```
1
2   /*
3   Decoding Table
4
5   where
6
7   *   cnt indicates previous address byte size
8
9   cnt == RD_Header && iBytePacket == 0XXX XXX1 then  oAddressEn = H, oAddress[7:2] =
    iBytePacket[6:1] ;
10  cnt == RD_Header && iBytePacket == 1XXX XXX1 then  oAddressEn = L, oAddress[7:2] =
    iBytePacket[6:1] , cnt = RD_Byte_1;
11
12  cnt == RD_Byte_1 && iBytePacket == 00XX XXXX then oAddressEn = H, oAddress[14:8] =
    iBytePacket[6:0];
13  cnt == RD_Byte_1 && iBytePacket == 01XX XXXX then oAddressEn = L, oAddress[14:8] =
    iBytePacket[6:0],  cnt = RD_Exception;
14  cnt == RD_Byte_1 && iBytePacket == 1XXX XXXX then oAddressEn = L, oAddress[14:8] =
    iBytePacket[6:0], cnt == RD_Byte_2;
15
16  cnt == RD_Byte_2 && iBytePacket == 00XX XXXX then oAddressEn = H, oAddress[21:15] =
    iBytePacket[6:0] has_excp = 0;
17  cnt == RD_Byte_2 && iBytePacket == 01XX XXXX then oAddressEn = L, oAddress[21:15] =
    iBytePacket[6:0], has_excp = 1;
18  cnt == RD_Byte_2 && iBytePacket == 1XXX XXXX then oAddressEn = L, oAddress[21:15] =
    iBytePacket[6:0], cnt == RD_Byte_3 excp = 0;
19
20  cnt == RD_Byte_3 && iBytePacket == 00XX XXXX then oAddressEn = H, oAddress[28:22] =
    iBytePacket[6:0]  cnt = RD_Header;
21  cnt == RD_Byte_3 && iBytePacket == 01XX XXXX then oAddressEn = L, oAddress[28:22] =
    iBytePacket[6:0], cnt = RD_Exception;
22  cnt == RD_Byte_3 && iBytePacket == 1XXX XXXX then oAddressEn = L, oAddress[28:22] =
    iBytePacket[6:0], cnt = RD_Byte_4;
23
24  cnt == RD_Byte_4 && iBytePacket == 0000 1XXX then oAddressEn = H, oAddress[31:29] =
    iBytePacket[2:0] cnt = RD_Header;
25  cnt == RD_Byte_4 && iBytePacket == 0100 1XXX then oAddressEn = H, oAddress[31:29] =
    iBytePacket[2:0] cnt = RD_Exception ;
26
27  cnt == RD_Exception NS <= iBytePacket[0]; Exception[3:0] <= iBytePacket[4:1];
28      iBytePacket[7] == H then    cnt = RD_Hyp;
29      iBytePacket[7] == L then    oAddressEn = H; cnt = RD_Header;
30
31  cnt == RD_Hyp then  Exception[8:4] = iBytePacket[4:0]; Hyp = iBytePacket[5]; cnt =
    RD_Header; oAddressEn = H;
32
33  */
34
35
36  module decoder(
37          input iClk,
38          input iRsn,
39          input iBytePacketEn,
40          input [7:0] iBytePacket,
41          output reg oAddressEn,
42          output reg [31:0] oAddress
43  );
44
45  //parameters
46  parameter H = 1'b1, L = 1'b0;
47
48  parameter ARM = 2;
49  parameter THUMB = 1;
50
51  parameter RD_Header    = 0;
52  parameter RD_Byte_1    = 1;
53  parameter RD_Byte_2    = 2;
54  parameter RD_Byte_3    = 3;
55  parameter RD_Byte_4    = 4;
56  parameter RD_Exception = 5;
57  parameter RD_Hyp       = 6;
58
59  // Declare Registers
```

```
60  reg [8:0] Exception;
61  reg NS, Hyp;
62  reg [2:0] cnt; // conut received Address Bytes
63
64
65  always @ ( posedge iClk or negedge iRsn ) begin
66
67      if(!iRsn) begin //Reset Func
68          NS <= L;
69          Hyp <= L;
70          cnt <= {3{L}};
71          oAddress <= {32{L}};
72          oAddressEn <= L;
73
74      end else if( iBytePacketEn ) begin // If Packet in Enabled
75
76          case ( cnt )
77
78              RD_Header : begin       //Read Header  Byte Signiture   XXXX XXX1
79
80                  if( iBytePacket[0] ) begin
81                      NS <= L;
82                      Hyp <= L;
83
84                      oAddress <= ( oAddress & { {24{H}}, {8{L}}} ) |
                        (iBytePacket[6:1] << ARM); //Preserve Upper Bits
85
86                      if( iBytePacket[7] ) begin
87                          cnt <= RD_Byte_1;
88                          oAddressEn <= L;
89
90                      end else begin
91                          oAddressEn <= H;
92                      end
93
94                  end
95
96              end
97
98              RD_Byte_1, RD_Byte_2, RD_Byte_3 : begin
99
100                 if( iBytePacket[7] ) begin  //Read Next Byte if 'C' Signiture is H
101
102                     oAddress <=  ( oAddress  &  ~(  {7{H}} << ( 7*(cnt-1) + ARM + 6
                        ) ) )  | iBytePacket[6:0] << ( 7*(cnt-1) + ARM + 6 );
103                     cnt <= cnt + 1;
104
105                 end else begin
106
107                     oAddress <=  ( oAddress  &  ~(  {6{H}} << ( 7*(cnt-1) + ARM + 6
                        ) ) )  | iBytePacket[5:0] << ( 7*(cnt-1) + ARM + 6 );
108                     if( iBytePacket[6] ) begin
109                         cnt <= RD_Exception;
110                     end else begin
111                         cnt <= RD_Header;
112                         oAddressEn <= H;
113                     end
114
115
116                 end
117
118             end
119
120             RD_Byte_4 : begin       //Read Last Byte
121
122                 oAddress <=  ( oAddress  &  ~(  {3{H}} << ( 27 + ARM ) ) )  |
                    iBytePacket[5:0] << ( 27 + ARM );
123
124                     if( iBytePacket[6] ) begin
125                         cnt <= RD_Exception;
126                     end else begin
127                         cnt <= RD_Header;
128                         oAddressEn <= H;
```

```
129                     end
130             end
131
132             RD_Exception : begin        //Read Exception Byte
133
134                     NS <= iBytePacket[0];
135                     Exception[3:0] <= iBytePacket[4:1];
136
137                     if ( iBytePacket[7] ) begin
138                         cnt <= RD_Hyp;
139                     end else begin
140                         oAddressEn <= H;
141                         cnt <= RD_Header;
142                     end
143
144             end
145
146             RD_Hyp : begin          //Read Hyp  Byte
147
148                     Exception[8:4] <= iBytePacket[4:0];
149                     Hyp <= iBytePacket[5];
150                     cnt <= RD_Header;
151                     oAddressEn <= H;
152
153             end
154
155         endcase
156
157     end else begin
158
159                 if( oAddressEn ) begin
160                     oAddressEn <= L;
161                 end
162
163     end
164
165 end
166
167 endmodule
168
169
170
171
```