A NEW APPROACH TO DATA CONVERSION: DIRECT ANALOG-TO-RESIDUE CONVERTER

D. Radhakrishnan and A.P. Preethy

Nanyang Technological University,
Nanyang Avenue, Singapore 639798

ABSTRACT

A novel design of a direct analog-to-residue converter is presented in this paper. The design makes use of two successive approximation analog-to-digital (A/D) converters, a few modulo adders and a small look-up table. One of the digital-to-analog converters is modified to generate outputs which are weighted by a constant factor, and one of the comparators is replaced by a difference amplifier. The look-up table needed is a very small percentage of the entire chip area and is shown to be only 840 bytes for a 36 bit residue number system converter.

1. INTRODUCTION

Residue number systems (RNS) are becoming very popular in many computation intensive applications like DSP [4]. An RNS is defined by a set of relatively prime integers $m_1, m_2, \ldots, m_r$ called the moduli of the number system. Such a system provides unique representation of numbers from 0 to M-1 where $M=\prod_{i=1}^{r} m_i$. Each integer X is represented by an r tuple ($x_1, x_2, \ldots, x_r$), where each residue x_i =X mod m_i , defined as the least remainder when X is divided by the moduli m_i . In RNS, arithmetic operations on large integers are done by converting them into smaller residues and performing the operations independently and all in parallel, thereby speeding up the whole operation. In present systems an analog signal is first converted into binary and then a binary-to-residue converter is used to generate the residues. The residue results produced at the end of the processing is finally reconverted back to binary. But this conversion from binary-to-residue and from residue-to-binary are complex operations for a general moduli set. Many researchers are therefore exploring new ways of tackling this problem - the one most sought after being the direct conversion of an analog signal to the residue form, an analog-to-residue (A/R) converter.

Recently a scheme for direct conversion from analog-to-residue form using a flash converter was proposed [2]. A flash converter basically consists of 2^n-1 comparators and 2^n resistors for n bit resolution [3]. The analog voltage sample applied at the input makes, as many comparators as possible within its voltage level, to switch their states to 1's. The rest of the comparators being in 0 states, the resulting output is a thermometer code. This thermometer code is converted into binary using a decoder.

The flash A/R converter proposed in [2] uses a PLA, latch, a code converter, a set of buffers and XOR gates in addition to the comparators and resistor elements as shown in Figure 1. The comparators are grouped into M/m_r groups, with m_r outputs in each group. The m_r-1 outputs corresponding to a non-zero residue is fed to a buffer stage. The buffer outputs are connected to a bus. The comparator with the lowest threshold voltage in each group is identified as having the base value for the group. The residues corresponding to the outputs of the comparators in each group range from 0 to m_r-1 with 0 assigned to the base comparator. Every input X belongs to one of the above groups and hence the number of 1 outputs from that group uniquely identifies the residue x_r. For a given input X only one of the buffers is enabled, this being identified by an XOR gate. A set of 2 input XOR gates are used for this purpose, with the inputs of each gate connected from the outputs of adjacent base value comparators. The outputs of the enabled buffer are pushed into a latch which feeds a PLA to generate the residue x_r. The XOR gate with its output at logic 1 uniquely identifies the group. Knowing the group and the number of 1 outputs from that group, each and every other residue can be uniquely identified. Since the number of XOR gates equals M/m_r, a code translator using OR gates is used to form its binary coding. The code translator output is also fed to the PLA to generate all the other residues. The size of the PLA in this case is $(l+m_r)\times\sum_{i=1}^{r}\lceil\log_2(m_i-1)\rceil$ bits, where $l=\left\lceil\log_2\left(\frac{M}{m_r}-1\right)\right\rceil$ As can be easily seen from the above, the PLA size, number of comparators and resistors become prohibitively large as the converter size increases. Hence this cannot be used for converter sizes of more than 8-10 bits.

The major drawbacks of the flash A/D converter are the large hardware requirement, high power dissipation and sensitivity to comparator offsets [1]. For an n-bit A/D converter 2^n-1 comparators are required with offset voltage of less than $1/2^n$. Because comparators with small offsets are expensive to build and difficult to design A/D converters with resolutions higher than 8 bits rarely use the flash architecture.

For almost all real world problems we need much higher resolutions than 8 bits. This forces us to find other methods for A/R conversion. One promising

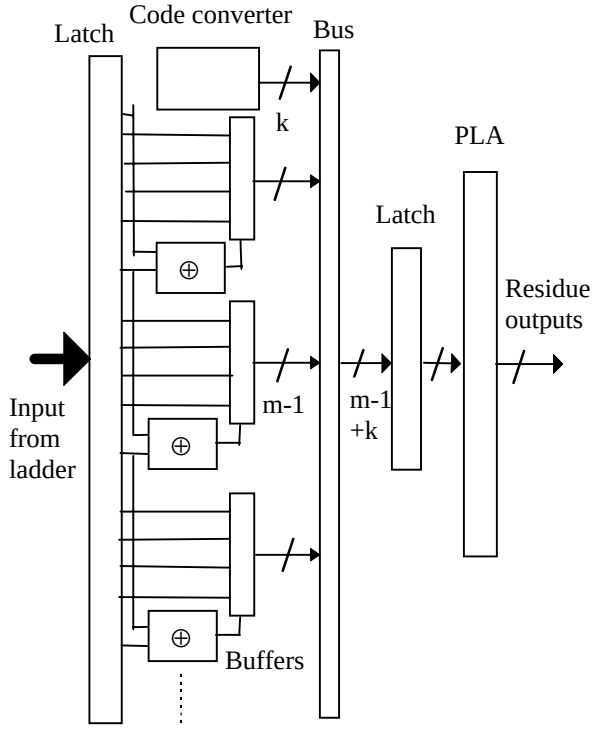

Figure 1. Multiple Residue Flash Converter

approach will be to use a successive approximation type of converter. In fact it may be noted that the most widely used general-purpose A/D converter is the successive approximation type. A block diagram of such a converter is shown in Figure 2. A sample of the analog input X appears at one input of the comparator. The digital word being generated at the output register is converted to analog voltage using a D/A converter and compared with the analog input sample X. Starting with all zeroes, a first 'trial' digital value is generated by changing the MSB of the output register to 1. The D/A converter converts this to an analog value V_R, which is then compared with X. If $X<V_R$, the assumed bit of 1 is rejected and replaced by 0. Otherwise, the 1 is retained. Focus is next directed to the second most significant bit of the trial value and is continued in a similar fashion till all bit positions in the output register are compared. The completion of conversion is triggered by a change in the state of the comparator. At this time the conversion stops and the output register contains the digital word. The total time (worst case) needed for conversion is n+1 clock cycles, as the trial bit has to traverse all n bit positions of the output register. A successive approximation A/D converter is reasonably fast for many applications. Hence this is used as the basic element in our A/R converter.

2. A SUCCESSIVE APPROXIMATION A/R CONVERTER

A block diagram of the A/R converter is shown in Figure 3. This is obtained by cascading two separate successive approximation A/D converters. The sampled analog input voltage X is applied to the input of the first A/D converter. This stage is modified by replacing the comparator with a difference amplifier (Amplifier 1) and adding a weighting factor (the largest modulus m_r) to the output of the D/A converter. Hence the difference amplifier 1 produces an output voltage equivalent to $X-i*m_r$ during each successive step of the iteration, where i represents the value stored in Register 1. The size of this register is chosen to have k bits with $k = \left\lceil \log_2\left(\frac{M}{m_r} - 1\right)\right\rceil$. Once all the k bit positions of Register 1 are identified, the difference amplifier output will become $X- R*m_r$, where R represents the value stored in Register 1. It can be easily verified that this voltage $X-R*m_r < m_r$. The first stage of conversion stops at this time.

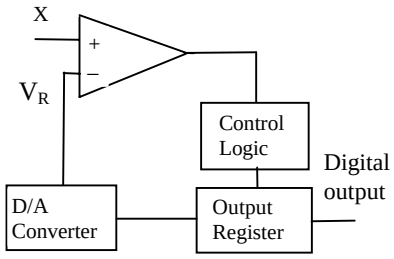

Figure 2. A Successive Approximation A/D Converter

The output of the difference amplifier 1 now represents the voltage equivalent of $x_r = X \bmod m_r$. This residue value x_r can be converted to a digital output using any one of the known A/D conversion techniques. In our design, a successive approximation A/D converter is used for this second stage also.

The second stage converter is exactly similar to the one shown in Figure 2. The output register size for this converter is $l = \lceil \log_2 m_r \rceil$ bits. The output of this register represents the residue x_r corresponding to the input X with respect to the modulus m_r. It may be noted that the second stage conversion is to be delayed till the completion of the conversion in the first stage. Furthermore, the extra amplifier stage with gain m_r shown in Figure 2 can be easily incorporated into the design of the D/A converter stage.

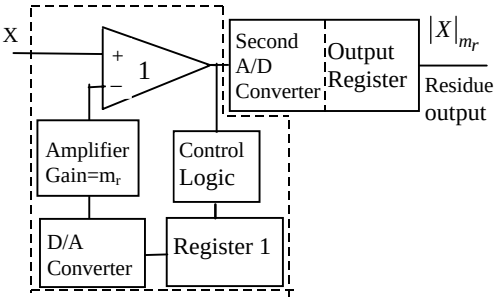

Figure 3. A Successive Approximation A/R Converter

The thrust of the whole design depends on the generation of the remaining r-1 residues. This must be done at the cost of minimum extra hardware and minimum delay. Our proposed architecture is based on the observation that the contents of Register 1 (Front end A/D converter) together with the output x_r (second stage A/D converter) uniquely identifies the original input X, since $X=R*m_r+x_r$. Hence this will also identify uniquely each one of the remaining residues $x_1, x_2, \ldots, x_{r-1}$. One solution is therefore to use a PLA to generate the above residues, with Register 1 and x_r as inputs to the PLA. But the size of this PLA becomes prohibitively large even for very small word length RNS converters. Hence a completely different design strategy by partitioning Register 1 into a number of equal length partitions is used in this paper. To efficiently use this technique, we assume that the moduli are all of equal word length with *l*-bits to represent each. First let us consider the generation of residue x_j corresponding to the modulus m_j. A residue generator for x_j is shown in Figure 4. Each partition group of Register 1 addresses a ROM that is programmed to produce its residue with respect to m_j. These residues are generated simultaneously and are added using modulo adders to get the final residue. A total of r-1 similar stages are needed to generate all the residues $x_1, x_2, \ldots, x_{r-1}$.

The conversion time for generating residue x_r equals the sum of the conversion times in both the A/D converter stages. This time is given by k+l+2 clock cycles, where k and l are the sizes of the registers in the first and second stage A/D converters respectively. A normal successive approximation A/D converter of the same word length uses more or less the same number of clock pulses for conversion of an analog signal to binary. The maximum number of extra clock pulses needed for the A/R converter will be 3 in the worst case.

The generation of the remaining residues need additional delay. This extra delay is due to the small ROM look-up tables and the modulo adder stages. If Register 1 is divided into 'p' partitions of *l*-bits each, then the number of modulo addition stages is $\lceil \log_2 (p+1) \rceil$. But as can be seen from Figure 4, a large portion of these delays overlap with the conversion time of the second stage. Hence all the residues are generated almost simultaneously.

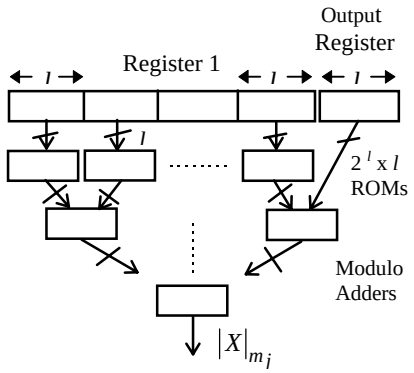

Figure 4. Residue Generation Logic

An example to illustrate the design of a 36 bit A/R converter is given below.

Example1: A 36 bit RNS converter can be implemented using eight 5 bit moduli 17,19,23,25,27,29,31 and 32. This gives a total range of more than 37 bits. The largest modulus in this case is m_r =32. The length of the registers for the first and second stage converters are *k*=31 and *l*=5 respectively. The conversion needs a total of 38 clock pulses to generate the first residue $|X|_{32}$, for any integer X within the range of 36 bits. Register 1 is divided into 7 partitions, six of 5 bits each and one last partition with a single bit. Each 5 bit partition uses a ROM of size 32X5 to generate the residues, thereby requiring a total of 192X5 per modulus. This amounts to a total of 840 bytes of storage for the entire system. A simple logic circuit can be used with the MSB bit to generate its equivalent residue value. The converter also

need seven 5 bit modulo adders per modulus, thereby requiring a total of 49 for the entire set.

3. CONCLUSIONS

The design of a direct A/R converter using two stages of successive approximation A/D converters is presented in this paper. The modifications needed in the A/D converters are very minimal, one of the comparators being replaced by a difference amplifier, and one D/A converter output being modified by adding an extra weighting factor. For generating all the residues, extra memory storage and a number of modulo adders are used. For a 36 bit converter this amounts to only 840 bytes of storage and forty nine 5 bit modulo adders.

The overall conversion speed of the proposed A/R converter is very close to that of a similar A/D converter, thereby eliminating the extra delay due to the binary-to-residue converters used in conventional approaches. Furthermore the conversion speed of the second stage A/D converter can be increased by using a flash A/D converter. This is possible since the resolution of this stage is usually less than 8 bits (5 bits for a 36 bit converter) even for a very large sized converter.

The amount of ROM storage can be reduced further by using multiple access techniques. By this method the ROM size for a 36 bit converter can be reduced to a mere 140 bytes of storage. But this in turn increases the overall delay for residue generation. This delay could be offset by using pipeline stages in the ROM, adder set up.

ACKNOWLEDGEMENTS

This work was supported in part by the NTU Research Grant under AcRF RG-22/95.

4. REFERENCES

[1] Cline D.W. “Noise, Speed, and Power Tradeoffs in Pipelined Analog-to-digital Converters”. *Ph.D. thesis*, University of California, Berkeley, 1995.

[2] Mandyam S. and Stouraitis T. “Efficient Analog-to-Residue Conversion Schemes”. *IEEE International Symposium on Circuits and Systems*. New Orleans, USA, pages 2885-2888, May 1990.

[3] Miner G.F. and Comer D.J. *Physical Data Acquisition for Digital Processing*, Prentice Hall, New Jersey, 1992.

[4] Soderstrand M.A., Jenkins W.K., Jullien G.A., and Taylor F.J. *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*, IEEE Press, New York, 1986.