```
 1: unit LCDDisplay;
 2:
 3: {$mode objfpc}{$H+}
 4:
 5: //=============================================================================================
 6: //
 7: //  LCDDisplay.pas
 8: //
 9: //  Calls: AppConstants
10: //         AppVariables
11: //         Main
12: //
13: //  Called By: BCCommand : TogglePTTBand
14: //             BUFCommand : BUFResponseHandler
15: //             BYCommand : BYResponseHandler
16: //             FAV : SetFAVChannel
17: //             Init : Initialize
18: //             Mem_VHF : TfrmMEM.bbtSelectClick
19: //             PSCommand : PSResponseHandler
20: //             Reverse : Toggle Reverse
21: //             RXCommand : RXResponseHandler
22: //             TXCommand : TXResponseHandler
23: //
24: //  Ver: 1.0.0
25: //
26: //  Date: 7 Dec 2013
27: //
28: //=============================================================================================
29:
30: interface
31:
32: uses
33:   Classes, Dialogs, Graphics, SysUtils,
34:   // Application Units
35:   AppConstants, AppVariables, Utilities;
36:
37: procedure LCDOff;
38: procedure LCDOn;
39: procedure UpdateLCDDisplay;
40:
41: procedure DisplayBCStatus;
42: procedure DisplayUHFBusyStatus(vstrStatus : string);
43: procedure DisplayVHFBusyStatus(vstrStatus : string);
44: procedure DisplayUHFCTStatus;
45: procedure DisplayVHFCTStatus;
46: procedure DisplayUHFChannelName;
47: procedure DisplayVHFChannelName;
48: procedure DisplayUHFChannelNr;
49: procedure DisplayVHFChannelNr;
50: procedure DisplayUHFDataSource;
51: procedure DisplayVHFDataSource;
52: procedure DisplayUHFDTSSStatus;
53: procedure DisplayVHFDTSSStatus;
54: procedure DisplayUHFRXFrequency;
55: procedure DisplayVHFRXFrequency;
56: procedure DisplayUHFReverseStatus;
57: procedure DisplayVHFReverseStatus;
58: procedure DisplayUHFRFPowerStatus;
59: procedure DisplayVHFRFPowerStatus;
60: procedure DisplayUHFRXStatus(vstrStatus : string);
```

```
 61: procedure DisplayVHFRXStatus(vstrStatus : string);
 62: procedure DisplayUHFShiftStatus;
 63: procedure DisplayVHFShiftStatus;
 64: procedure DisplayUHFTXStatus(vstrStatus : string);
 65: procedure DisplayVHFTXStatus(vstrStatus : string);
 66:
 67: implementation
 68:
 69: uses
 70:     BYCommand, Main;
 71:
 72: {=============================================================================================}
 73: {                       LCD PANEL                                                             }
 74: {=============================================================================================}
 75: procedure LCDOff;
 76: begin
 77:   frmMain.pnlLCD.Visible := False;
 78: end;// procedure LCDOff;
 79:
 80: //---------------------------------------------------------------------------------------------
 81: procedure LCDOn;
 82: begin
 83:   frmMain.pnlLCD.Visible := True;
 84: end;// procedure LCDOn;
 85:
 86: //---------------------------------------------------------------------------------------------
 87: procedure UpdateLCDDisplay;
 88: begin
 89:   DisplayBCStatus;
 90:   GetUHFBYStatus;
 91:   GetVHFBYStatus;
 92:   DisplayUHFChannelName;
 93:   DisplayVHFChannelName;
 94:   DisplayUHFChannelNr;
 95:   DisplayVHFChannelNr;
 96:   DisplayUHFCTStatus;
 97:   DisplayVHFCTStatus;
 98:   DisplayUHFDataSource;
 99:   DisplayVHFDataSource;
100:   DisplayUHFDTSSStatus;
101:   DisplayVHFDTSSStatus;
102:   DisplayUHFRXFrequency;
103:   DisplayVHFRXFrequency;
104:   DisplayUHFReverseStatus;
105:   DisplayVHFReverseStatus;
106:   DisplayUHFRFPowerStatus;
107:   DisplayVHFRFPowerStatus;
108:   DisplayUHFShiftStatus;
109:   DisplayVHFShiftStatus;
110:
111: end;// procedure UpdateLCDDisplay;
112:
113: {=============================================================================================}
114: {                         BC STATUS                                                           }
115: {=============================================================================================}
116: procedure DisplayBCStatus;
117:
118: const
119:   cbytPTTFreqSize = 48;
120:   cbytInactiveFreqSize = 32;
```

```
121:
122: begin
123:
124:   if gvstrPTTBand = gcstrVHF then // Current Band is VHF
125:   begin
126:     frmMain.lblVHFPTT.Visible := True;
127:     frmMain.lblUHFPTT.Visible := False;
128:     frmMain.lblVHFFreq.Font.Size := cbytPTTFreqSize;
129:     frmMain.lblVHFFreq.Font.Bold := True;
130:     frmMain.lblUHFFreq.Font.Size := cbytInactiveFreqSize;
131:     frmMain.lblUHFFreq.Font.Bold := False;
132:     DisplayVHFReverseStatus;
133:   end
134:   else // it is UHF
135:   begin
136:     frmMain.lblVHFPTT.Visible := False;
137:     frmMain.lblUHFPTT.Visible := True;
138:     frmMain.lblVHFFreq.Font.Size := cbytInactiveFreqSize;
139:     frmMain.lblVHFFreq.Font.Bold := False;
140:     frmMain.lblUHFFreq.Font.Size := cbytPTTFreqSize;
141:     frmMain.lblUHFFreq.Font.Bold := True;
142:     DisplayUHFReverseStatus;
143:   end;// if gvstrPTTBand = gcstrVHF
144:
145: end;// procedure DisplayBCStatus
146:
147: {=============================================================================================}
148: {                              BUSY STATUS                                                    }
149: {=============================================================================================}
150: procedure DisplayUHFBusyStatus(vstrStatus : string);
151: begin
152:   frmMain.lblUHFOnAirBusy.Caption := vstrStatus;
153: end;// procedure DisplayUHFBusyStatus
154:
155: //---------------------------------------------------------------------------------------------
156: procedure DisplayVHFBusyStatus(vstrStatus : string);
157: begin
158:   frmMain.lblVHFOnAirBusy.Caption := vstrStatus;
159: end;// procedure DisplayVHFBusyStatus
160:
161: {=============================================================================================}
162: {                             CHANNEL NAME                                                    }
163: {=============================================================================================}
164: procedure DisplayUHFChannelName;
165: begin
166:   frmMain.lblUHFChannelName.Caption := gvstrUHFChannelName;
167: end;// procedure DisplayUHFChannelName;
168:
169: //---------------------------------------------------------------------------------------------
170: procedure DisplayVHFChannelName;
171: begin
172:   frmMain.lblVHFChannelName.Caption := gvstrVHFChannelName;
173: end;// procedure DisplayVHFChannelName;
174:
175: {=============================================================================================}
176: {                            CHANNEL NR                                                       }
177: {=============================================================================================}
178: procedure DisplayUHFChannelNr;
179: begin
180:   frmMain.lblUHFChannelNr.Caption := gvstrUHFChannelNr;
```

```
181: end;// procedure DisplayUHFChannelNr;
182:
183: //-------------------------------------------------------------------------------------
184: procedure DisplayVHFChannelNr;
185: begin
186:   frmMain.lblVHFChannelNr.Caption := gvstrVHFChannelNr;
187: end;// procedure DisplayVHFChannelNr;
188:
189: {=====================================================================================}
190: {                                 CT STATUS                                           }
191: {=====================================================================================}
192: procedure DisplayUHFCTStatus;
193:  // This routine handles the status of both the UHF Tone and CTCSS functions. They are
194: // mutually exclusive so only one conition can be present. Either CTCSS, Tone or None.
195: VAR
196:   vwrdCTFreq : Word;
197:   vbytCode : Byte;
198:
199: begin
200:
201:   if gvstrUHFCTCSS = gcstrOn then
202:   begin
203:     frmMain.lblUHFTCT.Caption := 'CT';
204:     frmMain.lblUHFTCT.visible := True;
205:     frmMain.lblUHFTCTFreq.Caption := GetToneFrequencyFromToneNr(StrToInt(gvstrUHFCTCSSNr));
206:     frmMain.lblUHFTCTFreq.visible := True;
207:   end
208:   else if gvstrUHFTone = gcstrOn then
209:   begin
210:     frmMain.lblUHFTCT.Caption := 'T';
211:     frmMain.lblUHFTCT.visible := True;
212:     frmMain.lblUHFTCTFreq.Caption := GetToneFrequencyFromToneNr(StrToInt(gvstrUHFToneNr));
213:     frmMain.lblUHFTCTFreq.visible := True;
214:   end
215:   else
216:   begin
217:     frmMain.lblUHFTCT.visible := False;
218:     frmMain.lblUHFTCTFreq.visible := False;
219:   end;// if gvstrUHFTone = gcstrOn
220:
221: end;// procedure DisplayUHFCTCStatus
222:
223: //-------------------------------------------------------------------------------------
224: procedure DisplayVHFCTStatus;
225: // This routine handles the status of both the VHF Tone and CTCSS functions. They are
226: // mutually exclusive so only one conition can be present. Either CTCSS, Tone or None.
227: VAR
228:  vwrdCTFreq : Word;
229:  vbytCode : Byte;
230:
231: begin
232:
233:  if gvstrVHFCTCSS = gcstrOn then
234:  begin
235:    frmMain.lblVHFTCT.Caption := 'CT';
236:    frmMain.lblVHFTCT.visible := True;
237:    frmMain.lblVHFTCTFreq.Caption := GetToneFrequencyFromToneNr(StrToInt(gvstrVHFCTCSSNr));
238:    frmMain.lblVHFTCTFreq.visible := True;
239:  end
240:  else if gvstrVHFTone = gcstrOn then
```

```
241:  begin
242:    frmMain.lblVHFTCT.Caption := 'T';
243:    frmMain.lblVHFTCT.visible := True;
244:    frmMain.lblVHFTCTFreq.Caption := GetToneFrequencyFromToneNr(StrToInt(gvstrVHFToneNr));
245:    frmMain.lblVHFTCTFreq.visible := True;
246:  end
247:  else
248:  begin
249:    frmMain.lblVHFTCT.visible := False;
250:    frmMain.lblVHFTCTFreq.visible := False;
251:  end;// if gvstrVHFTone = gcstrOn
252:
253: end;// procedure DisplayVHFCTStatus
254:
255: {=============================================================================================}
256: {                            DATA SOURCE                                                      }
257: {=============================================================================================}
258: procedure DisplayUHFDataSource;
259: begin
260:   frmMain.lblUHFDataSource.Caption := gvstrUHFDataSource;
261: end;// procedure DisplayUHFDataSource;
262:
263: //---------------------------------------------------------------------------------------------
264: procedure DisplayVHFDataSource;
265: begin
266:   frmMain.lblVHFDataSource.Caption := gvstrVHFDataSource;
267: end;// procedure DisplayVHFDataSource;
268:
269: {=============================================================================================}
270: {                     DTSS STATUS                                                             }
271: {=============================================================================================}
272: procedure DisplayUHFDTSSStatus;
273: begin
274:
275:   if gvstrUHFDTSS = gcstrOn then
276:   begin
277:     frmMain.lblUHFDTSSCode.caption := gvstrUHFDTSSCode;
278:     frmMain.lblUHFDTSS.visible := True;
279:   end
280:   else
281:   begin
282:     frmMain.lblUHFDTSS.visible := False;
283:     frmMain.lblUHFDTSSCode.visible := False;
284:   end;// if gvstrUHFDTSS = gcstrOn
285:
286: end;// procedure DisplayUHFDTSSStatus;
287:
288: //---------------------------------------------------------------------------------------------
289: procedure DisplayVHFDTSSStatus;
290: begin
291:
292:     if gvstrVHFDTSS = gcstrOn then
293:     begin
294:       frmMain.lblVHFDTSSCode.caption := gvstrVHFDTSSCode;
295:       frmMain.lblVHFDTSS.visible := True;
296:     end
297:     else
298:     begin
299:       frmMain.lblVHFDTSS.visible := False;
300:       frmMain.lblVHFDTSSCode.visible := False;
```

```
301:     end;// if gvstrVHFDTSS = gcstrOn
302:
303: end;// procedure DisplayVHFDTSSStatus;
304:
305: {=============================================================================================}
306: {                  FREQUENCY                                                                  }
307: {=============================================================================================}
308: procedure DisplayUHFRXFrequency;
309: begin
310:   frmMain.lblUHFFreq.Caption := Copy(gvstrUHFRXFrequency,3,3) +
311:                                 '.' +
312:                                 Copy(gvstrUHFRXFrequency,6,3);
313: end;// procedure DisplayUHFRXFrequency;
314:
315: //---------------------------------------------------------------------------------------------
316: procedure DisplayVHFRXFrequency;
317: begin
318:   frmMain.lblVHFFreq.Caption := Copy(gvstrVHFRXFrequency,3,3) +
319:                                 '.' +
320:                                 Copy(gvstrVHFRXFrequency,6,3);
321: end;// procedure DisplayVHFRXFrequency;
322:
323: {=============================================================================================}
324: {                   REVERSE STATUS                                                            }
325: {=============================================================================================}
326: procedure DisplayUHFReverseStatus;
327: begin
328:
329:   if gvstrUHFReverseState = gcstrOn then
330:   begin
331:     frmMain.lblUHFReverse.visible := True;
332:     frmMain.bbtReverse.Font.Color := clRed;
333:     frmMain.bbtReverse.Font.Style := [fsBold];
334:   end
335:   else
336:   begin
337:     frmMain.lblUHFReverse.visible := False;
338:     frmMain.bbtReverse.Font.Color := clBlack;
339:     frmMain.bbtReverse.Font.Style := [];
340:   end;// if gvstrUHFReverseState = gcstrOn
341:
342: end;// procedure DisplayUHFReverseStatus
343:
344: {---------------------------------------------------------------------------------------------}
345: procedure DisplayVHFReverseStatus;
346: begin
347:
348:     if gvstrVHFReverseState = gcstrOn then
349:     begin
350:       frmMain.lblVHFReverse.visible := True;
351:       frmMain.bbtReverse.Font.Color := clRed;
352:       frmMain.bbtReverse.Font.Style := [fsBold];
353:     end
354:     else
355:     begin
356:       frmMain.lblVHFReverse.visible := False;
357:       frmMain.bbtReverse.Font.Color := clBlack;
358:       frmMain.bbtReverse.Font.Style := [];
359:     end;// if gvstrVHFReverseState = gcstrOn
360:
```

```
361: end;// procedure DisplayVHFReverseStatus
362:
363: {=============================================================================================}
364: {                    RF POWER STATUS                                                          }
365: {=============================================================================================}
366: procedure DisplayUHFRFPowerStatus;
367: begin
368:   case gvstrUHFRFPower of
369:     gcstrRFPowerLow : frmMain.lblUHFRFPwr.Caption := 'L';
370:     gcstrRFPowerMedium : frmMain.lblUHFRFPwr.Caption := 'M';
371:     gcstrRFPowerHigh : frmMain.lblUHFRFPwr.Caption := 'H';
372:   end;// case gvstrUHFRFPower of
373: end;// procedure DisplayUHFRFPowerStatus;
374:
375: //---------------------------------------------------------------------------------------------
376: procedure DisplayVHFRFPowerStatus;
377: begin
378:   case gvstrVHFRFPower of
379:     gcstrRFPowerLow : frmMain.lblVHFRFPwr.Caption := 'L';
380:     gcstrRFPowerMedium : frmMain.lblVHFRFPwr.Caption := 'M';
381:     gcstrRFPowerHigh : frmMain.lblVHFRFPwr.Caption := 'H';
382:   end;// case gvstrVHFRFPower of
383: end;// procedure DisplayVHFRFPowerStatus;
384:
385: {=============================================================================================}
386: {                              RX STATUS                                                      }
387: {=============================================================================================}
388: procedure DisplayUHFRXStatus(vstrStatus : string);
389: begin
390:   frmMain.lblUHFOnAirBusy.Caption := vstrStatus;
391: end;// procedure DisplayUHFRXStatus
392:
393: //---------------------------------------------------------------------------------------------
394: procedure DisplayVHFRXStatus(vstrStatus : string);
395: begin
396:   frmMain.lblVHFOnAirBusy.Caption := vstrStatus;
397: end;// procedure DisplayVHFRXStatus
398:
399: {=============================================================================================}
400: {                SHIFT STATUS                                                                 }
401: {=============================================================================================}
402: procedure DisplayUHFShiftStatus;
403: begin
404:
405:     if gvstrUHFShift = gcstrShiftSimplex then
406:       frmMain.lblUHFshift.Caption := ''
407:     else if gvstrUHFShift = gcstrShiftPlus then
408:       frmMain.lblUHFshift.Caption := '+'
409:     else
410:       frmMain.lblUHFshift.Caption := '-';
411:
412: end;// procedure DisplayUHFShiftStatus;
413:
414: //---------------------------------------------------------------------------------------------
415: procedure DisplayVHFShiftStatus;
416: begin
417:
418:   if gvstrVHFShift = gcstrShiftSimplex then
419:     frmMain.lblVHFshift.Caption := ''
420:   else if gvstrVHFShift = gcstrShiftPlus then
```

```
421:     frmMain.lblVHFshift.Caption := '+'
422:   else
423:     frmMain.lblVHFshift.Caption := '-';
424:
425: end;// procedure DisplayVHFShiftStatus;
426:
427: {=============================================================================================}
428: {                              TX STATUS                                                      }
429: {=============================================================================================}
430: procedure DisplayUHFTXStatus(vstrStatus : string);
431: begin
432:   frmMain.lblUHFOnAirBusy.Caption := vstrStatus;
433: end;// procedure DisplayUHFTXStatus
434:
435: //---------------------------------------------------------------------------------------------
436: procedure DisplayVHFTXStatus(vstrStatus : string);
437: begin
438:   frmMain.lblVHFOnAirBusy.Caption := vstrStatus;
439: end;// procedure DisplayVHFTXStatus
440:
441: {=============================================================================================}
442: end.// unit LCDDisplay
443:
```