Assumptions

1. Input Data Assumptions

1. **Input Format**:
    - Input test vectors are generated from the mayo-c implementation.
    - Each row of the input file contains 8-bit data values structured as:
        - **Expanded Public Key (expanded_pk)**
        - **Signature (sig)**
        - **Target Value (t)**

2. **File Handling**:
    - The input test vector file is processed by a JavaScript implementation of mayo_verify to calculate the intermediate value y.
    - The output file from the JavaScript implementation includes:
        - expanded_pk, sig, and the computed y for each test vector.
    - This file is the primary input for the hardware implementation.

3. **Data Integrity**:
    - All input values are assumed to be correctly formatted and validated prior to hardware processing.
    - No error handling for invalid or corrupted input data is implemented in hardware.

2. Hardware Memory Management Assumptions

1. **Memory Architecture**:
    - Dual-ported RAM with an **n×4-bit** width is used to store:
        - expanded_pk
        - sig

2. **Data Loading**:
    - Data from the input file is loaded into the memory using a Serial-In-Parallel-Out (SIPO) FIFO structure.
    - Each memory location stores **n/2 8-bit values**, assuming that the data is byte-aligned.

3. **Simplification**:
    - Data storage and retrieval from the memory are assumed to occur without any latency or bottlenecks.
    - Memory initialization and clearing are assumed to be pre-configured and error-free.

3. Processing and Computation Assumptions

1. **Start Signal**:
    - Computation begins when the calc signal is asserted (calc=1).
2. **Counter Management**:
    - Five independent counters are used to manage different iterations during processing:
        - Assumes that counters are reset and initialized correctly at the start of each computation cycle.
3. **Modular Design**:
    - The implementation is divided into multiple modules, each with specific functionality:
        - **4-bit Multiplication Module**: Handles gf16 multiplications with mod x^4+x+1.
        - **Array Multiplication Module**: Processes one row of data per cycle.
        - **Modular Reduction Module**: Performs reduction using the polynomial f(z).

4. Output Validation Assumptions

1. **Validation Logic**:
    - The computed y value is stored in a dedicated register (y_reg).
    - It is compared against the t value from the input file.
    - A match results in a high valid signal with the done signal asserted.
    - Any mismatch results in high impedance (Z) output.
2. **Timing Simplification**:
    - Assumes no delay in comparison or signal propagation to the output.
3. **Error Handling**:
    - Assumes all calculations are accurate, and no additional error-detection mechanisms are required.

This hardware implementation is generalized to work for 2 parameter sets MAYO1 and MAYO2