

Precise and scalable analogue matrix equation solving using resistive random-access memory chips

In the format provided by the
authors and unedited

Table of Contents:

Supplementary Notes	2
Supplementary Note 1: Algorithm of HP-INV circuit in Fig. 2a	2
Supplementary Note 2: Theoretical analysis on the HP-INV solver	3
Supplementary Note 3: Performance estimation of HP-INV solver	5
Supplementary Note 4: Area scalability of HP-INV solver	8
Supplementary Figures	10
Supplementary Fig. 1. Mapping method for HP-INV	10
Supplementary Fig. 2. Bit slicing for HP-MVM	11
Supplementary Fig. 3. Binary switching characteristics of wafer-level RRAM	12
Supplementary Fig. 4. 3-bit RRAM	13
Supplementary Fig. 5. Endurance performance of 3-bit RRAM	14
Supplementary Fig. 6. Retention performance of 3-bit RRAM	15
Supplementary Fig. 7. HP-INV solver system, consisting of LP-INV and HP-MVM experimental setups that are controlled by a personal computer (PC)	16
Supplementary Fig. 8. HP-MVM results in Fig. 2e	17
Supplementary Fig. 9. LP-INV results of a better conditioned matrix	18
Supplementary Fig. 10. Impact of condition number on the convergence of HP-INV	19
Supplementary Fig. 11. Impact of RME on the convergence of HP-INV	20
Supplementary Fig. 12. Impact of RME on the convergence of HP-INV using the same target matrix	21
Supplementary Fig. 13. The real-valued LP-INV circuit implemented with the bias-column method	22
Supplementary Fig. 14. Precision limits in HP-INV	23
Supplementary Fig. 15. HP-INV scheme with two-stage BlockAMC for complex-valued matrix A	24
Supplementary Fig. 16. Average convergence results of 16 HP-INV operations for computing the inverse of the 8×8 complex-valued matrix in Fig. 3f	25
Supplementary Fig. 17. Gray coding rule in massive MIMO	26
Supplementary Fig. 18. Processing results of image transmission by ZF detection with HP-INV in massive MIMO	27
Supplementary Fig. 19. Convergence behavior of HP-INV with and without wire resistance	28
Supplementary Fig. 20. Scaling behaviors of the chip area of HP-INV solver	30
Supplementary Tables	31
Supplementary Table 1. Summary of 4×4 numerical matrices utilized in experiment	31
Supplementary Table 2. Summary of large numerical matrices utilized in experiment	32
Supplementary References	33

Supplementary Note 1: Algorithm of HP-INV circuit in Fig. 2a.

Algorithm 1: HP-INV Algorithm

Input: matrix A and input vector b

Output: result x

```
1:    $\mathbf{x}^{(0)} = \mathbf{0}, \mathbf{r}^{(0)} = \mathbf{b}, \mathbf{d}^{(0)} = \beta^{(0)}\mathbf{r}^{(0)}$ 
2:    $A \rightarrow A_0, A_0 \rightarrow \alpha A_0 G_0$  (Normalizing and mapping)
3:   for  $k = 0 : L$ 
4:        $\Delta \mathbf{y}^{(k)} = A_0^{-1} \mathbf{d}^{(k)}$  (LP-INV)
5:        $\Delta \mathbf{r}^{(k)} = A \Delta \mathbf{y}^{(k)}$  (HP-MVM)
6:        $\Delta \mathbf{x}^{(k)} = \Delta \mathbf{y}^{(k)} / \beta^{(k)}$ 
7:        $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta \mathbf{x}^{(k)}$ 
8:        $\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \Delta \mathbf{r}^{(k)} / \beta^{(k)}$ 
9:       if  $-\log_2 \|\mathbf{r}^{(k+1)}\| > t$  or  $k = L$  break and output  $\mathbf{x}^{(k+1)}$ 
10:      else  $\mathbf{d}^{(k+1)} = \beta^{(k+1)} \mathbf{r}^{(k+1)}$ 
11:      end if
12:  end for
```

Supplementary Note 2: Theoretical analysis on the HP-INV solver.

Convergence condition for the HP-INV circuit. According to the iterative algorithm in Supplementary Note 1, the state function of the system is expressed as

$$\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \frac{\Delta \mathbf{r}^{(k)}}{\beta^{(k)}} = \mathbf{r}^{(k)} - \frac{\mathbf{A}(\mathbf{A}_0^{-1} \mathbf{d}^{(k)})}{\beta^{(k)}} = (\mathbf{I} - \mathbf{A}\mathbf{A}_0^{-1})\mathbf{r}^{(k)} \quad (\text{S1})$$

To assure the convergence of HP-INV circuit, the spectral radius of $\mathbf{I} - \mathbf{A}\mathbf{A}_0^{-1}$, *i.e.*, $\rho(\mathbf{I} - \mathbf{A}\mathbf{A}_0^{-1})$, should be less than 1.

Proof of effectiveness of approximating $\mathbf{A}_{4s}$ with $\mathbf{A}_4$ in BlockAMC for strictly diagonally dominant matrix. In BlockAMC of solving $\mathbf{A}\mathbf{x} = \mathbf{b}$, the matrix and vector are split as follows

$$\begin{bmatrix} \mathbf{A}_1 & \mathbf{A}_2 \\ \mathbf{A}_3 & \mathbf{A}_4 \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \end{bmatrix} \quad (\text{S2})$$

Accordingly, the solution is

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} (\mathbf{A}_1^{-1} + \mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_{4s}^{-1}\mathbf{A}_3\mathbf{A}_1^{-1}) & -\mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_{4s}^{-1} \\ -\mathbf{A}_{4s}^{-1}\mathbf{A}_3\mathbf{A}_1^{-1} & \mathbf{A}_{4s}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \end{bmatrix} \quad (\text{S3})$$

where

$$\mathbf{A}_{4s} = \mathbf{A}_4 - \mathbf{A}_3\mathbf{A}_1^{-1}\mathbf{A}_2 \quad (\text{S4})$$

If $\mathbf{A}_{4s}$ is approximated by $\mathbf{A}_4$, Eq. S3 becomes

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} (\mathbf{A}_1^{-1} + \mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_4^{-1}\mathbf{A}_3\mathbf{A}_1^{-1}) & -\mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_4^{-1} \\ -\mathbf{A}_4^{-1}\mathbf{A}_3\mathbf{A}_1^{-1} & \mathbf{A}_4^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{g} \end{bmatrix} \quad (\text{S5})$$

Define matrices $\mathbf{V} = \begin{bmatrix} (\mathbf{A}_1^{-1} + \mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_4^{-1}\mathbf{A}_3\mathbf{A}_1^{-1}) & -\mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_4^{-1} \\ -\mathbf{A}_4^{-1}\mathbf{A}_3\mathbf{A}_1^{-1} & \mathbf{A}_4^{-1} \end{bmatrix}$, and $\mathbf{W} = \mathbf{I} - \mathbf{A}\mathbf{V} = \begin{bmatrix} \mathbf{O} & \mathbf{O} \\ \mathbf{A}_3\mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_4^{-1}\mathbf{A}_3\mathbf{A}_1^{-1} & -\mathbf{A}_3\mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_4^{-1} \end{bmatrix}$. To assure the reliability of HP-INV with BlockAMC, the spectral radius of matrix $\mathbf{W}$ should be less than 1.

The spectral radius of $\mathbf{W}$ is $\rho(\mathbf{W}) = \max|\lambda_{\mathbf{W}}| = \sqrt{\max \lambda_{\mathbf{W}\mathbf{W}^T}}$, and $\mathbf{W}\mathbf{W}^T = \begin{bmatrix} \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{M}\mathbf{M}^T + \mathbf{N}\mathbf{N}^T \end{bmatrix}$, where $\mathbf{M} = \mathbf{A}_3\mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_4^{-1}\mathbf{A}_3\mathbf{A}_1^{-1}$ and $\mathbf{N} = -\mathbf{A}_3\mathbf{A}_1^{-1}\mathbf{A}_2\mathbf{A}_4^{-1}$. Define $\mathbf{Z} = \mathbf{M}\mathbf{M}^T + \mathbf{N}\mathbf{N}^T$, it turns out that

$$\rho(\mathbf{W}) = \sqrt{\max \lambda_{\mathbf{W}\mathbf{W}^T}} = \sqrt{\max \lambda_{\mathbf{Z}}} = \sqrt{\|\mathbf{Z}\|_2} \quad (\text{S6})$$

Matrix $\mathbf{Z}$ can be expanded as

$$\mathbf{Z} = \mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_2 \mathbf{A}_4^{-1} \mathbf{A}_3 \mathbf{A}_1^{-1} (\mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_2 \mathbf{A}_4^{-1} \mathbf{A}_3 \mathbf{A}_1^{-1})^T + \mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_2 \mathbf{A}_4^{-1} (\mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_2 \mathbf{A}_4^{-1})^T \quad (\text{S7})$$

Considering a symmetric matrix $\mathbf{A}$, there are $\mathbf{A}_1^T = \mathbf{A}_1$, $\mathbf{A}_4^T = \mathbf{A}_4$, $\mathbf{A}_2^T = \mathbf{A}_3$, and $\mathbf{A}_3^T = \mathbf{A}_2$. These equations are also true for a real-valued matrix converted from a complex-valued symmetric matrix, such as the one in Fig. 3d, namely $\mathbf{A}_1^T = \mathbf{A}_1$, $\mathbf{A}_4^T = \mathbf{A}_4$, $\mathbf{A}_2^T = -\mathbf{A}_2 = \mathbf{A}_3$, and $\mathbf{A}_3^T = -\mathbf{A}_3 = \mathbf{A}_2$. Based on these observations, Eq. S7 becomes

$$\mathbf{Z} = \mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_2 \mathbf{A}_4^{-1} (\mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_1^{-1} \mathbf{A}_2 + \mathbf{I}) \mathbf{A}_4^{-1} \mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_2 \quad (\text{S8})$$

Therefore, Eq. S6 becomes

$$\begin{aligned} \rho(\mathbf{W}) &= \sqrt{\|\mathbf{Z}\|_2} \leq \sqrt{\|\mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_2 \mathbf{A}_4^{-1}\|_2 \|\mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_1^{-1} \mathbf{A}_2 + \mathbf{I}\|_2 \|\mathbf{A}_4^{-1} \mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_2\|_2} \\ &\leq \sqrt{\left(\frac{\kappa(\mathbf{A}_1) \|\mathbf{A}_3\|}{\|\mathbf{A}_1\|} \frac{\kappa(\mathbf{A}_4) \|\mathbf{A}_2\|}{\|\mathbf{A}_4\|} \right)^2 (1 + \rho(\mathbf{A}_3 \mathbf{A}_1^{-1} \mathbf{A}_1^{-1} \mathbf{A}_2))} \\ &\leq \sqrt{\left(\frac{\kappa(\mathbf{A}_1) \|\mathbf{A}_3\|}{\|\mathbf{A}_1\|} \frac{\kappa(\mathbf{A}_4) \|\mathbf{A}_2\|}{\|\mathbf{A}_4\|} \right)^2 \left(1 + \left(\frac{\kappa(\mathbf{A}_1) \|\mathbf{A}_3\|}{\|\mathbf{A}_1\|} \frac{\kappa(\mathbf{A}_1) \|\mathbf{A}_2\|}{\|\mathbf{A}_1\|} \right) \right)} \quad (\text{S9}) \end{aligned}$$

where κ is condition number defined as $\kappa(\cdot) = \|\cdot\| \|(\cdot)^{-1}\|$, and $\rho(\cdot) = \|\cdot\|_2 = \max(\lambda)$ is true for a symmetric matrix. For strictly diagonally dominant matrices, there is $\|\mathbf{A}_{i_1}\| \gg \|\mathbf{A}_{i_2}\|$ ($i_1 = \{1,4\}$, $i_2 = \{2,3\}$), and $\kappa(\mathbf{A}_{i_1})$ is usually small. As a result, there are $\frac{\kappa(\mathbf{A}_1) \|\mathbf{A}_3\|}{\|\mathbf{A}_1\|} \frac{\kappa(\mathbf{A}_4) \|\mathbf{A}_2\|}{\|\mathbf{A}_4\|} \ll 1$ and $\frac{\kappa(\mathbf{A}_1) \|\mathbf{A}_3\|}{\|\mathbf{A}_1\|} \frac{\kappa(\mathbf{A}_1) \|\mathbf{A}_2\|}{\|\mathbf{A}_1\|} \ll 1$, and thus $\rho(\mathbf{W}) < 1$ is proved.

Supplementary Note 3: Performance estimation of HP-INV solver.

Here we assess the throughput and energy efficiency of the system. To estimate the performance, parameters including the number of operations in digital processors, solution time and energy dissipation of AMC circuits are required. To solve matrix equation $\mathbf{Ax} = \mathbf{b}$ with size of $N \times N$ by Cholesky decomposition (CD) method, a digital processor needs $C_1 = \frac{1}{6} \times N(N + 1)(2N + 1)$ multiply-accumulate (MAC) operations for decomposition of $\mathbf{A}$, *i.e.*, $\mathbf{A} = \mathbf{LL}^T$, and $C_2 = 2 \times \left(\frac{1}{2} \times N(N + 1) + \frac{1}{2} \times (N - 1)N \right) = 2N^2$ MAC operations for getting the final result $\mathbf{x}$ by solving $\mathbf{Ly}_{temp} = \mathbf{b}$ and $\mathbf{Lx} = \mathbf{y}_{temp}$.

In cases where BlockAMC is not needed, the solution time is simply $t = \frac{p}{p_0} (t_{INV} + t_{MVM} + 2t_{ADC})$, assuming the parallel processing of HP-MVM with all bit-sliced matrices, where p and p_0 are the target precision and the precision increasement per cycle, respectively. t_{INV} and t_{MVM} are convergence time of LP-INV and HP-MVM circuits, respectively. t_{ADC} is conversion time of ADC. Considering that DAC and AMC circuits are usually turned on simultaneously, and DAC is far faster than AMC circuits, the conversion time of DAC is not included in this estimation. As a result, the energy dissipation of LP-INV, HP-MVM, ADC and DAC are represented as

$$E_{INV} = NP_{INV}t_{INV} + N^2Wt_{INV} \quad (S10)$$

$$E_{MVM} = \frac{p^2}{m_0n_0} (NP_{MVM}t_{MVM} + N^2Wt_{MVM}) \quad (S11)$$

$$E_{ADC} = NP_{ADC}t_{ADC} + \frac{p^2}{m_0n_0} NP_{ADC}t_{ADC} \quad (S12)$$

$$E_{DAC} = NP_{DAC}t_{INV} + \frac{p^2}{m_0n_0} NP_{DAC}t_{MVM} \quad (S13)$$

where m_0 is the matrix precision per cell, n_0 is the input vector precision per slice, W is the average power of a single RRAM device, P_{INV} and P_{MVM} are the power of OPA in INV and MVM, respectively. P_{ADC} and P_{DAC} are the power of ADC and DAC, respectively. The overall energy dissipation is calculated by $E = \frac{p}{p_0} (E_{INV} + E_{MVM} + E_{ADC} + E_{DAC})$, and the throughput and energy efficiency are estimated by $\frac{C}{t}$ and $\frac{C}{E}$, respectively.

In cases where BlockAMC is used, the solution time and energy dissipation become

$$t = \frac{p}{p_0} \left((t_{INV} + t_{ADC}) \left(\frac{N}{N_0} \right)^{\log_2 3} + (t_{MVM} + t_{ADC}) \left(3 \left(\frac{N}{N_0} \right)^2 - 2 \left(\frac{N}{N_0} \right)^{\log_2 3} \right) \right),$$

$$E = \frac{p}{p_0} \left(E_{INV,block} \left(\frac{N}{N_0} \right)^{\log_2 3} + E_{MVM,block} \left(3 \left(\frac{N}{N_0} \right)^2 - 2 \left(\frac{N}{N_0} \right)^{\log_2 3} \right) \right),$$

where N_0 is the matrix size of a single array, and

$$E_{INV,block} = N_0 P_{INV} t_{INV} + N_0 P_{ADC} t_{ADC} + N_0^2 W t_{INV} + N_0 P_{DAC} t_{INV},$$

$$E_{MVM,block} = \frac{p^2}{m_0 n_0} (N_0 P_{MVM} t_{MVM} + N_0 P_{ADC} t_{ADC} + N_0^2 W t_{MVM} + N_0 P_{DAC} t_{MVM}).$$

In benchmarking results in Fig. 5c and 5d, the following parameters are used: $p = 24$, $p_0 = 3$, $m_0 = 3$, $n_0 = 4$, $N_0 = 32$, $P_{ADC} = 133 \text{ } \mu\text{W}$, $t_{ADC} = 6.7 \text{ ns}$, $P_{DAC} = 6.2 \text{ } \mu\text{W}$ and $W = 0.4 \text{ } \mu\text{W}^{S1-S2}$. For the two scenarios considered in the main text, the computing times and power consumptions of INV and MVM circuits are given as follows:

Experimental: $t_{INV} = 120 \text{ ns}$, $t_{MVM} = 60 \text{ ns}$, and $P_{INV} = P_{MVM} = 100 \text{ } \mu\text{W}$.

Assumed: $t_{INV} = 20 \text{ ns}$, $t_{MVM} = 10 \text{ ns}$, and $P_{INV} = P_{MVM} = 100 \text{ } \mu\text{W}$.

We also assess the computing throughput and energy efficiency of the state-of-the-art digital processors, including two GPUs and one ASIC. Nvidia H100 GPU contains 14592 core unified development architecture (CUDA) cores, built with 80 billion transistors. It was designed based on the Hopper microarchitecture, fabricated with the TSMC N4 process, and launched in 2022. Its peak performance is 60 TFLOPS at FP32 precision, which implies that the throughput of one core (with 5.48 million transistors) can be normalized to 4.11 GFLOPS. Its energy efficiency is 85 GFLOPS/W. AMD Vega 20 GPU contains 4096 cores with 13.23 billion transistors. It was fabricated with the TSMC 7 nm process and launched in 2018. Its single-precision FP performance is 14.7 TFLOPS, which implies that the peak performance of one core (with 3.58 million transistors) can be normalized to 3.59 GFLOPS. An experimental test with AMD Vega 20 GPU shows that 70% of peak performance can be achieved for LINPACK [Ref. S1], which is also about solving system of linear equations. The results also show that the performance is 1.02 GFLOPS per core on double-precision computations, which roughly implies that the single-precision performance should be around 2 GFLOPS, consistent with the AMD product specifications. Its energy efficiency is 49.2 GFLOPS/W.

The ASIC chip contains 1.4 million transistors (350 thousand gate) and was fabricated with the 28 nm FDSOI process. It can be used as either downlink precoder or uplink detector. In the latter case, 148 thousand gates in the chip is involved for the 128×8 massive MIMO detection. The throughput of ASIC is estimated by dividing the number of operations of running the digital algorithm (Cholesky decomposition) by the elapsed time.

Supplementary Note 4: Area scalability of HP-INV solver.

In cases where BlockAMC is not needed, the area of LP-INV and HP-MVM are represented as Eq. S14 and Eq. S15, respectively.

$$S_{INV} = (N^2 S_0 + N S_{OPA} + N S_{ADC} + N S_{DAC}) \quad (S14)$$

$$S_{MVM} = \frac{p^2}{m_0 n_0} (N^2 S_0 + N S_{OPA} + N S_{ADC} + N S_{DAC}) \quad (S15)$$

where N is the matrix size, S_0 is the average area of a single 1T1R device, S_{OPA} is the area of an operational amplifier (OPA) for INV and MVM operations, S_{ADC} and S_{DAC} are the area of ADC and DAC, respectively, p is the target precision, m_0 is the matrix precision per cell, n_0 is the input vector precision per slice. The overall area is represented as Eq. S16.

$$S = S_{INV} + S_{MVM} = \left(\frac{p^2}{m_0 n_0} + 1 \right) S_0 N^2 + \left(\frac{p^2}{m_0 n_0} + 1 \right) (S_{OPA} + S_{ADC} + S_{DAC}) N \quad (S16)$$

In cases where BlockAMC is used, the area of LP-INV and HP-MVM are represented as Eq. S17 and Eq. S18, respectively.

$$S_{INV,block} = \left(N_0^2 S_0 \times \frac{N}{N_0} + N_0 S_{OPA} + N_0 S_{ADC} + N_0 S_{DAC} \right) \quad (S17)$$

$$S_{MVM,block} = \frac{p^2}{m_0 n_0} (N^2 S_0 + N_0 S_{OPA} + N_0 S_{ADC} + N_0 S_{DAC}) \quad (S18)$$

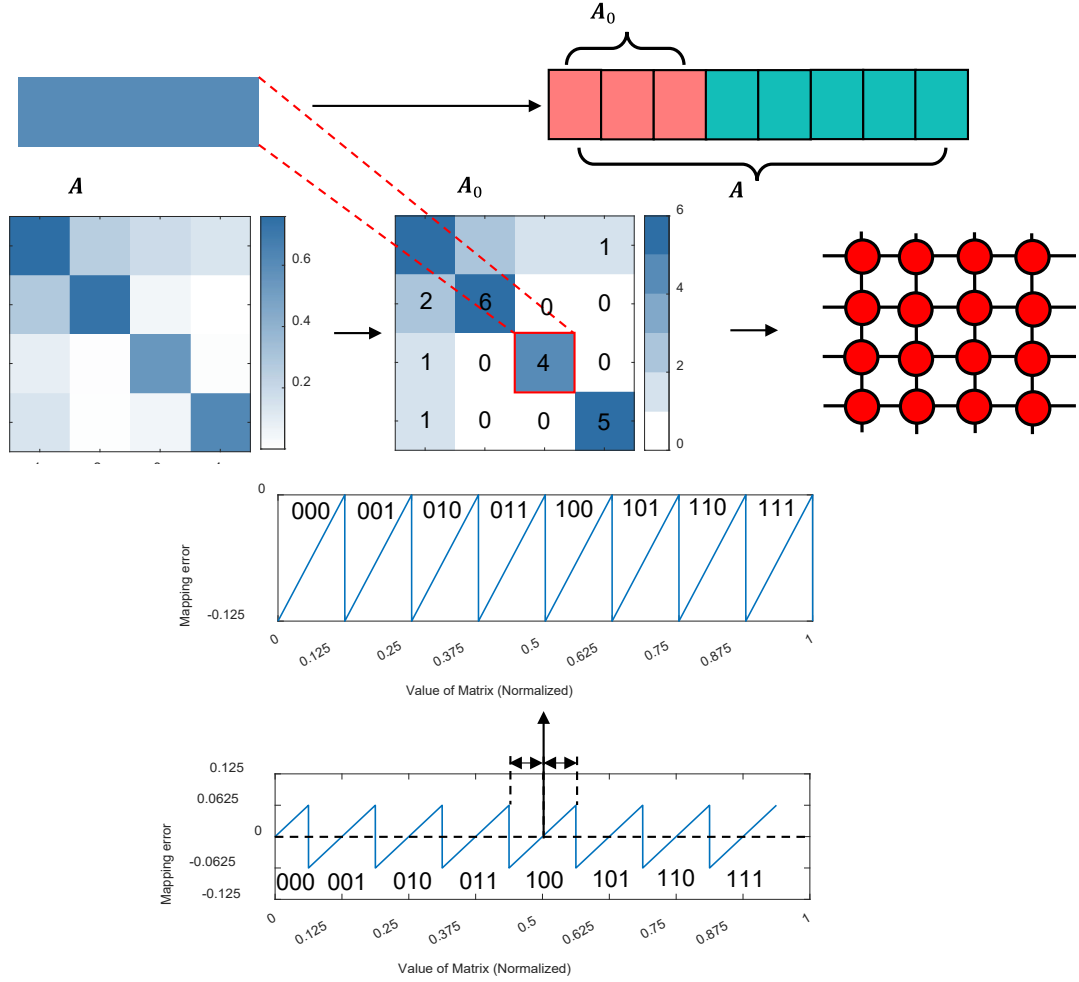
where N_0 is the matrix size of a single array. The overall area is $S_{block} = S_{INV,block} + S_{MVM,block}$, resulting in Eq. S19.

$$S_{block} = \frac{p^2}{m_0 n_0} S_0 N^2 + S_0 N_0 N + \left(\frac{p^2}{m_0 n_0} + 1 \right) (S_{OPA} + S_{ADC} + S_{DAC}) N_0 \quad (S19)$$

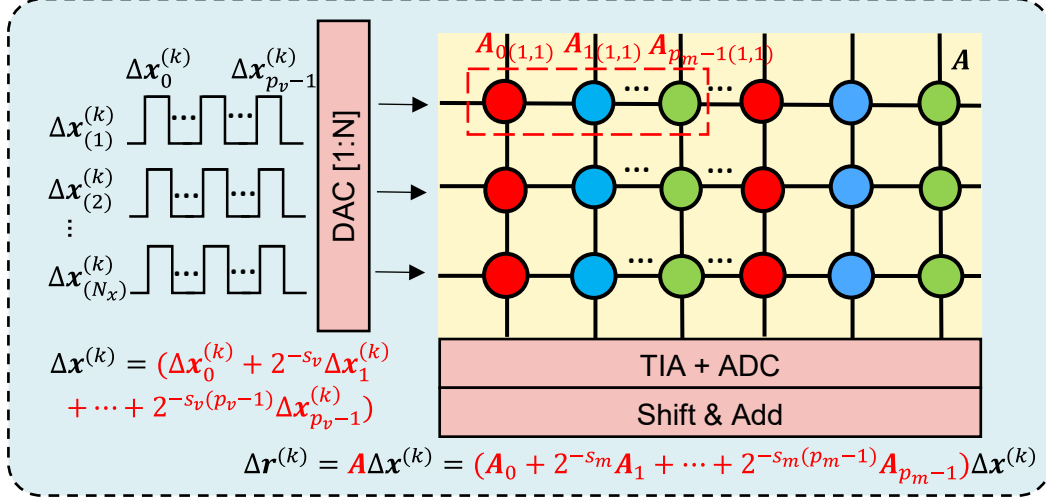
To estimate the chip area scalability, the following parameters are used: $N_0 = 32$, $S_0 = 0.36 \mu\text{m}^2$, $S_{OPA} = 76 \mu\text{m}^2$ [Ref. S2], $S_{ADC} = 62500 \mu\text{m}^2$ [Ref. S3], $S_{DAC} = 144 \mu\text{m}^2$ [Ref. S4]. The references are consistent with those used in the throughput and energy efficiency evaluation section.

On the other hand, the area cost by OPA grows linearly with GBWP. Here we also make an estimate about this area cost.

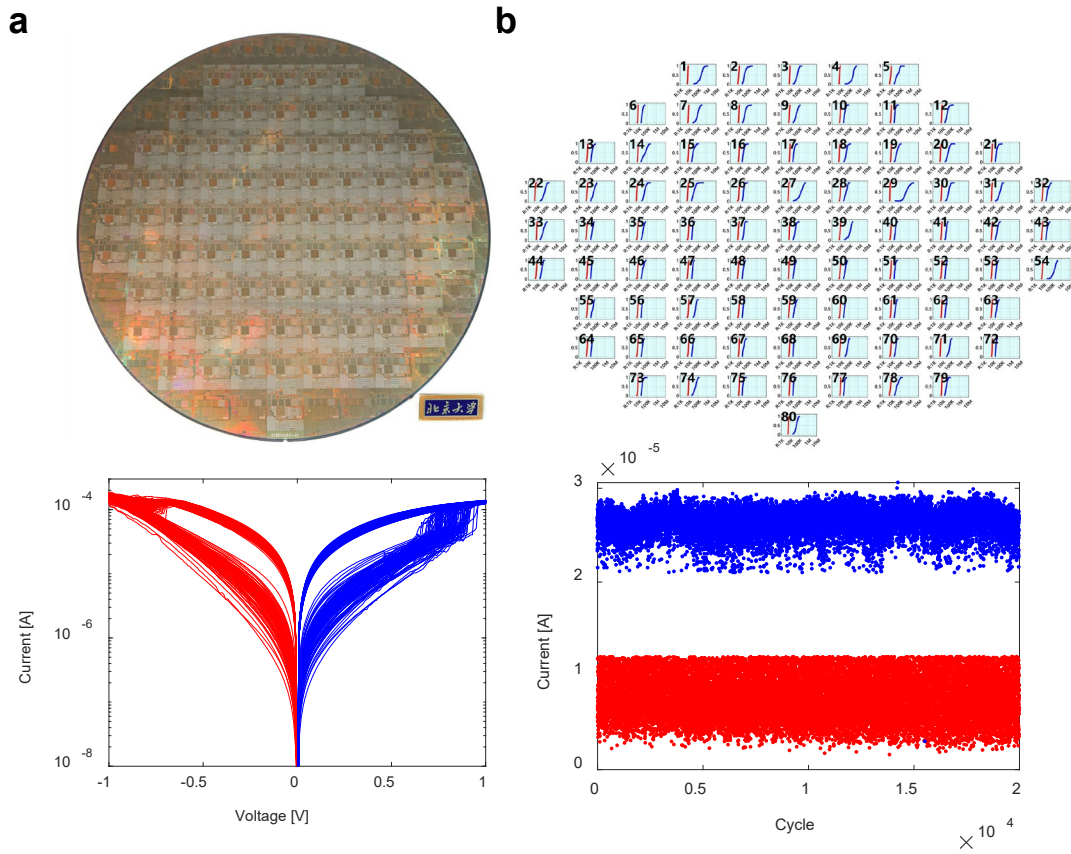
GBWP is proportional to the transistor transconductance, which, in turn, scales with $\sqrt{\frac{W}{L}} I_D$. Since $I_D \propto \frac{W}{L}$, there is a linear relationship between the area of OPA and GBWP. Therefore, the area of the OPA exhibits a linear relationship with GBWP. The area of the OPA used in experiment can be estimated as $S_{OPA,exp} = 76 \text{ } \mu\text{m}^2$. Since we assumed the speed of INV and MVM circuits can be optimized by 6 $\times$, the area of assumed OPA is $S_{OPA,asm} = 456 \text{ } \mu\text{m}^2$. By substituting these two parameters in Eq. S16 and Eq. S19 for AMC and BlockAMC, respectively, the scaling behaviors of the chip area are shown in Supplementary Fig. 18a. The two sets of curves (experimental vs assumed) evidence that the increasement of OPA area is almost invisible in the overall area. The increasement of OPA area cost can only be observed when S_{ADC} and S_{DAC} are excluded, as shown in Supplementary Fig. 18b. As matrix size increases, however, such an increasement declines due to the dominance of RRAM array area at large scale.



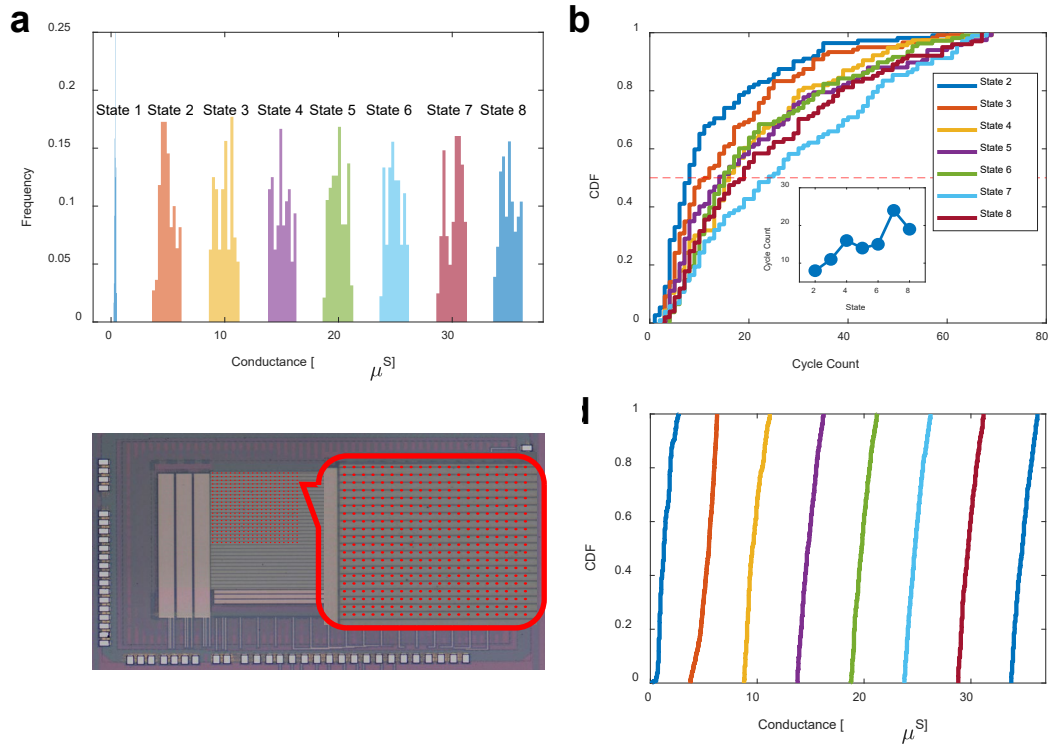
Supplementary Fig. 1. Mapping method for HP-INV. **a**, Quantization process. A_0 and A are programmed to INV and MVM circuit, respectively, and the remaining 9 bits are programmed to $A_1 \sim A_3$. **b**, The direct truncation method in MVM circuit. The error caused by the most significant bit (MSB) is compensated by the least significant bit (LSB), according to the bit slicing method. **c**, The round truncation method in INV circuit. As the truncation method may cause significant error to the mapping result and hence the INV result, the round truncation is used instead.



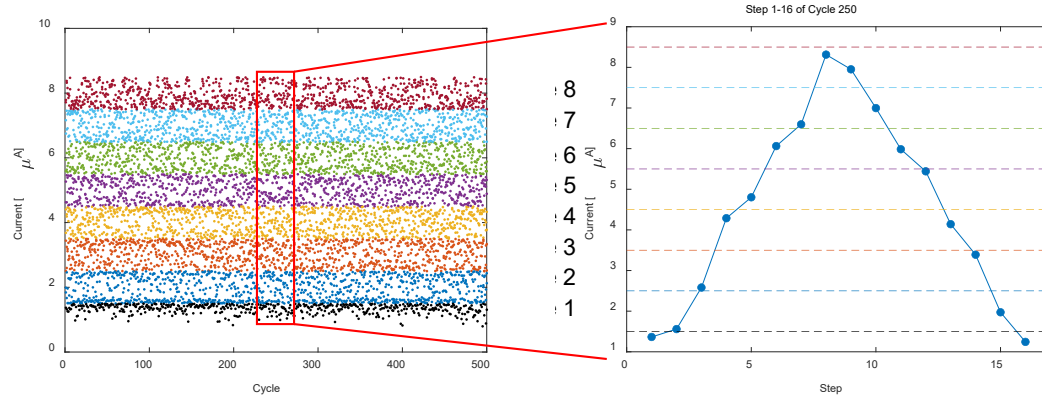
Supplementary Fig. 2. Bit slicing for HP-MVM. To compute $\Delta \mathbf{r}^{(k)} = A\Delta \mathbf{x}^{(k)}$, matrix A and vector $\Delta \mathbf{x}^{(k)} = [\Delta x_{(1)}^{(k)}, \dots, \Delta x_{(N_x)}^{(k)}]$ are split into slices with different weights, where N_x is the dimension of vector $\Delta \mathbf{x}^{(k)}$. For example, an $s_m p_m$ -bit matrix is split to p_m slice matrices ($A_i, i = 0, 1, \dots, p_m - 1$), where $A_{i(u,w)}$ is the u -th row and the w -th column of A_i . The s_m -bit data of each slice matrix are programmed in RRAM devices. In our case, $s_m = 3$. Similarly, an $s_v p_v$ -bit vector is split into p_v slice vectors ($\Delta x_j^{(k)}, j = 0, 1, \dots, p_v - 1$), each of which has s_v bits of data. For signed operations, the matrix and vector should additionally be split into positive and negative parts. After the analog MVM operations, outputs with different weights are accumulated in the shift-and-add module.



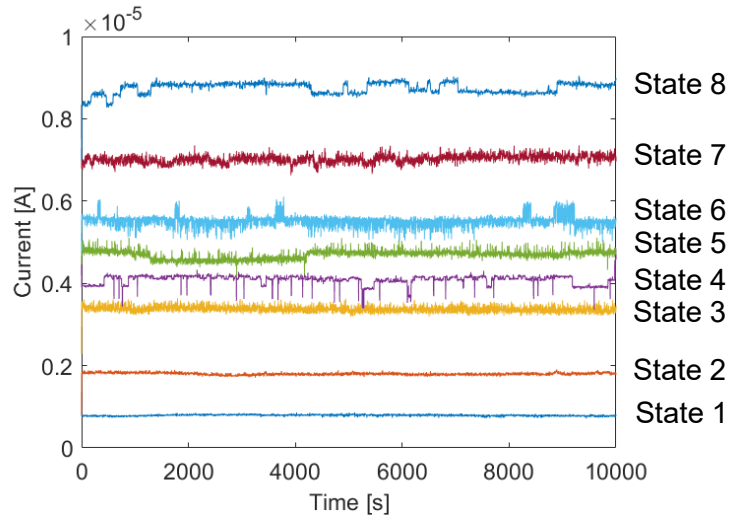
Supplementary Fig. 3. Binary switching characteristics of wafer-level RRAM. a, Photo of the 12-inch RRAM chip. **b**, CDF of high resistance state (HRS) and low resistance state (LRS) from 80 Process Control Monitor (PCM) dies across the wafer. **c**, I-V characteristics of RRAM device. 100 cycles of results are tested. **d**, Endurance behavior of binary states of RRAM device.



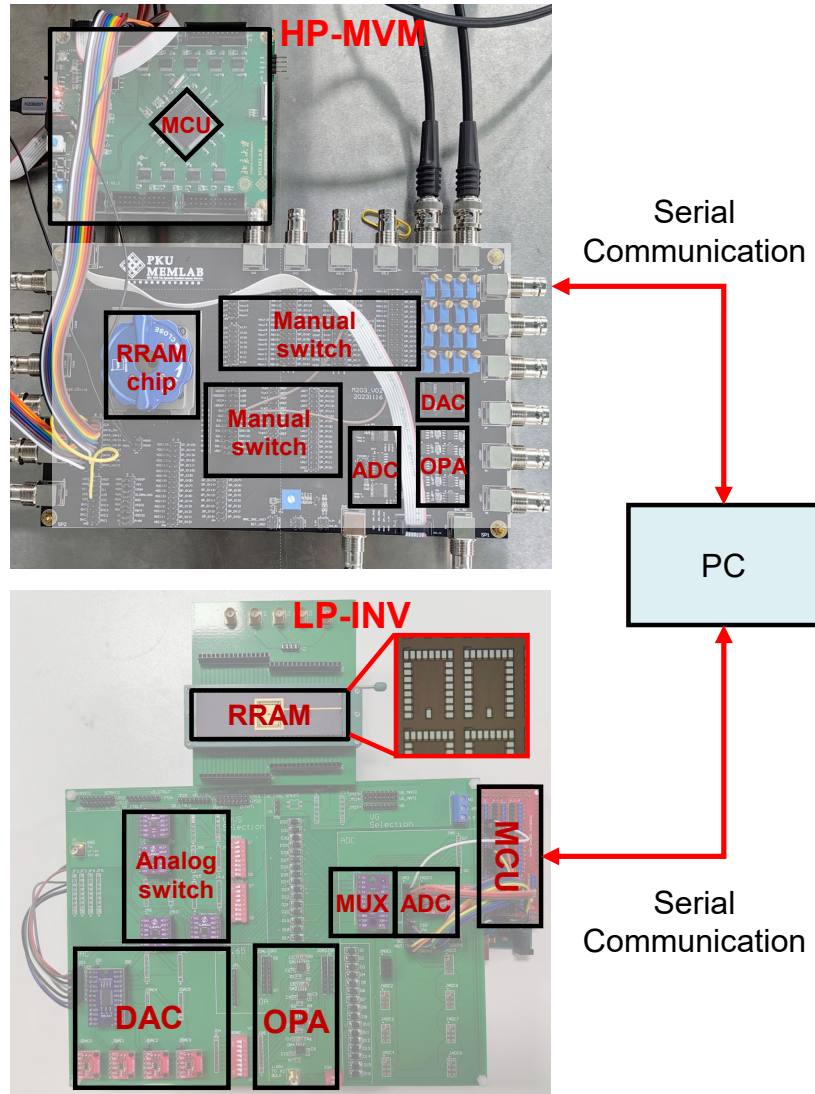
Supplementary Fig. 4. 3-bit RRAM. **a**, 8 conductance levels with distributions of RRAM, which are achieved with the write-verify method. **b**, Number of cycles required to program devices from state 1 to other states with the probability of 50%. **c**, 400 cells in the chip selected for testing yield of 3-bit programming of the 1Mb chip, marked with red dots. **d**, CDF plot of 8 conductance levels of the 400 tested devices.



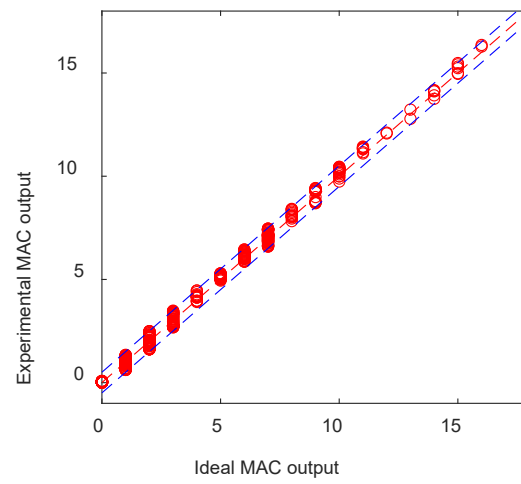
Supplementary Fig. 5. Endurance performance of 3-bit RRAM. The device is tested for 500 cycles, and each cycle includes 16 programming operations, namely from state 1 to state 8, and backwards from state 8 to state 1.



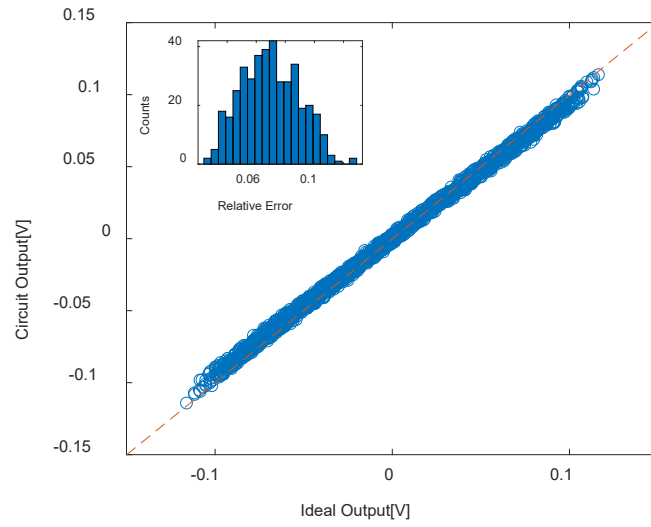
Supplementary Fig. 6. Retention performance of 3-bit RRAM. The 8 states of RRAM were tested for 10000 seconds.



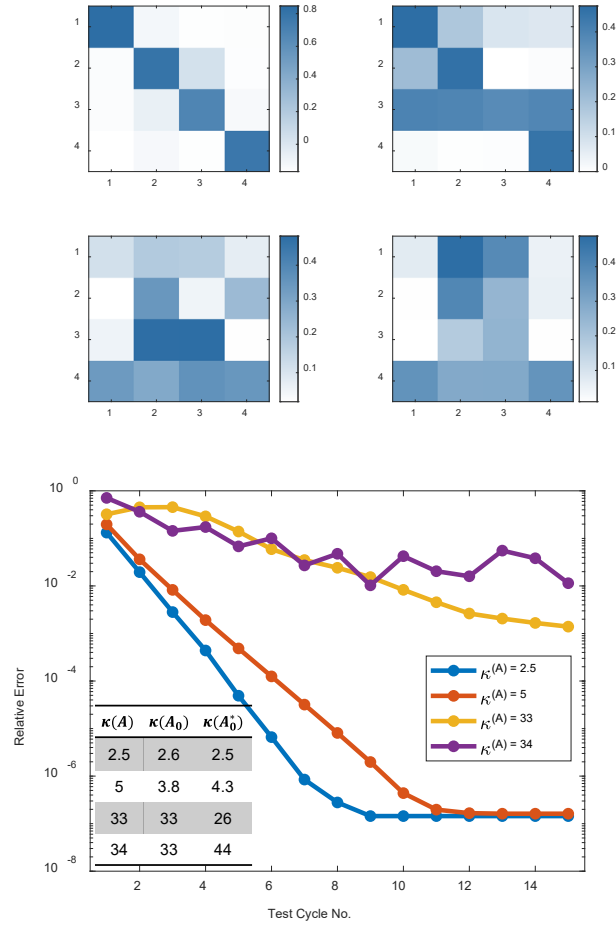
Supplementary Fig. 7. HP-INV solver system, consisting of LP-INV and HP-MVM experimental setups that are controlled by a personal computer (PC). The HP-MVM experiment PCB consists of a fully integrated chip, manual switches for changing the operation mode, such as on-chip mode, on-board mode (using DACs, ADCs and OPAs on the PCB for device programming and MVM), and off-board mode (using off-board instruments such as Keysight B1500A). The LP-INV experiment PCB consists of 8×8 RRAM arrays, OPAs, analog switches, MUX, DACs and ADCs. In the future, the entire setup will be integrated in a system-on-chip (SoC). It would consist of a digital unit and an analog unit. The analog unit account for LP-INV and MVM operations with RRAM arrays. The digital unit include a controller for selecting arrays and defining matrix operations, and a processor for operations other than INV and MVM, such as vector addition and subtraction. The controller would interact with all computing units through the exchange of instructions and input/output data.



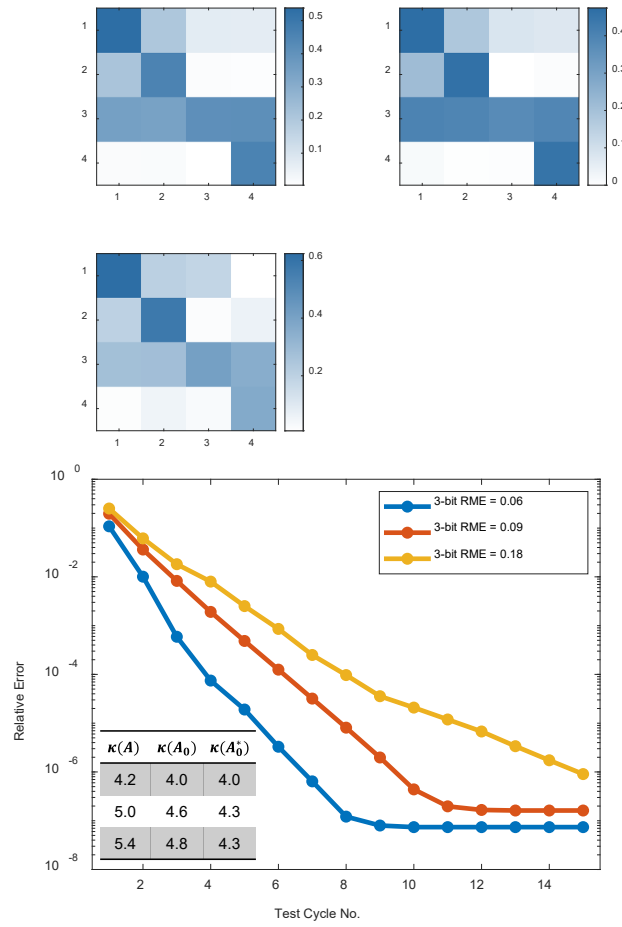
Supplementary Fig. 8. HP-MVM results in Fig. 2e. The red line indicates the ideal result, while the 2 blue line defines the boundary for accurate Multiply-Accumulate (MAC) output. No errors occur in 1152 MACs.



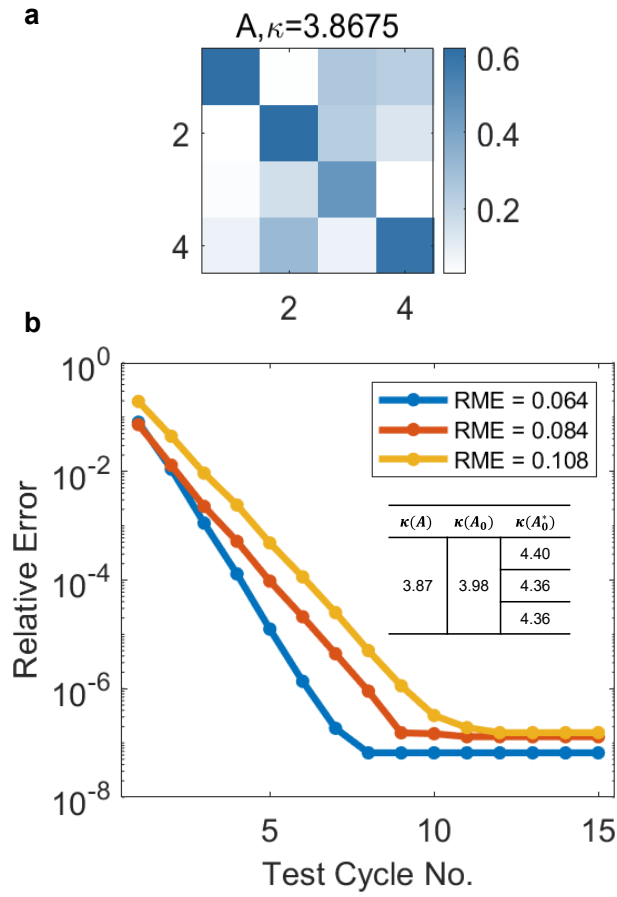
Supplementary Fig. 9. LP-INV results of a better conditioned matrix. The matrix size is 4×4 , and the condition number is 1.6. 500 experiments were performed.



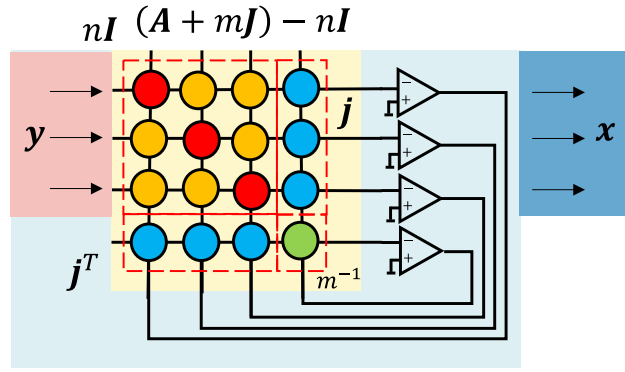
Supplementary Fig. 10. Impact of condition number on the convergence of HP-INV. a-d, The matrices with different condition numbers, namely 2.5, 5, 33, and 34. **e,** Convergence behavior of HP-INV, indicating that the convergence rate decreases as condition number increases. For each case, 15 cycles of HP-INV iterations were performed. In the inset, also the condition numbers of the most significant slice matrix A_0 are included, suggesting only slight changes.



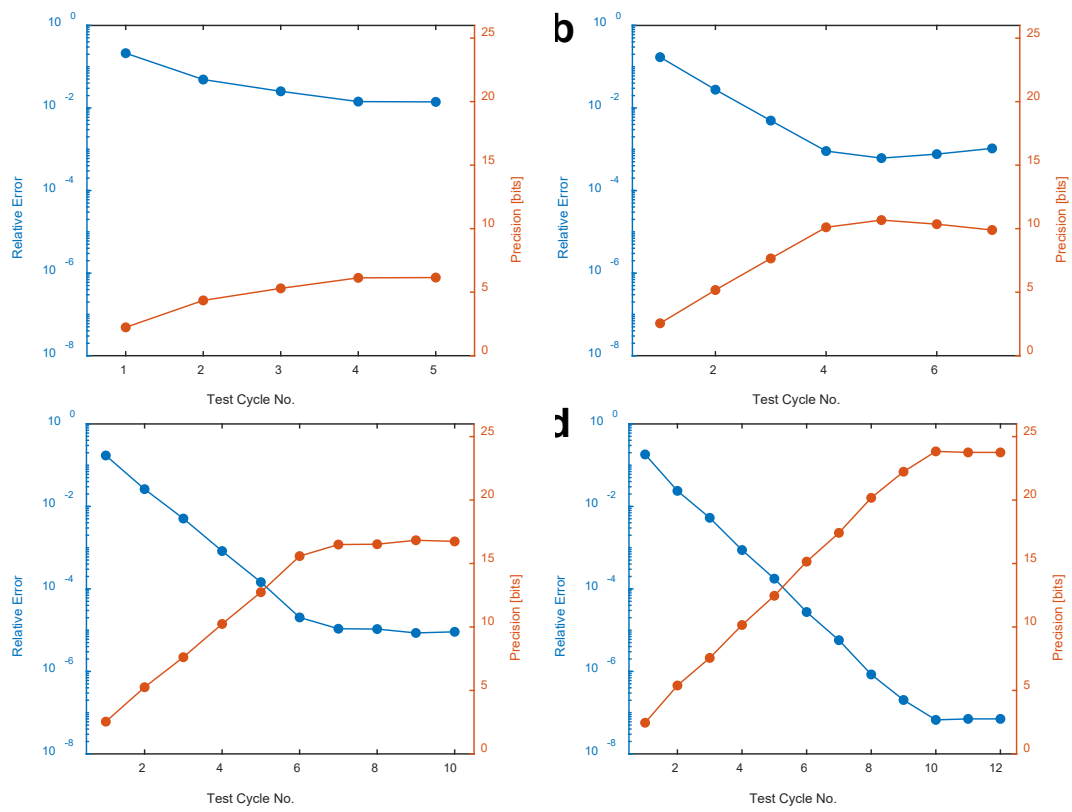
Supplementary Fig. 11. Impact of RME on the convergence of HP-INV. **a-c**, The matrices with similar condition numbers for study. **d**, Convergence behavior of HP-INV, indicating that the convergence rate decreases as RME increases. For each case, 15 cycles of HP-INV iterations were performed. In the inset, also the condition numbers of A_0 are included, suggesting only minor changes.



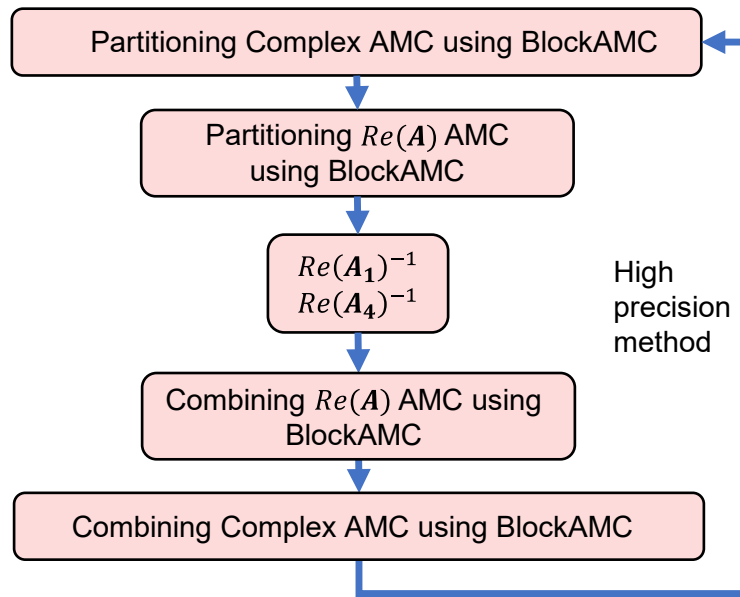
Supplementary Fig. 12. Impact of RME on the convergence of HP-INV using the same target matrix. a, The considered matrix A in the experiment. **b,** Convergence behavior of HP-INV, showing that the convergence rate decreases as the RME increases. 15 cycles of HP-INV iterations were performed for each case. The inset also presents the condition numbers of A_0 .



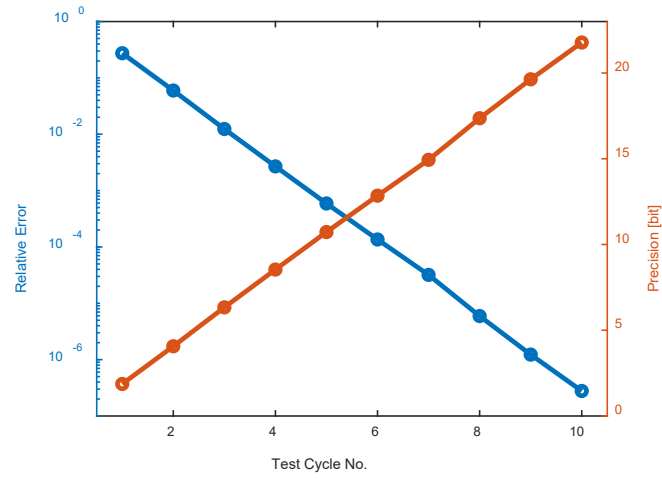
Supplementary Fig. 13. The real-valued LP-INV circuit implemented with the bias-column method. The matrix mJ provides negative bias, and the matrix nI provide diagonal bias. To avoid applying high voltage to the device, more than one column or row of bias matrix can be proposed, which means that J and m can be matrices.



Supplementary Fig. 14. Precision limits in HP-INV. The saturated precision of HP-INV with matrices quantized as **a**, 6 bits, **b**, 12 bits, **c**, 18 bits, and **d**, 24 bits.



Supplementary Fig. 15. HP-INV scheme with two-stage BlockAMC for complex-valued matrix A .

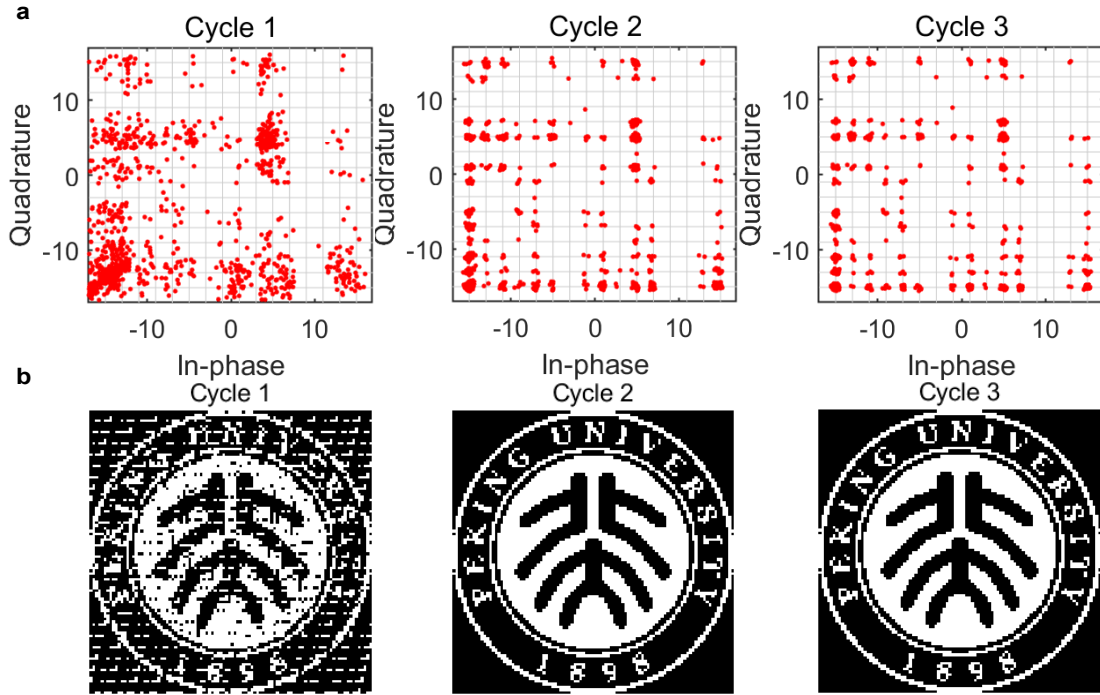


Supplementary Fig. 16. Average convergence results of 16 HP-INV operations for computing the inverse of the 8×8 complex-valued matrix in Fig. 3f.

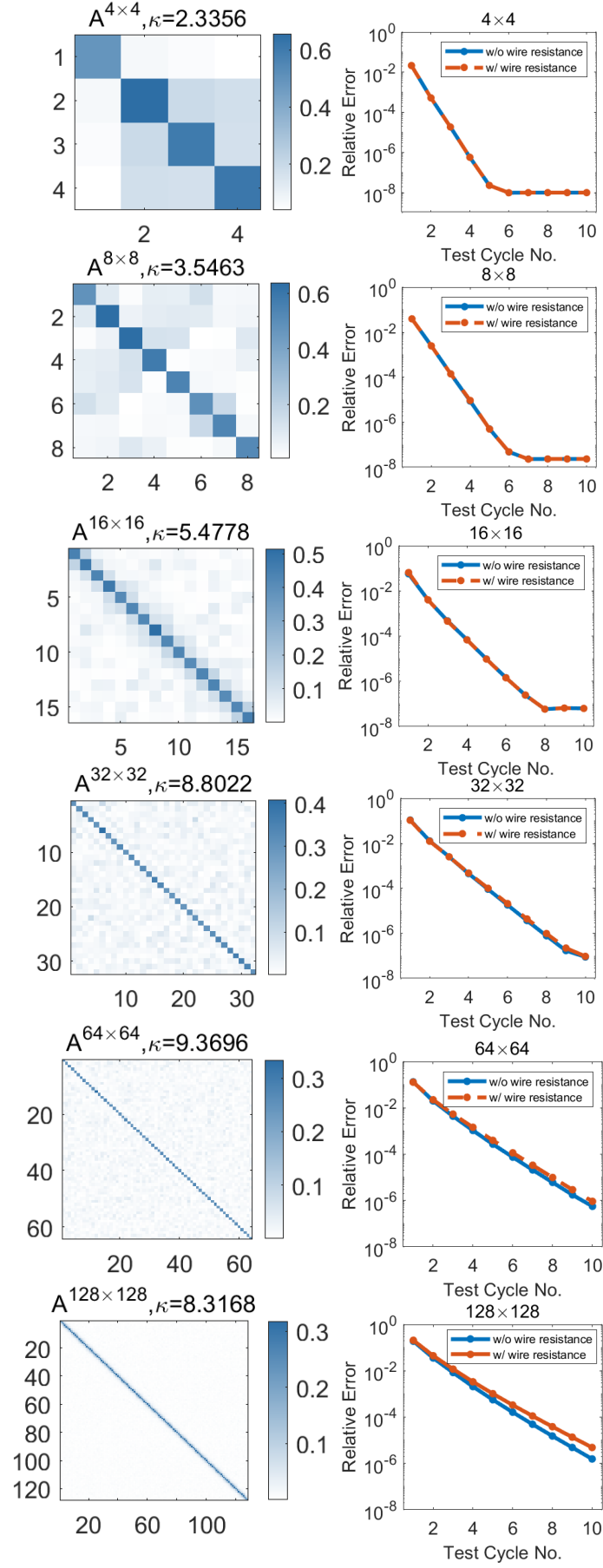
I/Q	Gray Code
-15	0
-13	1
-11	3
-9	2
-7	6
-5	7
-3	5
-1	4
1	12
3	13
5	15
7	14
9	10
11	11
13	9
15	8

I: In-phase Q: Quadrature

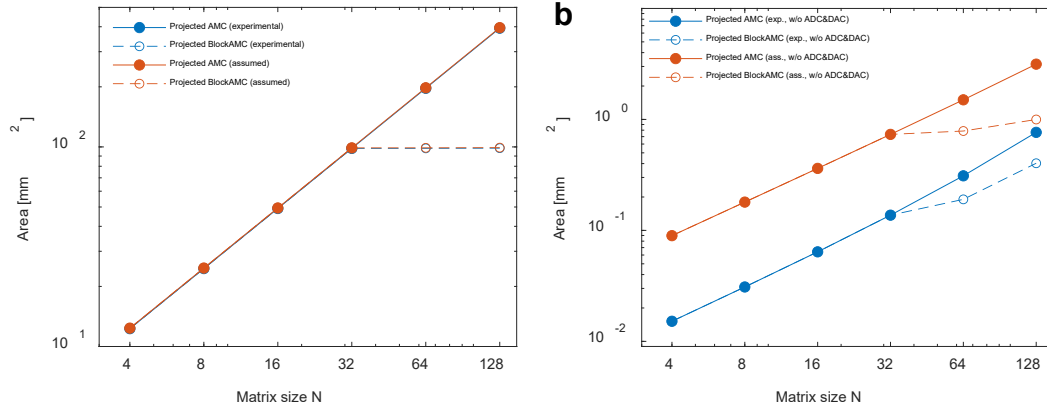
Supplementary Fig. 17. Gray coding rule in massive MIMO. The 8-bit number transmitted by each UE is split to two parts. The first 4-bit is the real-valued part, while the last 4-bit is the imaginary-valued part. Each part of the 4-bit number (Gray Code: 0-16) is mapped to the In-phase part (real-valued part) and Quadrature part (imaginary part).



Supplementary Fig. 18. Processing results of image transmission by ZF detection with HP-INV in massive MIMO. **a**, Constellation diagram results of 256-QAM symbols recovered by ZF detection, during 3 cycles of HP-INV. As the image size is 100×100 , there are 1250 points in each diagram. **b**, The recovered images by ZF detection, corresponding to the data in **a**.



Supplementary Fig. 19. Convergence behavior of HP-INV with and without wire resistance. Matrices of various sizes, ranging from 4×4 to 128×128 , were used for the study. Each matrix was randomly generated in the same manner as other matrices in this work. The condition number of each matrix is also provided, showing an increasing trend with matrix size. Despite the growth in matrix size and its condition number, the convergence behavior of HP-INV remains nearly identical with or without wire resistance, suggesting that wire resistance has a minimal impact on the convergence rate.



Supplementary Fig. 20. Scaling behaviors of the chip area of HP-INV solver. a, Results considering the overall area, including RRAM arrays, OPAs, DACs, and ADCs. **b,** Results considering only the area of RRAM arrays and OPAs.

Supplementary Table 1. Summary of 4×4 numerical matrices utilized in experiment.

Here the first 8 decimal places of matrices elements are displayed.

Matrices	Positions	κ
$\begin{bmatrix} 0.39793092 & 0.21119161 & 0.00814822 & 0.00489785 \\ 0.02482829 & 0.27987521 & 0.51724488 & 0.01312138 \\ 0.12348390 & 0.21195397 & 0.71211475 & 0.11480659 \\ 0.02361987 & 0.12717058 & 0.00505734 & 0.18088216 \end{bmatrix}$	Fig. 2d	7.7
$\begin{bmatrix} 2.00019916 & -0.26083176 & -0.38179648 & -0.36178361 \\ -0.34451004 & 1.89337106 & 0.09642510 & -0.37092546 \\ -0.35069423 & -0.15410866 & 1.60791353 & -0.30456058 \\ -0.39505508 & -0.29172644 & -0.37851576 & 1.83986141 \end{bmatrix}$	Fig. 3a Fig. S10a	2.5
$\begin{bmatrix} 1.10507812 & 0.109375 & 0.00546875 & 0.00332031 \\ 0.109375 & 1.22460937 & 0.01914062 & 0.20585937 \\ 0.00546875 & 0.01914062 & 0.99277343 & 0.10898437 \\ 0.00332031 & 0.20585937 & 0.10898437 & 1.05058593 \end{bmatrix}$	Fig. 3d $Re(A)$	1.7
$\begin{bmatrix} 1.15625 & 0.20097656 & -0.00078125 & 0.00351562 \\ 0.20097656 & 1.30175781 & 0.10292968 & 0.16679687 \\ -0.00078125 & 0.10292968 & 1.053125 & 0.1 \\ 0.00351562 & 0.16679687 & 0.1 & 1.08535156 \end{bmatrix}$	Fig. 3f $Re(A_1)$	1.6
$\begin{bmatrix} 1.29980468 & 0.12714843 & 0.06523437 & 0.00234375 \\ 0.12714843 & 1.32636718 & 0.23691406 & 0.07890625 \\ 0.06523437 & 0.23691406 & 0.92636718 & 0.10117187 \\ 0.00234375 & 0.07890625 & 0.10117187 & 1.159375 \end{bmatrix}$	Fig. 3f $Re(A_4)$	1.9
$\begin{bmatrix} 1.15 & 0.13 & 0 & 0.14 \\ 0.12 & 1.21 & 0.19 & 0.1 \\ 0.01 & 0.24 & 1.01 & 0.01 \\ 0.09 & 0.06 & 0 & 1.02 \end{bmatrix}$	Fig. S9	1.7
$\begin{bmatrix} 0.47408346 & 0.18173199 & 0.08428150 & 0.07638011 \\ 0.21597135 & 0.46091447 & 0 & 0.00790139 \\ 0.40297094 & 0.39506955 & 0.37663297 & 0.39243575 \\ 0.01580278 & 0.00263379 & 0.00526759 & 0.45301308 \end{bmatrix}$	Fig. S10b Fig. S11b	5
$\begin{bmatrix} 0.11826965 & 0.18699862 & 0.18164916 & 0.07652058 \\ 0.01918829 & 0.34841286 & 0.05361092 & 0.23619043 \\ 0.05523902 & 0.47738144 & 0.48121910 & 0.02116526 \\ 0.33945833 & 0.29235979 & 0.36457754 & 0.34887803 \end{bmatrix}$	Fig. S10c	33
$\begin{bmatrix} 0.07451536 & 0.48845533 & 0.39393472 & 0.05600517 \\ 0.01079760 & 0.40544426 & 0.25131137 & 0.05802231 \\ 0.00972971 & 0.18023700 & 0.25700681 & 0.01115357 \\ 0.36581348 & 0.29153543 & 0.29307794 & 0.36166055 \end{bmatrix}$	Fig. S10d	34
$\begin{bmatrix} 0.53791269 & 0.20976619 & 0.07295070 & 0.07071214 \\ 0.21898379 & 0.45837799 & 0.01158783 & 0.00882255 \\ 0.35290233 & 0.34513321 & 0.41268504 & 0.41452856 \\ 0.01593327 & 0.01830351 & 0.00342367 & 0.45903639 \end{bmatrix}$	Fig. S11a	4.2
$\begin{bmatrix} 0.62363648 & 0.20669022 & 0.18555000 & 0.00942395 \\ 0.20045003 & 0.57409705 & 0.02152227 & 0.06393006 \\ 0.27558696 & 0.28106304 & 0.41159753 & 0.35619997 \\ 0.01731969 & 0.05628902 & 0.02852656 & 0.36906239 \end{bmatrix}$	Fig. S11c	5.4
$\begin{bmatrix} 1.52 & 0.08 & 0.63 & 0.56 \\ 0.07 & 1.54 & 0.57 & 0.33 \\ 0.1 & 0.39 & 1.14 & 0.07 \\ 0.2 & 0.77 & 0.2 & 1.48 \end{bmatrix}$	Fig. S12a	3.9

Supplementary References

1. Simakov, N. A. et al. First Impressions of the NVIDIA Grace CPU Superchip and NVIDIA Grace Hopper Superchip for Scientific Workloads. in *Proceedings of the International Conference on High Performance Computing in Asia-Pacific Region Workshops*. 36-44 (2024).
2. Luo, Y. et al. Smaller, Faster, Lower-Power Analog RRAM Matrix Computing Circuits Without Performance Compromise. *SCI. CHINA Inform. Sci.* **68**, 122402 (2025).
3. Van der Plas, G. et al. A 150MS/s 133 μ W 7b ADC in 90nm digital CMOS Using a Comparator-Based Asynchronous Binary-Search sub-ADC. in *2008 IEEE International Solid-State Circuits Conference (ISSCC)*. (IEEE, 2008).
4. Lee, J. et al. A 17.5-fJ/bit Energy-Efficient Analog SRAM for Mixed-Signal Processing. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **25**, 2714 - 2723 (2017).