Rheinische Friedrich-Wilhelms-Universität Bonn
Forschungsinstitut für Diskrete Mathematik

Bachelorarbeit

im Studiengang Informatik

Ein Emulator der Z25
aus dem Jahr 1963 von Konrad Zuse

Fabian Hendrik Erdmann, Matrikelnummer 2680572
30.10.2017

Erstgutachterin: Prof. Dr. Ina Prinz
Zweitgutachter: Prof. Dr. Joachim K. Anlauf

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Der Erfinder Konrad Zuse

Konrad Zuse wurde am 22. Juni 1910 in Berlin-Wilmersdorf geboren. Da sein Vater Emil Zuse — ein Postbeamter — mehrfach versetzt wurde, zog er 1912 nach Braunsberg in Ostpreußen und 1924 nach Hoyerswerda in Sachsen, wo er am damaligen Reform-Realgymnasium 1927 sein Abitur ablegte. Schon in dieser Zeit beschäftigte Zuse sich mit Mechanik und Automaten. So konstruierte er zum Beispiel, mit einem Stabil Baukasten, einen Lastenkran und entwarf ein Konzept für das Verkehrsnetz einer Stadt. [Zuse 1986, S. 1-12]
Zuses zweites Hobby war das Malen und Zeichnen — ein Talent das er schon in seiner Grundschulzeit entdeckt und seitdem gepflegt hatte. So stand er nach dem Schulabschluss vor der Wahl Ingenieur oder Reklame Zeichner zu werden. Er entschied sich für Ersteres und begann ein Maschinenbaustudium an der Technischen Hochschule Berlin-Charlottenburg. Schon bald war er mit dem Studienfach unzufrieden und wechselte erst zu Architektur, und schließlich zum Bauingenieurs Studiengang. In diesem Fach sah Zuse sich mit vielen aufwendigen und insbesondere gleichförmigen Rechenaufgaben konfrontiert, was ihn das erste Mal auf die Idee einer programmierbaren Rechenmaschine brachte. 1935 schloss er sein Studium erfolgreich ab und fand eine Anstellung als Statiker bei den Henschel-Flugzeugwerken. Auch im Beruf, ebbten die monotonen Rechenaufgabe zu Zuses Leidwesen nicht ab. Schon bald verließ er die Henschel-Flugzeugwerke wieder um sich seiner Idee einer programmgesteuerten Rechenmaschine zu widmen. [Zuse 1986, S. 13 f. und S. 30]
Von seinen Eltern bekam Zuse ein Zimmer, in deren Berliner Wohnung, um seine Idee umzusetzen. Bis 1938, baute er dort die rein mechanische, programmierbare Rechenmaschine Z1, deren Rechenwerk aus gegeneinander verschiebbaren gefrästen Blechen bestand. Die Z1 rechnete im binären System mit Gleitkommazahlen. Zur Darstellung eines Bits, wurde ein Blech in eine von zwei Positionen geschoben. Welche der beiden Position ein Blech innehatte, konnte daraufhin von einem orthogonal bewegten Blech abgegriffen werden. Die Z1 hatte separate Bereiche für Befehlseingabe, Zahleneingabe, Zahlenausgabe, Steuerwerk, Rechenwerk und Speicher, und konnte Schleifen — aber keine bedingten Sprünge — ausführen. Beim Bau der Maschine, wurde Zuse von einigen seiner Kommilitonen, sowohl finanziell als auch handwerklich, unterstützt. Auch Zuses Schwester und Zuses Vater — der sich extra aus dem Ruhestand zurückholen ließ — halfen Zuse dabei, die für den Bau nötigen Gelder aufzubringen. Aufgrund unzureichender Präzision bei der Fertigung der Schaltbleche, funktionierte die Z1 nicht einwandfrei [Zuse 1986, S. 31-48]. Zudem legt der 1989 gefertigte Nachbau der Z1 nahe, dass es einen Konstruktionsfehler bei der Abhandlung von Überträgen gab. Die dafür nötige Stange hat sich beim Nachbau nämlich während des Betriebs verbogen.
Es folgte das Versuchsmodell Z2, in Relaistechnik, und, mit Unterstützung der deutschen Versuchsanstalt für Luft- und Raumfahrttechnik, die, am 12. Mai 1941 einsatzbereite, Z3. Die Z3 gilt heute als „der erste funktionsfähige, frei programmierbare,

auf dem binären Zahlensystem (Gleitkommazahlen) und der binären Schaltungstechnik basierende Rechner der Welt" [Dr. Zuse 2017a]. Sie wurde 1944, zusammen mit den beiden Vorgängermaschinen, im Bombenkrieg zerstört.
Nach dem Bau der Z3, gründete Zuse seine erste Firma — Zuse Ingenieurbüro und Apparatebau Berlin, — wo er schon bald 20 Mitarbeiter beschäftigte. [Zuse 1986, S. 57 f.]
1942 und 1943 entwickelte Zuse die S1 und S2 — Spezialrechner, die zur Flügelvermessung von ferngesteuerten Bomben verwendet wurden. Konkret wurden, in der Fertigung entstandene, Ungenauigkeiten der Stromlinienform gemessen, um diese bei der Steuerung der Bombe berücksichtigen zu können. Um die Daten der Messgeräte direkt in die Rechner einzuspeisen, entwickelte Zuse dabei die ersten Analog-Digital-Wandler. [Zuse 1986, S. 62-65]
Parallel zu den Flügelvermessungsmaschinen, wurde ab 1942 die Z4 entwickelt, die im März 1945 einsatzbereit war. Während dieser Zeit wurde Zuses Arbeit zunehmend durch den Bombenkrieg gestört. Zwischenzeitlich musste die Produktion sogar verlegt werden. Zuse selbst schreibt darüber: „Es herrschte Chaos, aber ein produktives Chaos." [Zuse 1986, S. 72]
Durch günstige Beziehungen gelang es, nach der Fertigstellung, die Z4 von Berlin nach Oberjoch und schließlich nach Hinterstein im Allgäu zu verlegen. Dass Zuse die Maschine ursprünglich als V4 (Versuchsmodell 4) bezeichnet hatte, stellte sich beim Transport als Glücksfall heraus, da sie an vielen Stellen für das, als kriegswichtig geltende, Nachfolgemodell der V2 Rakete gehalten wurde. [Zuse 1986, S. 81-83]
Die Z4 wurde in Hopferau bei Füssen aufgestellt, wo es allerdings keinen Anwendungszweck für sie gab. Gleichzeitig fehlten Zuse die Mittel um an einer neuen Maschine zu arbeiten, sodass Zuses Computer Entwicklung zum Stillstand kam. Zuse nutzte die Zeit um sich theoretischen Arbeiten zu widmen. Er erarbeitete seine Programmiersprache „Plankalkül" und beschäftigte sich das erste mal mit dem Konzept „Rechnender Raum", der Auffassung des Universums als Computer. [Zuse 1986, S. 91-93]
Die Existenz der Z4 sprach sich währenddessen herum. 1949 ließ sich Professor Eduard Stiefel von der ETH Zürich die Z4 vorführen und mietete sie schließlich für wissenschaftliche Zwecke an der ETH. Die gewonnen finanziellen Mittel nutzte Zuse um noch im selben Jahr die Zuse KG, zusammen mit Harro Stucken und Alfred Eckhard — mit denen er schon in Berlin gearbeitet hatte — zu gründen. Im Auftrag der Leitz AG begann 1950 der Bau der Z5, die 1953 ausgeliefert wurde. Sie wurde zur Berechnung von Lichtbrechung in der optischen Industrie benutzt. [Zuse 1986, S. 104-106]
Es folgte das Modell Z11, von dem 48 Stück ausgeliefert wurden. 1956 ließ die Zuse KG die Elektromechanik hinter sich und brachte die in Röhrentechnik arbeitende Z22 auf den Markt. Schon mit dem nächsten Modell, der 1961 erschienenen Z23, gab es den Wechsel zur Transistortechnik. [Dr. Zuse 2017b, Dr. Zuse 2017c, Dr. Zu-

se 2017d]
1961 wurde der Zeichentisch Zuse Z64 entwickelt, mit dem hochpräzise technische Zeichnung angefertigt werden konnten. Die Zuse KG war in diesem Sektor führend und lieferte 98 Geräte aus. [Dr. Zuse 2017e]
Mit den Modellen Z31 und Z25 musste die Firma herbe Verluste hinnehmen. 1962 wurde deshalb die Rheinstahl AG als Teilhaber ins Boot geholt. Trotzdem nahmen die Dinge für die Zuse KG einen negativen Verlauf. 1964 sah sich Konrad Zuse gezwungen als aktiver Teilhaber auszuscheiden und verkaufte 1969 seine letzten Firmenanteile. 1965 übernahm die Brown Boveri Company 100% der Anteile, die 1967 an die Siemens AG weitergingen. Am 1. April 1971 wurde der Firmenname von Siemens gelöscht. [Zuse 1986, S. 137-139]
Die arbeitsreiche Zeit als Unternehmer hinter sich, beschäftigte sich Zuse wieder im theoretischen Bereich. 1969 veröffentlichte er ein gleichnamiges Buch über den rechnenden Raum. Das Plankalkül arbeitete er weiter aus und stellte Überlegungen zu selbstreplizierenden Automaten an. Keines seiner Konzepte konnte sich im wissenschaftlichen Kreis behaupten. [Zuse 1986, S. 141-146 und 154-159]
Vor seinem Tod im Dezember 1995, wurden Zuse zahlreiche Auszeichnungen und Ehrungen zuteil, unter Anderem ein Ehrendoktortitel an der TH Berlin-Charlottenburg, der Werner von Siemens Ring und das große Bundesverdienstkreuz mit Stern und Schulterband. [Zuse 1986, S. 146 f.]

2 Historischer Hintergrund der Zuse Z25

Vorgänger der Z25 war das Modell Z23, das ab 1961 in Serie produziert wurde. Die Z23 rechnet mit 40 Bit Wortlänge bei einer Taktfrequenz von 140 kHz. Sie ist der erste Computer der Zuse KG der mit Transistoren rechnet — alle nachfolgenden Modelle stützen sich ebenfalls auf diese Technik. Gekostet hat die Maschine 180 000 DM oder mehr, je nach Ausführung, und war, mit 98 verkauften Exemplaren, ein Erfolg für die Zuse KG. Die Z23 war universell ausgerichtet und wurde in sehr unterschiedlichen Feldern, zum Beispiel Maschinenbau, Verkehrstechnik oder Bergbau, eingesetzt. [Dr. Zuse 2017d]
Die Z25 wurde ab 1963 vertrieben und über den gesamten Produktionszeitrum 128 mal verkauft. Sie ist, aufgrund ihrer technischen Kapazitäten, als eine logische Weiterführung der Z23 zu betrachten. Die Z25 rechnet mit 18 Bit Wörtern und hat eine Taktfrequenz von 294 kHz, was zu einer deutlich höheren Rechengeschwindigkeit — zu Lasten der Flexibilität des Befehlssatzes — führt. Preislich fing die Z25 bei 80 000 DM an, ist also günstiger als die Z23 — wobei zu beachten ist, dass die Grundausstattung der Z25 weniger Speicher hat (1024 Wörter bei der Z25, 8192 Wörter bei der Z23). Hauptanwendungsgebiet der Z25 waren Berechnungen zur Flurbereinigung, also die Neuordnung von agrarwirtschaftlichen Flächen nach ökonomischen Gesichtspunkt. Aber auch in verschiedenen anderen Industrien und Wissenschaften

ist die Z25 zum Einsatz gekommen. Im Schiffsbau und der Textilindustrie wurde die Z25 zum Beispiel zum Auslesen und Herstellen technischer Zeichnungen beziehungsweise Schnittmuster, unter Zuhilfenahme von Peripheriegeräten wie dem Haropen oder dem Z64 Graphomat, eingesetzt. [Dr. Zuse 2017f]
Bei der Fertigung der ersten Z25 Maschinen kam es zu einer schwerwiegenden Verzögerung, welche die Zuse KG viel Geld kostete und maßgeblich zu der Übernahme von Siemens und letztlich der Auflösung der Zuse KG beitrug. Grund dafür war, dass die Transistoren — eine neu Art, welche die Zuse KG zuvor noch nicht verbaut hatte — falsch verlötet wurden. Ob dieser Fehler auf Mitarbeiter der Zuse KG oder auf schlechte Kommunikation des Transistorherstellers zurückgeht, ist nicht bekannt. [Zuse 1986, S. 138]
Parallel zur Z25 wurde die Z31 entwickelt; eine Maschine die zunächst für kleine Anwendungen, wie zum Beispiel die Verwaltung von Banken oder Versicherungen, konzipiert wurde. Im Laufe der Entwicklung wurde das Gerät dann aber immer größer und teurer, sodass es letztlich nicht zu seinem ursprünglichen Zweck geeignet war. Die Z31 wurde für anderthalb bis zwei Millionen DM angeboten und nur siebenmal verkauft, was der Firma enorme Verluste einbrachte. [Dr. Zuse 2017g]
Das Nachfolgemodell der Z25, die Z26, welches nie in Serie produziert wurde, sollte in eine ganz andere Richtung als die bisherigen Zuse Rechner gehen. Der Rechner sollte mit einem Betriebssystem geliefert werden und in der Lage sein mehrere Programme parallel zu rechnen. Außerdem wurde ein Baukastenprinzip verwendet. Das ermöglichte es einzelne Teile des Computers unabhängig voneinander auszutauschen, um so mit dem Stand der Technik schritthalten zu können. Die Entwicklung wurde fallengelassen, da Siemens inzwischen die Kontrolle über die Zuse KG übernommen hatte und an einer Weiterführung nicht interessiert war. [Dr. Zuse 2017h]

3 Aufbau und Funktion

Abbildung 1: Screenshot aus dem Emulator. Eigene Grafik.

Hauptbestandteile der Z25 sind ein Ferritkern Arbeitsspeicher, ein Rechenwerk und ein Leitwerk zur Steuerung. Die zentrale Ein- und Ausgabemöglichkeit bildet eine, im mitgelieferten Schreibtisch verbaute, Schublade. Normschnittstellen für Peripheriegeräte und weitere Z25 Maschinen (zwecks Netzbetrieb) sind vorhanden. Der vorgestellte Emulator verfügt über zwei Peripheriegeräte: einen Fernschreiber und einen Magnettrommelspeicher. [Zuse KG 1969, S. 5-11]

3.1 Ein-/Ausgabeschublade

Die Schublade ist mit 25 beleuchtbaren Tasten ausgestattet. Außerdem gibt es 3 Leuchtfelder und ein elektrisches Schloss, die zur Magnettrommel gehören. Über die Schublade kann man einzelne Befehle eingeben und den Rechenprozess steuern. Die Ein- und Ausgabe per Schublade erfolgt in binär; die Eingabe durch Drücken der 18 Haupttasten — jede steht für ein Bit eines Wortes, — die Ausgabe über aufleuchten der oberen Tastenhälfte. Die übrigen Tasten dienen der Steuerung des Programmflusses, beziehungsweise dem Ein- und Ausschalten der Maschine. [Zuse KG 1967, Kap. 7 S. 3]

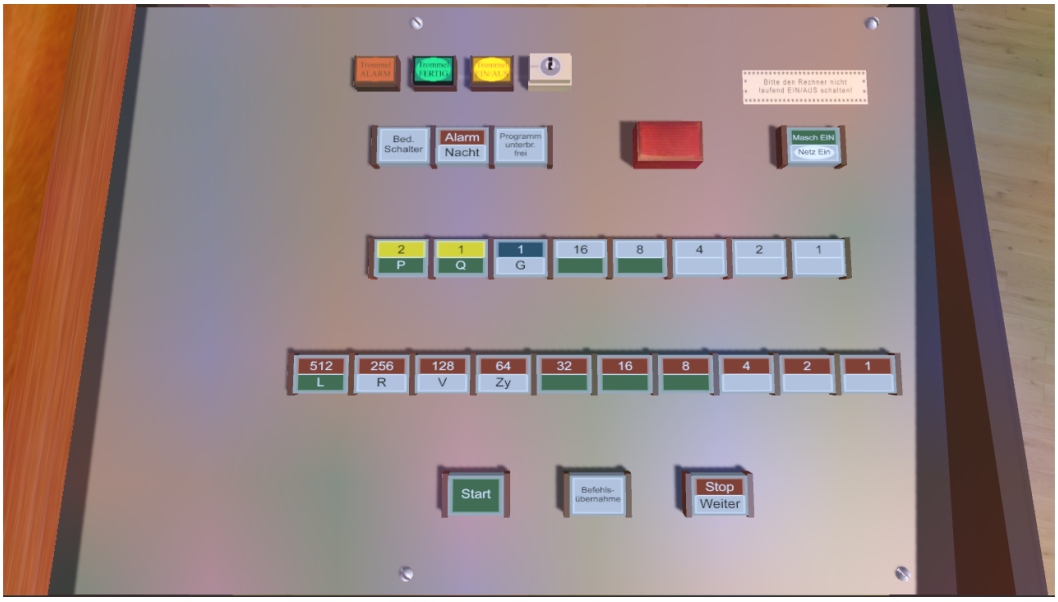

Abbildung 2: Screenshot der Ein-/Ausgabeschublade aus dem Emulator. Eigene Grafik.

Die Leuchtfelder, Trommel Alarm, Trommel Fertig und Trommel Ein, entsprechen den gleich beschrifteten Tasten der Magnettrommel. Das Schloss dient — genau wie das an der Magnettrommel angebrachte Schloss — dazu die Trommel ein- oder auszuschalten. Im Emulator kann nur das Schloss benutzt werden, das am Trommelschrank angebracht ist.

3.2 Speicher

In der Z25 sind ein Ferritkernspeicher, als Arbeitsspeicher, und Transistor Flipflops, für die Register des Computers, verbaut. Der Kernspeicher umfasst in Grundausstattung 1024 Speicherzellen und kann auf bis zu 16384 Zellen erweitert werden. Zusätzlich können zweimal 2048 Zellen an fest verdrahtetem ROM Speicher, für häufig genutzte Unterprogramme, angeschlossen werden. Die Sonderadressen 0-31 sind nicht Teil des Kernspeichers, auch wenn sie genau wie der Kernspeicher adressiert werden.
Jede Speicherzelle fasst ein 18 bit Wort und hat ein zusätzliches Bit zur Kontrolle der Quersummenparität beim Lesen. Der Hauptspeicher dient sowohl für Daten als auch für Programmcode, entspricht also der Von Neumann Architektur. Der Emulator benutzt 16384 Speicherzellen und hat keinen ROM. [Zuse KG 1967, Kap. 2 S. 6 f.]
Beim Ferritkernspeicher werden die Bits eines Wortes durch Magnetisierung eines ringförmigen Dauermagneten (Kern) dargestellt. Die Kerne haben zwei stabile ma-

gnetische Zustände und weisen eine fast quadratische Hystereseschleife auf, wodurch sie sich als sehr stabiles Speichermedium erweisen. Um einen Kern zu magnetisieren, wird in einem durch die Ringöffnung gelegten Draht ein Stromfluss erzeugt. Das entstehende Magnetfeld magnetisiert den Kern. Je nach Richtung des Stroms, wird der Kern in den einen oder den anderen Zustand gebracht. Um nicht für jeden Kern einen eigenen Draht legen zu müssen, sind die Kerne im Schachbrett-Muster angeordnet und durch jede Zeile und Spalte läuft je ein Draht. Um einen bestimmten Kern anzusteuern, legt man an dem Spaltendraht und dem Zeilendraht, die beide durch den gewünschten Kern führen, einen Strom an. Damit die anderen Kerne der selben Zeile und Spalte nicht auch beschrieben werden, wird auf jedem der zwei Drähte nur die Hälfte der zum Magnetisieren nötigen Stromstärke angelegt. Das resultierende Magnetfeld ist also nur am gewünschten Kern stark genug um die Hystereseschleife zu überwinden. [Zuse KG 1969, S. 35 f.]

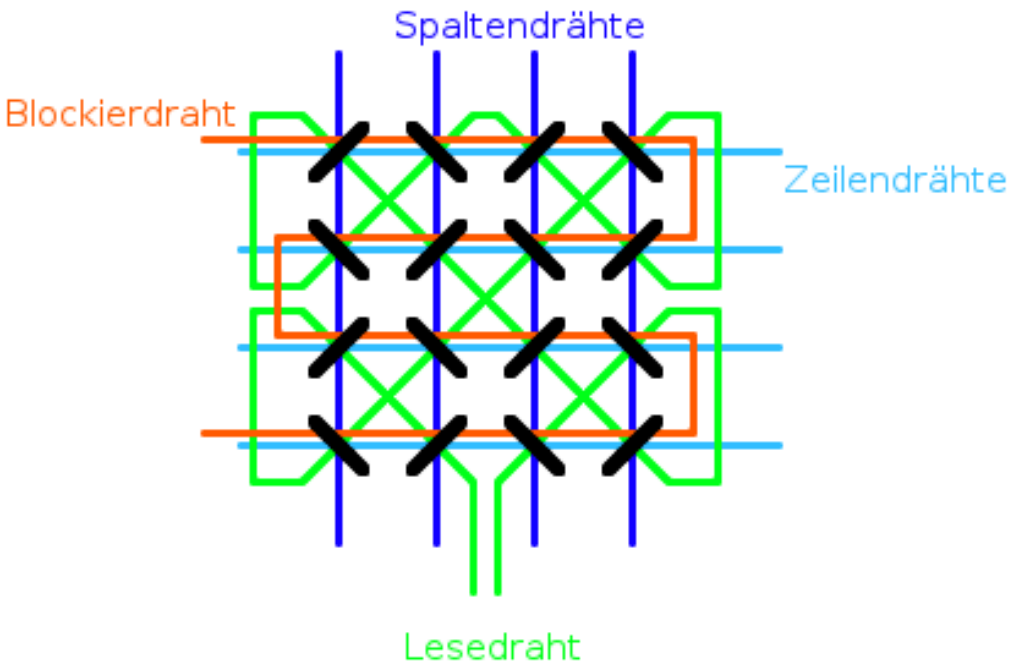

Abbildung 3: Eine Kernspeicherebene mit 16 Kernen. Nach [Zuse 1969, S. 38 f.]

Zum Lesen eines Kerns wird ein weiterer Draht benötigt. Für jede Speicherebene wird nur ein einziger Lesedraht verwendet, der durch alle Kerne der Ebene hindurchführt. Er wird diagonal zwischen Spalten- und Zeilendrähten hindurchgeführt, um von deren Magnetfeldern nicht zu einem Stromfluss angeregt zu werden. Um

einen bestimmten Kern nun auszulesen, wird dieser bestimmte Kern, so wie oben geschildert, mit 0 beschrieben. Befand sich besagter Kern vorher im Zustand 1, entsteht beim Umschalten ein Magnetfeld, das im Lesedraht einen Stromfluss induziert. War der Kern im Zustand 0, bleibt dieser Stromfluss aus. Da ausschließlich der gewünschte Kern geschaltet wird, kann der gemessene Strom in keinem der anderen Kerne entstanden sein, durch die der Lesedraht verläuft. [Zuse KG 1969, S. 37 f.]
Beim Lesen wir der Inhalt des Kerns zwangsweise gelöscht. Um den Ausgangszustand wiederherzustellen, wird deshalb über den entsprechenden Zeilen- und den entsprechenden Spaltendraht eine 1 in den Kern geschrieben. Falls vorm Lesen eine 0 im Kern gestanden hat, wäre das aber falsch, weshalb es noch einen vierten Draht, den Blockierdraht, gibt. Der Blockierdraht läuft, so wie der Lesedraht, durch alle Kerne einer Ebene. Im Gegensatz zum Lesedraht, verläuft der Blockierdraht aber parallel zu Spalten- oder Zeilendraht. Wenn im Leseschritt eine 0 gelesen wurde — der gelesene Kern also nicht in Zustand 1 gebracht werden darf — wird über den Blockierdraht ein Magnetfeld erzeugt, das dem schreibenden Magnetfeld von Spalten- und Zeilendraht entgegengesetzt ist. Dieses Magnetfeld ist nicht stark genug, um einen der durchlaufenen Kerne umzumagnetisieren, reicht aber aus, um das schreibende Magnetfeld so sehr zu schwächen, das der Kern nicht in Zustand 1 gebracht wird. [Zuse KG 1969, S. 38 f.]
Da die Ferritkerne selber keine Stromversorgung brauchen, bleiben gespeicherte Daten über das Ausschalten der Maschine hinweg erhalten.

Für die Register werden taktflankengesteuerte JK-Flipflops verwendet, mit der Besonderheit, dass beliebig viele Dateneingänge benutzt werden können, von denen jeder einen eigenen Takteingang hat. Der Grund ist, dass ein Flipflop zu mehreren unterschiedlichen Zeitpunkten einer Wortzeit geschaltet werden können soll und jede Bitzeit innerhalb einer Wortzeit ein eigenes Taktsignal benutzt.
Ein Register für 18 Bit (es gibt auch kürzere Register, wie das fünfstellige AER) besteht aus 18 hintereinander geschalteten Flipflops. Register können seriell oder parallel gelesen und beschrieben werden. Ein Register seriell zu lesen oder es zu beschreiben, dauert 18 Bitzeiten. In jeder Bitzeit wird das vorderste Bit ausgegeben (bzw. das hinterste Bit beschrieben); alle anderen Bits werden in das jeweils nächste Flipflop übertragen. Das Wort wird also mit jeder Bitzeit um eine Stelle verschoben. Soll das gespeicherte Wort nach dem Lesen erhalten bleiben, muss das ausgegebene Bit wieder in die hinterste Stelle geleitet werden. Bei der Übertragung der Operanden ins Rechenwerk, bringt die serielle Übertragung den Vorteil, dass das Rechenwerk immer nur eine Stelle bearbeiten muss und deshalb sehr simpel ist. Wenn Daten unverändert übertragen werden sollen, zum Beispiel zwischen den Steuerregistern, bietet sich die parallele Übertragung an, da sie in einer einzigen Bitzeit abgehandelt wird.
Die Datenübertragung zwischen Kernspeicher und Leitwerk ist parallel. Ein Puffer-

register ist zwischengeschaltet, um zwischen paralleler und serieller Übertragung zu wechseln. [Zuse KG 1969, S. 17-23]

3.3 Rechenwerk

Das Rechenwerk (oder Operationswerk) ist für die Ausführung der verschiedenen Operationen des Z25 Befehlssatzes zuständig. Im Rechenwerk kann sowohl mit Einzel- als auch Doppelwörtern gerechnet werden. Letzteres wird durch zwei Einzelwortoperationen erreicht, mit dem Unterschied, dass der Übertrag zwischen den beiden Operationen erhalten bleibt. [Zuse KG 1969, S. 31]
Es gibt vier verschiedene Operationen die gerechnet werden können: Addition, Subtraktion, Konjunktion und Konjunktion mit negativem zweiten Operand. Zwischen diesen Operationen wird anhand der Steuergrößen A*, S* und I* unterschieden.
Mit den gegebenen vier Operationen, können alle Zuse Z25 Befehle gerechnet werden. Unterschiede zwischen den Befehlen entstehen durch Nutzung unterschiedlicher Operanden (Addition gegenüber Konstantenaddition), durch Doppelwortbefehle oder durch wiederholtes Ausführen (zum Beispiel Multiplikation). Ein großer Teil des Befehlssatzes benutzt das Rechenwerk zudem überhaupt nicht (Transferbefehle und Sprungbefehle).
Die Z25 überträgt die Operanden seriell, das heißt die 18 Bit eines Wortes werden nicht gleichzeitig, sondern eins nach dem anderen übertragen. Für die Befehlsausführung bedeutet das, dass die rechnende Schaltung höchstens drei Bit beachten muss: Operand a, Operand b und, falls vorhanden, der Übertrag u der vorhergegangenen Stelle. Damit ergeben sich insgesamt acht verschiedene Eingabekombinationen. Jede dieser Kombinationen wird durch ein eigenes UND-Gatter geprüft. Es gibt also die Gatter: $a \wedge b \wedge u$, $a \wedge b \wedge \neg u$, $a \wedge \neg b \wedge u$ und so weiter. Diese Gatter werden durch die Steuersignale aktiviert oder deaktiviert und schließlich durch ein ODER-Gatter verbunden. Bei einer Addition werden zum Beispiel die Kombinationen $a \wedge b \wedge u$, $a \wedge \neg b \wedge \neg u$, $\neg a \wedge b \wedge \neg u$ und $\neg a \wedge \neg b \wedge u$ durch das Steuersignal A* zugeschaltet. Liegt eine dieser Kombinationen an, liefert das entsprechende UND-Gatter eine 1 und damit liefert auch das ODER-Gatter eine 1 als Resultat für die Addition der vorliegenden Bits. [Zuse KG 1969, S. 25-31]

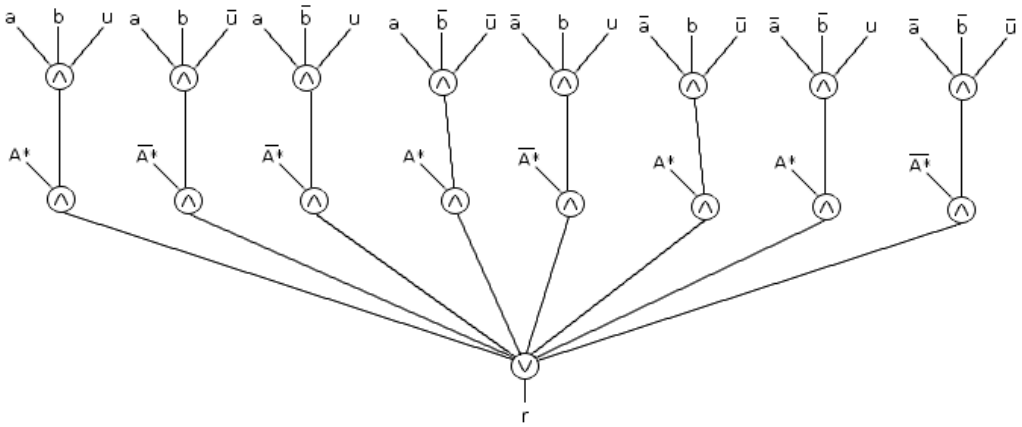

Abbildung 4: Der Schaltbaum für das Resultatbit. Nur Steuerleitung A* ist eingezeichnet. Vgl. [Zuse 1969, S. 28]

Für das Übertragsbit wird ein eigener Schaltbaum und ein Flipflop zum Zwischenspeichern benötigt. Der Schaltbaum funktioniert genau wie der für das Resultat.

3.4 Leitwerk

Das Leitwerk gibt die Steuersignale für das Rechenwerk und kümmert sich um den Programmablauf. Es entschlüsselt Befehle, prüft Bedingungen, inkrementiert den Programmzähler und sorgt dafür, dass der nächste Befehl geladen wird. Bei Befehlen die sich aus mehreren kleineren Befehlen zusammensetzen — zum Beispiel die Multiplikation, die durch mehrmaliges Addieren umgesetzt wird — unterbricht das Leitwerk den normalen Programmablauf und liefert selber die nötigen Einzelbefehle. Folgende Register sind Teil des Leitwerks:
Das Befehlsregister, in dem der aktuelle Befehl komplett gespeichert wird. Es ist mit den 18 Haupttasten der Eingabeschublade verbunden. Dort wird sein Inhalt in den oberen Leuchtfeldern angegeben und kann über die *Befehlsübernahme* Taste verändert werden.
Das Operationssteuerregister OSR, in das der Operationsteil nach Entschlüsselung geschrieben wird. Beim Entschlüsseln werden auch die Steuergrößen für das Rechenwerk gebildet.
Das Adressenerweiterungsregister AER. Es wird von verschiedenen Befehlen direkt angesprochen und wird benutzt um den Adressteil eines Befehls auf 15 Bit zu erweitern.
Das Adressensteuerregister ASR. Hier werden die 10 Bit des Adressteils plus die 5 Bit aus dem AER, falls eine Erweiterung stattfindet, gespeichert. Der gespeicherte Wert wird zum Ansteuern des Speichers verwendet.

Das Sonderadressregister SAR und das Sonderoperationsregister SOR werden benötigt, wenn eine der Sonderadressen 0-31 angesteuert wird.
Das Adressenzählregister AZR ist der Programmzähler der Z25. Es verweist auf die aktuelle Position im Programmcode. [Zuse KG 1969, S. 10 f.]

3.5 Taktung

Die Zuse Z25 hat eine Taktung von 294 kHz. 25 Takte bilden eine Wortzeit — die Dauer um einen einfachen Befehl auszuführen. 18 Davon werden für die Operation gebraucht, da jedes Wort seriell übertragen wird, also ein Bit pro Takt, und 18 Bit lang ist. Die übrigen sieben Bit werden zur Bildung von Steuergrößen und zur Ansteuerung von Speicheradressen verwendet.
Die Z25 benutzt mehrere verschiedene Taktsignale um eine Wortzeit in die verschiedenen Phasen der Befehlsausführung zu unterteilen. Die beiden wichtigsten Signale sind S_0 und $S_{0,5}$. Die steigende Taktflanke von S_0 fällt auf den Anfang einer Bitzeit. S_0 dauert eine viertel Bitzeit an. $S_{0,5}$ dauert ebenfalls eine viertel Bitzeit, die steigende Taktflanke fällt aber auf die Mitte einer Bitzeit. S_0 und $S_{0,5}$ sind also immer um eine halbe Bitzeit phasenverschoben.
Diesen beiden Taktsignalen liegt ein Rechteckgenerator, der mit vierfacher Bitfrequenz arbeitet, zugrunde. An den Generator ist eine aus zwei Flipflops bestehende Zählschaltung angeschlossen. Wenn die Zählschaltung auf 1 steht, wird das Taktsignal S_0 geschickt, wenn sie auf 3 steht, das Signal $S_{0,5}$.

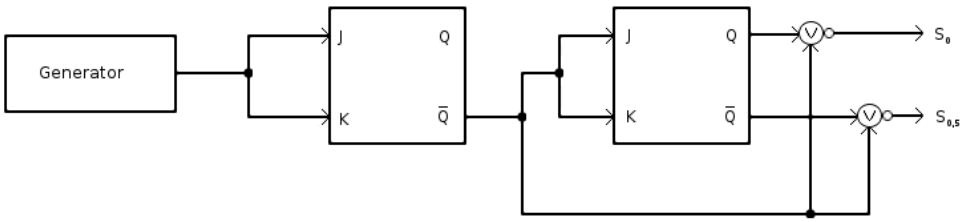

Abbildung 5: Schaltung zur Erzeugung von S_0 und $S_{0,5}$. Vgl. [Zuse 1969, S. 32]

Um nun die 25 Bitzeiten einer Wortzeit zu unterscheiden, werden 25 Taktsignale (Pl_0 bis Pl_{24}) benutzt. Jedes der Taktsignale dauert eine ganze Bitzeit und fällt mit der steigenden Taktflanke auf die Mitte einer Bitzeit. Die Signale werden von 25 Flipflops die zu einem Ring geschaltet sind erzeugt. Da Flipflops nach dem Computerstart in unbekanntem Zustand sind, werden sie zunächst alle auf 0 gestellt.

Daraufhin wird das für Pl_0 stehende Flipflop auf 1 gestellt. $S_{0,5}$ wird nun als Taktsignal an alle Flipflops angelegt. Mit jedem aufkommen von $S_{0,5}$ wandert das gesetzte Bit ein Flipflop weiter. Der Ring wechselt also immer zur Mitte jeder Bitzeit seinen Zustand. Von jedem der Flipflops wird das entsprechende Signal Pl_0 bis Pl_{24} abgezweigt.

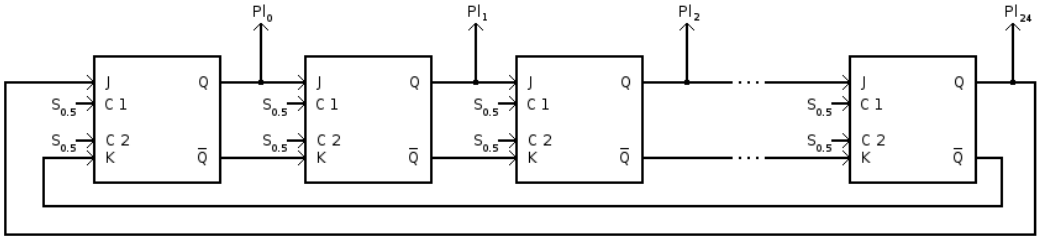

Abbildung 6: Der Schaltring zur Erzeugung von Pl_0 bis Pl_{24}. Die Schaltung zum Nullen nach dem Einschalten, ist nicht eingezeichnet. Vgl. [Zuse 1969, S. 33]

Wenn man eines dieser 25 Signale nun mit dem Signal S_0 durch ein UND-Gatter zusammenschaltet, erhält man ein Taktsignal dessen steigende Flanke auf den Beginn der entsprechenden Bitzeit fällt. Mit diesen Signalen können alle gewünschten Funktionen erreicht werden. [Zuse KG 1969, S. 13 f. und 32-34]

3.6 Fernschreiber

Die vorliegende Z25 wird mit einem Siemens T100 Fernschreiber betrieben; dieses Modell wird deshalb auch im Emulator benutzt. Der Fernschreiber kann zur Ein- und Ausgabe von Text benutzt werden. Ein Lochstreifenleser und ein Lochstreifenstanzer für 5-Kanal Lochstreifen sind ebenfalls vorhanden. Die Kommunikation zwischen Fernschreiber und Zentraleinheit erfolgt im 5 Bit CCIT-Code, der auch für die Lochstreifen benutzt wird. [Zuse KG 1967, Kap. 10 S. 1 f.]
Ein auf der Tastatur angeschlagenes Schriftzeichen wird nicht automatisch auf den Fernschreiber ausgegeben. Es muss von der Z25 eingelesen und wieder in den Fernschreiber zurückgeschrieben werden. Das Parsen der Eingabe in Befehlswörter, Zahlen und Bandbefehle erfolgt in einem Programm auf der Z25. Dieses Programm aus der Maschine auszulesen würde den Rahmen dieser Arbeit überschreiten, und auch in den vorliegenden Handbüchern wird das Programm nicht erläutert. Aus diesen Gründen unterscheidet sich der Emulator in diesem Punkt vom Original. Im Emulator werden eingegebene Zeichen direkt abgedruckt und erst wenn eine Informationseinheit vollständig eingegeben ist, wird sie geparst und an die Z25 übergeben. Auch die Bedienung des Fernschreibers ist von dieser Änderung betroffen; im entsprechenden Kapitel wird darauf genauer eingegangen.

3.7 Magnettrommelspeicher

Die Magnettrommel ist ein Zylinder, dessen Oberfläche magnetisierbar ist. Der Zylinder rotiert während des Betriebs um seine Längsachse. Entlang der Länge der Trommel sind 256 Lese-Schreibköpfe angebracht, die die Oberfläche des Zylinders magnetisieren, beziehungsweise ihre Magnetisierung überprüfen können. Dadurch, dass der Zylinder unter den Lese-Schreibköpfen rotiert, können diese jeder eine Spur auf der Oberfläche des Zylinders beschriften. [Tauschek 1937]
Der Umfang des Zylinders reicht aus, um auf einer Spur 69 Z25 Wörter, inklusive Quersummenkontrollbit, unterzubringen. Bei 256 Spuren, bedeutet das, die Magnettrommel hat eine Kapazität von 17 664 Wörtern, also 335 616 Bit.
Um einen sicheren Lese- und Schreibvorgang gewährleisten zu können, muss sich der Zylinder mit einer konstanten Geschwindigkeit drehen. Die Drehgeschwindigkeit sollte, für eine hohe Lesegeschwindigkeit, zudem möglichst schnell sein. Um diese Geschwindigkeit zu erreichen, muss die Trommel nach dem Einschalten erst etwa zwei Minuten anlaufen. Erst dann ist sie betriebsbereit. [Zuse KG 1967, Kap. 17]

4 Bedienung

Im Folgenden wird die Bedienung der Zuse Z25 und ihr Befehlssatz beschrieben und erläutert. Alles Aufgeführte gilt, wenn nicht anders angegeben, in gleicher Weise für die echte Zuse Z25 und den Emulator.
Zwei Sätze zur Notation: $< m >$ ist der Inhalt der Speicherzelle mit Adresse m. m_{0-9} Sind die Bits 0-9 des Wortes m.

4.1 Bedienung der Ein-/Ausgabeschublade

Die Ein-/Ausgabeschublade ist das zentrale Bedienelement der Zuse Z25. Sie ist die obere rechte Schublade des, mit der Z25 mitgelieferten, Schreibtisches. Es sind 25 Eingabetasten darauf angebracht, mit jeweils ein oder zwei Leuchtfeldern zur Ausgabe. Ein- und Ausgabe erfolgen ausschließlich in binär.

4.1.1 Ein- und Ausschalten der Maschine

Eingeschaltet wird die Maschine über die Taste *Masch EIN / Netz EIN*. Der obere Teil der Taste, ist im eingeschalteten Zustand beleuchtet. Der Untere zeigt an, ob die Maschine an den Strom angeschlossen ist. Direkt nach dem Einschalten befindet sich der Rechner im Stopp-Zustand; der obere Teil der Taste *Stop / Weiter* ist daher beleuchtet.
Zum Ausschalten drückt man ebenfalls die *Masch EIN / Netz EIN* Taste. Dabei ist zu beachten, dass man die Z25 nur im Stopp-Zustand ausschalten sollte, da es sonst zu fehlerhaften Speicherinhalten kommen kann. Da die Z25 Ferritkerne

zum Speichern benutzt, bleiben gespeicherte Inhalte nach dem Ausschalten erhalten. Für den Emulator ist zu beachten, dass Speicherinhalte nur über das Ausschalten der virtuellen Z25, nicht aber der Software selber, erhalten bleiben. Es ist deshalb möglich den Speicherinhalt zu sichern. [Zuse KG 1967, Kap. 7 S. 1 f.]

4.1.2 Einen Befehl eingeben

Um einen Befehl in die Maschine einzugeben, muss dieser auf den beiden mittleren Tastenreihen als binäres Befehlswort eingegeben werden. Diese achtzehn Tasten, im Folgenden als Haupttasten bezeichnet, repräsentieren das Befehlsregister des Computers. Befindet sich der Computer gerade im Stopp-Zustand, leuchtet für jedes gesetzte Bit im Befehlsregister die obere Hälfte der entsprechenden Haupttaste auf der Eingabe-Schublade. Durch drücken einer Haupttaste wird der untere Teil ein- oder ausgeschaltet, was zunächst keinen Einfluss auf das Befehlsregister hat. Erst durch drücken der Taste *Befehlsübernahme* wird die im unteren Teil der Haupttasten vorgenommene Einstellung in das Befehlsregister und somit in den oberen Teil übernommen. [Zuse KG 1967, Kap 7 S. 1f.]
Die oberen acht Tasten bilden den Operationsteil eines Befehlswortes, die unteren Zehn den Parameterteil. Die ersten beiden Tasten, *P* und *Q*, stehen für Bedingungsbits. Ist eine der Tasten oder sind Beide aktiviert, so überprüft der Rechner, ob die jeweilige Bedingung erfüllt ist, und führt nur in diesem Fall den Befehl tatsächlich aus. Dadurch haben Programmierer zum Beispiel die Option Zahlen miteinander zu vergleichen und insbesondere ist es möglich bedingte Sprünge durchzuführen.

Bedingung P: Befehl wird nur ausgeführt, wenn das Spezialregister BD gleich 1 ist. BD wird durch den Befehl PKm auf 1 gesetzt, falls der Inhalt von m negativ ist. Ist der Inhalt von m positiv setzt PKm BD gleich 0. BD bleibt nach der Überprüfung erhalten.
Bedingung Q: Befehl wird nur ausgeführt, wenn der Inhalt des Akkumulators a (Sonderadresse 4) negativ ist.
Bedingung PQ: Befehl wird nur ausgeführt, wenn der Inhalt von a genau null ist.

Die dritte Taste *G* gibt eine Adressenerweiterung oder -substitution an. Der an sich nur zehn Bit lange Parameterteil kann dadurch auf fünfzehn Bit verlängert werden, was für die Adressierung des gesamten Speichers nötig ist. Welche der beiden Optionen ausgeführt wird hängt von dem Befehl ab, der eine Stelle vorher im Speicher steht. Ist der vorige Befehl *Keine Operation*, und sind weder dessen Bedingungsbits noch G gesetzt, wird eine Adressensubstitution ausgeführt, sonst eine Adressenerweiterung.
Bei der Adressenerweiterung werden die Bits 0-9 unverändert aus dem Parameterteil übernommen, die Bits 10-14 werden mit dem Inhalt des Adressenerweiterungsregisters AER belegt. Der Befehl wird nun wie üblich ausgeführt, nur mit einem 15 Bit

langen Parameter. Der Inhalt des AER kann über den Befehl *Adressenerweiterungsregister setzen* festgelegt werden.
Bei der Adressensubstitution wird der Inhalt einer, an anderer Stelle im Speicher gelegenen, Speicherzelle als Parameter genutzt. Außerdem spielt der Parameter n (die Schrittweite) des vorangegangenen Befehls Kn eine Rolle.
Sei folgende Befehlsfolge gegeben:

Kn
GΦm

Wobei gilt: Φ ist ein beliebiger Befehl des Befehlssatzes der Z25. Der Parameter m wird als Adresse gewertet. Die Speicherzelle m wird als Indexspeicherzelle bezeichnet und ihr Inhalt als Index. Die höchsten drei Bits (P, Q und G) des Index modifizieren die Adressensubstitution. Wenn keines der drei Bits gesetzt ist, geschieht Folgendes: Die Bits 10-14 des Index werden in das AER übernommen. Φm̃ wird ausgeführt, wobei $\tilde{m} := Index_{[0-14]} + n$ ist.
Falls Bit 17 (P) des Index gesetzt ist, gehen nur die Bits 0-9 des Index in m̃ ein, der Rest wird mit dem AER aufgefüllt: $\tilde{m} :=< m >_{[0-9]} + n + AER * 2^{10}$. Das AER wird außerdem *nicht* mit dem Index überschrieben, außer es handelt sich bei Φ um einen Sprungbefehl E und die Indexspeicherzelle ist eine der Zellen 32 bis 63. Diese Ausnahme hängt wahrscheinlich mit der Programmunterbrechungsfunktion zusammen, diese speichert nämlich die Rücksprungadresse in den Bereich 32 bis 59.
Wenn Bit 16 (Q) gesetzt ist, wird der um die Schrittweite modifizierte Index in die Indexspeicherzelle geschrieben $< m > + n \rightarrow < m >$. Nur wenn diese Bit gesetzt ist, wird die Indexspeicherzelle modifiziert.
Bit 15 (G) zeigt eine mehrfache Adressensubstitution an. Das bedeutet, dass die Substitution mit dem neuen Parameter m̃ wiederholt wird.

Kn
GΦm̃

Die restlichen fünf Bits des Operationsteils werden zur Kodierung von den 32 verschiedenen Maschinenbefehlen der Z25 verwendet.
Im Parameterteil wird, im Gegensatz zu einer Architektur mit längeren Wörtern, nur ein einzelner Parameter angegeben. Dieser kann eine Konstante oder eine Speicheradresse, aus welcher der Operand ausgelesen werden soll, sein. Zweiter Operand und Zieladresse ist im Regelfall implizit der Akkumulator. [Zuse KG 1967, Kap. 4-5]

4.1.3 Befehlsausführung

Ist der gewünschte Befehl eingegeben und durch die *Befehlsübernahme* Taste in das Befehlsregister übernommen, wird er durch Drücken der Taste *Stop / Weiter*

ausgeführt. *Stop / Weiter* führt immer genau einen Befehl aus und kehrt danach in den Stopp-Zustand zurück. Nach der Befehlsausführung lädt der Computer den nächsten Befehl aus der Stelle des Speichers, auf den das Adressenzählregister (der Programm Counter) verweist. Wenn kein Sprungbefehl ausgeführt wurde, wird das Adressenzählregister in jedem Rechenschritt um eins erhöht.
Drückt man, statt der *Stop / Weiter* Taste, die Taste *Start*, führt der Computer automatisch immer weiter Befehle aus. Um wieder in den Stopp-Zustand zu kommen, muss entweder ein *Stop* Befehl ausgeführt werden oder die Taste *Befehlsübernahme* oder *Stop / Weiter* betätigt werden.
Über die *Befehlsübernahme* Taste werden zudem die Arbeit der Peripheriegeräte abgebrochen. Dies erweist sich als nützlich, falls die Peripherie versehentlich oder falsch angesteuert wurde.
Sollen einem Programm Parameter mitgegeben werden, ist die flexibelste Methode, die Parameter in festgelegte Speicherzellen zu schreiben, die vom Programm während der Laufzeit ausgelesen werden. Darüber hinaus kann man über die Taste *Bed.- Schalter* (Bedingungsschalter) ein einzelnes Bit übergeben. Ist die Taste gerastet, steht in Speicherzelle 6 der Wert 1, und sonst der Wert 0. Um den Programmablauf nun abhängig vom Bedingungsschalter zu variieren, lässt sich der Inhalt von Speicherzelle 6 vom Programm in den Akkumulator laden und mit Bedingung PQ prüfen. [Zuse 1967, Kap. 7 S. 1f.]

4.1.4 *Alarm/Nacht* und *Programm unterbr. frei*

Die Taste *Alarm/Nacht* zeigt im oberen Leuchtfeld an, wenn beim Lesen aus dem Arbeitsspeicher ein Fehler aufgetreten ist. Bei jedem Lesen aus dem Arbeitsspeicher wird überprüft, ob die Quersummenparität des gelesenen Wortes mit dem im Kontrollbit gespeicherten Wert übereinstimmt. Ist das nicht der Fall, wird die Maschine gestoppt und das Alarm Feld leuchtet auf. Der Befehle, bei dem der Lesefehler aufgetreten ist, wird vollständig abgearbeitet, das Adresszählregister AZR zeigt also schon auf den nächsten Befehl. Um die Fehlerstelle zu ermitteln, kann nach dem Alarm der Befehl *Sprung mit Notierung* Fm genutzt werden. Dieser überträgt den Inhalt des AZR in das RAS (Speicherzelle 5).
Wenn die Taste *Alarm/Nacht* gedrückt wird, leuchtet das untere Leuchtfeld auf. Dieses zeigt an, dass die Maschine im Nachtmodus ist. Wenn die Z25, bei eingeschaltetem Nachtmodus, aus dem Start-Zustand in den Stopp-Zustand wechselt, schaltet sie sich automatisch ab. Vor dem erneuten Einschalten muss die *Alarm/Nacht* Taste wieder gedrückt werden, um den Nachtmodus zu beenden. Das Wort „Nacht" ist hier keineswegs metaphorisch gedacht; in den Anwendungsgebieten der Z25 ist es üblich gewesen, den Computer unbeaufsichtigt über Nacht — wenn nicht sogar über mehrere Tage — rechnen zu lassen. Der Nachtmodus kann dem Betreiber in diesem Fall einiges an Stromkosten ersparen.
Mit der Taste *Programm unterbr. frei* wird für die Programmunterbrechungsfunk-

tion Freigabe erteilt oder zurückgezogen. Die Z25 verfügt über 28 Kanäle, an der verschiedene Peripheriegeräte für die Programmunterbrechung angeschlossen werden können. Ein an Kanal n angelegtes Peripheriegerät kann zu einem beliebigen Zeitpunkt einen Impuls an die Z25 senden. Diese unterbricht das aktuell laufende Programm und führt einen Sprung auf Adresse 512 + n aus. Die Rückkehradresse wird in Speicherzelle 32 + n gespeichert. Die Standardspeicherstelle für Rücksprungadressen, das RAS, wird deshalb nicht benutzt, da dieses vielleicht vom laufenden Programm benutzt wird und nicht gesichert werden kann. Die Programmunterbrechung ist im Emulator nicht implementiert, da keines der implementierten Peripheriegeräte die Funktion benutzt. [Zuse KG 1967, Kap. 7 S. 1 f. und Kap. 8]

4.2 Sonderadressen

Die Adressen 0 bis 31 des Arbeitsspeichers sind für spezielle Funktionen reserviert, die hier aufgeführt werden.

Adresse 0: Liefert konstant den Wert 0. Kann nicht überschrieben werden.

Adresse 1: Liefert konstant den Wert 2^{17}, also genau das Vorzeichenbit. Kann nicht überschrieben werden.

Adresse 2: Zählregister Z2.

Adresse 3: Akkumulatorverlängerung v. Wird für die Ausführung von Doppelwortbefehlen benötigt.

Adresse 4: Akkumulator a.

Adresse 5: Rückkehradressenspeicher RAS.

Adresse 6: Bedingungsschalter. Inhalt ist 1, wenn Taste *Bed.- Schalter* aktiviert, sonst ist der Wert 0.

Adresse 7: Externe Sonderadresse für den Fernschreiber (Tastatur und Druckwerk).

Adresse 8: Externe Sonderadresse für den am Fernschreiber angebauten Lochstreifenleser.

Adresse 9: Externe Sonderadresse für den schnellen Lochstreifenleser (im Emulator nicht implementiert).

Adresse 10: Externe Sonderadresse für den schnellen Lochstreifenstanzer (im Emu-

lator nicht implementiert).

Adresse 11: Externe Sonderadresse für den Zeichentisch Zuse Z64 Graphomat (im Emulator nicht implementiert).

Adressen 12 bis 29: Externe Sonderadressen für beliebige Peripheriegeräte (die Magnettrommel ist im Emulator an Adresse 12 angeschlossen).

Adressen 30 und 31: Sonderadressen zum Anschluss weiterer Zuse Z25 Zentraleinheiten um ein Mehrrechner Netzwerk zu bilden. [Zuse KG 1969, S. 12]

4.3 Befehlssatz der Z25

Die Z25 kann 32 verschiedene Befehle ausführen, die sich zum Teil in mehrere Unterbefehle aufteilen. Der Befehlsteil eines Wortes, die Bits 10-14, gibt an, um welchen Befehl es sich bei einem Wort handelt. Die Bits 0-9 bilden den Parameter m, falls nicht anders angegeben. Die Befehlsbeschreibungen sind [Zuse KG 1967, Kap. 3] entnommen.

4.3.1 Keine Operation

Km (0)
Nichts geschieht.

Ausnahme: wenn Bit 17 gesetzt ist, also Befehl PKm, ändert sich der Inhalt des Bedingungsregister BD, abhängig von Speicherzelle m. Ist der Inhalt von m negativ, wird BD auf 1 gesetzt, ist er positiv, setzt PKm BD auf 0. [Zuse KG 1967, Kap. 5]

4.3.2 Addition

Am (1)
Zum Inhalt des Akkumulators wird der Inhalt der Speicherstelle m addiert. Speicherstelle m wird nicht verändert.

$< a > + < m > \rightarrow < a >$

4.3.3 Doppelwortaddition

AAm (2)
Der Inhalt von Akkumulator und Akkumulatorverlängerung wird als Doppelwort, also ein Wort mit 36 Bit Länge, betrachtet. Die Stellen höheren Gewichts, also Bits 18-35, stehen im Akkumulator. Zu diesem Doppelwort wird das Doppelwort addiert, das sich aus den Speicherzellen m und m-1 ergibt. Bei diesem zweiten Doppelwort

stehen in m-1 die Bits höheren Gewichts.

$< a \, und \, v > + < m - 1 \, und \, m > \rightarrow < a \, und \, v >$

Der Übertrag aus der Hälfte niedrigeren Gewichts wird dabei in die Hälfte höheren Gewichts propagiert. Es handelt sich also nicht um zwei reguläre Einzelwortadditionen.
Die Speicherzellen m-1 und m bleiben unberührt.

4.3.4 Bringen

Bm (3)
Der Inhalt der Speicherzelle m wird in den Akkumulator übertragen. Speicherzelle m wird nicht verändert.

$< m > \rightarrow < a >$

4.3.5 Doppelwortbringen

BBm (4)
Das Doppelwort m-1 und m wird in Akkumulator und Akkumulatorverlängerung übertragen. m-1 und m werden nicht verändert.

$< m - 1 \, und \, m > \rightarrow < a \, und \, v >$

Oder einzeln betrachtet:
$< m - 1 > \rightarrow < a >$
$< m > \rightarrow < v >$

Im Akkumulator hält a die Hälfte des Doppelwortes mit höherem Wert, bei m-1 und m sind die gewichtigeren Bits in m-1 gespeichert. Die Zahl 3 wäre als Doppelwort also folgendermaßen gespeichert.

Im Akkumulator:
v: 000000 000000 000011
a: 000000 000000 000000

Im Kernspeicher:
m-1: 000000 000000 000000
m: 000000 000000 000011

Dies gilt für alle Doppelwort-Operationen.

4.3.6 Konjunktion

Im (5)
Stellenweise Konjunktion des Akkumulatorinhalts und des Inhalts von Speicherzelle m. Das Ergebnis wird in den Akkumulator geschrieben. Speicherzelle m bleibt unverändert.

$< a > \wedge < m > \rightarrow < a >$

4.3.7 Disjunktion

DIm (6)
Stellenweise Disjunktion des Akkumulatorinhalts und des Inhalts von Speicherzelle m. Das Ergebnis wird in den Akkumulator geschrieben. Speicherzelle m bleibt unverändert.

$< a > \vee < m > \rightarrow < a >$

4.3.8 Konstantenaddition

CAm (7)
Zum Inhalt des Akkumulators wird der Wert m addiert.

$< a > + m \rightarrow < a >$

4.3.9 Konstantenbringen

CBm (8)
Der Wert m wird in den Akkumulator geschrieben.

$m \rightarrow < a >$

4.3.10 Konstantenkonjunktion

CIm (9)
Stellenweise Konjunktion des Akkumulatorinhalts und des Wertes m. Das Ergebnis wird in den Akkumulator geschrieben.

$< a > \wedge m \rightarrow < a >$

4.3.11 Konstantensubtraktion

CSm (10)
Vom Inhalt des Akkumulators wird der Wert m abgezogen. Das Ergebnis wird in den Akkumulator geschrieben.

$< a > -m \rightarrow < a >$

4.3.12 Konjunktion mit negativer Konstante

CTm (11)
Stellenweise Konjunktion vom Inhalt des Akkumulators und dem Negativen von m (das Negative wird per Zweier-Komplement bestimmt). Das Ergebnis wird in den Akkumulator geschrieben.

$< a > \wedge - m \rightarrow < a >$

4.3.13 Konjunktion mit negativem Operand

ISm (12)
Stellenweise Konjunktion vom Inhalt des Akkumulators und dem Negativen des Inhaltes von Speicherzelle m (Zweier-Komplement). Das Ergebnis wird in den Akkumulator geschrieben. Speicherzelle m bleibt unverändert.

$< a > \wedge - < m > \rightarrow < a >$

4.3.14 Negativbringen

NSm (13)
Das Negative des Inhaltes von Speicherzelle m (Zweier-Komplement) wird in den Akkumulator übernommen. Speicherzelle m bleibt unverändert.

$- < m > \rightarrow < a >$

4.3.15 Subtraktion

Sm (14)
Vom Inhalt des Akkumulators wird der Inhalt der Speicherzelle m abgezogen. Speicherzelle m bleibt unverändert.

$< a > - < m > \rightarrow < a >$

4.3.16 Doppelwortsubtraktion

SSm (15)
Vom Doppelwort aus Akkumulator und Akkumulatorverlängerung wird das Doppelwort aus Speicherzellen m-1 und m abgezogen. Es gelten die selben Besonderheiten, wie bei der Doppelwortaddition.

$< a\ und\ v > - < m-1\ und\ m > \rightarrow < a\ und\ v >$

4.3.17 Umspeichern

Um (16)
Der Inhalt des Akkumulators wird in die Speicherzelle m übertragen. Hier bleibt der Akkumulator unverändert.

$< a > \rightarrow < m >$

4.3.18 Doppelwortumspeichern

UUm (17)
Das Doppelwort aus Akkumulator und Akkumulatorverlängerung wird in die Zellen m-1 und m gespeichert.

$< a\ und\ v > \rightarrow < m-1\ und\ m >$

Oder einzeln betrachtet:
$< a > \rightarrow < m-1 >$
$< v > \rightarrow < m >$

4.3.19 Multiplikation

Mm (18)
Die Inhalte der Speicherzelle m und des Akkumulators werden multipliziert. Der Multiplikator steht in a der Multiplikand in m. Das Ergebnis wird als Doppelwort in a und v gespeichert.

$< a > * < m > \rightarrow < a\ und\ v >$

Nur Kernspeicheradressen sind erlaubt, das heißt: $m \geqslant 32$ muss gelten. Der Inhalt der Speicherzelle m ändert sich durch die Rechnung nicht.
Nach Ausführung des Befehls ist Z2 immer gleich 0, da es benutzt wird um die Anzahl der durchgeführten Additionen zu zählen.

$0 \rightarrow < Z2 >$

Verkürzte Multiplikation:
Ist der Multiplikator weniger als 18 Binärstellen lang, ist es möglich die Rechenzeit der Multiplikation zu verkürzen. Hierzu muss vor dem Multiplikationsbefehl das Register ZS durch den Befehl Zählerladen Zn gegeben worden sein. n bezeichnet die Zahl der Binärstellen (einschließlich Vorzeichen) des Multiplikators. Der Computer führt nun nur n Additionen aus.
Das Ergebnis befindet sich nun um n Stellen nach links verschoben in $< a\,und\,v >$. Im rechten Teil des Akkumulators steht der noch nicht aus dem Register geschobene Teil des Multiplikators.
Nach Ausführung der verkürzten Multiplikation wird das Register ZS gelöscht.

4.3.20 Division

Dm (19)
Der Dividend, der als Doppelwort in Akkumulator und Akkumulatorverlängerung vorliegt, wird durch den Divisor, Inhalt der Speicherzelle m, geteilt. Das Ergebnis steht als einfaches Wort im Akkumulator. In der Akkumulatorverlängerung v steht der Rest.

$< a\,und\,v > / < m > \rightarrow < a >$
$< a\,und\,v > modulo < m > \rightarrow < a >$

Nur Kernspeicheradressen sind erlaubt, das heißt: $m \geqslant 32$ muss gelten.
Der Rest steht in veränderter Form r in v. Um den tatsächlichen Rest R zu erhalten gilt:

$R = (r + Divisor - 1)/2$

Nach Ausführung des Befehls ist Z2 immer gleich 0. Das liegt daran, dass die Z25 tatsächlich 18 Subtraktionen mit unterschiedlicher Stellenverschiebung hintereinander ausführt. Das Register Z2 wird benutzt um diese 18 Schritte mitzuzählen.

$0 \rightarrow < Z2 >$

Verkürzte Division:
Wenn bekannt ist, dass der Quotient einschließlich Vorzeichen nur eine Länge von $n < 18$ hat, ist es möglich die Rechenzeit der Division zu verkürzen. Hierzu muss vor dem Divisionsbefehl das Register ZS durch den Befehl Zählerladen Zn+1 gegeben worden sein. Der Computer führt daraufhin nicht 18, sondern nur n Subtraktionen durch.

Im Akkumulator befindet sich nun das Ergebnis in den Stellen 0 bis n-1.
In Stelle n befindet sich eine 1, falls der Quotient positiv ist, sonst eine 0.
In den Stellen n+1 bis 17 steht der um n+1 nach links verschobene Dividend.
Nach der Ausführung wird das Register ZS gelöscht.

4.3.21 Verschiebe-Befehl

SH (20)
Die Bits im Akkumulator, und gegebenenfalls in der Akkumulatorverlängerung, werden verschoben. Im Parameterteil des Befehls wird in den Stellen 6 bis 9 die Art, und in den Stellen 0 bis 5 die Anzahl der zu verschiebenden Stellen angegeben.
Die Funktionsbits 6 bis 9 sind folgendermaßen zugeordnet:
6: W (zyklisch)
7: V (verkoppelt)
8: R (rechts)
9: L (links)
Die möglichen Kombinationen sind im Folgenden beschreiben.

Linksverschieben (SHLm):
Der Akkumulatorinhalt wird um m+1 Stellen nach links verschoben. Von rechts werden Nullen nachgeschoben.

$< a > * 2^{m+1} \rightarrow < a >$

Verkoppeltes Linksverschieben (SHLVm):
Das Doppelwort in Akkumulator und Verlängerung wird um m+1 nach links verschoben. Von rechts werden Nullen nachgeschoben.

$< a\ und\ v > * 2^{m+1} \rightarrow < a\ und\ v >$

Rechtsverschieben (SHRm):
Der Akkumulatorinhalt wird um m Stellen nach rechts verschoben. Von links wird die Vorzeichenstelle nachgeschoben.

$< a > * 2^{-m} \rightarrow < a >$

Verkoppeltes Rechtsverschieben (SHRVm):
Das Doppelwort in Akkumulator und Verlängerung wird um m nach rechts verschoben. Die Vorzeichenstelle wird von links nachgeschoben.

$< a\ und\ v > * 2^{-m} \rightarrow < a\ und\ v >$

Zyklisches Rechtsverschieben (SHRWm):
Der Akkumulatorinhalt wird um m Stellen nach rechts verschoben. Dabei wird die nach rechts herausgeschobene Stelle links wieder eingeschoben.

Zyklisch verkoppeltes Rechtsverschieben (SHRVWm):
Das Doppelwort in Akkumulator und Verlängerung wird um m Stellen nach rechts verschoben. Dabei wird die nach rechts herausgeschobene Stelle links wieder eingeschoben.

Für alle Rechtsverschiebungen gilt. Ist $m > 18$ wird trotzdem nur um 18 Stellen verschoben und die Zahl der zu viel angegebenen Verschiebungen m-18 in Z2 geladen. Sonst wird Z2 gleich Null gesetzt.

4.3.22 Tausche Inhalt

TIm (21)
Hat nur für m $\leq$ 32 eine Wirkung. Der Inhalt des Adressenerweiterungsregisters AER und der Speicherzelle m werden getauscht. Da das AER nur fünf Stellen hat, werden die übrigen Stellen von m mit Nullen gefüllt

$< m >_{[0-4]} \rightarrow < AER >$
$< AER > \rightarrow < m >$

4.3.23 Umspeichertransfer

UTm (22)
Schreibt $< Z2 > +1$ Wörter ab der Stelle m aus dem Speicher der Z25 in den Eingabepuffer des, durch vorherige Freigabeaktivierung gewählten, Peripheriegerätes.

$< m > \rightarrow < n >$
$< m + 1 > \rightarrow < n + 1 >$
...
$< m + < Z2 >> \rightarrow < n + < Z2 >>$

Die Zieladresse n im Peripheriegerät kann durch Umspeichern von n in die Sonderadresse s des Peripheriegerätes gesteuert werden.
Nach der Ausführung ist Z2 gleich 0.

4.3.24 Bringtransfer

BTm (23)
Schreibt $< Z2 > +1$ Wörter ab der Stelle n aus einem externen Speicher nach Stelle m und Folgende der Z25.

$< n > \rightarrow < m >$
$< n + 1 > \rightarrow < m + 1 >$
...
$< n + < Z2 >> \rightarrow < m + < Z2 >>$

Die Adresse n im Peripheriegerät kann durch Umspeichern von n in die Sonderadresse s des Peripheriegerätes gesteuert werden. Das Peripheriegerät muss vor Ausführung des Bringtransfers durch Freigabeaktivierung aktiviert werden.
Nach der Ausführung ist Z2 gleich 0.

4.3.25 Adressenerweiterungsregister setzen

Hm (24)
Das, zur Adressenerweiterung bei gesetztem Bit G genutzte, Adressenerweiterungsregister AER wird mit m beschrieben. Nur die untersten 5 Bits von m werden geschrieben, da das AER nicht mehr Stellen hat.

$m_{[0-4]} \rightarrow < AER >$

4.3.26 Magnetbandoperation

MB(a + b) (25)
Operation zur Steuerung eines Magnetbandspeichers. Im Emulator wirkungslos, da kein Solcher implementiert ist.
Die Stellen 7 bis 9 des Operationsteils bilden a. Die Stellen 0 bis 2 bilden b. Über a wird eine der folgenden Operationen ausgewählt:
0: Vorwärtslesen
1: Rückwärtslesen
2: Vorwärtsschreiben
3: Rückwärtsschreiben
4: Vorwärtsspringen
5: Rückwärtsspringen
6: Entladen
7: Bandrücklauf
Die Zahl b gibt die Laufwerkadresse an, für die der Befehl ausgeführt werden soll. Für transferierende Befehle (0-3) muss nach MB noch ein Bringtransfer oder Umspeichertransfer geschaltet werden, um die Operation zu vollenden.

4.3.27 Schaltimpuls zur externen Einheit / Freigabeaktivierung

Y/X(g+s) (26)
Abhängig vom Bit an der Stelle 9 (Wert 512) wird einer von zwei Befehlen aus-

geführt. Ist 9 gleich 1 erfolgt Y(g+s), sonst X(g+s).

Schaltimpuls zur externen Einheit (Y(g+s)):
An das an g+s angeschlossene Peripheriegerät wird ein Schaltimpuls mit variabler Länge gegeben. Die Bits 0-4 entsprechen s und geben die Sonderadresse an, an der das Gerät angeschlossen ist. An jeder Sonderadresse können vier Geräte angeschlossen werden; welches genutzt werden soll, wird in den Bits g, 5 und 6, codiert. Die Länge des Schaltimpulses entspricht dem Inhalt des Zählregisters Z2 plus 1. Nach Ausführung des Befehls ist Z2 gleich 0 und das Register ZS ist gelöscht.

Freigabeaktivierung (X(g+s)):
Beim an g+s angeschlossenen Peripheriegerät wird um Freigabe gebeten. Die Freigabe kann durch Testsprung überprüft werden. Die Bits 0-4 entsprechen s und geben die Sonderadresse an, an der das Gerät angeschlossen ist. An jeder Sonderadresse können vier Geräte angeschlossen werden; welches genutzt werden soll, wird in den Bits g, 5 und 6, codiert.

4.3.28 Zählerladen

Zm (27)
Der Wert m wird in das Zählregister Z2 geschrieben und das einstellige Register ZS wird gesetzt.

$m \rightarrow < Z2 >$

Folgende Befehle löschen ZS:

Rechtsverschieben
Verkoppeltes Rechtsverschieben
Zyklisches Rechtsverschieben
Zyklisches verkoppeltes Rechtsverschieben
Freigabeaktivierung
Bringtransfer
Umspeichertransfer
Multiplikation
Divison

Außer den Verschiebe-Befehlen, setzen diese Befehle außerdem Z2 gleich Null. Die Verschiebe-Befehle verändern Z2 auf besondere Weise (siehe Verschiebe-Befehl).

4.3.29 Stop

STm (28)
Die Maschine wechselt in den Stopp-Zustand und lädt einen Sprungbefehl Em in das Befehlsregister. Der Sprung kann sofort durch *Stop/Weiter* ausgeführt werden. Falls G gesetzt war (GSTm), wird GEm in das Befehlsregister geladen.

4.3.30 Testsprung

Tm (29)
Testet das zuletzt angesteuerte Peripheriegerät auf Freigabe. Wenn diese nicht vorliegt wird auf Speicherzelle m gesprungen.

4.3.31 Sprung mit Notierung

Fm (30)
Der, zu diesem Zeitpunkt bereits um Eins erhöhte, Inhalt des Adressenzählregisters AZR — es zeigt an, wo sich der Rechner im Programmcode befindet — wird in das Rückkehradressenregister RAS geschrieben. Danach wird ein Sprungbefehl auf Speicherzelle m ausgeführt.

$< AZR > \rightarrow < RAS >$

Der Befehl wird benutzt um Unterprogramme zu starten. Um wieder ins Hauptprogramm zurück zu springen genügt ein Sprungbefehl E5. E5 springt automatisch, nicht auf die Adresse 5, sondern auf die in Speicherzelle 5 gespeicherte Adresse. Die Bits 10-14 des RAS werden dabei mit Einsen überschrieben.

4.3.32 Sprungbefehl

Em (31)
Statt die nächste Speicherzelle anhand des inkrementierten Inhalts von AZR aus dem Programmcode zu laden, wird das Programm an Stelle m im Speicher fortgesetzt.
Ist das Sprungziel das Rückkehradressenregister RAS (Speicherzelle 5; der Befehl E5 ist also gemeint), wird nicht auf Adresse 5 gesprungen, sondern auf die in Adresse 5 gespeicherte Adresse. Die Bits 10-14 des RAS werden dabei mit Einsen überschrieben. E5 dient somit als Rücksprung aus einem durch *Sprung mit Notierung* gestarteten Unterprogramm.

4.4 Ein- und Ausgabe per Fernschreiber

Der Fernschreiber hat vier verschiedene Funktionen. Eingabe per Tastatur, Drucken, Lochstreifenstanzen und Lochstreifenlesen. Alle Lese- und Schreibfunktionen laufen

mit etwa zehn Zeichen pro Sekunde.

Die Eingabe (sowohl Tastatur als auch Lochstreifen) funktioniert im Emulator anders, als am original Gerät; unter *Aufbau und Funktion* sind die Gründe aufgeführt. Beim Original wird die Eingabe in der Regel durch folgende Befehlsfolge gesteuert:

X(0+7)
B7
SHRW5
U7

X(0+7) erbittet Freigabe am Fernschreiber, der an Adresse 7 angeschlossen ist. Wenn der Lochstreifenleser benutzt wird muss für die ganze Sequenz 8 statt 7 benutzt werden. B7 bewirkt, dass ein Zeichen aus dem Fernschreiber eingelesen wird. Das Zeichen wird gemäß dem CCIT-Code mit fünf Bit gespeichert, die sich an den Stellen 0-4 des Wortes befinden. Der Verschiebe-Befehl SHRW5 bringt das kodierte Zeichen in die Stellen 13-17, die bei der Ausgabe statt 0-4 benutzt werden. Mit U7 wird das Zeichen an den Fernschreiber gegeben, der dieses nun ausdruckt. Ohne diese Befehlsfolge, wird ein angeschlagenes Zeichen nicht ausgedruckt.
Beim Emulator wird jedes an der Tastatur eingegebene Zeichen automatisch ausgedruckt. Wenn per Bringtransfer oder Schaltimpuls Zeichen eingelesen werden, müssen diese ebenfalls nicht, wie es beim Original der Fall ist, von der Z25 darauf geparst werden, ob es sich um Bandbefehle, Befehle, Text oder Zahlen handelt. Die eingegebenen Zeichen werden vom Fernschreiber zu den korrekten Z25 Wörtern kodiert und als solche an die Z25 übergeben.
Wenn der Fernschreiber oder Lochstreifenleser über den Befehl *Bringtransfer* zur Eingabe angeregt werden, wartet die Z25 solange bis die geforderte Zahl an Wörtern eingegeben ist und versetzt das Gerät dann zurück in den Ruhezustand. Wenn zur Anregung ein Schaltimpuls gesendet wurde, kann die Z25 weiter rechnen während die Eingabe erfolgt. Das Eingabegerät wird erst dann in Ruhezustand versetzt, wenn ein E-Bandbefehl eingegeben oder die Taste *Befehlsübernahme* gedrückt wird.
Dass die Tastatur zur Eingabe angeregt wurde, ist an der rechts angebrachten Lampe zu erkennen. Erlischt die Lampe, ist der Fernschreiber wieder im Ruhezustand und es können keine weiteren Zeichen eingegeben werden. Im CCIT-Code bedeuten ein fünf Bit Wort immer eines von zwei Zeichen. Welches der beiden Zeichen richtig ist, hängt davon ab ob zuletzt ein Buchstabenumschaltzeichen Bu (Taste *A...*) oder ein Ziffernumschaltzeichen Zi (Taste *1...*) gegeben wurde. Bu und Zi gelten immer solange bis das jeweils andere Zeichen eingegeben wurde. Beim original Gerät sind die Tasten gesperrt, die nicht mit dem gültigen Umschaltzeichen dargestellt werden, es muss also erst Bu oder Zi eingegeben werden. Der Emulator drückt die Umschalttasten automatisch, wenn versucht wird eine gesperrte Taste zu drücken.

Im Emulator kann die Tastatur sowohl über anklicken der gewünschten Tasten, als auch durch Drücken der entsprechenden Taste auf der Computertastatur bedient werden.
Der Lochstreifenleser zeigt nicht an, dass er zur Eingabe angeregt wurde. Er wartet mit dem Einlesen des Lochstreifens allerdings ab bis die Freigabe Taste (die mit dem ausgefüllten Kreis) am Lochstreifenleser gedrückt wurde. Im Emulator liest der Lochstreifenleser außerdem nur dann, wenn auch ein Lochstreifen eingelegt ist. Um einen Lochstreifen einzulegen, wählen sie den gewünschten Lochstreifen aus dem Lochstreifenmenü und klicken sie auf die Halterung an der Seite des Lochstreifenlesers.

Es gibt vier verschiedene Informationseinheiten, die eingegeben werden können: Bandbefehle, Befehle, Text und Zahlen. Einzelne Informationseinheiten müssen durch Leerzeichen, Wagenrücklauf oder Zeilentransport voneinander getrennt werden.
U-Bandbefehle haben die Form Ub+mU und E-Bandbefehle die Form Eb+mE. b+m gibt jeweils die Zieladresse des Befehls an. b kann ein Wert zwischen 0 und 31 sein und gibt den Speicherbereich an. Der Wert entspricht den Bits 10-14 einer Adresse. m kann zwischen 0 und 1023 liegen und entspricht den Bits 0-9. Der erste Wert und das „+" können weggelassen werden; b ist dann implizit gleich 0.
Ein U-Bandbefehl bewirkt, dass alle nachfolgenden Informationseinheiten, bis zum nächsten Bandbefehl, an die Speicherstelle b+m und Folgende gespeichert werden. Ein E-Bandbefehl bewirkt in der Z25 einen Sprungbefehl auf b+m, bringt die Z25 in Start-Zustand und versetzt das Eingabegerät in Ruhezustand.
Die folgende Folge an Informationseinheiten speichert B90 nach Zelle 100, CA5 nach 101, ST100 nach 102 springt auf Stelle 100 und startet das Programm

```
U100U
B90
CA5
ST100
E100E
```

Befehle werden durch ihre Kurzform, wie sie in *Befehlssatz* aufgeführt sind eingegeben — wobei die Parameter durch Zahlen ersetzt werden müssen. Wenn eines oder mehrere der Bits P,Q und G gesetzt werden soll, werden die Buchstaben einfach vorangestellt. Die Befehle werden nicht ausgeführt, sondern nur entsprechend U-Bandbefehl oder Bringtransfer in die Maschine geschrieben. Beispiele:

```
A30
MB3+0
PK0
```

PQGE100

Die Befehle MB(a+b) und X/Y(g+s) werden ohne Klammern eingegeben. Der Emulator akzeptiert aber auch Eingaben mit Klammern, um Verwirrung vorzubeugen. Außerdem kann der erste Parameter weggelassen werden — er wird dann als 0 interpretiert. Beispiel: X0+7 ist äquivalent zu X7.
Das Handbuch führt außerdem auf, dass Externbefehle mit einem Apostroph angeführt werden müssen. Der Begriff „Externbefehl" taucht an keiner anderen Stelle der Dokumentation auf, es ist deshalb nicht klar, welche Befehle damit gemeint sind.

Text beginnt und endet immer mit Doppelapostroph. Während der Texteingabe können Leerzeichen, Wagenrücklauf und Zeilentransport benutzt werden, ohne das die Informationseinheit abgebrochen wird — alle drei werden ganz einfach als Schriftzeichen übernommen. Je drei Schriftzeichen belegen zusammen ein Z25 Wort. Beispiel:

U100U "Text und Zahl 123"

Der Text steht in den Speicherzellen 100 bis 105 und kann über den Befehl *Umspeichertransfer* wieder auf den Fernschreiber gedruckt werden.

Zahlen können mit und ohne Komma angegeben werden, wobei das Komma als Punkt eingegeben werden muss. Ein Vorzeichen kann angegeben werden, sonst ist die Zahl implizit positiv. Folgende Kombinationen sind erlaubt:

123
+123.45
$-123_{10}3$
$123.45_{10} + 3$
$.123_{10} - 5$

Bei einer Gleitkommazahl werden Mantisse und Exponent auf zwei Speicherstellen aufgeteilt; erst Mantisse dann Exponent. Es ist zu beachten, dass der Exponent stets zur Basis Zehn gedacht ist.

Ist bei der Eingabe einer Informationseinheit (außer Text) ein Fehler aufgetreten, können die bisher eingegebenen Zeichen durch Eingabe ZiBu oder BuBu gelöscht werden. Dies ist nur solange möglich, wie die Informationseinheit noch nicht abgeschlossen wurde. Die Löschung ist am Ausdruck nicht erkennbar.
Wird ein fehlerhafter Bandbefehl eingegeben, druckt der Fernschreiber die Fehlermeldung „BDBF31+446". Der emulierte Fernschreiber druckt die selbe Fehlermel-

dung auch bei anderen fehlerhaften Informationseinheiten, damit ein Eingabefehler nicht ungesehen durchgeht.

Um Schriftzeichen über das Druckwerk des Fernschreibers auszugeben müssen die Schriftzeichen per Befehl *Umspeichern* oder Befehl *Umspeichertransfer* an den Fernschreiber übergeben werden. Beim Umspeichern wird das in Bits 13-17 stehende Zeichen des Wortes ausgegeben, beim Umspeichertransfer werden drei Bits pro Wort (Bits 13-17, 7-11, 1-5) und gegebenenfalls mehrere Wörter ausgegeben. Soll eine Zahl auf den Fernschreiber ausgegeben werden, muss diese erst in die entsprechenden Ziffern Schriftzeichen überführt werden; der Fernschreiber kann nur CCIT-Code ausgeben. Um die Lesbarkeit zu gewährleisten, druckt der emulierte Fernschreiber deutlich größer, als in Realität. Das führt dazu, dass jede Zeile weniger Zeichen aufnehmen kann; gegebenenfalls müssen mehr Zeilenumbrüche benutzt werden, als beim Original.

Der Lochstreifenstanzer ist an der linken Seite des Fernschreibers angebracht. Um einen Lochstreifen zu stanzen muss die Freigabetaste (die mit dem gefüllten Kreis) gedrückt werden. Nun wird alles was an der Fernschreiber Tastatur eingegeben wird in den Lochstreifen gestanzt. Um das Stanzen zu beenden wird die Taste mit dem leeren Kreis gedrückt. Im Emulator sorgt ein Doppelklick auf den gestanzten Lochstreifen dafür, dass der Lochstreifen abgerissen und ins Lochstreifenmenü übernommen wird. [Zuse KG ohne Jahr, Kap. IV]

4.5 Bedienung der Magnettrommel

Die Zuse Magnettrommel hat 256 Spuren mit je 69 Z25 Wörtern (inklusive Quersummenkontrolle), insgesamt 17 664 Wörter, an Speicher. An der Front der Magnettrommel sind drei Leuchtfelder und ein elektrisches Schloss angebracht.
Das Schloss wird benutzt um die Magnettrommel zu starten und wieder abzuschalten. Die Funktion ist bei laufender Z25 gesperrt, die Magnettrommel muss also vor der Z25 gestartet werden. Die Trommel benötigt etwa zwei Minuten Anlaufzeit, bevor sie betriebsbereit ist. Während der Anlaufzeit leuchtet das Feld *Vorwahl*. Wenn sie bereit ist, leuchtet das Feld *Fertig* auf und *Vorwahl* erlischt. Das Leuchtfeld *Alarm* leuchtet auf, wenn die Quersummenkontrolle beim Lesen einen Fehler feststellt.
Die Leuchtfelder *Alarm*, *Fertig* und *Trommel Ein* sind alle auch auf der Bedienschublade angebracht. Dort befindet sich auch noch ein zweites elektronisches Schloss mit gleicher Funktion. Im Emulator arbeitet nur das am Trommelschrank angebrachte Schloss.
Im Inneren des Trommelschrankes sind Kippschalter angebracht, mit denen Spuren zu je 16 Stück gesperrt werden können. Diese Schalter wurden im Emulator weggelassen. [Zuse KG 1967, Kap. 17]

4.6 Zusätzliche Funktionen des Emulators

Die verschiedenen Knöpfe der Zuse Z25 und der vorhandenen Peripheriegeräte werden über einfaches Anklicken bedient. Die Tastatur des Fernschreibers kann zudem mit der Computertastatur bedient werden. Bei gedrückter rechter Maustaste, kann die Kamera per Mausbewegung rotiert werden. Wird eines der Peripheriegeräte oder die Ein-/Ausgabeschublade angeklickt, bewegt sich die Kamera darauf zu. Wird der Fernschreiber aktiviert, springt die Kamera automatisch zu ihm.
In der oberen rechten Ecke befinden sich zwei Schaltflächen. Die Eine bringt die Kamera in die Ausgangsposition zurück, die Andere öffnet die Bedienungsanleitung. Am linken Rand des Bildschirms ist ein Knopf zu sehen. Wird dieser gedrückt, öffnet sich ein Interface auf der linken Seite. Über ein Dropdown Menü am oberen Interface Rand, wechselt man zwischen verschiedenen Anzeigemodi, die im Folgenden aufgeführt und erklärt werden.

4.6.1 Speicherübersicht

Unter *Speicherübersicht*, ist es dem Nutzer möglich den Speicher des Computers einzusehen. Es werden genau die Speicherzellen angezeigt, deren Inhalt ungleich 0 ist. Man kann zwischen Dezimal- und Binärdarstellung wechseln, sowie der Darstellung als Programm-Befehl. Mit der Option *Übersicht aktivieren* kann die Aktualisierung der Übersicht ein- oder ausgeschaltet werden. Die Option ist vorhanden, weil die ständige Aktualisierung der Speicherübersicht, bei zunehmend gefülltem Speicher, viel Rechenleistung erfordert.
Da der Computer Mikrobefehl für Mikrobefehl bedient wird, ist es nicht immer leicht den Überblick über den laufenden Rechenvorgang zu behalten. Deshalb werden im unteren Bereich der Speicherübersicht die letzten zehn gegebenen Befehle aufgelistet. Hier werden auch die Befehle aufgeführt, die aufgrund einer nicht erfüllten Bedingung nicht ausgeführt wurden, es sollten also auch die im Speicher angezeigten Ergebnisse und das AZR beachtet werden.
Rechts, zwischen den beiden Anzeigen, wird der Inhalt des Adressenzählregisters AZR angezeigt. Es fungiert als Programmzähler der Z25 und verweist auf die Stelle des Speichers, aus der der aktuelle Befehl gelesen wurde.
Diese Übersichten führen zu einer erheblichen Erleichterung der Bedienung. Die original Maschine kann Speicherzellen nur einzeln anzeigen, und das auch nur indem ein Sprungbefehl auf die entsprechende Speicherzelle durchgeführt wird. Eine Überwachung während des laufenden Programms ist nicht möglich. Es gibt zudem keine Möglichkeit den Inhalt des AZR, also die aktuelle Stelle im Programmcode, herauszufinden. Beides führt dazu, dass Debuggen fast nur mit Papier und Stift möglich ist. Am Emulator zu programmieren ist deshalb deutlich komfortabler, als an der echten Z25.

4.6.2 Befehlsübersicht

In der Befehlsübersicht sind alle 32 Befehle der Z25 aufgeführt. Zu jedem Befehl gehören zwei Knöpfe. Der Linke ist mit dem Namen des Befehls beschriftet, der Rechte mit seinem Operationscode. Ein Klick auf den linken Knopf öffnet ein Beschreibungsfenster, in dem die Funktion des Befehls genauer erläutert wird, ähnlich wie es hier im Kapitel *Befehlssatz* getan wird. Mit einem Klick auf den rechten Knopf wird der entsprechende Operationscode in die Haupttasten übernommen.

4.6.3 Lochstreifenmenü

Im Lochstreifenmenü werden alle aktuell geladenen Lochstreifen aufgelistet. Das Anklicken eines Lochstreifens markiert diesen. Ein markierter Lochstreifen kann gespeichert oder in den Lochstreifenleser eingehängt werden. Beide Optionen entfernen den Lochstreifen nicht aus dem Menü. Soll ein Lochstreifen gespeichert werden, muss ein Dateiname, unter dem er wieder geladen werden kann, angegeben werden. Die Lochstreifendateien werden im Ordner „Saves" im Data-Ordner des Emulators, mit Dateiendung „.ls", abgelegt. Wird ein Lochstreifen, durch einen Klick auf *Laden* und Eingabe des entsprechenden Dateinamens, geladen, erscheint er an letzter Stelle im Lochstreifenmenü. Das Selbe passiert mit einem Lochstreifen, der vom Lochstreifenstanzer abgerissen wird.
Dateien können überschrieben werden. Soll das für eine bestimmte Datei unterbunden werden, muss die Datei unter Windows als schreibgeschützt deklariert werden.

4.6.4 Optionen

Verschiedene Optionen zur Benutzung des Emulators. Über *Audio* wird der Ton ein- oder ausgeschaltet. Mit der Schaltfläche *Stromanschluss* kann bestimmt werden, ob der virtuelle Computer gerade an das Stromnetz angeschlossen ist oder nicht. Zu besseren Lesbarkeit der Knöpfe, sind die Knopfbeschriftungen im Emulator größer als im Original. Mit ausgeschalteter Option *Vergrößerte Knopfbeschriftung*, bekommt der Nutzer den korrekten Maßstab angezeigt. Der Knopf *Maschine zurücksetzen* bringt den Emulator wieder in Ausgangszustand: der Speicher ist leer. Durch Klick auf *Emulator beenden*, lässt sich die Software beenden.
Mit den Schaltflächen *Speichern* und *Laden* kann der Speicherinhalt von Z25 und Magnettrommel gesichert werden. Ein Dateiname muss eingegeben werden mit dem eine Speicherdatei erstellt wird. Die Dateiendung ist „.z25" und wird automatisch angehängt. Die Speicherdateien sind im Ordner „Saves" im Data-Ordner des Emulators zu finden. Zum Laden muss der entsprechende Dateiname in das Eingabefeld eingegeben werden.
Dateien können überschrieben werden. Soll das für eine bestimmte Datei unterbunden werden, muss die Datei unter Windows als schreibgeschützt deklariert werden.

4.7 Beispielprogramme

Im Folgenden sind zwei kommentierte Beispielprogramme aufgeführt. Das erste Programm summiert die Zahlen 1 bis x, wobei x die Zahl ist, die zum Programmstart im Akkumulator steht. Das zweite Programm findet alle Primzahlen bis Zahl x. Auch hier steht x zum Programmstart im Akkumulator.

4.7.1 Aufsummieren der Zahlen 1 bis x

Das Programm ist als Lochstreifen unter dem Namen „Aufsummieren" gespeichert und liegt dem Emulator bei. Der Lochstreifen enthält die nötigen Bandbefehle, um das Programm an der richtigen Stelle in den Speicher zu laden und zu starten. Soll das Programm manuell gestartet werden, muss zunächst die Zielzahl x in den Akkumulator gebracht werden. Danach kann mit Sprung auf 100, und anschließendem Drücken der *Start* Taste das Programm in Gang gebracht werden.
Zusätzlich zu dieser, liegt eine Version für Doppelwörter unter „AufsummierenDoppelwort" vor. Diese funktioniert bis auf die Verwendung von Doppelwortbefehlen genau gleich. Es gilt zu beachten, dass die höherwertigen Bits eines Doppelwortes im Akkumulator (Speicherzelle 4) liegen. Die beiden Versionen liefern bei gleicher Eingabe also unterschiedliche Ergebnisse.

Ab Speicherstelle 100:

U99	Speichert x nach Zelle 99.
CB0	97 und 98 werden gleich 0 gesetzt.
U97	97 hält Laufvariable i.
U98	In 98 wird das Zwischenergebnis gesichert.
B99	Überprüfung ob $i > x$.
S97	In dem Fall ist das Programm fertig.
QST98	
B98	Das Zwischenergebnis wird um i erhöht.
A97	
U98	
B97	i wird um 1 erhöht.
CA1	
U97	
E104	Sprung auf den Befehl BB99.

4.7.2 Primzahlen bis zur Zahl x

Das Programm geht die Zahlen 2 bis x nacheinander durch und prüft alle bisher gefundenen Primzahlen als Teiler. Jede Zahl für die kein Teiler gefunden wird, ist eine Primzahl und wird in eine der Speicherzellen ab Zelle 1025 gespeichert. Zum Programmstart ist die erste Primzahl 2 bereits bekannt (steht in Speicherstelle 1025) und x steht im Akkumulator. Von Speicherzelle 180 aus, kann das Programm gestartet werden.
Ein Lochstreifen, der das Programm in den Speicher lädt und automatisch startet, ist mit dem Namen „Primzahlen" gespeichert und liegt dem Emulator bei.

Speicherstellen:

164:	2	Aktuelle Prüfzahl i.
165:	0	Eingabe x.
166:	0	Laufvariable j.
167:	1	Anzahl Primzahlen bisher p.
168:	0	Aktueller Divisor d.
169:	QA0	Indexspeicherstelle. Verweist auf 1024.
170:	QA0	Zum Zurücksetzen des Index.
1025:	2	Erste Primzahl.

Programmstartpunkt. Ab Speicherstelle 180:

U165	x speichern.
CB2	i und p zurücksetzen, falls das Programm
U164	mehrmals hintereinander ausgeführt wird.
CB1	
U167	
B170	Index zurücksetzen.
U169	
B165	Schleifen Anfang.
S164	
QST179	Programmende falls $i > x$.
F195	Aufruf des Unterprogramms, das i prüft.
B164	i um 1 erhöhen.
CA1	
U164	
E187	Schleifen Ende.

Unterprogramm das prüft, ob die Zahl i eine Primzahl ist. Ab Speicherstelle 195:

B167	j gleich p, um p-mal die Schleife auszuführen.
U166	
B164	Schleifen Anfang.
U3	i nach v. Wird gleich zum dividieren benutzt.
K1	Adressensubstitution mit Veränderung des Index.
GB169	Bewirkt, dass die nächste Primzahl geladen wird.
U168	d gleich gerade geladene Primzahl.
CB0	i steht jetzt als Doppelwort im Akkumulator.
D168	i wird durch d geteilt.
B3	Rest wird in den Akkumulator gebracht.
A168	Rest wird korrigiert.
CS1	
SHR1	
PQE220	Falls Rest gleich 0, wurde ein Teiler gefunden.
B166	j um 1 verringern.
CS1	
PQE214	Falls j gleich 0 wurden alle Primzahlen ausprobiert.
U166	Sonst mit nächster Primzahl wiederholen.
E197	Schleifen Ende.
B164	Sprungziel, falls alle Primzahlen ausprobiert. i ist eine Primzahl.
K1	
GU169	Hinzufügen von i zur Liste der Primzahlen.
B167	p um 1 erhöhen.
CA1	
U167	
B170	Sprungziel, falls Teiler gefunden. i ist keine Primzahl.
U169	Index zurücksetzen.
E5	Rücksprung ins Hauptprogramm.

4.7.3 Weitere Programme

Neben den beiden vorgestellten, liegen dem Emulator noch weitere Programme bei, die hier, um nicht den Rahmen zu sprengen, nicht genauer erläutert werden. Die folgenden Programme liegen alle als *.z25* Dateien vor, müssen also nicht über das Lochstreifenmenü, sondern über die Optionen geladen werden.
Das als „Zahlenausgabe" gespeicherte Programm, kann eine Zahl auf den Fernschreiber ausgeben. Die Zahl muss dafür in den Akkumulator geladen, und dann das Programm ab Stelle 65 ausgeführt werden. Sowohl negative als auch positive Zahlen sind möglich.
Unter „AufsummierenMitAusgabe" ist das oben aufgeführte Programm zum Auf-

summieren gespeichert, mit dem Zusatz, dass das Ergebnis auf den Fernschreiber gedruckt wird. Genau wie oben, wird die Zielzahl x in den Akkumulator geladen, und dann das Programm ab Speicherstelle 100 gestartet.
„PrimzahlenMitAusgabe" enthält eine Version des obigen Primzahl-Programms, das die gefunden Primzahlen auf den Fernschreiber druckt. Die Zielzahl muss in den Akkumulator geladen, und dann das Programm ab Speicherstelle 65 gestartet werden.

5 Vorgehensweise beim Erstellen des Emulators

Die Emulator Software wurde in Unity 2017.1 Personal Edition mit der Programmiersprache C# entwickelt. Der Abgabe liegt eine für Windows kompilierte Datei bei, sowie eine Webbrowser Version. Die Webversion ist qualitativ eingeschränkt und die Funktionen zum Laden und Speichern funktionieren nicht. Mit zusätzlichem Aufwand, wäre es möglich, die Speicherfunktion (entweder via Cookies oder auf einem Server) auch für die Webanwendung zu aktivieren. Es wäre auch möglich die Funktionen ganz zu entfernen und alle Standardprogramm-Lochstreifen bei Start des Emulators automatisch zu laden.
Die verwendeten 3D Modelle sind in Blender 2.77 entstanden. Der Ton wurde an der echten Z25 aufgezeichnet. Für Fußboden, Tischplatte und Tapete, sind Bilder von *freestocktexture.com* und *photos-public-domain.com* verwendet.
Bei der Programmierung wurde darauf geachtet die internen Zustände der Maschine originalgetreu nachzubilden und die selben Rechenmethoden zu verwenden, so weit dies sinnvoll war. Zum Beispiel wird die Multiplikation durch stellenweise Verschiebung und Addition realisiert, um sicherzustellen, dass Rechenfehler — zum Beispiel durch Fehler in der Programmierung — korrekt dargestellt werden. Auf eine Umsetzung des seriellen Rechenwerks wurde hingegen verzichtet, da sich dieser Umstand auf die errechneten Ergebnisse nicht auswirkt und den Rechenaufwand stark erhöht.
Probleme traten im Verlauf der Entwicklung vor allem wegen Sonderfällen und undurchsichtigen Abhängigkeiten auf. Zum Beispiel können verschiedene Umstände dafür sorgen, die Maschine in Stopp-Zustand zu versetzen. Der Stopp-Zustand kann wiederum zum Beispiel den Nacht-Zustand auslösen. Diese Umstände werden an verschiedenen Stellen der Handbücher erklärt, so dass der Programmcode mit zunehmender Funktionalität immer wieder nachgebessert und umgeschrieben werden musste.
Unklarheiten gab es auch bei der Beschreibung der Peripheriegeräte. So wird zum Beispiel nicht geklärt, ob der Fernschreiber mehrere Schriftzeichen in den Ein-/Ausgabepuffer schreiben kann, oder ob jedes Zeichen erst von der Z25 gelesen werden muss, bevor das Nächste angeschlagen werden darf (für den Emulator wurde ein Puffer beliebiger Größe gewählt).

Literatur

[1] Konrad Zuse: *Der Computer - Mein Lebenswerk*. Springer, Heidelberg, 2. Auflage, 1986. [Zuse 1986]

[2] Professor Dr.-Ing. habil. Horst Zuse, „Z3". *http://www.horst-zuse.homepage.t-online.de/z3.html*. (Stand: 26.10.2017). [Dr. Zuse 2017a]

[3] Professor Dr.-Ing. habil. Horst Zuse, „Z11". *http://www.horst-zuse.homepage.t-online.de/z11.html*. (Stand: 26.10.2017). [Dr. Zuse 2017b]

[4] Professor Dr.-Ing. habil. Horst Zuse, „Z22". *http://www.horst-zuse.homepage.t-online.de/z22.html*. (Stand: 26.10.2017). [Dr. Zuse 2017c]

[5] Professor Dr.-Ing. habil. Horst Zuse, „Z23". *http://www.horst-zuse.homepage.t-online.de/z23.html*. (Stand: 26.10.2017). [Dr. Zuse 2017d]

[6] Professor Dr.-Ing. habil. Horst Zuse, „Z64". *http://www.horst-zuse.homepage.t-online.de/z64.html*. (Stand: 26.10.2017). [Dr. Zuse 2017e]

[7] Professor Dr.-Ing. habil. Horst Zuse, „Z25". *http://www.horst-zuse.homepage.t-online.de/z25.html*. (Stand: 26.10.2017). [Dr. Zuse 2017f]

[8] Professor Dr.-Ing. habil. Horst Zuse, „Z31". *http://www.horst-zuse.homepage.t-online.de/z31.html*. (Stand: 26.10.2017). [Dr. Zuse 2017g]

[9] Professor Dr.-Ing. habil. Horst Zuse, „Z26". *http://www.horst-zuse.homepage.t-online.de/z26.html*. (Stand: 26.10.2017). [Dr. Zuse 2017h]

[10] Zuse KG: *Datenverarbeitungssystem Zuse Z25 Beschreibung und Einführung in die Arbeitsweise*. 1967. [Zuse KG 1967]

[11] Zuse KG: *Aufbau und logische Abläufe: Zentraleinheit Zuse Z25 Handbuch*. 1969. [Zuse KG 1969]

[12] Zuse KG: *Anleitung zur Inbetriebnahme und Ablochvorschrift Zuse Z25 Handbuch*. ohne Jahr, Kapitel IV. [Zuse KG ohne Jahr]

[13] Gustav Tauschek: *Elektromagnetischer Speicher für Zahlen und andere Angaben, besonders für Buchführungseinrichtungen* . Deutsches Patent DE 643803A 17.04.1937. [Tauschek 1937]

Weiterführende Literatur

Wilhelm Mons, Horst Zuse, Roland Vollmar: *Konrad Zuse*. Ernst Freiberg-Stiftung, 2005.

Jürgen Alex, Hermann Flessner, Wilhelm Mons, Kurt Pauli, Horst Zuse: *Konrad Zuse: Der Vater des Computers*. Verlag Parzeller, 2000.

Eidesstattliche Erklärung zur Bachelorarbeit

Ich erkläre hiermit, dass ich die vorliegende Bachelorarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, sowie Zitate kenntlich gemacht habe.
Die Arbeit ist weder einer anderen Prüfungsbehörde vorgelegt, noch veröffentlicht worden.

Ort, Datum	Unterschrift