Max7219 mit 8x8 Ledmatrix, Beschreibung der Ansteuerung an Hand eines Beispiels

In diesem Artikel möchte ich das Ansteuern des Max7219, in Kombination mit einer 8x8 Ledmatrix, beschreiben.
Hier ein link zum Datenblatt Datenblatt als PDF
Ich beziehe mich auf diesen Modultyp.
Max7219-Modul.jpgBeschriftung.jpgAnreihungen.jpgVerdrahtung.jpg
Es gibt noch andere Bauformen, die aber nicht so elegant aneinander anreihbar sind. Ob die Anzeigen mit diesen Typen die gleichen wären, kann ich nicht beurteilen, ich habe nur die abgebildeten.

Das Modul:
Jede einzelne Led kann per Befehl an- oder ausgeschaltet werden. Die Helligkeit kann in 16 Stufen eingestellt werden. Zur Ansteuerung sind nur 3 Outputpins vom Kontroller nötig, selbst wenn viele Module aneinander gereiht werden. Das Modul benötigt eine Betriebsspannung von 5V.

Die Anschlüsse:
Die Anschlüsse sind auf der Platine beschriftet, VCC ist +5V Betriebsspannung, DIN ist der Dateneingang, CLK ist der Takteingang für die serielle Datenübertragung, CS steuert die Übernahme der Daten an die Logik des Max7219, GND ist der Masseanschluß.

Das Ansteuern:
Der Max7219 hat intern ein Schieberegister, das 2 bytes aufnehmen kann. Diese 2 byte werden bitweise über den DIN-Eingang und dem CLK-Eingang 'eingetaktet'. Das erste byte stellt einen 'Befehl' dar, das zweite byte 'Daten' für diesen Befehl. Werden weitere bits 'eingeschoben', erscheinen die vorher eingeschobenen bit am DOUT-Ausgang des Moduls. Jedes Modul kann also 16 bits zwischen speichern. Hat man beispielsweise 2 Module aneinander gereiht, und schiebt 2 bytes (byte 1 und byte 2) in den Din ein, befinden sich diese 2 bytes im ersten Modul, schiebt man weitere 2 bytes (byte 3 und byte 4)hinter her, werden die ersten beiden bytes aus dem ersten Modul über DOUT und DIN des zweiten Moduls in das zweite Modul geschoben und byte 3 und byte 4 befinden sich im Speicher vom ersten Modul. So kann man in jedes Modul durch passendes Anordnen der bytes den gewünschten Befehl platzieren. Um die jetzt für jedes Modul zugeordneten Befehle wirksam werden zu lassen, kommt der Anschluss CS in's Spiel. Ein Übergang von Pegel low zu Pegel high löst in den Modulen die Abarbeitung des Befehls aus.

Für dieses Modul sind folgende Befehle interessant:
'Matrixmodus':
Das sind 2 Befehle, die die Ansteuerung der Led-Matrix mit dem Max ermöglichen, im Gegensatz zur Ansteuerung von 7-Segmentdisplays.
Befehlsbyte =&H09, Datenbyte=0: Kein Decodieren der Daten für die Ausgabe an 7_Segmentdisplays.
Befehlsbyte =&H0B, Datenbyte=7: Alle Zeilen der Matrix sollen ansprechbar sein.

Shutdown mode:
Damit kann die komplette Matrix aus- bzw eingeschaltet werden. Nützlich zum Stromsparen, wenn keine Anzeige gebraucht wird, oder zum Blinken lassen, wenn die Matrix wechselweise ein- bzw ausgeschaltet wird.
Befehlsbyte =&H0C, Datenbyte=1: Shutdown=aus, Normalbetrieb, Datenbyte=0: Shutdown=ein, keine Anzeige.

Helligkeit:
Hiermit kann die Helligkeit in 16 Stufen eingestellt werden.
Befehlsbyte =&H0A, Datenbyte zwischen 0, 1/32tel Helligkeit (dunkel) bis 15, 31/32tel Helligkeit (maximal hell).

Displaytest: Die Komplette Matrix kann für einen Funktionstest ein- und wieder ausgeschaltet werden.
Befehlsbyte =&H0F, Datenbyte=0, kein Test, Datenbyte=1 zum Testen alle Leds an.

'Zeile schreiben':
Der Befehl ist dabei die Zeilennummer der Matrix und die Datenbits geben an, ob eine Led in dieser Zeile eingeschalten (bit ist auf 1 gesetzt) oder ausgeschalten wird (bit ist auf 0 gesetzt)
Befehlsbyte = Zeilennummer von 1 bis 8, Datenbyte bit entspricht der Led in der Zeile.

Das Format der Daten:

Die bytes werden bitweise in das Modul eingetaktet, wobei mit dem bit7 (MSB) begonnen wird und bit0 (LSB) das byte abschließt. Im Datenblatt steht noch: damit das angelegte Datenbit in den Speicher übernommen wird, muss der CLK-Anschluss von low auf high wechseln. Das muss man wissen, um den bascom-Befehl shiftout richtig zu konfigurieren. Dieser Befehl ist ideal zum Ansteuern des Max7219 geeignet.
shiftout.PNG
Hier ein Auszug aus der bascom-Hilfe.

Die Zuordnung der Leds:
Die Matrix kann man in Zeilen unterteilen, in jeder Zeile entspricht eine Led einem bit in dem Datenbyte
bit-Verteilung.jpg

Das war jetzt die Theorie, jetzt wird programmiert:
Der Jahreszeit entsprechend möchte ich auf 2 aneinander gereihten Modulen eine Kerze brennen lassen. Die Module werden nach dem Anreihen um 90° gedreht, der DIN-Anschluss befindet sich unten.

Der Max ist ja auch für 7-Segmentdisplays geeignet, daher muss man ihn für den Betrieb mit einer Matrix zu aller erst konfigurieren.

BASCOM-Quellcode

```
'Kerze mit 2 Max7219 Led8x8 Matrix
$regfile = "m168pdef.dat"
$crystal = 8000000
$hwstack = 34
$swstack = 32
$framesize = 64
'Alias
Din Alias Portd.0 'Mit DIN an der Max-Platine verbinden
Clk Alias Portd.1 'Mit Clk an der Max-Platine verbinden
Cs Alias Portd.2 'Mit Cs an der Max-Platine verbinden
'ports
Config Din = Output
Config Clk = Output
Config Cs = Output
'Variablen
Dim Befehl As Byte , Disp_daten As Byte
Dim N As Byte 'Zähler
Dim M As Byte 'Zähler
Dim Tabellenstart As Byte 'Nummer des Tabelleneintrags
Dim Pause As Word 'msec zwischen den einzelnen Flammenbildern
'Start Programm
Pause = 80
Cs = 1
Gosub Init_max7219 'initialisieren für die Verwendung mit der 8x8 Matrix
Do
Loop
End
Init_max7219: ' Initialisieren des MAX7219
Befehl = &H09 : Disp_daten = &H00 ' kein decode "B" für alle Digits weil 8x8 Matrix
Gosub Led_write_command
Befehl = &H0A : Disp_daten = &H04 ' Helligkeit 0-F
Gosub Led_write_command
Befehl = &H0B : Disp_daten = 7 ' Anzahl Digits 8 weil 8x8 Matrix
Gosub Led_write_command
Befehl = &H0F : Disp_daten = 0 ' Test Off
Gosub Led_write_command
Befehl = &H0C : Disp_daten = 1 ' Shutdown Off
Gosub Led_write_command
Return
```

```
46. Led_write_command:
47. Cs = 0
48. For N = 1 To 2 '2 Displays sind angeschlossen
49. Shiftout Din , Clk , Befehl , 1
50. Shiftout Din , Clk , Disp_daten , 1
51. Next N
52. Cs = 1 'Flanke von 0 zu 1 überträgt den Befehl und die Daten in den Max
53. Return
```

Alles anzeigen
Im code erkennt man das 'Befehlsbyte' (siehe auch Datenblatt) und das zugehörige 'Datenbyte'. Ich verwende 2 Module, also werden die Befehle 2x hintereinander gesendet, damit beide Module die Befehle im Speicher haben. Als Abschluss wird CS auf 1 gesetzt, um die Befehle auszuführen.
Jetzt sind beide Module eingerichtet und bereit, die Kerze anzuzeigen. Ja, wo ist die denn? Auf der oberen Matrix soll die Flamme angezeigt werden. Dazu müssen die passenden Leds der einzelnen Zeilen der Matrix gesetzt werden. 8 Zeilen der Matrix entsprechen 8 bytes. Da jetzt die Matrix um 90° gedreht ist, kann man von Spalten sprechen. Die unterste Led in einer Spalte entspricht bit0, die linkeste Spalte hat die Nummer 8. Also 8 bytes als data-Elemente eingetippt.

BASCOM-Quellcode

```
1. Kerzenflamme:
2. Data &B00000000
3. Data &B00001100
4. Data &B00111110
5. Data &B01111111
6. Data &B00111110
7. Data &B00001100
8. Data &B00000000
9. Data &B00000000 '
```

Neigt man den Kopf nach links, kann man mit etwas Mühe eine stilisierte Flamme in der binären Darstellung erkennen. Für die untere Matrix brauche ich 8 bytes, die den Kerzenstummel darstellen.

BASCOM-Quellcode

```
1. Kerze:
2. Data &B00000001
3. Data &B01111111
4. Data &B01111111
5. Data &B11111111
6. Data &B01111111
7. Data &B01111111
8. Data &B11111111
9. Data &B00000001 '
```

Damit ein bisschen Bewegung rein kommt, habe ich eine Abfolge von Flammen rein gedrückt, die in schneller Folge angezeigt werden sollen. Das sind die weiteren data-Zeilen.
Jetzt fehlt nur noch das Programm, was als 'Befehlsbyte' die Spaltennummer der Matrix aus gibt und das entsprechende byte für die Flamme und den Kerzenstummel.
Das schaut so aus:

BASCOM-Quellcode

```
1. 'Kerze mit 2 Max7219 Led8x8 Matrix
2. $regfile = "m168pdef.dat"
```

```
$crystal = 8000000
$hwstack = 34
$swstack = 32
$framesize = 64
'Alias
Din Alias Portd.0 'Mit DIN an der Max-Platine verbinden
Clk Alias Portd.1 'Mit Clk an der Max-Platine verbinden
Cs Alias Portd.2 'Mit Cs an der Max-Platine verbinden
'ports
Config Din = Output
Config Clk = Output
Config Cs = Output
'Variablen
Dim Befehl As Byte , Disp_daten As Byte
Dim N As Byte 'Zähler
Dim M As Byte 'Zähler
Dim Tabellenstart As Byte 'Nummer des Tabelleneintrags
Dim Pause As Word 'msec zwischen den einzelnen Flammenbildern
'Start Programm
Pause = 80
Cs = 1
Gosub Init_max7219 'initialisieren für die Verwendung mit der 8x8 Matrix
Do
For M = 0 To 10 'Muster der Kerze durchwechseln
Tabellenstart = M * 8 'Tabelleneinsprung berechnen für die jeweilige Flamme
For N = 0 To 7 'Bytenummer
Cs = 0
Befehl = N + 1 'Zeilennummer der Matrix
Disp_daten = Lookup(tabellenstart , Kerzenflamme) 'das letzte Display zuerst zeigt die Flamme
Shiftout Din , Clk , Befehl , 1
Shiftout Din , Clk , Disp_daten , 1
Disp_daten = Lookup(n , Kerze) 'das erste Display nach dem AVR zeigt den Kerzenstummel
Shiftout Din , Clk , Befehl , 1
Shiftout Din , Clk , Disp_daten , 1
Cs = 1 'low auf high-Flanke, Max übernimmt den Befehl und Daten
Incr Tabellenstart
Next N
Waitms Pause 'steuert die 'Bildfrequenz' zwischen den einzelnen Flammenbildern
Next M
Loop
End
Init_max7219: ' Initialisieren des MAX7219
Befehl = &H09 : Disp_daten = &H00 ' kein decode "B" für alle Digits weil 8x8 Matrix
Gosub Led_write_command
Befehl = &H0A : Disp_daten = &H04 ' Helligkeit 0-F
Gosub Led_write_command
Befehl = &H0B : Disp_daten = 7 ' Anzahl Digits 8 weil 8x8 Matrix
Gosub Led_write_command
Befehl = &H0F : Disp_daten = 0 ' Test Off
Gosub Led_write_command
Befehl = &H0C : Disp_daten = 1 ' Shutdown Off
Gosub Led_write_command
Return
Led_write_command:
Cs = 0
For N = 1 To 2 '2 Displays sind angeschlossen
Shiftout Din , Clk , Befehl , 1
Shiftout Din , Clk , Disp_daten , 1
```

```
Next N
Cs = 1 'Flanke von 0 zu 1 überträgt den Befehl und die Daten in den Max
Return
Kerzenflamme:
Data &B00000000
Data &B00001100
Data &B00111110
Data &B01111111
Data &B00111110
Data &B00001100
Data &B00000000
Data &B00000000 '
Data &B00000000 , &B00000100 , &B00001110 , &B00011111 , &B01111110 , &B00011100 , &B00000000 , &B00000000 '
Data &B00000000 , &B00000100 , &B00001110 , &B00001111 , &B00011110 , &B01111000 , &B00000000 , &B00000000 '
Data &B00000000 , &B00000000 , &B00000110 , &B00001111 , &B00001110 , &B00011100 , &B01110000 , &B00000000 '
Data &B00000000 , &B00000000 , &B00000110 , &B00001111 , &B00011110 , &B00111000 , &B00000000 , &B00000000 '
Data &B00000000 , &B00000000 , &B00000110 , &B00001111 , &B00111110 , &B00001100 , &B00000000 , &B00000000 '
Data &B00000000 , &B00000100 , &B00001110 , &B00111111 , &B00001110 , &B00000000 , &B00000000 , &B00000000 '
Data &B00000000 , &B00000100 , &B00111110 , &B00011111 , &B00001110 , &B00000000 , &B00000000 , &B00000000 '
Data &B00000000 , &B01110100 , &B00111110 , &B00011111 , &B00001110 , &B00000000 , &B00000000 , &B00000000 '
Data &B00000000 , &B00011100 , &B01111110 , &B00011111 , &B00001110 , &B00000000 , &B00000000 , &B00000000 '
Data &B00000000 , &B00000100 , &B00011110 , &B00111111 , &B00011110 , &B00000100 , &B00000000 , &B00000000 '
Kerze:
Data &B00000001
Data &B01111111
Data &B01111111
Data &B11111111
Data &B01111111
Data &B01111111
Data &B11111111
Data &B00000001 '
```

Alles anzeigen
Die Variable M ist zuständig, welches Bild der Flamme angezeigt wird, Disp_daten enthält ein byte aus den data-Tabellen und Befehl gibt die Spaltennummer auf der Matrix an.

Auch hier sieht man wieder, es werden für jedes Modul 2 bytes mit shiftout ausgegeben und mit dem Setzen von CS die Verarbeitung im Max ausgelöst.
Und wie ist das Ergebnis?
So
[/mediach