Project 1

MAC design: V1

Design a **sequential multiply–accumulate (MAC) circuit** in Verilog that takes two unsigned 8-bit input values **A** and **B** every clock cycle, multiplies them, and adds the result to an internal accumulator register named `ACC`. The value of `ACC` must be available on the output port `Y` at every clock cycle.

- Ensure a minimum number of product accumulation: 50 per run

Step 1 — Maximum product value

$$P_{\max} = 255 \times 255 = 65025$$

Step 2 — Maximum accumulated value

If you sum **50** such products starting from zero:

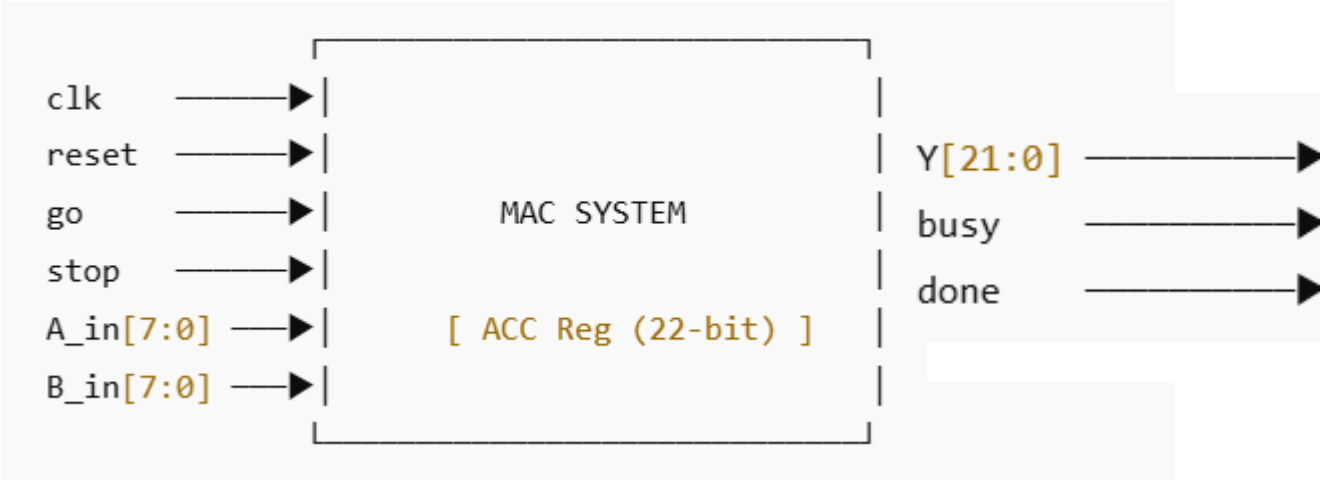
$$\text{Max Sum} = 50 \times 65025 = 3{,}251{,}250$$

Step 3 — Required accumulator width

We need $2^N - 1 \geq \text{Max Sum}$.

$$\boxed{N = 22 \text{ bits}}$$

MAC design: V2



Design a **Multiply–Accumulate (MAC) system** that:

- **Inputs**
  - `clk` : clock (period = 2 s)
  - `reset` : **asynchronous**, active-high (clears all state immediately)
  - `go` : **synchronous** start trigger (sampled on rising edge of `clk`)
  - `stop` : **synchronous** early-stop trigger (sampled on rising edge of `clk`)
  - `A_in[7:0]`, `B_in[7:0]` : 8-bit unsigned operands (sampled each clock while accumulating)
- **Outputs**
  - `Y[21:0]` : final accumulated value (from 22-bit accumulator register `ACC`)
  - `busy` : 1 while accumulation is in progress
  - `done` : 1 when accumulation completes (50 terms or early `stop`), held until next `go` or `reset`