



Arduino-Shield

microSD-Kartenschacht

Netzwerk für Arduino – NetworkShield for Arduino

Mit dem NetworkShield for Arduino lässt sich ein Arduino-Board einfach mit dem PC-Netzwerk verbinden. So können über das Netzwerk vom Arduino Steueraufgaben durchgeführt oder Rückmeldungen von Sensoren ausgelesen und am PC ausgewertet werden. Das NSA1 besitzt zusätzlich einen microSD-Kartenschacht und ist kompatibel zu den bereitstehenden Ethernet- und SD-Libraries von Arduino.

Rein ins Netz!

Die Arduino-Plattform ist ja anerkanntermaßen die universell einsetzbare Mikrocontroller-Plattform, vor allem auch beliebt wegen der vergleichsweise einfachen Programmierbarkeit über die zugehörige IDE. Da liegt natürlich auch der Einsatz als kleiner Webserver für Steuerungs- und Erfassungsaufgaben und damit die Einbindung in ein Computernetzwerk bzw. gar das Internet nahe.

Ein mögliches Szenario wäre etwa die Verbrauchsdatenerfassung an einem Strom- oder Gaszähler. Der

Arduino erfasst und sammelt hier die von Sensoren kommenden Daten (via S0-Schnittstelle, Impulsausgang oder optischem Abtaster), speichert diese als Datensatz auf einer Speicherkarte ab, und ein Datenbank- bzw. Visualisierungsprogramm auf einem PC im Netzwerk oder gar einem Smartphone fragt diese Daten via Netzwerk periodisch ab und stellt sie visualisiert zur Verfügung.

In gleicher Weise kann man zum Beispiel auch alle Temperaturen in der Heizungs- und Solaranlage zentral erfassen und zur Auswertung bereitstellen.

Aber auch umgekehrt geht es: Die Ports des Arduino können auch aus der Ferne als Ausgänge geschaltet werden und so z. B. Relaisstufen schalten, Dimmer, Motorregler oder Displays ansteuern.

Für alle dafür erforderlichen Programmschritte gibt es in der Arduino-Community fertige und an die eigenen Bedürfnisse einfach anpassbare Sketches, die man lediglich noch in das eigene Rahmenprogramm verpacken und verbinden muss.

Der reguläre PC-Anschluss des Arduino erfolgt ja über dessen USB-Port. Das stößt aber an Grenzen, wenn der Arduino abgesetzt betrieben wird. Zwar gibt es USB-zu-Ethernet-Konverter, aber diese sind nicht ganz billig und die mehrfache Protokollumsetzung erfordert auch wieder einen extra Programmieraufwand. Da liegt die Ergänzung des Arduino mit einer eigenen Ethernet-Schnittstelle und die Implementierung eines eigenen Webserver-Programms auf dem AVR des Arduino weit näher, zumal die Shield-Philosophie den Anschluss von Zusatzbaugruppen ohne jeden Verdrahtungsaufwand einfach macht.

Kurzbezeichnung:	NSA1
Versorgungsspannung:	5 Vdc
Stromaufnahme:	250 mA max.
Schnittstellen:	microSD-Kartenschacht, Ethernet (10/100 Mbit/s)
Max. Leitungslängen:	3 m
Art der Netzwerkleitung:	min. CAT.5 F/UTP
Anzeigen:	6 LEDs
Umgebungstemperatur:	5 bis 35 °C
Abmessungen (B x H x T):	74 x 56 x 26 mm
Gewicht (ohne Arduino-Board):	28 g

Das Ethernet-Shield

Unter dieser Prämisse entstand unser Ethernet-Shield, das zusätzlich gleich noch einen Kartenschacht für eine microSD-Speicherkarte trägt, um z. B. erfasste Daten einfach und auch netzausfallsicher zwischenspeichern zu können. Das Shield entspricht vollkommen den Arduino-Konventionen, weshalb auch die Ethernet Library von arduino.cc [1] direkt einsetzbar ist. Auch die vorhandenen Programmteile (SD Library, [2]) zum Auslesen und Beschreiben der SD-Karte sind hier nutzbar.

Technische Basis ist ein moderner Ethernet-Chip von WIZnet, der W5100. Er ist nicht nur deutlich stromsparender als der in vielen früheren Anwendungen zum Einsatz gekommene, inzwischen betagte ENC28J60, er ist auch vielseitiger, wie wir noch sehen werden. Zusätzlich ist eine saubere Pegelwandlung auf dem Board vorhanden, um die Signalpegel von Ethernet-Shield (betrieben mit 3,3 V) und Arduino-Board (betrieben mit 5 V) exakt anzupassen. Gerade dieses Feature findet man bei Weitem nicht bei allen auf dem Markt befindlichen Ethernet-Shields. Eine einfache Pegelwandlung, etwa durch Widerstandsnetzwerke, birgt aber immer die Gefahr zumindest von Datenverlusten, auch kann durchaus die Kommunikation zwischen den Boards gestört werden, mitunter führt dies auch zu Hardware-Schäden.

Die Spannungsversorgung erfolgt aus den 5 V des Arduino-Boards, auf dem Ethernet-Shield wird daraus die benötigte Betriebsspannung (3,3 V) durch eine eigene Stromversorgungsgruppe erzeugt.

Kernstück ist der bereits erwähnte Ethernet-Chip W5100, sein Blockschaltbild ist im Bild 1 zu sehen. Er enthält sämtliche Funktionsbaugruppen zur Behandlung von Ethernet-Protokollen, gesteuert durch einen eigenen internen Controller. Er beinhaltet einen IP-Stack, der sowohl TCP- als auch UDP-kompatibel ist, einen 16-kByte-Puffer für die Datenübertragung, und er ist in der Lage, bis zu 4 Socket-Verbindungen gleichzeitig zu realisieren. Somit ist der Arduino selbst komplett von allen Verwaltungs- und Protokollaufgaben zur Ethernet-Umsetzung entlastet.

Die Verbindung zum Arduino erfolgt via Pegelwandler über den SPI-Bus, wobei hier als Besonderheit zu beachten ist, dass sich Ethernet-Chip und microSD-Kartenslot diesen Bus teilen. Deshalb ist bei der Programmierung der Abläufe zu berücksichtigen, dass immer nur eine der beiden Komponenten auf dem Bus aktiv sein darf. Entsprechend ist jeweils ein Chip-Select-Signal auszugeben, für den Ethernet-Chip über Pin 10 (SS) des Arduino-Boards, für den Kartenslot über Pin 4. Aus diesem Grund sind die betroffenen I/Os in dieser Applikation nicht als GPIOs nutzbar. Unter [1] sind dazu weitere Hinweise zu finden.

Eine Besonderheit unseres Shields ist die Möglichkeit der Reaktion des Arduino-Controllers auf interruptbasierte Signale des Ethernet-Controllers. Diese Möglichkeit ist in der Standard-Ethernet-Library nicht enthalten, weshalb die Interrupt-Leitung nicht standardmäßig mit dem Arduino-Board verbunden ist. Erfahrene Programmierer können sie allerdings nach Schließen eines Lötjumpers (J3) auf der Platine nutzen. Wird hiervon nicht Gebrauch gemacht, bleibt J3 offen und der Pin 2 des Arduino-Boards bleibt weiter als normale I/O nutzbar.

Hat man einen kleinen Webserver auf dem Arduino installiert, ist dieser also via Ethernet-Shield wie ein normales Netzwerkgerät mit den üblichen Parametern IP, Portadresse, Subnetz- und Gateway-Adresse sowie MAC-Adresse ansprechbar.

Beispiel (abhängig von der jeweiligen Netzwerkkumgebung):

IP: 192.168.1.177

Port: 80

Subnet: 255.255.255.0

Gateway/Router-IP: 192.168.1.1

MAC-Adresse: siehe MAC-Adress-Aufkleber

Da für jedes Netzwerkgerät eine einmalige MAC-Adresse notwendig ist, wird dem Bausatz eine MAC-Adresse beigelegt. Diese befindet sich als Aufkleber auf der Platinenunterseite.

Der Webserver kann u. a. auch eigene HTML-Seiten zum Abruf zur Verfügung stellen. Wie man diese erzeugen kann, ist in [3] ausführlich erläutert.

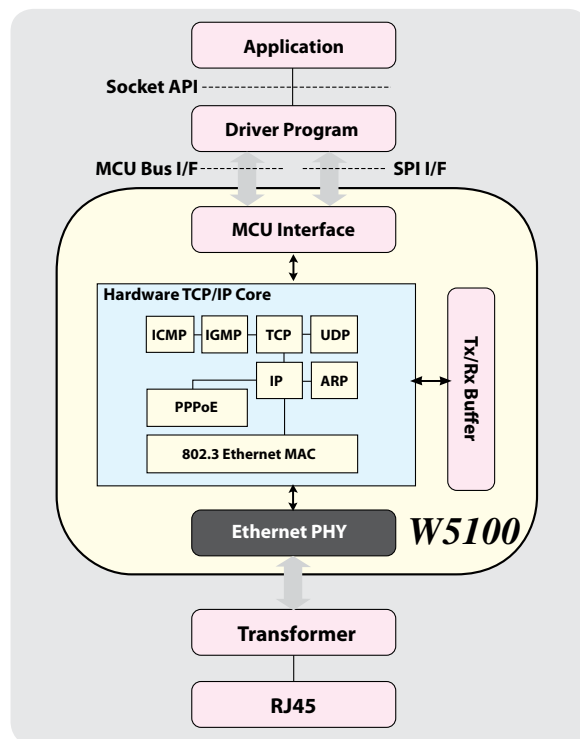


Bild 1: Das Blockschaltbild des Ethernet-Controllers W5100 von WIZnet

tert. Ein einfaches Webserver-Programm zum Auflisten des Inhalts der SD-Karte ist auf der Produktseite zum Download verfügbar, auch hierfür gibt es zahlreiche weitere Vorlagen und Hinweise, u. a. unter [1] und [3].



Wichtiger Hinweis:

Zur Gewährleistung der elektrischen Sicherheit muss es sich bei der speisenden Quelle um eine Sicherheits-Schutzkleinspannung handeln. Außerdem muss es sich um eine Quelle begrenzter Leistung gemäß EN60950-1 handeln, die nicht mehr als 15 W liefern kann. Üblicherweise werden beide Forderungen von handelsüblichen Stecker-Netzteilen mit entsprechender Leistung erfüllt.

Schaltung

Das Ethernet-Shield, dessen komplette Schaltung in Bild 2 aufgeführt ist, wird vom Arduino-Board aus mit 5 V versorgt. IC4 erzeugt daraus die Betriebsspannung von 3,3 V für den Ethernet-Controller und die Speicherkarte. Über ein Filter mit L2, C12, C13 wird von der Spannungsversorgung des Digitalteils eine entkoppelte Versorgungsspannung von 3,3 V für den Analogteil bereitgestellt. Der Ethernet-Controller erzeugt intern zusätzlich eine 1,8-V-Spannung, auch hier wird über ein Filter (L3, C17, C18) eine entkoppelte Versorgungsspannung für den Analogteil erzeugt. Alle weiteren Kondensatoren im Bereich der Spannungsversorgung und der Controller-Stromversorgungsanschlüsse haben eine Stabilisierungs- und Stützfunktion.

Die 5 V werden für die Pegelwandlerstufen von IC3 benötigt, um eine korrekte TTL-Pegelumsetzung zu gewährleisten.

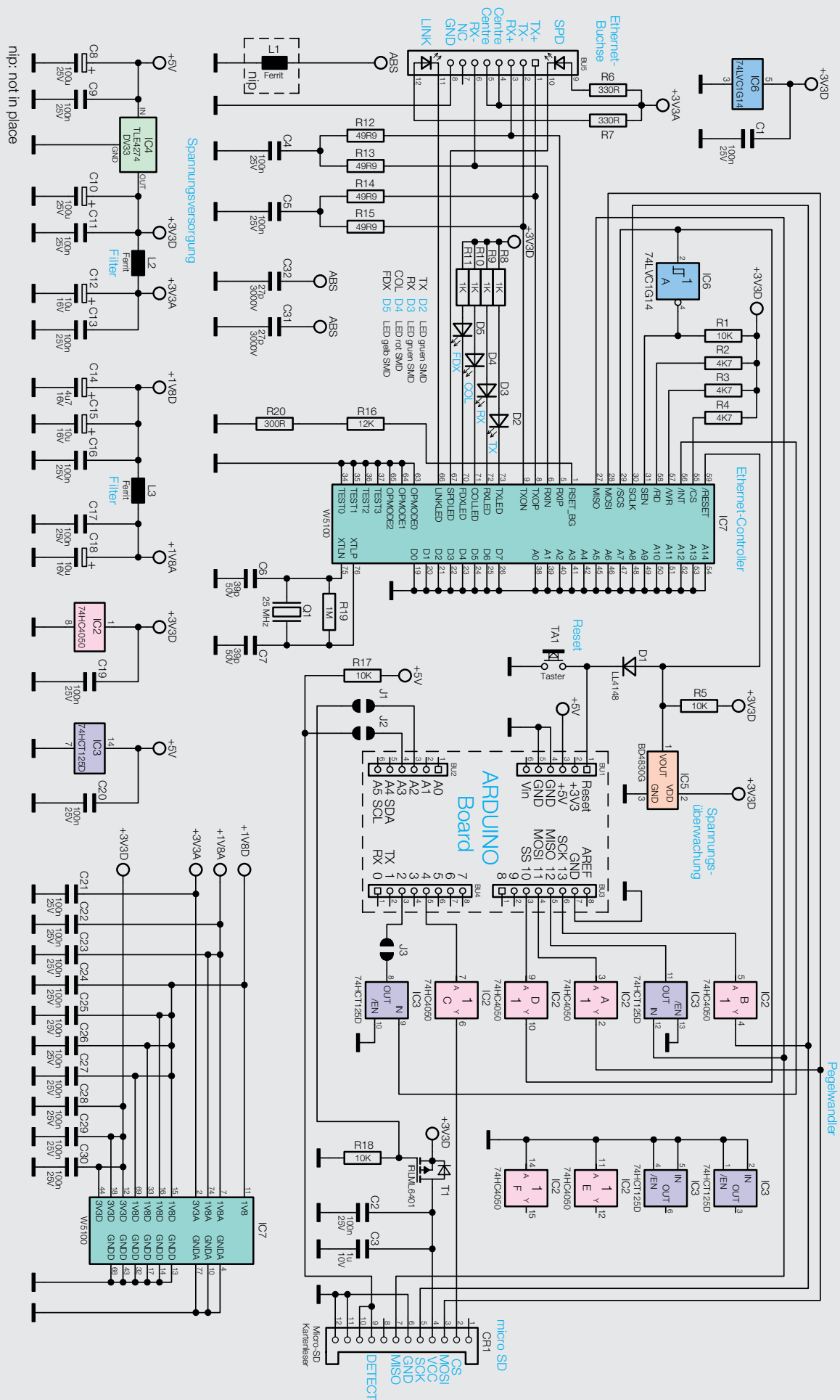


Bild 2: Schaltbild des NSA1

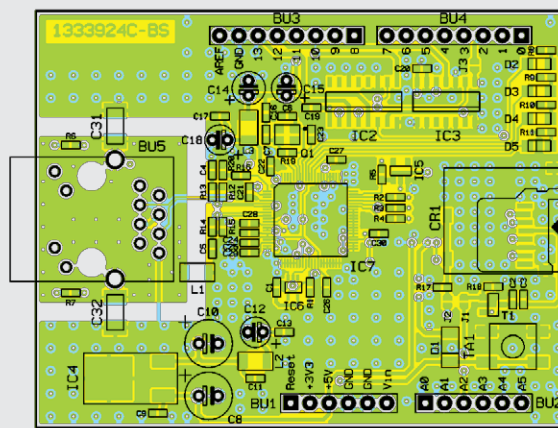
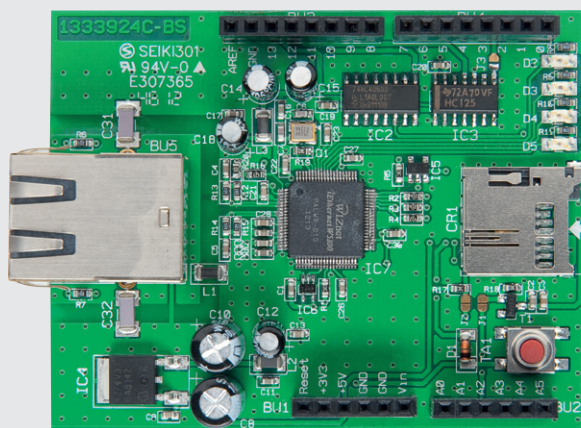


Bild 3: Die Platine des NSA1 mit dem entsprechenden Bestückungsdruck

Mit IC5 ist eine Spannungsüberwachung realisiert, die bei einem Einbruch der Betriebsspannung sowohl den Arduino als auch den Ethernet-Controller definiert zurücksetzt. Ein manueller Reset ist mit TA1 möglich.

Kernstück der Schaltung ist der Ethernet-Controller IC7. Seine Peripheriebeschaltung ist recht minimal. Der 25-MHz-Quarz Q1 dient mit C6, C7 und dem Lastwiderstand R19 der Erzeugung eines stabilen Takts. R16 und R20 legen den PHY-Status des Chips fest.

Die vier LEDs D2 bis D5 samt den zugehörigen Vorwiderständen R8 bis R11 dienen der Statusanzeige:

- D2: TX = blinkt, wenn das Shield Daten sendet
- D3: RX = blinkt, wenn das Shield Daten empfängt
- D4: COLL = blinkt bei einer Datenkollision im Netzwerk
- D5: FDX = zeigt Full-Duplex-Betrieb an

Die RC-Glieder R12 bis R15 mit C4 und C5 eliminieren Störeinflüsse auf den RX-/TX-Datenleitungen. Die Ethernet-Buchse BU5 enthält sowohl die erforderlichen Leitungsübertrager als auch die Anzeige-LEDs für die Netzwerk-Geschwindigkeit (SPD) sowie die Netzwerkverbindung (LINK).

Das Gatter IC6 A erzeugt aus dem vom Arduino-Board kommenden Chip-Select-Signal für die Aktivierung des Ethernet-Controllers das SPI-Enable-Signal, das den Zugriff des Controllers auf den SPI-Bus freigibt.

R1 bis R4 sind Pull-up-Widerstände, die entsprechend der Spezifikation des Controllers beschaltet sind. IC2 und IC3 dienen der Pegelwandlung bzw. Entkopplung der Signale von Arduino-Board und Ethernet-Shield. Mehr zur Funktion dieser Pegelwandlung findet sich unter „Elektronikwissen“.

CR1 ist der microSD-Slot, er erhält über IC2 C sein Chip-Select-Signal vom Arduino und ist über die Pegelwandler an den SPI-Bus angebunden. Um die Optionen für SD-Power und SD-Detect bei Bedarf nutzbar zu machen, sind die entsprechenden Pins des microSD-Slots an Lötjumper (J1 = SD-Power, J2 = SD-Detect) geführt. Diese können bei Bedarf geschlossen und die Steuersignale via A2 bzw. A3 des Arduino-Boards ausgegeben bzw. empfangen werden. T1 dient dabei als Schalter für Vcc. Die beiden Funktionen SD-Power und SD-Detect sind allerdings nicht in

der Standard-SD-Library berücksichtigt! Zum Abschalten der Spannungsversorgung für die microSD-Karte ist der Pin A2 als Ausgang auf High-Pegel zu setzen.

Für die Kartenerkennung ist Pin A3 als Eingang zu schalten (externer Pull-up vorhanden), er wird auf Low-Pegel gehalten, solange keine SD-Karte eingesetzt ist.

Der Lötjumper J3 schließlich ermöglicht die bereits beschriebene Interrupt-Verbindung zum Arduino.

Nachbau

Die Platine des Ethernet-Shields ist bereits ab Werk mit allen SMD-Bauteilen bestückt, so dass für den Aufbau des Shields nur noch wenige bedrahtete Bauteile zu bestücken sind. Allerdings sollte eine Bestückungskontrolle anhand des Platinenfotos und des Bestückungsplans erfolgen. In Bild 3 ist dazu die bestückte Platine sowie der dazugehörige Bestückungsplan zu sehen. Zusätzlich dient die Stückliste als Hilfe bei der Bestückung.



Wichtiger Hinweis:

Bei der Verwendung des NSA1 mit dem Arduino-Board ist darauf zu achten, dass die angeschlossene Netzwerkleitung keine mechanische Belastung auf die Netzwerkbuchse ausübt, da sonst das NSA1 aus den Buchsenleisten gehoben werden könnte und so ein Kurzschluss zwischen Netzwerkbuchse des NSA1 und der USB-Buchse des Arduino entstehen könnte.

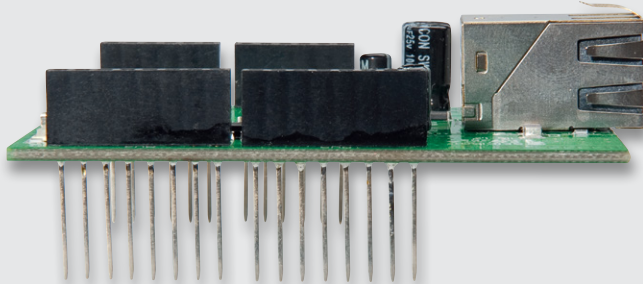


Bild 4: Die in die Platine eingesetzten und von unten verlöteten BU1 bis BU4



Bild 5: So ist die Ethernet-Buchse einzusetzen und zu verlöten.

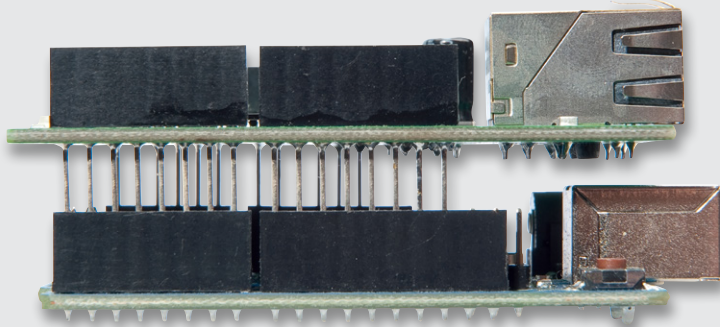


Bild 6: So wird die Ethernet-Shield-Platine in die Arduino-Platine eingesetzt.

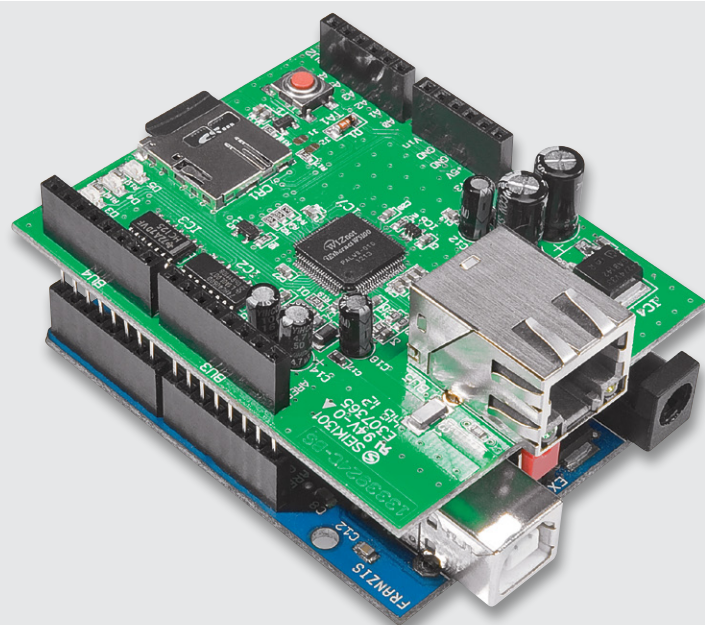


Bild 7: Betriebsbereit – die fertige Kombination aus Arduino und Ethernet-Shield NSA1

Diese beginnt mit dem polrichtigen Einsetzen der Elektrolyt-Kondensatoren C8, C10, C12, C14, C15 und C18. Hier ist darauf zu achten, dass im Bestückungsdruck der Pluspol markiert ist, aber am Kondensator selbst ist der Minuspol durch eine Strichmarkierung gekennzeichnet. Das Verlöten erfolgt auf der Platinenunterseite, überstehende Bauteilanschlussdrähte sind mit einem Seitenschneider abzuschneiden.

Danach erfolgt das Einsetzen und Verlöten der Stiftleisten-Buchsen BU1 bis BU4. Diese sind von oben in die Platine einzusetzen, so dass der Buchsenkörper plan auf der Platine aufliegt, und auf der Platinenunterseite zu verlöten. Bild 4 zeigt das Ergebnis. Dabei ist auch darauf zu achten, dass zum einen die Buchse senkrecht auf der Platine steht und zum anderen kein Lötzinn an die Steckkontakte gelangt, die in das Arduino-Board gesteckt werden.

Zum Schluss ist die Ethernet-Buchse BU5 von oben in die Platine einzusetzen, wobei darauf zu achten ist, dass alle Anschlusspins vor dem Bestücken gerade ausgerichtet sind. Nach dem Einsetzen verlötet man zunächst die Gehäuseanschlüsse bei plan aufsitzender Buchse mit reichlich Lötzinn, danach folgen dann die weiteren Pins. Bild 5 zeigt die exakt eingesetzte Buchse.

Damit ist der Aufbau bereits abgeschlossen und die Baugruppe kann nach Einsetzen in die Arduino-Platine (Bild 6) in Betrieb genommen werden. Bild 7 zeigt die betriebsbereite Kombination. **ELV**



Weitere Infos:

- [1] www.arduino.cc/en/Reference/Ethernet
- [2] www.arduino.cc/en/Reference/SD
- [3] www.arduino.cc/en/Tutorial/WebServer

Widerstände:

49,9 Ω /1 %/SMD/0603	R12–R15
300 Ω /1 %/SMD/0603	R20
330 Ω /SMD/0603	R6, R7
1 k Ω /SMD/0603	R8–R11
4,7 k Ω /SMD/0603	R2–R4
10 k Ω /SMD/0603	R1, R5, R17, R18
12 k Ω /1 %/SMD/0603	R16
1 M Ω /SMD/0603	R19

Kondensatoren:

27 pF/3 kV/SMD	C31, C32
39 pF/SMD/0603	C6, C7
100 nF/SMD/0603	C1, C2, C4, C5, C9, C11, C13, C16, C17, C19–C30
1 μ F/SMD/0603	C3
4,7 μ F/16 V	C14
10 μ F/16 V	C12, C15, C18
100 μ F/25 V/105 $^{\circ}$ C	C8, C10

Halbleiter:

74HC4050D/SMD	IC2
---------------	-----

74HCT125/SMD	IC3
TLE4274DV33/SMD	IC4
BD4830G/SMD	IC5
74LVC1G14/SMD	IC6
W5100/SMD	IC7
IRLML6401/SMD	T1
LL4148	D1
SMD-LED/gelb-grün	D2, D3
SMD-LED/rot	D4
SMD-LED/gelb	D5

Sonstiges:

Chip-Ferrit, 1206, 80 Ω bei 100 MHz	L1–L3
Quarz, 25 MHz, SMD	Q1
microSD-Kartenhalter TFLASH Push/Push	CR1
Buchsenleiste, 1x 6-polig, gerade, print	BU1, BU2
Buchsenleiste, 1x 8-polig, gerade, print	BU3, BU4
Modulare Einbaubuchse, 8-polig, abgesch.	BU5
Mini-Drucktaster, 1x ein, Tastknopflänge 1,1 mm	TA1
1 Ferrit-Ringkern, 25 x 12 mm	
1 Aufkleber mit MAC-Adresse, Matrix-Code	



Achtung:

Störaussendung

Beim Einsatz des Shields mit Arduino ist im Hinblick auf Störaussendung noch etwas zu beachten. Da die kommerziell erhältlichen Arduino-Boards beim Anschluss von externen Komponenten dazu neigen, Störsignale über die Zuleitungen auszusenden, muss ein Ferritring in die Zuleitung der Versorgungsspannung eingebracht werden. Hierzu wird die Zuleitung viermal durch den Ferritring geführt.



74er-Logikbausteine

Für die Pegelwandlung wurden Logik-Bausteine der 74er-Reihe verwendet, diese ist die am weitesten verbreitete Digital-IC-Familie. Dabei lässt sich diese Reihe noch unterteilen. So existieren z. B. auf Transistor-Transistor-Logik (TTL) oder auf Complementary-Metal-Oxide-Semiconductor-Technologie (CMOS) basierende Familien.

Der Vorteil von TTL ist immer noch die niedrige Verzögerungszeit, während der Vorteil von CMOS der geringere Stromverbrauch ist. Bei Mischung beider gibt es jedoch teilweise Probleme mit den unterschiedlichen Spannungspegeln (siehe Grafik), so dass bei CMOS extra Familien existieren, welche z. B. TTL-kompatible Eingänge bieten.

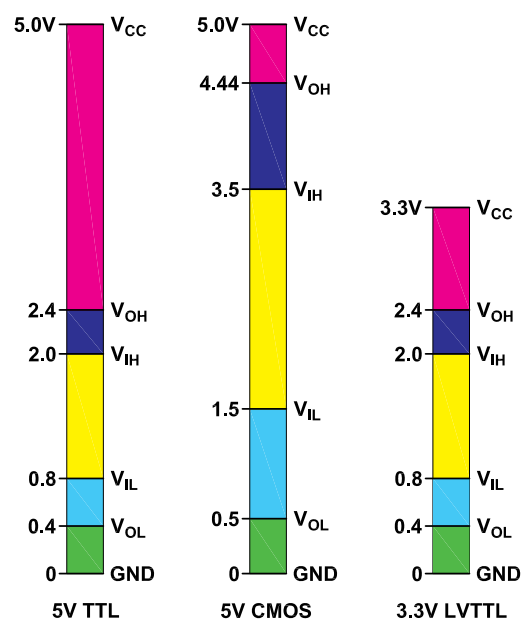
So folgt nach dem Präfix der Logikreihe die Kennzeichnung der einzelnen Familien mittels Buchstaben, die anschließend folgenden Zahlen geben die Logikfunktion der Schaltung wieder (siehe Auflistung und Tabelle).

Eine vollständige Auflistung der Logikfunktionen wäre zu umfangreich, dafür gibt es – unsere älteren Leser werden sie sicherlich noch kennen – TTL-Bücher, aber im digitalen Zeitalter findet man genügend Listen, z. B. unter:

www.mikrocontroller.net/articles/74xx

Deshalb erfolgt hier nur eine kleine Auswahl mit wichtigen Funktionen:

00	2 Input NAND
02	2 Input NOR
04	Inverter
08	2 Input AND
46	BCD -> 7-Segment Decoder (OC)
74	D Flipflop
78	JK Flipflop
86	2 Input XOR
125	Tri-State Buffer
165	8-Bit Parallel -> Serial (PISO) Shift Register
595	8-Bit Serial -> Parallel (SIPO) Shift Register
4050	Levelshifter



Vergleich der Standardpegel von TTL und CMOS

Neben der 74er-Reihe gibt es noch Reihen mit anderen Temperaturbereichen/Anwendungsgebieten:

Präfix	Temperaturbereich	Anwendungsgebiet
74...	0 bis 70 °C	kommerziell
84...	-25 bis +85 °C	industriell
54...	-55 bis +125 °C	militärisch

Jedoch wurde von den Herstellern der Temperaturbereich der 74er-Reihe in den neueren Generationen erweitert, so dass selbst Bauteile der 54er-Reihe nur noch selten zu finden sind.

Technologie	TTL			CMOS			
	7400	74LS00	74AS	74HC00	74HCT	74AHC/T	74LVC
Bezeichnung	Standard TTL	Low-Power-Schottky-TTL	Advanced-Schottky-TTL	High-Speed-CMOS	High-Speed-CMOS TTL	Advanced High-Speed-CMOS	Low-Voltage
Beschreibung							
Besondere Merkmale		verringert Strombedarf	sehr schnell		TTL-kompatible Eingänge	schneller als HC/T	geringere Versorgungsspannung
Versorgungsspannung	4,75–5,25 V	4,75–5,25 V	4,5–5,5 V	2–6 V	4,5–5,5 V	4,5–5,5 V (HCT) 2–5,5 V	1,65–3,6 V
V _{OH} min.	2,4	2,7	2,5	4,5	4,5	vgl. HC/HCT	2,4
V _{OL} max.	0,4	0,5	0,5	0,25	0,25	vgl. HC/HCT	0,4
V _{IH} min.	2	2	2	3,5	2	vgl. HC/HCT	2
V _{IL} max.	0,8	0,8	0,8	1,5	0,8	vgl. HC/HCT	0,8