A Continuous-Time Digital IIR Filter With Signal-Derived Timing and Fully Agile Power Consumption

Yu Chen, *Student Member, IEEE*, Xiaoyang Zhang, *Member, IEEE*, Yong Lian, *Fellow, IEEE*, Rajit Manohar, *Senior Member, IEEE*, and Yannis Tsividis, *Life Fellow, IEEE*

***Abstract*— Presented is the first continuous-time (CT) digital infinite impulse response (IIR) filter working on signal-derived timing in lieu of a clock. We introduce a novel design method which enables the design of high-order IIR filters using only two tap delays. An event-grouping technique is also introduced to prevent parasitic oscillations in the presence of tap delay mismatches. The 1.2-V, 65-nm CMOS prototype implements a sixth-order IIR filter, with a maximum input rate of 20 Msample/s and a stop-band rejection of more than 80 dB. Without using any power-down circuitry, the chip's power consumption tracks the input activity in a fully agile manner, and varies by more than 50×, from 0.04 to 2.32 mW. The filter achieves an figure of merit (FoM) which competes with that of discrete-time (DT) filters, while avoiding the use of a clock and an antialiasing filter. Compared to prior art in CT digital signal processings, the prototype achieves 45-dB improvement in stop-band rejection and 9× smaller delay line area. For the first time, the filtered CT digital signal is converted to synchronous mode at the end of signal chain, allowing integration with DT digital systems.**

***Index Terms*— Asynchronous circuits, continuous-time (CT) digital filter, CT digital signal processing (DSP), energy-efficient DSP, event-driven systems, infinite impulse response (IIR).**

I. Introduction

Battery-powered systems for many applications (sensor networks, biomedical instrumentation, and hand-held devices) have usually a very tight power budget. The input signals that these systems must handle are often sporadic and exhibit time-varying activities. When operating on such signals, conventional discrete-time (DT) digital systems use a fixed clock rate determined by the maximum frequency of the input, and hence waste power when the input changes slowly or is constant. Dieter *et al.* [1], Mostafa and El-Hagry [2], and Kurp *et al.* [3] proposed systems that adaptively change the clock rate according to the input activity. However, these systems still require always-on base clocks, resulting in suboptimal solutions. A scheme termed continuous-time (CT) digital signal processing (DSP) avoids such issues by eliminating the clock [4], [5], using only signal-derived timing. An accompanying CT analog-to-digital converter (ADC), e.g., a level-crossing sampler [6], converts an analog input signal into a CT digital signal (i.e., a set of binary CT waveforms) whose activity tracks the input. A data update in the CT digital signal is called an event. The inter-event spacing is nonuniform and is a key part of the signal representation. A CT finite impulse response (FIR) DSP filters the CT digital signal by scaling and summing the delayed versions of the input, using clockless digital circuits [6]–[8]. CT delay lines are used to delay the CT digital signal without compromising its timing. Operations in a CT DSP are event driven, hence power consumption responds to signals in a fully agile manner. Since no uniform sampling is performed in the absence of a clock, such systems are free of aliasing. Previous work on CT DSP was limited in several respects. First, only FIR filters were successfully demonstrated [6]–[8]; the only attempt at CT digital infinite impulse response (IIR) filters suffered from high noise and from large discrepancies between the measured and designed frequency responses [9]. Second, all previous CT DSPs required power-hungry and area-consuming delay lines, their number being equal to the order of the filter. This imposed a limit on the filter order. Finally, the outputs of these filters were non-synchronous and hence incompatible with DT systems. We solve all of these problems in this paper. We describe the first CT digital IIR filter with accurate frequency response, using a small number of delay taps independent of filter order, and with a synchronous output that allows interfacing to DT systems. The resulting CT DSP can directly process various types of time-based signals—sigma–delta modulated, pulse width modulation, pulse frequency modulation, etc., whether synchronous or not. This paper expands on the results presented in [10] by the following:

i) discussing design considerations when delay line mismatches are present, and explaining in detail

Manuscript received August 2, 2017; revised September 27, 2017; accepted October 22, 2017. Date of publication December 4, 2017; date of current version January 25, 2018. This work was supported by National Science Foundation under Grant CCF-0964606 and Grant CCF-1419949. This paper was approved by Associate Editor Andrea Baschirotto. *(Corresponding author: Yu Chen.)*

Y. Chen was with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA (e-mail: yuchen@cisl.columbia.edu).

X. Zhang is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583.

Y. Lian is with the Department of Electrical Engineering and Computer Science, Lassonde School of Engineering, York University, Toronto, ON M3J 1P3, Canada.

R. Manohar is with the Computer Systems Lab, Yale University, New Haven, CT 06520 USA.

Y. Tsividis is with the Department of Electrical Engineering, Columbia University, New York, NY 10027 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSSC.2017.2769339

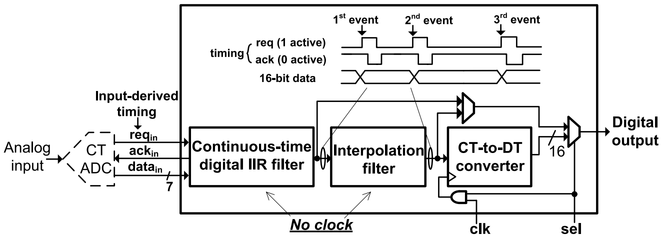


Fig. 1. Block diagram of the design.

a data grouping method that prevents parasitic oscillations;

ii) providing extensive circuit design information;

iii) describing the detailed design of a CT-to-DT output converter;

iv) providing more measurement results.

The remainder of this paper is organized as follows. Section II gives an overview of the system architecture. Section III analyzes a fundamental issue in CT digital IIR filter and provides a practical solution. Section IV presents the architecture of the IIR filter. Section V describes the interpolation filter and the CT-to-DT converter. Section VI gives the circuit implementation of various blocks. Section VII presents measurement results. Section VIII concludes this paper

II. Overall System Architecture

Fig. 1 (solid lines) shows the block diagram of the implemented system. The CT digital IIR filter implements the desired transfer function. An interpolation filter composed of CT digital FIR sections suppresses the noise and distortion power outside the baseband of the IIR filter's frequency response. The CT-to-DT converter converts the filtered signal into a synchronous digital output compatible with conventional DT systems. Both a CT digital output and a synchronous digital output are made accessible at the output of the system.

All signals between and within the blocks are CT digital signals with 2-bit timing information (*req* and *ack*) and a data bus. A four-phase bundled data protocol [11] is used for communication to ensure robustness. More details about four-phase bundled data are provided in Section IV-A. A rising edge on the high-active *req* denotes the occurrence of an event; a falling edge on the low-active *ack* means that an event is accepted by the following block. Although asynchronous digital techniques are used, a CT DSP is different from an asynchronous DSP. A standard asynchronous DSP operates on uniformly sampled input data. Delays between samples do not represent information; in fact some robust asynchronous circuit families (such as quasi delay insensitive (QDI) circuits) are designed to perform computation insensitive to gate delays. A CT DSP preserves the nonuniform inter-event spacing during processing, such spacing being an integral part of a CT digital signal representation. Except for the 7-bit input ($data_{\mathrm{in}}$), 16-bit word length is used everywhere else to minimize truncation error. The signal-derived timing req_{in} from the preceding CT ADC triggers the operations in the entire system, which is thus completely event driven.

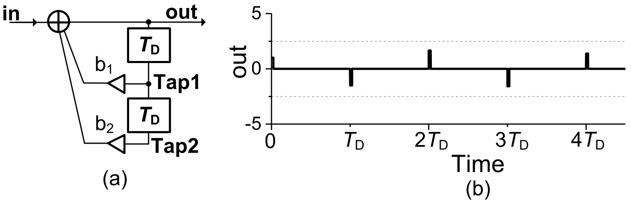


Fig. 2. (a) Ideal second-order IIR filter. (b) Unit pulse response for $b_1 = -1.5$ and $b_2 = -0.6$.

III. Preventing Parasitic Oscillation in CT Digital IIR Filters

A. Parasitic Oscillations and Their Causes

In contrast to DT digital filters, which use a clock to realize the tap delays, CT digital filters rely on CT delay lines to produce the required timing. In filters with an order higher than one, multiple delay lines are required, and mismatches between them are of concern. In an FIR filter, mismatches only cause errors in the frequency response. In CT digital IIR filters, however, mismatches can cause parasitic oscillations, as will now be explained via an example.

Consider the second-order IIR filter in Fig. 2(a), with a unit pulse at its input. Assume that $b_1 = -1.5$ and $b_2 = 0$, so that only the smaller loop is active. This system is unstable; the upper loop gain is larger than one, which causes the output to progressively become larger and larger, without bound. However, when the lower tap coefficient is changed to $b_2 = -0.6$, the system becomes stable, as can be deduced from its unit pulse response in Fig. 2(b). The reason is that the feedback signals from the two paths, which are aligned, combine in such a way that the output is prevented from growing. The stability of this system can also be verified with conventional z-domain analysis, which shows that the system poles are inside the unit circle.

Unfortunately, when mismatches exist, the above alignment of the signals from the two taps, which had "saved the show" above, cannot be guaranteed to occur. This is illustrated using the system in Fig. 3(a), where T_{D2} is assumed to be slightly different from T_{D1} due to mismatches. The pulses being returned from the two taps no longer coincide in time, as shown in Fig. 3(b) for a mismatch of 10%. (This value is exaggerated for clarity in the graphics.) The pulses no longer combine as a group, which would have yielded the result in Fig. 2(b); each pulse appears isolated and keeps growing, as seen, which is parasitic oscillation. It can be verified that if each pulse cluster (shown around multiples of T_D) is allowed to coincide by reducing the mismatch to 0, the response of Fig. 2(b) results.

The parasitic oscillation can also be explained in the frequency domain. Defining the fundamental period as $T'_D = 0.1T_D$, and using $z = e^{sT'_D}$, the z-domain transfer function of the system in Fig. 3(a) can be written as follows:

$$H(z) = \frac{1}{1 + 1.5z^{-10} + 0.6z^{-21}}. \quad (1)$$

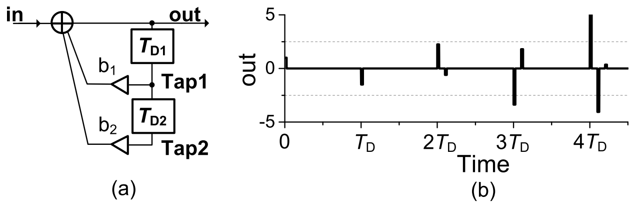


Fig. 3. (a) Second-order IIR filter with tap delay mismatch. (b) Unit pulse response for the same b_1 and b_2 values as in Fig. 2.

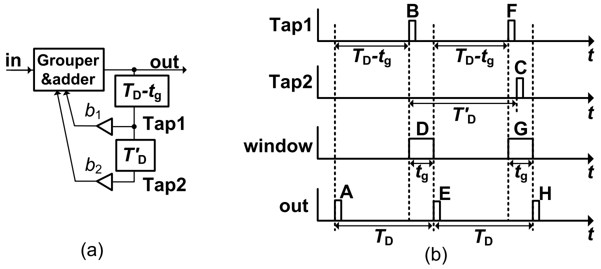


Fig. 4. (a) Second-order IIR with event grouping. (b) Timing diagram.

Some of its poles, e.g., $-1.01 \pm j0.32$, are outside of the unity circle and result in parasitic oscillation. One can prove a similar result when the delay mismatch has different values.

B. Solution

Since mismatches prevent the clusters from aligning with each other, we can force the pulses to coincide by grouping them together. This can be achieved by holding the pulses that appear at the adder inputs, and only generating an output when all anticipated pulses have arrived. We can compensate the delay t_g, which is purposely introduced to allow for grouping, by subtracting it from the first tap as shown in Fig. 4(a). As a result, the upper delay element becomes $T_D - t_g$. Fig. 4(b) illustrates the operation. Assume Pulse A at the output passes through the top delay and appears at Tap1 after a delay $T_D - t_g$, as Pulse B. This pulse does two things: 1) it goes through the bottom delay of length T'_D (of value to be discussed below), causing Pulse C to appear at Tap2 and 2) it activates a grouping window D of length t_g. Now consider the output of the grouper/adder. Since, in this example, the only pulse present in the first window is B, with no other pulse to be added to it, this pulse appears at the output of the adder, after a delay t_g, as Pulse E. The total delay caused by going from Pulse A, through the loop, to Pulse E, is $(T_D - t_g) + t_g = T_D$. Similarly, Pulse E is delayed and appears at Tap1 as Pulse F, activating a new window G. Now there are two pulses to be grouped together: F and C. The result appears at the end of the second window, as H. It can be seen from the marked intervals that H follows E with a delay T_D.

The value of T'_D is in principle not critical, as long as Pulse C falls within Window G, but both its position and the width of the window can be optimized by given information on mismatches. Thus, if τ represents the expected worst case delay tolerance, the window should be at least as wide as $t_g = 2\tau$, and T'_D should be such that Pulse C nominally falls in the center of the window, which implies that $T'_D = T_D + t_g/2$.

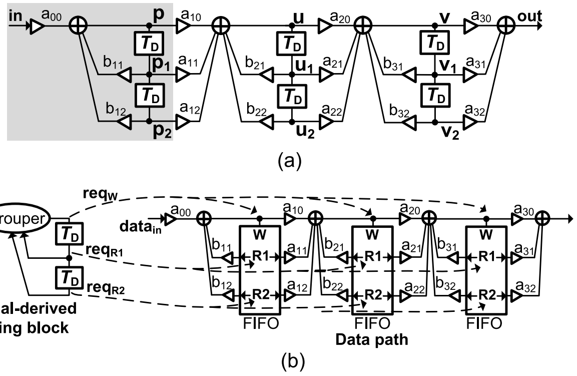


Fig. 5. (a) Sixth-order CT IIR implemented with cascaded second-order sections. (b) Implementation with shared timing signals derived from signals.

In addition to the pulses originating on Tap1 and Tap2 as in the above example, we may also have an input pulse (not shown in Fig. 4) that is also allowed to activate a grouping window. If no other pulse arrives during that window, the input pulse appears at the output after a delay t_g. If, however, a feedback pulse originated on Tap1 arrives during that window, the window is extended by t_g from such an arrival, so that the above approach, which is essential to prevent parasitic oscillations, is not compromised. At the end of this extended window, all pulses, including the input pulse, are grouped together. This can cause a small input-depended variation of the grouping delay; but as long as t_g is much smaller than T_D, the resulting distortion is small, as will be seen in Section VII.

Since pulses arriving within the same grouping window are combined together, the length of the window t_g determines the minimum interval between pulses getting out of the grouper. It also constrains the input signal granularity, which should be no less than t_g. Otherwise, two successive input events can collide in one window. Throwing away either one results in distortion. On the other hand, since the input sees a variable window length which can be slightly longer than t_g, the collision can still happen even if the granularity requirement is satisfied. If it does happen, the earlier input event is replaced by the newer one to keep the data updated. The detailed grouping algorithm is presented in Section IV-B. This algorithm has been tested with exhaustive simulations, and has been found to prevent parasitic oscillations as expected, in the presence of delay mismatches. The input collision scenario happens very rarely and the resulting distortion is small.

IV. Implementation of a Digital Signal Processor With Signal-Derived Timing

A. Ideal CT Digital IIR Filter Implementation

In previous CT digital filters [6]–[8], the number of tapped delays was the same as the filter order. Here, we introduce a new design method that allows one to implement a high-order CT digital IIR filter with only two tap delays, by means of a sixth-order example. As shown in Fig. 5(a), a sixth-order CT digital IIR filter can, in principle, be implemented with

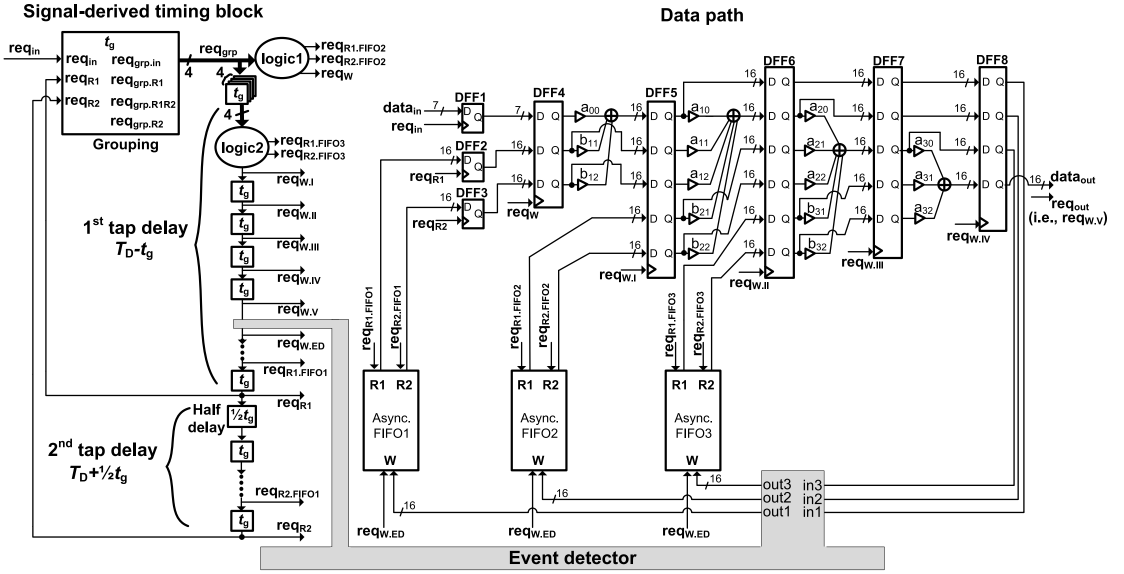


Fig. 6. Detailed architecture of the sixth-order CT digital IIR filter.

cascaded second-order sections in direct-II form. The signals at the input, output, and internal nodes of the filter are all CT digital signals containing timing bits and a data bus. An event at these nodes means an update of their data values, the timing of which is represented by a rising edge on the timing bit *req* (followed by four-phase handshaking). When an event arrives at the input in Fig. 5(a), it triggers an event at node p to reflect the data change. Assume first delay-free arithmetic. The new event at node p simultaneously triggers new events at nodes u and v. The events at nodes p, u, and v occur at the same instants. Because of identical tap delays, the events at nodes p_1, u_1, and v_1 also occur simultaneously, and so do the events at nodes p_2, u_2, and v_2. Thus, besides the input, the CT digital IIR filter has only three distinguishable timing signals, and they are all present in the first section of the filter [shadowed part in Fig. 5(a)]. Its timing path can be separated out and shared by the rest of the system, which results in the equivalent implementation of Fig. 5(b). Taking req_{in}[1] as its input, the timing block derives req_{W}, req_{R1}, and req_{R2}. These timing signals control the operations in the data path, including arithmetic operations and three FIFO operations with write (W) and two independent reads (R1 and R2) each, to generate the infinite impulse response of the IIR filter. The T_D ($2T_D$) delay in the data path is realized by writing data into an asynchronous FIFO and reading it out T_D ($2T_D$) later. This method decouples the link between the filter order and the number of necessary tap delays, and can be applied to CT digital IIR filter designs of any order. Note that the number of FIFOs in the IIR filter in Fig. 5(b) is the same as in the IIR filter in Fig. 5(a), and thus grows with the filter order.

B. Architecture of the Proposed CT Digital IIR Filter

a) Signal-Derived Timing Block: Fig. 6 (left) shows the architecture of the timing block. The grouping block and an extra half-delay cell in the second tap delay implement the grouping solution introduced in Section III. The length of the grouping window t_g (whose nominal value in this design is 25 ns) determines the minimum inter-event spacing in the feedback path. It is typically much smaller than the tap delay T_D (whose nominal value is 1 μs). In a CT DSP, the spacing between events carries information about the signal. Since the minimum supported inter-event delay is t_g, a CT delay line is constructed by using a chain of delay cells, each contributing a delay of t_g. This approach delays events without distorting their timing relative to each other. If a cell with a delay larger than t_g were used with an intention to reduce the number of delay cells, an event would have to wait at the input of a cell until the previous event gets out, which would distort the original timing between the two events. The logic1 and logic2 blocks will be explained later. The grouping block takes req_{in}, req_{R1}, and req_{R2} as its inputs and generates a 4-bit output, $req_{\text{grp}} = \{req_{\text{grp.in}}, req_{\text{grp.R1}}, req_{\text{grp.R1R2}}, req_{\text{grp.R2}}\}$.[2] The four output bits correspond to four possible grouping scenarios.

i) If a window only contains an input event from req_{in}, $req_{\text{grp.in}}$ is pulled up.[3]

ii) If a window contains an event from req_{R1} but no event from req_{R2}, $req_{\text{grp.R1}}$ is pulled up.

iii) If a window contains events both from req_{R1} and req_{R2}, $req_{\text{grp.R1R2}}$ is pulled up.

iv) The grouping solution introduced in Section III requires that the event from req_{R2} must be grouped in a window triggered by the previous event from req_{R1}. However, we retain the grouper's capability to handle req_{R2} events arriving at another time. When this situation happens, the grouping block immediately pulls up $req_{\text{grp.R2}}$ with-

[1] A complete timing signal consists of *req* and *ack*. Only *req* is discussed in this section for simplicity.

[2] $\{a_1, a_2, \ldots, a_N\}$ is the concatenation of N single-bit signal a_1, $a_2, \ldots, a_N$.

[3] For simplicity, we only mention the rising transition of *req* signals. In our actual implementation, a four-phase handshake is completed soon after this transition, pulling *req* signals back to zero so that the block can be ready to accept another event.

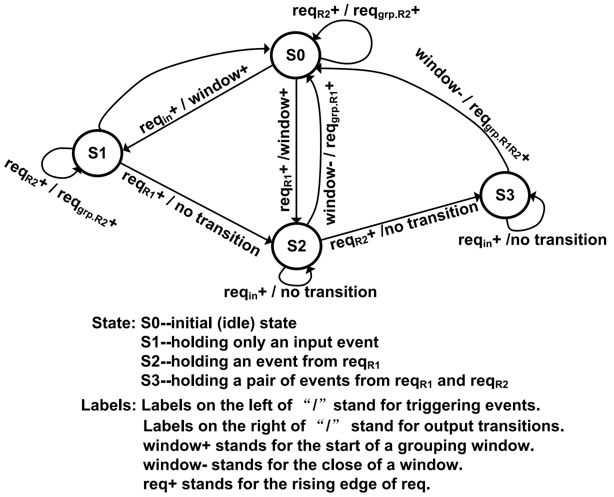

S0
S1
S2
S3
req_R2+ / req_grp.R2+
req_in+ / window+
req_R2+ / req_grp.R2+
req_R1+ / no transition
req_R1+ /window+
window- / req_grp.R1+
window- / req_grp.R1R2+
req_R2+ /no transition
req_in+ /no transition
req_in+ / no transition
State: S0--initial (idle) state
S1--holding only an input event
S2--holding an event from req_R1
S3--holding a pair of events from req_R1 and req_R2
Labels: Labels on the left of "/" stand for triggering events.
Labels on the right of "/" stand for output transitions.
window+ stands for the start of a grouping window.
window- stands for the close of a window.
req+ stands for the rising edge of req.

Fig. 7. State diagram of grouping block's operation.

out triggering a window. This capability is reserved for the event detection function which we will introduce in Section IV-B-c).

Fig. 7 shows the operations of the grouping block using an asynchronous state diagram. Its operation is described in detail in Section VI-B.

b) Data Path: Fig. 6 (right) shows the architecture of the data path. It is implemented in an asynchronous pipeline with bundled data to keep a high throughput. The adders of the sixth-order IIR filter are spread over four stages (DFF4 to DFF7), so that they can operate concurrently. The timing signals derived in the timing block are used to control the pipeline operations. As an example, consider three events, one from each of req_{in}, req_{R1}, and req_{R2}, arrive at the inputs of the grouping block within one grouping window. Upon their arrivals, the associated data are latched into DFF1–DFF3, respectively. The read operations in FIFO1, triggered by $req_{\mathrm{R1.FIFO1}}$ and $req_{\mathrm{R2.FIFO1}}$, always start t_g earlier than the triggering of DFF2–DFF3, leaving a t_g timing margin for the FIFO read. Because two feedback events are present in this grouping window, at the end of it, $req_{\mathrm{grp.R1R2}}$ is pulled up, which then pulls up req_{W}, $req_{\mathrm{R1.FIFO2}}$, and $req_{\mathrm{R2.FIFO2}}$ through the logic1 block (explained below). The rising edge on req_{W} latches all data in DFF1–DFF3 into DFF4, effectively aligning three events to one timing signal, and triggers a new set of arithmetic operations. Meanwhile, at the rising edges on $req_{\mathrm{R1.FIFO2}}$ and $req_{\mathrm{R2.FIFO2}}$ the system reads two different events, which are spaced by T_D in the time domain, from FIFO2 and prepares for the arithmetic operations in the next stage in the data path. The two read operations can happen simultaneously, because two events which are spaced by T_D arrive at the end of the first and second tap delay, respectively, at the same time. t_g later, $req_{\mathrm{W.I}}$, $req_{\mathrm{R1.FIFO3}}$, and $req_{\mathrm{R2.FIFO3}}$ are pulled up by the logic2 block, which latches data into DFF5 and triggers read operations in FIFO3. The calculated data then move forward in the pipeline with a constant inter-stage delay t_g. The final result is ready at $data_{\mathrm{out}}$ upon the rising edge of $req_{\mathrm{W.V}}$. Three intermediate results of feedback paths are stored in FIFOs when $req_{\mathrm{W,ED}}$ is asserted by the delay line after an event detector block (explained in Section IV-B-c).

Truncations are unavoidable at the outputs of each adder (except the last one) in a digital IIR filter. Moreover, the introduced truncation error may be amplified by the transfer functions of the following stages. To ensure that the power of the overall truncation error is less than the quantization error power from the input, we use 16-bit word length in the data path of this design.

As mentioned in Section IV-B-a), each of the four bits of req_{grp} in Fig. 6 (left) is asserted under different traffic scenarios. They trigger different FIFO operations and thus propagate separately in the delay line until no further FIFO operations are needed. Logic1's functions are the following.

i) $req_{\mathrm{R1.FIFO2}}$ is high if either $req_{\mathrm{grp.R1}}$ or $req_{\mathrm{grp.R1R2}}$ is high, indicating that Tap1 data will be read out from the R1 channel of each FIFO.
ii) $req_{\mathrm{R2.FIFO2}}$ is high if either $req_{\mathrm{grp.R2}}$ or $req_{\mathrm{grp.R1R2}}$ is high, indicating that Tap2 data will be read out from the R2 channel of each FIFO.
iii) req_{W} is high if any of req_{in}, $req_{\mathrm{grp.R1}}$, or $req_{\mathrm{grp.R1R2}}$ is high.

The three conditions are not mutual exclusive. Logic2 implements the same functions as logic1, but takes the delayed version of req_{grp} as its input.

We emphasize again that although an asynchronous pipeline is used, this CT DSP is fundamentally different from an asynchronous DSP, because the timing signals, which are key to the signal representation, are precisely being kept track of by the timing block.

c) Event Detector: The timing block in Fig. 6 contains closed-loop paths. If there is no data change at the arithmetic block outputs, activities regenerated in the timing block are not needed and waste energy. To remedy this, an event detector (gray in Fig. 6) is used to monitor the feedback traffic and control the timing block. Fig. 8 shows its implementation and the timing diagram. When an event enters the event detector, it triggers the left delay cells DC1 and DC2. The data on *in1*, *in2*, and *in3* are saved in DFF9 and compared with the data of the previous event stored in DFF10. The comparison delay should be shorter than $t_g/2$ so that the result (*cmp*) can be correctly latched. After a t_g delay, req_1 is pulled up. If all three values are the same, the event is identified as redundant. The signal *cmp* is set to high to prevent the event from entering the stage on the right. As a result, the redundant event will no longer be able to trigger any further operations in the filter; thus this event is effectively eliminated from the filter. A self-acknowledge is conducted to keep the asynchronous pipeline working. Only non-redundant events are written into FIFOs. By default, the comparators in the event detector take all 16 bits of the inputs for comparison. The comparator is designed such that its resolution can be programmed from 9 to 16 bit, with eight different options. When a low resolution is used, an event is more likely to be considered as redundant, and hence more power can be saved. However, a low-resolution comparison also introduces a large error power.

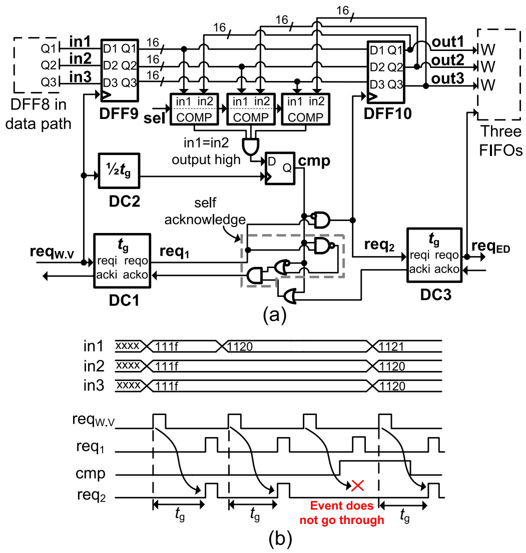


Fig. 8. Event detector. (a) Architecture. (b) Timing diagram.

The grouping solution introduced in Section III requires that the events from Tap2 should always arrive within the window activated by the previous event from Tap1. Notice from Fig. 6 that the event detector only controls the first tap delay. It is possible that after an event is eliminated from the first tap delay, its paired event in the second tap delay may arrive at the grouper outside of any window, which violates this requirement. The solution is to throw away this isolated event. When the isolated event from Tap2 arrives at the grouping block, $req_{\mathrm{grp.R2}}$ is pulled up without triggering a grouping window. Data are read out from R2 of FIFO1–FIFO3 sequentially, without being latched into any D flip flop (DFF) in the data path.

V. Output Synchronization

A. Interpolation Filter

Although the operations of a CT IIR filter track the input activity, the varying event intervals of the CT digital signal at its output prevent its integration with DT systems and limit its applications. Such integration becomes possible if a synchronous output is generated by sampling a CT signal with a clock. Because of the use of tap delays, the frequency response of the CT digital IIR filter is a function of $e^{j\omega T_D}$ and is periodic in the frequency domain [5]. For example, if the IIR's frequency response is lowpass in the baseband, it has repetitive passbands around every $1/T_D$. These repetitive passbands retain distortion and noise power from the input. They stay out of band if the final output is still a CT digital signal as in [8]. Once sampled, however, the out-of-band noise and distortion are aliased back and degrade the *signal-to-noise-and-distortion ratio (SNDR)*. A computationally efficient interpolator employing FIR filters are used before sampling to alleviate this problem (Fig. 9) [12], [13]. This interpolator is composed of four first-order FIR sections S1–S4 in Fig. 9(a), implementing notches [Fig. 9(b)] which suppress out-of-band distortion and noise power, and push away the first intact repetitive passband to $16/T_D$. Configuring the tap delays and coefficients, one can design a similar interpolation filter for high-pass and band-pass cases [12]. Because the timing signals in different sections of an FIR filter cannot be shared (in contrast to the case of IIR filters discussed earlier), we do not separate the FIR filters' timing and the data path. The architecture of the FIR filter is similar to the one in [8].

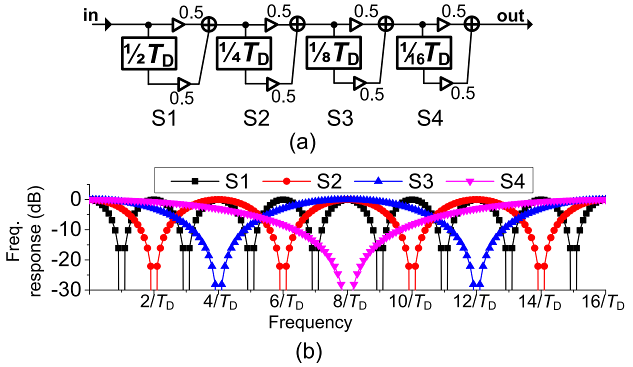


Fig. 9. (a) Architecture of the interpolation filter. (b) Frequency responses of the four sections of the interpolation filter.

B. Continuous-Time-to-Discrete-Time Converter

Directly sampling the CT digital signal with a clocked DFF coverts it into a synchronous digital signal. Traditionally two DFFs are used in cascade to minimize the possibility of the final outputs being in a metastable state [14]. Although the synchronous output is ensured to be in a stable state with good confidence, it is not necessarily in a correct stable state. If any bit settles to a wrong state, distortion can be introduced at the synchronous output. Consider an 8-bit sample as an example. Assume the sample value changes from 0000_1111 to 0001_0000. It is possible that the four LSBs change slightly earlier than the synchronization clock, while the fifth LSB changes slightly later, which results in a synchronized sample 0000_0000. A large distortion is introduced in this sample.

We can rely on the characteristics of the CT digital signal to alleviate this issue. Notice that a CT digital signal has a high effective event rate. The minimum spacing between two events is t_g. On the other hand, the tap delay of an IIR filter defines the baseband as $[0, 1/(2T_D)]$. So the maximum input frequency to the IIR filter is $1/(2T_D)$, which is much smaller than the effective event rate $1/t_g$ in this design. As a result, the amount of amplitude change between two successive events in a CT digital signal is limited to a certain range, which can be estimated by

$$\Delta V \leq \text{Max.Slope} * t_g = \frac{A_{\mathrm{FS}}\pi}{T_D} * t_g$$

where A_{FS} is the full-scale amplitude. In this design, the output resolution is 16 bit; thus $A_{\mathrm{FS}} = 2^{15}$LSB. T_D is 1 μs; t_g is 25 ns. This results in a ΔV smaller than $2^{11.3}$LSB. If the four most significant bits (MSBs) are represented in thermometer code, at most 1 bit changes at any time and hence the settling error is bounded to $\pm 2^{12}$LSB. The above-mentioned equation uses the maximum slope and is pessimistic. Measurement

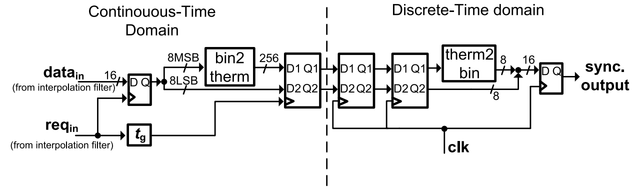


Fig. 10. Architecture of the CT-to-DT converter.

results show that ΔV is rarely larger than 2^8LSB. Thus, the eight MSBs of the CT signal are converted to thermometer code so that the settling error is bounded to $\pm 2^8$LSB. Fig. 10 shows the implementation of the CT-to-DT converter. The eight MSBs of the CT digital binary sample are converted to a thermometer code in the original CT domain. Before the thermometer-coded sample is used in the DT domain, it is latched by a DFF right after the bin2therm converter. Latching removes any glitches that have occurred in the bin2therm conversion. After synchronization, the thermometer code is converted back to binary.

VI. Circuit Implementation

A. Delay Cell

The delay cell [Fig. 11(a)] is based on the design in [15]. It uses four-phase handshaking to communicate with neighboring delay cells [11], [16]. A C-element [17] (the gate labeled "C") is used. A C-element is a memory circuit that outputs a high when both inputs are high, and a low when both inputs are low. If its two inputs are different, the output holds its previous value. Initially the delay cell is in the idle state. Both *reqi* and *reqo* are low, and *acki* and *acko* are high. Fig. 11(b) shows the timing diagram of the delay cell. The arrival of an event pulls up *reqi*. The logic high of *reqi* sets the SR latch and charges the capacitor. In the meantime, *acki* is pulled down to reset the preceding stage. When the voltage on node A falls below the threshold of the inverter, *reqo* is pulled up. The event passes from *reqi* to *reqo*. The time difference between the rising edges of these two signals is defined as the cell's delay t_g. Once the following stage accepts the event on *reqo*, *acko* is pulled down to reset the current delay cell. The value of t_g is tunable through the bias voltage of the current source M4, V_b. In this design, the nominal value of t_g is chosen to be 25 ns. A pMOS capacitor working in the depletion region, with a capacitance of 13 fF, is used. The ramp rate at node A is 15 MV/s. A half-delay cell is implemented with the same design, except that M4 is twice as wide, and hence its ramp rate at node A is twice as fast.

Power supply noise can cause the delay to vary, resulting in parasitic signals at the filter's output. A dedicated power supply line with good decoupling is used by the delay cells to alleviate this issue.

B. Grouping Block in IIR Filter

Fig. 12(a) shows the schematic of the grouping block in the IIR filter, which realizes the state diagram in Fig. 7. The schematic is composed of three channels, with interconnections between them. The feedforward and R1 channels are essentially two delay cells. The input-rising-edge t_g delay block is a simplified view of part of the delay cell in Fig. 11, including M1–M4, Cap, and Inv. The block implements a t_g delay for the rising edges at its input. The discussion of the block's operations refers to the state diagram in Fig. 7. The state of the block is stored in three registers: SR1, SR2, and DFF1. In the state of S0, Q of all latches and DFFs are low; the outputs of all C-elements are low; all *req* are low and all *ack* are high. When an input event arrives, indicated by a rising edge on req_{in} in Fig. 12(b), it triggers a delay operation in the feedforward channel by pulling up $Q1$. The block enters the state of S1. t_g later, input A of the arbiter [18] OD1 is pulled up. With $Q2$ stays low, the arbiter pulls $req_{\text{grp.in}}$ up

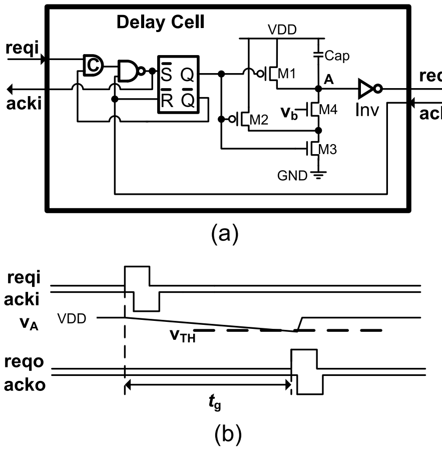


Fig. 11. Delay cell. (a) Circuit implementation. (b) Timing diagram.

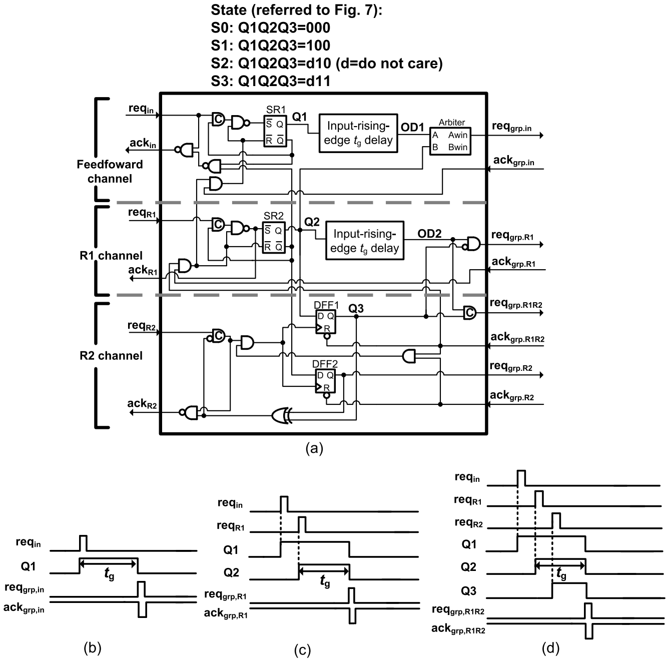


Fig. 12. Circuit implementation and timing diagrams of the grouping block in the IIR filter. (a) Schematic. (b) Grouping window with input events only. (c) Grouping window with an event from req_{R1} but no event from req_{R2}. (d) Grouping window with events both from req_{R1} and req_{R2}.

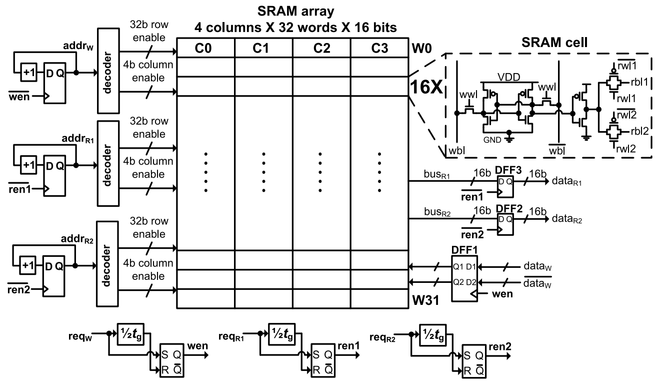


Fig. 13. Architecture of asynchronous FIFO with one write and two read channels.

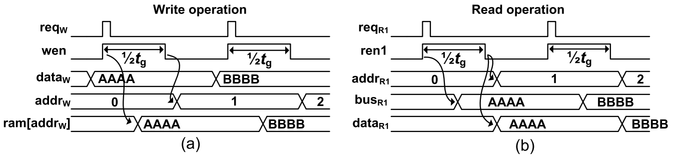


Fig. 14. Timing diagrams of write and read operations of asynchronous FIFO. (a) FIFO write operations. (b) FIFO Reads operation.

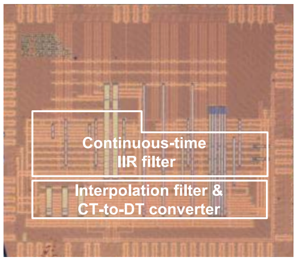

Continuous-time IIR filter
Interpolation filter & CT-to-DT converter

Fig. 15. Die photograph.

TABLE I
AREA BREAKDOWN

Core area	0.64 mm^2
IIR filter	0.38 mm^2
Delay line	0.05 mm^2
Interpolation filter	0.18 mm^2
CT-to-DT converter	0.08 mm^2

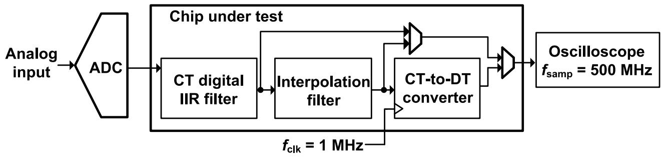


Fig. 16. Experimental setup.

and the input event is moved to the next stage. After the following stage accepts the event, $ack_{\rm grp.in}$ is pulled down to reset the feedforward channel. The block moves back to the state of S0. If an event arrives at $req_{\rm R1}$ during the state of S1, as shown in Fig. 12(c), the grouper enters S2. $Q2$ is pulled up to prevent $req_{\rm grp.in}$ being pulled up through the arbiter. Meanwhile, it triggers a delay operation in R1 channel. t_g later, OD2 is pulled up. With $Q3$ low, $req_{\rm grp.R1}$ is pulled up. However, if an event arrives at $req_{\rm R2}$ during the state of S2, as shown in Fig. 12(d), the block enters S3 and $Q3$ is pulled up. Instead of $req_{\rm grp.R1}$, $req_{\rm grp.R1R2}$ is pulled up upon the completion of the delay operation in channel R1. In either case, an acknowledge signal is received soon thereafter, to reset the block back to the state of S0. If an event arrives at $req_{\rm R2}$ while $Q2$ is low (i.e., $\overline{Q}2$ is high), DFF2 outputs a high which pulls $req_{\rm grp.R2}$ up immediately. Soon after that, $ack_{\rm grp.R2}$ is pulled low to reset DFF2. The transition of DFF2 from reset to set to reset again completes quickly without affecting the state of the grouper.

C. Asynchronous FIFO

The architecture of the asynchronous FIFO with one write (W) and two read (R1 and R2) channels is shown in Fig. 13. The FIFO depth is chosen to be slightly larger than the maximum number of events during an interval of $2T_D$. In this design, the minimum timing distance between two events is $t_g = 25$ ns, and the nominal value of T_D is 1 μs. Hence, 128 (4 columns by 32 rows) 16-bit-word cells are implemented in each FIFO. The static random-access memory (SRAM) cell is based on the fully static design in [19], but contains two read channels (triggered by $req_{\rm R1}$ and $req_{\rm R2}$, respectively). Fig. 14 shows the timing diagram of both write and read operations. They are triggered by the timing signals $req_{\rm W}$, $req_{\rm R1}$, and $req_{\rm R2}$, which are not synchronized to a clock. A write operation starts when $req_{\rm W}$ is pulled up [Fig. 14(a)]. A bundled data encoding scheme [11] assures that $data_{\rm W}$ is ready slightly before the timing signal. wen is pulled up to latch $data_{\rm W}$ into DFF1 and starts writing the cells pointed by the address $addr_{\rm W}$. After $t_g/2$, wen is pulled down and $addr_{\rm W}$ increases by one to prepare for next write operation. The write delay should be shorter than $t_g/2$ so that it can be completed before the $addr_{\rm W}$ changes. A read operation starts when either of $req_{\rm R1}$ or $req_{\rm R2}$ is pulled up. Fig. 14(b) illustrates an example when $req_{\rm R1}$ is pulled up. $ren1$ is then pulled up to read the SRAM cells pointed by the address $addr_{\rm R1}$. After a read delay shorter than $t_g/2$, data are ready on $bus_{\rm R1}$. $t_g/2$ after $req_{\rm R1}$ is pulled up, $ren1$ is pulled down and latches the new data into DFF3. Meanwhile, $addr_{\rm R1}$ increases by one to prepare for the next read operation. The three addresses $addr_{\rm W}$, $addr_{\rm R1}$, and $addr_{\rm R2}$ are increased by one at the falling edges of wen, $ren1$, and $ren2$, respectively, and are decoded into 32-bit row-enable and 4-bit column-enable signals each and select the wanted cells. Because the three decoders operate independently, the asynchronous FIFO supports one write and two reads operations simultaneously.

The FIFO used in the interpolation filter uses a similar structure but has one less read channel.

VII. MEASUREMENT RESULTS

The system is implemented in TSMC 65-nm low power CMOS process. Fig. 15 shows the die photograph. Table I shows its area breakdown. In contrast to [6] and [8], the delay

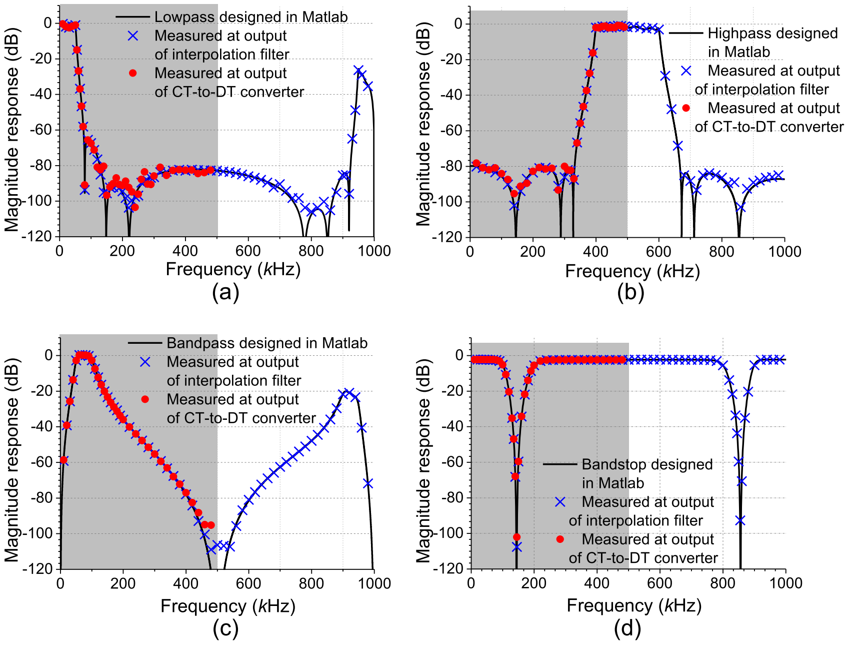


Fig. 17. Frequency responses. The gray part in each plot shows the baseband. (a) Lowpass. (b) Highpass. (c) Bandpass. (d) Bandstop.

line is a small portion of the system, both in area and power. The delay line area is 9× smaller than in [6], despite the fabrication process difference.

Fig. 16 shows the experimental setup. The system is measured with an off-chip ADC in the front, which converts analog test input to a 7-bit digital signal. The output of the chip is captured by an oscilloscope. Fig. 17 shows frequency responses of the system when the IIR filter is configured as a sixth-order low pass (LP), high pass (HP), band pass (BP) and band stop (BS), with T_D equal to 1 μs. Responses are measured both at the outputs of the interpolation filter and the CT-to-DT converter with a 1-MHz synchronization clock. Once the output is converted to a synchronous digital signal, the frequency response can only be measured up to half the clock frequency. In the LP and BP cases, the first repetitive passband around $1/T_D$ is suppressed by the notches formed in the interpolation filter. For LP and HP, the stop-band rejection is better than 80 dB; this is a 45-dB improvement over [6] and [8], due to both the IIR response and the larger number of internal bits retained. We manually tune the first tap delay in relation to a 1-μs clock period and rely on the matching of the two delay lines in the IIR filter to determine the exact delay in the second tap delay. Thanks to the event grouping method, the frequency response is insensitive to the exact value of the second tap delay. Although it is impossible to tune the first tap delay to be exactly 1 μs, a small difference has a negligible effect on the transfer function of the filter, which can be explained by the good matching between the designed and measured responses in Fig. 17. Automatic digital calibration would be straight-forward, as in [8]. With the LP configuration, the *SNDR* for full-scale input signals in the passband is 54 dB or above. The linearity of the designed filter is also tested with full-scale two-tone inputs. The in-band and out-band IM3 components are −59 and −58 dB, respectively. Fig. 18 shows the input and output spectra of the filter (before the CT-to-DT converter) with two out-band full-scale tones at 970 and 980 kHz. These two tones are in the repetitive passband of the IIR filter's frequency response, but are attenuated by the notches implemented in the interpolation filter. The output spectrum clearly demonstrates the alias-free feature of the CT digital filter. No components are aliased back into the baseband. By contrast, if the same input is fed into a DT digital filter with the same frequency response, full aliasing will be observed.

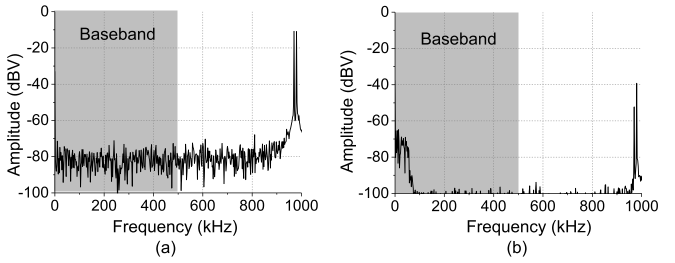


Fig. 18. Out-band two-tone test. (a) Spectrum of the filter input. (b) Spectrum of the filter output, before CT-to-DT converter.

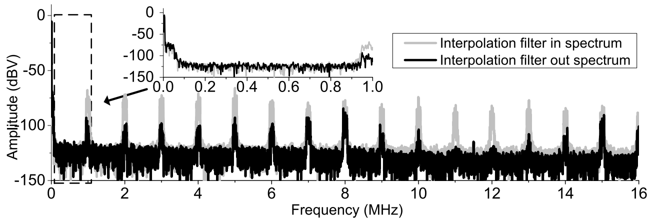


Fig. 19. Spectra at input and output of the interpolating filter, for a full-scale 5-kHz 7-bit sine wave input.

The effect of the interpolation filter is illustrated in Fig. 19. The gray curve shows the spectrum at the input of the

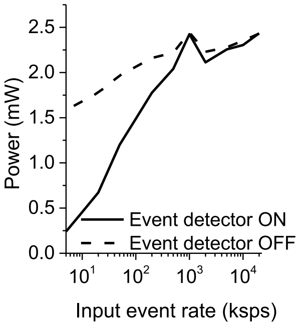


Fig. 20. Power consumption versus input event rate.

TABLE II

PERFORMANCE SUMMARY OF THE CHIP

Process/supply voltage	65 nm/1.2 V
Input resolution	7 bits
Filter coefficient resolution	10 bits
DSP arithmetic/output resolution	16 bits
T_D/t_g	1 μs/25 ns
Max. FIFO depth	128 words
Max. input data rate	20 Msps
Synchronized output data rate	1 Msps
*SNDR**	54-58 dB
In-band IM3† Out-band IM3‡	-59 dB -58 dB
Dynamic power @ 20MHz input data rate	2.32 mW
IIR filter	1.62 mW
Delay line	0.41 mW
SRAM	0.82 mW
Arithmetic blocks	0.40 mW
Interpolation filter & CT-to-DT converter	0.70 mW
Leakage	0.04 mW

*Measured after CT-to-DT converter; full-scale sinusoidal inputs in passband.

†Measured after CT-to-DT converter; full-scale dual-tone input at 20kHz and 30kHz.

‡Measured before CT-to-DT converter; full-scale dual-tone input at 970kHz and 980kHz.

Note: in both *SNDR* and IM3 measurements, the IIR filter is configured as a lowpass filter with a 50 kHz cutoff frequency.

interpolation filter (i.e., the output of the CT digital IIR filter) when the IIR filter is configured for lowpass operation as in Fig. 17(a). The testing input is a full-scale 5-kHz sine wave digitized by an off-chip 7-bit ADC at 20 MHz. Noise and distortion power are preserved at the repetitive passbands, which are around 1, 2 MHz…The black curve shows the spectrum at the output of the interpolation filter. Clearly, the noise and distortion power are suppressed by the notch formed in the interpolation filter. We acquire the CT digital data from the input and output of the interpolation filter using an oscilloscope with a 500-MHz sampling clock, as shown in Fig. 16. Since this rate is much higher than the bandwidth where the majority of the signal power is located (5 kHz), the acquired DT digital signal is a very good representation of the original CT digital signal. Conducting FFT analysis on both signals, we find the *SNDR*s at the interpolator input and output from 0 to 250 MHz are 48 and 57 dB, respectively.

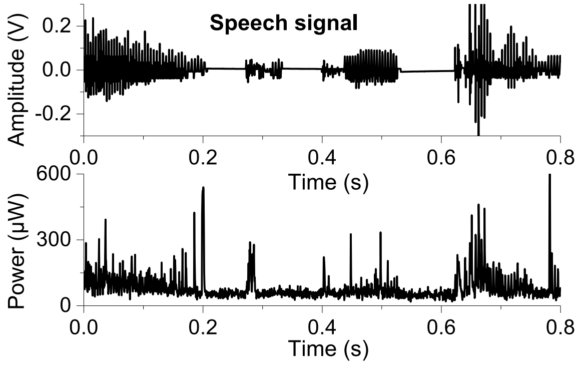


Fig. 21. Waveform of input speech signal (top) and the corresponding instantaneous power consumption (bottom) of the chip.

Fig. 20 shows the power consumption of the system versus the input event rate. When the event detector is ON, redundant events in the feedback loops are eliminated. The power sharply decreases when the input activity decreases. When the event detector is OFF, redundant events keep going around the feedback loops and maintain a high event rate regardless of the input. On the other hand, when the input rate is low, the data values in the system are less likely to change. Hence the power is still weakly dependent on the input. The power consumption of the system when the input event rate is 20 Msps is broken down in Table II. Fig. 21 shows a speech signal, fed into a 7-bit level-crossing sampler [6] followed by our test chip and the corresponding chip power consumption, demonstrating agility. When the input is quiet, the power drops to the leakage level (40 μW).

A performance summary for the entire system is shown in Table II. Table III compares the presented work with the state-of-the-art CT digital and DT analog and digital filters. A figure of merit (FoM) for comparing different signal processors is given by

$$\text{FoM} = \frac{\text{Power}}{f_{\max} 2^{\text{ENOB}} N K}$$

where $f_{\max}$ is the maximum input rate, N is the filter order, and K is the ratio by which an IIR's filter order is smaller than that of an FIR filter that would be needed in order to achieve the same frequency response specs. K is used in order to reveal the definite selectivity advantage of IIR filters, compared to FIR; if the filter under consideration is FIR, $K = 1$. The chip achieves an FoM better than all the CT digital design. In addition, for the first time, the chip implements an IIR architecture, signal-derived timing, and synchronous output. The system's 16-bit arithmetic resolution is much higher than that in CT digital filters in previous work [6]–[8]. Both the IIR architecture and the larger number of bits contribute to a much more accurate and versatile frequency response performance. Thanks to the small technology node used and the fact that only two delay blocks are used, the area of delay line in this design is 9× smaller[4] than in [6] and [8] (both used

[4] There are in fact two different comparisons involved. We compare the area of the delay line area with the 1-bit delay line in [6], and also compare the total area of the delay line and the FIFOs with that in [8]. Both comparison shows that our design is 9× smaller.

TABLE III
COMPARISON TO STATE-OF-THE-ART CT DIGITAL AND DT FILTERS

Parameter	[21]	[22]	[20]	[6]	[7]	[8]	**This work**
Signal domain	DT analog		DT digital	CT digital			**CT digital**
Output/ clock freq.	Analog	Analog	Sync/ 2.1 GHz	Async.	Async.	Async.	**Sync/ 1 MHz***
Type/order	FIR/15th	IIR/7th	IIR/3rd	FIR/15th	FIR/5th	FIR/15th	**IIR/6th**
Process	45 nm	65 nm	32 nm	90 nm	65 nm	130 nm	**65 nm**
Active Area	0.15 mm^2	0.42 mm^2		0.64mm^2	0.08 mm^2	5.6 mm^2	**0.64 mm^2**
Supply	1.1 V	1.2 V	1 V	1 V	1.2 V	1 V	**1.2 V**
DSP arithmetic resolution	6 bits	8 bits	8 bits	8 bits	4 bits	8 bits	**16 bits**
Stop-band rejection	22 dB	>100 dB	NA	35 dB	15 dB	25 dB	**80 dB**
SNDR	33 dB	41 dB	50 dB	47-62 dB	20-22 dB	40-51 dB	**>54 dB**
In-band IM3	-41 dB‡			-51 dB	-27 dB	-50 dB	**-59 dB**
Out-band IM3	-43 dB‡				-22 dB		**-57.6 dB**
Max. input rate	3.2 Gsps	800 Msps	2.1 Gsps	17 MHz	45 GHz	10 Msps	**20 Msps**
Power (Min/Max)†	48 mW**	1.96 mW**	- / 24 mW**	310 μW/ 3 mW	270 μW/ 9.2 mW	70 μW/ 3.3 mW	**40 μW/ 2.32 mW**
FoM (Min/Max)†	27 fJ	0.36 fJ	- / 1.3 fJ	1.2 pJ/ 12 pJ	0.1 fJ/ 4.3 fJ	1.6 fJ/ 76 fJ	**0.06 fJ/ 2.9 fJ**
Input antialiasing filter is required?	Yes	Yes	Yes	No	No	No	**No**

*For CT-to-DT converter only. **Not including clock generation and anti-aliasing filter.
†Varying with signal activity; without power down circuitry.
‡Inferred from IIP3 measurements, assuming full-scale combined input

16 delay blocks). Its FoM also compares well with that of DT analog and digital filters. However, as a digital filter, our chip has a programmability that cannot be achieved by analog filters [21], [22]. Our chip does not use a clock (except for output data synchronization) and an input antialiasing filter, both of which are necessary in DT filters [20]–[23] and can be power hungry. In contrast to analog designs [21], [22], the chip features agile power adaptation to input activity, a property it shares with other CT digital filters [6]–[8].

VIII. CONCLUSION

This paper has presented the detailed design of the first CT digital IIR filter with accurate frequency response and a synchronous output. The system includes an event grouping method which eliminates parasitic oscillation. A new design method was provided which allows one to use two tap delays to implement high-order CT digital IIR filters. The number of tap delays is not a function of the filter order. Based on these techniques, a sixth-order CT digital IIR filter was implemented in a 65-nm CMOS technology. Its power dissipation responds to input in a fully agile manner, varying by more than 50×, resulting in an FoM that varies from 2.9 to 0.06 fJ. The delay line area is 9× smaller than prior art. Frequency response accuracy is superior to that of previous CT digital filter; stop-band rejection is 45-dB better. A CT-to-DT converter allows integration with DT systems. The results suggest that the CT digital filtering approach has reached levels of performance previously unattainable by this technique, in fact approaching that of well-established analog and digital approaches, while avoiding a clock and an antialiasing filter, and featuring agile power dissipation without any power-down circuitry.

ACKNOWLEDGMENT

The authors would like to thank Prof. P. Kinget, K. Tien, Y. Xu, J. Zhu, S. Patil, and N. Guo for their helpful discussions.

REFERENCES

[1] W. R. Dieter, S. Datta, and W. K. Kai, "Power reduction by varying sampling rate," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, San Diego, CA, USA, Aug. 2005, pp. 227–232.

[2] A. Mostafa and M. El-Hagry, "Adaptive sampling of speech signals," *IEEE Trans. Commun.*, vol. COM-22, no. 9, pp. 1189–1194, Sep. 1974.

[3] T. Kurp, R. X. Gao, and S. Sah, "An adaptive sampling scheme for improved energy utilization in wireless sensor networks," in *Proc. IEEE Instrum. Meas. Technol. Conf. (I2MTC)*, Austin, TX, USA, May 2010, pp. 93–98.

[4] Y. Tsividis, "Continuous-time digital signal processing," *Electron. Lett.*, vol. 39, no. 21, pp. 1551–1552, Oct. 2003.

[5] Y. Tsividis, "Event-driven data acquisition and digital signal processing—A tutorial," in *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 8, pp. 577–581, Aug. 2010.

[6] B. Schell and Y. Tsividis, "A continuous-time ADC/DSP/DAC system with no clock and with activity-dependent power dissipation," *IEEE J. Solid-State Circuits*, vol. 43, no. 11, pp. 2472–2481, Nov. 2008.

[7] M. Kurchuk, C. Weltin-Wu, D. Morche, and Y. Tsividis, "Event-driven GHz-range continuous-time digital signal processor with activity-dependent power dissipation," *IEEE J. Solid-State Circuits*, vol. 47, no. 9, pp. 2164–2173, Sep. 2012.

[8] C. Vezyrtzis, W. Jiang, S. M. Nowick, and Y. Tsividis, "A flexible, event-driven digital filter with frequency response independent of input sample rate," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2292–2304, Oct. 2014.

[9] B. Schell, "Continuous-time digital signal processors: Analysis and implementation," Ph.D. dissertation, Dept. Elect. Eng., Columbia Univ., New York, NY, USA, 2008.

[10] Y. Chen, X. Zhang, Y. Lian, R. Manohar, and Y. Tsividis, "A continuous-time digital IIR filter with signal-derived timing, agile power dissipation and synchronous output," in Symp. VLSI Circuits (VLSI Circuits) Dig., Kyoto, Japan, Jun. 2017, pp. 272–273.

[11] J. Sparsø and S. Furber, *Principles Asynchronous Circuit Design*. Norwell, MA, USA: Kluwer, 2002.

[12] Y. Neuvo, D. Cheng-Yu, and S. Mitra, "Interpolated finite impulse response filters," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 32, no. 3, pp. 563–570, Jun. 1984.

[13] S. K. Mitra, *Digital Signal Processing: A Computer—Based Approach*. Boston, MA, USA: McGraw-Hill, 2001, pp. 686–688.

[14] R. Ginosar, "Metastability and synchronizers: A tutorial," *IEEE Des. Test Comput.*, vol. 28, no. 5, pp. 23–35, Sep./Oct. 2011.

[15] Y. Chen, R. Manohar, and Y. Tsividis, "Design of tunable digital delay cells," in *Proc. Custom Integr. Circuits Conf.*, Austin, TX, USA, Apr./May 2017, pp. 3–21.

[16] K. van Berkel, *Handshake Circuits: An Asynchronous Architecture for VLSI Programming*. Cambridge, U.K.: Cambridge Univ. Press, 1993.

[17] D. E. Muller, "Asynchronous logics and application to information processing," in *Switching Theory in Space Technology*. Redwood City, CA, USA: Stanford Univ. Press, 1963, pp. 289–297.

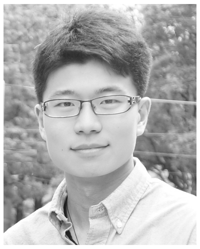
[18] C. L. Seitz, "System timing," in *Introduction to VLSI Systems*. Boston, MA, USA: Addison-Wesley, 1980, pp. 218–262.

[19] D. Kim, G. Chen, M. Fojtik, M. Seok, D. Blaauw, and D. Sylvester, "A 1.85 fW/bit ultra low leakage 10T SRAM with speed compensation scheme," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Rio de Janeiro, Brazil, May 2011, pp. 69–72.

[20] A. Agarwal *et al.*, "A 320mV-to-1.2V on-die fine-grained reconfigurable fabric for DSP/media accelerators in 32 nm CMOS," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2010, pp. 328–329.

[21] E. O'hAnnaidh, E. Rouat, S. Verhaeren, S. Le Tual, and C. Garnier, "A 3.2 GHz-sample-rate 800 mHz bandwidth highly reconfigurable analog FIR filter in 45 nm CMOS," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2010, pp. 90–91.

[22] M. Tohidian, I. Madadi, and R. B. Staszewski, "A 2 mW 800 MS/s 7th-order discrete-time IIR filter with 400 kHz-to-30 MHz BW and 100 dB stop-band rejection in 65 nm CMOS," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2013, pp. 174–175.



Yu Chen (S'13) received the B.S. degree in microelectronics from Fudan University, Shanghai, China, in 2010, and the M.S. and Ph.D. degrees in electrical engineering from Columbia University, New York, NY, USA, in 2012 and 2016, respectively.

He held several internships with the Bluetooth and Optical SerDes Groups, Broadcom, Irvine, CA, USA, and with the Microcontroller Unit Group, Silicon Labs, Austin, TX, USA. He is currently with Apple, Cupertino, CA, USA. His current research interests include continuous-time digital signal processing, mixed-signal integrated circuit design, and asynchronous digital circuit design.

Xiaoyang Zhang (S'09–M'15) received the B.Sc. degree in microelectronics from Peking University, Beijing, China, in 2009, and the Ph.D. degree in electrical engineering with the National University of Singapore, Singapore, in 2015.

His current research interests include low-power biomedical sensors and wearable computing.

Yong Lian (M'90–SM'99–F'09) received the B.Sc. degree from the College of Economics and Management, Shanghai Jiao Tong University, Shanghai, China, in 1984, and the Ph.D. degree from the Department of Electrical Engineering, National University of Singapore (NUS), Singapore, in 1994.

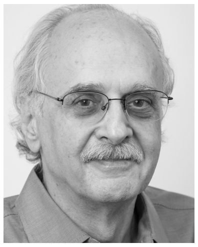
He was with the industry for nine years and joined NUS in 1996, where he was the Deputy Department Chair of research, the Area Director of IC and embedded systems with the Department of Electrical and Computer Engineering, and a member of the University Tenure and Promotion Committee and Senate Delegacy. He was the first one in the Department of Electrical and Computer Engineering, NUS, to be appointed as the Provost's Chair Professor in 2011. He is currently a Professor with York University, Toronto, ON, Canada. His current research interests include biomedical circuits and systems and signal processing.

Dr. Lian is a fellow of the Academy of Engineering Singapore. He received many awards including the IEEE Circuits and Systems Society's Guillemin-Cauer Award in 1996, the IEEE Communications Society Multimedia Communications Best Paper Award in 2008, the Institution of Engineers Singapore Prestigious Engineering Achievement Award in 2011, the Hua Yuan Association/Tan Kah Kee International Society Outstanding Contribution Award in 2013, the Chen-Ning Franklin Yang Award in Science and Technology for New Immigrant in 2014, and the Design Contest Award in 20th International Symposium on Low-Power Electronics and Design (ISLPED2015). As an Educator, he received the University Annual Teaching Excellent Award in two consecutive academic years from 2008 to 2010 and many other teaching awards from the Faculty of Engineering of NUS. Under his guidance, his students received many awards including the Best Student Paper Award in ICME 2007, the winner of 47th DAC/ISSCC Student Design Contest in 2010, and the Best Design Award in A-SSCC 2013 Student Design Contest. He was the Editor-in-Chief of the IEEE Transactions on Circuits and Systems Part II: Express Briefs from 2010 to 2013. He was the Guest Editor of eight special issues in the IEEE Transactions on Circuits and Systems Part I: Regular Papers, the IEEE Transactions on Biomedical Circuits and Systems, and the *Journal of Circuits, Systems Signal Processing*. He was the Vice President of Publications of the IEEE CAS Society from 2013 to 2015, the Vice President of the Asia–Pacific Region of the IEEE CAS Society from 2007 to 2008, an AdComm Member of the IEEE Biometrics Council from 2008 to 2009, a CAS Society Representative to the BioTechnology Council from 2007 to 2009, a Chair of the BioCAS Technical Committee of the IEEE CAS Society from 2007 to 2009, a Chair of the DSP Technical Committee of the IEEE CAS Society from 2010 to 2011, a member of the IEEE Medal for Innovations in Healthcare Technology Committee from 2010 to 2012, and a Distinguished Lecturer of the IEEE CAS Society from 2004 to 2005. He is the Founder of the International Conference on Green Circuits and Systems, the Asia–Pacific Conference on Postgraduate Research in Microelectronics and Electronics, and the IEEE Biomedical Circuits and Systems Conference. He is the President-Elect of the IEEE CAS Society, a member of the IEEE Fellow Committee, a Steering Committee Member of the IEEE Transactions on Biomedical Circuits and Systems.

Rajit Manohar (M'98–SM'10) received the B.S., M.S., and Ph.D. degrees from Caltech, Pasadena, CA, USA, in 1994, 1995, and 1998, respectively.

He was a Professor of electrical and computer engineering and a Stephen H. Weiss Presidential Fellow with Cornell, Ithaca, NY, USA, where he was the Associate Dean of research and graduate studies with Cornell Engineering. He was also an Associate Dean of academic affairs and an Associate Dean of research with Cornell Tech, New York, NY, USA. He has been with Yale Faculty, New Haven, CT, USA, since 2017, where his group conducts research on the design, analysis, and implementation of self-timed systems. He is currently the John C. Malone Professor of electrical engineering and a Professor of computer science with Yale. He is involved in the design and implementation of a number of self-timed VLSI chips including high-performance and low-power microprocessors, FPGAs, radiation-hardened FPGAs, and the first deterministic large-scale neuromorphic architecture. He founded Achronix Semiconductor, Santa Clara, CA, USA to commercialize high-performance asynchronous FPGAs.

Dr. Manohar was a recipient of the NSF Career Award, nine Best Paper Awards, nine teaching awards, and was named to MIT technology review's top 35 young innovators under 35 for contributions to low-power microprocessor design.



Yannis Tsividis (S'71–M'74–SM'81–F'86–LF'12) received the B.S. degree from the University of Minnesota, Minneapolis, MN, USA, in 1972, and the M.S. and Ph.D. degrees from the University of California, Berkeley, CA, USA, in 1973 and 1976, respectively.

He is Edwin Howard Armstrong Professor of Electrical Engineering, Columbia University, New York, NY, USA. He was with Motorola Semiconductor, Phoenix, AZ, USA and AT&T Bell Laboratories, Murray Hill, NJ, USA. He has taught at the University of California at Berkeley, the Massachusetts Institute of Technology, Cambridge, MA, USA, and the National Technical University of Athens, Greece.

Dr. Tsividis received the 1984 IEEE W.R.G. Baker Award for the best IEEE publication, Columbia's Presidential Award for Outstanding Teaching in 2003, the IEEE Undergraduate Teaching Award in 2005, the IEEE Circuits and Systems Education Award in 2010, the Outstanding Achievement Award of the University of Minnesota in 2013, and the IEEE Gustav Robert Kirchhoff Award in 2007. He was a recipient or co-recipient of the Best Paper Awards from the European Solid-State Circuits Conference in 1986, the IEEE International Solid-State Circuits Conference in 2003, and the IEEE Circuits and Systems Society (Darlington Award in 1987 and Guillemin-Cauer Award in 1998 and 2008). In 2012, he was elected Professor Honoris Causa at the University of Patras, Greece.