Explanation of the Components:

- **Input Ports**:
  - `clk`: Clock signal that drives the sequential logic.
  - `reset`: Asynchronous reset to initialize or clear the counter.
  - `syn_clr`: Synchronous clear that resets the counter within the clock cycle.
  - `load`: Load signal to input a parallel value into the counter.
  - `en`: Enable signal that allows the counter to increment.
  - `d`: Input data used when loading a value into the counter.
- **Output Ports**:
  - `max_tick`: A flag that becomes `1` when the counter reaches its maximum value (overflow condition).
  - `q`: The current count stored in the counter register.
- **Internal Signals**:
  - `r_reg`: The register that holds the current count value.
  - `r_next`: The next value to be loaded into the counter register in the next clock cycle.

Functionality:

1. **Synchronous Clear**: When `syn_clr` is asserted (`1`), the counter resets to `0` on the next clock edge.
2. **Load Operation**: If `load` is asserted, the counter will load the input data `d` into the counter register.
3. **Count Operation**: When `en` is high, the counter increments by 1.
4. **Pause**: If both `syn_clr`, `load`, and `en` are low, the counter maintains its current value.

This template is useful for implementing a universal counter, which can be customized based on bit-width or operations required in sequential circuits. The **max_tick** signal is especially useful for detecting when the counter reaches its maximum value.