Scalable Matrix Architecture (Integrated Facility Variant)

José Moreira
IBM Corporation

Summary (look at single-precision floating-point first)

- Let N be the size in (32-bit) words of a scalable vector:
  - $N = 4$: 128-bit vectors
  - $N = 8$: 256-bit vectors
  - $N = 16$: 512-bit vectors
- The architecture defines M accumulator registers (the M is fixed for a version)
  - Each accumulator is organized as N rows of N columns each – each element is a word
  - $\mathrm{ACC}[\mathrm{AT}][i][j]$ represents the word at row i, column j of accumulator AT
  - $\mathrm{ACC}[0{:}M-1][0{:}N-1][0{:}N-1]$ represents the space of words in accumulator file
- Outer-product operations are of the form

$$A\langle m_x, m_y\rangle = \pm xy^{\mathrm{T}} \pm A$$

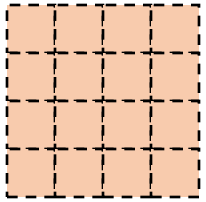
$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} = \pm \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \begin{bmatrix} y_1 & \cdots & y_n \end{bmatrix} \pm \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$$

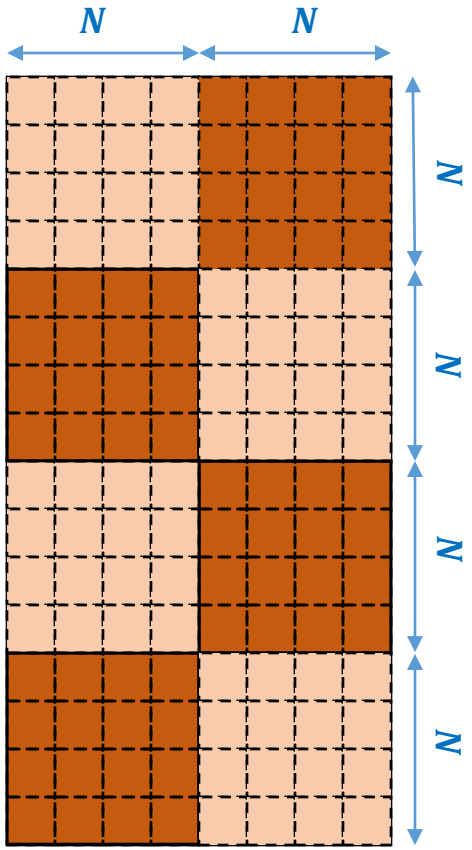
where

- A is an accumulator register ($\mathrm{ACC}[i \in [0, M)]$)
- x is an N-element column vector, contained in a scalable vector register
- y^{T} is an N-element row vector, contained in a scalable vector register
- m_x is a mask for the rows of the accumulator, contained in a vector (mask) register
- m_y is a mask for the columns of the accumulator, contained in a vector (mask) register

Example for $M = 8$ (8 accumulator registers)

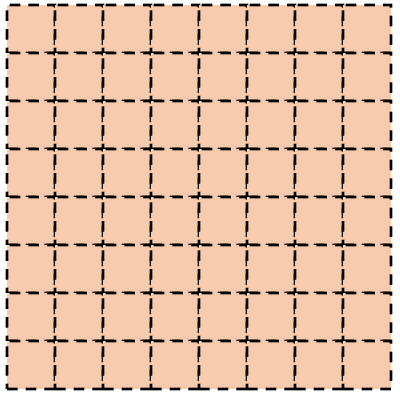
$N = 4$ (128-bit/16-byte scalable vectors)

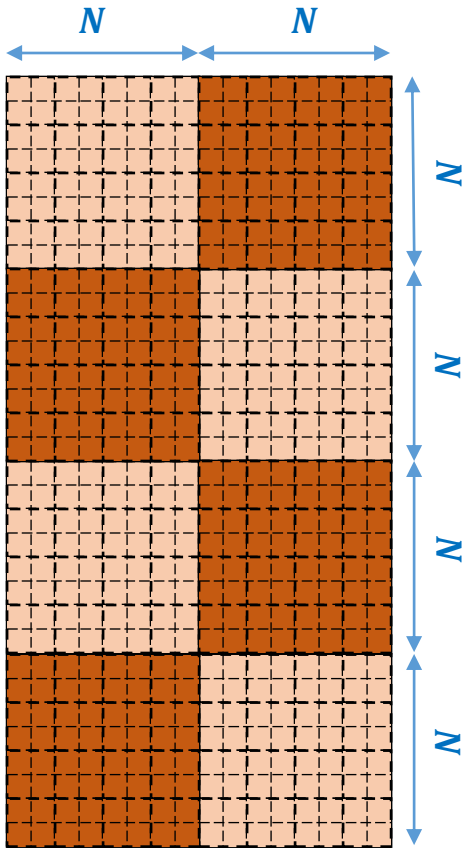




feature	value
each accumulator	4×4
accumulator file	16×8

$N = 8$ (256-bit/32-byte scalable vectors)





feature	value
each accumulator	8×8
accumulator file	32×16

SGEMM middle kernel ($4N \times 2N$ result panels)

$$A = XY^{\mathrm{T}}$$

$$\begin{bmatrix} [4N \times 2N] & \cdots & [4N \times 2N] \\ \vdots & \ddots & \vdots \\ [4N \times 2N] & \cdots & [4N \times 2N] \end{bmatrix} = \begin{bmatrix} [4N \times K] \\ \vdots \\ [4N \times K] \end{bmatrix} \times [[K \times 2N] \quad \cdots \quad [K \times 2N]]$$

$$[4N \times K] = \begin{bmatrix} X[0]+0 & X[1]+0 & \cdots \\ X[0]+1 & X[1]+1 & \cdots \\ \vdots & \vdots & \cdots \\ X[0]+N & X[1]+N & \cdots \\ \vdots & \vdots & \cdots \\ X[0]+2N & X[1]+2N & \cdots \\ \vdots & \vdots & \cdots \\ X[0]+3N & X[1]+3N & \cdots \\ \vdots & \vdots & \cdots \end{bmatrix}$$

$$[K \times 2N] = \begin{bmatrix} Y[0]+0 & Y[0]+1 & \cdots & Y[0]+N & \cdots \\ Y[1]+0 & Y[1]+1 & \cdots & Y[1]+N & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

SGEMM inner-most kernel ($4N \times 2N$ panel, $M = 8$)

```
void sgemm_kernel(X,Y,m_X[0:3],m_Y[0:1],N,K) {

    …

    vector<float> u[0:1], v[0:3];
    for (k=0; k<K; k++) {
        v[0]⟨m_X[0]⟩ ← X[k]; v[1]⟨m_X[1]⟩ ← X[k] + N; v[2]⟨m_X[2]⟩ ← X[k] + 2N; v[3]⟨m_X[3]⟩ ← X[k] + 3N;
        u[0]⟨m_Y[0]⟩ ← Y[k]; u[1]⟨m_Y[1]⟩ ← Y[k] + N;
        A[0]⟨m_X[0], m_Y[0]⟩ ← v[0] × u^T[0] + A[0];
        A[1]⟨m_X[0], m_Y[1]⟩ ← v[0] × u^T[1] + A[1];
        A[2]⟨m_X[1], m_Y[0]⟩ ← v[1] × u^T[0] + A[2];
        A[3]⟨m_X[1], m_Y[1]⟩ ← v[1] × u^T[1] + A[3];
        A[4]⟨m_X[2], m_Y[0]⟩ ← v[2] × u^T[0] + A[4];
        A[5]⟨m_X[2], m_Y[1]⟩ ← v[2] × u^T[1] + A[5];
        A[6]⟨m_X[3], m_Y[0]⟩ ← v[3] × u^T[0] + A[6];
        A[7]⟨m_X[3], m_Y[1]⟩ ← v[3] × u^T[1] + A[7];
    }

    …

}
```