

IEICE **TRANSACTIONS**

on Electronics

DOI:10.1587/transle.2023ECP5054

Publicized:2024/06/11

**This advance publication article will be replaced by
the finalized version after proofreading.**

A PUBLICATION OF THE ELECTRONICS SOCIETY



The Institute of Electronics, Information and Communication Engineers

Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3chome, Minato-ku, TOKYO, 105-0011 JAPAN

PAPER

VLSI Design and Implementation of ARS for Periods EstimationTakahiro SASAKI[†], *Member and* Yukihiro KAMIYA[†], *Nonmember*

SUMMARY This paper proposes two VLSI implementation approaches for periods estimation hardware of periodic signals. Digital signal processing is one of the important technologies, and to estimate periods of signals are widely used in many areas such as IoT, predictive maintenance, anomaly detection, health monitoring, and so on. This paper focuses on accumulation for real-time serial-to-parallel converter (ARS) which is a simple parameter estimation method for periodic signals. ARS is simple algorithm to estimate periods of periodic signals without complex instructions such as multiplier and division. However, this algorithm is implemented only on software, suitable hardware implementation methods are not clear. Therefore, this paper proposes two VLSI implementation methods called ARS-DFF and ARS-MEM. ARS-DFF is simple and fast implementation method, but hardware scale is large. ARS-MEM reduces hardware scale by introducing an SRAM macro cell. This paper also designs both approaches using SystemVerilog and evaluates VLSI implementation. According to our evaluation results, both proposed methods can reduce the power consumption to less than 1/1000 compared to the implementation on a microprocessor.

key words: VLSI, digital circuit, doppler sensor, parameter estimation, vital sensing, IoT.

1. Introduction

Digital signal processing is one of the important technologies for many applications such as IoT[1]–[6], predictive maintenance[7]–[9], anomaly detection[10]–[12], health monitoring[13]–[16], and so on. In the medical field, non-contact vital sensing which is the measurement of vital data such as heartbeats and breathing without putting sensors on the body is expected that the non-contact vital sensing enables us to make our daily life more secure and efficient[17].

Figure 1 depicts an example of vital sensing. A Doppler sensor detects the velocity of a moving objects, and it can be used as a non-contact vital sensor. The sensor emits microwaves and receives waves bounced back and outputs the phase difference between the transmitted wave and the reflected wave, called Doppler frequency. Since this frequency deviation is caused by the movement of the body surface, we can estimate the periods of heartbeats and breathing through digital signal processing, using the samples of the receiver output obtained by the analog-to-digital converter (ADC).

Figure 2 shows one of the applications of the proposed method. If we can measure two or more persons using one Doppler sensor, it is possible to reduce the number of the sensors to be installed in hospitals, which can reduce the cost of implementation[18]. To analyze frequency of signals, fast

Fourier transform (FFT)[19]–[21] is well used technique. FFT can analyze the frequency of multiple composite waveforms. Therefore, we can estimate the periods of two or more persons' heartbeats and breathing separately. However, frequency resolution of FFT is low in the low-frequency bands. To increase resolution, we need to use many signal samples, and to acquire them takes long time[19]. We assume to estimate the periods of two persons' heartbeats and breathing as shown in fig. 2, and sampling frequency f_s of Doppler sensor is 100 Hz. In order to distinguish the periods of two persons' breathing, 4.00 and 4.01, respectively, the frequency resolution δf should be larger than $\frac{f_s}{N}$, where N is the number of sampling data. To meet the condition, N must be equal or larger than 262144. Therefore, the sampling time is calculated as $f_s \times N \approx 43$ minutes which is not suitable for vital sensing.

As another period estimation methods, accumulation for real-time serial-to-parallel converter (ARS)[22], [23] was proposed. Similar approaches which estimate the period of a signal are proposed[24]–[26]. However, unlike our proposed method, those research never deals with multiple periodic signals. Although the methods presented in [27] and [28] are capable to cope with the multiple signals, they cannot estimate the fundamental waveform at the same time. Similarly, the several methods compared in [29] also do not estimate the fundamental waveform. A more advanced version in [30] was proposed for the same situation that this paper focuses on. It means that there are multiple bodies in front of a Doppler sensor and trying to conduct the parameter estimation for the two bodies. This method also allows us to estimate the fundamental waveforms. However, this method requires multiple antennas at the receiver since it separates the multiple signals by the difference of the angle of arrival (AOA). Therefore, naturally, this method cannot cope with multiple signals if they impinge on the antennas from the identical AOA. Of course, it costs a lot of computational complexity and hardware resources due to the usage of multiple antennas.

Furthermore, ARS has the advantage of high resolution in the low-frequency bands. The details of theoretical and numerical analysis of performance between ARS and FFT are shown in the [31], but the number of sampling data of ARS is smaller than that of FFT in low-frequency bands such as breathing. We assume that the sampling frequency of Doppler sensor is 100 Hz. If we want to distinguish the periods of two persons' breathing, 4.00 and 4.01, respectively,

[†]Graduate School of Information Science and Technology, Aichi Prefectural University, Aichi, Japan.

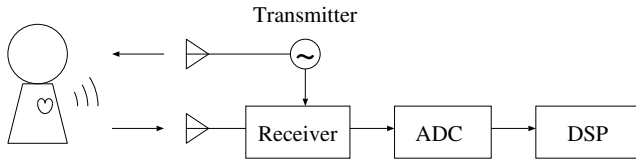


Fig. 1: Vital sensing using a Doppler sensor.

the number of minimum sampling data is 802 samples under ideal conditions[†]. According to the results of our preliminary experiments, we can distinguish the periods of two persons' breathing if the sampling time is 3 minutes, which is smaller than that of FFT (43 minutes). Therefore, this paper focuses on the ARS and does not further discuss FFT. Naturally, it is emphasized that ARS is not always superior to FFT.

However, ARS is implemented only on software, and hardware implementation to achieve better performance with lower power consumption is not proposed. Therefore, this paper proposes two VLSI implementation methods called ARS-DFF and ARS-MEM. ARS-DFF is simple and fast implementation method using D-flip flops, but hardware scale is large because ARS requires many accumulators. ARS-MEM reduces hardware scale by introducing an SRAM macro cell, but slower than ARS-DFF. This paper also designs both approaches using SystemVerilog and evaluates VLSI implementation. According to our design results, the maximum sampling frequency of input signal of ARS-DFF and ARS-MEM is 57.5MHz and 890KHz, respectively. However, hardware scale of ARS-MEM is less than 5% of ARS-DFF. Furthermore, both proposed methods can reduce the power consumption to less than 1/1000 compared to the implementation on a microprocessor.

The contributions of this paper are following:

- The first VLSI implementations of ARS are proposed.
- A formula to estimate memory capacity of working set of ARS is clarified.
- Hardware scale and performance of the two methods were evaluated in comparison.

The following sections are organized as follows: Algorithm of ARS is introduced in Section 2. Section 3 proposes our hardware implementation methods, and Section 5 evaluates VLSI design. Finally, we conclude this paper in Section 6.

2. Accumulation for real-time serial-to-parallel converter

Basic idea of ARS[22], [23] is simple. We assume that the input signal is a periodic signal which repeats [1, 3, 2, 4, 3], and the period is 5 samples. To explain simply, this paper assumes a simple waveform, but ARS can apply to composite waveform[22]. $x[k]$ denotes k -th input data from

[†]Each fundamental waveform is a sharp or abrupt waveform without noise.

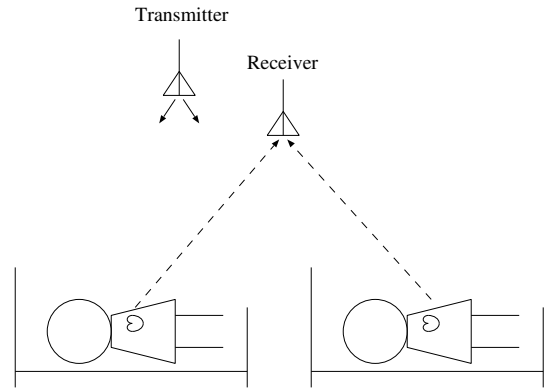


Fig. 2: Vital sensing for 2 persons using one Doppler sensor.

ADC. ARS fetches N samples from ADC as one block and accumulates each element of all blocks. Figure 3(a) depicts the case if the block size is equal to 5 which is the same as the period of the input signal. The elements of block[0], $[x[0], x[1], x[2], x[3], x[4]]$, are [1, 3, 2, 4, 3], the elements of block[1], $[x[5], x[6], x[7], x[8], x[9]]$, are [1, 3, 2, 4, 3], and the elements of block[2], $[x[10], x[11], x[12], x[13], x[14]]$, are [1, 3, 2, 4, 3]. All blocks have same element [1, 3, 2, 4, 3], and accumulated values of each element (accumulators in fig. 3(a)) becomes [3, 9, 6, 12, 9]. Here, the maximum element of accumulators is 12.

Figure 3(b) depicts the case if the block size is equal to 4 which is not the same as the period of the input signal. The elements of block[0], $[x[0], x[1], x[2], x[3]]$, are [1, 3, 2, 4], the elements of block[1], $[x[4], x[5], x[6], x[7]]$, are [3, 1, 3, 2], and the elements of block[2], $[x[8], x[9], x[10], x[11]]$, are [4, 3, 1, 3]. Therefore, accumulated values of each element (accumulators in fig. 3(b)) becomes [8, 7, 6, 9]. Here, the maximum element of accumulators is 9. The maximum value of accumulators of which block size is equals to the period of input signal has the maximum value in all accumulators.

If the minimum and the maximum period to estimate are 4, 7, respectively, ARS divides input data into four blocks of which block size are 4, 5, 6, and 7. The maximum value of each accumulator for block size 4, 5, 6, and 7 are 9, 12, 9, 10, respectively. The maximum value is 12 which is maximum value of block size 5, and we can find the period of this waveform is 5.

Before searching the maximum value from each accumulator, all values in accumulators should be normalized to improve estimation accuracy because the number of accumulated signals of each accumulator are not same. However, divider is required for normalization, and it increases circuit scale of VLSI. Therefore, we adopt a method to equalize the number of data to be accumulated. This method can be implemented using simple counters and comparators.

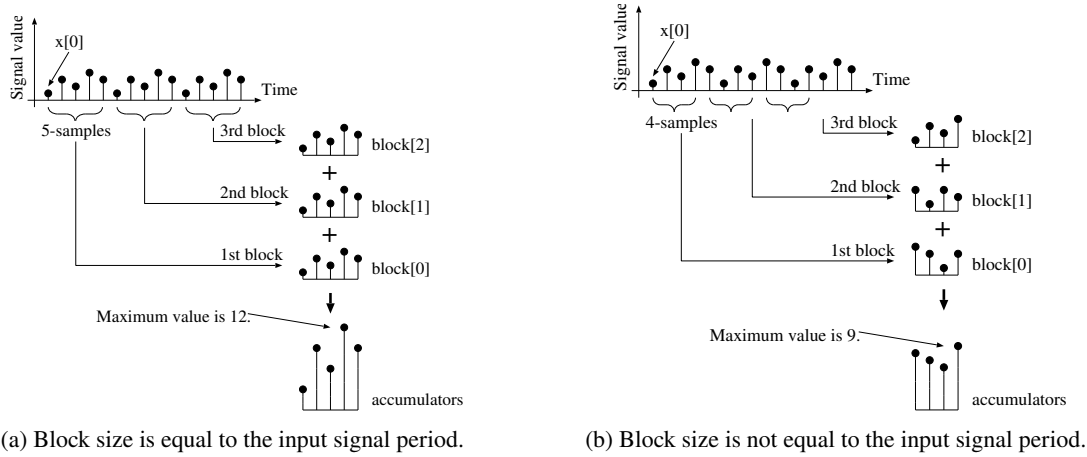


Fig. 3: Examples of signal processing.

3. Proposed Hardware implementation methods

In this section, we propose hardware implementation methods. Unlike software implementation, suitable hardware implementation method depends on application and condition such as maximum and minimum waveform periods to estimate, sampling frequency. ARS is simple algorithm and do not require complex logical unit such as multiplier, divider, and floating-point arithmetic unit. The dominant factor which affects hardware scale of ARS is memory capacity which constructs accumulator.

In addition, the analysis process can be performed on-line or offline. In this paper, assuming vital sensing to be performed in hospitals, etc., we try to implement the VLSI in an online type where analysis results can be obtained at short interval of time.

If we assume to implement accumulators as 2-dimensional array, total memory capacity for accumulators Mem_{acc} is in proportion to:

$$(P_{max} - P_{min}) \times P_{max},$$

where P_{max} and P_{min} denote maximum period of waveform (samples) and minimum period of waveform (samples), respectively. The Mem_{acc} is the number of words of memory unit, so if it is enough small, we can implement accumulators by D-Flip Flops (DFF). If not, accumulators should be implemented by memory unit such as an SRAM macro cell or an external DRAM module. The memory capacity is in proportion to both the square of the range of period to estimate and the square of sampling frequency. Therefore, these parameters strongly affect hardware cost and suitable implementation method.

We discuss exact memory capacity for accumulators. The accumulators for period P contain P accumulators, and the maximum value of each accumulator is:

$$\frac{Sig_{max} \times Time \times freq}{P},$$

where Sig_{max} , $Time$ and $freq$ are the maximum signal value

of ADC, measuring time (sampling time) and sampling frequency, respectively. Therefore, the number of bits (or the number of DFFs) of each accumulator is:

$$\left\lceil \log_2 \left(\frac{Sig_{max} \times Time \times freq}{P} + 1 \right) \right\rceil.$$

Memory capacity of accumulators for period P is:

$$P \times freq \times \left\lceil \log_2 \left(\frac{Sig_{max} \times Time \times freq}{P} + 1 \right) \right\rceil.$$

Therefore, total memory capacity of accumulators is:

$$\sum_{i=P_{min}}^{P_{max}} \left\{ i \times \log_2 \left(\frac{Sig_{max} \times Time \times freq}{i} + 1 \right) \right\}.$$

If we assume sampling frequency is 10Hz and try to estimate period between 2sec and 6sec, P_{min} and P_{max} equals 20 and 60, respectively. If measuring time is 180sec (3minutes), and Sig_{max} is 4095 (12bit ADC), total memory capacity is 29502 bits. If we implement accumulators by DFFs, the total number of DFFs is 29502. If sampling frequency is increased to 100Hz, P_{min} and P_{max} equals 200 and 600, respectively, and total memory capacity becomes 2884824 bits. If we implement accumulators by DFFs, the total number of DFFs is 2884824, and it is not feasible for VLSI implementation. In this case, accumulators should be implemented by a memory unit.

In this way, suitable implementation method depends on design parameters. In this paper, we propose two methods; one is D-flip flop approach called ARS-DFF which implements accumulators by adders and DFFs, and the other is memory approach called ARS-MEM which implements accumulators by an adder and a memory unit.

3.1 D-flip flop approach

This approach implements accumulator by adders and DFFs. We call this approach as ARS-DFF. Basic idea of ARS-DFF

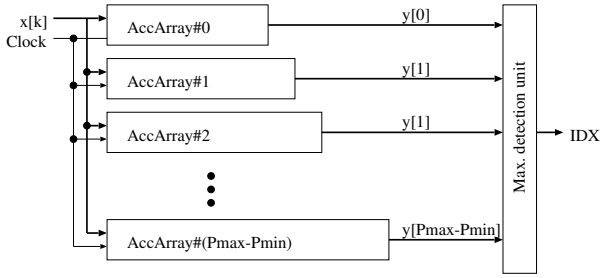


Fig. 4: Implementation with DFFs.

is almost same to [22] but optimized for hardware implementation. Figure 4 depicts block diagram of ARS-DFF. To explain the architecture simply, we assume that minimum periods, maximum periods, and sampling frequency are 4sec, 7sec, and 1Hz, respectively. The AccArray is an array of accumulators and accumulates stride input data. The AccArray also outputs the maximum value of accumulators.

In fig. 4, $x[k]$ is k -th input data from an ADC. AccArray#0 in fig. 4 divides input data into 4 samples, calculates the sum of each element, finds the maximum value from each accumulated value, and output that value as $y[0]$. AccArray#1 takes charge of 5 samples/block, and AccArray#2 takes charge of 6 samples/block.

Figure 5 depicts block diagram of AccArray unit shown in the fig. 4. P in Fig. 5 for AccArray# n is equals to $P_{min} + n$. AccArray#0 takes charge of 4 samples/block, so P is equals to 4. Acc in Fig. 5 is an accumulator, and DFFs labeled Dx, where x is 0 to $P - 1$, composes a shift register to activate only one accumulator. On reset, the DFF labeled D0 is set to 1, and other DFFs including accumulators are cleared as 0. Enable signal of only Acc0 is asserted, so Acc0 accumulates $x[0]$. On the next cycle, D1 becomes 1 and D0 becomes 0, therefore, Acc1 accumulates $x[1]$. In this way, Acc($k \bmod P$) accumulates $x[k]$, where mod denote modulo operation. The y is maximum value of all accumulators.

Max. detector unit in fig. 4 finds maximum value from $y[0]$ to $y[P_{max} - P_{min}]$, and outputs the index with the maximum value as IDX. Therefore, the estimated period is calculated as $IDX + P_{min}$. ARS can estimate the number of fundamental waveforms of combined signals and fundamental period of each waveform. However, the maximum number of the fundamental waveforms of the combined signals to estimate and the threshold to detect it depend on applications. Because this paper focuses on hardware design methods and its circuit scales, our designed VLSI outputs only one period, but to estimate plural periods is not difficult and circuit scale is not large.

Counter in fig. 5 is a saturating counter which counts the number of samples, in order to equalize the number of data to be accumulated described in Section 2. If the counter value is lower than or equal to $MV(P)$, the counter outputs 1 to accumulate the input data, otherwise the counter outputs 0 to stop accumulation. $MV(P)$ is fixed value calculated as follows:

$$\frac{\#samples}{P_{max}} \times P,$$

where #samples is the number of total samples to process.

ARS estimates the number of signals and each period using pre-defined threshold[22]. To handle plural periods, Max. detection unit checks if the maximum value of each AccArray is larger than the threshold. If the conditions are met, Max. detection unit outputs it's index, and the number of AccArrays which meet the condition is equals to the number of signals. The number of signals is the number of indices to output. Only a comparator and a counter are required to implement this method, however, to determine the threshold is difficult and depends on application. Therefore, our designed VLSI outputs only one period.

3.2 Memory approach

This approach implements accumulators by an adder and memory unit. We call this approach as ARS-MEM. As shown in fig. 6, ARS-MEM is constructed by an Address Generators, a Memory Controller, an adder, and a Memory unit.

This approach is similar to the implementation by software. Figure 7 depicts pseudo-code of ARS-MEM algorithm. The $acc[][]$ and $ptr[n]$ in fig. 7 correspond to Memory and Address Generator# n in fig. 6, respectively. The $ctr[]$ counts the number of data to accumulate, in order to equalize the number of data to be accumulated described in Section 2. Although a divider is used in the calculation of sat in fig. 7, no division unit is actually needed because the value is calculated statically at logical synthesis. In addition, the $acc[n][i]$ in fig. 7 corresponds to AccArray# n in fig. 4.

An Address Generator# n in the fig. 6 counts up to $n + P_{min} - 1$. If the counter reaches to the maximum value, the counter value is cleared as 0. If P_{min} and P_{max} are 4 and 7, respectively, the counter of the Address Generator#0 counts up to 3, the counter of the Address Generator#1 row counts up to 4, and the counter of the Address Generator#3 counts up to 6.

Figure 8 depicts 2-dimensional memory mapping. One row corresponds to one AccArray of ARS-DFF. We assume $x[k]$ is k -th signal data from the ADC. As initial condition, all $c_{i,j}$ is cleared. From now, we describe behavior of this approach.

To explain memory mapping, we assume simple parameters; P_{min} and P_{max} are 4 and 7, respectively. We assume that all accumulators are cleared as 0 initially. As first, k is 0 and the first signal $x[0]$ is accumulated in $c_{0,0}, c_{1,0}, c_{2,0}, c_{3,0}$. The next signal $x[1]$ is accumulated in $c_{0,1}, c_{1,1}, c_{2,1}, c_{3,1}$. In the same manner, $x[4]$ is accumulated in $c_{0,0}, c_{1,4}, c_{2,4}, c_{3,4}$. The counter in Address Generator#0 reached to the maximum value and reset to 0. So, $x[5]$ is accumulated in $c_{0,0}, c_{1,5}, c_{2,5}, c_{3,5}$. Unlike ARS-DFF, ARS-MEM takes multi-cycles to process one data because the number of memory units is one. In the above case, to process $x[0]$ takes 8 cycles;

Cycle 0: read from $c_{0,0}$ and add $x[0]$,

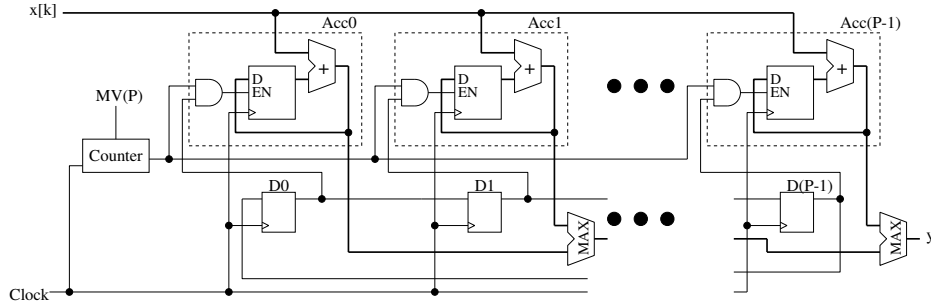


Fig. 5: Blockdiagram of AccArray.

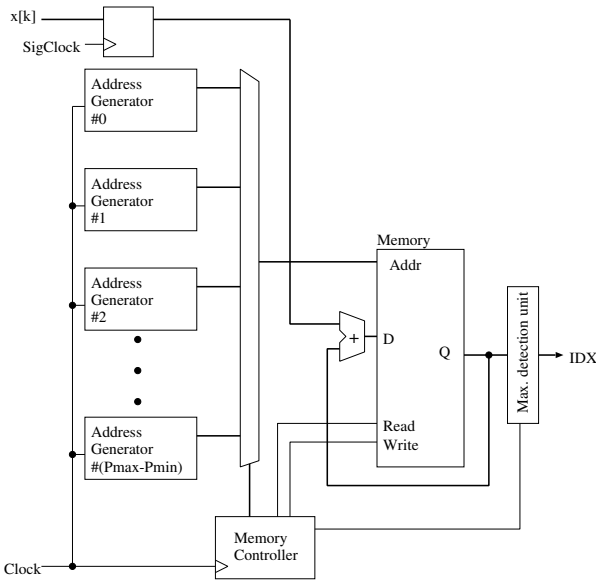


Fig. 6: Implementation with a memory.

```

var acc[Pmax - Pmin][Pmax]
var ptr[Pmax - Pmin]
var ctr[Pmax - Pmin]
var sat
var i, j

i = 0
sat = #samples / Pmax
while(true) {
  for(j = 0; j < Pmax - Pmin; j = j + 1) {
    var k = ptr[j]

    if(ctr[j] < sat) {
      acc[j][k] = acc[j][k] + x[i]
      k = k + 1
      if(k >= j + Pmin) {
        ctr[j] = ctr[j] + 1
        k = 0 # Reset address
      }
      ptr[j] = k
    }
  }
  i = i + 1
}
    
```

Fig. 7: Pseudo-code of ARS-MEM algorithm.

- Cycle 1:** write the sum to $c_{0,0}$,
- Cycle 2:** read from $c_{1,0}$ and add $x[0]$,
- Cycle 3:** write the sum to $c_{1,0}$,
- Cycle 4:** read from $c_{2,0}$ and add $x[0]$,
- Cycle 5:** write the sum to $c_{2,0}$,
- Cycle 6:** read from $c_{3,0}$ and add $x[0]$,
- Cycle 7:** write the sum to $c_{3,0}$.

In general, it takes $(P_{\max} - P_{\min} + 1) \times 2$ cycles to process one data from an ADC. Therefore, the frequency of the sampling clock (SigClock shown in fig. 4) is lower than that of the clock frequency of ARS-MEM main unit (Clock shown in fig. 6).

The function of Max. detector unit in fig. 6 is similar to the unit used in ARS-DFF. However, it finds the maximum value sequentially. To accumulate $x[k]$, the Memory outputs $x_{i,j}$. Max. detector unit fetches the value at that timing and find the maximum value.

4. Design verification

We implement both ARS-DFF and ARS-MEM as synthesizable SystemVerilog codes. To verify our design works

Period	Samples						
	0	1	2	3	4	5	6
4	$c_{0,0}$	$c_{0,1}$	$c_{0,2}$	$c_{0,3}$			
5	$c_{1,0}$	$c_{1,1}$	$c_{1,2}$	$c_{1,3}$	$c_{1,4}$		
6	$c_{2,0}$	$c_{2,1}$	$c_{2,2}$	$c_{2,3}$	$c_{2,4}$	$c_{2,5}$	
7	$c_{3,0}$	$c_{3,1}$	$c_{3,2}$	$c_{3,3}$	$c_{3,4}$	$c_{3,5}$	$c_{3,6}$

Fig. 8: Memory mapping.

correctly, we estimate period of various signal waves. We use Cadence Incisive 15.2 for RTL simulation. ARS can estimate various parameter[22], i.e., the estimation of the fundamental waveforms, the number of signals and its periods for the multiple signals contained in the observed noisy signal from accumulators. However, this paper focuses on a hardware implementation method, we evaluate estimation of periods. We use MATLAB to generate input waveforms, and table 1 shows default parameters. Although circuit scales and clock frequencies of ARS-DFF and ARS-MEM are different because they adopt different hardware implementation methods, they implement the same software algorithm and output the same results, so only the simulation results for

Table 1: Default parameters of simulations.

Parameter	Value
Period of input signal	451
Total number of samples	18000
Input SNR	0dB
Fundamental waveform	sinc
Number of signals	1 (single wavelength)
Maximum period	6sec
Minimum period	2sec
Sampling Clock	100Hz
ADC resolution	12bit

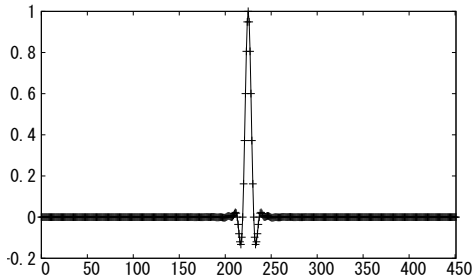


Fig. 9: Fundamental waveform of sinc.

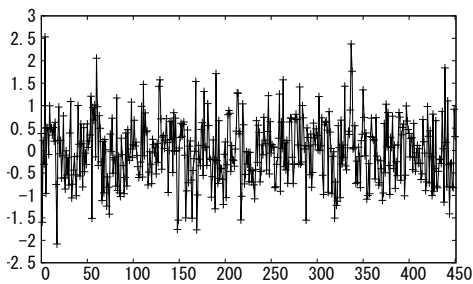


Fig. 10: The first 451 samples of sinc waveform with noise.

ARS-DFF are shown in this section.

The original ARS is designed by MATLAB, so it can handle floating point numbers. However, to calculate floating point values, hardware becomes complex. We assume to use our proposed design after analog-digital-converter (ADC). Therefore, we normalize the real number generated by MATLAB to integers.

4.1 Period estimation of a sinc function

Figure 9 is a fundamental sinc waveform, where the period is set at 451 samples. We generate the periodic sinc waveforms for input by concatenating the fundamental waveforms. To show robustness against noise, we add noise to the generated waveform. Figure 10 shows the first 451 samples with noise, where signal-to-noise power ratio (SNR) is 0dB.

Figure 11 shows the simulation results. Figure 11 depicts the maximum number of each accumulator. According to fig. 11, our designed hardware estimates the period as approximately 450. Figure 12 shows an enlarged view of the area in fig. 11, where the X-axis is 447 to 455 and the Y-axis is 8000 to 11000. According to fig. 12, our designed

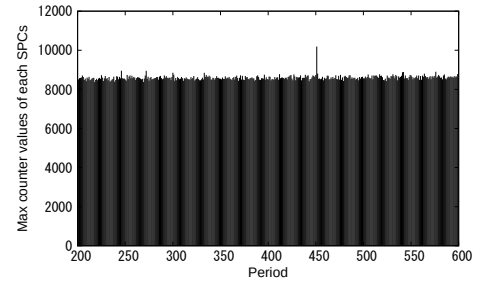


Fig. 11: The maximum numbers of each accumulator.

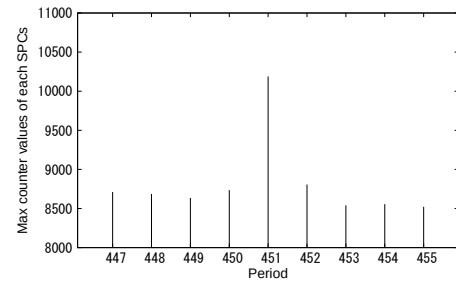


Fig. 12: The maximum numbers of each accumulator (enlarged).

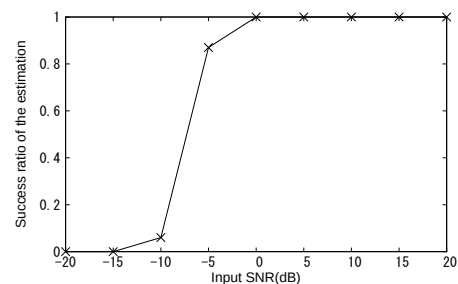


Fig. 13: Success ratio of the estimation vs. SNR.

hardware can estimate period as 451 correctly.

To show the robustness against noise, we evaluate the success rate of period estimation by varying SNR of the input signals. We use 100 series of input signals where SNR is varied from -20dB to 20dB by 5dB. Figure 13 shows the success ratio of the estimation for varying input signal noise from -20 ~ 20dB. If the SNR of input signal is -5dB, success ratio of period estimation is 87%, and if the SNR of input signal is larger than 0dB, the success ratio becomes 100%.

4.2 Periods estimation of combined sinc functions

ARS can estimate periods of the combined signals. Figure 14 is the outline of two fundamental sinc waveforms. To estimates two or three combined signals, we use combined sinc waveforms generated by $sinc_{451}(x) + sinc_{452}(x)$ and $sinc_{451}(x) + sinc_{452}(x) + sinc_{453}(x)$, respectively, where x is time and $sinc_y(x)$ is sinc function of period y . To generate $sinc_{452}(x)$ and $sinc_{453}(x)$, we add one or two extra zero into

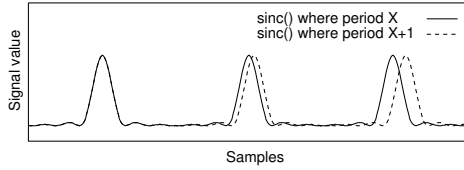


Fig. 14: Two sinc waveforms.

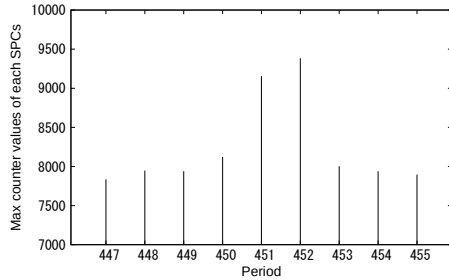


Fig. 15: Two combined sinc waveforms (enlarged).

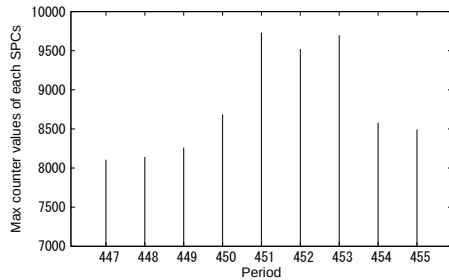


Fig. 16: Three combined sinc waveforms (enlarged).

the fundamental sinc waveform, which is the same manner as shown in [22].

Figure 15 and fig. 16 show the maximum numbers of each accumulator, where input signals are two and three combined sinc waveforms. To show the results clear, we show only the areas where the X-axis is 447 to 455 and the Y-axis is 7000 to 10000.

According to fig. 15, two values from the largest numbers are 451 and 452, which means our designed hardware can estimate the period of combined signals correctly. In the same way, fig. 16 shows that our designed hardware can estimate the period 451, 452 and 453 correctly.

5. VLSI implementation and performance evaluation

To evaluate both approaches, we design VLSI chip using Synopsys Design Compiler for logical synthesis and Synopsys IC Compiler II for layout on ROHM 0.18 μ m 5-metal layer CMOS process with standard cell library[32]. Default design parameters are shown in table 2, and table 3 shows design tools. In Section 4, the sampling frequency was set to 100 Hz and range of period to estimate was set from 2 to 6 seconds for the RTL simulation, which can estimate a period between 200 and 600 samples. However, the sampling fre-

Table 2: Default design parameters.

Parameter	Value
Maximum period	6sec
Minimum period	2sec
Sampling frequency	10Hz
ADC resolution	12bit

Table 3: Design tools.

Process	Tool
Design Verification	Cadence Incisive 15.2
Logical Synthesis	Synopsys Design Compiler 2021.06
Place & Route	Synopsys IC Compiler II 2021.06
Physical Verification	Mentor Graphics Calibre 2018.1
Power estimation	Synopsys PrimeSim XA 2022.06-SP1

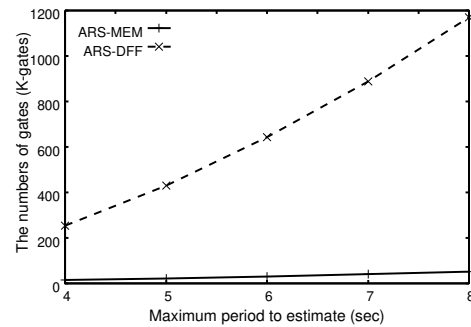


Fig. 17: The num. of gates vs. max. period to estimate.

quency was set to 10Hz for VLSI implementation because the circuit scale of ARS-DFF is too large to implement as VLSI. So the designed VLSI can estimate a period between 20 and 60 samples.

5.1 Circuit scale

Figure 17 shows the number of gates reported by logical synthesis. X-axis is maximum period to estimate, and Y-axis is the number of gates in terms of equivalent NAND gates. Solid line shows ARS-MEM, and dashed line shows ARS-DFF. ARS-MEM approach requires SRAM hardware macro cells, and we estimate the circuit scale of the memory unit from datasheet of ROHM 0.18 μ m library. According to fig. 17, ARS-MEM can implement ARS algorithm with fewer gates than that of ARS-DFF. The numbers of gates of both approaches are proportional to the square of the maximum period to estimate, because the memory capacity is proportional to the square of the maximum period to estimate discussed in Section 3, and the memory capacity is dominant factor affects circuit scale.

5.2 Circuit speed

Table 4 shows maximum clock frequency reported by logical synthesis. The maximum clock frequency of ARS-MEM is faster than that of ARS-DFF in all conditions. ARS-DFF can process one input data by one clock cycle. Therefore,

Table 4: Max clock frequency (MHz).

Max. period (sec)	ARS-DFF	ARS-MEM	ARS-MEM (sampling)
4	58.2	127.7	3.04
5	56.4	118.6	1.91
6	57.5	106.4	1.30
7	56.4	98.7	0.97
8	56.2	92.3	0.76

maximum sampling frequency is same to chip clock frequency. However, ARS-MEM need multi-cycles to process one data as shown in Section 3.2. The fourth column of table 4 shows maximum sampling frequency of ARS-MEM. For example, the ARS-MEM chip which can estimate period between 2 seconds and 4 second, P_{min} and P_{max} are 20 and 40, respectively. Therefore, the maximum clock frequency of ARS-MEM chip is 127.7MHz, however, the maximum sampling clock frequency is 3.04MHz because to process one signal from ADC requires $(P_{max} - P_{min} + 1) \times 2$ memory accesses. Compared to the second column and fourth column, ARS-DFF can process faster input signal. Generally, non-contact vital sensing such as health monitoring does not treat high frequency signal.

We assume that the target application is a non-contact vital sensing which measures vital data of breathing, sampling frequency is 10Hz, and range of period to estimate is 2 to 6 seconds. The clock frequency of ARS-DFF is same to the sampling frequency, however, the clock frequency of ARS-MEM is about 80 times higher than the sampling frequency. Therefore, the clock frequencies to drive the circuits of ARS-DFF and ARS-MEM are 10Hz and 80KHz, respectively. For such applications, ARS-MEM is better approach compared to ARS-DFF because of circuit scale shown in Section 5.1 and power consumption shown in Section 5.4.

Furthermore, if there is a margin in clock frequency, it has a possibility for further reduction of power consumption by adopting dynamic voltage and frequency scaling (DVFS) technique. However, since DVFS strongly depends on semiconductor device technology, this paper does not discuss further reduction by DVFS, so that we avoid too much dependence on a particular semiconductor technology.

5.3 Layout

We design VLSI chip of both ARS-DFF and ARS-MEM using Synopsys IC Compiler II. Design parameters for both methods are shown in table 2. Figure 18 shows ARS-DFF, and fig. 19 shows ARS-MEM. Chip size of ARS-DFF is $10.6mm^2$. Chip size of ARS-MEM without SRAM macro cell is $0.1mm^2$. We estimate the SRAM macro cell area from datasheet, and total chip size of ARS-MEM with SRAM macro cell is $0.436mm^2$.

ARS-MEM can be implemented in only 5% of the chip area of ARS-DFF. According to table 4, maximum sampling frequency of ARS-MEM is 1.30MHz, and it is lower than that of ARS-DFF (57.5MHz). However, it is not fatal problem because the required sampling frequency for applications

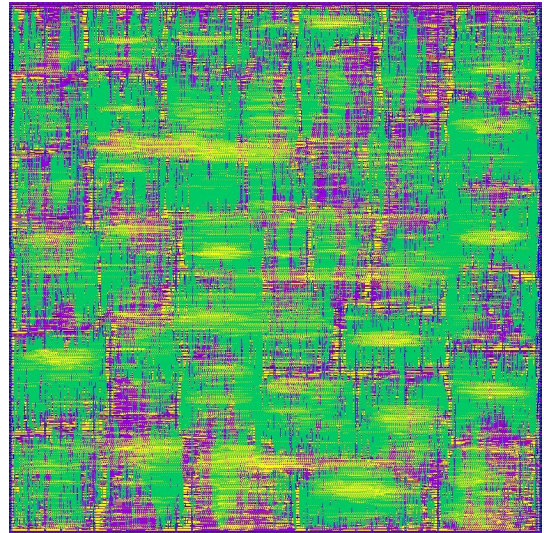


Fig. 18: Primary layout of ARS-DFF.

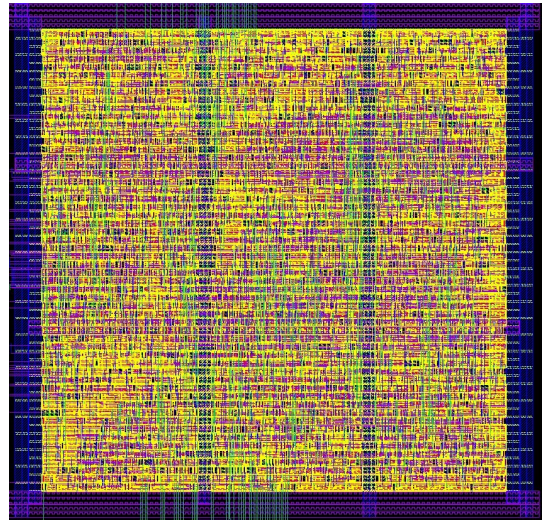


Fig. 19: Primary layout of ARS-MEM without SRAM macro cell.

such as vital sensing or health care are not very high.

5.4 Power consumption

We also estimate power consumption of the designs. To estimate power consumption, we use Synopsys PrimeSim XA. ARS-DFF is fully synthesizable and power consumption can be estimated by PrimeSim XA. ARS-MEM is also synthesizable, but it requires SRAM macro cell. However, we cannot estimate power consumption of SRAM macro cell using PrimeSim XA. Therefore, we estimate it from datasheet of SRAM macro cell.

To compare the power consumption if we implement the ARS algorithm on a microprocessor, this paper also shows the power consumption when implemented on Renesas Electronics RE01[33] which is 32bit microprocessor based on

Table 5: Power consumption.

Approach	Power consumption(μ W)
ARS-DFF	3.60
ARS-MEM	1.50
RE01	4050

ARM Cortex-M0+ core. We estimate the power consumption using Renesas Electronics Evaluation Kit RE01 1500KB and IWATSU VOAC7520H.

Table 5 shows power consumption of both approaches. According to our evaluation, the power consumption of ARS-MEM is the lowest among the three implementation methods. However, both ARS-DFF and ARS-MEM achieve a significant reduction in power consumption compared to implementation on a microprocessor.

6. Conclusions

This paper proposes two hardware implementation methods of ARS which can estimate periods of waveforms. ARS-DFF is simple and fast approach, and maximum sampling frequency is 57.5MHz. On the other hand, ARS-MEM is smaller circuit scale and lower power consumption approach. The sampling frequency of ARS-MEM is 1.3MHz, but hardware scale is only 5% compared to ARS-DFF. The power consumptions of ARS-DFF and ARS-MEM are 3.60μ W and 1.50μ W, respectively, and these power consumptions are dramatically small compared to 4050μ W which is the power consumption implemented on a microprocessor. Most of vital sensing applications do not require high sampling frequency. For example, let us think about the non-contact vital sensing for measuring breathing of a human. In this case, a sampling frequency setting of 10Hz is sufficient since the period of the breathing is typically assumed to be 2 to 6 seconds. Then, in this example, we can reduce the circuit scale and the power consumption of ARS-MEM till 5% and 42%, respectively, compared with ARS-DFF. Thus, it is obvious that ARS-MEM is a better approach rather than ARS-DFF.

As our future works, we will implement the hardware to handle threshold, fabricate VLSI chips, and evaluate performance. We will also evaluate the circuit scale and operating frequency based on layout results.

Acknowledgments

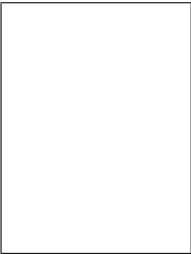
This work was supported through the activities of VDEC, The University of Tokyo, in collaboration with NIHON SYNOPSYS G.K., Cadence Design Systems, Mentor Graphics, TOOL Corporation, and Rohm Corporation.

References

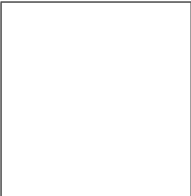
- [1] J. Yan, Y. Meng, L. Lu, and L. Li, "Industrial big data in an industry 4.0 environment: Challenges, schemes, and applications for predictive maintenance," *Ieee Access*, vol.5, pp.23484–23491, 2017.
- [2] Y. He, J. Guo, and X. Zheng, "From surveillance to digital twin: Challenges and recent advances of signal processing for industrial internet of things," *IEEE Signal Processing Magazine*, vol.35, no.5, pp.120–129, 2018.
- [3] E. Bertino, M.R. Jahanshahi, A. Singla, and R.T. Wu, "Intelligent iot systems for civil infrastructure health monitoring: a research roadmap," *Discover Internet of Things*, vol.1, pp.1–11, 2021.
- [4] Y. Cui, F. Liu, X. Jing, and J. Mu, "Integrating sensing and communications for ubiquitous iot: Applications, trends, and challenges," *IEEE Network*, vol.35, no.5, pp.158–167, 2021.
- [5] P. Bamurigire and A. Vodacek, "Validating algorithms designed for fertilization control in rice farming system," *Discover Internet of Things*, vol.3, no.1, p.4, 2023.
- [6] P. Shrivastava, V. Tewari, C. Gupta, and G. Singh, "Iot and radio telemetry based wireless engine control and real-time position tracking system for an agricultural tractor," *Discover Internet of Things*, vol.3, no.1, p.6, 2023.
- [7] Z. Wu, H. Luo, Y. Yang, P. Lv, X. Zhu, Y. Ji, and B. Wu, "K-pdm: Kpi-oriented machinery deterioration estimation framework for predictive maintenance using cluster-based hidden markov model," *IEEE Access*, vol.6, pp.41676–41687, 2018.
- [8] K. Tsuji, S. Imai, R. Takao, T. Kimura, H. Kondo, and Y. Kamiya, "A machine sound monitoring for predictive maintenance focusing on very low frequency band," *SICE Journal of Control, Measurement, and System Integration*, vol.14, no.1, pp.27–38, 2021.
- [9] R. Qi, J. Zhang, and K. Spencer, "A review on data-driven condition monitoring of industrial equipment," *Algorithms*, vol.16, no.1, p.9, 2022.
- [10] P. Henriquez, J.B. Alonso, M.A. Ferrer, and C.M. Travieso, "Review of automatic fault diagnosis systems using audio and vibration signals," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol.44, no.5, pp.642–652, 2013.
- [11] R. Srilakshmi, C. Ratnam, and V. Vitalrao, "A review on fault detection diagnosis and prognosis in vibration measurement through wavelets on machine elements," *International Journal of Applied Engineering Research*, vol.14, no.2, pp.547–555, 2019.
- [12] C. Jiang, C. Fu, Z. Zhao, and X. Du, "Effective anomaly detection in smart home by integrating event time intervals," *Procedia Computer Science*, vol.210, pp.53–60, 2022.
- [13] Y. Koyama, M. Nishiyama, and K. Watanabe, "Smart textile using hetero-core optical fiber for heartbeat and respiration monitoring," *IEEE Sensors Journal*, vol.18, no.15, pp.6175–6180, 2018.
- [14] T. Hall, D.Y. Lie, T.Q. Nguyen, J.C. Mayeda, P.E. Lie, J. Lopez, and R.E. Banister, "Non-contact sensor for long-term continuous vital signs monitoring: A review on intelligent phased-array doppler sensor design," *Sensors*, vol.17, no.11, p.2632, 2017.
- [15] T. Zhang, J. Sarrazin, G. Valerio, and D. Istrate, "Estimation of human body vital signs based on 60 ghz doppler radar using a bound-constrained optimization algorithm," *Sensors*, vol.18, no.7, p.2254, 2018.
- [16] Z.K. Yang, H. Shi, S. Zhao, and X.D. Huang, "Vital sign detection during large-scale and fast body movements based on an adaptive noise cancellation algorithm using a single doppler radar sensor," *Sensors*, vol.20, no.15, p.4183, 2020.
- [17] C. Li, V.M. Lubecke, O. Boric-Lubecke, and J. Lin, "A review on recent advances in doppler radar sensors for noncontact health-care monitoring," *IEEE Transactions on microwave theory and techniques*, vol.61, no.5, pp.2046–2060, 2013.
- [18] H. Lee, B.H. Kim, J.K. Park, and J.G. Yook, "A novel vital-sign sensing algorithm for multiple subjects based on 24-ghz fmcw doppler radar," *Remote Sensing*, vol.11, no.10, p.1237, 2019.
- [19] W.W. Smith and J. Smith, *Handbook of real-time fast Fourier transforms*, IEEE New York, 1995.
- [20] S. Brisard and L. Dormieux, "Fft-based methods for the mechanics of composites: A general variational framework," *Computational Materials Science*, vol.49, no.3, pp.663–671, 2010.
- [21] E.E. Swartzlander and H.H. Saleh, "Fft implementation with fused floating-point operations," *IEEE Transactions on Computers*, vol.61, no.2, pp.284–288, 2010.
- [22] Y. Kamiya, "A simple parameter estimation method for periodic sig-

- nals applicable to vital sensing using doppler sensors,” *SICE Journal of Control, Measurement, and System Integration*, vol.10, no.5, pp.378–384, 2017.
- [23] K. Yano and Y. Kamiya, “A simple signal detection and waveform estimation for biometrics using doppler sensors,” 2016 IEEE 13th International Conference on Signal Processing (ICSP), pp.1329–1332, IEEE, 2016.
- [24] C. Xia, Y. Li, J. Yan, Y. Wang, H. Yan, R. Guo, and F. Li, “A practical approach to wrist pulse segmentation and single-period average waveform estimation,” 2008 International Conference on BioMedical Engineering and Informatics, pp.334–338, IEEE, 2008.
- [25] P. Miao, T. Zhang, W. Zhang, G. Ma, and Y. Liu, “Period estimation of the pn sequence for direct sequence spread spectrum in multipath environment,” 2009 5th International Conference on Wireless Communications, Networking and Mobile Computing, pp.1–4, IEEE, 2009.
- [26] I. Sahin, N. Yilmazer, and M.A. Simaan, “A method for subsample fetal heart rate estimation under noisy conditions,” *IEEE transactions on biomedical engineering*, vol.57, no.4, pp.875–883, 2009.
- [27] L. Pang and C. Qi, “A novel joint parameter estimation method of single-channel and time-frequency overlapped multi-component signal,” 2013 Cross Strait Quad-Regional Radio Science and Wireless Technology Conference, pp.124–127, IEEE, 2013.
- [28] D.D. Muresan and T.W. Parks, “A new approach to period estimation,” 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100), pp.II709–II712, IEEE, 2000.
- [29] M. Bellani, J. Epps, and G.A. Huttley, “A comparison of periodicity profile methods for sequence analysis,” *Proceedings 2012 IEEE International Workshop on Genomic Signal Processing and Statistics (GENSIPS)*, pp.78–81, IEEE, 2012.
- [30] N. Petrochilos, M. Rezk, A. Host-Madsen, V. Lubecke, and O. Boric-Lubecke, “Blind separation of human heartbeats and breathing by the use of a doppler radar remote sensing,” 2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP’07, pp.I–333, IEEE, 2007.
- [31] R. Takao and Y. Kamiya, “A performance evaluation of periodic signal analysis by ars compared with frequency analysis by fft,” *SICE Journal of Control, Measurement, and System Integration*, vol.13, no.5, pp.240–248, 2020.
- [32] H. Onodera, A. Hirata, T. Kitamura, and K. Tamaru, “P2lib: process-portable library and its generation system,” *Proceedings of CICC 97-Custom Integrated Circuits Conference*, pp.341–344, IEEE, 1997.
- [33] Renesas Electronics, RE01 Group Evaluation Kit RE01 1500KB User’s Manual, Aug 2020.

1994 and 2000, respectively. He joined Kokusai Denshin Denwa (KDD) Co., Ltd. in 1994, and in 1995, he joined KDD R&D Laboratories where he had been engaged in the research on adaptive array antennas and digital signal processing. From October 1998 to February 2001, he was a researcher at ATR Adaptive Communications Research Laboratories. From March 2001, he was a Postdoctoral Research Fellow at Ecole Nationale Supérieure d’Ingenieurs de Constructions Aeronautiques (ENSICA), Toulouse, France. Then he was with PCOM:13 ApS, Aalborg, Denmark as a Project Manager till March 2003. He is currently an associate professor at Aichi Prefectural University. His interests include array signal processing for various wireless applications.



Takahiro SASAKI He received the B.S., M.S. and Ph.D. degrees from Hiroshima City University in 1998, 2000, and 2003 respectively. From 2003 to 2018, he was a assistant professor at Mie University. Since 2018, he has been an associate professor at Aichi Prefectural University. His research interests are in low-power and high-performance computing, embedded processors, multi-processor architecture, parallel processing and video codec hardware.



Yukihiro KAMIYA He received M.S. and Ph.D. degrees in information engineering and information electronics from Nagoya University in