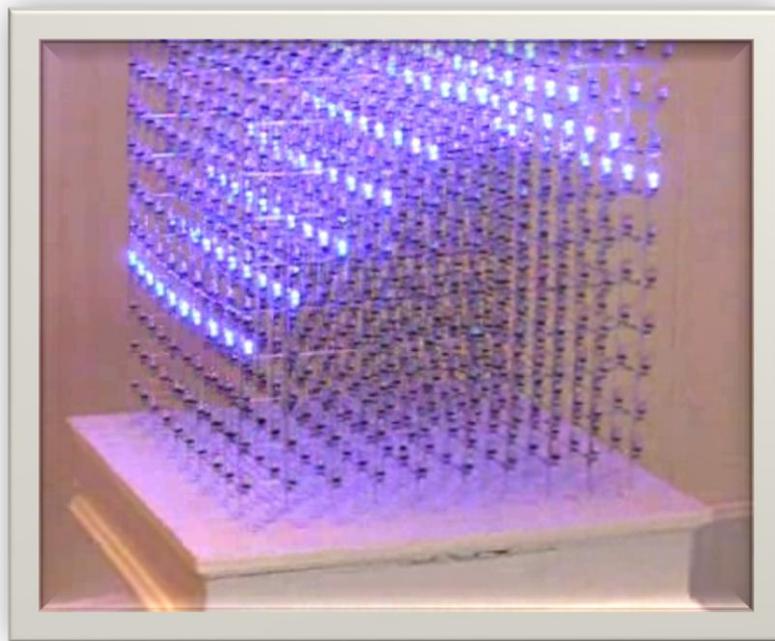


10x10x10 LED-Cube



Nachbauanleitung Projektdokumentation

V1.0 Stand 22.03.2012

by sungod

Ich übernehme keine Haftung für Schänden an Mensch, Tier, oder Gegenständen!

Wer Rechtschreibfehler findet, darf sie behalten ;-)

Einleitung

Beim durchstöbern von YouTube bin ich damals am 25.09.2011 auf Videos gestoßen von LED-Cube's. Diese haben mich wirklich sehr beeindruckt. Bei einigen Cube's bekam ich sogar eine Gänsehaut.

Zwei Beispiele:

<http://www.youtube.com/watch?v=1rMgoNGIly0>

<http://www.youtube.com/watch?v=6mXM-oGggrM>

Die nächsten Tage und Wochen verbrachte ich damit die Funktionsweise und Nachbauanregungen zu suchen.

Beispiele:

<http://www.instructables.com/id/Led-Cube-8x8x8>

<http://www.led-styles.de>

www.mikrocontroller.net

Es gab jede Menge davon im Netz. Ich würde sagen so ca. 90% aller Seiten bezogen sich auf 3x3x3 Cube's. Nun wollte ich aber nicht, wie fast überall beschrieben und geraten, erst einen 3er bauen...dann einen 5er...und dann ersten einen 10er. Ich wollte einfach nicht 3x Material, Geld und Zeit investieren.

So ein Cube ist ja im Prinzip immer gleich aufgebaut. Alle LED's einer Ebene sind miteinander verbunden und alle Säulen, also alle LED's übereinander, auch. Es wird ein bestimmtes Bitmuster an alle Säulen angelegt und dann jeweils eine Ebene nach der anderen durchgeschaltet.

Ich entschied mich nicht für einen RGB-Cube weil ich diesen nicht besonders toll fand. Ebenso finde ich bei den kleinen Cube's die Animationen kommen nicht so toll zur Geltung wie bei einen „Großen“. Dann war da noch die Frage nach der Ansteuerung. Da ich mich schon länger mit μ Controllern beschäftige war es klar einen ATmega zu benutzen. Die Software zum Programmieren des μ C benutzte ich BASCOM. Aus dem einfachen Grund, da ich schon mein Leben lang mit Basic programmiere und ich diese Sprache recht einfach finde und sie auch leicht nachzuvollziehen ist. Ich habe angefangen mit QBasic...PowerBasic...VisualBasic...u.s.w.

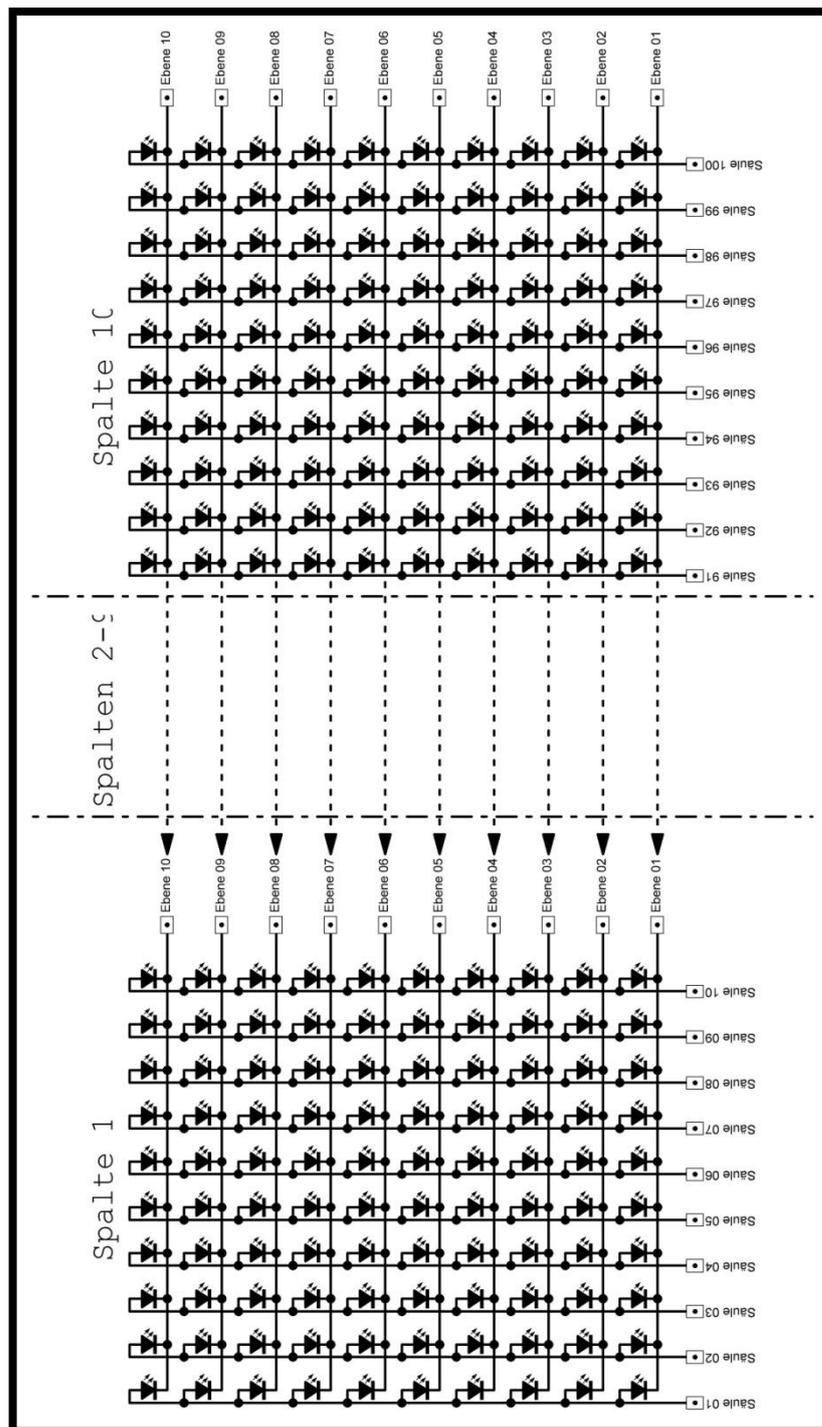
Aber das kann ja jeder für sich selber entscheiden. LED-Farbe? Stand gleich mal blau fest...sieht einfach cool aus. Im nach hinein hätte ich weiß genommen. Wer so einen Cube baut bzw. nachbauen möchte sollte wissen das er sehr viel Zeit in Anspruch nimmt. Finanziell gesehen nicht so stark...je nach dem wo man kauft und was man noch kaufen muss weil man es nicht hat. Wichtig ist auch noch sich zu überlegen was man für einen Unterbau benutzen will. Da der Würfel ja dann schon toll aussieht sollte man nicht als Unterbau einen Schuhkarton nehmen. Ich entschied mich für eine „Holzkiste“ die mit Holzleisten verschönert wurde.



Also fasse ich mal zusammen:

µC	ATMega@16MHz
LED	Blau 1000 Stück
Programmiersprache	BASCOM
Unterbau	Holzbox
Zeitaufwand	1-2 Stunden nach der Arbeit
Projektstart	25.09.2011

Hier nochmal die LED-Matrix:



Abgleich der Funktionsweise mit den Bauteilen

Ich habe in meiner Werkstatt „Unmengen“ an diversen Bauteilen. Ganz entscheidend war der IC SN74HC573 den ich in Massen hatte. Dieser Schaltkreis ist ein einfacher 1Byte (8 Bit) Speicher. Kurz gesagt, man kann ganz einfach ein Bitmuster durch Anlegen von H oder L (also High +5V oder Low 0V) an die 8 Eingänge beschreiben und diese Zustände werden dann an den Ausgängen gehalten.

Was brauche ich noch?

LED's! Viele LED's! Der Kauf in Elektronikversandhäusern fällt schon mal aus weil die einfach zu teuer in diesen Stückzahlen sind. Also ab zu eBay. Ich hatte Glück...ab und zu werden LED's in großen Stückzahlen angeboten. Ich bestellte also 2x 500 Stück + 1x 50 Stück sind 1050 LED's. 50 Stück als Reserve. Diese sind so billig da sie aus Thailand kommen. Kostenpunkt für 1050 Stück 5mm wasserklare blaue LED's 61,94 EUR.

Preisvergleich:

Conrad: 1050 Stück x 0,99 EUR = 1039,50 EUR

Reichelt: 1050 Stück x 0,28 EUR = 294 EUR

eBay: 1050 Stück = 61,94 EUR

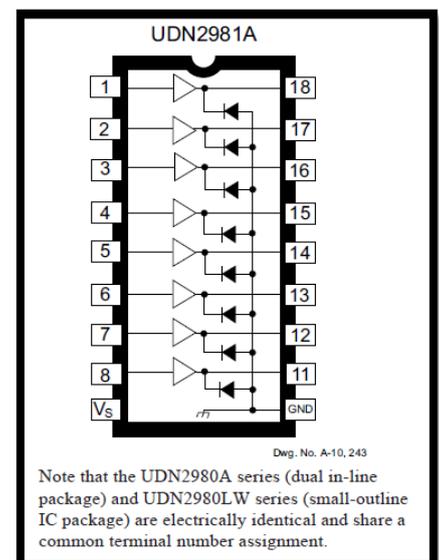
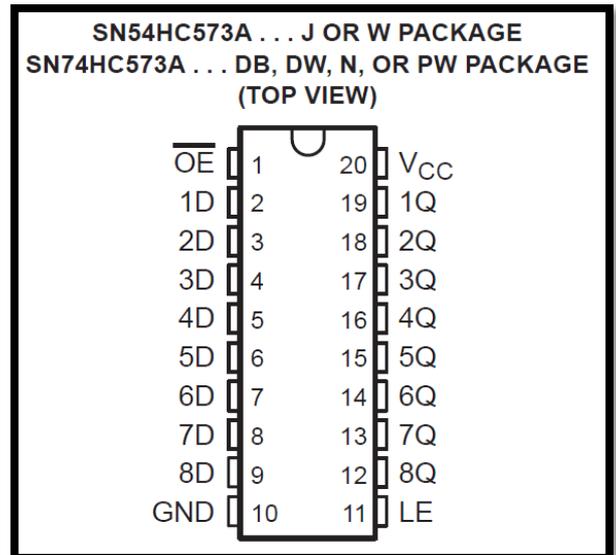
(Preise Stand: 9/2011)

Soweit so gut...: Nun haben wir also den μC , die Latches (8 Bit Speicher) und die 1050 LED's.

Nun wird ja so ein Würfel im Multiplexverfahren betrieben. Bedeutet daß, wenn man einen Würfel mit 10 Ebenen hat die alle nacheinander ganz schnell durchgeschaltet werden. Da das in der gleichen Zeit geschehen soll wie wenn nur eine LED alleine leuchtet geschieht sowas ganz schnell nacheinander. Also jede Ebene leuchtet nur 1/10 der Zeit, da wir ja 10 Ebenen haben. Wenn nun eine Ebene nur 1/10 der Zeit leuchtet dann muss sie ja auch 10x mehr Strom bekommen als sonst. Sonst leuchtet sie ja 10x dunkler.

Der derzeitige IST-Zustand: Der μC gibt ein Bitmuster zum Latch und dieser an die LED-Säulen?

Eine Normale LED hat eine Stromaufnahme von 0,025mA. Davon der 10x höhere Strom ergibt 250mA. Der Latch hat 8 Ausgänge...sollten alle Ausgänge auf HI sein müsste er 8 x 250mA = 2A!!! abgeben können...kann er aber nicht. Also muss eine LED-Treiberstufe her. Da sich bei meinen

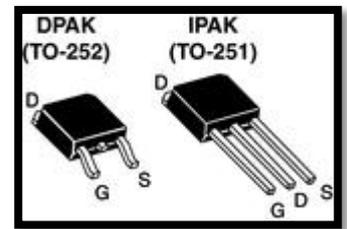


Würfel alle Anoden (+ der LED) zu den Säulen verbinden und alle Kathoden (- der LED) zu den Ebenen verbinden, entschied ich mich für den UDN2981A der den Latches nachgeschaltet wird. Danach noch die Vorwiderstände der LED's...und fertig wäre der Säulen-Teil.

Als Ebenentreiber wollte ich eigentlich einen (also 10, für jede Ebene einen) Transistor nehmen. Jedoch wurde ich in einem Forum darauf aufmerksam gemacht, dass der maximal fließende Strom in einer Ebene auf 25A!!!! steigt!!! Warum? Na ganz einfach...

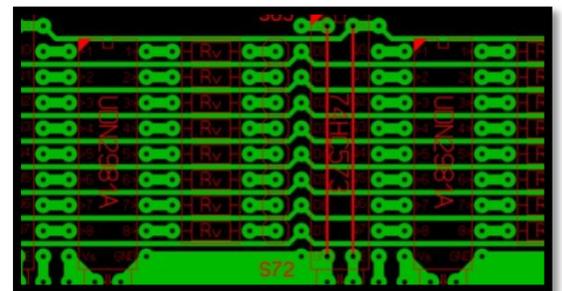
Eine LED nimmt normal 0,025A auf, x 10 (10x höherer Strom) x 100 LED's pro Ebene ergibt 25A.

Also fällt Transistor raus. Nun gut, ich wollte die LED's ja eh nicht soooo hell betreiben sondern nur mit 0,003A. Das ergäbe dann 3A. Ein netter User eines Forums gab den Tipp eine Leistungs-Mosfet zu nehmen. Dieser schaltet 10A und arbeitet mit LogicLevel. Bedeutet mit +5V oder 0V. genau das was aus den µC kommt. Perfekt. Nach einiger Googlezeit entschied ich mich für den IRLU024N.



IRLU024A

Soweit die Planung im Kopf...weiter ging nicht...es musste etwas aufs Papier bzw. es musste ein Schaltplan her. Als ersten Entwurf hatte ich die ganze Schaltung auf 2 Platinen untergebracht. Das war aber nicht gut da ich mit zu dicken Leiterbahnen gearbeitet hatte. Ich hatte diese nicht zwischen 2 Kontakten durchgezeichnet, da ich dachte das ich dieses so nicht ätzen könnte. Ich änderte also alles (siehe Bild rechts) und schon passte alles auf eine 200x150mm Platine. Meinen Schaltplan und das Platinenlayout entwickelte ich mit ABACOM SPlan und Sprint-Layout.



Nochmal eine Zusammenfassung: Also der µC schickt ein 8-Bit Bitmuster zum ersten Latch (für Säule 1 bis 8), setzt den Pin 11 (LE) auf HI/LO damit das Bitmuster gehalten wird. Danach neues Bitmuster für Säule 9 bis 16 ans zweite Latch u.s.w. bis alle 13 Latches gefüllt sind. Die Ebenen werden ja eigentlich durch die Mosfets geschaltet.

µC Pinbenutzung:

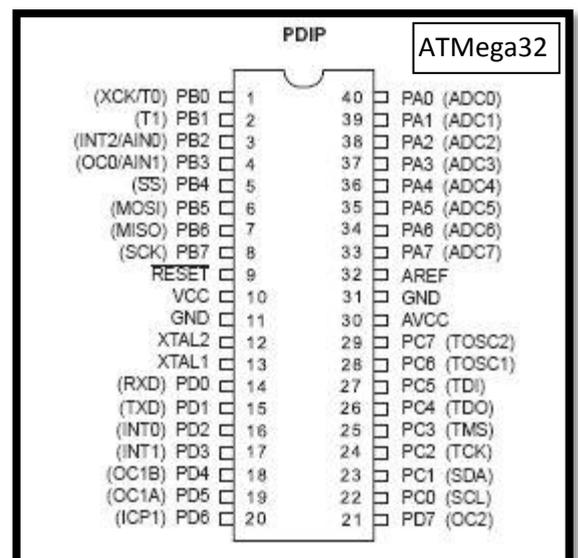
8 x Datenleitungen für Bitmuster zu den Latches

13 x Latch Enabled

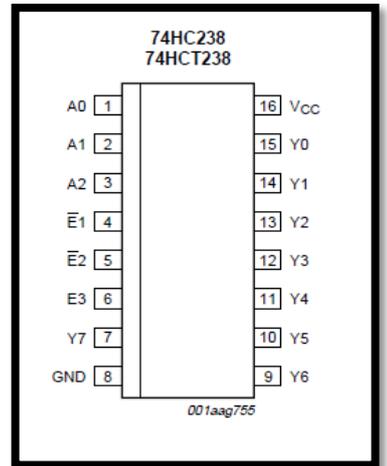
10 x Ebenensteuerung der Mosfets

31 Pins werden benötigt....bis jetzt...

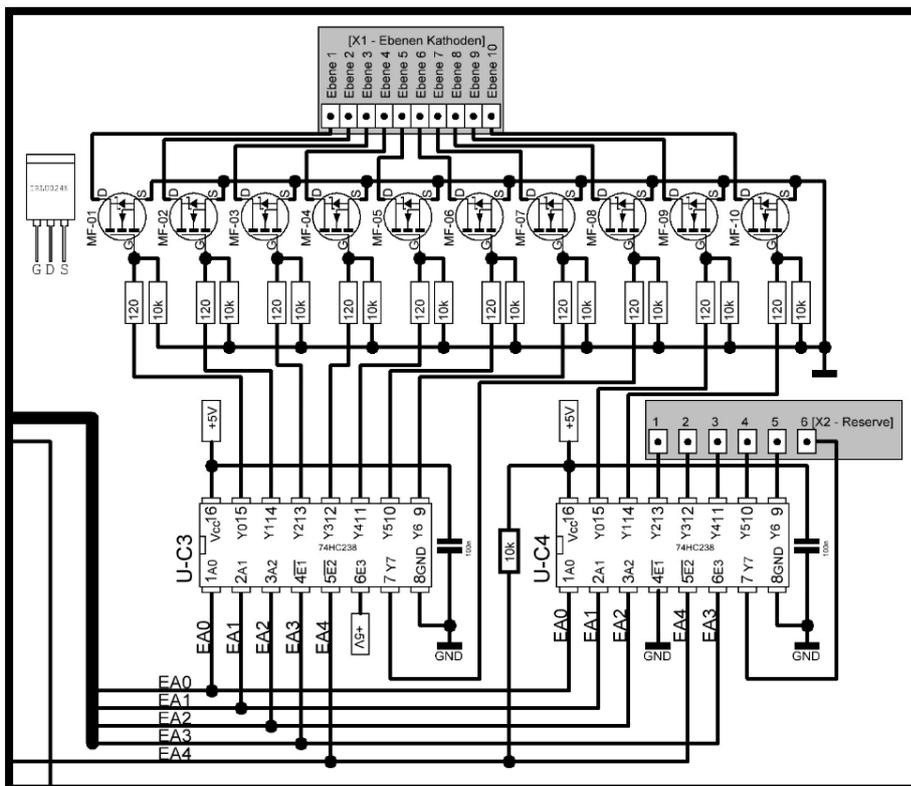
Nun hat ein ATmega 32 ja nur 40 IO-Pins...und ich hatte mir im Hinterkopf behalten, die Muster mal auf eine SD-Karte zu speichern und sie von dort aus anzeigen zu lassen.



Das 13te Latch ist nur bis zur Hälfte gefüllt! Weil 12 Latches x 8 Ausgänge sind 96 und 4 Ausgänge vom 13ten Latch sind 100. Wir haben auch 100 Säulen. Was macht man nun mit den letzten 4 Bits des 13ten Latches...die übrig sind? Da hat mich ein User eines Forums auf eine geniale Idee gebracht. Man könnte die letzten 4 Bits benutzen um die Ebenen auszuwählen. Das ist natürlich genial, man schickt ja sowieso ein Bitmuster zum letzten Latch...da können die letzten 4 Bits ja ruhig schon mal die entsprechende Ebene auswählen. Das bewerkstelligen wir mit einem 74HC238 (3-zu-8 Decoder/Demultiplexer). Da der Decoder aber nur 8 Ausgänge um die Mosfets anzusteuern hat aber unser Würfel 10 Ebenen hat die wir schalten müssen benötigen wir 2 Decoder. Diese entsprechend verschaltet können wir nun 10 Ebenen ansteuern mit nur 5 Leitungen. 4 Leitungen vom letzten (13ten) Latch und eine direkt vom μC um alle Ebenen komplett abzuschalten.



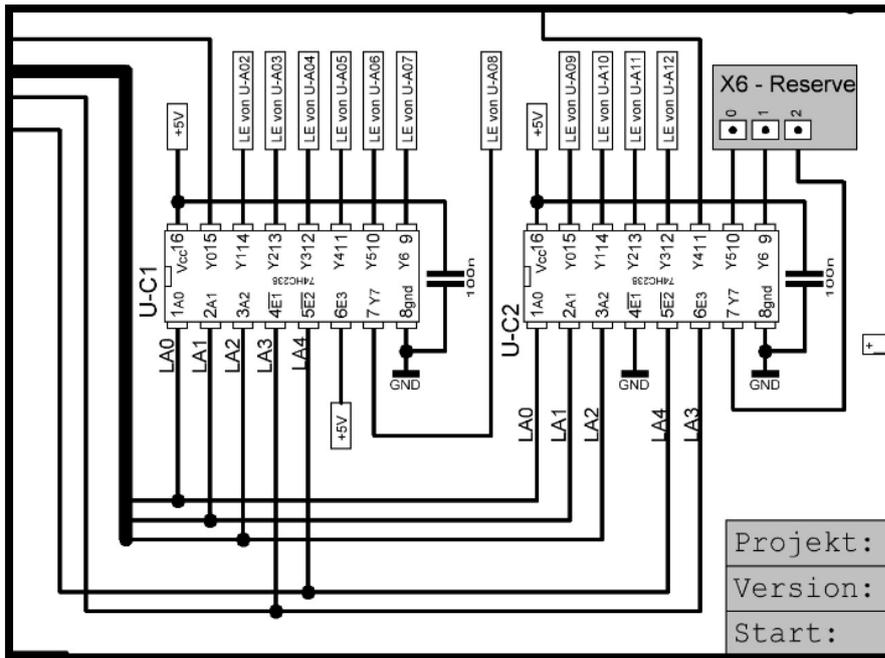
Hier mal ein Auszug vom Schaltplan:



EA0 bis EA3 (EA=EbenenAdresse) kommen vom 13ten Latch und EA4 kommt direkt vom μC . Nur wenn dieser (EA4) auf LO ist werden die Ebenen angesteuert. Mit EA3 wird einer der beiden Decoder ausgewählt. EA0 bis EA2 sind die Decodereingänge. Das ist ganz praktisch damit beim Programmieren des μC keine wirren Signale ausgegeben werden und die LED's des Würfels kaputt gehen. (Die Rv der LED's sind ja nur für den Multiplexbetrieb der LED's dimensioniert) Alle I/O-Pins des μC werden beim Programmieren eh auf LO gezogen und damit die Ebenensteuerung des Würfels ausgeschaltet.

Das gleiche Prinzip benutzen wir auch um die LE (LatchEnabled) Pins zu schalten. Davon haben wir ja schließlich 13 Stück.

Hier ein Auszug vom Schaltplan:



LA0 bis LA4 (LatchAdresse) kommen alle vom μC .
 LA4 auf LO und die Decoder sind enabled.
 Mit LA3 wird einer von den beiden Decoder ausgewählt.
 Und LA0 bis LA2 sind die Decodereingänge.

Mit diesen beiden Extraschaltungen verringert sich unser „Pin-Bedarf“ den μC folgendermaßen:

μC Pinbenutzung:

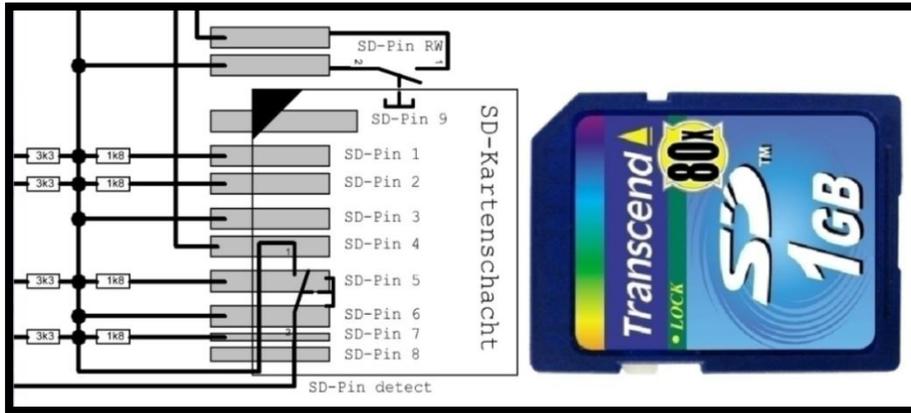
- 8 x Datenleitungen für Bitmuster zu den Latches
- 5 x Latch Enabled
- 1 x Ebenensteuerung der Mosfets

 14 Pins werden benötigt....bis jetzt...

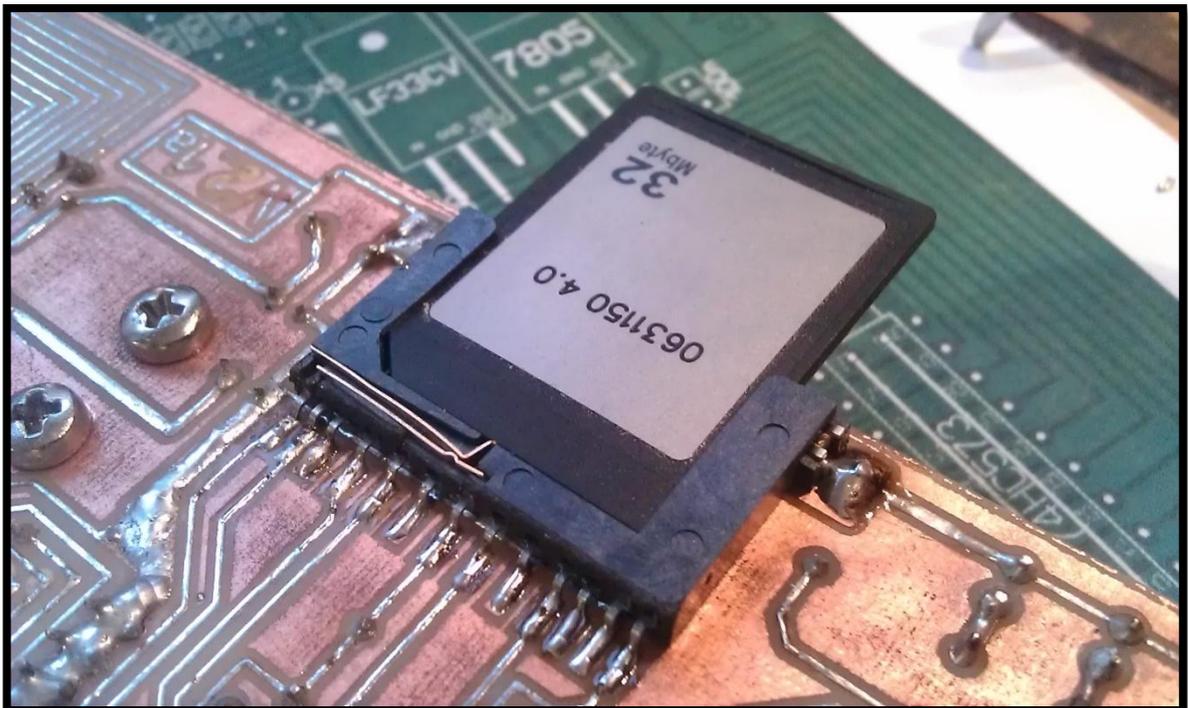
So weit so gut...

Nun dachte ich mir noch... es wäre doch toll wenn die Muster nicht im Speicher es μC sind sonder auf einer SD-Karte abgespeichert wären. Also lötete ich aus einem alten SD-Card-Reader einen SD-Karten-Slot aus und arbeitete ihn in den Schaltplan ein. Schön war auch das der Kartenschacht auch einen Kontakt hat zur Erkennung ob eine Karte eingesteckt ist und einen Kontakt zum Schreibschutz hat.

Hier mal ein Auszug aus dem Schaltplan:



Da die Speicherkarte eine Spannung von 3,3V möchte benötigen wir also noch einen 3,3V Positiv Festspannungsregler.



Was wir noch brauchen sind viele 100nF Kondensatoren, 3 Grüne LED's (Anzeige Versorgungsspannung, Anzeige 5V, Anzeige 3,3V) 16MHz Quarz, 100 Widerstände 40,2 Ω , Stiftsockelleisten....weiteres siehe Schaltplan....

aber erstmal die Bauteilliste...

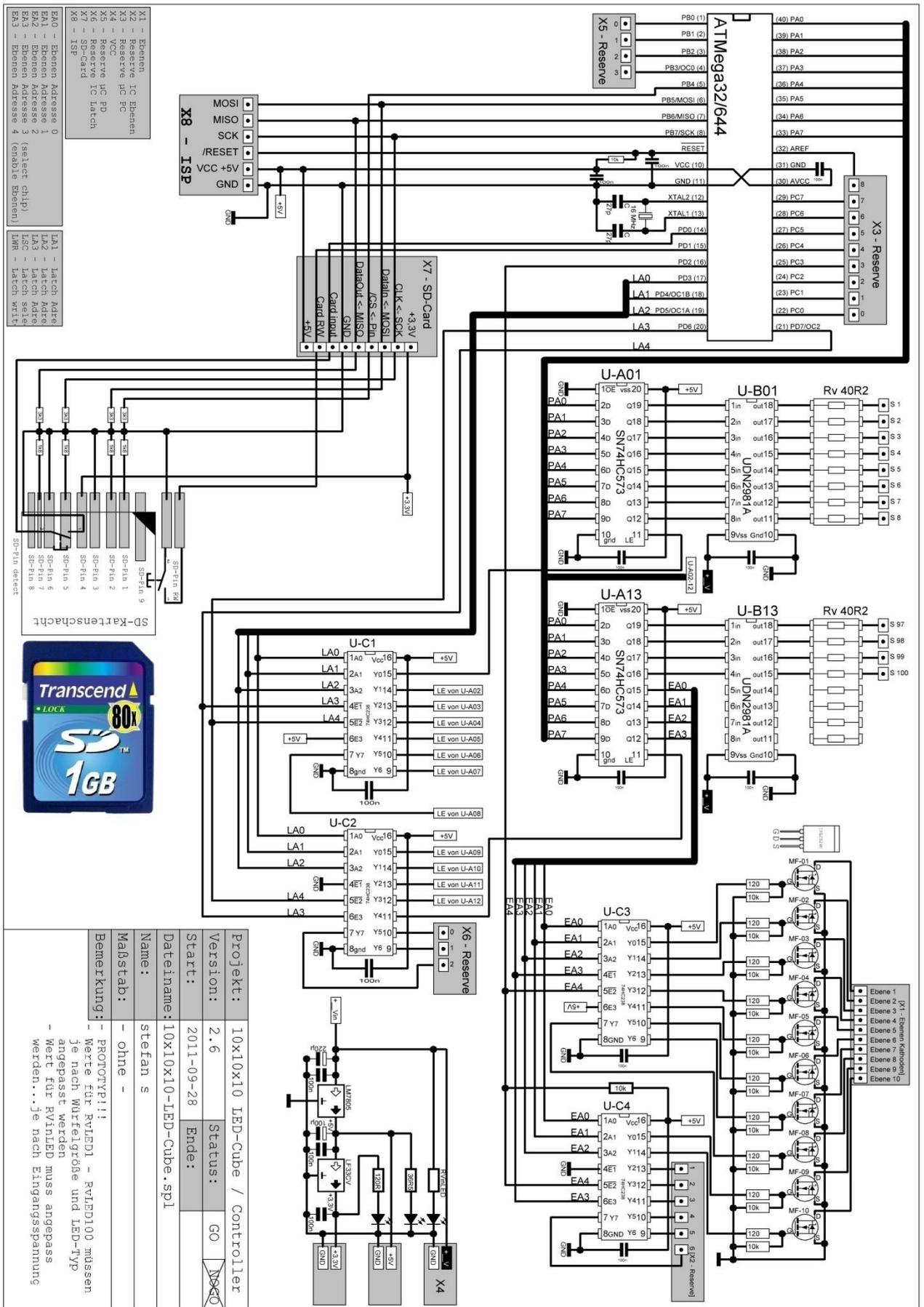
Bauteilliste (meine:-)

Bauteil	Stück	Preis	Anbieter
LED's	1050	61,94 €	eBay
µC (ATMega)	1	hatte ich	
Quarz 16MHz	1	hatte ich	
Latch (SN74HC573)	13	hatte ich	
LED-Treiber (UDN2981A)	13x á 1,50 €	15,00 €	Reichelt
Mosfets (IRLU024N)	10x á 0,42€	4,20 €	Reichelt
IC-Sockel	43	hatte ich	
Verzinnter Kupferdraht 0,91mm ² , 86m	2 Rollen á 16,78 €	33,74 €	Mercateo
Widerstände	1000 Stück	hatte ich	
3-zu-8 Decoder (74HC238)	4 x á 0,29 €	1,16 €	Reichelt
50pol. Stiftsockelleisten, gerade, RM 2,54	10x á 0,26 €	2,60 €	Reichelt
Festspannungsregler 3,3V TO-220	1	0,74 €	Reichelt
Festspannungsregler 5V TO-220	1	hatte ich	
Widerstände 40,2 Ω	100	hatte ich	
LED grün 3,5mm	3	hatte ich	
Kühlkörper für der Festspannungsregler	2	hatte ich	
Hauptkatalog ;-)	1	0,00 €	Reichelt

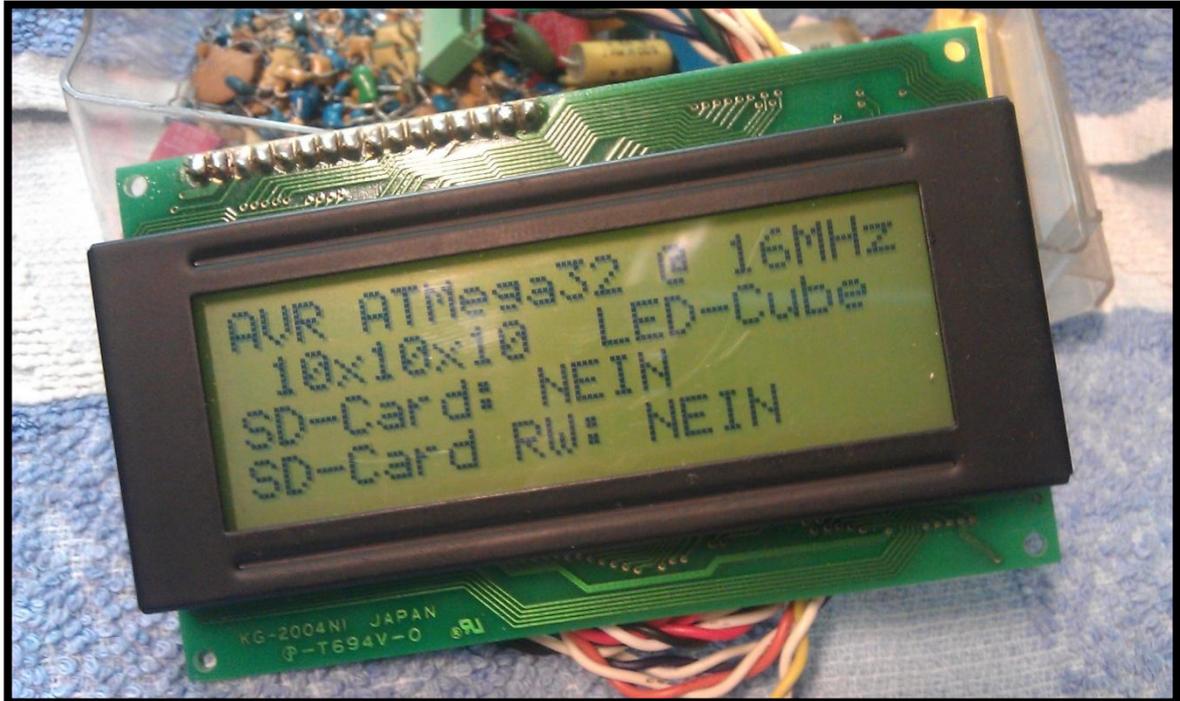
Die Preise können abweichen (gibt ja Rabatt ab einer bestimmten Stückzahl) und da ich mehr bestellt habe als ich für dieses Projekt benötige um in den Stückzahlrabatt zu kommen!

Jetzt aber zum Schaltplan....

Schaltplan



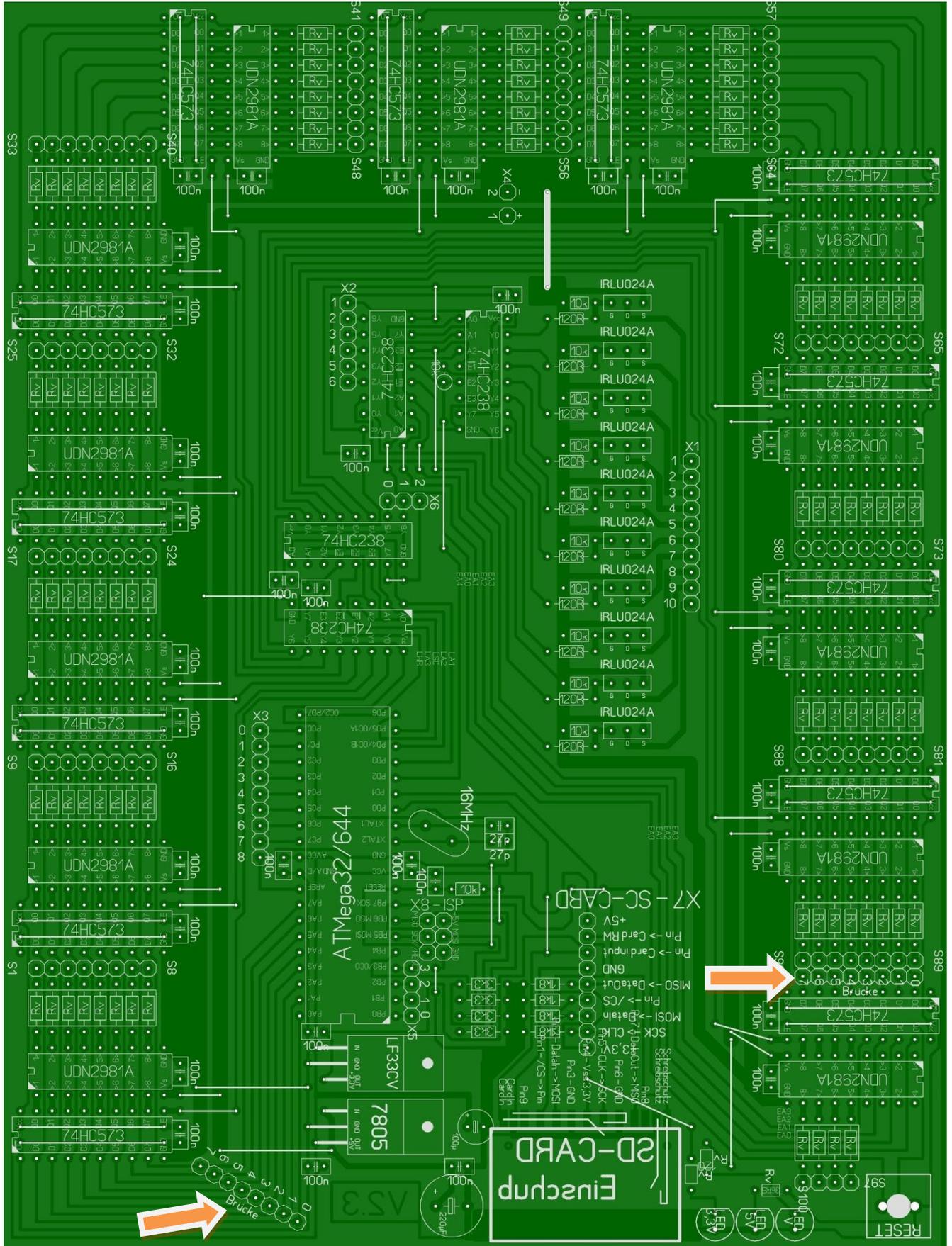
Wie man sehen kann wurden alle freien Pins auf Stiftsockelleisten gelegt um weitere Dinge anschließen zu können. Wie z.B. ein LCD-Display...neee.... heißt ja eigentlich LC-Display....um Statusmeldungen vom Würfel angezeigt zu bekommen.



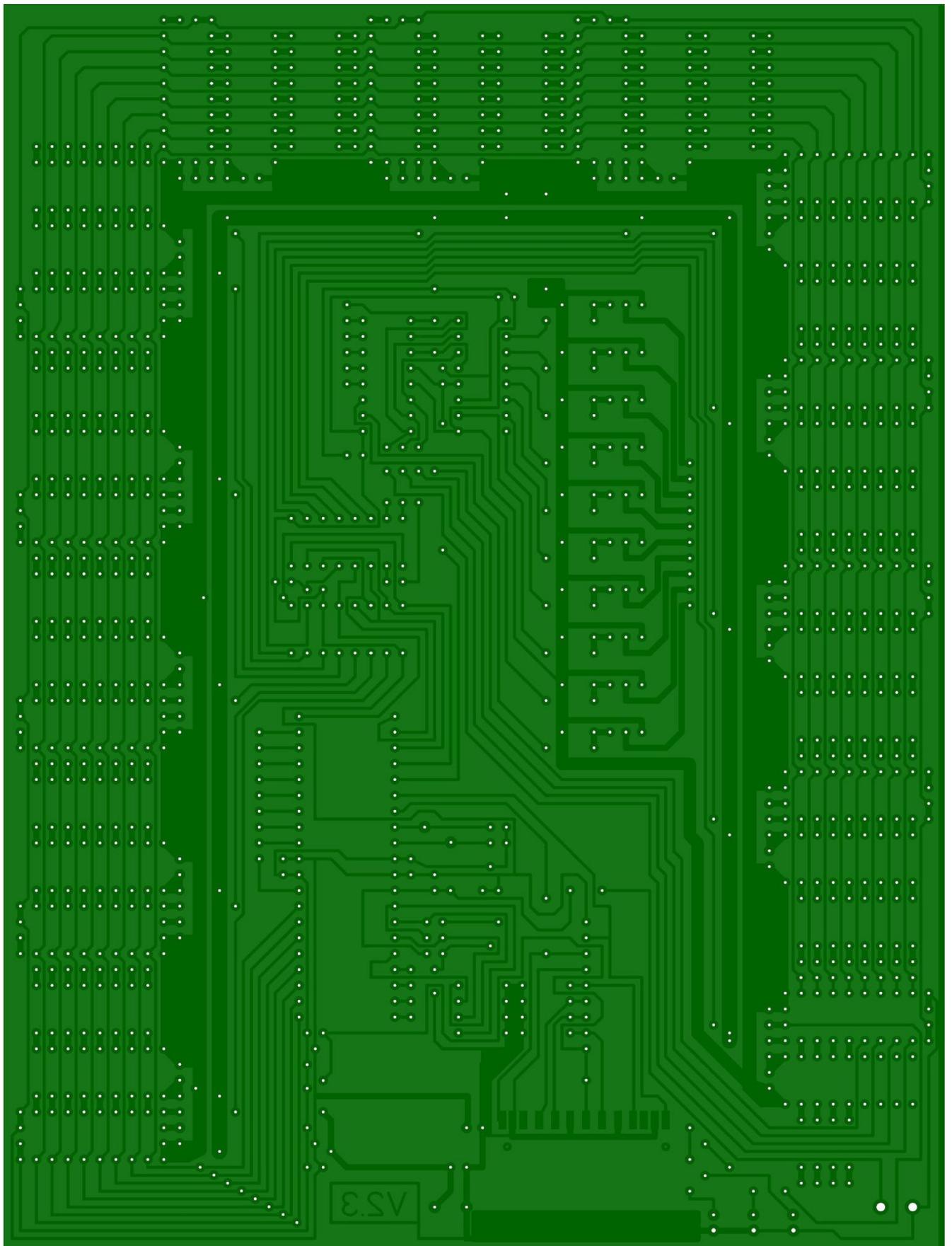
Weiter zum Platinenlayout erstellen. Das Layout habe ich mit Sprint-Layout von Abacom erstellt.

Hier das Layout von der Bestückungsseite, Lötseite und Bestückungsseite mit Bauteilen. Im folgendem Bild sind 2 Pfeile zu sehen. Da muss eine 8pol. Brücke rein. Warscheinlich sind die 8 Datenleitungen die rings rum verlaufen zu dünn bzw. haben einen zu hohen Widerstand...jedenfalls kommt an den letzten beiden Latches nur noch Müll an. Daher die Brücke NICHT vergessen!!!

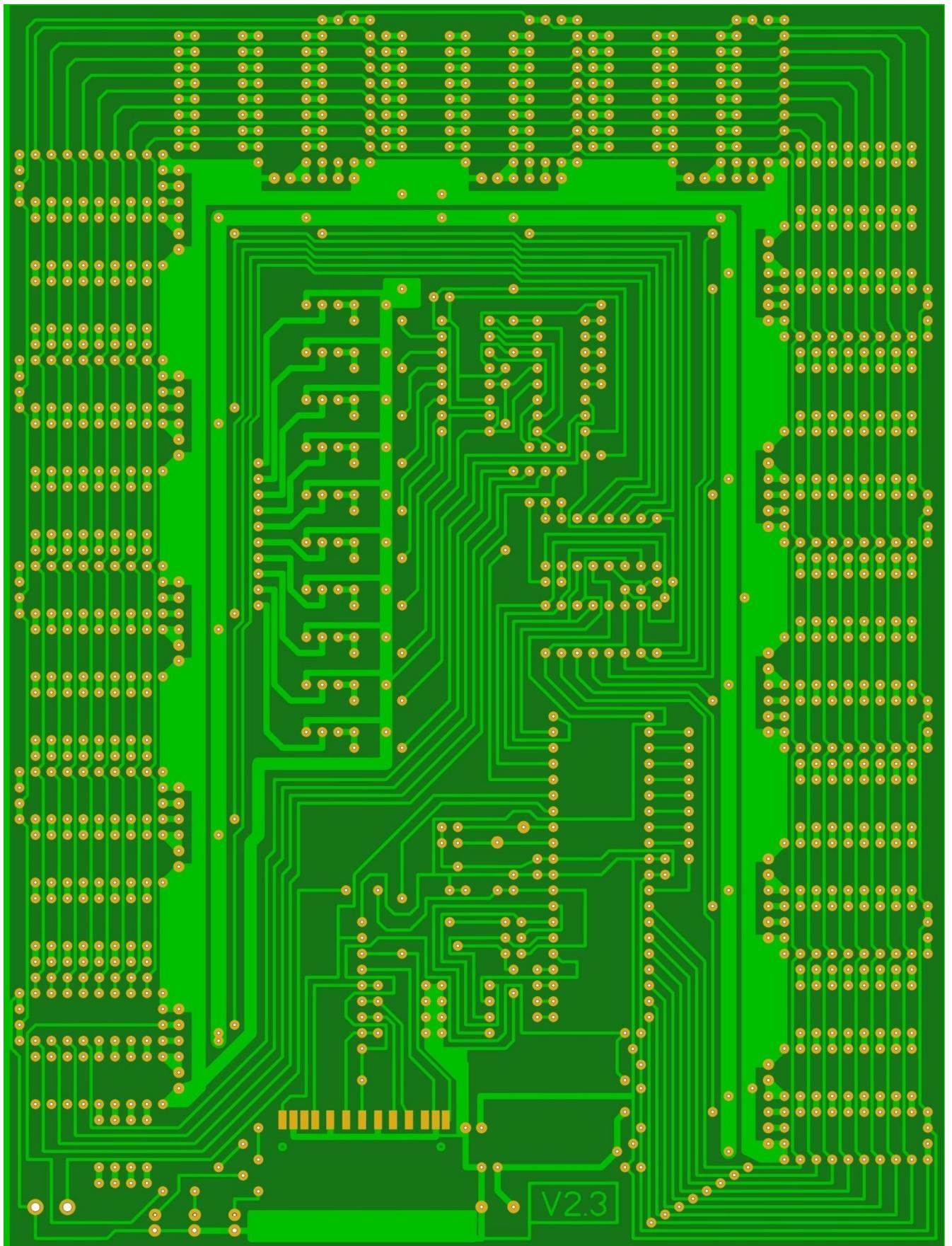
Platinenlayout V 2.3



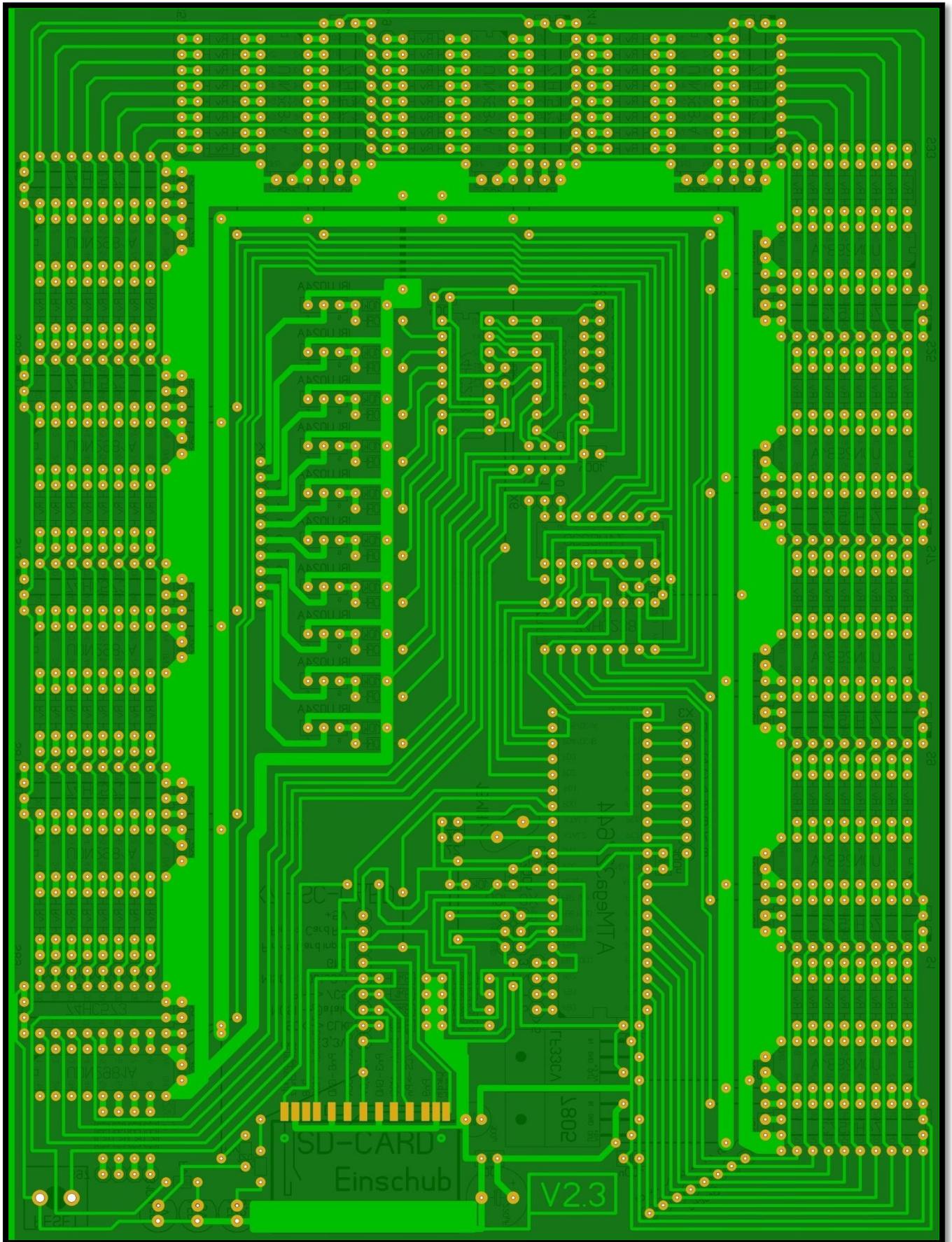
Bestückungsseite, Leiterbahnen auf der Unterseite durchscheinend
ACHTUNG!!! Brücke nicht vergessen (Pfeile)



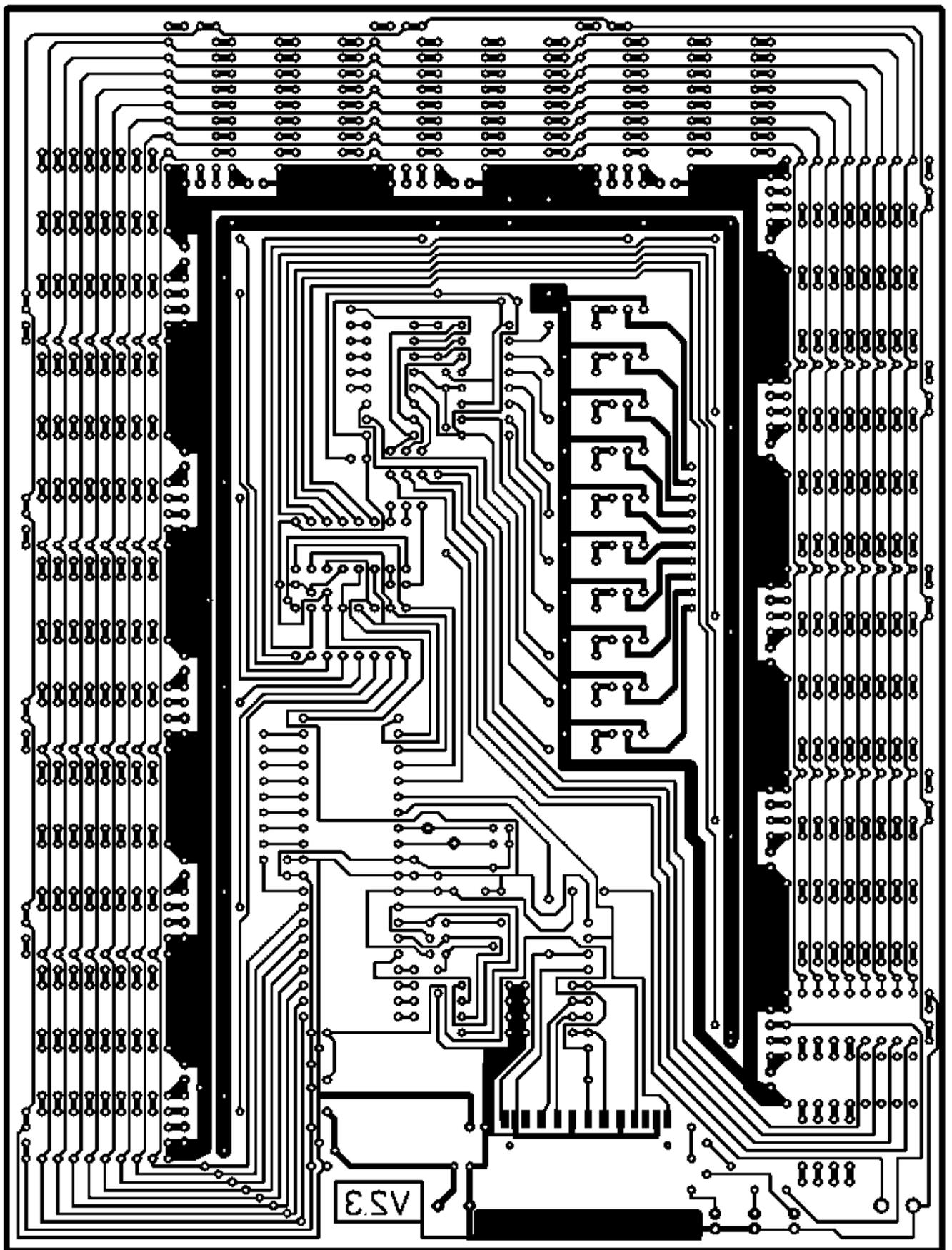
Bestückungsseite ohne Bauteile, Leiterbahnen auf der Unterseite durchscheinend



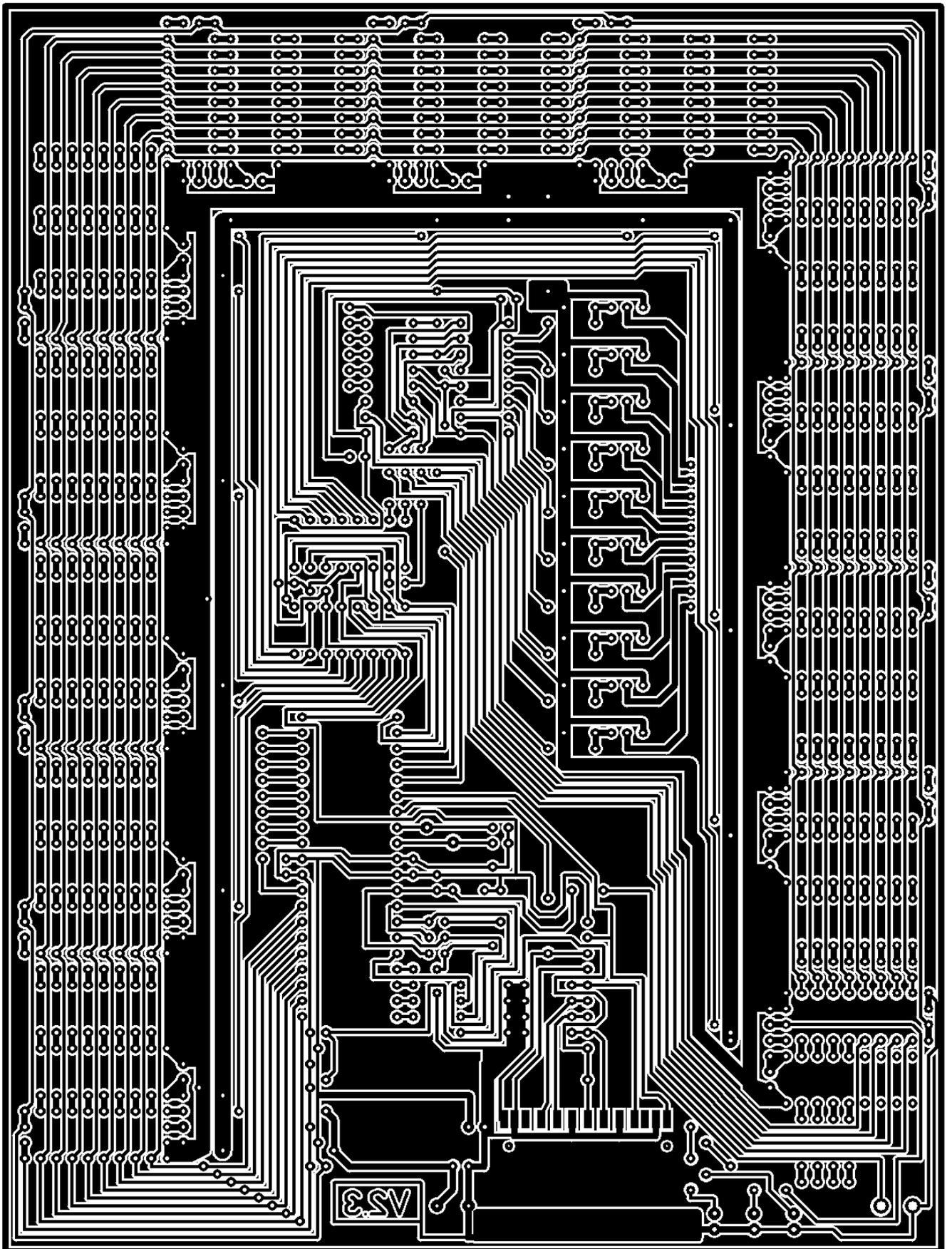
Lötseite



Lötseite, Bauteile auf der anderen Seite durchscheinend



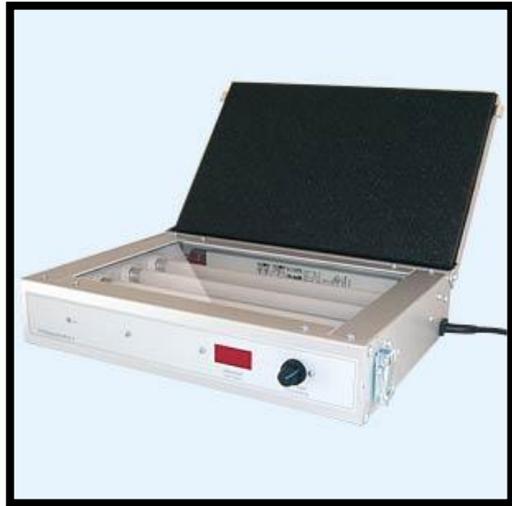
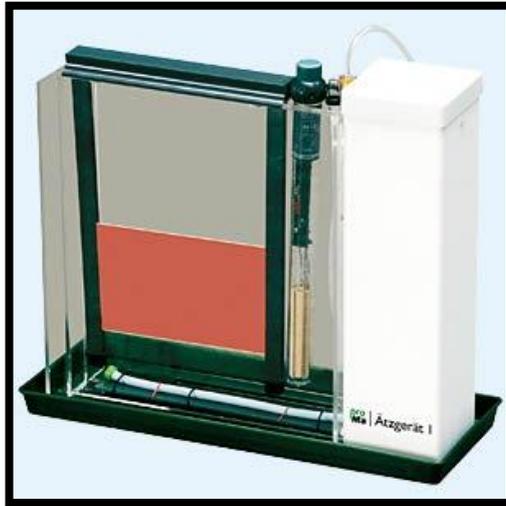
Vorlage (eventuell nicht 1:1) zum ätzen



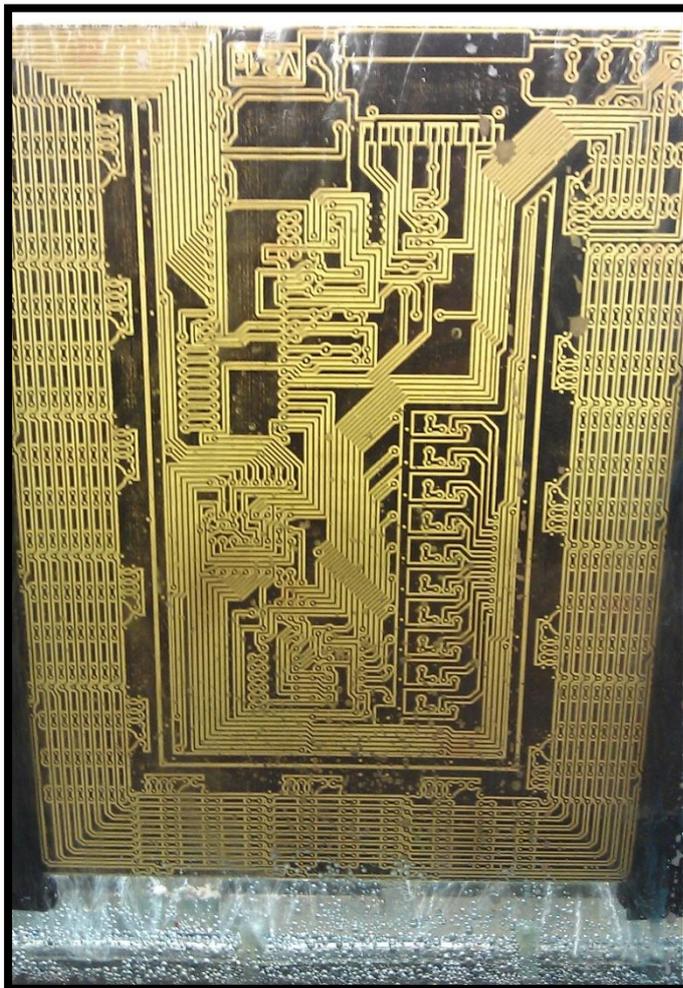
Vorlage mit AutoMasse (eventuell nicht 1:1) zum ätzen

Ätzen

Nachdem ich das Platinenlayout auf eine Transparentfolie via Laserdrucker ausgedruckt habe (Maßstabsgerecht!!!!) wurde eine Photobeschichtete Epoxyd 1,5mm Platine entsprechend mit dem Layout belichtet...in Entwickler Entwickelt und im Ätzbad geätzt...

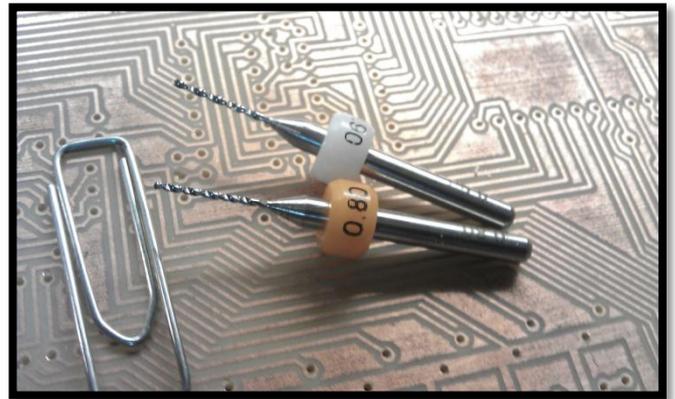


Hier mein Ätzgerät 1 von Reichelt und mein UV-Belichtungsgerät 1 von Reichelt

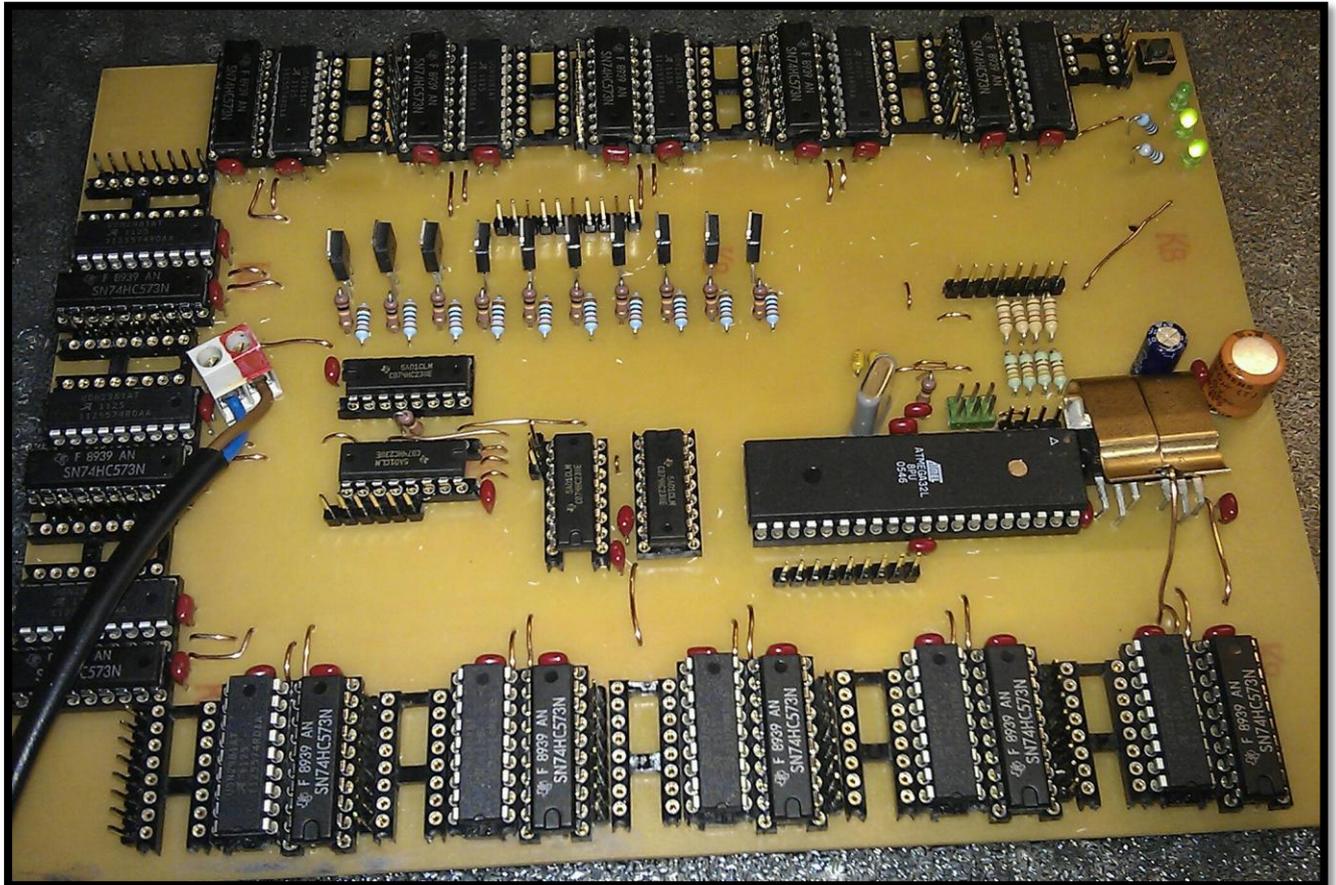


Platine wird geätzt...

Nachdem die Platine geätzt wurde und einige Leiterbahnen die dicht aneinander liegen auf „Nichtdurchgang“ geprüft wurden konnte es losgehen...**1286** Bohrungen zu bohren. Habe mir irgendwann mal diese Platinenbohrer gekauft...(Reichtl)...sind zwar nicht ganz billig mit 25€ aber die gehen super.



Danach ging es ans scheinbar unendliche bestücken...



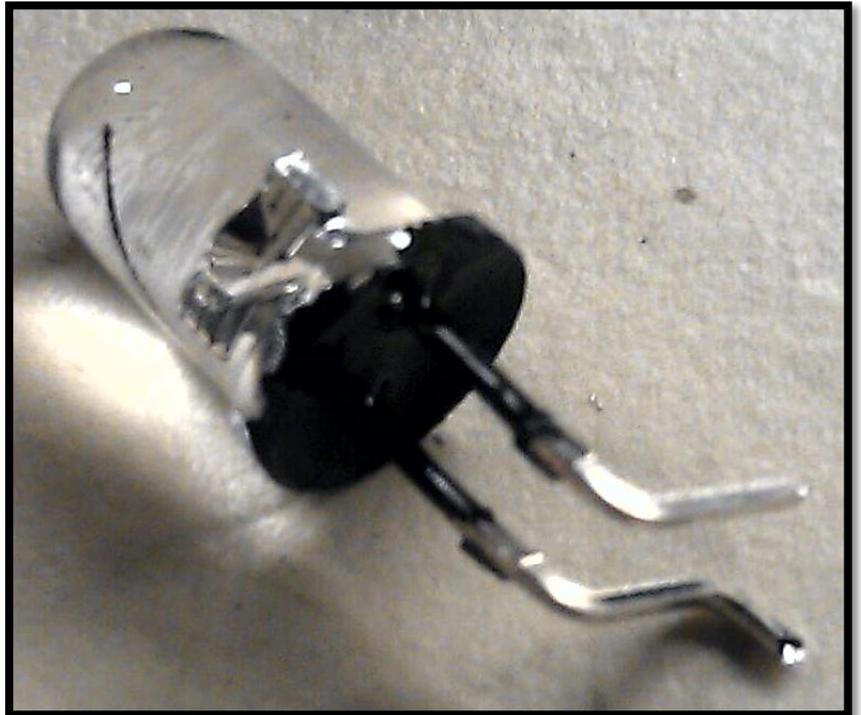
Auf die hier noch freien IC-Sockel kommen dann die Vorwiderstände der LED's. Aus der

Unterseite ist nur noch der SD-Karten-Slot. Auf der Unterseite deshalb, weil er als SMD zu löten ist.



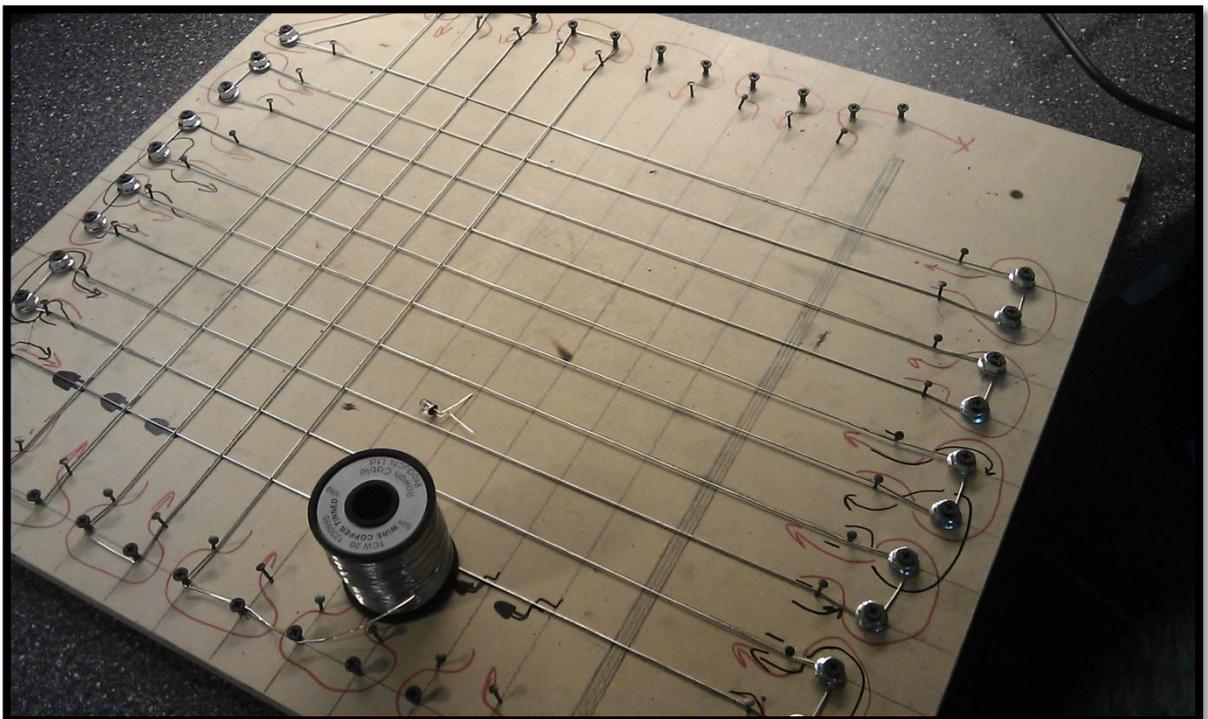
Würfel lötén

In mehreren Foren konnte ich lesen das (logischerweise) wasserklare LED's von unten in die darüberliegende reinstrahlen würden. Also bemalte ich alle 1050 LED's mit einem schwarzen Edding von unten an. Danach bog ich die Beinchen in die richtige Richtung und kürzte sie entsprechend. So wie auf dem folgenden Bild zu sehen ist. Das Beinchen welches nur 1x gebogen ist die die Kathode (Ebene) und das was 2x gebogen ist, ist die Anode (Säule).



Nun wie lötén?

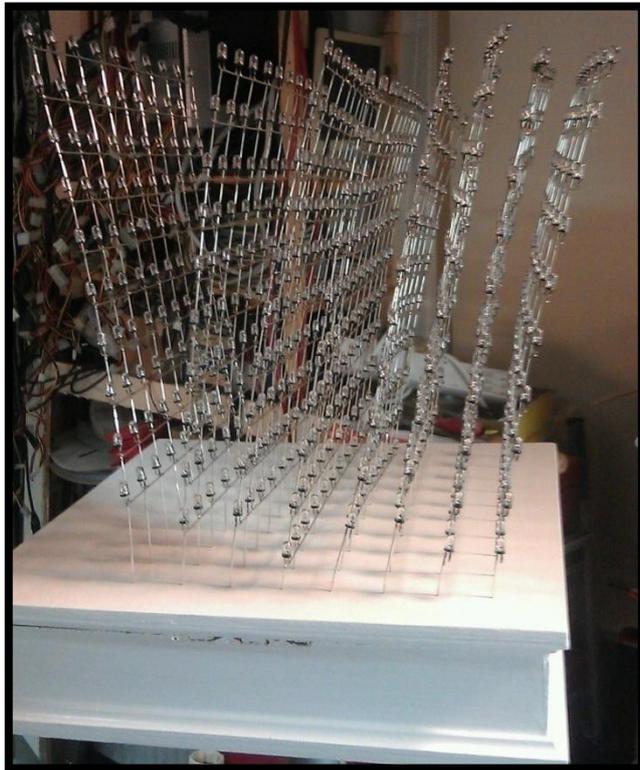
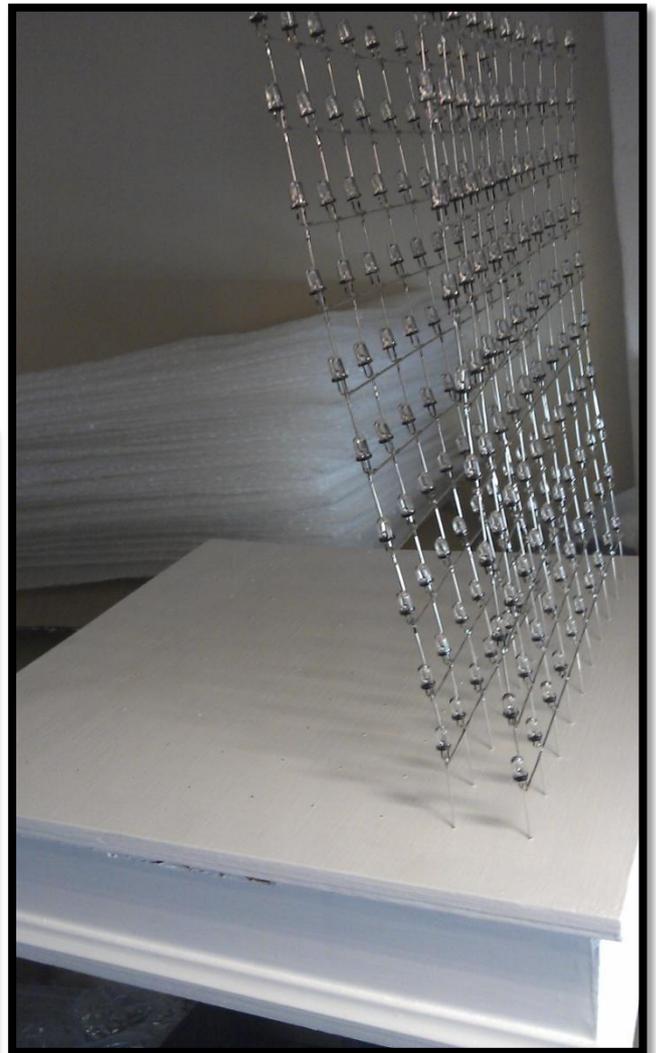
Ich baute mir aus einem Brett und Schrauben eine Halterung auf der ich den Verzinnten Kupferdraht spannte und ich nun recht komfortabel die LED's anlötén konnte.



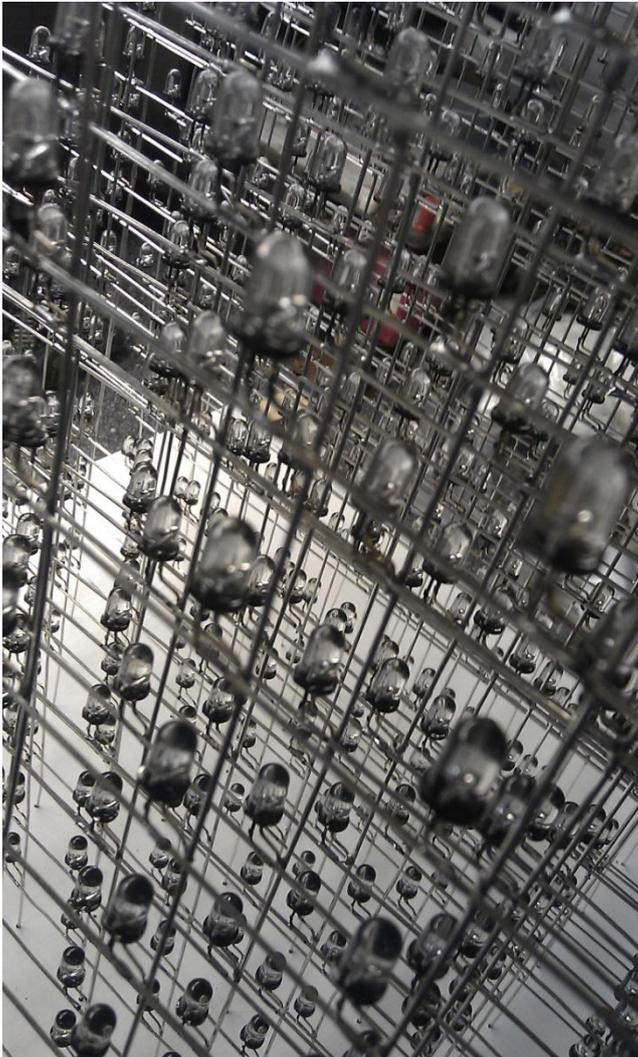
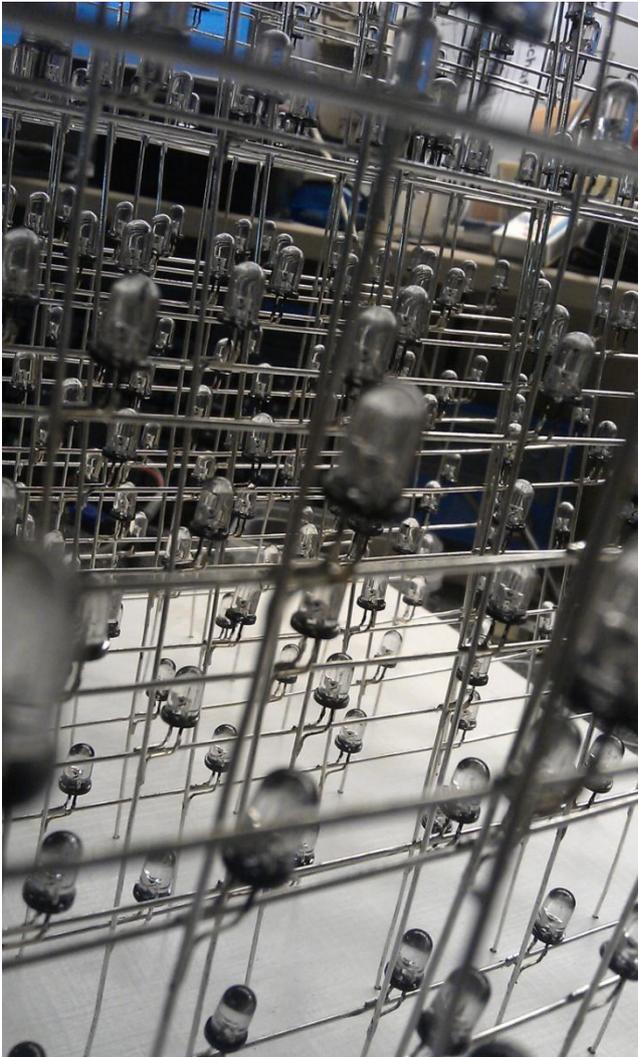
Nachdem eine „Spalte“ fertig gelötét war habe ich getestet ob auch alle funktionieren und alle die gleiche Helligkeit erzielten.



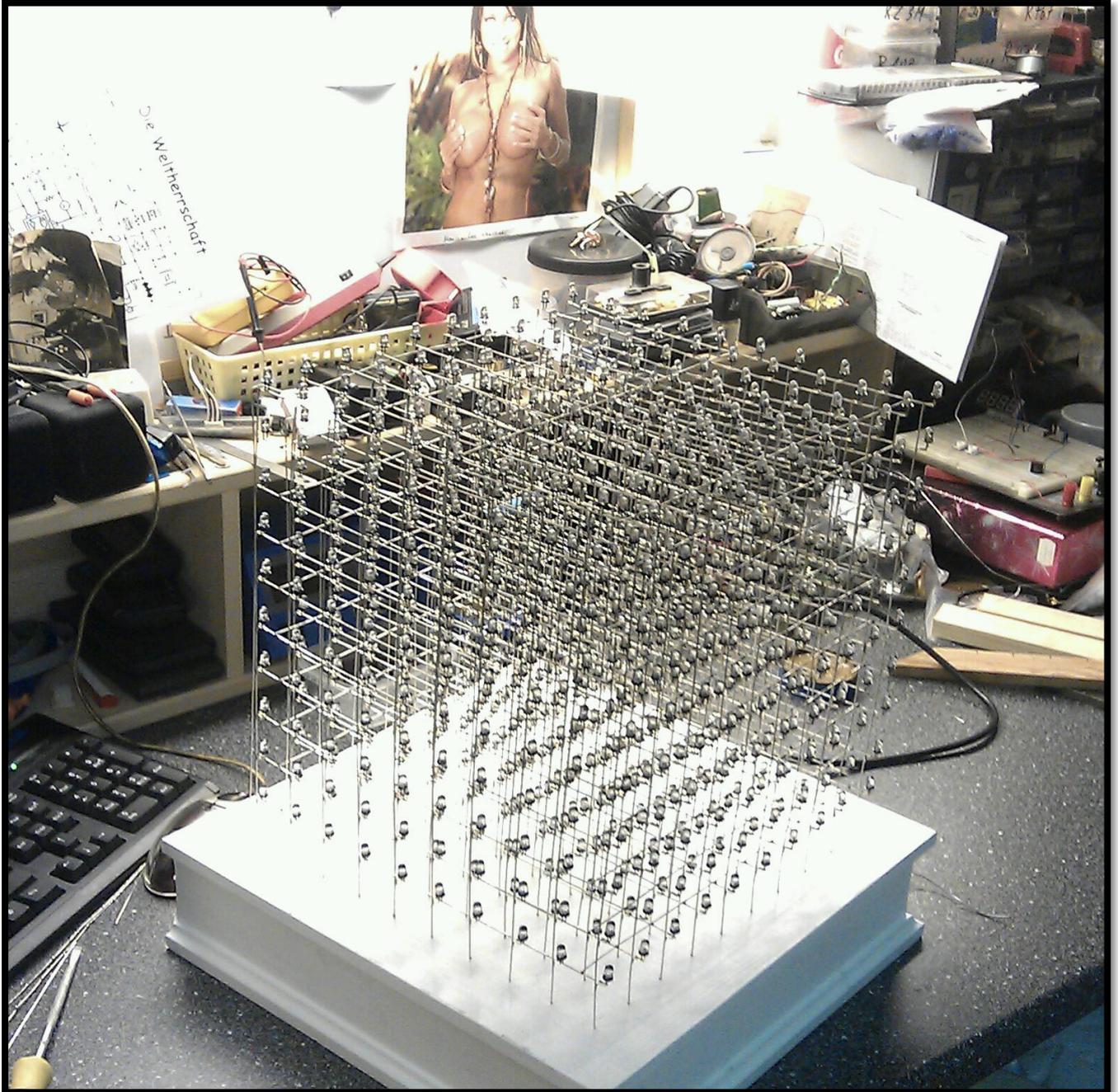
Wow...meine erste Spalte war fertig...nur wohin damit? Ich baute mir einen Holzkasten...verzierte ihn mit Zierleisten aus dem Baumarkt...lackierte ihn weiß...bohrte 110 kleine Löcher rein...100 für die Säulen und 10 für die Ebenen. Nun konnte ich meine erste Spalte schonmal einsetzen. Folgendes Foto entstand nach 2 Spalten. Das sieht dann so aus: →
Wenn man dann nicht die Lust verliert sieht es bald so aus: ↓



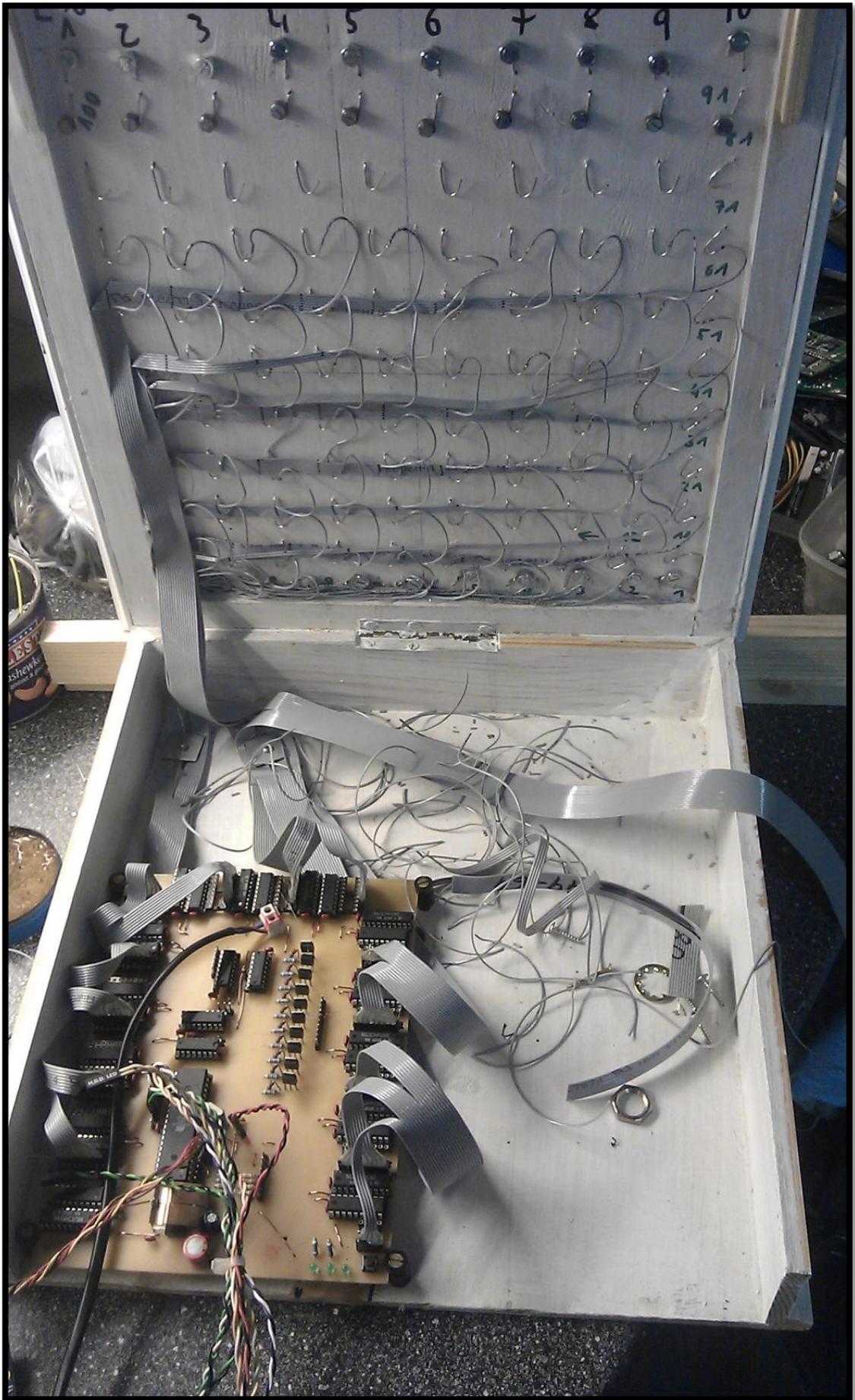
Hier noch einige coole Bilder



Nach dem ich nun auch die Ebenen verlötet hatte...sah es so aus...



Nun ging's ans inne verdrahten. Ich benutze Flachbandkabel. Naja Bilder sagen mehr als 1000 Worte ...



Zur Berechnung der Vorwiderstände der LED's.
Da ist bestimmt irgendwo ein Fehler drin...

Ich habe eine Versorgungsspannung von 9V=
Am UDN2981 fallen 1,6V und am IRLU024N fallen 0,1V ab.

Also: $9V - 1,6V - 0,1V = 7,3V$

$7,3V - 3,5V(\text{LED}) = 3,8V$

3,8V

----- = 38Ω also $39,2\Omega$ ich hab dann lieber $40,2\Omega$ genommen (hatte ich da)
 $0,100A$

↑LED mit 100mA Peak

Glaube so in etwa habe ich das berechnet...habe leider meinen Schmierzettel nicht gefunden wo die Rechnung drauf war...

Software

Zum Programmieren des ATMega's hab ich den ISP von IN-CIRCUIT (www.ic-board.de).

Und zwar den Programmieradapter „ICprog-AVR2.0“ .

Hier der Direktlink:

http://www.ic-board.de/product_info.php?info=p12_ICprog-AVR2-0.html

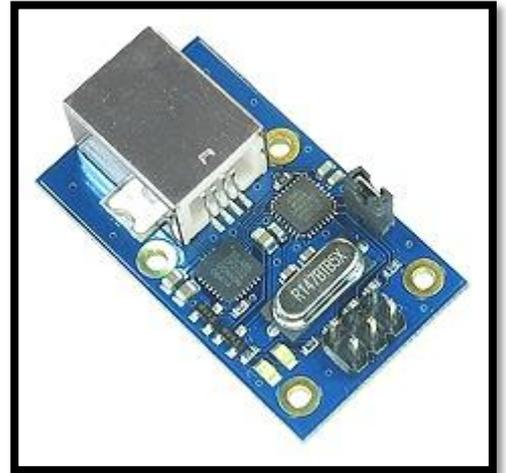
Als Software benutze ich BASCOM-AVR

(<http://www.mcselec.com>)

Hier direkt zum runterladen:

http://www.mcselec.com/index.php?option=com_docman&task=doc_download&gid=139&Itemid=54

Da der Code in der DEMO-Version nur 4k groß sein darf die Muster jedoch im Code mit drin sind musste ich das ganze immer bei einem Kumpel compilieren lassen.



SourceCode

Der Code ist nichts besonderes...der schwirrt überall im Netz rum....habe ihn nur soweit abgewandelt das man in einer Konfigurationstabelle folgende Angaben machen kann:

- Anzahl aller Animationen
- Anzahl der Bilder in jeder Animation
- Anzahl der Wiederholungen jeder Animation

Aber zunächst zum Bitmuster der Bilder:

Jede Animation besteht aus mehreren Bildern.

Das Bitmuster eines Bildes besteht aus 10 Zeilen (für jede Ebene eine).

Diese 10 Zeilen sind in 13 Blöcken (einer für jedes Latch) aufgeteilt.

```
Tabelle_animationen:
'Bild 1
Data &B00000000 , &B00000000
Data &B00000000 , &B00000001
Data &B00000000 , &B00000010
Data &B00000000 , &B00000011
Data &B00000000 , &B00000100
Data &B00000000 , &B00000101
Data &B00000000 , &B00000110
Data &B00000000 , &B00000111
Data &B00000000 , &B00001000
Data &B11111111 , &B11111001
```

In diesem Bild ist die ganze untere Ebene an.

Die letzten 4 Bits des 13ten Blocks ist die Ebenensteuerung!

```
Tabelle_animationen:
'Bild 1
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00000000
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00000001
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00000010
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00000011
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00000100
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00000101
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00000110
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00000111
Data &B00000000 , . . . , &B00000000 , &B00000000 , &B00000000 , &B00001000
Data &B11111111 , . . . , &B11111111 , &B11111111 , &B11111111 , &B11111001
```

Wenn man nun mehrere dieser Bilder hat....hat man schon eine Animation.

Die Konfigurationstabelle meines Würfels sieht so aus:

```
'Tabelle Konfiguration *****
Konfiguration:
'Anzahl aller Animationen
Data 10

'Animation 1 - 005 Welle 02
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 28 , 2

'Animation 2 - 002 schräg von unten links füllen und leeren
'Bilder in dieser Animation | Wiederholungen dieser Animation
```

```

Data 37 , 3

'Animation 3 - 013 - Welle loop
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 30 , 3

'Animation 4 - 008 Cube
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 1 , 30

'Animation 5 - 012 - von der mitte füllen und leeren
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 11 , 5

'Animation 6 - 011 - von innen nach außen und zurück
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 10 , 5

'Animation 7 - 004 - Spaltenweise von links füllen
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 20 , 3

'Animation 8 - 003 - Zeilenweise von unten füllen und leeren
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 20 , 3

'Animation 9 - 010 - Zeilen von unten nach oben
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 20 , 3

'Animation 10 - 009 - Spalten von links nach rechts
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 20 , 3

```

In der ersten **Data-Zeile** steht IMMER die Anzahl aller Animationen!!!

Mit der **nächsten Data-Zeile** wird angegeben das die folgende Animation aus 28 Einzelbildern besteht und diese Animation 2 mal wiederholt werden soll.

Danach folgt wie gesagt die „Tabelle_animationen“ ...

Hier mal der ganze Code:

(Es kann sein, das einige Definierten Variablen nicht benötigt werden...der Code ist halt noch in Bearbeitung)

```

*****
* Programmfunktion      : 10x10x10 LED-Cube          *
* Programmieradapter   : ICprog-AVR 2.0 (ISP)       *
* Board                : LED-Cube Controllerboard   *
* Controller           : ATmega32                  *
* Taktquelle           : XTAL                       *
* Taktfrequenz         : 16000000 Hz               *
* Code Version         : 1.6                       *
* Schaltplanversion    :                           *
* Programmiersprache   : BASCOM                    *
* Funktionsbeschreibung:                           *
*   Kurzfunktion: - SD-Card Ja/Nein                 *
*                 - Wenn Ja, Muster von Karte in Array laden *
*                 - Wenn Nein, Vordefiniertes Muster benutzen *
*                 - alle Latches für Ebene 1 füllen *
*                 - Ebene 1 an                       *
*                 - Ebene 1 aus                       *
*                 - alle Latches für Ebene 2 füllen *
*                 - Ebene 2 an                       *
*                 - Ebene 2 aus                       *
*                 - u.s.w.                           *
*
*   Detailbeschreibung: - Kontrolle ob SD-Card vorhanden ist oder nicht (PD0) *
*                       - wenn nein dann Muster aus Tabelle benutzen *
*                       - wenn ja dann Muster von SD-Card lesen *
*
*                       - LWR auf HI (alle Latch enable aus) *
*
*                       - Bitmuster anlegen (für ersten Latch, 8 Bits) *
*                       - ersten Latch enabled auf HI schalten *
*                         (Latch enabled Funktion) *
*                       - Bitmuster wird Ausgegeben (am ersten Latch) *
*                       - ersten Latch enabled auf LO schalten *
*
*                       - Bitmuster anlegen (für zweiten Latch, 8 Bits) *
*                       - zweiten Latch enabled auf HI schalten *
*                         (Latch enabled Funktion) *
*                       - Bitmuster wird Ausgegeben (am ersten Latch) *
*                       - zweiten Latch enabled auf LO schalten *
*
*                       - u.s.w. für Latch 3 bis 13 *
*
*
*   Latch Adresse (LA) *
*   ----- *
*   Latch 1 1 2 3 4 5 6 7 8 9 10 11 12 13 *
*   LA0 0 0 1 0 1 0 1 0 1 0 1 0 1 0 *
*   LA1 0 0 0 1 1 0 0 1 1 0 0 1 1 0 *
*   LA2 0 0 0 0 0 1 1 1 1 0 0 0 0 1 *
*   LA3 0 0 0 0 0 0 0 0 0 1 1 1 1 1 (select) *
*   LA4 1 0 0 0 0 0 0 0 0 0 0 0 0 0 (write) *
*   Y 0 1 1 1 1 1 1 1 1 1 1 1 1 1 *
*
*   LA3 - 1 - Latch speichert keine Daten *
*   LA3 - 0 - Latch speichert Daten *
*
*   LA4 - 0 - Latch 1 bis 8 *
*   LA4 - 1 - Latch 9 bis 13 *
*
*
*   Ebenen Adresse (EA) *
*   ----- *
*   EBENE - 1 2 3 4 5 6 7 8 9 10 *
*   EA0 - 0 1 0 1 0 1 0 1 0 1 0 *
*   EA1 - 0 0 1 1 0 0 1 1 0 1 *
*   EA2 - 0 0 0 0 1 1 1 1 0 0 *
*   EA3 - 0 0 0 0 0 0 0 0 1 1 *
*   EA4 1 0 0 0 0 0 0 0 0 0 *
*
*
*   Pinbelegungen *
*   ----- *
*   PA0 - Bitmuster D0 *
*   PA1 - Bitmuster D1 *
*   PA2 - Bitmuster D2 *
*   PA3 - Bitmuster D3 *
*   PA4 - Bitmuster D4 (13. Latch EA0) *
*   PA5 - Bitmuster D5 (13. Latch EA1) *
*   PA6 - Bitmuster D6 (13. Latch EA2) *
*   PA7 - Bitmuster D7 (13. Latch EA3) *
*
*   PB0 - frei *
*   PB1 - frei *
*   PB2 - frei *
*   PB3 - frei *
*   PB4 - SD-Card /CS *
*   PB5 - SD-Card DataIn, ISP-MOSI *
*   PB6 - SD-Card DataOut, ISP-MISO *
*   PB7 - SD-Card CLK, ISP-SCK *
*
*   PC0 - frei *
*   PC1 - frei *
*   PC2 - frei

```

```

'* PC3 - frei *
'* PC4 - frei *
'* PC5 - frei *
'* PC6 - frei *
'* PC7 - frei *
'* *
'* PD0 - SD-Card input detect - SDI *
'* PD1 - SD-Card Schreibschutz detect - SDW *
'* PD2 - Enable Cube - EA4 *
'* PD3 - Latch Adresse 0 - LA0 *
'* PD4 - Latch Adresse 1 - LA1 *
'* PD5 - Latch Adresse 2 - LA2 *
'* PD6 - Latch Adresse 3 (select) - LA3 *
'* PD7 - Latch Adresse 4 (write) - LA4 *
'* *
'* Bemerkungen: : *
'* *
'*****

$regfile "m32def.dat"
$crystal = 16000000

$hwstack = 32
$swstack = 32
$framesize = 32

' Pinkonfiguration *****

'Datenrichtungsregister Port A (offset), 1=Ausgang, 0=Eingang
'Pin 76543210
Ddra = &B11111111

'Datenrichtungsregister Port D, 1=Ausgang, 0=Eingang
'Pin 76543210
Ddrd = &B11111100

'PullUp aktivieren (1-an 0-aus) (Taster gegen GND)
'Pin 76543210
Portd = &B00000011

Sdi Alias Portd.0 'SD-Card input detect
Sdw Alias Portd.1 'SD-Card Schreibschutz detect
Ea4 Alias Portd.2 'Enable Cube, 1-an, 0-aus
La0 Alias Portd.3 'Latch Adresse 0
La1 Alias Portd.4 'Latch Adresse 1
La2 Alias Portd.5 'Latch Adresse 2
La3 Alias Portd.6 'Latch Adresse 3 select
La4 Alias Portd.7 'Latch Adresse 4 write

' Würfel vorbereiten *****
Ea4 = 1 'Enable Cube 0-AN, 1-AUS

La4 = 1 'alle Latches werden disabled

' Alle Latches leeren *****
Porta = &B00000000
La4 = 0 : Waitms 10
La0 = 0 : La1 = 0 : La2 = 0 : La3 = 0 : Waitms 10 'Latch 1
La0 = 1 : La1 = 0 : La2 = 0 : La3 = 0 : Waitms 10 'Latch 2
La0 = 0 : La1 = 1 : La2 = 0 : La3 = 0 : Waitms 10 'Latch 3
La0 = 1 : La1 = 1 : La2 = 0 : La3 = 0 : Waitms 10 'Latch 4
La0 = 0 : La1 = 0 : La2 = 1 : La3 = 0 : Waitms 10 'Latch 5
La0 = 1 : La1 = 0 : La2 = 1 : La3 = 0 : Waitms 10 'Latch 6
La0 = 0 : La1 = 1 : La2 = 1 : La3 = 0 : Waitms 10 'Latch 7
La0 = 1 : La1 = 1 : La2 = 1 : La3 = 0 : Waitms 10 'Latch 8
La0 = 0 : La1 = 0 : La2 = 0 : La3 = 1 : Waitms 10 'Latch 9
La0 = 1 : La1 = 0 : La2 = 0 : La3 = 1 : Waitms 10 'Latch 10
La0 = 0 : La1 = 1 : La2 = 0 : La3 = 1 : Waitms 10 'Latch 11
La0 = 1 : La1 = 1 : La2 = 0 : La3 = 1 : Waitms 10 'Latch 12
La0 = 0 : La1 = 0 : La2 = 1 : La3 = 1 : Waitms 10 'Latch 13
La4 = 1 : Waitms 10

' Watchdog Konfigurieren *****
Config Watchdog = 2048 'Timeout 2 Sekunden (2048 ms)
Start Watchdog ' Watchdog starten

' Variablen definieren *****

'Dim Maxbild As Word
Dim Bild As Word
Dim Dauer As Byte
Dim Ebene As Byte
Dim Temp As Word
Dim Offset As Word
Dim Startposition As Word
Dim T1 As Word
Dim T2 As Word

```

```

Dim Max_anim As Byte           'Anzahl dex Animationen
Dim Loops As Byte             'Anzahl der Wiederholungen der Animation
Dim Zeigerposition As Word    'Wert aus Konfigurationstabelle
Dim Animation As Word        'Schleife Animation
Dim Looping As Byte          'Schleife Wiederholungen der Animation
Dim Max_bild As Byte         'Anzahl der Bilder in einer Animation

'START LCD *****
Config Lcdpin = Pin , _
Db4 = Portc.3 , _
Db5 = Portc.2 , _
Db6 = Portc.1 , _
Db7 = Portc.0 , _
E = Portc.4 , _
Rs = Portc.6

Config Lcd = 20 * 4
Config Pinc.5 = Output
Portc.5 = 0

Initlcd
Cursor Off
Cls

Locate 1 , 1
Lcd " LED-Cube Status"

Locate 2 , 1
Lcd "-----"

Locate 3 , 1
Lcd "Looping : "

Locate 4 , 1
Lcd "Animation: "
'ENDE LCD *****

'Programmstart *****

Ea4 = 1           'Enable Cube 0-AN, 1-AUS
Reset Watchdog  'Watchdog zurücksetzen
Zeigerposition = 0 'Zeiger an erster (0-ter) Zelle
Max_anim = Lookup(zeigerposition , Konfiguration) 'Anzahl aller Animationen holen
Max_anim = Max_anim - 1 'Weil Schleife von 0 beginnt

Do
  Zeigerposition = 1 'Zeiger in zweite Zelle
  Startposition = 0

  For Animation = 0 To Max_anim 'Schleife der aktuellen Animation
    ' LCD ***** LCD
    Animation = Animation + 1
    Locate 4 , 12 : Lcd " "
    Locate 4 , 12 : Lcd Animation
    Animation = Animation - 1
    Locate 4 , 14 : Lcd "/"
    Max_anim = Max_anim + 1
    Locate 4 , 15 : Lcd Max_anim
    Max_anim = Max_anim - 1
    ' LCD ***** LCD

    Max_bild = Lookup(zeigerposition , Konfiguration) 'Bildanzahl der aktuellen Animation holen
    Max_bild = Max_bild - 1 ' Weil Schleife von 0 anfängt
    Zeigerposition = Zeigerposition + 1 'in nächste Zelle (3) springen für Loops
    Loops = Lookup(zeigerposition , Konfiguration)
    Loops = Loops - 1 ' Weil Schleife von 0 anfängt

    For Looping = 0 To Loops
      ' LCD ***** LCD
      Locate 3 , 12 : LCD " "
      Looping = Looping + 1
      Locate 3 , 12 : Lcd Looping
      Looping = Looping - 1
      Locate 3 , 14 : Lcd "/"
      Loops = Loops + 1
      Locate 3 , 15 : Lcd Loops
      Loops = Loops - 1
      ' LCD ***** LCD

      For Bild = 0 To Max_bild 'Schleife Bilderanzahl in der aktuellen animation
        For Dauer = 0 To 5 'Dauer wie lange ein Bild angezeigt werden soll (5 ist
ok)
          For Ebene = 0 To 9 '10 Ebenen durchmultiplexen
            Offset = Bild * 130 'Bytes pro Bild
            Temp = Ebene * 13 'Bytes pro Ebene
            Offset = Offset + Temp

```

```

Offset = Offset + Startposition
'steht 'Offset = Offset + 1 'kann weg wenn keine Bildanzahl an erster Stelle in der Mustertabelle

Porta = Lookup(offset , Tabelle_animationen) ' 1. Byte-Muster ausgeben
La0 = 0 : La1 = 0 : La2 = 0 : La3 = 0 : La4 = 0 : La4 = 1 'Latch 1
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 2. Byte-Muster ausgeben
La0 = 1 : La1 = 0 : La2 = 0 : La3 = 0 : La4 = 0 : La4 = 1 'Latch 2
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 3. Byte-Muster ausgeben
La0 = 0 : La1 = 1 : La2 = 0 : La3 = 0 : La4 = 0 : La4 = 1 'Latch 3
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 4. Byte-Muster ausgeben
La0 = 1 : La1 = 1 : La2 = 0 : La3 = 0 : La4 = 0 : La4 = 1 'Latch 4
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 5. Byte-Muster ausgeben
La0 = 0 : La1 = 0 : La2 = 1 : La3 = 0 : La4 = 0 : La4 = 1 'Latch 5
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 6. Byte-Muster ausgeben
La0 = 1 : La1 = 0 : La2 = 1 : La3 = 0 : La4 = 0 : La4 = 1 'Latch 6
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 7. Byte-Muster ausgeben
La0 = 0 : La1 = 1 : La2 = 1 : La3 = 0 : La4 = 0 : La4 = 1 'Latch 7
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 8. Byte-Muster ausgeben
La0 = 1 : La1 = 1 : La2 = 1 : La3 = 0 : La4 = 0 : La4 = 1 'Latch 8
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 9. Byte-Muster ausgeben
La0 = 0 : La1 = 0 : La2 = 0 : La3 = 1 : La4 = 0 : La4 = 1 'Latch 9
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 10. Byte-Muster ausgeben
La0 = 1 : La1 = 0 : La2 = 0 : La3 = 1 : La4 = 0 : La4 = 1 'Latch 10
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 11. Byte-Muster ausgeben
La0 = 0 : La1 = 1 : La2 = 0 : La3 = 1 : La4 = 0 : La4 = 1 'Latch 11
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 12. Byte-Muster ausgeben
La0 = 1 : La1 = 1 : La2 = 0 : La3 = 1 : La4 = 0 : La4 = 1 'Latch 12
Offset = Offset + 1
Porta = Lookup(offset , Tabelle_animationen) ' 13. Byte-Muster ausgeben
La0 = 0 : La1 = 0 : La2 = 1 : La3 = 1 : La4 = 0 : La4 = 1 'Latch 13 + Ebene

Ea4 = 0 'Würfel an
Waitms 1
Ea4 = 1 'Würfel aus

Reset Watchdog 'Watchdog zurücksetzen
Next Ebene
Reset Watchdog 'Watchdog zurücksetzen
Next Dauer
Reset Watchdog 'Watchdog zurücksetzen
Next Bild
Reset Watchdog 'Watchdog zurücksetzen
Next Looping
'Neuen Startpunkt in der Tabelle setzen *****
'berechnet sich wie folgt:
'alter Max_bild-Wert*130 (130 = Bytes pro Bild)
' + 10 Ebenen (0-9) * 13 (13 = Bytes pro Ebene)
' + 13 (plus die letzte Ebene)
' = erste Startposition des neuen Musters
T1 = Max_bild * 130
T2 = 9 * 13
T1 = T1 + T2
T1 = T1 + 13
Startposition = Startposition + T1
*****
Zeigerposition = Zeigerposition + 1 'eine zelle weiter in der Konfigurationstabelle

Next Animation

Reset Watchdog 'Watchdog zurücksetzen
Loop
Reset Watchdog 'Watchdog zurücksetzen

End

'Tabelle Konfiguration *****

Konfiguration:
'Anzahl aller Animationen
Data 10

'Animation 1 - 005 Welle 02
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 28 , 2

'Animation 2 - 002 schräg von unten links füllen und leeren

```

```
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 37 , 3

'Animation 3 - 013 - Welle loop
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 30 , 3

'Animation 4 - 008 Cube
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 1 , 30

'Animation 5 - 012 - von der mitte füllen und leeren
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 11 , 5

'Animation 6 - 011 - von innen nach außen und zurück
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 10 , 5

'Animation 7 - 004 - Spaltenweise von links füllen
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 20 , 3

'Animation 8 - 003 - Zeilenweise von unten füllen und leeren
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 20 , 3

'Animation 9 - 010 - Zeilen von unten nach oben
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 20 , 3

'Animation 10 - 009 - Spalten von links nach rechts
'Bilder in dieser Animation | Wiederholungen dieser Animation
Data 20 , 3

'Tabelle Animationen *****
Tabelle_animationen:                               '013 - Welle loop

'Bild 1
Data &B00000000, &B00000000
Data &B00000000, &B00000001
Data &B00000000, &B00000010
Data &B00000000, &B00000011
Data &B00000000, &B00000100
Data &B00000000, &B00000110
Data &B00000000, &B00000100
Data &B00000000, &B00000110
Data &B11111111, &B11111101
--
--
--
'Bild 28
Data &B00000000, &B00000000
Data &B00000000, &B00000001
Data &B00000000, &B00000010
Data &B00000000, &B00000011
Data &B00000000, &B00000100
Data &B00000000, &B00000110
Data &B00000000, &B00000100
Data &B00000000, &B00000110
Data &B11111111, &B11111101
--
--
--
'Muster: 002 - schräg von unten links füllen und leeren
'Bild 1
Data &B00000000, &B00000000
Data &B00000000, &B00000001
Data &B00000000, &B00000010
Data &B00000000, &B00000011
Data &B00000000, &B00000100
Data &B00000000, &B00000110
Data &B00000000, &B00000100
Data &B00000000, &B00000110
Data &B00100000, &B00010000, &B00001000, &B00000100, &B00000000, &B01000000, &B00100000, &B00000100, &B00000010, &B00000000, &B01000000, &B00100000, &B00000101
--
--
--
```

Also wie gesagt, der Code war ist nur zum Probieren des Würfels und des Musters...und ja ich weis das der µC voll ausgelastet läuft...und das die Muster noch schneller laufen wenn der Teil weggelassen wird wo Informationen zum LC-Display gesendet werden.

Hier mal ein Video: http://www.youtube.com/watch?v=f_tM3zozrC4

Zum Thema Muster von SD-Karte lesen...bin ich noch bei...werde dafür wohl einen ATmega 644 nehmen...zu noch ein MiniDos damit mal eine Textdatei mit Mustern öffnen kann. Aber daran arbeite ich noch...

Links

Alle hier aufgeführten Links haben zur Vollendung meines 10x10x10 LED-Cubes beigetragen!

Webseiten:

<http://zomgtronics.com>

<http://www.mylifeiscomputers.net/2010/09/embedded-systems-the-arduino>

<http://www.instructables.com/id/Led-Cube-8x8x8>

<http://www.led-styles.de>

www.mikrocontroller.net

Videos:

<http://www.youtube.com/watch?v=1rMgoNGIIY0>

<http://www.youtube.com/watch?v=6mXM-oGggrM>

Datenblätter:

UDN2981A

<http://www.alldatasheet.com/datasheet-pdf/pdf/120812/ALLEGRO/UDN2981A.html>

SN74HC573

<http://pdf1.alldatasheet.com/datasheet-pdf/view/27931/TI/SN74HC573A.html>

IRLU024N

<http://pdf1.alldatasheet.com/datasheet-pdf/view/93649/IRF/IRLU024N.html>

74HC238

http://www.nxp.com/documents/data_sheet/74HC_HCT238.pdf

Letztes

Wer Fragen hat oder Anregungen oder Fehler entdeckt hat....bitte mailen an:
ledcube@sungod-ra.de

Wer Rechtschreibfehler findet, darf sie behalten!

Bei Gelegenheit werde ich dieses Dokument etwas ausführlicher nacharbeiten.

Ach, und wer den Würfel mal sehen will....er steht in Berlin...einfach mal mailen!

Grüße und Viel Spaß beim Nachbau!!!!