEEE, 2004, pp. 478–483.

[6] C. Shi and R.W. Brodersen, "An automated floating-point to fixed-point conversion methodology", in Proceedings on IEEE International Conference on Acoustics, Speech, and Signal Processing (Vol. 2), IEEE, 2003, pp. 529–532.

[7] P. Banerjee, D. Bagchi, M. Haldar, A. Nayak, V. Kim and R. Uribe, "Automatic conversion of floating point MATLAB programs into fixed point FPGA based hardware design", in 11th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, IEEE, 2003, pp. 263–264.

[8] D. Menard and D. Chillet, "Automatic Floating-point to Fixed-point Conversion for DSP Code Generation", in Proceedings of International Conference on Compilers, Architecture and Synthesis for Embedded Systems, ACM, 2002, pp. 270–276.

[9] G. Caffarena, Á. Fernández-Herrero, J. A. López and C. Carreras, "Fast Fixed-Point Optimization of DSP Algorithms", in VLSI-SoC: Forward-Looking Trends in IC and Systems Design, Springer, 2012, pp. 182–205.

[10] O. Sarbishei, K. Radecka and Z. Zilic, "Analytical Optimization of Bit-Widths in Fixed-Point LTI Systems", in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, IEEE, March 2012, pp. 343–355.

[11] R. Nehmeh, D. Menard, A. Banciu, T. Michel and R. Rocher, "Integer word-length optimization for fixed-point systems", in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, May 2014, pp. 8321-8325.

[12] Alan V. Oppenheim and Roland W. Schafer, Discrete Time Signal Processing, Prentice-Hall, Inc., 1998

[13] John G. Proakis and Dimitris G. Manolakis, Digital Signal Processing-Principles, Algorithms, and Applications, Prentice-Hall, Inc., 1996