



STARPRINT LIMITED

Source code exported to PDF

Project Name	BarcodeMacros Version ..
Student Name	Joginder S Nahil
Lecturer's Name	Dr. Aaron Mann
Short Project Filename	Access 2000 Barcode Macros.mdb
General Date	07/10/2007 13:59:17
Long Date	07 October 2007
Short Date	07/10/2007
Long Time	13:59:17
Short Time	13:59

```

00001  '*****
00002  ' Visual Basic & VBA Functions for IDAutomation Barcode Fonts V7.10
00003  ' © Copyright, 2000-2007 IDAutomation.com, Inc. All rights reserved.
00004  ' Redistribution and use of this code in source and/or binary
00005  ' forms, with or without modification, are permitted provided
00006  ' that: ( 1 ) all copies of the source code retain the above
00007  ' unmodified copyright notice and this entire unmodified
00008  ' section of text, ( 2 ) You or Your organization owns a valid
00009  ' Developer License to this product from IDAutomation.com
00010  ' and, ( 3 ) when any portion of this code is bundled in any
00011  ' form with an application, a valid notice must be provided
00012  ' within the user documentation, start-up screen or in the
00013  ' help-about section of the application that specifies
00014  ' IDAutomation.com as the provider of the Software bundled
00015  ' with the application.
00016  '*****
00017
00018  'START OF DECLARATIONS
00019  Dim I As Integer
00020  Dim J As Integer
00021  Dim F As Integer
00022  Dim DataToPrint As String
00023  Dim OnlyCorrectData As String
00024  Dim PrintableString As String
00025  Dim Encoding As String
00026  Dim WeightedTotal As Long
00027  Dim WeightValue As Integer
00028  Dim CurrentValue As Long
00029  Dim CheckDigitValue As Integer
00030  Dim Factor As Integer
00031  Dim CheckDigit As Integer
00032  Dim CurrentEncoding As String
00033  Dim NewLine As String
00034  Dim msg As String
00035  Dim CurrentChar As String
00036  Dim CurrentCharNum As Integer
00037  Dim C128__StartA As String
00038  Dim C128__StartB As String
00039  Dim C128__StartC As String
00040  Dim C128__Stop As String
00041  Dim C128Start As String
00042  Dim C128CheckDigit As String
00043  Dim StartCode As String
00044  Dim StopCode As String
00045  Dim Fnc1 As String
00046  Dim LeadingDigit As Integer
00047  Dim EAN2AddOn As String
00048  Dim EAN5AddOn As String
00049  Dim EANAddOnToPrint As String
00050  Dim HumanReadableText As String
00051  Dim StringLength As Integer
00052  Dim CorrectFNC As Integer
00053  'The DLL referenced below may be downloaded from www.idautomation.com/fonts/tools/windows__dll/
00054  'and is only required to create USPS Intelligent Mail symbols. It is not required for other barcode types.
00055  'The file is free to use and distribute with this module, provided You or Your organization
00056  'owns a valid Developer License for the associated IDAutomation product or Barcode Font.
00057  Public Declare Function IDAutomation__Universal__OneCode _
00058      Lib "IDAutomationNativeFontEncoder.dll" _
00059      ( ByVal D2E As String, _
00060      ByVal out As String, _
00061      ByVal iSize As Long ) As Long
00062  'END OF DECLARATIONS

```

```

00063
00064  Public Function Code128 ( DataToFormat As String, Optional ReturnType As Integer = 0, Optional ApplyTilde
00065  » As Boolean = False ) As String
00066
00067      Dim DataToEncode As String
00068      CorrectFNC = 0
00069      PrintableString = ""
00070
00071      'Additional logic needed in case ReturnType is not correct
00072      If ReturnType < 0 Or ReturnType > 5 Then ReturnType = 0
00073
00074      '2006.2 BDA moved code to the ProcessTilde function
00075      If ApplyTilde Then DataToFormat = ProcessTilde ( DataToFormat)
00076
00077

```

```

00077 1 If ReturnType = 0 Or ReturnType = 2 Then
00078     'ReturnType 0 = format the data to the font
00079     'Select the character set A, B or C for the START character
00080     CurrentChar = Left( DataToFormat, 1)
00081     CurrentCharNum = AscW( CurrentChar)
00082     'TB 8/1/07 moved the StringLength variable to initialize it before If statement
00083     StringLength = Len( DataToFormat)
00084     If CurrentCharNum < 32 Then C128Start = ChrW( 203)
00085     If CurrentCharNum > 31 And CurrentCharNum < 127 Then C128Start = ChrW( 204)
00086     If ( ( StringLength > 3) And IsNumeric( Mid( DataToFormat, 1, 4) ) ) Then C128Start = ChrW( 205)
    » )
00087
00088     '202 & 212-215 is for the FNC1, with this Start C is mandatory
00089     If CurrentCharNum = 197 Then C128Start = ChrW( 204)
00090     If CurrentCharNum > 201 Then C128Start = ChrW( 205)
00091     If C128Start = ChrW( 203) Then CurrentEncoding = "A"
00092     If C128Start = ChrW( 204) Then CurrentEncoding = "B"
00093     If C128Start = ChrW( 205) Then CurrentEncoding = "C"
00094     'StringLength = Len( DataToFormat)
00095     For I = 1 To StringLength
00096         'Check for FNC1 in any set which is ASCII 202 and ASCII 212-215
00097         CurrentCharNum = AscW( Mid( DataToFormat, I, 1) )
00098         If CurrentCharNum > 201 Then
00099             DataToEncode = DataToEncode & ChrW( 202)
00100             'Check for switching to character set C
00101         ElseIf CurrentCharNum = 195 Then
00102             If CurrentEncoding = "C" Then
00103                 DataToEncode = DataToEncode & ChrW( 200)
00104                 CurrentEncoding = "B"
00105             End If
00106             DataToEncode = DataToEncode & ChrW( 195)
00107         ElseIf CurrentCharNum = 196 Then
00108             If CurrentEncoding = "C" Then
00109                 DataToEncode = DataToEncode & ChrW( 200)
00110                 CurrentEncoding = "B"
00111             End If
00112             DataToEncode = DataToEncode & ChrW( 196)
00113         ElseIf CurrentCharNum = 197 Then
00114             If CurrentEncoding = "C" Then
00115                 DataToEncode = DataToEncode & ChrW( 200)
00116                 CurrentEncoding = "B"
00117             End If
00118             DataToEncode = DataToEncode & ChrW( 197)
00119         ElseIf CurrentCharNum = 198 Then
00120             If CurrentEncoding = "C" Then
00121                 DataToEncode = DataToEncode & ChrW( 200)
00122                 CurrentEncoding = "B"
00123             End If
00124             DataToEncode = DataToEncode & ChrW( 198)
00125         ElseIf CurrentCharNum = 200 Then
00126             If CurrentEncoding = "C" Then
00127                 DataToEncode = DataToEncode & ChrW( 200)
00128                 CurrentEncoding = "B"
00129             End If
00130             DataToEncode = DataToEncode & ChrW( 200)
00131         ElseIf ( ( I < StringLength - 2) And ( IsNumeric( Mid( DataToFormat, I, 1) ) ) And ( IsNumeric(
    » Mid( DataToFormat, I + 1, 1) ) ) And ( IsNumeric( Mid( DataToFormat, I, 4) ) ) ) Or ( ( I <
    » StringLength) And ( IsNumeric( Mid( DataToFormat, I, 1) ) ) And ( IsNumeric( Mid( DataToFormat,
    » I + 1, 1) ) ) And ( CurrentEncoding = "C" ) ) Then
00132             '2005.12 BDA added this section
00133             'check to see if we have an odd number of numbers to encode,
00134             'if so stay in current set for 1 number and then switch to save space
00135             If CurrentEncoding <> "C" Then
00136                 J = I
00137                 Factor = 3
00138                 Do While J <= StringLength And IsNumeric( Mid( DataToFormat, J, 1) )
00139                     Factor = 4 - Factor
00140                     J = J + 1
00141                 Loop
00142                 If Factor = 1 Then
00143                     'if so stay in current set for 1 character to save space
00144                     DataToEncode = DataToEncode & ChrW( CurrentCharNum)
00145                     I = I + 1
00146                 End If
00147             End If
00148
00149             'Switch to set C if not already in it

```

IDAAutomationVBA

```

00150 1 2 3 4 If CurrentEncoding <> "C" Then DataToEncode = DataToEncode & ChrW( 199)
00151 CurrentEncoding = "C"
00152 CurrentChar = ( Mid( DataToFormat, I, 2) )
00153 CurrentValue = CInt( CurrentChar)
00154 'Set the CurrentValue to the number of String CurrentChar
00155 If ( CurrentValue < 95 And CurrentValue > 0) Then DataToEncode = DataToEncode & ChrW(
» CurrentValue + 32)
00156 If CurrentValue > 94 Then DataToEncode = DataToEncode & ChrW( CurrentValue + 100)
00157 If CurrentValue = 0 Then DataToEncode = DataToEncode & ChrW( 194)
00158 I = I + 1
00159 'Check for switching to character set A
00160 ElseIf ( I <= StringLength) And ( ( AscW( Mid( DataToFormat, I, 1) ) ) < 31) Or ( (
» CurrentEncoding = "A") And ( AscW( Mid( DataToFormat, I, 1) ) ) > 32 And ( AscW( Mid(
» DataToFormat, I, 1) ) ) < 96) ) ) Then
00161 'Switch to set A if not already in it
00162 If CurrentEncoding <> "A" Then DataToEncode = DataToEncode & ChrW( 201)
00163 CurrentEncoding = "A"
00164 'Get the ASCII value of the next character
00165 CurrentCharNum = AscW( Mid( DataToFormat, I, 1) )
00166 If CurrentCharNum = 32 Then
00167 DataToEncode = DataToEncode & ChrW( 194)
00168 ElseIf CurrentCharNum < 32 Then
00169 DataToEncode = DataToEncode & ChrW( CurrentCharNum + 96)
00170 ElseIf CurrentCharNum > 32 Then
00171 DataToEncode = DataToEncode & ChrW( CurrentCharNum)
00172 End If
00173 'Check for switching to character set B
00174 ElseIf ( I <= StringLength) And ( ( AscW( Mid( DataToFormat, I, 1) ) ) ) > 31 And ( AscW( Mid
» ( DataToFormat, I, 1) ) ) < 127 Then
00175 'Switch to set B if not already in it
00176 If CurrentEncoding <> "B" Then DataToEncode = DataToEncode & ChrW( 200)
00177 CurrentEncoding = "B"
00178 'Get the ASCII value of the next character
00179 CurrentCharNum = AscW( Mid( DataToFormat, I, 1) )
00180 If CurrentCharNum = 32 Then
00181 DataToEncode = DataToEncode & ChrW( 194)
00182 Else
00183 DataToEncode = DataToEncode & ChrW( CurrentCharNum)
00184 End If
00185 End If
00186 Next I
00187 End If
00188 'FORMAT TEXT FOR AIs
00189 If Returntype = 1 Then
00190 'Returntype 1 = format the data for human readable text only
00191 HumanReadableText = ""
00192 StringLength = Len( DataToFormat)
00193 For I = 1 To StringLength
00194 CorrectFNC = 0
00195 'Get ASCII value of each character
00196 CurrentCharNum = AscW( Mid( DataToFormat, I, 1) )
00197 'Check for FNC1
00198 If ( ( I < StringLength - 2) And ( ( CurrentCharNum = 202) Or ( ( CurrentCharNum > 211) And
» ( CurrentCharNum < 219) ) ) ) Then
00200 '2005.12 BDA updated the next if/else to eliminate errors from text after the AI
00201 'It appears that there is an AI
00202 'Get the value of the next 2 digits to try to determine the length of the AI, if text is used here
00203 'Set this value to 81 for a 4 digit AI
00204 If IsNumeric( Mid( DataToFormat, I + 1, 1) ) And IsNumeric( Mid( DataToFormat, I + 2, 1) )
» Then
00205 CurrentChar = Mid( DataToFormat, I + 1, 2)
00206 CurrentCharNum = CInt( CurrentChar)
00207 Else
00208 CurrentCharNum = 81
00209 End If
00210 'Is 2 digit AI by entering ASCII 212?
00211 If ( ( CorrectFNC = 0) And ( AscW( Mid( DataToFormat, I, 1) ) = 212) ) Then
00212 HumanReadableText = HumanReadableText & " (" & ( Mid( DataToFormat, I + 1, 2) ) & ") "
00213 I = I + 2
00214 CorrectFNC = 1
00215 'Is 3 digit AI by entering ASCII 213?
00216 ElseIf ( ( I < StringLength - 3) And ( CorrectFNC = 0) And ( AscW( Mid( DataToFormat, I, 1)
» ) = 213) ) Then
00217 HumanReadableText = HumanReadableText & " (" & ( Mid( DataToFormat, I + 1, 3) ) & ") "
00218 I = I + 3
00219 CorrectFNC = 1

```

IDAutomationVBA

```

00220 1 2 3 4 5 'Is 4 digit AI by entering ASCII 214?
00221      ) Elseif ( ( I < StringLength - 4 ) And ( CorrectFNC = 0 ) And ( AscW( Mid( DataToFormat, I, 1)
»      ) = 214 ) Then
00222          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 4 ) ) & " ) "
00223          I = I + 4
00224          CorrectFNC = 1
00225          'Is 5 digit AI by entering ASCII 215?
00226      ) Elseif ( ( I < StringLength - 5 ) And ( CorrectFNC = 0 ) And ( AscW( Mid( DataToFormat, I, 1)
»      ) = 215 ) Then
00227          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 5 ) ) & " ) "
00228          I = I + 5
00229          CorrectFNC = 1
00230          'Is 6 digit AI by entering ASCII 216?
00231      ) Elseif ( ( I < StringLength - 6 ) And ( CorrectFNC = 0 ) And ( AscW( Mid( DataToFormat, I, 1)
»      ) = 216 ) Then
00232          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 6 ) ) & " ) "
00233          I = I + 6
00234          CorrectFNC = 1
00235          'Is 7 digit AI by entering ASCII 217?
00236      ) Elseif ( ( I < StringLength - 7 ) And ( CorrectFNC = 0 ) And ( AscW( Mid( DataToFormat, I, 1)
»      ) = 217 ) Then
00237          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 7 ) ) & " ) "
00238          I = I + 7
00239          CorrectFNC = 1
00240          'Is 8 digit AI by entering ASCII 218?
00241      ) Elseif ( ( I < StringLength - 8 ) And ( CorrectFNC = 0 ) And ( AscW( Mid( DataToFormat, I, 1)
»      ) = 218 ) Then
00242          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 8 ) ) & " ) "
00243          I = I + 8
00244          CorrectFNC = 1
00245          'Is 4 digit AI by detection?
00246      ) Elseif ( ( I < StringLength - 4 ) And ( CorrectFNC = 0 ) And ( ( CurrentCharNum <= 81 And
»      CurrentCharNum >= 80 ) Or ( CurrentCharNum <= 34 And CurrentCharNum >= 31 ) ) Then
00247          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 4 ) ) & " ) "
00248          I = I + 4
00249          CorrectFNC = 1
00250          'Is 3 digit AI by detection?
00251      ) Elseif ( ( I < StringLength - 3 ) And ( CorrectFNC = 0 ) And ( ( CurrentCharNum <= 49 And
»      CurrentCharNum >= 40 ) Or ( CurrentCharNum <= 25 And CurrentCharNum >= 23 ) ) ) Then
00252          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 3 ) ) & " ) "
00253          I = I + 3
00254          CorrectFNC = 1
00255          'Is 2 digit AI by detection?
00256      ) Elseif ( ( CurrentCharNum <= 30 And ( CorrectFNC = 0 ) And CurrentCharNum >= 0 ) Or (
»      CurrentCharNum <= 99 And CurrentCharNum >= 90 ) ) Then
00257          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 2 ) ) & " ) "
00258          I = I + 2
00259          CorrectFNC = 1
00260          'If no AI was detected, set default to 4 digit AI:
00261      ) Elseif ( ( I < StringLength - 4 ) And ( CorrectFNC = 0 ) ) Then
00262          HumanReadableText = HumanReadableText & " ( " & ( Mid( DataToFormat, I + 1, 4 ) ) & " ) "
00263          I = I + 4
00264          CorrectFNC = 1
00265      End If
00266      ) Elseif ( AscW( Mid( DataToFormat, I, 1 ) ) < 32 ) Then
00267          HumanReadableText = HumanReadableText & " "
00268      ) Elseif ( ( AscW( Mid( DataToFormat, I, 1 ) ) > 31 ) And ( AscW( Mid( DataToFormat, I, 1 ) ) <
»      128 ) ) Then
00269          HumanReadableText = HumanReadableText & Mid( DataToFormat, I, 1)
00270      End If
00271  Next I
00272 End If
00273
00274 If ReturnType > 2 Then
00275     'ReturnType 3, 4 or 5 = format the data for human readable text only
00276     'inserting a space for every 3, 4 or 5 characters
00277     HumanReadableText = ""
00278     StringLength = Len( DataToFormat)
00279     J = 0
00280     For I = 1 To StringLength
00281         CurrentCharNum = AscW( Mid( DataToFormat, I, 1 ) )
00282         If CurrentCharNum > 31 And CurrentCharNum < 128 Then
00283             HumanReadableText = HumanReadableText & Mid( DataToFormat, I, 1)
00284             J = J + 1
00285         End If
00286         If ( J Mod ReturnType ) = 0 Then HumanReadableText = HumanReadableText & " "

```

```

00287 1 2 3 Next I
00288 End If
00289
00290 If ReturnType = 0 Or ReturnType = 2 Then
00291 '2006.2 BDA added the if block here for compatibility with different office versions
00292 DataToFormat = ""
00293 'Calculate Modulo 103 Check Digit
00294 WeightedTotal = AscW( C128Start) - 100
00295 StringLength = Len( DataToEncode)
00296 For I = 1 To StringLength
00297 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00298 If CurrentCharNum < 135 Then CurrentValue = CurrentCharNum - 32
00299 If CurrentCharNum > 134 Then CurrentValue = CurrentCharNum - 100
00300 If CurrentCharNum = 194 Then CurrentValue = 0
00301 CurrentValue = CurrentValue * I
00302 WeightedTotal = WeightedTotal + CurrentValue
00303 If CurrentCharNum = 32 Then CurrentCharNum = 194
00304 PrintableString = PrintableString & ChrW( CurrentCharNum)
00305 Next I
00306 CheckDigitValue = ( WeightedTotal Mod 103)
00307 If CheckDigitValue < 95 And CheckDigitValue > 0 Then C128CheckDigit = ChrW( CheckDigitValue + 32
» )
00308 If CheckDigitValue > 94 Then C128CheckDigit = ChrW( CheckDigitValue + 100)
00309 If CheckDigitValue = 0 Then C128CheckDigit = ChrW( 194)
00310 End If
00311
00312 DataToEncode = ""
00313 'ReturnType 0 returns data formatted to the barcode font
00314 If ReturnType = 0 Then Code128 = C128Start & PrintableString & C128CheckDigit & ChrW( 206)
00315 'ReturnType 1 returns data formatted for human readable text
00316 If ReturnType = 1 Or ReturnType > 2 Then Code128 = HumanReadableText
00317 'ReturnType 2 returns the check digit for the data supplied
00318 If ReturnType = 2 Then Code128 = C128CheckDigit
00319 End Function

```

```

00320
00321
00322 Public Function Code128a( DataToEncode As String) As String
00323
00324 PrintableString = ""
00325 WeightedTotal = 103
00326 PrintableString = ChrW( 203)
00327 StringLength = Len( DataToEncode)
00328 For I = 1 To StringLength
00329 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00330 If CurrentCharNum < 135 Then CurrentValue = CurrentCharNum - 32
00331 If CurrentCharNum > 134 Then CurrentValue = CurrentCharNum - 100
00332 CurrentValue = CurrentValue * I
00333 WeightedTotal = WeightedTotal + CurrentValue
00334 If CurrentCharNum = 32 Then CurrentCharNum = 194
00335 PrintableString = PrintableString & ChrW( CurrentCharNum)
00336 Next I
00337 CheckDigitValue = ( WeightedTotal Mod 103)
00338 If CheckDigitValue < 95 And CheckDigitValue > 0 Then C128CheckDigit = ChrW( CheckDigitValue + 32)
00339 If CheckDigitValue > 94 Then C128CheckDigit = ChrW( CheckDigitValue + 100)
00340 If CheckDigitValue = 0 Then C128CheckDigit = ChrW( 194)
00341 PrintableString = PrintableString & C128CheckDigit & ChrW( 206)
00342 Code128a = PrintableString
00343 End Function

```

```

00344
00345
00346 Public Function Code128b( DataToEncode As String) As String
00347
00348 PrintableString = ""
00349 WeightedTotal = 104
00350 PrintableString = ChrW( 204)
00351 StringLength = Len( DataToEncode)
00352 For I = 1 To StringLength
00353 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00354 If CurrentCharNum < 135 Then CurrentValue = CurrentCharNum - 32
00355 If CurrentCharNum > 134 Then CurrentValue = CurrentCharNum - 100
00356 CurrentValue = CurrentValue * I
00357 WeightedTotal = WeightedTotal + CurrentValue
00358 If CurrentCharNum = 32 Then CurrentCharNum = 194
00359 PrintableString = PrintableString & ChrW( CurrentCharNum)
00360 Next I
00361 CheckDigitValue = ( WeightedTotal Mod 103)
00362 If CheckDigitValue < 95 And CheckDigitValue > 0 Then C128CheckDigit = ChrW( CheckDigitValue + 32)
00363 If CheckDigitValue > 94 Then C128CheckDigit = ChrW( CheckDigitValue + 100)
1

```

```

00363 1 If CheckDigitValue = 0 Then C128CheckDigit = ChrW( 194)
00364 PrintableString = PrintableString & C128CheckDigit & ChrW( 206)
00365 Code128b = PrintableString
00366 End Function

```

```

00367
00368
00369 Public Function Code128c( DataToEncode As String, Optional ReturnType As Integer = 0) As String
00370 'Additional logic needed in case ReturnType is not entered
00371 If ReturnType <> 0 And ReturnType <> 1 And ReturnType <> 2 Then ReturnType = 0
00372 PrintableString = ""
00373 OnlyCorrectData = ""
00374 StringLength = Len( DataToEncode)
00375 For I = 1 To StringLength
00376     If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData & Mid(
    > DataToEncode, I, 1)
00377 Next I
00378 DataToEncode = OnlyCorrectData
00379 If ( Len( DataToEncode) Mod 2) = 1 Then DataToEncode = "0" & DataToEncode
00380 PrintableString = ChrW( 205)
00381 WeightedTotal = 105
00382 WeightValue = 1
00383 StringLength = Len( DataToEncode)
00384 For I = 1 To StringLength Step 2
00385     CurrentValue = CInt( Mid( DataToEncode, I, 2) )
00386     If CurrentValue < 95 And CurrentValue > 0 Then PrintableString = PrintableString & ChrW( CurrentValue
    > + 32)
00387     If CurrentValue > 94 Then PrintableString = PrintableString & ChrW( CurrentValue + 100)
00388     If CurrentValue = 0 Then PrintableString = PrintableString & ChrW( 194)
00389     CurrentValue = CurrentValue * WeightValue
00390     WeightedTotal = WeightedTotal + CurrentValue
00391     WeightValue = WeightValue + 1
00392 Next I
00393 CheckDigitValue = ( WeightedTotal Mod 103)
00394 If CheckDigitValue < 95 And CheckDigitValue > 0 Then C128CheckDigit = ChrW( CheckDigitValue + 32)
00395 If CheckDigitValue > 94 Then C128CheckDigit = ChrW( CheckDigitValue + 100)
00396 If CheckDigitValue = 0 Then C128CheckDigit = ChrW( 194)
00397 If ReturnType = 0 Then Code128c = PrintableString & C128CheckDigit & ChrW( 206)
00398 If ReturnType = 1 Then Code128c = DataToEncode & CheckDigitValue
00399 If ReturnType = 2 Then Code128c = Str( CheckDigitValue)
00400 End Function

```

```

00401
00402
00403 Public Function I2of5( DataToEncode As String) As String
00404 DataToPrint = ""
00405 DataToEncode = RTrim( LTrim( DataToEncode) )
00406 OnlyCorrectData = ""
00407 'Check to make sure data is numeric and remove dashes, etc.
00408 StringLength = Len( DataToEncode)
00409 For I = 1 To StringLength
00410     'Add all numbers to OnlyCorrectData string
00411     '2006.2 BDA modified the next 3 lines for compatibility with different office versions
00412     'If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData &
    > Mid( DataToEncode, I, 1)
00413     CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00414     If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
    > DataToEncode, I, 1)
00415 Next I
00416 DataToEncode = OnlyCorrectData
00417 'Check for an even number of digits, add 0 if not even
00418 If ( Len( DataToEncode) Mod 2) = 1 Then DataToEncode = "0" & DataToEncode
00419 'Assign start and stop codes
00420 StartCode = ChrW( 203)
00421 StopCode = ChrW( 204)
00422 StringLength = Len( DataToEncode)
00423 For I = 1 To StringLength Step 2
00424     'Get the value of each number pair
00425     CurrentCharNum = Val( ( Mid( DataToEncode, I, 2) ) )
00426     'Get the ASCII value of CurrentChar
00427     If CurrentCharNum < 94 Then DataToPrint = DataToPrint & ChrW( CurrentCharNum + 33)
00428     If CurrentCharNum > 93 Then DataToPrint = DataToPrint & ChrW( CurrentCharNum + 103)
00429 Next I
00430 'Get Printable String
00431 PrintableString = StartCode + DataToPrint + StopCode
00432 'Return PrintableString
00433 I2of5 = PrintableString
00434 End Function

```

```

00435
00436
00437 Public Function Code39Mod43 ( DataToEncode As String, Optional ReturnTyp
00438 e As Integer = 0) As String
00439 'Additional logic needed in case ReturnTyp
00440 e is not correct
00441 If ReturnTyp
00442 e <> 0 And ReturnTyp
00443 e <> 1 And ReturnTyp
00444 e <> 2 Then ReturnTyp
00445 e = 0
00446 DataToEncode = UCase( DataToEncode)
00447 DataToPrint = ""
00448 OnlyCorrectData = ""
00449 'Only pass correct data
00450 StringLength = Len( DataToEncode)
00451 For I = 1 To StringLength
00452 'Get each character one at a time
00453 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00454 'Get the value of CurrentChar according to MOD43
00455 '0-9
00456 If CurrentCharNum < 58 And CurrentCharNum > 47 Then OnlyCorrectData = OnlyCorrectData & Mid(
00457 DataToEncode, I, 1)
00458 'A-Z
00459 If CurrentCharNum < 91 And CurrentCharNum > 64 Then OnlyCorrectData = OnlyCorrectData & Mid(
00460 DataToEncode, I, 1)
00461 'Space
00462 If CurrentCharNum = 32 Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
00463 '.
00464 If CurrentCharNum = 45 Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
00465 '.
00466 If CurrentCharNum = 46 Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
00467 '$
00468 If CurrentCharNum = 36 Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
00469 '/
00470 If CurrentCharNum = 47 Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
00471 '+
00472 If CurrentCharNum = 43 Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
00473 '%
00474 If CurrentCharNum = 37 Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
00475 Next I
00476 DataToEncode = OnlyCorrectData
00477 WeightedTotal = 0
00478 StringLength = Len( DataToEncode)
00479 For I = 1 To StringLength
00480 'Get each character one at a time
00481 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00482 'Get the value of CurrentChar according to MOD43
00483 '0-9
00484 If CurrentCharNum < 58 And CurrentCharNum > 47 Then CurrentValue = CurrentCharNum - 48
00485 'A-Z
00486 If CurrentCharNum < 91 And CurrentCharNum > 64 Then CurrentValue = CurrentCharNum - 55
00487 'Space
00488 If CurrentCharNum = 32 Then CurrentValue = 38
00489 '.
00490 If CurrentCharNum = 45 Then CurrentValue = 36
00491 '.
00492 If CurrentCharNum = 46 Then CurrentValue = 37
00493 '$
00494 If CurrentCharNum = 36 Then CurrentValue = 39
00495 '/
00496 If CurrentCharNum = 47 Then CurrentValue = 40
00497 '+
00498 If CurrentCharNum = 43 Then CurrentValue = 41
00499 '%
00500 If CurrentCharNum = 37 Then CurrentValue = 42
00501 'To print the barcode symbol representing a space it is necessary
00502 'to type or print "=" ( the equal character) instead of a space character.
00503 If CurrentCharNum = 32 Then CurrentCharNum = 61
00504 'Gather data to print
00505 DataToPrint = DataToPrint & ChrW( CurrentCharNum)
00506 'Add the values together
00507 WeightedTotal = WeightedTotal + CurrentValue
00508 Next I
00509 'Divide the WeightedTotal by 43 and get the remainder, this is the CheckDigit
00510 CheckDigitValue = ( WeightedTotal Mod 43)
00511 'Assign values to characters
00512 '0-9
00513 If CheckDigitValue < 10 Then CheckDigit = CheckDigitValue + 48
00514 'A-Z
00515 If CheckDigitValue < 36 And CheckDigitValue > 9 Then CheckDigit = CheckDigitValue + 55
00516 'Space
00517 If CheckDigitValue = 38 Then CheckDigit = 61
00518 '.

```

IDAutomationVBA

```

00511 1 If CheckDigitValue = 36 Then CheckDigit = 45
00512 .
00513 If CheckDigitValue = 37 Then CheckDigit = 46
00514 '$
00515 If CheckDigitValue = 39 Then CheckDigit = 36
00516 '/'
00517 If CheckDigitValue = 40 Then CheckDigit = 47
00518 '+'
00519 If CheckDigitValue = 41 Then CheckDigit = 43
00520 '%'
00521 If CheckDigitValue = 42 Then CheckDigit = 37
00522 'ReturnType 0 returns data formatted to the barcode font
00523 If ReturnType = 0 Then Code39Mod43 = "!" & DataToPrint & ChrW( CheckDigit) & "!" & " "
00524 'ReturnType 1 returns data formatted for human readable text
00525 If ReturnType = 1 Then Code39Mod43 = DataToPrint & ChrW( CheckDigit)
00526 'ReturnType 2 returns the check digit for the data supplied
00527 If ReturnType = 2 Then Code39Mod43 = ChrW( CheckDigit)
00528 End Function

00529
00530
00531 Public Function Code39( DataToEncode As String) As String
00532 DataToEncode = RTrim( LTrim( DataToEncode) )
00533 'Check for spaces in code
00534 StringLength = Len( DataToEncode)
00535 For I = 1 To StringLength
00536 'Get each character one at a time
00537 CurrentChar = ( Mid( DataToEncode, I, 1) )
00538 'To print the barcode symbol representing a space it is necessary
00539 'to type or print "=" ( the equal character) instead of a space character.
00540 If CurrentChar = " " Then CurrentChar = "="
00541 DataToPrint = DataToPrint & CurrentChar
00542 Next I
00543 'Get Printable String
00544 Code39 = "!" & DataToPrint & "!"
00545 End Function

00546
00547
00548 Public Function I2of5Mod10( DataToEncode As String, Optional ReturnType As Integer = 0) As String
00549 'Additional logic needed in case ReturnType is not entered
00550 If ReturnType <> 0 And ReturnType <> 1 And ReturnType <> 2 Then ReturnType = 0
00551 DataToPrint = ""
00552 OnlyCorrectData = ""
00553 'Check to make sure data is numeric and remove dashes, etc.
00554 StringLength = Len( DataToEncode)
00555 For I = 1 To StringLength
00556 'Add all numbers to OnlyCorrectData string
00557 '2006.2 BDA modified the next 3 lines for compatibility with different office versions
00558 'If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData &
» Mid( DataToEncode, I, 1)
00559 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00560 If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
» DataToEncode, I, 1)
00561 Next I
00562 DataToEncode = OnlyCorrectData
00563 'Calculate Check Digit
00564 Factor = 3
00565 WeightedTotal = 0
00566 For I = Len( DataToEncode) To 1 Step -1
00567 'Get the value of each number starting at the end
00568 CurrentCharNum = Mid( DataToEncode, I, 1)
00569 'Multiply by the weighting factor which is 3,1,3,1...
00570 'and add the sum together
00571 WeightedTotal = WeightedTotal + CurrentCharNum * Factor
00572 'Change factor for next calculation
00573 Factor = 4 - Factor
00574 Next I
00575 'Find the CheckDigit by finding the smallest number that = a multiple of 10
00576 I = ( WeightedTotal Mod 10)
00577 If I <> 0 Then
00578 CheckDigit = ( 10 - I)
00579 Else
00580 CheckDigit = 0
00581 End If
00582 'Add check digit
00583 DataToEncode = DataToEncode & CheckDigit
00584 'Check for an even number of digits, add 0 if not even
00585 If ( Len( DataToEncode) Mod 2) = 1 Then DataToEncode = "0" & DataToEncode

```

```

00586 1   StringLength = Len( DataToEncode)
00587   For I = 1 To StringLength Step 2
00588     'Get the value of each number pair
00589     CurrentCharNum = ( Mid( DataToEncode, I, 2) )
00590     'Get the ASCII value of CurrentChar
00591     If CurrentCharNum < 94 Then DataToPrint = DataToPrint & ChrW( CurrentCharNum + 33)
00592     If CurrentCharNum > 93 Then DataToPrint = DataToPrint & ChrW( CurrentCharNum + 103)
00593   Next I
00594   'Return Type 0 returns data formatted to the barcode font
00595   If Return Type = 0 Then I2of5Mod10 = ChrW( 203) & DataToPrint & ChrW( 204)
00596   'Return Type 1 returns data formatted for human readable text
00597   If Return Type = 1 Then I2of5Mod10 = DataToEncode
00598   'Return Type 2 returns the check digit for the data supplied
00599   If Return Type = 2 Then I2of5Mod10 = Str$( CheckDigit)
00600 End Function

00601
00602
00603 Public Function MSI( DataToEncode As String, Optional Return Type As Integer = 0) As String
00604   'Additional logic needed in case Return Type is not entered correctly
00605   If Return Type <> 0 And Return Type <> 1 And Return Type <> 2 Then Return Type = 0
00606   'The MSI encoding function will only accept digits. Any non-numeric characters
00607   'will be discarded
00608   Dim DataToPrint As String 'output for function
00609   Dim OnlyCorrectData As String 'Only numeric characters pulled from DataToEncode
00610   Dim StringLength As Long 'Length of string
00611   Dim Idx As Integer 'for loop counter
00612   Dim OddNumbers As String 'String of odd position numbers used to create check digit
00613   Dim EvenNumberSum As Long 'all of the even position numbers added up
00614   Dim OddNumberProduct As Long 'Product of OddNumbers variable
00615   Dim sOddNumberProduct As String 'String version of OddNumberProduct variable
00616   Dim OddNumberSum As Long 'Sum of individual digits in sOddNumberProduct
00617   Dim OddDigit As Boolean 'Used to determine even/odd position digits.
00618   Dim CheckDigit As String 'This is the CheckDigit
00619   DataToPrint = ""
00620   OnlyCorrectData = ""
00621   'Check to make sure data is numeric
00622   StringLength = Len( DataToEncode)
00623   For I = 1 To StringLength
00624     'Add all numbers to OnlyCorrectData string
00625     '2006.2 BDA modified the next 3 lines for compatibility with different office versions
00626     'If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData &
» Mid( DataToEncode, I, 1)
00627     CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00628     If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
» DataToEncode, I, 1)
00629   Next I
00630   DataToPrint = OnlyCorrectData
00631   '<<<< Calculate Check Digit >>>>'
00632   'To create the check digit follow these steps:
00633   '1) Starting from the units position, create a new number with all of the odd
00634   ' position digits in their original sequence.
00635   '2) Multiply this new number by 2.
00636   '3) Add all of the digits of the product from step two.
00637   '4) Add all of the digits not used in step one to the result in step three.
00638   '5) Determine the smallest number which when added to the result in step four
00639   ' will result in a multiple of 10. This is the check character.
00640   'Step 1 -- Create a new number of the odd position digits starting from the right and going left, but store the
00641   'digits from left to right.
00642   'Create the odd position number & prepare for Step 4 by getting the sum of all even position characters
00643   StringLength = Len( DataToEncode)
00644   OddNumbers = ""
00645   OddDigit = True
00646   EvenNumberSum = 0
00647   For Idx = StringLength To 1 Step -1
00648     If OddDigit = True Then
00649       OddNumbers = Mid( DataToEncode, Idx, 1) & OddNumbers
00650       OddDigit = False
00651     Else
00652       EvenNumberSum = EvenNumberSum + Val( Mid( DataToEncode, Idx, 1) )
00653       OddDigit = True
00654     End If
00655   Next Idx
00656   'Step 2 -- Multiply this new number by 2.
00657   OddNumberProduct = Val( OddNumbers) * 2
00658   'Step 3 -- Add all of the digits of the product from step two.
00659   sOddNumberProduct = Format( OddNumberProduct)
00660   StringLength = Len( sOddNumberProduct)

```

```

00661 1   OddNumberSum = 0
00662   For Idx = 1 To StringLength
00663     OddNumberSum = OddNumberSum + Val( Mid( sOddNumberProduct, Idx, 1) )
00664   Next Idx
00665   'Step 4 -- Add all of the digits not used in step one to the result in step three.
00666   'We will store the result in OddNumberSum just so we don't have to create another variable
00667   OddNumberSum = OddNumberSum + EvenNumberSum
00668   'Step 5 -- Determine the smallest number which when added to the result in step four
00669   'will result in a multiple of 10. This is the check character.
00670   OddNumberSum = OddNumberSum Mod 10
00671   If OddNumberSum <> 0 Then
00672     CheckDigit = Format( 10 - OddNumberSum)
00673   Else
00674     CheckDigit = "0"
00675   End If
00676   Select Case Return Type
00677   Case 0 'Returns formatted data for barcode
00678     DataToPrint = "(" & DataToEncode & CheckDigit & ")"
00679   Case 1 'Returns data formatted for human readable text.
00680     'Which means all of the invalid characters are
00681     'stripped out.
00682     DataToPrint = DataToEncode
00683   Case 2 'Returns just the check digit
00684     DataToPrint = CheckDigit
00685   End Select
00686   MSI = DataToPrint
00687 End Function

00688
00689
00690 Public Function UPCa( DataToEncode As String) As String
00691   DataToPrint = ""
00692   OnlyCorrectData = ""
00693   'Check to make sure data is numeric and remove dashes, etc.
00694   StringLength = Len( DataToEncode)
00695   For I = 1 To StringLength
00696     'Add all numbers to OnlyCorrectData string
00697     '2006.2 BDA modified the next 3 lines for compatibility with different office versions
00698     'If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData &
»   Mid( DataToEncode, I, 1)
00699     CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00700     If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
»   DataToEncode, I, 1)
00701   Next I
00702   '2006.2 BDA added the next line for general compatibility
00703   StringLength = Len( OnlyCorrectData)
00704   'Remove check digits if they added one
00705   If StringLength < 11 Then OnlyCorrectData = "00000000000"
00706   If StringLength = 15 Then OnlyCorrectData = "00000000000"
00707   If StringLength > 18 Then OnlyCorrectData = "00000000000"
00708   If StringLength = 12 Then OnlyCorrectData = Mid( OnlyCorrectData, 1, 11)
00709   If StringLength = 14 Then OnlyCorrectData = Mid( OnlyCorrectData, 1, 11) & Mid( OnlyCorrectData, 13, 2
»   )
00710   If StringLength = 17 Then OnlyCorrectData = Mid( OnlyCorrectData, 1, 11) & Mid( OnlyCorrectData, 13, 5
»   )
00711   EAN2AddOn = ""
00712   EAN5AddOn = ""
00713   EANAddOnToPrint = ""
00714   '2006.2 BDA added the next line for general compatibility
00715   StringLength = Len( OnlyCorrectData)
00716   If StringLength = 16 Then EAN5AddOn = Mid( OnlyCorrectData, 12, 5)
00717   If StringLength = 13 Then EAN2AddOn = Mid( OnlyCorrectData, 12, 2)
00718   'split 12 digit number from add-on
00719
00720   DataToEncode = Mid( OnlyCorrectData, 1, 11)
00721   '<<<< Calculate Check Digit >>>>
00722   Factor = 3
00723   WeightedTotal = 0
00724   For I = Len( DataToEncode) To 1 Step -1
00725     'Get the value of each number starting at the end
00726     CurrentCharNum = Mid( DataToEncode, I, 1)
00727     'multiply by the weighting factor which is 3,1,3,1...
00728     'and add the sum together
00729     WeightedTotal = WeightedTotal + CurrentCharNum * Factor
00730     'change factor for next calculation
00731     Factor = 4 - Factor
00732   Next I
00733   'Find the CheckDigit by finding the number + WeightedTotal that = a multiple of 10

```

```

00734 1 'Divide by 10, get the remainder and subtract from 10
00735 I = ( WeightedTotal Mod 10)
00736 'If I <> 0 Then
00737     CheckDigit = ( 10 - I)
00738 } Else
00739     CheckDigit = 0
00740 } End If
00741 DataToEncode = DataToEncode & CheckDigit
00742 'Now that have the total number including the check digit, determine character to print
00743 'for proper barcoding
00744 StringLength = Len( DataToEncode)
00745 For I = 1 To StringLength
00746     'Get the ASCII value of each number
00747     CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00748     'Print different barcodes according to the location of the CurrentChar
00749     Select Case I
00750     Case 1
00751         'For the first character, print the human readable character, the normal
00752         'guard pattern, and then the barcode without the human readable character
00753         '2006.2 BDA modified the next 2 lines for general compatibility
00754         If ( CurrentCharNum - 48) > 4 Then DataToPrint = ChrW( CurrentCharNum + 64) & "(" & ChrW(
» CurrentCharNum + 49)
00755         If ( CurrentCharNum - 48) < 5 Then DataToPrint = ChrW( CurrentCharNum + 37) & "(" & ChrW(
» CurrentCharNum + 49)
00756     Case 2
00757         DataToPrint = DataToPrint & ChrW( CurrentCharNum)
00758     Case 3
00759         DataToPrint = DataToPrint & ChrW( CurrentCharNum)
00760     Case 4
00761         DataToPrint = DataToPrint & ChrW( CurrentCharNum)
00762     Case 5
00763         DataToPrint = DataToPrint & ChrW( CurrentCharNum)
00764     Case 6
00765         'Print the center guard pattern after the 6th character
00766         DataToPrint = DataToPrint & ChrW( CurrentCharNum) & "*"
00767     Case 7
00768         'Add 27 to the ASCII value of characters 6-12 to print from character set C
00769         'This is required when printing to the right of the center guard pattern
00770         DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
00771     Case 8
00772         DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
00773     Case 9
00774         DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
00775     Case 10
00776         DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
00777     Case 11
00778         DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
00779     Case 12
00780         'For the last character, print the barcode without the human readable character,
00781         'the normal guard pattern, and then the human readable character.
00782         '2006.2 BDA modified the next 2 lines for general compatibility
00783         If ( CurrentCharNum - 48) > 4 Then DataToPrint = DataToPrint & ChrW( CurrentCharNum + 59) &
» "(" & ChrW( CurrentCharNum + 64)
00784         If ( CurrentCharNum - 48) < 5 Then DataToPrint = DataToPrint & ChrW( CurrentCharNum + 59) &
» "(" & ChrW( CurrentCharNum + 37)
00785     End Select
00786 Next I
00787 'Process add-ons if they exist
00788 If Len( EAN2AddOn) = 2 Then DataToPrint = DataToPrint & ProcessEAN2AddOn( EAN2AddOn)
00789 If Len( EAN5AddOn) = 5 Then DataToPrint = DataToPrint & ProcessEAN5AddOn( EAN5AddOn)
00790 'Return PrintableString
00791 UPCa = DataToPrint
00792 End Function

00793
00794 Private Function UPCe7To11( DataToExpand As String) As String
00795     StringLength = Len( DataToExpand)
00796     If StringLength = 6 Then DataToExpand = "0" & DataToExpand
00797     'Expect 7 digits; the first digit is the number system
00798     If StringLength <> 7 Then DataToExpand = "0000000"
00799     Dim D1 As String
00800     Dim D2 As String
00801     Dim D3 As String
00802     Dim D4 As String
00803     Dim D5 As String
00804     Dim D6 As String
00805     Dim D7 As String
00806     D1 = Mid( DataToExpand, 1, 1)

```

```

00807 1 D2 = Mid( DataToExpand, 2, 1)
00808 D3 = Mid( DataToExpand, 3, 1)
00809 D4 = Mid( DataToExpand, 4, 1)
00810 D5 = Mid( DataToExpand, 5, 1)
00811 D6 = Mid( DataToExpand, 6, 1)
00812 D7 = Mid( DataToExpand, 7, 1)
00813 Select Case D7
00814 { Case "0"
00815     UPCe7To11 = D1 & D2 & D3 & "00000" & D4 & D5 & D6
00816 { Case "1"
00817     UPCe7To11 = D1 & D2 & D3 & D7 & "0000" & D4 & D5 & D6
00818 { Case "2"
00819     UPCe7To11 = D1 & D2 & D3 & D7 & "0000" & D4 & D5 & D6
00820 { Case "3"
00821     UPCe7To11 = D1 & D2 & D3 & D4 & "00000" & D5 & D6
00822 { Case "4"
00823     UPCe7To11 = D1 & D2 & D3 & D4 & D5 & "00000" & D6
00824 { Case "5"
00825     UPCe7To11 = D1 & D2 & D3 & D4 & D5 & D6 & "0000" & D7
00826 { Case "6"
00827     UPCe7To11 = D1 & D2 & D3 & D4 & D5 & D6 & "0000" & D7
00828 { Case "7"
00829     UPCe7To11 = D1 & D2 & D3 & D4 & D5 & D6 & "0000" & D7
00830 { Case "8"
00831     UPCe7To11 = D1 & D2 & D3 & D4 & D5 & D6 & "0000" & D7
00832 { Case "9"
00833     UPCe7To11 = D1 & D2 & D3 & D4 & D5 & D6 & "0000" & D7
00834 End Select
00835 '2006.2 BDA modified the next line for compatibility with different office versions
00836 StringLength = Len( UPCe7To11)
00837 End Function

00838
00839 Public Function UPCe( DataToEncode As String) As String
00840 OnlyCorrectData = ""
00841 DataToPrint = ""
00842 ' Check to make sure data is numeric and remove dashes, etc.
00843 StringLength = Len( DataToEncode)
00844 For I = 1 To StringLength
00845     'Add all numbers to OnlyCorrectData string
00846     '2006.2 BDA modified the next 3 lines for compatibility with different office versions
00847     'If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData &
00848     Mid( DataToEncode, I, 1)
00849     CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
00849     If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
00850     DataToEncode, I, 1)
00851 Next I
00851 'If UPCe is not expanded, then expand
00852 '2006.2 BDA added the next 4 lines for compatibility with different office versions
00853 StringLength = Len( OnlyCorrectData)
00854 If StringLength = 6 Then OnlyCorrectData = UPCe7To11( "0" & OnlyCorrectData)
00855 If StringLength = 7 Then OnlyCorrectData = UPCe7To11( OnlyCorrectData)
00856 If StringLength = 8 Then OnlyCorrectData = UPCe7To11( Mid( OnlyCorrectData, 1, 7) )
00857
00858 '2006.2 BDA added the next line for compatibility with different office versions
00859 StringLength = Len( OnlyCorrectData)
00860 If StringLength < 11 Then OnlyCorrectData = "00005000000"
00861 If StringLength = 15 Then OnlyCorrectData = "00005000000"
00862 If StringLength > 18 Then OnlyCorrectData = "00005000000"
00863 If StringLength = 12 Then OnlyCorrectData = Mid( OnlyCorrectData, 1, 11)
00864 If StringLength = 14 Then OnlyCorrectData = ( Mid( OnlyCorrectData, 1, 11) & Mid( OnlyCorrectData, 13,
00865 2) )
00865 If StringLength = 17 Then OnlyCorrectData = ( Mid( OnlyCorrectData, 1, 11) & Mid( OnlyCorrectData, 13,
00866 5) )
00866 EAN2AddOn = ""
00867 EAN5AddOn = ""
00868 EANAddOnToPrint = ""
00869 '2006.2 BDA added the next line for compatibility with different office versions
00870 StringLength = Len( OnlyCorrectData)
00871 If StringLength = 16 Then EAN5AddOn = Mid( OnlyCorrectData, 12, 5)
00872 If StringLength = 13 Then EAN2AddOn = Mid( OnlyCorrectData, 12, 2)
00873 'split 12 digit number from add-on
00874 DataToEncode = Mid( OnlyCorrectData, 1, 11)
00875 'Calculate Check Digit
00876 Factor = 3
00877 WeightedTotal = 0
00878 For I = Len( DataToEncode) To 1 Step -1
00879     'Get the value of each number starting at the end

```

```

00880 1 2 CurrentCharNum = Mid( DataToEncode, I, 1)
00881 'Multiply by the weighting factor which is 3,1,3,1...
00882 'and add the sum together
00883 WeightedTotal = WeightedTotal + CurrentCharNum * Factor
00884 'Change the factor for next calculation
00885 Factor = 4 - Factor
00886 Next I
00887 'Find the CheckDigit by finding the number + WeightedTotal that = a multiple of 10
00888 'Divide by 10, get the remainder and subtract from 10
00889 I = ( WeightedTotal Mod 10)
00890 If I <> 0 Then
00891     CheckDigit = ( 10 - I)
00892 Else
00893     CheckDigit = 0
00894 End If
00895 DataToEncode = DataToEncode & CheckDigit
00896 'Compress UPC-A to UPC-E if possible
00897 Dim D1 As String
00898 Dim D2 As String
00899 Dim D3 As String
00900 Dim D4 As String
00901 Dim D5 As String
00902 Dim D6 As String
00903 Dim D7 As String
00904 Dim D8 As String
00905 Dim D9 As String
00906 Dim D10 As String
00907 Dim D11 As String
00908 Dim D12 As String
00909 D1 = Mid( DataToEncode, 1, 1)
00910 D2 = Mid( DataToEncode, 2, 1)
00911 D3 = Mid( DataToEncode, 3, 1)
00912 D4 = Mid( DataToEncode, 4, 1)
00913 D5 = Mid( DataToEncode, 5, 1)
00914 D6 = Mid( DataToEncode, 6, 1)
00915 D7 = Mid( DataToEncode, 7, 1)
00916 D8 = Mid( DataToEncode, 8, 1)
00917 D9 = Mid( DataToEncode, 9, 1)
00918 D10 = Mid( DataToEncode, 10, 1)
00919 D11 = Mid( DataToEncode, 11, 1)
00920 D12 = Mid( DataToEncode, 12, 1)
00921 '2005.12 BDA updated the next line
00922 If D1 = "0" Or D1 = "1" Then
00923     'Condition A
00924     'EX: 02345600007
00925     If ( D11 = "5" Or D11 = "6" Or D11 = "7" Or D11 = "8" Or D11 = "9" ) And D6 <> "0" And ( D7 = "0" And
> DB = "0" And D9 = "0" And D10 = "0" ) Then
00926         DataToEncode = D2 & D3 & D4 & D5 & D6 & D11
00927     End If
00928     'Condition B
00929     'EX: 02345000001
00930     If ( D6 = "0" And D7 = "0" And D8 = "0" And D9 = "0" And D10 = "0" ) And D5 <> "0" Then
00931         DataToEncode = D2 & D3 & D4 & D5 & D11 & "4"
00932     End If
00933     'Condition C
00934     'EX: 06320000971
00935     If ( D5 = "0" And D6 = "0" And D7 = "0" And D8 = "0" ) And ( D4 = "1" Or D4 = "2" Or D4 = "0" ) Then
00936         DataToEncode = D2 & D3 & D9 & D10 & D11 & D4
00937     End If
00938     'Condition D
00939     'EX: 08670000093
00940     If ( D5 = "0" And D6 = "0" And D7 = "0" And D8 = "0" And D9 = "0" ) And ( D4 = "3" Or D4 = "4" Or D4 =
> "5" Or D4 = "6" Or D4 = "7" Or D4 = "8" Or D4 = "9" ) Then
00941         DataToEncode = D2 & D3 & D4 & D10 & D11 & "3"
00942     End If
00943 End If
00944 'Run UPC-E compression only if DataToEncode = 6
00945 If Len( DataToEncode ) = 6 Then
00946     '2005.12 BDA updated this section for number system 1 compatibility
00947     'Encode the check character into the symbol
00948     'by using variable parity between character sets A and B
00949     'The UPC-E starts with a 0 or 1 which indicates the number system
00950     'Number system is 1 only if first digit is 1
00951     If D1 = "1" Then
00952         Select Case D12
00953         Case "0"
00954             Encoding = "AAABBB"
00955         Case "1"

```

IDAutomationVBA

```

00956 1 2 3 4 Encoding = "AABABB"
00957      } Case "2"
00958      Encoding = "AABBAB"
00959      } Case "3"
00960      Encoding = "AABBBA"
00961      } Case "4"
00962      Encoding = "ABAABB"
00963      } Case "5"
00964      Encoding = "ABBAAB"
00965      } Case "6"
00966      Encoding = "ABBBAA"
00967      } Case "7"
00968      Encoding = "ABABAB"
00969      } Case "8"
00970      Encoding = "ABABBA"
00971      } Case "9"
00972      Encoding = "ABBABA"
00973      End Select
00974  } Else
00975      'Number system 0
00976      D1 = "0"
00977      Select Case D12
00978      } Case "0"
00979      Encoding = "BBBAAA"
00980      } Case "1"
00981      Encoding = "BBABAA"
00982      } Case "2"
00983      Encoding = "BBAABA"
00984      } Case "3"
00985      Encoding = "BBAAAB"
00986      } Case "4"
00987      Encoding = "BABBBAA"
00988      } Case "5"
00989      Encoding = "BAABBA"
00990      } Case "6"
00991      Encoding = "BAAABB"
00992      } Case "7"
00993      Encoding = "BABABA"
00994      } Case "8"
00995      Encoding = "BABAAB"
00996      } Case "9"
00997      Encoding = "BAABAB"
00998      End Select
00999  End If
01000  StringLength = Len( DataToEncode)
01001  For I = 1 To StringLength
01002      'Get the ASCII value of each number
01003      CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01004      CurrentEncoding = Mid( Encoding, I, 1)
01005      Select Case CurrentEncoding
01006      } Case "A"
01007      DataToPrint = DataToPrint & ChrW( CurrentCharNum)
01008      } Case "B"
01009      DataToPrint = DataToPrint & ChrW( CurrentCharNum + 17)
01010      End Select
01011      'Add in the 1st character along with guard patterns at the correct locations
01012      Select Case I
01013      } Case 1
01014      'For the LeadingDigit print the human readable character of the number system,
01015      'the normal guard pattern and then the rest of the barcode
01016      If D1 = "0" Then DataToPrint = ChrW( 85) & "(" & DataToPrint
01017      If D1 = "1" Then DataToPrint = ChrW( 86) & "(" & DataToPrint
01018      } Case 6
01019      'Print the SPECIAL guard pattern and check character
01020      If Cint( D12) > 4 Then DataToPrint = DataToPrint & ")" & ChrW( AscW( D12) + 64)
01021      If Cint( D12) < 5 Then DataToPrint = DataToPrint & ")" & ChrW( AscW( D12) + 37)
01022      End Select
01023      Next I
01024  End If
01025
01026  If Len( DataToEncode) <> 6 Then
01027      StringLength = Len( DataToEncode)
01028      For I = 1 To StringLength
01029          'Get the ASCII value of each number
01030          CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01031          Select Case I
01032          } Case 1
01033          'For the first character, print the human readable character, the normal

```

IDAutomationVBA

```

01034 1 2 3 4 'guard pattern, and then the barcode without the human readable character
01035 If ChrW( CurrentCharNum) > 4 Then DataToPrint = ChrW( CurrentCharNum + 64) & "( " &
» ChrW( CurrentCharNum + 49)
01036 If ChrW( CurrentCharNum) < 5 Then DataToPrint = ChrW( CurrentCharNum + 37) & "( " &
» ChrW( CurrentCharNum + 49)
01037 Case 2
01038 DataToPrint = DataToPrint & ChrW( CurrentCharNum)
01039 Case 3
01040 DataToPrint = DataToPrint & ChrW( CurrentCharNum)
01041 Case 4
01042 DataToPrint = DataToPrint & ChrW( CurrentCharNum)
01043 Case 5
01044 DataToPrint = DataToPrint & ChrW( CurrentCharNum)
01045 Case 6
01046 'Print the center guard pattern after the 6th character
01047 DataToPrint = DataToPrint & ChrW( CurrentCharNum) & ""
01048 Case 7
01049 'Add 27 to the ASCII value of characters 6-12 to print from character set C
01050 'This is required when printing to the right of the center guard pattern
01051 DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01052 Case 8
01053 DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01054 Case 9
01055 DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01056 Case 10
01057 DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01058 Case 11
01059 DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01060 Case 12
01061 'For the last character, print the barcode without the human readable character,
01062 'the normal guard pattern, and then the human readable character.
01063 If ChrW( CurrentCharNum) > 4 Then DataToPrint = DataToPrint & ChrW( CurrentCharNum + 59)
» ) & "( " & ChrW( CurrentCharNum + 64)
01064 If ChrW( CurrentCharNum) < 5 Then DataToPrint = DataToPrint & ChrW( CurrentCharNum + 59)
» ) & "( " & ChrW( CurrentCharNum + 37)
01065 End Select
01066 Next I
01067 End If
01068
01069 'Process add-ons if they exist
01070 If Len( EAN2AddOn) = 2 Then DataToPrint = DataToPrint & ProcessEAN2AddOn( EAN2AddOn)
01071 If Len( EAN5AddOn) = 5 Then DataToPrint = DataToPrint & ProcessEAN5AddOn( EAN5AddOn)
01072 'Return PrintableString
01073 UPCe = DataToPrint
01074 End Function

01075
01076 Public Function EAN13( DataToEncode As String) As String
01077 DataToPrint = ""
01078 OnlyCorrectData = ""
01079 'Check to make sure data is numeric and remove dashes, etc.
01080 StringLength = Len( DataToEncode)
01081 For I = 1 To StringLength
01082 'Add all numbers to OnlyCorrectData string
01083 '2006.2 BDA modified the next 3 lines for compatibility with different office versions
01084 'If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData &
» Mid( DataToEncode, I, 1)
01085 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01086 If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
» DataToEncode, I, 1)
01087 Next I
01088 '2006.2 BDA added the next line for general compatibility
01089 StringLength = Len( OnlyCorrectData)
01090 If StringLength < 12 Then OnlyCorrectData = "000000000000"
01091 If StringLength = 16 Then OnlyCorrectData = "000000000000"
01092 If StringLength = 13 Then OnlyCorrectData = Mid( OnlyCorrectData, 1, 12)
01093 If StringLength = 15 Then OnlyCorrectData = ( Mid( OnlyCorrectData, 1, 12) & Mid( OnlyCorrectData, 14,
» 2) )
01094 If StringLength > 17 Then OnlyCorrectData = ( Mid( OnlyCorrectData, 1, 12) & Mid( OnlyCorrectData, 14,
» 5) )
01095 Dim EAN2AddOn As String
01096 Dim EAN5AddOn As String
01097 EAN2AddOn = ""
01098 EAN5AddOn = ""
01099 '2006.2 BDA added the next line for general compatibility
01100 StringLength = Len( OnlyCorrectData)
01101 If StringLength = 17 Then EAN5AddOn = Mid( OnlyCorrectData, 13, 5)

```

```

01102 1 If StringLength = 14 Then EAN2AddOn = Mid( OnlyCorrectData, 13, 2)
01103 'Remove digit number from add-ons and check digit
01104 DataToEncode = Mid( OnlyCorrectData, 1, 12)
01105 'Calculate Check Digit
01106 Factor = 3
01107 WeightedTotal = 0
01108 For I = Len( DataToEncode) To 1 Step -1
01109 'Get the value of each number starting at the end
01110 CurrentCharNum = Mid( DataToEncode, I, 1)
01111 'Multiply by the weighting factor which is 3,1,3,1...
01112 'and add the sum together
01113 WeightedTotal = WeightedTotal + CurrentCharNum * Factor
01114 'Change factor for next calculation
01115 Factor = 4 - Factor
01116 Next I
01117 'Find the CheckDigit by finding the number + WeightedTotal that = a multiple of 10
01118 'Divide by 10, get the remainder and subtract from 10
01119 I = ( WeightedTotal Mod 10)
01120 If I <> 0 Then
01121 CheckDigit = ( 10 - I)
01122 Else
01123 CheckDigit = 0
01124 End If
01125 'Encode the leading digit into the left half of the EAN-13 symbol
01126 'by using variable parity between character sets A and B
01127 LeadingDigit = Mid( DataToEncode, 1, 1)
01128 Select Case LeadingDigit
01129 Case 0
01130 Encoding = "AAAAAACCCCC"
01131 Case 1
01132 Encoding = "AABABBCCCCC"
01133 Case 2
01134 Encoding = "AABBABCCCCC"
01135 Case 3
01136 Encoding = "AABBBACCCCC"
01137 Case 4
01138 Encoding = "ABAABBCCCCC"
01139 Case 5
01140 Encoding = "ABBAABCCCCC"
01141 Case 6
01142 Encoding = "ABBBAACCCCC"
01143 Case 7
01144 Encoding = "ABABABCCCCC"
01145 Case 8
01146 Encoding = "ABABBACCCCC"
01147 Case 9
01148 Encoding = "ABBABACCCCC"
01149 End Select
01150 'Add the check digit to the end of the barcode & remove the leading digit
01151 DataToEncode = Mid( DataToEncode, 2, 11) & CheckDigit
01152 'Determine character to print for proper barcoding
01153 StringLength = Len( DataToEncode)
01154 For I = 1 To StringLength
01155 'Get the ASCII value of each number excluding the first number because
01156 'it is encoded with variable parity
01157 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01158 CurrentEncoding = Mid( Encoding, I, 1)
01159 'Print different barcodes according to the location of the CurrentChar and CurrentEncoding
01160 Select Case CurrentEncoding
01161 Case "A"
01162 DataToPrint = DataToPrint & ChrW( CurrentCharNum)
01163 Case "B"
01164 DataToPrint = DataToPrint & ChrW( CurrentCharNum + 17)
01165 Case "C"
01166 DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01167 End Select
01168 'Add in the 1st character along with guard patterns
01169 Select Case I
01170 Case 1
01171 'For the LeadingDigit, print the human readable character,
01172 'the normal guard pattern, and then the rest of the barcode
01173 If LeadingDigit > 4 Then DataToPrint = ChrW( ( LeadingDigit + 48) + 64) & "( " & DataToPrint
01174 If LeadingDigit < 5 Then DataToPrint = ChrW( ( LeadingDigit + 48) + 37) & "( " & DataToPrint
01175 Case 6
01176 'Print the center guard pattern after the 6th character
01177 DataToPrint = DataToPrint & "***"
01178 Case 12
01179 'For the last character ( 12) , print the the normal guard pattern after the barcode

```

1 2 3

IDAutomationVBA

```

01180 1 2 3 DataToPrint = DataToPrint & "( "
01181      End Select
01182      Next I
01183      'Process add-ons if they exist
01184      If Len( EAN2AddOn) = 2 Then DataToPrint = DataToPrint & " " & ProcessEAN2AddOn( EAN2AddOn)
01185      If Len( EAN5AddOn) = 5 Then DataToPrint = DataToPrint & " " & ProcessEAN5AddOn( EAN5AddOn)
01186      'Return PrintableString
01187      EAN13 = DataToPrint
01188      End Function

01189
01190
01191      Public Function EAN8( DataToEncode As String) As String
01192          DataToPrint = ""
01193          OnlyCorrectData = ""
01194          'Check to make sure data is numeric and remove dashes, etc.
01195          StringLength = Len( DataToEncode)
01196          For I = 1 To StringLength
01197              'Add all numbers to OnlyCorrectData string
01198              '2006.2 BDA modified the next 3 lines for compatibility with different office versions
01199              'If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData &
»      Mid( DataToEncode, I, 1)
01200              CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01201              If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
»      DataToEncode, I, 1)
01202          Next I
01203          '2006.2 BDA modified the next 14 lines for add-on compatibility
01204          StringLength = Len( OnlyCorrectData)
01205          If StringLength < 7 Then OnlyCorrectData = "0000000"
01206          If StringLength = 11 Then OnlyCorrectData = "0000000"
01207          If StringLength = 8 Then OnlyCorrectData = Mid( OnlyCorrectData, 1, 7)
01208          If StringLength = 10 Then OnlyCorrectData = ( Mid( OnlyCorrectData, 1, 7) & Mid( OnlyCorrectData, 9, 2
»      ) )
01209          If StringLength > 12 Then OnlyCorrectData = ( Mid( OnlyCorrectData, 1, 7) & Mid( OnlyCorrectData, 9, 5
»      ) )
01210          Dim EAN2AddOn As String
01211          Dim EAN5AddOn As String
01212          EAN2AddOn = ""
01213          EAN5AddOn = ""
01214          '2006.2 BDA added the next line for general compatibility
01215          StringLength = Len( OnlyCorrectData)
01216          If StringLength = 12 Then EAN5AddOn = Mid( OnlyCorrectData, 8, 5)
01217          If StringLength = 9 Then EAN2AddOn = Mid( OnlyCorrectData, 8, 2)
01218          'Remove digit number from add-ons and check digit
01219          DataToEncode = Mid( OnlyCorrectData, 1, 7)
01220          'Calculate Check Digit
01221          Factor = 3
01222          WeightedTotal = 0
01223          For I = Len( DataToEncode) To 1 Step -1
01224              'Get the value of each number starting at the end
01225              CurrentCharNum = Mid( DataToEncode, I, 1)
01226              'Multiply by the weighting factor which is 3,1,3,1...
01227              'and add the sum together
01228              WeightedTotal = WeightedTotal + CurrentCharNum * Factor
01229              'Change factor for next calculation
01230              Factor = 4 - Factor
01231          Next I
01232          'Find the CheckDigit by finding the number + WeightedTotal that = a multiple of 10
01233          'Divide by 10, get the remainder and subtract from 10
01234          I = ( WeightedTotal Mod 10)
01235          If I <> 0 Then
01236              CheckDigit = ( 10 - I)
01237          Else
01238              CheckDigit = 0
01239          End If
01240          DataToEncode = DataToEncode & CheckDigit
01241          'Determine character to print for proper barcoding
01242          StringLength = Len( DataToEncode)
01243          For I = 1 To StringLength
01244              'Get the ASCII value of each number
01245              CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01246              CurrentEncoding = Mid( Encoding, I, 1)
01247              'Print different barcodes according to the location of the CurrentChar and CurrentEncoding
01248              Select Case I
01249              Case 1
01250                  'For the first character, print the normal guard pattern
01251                  'and then the barcode, without the human readable character
01252                  DataToPrint = "( " & ChrW( CurrentCharNum)

```

```

01253 1 2 3 } Case 2
01254      DataToPrint = DataToPrint & ChrW( CurrentCharNum)
01255      } Case 3
01256      DataToPrint = DataToPrint & ChrW( CurrentCharNum)
01257      } Case 4
01258      'Print the center guard pattern after the 6th character
01259      DataToPrint = DataToPrint & ChrW( CurrentCharNum) & ""
01260      } Case 5
01261      DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01262      } Case 6
01263      DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01264      } Case 7
01265      DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27)
01266      } Case 8
01267      'Print the check digit as 8th character + normal guard pattern
01268      DataToPrint = DataToPrint & ChrW( CurrentCharNum + 27) & "( "
01269      End Select
01270      Next I
01271      '2006.2 BDA modified the next 3 lines for add-on compatibility
01272      'Process add-ons if they exist
01273      If Len( EAN2AddOn) = 2 Then DataToPrint = DataToPrint & " " & ProcessEAN2AddOn( EAN2AddOn)
01274      If Len( EAN5AddOn) = 5 Then DataToPrint = DataToPrint & " " & ProcessEAN5AddOn( EAN5AddOn)
01275      'Return PrintableString
01276      EAN8 = DataToPrint
01277      End Function

```

```

01278
01279      Public Function UCC128( UCCToEncode As String) As String
01280      'Check for FNC1 which can be ASCII 202 and ASCII 212-217
01281      CurrentCharNum = AscW( Mid( UCCToEncode, 1, 1) )
01282      If ( ( CurrentCharNum = 202) Or ( ( CurrentCharNum > 211) And ( CurrentCharNum < 219) ) )
01283      >> Then
01284      'do nothing, AI is already in the data
01285      Else
01286      UCCToEncode = ChrW( 202) & UCCToEncode
01287      End If
01288      UCC128 = Code128( UCCToEncode, 0, True)
01289      End Function

```

```

01290
01291      Public Function Code11( DataToEncode As String) As String
01292      DataToPrint = ""
01293      OnlyCorrectData = ""
01294      ' Check to make sure data is numeric and remove dashes, etc.
01295      StringLength = Len( DataToEncode)
01296      For I = 1 To StringLength
01297      'Add all numbers to OnlyCorrectData string
01298      '2006.2 BDA modified the next 2 lines for compatibility with different office versions
01299      CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01300      If ( CurrentCharNum > 47 And CurrentCharNum < 58) Or CurrentCharNum = 45 Then OnlyCorrectData
01301      >> = OnlyCorrectData & Mid( DataToEncode, I, 1)
01302      Next I
01303      DataToEncode = OnlyCorrectData
01304      'Calculate Check Digit
01305      Factor = 1
01306      WeightedTotal = 0
01307      For I = Len( DataToEncode) To 1 Step -1
01308      'Get the value of each number starting at the end
01309      CurrentChar = Mid( DataToEncode, I, 1)
01310      'Set the "-" character to the value of 10
01311      If CurrentChar = "-" Then CurrentChar = "10"
01312      'Multiply by the weighting character and add together
01313      WeightedTotal = WeightedTotal + Val( CurrentChar) * Factor
01314      'Change factor for next calculation
01315      Factor = Factor + 1
01316      Next I
01317      'Find the Modulo 11 check digit
01318      CheckDigit = WeightedTotal Mod 11
01319      Code11 = "( " & DataToEncode & CheckDigit & " ) "
01320      End Function

```

```

01321
01322      Public Function RM4SCC( DataToEncode As String) As String
01323      DataToEncode = UCase( DataToEncode)
01324      OnlyCorrectData = ""
01325      StringLength = Len( DataToEncode)
01326      1

```

```

01327 1 For I = 1 To StringLength
01328     'Get each character one at a time
01329     CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01330     'Get the value of CurrentChar according to MOD43
01331     '0-9
01332     If CurrentCharNum < 58 And CurrentCharNum > 47 Then OnlyCorrectData = OnlyCorrectData & Mid(
»     DataToEncode, I, 1)
01333     'A-Z
01334     If CurrentCharNum < 91 And CurrentCharNum > 64 Then OnlyCorrectData = OnlyCorrectData & Mid(
»     DataToEncode, I, 1)
01335     Next I
01336     DataToEncode = OnlyCorrectData
01337     Dim r As Integer
01338     Dim C As Integer
01339     Dim Rtotal As Long
01340     Dim Ctotal As Long
01341     Rtotal = 0
01342     Ctotal = 0
01343     WeightedTotal = 0
01344     StringLength = Len( DataToEncode)
01345     For I = 1 To StringLength
01346         'Get each character one at a time
01347         CurrentChar = Mid( DataToEncode, I, 1)
01348         'Get the values of CurrentChar
01349         Select Case CurrentChar
01350             Case "0"
01351                 r = 1
01352                 C = 1
01353             Case "1"
01354                 r = 1
01355                 C = 2
01356             Case "2"
01357                 r = 1
01358                 C = 3
01359             Case "3"
01360                 r = 1
01361                 C = 4
01362             Case "4"
01363                 r = 1
01364                 C = 5
01365             Case "5"
01366                 r = 1
01367                 C = 0
01368             Case "6"
01369                 r = 2
01370                 C = 1
01371             Case "7"
01372                 r = 2
01373                 C = 2
01374             Case "8"
01375                 r = 2
01376                 C = 3
01377             Case "9"
01378                 r = 2
01379                 C = 4
01380             Case "A"
01381                 r = 2
01382                 C = 5
01383             Case "B"
01384                 r = 2
01385                 C = 0
01386             Case "C"
01387                 r = 3
01388                 C = 1
01389             Case "D"
01390                 r = 3
01391                 C = 2
01392             Case "E"
01393                 r = 3
01394                 C = 3
01395             Case "F"
01396                 r = 3
01397                 C = 4
01398             Case "G"
01399                 r = 3
01400                 C = 5
01401             Case "H"
01402                 r = 3

```

```

01403 1 2 3 C = 0
01404      } Case "I"
01405      r = 4
01406      C = 1
01407      } Case "J"
01408      r = 4
01409      C = 2
01410      } Case "K"
01411      r = 4
01412      C = 3
01413      } Case "L"
01414      r = 4
01415      C = 4
01416      } Case "M"
01417      r = 4
01418      C = 5
01419      } Case "N"
01420      r = 4
01421      C = 0
01422      } Case "O"
01423      r = 5
01424      C = 1
01425      } Case "P"
01426      r = 5
01427      C = 2
01428      } Case "Q"
01429      r = 5
01430      C = 3
01431      } Case "R"
01432      r = 5
01433      C = 4
01434      } Case "S"
01435      r = 5
01436      C = 5
01437      } Case "T"
01438      r = 5
01439      C = 0
01440      } Case "U"
01441      r = 0
01442      C = 1
01443      } Case "V"
01444      r = 0
01445      C = 2
01446      } Case "W"
01447      r = 0
01448      C = 3
01449      } Case "X"
01450      r = 0
01451      C = 4
01452      } Case "Y"
01453      r = 0
01454      C = 5
01455      } Case "Z"
01456      r = 0
01457      C = 0
01458      } End Select
01459      'add the values together
01460      Rtotal = Rtotal + r
01461      Ctotal = Ctotal + C
01462      } Next I
01463
01464
01465      'divide the Totals by 6 and get the remainder, this is a reference
01466      'to the Check Digit.
01467      'set check digit to CurrentChar ( a string)
01468      Rtotal = ( Rtotal Mod 6)
01469      Ctotal = ( Ctotal Mod 6)
01470      Select Case Rtotal
01471      } Case 1
01472      } Select Case Ctotal
01473      } Case 1
01474      CurrentChar = "0"
01475      } Case 2
01476      CurrentChar = "1"
01477      } Case 3
01478      CurrentChar = "2"
01479      } Case 4
01480      CurrentChar = "3"

```

```

01481 1 2 3 } Case 5
01482      CurrentChar = "4"
01483 } Case 0
01484      CurrentChar = "5"
01485 } End Select
01486 } Case 2
01487      Select Case Ctotal
01488      } Case 1
01489          CurrentChar = "6"
01490      } Case 2
01491          CurrentChar = "7"
01492      } Case 3
01493          CurrentChar = "8"
01494      } Case 4
01495          CurrentChar = "9"
01496      } Case 5
01497          CurrentChar = "A"
01498      } Case 0
01499          CurrentChar = "B"
01500      } End Select
01501 } Case 3
01502      Select Case Ctotal
01503      } Case 1
01504          CurrentChar = "C"
01505      } Case 2
01506          CurrentChar = "D"
01507      } Case 3
01508          CurrentChar = "E"
01509      } Case 4
01510          CurrentChar = "F"
01511      } Case 5
01512          CurrentChar = "G"
01513      } Case 0
01514          CurrentChar = "H"
01515      } End Select
01516 } Case 4
01517      Select Case Ctotal
01518      } Case 1
01519          CurrentChar = "I"
01520      } Case 2
01521          CurrentChar = "J"
01522      } Case 3
01523          CurrentChar = "K"
01524      } Case 4
01525          CurrentChar = "L"
01526      } Case 5
01527          CurrentChar = "M"
01528      } Case 0
01529          CurrentChar = "N"
01530      } End Select
01531 } Case 5
01532      Select Case Ctotal
01533      } Case 1
01534          CurrentChar = "O"
01535      } Case 2
01536          CurrentChar = "P"
01537      } Case 3
01538          CurrentChar = "Q"
01539      } Case 4
01540          CurrentChar = "R"
01541      } Case 5
01542          CurrentChar = "S"
01543      } Case 0
01544          CurrentChar = "T"
01545      } End Select
01546 } Case 0
01547      Select Case Ctotal
01548      } Case 1
01549          CurrentChar = "U"
01550      } Case 2
01551          CurrentChar = "V"
01552      } Case 3
01553          CurrentChar = "W"
01554      } Case 4
01555          CurrentChar = "X"
01556      } Case 5
01557          CurrentChar = "Y"
01558      } Case 0
1 2 3

```

```

01559 1 2 3 CurrentChar = "Z"
01560 End Select
01561 End Select
01562 'Return Printable String
01563 RM4SCC = "(" & DataToEncode & CurrentChar & ")" "
01564 End Function

01565
01566
01567 Public Function Codabar( DataToEncode As String) As String
01568 DataToPrint = ""
01569 OnlyCorrectData = ""
01570 StringLength = Len( DataToEncode)
01571 'Check to make sure data is numeric, $, +, -, /, or :, and remove all others.
01572 For I = 1 To StringLength
01573 '2006.2 BDA modified the next 9 lines for compatibility with different office versions
01574 CurrentChar = Mid( DataToEncode, I, 1)
01575 CurrentCharNum = AscW( CurrentChar)
01576 If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
» DataToEncode, I, 1)
01577 If CurrentChar = "$" Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
01578 If CurrentChar = "+" Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
01579 If CurrentChar = "-" Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
01580 If CurrentChar = "/" Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
01581 If CurrentChar = "." Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
01582 If CurrentChar = ":" Then OnlyCorrectData = OnlyCorrectData & Mid( DataToEncode, I, 1)
01583 Next I
01584 DataToPrint = OnlyCorrectData
01585 'Get Printable String
01586 Codabar = "A" & DataToPrint & "B"
01587 End Function

01588
01589
01590 Public Function Postnet( DataToEncode As String, Optional ReturnType As Integer = 0) As String
01591 'Additional logic in case ReturnType is not correct
01592 If ReturnType <> 0 And ReturnType <> 1 And ReturnType <> 2 Then ReturnType = 0
01593 DataToPrint = ""
01594 OnlyCorrectData = ""
01595 'Check to make sure data is numeric and remove dashes, etc.
01596 StringLength = Len( DataToEncode)
01597 For I = 1 To StringLength
01598 'Add all numbers to OnlyCorrectData string
01599 '2006.2 BDA modified the next 3 lines for compatibility with different office versions
01600 'If IsNumeric( Mid( DataToEncode, I, 1) ) Then OnlyCorrectData = OnlyCorrectData &
» Mid( DataToEncode, I, 1)
01601 CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01602 If CurrentCharNum > 47 And CurrentCharNum < 58 Then OnlyCorrectData = OnlyCorrectData & Mid(
» DataToEncode, I, 1)
01603 Next I
01604 DataToEncode = OnlyCorrectData
01605 'Calculate Check Digit
01606 WeightedTotal = 0
01607 StringLength = Len( DataToEncode)
01608 For I = 1 To StringLength
01609 'Get the value of each number
01610 CurrentCharNum = Mid( DataToEncode, I, 1)
01611 'Add the values together
01612 WeightedTotal = WeightedTotal + CurrentCharNum
01613 Next I
01614 'Find the CheckDigit by finding the number + WeightedTotal that = a multiple of 10
01615 'divide by 10, get the remainder and subtract from 10
01616 I = ( WeightedTotal Mod 10)
01617 If I <> 0 Then
01618 CheckDigit = ( 10 - I)
01619 Else
01620 CheckDigit = 0
01621 End If
01622 'ReturnType 0 returns data formatted to the barcode font
01623 If ReturnType = 0 Then Postnet = "(" & DataToEncode & CheckDigit & ")" "
01624 'ReturnType 1 returns data formatted for human readable text
01625 If ReturnType = 1 Then Postnet = DataToEncode & CheckDigit
01626 'ReturnType 2 returns the check digit for the data supplied
01627 If ReturnType = 2 Then Postnet = Str$( CheckDigit)
01628 End Function

01629
01630
01631 Public Function Code93( DataToEncode As String) As String

```

```

01632 DataToEncode = UCase( DataToEncode)
01633 DataToPrint = ""
01634 OnlyCorrectData = ""
01635 'Only pass correct data
01636 StringLength = Len( DataToEncode)
01637 For I = 1 To StringLength
01638     CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01639     If Code93Val( CurrentCharNum) < 47 Then
01640         If CurrentCharNum = 32 Then CurrentCharNum = 61
01641         OnlyCorrectData = OnlyCorrectData & ChrW( CurrentCharNum)
01642     End If
01643 Next I
01644 DataToEncode = OnlyCorrectData
01645 CurrentCharNum = 0
01646 StringLength = Len( DataToEncode)
01647 Dim C As Integer
01648 Dim K As Integer
01649 Dim CW As Integer
01650 Dim KW As Integer
01651 Dim CWSum As Integer
01652 Dim KWSum As Integer
01653 CW = 1
01654 KW = 2
01655 I = 1
01656 '2006.2 BDA modified the next line for compatibility with different office versions
01657 For I = StringLength To 1 Step -1
01658     'Get each character one at a time from the back
01659     CurrentCharNum = AscW( Mid( DataToEncode, I, 1) )
01660     'Get the value
01661     CurrentValue = Code93Val( CurrentCharNum)
01662     'Calculate check digit C
01663     CWSum = CWSum + ( CurrentValue * CW)
01664     CW = CW + 1
01665     If CW = 21 Then CW = 1
01666     'Calculate check digit K
01667     KWSum = KWSum + ( CurrentValue * KW)
01668     KW = KW + 1
01669     If KW = 16 Then KW = 1
01670     'Gather data to print
01671     DataToPrint = ChrW( CurrentCharNum) & DataToPrint
01672 Next I
01673 'Divide the C sum by 47 and get the remainder, this is the Check Digit
01674 C = ( CWSum Mod 47)
01675 'Add the last digit to the K sum
01676 KWSum = KWSum + C
01677 'Divide the K sum by 47 and get the remainder, this is the Check Digit
01678 K = ( KWSum Mod 47)
01679 Code93 = "( " & DataToPrint & Code93Char( C) & Code93Char( K) & ")" "
01680 End Function

01681
01682 Private Function Code93Char( CharValue As Integer) As String
01683     'Returns a character from a character value
01684     'An invalid character value returns nothing
01685     Code93Char = ""
01686     If CharValue < 10 And CharValue > -1 Then Code93Char = ChrW( CharValue + 48)
01687     'A-Z
01688     If CharValue < 36 And CharValue > 9 Then Code93Char = ChrW( CharValue + 55)
01689     '-'
01690     If CharValue = 36 Then Code93Char = ChrW( 45)
01691     '.'
01692     If CharValue = 37 Then Code93Char = "."
01693     'Space
01694     If CharValue = 38 Then Code93Char = "="
01695     '$
01696     If CharValue = 39 Then Code93Char = "$"
01697     '/'
01698     If CharValue = 40 Then Code93Char = "/"
01699     '+'
01700     If CharValue = 41 Then Code93Char = "+"
01701     '%'
01702     If CharValue = 42 Then Code93Char = "%"
01703     '!'
01704     If CharValue = 43 Then Code93Char = "!"
01705     '#'
01706     If CharValue = 44 Then Code93Char = "#"
01707     '&'
01708     If CharValue = 45 Then Code93Char = "&"
01709     '@

```

```

01710 1 If CharValue = 46 Then Code93Char = "@"
01711 End Function

01712
01713 Private Function Code93Val( CharASCValue As Integer) As Integer
01714 'Returns a character value from a character
01715 'An invalid character value returns 99
01716 Code93Val = 99
01717 '0-9
01718 If CharASCValue < 58 And CharASCValue > 47 Then Code93Val = CharASCValue - 48
01719 'A-Z
01720 If CharASCValue < 91 And CharASCValue > 64 Then Code93Val = CharASCValue - 55
01721 'Space
01722 If CharASCValue = 32 Then Code93Val = 38
01723 '='
01724 If CharASCValue = 61 Then Code93Val = 38
01725 '-'
01726 If CharASCValue = 45 Then Code93Val = 36
01727 '.'
01728 If CharASCValue = 46 Then Code93Val = 37
01729 '$
01730 If CharASCValue = 36 Then Code93Val = 39
01731 '/'
01732 If CharASCValue = 47 Then Code93Val = 40
01733 '+'
01734 If CharASCValue = 43 Then Code93Val = 41
01735 '%'
01736 If CharASCValue = 37 Then Code93Val = 42
01737 '!'
01738 If CharASCValue = 33 Then Code93Val = 43
01739 '#'
01740 If CharASCValue = 35 Then Code93Val = 44
01741 '&'
01742 If CharASCValue = 38 Then Code93Val = 45
01743 '@'
01744 If CharASCValue = 64 Then Code93Val = 46
01745 End Function

01746
01747
01748 » Public Function SpliceText( DataToFormat As String, Optional SpacingNumber As Integer = 4, Optional
ApplyTilde As Boolean = False) As String
01749 'This function inserts a space for every SpacingNumber of characters
01750 '2006.2 BDA added the next line to move code to the ProcessTilde function
01751 If ApplyTilde Then DataToFormat = ProcessTilde( DataToFormat)
01752 HumanReadableText = ""
01753 StringLength = Len( DataToFormat)
01754 J = 0
01755 For I = 1 To StringLength
01756 CurrentCharNum = AscW( Mid( DataToFormat, I, 1) )
01757 If CurrentCharNum > 31 And CurrentCharNum < 128 Then
01758 HumanReadableText = HumanReadableText & Mid( DataToFormat, I, 1)
01759 J = J + 1
01760 End If
01761 If ( J Mod SpacingNumber) = 0 Then HumanReadableText = HumanReadableText & " "
01762 Next I
01763 SpliceText = HumanReadableText
01764 End Function

01765
01766
01767 Public Function MOD10( M10NumberData As String) As Integer
01768 *****
01769 ' This is a general MOD10 function compatible with EAN and UPC standards
01770 *****
01771 Dim M10StringLength As Integer
01772 Dim M10OnlyCorrectData As String
01773 Dim M10Factor As Integer
01774 Dim M10WeightedTotal As Integer
01775 Dim M10CheckDigit As Integer
01776 Dim M10I As Integer
01777 M10OnlyCorrectData = ""
01778 M10StringLength = Len( M10NumberData)
01779 'Check to make sure data is numeric and remove dashes, etc.
01780 For M10I = 1 To M10StringLength
01781 'Add all numbers to OnlyCorrectData string
01782 '2006.2 BDA modified the next 2 lines for compatibility with different office versions
01783 CurrentCharNum = AscW( Mid( M10NumberData, M10I, 1) )
01784 If CurrentCharNum > 47 And CurrentCharNum < 58 Then M10OnlyCorrectData = M10OnlyCorrectData &

```

```

01785 Mid( M10NumberData, M10I, 1)
01786     Next M10I
01787     'Generate MOD 10 check digit
01788     M10Factor = 3
01789     M10WeightedTotal = 0
01790     M10StringLength = Len( M10NumberData)
01791     For M10I = M10StringLength To 1 Step -1
01792         'Get the value of each number starting at the end
01793         'CurrentCharNum = Mid( M10NumberData, I, 1)
01794         'Multiply by the weighting factor which is 3,1,3,1...
01795         'and add the sum together
01796         M10WeightedTotal = M10WeightedTotal + ( Val( Mid( M10NumberData, M10I, 1) ) * M10Factor)
01797         'Change factor for next calculation
01798         M10Factor = 4 - M10Factor
01799     Next M10I
01800     'Find the CheckDigit by finding the smallest number that = a multiple of 10
01801     M10I = ( M10WeightedTotal Mod 10)
01802     If M10I <> 0 Then
01803         M10CheckDigit = ( 10 - M10I)
01804     Else
01805         M10CheckDigit = 0
01806     End If
01807     MOD10 = Str( M10CheckDigit)
End Function

```

```

01808
01809
01810 Public Function ProcessTilde ( StringToProcess As String) As String
01811     ProcessTilde = ""
01812     Dim OutString As String
01813     StringLength = Len( StringToProcess)
01814     For I = 1 To StringLength
01815         If ( I < StringLength - 2) And Mid( StringToProcess, I, 2) = "-m" And IsNumeric( Mid(
»     StringToProcess, I + 2, 2) ) Then
01816             Dim StringToCheck As String
01817             WeightValue = Val( Mid( StringToProcess, I + 2, 2) )
01818             'TB 8/3/2007 now we must walk through the outstring backwards starting one spot before the ~
01819             Dim CharsAdded As Integer
01820             For J = I To 1 Step -1
01821                 If IsNumeric( Mid( OutString, J, 1) ) Then
01822                     StringToCheck = StringToCheck & Mid( OutString, J, 1)
01823                     CharsAdded = CharsAdded + 1
01824                 End If
01825                 'when the number of digits added to StringToCheck equals the weight value
01826                 'we must exit the for loop
01827                 If CharsAdded = WeightValue Then
01828                     Exit For
01829                 End If
01830             Next J
01831             CheckDigitValue = MOD10( StrReverse( StringToCheck) )
01832             OutString = OutString & ChrW( CheckDigitValue + 48)
01833             I = I + 3
01834         ElseIf ( I < StringLength - 2) And Mid( StringToProcess, I, 1) = "-" And IsNumeric( Mid(
»     StringToProcess, I + 1, 3) ) Then
01835             CurrentCharNum = Val( Mid( StringToProcess, I + 1, 3) )
01836             OutString = OutString & ChrW( CurrentCharNum)
01837             I = I + 3
01838         Else
01839             OutString = OutString & Mid( StringToProcess, I, 1)
01840         End If
01841     Next I
01842     ProcessTilde = OutString
01843     StringToProcess = ""
01844 End Function

```

```

01845
01846
01847 Public Function ProcessEAN5AddOn ( EAN5AddOn As String) As String
01848     If Len( EAN5AddOn) = 5 Then
01849         EANAddOnToPrint = ""
01850         'Get the check digit for the add on
01851         Factor = 3
01852         WeightedTotal = 0
01853         For I = Len( EAN5AddOn) To 1 Step -1
01854             'Get the value of each number starting at the end
01855             CurrentCharNum = Mid( EAN5AddOn, I, 1)
01856             'Multiply by the weighting factor which is 3,9,3,9.
01857             'and add the sum together
01858             If Factor = 3 Then WeightedTotal = WeightedTotal + CurrentCharNum * 3

```

```

01859 1 2 3 If Factor = 1 Then WeightedTotal = WeightedTotal + CurrentCharNum * 9
01860 'Change factor for next calculation
01861 Factor = 4 - Factor
01862 Next I
01863 'Find the CheckDigit by extracting the right-most number from WeightedTotal
01864 CheckDigit = Val( Right$( WeightedTotal, 1) )
01865 'Encode the add-on CheckDigit into the number sets
01866 'by using variable parity between character sets A and B
01867 Select Case CheckDigit
01868 Case 0
01869 Encoding = "BAAA"
01870 Case 1
01871 Encoding = "BABAA"
01872 Case 2
01873 Encoding = "BAABA"
01874 Case 3
01875 Encoding = "BAAAB"
01876 Case 4
01877 Encoding = "ABBAA"
01878 Case 5
01879 Encoding = "AABBA"
01880 Case 6
01881 Encoding = "AAABB"
01882 Case 7
01883 Encoding = "ABABA"
01884 Case 8
01885 Encoding = "ABAAB"
01886 Case 9
01887 Encoding = "AABAB"
01888 End Select
01889 'Determine the characters to print for proper barcoding
01890 For I = 1 To Len( EAN5AddOn)
01891 'Get the value of each number encoded with variable parity
01892 CurrentChar = Mid( EAN5AddOn, I, 1)
01893 CurrentEncoding = Mid( Encoding, I, 1)
01894 'Print different barcodes according to the location of the CurrentChar and CurrentEncoding
01895 Select Case CurrentEncoding
01896 Case "A"
01897 If CurrentChar = "0" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 34)
01898 If CurrentChar = "1" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 35)
01899 If CurrentChar = "2" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 36)
01900 If CurrentChar = "3" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 37)
01901 If CurrentChar = "4" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 38)
01902 If CurrentChar = "5" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 44)
01903 If CurrentChar = "6" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 46)
01904 If CurrentChar = "7" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 47)
01905 If CurrentChar = "8" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 58)
01906 If CurrentChar = "9" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 59)
01907 Case "B"
01908 If CurrentChar = "0" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 122)
01909 If CurrentChar = "1" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 61)
01910 If CurrentChar = "2" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 63)
01911 If CurrentChar = "3" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 64)
01912 If CurrentChar = "4" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 91)
01913 If CurrentChar = "5" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 92)
01914 If CurrentChar = "6" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 93)
01915 If CurrentChar = "7" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 95)
01916 If CurrentChar = "8" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 123)
01917 If CurrentChar = "9" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 125)
01918 End Select
01919 'Add in the space & add-on guard pattern
01920 Select Case I
01921 Case 1
01922 EANAddOnToPrint = ChrW( 43) & EANAddOnToPrint & ChrW( 33)
01923 'Print add-on delineators between each add-on character
01924 Case 2
01925 EANAddOnToPrint = EANAddOnToPrint & ChrW( 33)
01926 Case 3
01927 EANAddOnToPrint = EANAddOnToPrint & ChrW( 33)
01928 Case 4
01929 EANAddOnToPrint = EANAddOnToPrint & ChrW( 33)
01930 Case 5
01931 EANAddOnToPrint = EANAddOnToPrint
01932 End Select
01933 Next I
01934 End If
01935 ProcessEAN5AddOn = EANAddOnToPrint
01936 End Function

```

```

01937
01938
01939 Public Function ProcessEAN2AddOn( EAN2AddOn As String) As String
01940   'Process the 2 digit add on
01941   EANAddOnToPrint = ""
01942   If Len( EAN2AddOn) = 2 Then
01943     'Get encoding for add on
01944     For I = 0 To 99 Step 4
01945       If Val( EAN2AddOn) = I Then Encoding = "AA"
01946       If Val( EAN2AddOn) = I + 1 Then Encoding = "AB"
01947       If Val( EAN2AddOn) = I + 2 Then Encoding = "BA"
01948       If Val( EAN2AddOn) = I + 3 Then Encoding = "BB"
01949     Next I
01950     For I = 1 To Len( EAN2AddOn)
01951       'Get the value of each number
01952       'It is encoded with variable parity
01953       CurrentChar = Mid( EAN2AddOn, I, 1)
01954       CurrentEncoding = Mid( Encoding, I, 1)
01955       'Print different barcodes according to the location of the CurrentChar and CurrentEncoding
01956       Select Case CurrentEncoding
01957         Case "A"
01958           If CurrentChar = "0" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 34)
01959           If CurrentChar = "1" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 35)
01960           If CurrentChar = "2" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 36)
01961           If CurrentChar = "3" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 37)
01962           If CurrentChar = "4" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 38)
01963           If CurrentChar = "5" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 44)
01964           If CurrentChar = "6" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 46)
01965           If CurrentChar = "7" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 47)
01966           If CurrentChar = "8" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 58)
01967           If CurrentChar = "9" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 59)
01968         Case "B"
01969           If CurrentChar = "0" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 122)
01970           If CurrentChar = "1" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 61)
01971           If CurrentChar = "2" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 63)
01972           If CurrentChar = "3" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 64)
01973           If CurrentChar = "4" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 91)
01974           If CurrentChar = "5" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 92)
01975           If CurrentChar = "6" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 93)
01976           If CurrentChar = "7" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 95)
01977           If CurrentChar = "8" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 123)
01978           If CurrentChar = "9" Then EANAddOnToPrint = EANAddOnToPrint & ChrW( 125)
01979       End Select
01980       'Add in the space & add-on guard pattern
01981       Select Case I
01982         Case 1
01983           EANAddOnToPrint = ChrW( 43) & EANAddOnToPrint & ChrW( 33)
01984           'Print add-on delineators between each add-on character
01985         Case 2
01986           EANAddOnToPrint = EANAddOnToPrint
01987       End Select
01988     Next I
01989   End If
01990   ProcessEAN2AddOn = ProcessEAN2AddOn & EANAddOnToPrint
01991 End Function

01992
01993 Public Function IntelligentMail( DataToEncode As String) As String
01994   On Error GoTo ErrorHandler
01995   Dim myOut As String
01996   Dim iSize As Long
01997   Dim IRetVal As Long
01998   myOut = String( 512, " ")
01999   IRetVal = IDAutomation_ Universal_ OneCode( DataToEncode, myOut, iSize)
02000   IntelligentMail = RTrim( myOut)
02001   If J = 2 Then IntelligentMail = "ERROR"
02002   Exit Function
02003 ErrorHandler:
02004   If J <> 2 Then
02005     MsgBox "The IDAutomationNativeFontEncoder.dll file does not appear to be installed. To create USPS
>> Intelligent Mail symbols, the Native Font Encoder DLL file IDAutomationNativeFontEncoder.dll must be
>> installed in the Windows System directory. This file may be downloaded from
>> www.idautomation.com/fonts/tools/windows_dll/ " & Chr( 13) & Err.Description, vbOKOnly, "Notice:"
02006     J = 2
02007   End If
02008   Resume Next
02009 End Function
02010

```

```
02011 '*****
02012 ' © Copyright, IDAutomation.com, Inc. All rights reserved.
02013 ' Redistribution and use of this code in source and/or binary
02014 ' forms, with or without modification, are permitted provided
02015 ' that: ( 1 ) all copies of the source code retain the above
02016 ' unmodified copyright notice and this entire unmodified
02017 ' section of text. ( 2 ) You or Your organization owns a valid
02018 ' Developer License to this product from IDAutomation.com
02019 ' and. ( 3 ) when any portion of this code is bundled in any
02020 ' form with an application, a valid notice must be provided
02021 ' within the user documentation, start-up screen or in the
02022 ' help-about section of the application that specifies
02023 ' IDAutomation.com as the provider of the Software bundled
02024 ' with the application.
02025 '*****
02026 '
02027 ' Internal Version 2007.9.20
02028 '
02029 '
```

Procedure Index

Procedure Index for Project - BarcodeMacros

Pg #	Name	Referenced on Pages	Scope	Type
2	IDAutomationVBA	3-29		Module
23	Codabar		Public	Function
19	Code11		Public	Function
2	Code128	6, 7, 19	Public	Function
6	Code128a		Public	Function
6	Code128b	7	Public	Function
7	Code128c		Public	Function
9	Code39	8	Public	Function
7	Code39Mod43	8, 9	Public	Function
23	Code93	24, 25	Public	Function
24	Code93Char	25	Private	Function
25	Code93Val	24	Private	Function
16	EAN13	18	Public	Function
18	EAN8	19	Public	Function
7	I2of5	9, 10	Public	Function
9	I2of5Mod10	10	Public	Function
28	IntelligentMail		Public	Function
25	MOD10	9, 10, 26	Public	Function
10	MSI	11	Public	Function
23	Postnet		Public	Function
27	ProcessEAN2AddOn	12, 16, 18, 19, 28	Public	Function
26	ProcessEAN5AddOn	12, 16, 18, 19, 27	Public	Function
26	ProcessTilde	2, 25	Public	Function
19	RM4SCC	23	Public	Function
25	SpliceText		Public	Function
19	UCC128		Public	Function
11	UPCa	12	Public	Function
13	UPCe	12, 16	Public	Function
12	UPCe7To11	13	Private	Function

Table of Contents

IDAutomationVBA	2
Code128	2
Code128a	6
Code128b	6
Code128c	7
I2of5	7
Code39Mod43	7
Code39	9
I2of5Mod10	9
MSI	10
UPCa	11
UPCe7To11	12
UPCe	13
EAN13	16
EAN8	18
UCC128	19
Code11	19
RM4SCC	19
Codabar	23
Postnet	23
Code93	23
Code93Char	24
Code93Val	25
SpliceText	25
MOD10	25
ProcessTilde	26
ProcessEAN5AddOn	26
ProcessEAN2AddOn	27
IntelligentMail	28
