Rainbow.lib v1.0

25.Juli 2014

- Unterstützt bis zu 8 LED-Bänder unabhängig voneinander.
- LED-Bänder mit einer Länge > 256 können bedient werden (nicht erprobt)
- Swap implementiert
- Rotate_Left implementiert
- Rotate_Right implementiert
- SetColor implementiert
- SetTableColor implementiert
- Reset implementiert
- Clear implementiert
- Implementierte Treiber: WS2812b [8 / 9.6 / 14.7456 / 16 MHz]

Die Library ermöglicht eine einfache Verwendung von LED-Streifen. Dazu dimensioniert, verwaltet und bedient die Libary die nötigen Speicherräume.

Mindestvorraussetzungen

64Byte SRAM, 1kByte FlashROM, 8MHz

Initialisierung

Um die Library verwenden zu können sind die folgenden Zeilen in den Programmcode zu integrieren:

```
$Lib "Rainbow.lib"  'Implementierung
$external WS2812b   'Driver

Declare sub BOW_INIT()

Declare sub BOW_Reset()

Declare sub BOW_Clear(Byval Set_Direct as Byte)

Declare sub BOW_Select(Bylabel WS2812b_Channel as Word)

Declare Sub BOW_SetColor(Byval LedNr as Word , Byval Color as long , _
            Byval Set_Direct as Byte)

Declare Sub BOW_SetTableColor(Byval LedNr as Word , _
            Byval ColorIndex as Byte ,  Byval Set_Direct as Byte)

Declare Sub BOW_Swap(Byval Led1 as Word , Byval Led2 as word , _
            Byval Set_Direct as Byte)

Declare Sub BOW_Rotate_left(Byval Left_Index as Word , Byval Width as Word,_
            Byval Set_Direct as Byte)

Declare Sub BOW_Rotate_right(Byval Left_Index as Word , Byval Width as Word,_
            Byval Set_Direct as Byte)
```

Implementierung

Um die Routinen der Library auf einen LED-Streifen durch den gewählten Treiber anwenden zu können, muss mindestens ein Kanal implementiert werden.

```
'#CHANNEL0
Const BOW_CH0_Len = 10      'Stripe len
Const BOW_CH0_Port = PortC
Const BOW_CH0_Pin = PC3
```

Die Konstantennamen sind fest vorgegeben und beginnen wie auch die Routinen mit „BOW“, gefolgt von einer Kanalbezeichnung, z.B. „CH1“ und der Zweckbindung, z.B. „Port“, jeweils durch einen Unterstrich getrennt.

Momentan werden bis zu 8 Kanäle unterstützt, Ch0 bis Ch7.

`BOW_CH1_Len` - Die Anzahl der LED die der Kanal verwalten soll
`BOW_CH1_Port` - der Port und Pin, die dem der Kanal zugeordnet werden sollen
`BOW_CH1_Pin`

Es ist durchaus möglich gleich mehrere Kanäle auf denselben Pin anzuwenden.

Ein explizites Konfigurieren eines Pins als Ausgang ist nicht erforderlich, - das erledigt der folgende aufzurufende Befehl gleich mit.

```
Call BOW_Init()
```

Routinen – die Gemeinsamkeiten

- Damit eine aufgerufene Routinen auf einen gewünschten Kanal angewendet wird, muss dieser zuvor selektiert werden.

  ```
  Call BOW_select(BOW_CH0)
  ```

  Solange der Kanal nicht gewechselt werden soll, oder es wird nur ein Kanal verwendet, genügt es, diesen Befehl nur einmal auszuführen.

- Jede Routine erwartet den Parameter `Set_Direct`, der 1 oder 0 sein kann. Damit wird bestimmt ob direkt nach Abarbeitung des Befehls, eine Ausgabe auf den dem Kanal zugeordneten Pin erfolgen soll.

  Zur besseren Lesbarkeit empfiehlt es sich daher, für diese Parameter symbolische Konstanten anzulegen:

  ```
  Const Send = 1
  Const NoSend = 0
  ```

- Index-Parameter sind grundsätzlich nullbasierend, LED Indizes darüber hinaus 2 Byte groß.

Farbe

Die Library stellt zwei Routinen bereit um einer LED eine Farbe zuzuordnen.

- **SetColor**

```
Sub BOW_SetColor(Byval LedNr as Word , _
                 Byval Color as long , _
                 Byval Set_Direct as Byte)
```

Mit `BOW_SetColor` kann einer LED eine Farbe in Form einer Konstante zugewiesen werden.
Der Parameter `Color` (4 Byte) kann als symbolische Konstante oder direkt übergeben werden.

```
'     Name =  RRGGBB
Const Rot  = &HFF0000
Const Blau = &H00FF00
Const Gruen = &H0000FF
Const Chuck_Berry_Red = &HDD2010
Const Peppermint = &H10ff30

Call BOW_SetColor(0, Peppermint , NoSend)   '  1. LED
Call BOW_SetColor(1, Blau , Send)           '  2. LED + aktualisieren
```

- **SetTableColor**

```
Sub BOW_SetTableColor(Byval LedNr as Word , _
                      Byval ColorIndex as Byte , _
                      Byval Set_Direct as Byte)
```

RGB-Daten können auch zu einer Tabelle zusammengefasst werden. Die Library stellt speziell dazu einen Befehl zum Auslesen der 3 Byte langen RGB-Daten bereit.
Diese Daten sind unter dem Label `Rainbow_Colors:` aufzuführen und mit folgendem Befehl zu verwenden.

```
Call BOW_SetTableColor(0, 2, Send) ' 1. Led blau

Rainbow_Colors:
   '      R  ,  G  ,  B                    index
   Data &HFF , &H00 , &H00       'Red        0
   Data &H00 , &HFF , &H00       'Green      1
   Data &H00 , &H00 , &HFF       'Blue       2
   Data &HFF , &HA5 , &H00       'Orange     3
   Data &HFF , &HFF , &H00       'Yellow     4
   Data &HFF , &H69 , &HB4       'HotPink    5
```

Der Zugriff auf die Tabelle ist standardmäßig aktiviert, d.h. der Compiler erwartet das Label `Rainbow_Colors:`.
Soll nicht mit der Farbtabelle gearbeitet werden, kann diese Option mit `Const NoSetTableColor = 1` abgeschaltet werden.

Farbbewegung mit SWAP

```
Sub BOW_Swap(Byval Led1 as Word ,
             _Byval Led2 as word ,
             _Byval Set_Direct as Byte)
```

Swap tauscht die Farbinhalte zweier angegebener LED-Positionen miteinander.

Beispiel:
```
Call BOW_Swap(0 , 3, send) 'tausche 1 und 4 LED
```

Farbbewegung mit `Rotate_left` und `Rotate_Right`

Mit diesen Befehlen können alle Farben oder auch nur Farben einzelner Abschnitte einer LED-Kette bewegt werden.

Mit `LeftIndex` wird die erste LED angegeben, mit `Width` die Anzahl der LED dessen Farben einer Positionsverschiebung unterzogen werden sollen.

```
Declare Sub BOW_Rotate_left(Byval LeftIndex as Word , _
                            Byval Width as Word , _
                            Byval Set_Direct as Byte)


Declare Sub BOW_Rotate_right(Byval LeftIndex as Word , _
                             Byval Width as word , _
                             Byval Set_Direct as Byte)



Dim i as word

Call BOW_SetColor (0 , Chuck_berry_red , Send) '1.  LED
Call BOW_SetColor (9 , Chuck_berry_red , Send) '10. LED

For i = 1 to 8 '2. bis 9. LED
  Call BOW_SetColor (i , Peppermint , Send)
Next

'EFFEKT- zwei Abschnitte der Länge 5 scheinen sich von ihrer gemeinsamen Mitte
voneinander weg zu bewegen.

Do
   Call BOW_Rotate_left(0 , 5 , nosend)
   Call BOW_Rotate_right(5 , 5 , send)
   Waitms 200
loop
```

Clear und Reset

```
Sub BOW_Reset()
```

Löscht die Farben der gesamten LED-Kette. Die Farbinformationen im Array des aktiven Kanals bleiben erhalten.

```
Sub BOW_Clear(Set_Direct)
```

Löscht die Farbinformationen im Array des aktiven Kanals.

```
Set_Direct  = Send       ´ =1 :Farben der LED-Kette werden gelöscht
Set_Direct  = NoSend     ´ =0 :Farben der LED-Kette bleiben erhalten
```

Weiterführende Informationen

Datenverwaltung

Die Library legt einige Variablen zur Verwaltung und ein für jeden definierten Kanal zugehöriges Array an.

Verwaltung:

- BOW__Cur_Port Byte
- BOW__Cur_Pin Byte
- BOW__Cur_Size Word
- BOW__Cur_ADR Word

Kanalgebundenes Array:

- BOW_ch**n**_(BOW_CH**n**_Len *3) 'Byte-Array

Darüber hinaus werden die für jeden angemeldeten Kanal zugehörigen Parameter in eine Konstantentabelle, als fester Bestandteil des Programmcodes eingebunden und bei Bedarf aus dem Programmspeicher geladen.
Gespeichert sind dort der Port (1 Byte) und der Pin (1 Byte), sowie die Adresse des angelegten Arrays (2 Byte) und dessen Länge (2 Byte). Mit **BOW_select**(BOW_CH**n**) werden die entsprechenden Informationen in die Verwaltungsvariablen überführt.

Ausgabe-Pin zur Laufzeit ändern

Bedingt dadurch, dass die Port-Pin Kombination beim selektieren eines Kanals in das SRAM, bzw. in Variablen geladen wird, ist es möglich die Ausgabe eines Kanals zur Laufzeit auf einen anderen Pin umzuleiten.
Der Befehl **BOW_select**(BOW_CH**n**) stellt dann wieder den Urzustand her. Hinter BOW_CH**n** verbirgt sich die Adresse unter der die Parameter des angegebenen Kanals abzurufen sind, siehe *Datenverwaltung*.

Die Zuweisung einer Portadresse gelingt in folgender Weise:

```
Const PORTneu = PORTB + &H20
BOW__Cur_Port = PORTneu
```

Der Pin kann direkt als Ganzzahl von 0 bis 7 übergeben werden.
Wird der Portpin nicht schon für einen anderen Kanal verwendet, muss sichergestellt sein, dass dieser als Ausgang konfiguriert ist.

Codereduzierung

Standardmäßig wird die Library mit ihrem vollen Funktionsumfang kompiliert, was bei nicht Gebrauch einiger Routinen zu einer unnötigen Verschwendung wertvollen Programmspeichers führt.
Um dem entgegenzutreten können einige Routinen „abgewählt“ werden.

```
Const NoSetTableColor = 1
Const NoSwap = 1
Const NoRotate = 1
```

Stapelspeicherkonsum

BEFEHL(PARAMETER)	HW-stack	SW-stack	Frame
BOW_Init()	4	0	0
BOW_Select(Word)	2	2	2
BOW_SetColor(Word, Long, Byte)	4 / 8*	6	7
BOW_SetTableColor(Word , Byte, Byte)	4 / 8*	6	6
BOW_Swap(Word ,Word, Byte)	4 / 8*	6	5
BOW_Rotate_left(Word, Word, Byte)	4 / 8*	6	7
BOW_Rotate_right(Word, Word, Byte)	4 / 8*	6	7
BOW_Reset()	6	0	0
BOW_Clear(Byte)	2/ 6*	2	1

* Option: Send