
Matrix
Multiplication

Matrix Multiplication

$$\begin{bmatrix} a00 & a01 \\ a10 & a11 \end{bmatrix} * \begin{bmatrix} b00 & b01 \\ b10 & b11 \end{bmatrix}$$

$$c00 = a00 * b00 + a10 * b01$$

$$c10 = a10 * b00 + a11 * b10$$

For matrix size *L*:

$$cmn = a_{m0} * b_{0n} + a_{m1} * a_{1n} + \ldots\ldots\ldots\ldots\ldots\ldots + a_{mL-1} * b_{L-1}$$

$$\begin{bmatrix} a00 & a01 \\ a10 & a11 \end{bmatrix} * \begin{bmatrix} b00 & b01 \\ b10 & b11 \end{bmatrix}$$

Matrix Multiplication

- For *LxL* matrix to multiply using naïve multiplication it would require:
  - L^3 multiplications and $L^2*(L-1)$ Additions
  - *Many algorithms are used to minimize and reduce the computation needed for matrices.*

Assignment

- Write UC/Arduino Code that implements Matrix Multiplication
- Tracking your instructions and UC datasheet try to predict how many cycles will the program take
- Make the code modular to allow the ability of increasing matrix size later on
- Optimize your code for least time to compute as possible.

- The winner of best implementation will have a CCs gift card ;)

Strassen Algorithm (One way to accelerate Matrix Multiplication)

$$M1 = (A00 + A11)(B00 + B11)$$

$$M2 = (A10 + A11) * B00$$

$$M3 = A00 * (B01 - B11)$$

$$M4 = A11 * (B10 + B00)$$

$$M5 = (A00 + A01) * B11$$

$$M6 = (A10 - A00)(B00 + B01)$$

$$M7 = (A01 - A11)(B10 + B11)$$

$$C00 = M1 + M4 - M5 + M7$$

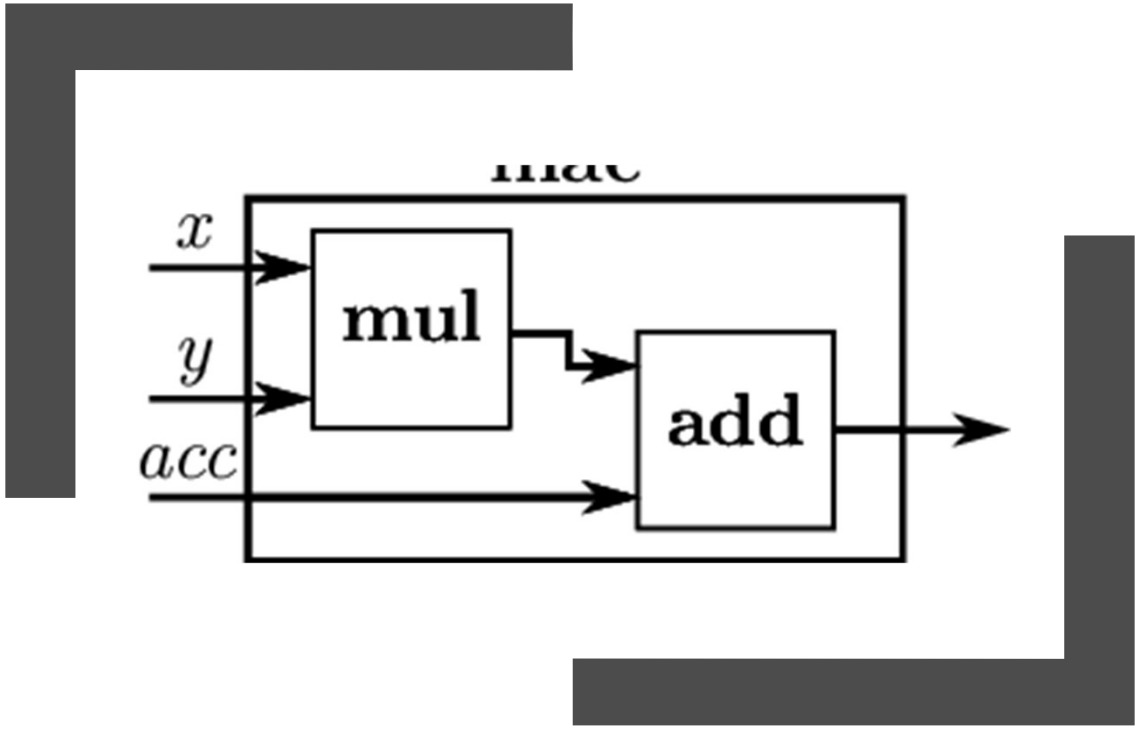
$$C01 = M3 + M5$$

$$C10 = M2 + M4$$

$$C11 = M1 - M2 + M3 + M6$$

Speed α $O(L^{2.8})$

Verilog Assignment



- Design Multiply Accumulate block whose block diagram looks like the following.
- Each parameter is an 8-bit Block

Verilog Assignment

- Test pattern
  - A = 1; B = 10
  - A = 1; B = 100
  - A = 1; B = 255
  - A = 5; B = 10
  - A = 5; B = 100
  - A = 5; B = 255
  - A = 100; B = 100
  - A = 100; B = 255
  - A = 255; B = 255