200 MS/s ADC implemented in a FPGA employing TDCs

Harald Homulle
Delft University of Technology
Delft, The Netherlands
h.a.r.homulle@tudelft.nl

Francesco Regazzoni
USI - ALaRI
Lugano, Switzerland
regazzoni@alari.ch

Edoardo Charbon
Delft University of Technology
Delft, The Netherlands
e.charbon@tudelft.nl

ABSTRACT

Analog signals are used in many applications and systems, such as cyber physical systems, sensor networks and automotive applications. These are also applications where the use of FPGAs is continuously growing. To date, however there is no direct integration between FPGAs, which are digital, and the analog world (except for the newest generation of FPGAs). Currently, an external analog-to-digital converter (ADC) has to be added to the system, thus limiting its overall compactness and flexibility.

To address this issue we propose a novel architecture implementing a high speed ADC in reconfigurable devices. The system exploits picosecond resolution time-to-digital converters (TDCs) to reach a conversion as fast as its clock speed. The resulting analog-through-time-to-digital converter (ATDC) can achieve a sampling rate of 200 MS/s with a 7 bit resolution for signals ranging from 0 to 2.5 V. Except for the external resistor needed for the analog reference ramp, the system is fully integrated inside the target FPGA. Moreover, our design can be easily scaled for multichannel ADCs, proving the suitability of reconfigurable devices for applications requiring a deep integration between analog and digital world.

Keywords

ADC; TDC; FPGA; analog-to-digital; time-to-digital

1. INTRODUCTION

Analog-to-digital Conversion is needed in a large number of applications, ranging from sensor nodes, industrial control systems to (high energy) physics experiments. Devices used in these applications typically consist of an analog component which senses one or more parameters (for instance temperature, position, ECG or EEG), that is connected to a digital world, often a system-on-chip, composed of DSPs, processors and coprocessors. The analog part of the system is used to collect the data, while the digital part, often a reconfigurable circuit, e.g. a FPGA, analyses the data and reacts accordingly.

The conventional approach is to use an external device to carry out the conversion between analog and digital. This approach is not optimal, because it has a limited scalability, limited flexibility, and cannot achieve the compactness needed for several applications. When an external ADC is used, the interface with the FPGA is fixed at design time, and cannot be easily changed. As an additional component is needed, the overall size of the system increases, usually requiring extra connectors and PCBs. Finally, the use of an external ADC increases the overall system power consumption.

A better approach would be to completely integrate an ADC in the FPGA, either an ASIC ADC on-chip or a reconfigurable ADC.

This need has been recognised by FPGA manufacturers. E.g. Xilinx now includes an on-chip ADC (XADC) in the new 7 Family generation of FPGAs, ranging from the low-cost Artix 7 to the high-end Virtex 7. Advantages over external ADCs are the smaller size, lower power and easier interfacability. The main drawback of this solution is that it is fixed in terms of conversion rate, voltage range and number of input channels. Secondly this ASIC takes die area that one cannot use for anything else, whether one plans to use the XADC or not.

Another solution is an ADC implemented directly into reconfigurable hardware, that eventually can be used as any other IP. This would guarantee the advantage of having the digital information directly available inside the FPGAs, allowing faster processing of the digitized data. Furthermore, such an ADC could be interfaced easily and in a flexible way, allowing, for instance, to adjust the number of ADCs to the needs of the target application. Finally, the overall size of the system would be smaller allowing for an easy integration also in compact devices.

In this paper, we tackle this problem and we propose an ADC architecture which can be implemented into reconfigurable devices. Except for a small single resistor needed to create the analog reference ramp (to date, the creation of analog signals inside the FPGA is not possible), the whole ADC is completely implemented in the FPGA. The design comprises two TDCs and a $LVDS$ transceiver, and it is synthesized, placed, routed, and tested on a Spartan 6 FPGA.

Similarly to soft-core processors, which are used even in FPGAs with integrated hard-core processors, our soft-core

FPGA'15, February 22–24, 2015, Monterey, California, USA.

http://dx.doi.org/10.1145/2684746.2689070.

ADC architecture complements and extends the possibilities offered by hard-core ADCs.

The remainder of this paper is organized as follows. In Section 2 we review the state-of-the-art of low cost ADCs and TDCs implemented using reconfigurable devices. In Section 3 we present our architecture for implementing ADCs using FPGAs. Finally Section 4 reports the area occupation and the performance of our design and compares them with results reported in literature.

2. RELATED WORKS

In the past several works addressed the problem of creating low cost, highly precise and high speed digital converters using reconfigurable hardware. Most of them address the problem of time-to-digital conversion, but few attempts of implementing ADCs were also undertaken.

Favi and Charbon [1] proposed the first picosecond resolution TDC implemented using a FPGA. The main problem which designers have to face, while implementing a TDC using reconfigurable hardware, is how to reach a time resolution greater than the clock frequency. This is usually obtained with a so called thermometer. A number of registers is connected to build a chain. Each of the registers has a specific and small propagation delay towards the next register.

At the beginning of the measurement, a '1' is written to the first register and, till the stop signal is high, it is propagated into the next registers. At the end of the measurement, the chain is read out, and the number of ones gives the time stamp of the thermometer. The registers are formed by a chain of Carry4 elements: the elements in the FPGA with the smallest propagation delay, as it is a single chain with a dedicated path. The same base architecture was improved in a number of follow-up works [2, 3, 4].

Previous works implementing ADCs on FPGAs are usually based on delta sigma modulators [5, 6]. In this approach, a stream of pulses is generated and compared with the analog input signal using a feedback loop. This approach however is capable of detecting changes which occurred in the analog signal rather than attempting to read its absolute value.

Attempts to implement ADCs with a TDC using single or double slope schemes were also proposed in the past. Among them, the design of Wu *et al.* [7] is probably the most relevant for our work. The ADC is implemented using a reduced number of external components (3 resistors and 1 capacitor) and the reference ramp is generated using this passive RC network. The comparators are built using the differential inputs of the FPGA, and the signals are digitized using TDCs based on different phases of the clock. However, the performance of this design was still very limited, and the level of integration and compactness was limited by the amount of components, which were not integrated into the FPGA.

3. SYSTEMS ARCHITECTURE

The proposed architecture is depicted in Figure 1. The internal structure includes a $LVDS$ transceiver, acting as comparator, two carrychain TDCs for the time measurement (indicated as TDC 1 and TDC 2), and the logic needed for controlling the clock (Digital Clock Management Tiles or DCM) and readout. Furthermore two inverters are added

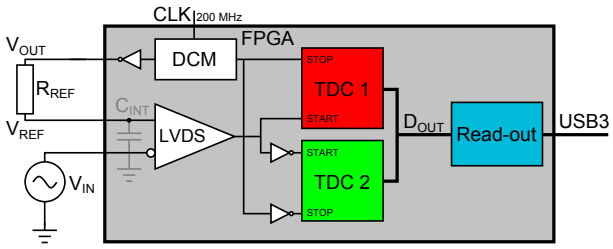


Figure 1: High level block diagram of the proposed analog-through-time-to-digital converter. The architecture comprises two TDCs, one $LVDS$, a DCM and a resistor R_{REF}. The analog signal V_{IN} is digitized into the digital signal D_{OUT}.

to measure both the rising and falling edge of the comparator.

Additionally, an external resistor R_{REF} is needed to generate the reference voltage. The system has three inputs (the input analog signal V_{IN}, the reference voltage generated by the RC circuit V_{REF}, and a clock signal CLK operating at 200 MHz) and two outputs (the digital conversion of the input analog signal (D_{OUT}), and the output voltage (V_{OUT}) used to generate the reference voltage for the $LVDS$ comparator).

The waveforms representing the timing diagram of the whole systems are depicted in Figure 2; the clock signal controls the overall operation of the circuit. It is used as stop signal for TDC 1 and, inverted, as stop signal for the TDC 2. Additionally, again inverted, it drives the output signal V_{OUT} connected to the RC network, generating a semi-exponential ramp. The output voltage is set to 3.3 V, whereas the maximum voltage reached on the input is 2.5 V. This ensures that the ramp operates on the most linear possible regime on the RC curve.

The input voltage and reference voltage are compared with the $LVDS$ transceiver. A '1' is generated when the reference ramp V_{REF} becomes higher than the input voltage V_{IN}. The time interval between a '0'→'1' transition caused by the $LVDS$ and the next rising edge of the clock

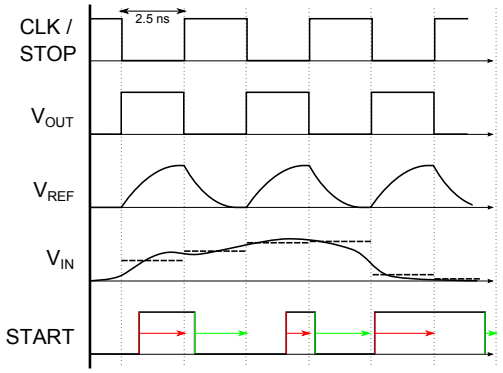


Figure 2: Timing diagram of the proposed ATDC. The reference clock signal is on top. The clock signal, inverted, is also used to drive V_{OUT}, and to generate V_{REF}. The analog input signal V_{IN} is sampled two times per clock period by comparing with V_{REF}.

is measured by TDC 1 and stored in a register. The time interval between the '1'→'0' transition caused by the $LVDS$ and the next rising edge of the inverted clock is measured in TDC 2 and again stored in a register.
The two timing intervals measured by TDC 1 and TDC 2 are then added. This addition guarantees that our design produces a single digital word corresponding to each analog input voltage with a 200 MS/s rate. It also allows to achieve one additional bit of resolution and to filter noise. The converted signals are finally transferred to the external world, in this case a standard personal computer, using the USB protocol implemented using an FX3 controller [8].

3.1 Implementation

A more detailed block representation of our ADC is depicted in Figure 3. From the figure, we can see the carrychains, the flip-flops, the thermometer decoder, and the read-out. In each clock period the $LVDS$ generates a '1'. This signal is delayed through the carrychain, implemented using Carry4 blocks. The carry value in each Carry4 block is latched on the next clock edge. The time stamp measured with this thermometer is converted into a binary time stamp. The decoder for the thermometer is a decoder implemented using multiplexers that select to forward the upper or the lower part of the thermometer depending on the value stored in the middle bit of the thermometer, as will be detailed in the rest of this section.

The main reason to use the carrychains is the fact they are the fastest elements available in FPGAs. The carry-to-carry delay is approximately 20 ps (the exact value is obviously dependent on the specific FPGA platform).

To reach this performance, it is for optimal performance best to manually place the TDCs carrychains. Manual placement avoids (many) clock domain crossings and ensures the best possible linearity. The rest of the whole design is automatically placed and routed using standard FPGA design tools.

To measure the analog input voltage with a TDC, the analog signal has to be converted into the time domain. This step requires a small external circuit. By using the parasitic capacitance of the system, we can achieve a high sample rate for our ADC. The internal pin capacitance is dominated by the PCB, therefore it is different from board to board. In

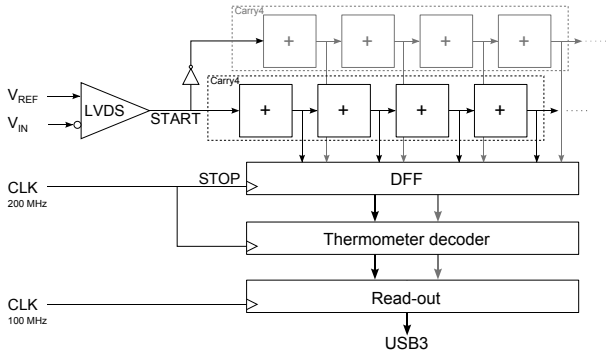


Figure 3: More detailed block diagram of our ATDC. It is possible to see the thermometer implemented using Carry4 elements, the LVDS, the flip-flops, the thermometer decoder, and the interface for the readout.

our setup, the C_{INT} was measured to be in the order of 50 pF. By combining C_{INT} with an external resistor R_{REF} of 100 Ω we can achieve a ramp with a rise time τ of approximately 5 ns. This is indeed larger than the half of the clock period and it ensures the maximum linearity of the input ramp.

3.2 Resources

Considering the target application of foreseen for our ADC, we synthesized our design using a low cost FPGA (Spartan 6 [9]). The reported results were obtained using a XC6SLX100-3FGG676 FPGA on a custom PCB, augmented with an onboard clock crystal which generates the system clock signal at a frequency of 200 MHz. The design occupies 366 FPGA slices, roughly 2% of what is available in the target platform. The power consumption, estimated with the Xilinx XPower Analyzer, is 410 mW (not considering, however, the power dissipated by the data transfer). The onboard clock jitter (σ), measured with a LeCroy WaveMaster 8600A, was measured to 30 ps after going through the FPGA's DCM tiles.

The reference ramp created through the resistor is driven by the same clock, thus the same jitter applies. However since the measurement is started from one clock period and finishes at the next one, the measurement jitter will be in the order of 40 ps. This jitter does not meet the ADC's jitter requirement:

$$\text{Jitter} < \frac{1}{2^q \pi f_s}, \tag{1}$$

which states that the jitter should be < 22 ps (for a 7 bit ADC). Therefore for a higher performance a clock with lower jitter is needed, although for our application this precision is not required.

The noise introduced into the system by the 100 Ω resistor follows Boltzmann's law $V_{noise\ RMS} = \sqrt{4kTRB} < 20\,\mu V$, at $40°C$ and a 200 MHz bandwidth. This is significantly less than the quantization error, therefore it can be ignored. The resistor does not require a strict tolerance as this is a one time calibration problem. Therefore a simple and cheap resistor will fulfil our needs.

3.3 Calibration

Measuring the transfer curve of the ADC, the input - output relation will produce the uncalibrated result as shown in Figure 4. Each measurement has a standard deviation error σ of 1.9 LSB, see Subsection 4.1. The resolution of the ADC, found by fitting through the transfer curve, is 17 mV = 1 LSB. The digital range spans 7.2 bits. The performance of the system is bound by the quantization error, where the signal-to-quantization noise ratio is

$$SQNR = 20 \cdot \log_{10}(2^q) \tag{2}$$

For a system with 7.2 bits, the $SQNR$ = 43 dB. This will be reduced by jitter and non linearities. The result can be eventually improved by applying a number of calibration steps to the system.

3.3.1 Bubbles

A problem to address is that of the so-called bubbles. Bubbles are unwanted 'gaps' in the thermometer code. For instance, a consistent thermometer code is

$\cdots$0000111111$\cdots$

the carry propagates right to left.

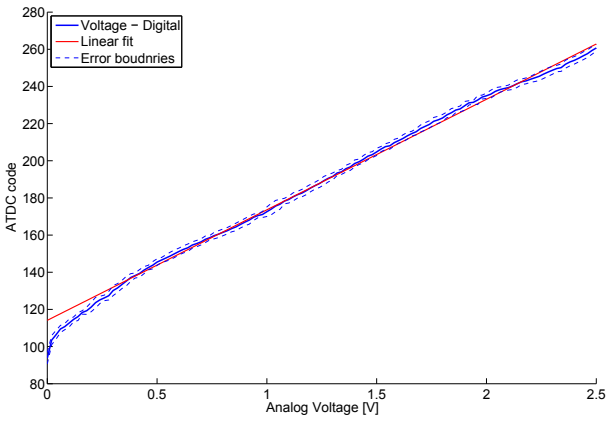


Figure 4: Analog voltage vs. digital output (before calibration). The figure shows the measured transfer function (blue), the ideal transfer function (red) and the error on the measurement (dashed). It is possible to notice the non perfect linearity of the deviation between the measured and the ideal transfer functions.

On the contrary, a bubble is present in the following code:

$\cdots$0001011111$\cdots$

Bubbles are caused by different phenomena: fast carry look ahead, clock slew, slack, and setup and hold time violations are the most common ones. The most common approach to correct a bubble is to fill the gap by adding the missing '1', as for instance in the following code:

$\cdots$0001111111$\cdots$

This however causes an overestimation of the original value which had to be stored in the thermometer. As a result, if the bubbles always occur in the same way at a delay, i.e. they are consistent with respect to time, the bubble should not be corrected at all, as it is intended behaviour of the carry structure.

For this reason, the last stages of our thermometer to binary decoder are not based on the same structure based on multiplexers, but on a simple bit counter. The bits that are high are counted to give the exact amount of all bits that were '1' in the thermometer code without filling any of the potential gaps caused by bubbles.

3.3.2 Dithering

From a density test, we can estimate the frequency of the ADC bins, where frequency refers to the rate at which a given event occurs in a particular bin. This frequency is estimated assuming that the non linearities of the system are the ones discussed in Subsection 4.2. When the bins are uniform, namely when all the bins have size equivalent to the same time interval, all bins would have the same frequency. However some carry blocks are faster than others, as can be noticed also from the simulation reported in Table 1. This is the case, for instance of bin 31 with a delay difference of 36 ps. Those large bins cause large non linearities in the TDC output. Dithering is a way to pseudo compensate the bin frequency. Dithering is a random non linear mapping, in which each bin is randomly divided over multiple bins dependent on the bin delay. The operating principle is shown in Figure 5.

Using this approach, bin 31 can be divided into three bins,

Table 1: Delay of the carrychain simulated with Xilinx Spartan 6 post place and route model. For each bin, we report the exact delay in ns and the difference from the previous bin in ps. It is possible to notice the large difference between the delays of the different bins (bin 31 in particular has an exceptionally large delay).

Bin	Delay [ns]	Delay difference [ps]
⋮	⋮	⋮
28	1.243	18
29	1.256	13
30	1.266	10
31	1.302	**36**
32	1.319	17
33	1.333	14
⋮	⋮	⋮

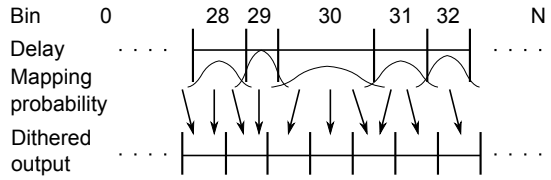


Figure 5: Dithering applied to the carrychain output. The mapping between the actual value and the dithered one is obtained by estimating the delay of each bin with the ADC density test of Subsection 4.2. Dithering allows to correct the delay of the carrychain by applying a Gaussian map between the estimated bin delay and a fixed bin width.

where the side bins partially overlap with the original bins 30 and 32 to equalize the delays. The probability that bin 31 corresponds to a specific time is a Gaussian probability function. It must be noted that, although dithering introduces random noise, it increases the linearity of the system as the bins are more equally spaced in time.

The transfer function after adding dithering and offset correction is given in Figure 6. There is no downsampling or any other technique applied to improve linearities. Comparing Figure 6 with Figure 4, a more linear characteristic can be immediately noticed. This comes at the expense of a slightly larger deviation on the measurements (due to dithering) of 0.1 LSB and a slight increase in random noise.

3.4 Design alterations

The design presented can be tailored to the specific needs of the target application. The most important parameters which a designer can trade are:

- **Sample rate**: Reducing the sample rate to 100 MS/s increases the resolution of 1 bit. A sample rate of 50 MS/s adds an additional bit.
- **Resources**: Reducing the required resources by using only one TDC (instead of two) decreases the resolution by 1 bit. By decreasing also the sample rate to 100 MS/s the resolution is unchanged.
- **Voltage range**: The input voltage range (0-2.5 V) can be adjusted by increasing or decreasing the output drive strength. The same result can be achieved by changing the resistor value.

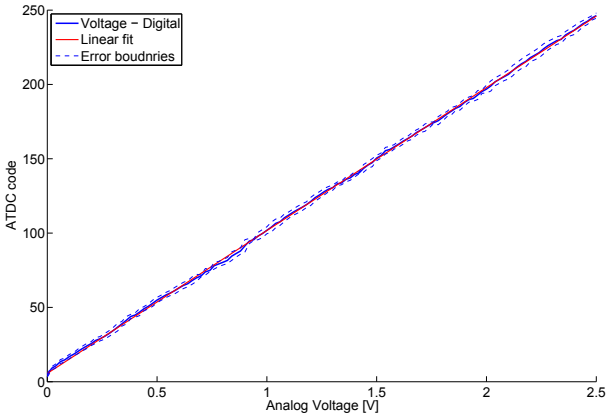


Figure 6: Analog voltage vs. digital output (after calibration). Comparing this plot with Figure 4, it is possible to notice that the calibration increases the linearity of the transfer function.

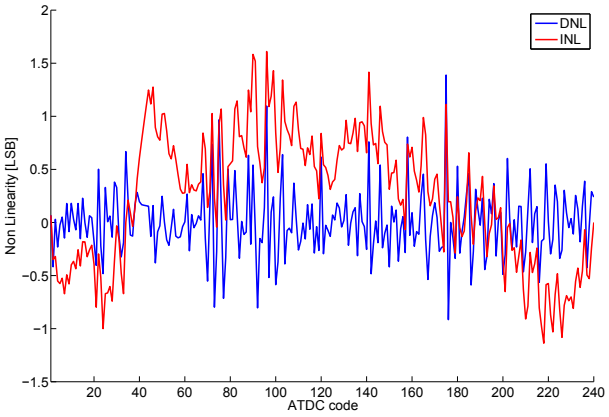


Figure 8: Non linearities (DNL and INL) extracted from ADC density test. The ADCs density is produced by applying a ramp that exceeds both negative and positive input range to the input of the ADC.

4. RESULTS

In this section we report and discuss the results of the full characterization of our architecture, implemented using a configuration consisting of a single channel ATDC.

4.1 DC measurement

The first experiment consisted in a single shot measurement. A DC voltage is applied to the input of the ADC. This measurement shows the fluctuation of measuring a single voltage; i.e. the measurement error or jitter.

After accumulating single shot statistics, we estimate the error by applying a Gaussian distribution. The result is shown in Figure 7. Over the entire measurement campaign, the average error was $\leq$ 2 LSB ($1{\cdot}\sigma$).

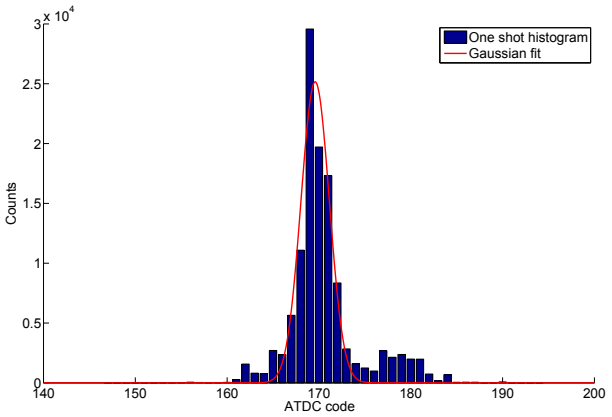


Figure 7: Single shot measurement obtained by applying a DC voltage to the ATDCs input. The measurement error over the entire measurement range, calculated using a Gaussian fit, is $\leq$ 2 LSB ($1{\cdot}\sigma$).

4.2 Non linearities: density test

The non linearities can be identified in different ways. One possible way is to compare the ideal and the measured transfer curves. Another is to generate measurement points over the entire input range. As a result, a ramp that exceeds both the positive and the negative ADC input range is generated. In case of a perfect linear system, the points are equally spread over the entire ADC range. In presence of a non linear transfer function, some bins have more counts than others. The results in each bin are summed to compute the ADC density. Each code n has a total counts $C(n)$. In an ideal system, the counts in each bin would be equal to the average $\overline{C} = \frac{\sum_{i=1}^{N} C(n)}{N}$.

From the ADC density the differential and integral non linearities (DNL and INL) can be found using the following formulas:

$$\mathrm{DNL}(n) = \frac{C(n)}{\overline{C}} \tag{3a}$$

$$\mathrm{INL}(n) = \sum_{i=1}^{n} \mathrm{DNL}(i) \tag{3b}$$

The DNL and INL for the system are in the range [-0.9 1.4] LSB and [-1.1 1.6] LSB, respectively.

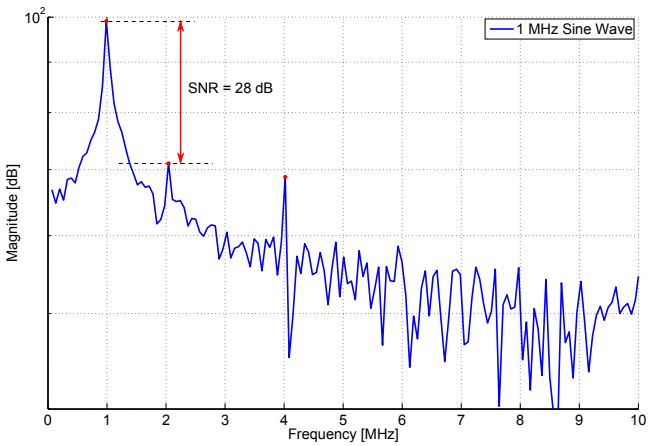
The main contributors to DNL and INL in the system are the intrinsic non linear transfer function of the reference ramp, the clock distribution in the FPGA and the unevenly spaced delays of the carry blocks. The last two factors are intrinsic FPGA problems and occur everywhere in the device. They can be partially overcome by applying dithering, as discussed. However in the placement of the delayline it is essential to avoid so-called bad spots: particular carrychains that exhibit higher non linearities and have lower resolution.

4.3 AC measurement

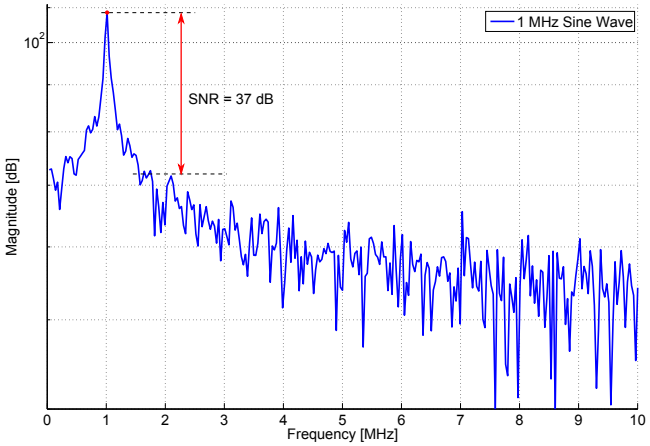
The AC measurement allows to estimate the signal-to-noise-and-distortion-ratio ($SNDR$) and the effective number of bits ($ENOB$). To do so, we applied a 2.2 V_{pk-pk} sine wave with a 1 MHz oscillation frequency.

The external AC signal of 1 MHz is generated with a Rohde & Schwarz HMF2550 function generator. The signal has a jitter of 4 ns (implying a 0.25 MHz deviation). All harmonics are suppressed > 40 dB, as measured with a LeCroy WaveMaster 8600A oscilloscope.

The signal in time domain, as measured with the implemented ATDCs is plotted in Figure 10. This is a result



(a) Before calibration.



(b) After calibration.

Figure 9: FFT of the 1 MHz 2.2 V_{pk-pk} sinewave for $SNDR$ estimation.

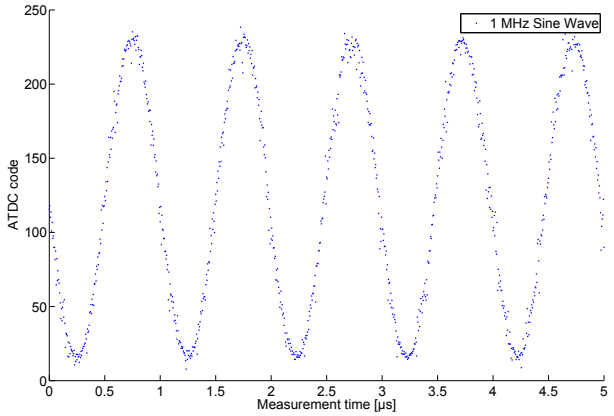


Figure 10: 1 MHz 2.2 V_{pk-pk} sinewave digitized at 200 MS/s.

after calibration.

Using the FFT, the frequency domain plots of Figure 9 are derived. They show the frequency components in the input signal, respectively before and after calibration.

The peak at 1 MHz is approximately 37 dB above the noise threshold (this result is obtained after calibration). From this $SNDR$, the $ENOB$ can be calculated with Equation 4 to be approximately 6 bits.

$$ENOB = \frac{SNDR - 1.76}{6.02} \quad (4)$$

All the harmonics of the 1 MHz sinewave are below the noise threshold. This result is compared with the measurement obtained before calibration. The harmonics are above the noise threshold and the $SNDR$ is only 28 dB. Therefore calibration significantly enhances the result with an increase in $ENOB$ of over 1.5 bit.

Although both $SNDR$ and $ENOB$ are improved by calibration (and thus dithering), the noise floor is increased due to the random noise added in the dithering process.

For higher frequencies, up to the Nyquist sampling rate, the $SNDR$ drops to roughly 34 dB.

4.4 Comparison

Our design is compared with existing implementations of reconfigurable ADCs on FPGAs and the built-in XADC. Other implementations are based on Clock phase TDC [7] and delta sigma modulation [5, 6]. The XADC is a dual-channel ADC integrated in the 7 Family of Xilinx FPGAs [10]. A comparison is given in Table 2. The details in the two works on reconfigurable ADCs we used for comparison, are generally not sufficient to carry out an exhaustive comparison. Nevertheless, the comparison of several parameters, such as sampling rate and resolution, can still be carried out in a meaningful way.

A first comparison between the implementation based on Clock phase TDC and our work, shows that our design is capable of reaching both a higher resolution and a much higher sampling rate. An immediate fair comparison with the implementation based on delta sigma modulation is not possible. Nevertheless we can extrapolate the performance of our design assuming that it is running at a lower sampling rate. As described in Subsection 3.4 by halving the sampling rate, an additional bit in digital resolution is gained. Extrapolating this trend, an $ENOB$ of 9 bits can be achieved by 2^3 divisions of the sampling rate to 25 MS/s, outperforming the design based on delta sigma modulation in speed by a factor of 20, even assuming an additional 2× division of our sampling rate.

As a result, to the best of our knowledge, the ADC design proposed in this paper outperforms the sampling rate of all previously reported ADCs implemented using reconfigurable logic.

For completeness, we compared our design with ASIC implementations, with the FPGA built-in XADC as reference. As also shown in Table 2 the XADC has an $ENOB$ of 10 bits with a maximum sampling rate of 1 MS/s. The XADC is limited to two channels, whereas our implementation can scale to a number of channels limited only by the slice utilization. In contrast to reconfigurable FPGA ADCs, this built-in ADC doesn't require any additional external com-

Table 2: Comparison between our architecture and previous ADCs implemented in reconfigurable hardware. Additionally we compare with the XADC available in Xilinx 7 series FPGAs. We use *n.a.* to indicate that the corresponding data is not available.

ADC based on:		Carrychain TDC [this work]	Clock phase TDC [7]	Delta Sigma Modulator [5, 6]	XADC [10]
Clock speed	[MHz]	200	360	100	
Conversion rate	[MS/s]	200	22.5	0.5 to 0.05	1
Voltage range	[V]	0 - 2.5	0 - 3.3	0 - 3.3	0 - 1.0
Digital range		7.2 bit	6 bit	10 to 16 bit	12 bit
ENOB		6 bit	n.a.	9 bit	10 bit
Resolution (LSB)	[mV]	17	52	3	0.25
DNL	[LSB]	[-0.9 1.4]	n.a.	n.a.	$\pm$ 1
INL	[LSB]	[-1.1 1.6]	n.a.	n.a.	$\pm$ 2
Error σ	[LSB]	2	n.a.	n.a.	1
Power consumption	[mW]	410	n.a.	n.a.	n.a.
FPGA slices		$<$ 400	n.a.	$<$ 80	dedicated
External components	n.a.	1	4	2 - 3	0

ponents nor any FPGA slices. This of course is comes at a cost of an additional space requirement on the FPGA die.

Moreover, our design has the possibility to reach both a higher resolution and conversion rate, whereas XADC is fixed in all ways.

For the comparison with other ASIC ADCs, we referred in particular to the tables collected by Murmann [11], who keeps a yearly updated ranking of ADCs published in major design conferences such as the International VLSI symposium [12] and ISSCC [13]. The results of our comparison are reported in Figure 11. The figures show an immediate visual positioning of a particular design for ranking different Figure of Merits commonly used for ADCs, such as speed, resolution and power consumption.

Obviously, the comparisons of our design, implemented using reconfigurable logic, with ASIC implementations is unfair, as it is well known that even state of the art FPGAs cannot reach the performance achieved by the more recent ASIC designs reported. However, we can see that our design is already comparable to ASIC implementations of ADCs realized with somewhat older technologies.

Another interesting information from these graphs is the trend for ASICs: we are moving towards smaller feature sizes, with the direct consequences that the designs will achieve lower power consumption and higher speeds. A similar trend can be expected for FPGAs: the next generations will be based on smaller geometries. Because of this, we also expect for our design a reduction in power consumption as well as an increase of the speed.

5. CONCLUSIONS

A novel architecture for a high speed ADC was presented, using low cost Spartan 6 FPGA families, and fully characterized. Our design can convert up to 200 MS/s with a 7 bit resolution. Additionally, our system shows a high linearity (DNL [-0.9 1.4] LSB and INL [-1.1 1.6] LSB), an $ENOB$ of 6 bit (37 dB $SNDR$) and can be implemented using only 366 FPGA slices (2% of what is available). To the best of our knowledge, this is the fastest ADC fully implemented in a FPGA reported. Our system enables an easy integration between the analog world and FPGAs, without the need of external ADCs, paving the way to a large number of low cost industrial, medical, and sensing applications where analog signals have to be converted in a cheap and reliable way.

6. SUPPLEMENT

The VHDL code and accompanying documentation is available on: `http://cas.tudelft.nl/fpga_tdc/`.

7. ACKNOWLEDGEMENTS

The authors are grateful to Xlinix Inc. for hardware donations.

8. REFERENCES

[1] C. Favi and E. Charbon, "A 17 ps Time to Digital Converter implemented in 65 nm FPGA technology," in *Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, ser. FPGA '09. New York, USA: ACM, 2009, pp. 113–120.

[2] H. Menninga, C. Favi, M. Fishburn, and E. Charbon, "A multi-channel, 10 ps resolution, FPGA-based TDC with 300ms/s throughput for open-source PET applications," in *Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2011 IEEE*, Oct. 2011, pp. 1515–1522.

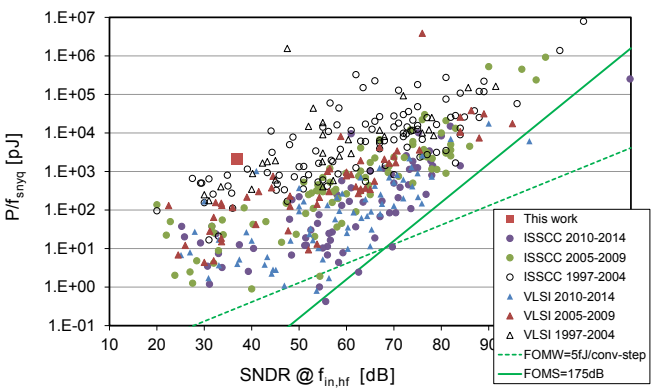
[3] S. Bourdeauducq, "Time to Digital Converter core for Spartan 6 FPGAs," Nov. 2011. [Online]. Available: http://www.ohwr.org/attachments/855/tdc_v3.pdf

[4] J. Wu and Z. Shi, "The 10 ps wave union TDC: Improving FPGA TDC resolution beyond its cell delay," in *Nuclear Science Symposium Conference Record, 2008. NSS '08. IEEE*, Oct. 2008, pp. 3440–3446.

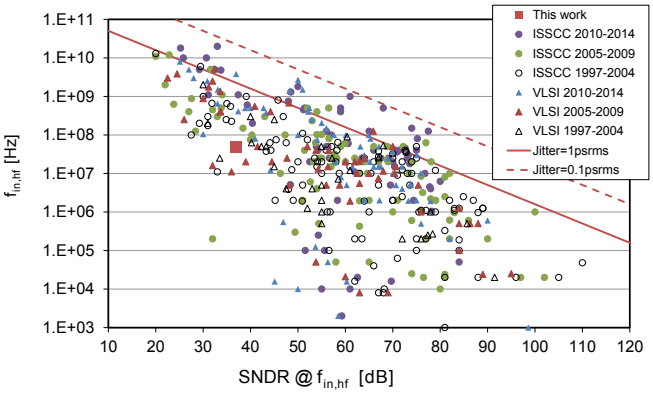
[5] "Leveraging FPGA and CPLD digital logic to implement analog to digital converters," Lattice Semiconductor, Mar. 2010. [Online]. Available: http://www.latticesemi.com/~/media/Documents/WhitePapers/AG/CreatingAnADCUsingFPGAResources.PDF?document_id=36525

[6] "Integrated ADC for Altera Cyclone-IV devices," Missing Link Electronics, Apr. 2011. [Online]. Available: http://www.missinglinkelectronics.com/devzone/files/papers/MLE-TB20110419.pdf

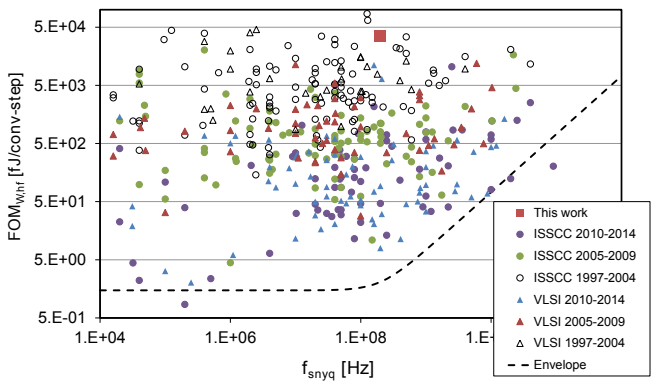
[7] J. Wu, S. Hansen, and Z. Shi, "ADC and TDC implemented using FPGA," in *Nuclear Science Symposium Conference Record, 2007. NSS '07. IEEE*, Oct. 2007, pp. 281–286.



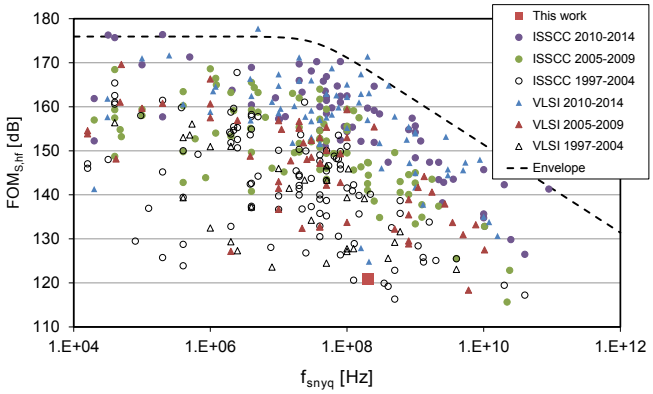
(a) Energy vs. $SNDR$



(b) Aperture vs. $SNDR$



(c) Walden FoM vs. speed



(d) Schreier FoM vs. speed

Figure 11: Graphical comparison of the ADC presented in this work (red square) with different ASIC implementations of ADCs, using the following Figures of Merit: (a) energy per conversion versus $SNDR$ (b) frequency of the measured signal versus $SNDR$. (c) energy per conversion versus sampling rate. (d) Schreier FoM = $SNDR + 10 \cdot \log(f_{sample/2}/Power)$ versus sampling rate. It is possible to notice that our design, despite the use of a FPGA, compares with other ASIC designs. The comparison is based on the survey maintained by [11].

[8] "EZ-USB FX3 SuperSpeed USB 3.0 peripheral controller," Cypress. [Online]. Available: http://www.cypress.com/fx3/

[9] "Spartan 6 FPGA family," Xilinx. [Online]. Available: http://www.xilinx.com/products/silicon-devices/fpga/spartan-6/

[10] "7 series FPGAs and Zynq-7000 all programmable SoC XADC," Xilinx. [Online]. Available: http://www.xilinx.com/support/documentation/user_guides/ug480_7Series_XADC.pdf

[11] B. Murmann, "ADC performance survey 1997-2014," June 2014. [Online]. Available: http://web.stanford.edu/~murmann/adcsurvey.html

[12] "VLSI Symposia on VLSI technology and circuits." [Online]. Available: http://www.vlsisymposium.org/

[13] "IEEE international solid-state circuits conference." [Online]. Available: http://isscc.org/