```
package cosc441_mac;

simple CsmaMac like InterfaceMac
{
    parameters:
        @signal[bufferLossSig](type=bool);        // number of packets lost at MAC
buffer
        @signal[bufferEnteredSig](type=bool);     // number of packets admitted to
MAC buffer (and transmission)
        @signal[numberAttemptsSig](type=long);    // number of attempts per packet
        @signal[accessFailedSig](type=bool);      // number of packets for which no
ACK is ever received
        @signal[accessSuccessSig](type=bool);     // number of packets for which an
ACK is received
        @statistic[bufferLossSig](record=count; title="MAC: number of higher-layer
packets lost at buffer");
        @statistic[bufferEnteredSig](record=count; title="MAC: number of
higher-layer packets admitted to buffer");
        @statistic[numberAttemptsSig](record=stats; title="MAC: number of attempts
per packet");
        @statistic[accessFailedSig](record=count; title="MAC: number of MAC packets
for which no ACK was received");
        @statistic[accessSuccessSig](record=count; title="MAC: number of MAC packets
for which an ACK was received");

        int ownAddress;                       // own node address
        int bufferSize;                       // max number of application messages
the MAC can hold
        int maxBackoffs;                      // max number of backoff operations
per attempt
        int maxAttempts;                      // max number of attempts
        int macOverheadSizeData @unit(byte);  // Overhead for a MAC data packet
        int macOverheadSizeAck @unit(byte);   // Overhead for a MAC acknowledgement
        double macAckDelay @unit(s);             // fixed waiting time for sending
ACK
        double ackTimeout @unit(s);              // timeout for ACK packet
        volatile double csBackoffDistribution @unit(s) = default(uniform(0ms, 3ms));
   // backoff after busy CS
        volatile double attBackoffDistribution @unit(s) = default(uniform(0ms,
10ms));   // backoff after failed attempt
        volatile double succBackoffDistribution @unit(s) = default(uniform(0ms,
3ms));    // backoff after successful attempt

    gates:
        input fromHigher;
        output toHigher;
        input fromTransceiver;
        output toTransceiver;
}
```