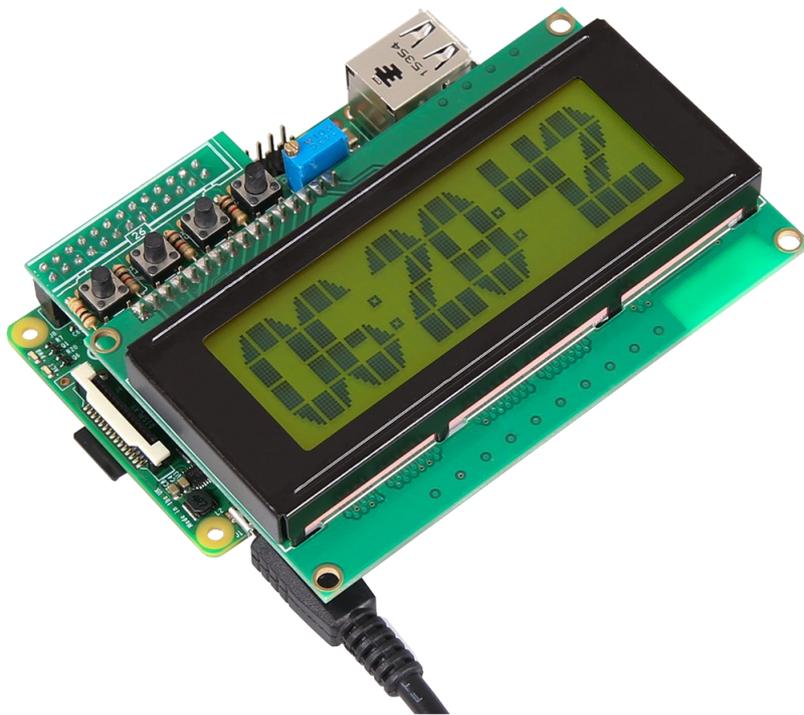


# JOY-IT

## LCD Display 20x4 mit Buttons



Index

---

1. Anschluss des Displays
2. Installation des Systems
3. Verwendung & Beispiel-Code
4. Support

Sehr geehrter Kunde,

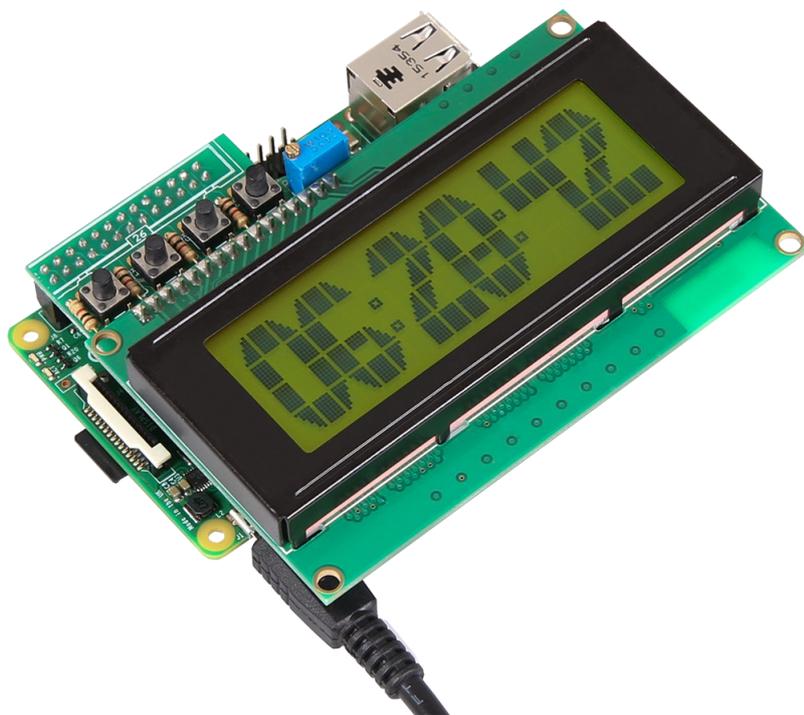
vielen Dank, dass Sie sich für unser Produkt entschieden haben.

Im Folgenden haben wir aufgelistet, was bei der Inbetriebnahme zu beachten ist:

## 1. Anschluss des Displays

Stecken Sie das Display so auf den Pin-Header Ihres Raspberry Pi auf, dass sich die Display-Einheit genau über Ihrem Raspberry Pi befindet.

Sobald Sie die Stromversorgung zu Ihrem Raspberry Pi herstellen, sollte nun auch die Hintergrundbeleuchtung des Displays zu leuchten beginnen.



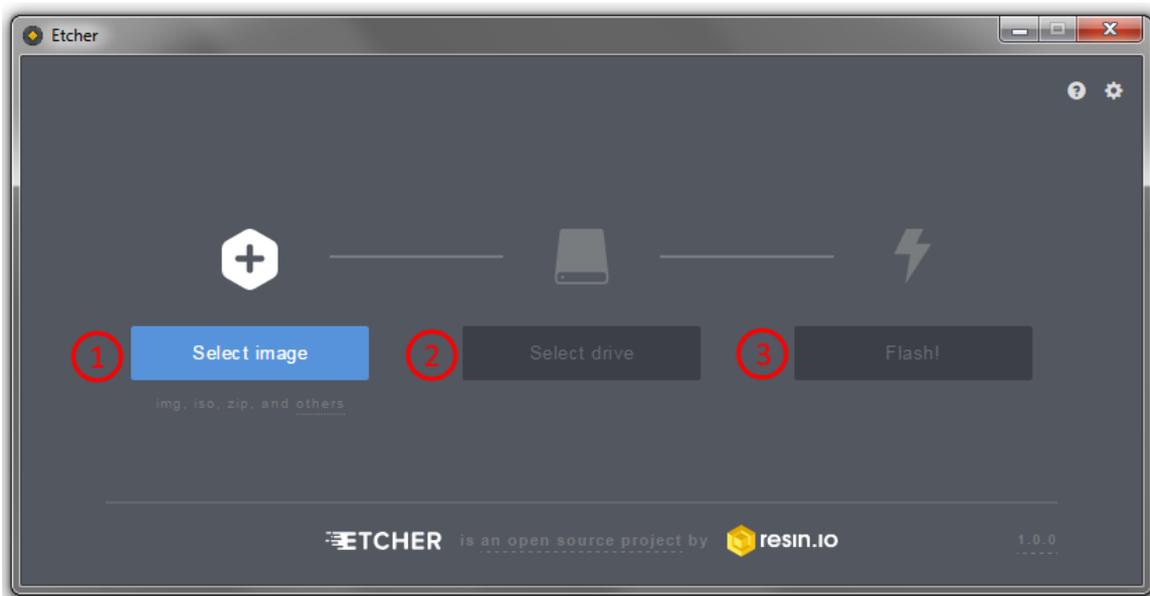
## 2. Installation des Systems

Wenn Sie auf Ihrem Raspberry Pi bereits die aktuellste Raspbian Software verwenden, können Sie diesen Schritt überspringen und sofort mit dem nächsten Schritt fortfahren. Andernfalls folgen Sie bitte den weiteren Anweisungen.

Installieren Sie auf Ihre SD-Karte bitte das aktuellste Raspbian System-Image.

Dieses können Sie [hier](#) herunterladen.

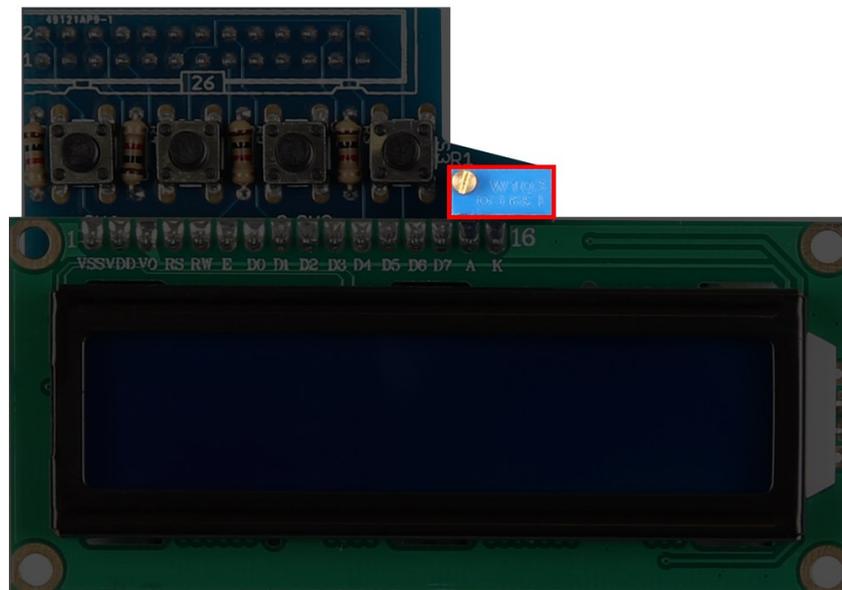
Mit einem geeigneten Programm (z.B. Etcher) können Sie das Image dann auf Ihre SD Karte übertragen. Nach erfolgreicher Übertragung kann die SD Karte in den Raspberry Pi eingelegt und das System gestartet werden.



### 3. Verwendung & Code-Beispiel

Das Display ist nach dem Anschluss einsatzbereit und bedarf keiner weiteren Installation. Es ist unter Umständen möglich, dass Sie zunächst den Kontrast manuell anpassen müssen, bevor Sie auf dem Display eine Ausgabe erkennen können.

Justieren Sie hierfür den Kontrast, indem Sie mit einem kleinen Schraubenzieher an der dafür vorgesehenen Einstell-Schraube drehen.



Um das Display nun vollständig in Betrieb zu nehmen, können Sie sich entweder den Beispiel-Code [herunterladen](#), oder selbst eine neue Datei erstellen und den nachfolgenden Code dort einfügen.

Um eine neue Datei zu erstellen, verwenden Sie den folgenden Befehl:

```
sudo nano lcd20x4.py
```

```

import time
import RPi.GPIO as GPIO

# PIN-Configuration
LCD_RS = 7 #GPIO7 = Pi pin 26
LCD_E = 8 #GPIO8 = Pi pin 24
LCD_D4 = 17 #GPIO17 = Pi pin 11
LCD_D5 = 18 #GPIO18 = Pi pin 12
LCD_D6 = 27 #GPIO21 = Pi pin 13
LCD_D7 = 22 #GPIO22 = Pi pin 15
OUTPUTS = [LCD_RS,LCD_E,LCD_D4,LCD_D5,LCD_D6,LCD_D7]

#Button-PINS
SW1 = 4 #GPIO4 = Pi pin 7
SW2 = 23 #GPIO16 = Pi pin 16
SW3 = 10 #GPIO10 = Pi pin 19
SW4 = 9 #GPIO9 = Pi pin 21
INPUTS = [SW1,SW2,SW3,SW4]
#HD44780 Controller Commands
CLEARDISPLAY = 0x01
SETCURSOR = 0x80

#Line Addresses. (Pick one. Comment out whichever doesn't apply)
LINE = [0x00,0x40,0x14,0x54] #for 20x4 display
#LINE = [0x00,0x40] #for 16x2 display
#####

def InitIO():
    #Sets GPIO pins to input & output, as required by LCD board
    GPIO.setmode(GPIO.BCM)
    GPIO.setwarnings(False)
    for lcdLine in OUTPUTS:
        GPIO.setup(lcdLine, GPIO.OUT)
    for switch in INPUTS:
        GPIO.setup(switch, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def CheckSwitches():
    #Check status of all four switches on the LCD board
    val1 = not GPIO.input(SW1)
    val2 = not GPIO.input(SW2)
    val3 = not GPIO.input(SW3)
    val4 = not GPIO.input(SW4)
    return (val4,val1,val2,val3)

```

```

def PulseEnableLine():
    #Pulse the LCD Enable line; used for clocking in data
    mSec = 0.0005 #use half-millisecond delay
    time.sleep(mSec) #give time for inputs to settle
    GPIO.output(LCD_E, GPIO.HIGH) #pulse E high
    time.sleep(mSec)
    GPIO.output(LCD_E, GPIO.LOW) #return E low
    time.sleep(mSec) #wait before doing anything else

def SendNibble(data):
    #sends upper 4 bits of data byte to LCD data pins D4-D7
    GPIO.output(LCD_D4, bool(data & 0x10))
    GPIO.output(LCD_D5, bool(data & 0x20))
    GPIO.output(LCD_D6, bool(data & 0x40))
    GPIO.output(LCD_D7, bool(data & 0x80))

def SendByte(data,charMode=False):
    #send one byte to LCD controller
    GPIO.output(LCD_RS,charMode) #set mode: command vs. char
    SendNibble(data) #send upper bits first
    PulseEnableLine() #pulse the enable line
    data = (data & 0x0F)<< 4 #shift 4 bits to left
    SendNibble(data) #send lower bits now
    PulseEnableLine() #pulse the enable line

def InitLCD():
    #initialize the LCD controller & clear display
    SendByte(0x33) #initialize
    SendByte(0x32) #set to 4-bit mode
    SendByte(0x28) #2 line, 5x7 matrix
    SendByte(0x0C) #turn cursor off (0x0E to enable)
    SendByte(0x06) #shift cursor right
    SendByte(CLEARDISPLAY) #remove any stray characters on display
    #####

def SendChar(ch):
    SendByte(ord(ch),True)

def ShowMessage(string):
    #Send string of characters to display at current cursor position
    for character in string:
        SendChar(character)

```

```
def GotoLine(row):
    #Moves cursor to the given row
    #Expects row values 0-1 for 16x2 display; 0-3 for 20x4 display
    addr = LINE[row]
    SendByte(SETCURSOR+addr)

#####
# Main Program
print "LCD program starting. Press CTRL+C to stop."
InitIO()
InitLCD()
ShowMessage('Press a button!')
while (True):
    GotoLine(1)
    switchValues = CheckSwitches()
    decimalResult = " %d %d %d %d" % switchValues
    ShowMessage(decimalResult)
# time.sleep(0.2)
```

Die Datei können Sie mit der Tastenkombination **STRG+O** können Sie die Datei speichern und den Editor mit **STRG+X** verlassen.

Anschließend können Sie die Datei mit dem folgenden Befehl ausführen:

```
sudo python lcd20x4.py
```

## 4. Support

Wir sind auch nach dem Kauf für Sie da. Sollten noch Fragen offen bleiben oder Probleme auftauchen stehen wir Ihnen auch per E-Mail, Telefon und Ticket-Supportsystem zur Seite.

E-Mail: [service@joy-it.net](mailto:service@joy-it.net)

Ticket-System: <http://support.joy-it.net>

Telefon: +49 (0)2845 98469 – 66 (11- 18 Uhr)

Für weitere Informationen besuchen Sie unsere Website:

[www.joy-it.net](http://www.joy-it.net)