# MATLAB und Arduino

**Allgemeines:**
MATLAB werden Sie in zwei weiteren Messtechnik-Praktikumsversuchen benötigen und in vielen Lehrveranstaltungen im Laufe Ihres weiteren Studiums.

**Hinweise zur Installation auf Ihrem Rechner und Inbetriebnahme**

1. Aktuelles Arduino Studio installieren, siehe https://www.arduino.cc/
   Dies wird für die Treiber benötigt und dafür, dass MATLAB auf den Arduino zugreifen kann.
2. In MATLAB die **add ons** für den Arduino installieren, d.h. das **MATLAB support package for Arduino** Hardware auswählen und anklicken. Dazu brauchen Sie Ihre Kennung und Ihr Passwort für MATLAB. Eventuell müssen Sie das package zuvor noch herunterladen.
3. Einige MATLAB Beispiele ausprobieren, die z.B. in der Hilfe zu finden sind, um einen Eindruck fürs Programmieren zu bekommen.

**Hardware**
Falls Sie keinen Arduino Uno besitzen, bekommen Sie einen bei den Präsenzversuchen ausgeliehen. Das notwendige Shield für den Versuch haben Sie in der Elektronik Praxis selbst hergestellt.

**Aufgabenstellung**
**Die Anleitung ist absichtlich relativ vage und nicht so detailliert. Viele der Aufgabenstellungen erfordern daher unter anderem die Verwendung der Hilfe Funktion von MATLAB und die Verwendung des Internet. Aber es ist ja auch ein wesentliches Ziel, dass Sie lernen, selbst etwas auf die Beine zu stellen.**

Schreiben Sie bitte für die Ziele je ein Matlab-Skript (= m-file) mit dem Namen „Nachname_Vorname_Arduino_Ziel_x.m“ mit x = 1, 2, 3 oder 4.

**Ziel 1**

Bringen Sie den Arduino unter MATLAB zum Laufen. Dazu gibt es bei MATLAB und im Internet genügend Informationen. Es gehört zur Aufgabenstellung, sich die notwendigen Infos selbst zu besorgen.

Auf Ihrem Shield befindet sich ein Fotowiderstand LDR101, der mit dem Widerstand R102 einen Spannungsteiler bildet (siehe Schaltplan auf der letzten Seite). Die sich ergebende Spannung kann am Pin A0 mit Hilfe des ADCs im Prozessor gemessen werden.
Erstellen Sie ein Skript ( = m-File) das die Spannung dort misst und mit dem Befehl *disp* einschließlich Erläuterung im Kommandofenster ausgibt, d.h. es sollte z.B. für 1.567 V folgende Meldung erscheinen:

gemessene Spannung am Fotowiderstand: 1.567 V

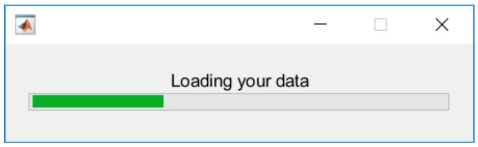
Messen Sie die Spannung mit und ohne Abschattung des Fotowiderstandes. Es muss sich ein deutlicher Unterschied ergeben.

**Ziel 2**

**In diesem Ziel soll die Spannung laufend gemessen und ausgegeben werden. Auf Knopfdruck soll das Programm beendet werden können.**

Das könnten Sie prinzipiell mit einer *while* Schleife tun. Das Problem ist aber, die *while* Schleife gezielt zu beenden. Den in C dafür vorhandenen Befehl *kbhit()* zur Abfrage, ob irgendein Zeichen auf der Tastatur eingegeben worden ist, gibt es in MATLAB leider nicht. Die interaktive Bedienung von MATLAB erfolgt standardmäßig mit einer grafischen Benutzeroberfläche, GUI = **G**raphical **U**ser **I**nterface genannt. Es ist aber nicht sinnvoll, im Rahmen einer Einführung eine ganze grafische Benutzeroberfläche zu erstellen.

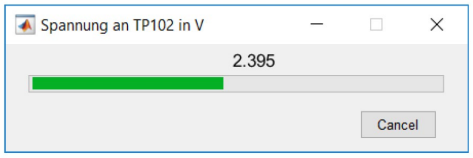
Wir verwenden ein von MATLAB bereitgestelltes grafisches Element, das für unsere Zwecke geeignet ist und *waitbar* genannt wird. Sie kennen es sicher:




MATLAB erlaubt es, dieses Element abzuändern. Man kann

1. ihm einen Namen geben (hier *Spannung an TP102 in V* ),
2. Zahlenwerte ausgeben (hier 2.395),
3. es mit einem Button ‚Cancel' versehen, den man ‚drücken' kann
4. und man kann den grünen Balken dazu verwenden, die Höhe der Spannung als Prozentsatz von 3,3 V (= Versorgungsspannung für den Spannungsteiler unabhängig von der Arduino Version) auszudrücken.

Es sieht dann so aus:




Das zugehörige Skipt zur Demonstration der Funktionsweise des waitbar – Objektes in MATLAB ist:

```
% Definition of the modified waitbar figure.
% wb_identifier is an identifier, so that following statements know,
% where to display numbers or where to get information from.
% The 3 dots ... indicate, that the statement continues in the next line.
wb_identifier = waitbar(0,'1','Name','Spannung an TP102 in V',...
'CreateCancelBtn','setappdata(gcbf,''canceling'',1)');
% example only - exchange it by your program, that measures the voltage at
% the light sensor
voltage = 0.2; % example only
while true
      if getappdata(wb_identifier,'canceling')
            break
      end
% example only - exchange it by your program, that measures the voltage at
% the light sensor
```

```
      voltage = voltage + 0.1; % example only
      pause(1) % example only
      waitbar(voltage/3.3, wb_identifier,sprintf('%12.3f',voltage))
end % while Schleife
delete(wb_identifier)
```

Modifizieren Sie das gegebene Skript so, dass es die oben geforderte Funktion erfüllt.

**Ziel 3**

**In Ziel 3 soll bestimmt werden, wie lange eine Messung der Spannung am Fotowiderstand dauert und die Messergebnisse über einen Zeitraum von etwa 10s sollen grafisch nach der Messung mit einem plot-Befehl dargestellt werden.**

Programmieren Sie eine Schleife, die gut 10 Sekunden dauert, in der Sie die Spannung am Fotowiderstand laufend messen und abspeichern und bestimmen Sie mit „tic“ und „toc“, wie lange **eine** Spannungsmessung dauert. Berechnen Sie die Abtastfrequenz und geben Sie sie mit einem disp Befehl einschließlich Text zur Erläuterung aus.
Speichern Sie die Messwerte in einem Array und geben Sie sie grafisch mit dem plot – Befehl aus. Damit sich etwas ändert, können Sie den Finger über den Lichtsensor bewegen und damit die Helligkeit verändern.
Damit Sie wissen, wann die Messung wirklich beginnt und endet, sollten Sie dies mit einer der beiden LED auf dem Shield anzeigen.
Wenn Sie nur den plot-Befehl in der einfachsten Version verwenden, werden die Messwerte über dem Index aufgetragen. Ändern Sie waagrechte Achse so ab, dass dort direkt die Zeit steht.
Geben Sie dem Diagramm einen Titel und beschriften Sie die waagrechte und senkrechte Achse sinnvoll.

**Ziel 4**

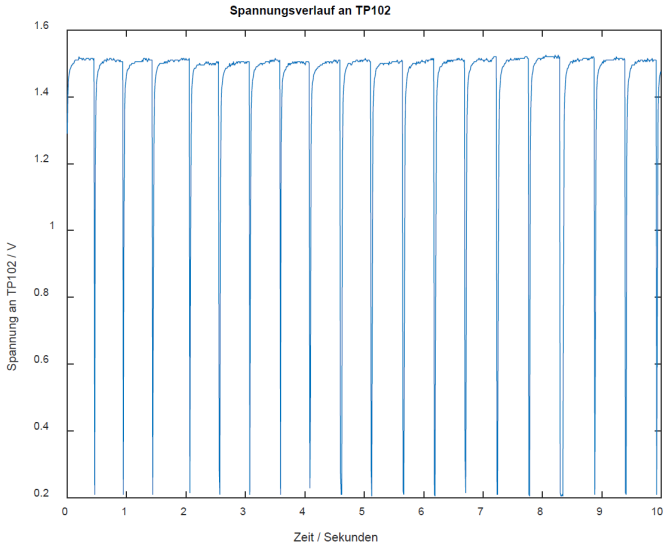
Installieren Sie auf Ihrem Handy die kostenlose App “Strobe Light“. Sie ist an folgendem Symbol erkennbar:

Mit dieser App können Sie (fast) regelmäßige Lichtimpulse mit Frequenzen zwischen 1 und 30 Hz erzeugen – gerade richtig für die Messgeschwindigkeit des Arduino Uno.

Speichern Sie das in Ziel 3 erzeugte m-File unter Arduino_Ziel_4.m ab und erweitern Sie es, um folgende Aufgabenstellungen erfüllen zu können:
**Die Anzahl der Messungen muss flexibel sein, d.h. wenn Sie eine Messzeit vorgeben, berechnet das Programm selbständig die dafür notwendige Anzahl der Messungen *NumberOfMeasurements*.**

Stellen Sie in Ihrer Handy App eine nominale Frequenz von 2 Hz ein (Hinweis: Nicht bei jedem Handy ist die dann erzeugte Frequenz genau 2 Hz, sondern kann um wenige Zehntel Hz abweichen). Weiterhin sind nicht unbedingt alle Lichtpulse gleich lang. Das bedeutet, dass Sie nicht notwendigerweise ein perfekt periodisches Signal bekommen und damit auch die DFT einen nicht unerheblichen Untergrund haben kann.

Im Zeitbereich sollten Sie in etwa folgende Abbildung bekommen (Messzeit ca. 10s):

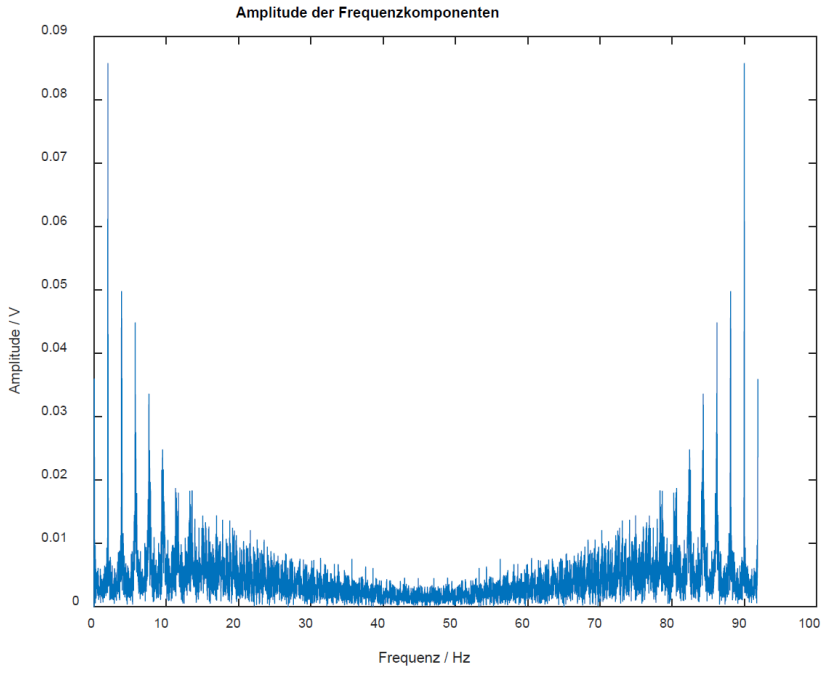



Es ist sinnvoll, vor der Berechnung der DFT den Gleichanteil zu berechnen und abzuziehen.

Unterwerfen Sie den AC-Anteil der Messwerte der Spannung an TP102 einer DFT und geben Sie das Ergebnis der DFT (d.h. im Wesentlichen den Betrag der X[k] ) grafisch mit *plot* in einer zweiten Abbildung aus, so dass:

1. in der waagrechten Achse die Frequenz angezeigt wird
2. in der senkrechten Achse die Amplitude der Spannungskomponente bei der zugehörigen Frequenz angegeben wird und
3. beschriften Sie die Grafik sinnvoll

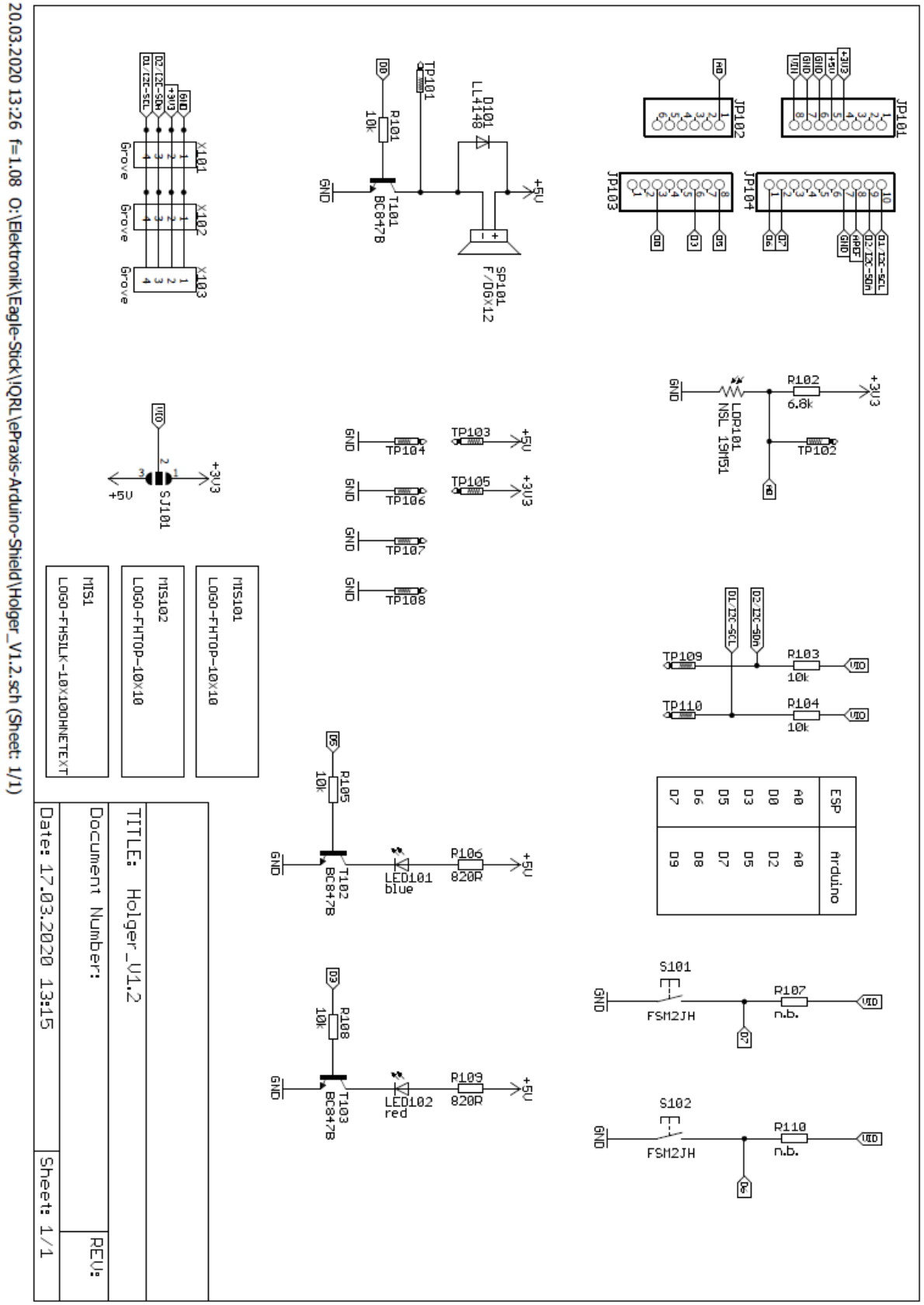
Das Ganze sieht ungefähr dann aus, wie in der folgenden Abbildung (Messzeit ca. 50 s):



Bedenken Sie, dass in MATLAB die Indices mit 1 beginnen und nicht wie in den meisten anderen Programmiersprachen mit 0. Beispiel: Unser X[10] aus der Messtechnik Vorlesung finden Sie daher in MATLAB beim Index 11.

Stellen Sie die Messzeit so ein, dass Sie eine Frequenzauflösung von etwa 0,05 Hz erreichen und messen Sie die Frequenz der Lichtpulse.

Zur Verifizierung, dass die Frequenzmessung auch bei anderen Frequenzen funktioniert, stellen Sie bitte in der App auch andere Frequenzen ein.


20.03.2020 13:26 f=1.08 O:\Elektronik\Eagle-Stick\IQRL\ePraxis-Arduino-Shield\Holger_V1.2.sch (Sheet: 1/1)
TITLE: Holger_V1.2
Document Number:
REV:
Date: 17.03.2020 13:15
Sheet: 1/1
JP101
JP102
JP103
JP104
X101
X102
X103
Grove
SJ101
SP101
F/D6X12
D101
LL4148
R101
10k
T101
BC847B
TP101
R102
6.8k
LDR101
NSL 19M51
TP102
TP103
TP104
TP105
TP106
TP107
TP108
R103
10k
R104
10k
TP109
TP110
R105
10k
T102
BC847B
LED101
blue
R106
820R
R108
10k
T103
BC847B
LED102
red
R109
820R
S101
FSM2JH
R107
n.b.
S102
FSM2JH
R110
n.b.
ESP
Arduino
MIS101
LOGO-FHTOP-10X10
MIS102
LOGO-FHTOP-10X10
MIS1
LOGO-FHSILK-10X100HNETEXT
GND
+5V
+3V3