Mini-Arcade-Machine

Seminararbeit

von Felix Neumann

Matr. Nr. 215892

Ein Bild, das Box, Design, Im Haus, gelb enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abstract

Inhalt

[2 Einleitung 1](#_Toc221266857)

[3 Grundlagen 3](#_Toc221266858)

[3.1 Callback-Funktionen 3](#_Toc221266859)

[3.2 Bresenham-Algorithmus 4](#_Toc221266860)

[3.3 Midpoint-Algorithmus 7](#_Toc221266861)

[3.4 Thin-Film-Transistor Liquid-Crystal Displays 9](#_Toc221266862)

[3.5 Tiefsetzsteller 10](#_Toc221266863)

[3.6 Low-Drop-Spannungsregler 11](#_Toc221266864)

[4 Stand der Technik 12](#_Toc221266865)

[4.1 Mikrokontroller SAM D21 J18A-A 12](#_Toc221266866)

[4.2 Displaycontroller ST7735S 15](#_Toc221266867)

[5 Methodik 17](#_Toc221266868)

[6 Implementierung Hardware 18](#_Toc221266869)

[6.1 Einleitung Hardware 18](#_Toc221266870)

[6.2 Spannungsversorgung 19](#_Toc221266871)

[6.2.1 Spannungsquelle 19](#_Toc221266872)

[6.2.2 Tiefsetzsteller 20](#_Toc221266873)

[6.2.3 Low-Drop-Spannungsregler 21](#_Toc221266874)

[7 Implementierung Software 23](#_Toc221266875)

[7.1 Einleitung Software 23](#_Toc221266876)

[7.2 Displaytreiber 23](#_Toc221266877)

[7.2.1 Initialisierung 23](#_Toc221266878)

[7.2.2 Controller Befehle 25](#_Toc221266879)

[7.2.3 Datenaustausch 25](#_Toc221266880)

[7.2.4 Bilddaten 26](#_Toc221266881)

[7.2.5 Bildwiederholrate 27](#_Toc221266882)

[7.3 Grafik-Library 29](#_Toc221266883)

[8 Literaturverzeichnis 30](#_Toc221266884)

# Einleitung

# Grundlagen

## Callback-Funktionen

Callback Funktionen zu Deutsch Rückruffunktionen bezeichnen in der Informatik Funktionen, die anderen Funktionen als Parameter übergeben werden. Die Callback Funktion wird durch eine Referenz übergeben und unter definierter Bedingung mit definierten Argumenten aufgerufen. [1] In der Programmiersprache C ist diese Referenz ein Funktionszeiger. [2]

Callback Funktionen eignen sich besonders für die ereignisorientierte Programmierung. Dabei wird das Programm nicht rein sequenziell durchlaufen, sondern Ereignisbehandlungsroutinen im englischen Eventhandler, werden ausgeführt, sobald ein bestimmtes Ereignis auftritt. Die ereignisorientierte Programmierung ist vor allem dann sinnvoll, wenn Ereignisse spontan auftreten können, wie z. B. Timer-Abläufe oder das Drücken eines Tasters. Seit 1960 wird dieses Konzept technisch durch Hardware-interrupts und Callback Funktionen realisiert. [3]

Grundsätzlich unterscheidet man zwischen synchronen und asynchronen Rückruf. Bei synchronen Callback wird die Rückruffunktion im selben Ablauf aufgerufen, siehe Abbildung 1. Dadurch wird der Programmablauf blockiert, bis die Callback Funktion vollständig ausgeführt wurde. Bei asynchronem Rückruf ist es möglich, dass die Callback Funktion zu einem späteren Zeitpunkt aus einem anderen Thread aufgerufen wird. [1]

Ein Bild, das Text, Screenshot, Reihe, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung 1 Synchroner Rückruf

## Bresenham-Algorithmus

Der Bresenham-Algorithmus wurde 1962 von dem Programmierer Jack Bresenham zur Steuerung von Digitalplottern entwickelt. Er ist ein Algorithmus aus der Computergrafik zum Zeichnen von Linien auf einer Rasteranzeige. Das Besondere an dem Algorithmus ist das er die Rundungsfehler, die bei der Diskretisierung von kontinuierlichen Koordinaten entstehen, minimiert und dabei ohne Multiplikation, Division oder Gleitkommazahlen auskommt. [4]

Die einfachste Methode eine Linie zwischen einem Startpunkt (xa | ya) und einem Endpunkt (xe | ye) zu rastern ist die direkte Umsetzung der Geradengleichung.

(x – xa) + ya

Gleichung 1 Geradengleichung

Die Linie wird gezeichnet, indem eine Schleife von xa bis xe läuft und für jede X-Koordinate nach Gleichung 1 die entsprechende Y-Koordinate berechnet und auf eine Ganzzahl rundet. Diese Methode ist unnötig langsam und komplex, da sie in jedem Schleifendurchgang eine Multiplikation durchführt. [4]

Ein Bild, das Reihe, Diagramm, Zahl, Text enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung 2 Naive Linien-Rasterung

Die grundlegende Idee des Bresenham Algorithmus ist es, in jedem Schritt zu prüfen, ob sich für den nächsten Pixel die Y-Koordinate verändert. Dazu wird der Mittelpunkt zwischen den beiden möglichen nächsten Pixelpositionen mit der idealen Geraden verglichen, siehe Abbildung 3. Befindet sich der Mittelpunkt über der idealen Linie, kann die aktuelle Y-Koordinate beibehalten werden, liegt der Mittelpunkt unter der Linie, muss die Y-Koordinate inkrementiert werden. [4]

Ein Bild, das Reihe, Diagramm, Schrift, Screenshot enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung 3 Wahl des nächsten Pixels

Um zu prüfen, ob der Mittelpunkt oberhalb oder unterhalb der idealen Geraden liegt, wird die Geradengleichung in die folgende Form gebracht, siehe Gleichung 2. Dabei ergibt die Gleichung Null für Punkte, die exakt auf der Linie liegen, negative Werte für Punkte oberhalb der Linie und positive Werte für Punkte unterhalb der Linie.

Gleichung 2

Setzt man die Koordinaten des Mittelpunktes in Gleichung 2 ein und weißt den Wert einer Kontrollvariable di zu, so erhält man Gleichung 3. Anhand des Vorzeichens der Kontrollvariable di lässt sich dann entscheiden, ob für den nächsten Pixel die Y-Koordinate inkrementiert werden soll oder nicht.

Gleichung 3

Die Kontrollvariable di wird im Bresenham-Algorithmus inkrementell berechnet. Die Änderung der Kontrollvariable zwischen zwei aufeinanderfolgenden Schritten hängt davon ab, ob die Y-Koordinate inkrementiert wurde oder nicht. Für die beiden Fälle ergeben sich die folgenden Differenzen für die Kontrollvariable, siehe Gleichung 4 und   
Gleichung 5.

Gleichung 4 Differenz ohne inkrementierte Y-Koordinate

Gleichung 5 Differenz bei inkrementierte Y-Koordinate

Der Anfangswert der Kontrollvariable lässt sich nach dem gleichen Vorgehen berechnen. Dabei ergibt der Term Null, da der Anfangspunkt (xa, ya) genau auf der Linie liegt.

Gleichung 6 Anfangswert der Kontrollvariable

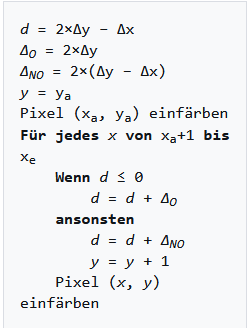
Um die Division durch Zwei im Anfangswert der Kontrollvariable zu vermeiden, werden alle Gleichungen mit Faktor Zwei erweitert. Das ändert nichts an der Funktion des Algorithmus, da das Vorzeichen der Kontrollvariable erhalten bleibt. Für eine Linie mit einer Steigung zwischen 0 und 1 lässt sich der Bresenham-Algorithmus mit dem folgenden Pseudo-Code ausdrücken, siehe Abbildung 4. [4]

Abbildung 4 Pseudocode Bresenham-Algorithmus

Der hier erklärte Algorithmus beschränkt sich auf die Rasterung von Linien mit einer Steigung zwischen 0 und 1, was einem Winkel von 0° bis 45° zur Horizontalen entspricht. Der Algorithmus lässt sich durch Nutzung von Symmetrieeigenschaften auch auf andere Linien anpassen. [4]

## Midpoint-Algorithmus

Der Midpoint-Algorithmus ist eine Variante des Bresenham-Algorithmus zur Rasterung von Kreisen. [5] Die Grundlage für den Algorithmus ist die implizite Kreisgleichung, siehe Gleichung 7. Setzt man die Koordinaten eines Punktes in Gleichung 7, ergibt der Funktionswert Null, wenn der Punkt auf dem Kreis liegt, einen negativen Wert, wenn der Punkt innerhalb des Kreises liegt und einen positiven, wenn der Punkt außerhalb des Kreises liegt.

Gleichung 7 Implizite Kreisgleichung

Ein Bild, das Reihe, Diagramm, Kreis, parallel enthält.

KI-generierte Inhalte können fehlerhaft sein.Der Midpoint-Algorithmus startet bei der größten Y-Koordinate. Wie beim Bresenham-Algorithmus wird in jedem Schritt zwischen den nächsten beiden möglichen Pixelpositionen entschieden, dazu wird der Mittelpunkt zwischen den beiden Pixelpositionen betrachtet, siehe Abbildung 5.

Abbildung 5 Wahl des Pixels

Die Koordinaten des Mittelpunktes werden in Gleichung 7 eingesetzt und einer Kontrollvariable di zugewiesen. Ist die Kontrollvariable positiv wird die Y-Koordinate dekrementiert, ansonsten wird die aktuelle Y-Koordinate beibehalten. Die Kontrollvariable wird, wie beim Bresenham-Algorithmus inkrementell berechnet. Die Änderung der Kontrollvariable hängt dabei davon ab, ob die Y-Koordinate dekrementiert wurde oder nicht, siehe Gleichung 8 und Gleichung 9.

Gleichung 8 Differenz bei gleicher Y-Koordinate

Gleichung 9 Differenz bei dekrementierter Y-Koordinate

Der Anfangswert der Kontrollvariable beträgt , der Bruch lässt sich vermeiden, indem man von der Kontrollvariabel abzieht. Dadurch ergibt sich für den Anfangswert und der Vergleich für die Wahl des nächsten Pixels ändert sich  
 zu . Die Änderung des Vergleichs kann vernachlässigt werden und zu gerundet werden. [6] Mit dem eben beschriebenen Vorgehen wird nur ein Achtelkreis berechnet. Der Rest des Kreises wird einfach durch eine Spiegelung auf die restlichen Oktanten gezeichnet, siehe Abbildung 6.

Ein Bild, das Reihe, Diagramm, Kreis, Text enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung 6 Spiegelung eines berechneten Pixels

## Thin-Film-Transistor Liquid-Crystal Displays

Thin-Film-Transistor Liquid-Crystal Displays kurz TFT-LCDs, sind Flüssigkeitskristallbildschirme, die über Dünnschichttransistoren angesteuert werden. [7] Flüssigkeitskristalle haben die Eigenschaft, dass sie ihre Durchlässigkeit für polarisiertes Licht ändern, wenn eine elektrische Spannung an ihnen angelegt wird. Das polarisierte Licht wird mit Polarisationsfiltern erzeugt, welche das einfallende Umgebungslicht oder eine Hintergrundbeleuchtung filtern. Damit das Display beliebige Inhalte anzeigen kann werden die Segmente in gleichmäßigen Rastern angeordnet. [8]

Jeder Pixel eines Farbdisplays besteht aus drei Flüssigkeitskristallen. Um die Anzahl an Anschlüssen in einem Display zu reduzieren, werden die Flüssigkeitskristalle in Reihen und Spalten angeordnet, siehe Abbildung 7. Jeder Flüssigkeitskristall wird durch einen Transistor angesteuert. In einer Reihe oder Spalte sind die Anschlüsse der Transistoren elektrisch miteinander verbunden. Da der Transistor Strom nur in eine Richtung durchlässt, wird verhindert, dass sich die Spannung an den Flüssigkeitskristallen entlädt, bevor das Display wieder aktualisiert wird. Nochmal überarbeiten ggf.

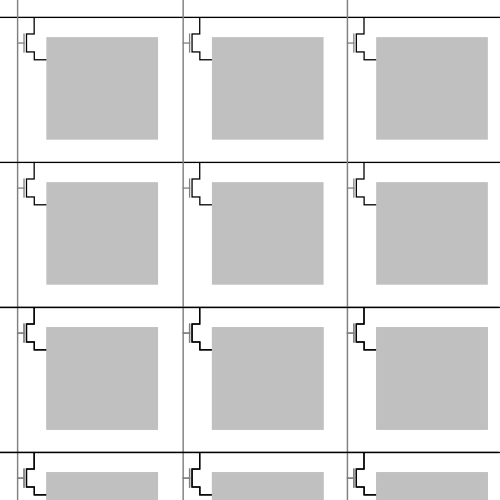


Abbildung 7 Pixel Anordnung in einem LCD

## Tiefsetzsteller

Ein Tiefsetzsteller, im Englischen Step-Down Converter genannt, ist ein geschalteter  
Gleichspannungswandler der die Eingangsspannung UE in eine niedrigere Ausgangsspannung UA wandelt.

In Abbildung 7 ist der grundlegende Aufbau eines Tiefsetzstellers zu sehen. Der Schalter S wird meist durch einen Transistor realisiert, der einige Hundert, bis mehrere Millionen Mal in der Sekunde schaltet. Die Höhe der Ausgangsspannung kann über die Einschaltzeit und Ausschaltzeit des Transistors eingestellt werden. Durch einen geschlossenen Regelkreis wird die Ausgangsspannung auf einem gewünschten Wert gehalten.

Die Spule L und der Kondensator C versorgen während der Ausschaltzeit TE den Verbraucher mit Energie. Während der Einschaltzeit TE sperrt, die Diode D und der Strom fließt über die Spule L. In der Ausschaltphase TA wird die im Magnetfeld der Spule gespeicherte Energie abgebaut. Der Laststrom fließt über die Spule zum Verbraucher und über die Diode zurück zur Spule. Durch den Kondensator wird die Ausgangsspannung UA gestützt.

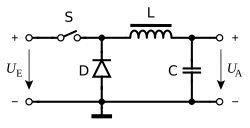


Abbildung 8 Schaltungsschema Tiefsetzsteller

Beim Tiefsetzsteller unterscheidet man grundsätzlich zwischen zwei Betriebsarten. Beim nicht lückenden Betrieb, im Englischen Continuous Current Mode, fließt jederzeit Strom durch die Spule. Die Einschaltzeit beginnt wieder, bevor die Energie im Magnetfeld der Spule vollständig abgebaut wurde. Im lückenden Betrieb, im Englischen   
Discontinuous Current Mode, gibt es eine dritte Phase, in der kein Strom durch die Spule fließt und die angeschlossene Last ausschließlich vom Kondensator gestützt wird.

Beim Tiefsetzsteller treten Verluste in der Spule, dem Transistor und der Freilaufdiode auf. In der Spule treten ohmsche Verluste durch den Wicklungswiderstand sowie magnetische Verluste auf. Der Transistor hat im eingeschalteten Zustand einen Spannungsabfall sowie Schaltverluste. An der Diode fallen je nach Bauart 0,4 bis 1 Volt ab. Die Verluste der Diode lassen sich reduzieren, indem man sie durch einen Transistor ersetzt. Wird die Diode durch einen Transistor mit der zugehörigen Steuerlogik ersetzt, erhält man einen Synchronwandler. [9]

## Low-Drop-Spannungsregler

Ein Low-Drop Spannungsregler, abgekürzt LDO (Low Dropout Regulator), ist ein linearer Spannungsregler, der Eingangsspannungen regeln kann, die sehr nah an der Ausgangsspannung liegen. Die Vorteile gegenüber geschalteten Gleichspannungsreglern sind ein simpleres Design, eine kleinere Bauform und keine zusätzlichen Störsignale durch die hohen Schaltfrequenzen. Der große Nachteil ist, dass bei Linearen Spannungsreglern viel Abwärme entsteht. LDOs bestehen grundlegend aus einem   
Leistungs-Feldeffekttransistor und einem Differenzverstärker, siehe Abbildung 8.

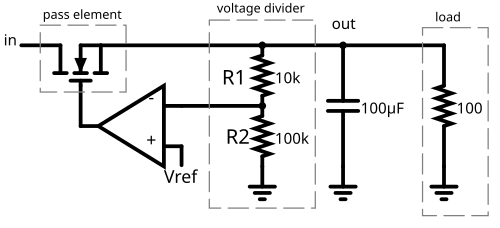


Abbildung 9 Aufbau eines LDOs

Am invertierenden Eingang des Operationsverstärkers liegt ein Teil der Ausgangsspannung an, die über den Spannungsteiler R1, R2 eingestellt wird,   
siehe Gleichung 10.

Gleichung 10 Spannung am invertierenden Eingang

Am nichtinvertierenden Eingang liegt eine feste Referenzspannung Vref an. Der Ausgang des Operationsverstärkers ist mit dem Gate-Anschluss des Feldeffekttransistors verbunden. Steigt die Ausgangsspannung zu hoch und damit auch die Teilspannung am invertierenden Eingang, verändert der Operationsverstärker seine Ausgangsspannung Vout, um die Teilspannung V- an die Referenzspannung Vref anzugleichen. Für die Spannung am Gate-Anschluss des Transistors gilt Gleichung 11.

Gleichung 11 Spannung am Gate-Anschluss des Transistors

Die minimal Spannungsdifferenz zwischen Eingangs- und Ausgangsspannung des LDOs wird als Dropout Spannung VDO bezeichnet. Sie ist notwendig damit der LDO die Ausgangsspannung regulieren kann. Die minimal Spannungsdifferenz ist dabei durch die Sättigungsspannung des Feldeffekttransistors begrenzt. [10]

# Stand der Technik

## Mikrokontroller SAM D21 J18A-A

Der SAM D21 J18A-A ist ein Mikrokontroller von dem amerikanischen Halbleiterhersteller Microchip Technology mit einem 32-Bit ARM-Cortex-M0+ Prozessor, 64 konfigurierbaren Pins, einem 256 KB Flash Speicher und 32 KB flüchtigem Speicher. Der SAM D21 arbeitet mit einer maximalen Taktrate von 48 MHz und erreicht laut Hersteller eine Leistung von bis zu 2,46 . Der SAM D21 bietet eine Vielzahl an Funktionen, verschiedene serielle Schnittstellen, Timer-Counter und programmierbare I/O Pins. Im Folgenden werden die Funktionen des SAM D21, welche für die Umsetzung der Projektarbeit besonders relevant waren, genauer beschrieben. [11, S. 13]

Ein Bild, das Text, Screenshot, Diagramm, Plan enthält.

KI-generierte Inhalte können fehlerhaft sein.Der I/O Pin Controller kontrolliert die I/O Pins des SAM D21. Die I/O Pins sind in   
PORT-Gruppen mit bis zu 32 Pins aufgeteilt und können einzeln oder in der Gruppe angesteuert werden. Jeder Pin kann direkt als Ein- oder Ausgang genutzt werden oder einem Peripheriemodul zugewiesen werden. Die Peripheriemodule werden den Pins durch softwaregesteuertes Multiplexen zugewiesen. [11, S. 363]

Abbildung 10 Blockdiagramm einer PORT-Gruppe

Jeder Pin in einer PORT-Gruppe wird einem Bit im Data Direction (DIR) und Data Output (OUT) Register zugewiesen. Im Data Direction Register wird konfiguriert, ob ein Pin als Ein- oder Ausgang genutzt wird. Ist das entsprechende Bit auf 1 gesetzte wird der Pin als Ausgang genutzt. Der Ausgangpegel des Pins wird durch das Data Output Register festgelegt. Wird das entsprechende Bit auf 1 gesetzt, ist der Ausgangspegel des Pins Logisch-High. [11, S. 367] Über das Pin Configuration Register (PINCFG) lassen sich zusätzliche Pin-Konfigurationen einstellen. Wird das PULLEN Bit auf 1 gesetzt, lässt sich ein Eingang über das Data Output Register als Pull-Up oder Pull-Down konfigurieren. [11, S. 368]

Die seriellen Kommunikations-Schnittstellen (SERCOM) des SAMD21 können als I2C, SPI oder USART Schnittstelle konfiguriert werden. Die SERCOM-Schnittstelle besteht aus einem Sender und Empfänger, einem Baudratengenerator und einer Adressabgleichfunktion. Sie kann einen internen oder externen Taktgeber verwenden.   
[11, S. 408]

Ein Bild, das Text, Screenshot, Diagramm, Reihe enthält.

KI-generierte Inhalte können fehlerhaft sein.Serial Peripheral Interface, kurz SPI, ist eine der verfügbaren seriellen Schnittstellen. Die SPI-Schnittstelle kann als Master oder Slave betrieben werden. Beide Modi bestehen aus einem Schieberegister, einem Sendepuffer und einem zweistufigen Empfangspuffer. Zusätzlich dazu nutzt die Schnittstelle im Mastermode noch den Baudraten-Generator für die Clock. Im Slave-Modus wird noch zusätzlich der Adressabgleich genutzt. In Abbildung 11 ist die Kommunikation zwischen Master und Slave als Blockdiagramm dargestellt. [11, S. 448]

Abbildung 11 Blockdiagramm Kommunikation zwischen SPI Master und Slave

Die SPI-Schnittstelle kann in den Modi null bis drei betrieben werden. Die vier Modi ergeben sich aus einer Kombination der beiden Parameter Clock-Polarity und Clock-Phase. Die beiden Parameter lassen sich im CTRLA-Register über die Bits CPOL und CPHA konfigurieren. CPOL definiert dabei den Ruhezustand des Taktes, Logisch-High oder Logisch-Low. Über CPHA wird eingestellt, ob die Daten bei der führenden oder hinteren Flanke des Taktes gesampelt werden. Tabelle 1 gibt einen Überblick über die vier SPI-Modi. [11, S. 452]

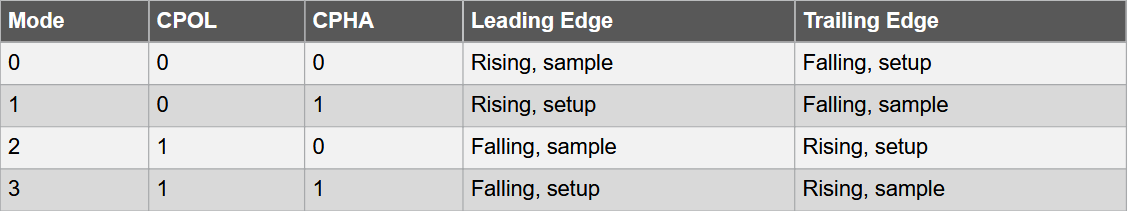


Tabelle 1 SPI-Modi

Die Timer/Counter for Control Applications Peripheire, kurz TCC, ist eine Timer-Komponente des SAMD21 für anspruchsvolle Steueranwendungen wie Pulsweitenmodulation oder Frequenz-Generierung. Der SAMD21 verfügt über drei TCC-Instanzen, die unabhängig voneinander arbeiten. Jede Instanz besteht aus einem Zähler, einem Prescaler und einem Compare/Capture Kanal, kurz CC. [11, S. 616]

In Abbildung 12 ist die allgemeine Funktionsweise des TCC als Blockdiagramm dargestellt. Jede Instanz verfügt über ein Zähler-Register, in dem der aktuelle Zählwert gespeichert wird, ein gepuffertes Register für die Periode und jeweils ein gepuffertes Register für die vier CC-Kanäle. Im normalen Betrieb wird kontinuierlich geprüft, ob der Zählwert den TOP- oder ZERO-Wert erreicht hat. In beiden Fällen kann der TCC Interrupts auslösen, DMA-Transaktionen ausführen oder Events für das Event System generieren. Die CC-Kanäle werden kontinuierlich mit dem Zählwert verglichen, um daraus Wellenformen zu generieren. Die generierten Signale werden einem Waveform-Generation-Output, kurz WO, zugewiesen. Der WO wird dann über den I/O Pin Controller einem physischen Pin zugewiesen. [11, S. 619]

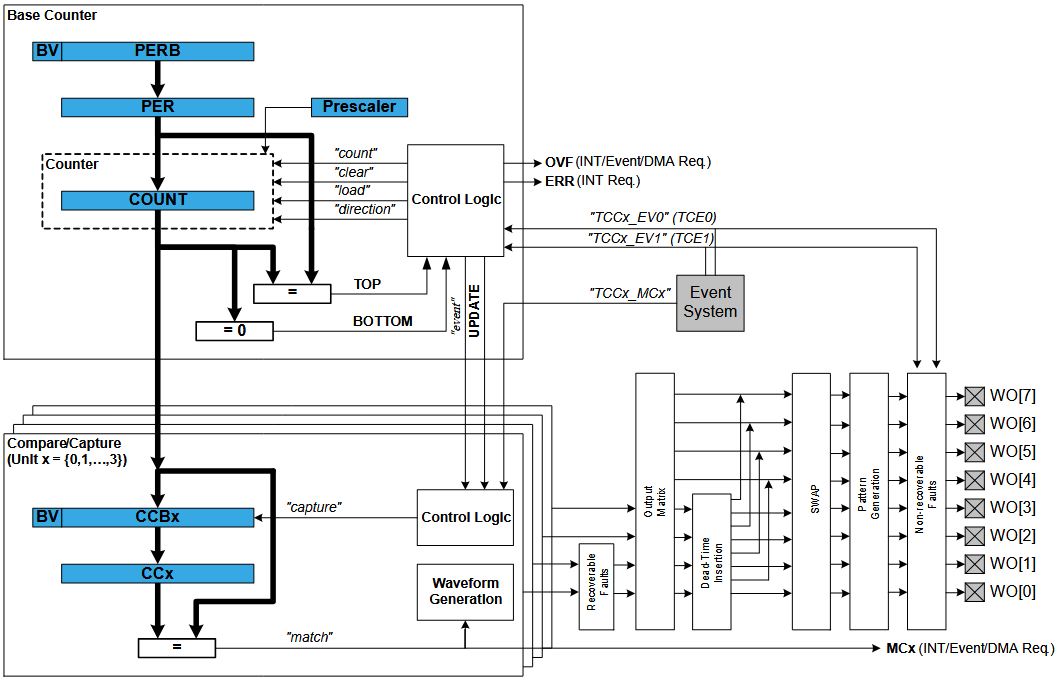


Abbildung 12 Block Diagramm der Timer/Counter Control Application

Der Normal-Frequency-Mode, kurz NFRQ, generiert ein Signal mit beliebiger Frequenz. Die Periodendauer des Signals wird über den Wert im Perioden-Register festgelegt. Die Wellenform selbst wird durch den CC-Kanal generiert. Jedes Mal, wenn der Zählwert den Wert im CC-Register erreicht, wechselt das Ausgangssignal zwischen Logisch-High und Logisch-Low. In Abbildung 13 ist der Signalfluss im NFRQ-Mode dargestellt. [11, S. 625]

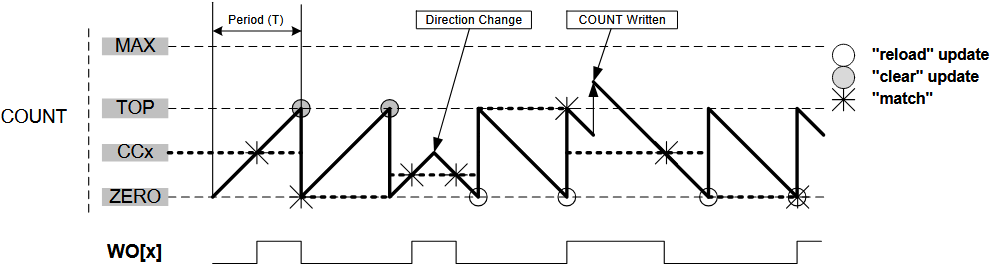


Abbildung 13 Normal Frequency Mode (NFRQ)

## Displaycontroller ST7735S

Der ST7735S ist ein Displaycontroller für TFT-LCDs von der Firma Sitronix. Der Chip kann direkt an einen Mikrokontroller angeschlossen werden und über eine SPI oder parallele Schnittstelle angesteuert werden. Der Chip verfügt über einen RAM der Bilddaten bis zu einer Auflösung von speichern kann. [12, S. 10] Im Folgenden werden, die für die Umsetzung des Projekts wichtigen Funktionen genauer beschrieben.

In Abbildung 14 ist das Timing für die serielle Schnittstelle des ST7735S zu sehen. Um die Datenübertragung zu beginnen, muss die Chip Select Leitung auf Logisch-Low gezogen werden. Die Chip Select Leitung bleibt auf Low, bis die Datenübertragung vollständig abgeschlossen ist. Der Ruhezustand der Serial Clock ist Logisch-High und die Daten werden bei einer steigenden Flanke gesampelt, das entspricht SPI-Modus 0. [12, S. 45] Die Serial Clock kann dabei mit einer Frequenz von bis zu 14 MHz getaktet werden. [12, S. 36] Der Datenaustausch zwischen Master und Slave läuft über die SDA-Leitung in beide Richtungen. Die D/CX Leitung legt fest, ob Daten oder ein Befehl gesendet werden, dabei wird Logisch-Low als Befehl interpretiert und umgekehrt. [12, S. 44]

Ein Bild, das Text, Diagramm, Reihe, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung 14 Timing der SPI-Schnittstelle des ST7735S

Die Bilddaten werden im RGB-Format an den Display-Controller gesendet, dabei kann zwischen drei möglichen Farbtiefen gewählt werden, 4K, 65K und 262K. In Abbildung 15 ist die serielle Schnittstelle für eine Farbtiefe von 65K dargestellt.

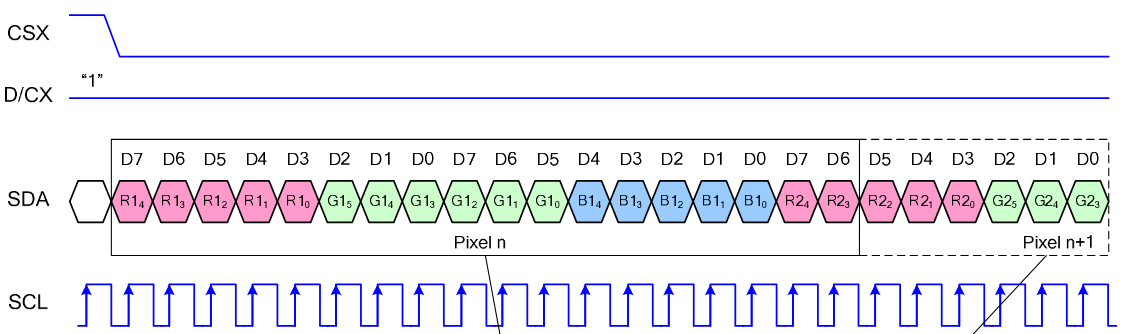


Abbildung 15 Bilddaten im RGB 5-6-5 Format

Ein Bild, das Text, Diagramm, Screenshot, Reihe enthält.

KI-generierte Inhalte können fehlerhaft sein.Die gesendeten Bilddaten werden pixelweise, entsprechend der ausgewählten Bildtiefe, in den integrierten RAM geschrieben. Der Adressen-Pointer legt dabei fest, an welche Stelle im RAM die gesendeten Bildpunkte geschrieben werden. Die Adressen gehen in X-Richtung von 0 bis 131 und in Y-Richtung von 0 bis 161. Bevor der RAM beschrieben werden kann, muss ein Fenster definiert werden, welches beschrieben werden soll. [12, S. 74] Das Fenster wird über die beiden Befehle CASET und RASET definiert. [12, S. 128–131] Die Scanrichtung wird über den Befehl MADCTL eingestellt.

Abbildung 16 Struktur des Display RAM

# Methodik

# Implementierung Hardware

## Einleitung Hardware

Das Ziel der Hardware-Entwicklung ist ein Lötbausatz für Schüler in Form eines voll funktionsfähigen Arcade-Automaten. Die Mini-Arcade-Machine soll ein modernes RGB-Display besitzen, sowie eine Soundausgabe und einen Controller zur Steuerung. Die Spannungsversorgung soll netzunabhängig sein und darf kein Sicherheitsrisiko für die Schüler darstellen.

Die Mini-Arcade-Machine besteht aus einer separaten Platine für die Spannungsversorgung, einer Platine für den Controller sowie einem Mainboard. Zusätzlich zu den drei eben genannten gibt es weitere Platinen, die als Verbindungselemente dienen. Die gesamte Hardware wurde mit dem EDA-Tool KiCad konstruiert. Zusätzlich zu KiCad wurde Autodesk Inventor genutzt, um die einzelnen Platinen in einer Baugruppe darzustellen. In Abbildung 17 ist das 3D-Modell der Mini-Arcade-Machine zu sehen. Im Folgenden werden die einzelnen Komponenten im Detail erklärt.



Abbildung 17 3D-Modell Mini-Arcade-Machine

## Spannungsversorgung

Die Aufgabe der Spannungsversorgung ist es, regulierte 3,3 V Gleichstrom für den SAM D21 und das Display bereit zu stellen. Um die Störeinflüsse der Leistungselektronik auf die restliche Elektronik zu minimieren, wurde die gesamte Spannungsversorgung auf eine separate Platine ausgelagert.

### Spannungsquelle

Da das Ziel der Entwicklung ein Lötbausatz für Schüler ist, soll die Spannungsquelle eine Batterie sein. Einerseits aus Sicherheitsgründen, andererseits wegen der Benutzerfreundlichkeit. Um eine geeignete Batterie auszuwählen, wurde anhand der Datenblätter der verwendeten Komponenten ein Richtwert für den Betriebsstrom ermittelt. Insgesamt ist ein Betriebsstrom von ca. 132 mA anzunehmen. Nach ausgiebiger Recherche fiel die Auswahl auf AAA-Zellen. Microzellen sind ein guter Kompromiss aus Baugröße und Kapazität. Alkaline AAA-Zellen haben eine typische Kapazität von   
1200 mAh. [13] Damit ist bei dem ermittelten Betriebsstrom ein Betrieb von mehreren Stunden gewährleistet. Um ein Spannungsniveau über 3,3 V zu erreichen, werden drei AAA-Zellen in Reihe geschaltet. Durch die Reihenschaltung ergibt sich eine Nennspannung von 4,5 V.

Da es sich bei dem entwickelten Bausatz um einen ersten Prototyp handelt, wurde zusätzlich zu den AAA-Zellen eine USB-C Schnittstelle als Spannungsquelle für den Testbetrieb vorgesehen. Um zwischen den beiden Spannungsquellen zu wechseln, wird eine Brücke an einer dreipoligen Stiftleiste umgesteckt, siehe   
Abbildung 17.

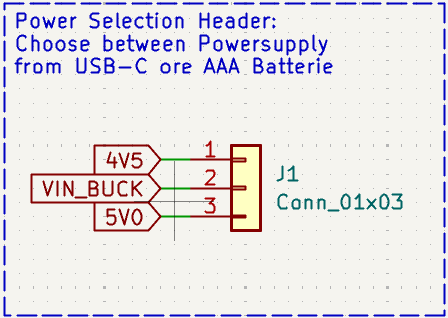


Abbildung 18 Stiftleiste zur Auswahl der Spannungsquelle

Um die 4,5 V der Microzellen und die 5 V der USB-C Schnittstelle in 3,3 V zu wandeln, werden ein Tiefsetzsteller und ein linearer Spannungsregler in Reihe geschaltet. Der Tiefsetzsteller wandelt die Eingangsspannung in eine Ausgangsspannung von 3,5 V. Der Spannungsregler reguliert die Eingangsspannung auf 3,3 V.

### Tiefsetzsteller

Als Tiefsetzsteller wurde der IC AP61100Z6-7 von dem Hersteller Diodes Incorporate ausgewählt. Die Hauptkriterien für die Auswahl des Tiefsetzstellers waren der 1 Ampere zulässiger Ausgangsstrom, der breite Eingangsspannungsbereich von 2,3 V bis 5,5 V sowie die ausführliche Dokumentation des Herstellers. [14, S. 1] In Abbildung 18 ist der Schaltplan des Tiefsetzstellers zu sehen.

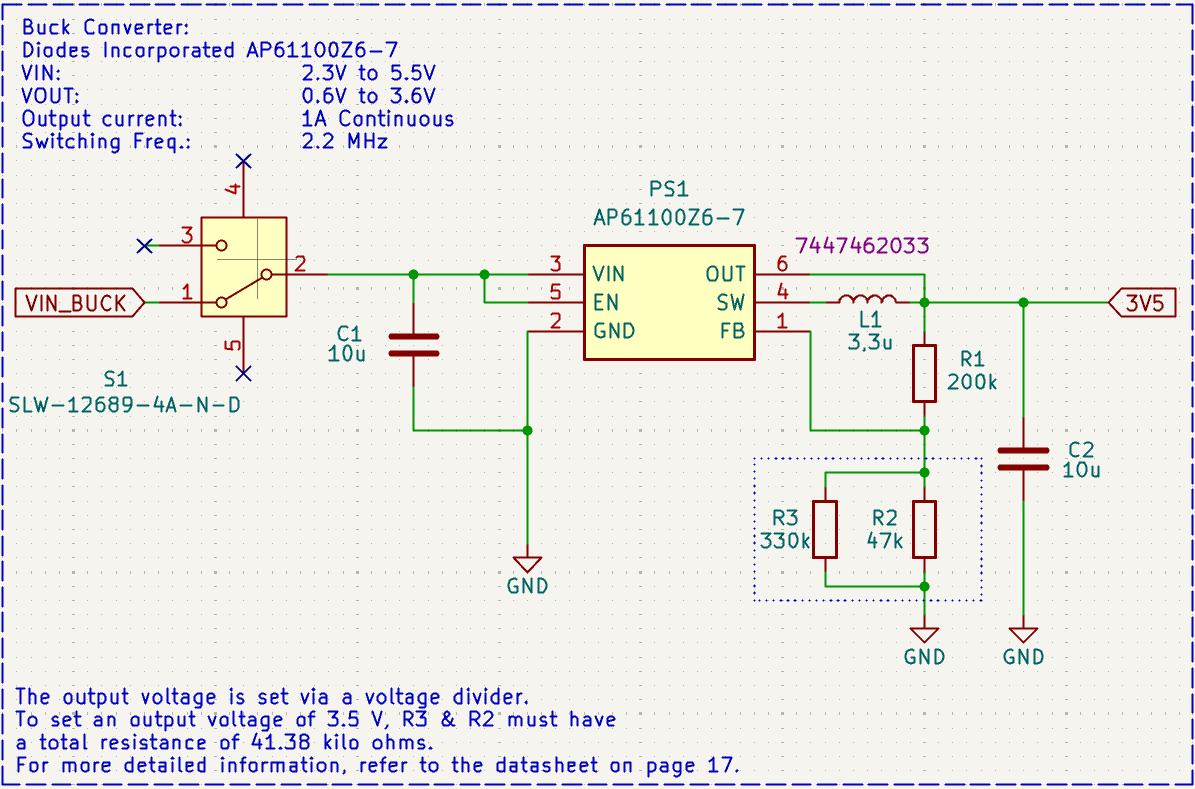


Abbildung 19 Schaltplan des Tiefsetzstellers

Zwischen der Stiftleiste für die Auswahl der Spannungsquelle und dem Tiefsetzteller ist der Hauptschalter S1. Mit dem Hauptschalter lässt sich die gesamte Versorgungsspannung abschalten. Der Tiefsetzsteller hat eine variable Ausgangsspannung. Über den Spannungsteiler R2 und R3 wird die Ausgangsspannung auf 3,5 V eingestellt. Die Induktivität L1 wurde nach der Gleichung 12 dimensioniert. [14, S. 17] Aus der Berechnung ergibt sich für die Induktivität ein Wert von 3,3 µH.

Gleichung 12

In Abbildung 19 ist das Routing des Tiefsetzstellers zu sehen. Beim Routing müssen ausreichend große Kupferflächen für die Wärmeabfuhr berücksichtigt werden.   
[14, S. 19] Die Herausforderung dabei ist es, einen Kompromiss zwischen einer guten Wärmeabfuhr und der Lötbarkeit des Bauteils zu finden.

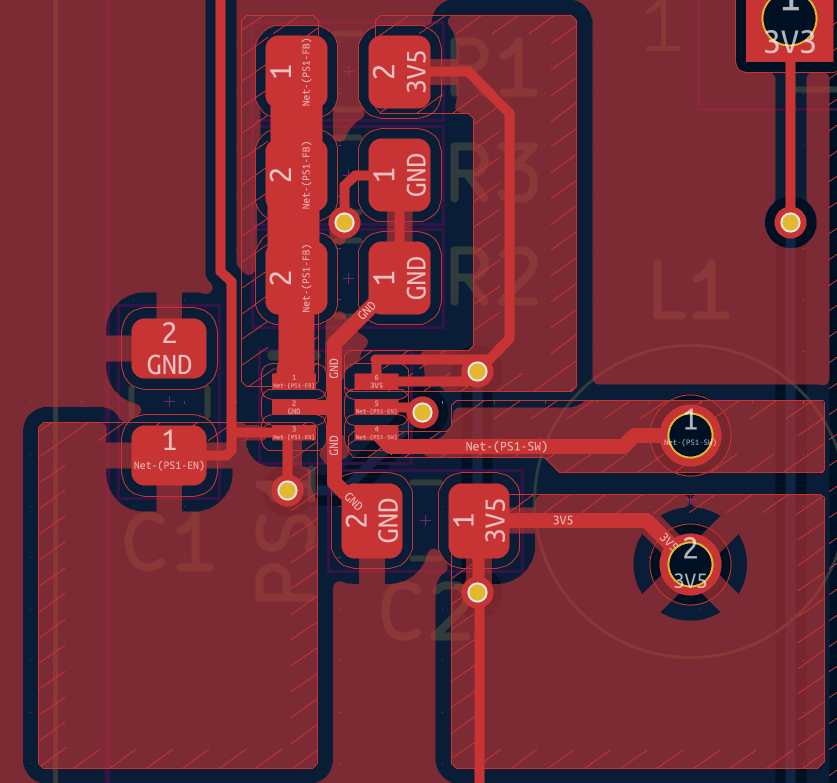


Abbildung 20 Routing des Tiefsetzstellers

### Low-Drop-Spannungsregler

Ein Bild, das Text, Screenshot, Reihe, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.Für den Spannungsregler wurde der MIC94310 von Microchip Technology ausgewählt. Der LDO regelt die 3,5 V Eingangsspannung vom Tiefsetzsteller auf 3,3 V. In Abbildung 20 ist der Schaltplan des Spannungsreglers zu sehen. Der Spannungsregler hat eine feste Ausgangsspannung von 3,3V und einen zulässigen Ausgangsstrom von 200 mA.   
[15, S. 1]

Abbildung 21 Schaltplan Low-Drop-Spannungsregler

## Controller

Der Controller besteht aus einem Joystick und einem Directional-Pad. In Abbildung 21 ist das 3D-Modell des Controllers zu sehen. Der Joystick kann in der Platinen-Ebene nach oben, unten, rechts und links geschaltet werden. Zusätzlich dazu besitzt der Joystick einen fünften Kontakt in Z-Richtung. Das Directional-Pad besteht aus vier Schaltern, die unabhängig voneinander geschaltet werden.

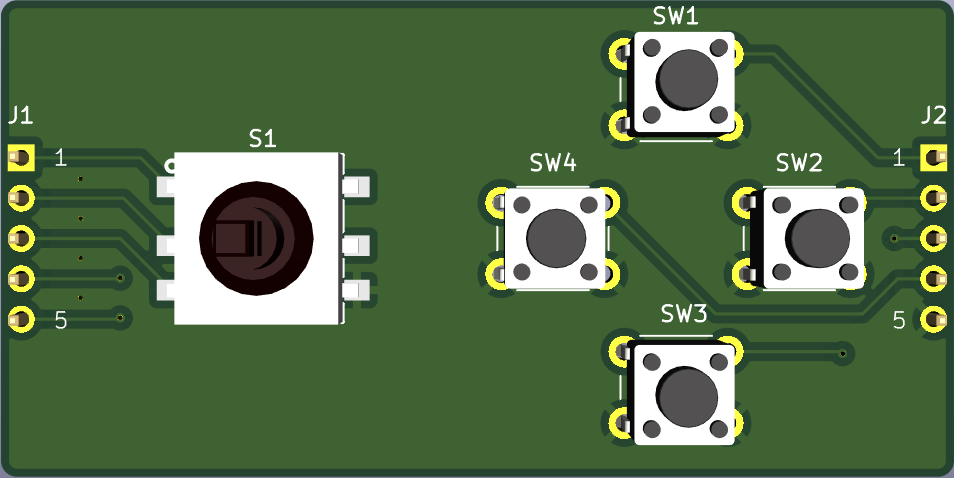


Abbildung 22 3D-Modell des Controllers

Alle Kontakte des Joysticks und des Directional-Pads werden gegen Masse geschaltet. Über die beiden Stiftleisten J1 und J2 sind die Kontakte mit den GPIO-Pins des SAM D21 verbunden. Die GPIO-Pins sind als Pull-Up konfiguriert und in der Software über den gleichen Namen ansteuerbar wie der Bezeichner im Schaltplan, siehe Abbildung 22. Das bedeutet, wenn am Pin des SAM D21 ein Logische-Low Signal anliegt, wurde der entsprechende Controller-Kontakt geschaltet.

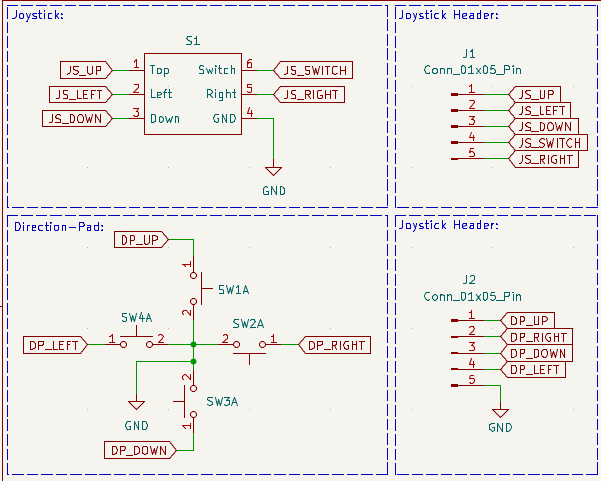


Abbildung 23 Schaltplan des Controllers

## Mainboard

Das Mainboard ist das Herzstück der Mini-Arcade-Machine. Es besteht aus dem Mikrokontroller SAM D21, dem Anschluss für das Display, dem Buzzer zur Soundausgabe und einer Debug-Schnittstelle. In Abbildung 23 ist das 3D-Modell des Mainboards zu sehen.

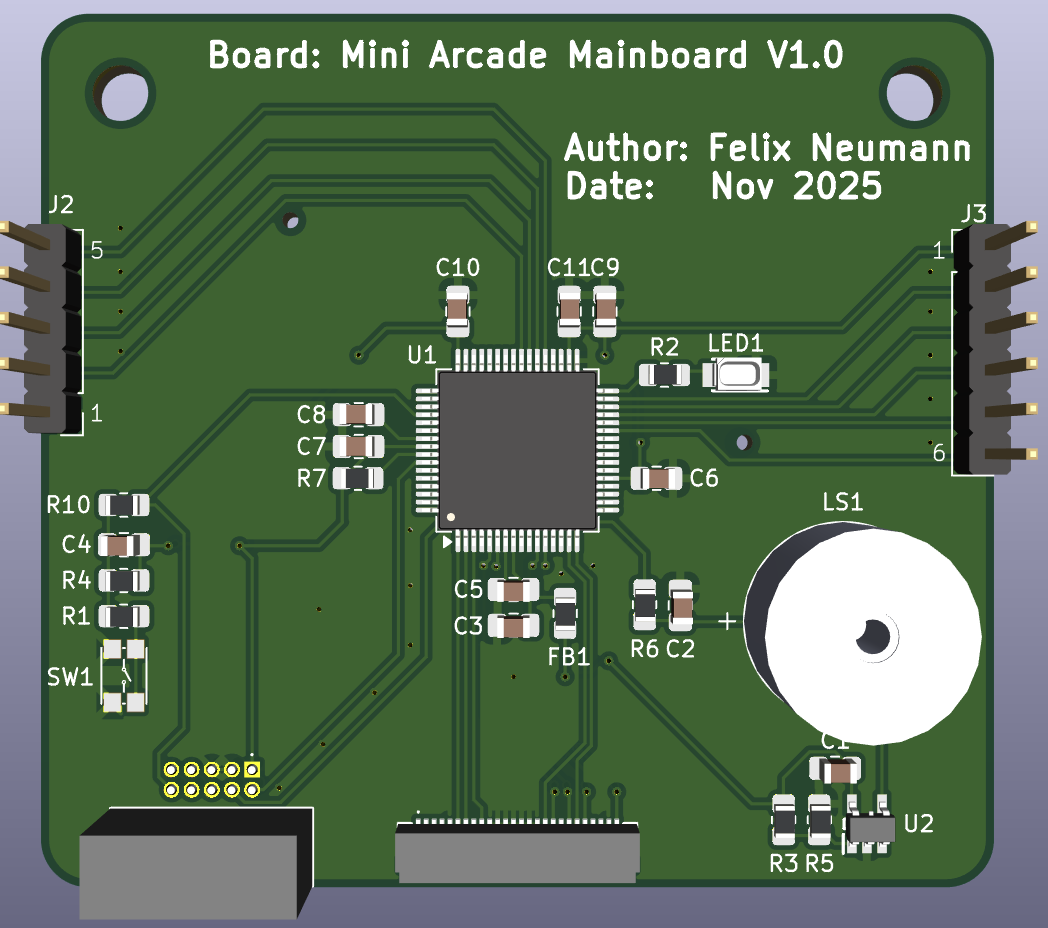


Abbildung 24 3D-Modell des Mainboards

### Mikrokontroller SAM D21

Der SAM D21 ist die zentrale Peripherie der Mini-Arcade-Machine. Über den Mikrokontroller läuft die gesamte Datenverarbeitung. Die Generierung der Bild- und Audiodaten, der Programmablauf und die Auswertung der Controller-Signale. Für ein simples Routing wurde der SAM D21 im Zentrum der Platine platziert. Die verschiedenen Funktionsgruppen sind entsprechend ihrer zugehörigen Pins um den Mikrokontroller angeordnet.

In Abbildung 24 ist der Schaltplan der Spannungsversorgung des SAM D21 zu sehen. Der SAM D21 hat drei separate Versorgungsspannungen. Eine Versorgungsspannung für die analogen Peripherien, eine für die digitalen Peripherien und eine interne Versorgungsspannung für den Kern des SAM D21. Um die analoge Versorgungsspannung „VDDANA“ zu entstören, wird ein Ferrit-Bead in Reihe geschaltet. Die Masse der analogen Versorgungsspannung „GNDANA“ wird durch eine Net Tie entkoppelt.

Für einen fehlerfreien Betrieb des SAM D21 benötigen die drei Versorgungsspannungen mehrere Stützkondensatoren. Die Stützkondensatoren müssen möglichst nah an den Versorgungspins des SAM D21 platziert werden. Im Schaltplan in Abbildung 24 sind alle Kondensatoren, die nah am Mikrokontroller platziert werden müssen, rechts von der gestrichelten Linie dargestellt. [12, S. 112–113]

Um den Mikrokontroller manuell zurückzusetzen, befindet sich auf der Platine ein Reset-Taster. In Abbildung 25 ist der Schaltplan des Reset-Tasters dargestellt. Bei einem Logisch-Low Signal am Reset-Pin wird der SAM D21 zurückgesetzt. Der Pull-Up-Widerstand R4 hält den Reset-Pin auf Logisch-High damit der Mikrokontroller nicht aus Versehen zurückgesetzt wird. Der Kondensator C4 mindert die Störanfälligkeit des Reset-Schaltkreises. Der Widerstand R1 verhindert einen Kurzschluss, wenn der Reset-Taster gedrückt wird. [11, S. 114–115] Widerstand R10 entkoppelt den Reset-Taster von der Debug-Schnittstelle, welche auch den Reset-Pin des SAM D21 nutzt.

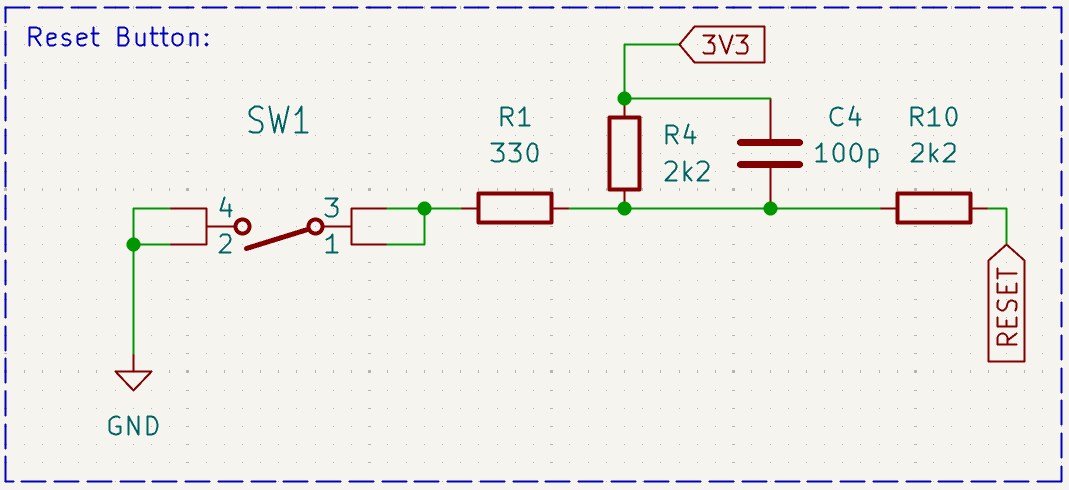


Abbildung 25 Schaltplan des Reset-Tasters



Abbildung 26 Schaltplan der Versorgungsspannung des SAM D21

Zum Programmieren oder Debuggen besitzt der SAM D21 eine Serial Wire Debug Schnittstelle. Als Programmiergerät wird ein JTAGICE3 verwendet, welcher im Elektroniklabor der Hochschule bereits vorrätig ist. Das Programmiergerät wird über einen 10 poligen 50 mil. Cortex Debug Connector mit dem SAM D21 verbunden. In Abbildung 26 ist der Schaltplan der JTAG-Schnittstelle zu sehen. Der Datenaustausch zwischen Mikrokontroller und Programmiergerät findet über die bidirektionale Datenleitung „SWDIO“ und die Clock „SWCLK“ statt. Die Clock und die Reset-Leitung sind jeweils mit einem Pullup-Widerstand verbunden. Zusätzlich zu den Datenleitungen brauch das Programmiergerät noch eine Referenz der Versorgungsspannung des SAM D21. [11, S. 1118–1121]

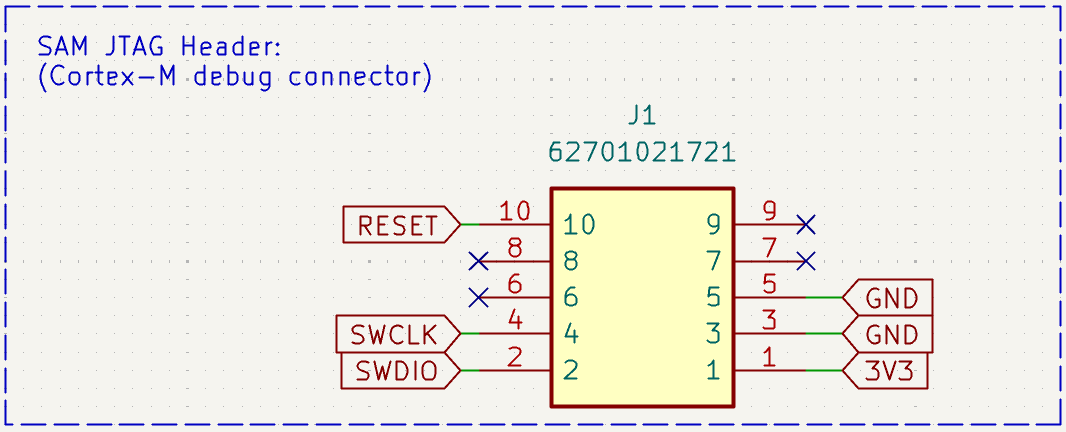


Abbildung 27 Schaltplan der JTAG-Schnittstelle

Zur allgemeinen Nutzung und speziell für die Fehlersuche wurde auf der Platine eine User-LED angebracht. Bei der LED handelt es sich um eine Low Current LED. Die LED ist mit dem PIN PA15 des SAM D21 verbunden, damit die LED leuchtet, muss der Pin auf Logisch-High gesetzt werden. In Abbildung 27 ist der Schaltplan der User LED zu sehen.

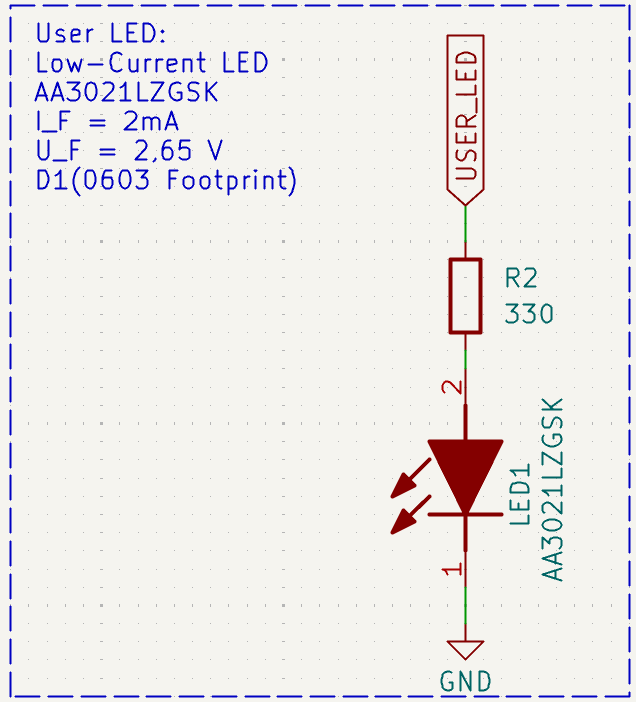


Abbildung 28 Schaltplan der User LED

### Buzzer

Ein Bild, das Text, Diagramm, Screenshot, Reihe enthält.

KI-generierte Inhalte können fehlerhaft sein.Eine der Entwicklungsanforderungen war eine Soundausgabe für die Mini-Arcade-Machine, um z. B. 8-Bit-Musik abzuspielen. Die Soundausgabe der Mini-Arcade-Machine wurde mit einem passiven Piezo-Buzzer realisiert. Piezo-Buzzer sind eine simple Lösung für eine primitive Soundausgabe. Für den Piezo-Buzzer wurde der CPT-1207-3TH von dem Hersteller Same Sky ausgewählt. Der ausgewählte Buzzer braucht eine bidirektionale Versorgungsspannung mit einer Amplitude von 3 V, hat einen Stromverbrauch von 3 mA und eine Lautstärke von ca. 75 dB. [16, S. 1] In Abbildung 28 ist der Schaltplan des Piezo-Buzzers zu sehen.

Abbildung 29 Schaltplan des Piezo-Buzzers

Bei dem Piezzo-Buzzer handelt es sich um einen passiven Buzzer, das bedeutet, der Buzzer besitzt keinen integrierten Oszillator. Um den Buzzer in Schwingung zu versetzten muss er mit einer Wechselspannung angesteuert werden. Dazu ist der Buzzer mit dem Pin PA04 des SAM D21 verbunden. Dem Pin wurde der Waveform-Generation-Output 0 der TCC0 Instanz des SAM D21 zugewiesen. Mit dem TCC0 lassen sich auf den zugewiesenen Pin Rechtecksignale mit einer beliebigen Frequenz modulieren. Die genaue Konfiguration der TCC0 Instanz wird später in KapitelXY genauer erklärt.

Mit der TCC0 Instanz erzeugt der Mikrokontroller ein unidirektionales Rechtecksignal von 0 bis 3,3 V. Der Buzzer benötigt jedoch ein bidirektionales Signal mit einer Amplitude von 3 V. Um dieses Problem zu lösen, wird der Buzzer mit einer virtuellen Masse betrieben. Die virtuelle Masse hat ein Potential von 1,65 V und ist mit dem negativen Pin des Buzzers verbunden. Durch die virtuelle Masse wird das unidirektionale Rechtecksignal für den Buzzer zu einer bidirektionalen Wechselspannung.

Das Potential der virtuellen Masse wird mit einem Spannungsfolger erzeugt, siehe Abbildung 28. Die Kombination aus Spannungsteiler und Operationsverstärker ist eine sehr simple Lösung, um ein niederohmiges Potential der Versorgungsspannung zu erzeugen.

Um den Klang des Piezo-Buzzers zu verbessern wird das Rechtecksignal des Mikrokontrollers mit einem Tiefpassfilter an einen Sinus angenähert. Um den Tiefpassfilter zu dimensionieren, wurde die Eckfrequenz fc oberhalb der Note C5 zu 630 Hz gewählt. Die Kapazität C2 des Tiefpassfilters berechnet sich aus der Eckfrequenz nach Gleichung 13. Der Widerstand R6 wurde zu 100 Ω gewählt, um den Spannungsabfall möglichst gering zu halten. Berechnet man die Kapazität nach Gleichung 13 bei einem Widerstand von 100 Ω, ist der nächstgelegene Standardwert aus der E12-Reihe 2,2 µF. Mit einer Kapazität von 2,2 µF und einem Widerstand von 100 Ω ergibt sich ein Tiefpassfilter mit einer Eckfrequenz von 723 Hz.

Gleichung 13 Berechnung der Kapazität aus der Eckfrequenz

### FPC-Steckverbinder

Das Display ER-TFT1.44-1 wird über einen FPC-Steckverbinder mit dem SAM D21 verbunden. Der FPC-Steckverbinder hat 26 Pins und einen Pitch von 0,5 mm. Die FPC-Buchse sitzt auf der Vorderseite der Platine, mit dem Rest der Elektronik. Das Display wird auf der Vorderseite angeschlossen und dann auf der Rückseite der Platine befestigt. Auf der Rückseite wird das Display mit doppelseitigem Klebeband befestigt und durch zwei Bohrungen ausgerichtet, siehe Abbildung 29. Da das Display auf der Vorderseite der Platine eingesteckt und auf der Rückseite befestigt wird, ist die Pinbelegung der FPC-Buchse genau spiegelverkehrt zum FPC-Steckverbinder des Displays.

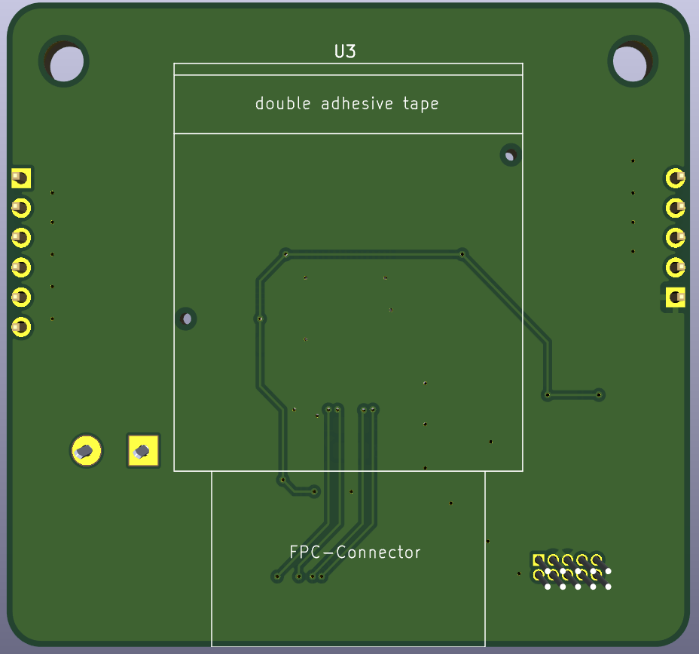


Abbildung 30 Befestigung des Displays

## Verbindungssystem

Die Spannungsversorgung, der Controller und das Mainboard werden über die Seitenwände der Mini-Arcade-Machine miteinander verbunden. Die Platinen werden über Stiftleisten mechanisch und elektrisch miteinander verbunden, siehe Abbildung 30. Da es sich beim aktuellen Entwicklungsstand noch um einen Prototyp handelt, wurden die Verbindungen bewusst steckbar ausgeführt.

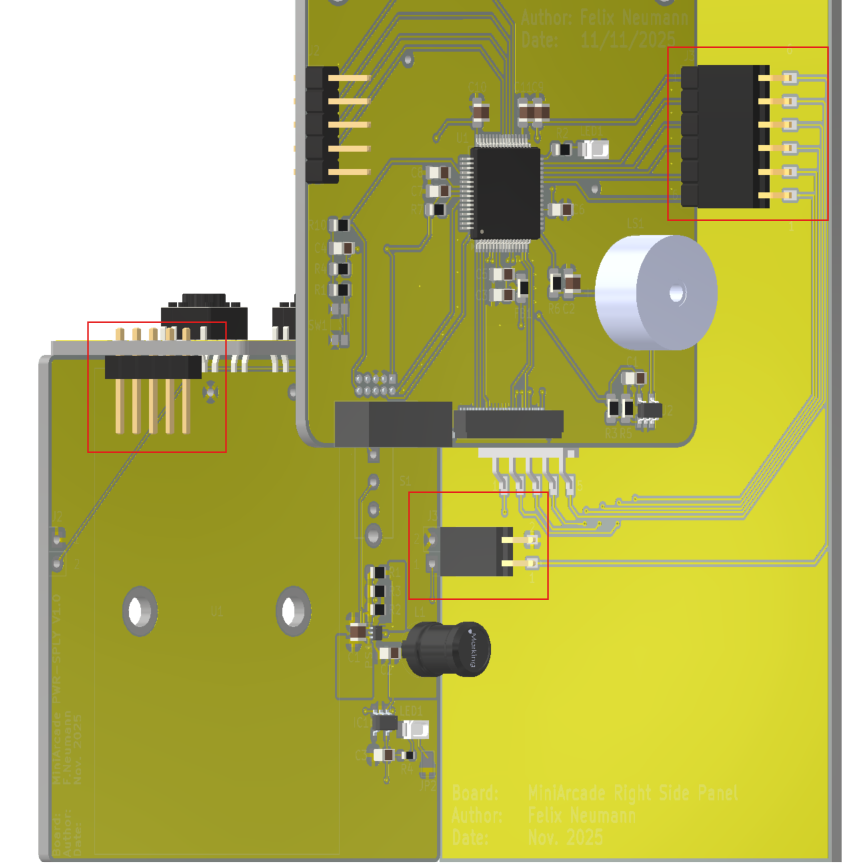


Abbildung 31 Verbindungssystem mit Stiftleisten

# Implementierung Software

## Einleitung Software

Ziel der Software-Entwicklung war, für die entwickelte Hardware ein abstraktes Grundgerüst zu schaffen, welches es zukünftigen Entwicklern ermöglicht, beliebige Arcade-Games zu programmieren, ohne Kenntnisse über die Hardware-Layer zu besitzen. Dazu wurde ein Displaytreiber für den LCD-Controller ST7735S sowie ein Grafik-Library und eine Font-Library programmiert. Für die Softwareentwicklung wurde die Entwicklungsumgebung MPLAB X IDE von dem amerikanischen Halbleiterhersteller Microchip genutzt. Der gesamte Code wurde in der Programmiersprache C geschrieben. Im Folgenden werden die entwickelten Softwarekomponenten im Detail beschrieben.

## Displaytreiber

Für die Schnittstelle zwischen dem SAM D21 und dem Display-Controller ST7735S wurde ein Displaytreiber entwickelt. Der Displaytreiber besteht aus der Headerdatei st7735s.h und einer gleichnamigen Quellcode-Datei. Abgesehen von C Standardbibliotheken nutzt der Displaytreiber die Hardware-Abstraction-Layer des SAM D21 sowie eine Delay-Funktion, die in einer anderen Quellcode-Datei ausprogrammiert wurden.

### Initialisierung

Bevor der Mikrokontroller Bilddaten senden kann, muss der Display-Controller einmali initialisiert werden. Die gesamte Initialisierung wurde in der Unterfunktion „st7735s\_init()“ ausprogrammiert. In Abbildung 17 ist die Initialisierungsroutine als Ablaufdiagramm dargestellt.

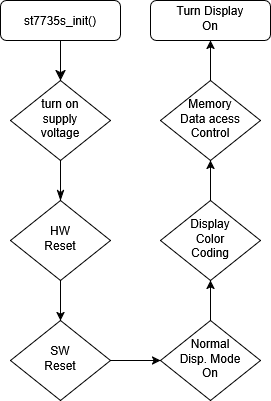


Abbildung 32 Initialisierungsroutine

Zu Beginn wird die Versorgungsspannung für den Display-Controller und die Hintergrundbeleuchtung eingeschaltet. Nachdem die Versorgungsspannung eingeschaltet wurde, muss ein Software- und Hardware-Reset durchgeführt werden. Beim Software- sowie Hardware-Reset muss laut Hersteller eine vorgeschriebene Verzögerung eingehalten werden. Dazu wurde in einer separaten Header- und Quellcode-Datei eine Delay-Funktion ausprogrammiert. Die Delay-Funktion nutzt den Systemtakt des SAM D21 und verzögert den Programmablauf um eine beliebige Zeit.

Nach dem Software-Reset muss der Controller den Sleep In Modus verlassen, um die volle Funktionalität des Displays nutzen zu können. Dazu wird der Befehl „SLPOUT“ an den Display-Controller gesendet. [12, S. 87] Nachdem der Sleep In Modus verlassen wurde, kann die eigentliche Initialisierung des Controllers beginnen.

Im aktuellen Code werden nur die Farbtiefe und der Zugriff auf den RAM verändert, ansonsten werden die Standardeinstellungen des Controllers beibehalten. Für die Farbtiefe des Displays wurde das Farbmodell RGB565 gewählt, das entspricht einer Auflösung von 16 Bit pro Pixel. Die gewählte Farbtiefe ist ein guter Kompromiss zwischen Farbgenauigkeit und der Bildwiederholrate.

Über den Befehl „MADCTL“ wurde der Zugriff auf den RAM verändert. Die Bits „MY“ und „MX“ wurden auf Eins gesetzt, dadurch werden die Reihenadressen und Spaltenadressen gespiegelt, siehe Abbildung 18. [12, S. 142] Durch die Spiegelung der Adressen sitzt der Nullpunkt des Display-Koordinatensystems in der oberen linken Ecke, der neu gewählte Nullpunkt ist deutlich intuitiver. Zusätzlich zu den Speicheradressen wurde der BGR-Farbfilter aktiviert, damit die übermittelten Farbwerte korrekt dargestellt werden. Dazu wurde das Bit „RGB“ auf Eins gesetzt. Zuletzt wird das Display noch über den Befehl „DISPON“ eingeschaltet. [12, S. 127]

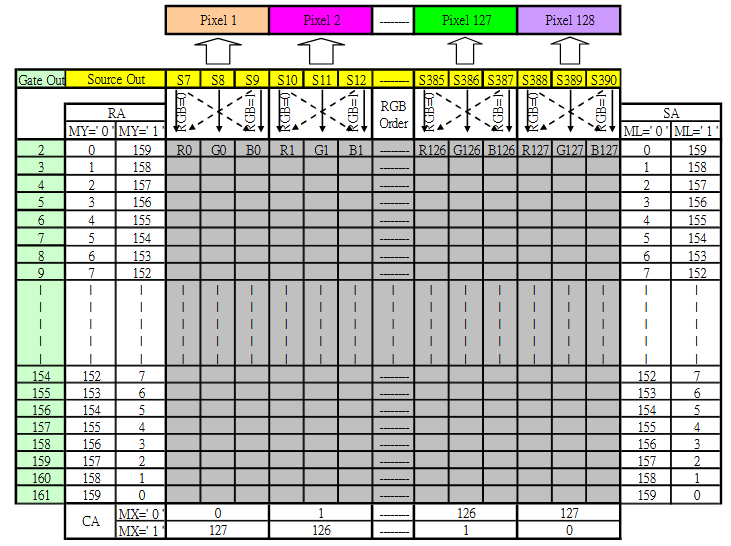


Abbildung 33 Organisation der Speicheradressen

### Controller Befehle

Jeglicher Datenaustausch zwischen Mikrokontroller und Display wird durch den entsprechenden Befehl des Display-Controllers eingeleitet. In der Headerdatei des Displaytreibers wurde mit Präprozessorkonstanten eine Befehlsliste für alle möglichen Befehle des ST7735S Display-Controllers angelegt. Mit der Funktion „st7735s\_send\_command()“ wird ein Befehl an den Display-Controller gesendet. In Abbildung 19 ist der Ablauf der Funktion dargestellt. Der übergebene Befehl wird über das Präprozessor-Makro „Send\_Data()“ in das SERCOM-Datenregister des SAM D21 geschrieben. Mit dem Präprozessor-Makro „Buisy\_Wait()“ wird der Programmablauf verzögert, bis die Transmit-Complete-Flag gesetzt und die Daten vollständig gesendet wurden.

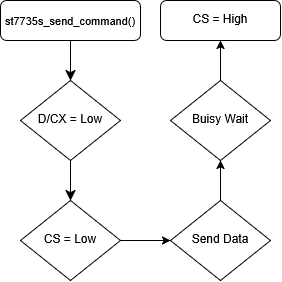


Abbildung 34 Ablaufdiagramm „st7735s\_send\_command()“

### Datenaustausch

Für die Initialisierung sowie den Austausch der Bilddaten müssen je nach Parameter Daten mit einer Größe von 8 Bit oder 16 Bit an den Display-Controller gesendet werden. Dafür wurden die beiden Funktionen „st7735s\_send\_data8bit()“ und „st7735s\_send\_data16bit ()“ implementiert. Die Funktion „st7735s\_send\_data8bit()“ ist genau gleich aufgebaut wie die Funktion, um einen Befehl zu senden, lediglich der „D/CX“ Pin wird auf Logisch-High gesetzt. Dadurch interpretiert der Controller das empfangene Signal als Datenaustausch. [12, S. 45] Um Daten mit einer Größe von 16 Bit zu senden, nutzt die Funktion „st7735s\_send\_data16bit()“ die beiden Präprozessor-Makros „Send\_lower\_Byte()“ und „Send\_upper\_Byte()“. Durch die beiden Bitmasken werden die 16 Bit großen Daten in zwei 8 Bit große Pakete aufgeteilt.

### Bilddaten

Um Bilddaten an den Display-Controller zu senden, muss zuerst das Fenster definiert werden, welches im RAM beschrieben werden soll. Dafür wurde die Funktion „st7735s\_set\_window()“ implementiert. In Abbildung 20 ist der Ablauf der Funktion dargestellt. Um das Fenster im RAM zu definieren, werden der Funktion jeweils die Start- und Endkoordinaten für die Reihen- und Spaltenadresse übergeben. Als erstes wird, geprüft ob die übergebenen Adressen innerhalb der Displaydimensionen sind und die Startadresse nicht größer als die zugehörige Endadresse ist. Wenn die übergebenen Adressen zulässig sind, werden sie mit den Befehlen „RASET“ und „CASET“ an den Display-Controller gesendet. [12, S. 128–132]

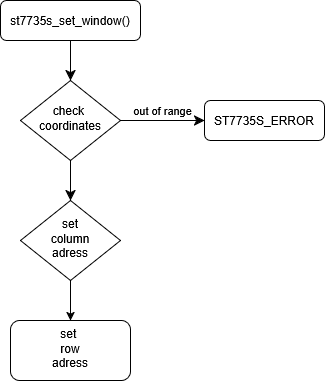
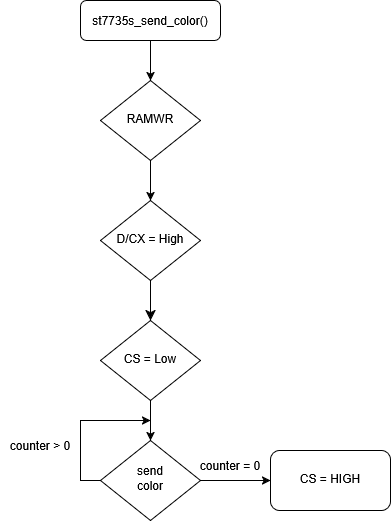


Abbildung 35 Ablaufdiagramm „st7735s\_set\_window()“

Nach dem das Fenster im RAM definiert wurde können mit dem Befehl „RAMWR“ die Bilddaten für das definierte Fenster gesendet werden. Dazu wurde die Funktion „st7735s\_send\_color()“ implementiert. In Abbildung 21 ist das Ablaufdiagramm der Funktion dargestellt. Da der übergebene Farbwert eine Größe von 16 Bit hat, ist die Funktion ähnlich aufgebaut wie die Funktion „st7735s\_send\_data16bit()“. Zusätzlich zu dem Farbwert wird der Funktion noch ein Counter übergeben, der festlegt, wie oft der Farbwert gesendet wird.

### Bildwiederholrate

Abbildung 36 Ablaufdiagramm „st7735s\_send\_color()“



Die Bildwiederholrate war eine zentrale Herausforderung bei der Entwicklung des Displaytreibers. Um das gesamte Display auf einmal zu aktualisieren, muss man, bei einer Farbtiefe von 16 Bit pro Pixel, senden. Serielle Schnittstellen kommen bei diesen Datengrößen schnell an ihre Grenzen. Die SPI-Schnittstelle ist durch den Displaycontroller auf eine maximale Clock-Frequenz von 14 MHz begrenzt. [12, S. 36] Das ergibt eine theoretische Bildwiederholrate von . Diese Betrachtung ist stark vereinfacht und berücksichtigt nicht, dass die Ausführung des Programms auch Zeit benötigt, in der keine Daten gesendet werden können.

In einer früheren Version des Displaytreibers wurden zum Senden von Daten die von Microchip bereitgestellten Funktionen genutzt. Um Daten in das SERCOM-Datenregister des SAM D21 zu schreiben, wurden mehrere Funktionsaufrufe durchgeführt. In Abbildung 22 ist die Chip Select- und SDA-Leitung für eine Datenübertragung von   
8 Bit dargestellt. Es ist klar zu erkennen, dass durch die vielen Funktionsaufrufe die tatsächliche Datenübertragung unnötig lang wird.

Ein Bild, das Screenshot, Multimedia-Software enthält.

KI-generierte Inhalte können fehlerhaft sein.

Abbildung 37 8 Bit Datenübertagung mit den von Microchip bereitgestellten Funktionen

Um dieses Problem zu lösen, wurden die in Kapitel 7.2.3 beschriebenen Funktionen für den Datenaustausch programmiert. Dabei lag der Fokus darauf, möglichst effiziente Funktionen zu programmieren, deren Ablauf trotzdem nachvollziehbar und sicher ist. Um unnötige Funktionsaufrufe zu vermeiden, wurde mit Funktions-Makros gearbeitet. Bei jedem Datenaustausch muss die Transmit-Complete-Flag des SAM D21 abgefragt werden, um zu verhindern, dass Daten überschrieben werden. Diese Routine wurde mit dem Funktions-Makro „Buisy\_Wait()“ realisiert, anstatt sie in eine separate Unterfunktion auszulagern.

Außerdem wurden alle Funktionen, die am Datenaustausch beteiligt sind, als Inline-Funktionen ausgeführt. Bei Inline-Funktionen wird der Code der Funktion vom Compiler am Ort des Aufrufes eingefügt. Dadurch wird die Funktion schneller ausgeführt, allerdings mehr Programmspeicher beansprucht. Inline-Funktionen sind dann sinnvoll, wenn sie oft aufgerufen werden und wenig ausführbaren Code enthalten.

Bild neue Schnittstelle einfügen mit schnelleren Funktionen.

## Grafik-Library

# Literaturverzeichnis

[1] „Rückruffunktion“, *Wikipedia*. 18. Juni 2025. Zugegriffen: 29. Dezember 2025. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=R%C3%BCckruffunktion&oldid=257131550

[2] „Callbacks in C“, GeeksforGeeks. Zugegriffen: 29. Dezember 2025. [Online]. Verfügbar unter: https://www.geeksforgeeks.org/c/callbacks-in-c/

[3] „Ereignis (Programmierung)“, *Wikipedia*. 14. April 2024. Zugegriffen: 29. Dezember 2025. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Ereignis\_(Programmierung)&oldid=244033128

[4] „Rasterung von Linien“, *Wikipedia*. 23. April 2024. Zugegriffen: 31. Dezember 2025. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Rasterung\_von\_Linien&oldid=244319003#Einfache\_Methoden

[5] „Bresenham-Algorithmus“, *Wikipedia*. 7. August 2025. Zugegriffen: 31. Dezember 2025. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Bresenham-Algorithmus&oldid=258677015

[6] „Rasterung von Kreisen“, *Wikipedia*. 23. Dezember 2025. Zugegriffen: 7. Januar 2026. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Rasterung\_von\_Kreisen&oldid=262671255#Midpoint-Algorithmus

[7] „TFT LCD“, *Wikipedia*. 9. Januar 2026. Zugegriffen: 14. Januar 2026. [Online]. Verfügbar unter: https://en.wikipedia.org/w/index.php?title=TFT\_LCD&oldid=1332044184

[8] „Flüssigkristallanzeige“, *Wikipedia*. 26. November 2025. Zugegriffen: 14. Januar 2026. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Fl%C3%BCssigkristallanzeige&oldid=261891663

[9] „Abwärtswandler“, *Wikipedia*. 27. Dezember 2024. Zugegriffen: 9. Januar 2026. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Abw%C3%A4rtswandler&oldid=251613339

[10] „Low-dropout regulator“, *Wikipedia*. 23. November 2025. Zugegriffen: 13. Januar 2026. [Online]. Verfügbar unter: https://en.wikipedia.org/w/index.php?title=Low-dropout\_regulator&oldid=1323761443

[11] Microchip, „SAM D21/DA1 Family“. [Online]. Verfügbar unter: https://www.mouser.de/datasheet/3/282/1/SAM-D21-DA1-Family-Data-Sheet-DS40001882H.pdf

[12] Sitronix, „ST7735S Datasheet“. [Online]. Verfügbar unter: https://www.hpinfotech.ro/ST7735S.pdf

[13] „Micro (Batterie)“, *Wikipedia*. 6. Februar 2024. Zugegriffen: 4. Februar 2026. [Online]. Verfügbar unter: https://de.wikipedia.org/w/index.php?title=Micro\_(Batterie)&oldid=241934353

[14] Diodes Incorporated, „AP61100Z6-7“. [Online]. Verfügbar unter: https://4donline.ihs.com/images/VipMasterIC/IC/DIOD/DIOD-S-A0010726746/DIOD-S-A0010880068-1.pdf?hkey=CECEF36DEECDED6468708AAF2E19C0C6

[15] Microchip, „MIC94310“. [Online]. Verfügbar unter: https://www.mouser.de/datasheet/3/282/1/MIC94310-Data-Sheet-20006105B.pdf

[16] Same Sky, „CPT-1207-3TH“. [Online]. Verfügbar unter: https://www.mouser.de/datasheet/3/6118/1/cpt\_1207\_3th.pdf

Mögliche To Dos:

* TFT LCD überabeiten
* Software Bidlwiederholrate zweite Grafik einfügen