IMTC 2004 - Instrumentation and Measurement
Technology Conference
Como, Italy, May 18–20, 2004

On the Monotonicity and Linearity of Ideal Radix-based A/D Converters

János Márkus and István Kollár
Department of Measurement and Information Systems
Budapest University of Technology and Economics
P.O. Box 91, H-1521 Budapest, Hungary
Phone: + 36 1 463-3583, Fax: + 36 1 463-4112, Email: markus@mit.bme.hu, kollar@mit.bme.hu

Abstract – *Both cyclic and pipelined analog-to-digital converters are getting more and more popular, as they are relatively easy to design and either have a high throughput (pipelined converters) or small area- and power-consumption (cyclic/algorithmic converters). To avoid saturation and to ensure effective digital calibration, in the analog stage(s) of these converters, instead of the ideal two, often a smaller nominal gain (called radix number) is used. In this paper properties of these radix-based converters are discussed. First, it is shown that these type of converters produce non-monotonic output. The causes of this phenomena are discussed in detail and a method to avoid non-monotonicity is suggested. Second, it is shown that even the ideal sub-radix converters have limited linearity. Lower bound for the differential non-linearity (DNL) is calculated. The results about monotonicity can be used either to quickly locate and avoid non-monotonic code transitions in a converter. The derived expressions for the lower bound of the DNL can be used to estimate the minimum required number of cycles (stages) for a converter to push the DNL below the specification.*

Keywords – *Analog-digital conversion, multi-stage pipelined, cyclic, algorithmic, sub-ranging A/D converter, non-radix-2, radix less then 2, sub-radix ADC, monotonicity, linearity, differential non-linearity, DNL*

I. INTRODUCTION

Sub-ranging analog-to-digital (A/D) converters [1] are getting more and more popular in different applications. In these converters the input signal is quantized first by a coarse (often 1-bit) quantizer, then the analog residue is calculated and requantized either by the same circuit (algorithmic/cyclic converter, [1], Fig. 1) or by another similar stage (sub-ranging/pipelined converter, [1]).

Cyclic converters are easy to design and have very low area- and power-consumption. Pipelined converters can exhibit high throughput at medium or high resolution, and are commonly used in high-speed digital communication systems. Due to analog component mismatches, the resolution of these converters is limited to 9-11 bits using standard technologies.

To enhance the resolution, instead of the expensive individual laser wafer trimming, self-calibration can be used, where the converter measures its own error and subtracts it from the output. The subtraction can be done either in the analog domain (mixed-mode calibration, e.g. [2]) or in the digital domain (fully-digital calibration, e.g. [3–9]). Fully digital calibration is the most preferable, as it does not require high-precision analog or mixed-mode components such as capacitor-arrays or other digital-to-analog converter (DAC) in the system.

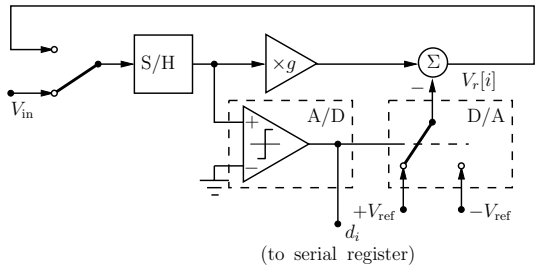

Fig. 1. Block diagram of an algorithmic converter. $V_{\rm in}$ is the input signal, $V_{\rm ref}$ is the reference signal, $V_r[i]$ is the residue signal in the ith cycle, g is the radix number equal to two in the ideal case, and d_i is the one-bit digital output.

Fig. 1 shows the block diagram of a 1-bit/stage cyclic converter. Its operation is as follows. In the first cycle, the input signal sampled, then quantized by a one-bit A/D (i.e., a comparator), then the quantized output is converted back to analog again by the D/A and subtracted from the input multiplied by the radix number g. The value of g is equal to two in the ideal case. In the next cycle, this residue is used as an input signal to obtain the second MSB, and so on, up to n. The output of the converter is the sequence of the one-bit d_i-s, which is the binary representation of the input signal in the ideal case. Although the cyclic converter belongs to the family of 1-bit/stage sub-ranging converters, its operation is very similar to the successive approximation converter. The main difference is that the latter one requires precise multibit feedback DAC.

Due to the finite precision of analog components, circuit non-ideality errors affect the accuracy of the converter. The most important error source is the capacitor mismatch which causes g to become inaccurate. As a result, two types of error can occur [4, 5]. If $g > 2$, at specific inputs at least one stage will be saturated, causing missing-decision-level error (i.e., the output does not change for a wide range of the input signal). If $g < 2$, codes will be missing in the output (i.e., the output jumps larger than one LSB at a code transition). Note that digital calibration, which, using the measured errors, simply reassigns digital codes to other ones, does not correct for missing analog decision levels (there is no information in the digital domain about the range of the input signal causing the same output code). Therefore, the uncalibrated converter must provide

decision levels spaced at no more then one LSB of the target resolution [4]. This can be ensured generally two ways: the first alternative is to use a nominal $g < 2$ in the circuit to make sure it won't turn over 2 even in a worst-case mismatch error, and use more stages to compensate for the resolution loss (*analog redundancy* [4, 6]). The second alternative is to use more then one bit/stage in the converter, and still use a nominal gain $g = 2$ (*digital redundancy*, [5]). Both of these techniques can be later compensated in the digital domain, and they can also be transformed into each other. The first approach uses simple analog hardware, but more complex digital algorithm, the second one uses more comparators and other analog elements in one stage, but the implementation of the digital correction becomes easier.

In the following the 1 bit/stage converters with $g < 2$ [4, 6] are examined in details. In Sec. II it is shown that these type of converters produce non-monotonic output. Although this behaviour occurs only sparingly (about 1% of the input range), in some applications (control loops, process monitoring), non-monotonicity should be avoided. A method to avoid this behaviour is also suggested. Monotonicity of non-ideal pipelined converters has been addressed in [10], however, this paper shows that even the ideal radix-based converters can exhibit non-monotonicity, and an easy fully-digital technique is suggested to remove the non-monotonic transitions, which can be used with traditional digital calibration techniques.

In Sec. III the linearity of these converters is discussed. It is shown that even the ideal radix-based converters have limited differential linearity. This minimum differential non-linearity (DNL) can be easily estimated from the number of cycles the converter operates (n), the radix number g, and the target resolution $n_{\rm bit}$. Although this linearity cannot be eliminated using radix-based digital correction techniques, it can be made as small as required for a given application. Linearity of pipelined converters has been addressed in [11]. Here a less complex estimation of the minimum DNL is given and shown that this can be further decreased by adding some more stages.

II. MONOTONICITY OF RADIX-BASED CONVERTERS

A. Operation Details

In [4], a radix-based pipeline converter was introduced, where the first stages used a nominal gain $g < 2$ value, and the last 11 stages used a nominal gain of 2. During the calibration process, these last stages were used to measure the residue jumps of the former stages. The need for two kinds of stages makes the implementation of this calibration technique to algorithmic converters difficult. Thus, in [6] equal radix numbers ($g = 1.95$) were suggested for all stages. In this case the residues during the conversion can be obtained as follows (cf. Fig. 1):

$$\begin{aligned} V_r[1] &= gV_{\rm in} - d_1V_{\rm ref} \\ V_r[2] &= gV_r[1] - d_2V_{\rm ref} = \\ &= g^2V_{\rm in} - gd_1V_{\rm ref} - d_2V_{\rm ref} \end{aligned}$$

$$\begin{aligned} V_r[n] &= g^nV_{\rm in} - g^{n-1}d_1V_{\rm ref} - \ldots - d_nV_{\rm ref} = \\ &= g^n\left(V_{\rm in} - \sum_{i=1}^{n} g^{-i}d_iV_{\rm ref}\right), \end{aligned} \tag{1}$$

where $V_{\rm in}$ is the input signal, $V_{\rm ref}$ is the reference voltage of the converter, $V_r[i]$ is the residue of the ith conversion cycle, $d_i \in \{-1, 1\}$ is the ith decision of the comparator and n is the number of cycles the converter operates. Rearranging Eq. (1), one can get

$$\frac{V_{\rm in}}{V_{\rm ref}} = \sum_{i=1}^{n} g^{-i}d_i + \epsilon \qquad |\epsilon| \le \frac{1}{g^n}, \tag{2}$$

where ϵ is the quantization error. Another calculation method of the input signal is

$$V_{\rm in} = V_{\rm LSB}\left(\frac{1}{2}\sum_{i=1}^{n} g^{n-i}d_i + \epsilon'\right), \tag{3}$$

where $V_{\rm LSB} = 2V_{\rm ref}/g^n$ and $\epsilon' = \epsilon g^n/2 \le 0.5$.

Thus, the digital output can be calculated as

$$D = \frac{1}{2}\sum_{i=1}^{n} g^{n-i}d_i. \tag{4}$$

Here, d_1 means the MSB, while d_n denotes the LSB value of the digital word. The factor of $1/2$ comes from the fact that $d_i \in \{-1, 1\}$, and it can be even omitted by mapping d_i into the $\{0, 1\}$ binary representation, which transforms into a DC offset in the digital output.

With an even traditional representation, when d'_0 is the LSB and $d'_i \in \{0, 1\}$, the output can be calculated simply as

$$D = \sum_{j=0}^{n-1} g^j d'_j, \tag{5}$$

where $d'_0 = d_n$, $d'_1 = d_{n-1}$, …, $d'_{n-1} = d_1$.

B. Output Calculation

Calculating Eq. (5) for all possible input codes would exhibit large jumps at major code transitions, resulting in non-equivalent mapping from the analog input to the digital codes. As an example, for an input code of $0111\,1111\,1111$, $D = \sum_{j=0}^{n-2} g^j = 1630.710$, while for the input code of $1000\,0000\,0000$, $D = g^{n-1} = 1550.175$, resulting in a huge negative jump of 80 LSBs at the MSB transition.

In reality, as discussed in the introduction, an algorithmic converter with $g < 2$ will exhibit missing codes. Moreover, it was proven previously in Eq. (3) that (due to the negative feedback) the operation of the cyclic converter ensures that the absolute difference between the input signal and the *radix-based* representation of the input signal will be always less then half LSB.

TABLE I

CODE TRANSITIONS OF A 12-CYCLE CONVERTER WITH $g = 1.95$

#	Code Transition	Step size in LSB_{12}
#0	xxxx xxxx xxx0→xxxx xxxx xxx1	1
#1	xxxx xxxx xx01→xxxx xxxx xx10	0.95
#2	xxxx xxxx x011→xxxx xxxx x100	0.853
#3	xxxx xxxx 0111→xxxx xxxx 1000	0.662
#4	xxxx xxx0 1111→xxxx xxx1 0000	0.292
#5	xxxx xx01 1111→xxxx xx10 0000	-0.431
#6	xxxx x011 1111→xxxx x100 0000	-1.841
⋮	⋮	⋮
#11	0111 1111 1111→1000 0000 0000	-80.53

Unfortunately, this algorithm does not guarantee also monotonicity. Although this behaviour ensures that the difference between two consecutive digital output is less then or equal to one LSB, it allows this difference to be negative, thus, it allows non-monotonic behaviour, if such transition exists. Table I shows the the major code transitions of a cyclic converter with $g = 1.95$. It can be seen that code transition #5 is negative and its absolute value is less then one LSB.

Based on this derivation, it is expected that an ideal radix-based converter with $g = 1.95$ and $n = 12$ produces major code transitions #0 to #5, but no major code transitions #6 and above. Due to the missing codes, other (positive or negative) step sizes less then one LSB can also be produced. According to these derivations, Fig. 2(a) shows the transfer characteristics of an ideal radix-based cyclic converter, while Fig. 2(b) shows the step-sizes in radix-g 12-bit LSB. Although the general transfer characteristic seems to be smooth without non-monotonic jumps, the inset of Fig. 2(a) shows that non-monotonic code transition happens. The example shown is the transition of $1000\,1101\,1111 \rightarrow 1000\,1110\,0000$, containing the major code transition of #5.

Comparing Tab. I with Fig. 2(b), it can be seen that most step sizes in the simulated ideal converter belong to the code transition #0–#5 derived above (one can quickly locate that in Fig. 2(b) most dots belongs to $1, 0.95, 0.853, \ldots, -0.431$). Due to the missing codes, there exist some other code transitions as well (in this case only 4 different types), which are responsible for the other step-sizes. For example, the most negative step (-0.69 LSB) occurs at the code transitions $\mathrm{xxxx}\,0111\,1101 \rightarrow \mathrm{xxxx}\,1000\,0010$, which is not a simple major code transition. Note that it still contains the key feature of the code transition #5, i.e. this code transition also contains 5 consecutive ones and then 5 consecutive zeros. The difference is that the algorithm uses the available LSBs to refine the digital code and to get a better representation of the input signal.

The above derivation can be extended to arbitrary g the following way. Let us denote the number of consecutive ones turning to zero in a code transition, which is negative, but with absolute value is less then 1 LSB, n_0. Thus, the code transition can be denoted as $\ldots 0\underbrace{111\ldots1}_{n_0} \rightarrow \ldots 1\underbrace{000\ldots0}_{n_0}$. To find n_0, the mathematical formalism of the above conditions is the following:

$$-1 < g^{n_0} - \sum_{i=0}^{n_0-1} g^i < 0, \tag{6}$$

i.e.,

$$-1 < \frac{(g-2)g^{n_0}+1}{g-1} < 0. \tag{7}$$

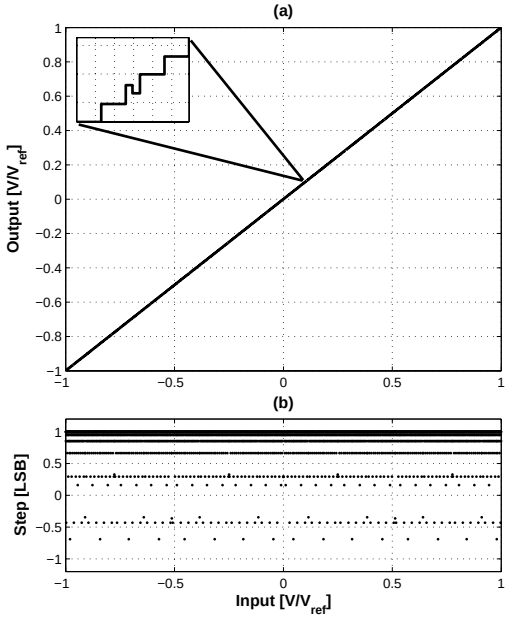

Fig. 2. (a) Output of a 12-cycles ideal radix-based converter. The inset shows a detailed transition, which is non-monotonic. (b) Steps of the output code transitions. Steps less then 0 correspond to non-monotonic code transitions.

Ignoring the exhaustive derivation, the following statements can be proven:

- As Eq. (7) is a continuous function of n_0, there exists a real $n_{0,-1}$ and $n_{0,0}$ for which Eq. (7) becomes equal to -1 and 0, respectively;
- $n_{0,-1} = n_{0,0} + 1$;
- $n_0 = \lfloor n_{0,0} \rfloor + 1 = \lfloor n_{0,-1} \rfloor$, where $\lfloor . \rfloor$ denotes the integer part of a rational number.

Fig. 3 illustrates the statements above for $g \in (1.9, 1.99)$. ∇ and $\triangle$ shows $n_{0,-1}$ and $n_{0,0}$ (the limits for which Eq. (7) becomes -1 and 0), respectively. The solid line shows the calculated n_0, while $\times$ denotes simulation results for some gs. Note that there exist a few g (e.g. 1.9276, 1.9659, 1.9836) for which the worst case code transition is exactly zero, but as g is a random variable, one cannot rely on them.

C. Requantization

In the discussion above, calculation of the output code (Eq. (5)) was assumed to be infinitely precise. In a real hardware the output code is calculated, then requantized to $n_{\mathrm{bit}} < n$

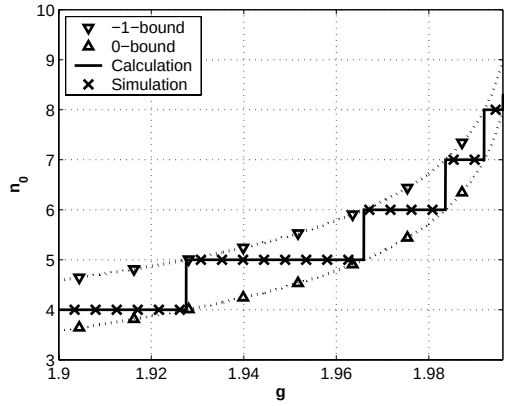

Fig. 3. Calculation and simulation results of n_0, which causes non-monotonic transition in a cyclic converter. ∇ and $\triangle$ illustrates the limits -1 and 0 in Eq. (6), respectively. Solid line is the calculated n_0, while $\times$ are simulation results of an ideal converter.

bit, which is the target resolution of the converter [6]. As $n_{\text{bit}} \leq n-2$, the final LSB size will be 2–3 times larger then the step size of the radix-based converter. Thus, most of the non-monotonic transitions will be smoothed out by the requantization process. However, due to the large number of non-monotonic transitions and the fact that that the step-sizes of the radix-based converter are uneven, there will always be some transitions which crosses one of the quantization thresholds, causing non-monotonic behaviour in the final output.

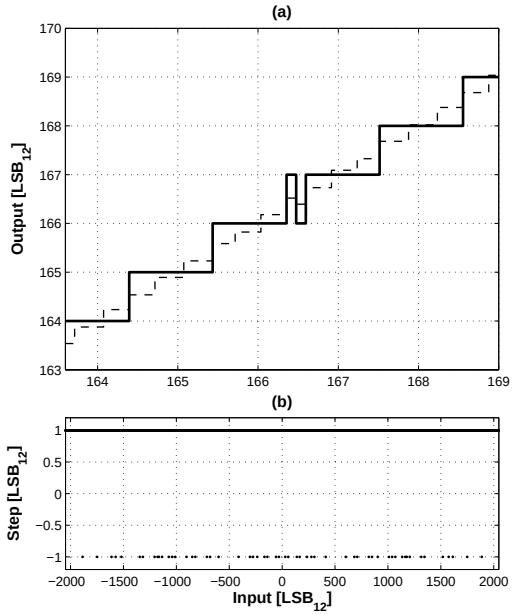

Fig. 4. (a) Enlarged output of a 14-cycles 12-bit converter (dashdot line: 14-cycle output code, solid line: 12-bit requantized binary output code), showing a non-monotonic code transition. (b) Steps of all output 12-bit binary code. Steps equal to -1 correspond to non-monotonic output transitions.

Fig. 4 shows simulation results of a 12-bit, 14-cycle, $g = 1.95$ converter. Fig. 4(a) shows an enlarged part of the converter transfer curve with one non-monotonic code transitions, while Fig. 4(b) shows all the step-sizes. It can be seen that there are several transitions when the output code is not monotonic even after the final quantization (Fig. 4(b)). It is also shown that every non-monotonic transient output exists only within 0.2-0.3 LSB of the input signal (Fig. 4(a)). Table II shows the statistics of the same example, showing that in the final output about 1% of the input signal will be mapped wrongly. In most cases it does not cause any problem.

TABLE II
NUMBER OF NON-MONOTONIC TRANSITIONS IN A 14-CYCLE, 12-BIT CONVERTER WITH $g = 1.95$

# of g-based codes	13101
# of g-based non-monotonic transition	326
Wrong g-based codes (% of all g codes)	4.97%
Wrong g-based codes (% of input range)	1.49%
# of 2-based codes	4095
# of 2-based non-monotonic transition	56
Wrong 2-based codes (% of all 2 codes)	2.74%
Wrong 2-based codes (% of input range)	0.84%

If this behaviour is not acceptable, two methods can be used to avoid non-monotonicity. First, as the width of this transition is inversely proportional to n, the number of cycles the converter operates, increasing n will further decrease the width and also the number of non-monotonic code transitions. Note that this method does not eliminate these transitions completely.

Alternatively, it was shown previously that this type of transition occurs only if n_0 consecutive ones changes to n_0 consecutive zeros in the output code. This n_0 can be calculated from g, e.g. if $g = 1.95$, $n_0 = 5$ (cf. Sec. II). Therefore, with a simple digital hardware subtracting one from any code containing n_0 consecutive ones and adding one to any code containing n_0 consecutive zeros before the requantization process will eventually swap the two codes, thus removing all non-monotonic transitions from the digitally calibrated output code.

D. Noise in the converter

In the previous subsections ideal noiseless converters were evaluated. In a real converter, however, analog noise cannot be avoided. In the presence of noise, the definition of monotonicity must be modified: a converter is monotonic only if the *mean value* of its output is monotonic. The monotonicity of the converter's output thus depends on the noise level and the

number of samples used in the estimation of the mean value of the converter's output. The following preliminary results has been achieved and verified by simulation:

1. If the noise variance (σ_n^2) is much larger then that of the quantization noise ($q^2/12$, where q is the quantization interval), the output of the converter will be very noisy, thus, deterministic non-monotonic code transitions cannot be localized.
2. If σ_n^2 is in the order of $q^2/12$, the analog noise behaves as a dither signal, causing many 1-bit non-deterministic, non-monotonic code transitions in the signal. The deterministic non-monotonicity will be covered up by the dither signal. Note that if averaging is used at the output of the converter, the deterministic non-monotonic transitions can be localized and removed by the methods described at the end of the previous subsection.
3. If the noise variance is much smaller then that of the quantization noise, only the deterministic non-monotonic transitions appear in the output code sequence, and can be removed by the methods described earlier.

III. LINEARITY OF RADIX-BASED CONVERTERS

Another important property of radix-based converters is that even the ideal converter have limited linearity [11]. In this section, this property is examined and the minimum achievable differential non-linearity is estimated. The non-monotonic codes (cf. Sec. II) are not taken into consideration throughout this section.

As it was discussed in the previous section, the step-sizes of a radix-based converter with $g < 2$ are not equal (cf. Tab. I), meaning that in the case of a slow ramp input signal the duration time of each 12-bit radix-based codes will differ. As these codes are calculated by the algorithm in the analog domain, the duration time of the codes cannot be changed later in the digital domain. Thus, even the final requantized output values of the converter will map the input signal unevenly to the output digital codes. Fig. 5 shows two examples of this effect. Fig. 5(a) shows an example when the DNL is greater then 0, i.e. the analog input value changes more then one V_{LSB} within a digital code, while Fig. 5(b) shows an example when the DNL is less then 0, i.e. the analog input value changes less then one V_{LSB} within a digital code.

To estimate the DNL of the converter, the following method is proposed. As it is shown in Fig. 5, by adding the step-sizes of consecutive code transitions of the 12-cycle converter, one can get an estimate of the duration of the 10-bit final output code. The number of consecutive code transitions to be added can be determined by the relative width of the 12-cycle and the 10-bit codes. Mathematically, the sum of the consecutive 12-cycles code transitions must be within 1 $\mathrm{LSB}_{10} \pm 1\ \mathrm{LSB}_{12}$. In the current example the number of code transitions to be taken into account is 3 or 4.

Tab. III shows the duration of the 12-cycle code transition in LSB of the 10-bit final output. Tab. IV shows all the possible 12-cycle code transitions, which are mapped into one 10-bit transition, and the DNL error associated with them. Note that there exist other code transitions which are simple permutations of the ones listed in the table, causing the same DNL error.

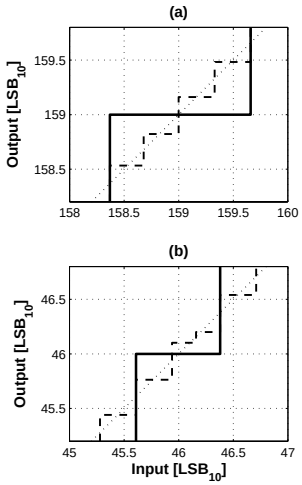

Fig. 5. (a) An example, when $\mathrm{DNL} > 0$ in a 12-cycles 10-bit converter. (b) An example, when $\mathrm{DNL} < 0$ (dotted line: input signal, dashdot line: 12-cycle output code, solid line: 10-bit requantized binary output code)

TABLE III
CODE TRANSITIONS IN 10-BIT LSBS OF A 12-CYCLE (10-BIT) CONVERTER WITH $g = 1.95$

#	Code Transition	Step size in LSB_{10}
#0	...xx xxx0→...xx xxx1	0.339
#1	...xx xx01→...xx xx10	0.322
#2	...xx x011→...xx x100	0.289
#3	...xx 0111→...xx 1000	0.224
#4	...x0 1111→...x1 0000	0.099

Finally, Fig. 6 shows simulation results of the ideal radix-based converter. Fig. 6(a) shows the traditional representation of the DNL curve, from which one cannot see any specific DNL levels supporting the derivation above. However, representing the DNL with dots (Fig. 6(b)), one can clearly see the distinct levels of DNL caused by the different code transitions. Most of the levels are close to the derived ones of Tab. IV (see the gray lines of Fig. 6(b)). There are some codes, however, where the DNL differs from the predicted ones. These are 10-bit output codes which contain 12-bit code transitions with missing codes, thus they cannot be predicted from regular major code transitions.

Note that code transition #0→#1→#0 never happens alone, but always with one more code transition (either →#2, →#3 or →#4), as the 4 consecutive transition still fit into the $1\ \mathrm{LSB}_{10} + 1\ \mathrm{LSB}_{12}$ criteria. This is the reason why there is no output code with approximately 0 DNL error despite of its prediction

TABLE IV
MULTIPLE 12-BIT CODE TRANSITIONS CAUSING ONE 10-BIT TRANSITION IN THE REQUANTIZED FINAL OUTPUT.

Code Transition	Step size in LSB_{10}	DNL in LSB_{10}
#0 →#1 →#0	1.00	-0.00
#0 →#2 →#0	0.97	-0.03
#0 →#3 →#0	0.90	-0.1
#0 →#4 →#0	0.78	-0.22
#0 →#1 →#0 →#2	1.29	+0.29
#0 →#1 →#0 →#3	1.22	+0.22
#0 →#1 →#0 →#4	1.10	+0.10
#1 →#0 →#2	0.95	-0.05
#1 →#0 →#3	0.88	-0.12
#1 →#0 →#4	0.76	-0.24

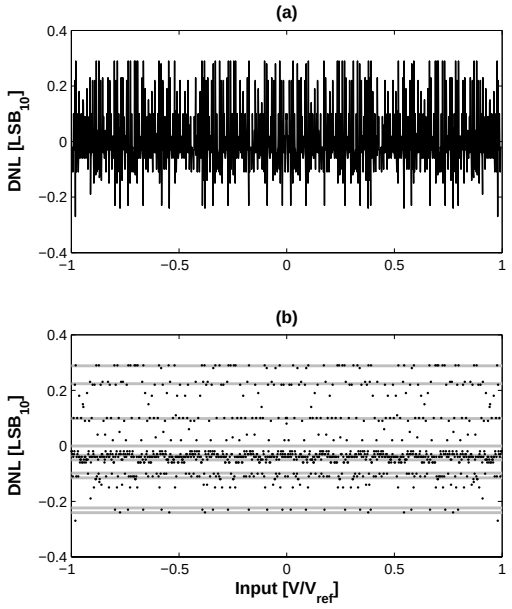

Fig. 6. (a) Traditional representation of the DNL of the ideal converter. (b) Representation of the DNL with dots. Solid gray lines show the estimated levels of the DNL error (cf. Tab. IV).

in Tab. IV, row 1.

Using Tab. IV, the minimum DNL of a radix-based converter can be estimated as 0.29 LSB. This DNL is proportional to the ratio of the code width of the n-cycle converter and the target code width of the converter. Thus, increasing the number of cycles in the converter while keeping the target bit number the same will decrease this DNL error to an acceptable level for a desired application.

IV. CONCLUSION

In this paper properties of radix-based converters were discussed.

First, it was shown that even ideal converters produce non-monotonic output. The mathematical conditions derived in the paper for the occurrence of the non-monotonic code transitions can be used to easily detect these transitions in the output of a radix-based converter. Using the method suggested at the end of Sec. II designers can completely eliminate non-monotonic transitions in the digitally calibrated radix-based converter.

Second, it was shown that even ideal radix-based converters have limited linearity, which is inherited from the topology, thus it cannot be removed from the system. The expressions derived for the minimum differential non-linearity can be used to estimate the required number of stages n in a given design, to push the minimum linearity error below the required value of the specification.

ACKNOWLEDGEMENT

The authors are thankful to Prof. Un-Ku Moon, his research group, and the anonymous reviewers for reviewing the preliminary version of the paper.

This research was supported in part by the Hungarian Fund for Scientific Research, under contract No. OTKA T 033 053, by the László Schnell Instrumentation and Measurement Foundation and by the Siemens AG.

REFERENCES

[1] D. Johns and K. Martin, *Analog Integrated Circuit Design*, 1st ed. Hoboken, NJ, USA: John Wiley & Sons, Nov. 1996, ch. 13 (Nyquist-Rate A/D Converters), pp. 487–530.

[2] H. Lee, D. Hodges, and P. Gray, "A self-calibrating 15-b CMOS A/D converter," *IEEE Journal of Solid-State Circuits*, vol. 19, no. 12, pp. 813–19, Dec. 1984.

[3] S.-H. Lee and B.-S. Song, "Digital-domain calibration of multistep analog-to-digital converters," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 12, pp. 1679–88, Dec. 1992.

[4] A. N. Karanicolas, H.-S. Lee, and K. L. Bacrania, "A 15-b 1-Msample/s digitally self-calibrated pipeline ADC," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 12, pp. 1207–15, Dec. 1993.

[5] H.-S. Lee, "A 12-b 600 ks/s digitally self-calibrated pipelined algorithmic ADC," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 4, pp. 509–15, Apr. 1994.

[6] O. E. Erdoğan, P. J. Hurst, and S. H. Lewis, "A 12-b digital-background-calibrated algorithmic ADC with -90-dB THD," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 12, pp. 1812–20, Dec. 1999.

[7] D.-Y. Chang and U.-K. Moon, "Radix-based digital calibration technique for multi-stage ADC," in *Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS'02)*, vol. 2, Scottsdale, Arizona, USA, 26–29 May 2002, pp. II–796–II–799.

[8] J. Li and U.-K. Moon, "An extended radix-based digital calibration technique for multi-stage ADC," in *Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS'03)*, vol. 1, Bangkok, Thailand, 25–28 May 2003, pp. I–829–I–832.

[9] ——, "Background calibration techniques for multistage pipelined ADCs with digital redundancy," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 50, no. 9, pp. 531–538, Sept. 2003.

[10] W. Law, J. Guo, C. T. Peach, W. J. Helms, and D. J. Allstot, "A monotonic digital calibration technique for pipelined data converters," in *Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS'03)*, vol. 1, Bangkok, Thailand, 25–28 May 2003, pp. I–873–I–876.

[11] Y. Ren, B. H. Leung, and Y.-M. Lin, "A mismatch-independent DNL pipelined analog-to-digital converter," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 5, pp. 517–26, May 1999.