Freie Universität Berlin
Fachbereich Mathematik und Informatik

Bachelorarbeit der Informatik

3D-Simulation des Additionswerkes der Z1 von Konrad Zuse

von *Jakob Mischek*

Betreuer:
Prof. Dr. Raúl Rojas

10. Dezember 2012

Zusammenfassung

Die Z1 von Konrad Zuse war der erste tatsächlich gebaute programmierbare Rechner der Menschheitsgeschichte. Was sie bis heute relevant macht, ist, dass sie bereits einige Konzepte der Rechnerarchitektur vorwegnahm, die auch in der heutigen Computertechnik noch verwendet werden, wie beispielsweise das Rechnen mit binären Zahlen in Gleitkommadarstellung. In einem Punkt aber unterschied sich die Z1 von allen nachfolgenden Computern: sie funktionierte komplett mechanisch, indem sich speziell geformte Metallteile gegeneinander verschoben. Wie dies genau funktionierte, lässt sich nur schwer nachvollziehen, unter anderem weil die originale Maschine nicht mehr existiert und auch der zu Anschauungszwecken von Zuse gefertigte Nachbau nicht mehr in Betrieb ist.

Im Rahmen dieser Bachelorarbeit wurde auf Basis einer Patenschrift von Konrad Zuse eine 3D-Simulation der zentralen, für mathematische Operationen zuständigen Komponente der Z1, des Additionswerkes angefertigt. Für die einzelnen Rechenschritte werden die Bewegungen der Metallteile animiert dargestellt. Durch Ausblenden von Teilen und Legen von Querschnitten durch die Maschine können Einsichten gewonnen werden, wie sie an einem realen Nachbau nicht möglich sind. Darüber hinaus verknüpft ein logischer Schaltplan einzelne Bereiche des Additionswerkes mit Informationen über deren Funktion.

Des Weiteren beschreibt die vorliegende Arbeit, welche Ergänzungen und Korrekturen an Zuses Plänen aus der Patentschrift gemacht werden mussten, um ein funktionsfähiges Additionswerk zu erhalten.

Eidesstattliche Erklärung

Ich versichere, die Bachelorarbeit selbstständig und lediglich unter Benutzung der angegebenen Quellen und Hilfsmittel verfasst zu haben. Ich erkläre weiterhin, dass die vorliegende Arbeit noch nicht im Rahmen eines anderen Prüfungsverfahrens eingereicht wurde.

Berlin, den 10. Dezember 2012 _____

Jakob Mischek

Inhaltsverzeichnis

1 Einleitung **3**
1.1 Zielsetzung 4
1.2 Struktur der Arbeit 4

2 Einordnung **5**

3 Funktionsprinzipien **6**
3.1 Das mechanische Schaltglied 6
3.2 Mechanische Schaltung 6
3.3 Wiederverwendbarkeit 7
3.4 Taktmodell der Z1 8
3.5 Logischer Aufbau 9

4 Rekonstruktion **11**
4.1 Digitalisierung der Blechformen 11
4.1.1 Das Raster 11
4.1.2 Die Blechausschnitte 13
4.1.3 Die Blechumrisse 14
4.1.4 Ebene A 16
4.1.5 Ebene B 16
4.1.6 Ebenen C und D 17
4.2 Vertikaler Aufbau 17

5 Implementierung **20**
5.1 Technologie 20
5.1.1 Flash 20
5.1.2 Java-Applet 21
5.1.3 WebGL 21
5.1.4 Three.js 22
5.2 Funktionen 22
5.2.1 3D-Ansicht 22
5.2.2 Steuerung 23
5.2.3 Ausschnitte 24
5.2.4 Logischer Schaltplan 24
5.2.5 Darstellungsoptionen 24
5.3 Aufbau des 3D-Modells 24
5.4 Shader zum Anzeigen von Ausschnitten 25
5.5 Simulation der Bewegungsabläufe 26

6 Fazit und Ausblick **28**

7 Anhang **29**
7.1 Originalpläne 29
7.1.1 Ebene A 29
7.1.2 Ebene B 30
7.1.3 Ebene C 31
7.2 Rekonstruktionspläne 33
7.2.1 Blechausschnitte 33
7.2.2 Blechformen 34
7.3 Weitere Abbildungen 60

Abbildungsverzeichnis **61**

Quellenverzeichnis **62**
Literatur 62
Webseiten 63

1 Einleitung

Die Z1 von Konrad Zuse wird oft als erster Computer der Welt bezeichnet. Nach einem flüchtigen Blick auf die Maschine wird man allerdings wenig von dem wiederfinden, was für uns heute das Wort Computer bedeutet. Von ihrem Funktionsumfang her gesehen, scheint die Bezeichnung „programmierbarer Rechner“ angebrachter. Erst wer genauer hinguckt, kann viele Konzepte entdecken, die noch heute in der Computertechnik verwendet werden. Die Z1 rechnet intern mit binären Zahlen in Gleitkommadarstellung. Mit ihrer Trennung von Speicher, Prozessor und Ein-/Ausgabe führte sie ein Prinzip ein, das als Vorläufer der Von-Neumann-Architektur gelten kann [4]. Dies sind Konzepte, die sich bis heute als praktikable Strategien behauptet haben. Es kann deshalb nicht stark genug gewürdigt werden, dass Konrad Zuse bereits 1937 diese Konzepte in alleiniger Forschung als optimalen Weg der Rechentechnik erkannte.

Kurz zuvor, 1936, hatte Alan Turing bereits mit seiner Turingmaschine ein theoretisches Modell eines Computers vorgestellt. Doch die Z1 war dann nicht nur der erste Rechner, der wirklich gebaut wurde, im Unterschied zur Turinmaschine war sie zur praktischen Anwendung konzipiert und hätte für Berechnungen eingesetzt werden können.

Und in noch einem weiteren Punkt nimmt die Z1 eine besondere Stellung ein: sie funktionierte komplett mechanisch und ließ sich ganz ohne Strom mit einer Handkurbel antreiben. Intern wird die Drehbewegung zur Verschiebung von Metallblechen genutzt, mit denen Schaltungen realisiert sind.

Wie dies genau funktioniert, lässt sich nur schwer nachvollziehen, unter anderem weil die originale Maschine nicht mehr existiert und auch der zu Anschauungszwecken von Zuse gefertigte Nachbau nicht mehr in Betrieb ist.

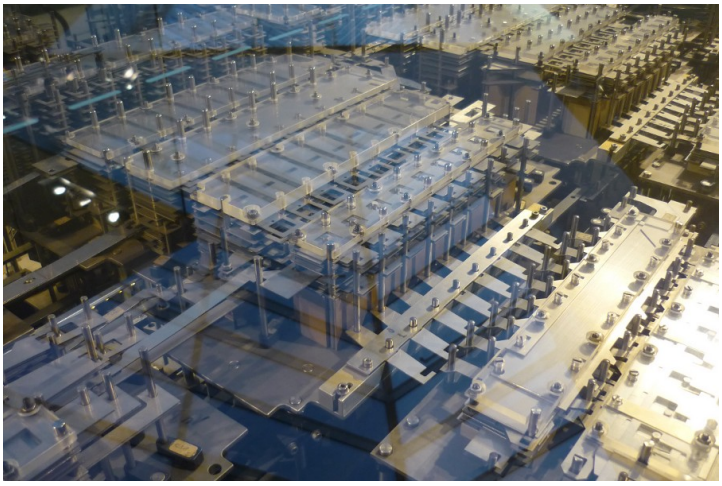


Abbildung 1: Detail des Z1-Nachbaus im Berliner Technikmuseum

1.1 Zielsetzung

An dieser Stelle setzt die vorliegende Arbeit an. Mit einer 3D-Simulation einer zentralen Komponente der Z1, des Additionswerkes, soll die Mechanik mit ihren Bewegungen gezeigt werden. Darüber hinaus sollen Funktionen implementiert werden, die tiefere Einsichten in die Maschine zulassen, als dies an einem Nachbau möglich ist. Erstrebt wird dabei die Funktionsweise der Z1 verstehbar zu machen.

Einsetzbar sollte die Simulation als Informationsangebot zur individuellen Erkundung sowie auch als unterstützende Demonstration in Lehrveranstaltungen sein.

1.2 Struktur der Arbeit

Im Kapitel 2 erfolgt eine Einordnung des Additionswerkes bezüglich seiner Bedeutung für die Z1 und die Arbeiten Konrad Zuses.

Im Kapitel 3 werden die Funktionsprinzipien hinter dem Additionswerk erklärt.

Im Kapitel 4 wird beschrieben, wie aus den Plänen von Zuse das Modell des Additionswerkes rekonstruiert wurde und welche Ergänzungen und Änderungen gegenüber den originalen Plänen gemacht wurden.

Im Kapitel 5 soll zunächst auf die verwendete Technologie eingegangen werden. Anschließend werden die umgesetzten Funktionen der 3D-Simulation beschrieben und danach wichtige Aspekte der Implementierung einzeln erörtert.

Im Kapitel 6 wird zusammengefasst, was im Rahmen dieser Bachelorarbeit erreicht wurde und aufgezeigt, welche Möglichkeiten zur Weiterentwicklung bestehen.

Im Anhang 7 sind die originalen und rekonstruierten Pläne zu finden.

2 Einordnung

Das in dieser Bachelorarbeit behandelte Additionswerk ist nur eine Komponente der Z1. Es übernimmt Additionen im binären Zahlensystem und ist in seiner hier dargelegten Form für vierstellige Zahlen ausgelegt. Die Z1 hingegen enthielt Additionsvorrichtungen für 24 Stellen. Durch das Verwenden einer Gleitkommadarstellung wurde jedoch ein weitaus höherer Wertebereich als 2^{24} erreicht. Auch die anderen von der Z1 beherrschten Rechenarten Subtraktion, Multiplikation und Division werden als Folge von Additionen über diesen Teil der Machine ausgeführt [4].

Als Grundlage dieser Bachelorarbeit diente eine von Zuse 1965 verfasste Patentschrift [9, 10] über die Z1. Weil die originale Z1 mitsamt ihren Bauplänen im Zweiten Weltkrieg zerstört wurde, musste Konrad Zuse nach dem Krieg die Patentschrift seinen Erinnerungen nach aufschreiben und die Details neu herausarbeiten. Dadurch dürften sich die hier später noch behandelten Unstimmigkeiten eingeschlichen haben. Außerdem umfasst die Patentschrift nicht die komplette Rechenmaschine Z1, sondern behandelt lediglich einige Komponenten.

Für den 1989 angefertigten Nachbau der Z1 legte Konrad Zuse neue Pläne an. Das Additionswerk aus der hier realisierten Simulation lässt sich somit nicht eins zu eins am Nachbau wiederfinden.

3 Funktionsprinzipien des Additionswerkes

3.1 Das mechanische Schaltglied

Grundlage der Z1 ist das mechanische Schaltglied, ein eingetragenes Patent von Konrad Zuse [7]. Wie in Abbildung 2 zu sehen, besteht es aus Blechen mit Ausschnitten und einem Stift. Die Funktionsweise ist einfach erklärt:

- Eine eingehende Kraft wirkt auf das Blech A und bewegt dieses in positive x-Richtung und wieder zurück.
- Das Steuerblech C kann in y-Richtung bewegt werden und verschiebt dabei den Schaltstift S zwischen zwei möglichen Positionen hin und her.
- In seiner Stellung in positiver y-Richtung liegt der Stift in einem Ausschnitt des Bleches A und wird von diesem bei seinen Bewegungen mitgeführt.
- Eine Bewegung von S in x-Richtung wirkt sich auch auf das Blech B aus.
- Mit dem Steuerblech kann also entschieden werden, ob eine Bewegung von A nach B übertragen wird.
- Das Fest- oder Grundblech D dient zur Führung des Stiftes. Man beachte, dass es keinen rechteckigen Ausschnitt besitzt, sondern eine, bei Zuse Sperrnase genannte, hervorstehende Ecke hat. Diese Sperrnase verhindert, dass sich der Stift in seiner ruhenden Position fälschlicherweise, beispielsweise durch Reibung mit der Kante von A, in x-Richtung bewegen kann.

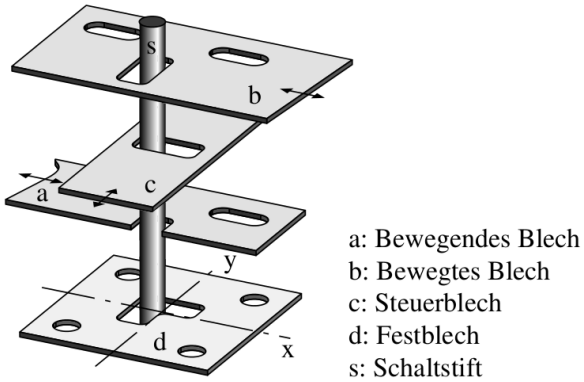


Abbildung 2: Einfaches mechanisches Schaltglied [7]

3.2 Einfache mechanische Schaltung

Das mechanische Schaltglied entspricht in seiner Funktionalität der eines Relais. Nur das bei einem Relais keine Bewegung sondern Strom weitergeleitet wird und die Steuerung mit einer

elektromagnetischen Spule erfolgt. Zuses Nachfolgemodelle der Z1 und auch andere frühe Computer wie der Harvard Mark I wurden aus Relaisschaltungen aufgebaut. Sehr schnell wurden dann aber Relais in der Computertechnik durch Elektronenröhren und später durch Transistoren ersetzt, das Prinzip blieb aber bestehen.

Die Schaltpläne von Zuses Relaiscomputer Z3 und andere Relaisschaltungen sind bekannt. Man könnte sie eins zu eins aus mechanischen Schaltgliedern nachbauen. Eine 2D-Simulation eines Z1-ähnlichen Additionswerkes von Götz Widiger [23] basiert auf dieser Vorgehensweise, siehe Abbildung 3. Es gibt jedoch einige Gründe, warum Zuses Additionswerk in der Z1 komplizierter ist.

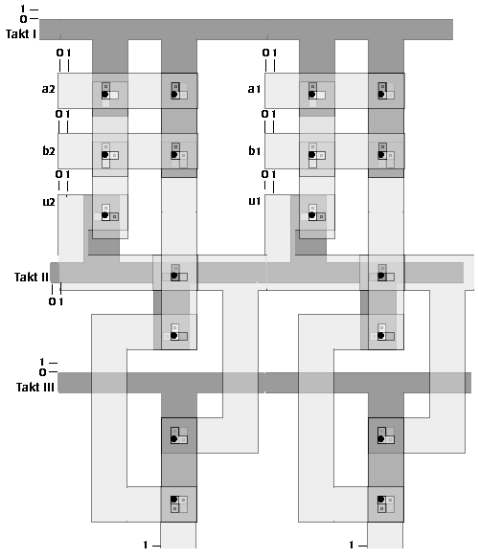


Abbildung 3: Einfache Simulation eines Additionswerkes [23]

3.3 Wiederverwendbarkeit von Schaltungen

Das erste Problem ergibt sich bereits aus der Konstruktion des mechanischen Schaltgliedes. Die Sperrnase verhindert nicht nur eine unbeabsichtigte Bewegung des Stiftes in Arbeitsrichtung, sie verhindert ebenfalls, dass nach einer übertragenen Bewegung das Steuerblech bewegt werden kann. Eine Änderung der Schaltstellung ist also erst nach Rückbewegung von A und B mögliche, siehe Abbildung 2.

Für den Aufbau der Simulation in Abbildung 3 bedeutet dies folgendes: nachdem die drei markierten Takte ausgeführt wurden und das Ergebnis vorliegt, müssen die Taktbewegungen in umgekehrter Reihenfolge rückgängig gemacht werden. Erst dann können die Steuerbleche der Eingabe umgeschaltet werden. Für den Aufbau eines realen Rechners wie der Z1 ist dies

unmöglich. Es würde bedeuten, dass alle Schaltungen bis zum Ende einer Berechnung blockiert wären. Die Z1 gewinnt ihre Leistungsfähigkeit aber gerade daraus, dass im Laufe einer Rechnung das Additionswerk ständig wiederverwendet wird und so über eine kleine Recheneinheit komplexe Rechnungen möglich werden.

Dies zu realsieren dürfte eine der größten Schwierigkeiten gewesen sein, die sich Zuse bei der Planung der Z1 stellten. Um eine zuverlässige Berechnung zu garantieren, konnte er auf die Sperrnasen nicht verzichten. Eine Lösung des Problems besteht darin, für die Sperrnasen eigene Bleche zu benutzen, die sich zur Seite bewegen können, wenn das Steuerblech verschoben werden soll, wie Zuse in seiner Patentschrift beschreibt [9, S. 9]. Im simulierten Additionswerk finden sich solche ausweichenden Sperrnasen beispielsweise oben am Blech P in Ebene A, siehe Seite 37.

Eine andere Möglichkeit ist, mit Federn einen entsprechenden Widerstand gegenüber der Bewegung zu erzeugen, der nur bei beabsichtigter Bewegung überwunden werden kann [9, S. 11]. Im Additionswerk dient die Kerbe am Blech K auf den Ebenen C und D (Seite 50) dazu, um bei einer Feder einzurasten, auch wenn Zuse dies nicht explizit erwähnt. Die Federn sind nur in Fig. 60 [10] angedeutet.

3.4 Taktmodell der Z1

Ein weiteres Problem, dass sich aus der einfachen Schaltung (3.2) ergibt, ist die Taktabfolge. Angenommen das Problem der Sperrnasen wäre behoben, wäre immer noch offen, wann die Rückbewegung der Bleche erfolgen sollte. Zuse hat deshalb für die Z1 ein einheitliches Taktmodell entwickelt. Es ist in Abbildung 4 schematisch dargestellt.

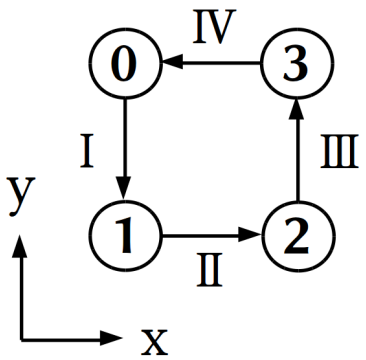


Abbildung 4: Takt und Positionen

Es gibt vier Takte I bis IV, jeweils abwechselnd in x- und in y-Richtung. Gegenüberliegende Takte stehen damit als Bewegung und Rückbewegung zueinander und können von einem Impulsglied erzeugt werden. Es sind demnach nur zwei Impulsglieder, für jede Richtung eines, nötig [9, S. 8]. Die Bewegungen der Impulsglieder können aus einer kontinuierlichen

Drehbewegung erzeugt werden [10, Fig. 21]. Die Z1 in etwa ließ sich durch eine Handkurbel antreiben.

Zur Umsetzung der Simulation wurden die einzelnen Position, die im Taktkreislauf möglich sind, von 0 bis 3 nummeriert (Abbildung 4). Die Bezeichnung der Richtungen hält sich an Zuses Schriften:

> Da die Bezeichung waagerecht und senkrecht nur für die zeichnerische Darstellung Geltung hätte und räumlich sinnlos wäre, werden in folgendem die Begriffe x- und y-Richtung benutzt. [8, S. 3]

3.5 Logischer Aufbau

Das in der Patentschrift beschriebene Additionswerk kann zwei 4-stellige Binärzahlen addieren. Die erste Zahl sei a genannt und hat die Ziffern a_0 bis a_3, die zweite Zahl b die Ziffern b_0 bis b_3. Das Ergebnis k besitzt ebenfalls vier Stellen k_0 bis k_3. Geht das Ergebnis über eine vierstellige Zahl hinaus, so wird dies am Ausgang f_4 als Überlauf signalisiert. Es existiert dann also ein Übertrag auf die nicht vorhandene fünfte Stelle. Es können mehrere dieser Additionswerke aneinander gereiht werden, der Überlauf wird dann vom nächsten Additionswerk ausgewertet. Dafür besitzt das Additionswerk den Eingang f_0, also einen Übertrag an Stelle 0 von der möglicherweise davorliegenden Stelle.

In Abbildung 5 ist der logische Schaltplan für die Addition einer Stelle abgebildet. Die einzelnen Logikgatter sind von α bis η beschriftet.

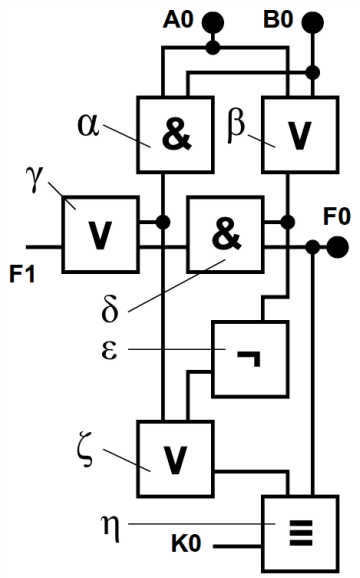


Abbildung 5: Logischer Schaltplan an der Stelle 0

Die Funktion der einzelnen Gatter:

- α entspricht der Aussage: beide Ziffern sind 1
- β entspricht der Aussage: mindestens eine Ziffer ist 1
- ε entspricht der Aussage: keine Ziffer ist 1
- ζ entspricht der Aussage: die Summe der Ziffern ist gerade, also beide Ziffern sind 1 oder keine.
- η ist das Ergebnis an dieser Stelle. Die Wahrheitstabelle:

ζ	f_0	$\eta = (\zeta \equiv f_0)$	wörtliche Rechnung
0	0	1	Summe ungerade und kein Übertrag
0	1	0	Summe ungerade und Übertrag
1	0	0	Summe gerade und kein Übertrag
1	1	1	Summe gerade und Übertrag

Die Gatter der Übertragskette:

- δ Es wird ein Übertrag von der vorherigen Stelle weitergegeben, genau dann, wenn es einen Übertrag von der vorherigen Stelle gibt **UND** mindestens eine Ziffer 1 ist.
- γ Es wird ein Übertrag an die nächste Stelle signalisiert, genau dann, wenn ein Übertrag von der vorherigen Stelle weitergegeben **ODER** an dieser Stelle ein neuer (d.h. beide Ziffern 1) erzeugt wird.

Der komplette Schaltplan ist im Anhang unter Abbildung 20 abgebildet.

4 Rekonstruktion

Der Arbeitsschritt, aus den Zeichnungen von Konrad Zuse ein komplettes Modell des Additionswerkes anzufertigen, sei hier Rekonstruktion genannt, denn er beinhaltet die Ergänzung vieler in den Zeichnungen nicht enthaltener Details. An einigen Stellen bedarf es zusätzlich einer Korrektur von Zuses Plänen. Diese Änderungen und Ergänzungen zu Zuses Patentschrift sind sicherlich der allgemein interessanteste Teil dieser Bachelorarbeit, denn die selben Probleme werden sich jedem stellen, der sich mit Zuses Patentschrift befasst. Deshalb soll hier ausführlich dargelegt werden, wie die Informationen aus den Zeichnungen übernommen und welche Änderungen gemacht wurden.

4.1 Digitalisierung der Blechformen

Der wahrscheinlich einfachste Ansatz die Blechumrisse in eine weiterverarbeitbare Form zu bringen besteht darin, sie in einem Vektorgrafikprogramm mit den Plänen als Hintergrund nachzuzeichnen. Dieses Vorgehen wurde so auch von Kostulski [2] beschrieben. Allerdings müsste dann auch erläutert werden, mit welcher Genauigkeit die ursprünglichen Pläne gezeichnet wurden. Weiterhin ergibt sich das Problem, auf welcher Grundlage die Bleche ohne vorhandene Zeichnung erstellt werden, worauf Kostulski leider nicht detailliert eingeht.

Aufgrund dieser Bedenken wurde für diese Arbeit ein anderer Ansatz gewählt, der daraus besteht, die Blechumrisse auf einem zuvor definierten Raster neu zu konstruieren.

4.1.1 Das Raster

Als Grundlage wurde ein gitterförmiges Raster mit einem Linienabstand von 10 Einheiten gewählt. Vergleiche dazu Abbildung 6.

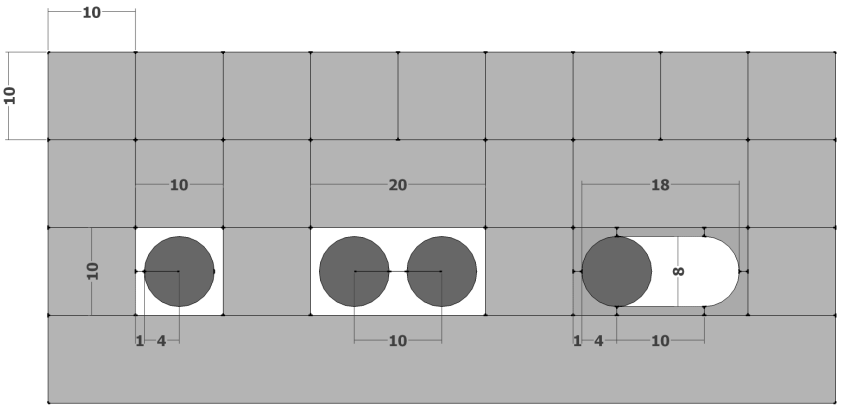


Abbildung 6: Raster mit Blechausschnitten und Stiften

Ein einfacher Blechausschnitt besitzt in diesem Raster die Abmessungen 10×10 Einheiten. Ein Stift, der durch diesen Blechausschnitt geht, hat einen kreisförmigen Querschnitt mit einem Durchmesser von 8 Einheiten. Er besitzt folglich zu den Rändern des Ausschnitts in alle Richtungen einen Abstand von 1 und berührt diese nicht.

Die standardisierte Bewegungslänge eines Stiftes entspricht der Wanderung vom Mittelpunkt eines Rasterquadrates in das nächste und misst somit 10 Einheiten.

In Ausschnitten, die den Stift durch Berührung halten oder führen sollen, ist der Ausschnitt an allen Seiten mit einem Offset von 1 nach innen gerückt.

Der weitere Aufbau eines Bleches orientiert sich an einem Grundlayout, abgebildet in Abbildung 7. Dieses Grundlayout lässt sich vor allem an den Grundblechen erkennen (zum Beispiel die Abbildungen auf Seite 38 und 44). Im Grundlayout sind Ausschnitte von 20×20 Einheiten, in denen die beweglichen Stifte sich in einer kompletten Runde bewegen können. Zwischen den Ausschnitten besteht ein Abstand von ebenfalls 20 Einheiten. Auf dem Kreuz zwischen vier Ausschnitten ist mittig ein kleinerer Ausschnitt, der einen Stift festhält. Hier stecken die statischen Stifte, welche die Grundbleche zusammenhalten und als Führung für die beweglichen Bleche dienen.

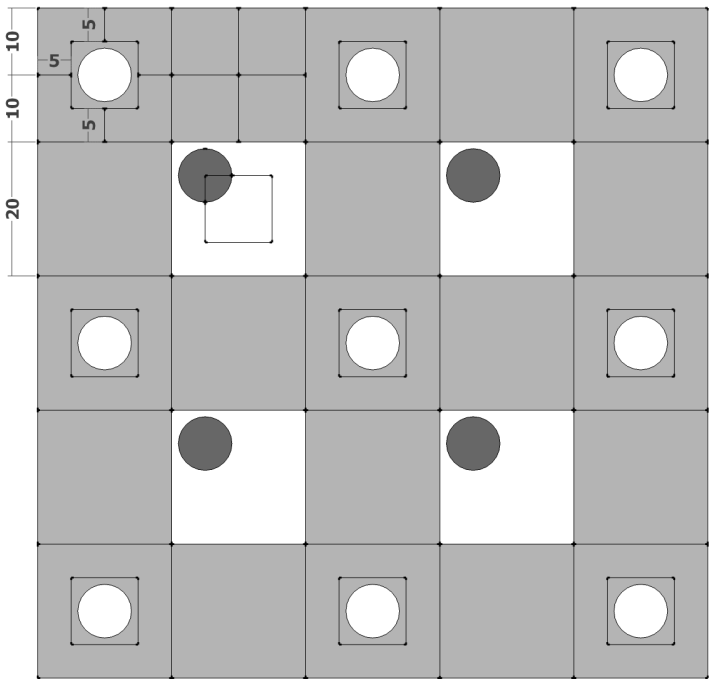


Abbildung 7: Ausschnitte im Grundlayout

4.1.2 Die Blechausschnitte

Die Formen der Blechausschnitte richten sich nach ihrer Funktion. In Abbildung 8 sind vier verschiedene Blechausschnitte zu sehen.

- Im ersten Ausschnitt kann der Stift eine komplette Taktrunde gehen ohne das Blech zu berühren.
- Im zweiten Ausschnitt kann sich der Stift nur in Takt II und IV frei bewegen. Wenn der Stift auch die Bewegungen der Takte I und III mitmacht, so wird er dabei das Blech mitnehmen.
- Im dritten Ausschnitt stößt der Stift in Takt I ebenfalls an das Blech und bewegt es mit, nimmt es aber in Takt III nicht wieder mit zurück.
- Im vierten Ausschnitt schließlich wird der Stift das Blech nur bewegen, wenn er sich rechts im Ausschnitt befindet. In der linken Position kann er sich frei in den Ausschnitt nach vorne bewegen. Zurücknehmen wird er das Blech aber egal ob er sich dann links oder rechts befindet.

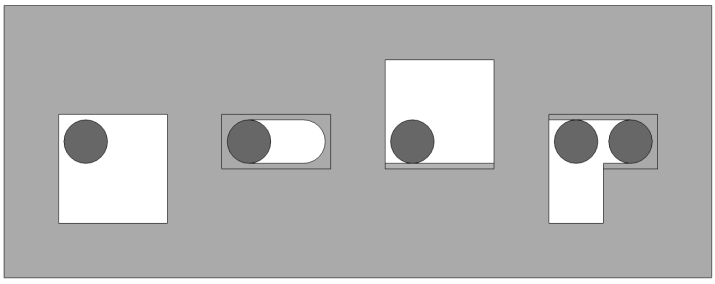


Abbildung 8: Blechausschnitte

In Abbildung 19 im Anhang sind alle im Additionswerk verwendeten Blechausschnitte dargestellt. Zu ihrer Organisation wurde ein Nummerierungssystem entworfen.

- Die ersten beiden Ziffern geben die Größe in Höhe mal Breite an, interpretiert in Schritten von 10 Einheiten.
- Eine 0 signalisiert einen eng umfassenden Ausschnitt.
- Bei den weiteren Stellen zeigt eine Ziffer zwischen 1 und 4 eine Sperrnase im jeweiligen Quadranten an.
- Die Ziffern 5 bis 8 stehen für eine berührend ausgeführte Kante, beginnend mit 5 für oben und gegen den Uhrzeigersinn folgend.
- Die Ziffer 9 ist für Ausschnittsmerkmale außerhalb dieses Systems reserviert.

Das System ist nicht eindeutig, reicht aber zur Benennung der verwendeten Ausschnitte aus.

Insbesondere ist nicht geklärt, welchen Unterschied es macht, ob bei einem Ausschnitt mit Sperrnase und berührender Kante die eng anliegende Ausbuchtung rund oder eckig ausgeführt ist (Unterschied zwischen Auschnitt 2227 und 2228, Abbildung 19). Bei Zuse existieren beide Ausführungen, allerdings nicht für die selbe Ausschnittsform. Es konnte nicht ermittelt werden, ob es einen funktionalen Unterschied zwischen den Varianten gibt. Deshalb übernimmt die Rekonstruktion dieses Detail unverändert aus Zuses Plänen.

Einige Ausschnitte sind offensichtlich für dickere Stifte gedacht. Auch dies richtet sich nach Zuses Plänen, wo die Impulsgeber und zwei, alle Ebenen verbindenden, statischen Stifte dicker ausgeführt sind. Für den Durchmesser dieser Stifte wurden 12 Einheiten gewählt. Außer einer visuellen Unterscheidung hat die Wahl eines dickeren Stiftes keine Auswirkungen.

4.1.3 Die Blechumrisse

Mit den vorangegangenen Überlegungen konnten die Blechumrisse auf dem Raster neu gezeichnet werden. Dazu wurde die Software SketchUp [21] verwendet. SketchUp ermöglicht exaktes Konstruieren im dreidimensionalen Raum und erlaubt dank einfacher Bedienung die schnelle Umsetzung von Änderungen an Blechformen. In Abbildung 9 sind die konstruierten Blechformen in SketchUp zu sehen.

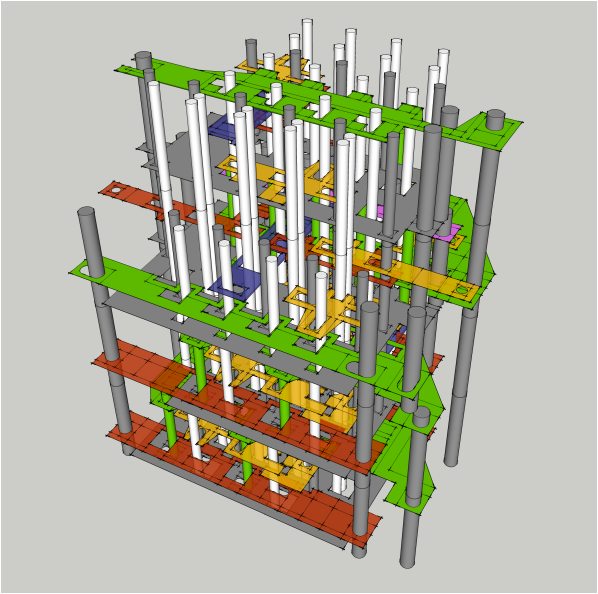


Abbildung 9: Konstruierte Blechformen in SketchUp

Aus den Plänen von Zuse wurden nur die ungefähren Abmessungen entnommen. Denn die genaue Lage der Linie ergibt sich in fast jedem Fall aus ihrer Funktion.

Dies soll am Beispiel von Blech C auf Ebene B erklärt werden und ist in Abbildung 10 visualisiert.

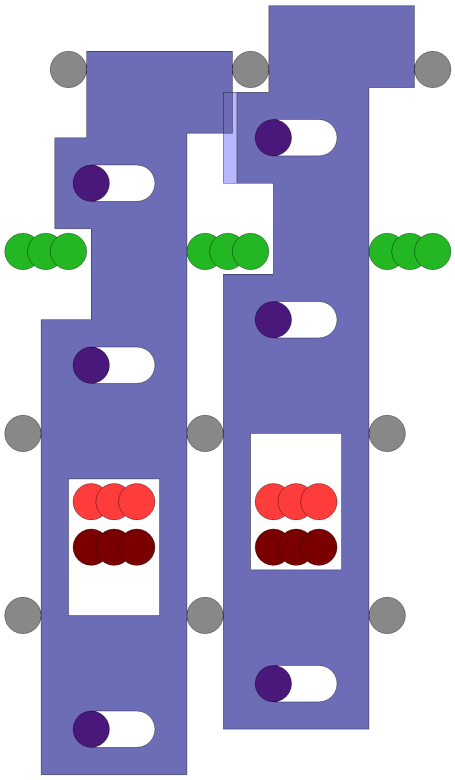


Abbildung 10: Die Rekonstruktion des Blech C

Die Position der Ausschnitte wird durch das Raster vorgegeben, ihre Form durch ihre Funktion. Der dritte Ausschnitt von oben beispielsweise ist so groß gewählt, weil er nicht mit denen sich ihn ihm befindenden Stiften interagiert. Bewegungen des blauen Bleches in y-Richtung als auch die Bewegungen des roten und des dunkelroten Stiftes in x-Richtung führen zu keiner Berührung. Die Breite des Bleches ergibt sich, weil die Seiten zur Führung des Bleches die statischen Stifte berühren müssen. Weil oben die statischen Stifte zum Grundlayout verschoben angelegt sind, ist auch das blaue Blech dort versetzt. Die untere Ausdehnung ergibt sich als Rand um den untersten Ausschnitt. Die Ausdehnung nach oben stellt sicher, dass das Blech bei seiner Bewegung in negativer y-Richtung nicht aus den statischen Stiften rutscht.

Weil sich neben dem Blech auch die grün markierten, beweglichen Stifte befinden, ist dort im Blech auf der linken Seite ein Einschnitt, der genau so groß ist, wie es die freie Bewegbarkeit der Stifte erfordert.

Über dem Einschnitt ist das Blech allerdings nicht so breit wie darunter, weil es sonst zur Kollision mit dem versetzten oberen Teil des benachbarten Bleches während dessen Bewegung in y-Richtung kommen könnte (hellblau markiert), wenn diese auf der gleichen oder benachbarten Schicht liegen. Im Originalplan von Zuse existiert diese Verjüngung nicht (Abbildung 16).

Es gibt nun verschiedene Möglichkeiten:

1. Bei Zuse waren die Bleche nicht auf der gleichen oder benachbarten Schicht, was aber im vertikalen Aufbau umständlich wäre.
2. Zuse hat an dieses Detail nicht gedacht und einen Fehler gemacht.

In diesem Fall lässt sich wohl nicht mehr klären, was der Wahrheit entspricht. Aber auch wenn man mit der Behauptung, dass Zuse einen Fehler gemacht hat, nur sehr vorsichtig umgeht, wird man in seinen Plänen des Additionswerkes Stellen finden, die ganz offensichtlich nicht funktionieren. Dies schmälert keinesfalls Zuses Leistung, denn die Konzepte hinter dem Additionswerk sind durchdacht und stimmig. Zudem stand ihm keine Software zum Testen seiner Pläne zur Verfügung, wie sie für diese Bachelorarbeit genutzt wurde. Zuse wird seine Pläne wahrscheinlich hauptsächlich im Kopf durchdacht haben, was leicht zu kleinen Unstimmigkeiten führen kann.

Im Folgenden sollen deshalb die Unklarheiten und Fehler, die mir aufgefallen sind, sowie meine Lösungen beschrieben werden. Denn genau wie Zuse sich an einigen Stellen geirrt hat, kann auch ich Dinge übersehen haben. Die Überlegungen hinter meinen Änderungen können hier nachgelesen und diskutiert werden.

4.1.4 Ebene A

Originalplan unter 7.1.1.

Blech A Es ist unklar warum der Einschnitt α eine Bewegung des Stiftes in y-Richtung zulassen würde, denn dies ist eigentlich nicht nötig. Das Detail wurde trotzdem übernommen. (Seite 34)

4.1.5 Ebene B

Originalplan unter 7.1.2.

Blech A Unklar ist, warum die schrägen Seiten in einem anderen Winkel ausgeführt sind als beispielsweise bei Blech A auf Ebene A. Im Sinne einer Vereinheitlichung wurden die Schrägen in der Rekonstruktion angepasst. (Seite 39)

Blech C Blech C wurde bereits unter 4.1.3 ausführlich beschrieben.

Blech F Die Nase β musste ergänzt werden. Sie ist nötig, damit von einer Stelle, an der beide Ziffern 1 waren, ein Übertrag an die nächste Stelle signalisiert werden kann. Des Weiteren fehlt der mittlere Ausschnitt in den originalen Plänen. (Seite 42)

Blech F0 und F4 Diese beiden Bleche sind nicht in den Plänen. Sie leiten sich aus der Form von Blech F ab. (Seite 42)

Blech N Auch zu diesem Blech existiert kein originaler Plan. Seine Aufgabe ist es, das Blech C zu bewegen, falls $a_i = 1$ und $b_i = 0$. Wurde das Blech A bewegt, so landet der Stift im eng anliegenden Ausschnitt γ auf Blech N und wird in negativer y-Richtung mitgenommen. (Seite 45)

Blech Q' Der Ausschnitt δ ist etwas unglücklich zustande gekommen. So wurde auf Ebene A bereits die Position der Impulsstifte festgelegt. Weil auf der Ebene B die oberste Reihe von statischen Stiften versetzt ist, wurde es an dieser Stelle eng. Der Ausschnitt δ musste deshalb an zwei Seiten eng anliegend ausgeführt werden, auch wenn dies bei Zuse nicht der Fall ist. (Seite 46)

4.1.6 Ebenen C und D

Originalplan unter 7.1.3. Zuse hat nur Pläne für die Ebene C gezeichnet, weil sich die Ebene D analog zu Ebene C verhält, nur dass die Elemente um eine Stelle versetzt sind.

Blech K In die Kerbe ε (Seite 50) kann eine Feder einrasten (siehe Abschnitt 3.3).

Blech M Zum Blech M wurde die Einbuchtung ζ hinzugefügt. Sie verhindert bei Bewegung in negativer y-Richtung einen Zusammenstoß mit der Feder für Blech K. Dies hängt natürlich davon ab, wie die Feder gestaltet ist. So wie Zuse die Feder in Fig. 60 [10] andeutet, könnte diese zwischen zwei Stiften gespannt sein. In diesem Fall ist die Einbuchtung ζ unbedingt nötig. (Seite 51)

Blech S Die Ausschnitte η sind keine führenden Ausschnitte. Es ist deshalb verwirrend, dass Zuse sie rund gezeichnet hat (Abbildung 17), was normalerweise nur eng anliegende Ausschnitte sind. Allerdings auch im Blech T (Abbildung 18) verwendet Zuse runde Ausschnitte, die nicht eng anliegend sind. Dort ist allerdings besser als in Blech S zu erkennen, dass diese Spielraum für die Stifte bieten.

Des Weiteren zeigen die Ausschnitte in die falsche Richtung. Den mittleren Ausschnitten und der Stellung des Impulsgebers zufolge, muss sich S in negativer x-Richtung bewegen können.

Es war ferner eine zusätzliche Einkerbung θ notwendig.

Blech U Die Ausschnitte in diesem Blech sind bei Zuse falsch gewählt.

4.2 Vertikaler Aufbau

Aus dem Kapitel der Patentschrift über das Additionswerk lassen sich nur wenige Informationen über den vertikalen Aufbau entnehmen. Die Anordnung der Ebenen ist beispielsweise festgehalten, allerdings existieren keine konstruktiven Gründe für diesen Aufbau. Das Additionswerk würde genauso funktionieren, wenn man die Ebenen in umgekehrter Reihenfolge aufeinandersetzen würde. Sogar gegen eine beliebig vertauschte Anordnung spricht nichts,

denn mit Ausnahme des Stiftes zur Bewegung von Blech V gehen alle Stifte entweder über die gesamte vertikale Länge oder sind auf eine Ebene beschränkt. Was letztendlich für die von Zuse gewählte Anordnung spricht, ist, dass sie den logischen Ablauf der Rechnung weitestgehend von oben nach unten durchführt, was der gewohnten Leserichtung entspricht.

Über die Schichtung der Bleche innerhalb einer Ebene lässt sich hingegen aus der Patentschrift nur wenig entnehmen. Lediglich aus den Draufsichten, in denen alle Bleche zusammen dargestellt sind (zum Beispiel [10, Fig. 55]), kann teilweise erkannt werden, dass ein Blech über einem anderen liegt, weil es Teile dessen verdeckt. Ob hinter dieser Information Überlegungen von Zuse stecken ist unklar. Wahrscheinlicher scheint, dass er die Schichtung in diesen Darstellungen danach gewählt hat, wie gut sie sich als Draufsicht zum Erkennen aller Teile eignen.

Auch mit einfachen logischen Überlegungen kommt man kaum auf Gründe, die für eine bestimmte Schichtung sprechen. Trotzdem ist dieser Punkt für einen realen Nachbau des Additionswerkes keinesfalls irrelevant. Wie bereits im Abschnitt 4.1.3 beschrieben, ergeben sich teilweise strengere Anforderungen an Blechformen, wenn diese auf einer Schicht liegen. Aber auch zwischen Blechen auf benachbarten Schichten kommt es zur Berührung und kann zu Verkantungen führen. Fast überall wo die Blechformen schräge Kanten aufweisen, geschieht dies aus diesem Grund. Denn so kann verhindert werden, dass eine gerade Kante in der Bewegung von einem darunterliegenden Blech rutscht und sich bei der Rückbewegung aufgrund der Schwerkraft nicht wieder einfach über das andere Blech schiebt [5, S. 51].

Ein weiterer interessanter Punkt ist am Nachbau der Z1 sichtbar. Man erkennt deutlich Bleche mit gebogenen Teilen (Abbildung 11). In der Patentschrift werden solche Fälle an keiner Stelle erwähnt.

Abbildung 11: Vertikaler Aufbau am Z1-Nachbau

Für die 3D-Simulation sind viele Überlegungen natürlich nachrangig, denn sie ist nicht der Schwerkraft ausgesetzt. So schweben die Bleche letztendlich in der Implementierung mit Abständen übereinander, ohne darauf zu achten, ob es an dieser Position darunterliegende Bleche gibt, die sie auf dieser Höhe halten könnten. Die Schichtung wurde in der 3D-Simulation vielmehr nach folgenden Gesichtspunkten angelegt:

- Die Ebene soll mit möglichst wenigen Blechschichten auskommen, um den Aufbau nicht visuell zu verkomplizieren. Der letztendliche Abstand, der zum verdeckungsfreien Betrachten eines Bleches nötig ist, kann trotzdem noch variiert werden, so wie dies auch in der Implementierung umgesetzt wurde.
- Daraus ergibt sich auch, dass gleiche Teile, wenn möglich, auf der selben oder benachbarten Schicht liegen sollten, was ebenfalls entscheidend für eine verständliche Ordnungsstruktur ist.
- Große Bleche sollten möglichst auf der untersten Schicht liegen, um im Fall der standardmäßig von oben gerichteten Betrachtungssicht andere Bleche nicht zu verdecken.

Einige weitere Aspekte des vertikalen Aufbaus solcher Schaltkonstrukte finden sich in anderen Kapiteln der Patentschrift, treffen aber ebenfalls auf das Additionswerk zu. So müssen laut Zuse, mit Ausnahme eines mittig liegenden Bleches, alle Bleche in zweifacher Ausführung vorhanden sein, und so übereinander angeordnet werden, dass sich ein spiegelsymmetrischer Aufbau ergibt. Ohne diesen symmetrischen Aufbau würde eine auf das Stiftende gerichtete Kraft diesen eher zum Kippen als zur geraden, horizontalen Bewegung bringen [9, 5]. Dieser Aspekt wurde in der 3D-Simulation nicht übernommen, denn eine Verdopplung der Blechanzahl würde die Übersichtlichkeit der Darstellung stark einschränken. Außerdem ist der symmetrische Aufbau in einer Simulation, die keinen physikalischen Störungen ausgesetzt ist, nicht nötig. Lediglich die Grundbleche wurden doppelt ausgeführt und belegen auf jeder Ebene die oberste und unterste Schicht.

Auch auf die Nachbildung weiterer Elemente des vertikalen Aufbaus, die sich um die Einbettung des Additionswerkes in das Gerüst der Maschine und die Generierung und Weiterleitung der Impulsbewegungen kümmern würden, wurde in der Simulation verzichtet.

Hinzugefügt wurden aber im Kapitel des Additionswerkes bei Zuse nicht beschriebene Zwischenebenen. Sie bilden in der Simulation eine visuelle Trennung der Ebenen. Aber auch physikalisch hätten sie eine Funktion: sie verhindern, dass ebenenspezifische Stifte in darunterliegende Ebenen rutschen können. In den Zwischenebenen sind deshalb nur Ausschnitte für die ebenenübergreifenden Verbindungsstifte (Abbildung 21).

5 Implementierung

In diesem Kapitel soll zunächst auf die verwendete Technologie eingegangen werden. Anschließend werden die umgesetzten Funktionen der 3D-Simulation beschrieben und danach wichtige Aspekte der Implementierung einzeln erörtert.

5.1 Technologie

Zur Umsetzung der 3D-Simulation wurde eine Technologie benötigt mit der sich animierte, interaktive 3D-Grafiken realisieren lassen. Des Weiteren schien es sinnvoll das Ergebnis auf einer Webseite präsentierbar zu machen, damit es vielen Menschen möglichst einfach zugänglich ist.

Natürlich wäre es möglich gewesen, eine Installationsdatei zum Herunterladen und Installieren anzubieten. Aber bereits das würde für viele Benutzer eine Hürde darstellen, die sie eher abschreckt. Insbesondere weil die zu implementierende Anwendung von den meisten Benutzern nicht viel mehr als aus einmaligem Interesse verwendete werden würde, wäre eine Installation eine unverhältnismäßige Forderung. Nicht zuletzt sind die Technologien für eine Umsetzung als Anwendung im Browser vorhanden, werden an vielen Stellen bereits eingesetzt und führen dazu, dass sich bei den Anwendern die Erwartungshaltung, gerade was Informationsangebote angeht, zunehmend in Richtung Webanwendung verschiebt.

Dreidimensionale Grafiken sind aber gerade einer der Punkte, die sich im Browser noch nicht durchsetzen konnten. Wesentliche Ursache dürfte sein, dass sich in der bestehenden Technologievielfalt bislang keine Lösung eine breite Akzeptanz sichern konnte. Im Folgenden sollen Vorteile und Nachteile der wichtigsten in Frage kommenden Technologien erläutert werden.

5.1.1 Flash

Nachdem Flash jahrelang die am häufigsten im Internet anzutreffende Lösung für animierte und interaktive Grafiken war [19], ist die Zukunft für dieses Format unsicher geworden. Was den Einsatz von Flash möglich machte, war die große Verbreitung des nötigen Browser-Plugins [11]. Allerdings nicht ohne Hindernisse, denn für Linux-Systeme existiert kein Version, die den gleichen Funktionsumfang wie die Windows-Version bietet [22]. Aber auch auf anderen Plattformen wird Flash für Ressourcenhunger, Instabilität und Sicherheitsprobleme kritisiert [17]. Weil es sich zudem nicht um ein offenes Format handelt, müsste die besitzende Firma Adobe die Fehler beseitigen, was sie aber oft nur unzureichend tut [20]. Performanceprobleme waren auch der offizielle Grund für Apple, ihre iPhone und iPad-Geräte nicht mit Flash-Playern auszustatten [16]. Nicht viel später kündigte dann Adobe an, Flash auch für andere mobile Geräte nicht mehr weiter zu entwickeln [15]. Inzwischen lassen sich die meisten Funktionen von Flash durch HTML5-Technologien ersetzen [24]. Flashs Potential für echte

3D-Grafiken wurde sowieso nur selten ausgenutzt, die meisten Flash-Anwendungen verwenden 2D-Animationen und Videomaterial.

5.1.2 Java-Applet

Java-Applets konnten sich nie richtig im Web durchsetzen [6]. Ursachen waren lange Ladezeiten, schlechtes Einblenden mit anderen Inhalten auf einer Webseite und die enorme Dateigröße des Plugins [6]. Trotzdem sprechen Statistiken meist für eine gute Verbreitung des Plugins beim Endanwender [2] und der Java-Unterbau verspricht eine große Plattformunabhänigikeit, die aber in der Realität oft nicht gehalten werden kann[1]. Das Java-Applets Hardwarebeschleunigung unterstützen war lange Zeit ein Vorteil, inzwischen bieten aber auch Canvas[2] und WebGL diese Funktionalität [26]. Nicht vergessen werden darf schließlich, dass aufgrund von Sicherheitsproblemen den Anwendern zur Zeit davon abgeraten wird jegliche Java-Applets auszuführen [18].

5.1.3 WebGL

Für die Umsetzung dieses Projektes fiel die Entscheidung letztendlich auf die Technologie WebGL. Dass mag angesichts der aufgeführten Nachteile der beiden vorangegangenen Technologien kaum überraschen, aber auch WebGL ist nicht unproblematisch.

WebGL gehört zur Familie der HTML5-Technologien, ist aber kein Teil der W3C-Standards [25], sondern wird von der Khronos Group als lizenzfreier Standard entwickelt. Es bietet eine Schnittstelle um im Webbrowser OpenGL ES[3] anzusprechen [1]. Zu den Befürwortern von WebGL gehören viele große Firmen. Google hat beispielsweise die Entwicklung eines eigenen Plugins für 3D-Grafiken zu Gunsten von WebGL eingestellt.

WebGL wird von den meisten Browsern[4] in ihrer aktuellen Version unterstützt [13]. Im Unterschied zu den vorher genannten Techologien ist die Unterstützung im Browser nativ, das heißt es ist kein Plugin nötig. Eine Ausnahme bildet der Internet Explorer, weil von Microsoft bisher keine Pläne zur WebGL-Integration bekannt sind. Es existieren aber Plugins um den Internet Explorer mit WebGL aufzurüsten[5]. Nachteilig bleibt, dass diese Plugins wahrscheinlich nicht die starke Verbreitung wie der Flash-Player und die Runtime für Java-Applets finden werden. Die Anzahl der Nutzer mit nativer WebGL Unterstützung wird sich hingegen kontinuierlich erhöhen, wenn diese auf eine aktuelle Version ihres Browsers umsteigen. Für

[1] Beispielsweise die von Kostulski [2] umgesetzte 3D-Simulation des Z1-Addierers konnte vom Autor nicht unter Ubuntu ausgeführt werden und auch andere Benutzer berichteten von Problemen [14].

[2] HTML5-Element für programmierbare Bitmap-Grafiken

[3] *Open Graphics Library for Embedded Systems* ist eine vereinfachte Version der OpenGL-Spezifikation zur hardwarebeschleunigten Berechnung von 2D- und 3D-Grafiken

[4] Firefox, Safari, Chrome, Opera und Browser auf Basis der WebKit- oder Gecko-Engine

[5] zum Beispiel Google Chrome Frame oder IEWebGL

kommerzielle Webseiten ist dies zur Zeit verständlicherweise Grund genug auf WebGL zu verzichten, weil sie es sich nicht leisten können, die Nutzer mit veralteten Browsern von ihren Angeboten auszuschließen. Im Rahmen von universitären Projekten wie der vorliegenden Arbeit ist WebGL aber bereits einsetzbar und kann dabei helfen den Standard beim Anwender populär zu machen.

5.1.4 Three.js

Weil WebGL in seiner reinen Form eine Low-Level-API[6] ist, empfielt sich der Einsatz einer Bibliothek. Dadurch kann vermieden werden, einige grundlegende Funktionen selber implementieren zu müssen. Der Zugriff auf die reinen OpenGL ES Funktionen bleibt, falls benötigt, weiterhin möglich. Für die vorliegende Arbeit wurde die Bibliothek Three.js [12] verwendet. Es handelt sich dabei um eine populäre WebGL-Bibliothek in JavaScript[7]. Sie wird inzwischen in Büchern über WebGL behandelt [3] und fand in einer Reihe von Projekten Verwendung.

Ein paar Schwierigkeiten bei der Umsetzung des Projektes mit Three.js entstanden, weil die Bibliothek zur Zeit sehr aktiv weiterentwickelt wird. Dabei kam es auch zu Änderungen an der API der Bibliothek. Bei Aktualisierung der verwendeten Bibliothek auf neue Versionen waren einige Nacharbeiten in der Simulation nötig. Zu diesen Problemen kam es insbesondere auch deshalb, weil ich Bibliotheksklassen mit eigenen Funktionen erweiterte. Diese mussten dann auch eigenständig weitergepflegt werden. Sinnvollerweise könnten diese Erweiterungen zur Aufnahme in die Bibliothek eingereicht werden. Dazu müssten sie aber erst noch stärker verallgemeinert werden, als dies im Rahmen dieses Projektes erfolgte. Für die Weiterführung der Simulation ergeben sich aus solchen Abhängigkeiten zunächst keine Probleme, denn die Bibliothek kann in alten Versionen weiter benutzt werden. Eine Aktualisierung der Bibliothek ist nur notwendig, wenn man neue Funktionen aus Updates der Bibliothek nutzen möchte.

5.2 Funktionen

Folgende Funktionen wurden für die 3D-Simulation umgesetzt:

5.2.1 3D-Ansicht

- Das Additionswerk wurde als 3D-Modell aufgebaut. In Abbildung 12 ist das Additionswerk in WebGL zu sehen.

[6] Schnittstelle, die nur wenig Abstraktion von der Hardwareimplementierung bietet.
[7] Skriptsprache, die von allen gängigen Browsern verstanden wird.

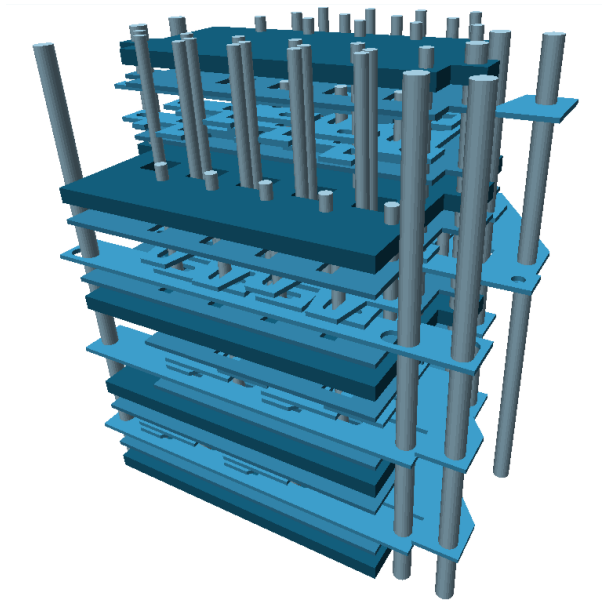


Abbildung 12: Das 3D-Modell des Additionswerkes in WebGL

- Jedes Element ist ein dreidimensionaler Körper mit Tiefe. Die Blechformen werden aus SVG-Dateien geladen, extrudiert[8] und trianguliert[9].
- Das Modell lässt sich Drehen, Verschieben und Vergrößern. Die Ansicht kann zurückgesetzt werden.

5.2.2 Steuerung

- Die Eingabe für einen Rechenvorgang umfasst zwei vierstellige, binäre Zahlen und den Übertrag an der Stelle 0.
- Als Ausgabe erhält man das vierstellige Ergebnis und eine Information, falls ein Stellenüberlauf auftrat.
- Die Rechenschritte werden animiert dargestellt und die bewegten Teile farblich markiert.
- Die farbliche Markierung der bewegten Teile ist auch vor der Bewegung als Vorschau möglich.
- Bewegungen können wiederholt werden oder rückgängig gemacht werden.

[8] aus einer Fläche einen Körper mit Tiefe erzeugen
[9] Approximation eines Körpers aus Dreiecksflächen

5.2.3 Ausschnitte

- Es lassen sich rechteckige Ausschnitte des 3D-Modells anzeigen. An den Grenzen des Ausschnitts werden Querschnitte durch das Modell angezeigt. Vergleiche dazu Abbildung 14.
- In der Höhe ist das Ausblenden von Ebenen möglich.
- Der Ausschnitt kann durch Klick auf eine Auswahlbox im Modell verändert werden.

5.2.4 Logischer Schaltplan

- Ein logischer Schaltplan kann eingeblendet und in seiner Größe verändert werden. Der Schaltplan ist im Anhang unter Abbildung 20 abgebildet.
- Zu den Logikgattern auf dem Schaltplan lassen sich die zugehörigen Ausschnitte des 3D-Modells anzeigen.

5.2.5 Darstellungsoptionen

- Die Elemente des 3D-Modells sind in Klassen nach beweglich/statisch und Blech/Stift geteilt. Für jede Elementklasse kann ein Transparenzwert eingestellt oder eine komplette Ausblendung erreicht werden.
- Die vertikalen Abstände innerhalb einer Ebene sind veränderbar, um bessere Einsichten in das Modell zu gewähren.
- Es lässt sich eine Funktion aktivieren, die eine farbliche Markierung von Elementen bei Mauskontakt erlaubt.

5.3 Aufbau des 3D-Modells

Ein wichtiger Aspekt der Implementierung ist, wie die rekonstruierten Pläne in ein 3D-Modell umgesetzt werden. Eine durchaus übliche Herangehensweise wäre, die Blechformen in eine 3D-Modellierungs-Software zu laden. Dort können diese extrudiert und als Drahtgittermodell exportiert werden. Nachteilig an dieser Lösung ist, dass bei Änderungen am Blechumriss der komplette Arbeitsablauf wiederholt werden muss. Zudem schränkt es bei Weiterführung des Projektes die Mitarbeit von Personen, denen die verwendete Software oder die Kenntnisse in ihrer Bedienung nicht zur Verfügung stehen, ein. Aus diesen Gründen wurde eine andere Herangehensweise gewählt.

Die Blechformen aus den Rekonstruktionsplänen wurden als SVG-Dateien angelegt. SVG ist ein offenes Dateiformat für Vektorgrafiken. Es kann, ohne das spezielle Software nötig wäre, in modernen Browsern angezeigt und mit einem Texteditor bearbeitet werden. Aus den

Blechformen in den SVG-Dateien werden durch die Bibliothek Three.js Gitternetzmodelle mit Tiefe erzeugt. Allerdings enthält Three.js keine Import-Funktion für SVG-Dateien. Formen, die in Three.js extrudiert werden sollen, müssen mit eigenen Funktionen von Three.js angelegt werden. Um diese Lücke zu schließen, wurde ein rudimentärer SVG-Importer implementiert. Aufgrund der sehr klaren Anforderungen war es möglich sich auf einen sehr kleinen, benötigten Funktionsumfang des Importers zu beschränken. Deshalb können im Endeffekt nur SVG-Formen importiert werden, die sich genau an den immer gleichen Aufbau der Blechformen halten.

Ein weiteres Problem stellte die Triangulation der Objekte dar. Hierfür wurden bereits in Three.js vorhandene Methoden genutzt. Weil Triangulation aber keine triviale Angelegenheit ist, kam es dabei immer wieder zu Fehlern, also Formen bei denen der Triangulationsalgorithmus fehlschlug. Abhilfe schaffte in diesem Fall die langen, geraden Strecken der Blechformen beim Import aus mehreren kleineren Strecken zusammenzusetzen, wie in Abbildung 13 zu sehen. Die so entstanden neuen Punkte werden vom Triangulationsalgorithmus als Eckpunkte im Dreiecksgitter benutzt und vereinfachen die Triangulation. Schlug die Triangulation trotzdem fehl, half es meistens den Abstand der Unterteilungen zu verändern. Der gewählte Abstand ist als Attribut `stepSize` des Path-Elementes in den SVG-Dateien gespeichert.

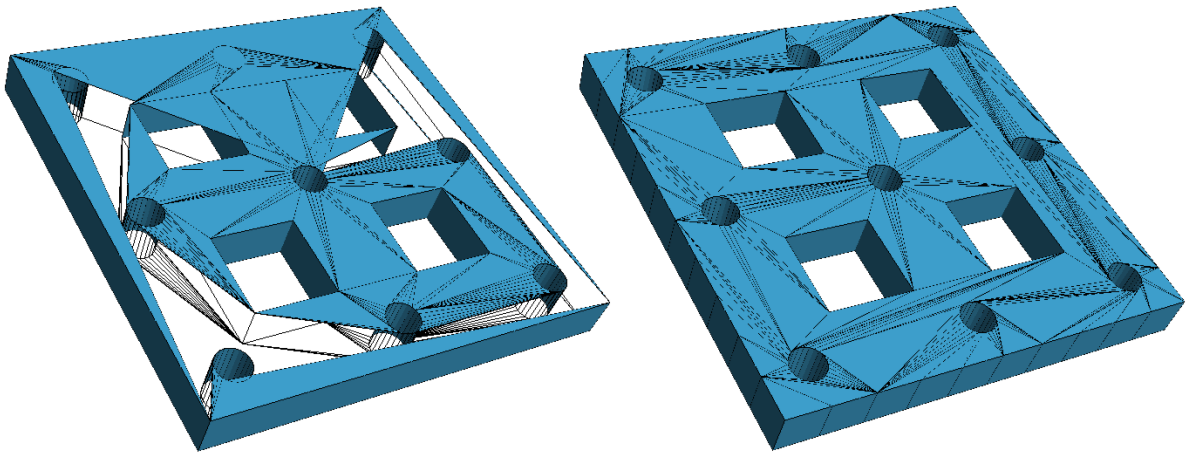


Abbildung 13: Fehlgeschlagene und fehlerfreie Triangulation

5.4 Shader zum Anzeigen von Ausschnitten

Eine wesentliche Funktion der 3D-Simulation ist, dass sich Ausschnitte des Additionswerkes anzeigen lassen, während der Rest ausgeblendet wird. Dazu können Schnitte durch die Geometrien gelegt werden, was Einblicke gewährt, die an einem realen Modell nicht möglich wären. Um dies zu ermöglichen, musste ein Weg gefunden werden, die Blechobjekte nur teilweise anzeigen zu können.

Eine simple Lösung an dieser Stelle wäre, dafür die Geometrie der Gitternetze zu verändern, zum Beispiel durch Boolesche Intersektion[10], was aber viele Nachteile enthält. So wurde bereits erwähnt, dass die Triangulation fehleranfällig ist, und nur weil stets dieselben Formen trianguliert werden zuverlässig funktioniert. Müssten in diesem Schritt neue Gitternetze berechnet werden, so könnten Fehler bei der Triangulation nicht mehr ausgeschlossen werden. Außerdem ist jede reale Änderung der Gitternetzgeometrie rechenintensiv. Sowohl im Schritt der Bearbeitung selbst, als auch für die Rendering-Pipeline, weil Buffer neu beschrieben werden müssen.

Eleganter ist es diese Funktion mit einem Shader[11] zu implementieren. Die Geometrien brauchen dann nicht mehr verändert werden, es entscheidet lediglich der Shader was gezeichnet wird und was nicht. Aufgrund der schnellen Berechnungsgeschwindigkeit von Shadern kann die Ausschnittsveränderung dann auch animiert dargestellt werden. Der Shader wurde als Pixel-Shader implementiert und vergleicht die Position jedes von ihm zu zeichnenden Pixels mit einem Auswahlrahmen, der ihm über Shader-Variablen überreicht wird. Des Weiteren können für den ausgewählten Teil und den nicht ausgewählten Teil Transparenzen festgelegt werden. Sinnvollerweise ist der nicht ausgewählte Teil immer transparenter als der ausgewählte Teil. Weil der Transparenzwert für jede Instanz des Shaders eine eigene Variable ist, können mit unterschiedlichen Materialien belegte Objektfamilien unterschiedlich transparent gezeichnet werden. Die Variablen für den Auswahlrahmen hingegen existieren nur einmal, wodurch sich das Festlegen einer Auswahl automatisch auf alle Elemente auswirkt.

Ein Nachteil des Pixel-Shaders ist, dass an der Auswahlgrenze die Elemente hohl angezeigt werden. Denn natürlich sind die Gitternetze der Objekte nur eine Hülle. Für den visuellen Eindruck ist dies nicht optimal, denn der Betrachter würde erwarten, dass die Dicke eines Bleches wie in der Realität mit Material gefüllt ist. Möglicherweise könnte dies durch den Geometrie-Shader, der in der Verarbeitungsabfolge vor dem Pixel-Shader liegt, simuliert werden. Dies wäre ein Punkt, der möglicherweise später umgesetzt werden kann. Den Rahmen der vorliegenden Arbeit hätte dies aber überschritten. Stattdessen wurde zum Shader noch eine Funktion ergänzt, mit der sich der Übergang an der Auswahlgrenze langsam ausblenden lässt, was ebenfalls den visuellen Eindruck verbessert, zu sehen in Abbildung 14.

5.5 Simulation der Bewegungsabläufe

Welche Elemente in einem Takt bewegt werden, legt eine Reihe von Regeln über Bewegungszusammenhänge fest. Diese bestehen aus Informationen darüber, welche Elemente von einem Element in einem bestimmten Takt angeschoben werden und unter welchen Bedingungen dies geschieht. In jedem Takt brauchen nun nur die Impulsgeber angestoßen zu werden und die Bewegung setzt sich automatisch fort. Dies erfordert allerdings äußerste Korrektheit der Bewegungsregeln. Einfacher wäre es, die Logik hinter dem Addierer nachzubauen und jedes

[10] Anwendung des Booleschen Und-Operators auf Körper.
[11] Ein auf dem Grafikprozessor ausgeführtes Programm zur Berechnung der Darstellung von Objekten.

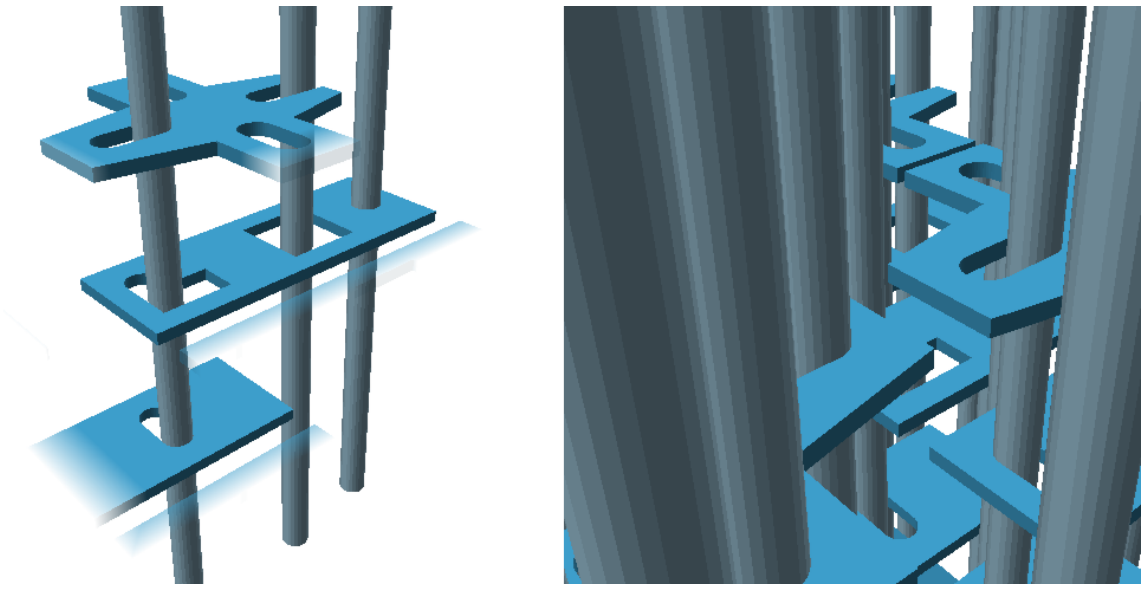


Abbildung 14: Anzeige einer Auswahl und gleiche Ansicht ohne Ausblendung

Element seiner logischen Gleichung entsprechend zu animieren. Einzelne Fehler beim Aufstellen der Regeln führen dann nicht mehr zu einem falschen Endergebnis, sondern brechen die Simulation nur lokal an einer Stelle. Die aufwändigere Simulation über Bewegungsregeln bietet aber den Vorteil Informationen über die Anstoßreihenfolge zu generieren. In der umgesetzten Simulation sind diese Bewegungsabläufe also enthalten, werden aber nicht angezeigt. Hier eine Möglichkeit zu finden, wie dargestellt werden kann, welches Elemente ein anderes anschiebt, wäre eine interessanter Punkt zur Weiterentwicklung der Simulation.

6 Fazit und Ausblick

Mit der vorliegenden Arbeit konnte eine funktionierende Simulation des Additionswerkes der Z1 umgesetzt werden. Diese erfüllt ihren Nutzen zunächst als Anschauungsmaterial zur Z1, deren Nachbau nicht mehr in Bewegung gesehen werden kann.

Wichtiger Antrieb dieses Projektes war jedoch darüber hinaus die Funktionsweise der mechanischen Schaltungen verstehbar zu machen. Dazu wurden Funktionen umgesetzt, welche ein realer Nachbau nicht bereitstellen kann. Im Wesentlichen sind dies das farbige Markieren von Elementen während der Bewegung und als Vorschau, das Anzeigen von Ausschnitten der Maschine, das Ausblenden von Elementen, das Expandieren von Ebenen und die Verknüpfung mit einem logischen Schaltplan. Einige dieser Funktionen wurden durch Reaktionen auf die Simulation von Kostulski [2] angeregt. Genauso wird auch das Benutzer-Feedback zu dieser Simulation möglicherweise zu weiteren Funktionen führen. Einige Vorschläge wurden bereits in dieser Arbeit gemacht, beispielsweise die Anstoßreihenfolge durchlaufbar zu machen.

Ferner fehlen in der Simulation bislang noch zwei Komponenten aus der Patentschrift. Dies sind zum einen die Federn, die eine fälschliche Bewegung von Blech K verhindern sollen. Zum anderen fehlt eine Realisierung der steuerbaren Rückübertragung des Additionsergebnisses auf die Eingabe B. Die Bleche hierfür sind bereits vorhanden, die Bewegungsregeln sind aber noch nicht ausgearbeitet und es fehlt ein Konzept, wie diese Funktion zu den Eingabeelementen hinzugefügt werden kann, ohne Verwirrungen beim Benutzer zu verursachen.

Neben solchen funktionalen Erweiterungen kommen noch mögliche Verbesserungen der Benutzeroberfläche hinzu. Über die bislang verwendeten Eingabeelemente unter Benutzung der jQuery UI Bibliothek bin ich zunehmend unzufrieden. Eine Überarbeitung des Bedienkonzeptes dürfte daher einer der dringlichsten Punkte sein. In diesem Zusammenhang ist eine Optimierung für Touch-Bedienung in Erwägung. Dadurch könnten die Einsatzgebiete der Simulation erweitert werden, insbesondere eine mögliche Präsentation im Museum.

Nicht zuletzt stellt das Additionswerk nur einen kleinen Teil der Z1 dar und könnte um weitere Module ergänzt werden.

Zur Präsentation der Simulation wurde die Domain http://www.zuse-z1.de registriert. Es ist geplant den Quellcode als Open-Source-Projekt auf GitHub zu veröffentlichen.

7 Anhang

7.1 Originalpläne

7.1.1 Ebene A

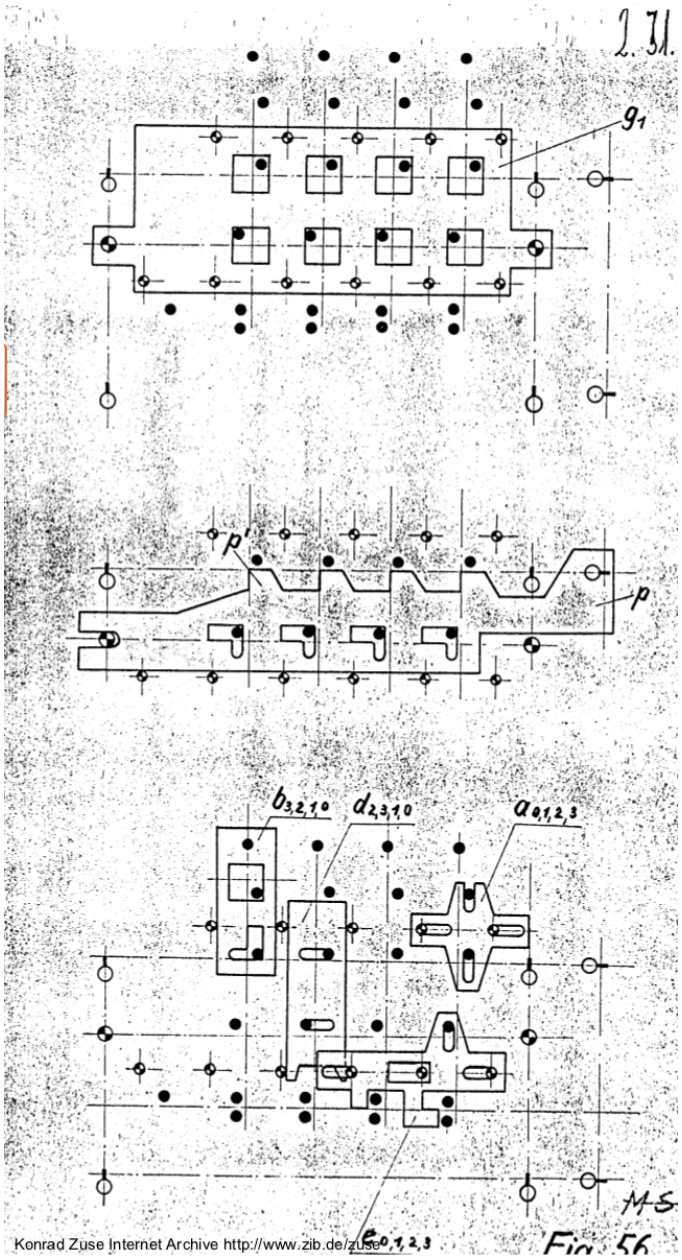


Abbildung 15: Zuses Originalplan der Ebene A [10, Fig. 56]

7.1.2 Ebene B

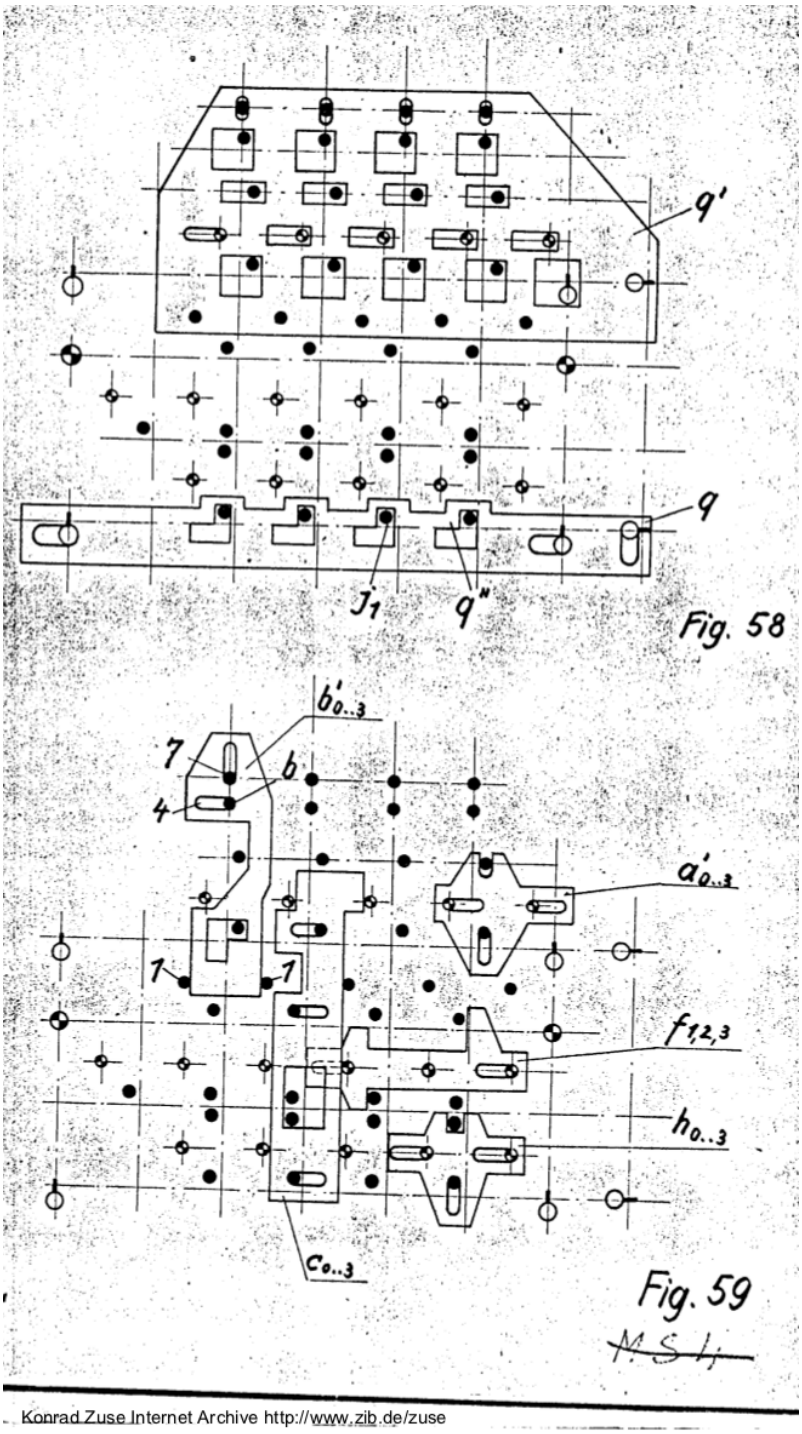


Abbildung 16: Zuses Originalplan der Ebene B [10, Fig. 58, 59]

7.1.3 Ebene C

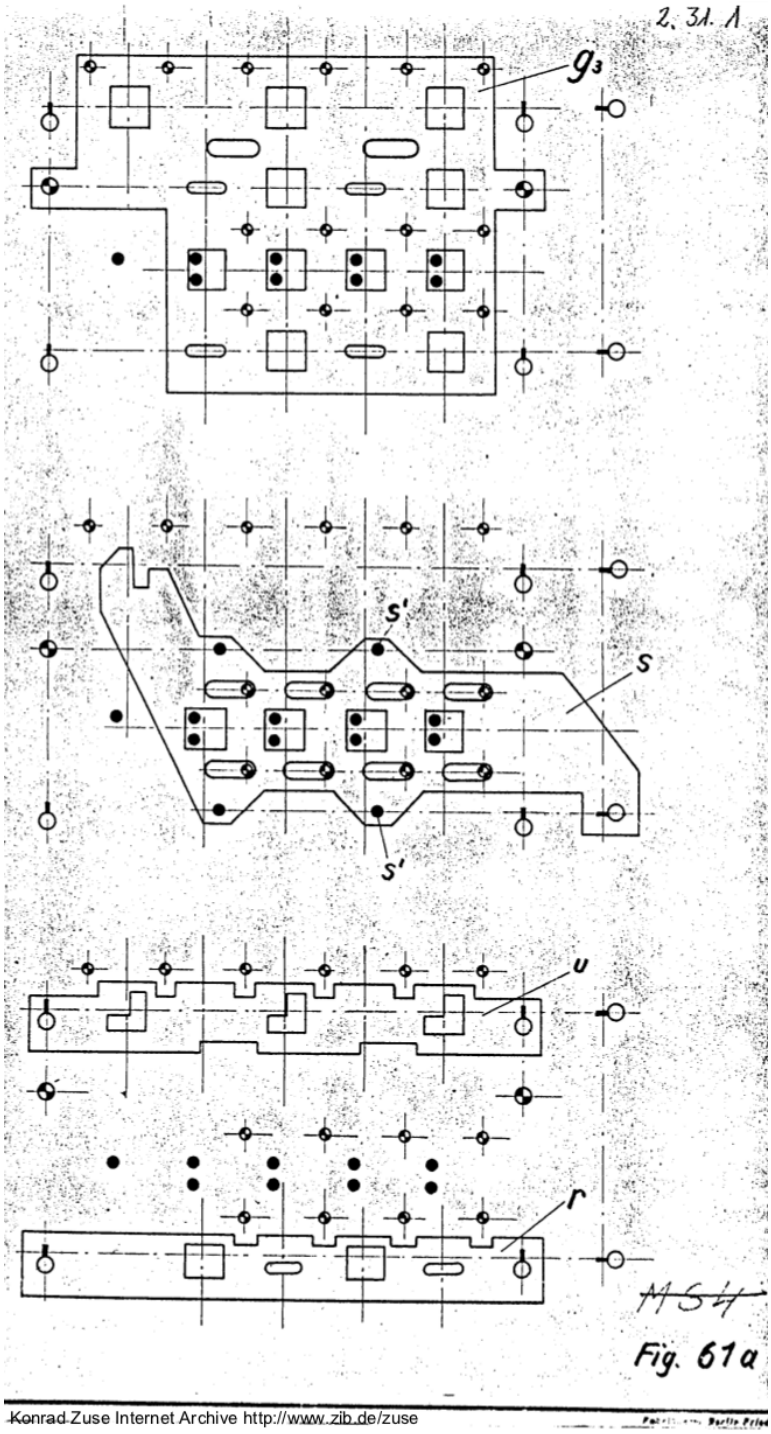


Abbildung 17: Zuses Originalplan der Ebene C, Teil 1 [10, Fig. 61a]

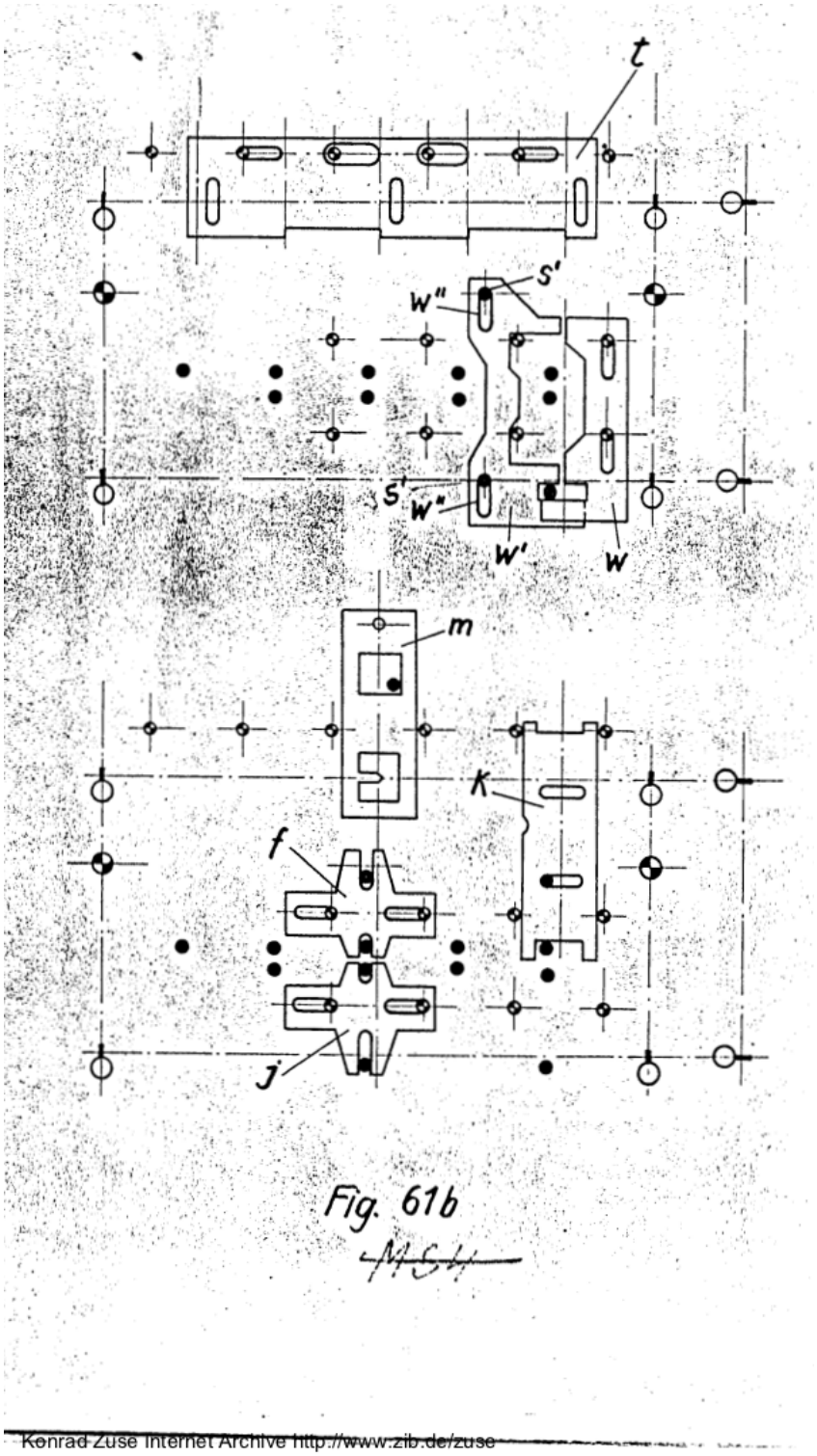


Abbildung 18: Zuses Originalplan der Ebene C, Teil 2 [10, Fig. 61b]

7.2 Rekonstruktionspläne

7.2.1 Blechausschnitte

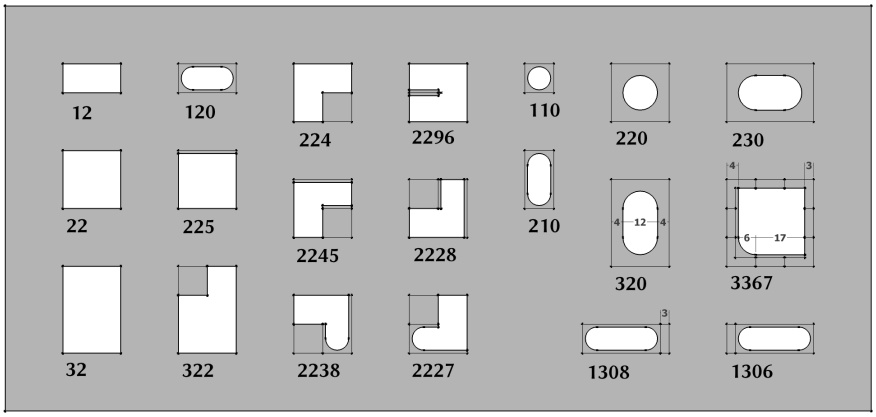


Abbildung 19: Alle Arten von Blechausschnitten im Additionswerk

Nummer	SVG-Pfad
110	`<circle cx="5" cy="5" r="4"/>`
220	`<circle cx="10" cy="10" r="6"/>`
12	`<path d="M 0,0 l 20,0 0,10 -20,0 0,-10 Z"/>`
22	`<path d="M 0,0 l 20,0 0,20 -20,0 0,-20 Z"/>`
32	`<path d="M 0,0 l 20,0 0,30 -20,0 0,-30 Z"/>`
120	`<path d="M 5,1 l 10,0 a 4,4,0,0,1,0,8 l -10,0 a 4,4,0,0,1,0,-8 Z"/>`
210	`<path d="M 1,15 l 0,-10 a 4,4,0,0,1,8,0 l 0,10 a 4,4,0,0,1,-8,0 Z"/>`
224	`<path d="M 0,0 l 10,0 0,10 10,0 0,10 -20,0 0,-20 Z"/>`
225	`<path d="M 0,0 l 20,0 0,19 -20,0 0,-19 Z"/>`
230	`<path d="M 10,4 l 10,0 a 6,6,0,0,1,0,12 l -10,0 a 6,6,0,0,1,0,-12 Z"/>`
320	`<path d="M 4,20 l 0,-10 a 6,6,0,0,1,12,0 l 0,10 a 6,6,0,0,1,-12,0 Z"/>`
322	`<path d="M 0,0 l 20,0 0,30 -10,0 0,-10 -10,0 0,-20 Z"/>`
1306	`<path d="M 8,1 l 17,0 a 4,4,0,0,1,0,8 l -17,0 a 4,4,0,0,1,0,-8 Z"/>`
1308	`<path d="M 5,1 l 17,0 a 4,4,0,0,1,0,8 l -17,0 a 4,4,0,0,1,0,-8 Z"/>`
2227	`<path d="M 5,1 l 15,0 0,19 -10,0 0,-11 -5,0 a 4,4,0,0,1,0,-8 Z"/>`
2228	`<path d="M 0,0 l 19,0 0,20 -8,0 0,-10 -11,0 0,-10 Z"/>`
2238	`<path d="M 0,10 l 11,0 0,-5 a 4,4,0,0,1,8,0 l 0,15 -19,0 0,-10 Z"/>`
2245	`<path d="M 0,0 l 10,0 0,11 10,0 0,8 -20,0 0,-19 Z"/>`
2296	`<path d="M 0,0 l 20,0 0,20 -20,0 0,-9 10,0 1,-1 -1,-1 -10,0 0,-9 Z"/>`
3367	`<path d="M 10,4 l 17,0 0,23 -23,0 0,-17 a 6,6,0,0,1,6,-6 Z"/>`

Tabelle 1: SVG-Pfade der Blechausschnitte

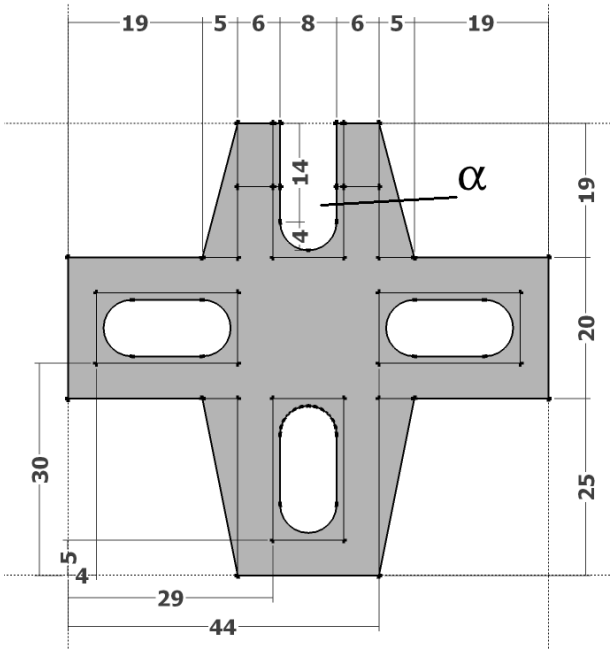
7.2.2 Blechformen

- Die Position ist immer für die linke, untere Ecke des umfassenden Rahmen der Blechform angegeben.
- Die Notation 171 ↔ 181 bedeutet, dass sich das Blech zwischen diesen Positionen hin und her bewegen kann.

Ebene A, Blech A

Instanzen:

Nummer	Position x,	y	Schicht
0	171 ↔ 181,	125	4
1	131 ↔ 141,	125	3
2	91 ↔ 101,	125	4
3	51 ↔ 61,	125	3



SVG-Pfad:

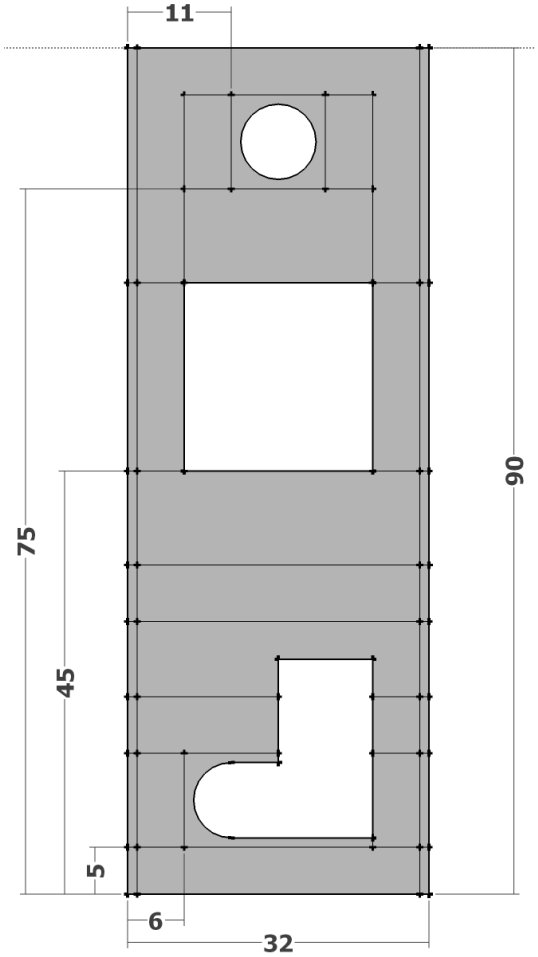
```
<path d="M 0,25 l 0,20 19,0 5,19 6,0 0,-14 a 4,4,0,0,1,8,0 l 0,14 6,0 5,-19 19,0
0,-20 -19,0 -5,-25 -20,0 -5,25 -19,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y
210	29,	5
120	4,	30
120	44,	30

Ebene A, Blech B

	Nummer	Position x,	y	Schicht
Instanzen:	0	194,	135 ↔ 125	2
	1	154,	135 ↔ 125	2
	2	114,	135 ↔ 125	2
	3	74,	135 ↔ 125	2



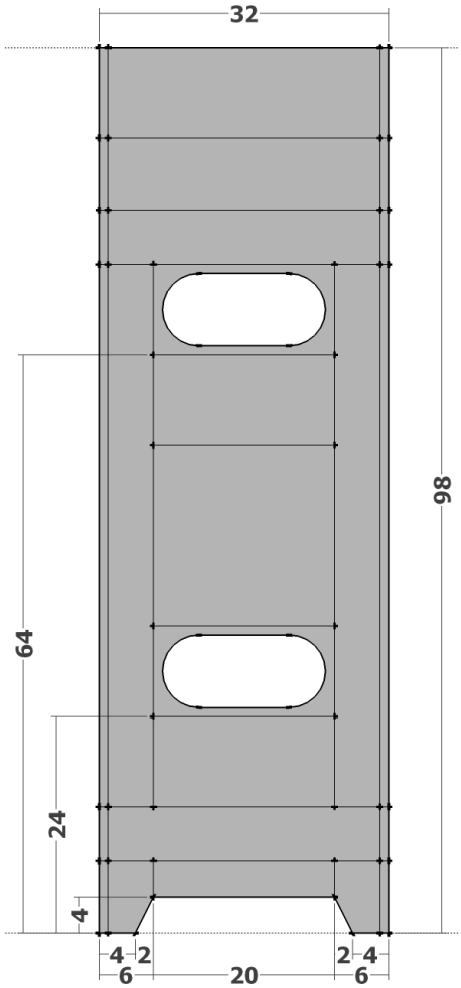
SVG-Pfad: `<path d="M 0,0 l 0,90 32,0 0,-90 -32,0 Z"/>`

Ausschnitte:

Nummer	Position x,	y
2227	6,	5
22	6,	45
110	11,	75

Ebene A, Blech D

	Nummer	Position x,	y	Schicht
	0	194,	76 ↔ 66	1
Instanzen:	1	154,	76 ↔ 66	1
	2	114,	76 ↔ 66	1
	3	74,	76 ↔ 66	1



SVG-Pfad: `<path d="M 0,0 l 0,98 32,0 0,-98 -4,0 -2,4 -20,0 -2,-4 -4,0 Z"/>`

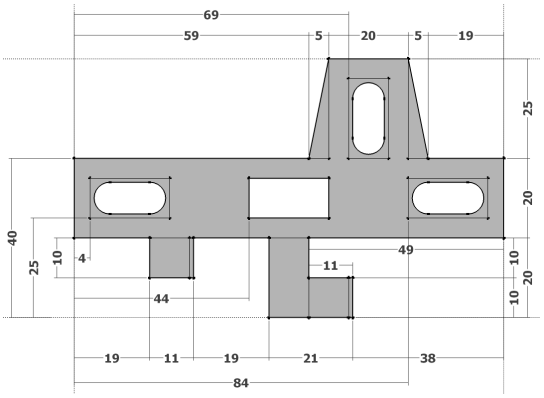
Ausschnitte:

Nummer	Position x,	y
120	6,	24
120	6,	64

Ebene A, Blech E

Instanzen:

Nummer	Position x,	y	Schicht
0	131 ↔ 141,	50	2
1	91 ↔ 101,	50	3
2	51 ↔ 61,	50	4
3	11 ↔ 21,	50	2



SVG-Pfad:

```
<path d="M 0,20 l 0,20 59,0 5,25 20,0 5,-25 19,0 0,-20 -49,0 0,-10 11,0 0,-10 -21,0
0,20 -19,0 0,-10 -11,0 0,10 -19,0 Z"/>
```

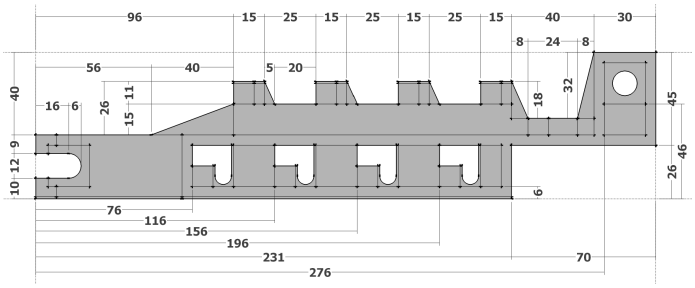
Ausschnitte:

Nummer	Position x,	y
120	4,	25
12	44,	25
120	84,	25
210	69,	40

Ebene A, Blech P

Instanzen:

Nummer	Position x,	y	Schicht
0	-6 ↔ 4,	84	5



SVG-Pfad:

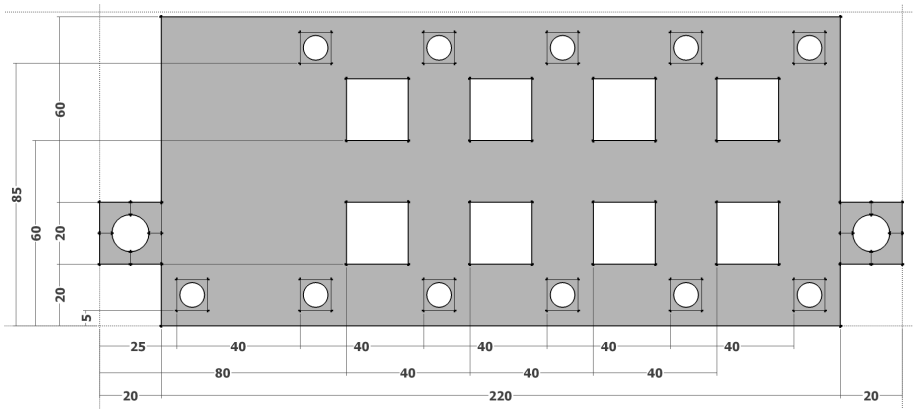
```
<path d="M 0,0 l 0,10 16,0 a 6,6,0,0,1,0,12 l -16,0 0,9 56,0 40,15 0,11 15,0 5,-11
20,0 0,11 15,0 5,-11 20,0 0,11 15,0 5,-11 20,0 0,11 15,0 8,-18 24,0 8,32 30,0 0,-45
-70,0 0,-26 -231,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y
2238	76,	6
2238	116,	6
2238	156,	6
2238	196,	6
220	276,	46

Ebene A, Blech G

	Nummer	Position x,	y	Schicht
Instanzen:	0	0,	70	0
	1	0,	70	6



SVG-Pfad: `<path d="M 20,0 l 0,20 -20,0 0,20 20,0 0,60 220,0 0,-60 20,0 0,-20 -20,0 0,-20 -220,0 Z"/>`

Ausschnitte:

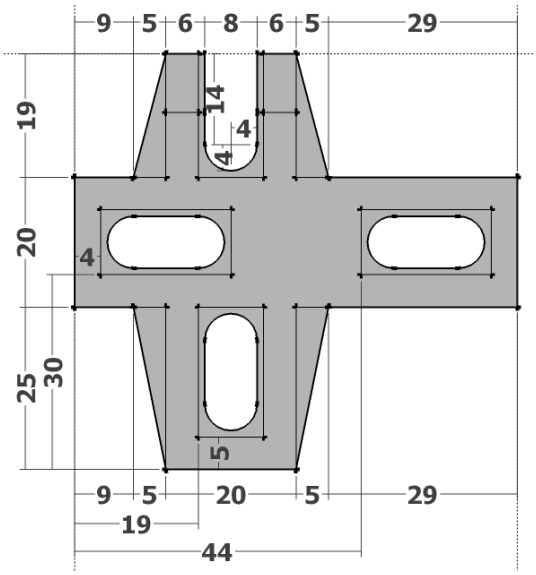
Nummer	Position x,	y
110	25,	5
110	65,	5
110	105,	5
110	145,	5
110	185,	5
110	225,	5
220	0,	20
22	80,	20
22	120,	20
22	160,	20
22	200,	20

Nummer	Position x,	y
220	240,	20
22	80,	60
22	120,	60
22	160,	60
22	200,	60
110	65,	85
110	105,	85
110	145,	85
110	185,	85
110	225,	85

Ebene B, Blech A

Instanzen:

Nummer	Position x,	y	Schicht
0	181 ↔ 191,	125	5
1	141 ↔ 151,	125	4
2	101 ↔ 111,	125	5
3	61 ↔ 71,	125	4



SVG-Pfad:

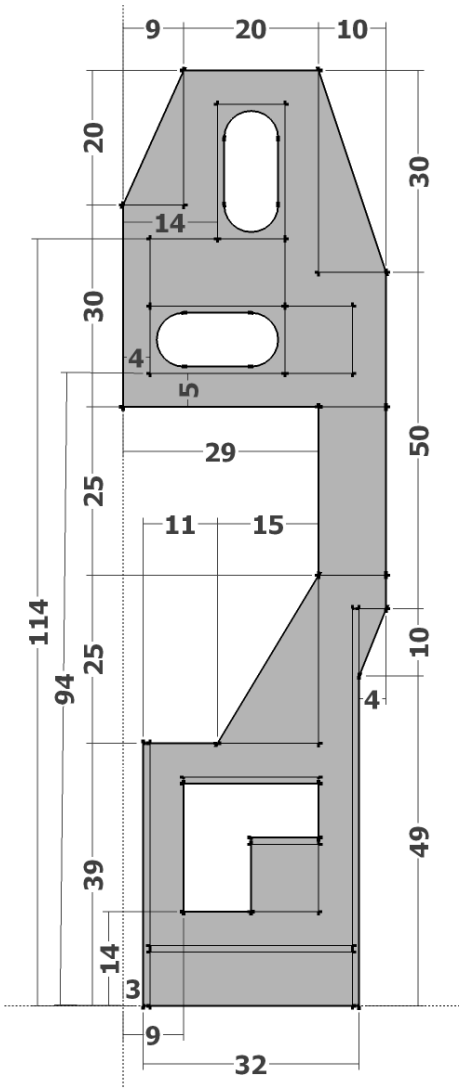
```
<path d="M 0,25 l 0,20 9,0 5,19 6,0 0,-14 a 4,4,0,0,1,8,0 l 0,14 6,0 5,-19 29,0 0,-20
-29,0 -5,-25 -20,0 -5,25 -9,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y
210	19,	5
120	4,	30
120	44,	30

Ebene B, Blech B

	Nummer	Position x,	y	Schicht
	0	191 ↔ 201,	116 ↔ 106	3
Instanzen:	1	151 ↔ 161,	116 ↔ 106	3
	2	111 ↔ 121,	116 ↔ 106	3
	3	71 ↔ 81,	116 ↔ 106	3



SVG-Pfad:

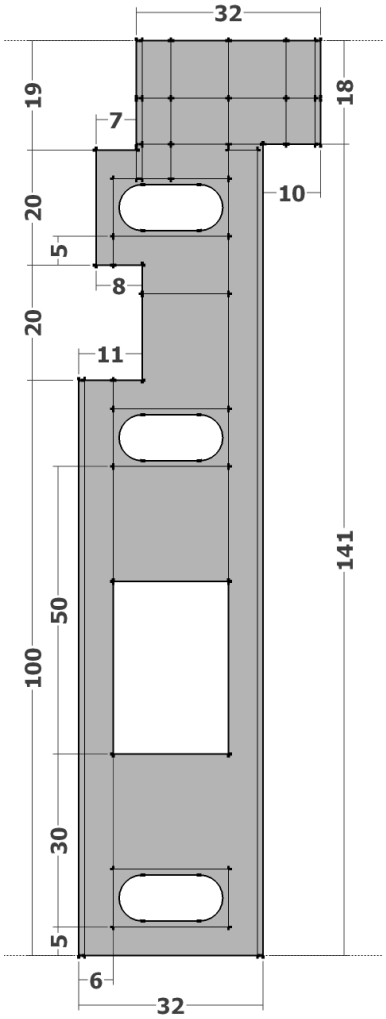
```
<path d="M 3,0 l 0,39 11,0 15,25 0,25 -29,0 0,30 9,20 20,0 10,-30 0,-50 -4,-10 0,-49
-32,0 z"/>
```

Ausschnitte:

Nummer	Position x,	y
2245	9,	14
120	4,	94
210	14,	114

Ebene B, Blech C

	Nummer	Position x,	y	Schicht
	0	194,	15 ↔ 5	2
Instanzen:	1	154,	15 ↔ 5	2
	2	114,	15 ↔ 5	2
	3	74,	15 ↔ 5	2



SVG-Pfad: `<path d="M 0,0 l 0,100 11,0 0,20 -8,0 0,20 7,0 0,19 32,0 0,-18 -10,0 0,-141 -32,0 Z"/>`

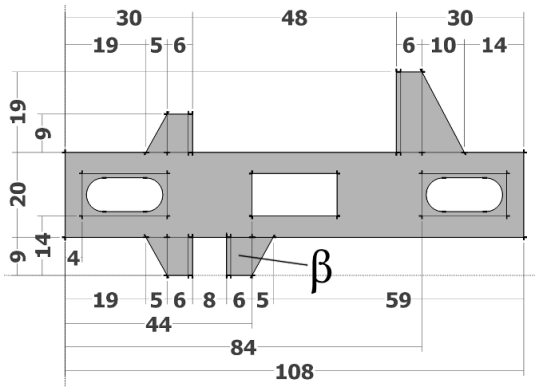
Ausschnitte:

Nummer	Position x,	y
120	6,	5
32	6,	35
120	6,	85
120	6,	125

Ebene B, Blech F

Instanzen:

Nummer	Position x,	y	Schicht
1	131 ↔ 141,	61	5
2	91 ↔ 101,	61	6
3	51 ↔ 61,	61	4



SVG-Pfad:

```
<path d="M 0,9 l 0,20 19,0 5,9 6,0 0,-9 48,0 0,19 6,0 10,-19 14,0 0,-20 -59,0 -5,-9
-6,0 0,9 -8,0 0,-9 -6,0 -5,9 -19,0 Z"/>
```

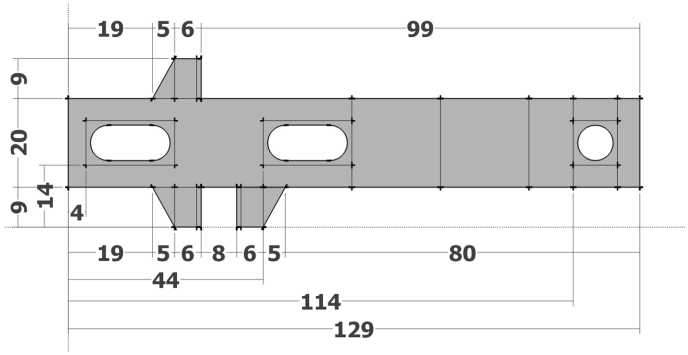
Ausschnitte:

Nummer	Position x,	y
120	4,	14
12	44,	14
120	84,	14

Ebene B, Blech F0

Instanzen:

Nummer	Position x,	y	Schicht
0	171 ↔ 181,	61	4



SVG-Pfad:

```
<path d="M 0,9 l 0,20 19,0 5,9 6,0 0,-9 99,0 0,-20 -80,0 -5,-9 -6,0 0,9 -8,0 0,-9
-6,0 -5,9 -19,0 Z"/>
```

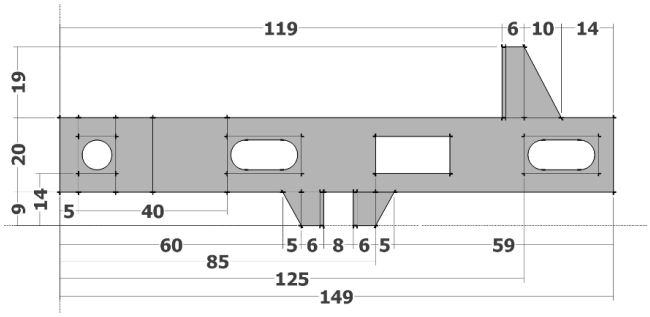
Ausschnitte:

Nummer	Position x,	y
120	4,	14
120	44,	14
110	114,	14

Ebene B, Blech F4

Instanzen:

Nummer	Position x,	y	Schicht
0	-30 ↔ -20,	61	5



SVG-Pfad:

```
<path d="M 0,9 l 0,20 119,0 0,19 6,0 10,-19 14,0 0,-20 -59,0 -5,-9 -6,0 0,9 -8,0 0,-9
-6,0 -5,9 -60,0 Z"/>
```

Ausschnitte:

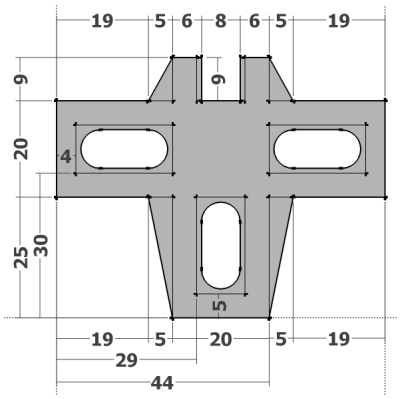
Nummer	Position x,	y
110	5,	14
120	45,	14

Nummer	Position x,	y
12	85,	14
120	125,	14

Ebene B, Blech H

Instanzen:

Nummer	Position x,	y	Schicht
0	171 ↔ 181,	5	3
1	131 ↔ 141,	5	4
2	91 ↔ 101,	5	3
3	51 ↔ 61,	5	4



SVG-Pfad:

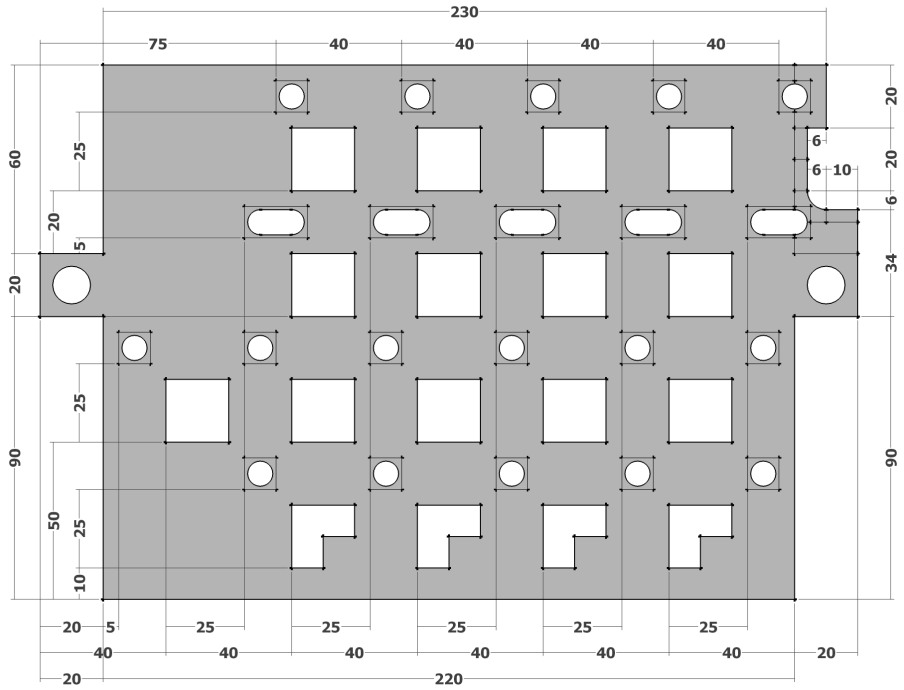
```
<path d="M 0,25 l 0,20 19,0 5,9 6,0 0,-9 8,0 0,9 6,0 5,-9 19,0 0,-20 -19,0 -5,-25
-20,0 -5,25 -19,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y
210	29,	5
120	4,	30
120	44,	30

Ebene B, Blech G

	Nummer	Position x,	y	Schicht
Instanzen:	0	0,	0	0
	1	0,	0	7



SVG-Pfad: `<path d="M 250,124 l 10,0 0,-34 -20,0 0,-90 -220,0 0,90 -20,0 0,20 20,0 0,60 230,0 0,-20 -6,0 0,-20 a 6,6,0,0,1,6,-6 Z"/>`

Ausschnitte:

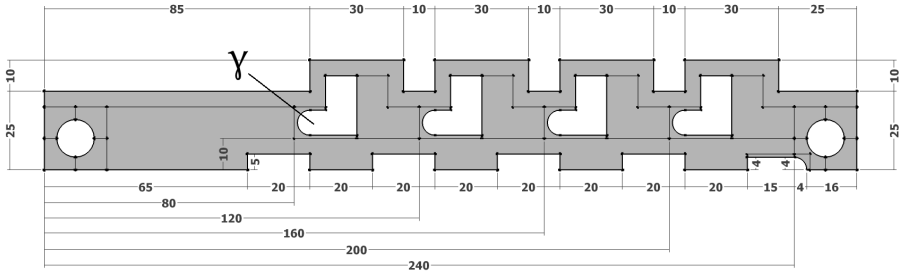
Nummer	Position x,	y
110	75,	155
22	160,	130
22	200,	130
224	80,	10
22	80,	130
224	120,	10
224	200,	10
110	65,	35
110	105,	35
110	145,	35
110	185,	35
110	225,	35
22	40,	50
22	80,	50
22	120,	50
22	160,	50
22	200,	50
110	25,	75
110	65,	75
110	105,	75

Nummer	Position x,	y
110	145,	75
22	120,	130
110	185,	75
110	225,	75
220	0,	90
22	80,	90
22	120,	90
22	160,	90
22	200,	90
220	240,	90
120	65,	115
120	105,	115
120	145,	115
120	185,	115
120	225,	115
110	115,	155
110	155,	155
110	195,	155
110	235,	155
224	160,	10

Ebene B, Blech N

Instanzen:

Nummer	Position x,	y	Schicht
0	0,	130 ↔ 120	6



SVG-Pfad:

```
<path d="M 240,4 l -15,0 0,-4 -20,0 0,5 -20,0 0,-5 -20,0 0,5 -20,0 0,-5 -20,0 0,5
-20,0 0,-5 -20,0 0,5 -20,0 0,-5 -65,0 0,25 85,0 0,10 30,0 0,-10 10,0 0,10 30,0 0,-10
10,0 0,10 30,0 0,-10 10,0 0,10 30,0 0,-10 25,0 0,-25 -16,0 a 4,4,0,0,1,-4,4 Z"/>
```

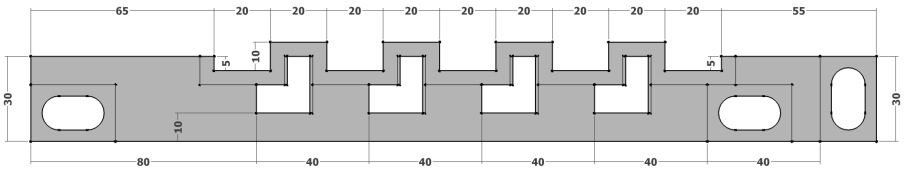
Ausschnitte:

Nummer	Position x,	y	Nummer	Position x,	y
220	0,	0	2227	160,	10
2227	80,	10	2227	200,	10
2227	120,	10	220	240,	0

Ebene B, Blech Q

Instanzen:

Nummer	Position x,	y	Schicht
0	-10 ↔ 0,	10 ↔ 0	1



SVG-Pfad:

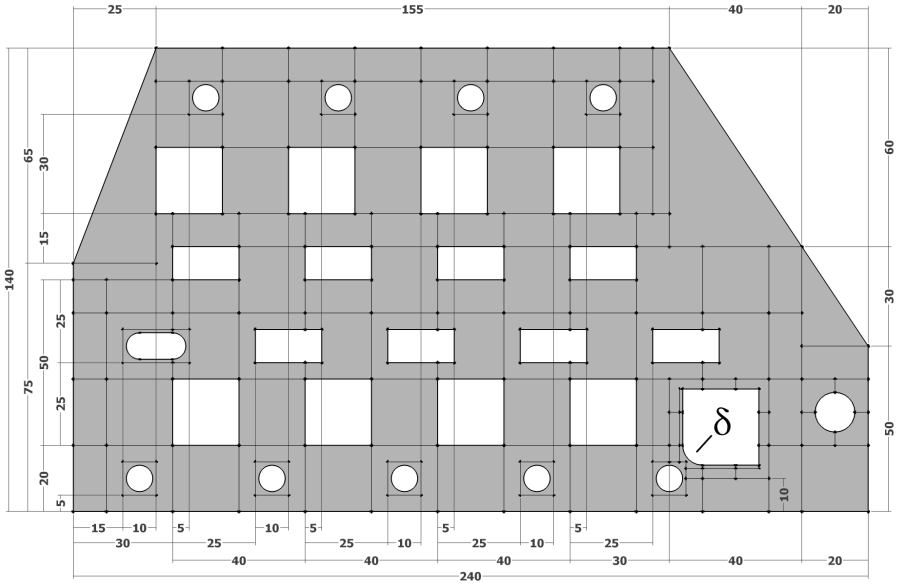
```
<path d="M 0,0 l 0,30 65,0 0,-5 20,0 0,10 20,0 0,-10 20,0 0,10 20,0 0,-10 20,0 0,10
20,0 0,-10 20,0 0,10 20,0 0,-10 20,0 0,5 55,0 0,-30 -300,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y	Nummer	Position x,	y
230	0,	0	2228	200,	10
2228	80,	10	230	240,	0
2228	120,	10	320	280,	0
2228	160,	10			

Ebene B, Blech Q'

	Nummer	Position x,	y	Schicht
Instanzen:	0	50 ↔ 60,	110	1



SVG-Pfad: `<path d="M 0,0 l 0,75 25,65 155,0 40,-60 20,-30 0,-50 -240,0 Z"/>`

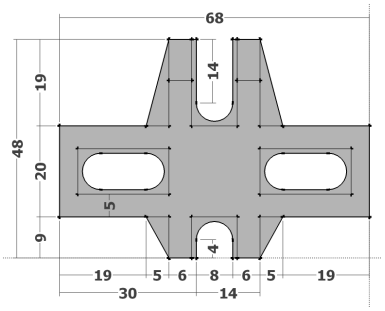
Ausschnitte:

Nummer	Position x,	y	Nummer	Position x,	y
110	15,	5	12	135,	45
110	55,	5	12	175,	45
110	95,	5	12	30,	70
110	135,	5	22	25,	90
110	175,	5	22	65,	90
22	30,	20	22	105,	90
22	70,	20	22	145,	90
22	110,	20	110	35,	120
22	150,	20	110	75,	120
3367	180,	10	110	115,	120
220	220,	20	110	155,	120
120	15,	45	12	70,	70
12	55,	45	12	110,	70
12	95,	45	12	150,	70

Ebene C & D, Blech F

Instanzen:

Nummer	Position x,	y	Ebene	Schicht
0	171 ↔ 181,	61	C	5
1	131 ↔ 141,	61	D	5
2	91 ↔ 101,	61	C	5
3	51 ↔ 61,	61	D	5



SVG-Pfad:

```
<path d="M 0,9 l 0,20 19,0 5,19 6,0 0,-14 a 4,4,0,0,1,8,0 l 0,14 6,0 5,-19 19,0 0,-20
-19,0 -5,-9 -6,0 0,4 a 4,4,0,0,1,-8,0 l 0,-4 -6,0 -5,9 -19,0 Z"/>
```

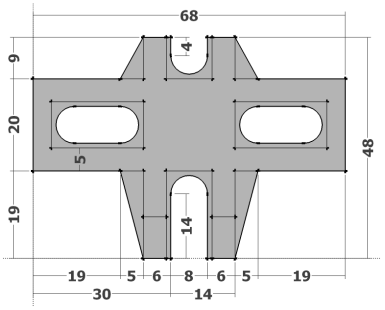
Ausschnitte:

Nummer	Position x,	y
120	4,	14
120	44,	14

Ebene C & D, Blech J

Instanzen:

Nummer	Position x,	y	Ebene	Schicht
0	171 ↔ 181,	11	C	5
1	131 ↔ 141,	11	D	5
2	91 ↔ 101,	11	C	5
3	51 ↔ 61,	11	D	5



SVG-Pfad:

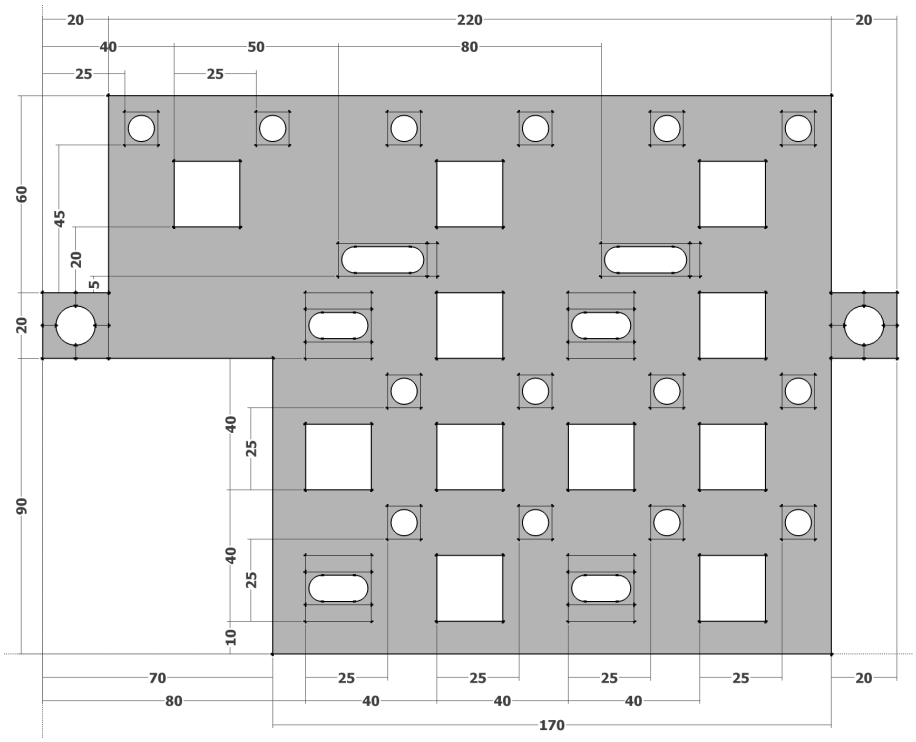
```
<path d="M 0,19 l 0,20 19,0 5,9 6,0 0,-4 a 4,4,0,0,1,8,0 l 0,4 6,0 5,-9 19,0 0,-20
-19,0 -5,-19 -6,0 0,14 a 4,4,0,0,1,-8,0 l 0,-14 -6,0 -5,19 -19,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y
120	4,	24
120	44,	24

Ebene C, Blech G

	Nummer	Position x,	y	Schicht
Instanzen:	0	0,	0	0
	1	0,	0	6



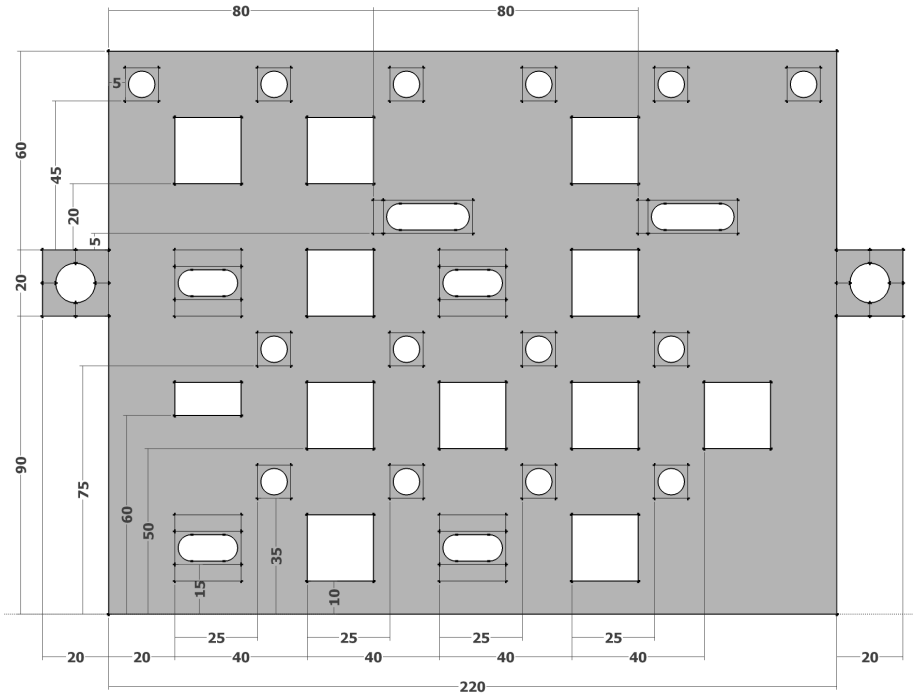
SVG-Pfad: `<path d="M 0,90 l 0,20 20,0 0,60 220,0 0,-60 20,0 0,-20 -20,0 0,-90 -170,0 0,90 -70,0 Z"/>`

Ausschnitte:

Nummer	Position x,	y	Nummer	Position x,	y
120	80,	15	120	80,	95
22	120,	10	22	120,	90
120	160,	15	120	160,	95
22	200,	10	22	200,	90
110	105,	35	220	240,	90
110	145,	35	1308	90,	115
110	185,	35	1308	170,	115
110	225,	35	22	40,	130
22	80,	50	22	120,	130
22	120,	50	22	200,	130
22	160,	50	110	25,	155
22	200,	50	110	65,	155
110	105,	75	110	105,	155
110	145,	75	110	145,	155
110	185,	75	110	185,	155
110	225,	75	110	225,	155
220	0,	90			

Ebene D, Blech G

	Nummer	Position x,	y	Schicht
Instanzen:	0	0,	0	0
	1	0,	0	6



SVG-Pfad:

```
<path d="M 0,90 l 0,20 20,0 0,60 220,0 0,-60 20,0 0,-20 -20,0 0,-90 -220,0 0,90 -20,0 Z"/>
```

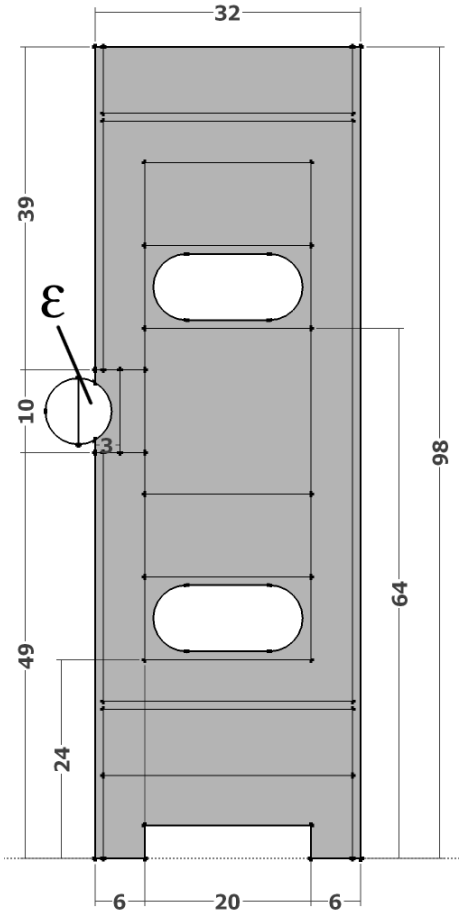
Ausschnitte:

Nummer	Position x,	y
120	40,	15
22	80,	10
120	120,	15
22	160,	10
110	65,	35
110	105,	35
110	145,	35
110	185,	35
12	40,	60
22	80,	50
22	120,	50
22	160,	50
22	200,	50
110	65,	75
110	105,	75
110	145,	75
110	185,	75

Nummer	Position x,	y
220	0,	90
120	40,	95
22	80,	90
120	120,	95
22	160,	90
220	240,	90
1306	100,	115
1306	180,	115
22	40,	130
22	80,	130
22	160,	130
110	25,	155
110	65,	155
110	105,	155
110	145,	155
110	185,	155
110	225,	155

Ebene C & D, Blech K

	Nummer	Position x,	y	Ebene	Schicht
	0	194,	76 ↔ 66	C	1
Instanzen:	1	154,	76 ↔ 66	D	1
	2	114,	76 ↔ 66	C	1
	3	74,	76 ↔ 66	D	1



SVG-Pfad: `<path d="M 0,0 l 0,49 0,1.5 q 4,3.5,0,7 l 0,1.5 0,39 32,0 0,-98 -6,0 0,4 -20,0 0,-4 -6,0 Z"/>`

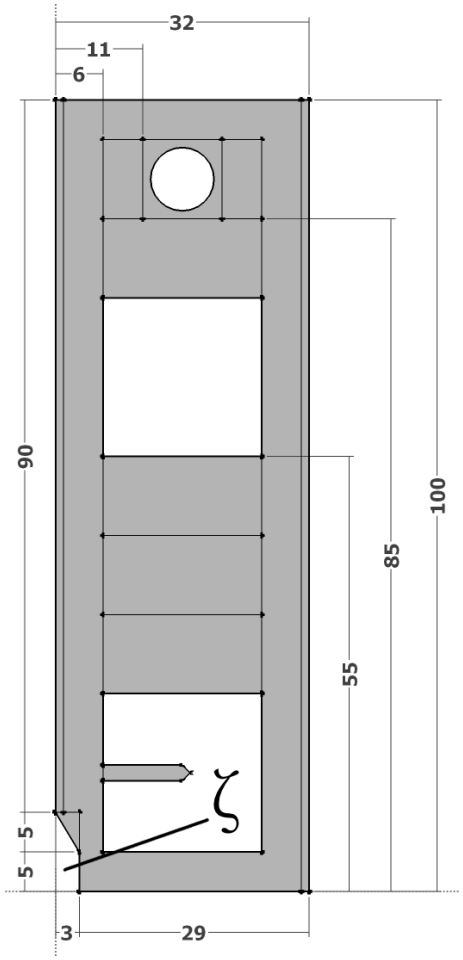
Auf Ebene D ist die Kerbe auf der anderen Seite: `<path d="M 0,0 l 0,98 32,0 0,-39 0,-1.5 q -4,-3.5,0,-7 l 0,-1.5 0,-49 -6,0 0,4 -20,0 0,-4 -5,0 Z"/>`

Ausschnitte:

Nummer	Position x,	y
120	6,	24
120	6,	64

Ebene C & D, Blech M

	Nummer	Position x,	y	Ebene	Schicht
Instanzen:	0	194,	125 ↔ 115	C	5
	1	154,	125 ↔ 115	D	5
	2	114,	125 ↔ 115	C	5
	3	74,	125 ↔ 115	D	5



SVG-Pfad: `<path d="M 0,10 l 0,90 32,0 0,-100 -29,0 0,5 -3,5 Z"/>`

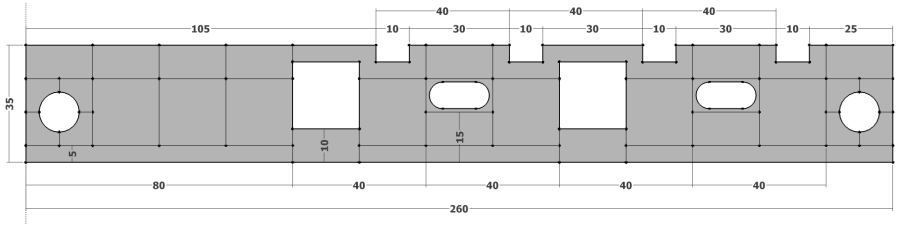
Auf Ebene D ist die Kerbe auf der anderen Seite: `<path d="M 0,0 l 0,100 32,0 0,-90 -3,-5 0,-5 -29,0 Z"/>`

Ausschnitte:

Nummer	Position x,	y
2296	6,	5
22	6,	55
110	11,	85

Ebene C, Blech R

	Nummer	Position x,	y	Schicht
Instanzen:	0	0,	5 ↔ -5	1



SVG-Pfad:

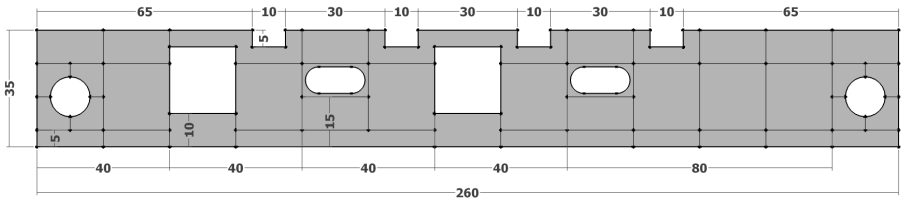
```
<path d="M 0,0 l 0,35 105,0 0,-5 10,0 0,5 30,0 0,-5 10,0 0,5 30,0 0,-5 10,0 0,5 30,0
0,-5 10,0 0,5 25,0 0,-35 -260,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y	Nummer	Position x,	y
220	0,	5	22	160,	10
22	80,	10	120	200,	15
120	120,	15	220	240,	5

Ebene D, Blech R

	Nummer	Position x,	y	Schicht
Instanzen:	0	0,	5 ↔ -5	1



SVG-Pfad:

```
<path d="M 0,0 l 0,35 65,0 0,-5 10,0 0,5 30,0 0,-5 10,0 0,5 30,0 0,-5 10,0 0,5 30,0
0,-5 10,0 0,5 65,0 0,-35 -260,0 Z"/>
```

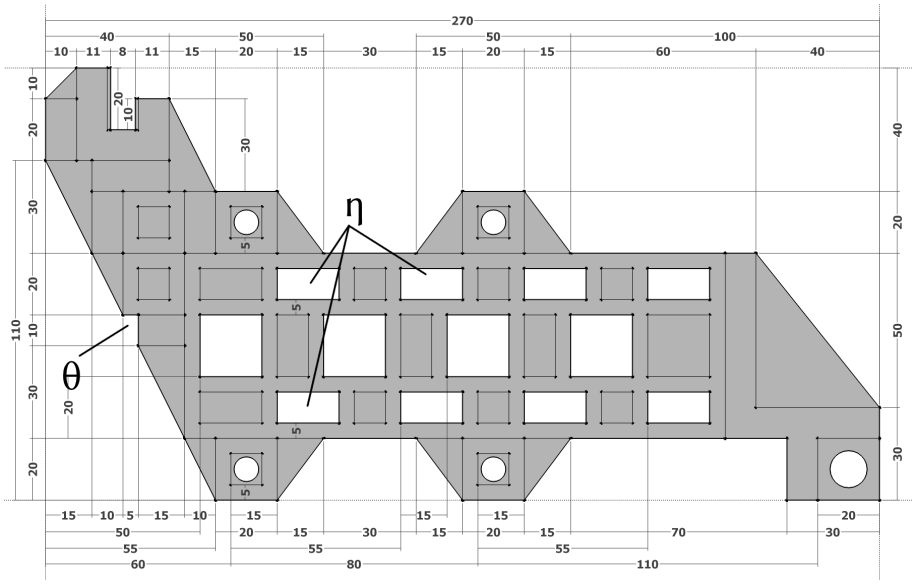
Ausschnitte:

Nummer	Position x,	y	Nummer	Position x,	y
220	0,	5	22	120,	10
22	40,	10	120	160,	15
120	80,	15	220	240,	5

Ebene C, Blech S

Instanzen:

Nummer	Position x,	y	Schicht
0	20 ↔ 30,	10	2



SVG-Pfad:

```
<path d="M 0,110 l 0,20 10,10 11,0 0,-20 8,0 0,10 11,0 15,-30 20,0 15,-20 30,0 15,20
20,0 15,-20 60,0 20,-25 20,-25 0,-30 -30,0 0,20 -70,0 -15,-20 -20,0 -15,20 -30,0
-15,-20 -20,0 -10,20 -15,30 0,10 -5,0 -10,20 -15,30 Z"/>
```

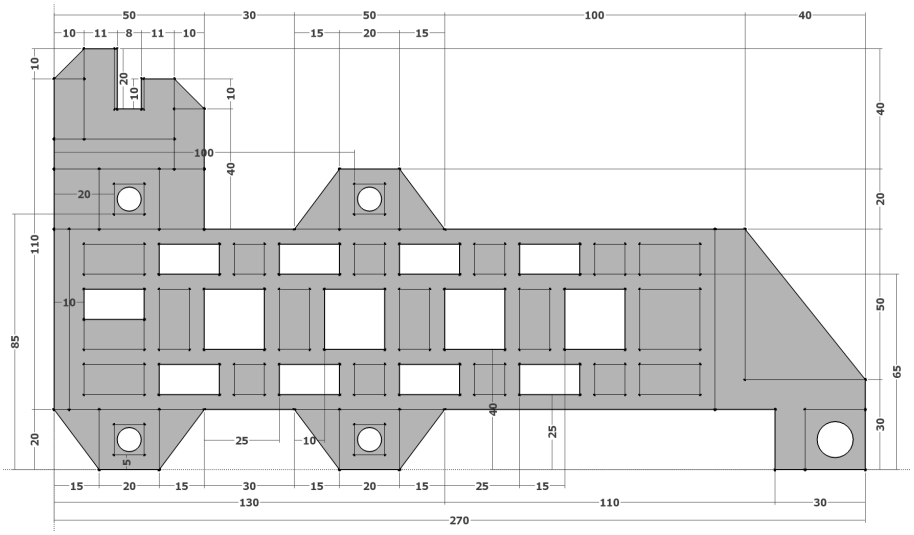
Ausschnitte:

Nummer	Position x,	y
22	50,	40
110	60,	5
110	60,	85
129	75,	25
129	75,	65
22	90,	40
129	115,	25
129	115,	65
22	130,	40

Nummer	Position x,	y
110	140,	5
110	140,	85
129	155,	25
129	155,	65
22	170,	40
129	195,	25
129	195,	65
220	250,	0

Ebene D, Blech S

Instanzen:	Nummer	Position x,	y	Schicht
	0	20 ↔ 30,	10	2



SVG-Pfad:

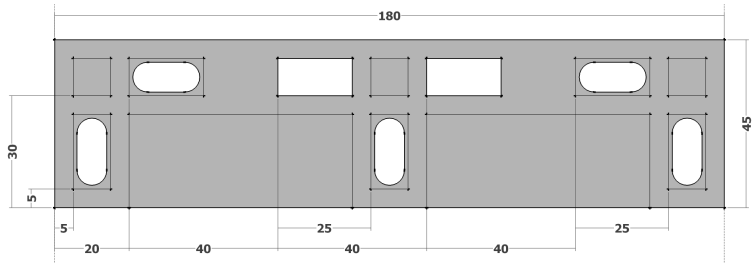
```
<path d="M 0,20 l 0,110 10,10 11,0 0,-20 8,0 0,10 11,0 10,-10 0,-40 30,0 15,20 20,0
15,-20 100,0 20,-25 20,-25 0,-30 -30,0 0,20 -110,0 -15,-20 -20,0 -15,20 -30,0 -15,-20
-20,0 -15,20 Z"/>
```

Ausschnitte:

Nummer	Position x,	y	Nummer	Position x,	y
110	20,	5	129	35,	65
110	100,	5	129	75,	65
110	20,	85	129	115,	65
110	100,	85	129	155,	65
220	250,	0	129	10,	50
129	35,	25	22	50,	40
129	75,	25	22	90,	40
129	115,	25	22	130,	40
129	155,	25	22	170,	40

Ebene C, Blech T

	Nummer	Position x,	y	Schicht
Instanzen:	0	35 ↔ 45,	125	3



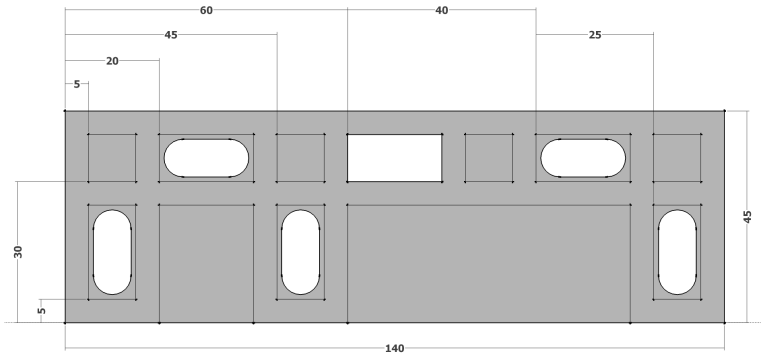
SVG-Pfad: `<path d="M 0,0 l 0,45 180,0 0,-45 -180,0 Z"/>`

Ausschnitte:

Nummer	Position x,	y
210	5,	5
210	85,	5
210	165,	5
120	20,	30
12	60,	30
12	100,	30
120	140,	30

Ebene D, Blech T

	Nummer	Position x,	y	Schicht
Instanzen:	0	35 ↔ 45,	125	3



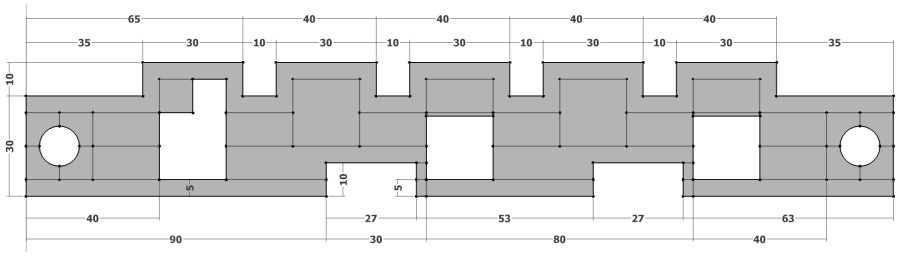
SVG-Pfad: `<path d="M 0,0 l 0,45 140,0 0,-45 -140,0 Z"/>`

Ausschnitte:

Nummer	Position x,	y
210	5,	5
210	45,	5
210	125,	5
120	20,	30
12	60,	30
120	100,	30

Ebene C, Blech U

	Nummer	Position x,	y	Schicht
Instanzen:	0	0,	125 ↔ 115	4



SVG-Pfad:

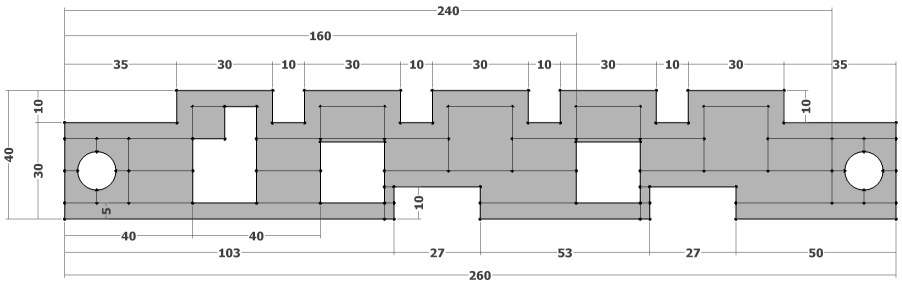
```
<path d="M 0,0 l 0,30 35,0 0,10 30,0 0,-10 10,0 0,10 30,0 0,-10 10,0 0,10 30,0 0,-10
10,0 0,10 30,0 0,-10 10,0 0,10 30,0 0,-10 35,0 0,-30 -63,0 0,10 -27,0 0,-10 -53,0
0,10 -27,0 0,-10 -90,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y
220	0,	5
322	40,	5
225	120,	5
225	200,	5
220	240,	5

Ebene D, Blech U

	Nummer	Position x,	y	Schicht
Instanzen:	0	0,	125 ↔ 115	4



SVG-Pfad:

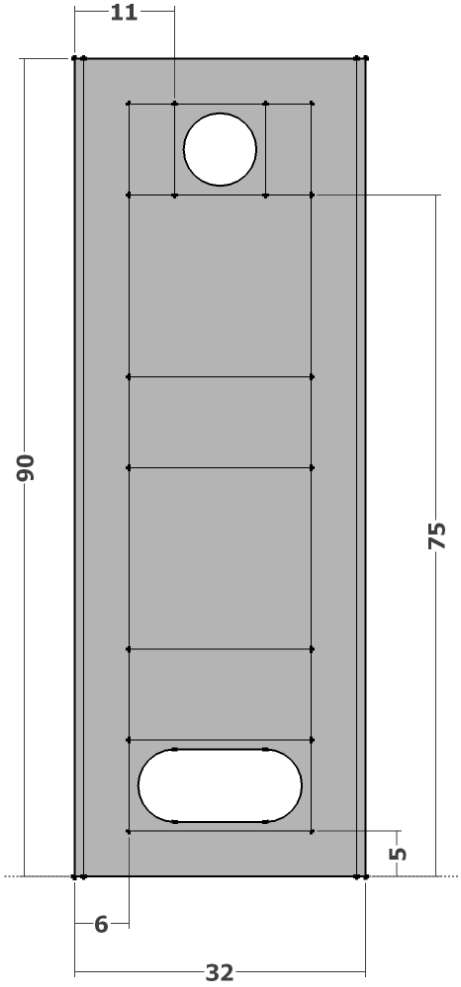
```
<path d="M 0,0 l 0,30 35,0 0,10 30,0 0,-10 10,0 0,10 30,0 0,-10 10,0 0,10 30,0 0,-10
10,0 0,10 30,0 0,-10 10,0 0,10 30,0 0,-10 35,0 0,-30 -50,0 0,10 -27,0 0,-10 -53,0
0,10 -27,0 0,-10 -103,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y
220	0,	5
322	40,	5
225	80,	5
225	160,	5
220	240,	5

Ebene C & D, Blech V

	Nummer	Position x,	y	Ebene	Schicht
Instanzen:	0	34,	135 ↔ 125	C	1
	0	34,	135 ↔ 125	D	1



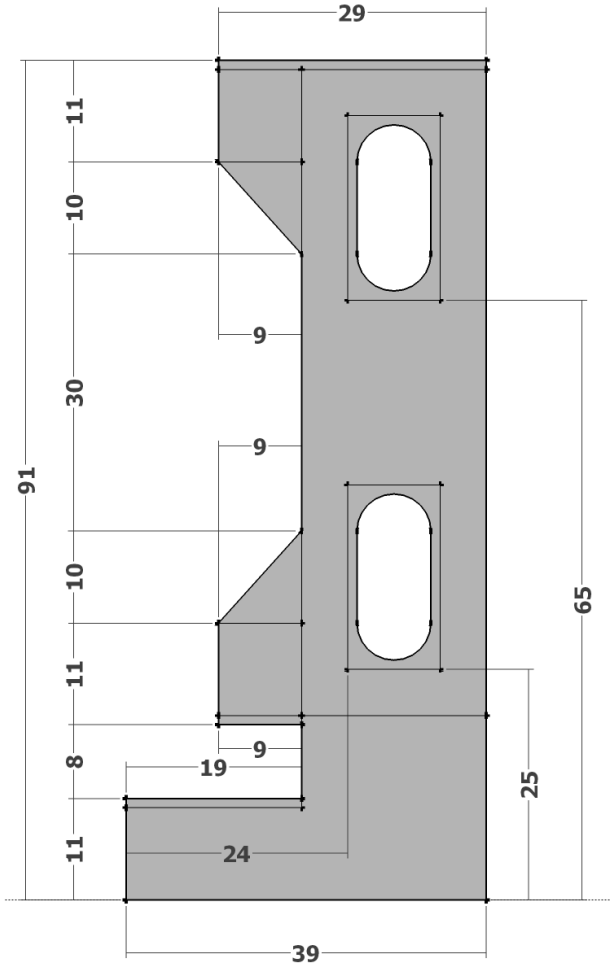
SVG-Pfad: `<path d="M 0,0 l 0,90 32,0 0,-90 -32,0 Z"/>`

Ausschnitte:

Nummer	Position x,	y
120	6,	5
110	11,	75

Ebene C & D, Blech W

	Nummer	Position x,	y	Ebene	Schicht
	0	201,	10 ↔ 0	C	4
Instanzen:	1	161,	10 ↔ 0	D	4
	2	121,	10 ↔ 0	C	4
	3	81,	10 ↔ 0	D	4



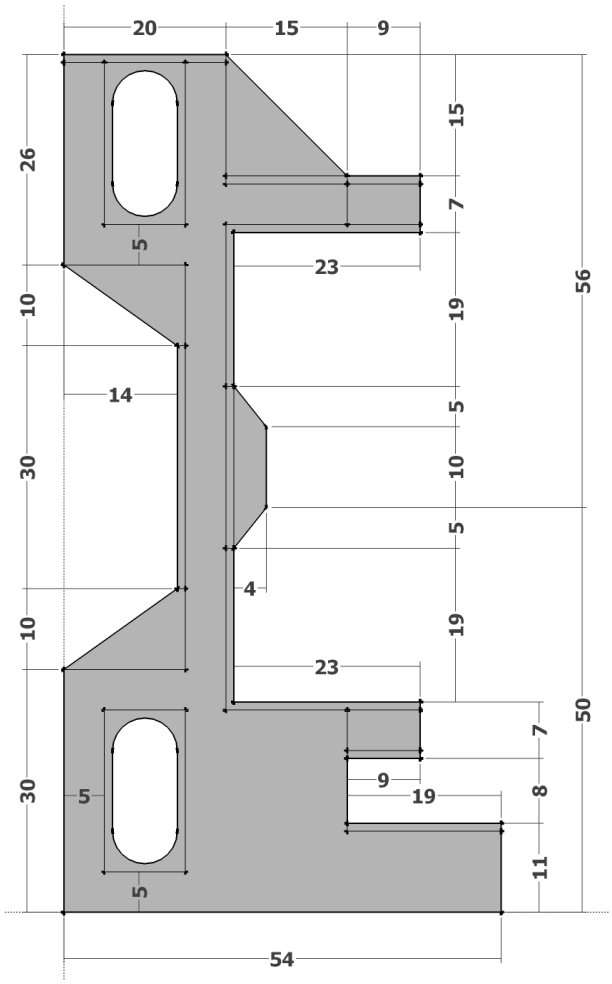
SVG-Pfad: `<path d="M 0,0 l 0,11 19,0 0,8 -9,0 0,11 9,10 0,30 -9,10 0,11 29,0 0,-91 -39,0 Z"/>`

Ausschnitte:

Nummer	Position x,	y
210	24,	25
210	24,	65

Ebene C & D, Blech W'

	Nummer	Position x,	y	Ebene	Schicht
	0	155 ↔ 165,	10 ↔ 0	C	3
Instanzen:	1	115 ↔ 125,	10 ↔ 0	D	3
	2	75 ↔ 85,	10 ↔ 0	C	3
	3	35 ↔ 45,	10 ↔ 0	D	3



SVG-Pfad:

```
<path d="M 0,0 l 0,30 14,10 0,30 -14,10 0,26 20,0 15,-15 9,0 0,-7 -23,0 0,-19 4,-5
0,-10 -4,-5 0,-19 23,0 0,-7 -9,0 0,-8 19,0 0,-11 -54,0 Z"/>
```

Ausschnitte:

Nummer	Position x,	y
210	5,	5
210	5,	85

7.3 Weitere Abbildungen

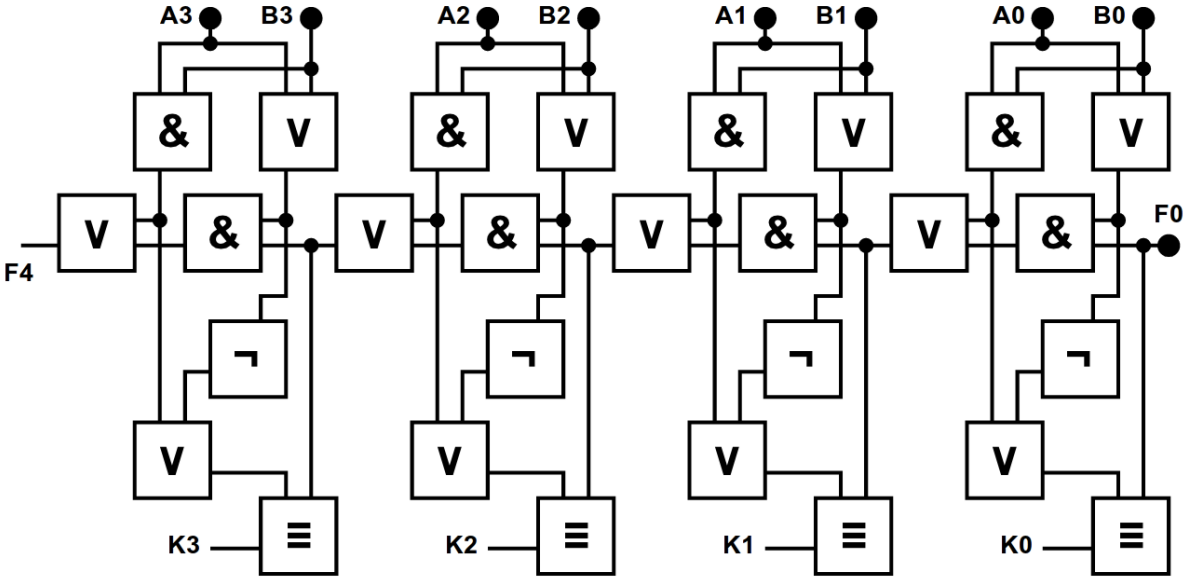


Abbildung 20: Logischer Schaltplan für alle 4 Stellen

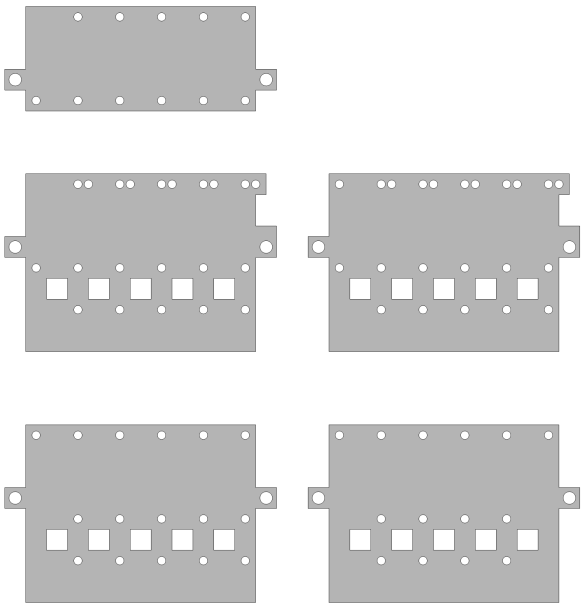


Abbildung 21: Formen der Zwischenebenen

Abbildungsverzeichnis

1 Detail des Z1-Nachbaus im Berliner Technikmuseum 3
2 Einfaches mechanisches Schaltglied [7] 6
3 Einfache Simulation eines Additionswerkes [23] 7
4 Takt und Positionen 8
5 Logischer Schaltplan an der Stelle 0 9
6 Raster mit Blechausschnitten und Stiften 11
7 Ausschnitte im Grundlayout 12
8 Blechausschnitte 13
9 Konstruierte Blechformen in SketchUp 14
10 Die Rekonstruktion des Blech C 15
11 Vertikaler Aufbau am Z1-Nachbau 18
12 Das 3D-Modell des Additionswerkes in WebGL 23
13 Fehlgeschlagene und fehlerfreie Triangulation 25
14 Anzeige einer Auswahl und gleiche Ansicht ohne Ausblendung 27
15 Zuses Originalplan der Ebene A [10, Fig. 56] 29
16 Zuses Originalplan der Ebene B [10, Fig. 58, 59] 30
17 Zuses Originalplan der Ebene C, Teil 1 [10, Fig. 61a] 31
18 Zuses Originalplan der Ebene C, Teil 2 [10, Fig. 61b] 32
19 Alle Arten von Blechausschnitten im Additionswerk 33
20 Logischer Schaltplan für alle 4 Stellen 60
21 Formen der Zwischenebenen 60

Quellenverzeichnis

Literatur

[1] Brian P. Hogan. *HTML5 & CSS3: Webentwicklung mit den Standards von morgen.* O'Reilly, 2011.

[2] Jan Kostulski. Webbasierte 3D-Simulation des Z1 Addierers von Konrad Zuse. Bachelorarbeit, Freie Universität Berlin, 2012.

[3] Tony Parisi. *WebGL: Up and Running.* O'Reilly, 2012.

[4] Raúl Rojas. Z1, Z2, Z3 and Z4. http://www.zib.de/zuse/Inhalt/Kommentare/Pdf/0680.pdf, 1999. Zuse Internet Archive Nr. 0680.

[5] Ursula Schweier and Dietmar Saupe. Funktions- und Konstruktionsprinzipien der Programmgesteuerten mechanischen Rechenmaschine Z1. http://www.zib.de/zuse/Inhalt/Kommentare/Pdf/0681.pdf, 1988. Zuse Internet Archive Nr. 0681.

[6] Thomas Walter. *Kompendium der Web-Programmierung: Dynamische Web-Sites.* Springer Berlin Heidelberg, 2008.

[7] Konrad Zuse. Mechanisches Schaltglied. http://www.zib.de/zuse/Inhalt/Texte/Chrono/30er/Pdf/0115.pdf, 1936. Zuse Internet Archive Nr. 0115.

[8] Konrad Zuse. Aus mechanischen Schaltgliedern aufgebautes Speicherwerk. Zusatz zum Zusatzpatent 919 017. http://www.zib.de/zuse/Inhalt/Texte/Chrono/30er/Pdf/0114.pdf, 1937. Zuse Internet Archive Nr. 0114.

[9] Konrad Zuse. Rechenvorrichtungen aus mechanischen Schaltgliedern. http://www.zib.de/zuse/Inhalt/Texte/Chrono/50er/Pdf/0169.pdf, 1952. Zuse Internet Archive Nr. 0169, konvertierter Text, mit Lücken, ohne Abbildungen.

[10] Konrad Zuse. Rechenvorrichtungen aus mechanischen Schaltgliedern. http://www.zib.de/zuse/Inhalt/Texte/Chrono/50er/Pdf/0169scan.pdf, 1952. Zuse Internet Archive Nr. 0169, Scan vom Original, teilweise unleserlich, mit Abbildungen.

Webseiten

[11] Adobe. http://www.adobe.com/products/flashruntimes/statistics.html.

[12] Ricardo Cabello. Three.js JavaScript 3D library. https://github.com/mrdoob/three.js/.

[13] Khronos Group. Getting a WebGL Implementation. http://www.khronos.org/webgl/wiki/Getting_a_WebGL_Implementation.

[14] heise online. http://www.heise.de/newsticker/foren/S-Konrad-Zuse-im-Netz/forum-236730/list/.

[15] heise online. Adobe stellt mobiles Flash-Plugin ein. http://www.heise.de/developer/meldung/Adobe-stellt-mobiles-Flash-Plugin-ein-1375503.html.

[16] heise online. Kein Flash fürs iPhone. http://www.heise.de/newsticker/meldung/Kein-Flash-fuers-iPhone-187788.html.

[17] heise online. Windows 8 vor Veröffentlichung mit Flash-Lücke. http://www.heise.de/security/meldung/Windows-8-vor-Veroeffentlichung-mit-Flash-Luecke-1703223.html.

[18] heise security. http://www.heise.de/security/dienste/Java-403125.html.

[19] Ralf Kunze. Der Flash Standard. http://www-lehre.inf.uos.de/~rkunze/flashweather/Diplomarbeit/node9.html.

[20] Timothy B. Lee. The Death of Flash and the Quiet Triumph of Open Standards. http://www.forbes.com/sites/timothylee/2011/09/28/the-death-of-flash-and-the-quiet-triumph-of-open-standards/.

[21] SketchUp. http://www.sketchup.com/intl/de/.

[22] ubuntuusers. http://wiki.ubuntuusers.de/Adobe_Flash.

[23] Götz Widiger. Simulation of the Z1 adder. http://www.zib.de/zuse/Inhalt/Programme/Simulationen/Z1/Simulation/addierer.html.

[24] Wikipedia. Comparison of HTML5 and Flash. http://en.wikipedia.org/wiki/Comparison_of_HTML5_and_Flash.

[25] Wikipedia. HTML5. http://en.wikipedia.org/wiki/HTML5.

[26] Wikipedia. Java applet. http://en.wikipedia.org/wiki/Java_applet.