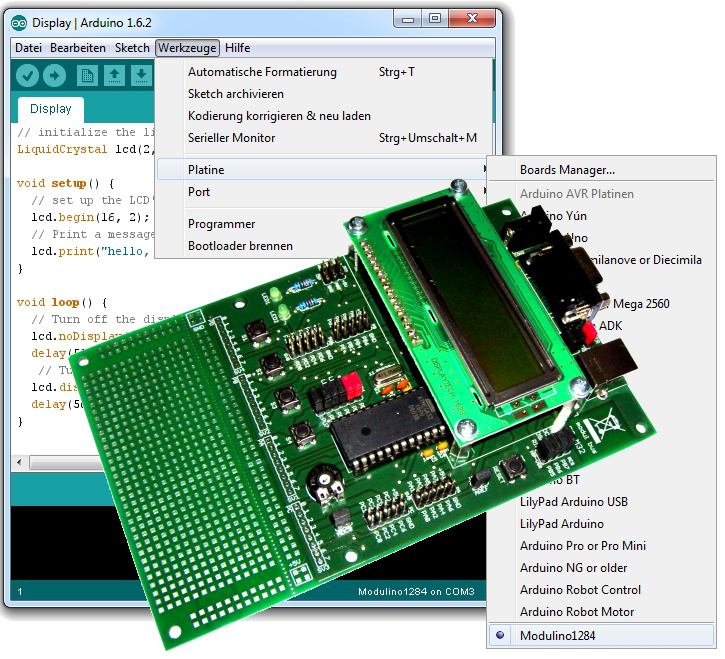
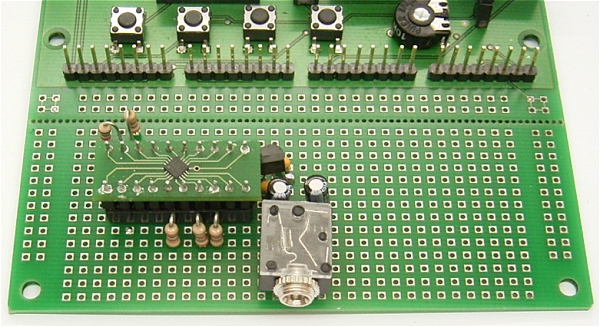
**Der Modulino1284**



|  |
| --- |
| Modul-Bus stellt ein neues Entwicklungssystem vor. Der Modul-Lino1284 ist ein Arduino-kompatibles Entwicklungssystem auf der Basis des ATmega1284. Manch einer wird vielleicht sagen: Noch ein Arduino, gibt es da nicht schon genügend unterschiedliche Systeme? Kann schon sein, aber wir haben uns entschlossen, die Vorteile des bewährten [**ES-M32**](http://elexs.de/ESM32.htm) nun auch denjenigen zugänglich zu machen, die gern mit der Arduino-Entwicklungsumgebung arbeiten.   Als das ES-M32 entwickelt wurde stand die universelle Verwendbarkeit im Mittelpunkt. Was braucht man in den meisten Fällen, was sollte gleich mit an Bord sein. Das Konzept hat sich bei zahlreichen Entwicklungen bewährt. So wurde z.B. das Elektor-DSP-Radio anfangs auf dem ES-M32 entwickelt. Und zahlreiche kundenspezifische Entwicklungen wurden bei Modul-Bus auf dieser Basis begonnen, um dann später eine kompaktere Platine für die Serie zu entwickeln. |



[Radio-Enwicklung](http://elexs.de/SI4735g.html)

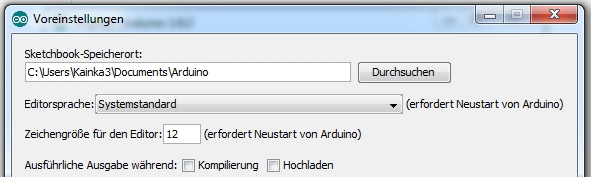
|  |
| --- |
| Der heutige Quasi-Standard in der Arduino-Welt ist der Arduino Uno, ebenfalls mit 32 KB Flash. Mit diesem System kann man gut arbeiten. Aber man stößt relativ schnell an eine Grenze, bei der man zusätzliche Hardware in Form eines Shields einsetzen muss. Und oft werden die Portanschlüsse knapp. Setzt man z.B. ein LCD ein, bleiben für manche Aufgaben nicht mehr genügen Pins frei.   Mit dem Modulino ist das anders. Das Wichtigste ist schon vorhanden, sodass man in den meisten Fällen ohne zusätzliche Verdrahtung gleich loslegen kann:  -          ATmega1284, 16 MHz, 32 IO-Ports, gesockelt  -          On-Board ISP-Programmer  -          Wahlweise RS232 oder USB verwendbar  -          Stromversorgung über USB oder extern  -          Portanschlüsse STK500-kompatibel mit GND und VCC  -          LC-Display on Board  -          Poti, vier Taster, zwei LEDs  -          Lochrasterfeld für Hardware-Erweiterungen     Und wenn ein Projekt fertig ist kann man den programmierten Mega1284 aus dem Sockel nehmen und auf einer eigenen Platine einsetzen. Ein neuer Mega1284 wird dann gleich auf der Platine ohne ein zusätzliches Programmiergerät mit dem Arduino-Bootloader (basierend auf dem Optioboot-Bootloader) programmiert, und schon kann es weiter gehen.  (!!!)  Wer das ES-M32 kennt wird nur einen Unterschied entdecken: Die Taktfrequenz wurde im Interesse einer besseren Kompatibilität auf 16 MHz hochgesetzt. Alles andere betrifft die Software. Mit der erweiterten Programmiersoftware ESM32.exe kann nun der Arduino-Bootloader in den Mega1284 gebrannt werden. Und damit ist das System fit für die Verwendung mit der Arduino-Entwicklungsumgebung.   Die Arduino-IDE unterstützt eigentlich keinen Mega1284. Man kann aber ein eigenes Board definieren und einfügen. Die Umsetzung der Implementierung basiert auf einem Projekt mit dem Namen Sanguino. Dabei haben folgende Quellen geholfen:  <https://github.com/Lauszus/Sanguino>  <http://blog.stevemarple.co.uk/2013/01/how-to-use-atmel-atmega1284-non-p.html>  <https://github.com/arduino/Arduino/wiki/Arduino-Hardware-Cores-migration-guide-from-1.0-to-1.6> |

**Installation des Modulino1284 Boards**

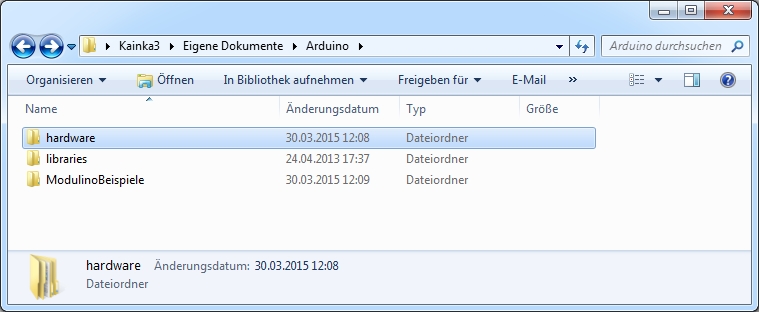
Diese Anleitung und diese Dateien beziehen sich auf eine Arduino Version ab 1.6.

**Download: hardware.zip (!!!)**

Zum Ausführen der Installation sind einige wenige Schritte nötig.

* Laden Sie den oben verlinkten Zip-Ordner herunter
* Schließen Sie die Arduino-Oberfläche
* Kopieren Sie den Ordner „hardware“ inklusive aller darin enthaltenden Ordner und Dateien in Ihren Sketchordner  
  + Wenn Sie sich nicht sicher sind, wo sich Ihr Sketchorder befindet sollten Sie in den Einstellungen nachsehen (unter Datei/Einstellungen oder mit der Tastenkombination [Strg] + [,]) 
  + Sollte sich in Ihrem Sketchordner bereits ein Ordner mit dem Namen „hardware“ befinden, integrieren Sie die Dateien des geladenen hardware-Ordners in den bestehenden
* Starten Sie die Arduino Oberfläche
* Wählen Sie unter Werkzeuge-> Platine den Punkt Modulino1284 aus
* Testen Sie die Funktionalität mit dem Blink-Beispiel unter Datei->Beispiele->01. Basics->Blink

**((Bootloader installieren (!!!) nicht nötig, schon drin. Optional, später ebschreiben))**  
Bei Bedarf können Sie mit dem Programm ESM32.exe den Booatloader in den Controller brennen. In **\hardware\Modulino\avr\bootloaders\optiboot\** steht die Hexdatei des Bootloaders **optiboot\_atmega1284p.hex**, die Sie bei Bedarf mit dem Programm [**ESM32.exe**](http://elexs.de/ESM320o.html) in den Controller brennen können. In den meisten Fällen wird das allerdings nicht nötig sein, weil der Bootloader bereits bei der Auslieferung des ES-Modulino geladen ist.



**Arduino-Pins**

Eine Besonderheit der Arduino-Software ist, dass alle Pins einfach nur Nummern tragen. Der Mega1284 hat deutlich mehr IO-Ports als der Mega328 im Arduino Uno. Daher ergibt sich eine andere Nummerierung der Anschlüsse, die man beim Programmieren berücksichtigen muss:

// ATMEL ATMEGA644/ATMEGA1284 / SANGUINO

//

// +---\/---+

// (D 0) PB0 1| |40 PA0 (AI 0 / D31)

// (D 1) PB1 2| |39 PA1 (AI 1 / D30)

// INT2 (D 2) PB2 3| |38 PA2 (AI 2 / D29)

// PWM (D 3) PB3 4| |37 PA3 (AI 3 / D28)

// SS PWM (D 4) PB4 5| |36 PA4 (AI 4 / D27)

// MOSI (D 5) PB5 6| |35 PA5 (AI 5 / D26)

// MISO (D 6) PB6 7| |34 PA6 (AI 6 / D25)

// SCK (D 7) PB7 8| |33 PA7 (AI 7 / D24)

// RST 9| |32 AREF

// VCC 10| |31 GND

// GND 11| |30 AVCC

// XTAL2 12| |29 PC7 (D 23)

// XTAL1 13| |28 PC6 (D 22)

// RX0 (D 8) PD0 14| |27 PC5 (D 21) TDI

// TX0 (D 9) PD1 15| |26 PC4 (D 20) TDO

// INT0 RX1 (D 10) PD2 16| |25 PC3 (D 19) TMS

// INT1 TX1 (D 11) PD3 17| |24 PC2 (D 18) TCK

// PWM (D 12) PD4 18| |23 PC1 (D 17) SDA

// PWM (D 13) PD5 19| |22 PC0 (D 16) SCL

// PWM (D 14) PD6 20| |21 PD7 (D 15) PWM

// +--------+

//

(Auszug aus der pins\_arduino.h-Datei)

Dementsprechend liegt

**LED1** an **D12** (PWM fähig)

**LED2** an **D13** (PWM fähig)

**S1** an **D10** (Hardware Interrupt0)

**S2** an **D11** (Hardware Interrupt1)

**S3** an **D14**

**S4** an **D15**

**Display**

**LCD RS** an **D2**

**LCD Enable** an **D3**

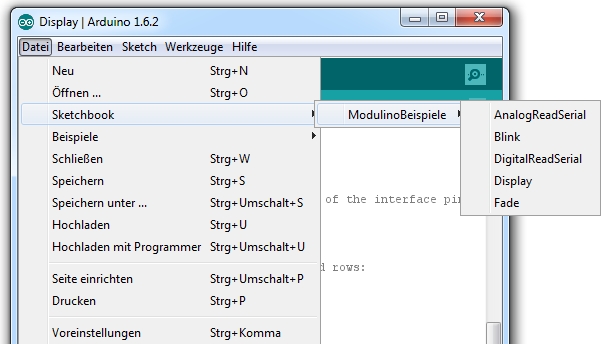
**LCD D4 pin** an **D4**

**LCD D5 pin** an **D5**

**LCD D6 pin** an **D6**

**LCD D7 pin** an **D7**

|  |
| --- |
| **Testbeispiele**  Die Arduino-Software wird mit zahlreichen Beispielprogrammen (Sketches) geliefert, die man teilweise leicht anpassen muss. Der Beispiel-Sketch Blink allerdings funktioniert auf Anhieb, da der Pin D13 an LED2 angeschlossen ist. Eine Auswahl weiterer Beispiele (inklusive ein Beispiel zur Ansteuerung des Displays) finden Sie in dem unten verlinkten Zip-Ordner, welchen Sie einfach in Ihrem Sketchorder kopieren können. |



Download: Modulino1284.zip

|  |
| --- |
| Damit das Programm in das System geladen werden kann muss das richtige Board (Modulino1284) und die richtige Schnittstelle (z.B. COM2) eingestellt sein: Nach einem Klick auf den Upload-Button erscheint zunächst die Meldung „Sketch kompilieren“ und kurz darauf „Uploaden….“. |