Recursive Filtering

Haotian Zhai and Bernd-Peter Paris
Department of Electrical and Computer Engineering
George Mason University
Fairfax, VA 22030
{hzhai,pparis}@gmu.edu

***Abstract*—**
***Index Terms*—**

I. The Proposed Algorithm

The form of a general second order IIR filter is given by

$$y_n = x_n + b_1 x_{n-1} + b_2 x_{n-2} + a_1 y_{n-1} + a_2 y_{n-2} \tag{1}$$

where x_n and y_n denote the input and output, respectively. The proposed algorithm in this paper processes multiple blocks (SIMD vectors) of data simultaneously so that it utilizes both intra-block and inter-block parallelism. Define M be the length of SIMD vector. The input data of multiple blocks is exactly distributed in a squared matrix (M by M), i.e.,

$$\boldsymbol{X} = \left[\begin{array}{c|c|c|c} x_0 & x_M & \cdots & x_{M^2-M} \\ x_1 & x_{M+1} & \cdots & x_{M^2-M+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{M-1} & x_{2M-1} & \cdots & x_{M^2-1} \end{array}\right]$$

where each block of data is aligned with column and seperated by line. We use $\boldsymbol{X}$ to denote the matrix version of input data. Furthermore, $\boldsymbol{X}[n]$ means the nth block and $\boldsymbol{X}[n][m]$ means the mth sample at nth block of input matrix $\boldsymbol{X}$.

To deal with the dependency problem of second order IIR filter equation, the algorithm that computes the solution of finite impulse response (FIR), the particular and homogeneous solutions apart and add them to get the complete solutions is proposed.

A. Finite Impulse Response (FIR)

To compute (1), we start with

$$v_n = x_n + b_1 x_{n-1} + b_2 x_{n-2} \tag{2}$$

where v_n is the output of second order FIR filter.

1) Non-transposed multi-block filtering of FIR: It is easy to apply the multi-block filtering method on FIR filter because of no dependency. The algorithm that computes the output of second order FIR filter with non-transposed matrix of data is given by

$$\begin{aligned} \boldsymbol{V} &= \left[\begin{array}{c|c|c|c} v_0 & v_M & \cdots & v_{M^2-M} \\ v_1 & v_{M+1} & \cdots & v_{M^2-M+1} \\ \vdots & \vdots & \ddots & \vdots \\ v_{M-1} & v_{2M-1} & \cdots & v_{M^2-1} \end{array}\right] \\ &= \boldsymbol{X} + b_1 \left[\begin{array}{c|c|c|c} x_{-1} & x_{M-1} & \cdots & x_{M^2-M-1} \\ x_0 & x_M & \cdots & x_{M^2-M} \\ \vdots & \vdots & \ddots & \vdots \\ x_{M-2} & x_{2M-2} & \cdots & x_{M^2-2} \end{array}\right] \\ &+ b_2 \left[\begin{array}{c|c|c|c} x_{-2} & x_{M-2} & \cdots & x_{M^2-M-2} \\ x_{-1} & x_{M-1} & \cdots & x_{M^2-M-1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{M-3} & x_{2M-3} & \cdots & x_{M^2-3} \end{array}\right] \end{aligned} \tag{3}$$

where $\boldsymbol{V}$ denotes the matrix version of output data of FIR. (3) can be computed among multiple blocks exploiting both intra-block and inter-block parallelism. The execution procedure of SIMD operation is as follows:

1. Load and broadcast two coefficients b_1 and b_2 in SIMD vector.

2. Load the first block (SIMD vector) of input $\boldsymbol{X}$ and shuffle it twice by x_{-1}, x_{-2} to get the initial conditions.

3. Compute two fused multiply and add (FMA) operations to get the first output block $\boldsymbol{V}[0]$.

4. Repeat step 2 and 3 by shuffling with the last two input data of previous block for $M-1$ times to get the entire $\boldsymbol{V}$.

5. Store the last block of $\boldsymbol{X}$ in the shift register as the initial conditions for next FIR filtering.

2) Transposed multi-block filtering of FIR: Next, We provide another solution to computing the output of second order FIR filter, which is based on transposed matrix version of data. The algorithm also exhibits both intra-block and inter-block parallelism and is given by

$$
\begin{aligned}
\boldsymbol{V}^T &= \left[\begin{array}{c|c|c|c} v_0 & v_1 & \cdots & v_{M-1} \\ v_M & v_{M+1} & \cdots & v_{2M-1} \\ \vdots & \vdots & \ddots & \vdots \\ v_{M^2-M} & v_{M^2-M+1} & \cdots & v_{M^2-1} \end{array}\right] \\
&= \boldsymbol{X}^T + b_1 \left[\begin{array}{c|c|c|c} x_{-1} & x_0 & \cdots & x_{M-2} \\ x_{M-1} & x_M & \cdots & x_{2M-2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{M^2-M-1} & x_{M^2-M} & \cdots & x_{M^2-2} \end{array}\right] \\
&+ b_2 \left[\begin{array}{c|c|c|c} x_{-2} & x_{-1} & \cdots & x_{M-3} \\ x_{M-2} & x_{M-1} & \cdots & x_{2M-3} \\ \vdots & \vdots & \ddots & \vdots \\ x_{M^2-M-2} & x_{M^2-M-1} & \cdots & x_{M^2-3} \end{array}\right]
\end{aligned} \tag{4}
$$

Compared with non-transposed data matrix, the initial conditions for each block already exist at the previous two blocks except for the first block. Thus, to compute (4), only two shuffles are needed to provide the initial conditions for the first block. The execution procedure is as follows:

1. Load and broadcast two coefficients b_1 and b_2 in SIMD vector.

2. Load the last two blocks of input $\boldsymbol{X}^T$ and shuffle those by x_{-2}, x_{-1} respectively to get the initial conditions for the first block.

3. Apply two FMA operations among the current block and the previous two blocks for M times to get the entire $\boldsymbol{V}^T$.

4. Store (another shuffle) the last two input data located at the last two blocks in the shift register as the initial conditions for next FIR filtering.

B. Infinite Impulse Response (IIR)

After computing (2), the general equation of second order IIR filter in (1) is simplified as

$$
y_n = v_n + a_1 y_{n-1} + a_2 y_{n-2} \tag{5}
$$

We extend (5) for the first couple of samples as follows:

$$
\begin{aligned}
y_0 &= \boxed{v_0} + a_1 y_{-1} + a_2 y_{-2} \\
y_1 &= v_1 + a_1 y_0 + a_2 y_{-1} = \boxed{v_1 + a_1 v_0} + (a_1^2 + a_2) y_{-1} + a_1 a_2 y_{-2} \\
y_2 &= v_2 + a_1 y_1 + a_2 y_0 = \boxed{v_2 + a_1(v_1 + a_1 v_0) + a_2 v_0} + \\
&\qquad a_1^2(a_1 y_{-1} + a_2 y_{-2}) + 2a_1 a_2 y_{-1} + a_2^2 y_{-2} \\
&\qquad \vdots
\end{aligned} \tag{6}
$$

where the terms in rectangle denote the particular solution and the rest of terms are the homogeneous part of the second order IIR equation. In this paper, we first calculate the particular solution by zeroing the homogenous part, where the process is called zero initial condition (ZIC). After that, we correct the initial conditions (homogeneous part) for getting the complete output. The latter process is called initial condition correction (ICC). Note, the above algorithms are all processed by multi-block filtering method.

1) Transposed zero initial condition (ZIC): From (6), the algorithm that computes the particular solution of second order IIR filter with transposed matrix of data is given by

$$
\begin{aligned}
\boldsymbol{W}^T &= \left[\begin{array}{c|c|c|c} w_0 & w_1 & \cdots & w_{M-1} \\ w_M & w_{M+1} & \cdots & w_{2M-1} \\ \vdots & \vdots & \ddots & \vdots \\ w_{M^2-M} & w_{M^2-M+1} & \cdots & w_{M^2-1} \end{array}\right] \\
&= \boldsymbol{V}^T + a_1 \left[\begin{array}{c|c|c|c} 0 & v_0 & v_1 + a_1 v_0 & \cdots \\ 0 & v_M & v_{M+1} + a_1 v_M & \cdots \\ \vdots & \vdots & \vdots & \ddots \\ 0 & v_{M^2-M} & v_{M^2-M+1} + a_1 v_{M^2-M} & \cdots \end{array}\right] \\
&+ a_2 \left[\begin{array}{c|c|c|c} 0 & 0 & v_0 & \cdots \\ 0 & 0 & v_M & \cdots \\ \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & v_{M^2-M} & \cdots \end{array}\right] \\
&= \boldsymbol{V}^T + a_1 \left[\begin{array}{c|c|c|c|c} 0 & w_0 & w_1 & \cdots & w_{M-2} \\ 0 & w_M & w_{M+1} & \cdots & w_{2M-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & w_{M^2-M} & w_{M^2-M+1} & \cdots & w_{M^2-2} \end{array}\right] \\
&+ a_2 \left[\begin{array}{c|c|c|c|c} 0 & 0 & w_0 & \cdots & w_{M-3} \\ 0 & 0 & w_M & \cdots & w_{2M-3} \\ \vdots & \vdots & \ddots & \ddots & \ddots \\ 0 & 0 & w_{M^2-M} & \cdots & w_{M^2-3} \end{array}\right]
\end{aligned} \tag{7}
$$

where $\boldsymbol{W}^T$ is the transposed output matrix of ZIC. Note, (7) doesn't compute the complete particular solution of (6). For example, for each sample in the first block, the compensations (multiplications by a_1 and a_2) are two empty blocks. This means the samples except for w_0 will lack of some information of past few samples due to the recursive property, e.g., the complete solution of w_m should be a combination of v_0, v_1, $\cdots$, v_m. Thus, when we do initial condition correction (ICC), each block should be corrected based on the last two output samples of the previous block instead of the original states y_{-1} and y_{-2} only. We will explain more details later in section ICC. The reason for doing (7) is to leverage the intra-block parallelism.

Also note, because of the intrinsic dependency problem of IIR filter, inter-block parallelism can't be achieved simultaneously when process (7). The execution procedure is as follows:

1. Load and broadcast two coefficients a_1 and a_2 in SIMD vector.

2. load the first block of $\boldsymbol{V}^T$ and store it as the first block in the output matrix $\boldsymbol{W}^T$.

3. load the second block of $\boldsymbol{V}^T$ and apply one FMA with the previous output block to get the second block of $\boldsymbol{W}^T$.

4. load the rest blocks for $M-2$ times and apply two FMAs with the previous two output blocks to get the entire $\boldsymbol{W}^T$.

2) Non-transposed zero initial condition: If we re-write (7) with non-transposed data version, i.e.,

$$
\begin{aligned}
\boldsymbol{W} &= \left[\begin{array}{c|c|c|c} w_0 & w_M & \cdots & w_{M^2-M} \\ w_1 & w_{M+1} & \cdots & w_{M^2-M+1} \\ \vdots & \vdots & \ddots & \vdots \\ w_{M-1} & w_{2M-1} & \cdots & w_{M^2-1} \end{array}\right] = \boldsymbol{V} \\
&+ a_1 \left[\begin{array}{c|c|c|c} 0 & 0 & \cdots & 0 \\ v_0 & v_M & \cdots & v_{M^2-M} \\ v_1{+}a_1v_0 & v_{M+1}{+}a_1v_M & \cdots & v_{M^2-M+1}{+}a_1v_{M^2-M} \\ \vdots & \vdots & \ddots & \vdots \end{array}\right] \\
&+ a_2 \left[\begin{array}{c|c|c|c} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ v_0 & v_M & \cdots & v_{M^2-M} \\ \vdots & \vdots & \ddots & \vdots \end{array}\right]
\end{aligned} \tag{8}
$$

We can see that (8) can no longer be computed among multiple blocks. This is because the dependency problem now occurs inside each block instead of happening among blocks as (7). However, (8) can still achieve intra-block parallelism by block-filtering algorithm based on (6) as shown below

$$
\begin{aligned}
\boldsymbol{W} &= \left[\begin{array}{c|c|c|c} w_0 & w_M & \cdots & w_{M^2-M} \\ w_1 & w_{M+1} & \cdots & w_{M^2-M+1} \\ \vdots & \vdots & \ddots & \vdots \\ w_{M-1} & w_{2M-1} & \cdots & w_{M^2-1} \end{array}\right] \\
&= \left[\begin{array}{ccccc} 1 & 0 & \cdots & \cdots & 0 \\ a_1 & 1 & \ddots & 0 & \vdots \\ a_1^2+a_2 & a_1 & 1 & \ddots & \vdots \\ a_1^3+2a_1a_2 & a_1^2+a_2 & a_1 & 1 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{array}\right] \boldsymbol{V} = \boldsymbol{H}\boldsymbol{V}
\end{aligned} \tag{9}
$$

where (9) performs a matrix multiplication and $\boldsymbol{H}$ denotes the transition matrix from $\boldsymbol{V}$ to $\boldsymbol{W}$. $\boldsymbol{H}$ can be pre-computed by an impulse response, $\left[0\ 1 \,\vdots\, a_1\ a_1^2+a_2\ (a_1^2+a_2)a_1{+}a_1a_2\ \cdots\right]$, where each response is obtained by multiplying the previous two responses with a_2 and a_1. The vertical dashed line is used to seperate the initial state and the resulting response sequence, which is denoted by $\boldsymbol{h}_1 = \left[a_1\ a_1^2+a_2\ (a_1^2+a_2)a_1{+}a_1a_2\ \cdots\right]$. The length of $\boldsymbol{h}_1$ is M. The execution procedure of (9) is as follows:

1. Pre-calculate and load the transition matrix $\boldsymbol{H}$. Note, this won't take the run time.

2. Load and broadcast every sample in the first block of $\boldsymbol{V}$. Apply M times FMAs with each SIMD vectors at the same index in $\boldsymbol{H}$ and accumulate them together to get the first block of $\boldsymbol{W}$.

3. Repeat step 2 for $M-1$ times for the rest blocks to get the entire $\boldsymbol{W}$.

Note, the resulting $\boldsymbol{W}$ from (9) is exactly the same as $\boldsymbol{W}^T$ from (7) without matrix transpose.

3) Non-transposed initial condition correction (ICC): After computing the particular solution of second order IIR filter, we now compensate for the initial conditions to get the complete solution. Recall that the output matrix of ZIC in either (7) or (9) only has partial particular solution. Thus, based on (6) and along with ZIC, the algorithm that computes the homogeneous solution of second order IIR filter with non-transposed matrix of data is given by

$$
\begin{aligned}
\boldsymbol{Y} &= \left[\begin{array}{c|c|c|c} y_0 & y_M & \cdots & y_{M^2-M} \\ y_1 & y_{M+1} & \cdots & y_{M^2-M+1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{M-1} & y_{2M-1} & \cdots & y_{M^2-1} \end{array}\right] = \boldsymbol{W} + \\
&\left[\begin{array}{cc} a_1 & a_2 \\ a_1^2+a_2 & a_1a_2 \\ a_1^3+2a_1a_2 & a_1^2a_2+a_2^2 \\ \vdots & \vdots \end{array}\right] \left[\begin{array}{cccc} y_{-1} & y_{M-1} & \cdots & y_{M^2-M-1} \\ y_{-2} & y_{M-2} & \cdots & y_{M^2-M-2} \end{array}\right] \\
&= \boldsymbol{W} + \boldsymbol{h}\boldsymbol{Y}_p
\end{aligned} \tag{10}
$$

where $\boldsymbol{h}$ denotes the transition matrix for the pre-state matrix $\boldsymbol{Y}_p$, and it is composed by two impulse response sequences, $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$, where $\boldsymbol{h}_2$ is the resulting response sequence of $\left[1\ 0 \,\vdots\, a_2\ a_1a_2\ a_1^2a_2{+}a_2^2\ \cdots\right]$, following the same rule as $\boldsymbol{h}_1$. Compared with (6), because the output samples of ZIC from (7) or (9) only has partial particular solution, we can't compensate for every block by the original initial conditions y_{-1} and y_{-2}. Instead, we correct each block based on the pre-states of the start sample at each block, e.g., $\boldsymbol{W}[1]$ should be compensated by $\boldsymbol{Y}[0][M-1]$ and $\boldsymbol{Y}[0][M-2]$. Furthermore, (10) exploits intra-block parallelism and multi-block filtering method. The execution procedure is as follows:

1. Pre-calculate and load the transition matrix $\boldsymbol{h}$.

2. Load and broadcast the first two initial conditions y_{-1}, y_{-2} and apply two FMAs with two SIMD vectors in $\boldsymbol{h}$ to get the first block of $\boldsymbol{Y}$.

3. Repeat step 2 for $M-1$ times with two initial conditions, which are the last two samples of the prvious output block, to get the entire $\boldsymbol{Y}$.

4. Store the last block of $\boldsymbol{Y}$ in the shift register as the initial conditions for next IIR filtering.

4) Transposed initial condition correction (ICC): To compute the homogeneous solution with transposed matrix of data, we re-write (10) in transposed version, i.e.,

$$
\begin{aligned}
\boldsymbol{Y}^T &= \left[\begin{array}{c|c|c|c} y_0 & y_1 & \cdots & y_{M-1} \\ y_M & y_{M+1} & \cdots & y_{2M-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{M^2-M} & y_{M^2-M+1} & \cdots & y_{M^2-1} \end{array}\right] = \boldsymbol{W}^T + \\
&\left[\begin{array}{c|c} y_{-1} & y_{-2} \\ y_{M-1} & y_{M-2} \\ \vdots & \vdots \\ y_{M^2-M-1} & y_{M^2-M-2} \end{array}\right] \left[\begin{array}{cccc} a_1 & a_1^2+a_2 & a_1^3+2a_1a_2 & \cdots \\ a_2 & a_1a_2 & a_1^2a_2+a_2^2 & \cdots \end{array}\right] \\
&= \boldsymbol{W}^T + \boldsymbol{Y}_p^T\boldsymbol{h}^T
\end{aligned} \tag{11}
$$

Compared with (10), we need one extra step to process (11) that is to compute the two initial-condition vectors in $\boldsymbol{Y}_p^T$. The two initial conditions can be computed at first since they only depend on $\boldsymbol{W}$ and the two pre-states y_{-2} and y_{-1}. Thus, the algorithm of calculating $\boldsymbol{Y}_p^T$ is given by

$$
\begin{bmatrix} y_{M-2} \\ y_{2M-2} \\ \vdots \\ y_{M^2-2} \\ \hline y_{M-1} \\ y_{2M-1} \\ \vdots \\ y_{M^2-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{h}_{2,M-2} & \boldsymbol{h}_{1,M-2} \\ \boldsymbol{h}_{2,M-2}\boldsymbol{h}_{2,M-2} & \boldsymbol{h}_{1,M-2}\boldsymbol{h}_{2,M-2} \\ +\boldsymbol{h}_{2,M-1}\boldsymbol{h}_{1,M-2} & +\boldsymbol{h}_{1,M-1}\boldsymbol{h}_{1,M-2} \\ \vdots & \vdots \\ \hline \boldsymbol{h}_{2,M-1} & \boldsymbol{h}_{1,M-1} \\ \boldsymbol{h}_{2,M-2}\boldsymbol{h}_{2,M-1} & \boldsymbol{h}_{1,M-2}\boldsymbol{h}_{2,M-1} \\ +\boldsymbol{h}_{2,M-1}\boldsymbol{h}_{1,M-1} & +\boldsymbol{h}_{1,M-1}\boldsymbol{h}_{1,M-1} \\ \vdots & \vdots \end{bmatrix} \begin{bmatrix} y_{-2} \\ y_{-1} \end{bmatrix} +
$$

$$
\left[\begin{array}{ccc|ccc} 1 & 0 & \cdots & 0 & \cdots & \cdots \\ \boldsymbol{h}_{2,M-2} & 1 & 0 & \boldsymbol{h}_{1,M-2} & 0 & 0 \\ \boldsymbol{h}_{2,M-2}\boldsymbol{h}_{2,M-2}+ & \ddots & \ddots & \boldsymbol{h}_{1,M-2}\boldsymbol{h}_{2,M-2}+ & \ddots & \ddots \\ \boldsymbol{h}_{2,M-1}\boldsymbol{h}_{1,M-2} & & & \boldsymbol{h}_{1,M-1}\boldsymbol{h}_{1,M-2} & & \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \\ \hline 0 & \cdots & \cdots & 1 & 0 & \cdots \\ \boldsymbol{h}_{2,M-1} & 0 & 0 & \boldsymbol{h}_{1,M-1} & 1 & 0 \\ \boldsymbol{h}_{2,M-2}\boldsymbol{h}_{2,M-1}+ & \ddots & \ddots & \boldsymbol{h}_{1,M-2}\boldsymbol{h}_{2,M-1}+ & \ddots & \ddots \\ \boldsymbol{h}_{2,M-1}\boldsymbol{h}_{1,M-1} & & & \boldsymbol{h}_{1,M-1}\boldsymbol{h}_{1,M-1} & & \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \end{array}\right]
$$

$$
\cdot \left[\begin{array}{ccc|ccc} w_{M-2} & w_{2M-2} & \cdots & w_{M^2-2} & w_{M-1} & \cdots \end{array} \; w_{M^2-1}\right]^T
$$

$$
= \boldsymbol{D}\boldsymbol{y}_p + \boldsymbol{T} \cdot \begin{bmatrix} \boldsymbol{W}^T[M-2] \\ \boldsymbol{W}^T[M-1] \end{bmatrix} \tag{12}
$$

where $\boldsymbol{h}_{x,n}$ is equivalent to $\boldsymbol{h}_x[n]$ and $\boldsymbol{D}$ and $\boldsymbol{T}$ are two transition matrices for the pre-state vector $\boldsymbol{y}_p$ and the corresponding part of $\boldsymbol{W}$. $\boldsymbol{D}$ and $\boldsymbol{T}$ can be pre-computed by the following two joint impulse responses, which are

1. $\left[1\ 0 \;\vdots\; \boldsymbol{h}_{2,M-2}\ \boldsymbol{h}_{2,M-1}\ \boldsymbol{h}_{2,M-2}\boldsymbol{h}_{2,M-2}+\boldsymbol{h}_{2,M-1}\boldsymbol{h}_{1,M-2}\ \boldsymbol{h}_{2,M-2}\boldsymbol{h}_{2,M-1}+\boldsymbol{h}_{2,M-1}\boldsymbol{h}_{1,M-1}\cdots\right]$
2. $\left[0\ 1 \;\vdots\; \boldsymbol{h}_{1,M-2}\ \boldsymbol{h}_{1,M-1}\ \boldsymbol{h}_{1,M-2}\boldsymbol{h}_{2,M-2}+\boldsymbol{h}_{1,M-1}\boldsymbol{h}_{1,M-2}\ \boldsymbol{h}_{1,M-2}\boldsymbol{h}_{2,M-1}+\boldsymbol{h}_{1,M-1}\boldsymbol{h}_{1,M-1}\cdots\right]$

The first position of initial state corresponds to $\boldsymbol{h}_2$ while the second one points to $\boldsymbol{h}_1$. Furthermore, each of the above two impulse sequences calculates a pair of impulse responses simultaneously and interleavedly. The current pair of impulse response are calculated based on the previous pair and the former determines multiplying $\boldsymbol{h}_x$ of position $M-2$ while the latter points to $M-1$.

Note, after computing (12), the pre-state matrix $\boldsymbol{Y}_p^T$ in (11) can be then obtained by shuffling with y_{-1} and y_{-2}. Furthermore, since $\boldsymbol{h}^T$ can be pre-computed, the process of computing (11) achieves both intra-block and inter-block parallelism. The execution procedure of (12) and (11) is as follows:

1. Pre-compute and load two transition matrices $\boldsymbol{D}$ and $\boldsymbol{T}$.
2. Load and broadcast y_{-1}, y_{-2} and each sample in the last two blocks of $\boldsymbol{W}$, then apply $4M+4$ FMAs to get the last two blocks of $\boldsymbol{Y}^T$ following (12).
3. Shuffle the obtained two output blocks at step 2 with y_{-1} and y_{-2} respectively to get the pre-state matrix $\boldsymbol{Y}_p^T$.
4. Load and broadcast each element in $\boldsymbol{h}^T$ and apply two FMAs for $M-2$ times to get the entire $\boldsymbol{Y}^T$ following (11).

C. Matrix transpose

Since the proposed algorithm accepts both transposed and non-transposed data at each processing phase, it is necessary to discuss how matrix transpose is performed in SIMD operation. In this paper, the matrix transpose for multiple blocks of data is executed as follows:

1. Swap lower left part and upper right of the matrix, i.e.,

$$
\boldsymbol{X} \rightarrow \left[\begin{array}{ccc|ccc|c|ccc|c} x_0 & & & x_M & & & \cdots & x_{M/2} & & & \\ x_1 & & & x_{M+1} & & & & x_{M/2+1} & & & \\ \vdots & & & \vdots & & & & \vdots & & & \\ x_{M/2-1} & & & x_{M+M/2-1} & & & & x_{M-1} & & & \\ x_{M/2\cdot M} & & & x_{(M/2+1)\cdot M} & & & & x_{M/2\cdot M+M/2} & & & \\ x_{M/2\cdot M+1} & & & x_{(M/2+1)\cdot M+1} & & & & x_{M/2\cdot M+M/2+1} & & & \\ \vdots & & & \vdots & & & & \vdots & & & \\ x_{M/2\cdot(M+1)-1} & & & x_{(M/2+1)\cdot M+M/2-1} & & & & x_{(M/2+1)M-1} & & & \end{array}\right]
$$

2. Swap lower left part and upper right within each quadrant,

$$
\rightarrow \left[\begin{array}{ccc|ccc|ccc|c} x_0 & & & x_M & & & x_2 & & & \\ \vdots & & & \vdots & & & \vdots & & & \\ x_{M/4-1} & & & x_{M+M/4-1} & & & x_{M/2-1} & & & \\ x_{M/4\cdot M} & & & x_{(M/4+1)\cdot M} & & & x_{M/4\cdot M+2} & & & \\ \vdots & & & \vdots & & & \vdots & & & \\ x_{M/4\cdot M+M/4-1} & & & x_{M/4\cdot M+5M/4-1} & & & x_{M/4\cdot M+M/2-1} & & & \\ x_{M/2\cdot M} & & & x_{(M/2+1)\cdot M} & & & x_{M/2\cdot M+2} & & & \\ \vdots & & & \vdots & & & \vdots & & & \\ x_{M/2\cdot M+M/4-1} & & & x_{M/2\cdot M+5M/4-1} & & & x_{M/2\cdot M+M/2-1} & & & \\ x_{3M/2\cdot M} & & & x_{(3M/2+1)\cdot M} & & & x_{3M/2\cdot M+2} & & & \\ \vdots & & & \vdots & & & \vdots & & & \\ x_{3M/2\cdot M+M/4-1} & & & x_{3M/2\cdot M+5M/4-1} & & & x_{3M/2\cdot M+M/2-1} & & & \end{array}\right]
$$

3. Repeat step 2 with each sub-quadrant until step 4 is ready
4. Swap anti-diagonally adjacent elements to get $\boldsymbol{X}^T$.

It can be seen that the entire process of matrix transpose in SIMD operation requires $M\log(M)$ shuffles.