



CSCE 491 Project Specification – Part 3

Dr. James P. Davis
jimdavis@cse.sc.edu

Exception Codes

The following list of exception codes, shown in the table, are likely to be generated from within the various blocks of the 802.11 MAC Layer Receiver block and its sub-blocks. The bus 'ErrCode' encodes the exception code in a 4-bit value (shown in binary), where the MSB is on the left, LSB is on the right.

#	Code	Exception	Description
1	0000	No Exception	This should be the default value for this bus, and is the value that the Receiver block would make available to the MAC Control block (not part of this project) on completion of processing of the current received frame.
2	0001	CRC Error	Generated by the FCS_Decoder block, this exception indicates that an error in the CRC calculation was detected. It is sufficient to note the error, not where the error actually occurred.
3	0010	Protocol Version Error	Generated by FCH_Decoder block, indicates that a value for the protocol version in the Frame Control word is a value other than the base supported (namely, '00').
4	0011	Frame Type/Subtype Mismatch	Generated by FCH_Decoder, indicates that the value for the subtype doesn't match that of the specified type encountered in the control word. Could indicate a CRC error (which would be found later in processing) or possible intrusion attempt using frame spoofing.
5	0100	Frame Address Sync Error	Generated in Addr_Decoder block, covers the case where the RA is equal to the TA. This is an erroneous case, signifying that the station is sending frames to itself. Might indicate spoofing. Only occurs for RTS and Data frames involving 2 addresses.
6	0101	Fragmentation Sync Error	Generated in the Seq_Control_Decoder block, where we have a Data frame fragment, and the Fragment No. is defined in proper sequence, but is less than '1111'; however, we have the MoreFragments bit reported by the FCH_decoder block set to zero.
7	0110	Erroneous Fragment Error	Generated by the Seq_Control_Decoder block, where the previous frame had a fragment number ('b0000 – b1111') with MoreFragments bit set, but the current frame has its MoreFragments bit set but no Fragment No. following the sequence (either zeros or fragment number other than that expected).

#	Code	Exception	Description
8	0111	Duplicate Sequence Number Error	Generated in the Seq_Control_Decoder block, where the Sequence No. is zero, or some non-zero value, for more than one non-fragmented frame in sequence. This implies that the Sequence No. has the same value as that of the previous received frame (must be from the same sender), there is no fragmentation indicated (i.e., the MoreFrag bit is not set) and the Retry bit is also not set.
9	1000	Sequence Sync Error	Generated in the Seq_Control_Decoder block, where the current frame's Sequence No. is some value other than that of the previous frame's, plus one, or if zero (following the wrap of Sequence No. values from 2^{12} minus 1, wrapping to zero as the next value in frame sequence). This would be for a given frame sequence from a specific sender address.
10	1001	Address Format Error	Generated by the Addr_Decoder block. The addresses received for TA or RA do not have their MSB 2 bits as zero, as provided by the specification, for the various frame subtypes we are using.
11	1010	Byte Count Error	For each frame of a given type and subtype, we come up short of the total number of words that should be shifted into the Receiver block. Each of these has a pre-defined number of words, so we should maintain a word count to insure that all blocks should be getting their appropriate data. This might manifest itself by the Sequencer getting out of sync with the shifter, in that some of the decoder blocks never get enabled, because there isn't any more data from the PHY Layer to shift and pass on to them. The frame is too short. This could be detected in several places in the design, but perhaps the best place is in the shifter (hence, the word counter that we haven't used up to this point).
12	1011	Retry Sync Error	Detected in the Seq_Control_Decoder block, based on the posting of the RetryFlag by the FCH_decoder block. If the Sequence No. is the next one in the sequence
13	1100	Duplicate Frame Error	Detected in the Seq_Control_Decoder block, based on checking the Sequence No., or Frag No. (if we are fragmenting), and it is the same as the previous frame received from the given sender (and we are the targeted receiver), and the RetryBit detected by FCH_Decoder is not set to '1' indicating this is NOT a retry frame. It could be a reflection of the frame, arriving at a later time than the original frame's image. Presuming we received the original correctly, the Receiver would have already signaled the Transmitter to send an appropriate ACK response. Note this is valid only for Data frames.
14	1101	Retry Frame Encountered	Detected in the Seq_Control_Decoder block, based on checking the Sequence No., or Frag No. (if we are fragmenting), and it is the same as the previous frame received from the given sender (and we are the targeted receiver), and the RetryBit detected by FCH_Decoder is set to '1' indicating this IS a retry frame. In this case, we have already seen a good version of this frame, but a delay in response failed to arrive back to the sender.

#	Code	Exception	Description
15	1110	Future use.	You may have found some other cases.
16	1111	Future use.	You may have found some other cases.

The behavior of the design should be such that, when one block asserts the ERR flag and posts an ErrCode, processing in all the blocks should halt, and control in each thread should return to its base poll loop. This is to conserve power as much as possible, since the assumption of the protocol is that the frame-sender will eventually timeout waiting for a response from the receiver, and will send a retry.

Note that, when we are comparing current versus previous frame Sequence or Fragment numbers, we must be sure that these previously stored numbers are for frames that had no exceptions in their processing. In the case of detecting a framing error, we want to not keep the extracted Sequence and/or Fragment number, and we also don't want to save the DID value in the NAV_Register (when the frame is not addressed to this station).

Note also that determining that a frame is not destined for the receiving station isn't an exception, but we would stop processing the frame in the same manner. We simply save the DID value in the NAV_Register and prepare to receive the next frame data stream.