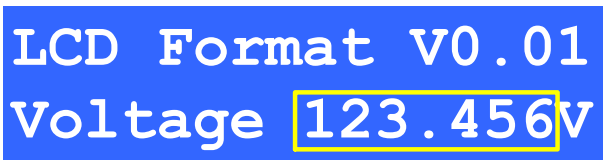
Zahlen in LCD-Anzeige einpassen

DL6GL, 18.06.2022



Eine Displayanzeige sollte leicht ablesbar sein. Dazu gehört auch, dass Messwerte, hier Spannungen, immer am erwarteten Ort erscheinen, 123.456V ebenso wie 0.123V. Bei Gleitkommazahlen orientiert sich unser Auge am Dezimalpunkt/Dezimalkomma.

Die Formatierungsoptionen von BASCOM für Zahlen, "Format" und "Fusing", ergeben immer linksbündige Zeichenketten. Wir müssen also zusehen, die formatierte Zeichenkette rechtsbündig in den vorgesehenen gelben Rahmen im obigen Bild einzufügen.

Vorgehensweise in allen drei gezeigten Beispielen:

1. Vorgabe der Anzeigemaske und des Anzeigebereichs für die Zahlen.
   Dabei ist bei Gleitkommazahlen und bei Festkommazahlen vom Typ Integer oder Long ggf. Platz für das Vorzeichen zu berücksichtigen,
2. Umwandlung der Zahl in einen String,
3. bei Gleitkommazahlen Formatierung auf die gewünschte Nachkommazahl,
4. Sortieren jedes einzelnen Zeichens von rechts nach links (rückwärts) in den Anzeigebereich.
   Dabei kommt uns das hier (https://dl6gl.de/bytes-im-gaensemarsch-mit-bascom-overlay.html) beschriebene BASCOM Overlay zur Hilfe.

1 Formatieren mit BASCOM 'Fusing'

Zunächst naheliegend für die Formatierung von Gleitkomma (Real)-Zahlen ist die Funktion 'Fusing'.

target = **Fusing**(source, "mask")

target	Ergebnis der Funktion als formatierter String
source	Real-Variable, die konvertiert werden soll, nur (!)vom Typ Single
"mask"	Maske (Formatierungsanweisung) für source Muss immer mit # beginnen, beidseitig abgeschlossen mit "…". "#.###" formatiert z.B. 123.4567 in 123.457 mit Rundung der letzten Stelle "#.&&&" formatiert z.B. 123.4567 in 123.456 ohne Rundung der letzten Stelle.

Programmbeispiel LCD Format V0.01 (Ausschnitt):

```
sngTest = 1234.567
sngTmp0 = sngTest
bytRun = 0
strDis = "Voltage        V"                    'Display mask
strSav = Space(16)

Do
   Incr bytRun
   'Use 'Fusing' for string conversion
   strNum = Fusing(sngTmp0 , "#.###")
   If strNum <> strSav Then                    'Display if number was changed
      Call DisplayReal(7 , 15)
      Locate 2 , 1
      LCD strDis
      strSav = strNum
      Wait 2
   End If

   If bytRun = 5 Then
      bytRun = 0
      sngTmp0 = sngTest
```

```
   Else
      sngTmp0 = 0.1 * sngTmp0
   End If
Loop
```

Mit `strDis = "Voltage       V"` wird die Anzeigemaske mit 16 Zeichen für ein 16x2 LCD vorgegeben. Die graue Schraffierung zeigt den Platz an, in den die formatierte Zahl eingefügt werden soll.

In der Do…Loop wird die anzuzeigende Zahl sngTmp0 (Typ Single) schrittweise verkleinert. Die Überprüfung, ob eine geänderte Zahl anzuzeigen ist, ist hier also eigentlich sinnfrei. Wenn aber sich ständig ändernde Messwerte fortlaufend in einer Do…Loop anstehen, müssen gleiche Werte ja nicht immer wieder neu angezeigt werden. Das spart Zeit.

Mit `strNum = Fusing(sngTmp0 , "#.###")` erfolgt die jeweilige Formatierung mit 3 Nachkommastellen, wobei der String strNum linksbündig ist.
`DisplayReal(7 , 15)` sortiert den String strNum von rechts ab Position 15 in die Anzeigemaske strDis ein. Die verfügbare Länge, siehe obige Schraffierung, ist 7 Zeichen.

```
Sub DisplayReal(byVal bytLenMax As Byte , byVal bytPosLast As Byte)
'Insert temporary string  bytNum to display string bytDis
'from position bytPosLast backwards, max. bytLenMax characters
'Input:  bytNum    (=strNum by overlay) string to be inserted in strDis
'        bytLenMax   Max. possible length of string strNum to be inserted
'        bytPosLast  Right position in bytDis (=strDis by overlay)
'Output: strDis      Completed string to be displayed on LCD

         bytTmp1 = bytPosLast - bytLenMax
         bytTmp1 = bytTmp1 + 1
         bytLen = Len(strNum)
         If bytLen > bytLenMax Then                'No room, overflow
            For bytTmp0 = bytTmp1 To bytPosLast
               bytDis(bytTmp0) = "#"
            Next bytTmp0
            Exit Sub
         Else                                      'Fits available space
            For bytTmp0 = bytTmp1 To bytPosLast
               bytDis(bytTmp0) = " "               'Fill with blanks
            Next bytTmp0
         End If

         bytTmp1 = bytPosLast + 1
         For bytTmp0 = bytLen To 1 Step -1         'From right to left
             bytTmp1 = bytTmp1 - 1
             bytDis(bytTmp1) = bytNum(bytTmp0)
         Next bytTmp0

End Sub
```

In der Sub DisplayReal wird zunächst geprüft, ob der formatierte Zahlen-String strNum in den verfügbaren Platz (bytLenMax) passt. Wenn nicht, wird der Anzeigebereich in strDis mit einem Überlaufbalken ####### gefüllt.
Wenn ja, wird der Anzeigebereich zunächst mit Blanks, anschließend von rechts nach links (rückwärts) mit den Zeichen aus strNum gefüllt.

Um die zeichenweise Verarbeitung zu ermöglichen, sind strNum und strDis mit Byte-Overlays definiert, z.B.

```
Dim strDis As String * 16                          'Text to be dispayed on LCD
Dim bytDis(16) As Byte At strDis Overlay           'the same as byte array
```

2 Formatieren mit BASCOM 'Format'

Die Schreibweise ist identisch mit 'Fusing':

target = **Format**(source, "mask")

target	Ergebnis der Funktion als formatierter String
source	String aus beliebiger numerischer Variablen, die konvertiert werden soll Praktisch ist es, Festkommazahlen (Word, Integer, DWord, Long) zu verwenden und die zu formatierende Gleitkomazahl mit einer Zehnerpotenz passend zur Nachkommazahl zu multiplizieren, z.B. *1000 für 3 Nachkommastellen, *100 für 2.
"mask"	Maske (Formatierungsanweisung) für source beidseitig abgeschlossen mit "…". "0.000" formatiert z.B. 123456 in 123.456 mit 3 Nachkommastellen.

Ein paar mehr Möglichkeiten bietet 'mask' noch:

- Mit führenden Nullen in mask, z.B. "0000.000", kann die Länge des resultierenden Strings bestimmt werden, solange die Ziffernzahl kleiner ist als die vorgegebene Formatmaske.
  Der o.g. String 123456 würde mit mask = "0000.000" ergeben: 0123.456, String 12345 ergäbe 0012.345. In beiden Fällen feste Länge, die, ab einer bestimmten LCD-Position geschrieben, den vorgesehenen Platz richtig ausfüllen. Sieht aber nicht schön aus.
- Führende Blanks in mask, z.B. "_ _000.000" (2 führende Blanks), verlängern den resultierenden Strings um eben diese Blanks, solange (wie oben) die Ziffernzahl kleiner ist als die vorgegebene Formatmaske.
  Der o.g. String 123456 würde mit dieser mask ergeben: _ _123.456. Auch feste Länge.

Bei nicht vorhersagbarer Ziffernzahl ist es nicht möglich, alleine mit der Format-Funktion eine ansehnliche rechtsbündige Darstellung mit fester Position des Dezimalpunktes hinzubekommen.

Programmbeispiel LCD Format V0.02 (Ausschnitt):

```
sngTest = 1234.567
sngTmp0 = sngTest
bytRun = 0
strDis = "Voltage        V"                    'Display mask
strSav = Space(16)

Do
   Incr bytRun
   'Use 'Format' for string conversion
   dwdTmp0 = sngTmp0 * 1000                    'Factor for 3 decimals
   strNum = str(dwdTmp0)
   strNum = Format(strNum , "0.000")
   If strNum <> strSav Then                    'Display if number was changed
      Call DisplayReal(7 , 15)
      Locate 2 , 1
      LCD strDis
      strSav = strNum
      Wait 2
   End If

   If bytRun = 5 Then
      bytRun = 0
      sngTmp0 = sngTest
   Else
      sngTmp0 = 0.1 * sngTmp0
   End If
Loop
```

Die Schleife über die Single-Zahlen sngTmp0 in der Do…Loop ist identisch. Für die gleiche Darstellung mit 3 Nachkommastellen wird sngTmp0 jeweils mit 1000 multipliziert und der DWord-Variablen dwdTmp0 zugewiesen, anschließend mit `strNum = str(dwdTmp0)` in einen Zeichen-

String gewandelt und in der Folge mit `strNum = Format(strNum , "0.000")` mit 3 Nachkommastellen formatiert. Damit wird die Single-Zahl richtig angezeigt.

DWord (32 Bit) und Word (16 Bit) mit kleinerem Zahlenbereich sind nur tauglich für positive Zahlen. Sollen auch negative Zahlen dargestellt werden, sind Long (32 Bit) und Integer (16 Bit), beide mit Vorzeichen-Bit, die Wahl.

Der Rest ist identisch mit LCD Format V0.01, einschließlich Sub DisplayReal.

Trotz einiger zusätzlicher Variablen und Codezeilen ist der Hex-Code um 116 Bytes kürzer als der von LCD Format V0.01 mit der Funktion 'Fusing'. Bei der Programmierung von AVR mit wenig Flash könnte das von Vorteil sein.

3 Formatieren von Festkommazahlen

Insbesondere wenn längere Festkommazahlen, z.B. Frequenzen in Hertz, angezeigt werden sollen, verbessert das Einfügen von Tausender-Trennzeichen die Lesbarkeit der LCD-Anzeige. Dafür bietet BASCOM keine fertige Formatierungsfunktion. Hier ist Handarbeit angesagt.

Programmbeispiel LCD Format V0.03 (Ausschnitt):

```
dwdTest = 1234567890                          'Start frequency
dwdTmp0 = dwdTest
bytNumMax = 9                                 'Max number length excl. separators
bytLCDStart = 4                               'Start position of frequency
bytLCDEnd = 14                                'End position of frequency
bytLCDLen = bytLCDEnd - bytLCDStart
bytLCDLen = bytLCDLen + 1                     'Length of frequency string
bytSep = ","                                  'Thousand separator ','
'bytSep = "."                                 'Thousand separator '.'
'bytSep = "-"                                 'No Thousand separator
bytRun = 0
strDis = "f             Hz"                   'Display mask
strSav = Space(16)

Do
   Incr bytRun
   strNum = str(dwdTmp0)
   If strNum <> strSav Then                   'Display if number was changed
      Call DisplayInt(bytLCDLen , bytLCDEnd , bytSep)
      Locate 2 , 1
      LCD strDis
      strSav = strNum
      Wait 2
   End If

   If bytRun = 10 Then
      bytRun = 0
      dwdTmp0 = dwdTest
   Else
      dwdTmp0 = dwdTmp0 / 10
   End If
Loop
```

Darstellung der hohen Frequenz in Hz mit der DWord-Variablen dwdTmp0. Die Display-Maske wird wie oben mit strDis = … festgelegt, dazu der Startpunkt bytLCDStart und der Endpunkt bytLCDEnd für den Anzeigebereich der Frequenz. Die Tausender-Trennzeichen können als ',' oder '.' festgelegt werden; mit '-' werden keine Tausender-Trennzeichen angezeigt.

Der Aufbau der Do…Loop ist vergleichbar mit den obigen Beispielen.

Für die Konvertierung der DWord-Variablen dwdTmp0 in den String strNum reicht die Funktion str ohne gesonderte Formatierung mit Format.

Zahlen in LCD-Anzeige einpassen

DL6GL, 18.06.2022

Die Aufbereitung mit Tausender-Trennzeichen nimmt Sub DisplayInt vor.

```
Sub DisplayInt(byVal bytLenMax As Byte , byVal bytPosLast As Byte , byVal bytSep
As Byte)
'Insert integer number string  bytNum to display string bytDis
'from position bytPosLast backwards, max. bytLenMax characters
'Thousand separators are added.
'Input:  bytNum      (=strNum by overlay) string to be inserted in strDis
'        bytNumMax   Max. digits exclusive thousand separator
'        bytLenMax   Max. possible length of string to be inserted
'        bytPosLast  Right position in bytDis (=strDis by overlay)
'        bytSep      Thousand separator "," or "."
'                    = "-": No separator
'Output: strDis      Completed string to be displayed on LCD

   bytLen = Len(strNum)
   If bytLen > bytNumMax Then                 'No room, overflow
      For bytTmp0 = bytLCDStart To bytLCDEnd
          bytDis(bytTmp0) = "#"
      Next bytTmp0
      Exit Sub
   Else
      For bytTmp0 = bytLCDStart To bytLCDEnd
         bytDis(bytTmp0) = " "
      Next bytTmp0
   End If
   bytTmp1 = bytLCDEnd + 1
   bytDigit = 0
   For bytTmp0 = bytLen To 1 Step -1          'From right to left
       bytDigit = bytDigit + 1                'Current digit number from right
       bytTmp1 = bytTmp1 - 1
       bytTmp2 = bytDigit Mod 3
       If bytTmp2 = 0 Then                    'Next positon is thousand separator
          bytDis(bytTmp1) = bytNum(bytTmp0)
          If bytTmp0 > 1 And bytSep <> "-" Then   'Add separator
            bytTmp1 = bytTmp1 - 1
            bytDis(bytTmp1) = bytSep
          End If
       Else
         bytDis(bytTmp1) = bytNum(bytTmp0)
       End If
   Next bytTmp0

End Sub
```

Wie vorher wird zunächst geprüft, ob der Zahlen-String strNum in den verfügbaren Platz (bytNumMax) passt. Wenn nicht, wird der Anzeigebereich in strDis mit einem Überlaufbalken ######## gefüllt.
Wenn ja, wird der Anzeigebereich zunächst mit Blanks, anschließend von rechts nach links (rückwärts) mit den Zeichen aus strNum gefüllt. Sind jeweils 3 Zeichen bearbeitet, wird das festgelegte Tausender-Trennzeichen anschließend (links davor) eingefügt (oder auch nicht).

Ergebnis z.B.