```
 1: unit MEM_VHF;
 2:
 3: {$mode objfpc}{$H+}
 4:
 5: //=============================================================================================
 6: //
 7: //  Mem_VHF.pas
 8: //
 9: //  Calls: AppConstants
10: //         AppVariables
11: //         BCCommand : SetVHFBand
12: //         BUFCommand : SetBuffer
13: //         LCDDisplay : UpdateLCDDisplay
14: //         Mem
15: //         Utilities : GetToneFrequencyFromToneNr
16: //
17: //  Called By: MEM : TfrmMEM.Setup
18: //                   SetVHFChannel
19: //
20: //  Ver: 1.0.0
21: //
22: //  Date: 11 Aug 2013
23: //
24: //=============================================================================================
25:
26: interface
27:
28: uses
29:   Classes, Dialogs, SysUtils,
30:   // Application Units
31:   AppConstants, AppVariables, BCCommand, BUFCommand, LCDDisplay, Utilities;
32:
33: procedure LoadVHFStringGrid;
34: procedure SetVHFChannel;
35:
36: implementation
37:
38: uses
39:   Mem;
40:
41: //=============================================================================================
42: procedure LoadVHFStringGrid;
43:
44: var
45:   vbytTemp : Byte;
46:   vstrTStr : String;
47:
48: begin
49:
50:   for vbytTemp := 1 to gcbytMaxVHFChannels do
51:   begin
52:
53:     // Channel Nr
54:     frmMem.sgrVHF.Cells[gcbytChMemNrCol, vbytTemp] := IntToStr(vbytTemp);
55:
56:     // Channel Name
57:     frmMem.sgrVHF.Cells[gcbytNameCol, vbytTemp] :=
58:                     gvstrVHFChannelDataArray[vbytTemp,gcbytChannelNameField];
59:
```

```
 60:     // RX FREQUENCY
 61:     if Length (gvstrVHFChannelDataArray[vbytTemp,gcbytRXFrequencyField]) > 0 then
 62:       frmMem.sgrVHF.Cells[gcbytRXFreqCol, vbytTemp] :=
 63:          Copy(gvstrVHFChannelDataArray[vbytTemp,gcbytRXFrequencyField],3,3) +
 64:          '.' +
 65:          Copy(gvstrVHFChannelDataArray[vbytTemp,gcbytRXFrequencyField],6,3)
 66:     else
 67:       frmMem.sgrVHF.Cells[gcbytRXFreqCol, vbytTemp] := '';
 68:
 69:     // SHIFT
 70:     vstrTStr := gvstrVHFChannelDataArray[vbytTemp,gcbytShiftCol+1];
 71:     case vstrTStr of
 72:       gcstrShiftSimplex : frmMem.sgrVHF.Cells[gcbytShiftCol, vbytTemp] := gcstrTMV7ShiftSimplex
    ;
 73:       gcstrShiftPlus : frmMem.sgrVHF.Cells[gcbytShiftCol, vbytTemp] := gcstrTMV7ShiftPlus;
 74:       gcstrShiftMinus : frmMem.sgrVHF.Cells[gcbytShiftCol, vbytTemp] := gcstrTMV7ShiftMinus;
 75:     end;// case vstrTStr
 76:
 77:     // Offset
 78:     if Length(gvstrVHFChannelDataArray[vbytTemp,gcbytShiftOffsetField]) > 0 then
 79:       if gvstrVHFChannelDataArray[vbytTemp,gcbytShiftCol+1] = gcstrShiftSimplex then
 80:         frmMem.sgrVHF.Cells[gcbytOffsetCol, vbytTemp] := ''
 81:       else
 82:         frmMem.sgrVHF.Cells[gcbytOffsetCol, vbytTemp] :=
 83:               Copy(gvstrVHFChannelDataArray[vbytTemp,gcbytShiftOffsetField],2,2) +
 84:               '.' +
 85:               Copy(gvstrVHFChannelDataArray[vbytTemp,gcbytShiftOffsetField],4,2)
 86:     else
 87:       frmMem.sgrVHF.Cells[gcbytOffsetCol, vbytTemp] := '';
 88:
 89:     // Tone or CTCSS
 90:     // We only load this field if there is a valid record
 91:     if Length (gvstrVHFChannelDataArray[vbytTemp,gcbytChannelNameField]) > 0 then
 92:     begin
 93:       if gvstrVHFChannelDataArray[vbytTemp,gcbytToneField] = gcstrOn then
 94:         frmMem.sgrVHF.Cells[gcbytToneCTCSSCol, vbytTemp] := gcstrTMV7Tone
 95:       else if gvstrVHFChannelDataArray[vbytTemp,gcbytCTCSSField] = gcstrOn then
 96:         frmMem.sgrVHF.Cells[gcbytToneCTCSSCol, vbytTemp] := gcstrTMV7CTCSS
 97:       else frmMem.sgrVHF.Cells[gcbytToneCTCSSCol, vbytTemp] := gcstrTMV7None;
 98:     end;// if Length (gvstrVHFChannelDataArray[vbytTemp,gcbytChannelNameField]) > 0
 99:
100:     // Tone Freq
101:     // We only load the tone Frequency if the record is valid and a tone is selected
102:    if Length (gvstrVHFChannelDataArray[vbytTemp,gcbytChannelNameField]) > 0 then
103:    begin
104:      // We have a valid record we now check to see if there is a Tone or CTCSS on
105:      case frmMem.sgrVHF.Cells[gcbytToneCTCSSCol, vbytTemp] of
106:        gcstrTMV7Tone : begin
107:                          frmMem.sgrVHF.Cells[gcbytToneCTCSSFreqCol, vbytTemp] :=
108:                            GetToneFrequencyFromToneNr
109:                           (StrToInt (gvstrVHFChannelDataArray[vbytTemp,gcbytToneNrField]));
110:                        end;// gcstrTMV7Tone
111:        gcstrTMV7CTCSS : begin
112:                           frmMem.sgrVHF.Cells[gcbytToneCTCSSFreqCol, vbytTemp] :=
113:                             GetToneFrequencyFromToneNr
114:                              (StrToInt (gvstrVHFChannelDataArray[vbytTemp,gcbytCTCSSNrField]));
115:                        end;// gcstrTMV7CTCSS
116:      else // gcstrTMV7None
117:        frmMem.sgrVHF.Cells[gcbytToneCTCSSFreqCol, vbytTemp] := '';
```

```
118:      end;// case frmMem.sgrVHF.Cells[cbytToneCTCSSCol, vbytTemp]
119:
120:    end;// Length (gvstrVHFChannelDataArray[vbytTemp,gcbytChannelNameField]) > 0
121:
122:        // RF Power
123:     vstrTStr := gvstrVHFChannelDataArray[vbytTemp,gcbytRFPowerField];
124:     case vstrTStr of
125:       gcstrRFPowerLow : frmMem.sgrVHF.Cells[gcbytRFPowerCol, vbytTemp] := gcstrTMV7RFPowerLow;
126:       gcstrRFPowerMedium : frmMem.sgrVHF.Cells[gcbytRFPowerCol, vbytTemp] := gcstrTMV7
    RFPowerMedium;
127:       gcstrRFPowerHigh : frmMem.sgrVHF.Cells[gcbytRFPowerCol, vbytTemp] := gcstrTMV7RFPowerHigh
    ;
128:     end;//  case vstrTStr of
129:
130:     // DTSS
131:     vstrTStr := gvstrVHFChannelDataArray[vbytTemp,gcbytDTSSField];
132:     case vstrTStr of
133:       gcstrOn : frmMem.sgrVHF.Cells[gcbytDTSSCol, vbytTemp] := gcstrTMV7On;
134:       gcstrOff : frmMem.sgrVHF.Cells[gcbytDTSSCol, vbytTemp] := gcstrTMV7Off;
135:     else
136:       frmMem.sgrVHF.Cells[gcbytRFPowerCol, vbytTemp] := '';
137:     end;//  case vstrTStr of
138:
139:     // DTSS CODE
140:    if Length (gvstrVHFChannelDataArray[vbytTemp,gcbytChannelNameField]) > 0 then
141:    begin
142:       if gvstrVHFChannelDataArray[vbytTemp,gcbytDTSSField] = gcstrOn then
143:         frmMem.sgrVHF.Cells[gcbytDTSSCodeCol, vbytTemp] :=
144:                     gvstrVHFChannelDataArray[vbytTemp,gcbytDTSSCodeField]
145:       else
146:         frmMem.sgrVHF.Cells[gcbytDTSSCodeCol, vbytTemp] := '';
147:    end;// if Length (gvstrVHFChannelDataArray[vbytTemp,gcbytChannelNameField]) > 0
148:
149:     // REVERSE
150:     vstrTStr := gvstrVHFChannelDataArray[vbytTemp,gcbytReverseField];
151:     case vstrTStr of
152:       gcstrOn : frmMem.sgrVHF.Cells[gcbytReverseCol, vbytTemp] := gcstrTMV7On;
153:       gcstrOff : frmMem.sgrVHF.Cells[gcbytReverseCol, vbytTemp] := gcstrTMV7Off;
154:     else
155:       frmMem.sgrVHF.Cells[gcbytReverseCol, vbytTemp] := '';
156:     end;//  case vstrTStr of
157:
158:     // SCAN
159:     vstrTStr := gvstrVHFChannelDataArray[vbytTemp,gcbytScanField];
160:     case vstrTStr of
161:       gcstrOn : frmMem.sgrVHF.Cells[gcbytScanCol, vbytTemp] := gcstrTMV7On;
162:       gcstrOff : frmMem.sgrVHF.Cells[gcbytScanCol, vbytTemp] := gcstrTMV7Off;
163:     else
164:       frmMem.sgrVHF.Cells[gcbytScanCol, vbytTemp] := '';
165:     end;//  case vstrTStr of
166:
167:     // Step
168:     if Length (gvstrVHFChannelDataArray[vbytTemp,gcbytStepField]) > 0 then
169:       vstrTStr := gvstrVHFChannelDataArray[vbytTemp,gcbytStepField]
170:     else vstrTStr := '';
171:
172:     if Length(vstrTStr) > 0 then
173:       frmMem.sgrVHF.Cells[gcbytStepCol, vbytTemp] := gvstrStepArray[StrToInt(vstrTStr)]
174:     else
```

```
175:       frmMem.sgrVHF.Cells[gcbytStepCol, vbytTemp] := '';
176:
177:     // COMMENTS
178:     frmMem.sgrVHF.Cells[gcbytCommentCol, vbytTemp] :=
179:                     gvstrVHFChannelDataArray[vbytTemp,gcbytCommentsField];
180:
181:   end;// for vbytTemp := 1 to gcbytMaxVHFChannels do
182:
183: end;// procedure TfrmMEM.LoadVHFStringGrid;
184:
185: //---------------------------------------------------------------------------
186: procedure SetVHFChannel;
187: begin
188:
189:   // vbytChannelNr is the index into the gvstrFAVChannelDataArray table.
190:   // First we make sure that we have a valid data record at this position by ensuring
191:   // the Channel Name contains data (Mandatory field).
192:   if gvintSelectedRow = 0 then gvintSelectedRow := 1;
193:
194:   if Length ( gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytChannelNameField ] ) <
195:               gcbytMinChannelNameLength then
196:   begin
197:     showmessage('No Entry');
198:     Exit;
199:   end;// if Length ( gvstrFAVChannelDataArray
200:
201:   // Here we have a valid data record so we load the appropriate buffer based on the
202:   // VFO field
203:   gvstrVHFDataSource := 'MEM';
204:   gvstrVHFRXFrequency := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytRXFrequencyField ];
205:   gvstrVHFStep := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytStepField ];
206:   gvstrVHFShift := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytShiftField ];
207:   gvstrVHFReverse := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytReverseField ];
208:   gvstrVHFTone := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytToneField ];
209:   gvstrVHFCTCSS := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytCTCSSField ];
210:   gvstrVHFDTSS := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytDTSSField ];
211:   gvstrVHFToneNr := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytToneNrField ];
212:   gvstrVHFDTSSCode := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytDTSSCodeField ];
213:   gvstrVHFCTCSSNr := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytCTCSSNrField ];
214:   gvstrVHFOffset := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytShiftOffsetField ];
215:   gvstrVHFScan := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytScanField ];
216:   gvstrVHFRFPower := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytRFPowerField ];
217:   gvstrVHFChannelName := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytChannelNameField ];
218:   gvstrVHFChannelComments := gvstrVHFChannelDataArray[ gvintSelectedRow, gcbytCommentsField ];
219:
220:   //***      DisplayVHFBuffer;
221:
222:   SetBuffer(gcstrVHFVFO);
223:   SetVHFBand;
224:   UpdateLCDDisplay;
225:
226: end;// procedure SetVHFChannel;
227:
228: //===========================================================================
229: end.// unit MEM_VHF;
230:
```