Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

WOLFGANG K. GILOI

# Die Ungnade der frühen Geburt

*Vortrag, gehalten am 12.12.1996 in memoriam Konrad Zuse*

Preprint SC 97-09 (Juni 1997)

# Die Ungnade der frühen Geburt

*Vortrag gehalten am 12.12.1996 in memoriam Konrad Zuse*

Wolfgang K. Giloi


*Abb. 1 Konrad Zuse mit dem Nachbau der Z1*

## Einleitung

Konrad Zuse ist in unserem Land bekannt als der Computerpionier, der den ersten funktionstüchtigen Rechner in der Geschichte der Menschheit realisierte. Weniger bekannt ist aber immer noch die Tatsache, daß er mit seinem Plankalkül auch die erste höhere Programmiersprache der Welt entwickelte. Zuse hatte bereits 1945 nicht nur klare Vorstellungen über die Konzepte von höheren

Programmiersprachen, sondern auch über die Programmierung von Anwendungen der künstlichen Intelligenz wie der relationalen Datenspeicherung oder dem Schachspiel. Dies macht ihn auch zum ersten Informatiker der Welt und darüber hinaus zum Erfinder der künstlichen Intelligenz.

Auf die ein Jahrzehnt nach Zuses Pionierleistungen beginnende Entwicklung der Programmiermethodik und der künstlichen Intelligenz hatten diese weitgehenden Vorarbeiten jedoch keinen Einfluß. International blieben sie durch die Kriegs- und Nachkriegsverhältnisse lange unbekannt, aber auch im eigenen Lande waren sie zu früh gekommen, um in ihrer Bedeutung verstanden zu werden. Bis heute würdigt die internationale Gemeinde der Informatiker, Computerwissenschaftler und Ingenieure den Genius von Konrad Zuse nicht in gebührendem Maße. In diesem Vortrag werden wir versuchen, die Beiträge, die Konrad Zuse als Erfinder und Wissenschaftler geleistet hat sowie die Prioritäten, die er auf verschiedenen Gebieten beanspruchen kann, gebührend zu würdigen. Dabei werden wir auch versuchen, den Menschen Konrad Zuse, so wie wir ihn aus vielen Begegnungen kannten, hervortreten zu lassen.

## Der Erfinder

Konrad Zuse ist mitunter in der ffentlichkeit als der Erfinder der programmgesteuerten digitalen Rechenmaschine, kurz "Computer" genannt, bezeichnet worden. Dies ist jedoch eine journalistische Verkürzung, da die Idee einer programmierbaren Rechenmaschine in allen wesentlichen Zügen bereits von dem englischen Mathematiker Charles Babbage (1792-1871) entwickelt wurde. Babbage arbeitete viele Jahre an der Realisierung eines programmierbaren mechanischen Computers, den er Analytical Engine nannte. Die Konstruktion der Analytical Engine überstieg jedoch die technischen Möglichkeiten seiner Zeit, so daß es Babbage nicht gelang, ein funktionsfähiges Modell zu erstellen. Daher stellte 1842 die britische Regierung, die Babbages Entwicklung jahrelang finanziell unterstützt hatte, die weitere Förderung ein.

Den ersten funktionsfähigen mechanischen Digitalrechner, aber auch den ersten mit Relais arbeitenden Computer, realisierte in der Tat Konrad Zuse. Kann man ihn deshalb den Erfinder des Computers nennen?

Nicht nach amerikanischer Rechtsprechung. Im Jahre 1964 entschied der ehrenwerte Richter Earl Larson vom US District Court in Minneapolis in dem Rechtsstreit *Honeywell vs. Sperry*, daß der Erfinder des Computers ein gewisser Dr. John Vincent Atanasoff sei [1]; also weder Babbage, noch Zuse, noch Eckert und Mauchley, die Konstrukteure des ENIAC, des ersten arbeitsfähigen elektronischen Rechenautomaten, noch John von Neumann, nach dem der sequentiell arbeitende Computer auch als *von Neumann - Maschine* bezeichnet wird, sondern Atanasoff, von dem bis dahin kaum jemand etwas wußte. Dieses Urteil, dem die Fachwelt bis heute mit Kopfschütteln bis Gelächter begegnet, gilt immer noch. Die Stadt Hünfeld, die für ihren Ehrenbürger Zuse lediglich den Anspruch erhebt, daß er den ersten *funktionsfähigen* Computer der Welt baute, war offenbar rechtlich gut beraten.

Die internationale Anerkennung dieser Priorität blieb Zuse jedoch versagt. Die Computer Society des *Institute of Electrical and Electronics Engineers* (IEEE), der Welt größte Vereinigung von Computerwissenschaftlern und - ingenieuren, schuf 1980 den *Computer Pioneer Award*, gedacht als eine Art Nobelpreis für Computerpioniere. Der Preis für den ersten funktionsfähigen Computer ging aber nicht an Konrad Zuse, dessen mit Relais arbeitender Digitalrechner Z3 bereits 1941 lief, sondern an Howard Aiken, dessen (ebenfalls mit Relais arbeitender) Rechner Mark I erst drei Jahre später funktionsfähig war. Da man im Jahre 1980 nicht mehr vorgeben konnte, von Zuses Arbeiten nichts zu wissen, gab man auch ihm den Pioneer Award "für den ersten Prozeßrechner".

Zuse wurde bei Kriegsausbruch zunächst eingezogen, bald darauf aber als Statiker bei den Henschel-Flugzeugwerken "uk-gestellt". Bei Henschel hatte er die Aufgabe, die Korrekturen zu berechnen, die individuell an jedem der in billiger Blechtechnik mit groben Toleranzen gefertigten unbemannten Flugkörper HS 293 an den Tragflügeln vorzunehmen waren, um diese flugfähig zu machen. Für Zuse war nichts naheliegender, als dazu seinen Rechner einzusetzen. Zu diesem Zweck baute er die mit Relais arbeitenden Spezialrechner S1 und S2. Der S2 verwirklichte die absolut neuartige Idee, durch einen in den Rechner integrierten "Analog/Digital-Wandler" die von Meßfühlern gelieferten *analogen Meßwerte* unter Programmkontrolle in entsprechende *digitale Größen* umzuwandeln, damit der S2 sie dann weiterverarbeiten konnte. Er hat es später sehr bedauert, für die Verbindung eines Analog/Digital-Wandlers mit einem Rechenautomaten kein Patent angemeldet zu haben.

Die Ironie ist, daß Zuse für den praktisch eingesetzten Rechner S2 nur den Preis für eine spezielle Anwendung erhielt, der Preis für den ersten funktionsfähigen Rechner der Welt schlechthin aber für eine Entwicklung vergeben wurde, die in gleicher Technik und nach gleichen Prinzipien, aber erst drei Jahre später arbeitete. Es sei noch angemerkt, daß die Arbeit bei Henschel Zuse nicht nur den Pioneer Award eintrug, sondern in den siebziger Jahren auch Angriffe von Studenten, die ihn als "Bombenbastler Zuse" verunglimpften. Es war für mich damals sehr eindrucksvoll, die geistige Sovereignität Zuses zu erleben, mit der dieser nicht nur auf diese Angriffe reagierte, sondern auch dazu stand, daß er als in Deutschland lebender Ingenieur ungewollt zu einem Rädchen in der Kriegsmaschine des "Dritten Reiches" wurde. Im übrigen wurden seine Rechnerentwicklungen als "nicht kriegswichtig" eingestuft und damit (damals zu seinem Leidwesen) nicht staatlich gefördert.

Im Jahre 1945 schrieben *von Neumann, Burke, and Goldstine* die berühmt gewordene Studie "*Preliminary Discussion of the Logical Design of an Electronic Computing Instrument*" [2], in der erstmalig die Arbeitsprinzipien des sequentiell arbeitenden Digitalrechners [1] beschrieben wird. Dies führte dazu, daß bis heute der sequentielle Rechner auch als "von Neumann - Maschine" bezeichnet wird. Die Arbeitsprinzipien des sequentiellen Digitalrechners hatte Zuse aber

[1] Der klassische Digitalrechner verarbeitet mit jedem Rechenschritt einen einzelnen (skalaren) Wert. Um das gewünschte Ergebnis zu erzielen, sind damit viele Rechenschritte nötig. Erst in den sechziger Jahren entstand die Idee des Parallelrechners, der mit jedem Rechenschritt viele Transformationen gleichzeitig ausführen kann. Der Parallelrechner ist somit keine von Neumann - Maschine.

bereits in einer Patentschrift von 1941 ähnlich beschrieben, wenn auch nicht in der geschliffenen Prosa der Herren von Neumann, Burke and Goldstine, sondern in dem vorgeschriebenen Patentdeutsch. Zuses Visionen gingen aber schon Jahre zuvor über den sequentiellen Rechner hinaus - seine Patentanmeldung von 1936 [3] erwähnt bereits die Möglichkeit der Konstruktion von Feldrechnern und parallelen Rechnern. Im übrigen enthält die Patentschrift von 1936 auch die Erfindung der Gleitpunkt- Zahlendarstellung, ohne die kein numerischer Rechner auskommt.

Zuse hat sich über die Bezeichnung von *Neumann - Rechner* geärgert; er war der Meinung, daß diese Maschine eher *Babbage-Rechner* oder besser noch *Zuse-Rechner* heißen sollte [4]. In der Tat hatte Zuse und nicht Babbage als erster die Idee des adressierbaren Speichers mit binär kodierten Adressen - das Kernstück des von Neumann - Rechners (Babbage adressierte den Speicher dadurch, daß seine Lochkarten für jede Speicherzelle eine Lochposition hatten). Dafür gab es in Zuses ersten Rechnern nicht wie in Babbages Konzept *Sprungbefehle* [2] zum Verlassen des linearen Befehlsflusses. Bei Zuses ersten Rechnern Z1 bis Z3 wurde die Folge der Befehle eines Programms in einen Lochstreifen gestanzt (er verwendete dazu Kinofilm, den er als Abfall von den Babelsberger Studios erhielt), und sie konnten nur in dieser Reihenfolge gelesen werden. Die von Neumann - Maschine, die Befehle wie Daten in ihrem Speicher ablegt, kann hingegen auf jeden beliebigen Befehl wahlfrei zugreifen.

Zuse hat später betont, daß er von Anfang an die Möglichkeit der Speicherung von Daten und Befehlen gesehen habe. Da er zunächst aber die Konsequenzen von Sprungbefehlen nicht überschauen konnte, habe er lieber darauf verzichtet [5,6]. Es liegt aber auch die Vermutung nahe, daß er den kostbaren Speicher nicht zum Speichern von Befehlen verschwenden wollte, zumal die ihm vertrauten numerischen Rechnungen weitgehend ohne Programmschleifen ausgeführt werden konnten. Es bleibt festzuhalten, daß erst die Z4 wirklich in allen Zügen ein *von Neumann - Rechner* war.

Schlimmer noch als mit der ihm nur zögernd und nicht in vollem Maße gewährten Anerkennung ist es Zuse mit seinen Patentanmeldungen ergangen [5]. Zwar führte die Anmeldung von 1936 zu einem Patent auf die in der Z1 und Z2 verwandte mechanische Schaltkreis- und Speichertechnik. Eine 1937 getätigte erste Anmeldung von Programmsteuerung und Rechenwerken wurde wegen mangelnder Offenbarung zurück gezogen. Der gleichzeitigen Anmeldung in den USA wurde Babbage entgegengehalten.

1941 meldete Zuse ein Patent mit 51 Ansprüchen auf alle wichtigen Einrichtungen des Relaisrechners Z3 an [5]. Die Ansprüche bezogen sich insbesondere auf eine *programmgesteuerte Rechenmaschine* mit einem adressierbaren *Speicher*, einem *Rechenwerk* für Gleitpunkt-Operationen und einem *Programmwerk* zur Steuerung der Rechenabläufe - dies sind genau die Komponenten, die den *von Neumann - Rechner* ausmachen. Wegen der im Kriege bestehenden Geheimhaltungsvorschrift wurde die Anmeldung erst 1951 bekanntgemacht. Gegen die

[2] Sprungbefehle werden zum Beispiel benötigt, um Programmschleifen zu programmieren, d.h. das mehrmalige Ausführen einer bestimmte Befehlsfolge, wie es bei iterativen Algorithmen notwendig ist.

Patenterteilung wurde zunächst von der Firma Triumph und später auch von IBM Einspruch erhoben. Erst 1967, 26 Jahre nach der Anmeldung, entschied das Bundespatentgericht auf Ablehnung mit folgender Begründung: "Die Neuheit und Fortschrittlichkeit des mit dem Hauptantrag beanspruchten Gegenstandes sind nicht zweifelhaft. Indessen kann auf ihn mangels Erfindungshöhe kein Patent erteilt werden". Eine Rechtsbeschwerde beim Bundesgerichtshof wurde nicht zugelassen. Die Absurdität dieses Urteils stellt den erwähnten Spruch des Honorable Judge Larson, der Eckert und Mauchly ihre ENIAC-Patente aberkannte, bei weitem in den Schatten! Anders als anderen, glücklicheren Rechnerpionieren erwuchs so Zuse kein Nutzen aus seiner epochalen Erfindung.

## Der Wissenschaftler

*Verwendung von Aussagenlogik und Boolescher Algebra*

In der deutschen ffentlichkeit entstand über die Jahre das Bild vom "Computerbastler Zuse" (eine beliebte Formulierung eines deutschen Nachrichtenmagazins). Zuse selbst hat mit seiner Autobiographie [5] zu diesem Bild beigetragen, wenn er beschreibt, wie er im elterlichen Wohnzimmer in Berlin mit der Laubsäge die Blechteile für den mechanischen Rechner Z1 ausschnitt.

Inspiriert von der Dyadik des Gründers unserer Akademie Gottfried Wilhelm Leibniz (1646-1716), machte auch Zuse die Verwendung der Binärzahlen zum Ausgangspunkt seiner Rechnerentwicklungen. Zum Rechnen mit Binärzahlen benötigt man nur drei sehr einfache Grundoperationen: *Konjunktion* (logisches UND), *Disjunktion* (logisches ODER) und *Negation* (logisches NICHT). Seine erste Erfindung waren mechanische Schaltelemente zur Ausführung dieser Operationen. Abb. 2 zeigt das mechanische UND-Gatter zur Ausführung der Konjunktion.

Das UND-Gatter arbeitet mit drei Blechstreifen, die hin- und herbewegt werden können und entsprechend ihrer Endlage die Werte 0 oder 1 darstellen. Die Steuereingabe S bringt dabei den Mitnahmestift in die obere oder untere Stellung. In der unteren Stellung (S=0) kann sich die Eingabe E nach links (E=0) oder rechts (E=1) bewegen, ohne daß der Stift die Ausgabe A "mitnimmt". In der oberen Stellung (S=1) hingegen folgt die Ausgabe A der Eingabe E. Die Wahrheitstabelle dieser Operation ist somit die der Konjunktion.

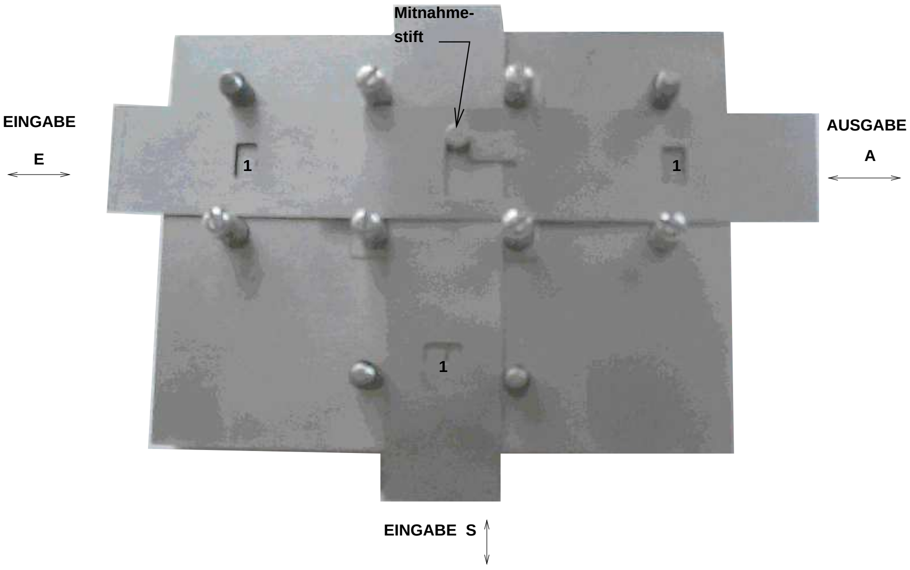


*Abb.2 Mechanisches UND-Gatter*

Zuses erster mechanischer Digitalrechner, Z1, bestand aus der Zusammenschaltung von tausenden solcher mechanischer UND-Gatter, wie in Abb. 3 zu sehen.

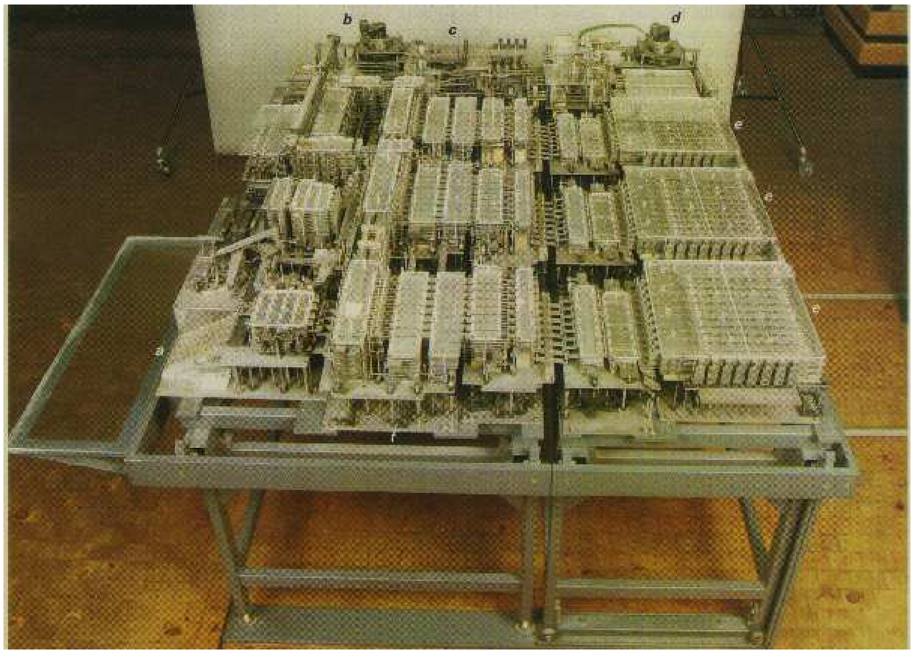

*Abb. 3 Der mechanische programmierbare Rechner Z1*

Zuse erkannte sehr bald, daß er die logischen Grundoperationen statt mit seinen mechanischen Schaltelementen auch mit Telefonrelais realisieren konnte. Fast zehn Jahre später lernte er dann von seinem Freund Helmut Schreyer, daß auch Elektronenröhren die benötigten Schaltvorgänge ausführen können [5].

Zuse war alles andere als ein Bastler. Er erkannte frühzeitig, daß sich die Funktionsweise seines mit Dualzahlen arbeitenden Rechners mathematisch mit den Mitteln der Aussagenlogik beschreiben ließ. Der Aussagekalkül bzw. die Boolesche Algebra wurde so bei ihm von Anfang an (wie auch heute noch) zur Grundlage für den logischen Entwurf seiner Rechner. Daraus erwuchs ein starkes, allgemeineres Interesse an der mathematischen Logik, was auch die Beschäftigung mit der Prädikatenlogik einschloß.

Zuse wurde oft gefragt, was ihn in den dreißiger Jahren dazu motivierte, einen programmierbaren Rechenautomaten bauen zu wollen. Seine Standard-Antwort war: Faulheit! Er pflegte dann anzufügen, daß er die mühsamen statischen Berechnungen, die er als Bauingenieur auszuführen hatte, lieber einer Maschine übertragen wollte, statt sie selbst durchführen zu müssen. Diese Antwort, die ihm immer Punkte bei Journalisten einbrachte, ist bestenfalls teilweise wahr. Daß die Hauptanwendungen seiner Rechner in numerischen Berechnungen bestehen würden, war Zuse von Anfang an klar, und dafür entwarf er auch seine ersten Maschinen. Sein besonderes Interesse galt aber von Anfang an Anwendungen, die man später dem Bereich der künstlichen Intelligenz zurechnete, wie zum Beispiel dem Schachspiel [5,6]. Die von Zuse 1945 entwickelte, *Plankalkül* genannte Programmiermethodik [6] war daher bereits ebenso für die Behandlung von numerischen wie von nicht-numerischen Anwendungen konzipiert.

### *Zuses größte wissenschaftliche Leistung: Der Plankalkül*

Der Plankalkül (PK) war die erste höhere Programmiersprache in der Geschichte der Menschheit. Der PK beweist mehr als jede andere Tätigkeit von Konrad Zuse, wie falsch das Bild vom "Computerbastler" ist. Er entstand 1945 im bayrischen Hinterstein, einem Dorf, in das Zuse zusammen mit der unvollendeten Z4 kurz vor Kriegsende aus Berlin ausgelagert worden war. Da es Zuse dort zunächst unmöglich war, weiter an der Entwicklung der Z4 zu arbeiten, wendete sich sein rastloser Geist der Frage zu, wie digitale Rechner in mathematisch-abstrakter Form programmiert werden und dabei insbesondere auch Probleme der künstlichen Intelligenz angepackt werden könnten.

Nach eigener Aussage beabsichtige Zuse mit dem PK, ein formales Schema, einen Kalkulus für die Konstruktion von *Rechenplänen* (heute sprechen wir von *Programmen*) aufzustellen, welches "eine rein formale Beschreibung für jede beliebige Rechenprozedur liefert". Zu seinen Intentionen schreibt er [6, S.25], daß der Schwerpunkt seines Interesses weniger bei den numerischen Anwendungen lag - die für ihn trivial war - als bei logischen Problemen. Der PK war daher ebenso für die Behandlung kombinatorischer Probleme wie für die Ausführung numerischer Algorithmen gedacht. Das Musterbeispiel eines kombinatorischen Problems war für Zuse das Schachspiel: In seiner Schrift über den PK [6] sind allein 44 Seiten dieser Anwendung gewidmet. Zuse hatte von Anfang an die Vision, daß seine Maschinen am Ende besser Schach spielen würden als der beste Schachmeister - und auch damit behielt er am Ende recht.

Oberflächlich gesehen nahm der PK wesentliche Züge der späteren algorithmischen Sprachen wie Algol vorweg [7]. So gibt es Entsprechungen für die gängigen Kontrollkonstrukte wie IF- und REPEAT-UNTIL-Anweisung. Und man findet auch alle in höheren Programmiersprachen übliche skalaren Datentypen wie:

- boolean (logische Werte)
- integer (ganze Zahlen)
- real (reelle Zahlen)
- complex (komplexe Zahlen).

Eine genauere Betrachtung [8] zeigt aber doch wesentliche Unterschiede zu den frühen höheren Programmiersprachen (die erst über ein Jahrzehnt später entstanden) auf, nämlich

- die Sichtbarkeit der binären Darstellung (*representation*) der genannten Datentypen
- die Existenz von Datenstrukturtypen.

Der Begriff des Datentyps und seiner Repräsentation ist zu erläutern. Mit *Datentypen* bezeichnet man die Objektarten, mit denen in den Anwendungen gerechnet wird (z.B. logische Größen, Zahlen der verschiedensten Art, abstrakte Bezeichner, usw.), einschließlich der Operationen, die auf die einzelnen Objektarten angewandt werden können. Auf Hardwareebene werden alle diese Objektarten immer durch *Bitketten*, d.h. Folgen von Nullen und Einsen, repräsentiert. Bereits bei den frühen höheren Programmiersprachen Fortran und Algol galt das Prinzip, daß der Programmierer nach Möglichkeit nur die Typen der Sprache sieht, nicht jedoch die sie repräsentierenden Bitketten (die nur der Maschine bekannt sein müssen). Das Verbergen der Datenrepräsentation gibt dem Programmierer eine abstrakte, anwendungsbezogene Sicht der Datentypen und macht so die Programmierung einfacher und fehlerfreier. Dadurch wird dem Programmierer andererseits die Möglichkeit genommen, eigene Typen definieren zu können.

Der PK kennt genau betrachtet nur *einen* elementaren Datentyp: das einzelne Bit. Alle anderen, anwendungsorientierten Datentypen werden vom Programmierer bereits als Bitmuster, d.h. als Strukturen deklariert, deren Interpretation nicht wie in den höheren Programmiersprachen a priori definiert ist, sondern erst durch eine vom Programmierer zu liefernde Typenspezifikation. Dadurch sind die den Anwendungs- Datentypen zugrundeliegenden Bitmuster dem Benutzer sichtbar, und im Vergleich zu Fortran und Algol hat hier der PK eine niedrigere Abstraktionshöhe.

Andererseits gibt es im PK Datenstrukturtypen wie den *binären Baum*, das *Array* (Feld) und die *Liste*, insbesondere die *Liste von Wertepaaren*, die die Darstellung von generalisierten Graphen [9] (d.h. beliebigen Relationen) ermöglicht. Damit können z.B. geometrische Strukturen aufgebaut werden [6]. Strukturen können im PK dynamisch sein, d.h. während der Programmausführung erzeugt werden. Listen sind immer dynamisch; sie können wachsen und schrumpfen. Die leere Variable dient als Platzhalter für dynamisch erzeugte Elemente, und es gibt Listenoperationen für

- Erzeugung von Unterlisten aus denjenigen Elementen, die ein bestimmtes Prädikat erfüllen;
- Abfrage der Anzahl der Listenelemente;
- Lesen des ersten oder letzten Elements;
- Suche nach dem kleinsten oder größten Element;
- Anfügen eines Elements am Anfang oder Ende der Liste;
- Konkatenation von zwei Listen.

Diese Züge des PK sind erst in den ein Jahrzehnt später entstehenden Sprachen *Lisp* und *APL* erneut zu finden. Lisp basiert auf der Liste als Grundtyp und wurde damit zur Sprache der *künstlichen Intelligenz*. APL (*A Programming Language*) [10] war die erste implementierte Sprache für die dynamische Erzeugung und Manipulation von komplexen Datenstrukturen. Alle diese Eigenschaften geben dem PK (wie auch Lisp und APL) einen sehr viel höheren Abstraktionsgrad, als ihn die üblichen höheren Programmiersprachen haben, die letztlich nur die Fähigkeit der "von-Neumann-Maschine" abbilden, mit jedem Rechenschritt den Inhalt eines einzelnen Speicherplatzes zu transformieren, statt eine ganze Datenstruktur.

*Beispiel: Relationale Datenspeicherung*

Die Sichtbarkeit der binären Strukturobjekte im PK gestattet es, diesen jede beliebige semantische Bedeutung zu geben. Das heißt, der Benutzer kann seine eigenen, anwendungsgerechten Datentypen definieren. Zuse nennt als Beispiele [5, p.12] die folgenden Typen:

- Personen
- Alter
- Geschlecht
- Ehestand
- andere Personaldaten
- die Felder des Schachbretts

- die Schachfiguren, einschließlich der Definition ihrer Zugmöglichkeiten
- die Kanten eines Graphen
- oder was immer sonst die Anwendung erfordert.

Alle Operationen auf benutzerdefinierten Typen sind Ausdrücke der Aussagenlogik oder der Prädikatenlogik, die auf den binären Repräsentationen ausgeführt werden. Dies erlaubt es, Untermengen aufgrund vorgegebener Eigenschaften zu bilden. Bild 4 gibt dafür ein Beispiel.

**Gegeben sei die Bitkette**

$$b_9 b_8 b_7 b_6 b_5 b_4 b_3 b_2 b_1 b_0$$

**die eine Person wie folgt beschreibt:**

DEFINITIONEN

$b_6 b_5 b_4 b_3 b_2 b_1 b_0$ = ALTER $\qquad 0 \le AGE \le 127$

$b_7$ = GESCHLECHT $\qquad b_7$ = 0: weiblich
1: männlich

$b_9 b_8$ = FAMILIENSTAND $\qquad b_9 b_8$ = 0 0: ledig
0 1: verheiratet
1 0: verwitwet
1 1: geschieden

**Das Prädikat**

$$\overline{b}_7 \wedge (b_9 \vee \overline{b}_8) \wedge \overline{b}_6 \wedge \overline{b}_5 \wedge b_4$$

**bezeichnet alle unverheirateten Frauen in der Altersgruppe von 16 bis 31 Jahren.**

*Abb. 4 Beispiel eines benutzerdefinierten Datentyps*

*Beispiel: Graphen-Analyse*

Als Demonstration für die quasi-mathematischen Programmtexte des PK zeigen wir in Bild 5 ein Programm, welches herausfindet, wieviel Kanten jeden Knoten eines unidirektionalen Graphen verlassen [8]. Der Graph wird dabei als *Liste von Paaren* dargestellt, wobei die Paare die Quell- und Zielknoten der Kanten angeben. Die Funktion ist von der Art: $(V_0, V_{10}) \Rightarrow R$, wobei $V_0$ die Liste von Paaren, $V_1$ ein Paar in der Liste und $V_{10}$ das erste Element dieses Paares bezeichnen. R ist die Zahl der Paare in $V_0$ in denen $V_{10}$ als erstes Element auftritt und damit die Anzahl der Kanten, die $V_{10}$ verlassen. Das Programm benutzt zwei spezielle Funktionen des PK, den Operator L, der die Zahl der Listenelemente liefert [6, p.46] und den THOSE WHICH - Operator, $\hat{x}$ der eine Unterliste aller Elemente einer Liste erzeugt, für die ein gegebenes Prädikat (eine Aussage) wahr ist [6, p.70]. Damit liest sich das Programm fast wie eine mathematische Formel.

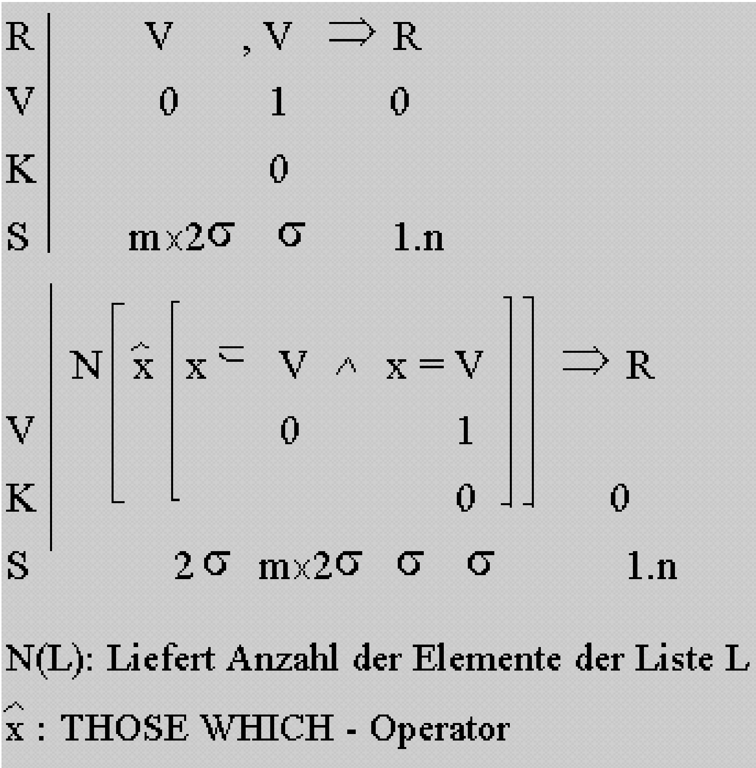


*Abb. 5 Programm zur Ermittlung der Zahl der Kanten in den Knoten eines Graphen*

Datenstrukturen wie *Liste von Paaren* und Operatoren wie N(L) und THOSE WHICH sind in den Ende der fünfziger Jahre entstehenden höheren Programmiersprachen nicht zu finden. Etwas ähnliches findet man erst in der "KI-Sprache" *Lisp* oder der quasi-algebraischen Operatorensprache *APL*.

Konrad Zuse hatte bereits 1945 bei der Entwicklung des Plankalküls eine klare Vorstellung eines abstrakten Programmiermodells mit abstrakten Datentypen und Operationen und einer für numerische wie nicht-numerische Anwendungen gleichermaßen geeigneten Datenrepräsentation, Konzepte, die eigentlich erst in den siebziger Jahren ihren Einzug in die Informatik hielten. Dies macht ihn zum ersten Informatiker. In seiner Schrift über den PK führt er bis in das letzte Detail aus, wie die Datenstrukturtypen des PK mit ihren mächtigen Operationen dazu benutzt werden können, das Schachbrett mit jeder beliebigen Konstellation der Figuren sowie die Zugmöglich keiten der einzelnen Figuren darzustellen und damit Schachprogramme zu realisieren. Dies macht ihn zum Erfinder des Konzepts der künstlicher Intelligenz.

## Die Ungnade der frühen Geburt

Weder als der erste Informatiker noch als Erfinder der künstlichen Intelligenz ist Konrad Zuse bis heute anerkannt, ebensowenig, wie ihm seine Erfindungen auf dem Gebiet der Hardwarearchitektur Nutzen gebracht haben. Von ihm selbst stammt der Ausspruch: "Die besten Erfindungen taugen nichts, wenn sie zu früh kommen".

Seine Erfindungen entstanden im NS-Staat, der sich selbst aus der freien Völkergemeinschaft mit freiem Austausch von Ideen ausgeschlossen hatte. Durch die Vertreibung oder Vernichtung eines wesentlichen Teils der geistigen Elite Deutschlands war die traditionelle Rolle der deutschen Sprache als einer wichtigen Wissenschaftssprache bereits vor dem Krieg geschwunden. Die Kriegs- und Nachkriegsverhältnisse schlossen erst recht einen wissenschaftlichen Austausch aus. Als die Bundesrepublik Deutschland graduell wieder in die Völkergemeinschaft aufgenommen wurde, war die Entwicklung inzwischen in den USA soweit vorangeschritten, daß dort mit einer gewissen Berechtigung die Meinung vorherrschte (und bis heute vorherrscht), daß alle wichtigen Erfindungen und Konzepte auf dem Gebiet der Rechnertechnologie und der Informatik in den USA entstanden sind. Wir wagen zu sagen, daß manche Entwicklungen auf dem Gebiet der höheren Programmiersprachen und der künstlichen Intelligenz Jahre früher begonnen hätten, wären Zuses Ideen und Vorarbeiten bekannt gewesen.

Erst Brian Randells so verdienstvolles Buch "The Origins of Digital Computers" [3] machten die interessierte Fachwelt mit Zuses Erfindungen bekannt. Die Monographie über den Plankalkül wurde erst 1976 als GMD-Bericht veröffentlicht, und zwar in englischer bersetzung. Diese Schrift wurde aber international kaum zur Kenntnis genommen (was sicher auch die Tatsache widerspiegelt, daß GMD-Berichte wenig gelesen werden).

Wäre Konrad Zuse in den fünfziger Jahren ein jüngerer Mann mit akademischen Interessen gewesen, der seine Arbeiten in amerikanischen Journalen hätte publizieren können, seine Rolle als Pionier neuartiger Konzepte der Informatik und künstlichen Intelligenz wäre eher anerkannt worden. Zuse hatte in den fünfziger und sechziger Jahren aber einen ganz anderen Ehrgeiz, nämlich den, leistungsfähige und wirtschaftliche Rechner zu entwickeln, herzustellen und auf den Markt zu bringen. Zuse selbst hat öfters in der ihm eigenen Selbstironie auf die Frage, als was er sich letztlich sieht, die Antwort gegeben: "Als gescheiterter Unternehmer". Auch mit diesem Ausspruch verkauft er sich unter Wert.

Seine Firma hatte den Verdienst, es der Deutschen Forschungsgemeinschaft zu ermöglichen, in den späten fünfziger und sechziger Jahren die deutschen Universitäten mit dem preiswerten wissenschaftlichen Rechner Z22 auszustatten. Die erste Generation von Informatikern in der Bundesrepublik (zu der auch der Verfasser gehört) verdankt der Z22 ihre ersten Erfahrungen mit digitalem Rechnen und digitaler Programmierung.

Wie ist der Mensch Konrad Zuse damit fertig geworden, daß seinen Pionierleistungen außerhalb Deutschlands nicht in gebührendem Maße anerkannt worden sind? Wir glauben, daß sich diese Frage für jeden beantwortet, der den Maler Konrad Zuse kennt. Wer das Privileg hatte, ihn in seinem Atelier zu erleben und die Originalität und Fülle seines künstlerischen Schaffens zu erleben, glaubt zu erahnen, worin Konrad Zuse Trost über alle Enttäuschungen fand. Für uns bleibt aber die Verpflichtung, auch nach seinem Tode das unsere zu tun, die wahre Bedeutung des großen Computerpioniers und Informatikers Konrad Zuse der Nachwelt deutlich zu machen.

# Literatur

[1 ] Lee, J.A.N.: looking.back. In: Computer Magazine 2/1996, (looking back: monthly column in the 1996 issues of Computer to mark the Computer Societys 50th anniversary)

[2 ] Burks, A.W. et al.(1963): Preliminary Discussion of the Logical Design of an Electronic Computing Instrument, in: Taub A.H.(Hg.): Collected Works of John von Neumann, vol.5, New York: MacMillan, S. 34-79

[3 ] Randell B.(Hg..) (1973): The Origins of Digital Computers, Berlin-Heidelberg-New York: Springer-Verlag

[4 ] Zuse K. (1993): Brief an W.K. Giloi vom 27.12.1993

[5 ] Zuse K.(1970): Der Computer Mein Lebenswerk, München, Verlag Moderne Industrie

[6 ] Zuse K.(1989): The Plankalkül, GMD Report no. 175, 2nd edition, München-Wien: R Oldenbourg-Verlag

[7 ] Bauer F.L., Wössner H. (1972): The Plankalkül of Konrad Zuse, a Forerunner of Todays Programming Languages, In: Elektronische Rechenanlagen 2/1972

[8 ] Giloi W.K.(1997): Konrad Zuses Plankalkül: The First High-Level, nnon von Neumann" Programming Language, In: Annals of the History of Computing 4/1997

[9 ] Knuth D.E.(1975): The Art of Computer Programming, Vol.1, (2nd edition) Reading, Mass.: Addison-Wesley

[10 ] Iverson K.E. (1962): A Programming Language, New York: J.Wiley & Sons