```
 1: unit Reverse;
 2:
 3: {$mode objfpc}{$H+}
 4:
 5: //=============================================================================================
 6: //
 7: //  Reverse.pas
 8: //
 9: //  This unit provides Reverse functionality for the currently selected channels.
10: //
11: //  The default value for Reverse is OFF.
12: //
13: //  The Reverse button is a "Toggle". When pressed, state of the currently selected PTT
14: //  channel is changed.
15: //
16: //  Calls: Final : Finalize
17: //                 LCDDisplay : DisplayUHFReverseStatus
18: //                      DisplayVHFReverseStatus
19: //                 Main
20: //                 PSCommand : TogglePowerOnOff
21: //
22: //  Called By: Main : TfrmMain.btnReverseClick
23: //
24: //  Ver: 1.0.0
25: //
26: //  Date: 9 Dec 2013
27: //
28: //=============================================================================================
29:
30: interface
31:
32: uses
33:   Classes, Dialogs, Graphics, SysUtils,
34:   // Application  Units
35:   AppConstants, AppVariables, LCDDisplay;
36:
37: procedure ToggleReverse;
38: procedure UHFReverseOff;
39: procedure VHFReverseOff;
40:
41: //=============================================================================================
42: implementation
43:
44: uses
45:   BufCommand, Main;
46:
47: procedure UHFReverseOff;
48: begin
49:   // Change the Button colours
50:   frmMain.bbtReverse.Font.Color := clBlack;
51:   frmMain.bbtReverse.Font.Style := [];
52:   gvstrUHFReverseState := gcstrOff;
53:   DisplayUHFRXFrequency;
54:   DisplayUHFReverseStatus;
55:   DisplayUHFShiftStatus;
56:   DisplayUHFCTStatus;
57: end;// procedure UHFReverse
58:
59: //=============================================================================================
60: procedure VHFReverseOff;
```

```
 61: begin
 62:   // Change the Button colours
 63:   frmMain.bbtReverse.Font.Color := clBlack;
 64:   frmMain.bbtReverse.Font.Style := [];
 65:   gvstrVHFReverseState := gcstrOff;
 66:   DisplayVHFRXFrequency;
 67:   DisplayVHFReverseStatus;
 68:   DisplayVHFShiftStatus;
 69:   DisplayVHFCTStatus;
 70: end;// procedure VHFReverse
 71:
 72: //=============================================================================================
 73: function RevRXFrequency : string;
 74:
 75: var
 76:   vsngTFrequency : single;
 77:   vsngTOffset : single;
 78:
 79:
 80: begin
 81:
 82:   // The Shift offset depends on the PTTBand and the Shift (Plus or Minus.
 83:   // Simplex never gets here.
 84:   if gvstrPTTBand = gcstrVHF then
 85:   begin
 86:     // Save the Original RXFrequency, Shift and Tone
 87:     gvstrVHFOrigRXFrequency := gvstrVHFRXFrequency;
 88:     gvstrVHFOrigShift := gvstrVHFShift;
 89:     gvstrVHFOrigTone := gvstrVHFTone;
 90:     gvstrVHFOrigCTCSS := gvstrVHFCTCSS;
 91:     // First we have to Reverse the Shift
 92:     if gvstrVHFOrigShift = gcstrShiftPlus then
 93:       gvstrVHFShift := gcstrShiftMinus
 94:     else
 95:       gvstrVHFShift := gcstrShiftPlus;
 96:     // Now turn off the Tone
 97:    gvstrVHFTone := gcstrOff;
 98:    gvstrVHFCTCSS := gcstrOff;
 99:
100:     // Now we calculate a Reversed RXFrequency based on the Original Shift
101:     vsngTFrequency := StrToFloat( gvstrVHFRXFrequency );
102:     vsngTOffset := StrToFloat(gcstrVHFShiftOffset);
103:     if gvstrVHFOrigShift = gcstrShiftPlus then
104:     begin
105:      vsngTFrequency := vsngTFrequency + 600000;
106:     end
107:     else
108:     begin
109:       vsngTFrequency := vsngTFrequency - 600000;
110:     end;// if gvstrVHFShift = gcstrShiftPlus
111:   end
112:   else
113:   begin
114:     // Save the Original RXFrequency and Shift
115:     gvstrUHFOrigRXFrequency := gvstrUHFRXFrequency;
116:     gvstrUHFOrigShift := gvstrUHFShift;
117:     gvstrUHFOrigTone := gvstrUHFTone;
118:     gvstrUHFOrigCTCSS := gvstrUHFCTCSS;
119:     // First we have to Reverse the Shift
120:     if gvstrUHFShift = gcstrShiftPlus then
```

```
121:       gvstrUHFShift := gcstrShiftMinus
122:     else
123:       gvstrUHFShift := gcstrShiftPlus;
124:     // Now turn off the Tone
125:     gvstrUHFTone := gcstrOff;
126:     gvstrUHFCTCSS := gcstrOff;
127:
128:     // Now we calculate a Reversed RXFrequency based on the Original Shift
129:     vsngTFrequency := StrToFloat( gvstrUHFRXFrequency );
130:     vsngTOffset := StrToFloat(gcstrUHFShiftOffset);
131:     if gvstrUHFOrigShift = gcstrShiftPlus then
132:     begin
133:      vsngTFrequency := vsngTFrequency + 5000000;
134:     end
135:     else
136:     begin
137:       vsngTFrequency := vsngTFrequency - 5000000;
138:     end;// if gvstrVHFShift = gcstrShiftPlus
139:
140:   end;// if gvstrPTTBand = gcstrVHF
141:
142:   Result := '00' + FloatToStr(vsngTFrequency);
143:
144: end;// function RevRXFrequency
145:
146: //=============================================================================================
147: procedure ToggleReverse;
148: begin
149:
150:
151:   // Check to see which Band Channel is selected
152:   if gvstrPTTBand = gcstrVHF then
153:   begin
154:
155:     // Check to see if this is a simplex channel. If it is, there is no point in going
156:     // to Reverse
157:     if gvstrVHFShift = gcstrShiftSimplex then
158:     begin
159:       ShowMessage( 'Simplex Channel' );
160:       Exit;
161:     end;// if gvstrVHFShift = gcstrShiftSimplex
162:
163:      // Current Band is VHF now we toggle VHFReverse
164:     if gvstrVHFReverseState = gcstrOff then
165:     begin
166:        // Calculate the new RX Frequency using the configured shift and set the Buffer
167:       gvstrVHFRXFrequency := RevRXFrequency;
168:       SetBuffer(gcstrVHF);
169:       // Change the Button colours
170:       frmMain.bbtReverse.Font.Color := clRed;
171:       frmMain.bbtReverse.Font.Style := [fsBold];
172:       gvstrVHFReverseState := gcstrOn;
173:       DisplayVHFReverseStatus;
174:       DisplayVHFShiftStatus;
175:       DisplayVHFCTStatus;
176:     end
177:     else
178:     begin
179:       // Restore the Original RX Frequency, Shift, Tone and set the Buffer
180:       gvstrVHFRXFrequency := gvstrVHFOrigRXFrequency;
```

```
181:       gvstrVHFShift := gvstrVHFOrigShift;
182:       gvstrVHFTone := gvstrVHFOrigTone;
183:       gvstrVHFCTCSS := gvstrVHFOrigCTCSS;
184:       SetBuffer(gcstrVHF);
185:       VHFReverseOff;
186:     end;// if gvstrUHFReverseState = gcstrOff
187:
188:   end
189:   else
190:   begin
191:
192:     // Check to see if this is a simplex channel. If it is, there is no point in going
193:     // to Reverse
194:     if gvstrUHFShift = gcstrShiftSimplex then
195:     begin
196:       ShowMessage( 'Simplex Channel' );
197:       Exit;
198:     end;// if gvstrUHFShift = gcstrShiftSimplex
199:
200:     // Current Band is UHF now we toggle UHFReverse
201:     if gvstrUHFReverseState = gcstrOff then
202:     begin
203:       // Calculate the new RX Frequency using the configured shift and set the Buffer
204:       gvstrUHFRXFrequency := RevRXFrequency;
205:       SetBuffer(gcstrUHF);
206:       // Change the Button colours
207:       frmMain.bbtReverse.Font.Color := clRed;
208:       frmMain.bbtReverse.Font.Style := [fsBold];
209:       gvstrUHFReverseState := gcstrOn;
210:       DisplayUHFReverseStatus;
211:       DisplayUHFShiftStatus;
212:       DisplayUHFCTStatus;
213:     end
214:     else
215:     begin
216:       // Restore the Original RX Frequency, Shift, Tone and set the Buffer
217:       gvstrUHFRXFrequency := gvstrUHFOrigRXFrequency;
218:       gvstrUHFShift := gvstrUHFOrigShift;
219:       gvstrUHFTone := gvstrUHFOrigTone;
220:       gvstrUHFCTCSS := gvstrUHFOrigCTCSS;
221:       SetBuffer(gcstrUHF);
222:       UHFReverseOff;
223:     end;// if gvstrUHFReverseState = gcstrOff
224:
225:   end;// if gvstrPTTBand = gcstrVHF
226:
227: end;// procedure ToggleReverse
228:
229: //=============================================================================================
230: end.// Reverse;
231:
```