Konrad Zuse built the machine Z3 from 1939 to 1941 in the Methfesselstraße 7 in Berlin-Kreuzberg with some friends and a small support by the government. With the Z3 Konrad Zuse wanted to show, that it is possible to build a reliable working machine for very complicated arithmetic calculations, which is freely programmable and is based on a binary floating point number and switching system. For reliability reasons he used relays for the entire machine.

Picture 1: The design of the Z3 by Konrad Zuse in 1939. There does not exist a photo of the Z3 because all documents and photos were destroyed by allied air raids during the war.

Konrad Zuse rebuilt the Z3 in 1960/61 in his Zuse KG in order to show the patent justice and the world the performance of this machine and to demonstrate this machine for advertising purposes.

Picture 2: Left: The rebuilt computer Z3 in 1964 with Konrad Zuse. The memory is on the left side (The logo Z of the Zuse KG can be seen) and the arithmetic unit with the stepwise relays are on the right side. Left on the front the console with the punch tape reader can be seen.
Right: The diary of Konrad Zuse at May 12, 1941: Konrad Zuse presented the working Z3 to scientists in Berlin.

Today, it is undisputed, that the Z3 was the first freely programmable, based on a binary floating number and switching system, really working computer of the world. The Z3 did not store the program in the memory, but it contained in 1941 almost all the components of a modern computer as required by John von Neumann et al. in 1946 /BURK46/. One exception is the ability to store the program in the memory together with the data. Konrad Zuse did not implement this feature in the Z3 because the memory of 64 words was too small for this. He wanted to calculate thousands of instructions in a meaningful order. The machine he constructed should be freely programmable. For this reason, he only used the memory for values or numbers.

Picture 3: The block structure of the Z3 and the rebuilt machine Z3.

The block structure of the Z3 is very similar to a modern computer. The Z3 disposed of separate units, like punch tape reader, control unit, floating point arithmetic unit and input-/ output devices. With respect to this theme, Burks et al. /BURK46/ wrote as late as in 1946: *In as much as the completed device will be a general-purpose computing machine it should contain main organs relating to arithmetic, memory-storage, control and connection with the human operator. It is intended that the machine be fully automatic in character, i.e. independent of the human operator after the computation starts.* It is important to notice, that the Z3 contained organs like an arithmetic unit, memory-storage, control and connection with the human operator. The machine Z3 contained a special operating modus. With the instruction *Lu* the input device was activated and the program was stopped. The human operator could check, among others, the Registers R1 and R2 of the arithmetic unit and he could make intermediate calculations using the Registers R1 and R2. Then he could proceed the program. This is a very early interactive communication with the machine.

Let us consider the discussion of floating point numbers in the paper of Burks et al. /BURK46/: *It would therefore seem to us not at all clear whether the modest advantages of a floating binary point offset the loss of memory capacity and the increased complexity of the arithmetic and control circuits.* In /NEUM45/ the use of the binary system for computers is proposed. It is very important to notice, that floating point numbers were discussed as late as in 1946 by Burks et al. and they were not really convinced that this is a good concept. Konrad Zuse described in 1934 the concept of a floating processor and implemented in 1936 such a
floating point processor. This is ten years before Burks et al. /BURK46/ postulated it as a concept for modern computers (not a built machine). As shown above and will be shown below, Konrad Zuse also implemented a powerful arithmetic exception unit. Let us consider the logical concepts of the Z3 in detail.

Parallel Machine: The Z3 was a parallel working machine. The 22 bits from the memory to the Register R1 and vice versa were moved in one step (cycle). The same holds for the binary arithmetic unit, where, among others, two parallel adders (exponent, mantissa) were used.

Memory: The memory of the Z3 consisted of 64 words à 22 bits. Each word was directly addressable by the instructions Pr z or Ps z, where z is the address in the range: 64 £ z £ 1. For each bit a relay was needed.

Floating Point Numbers: Konrad Zuse used a semi-logarithm representation of numbers. Let us take the number 100. In the binary system this number can be written as:

$$100 = 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 = 64 + 32 + 4.$$ 

The number 100 can be created by the power of 2 related to the numbers: 2^6, 2^5 and 2^2 = 64 + 32 + 4. In the binary representation we use the 1 or 0 in order to represent the number. (These are exactly the factors 1 and 0 before the multiplication sign. In the binary digit system the number 100 can be written as: 1100100. This is a binary number without a comma. The number 100,5 can be represented as:

$$100,5 = 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} = 64 + 32 + 4 + 0,5.$$ 

In the binary system we write this as: 1100100,1. The most right 1 of the comma is 2^{-1} = 0,5. These are binary numbers with a comma, we also say fixed point numbers. To represent numbers in the 0 or 1 notation has big advantages for computers. This has not changed till today. Since only two states or two numbers are considered, we can use pair circuits to store numbers and to operate with them. In 1939 relays were such building blocks. However, the above representation of numbers has a disadvantage. Konrad Zuse used 22 bits for his number, but with 22 bits we can only represent small numbers in a range £ 2^{22} - 1. For this reason Konrad Zuse used the binary floating point representation (semi-logarithmic representation) for the numbers. Let us consider the number 100 again.

$$100 = 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 = 64 + 32 + 4.$$ 

Konrad Zuse divided the number in an exponent and a mantissa. As the exponent he used the power of two of the highest exponent, in our case 6. This number was converted into the binary system and we got 110. As the mantissa we used the binary number 1100100. The new number – a floating point number – is now:

$$00000110 1100100000000000.$$ 

The first digit of the mantissa always is 1. For this reason the first 1 is left out and the mantissa is 100100 instead of 1100100. Doing this, Konrad Zuse did win one digit for a higher accuracy of his numbers. Instead of 14 bits he had 15 bits for the mantissa. The number 100 is now:

$$00000110 1001000000000000.$$
The number 100.5 in the binary system is: 1100100,1. The floating point number is:

00000110 110010010000000.

In the Z3 the number is:

00000110 10010010000000.

Using this concept, Konrad Zuse was able to handle very big and very small numbers in a range of approximately ±2⁻⁶³ to ±2⁶³.

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Function</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pr z</td>
<td>To read the contents of the memory cell z into the Registers R1 or R2.</td>
<td>1</td>
</tr>
<tr>
<td>Ps z</td>
<td>To write the contents of Register R1 in the memory cell z.</td>
<td>0-1</td>
</tr>
<tr>
<td>Ls1</td>
<td>Addition: R1 := R1 + R2</td>
<td>3</td>
</tr>
<tr>
<td>Ls2</td>
<td>Subtraction: R1 := R1 + R2</td>
<td>4-5</td>
</tr>
<tr>
<td>Lm</td>
<td>Multiplication: R1 := R1 + R2</td>
<td>16</td>
</tr>
<tr>
<td>Li</td>
<td>Division: R1 := R1 + R2</td>
<td>18</td>
</tr>
<tr>
<td>LW</td>
<td>Square root: R1 := SQR(R1)</td>
<td>20</td>
</tr>
<tr>
<td>Lu</td>
<td>To call the input device for decimal numbers.</td>
<td>9-41</td>
</tr>
</tbody>
</table>

Picture 4: Floating point numbers of the Z3. VZ means the sign for the mantissa, E7 to E1 are the bits of the exponent and the numbers 14 to 1 describe the 14 bits of the mantissa. The picture is from the Konrad Zuse Multimedia Show /ZUSE98/.

Instructions: The Z3 disposed of the following instruction set and cycles. We introduce the following notation: R1 and R2 are Registers à 22 bits. The notation: R1 := R1 + R2 means: The contents of Register R1 is added to the contents of Register R2 and the result is stored in Register R1. Register R2 is set to empty after arithmetic the arithmetic operation.

**Instruction Function Cycles**
Ld To call the output device for decimal numbers. 9-41

The two basic arithmetic operations of the Z3 are the addition and substraction of the exponent and signs. Addition and subtraction require more than one cycle because, in the case of floating point numbers, care has to be taken to set the size of the exponent of both arguments to the same value. This requires some extra comparisons and shifting. The number of cycles needed for the Lu and Ld instructions is variable, because it depends on the exponent of the argument. A number can be stored in memory (Ps) in zero cycles when the result of the last arithmetic operation can be redirected to the desired memory address. In this case, the cycle needed for the store instruction overlaps the last cycle of the arithmetical operations.

It is a little bit surprising, that the conditional branch is missing. Konrad Zuse described in his papers from 1936 till 1945 a lot of scientific and numerical problems, which he wanted to solve with his machines. For this calculations he did not need the conditional branch. At least in 1944, he mentioned the conditional branch indirectly in the context of free programs /ZUSE45/, p16. Konrad Zuse introduced Free Programs (Freie Rechenpläne): The definition is: Bei den freien Rechenplänen beeinflussen die eigentlichen Variablen den Ablauf der Rechnung. Zunächst können die bei den quasistarren Rechenplänen besprochenen Planvariablen wie variable Operationszeichen, Strukturzeichen usw. Funktionen der eigentlichen Variablen sein. Es kann z.B. die Art der Operation in einer Rechengeleichung erst errechnet werden.

Translation: With the free programs the actual variables influence the execution of the calculation. First of all the variables, like operation signs, structure signs, as defined with the quasi rigid (quasi starr) programs, etc., can be functions of the actual variable. It is, for example, possible to calculate the type of operation in a calculation equation. However, all the other machines at this time did not have the conditional branch.

Arithmetic Unit and Carry Ahead: The arithmetic unit of the Z3 is Konrad Zuses masterstroke. He reduced all the arithmetic operations to addition or substraction. For the realization of the addition (subtraction is an addition of the complement of one number and the number) Konrad Zuse implemented a special switch because he wanted to avoid too many cycles for the addition of two binary floating point numbers. Using the special switch, he could reduce the addition from at least 14 cycles with a serial addition down to three cycles with a parallel addition. Although there were only five instructions (Ls1, Ls2, Lm,Li, and Lw) for arithmetic operations, some more operations were implemented which could be called from the input device. He also simplified the execution of the arithmetic operations with micro-sequences controlled by stepwise relays.

<table>
<thead>
<tr>
<th>Arithmetic Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic Operations:</td>
</tr>
<tr>
<td>+, -, *, /, SQR, *0.1, *10, /0.1, /10, *2, *1/2, ^-1</td>
</tr>
<tr>
<td>Exponent</td>
</tr>
<tr>
<td>Register R1</td>
</tr>
</tbody>
</table>

Picture 5: Left: The arithmetic unit of the Z3. It could be used from the program but also for intermediate operations from the input device. Right: A relay of the Z3.
Konrad Zuse used a self developed carry look-ahead circuit of relays for the addition of floating point numbers. With this concept he could add two floating point numbers in three cycles. Assuming the numbers to be added were still in the memory, then he needed five cycles, two for the instructions Pr and three for the instruction Ls1. With the first cycle, the first operand was transferred to Register R1 and with the second cycle, the second operand was transferred to Register R2. Then, the exponent and mantissa were added and the result was stored in Register R1. After this Register R2 is set empty. We write this as: Ls1: R1 := R1 + R2. Konrad Zuse used additional internal registers which are not shown on the picture above. These registers were two bits longer than the mantissa in order to get a higher accuracy of his results.

The multiplication algorithm of the Z3 is like the one used for decimal multiplication by hand, that is, it is based on repeated additions of the multiplicator according to the individual digits of the multiplicand.

The division algorithm is similar to the multiplication one, but the repeated substruction is used.

The instruction algorithm behind the instruction Lw is a jewel of the Z3. The main idea is to reduce the square root operation to a division (See also the Machine Z4).

Arithmetic Exception Handling: The Z3 disposed of an arithmetic exception handling. Konrad Zuse implemented the exception handling because he wanted to be sure that the Z3 calculates the numbers correctly even when the Z3 is working without a supervisor. The Z3 recognized the following operations:

- Underflow of the range of numbers is: 0.
- Overflow of the range of numbers is: \( \infty \).
- For the calculation with 0 holds: \( 0 + x = x, 0 \times x = 0, x / 0 = \infty, 0 / 0 = ? \)
- For the calculation with \( \infty \) holds: \( \infty + x = \infty, \infty \times x = \infty, 1 / \infty = 0 \).
- All operations with ? have the result ?.
- \( 0/0 = \text{undefined} \)
- \( \infty - \infty = \text{undefined} \)
- \( \infty / \infty = \text{undefined} \)
- \( 0 \times \infty = \text{undefined} \)

The undefined state was shown on the output device on the left side with small lights. For the numbers 0 and \( \infty \) Konrad Zuse used special bit codes in the exponent. An exponent of \(-64\) is the decimal 0. An exponent of \(-63\) or \(+63\) represents \( \pm \infty \). The Z3 calculates always correctly, if an argument is 0 or \( \infty \) and the other argument is in the allowed range.
The input- and output devices of the Z3. At the front the numbers could be put in by buttons. There were four buttons for the mantissa and 17 buttons for the exponent (from -8 to +8). The results were shown by lamps. Right: The output device of the Z3 with the lamps for the decimal numbers (right) and the arithmetic exception handling on the left side.

The binary floating point numbers were converted to decimal floating numbers. For this conversions he needed between 9 and 41 cycles depending on the exponent. The mantissa consisted of four decimal digits (five digits for the 1) and the exponent was between –8 and +8. The biggest decimal number which could be shown was 19999E10^8.

Clocked Maschine / Clock Frequency: The Z3 is a clocked machine. Konrad Zuse used this principle to synchronize the different components of the machine. In order to do this he implemented a special impuls generator with a drum.

The impuls generator stepwise provides the different relay groups with electrical power. The clock frequency was around 5.3 Hertz depending on the quality of the telephone relays, so the faster the clock then the faster the machine. In talks with Rojas /ROJA98a/ and me Konrad Zuse mentioned that a multiplication of two floating point numbers took about three seconds. The Z3 needed 16 cycles for a multiplication. For this reason the clock frequency is 16 / 3 ≈ 5.3 Hertz. In order to avoid sparking of the relay contacts when switching them, he used the drum to drop off the electricity at the switching time. This principle avoided material rust and guaranteed a long functionality of the relay contacts.

Micro Sequencer and Pipelining: The heart of the control unit of the Z3 are the micro sequencers. Every cycle in the Z3 was divided in five steps I to V.
The steps IV and V are used to transfer information from one part of the Z3 to another one, for example the contents of a register to a memory cell. During the steps I II and III an addition or subtraction in both parts of the arithmetic unit is executed. This is the execution phase of an instruction. A typical instruction takes the arguments of the operation, executes an operation and writes the result back. Konrad Zuse considered it as very important to save execution time, that he executed the instruction in an overlapping way (See the picture). It is kind of a pipelining principle.

It is not well known, that Konrad Zuse in 1949 applied for a patent for a pipelining computer in 1949.

At June 30, 1949 Konrad Zuse applied for a patent of the first pipeline computer of the world and got it as late as on December 6, 1962. The title of the patent is: Rechenmaschine zur Durchführung arithmetischer Rechenoperationen (z.B. Multiplikationen. Translation: Calculator for the execution of arithmetic operations, for example multiplication). It was a machine consisting of eight independent arithmetic units which could execute partial arithmetic operations on certain data. With this machine it was possible to execute arithmetic operations for a large set of data.

We now are coming back to the micro-sequencers of the Z3.
With stepwise relays Konrad Zuse implemented so-called micro sequencers. The sequencers were part of the control unit and were used to enable a sequence of repeated micro-instructions. For example, the instruction Lm is realized by a sequence of additions. In the Z3 we can find micro-sequences for the instructions Lw, Lm, Li, Lu and Ld. It is not micro-programming, but it is a hardware wired micro-sequencing.

An Example Program (Polynomial): We now present an example program of the Z3. We want to calculate the polynomial: \((a_4 x + a_3)x + a_2)x + a_1\). The numbers for \(a_4,a_3,a_2\) and \(a_1\) are already stored in the memory cells 4, 3, 2, 1, which had to be done by the input device. The number for \(x\) is required by the input device with the instruction Lu.

```
Lu To call the input device for the variable x.
P 5 To store variable x in memory cell 5.
Pr 4 Load a1 in Register R1.
Pr 5 Load x in Register R2.
Lm Multiply: R1 := R1 x R2.
Pr 3 Load a3 in Register R2
Ls1 Add: R1 := R1 + R2.
Pr 5 Load x in R2.
Lm Multiply: R1 := R1 x R2.
Pr 2 Load a2 in Register R2.
Ls1 Add: R1 := R1 + R2.
Pr 5 Load x in Register R2.
Lm Multiply: R1 := R1 x R2.
Ppr 1 Load a1 in Register R2.
Ls1 Add: R1 := R1 + R2.
Ld Shows the result as a decimal number.
```

After the execution of the program the Z3 stopped and showed the result via the instruction Ld.

**Minimal Design Principle and Universal Computer (Turing)**

Konrad Zuse followed a minimal design principle with the construction of his computers. He tried to build a powerful computing machine with minimal effort and costs. He had no other way because his parents were not rich and he had not much money. For Konrad Zuse, the Z3 was the last machine in a series of trial machines (Z1-Z3) in order to build a machine which was able to solve the mathematical problems of engineers and scientists. Although he was convinced, that his machine could calculate all mathematical problems, he could not prove it. He tried to show this with many examples and chess. In 1941 he said to friends, that his machine can play chess. However, he had the right feeling, that his machine could calculate
everything. From 1941 till his death in 1995 he did not know, that his computer Z3 was an universal computer in the sense of Turing. In 1998 Raul Rojas /ROJA98a/ formulated the proof, that the Z3 is an universal computer.

Literatur

Some of the referenced literature of Konrad Zuse were not published in journals. The reason was the political situation in Germany from 1933 and the post war area.

This is an article written in the German language where F.L. Bauer discusses the dispute of the invention of the stored program computer.

This is the article from 1946 where the basic principles of a computer are explained. Here, the separation of the control unit, arithmetic unit, memory and input- and output devices are required. Burks et al. also propose to store the program together with the data in the memory. However, Konrad Zuse implemented the separation of the different units of a computer already in 1938 with the Z1 and in 1941 with the Z3. It is correct, that Konrad Zuses machines did not store the program in the memory of the machine. However, Konrad Zuse mentioned this possibility already in 1936/37 /ZUSE36/, /ZUSE37a/.

In this article Giloi describes the fundamental ideas of Konrad Zuses Plankalkül and points out that this proposal is much closer to Prolog than to the imperative languages.

This is the IEEE standard for floating point numbers from 1985. The standard also uses a semi-logarithm representation of floating point numbers. The binary floating point numbers are divided in a mantissa and an exponent. The representation of the numbers 0 and ±¥ are identical to Konrad Zuses implementations in the Z1 (1938) and the following Z-machines.

In this paper John von Neumann describes the architecture of computing machines. Among others, John von Neumann proposes the principle of the separation of the units of a computer (control unit, memory, arithmetic unit, input-/output devices). He also proposes the principle of the stored program computer. However, it is important to mention here, that Konrad Zuse implemented the separation of the units of a computer already in 1938 with the Z1. In /ZUSE36/ and /ZUSE37/ Konrad Zuse mentions the possibility to store various things in a memory, among others, instructions.

Informationsverarbeitung in ausgewählten Beiträgen, Braunschweig, 1962, pp. 269-2311.

This is a very good book about the history of computing.

This is a very good article of the architecture of the machines Z1 and Z3 of Konad Zuse.

Raul Rojas proves, that the Z3 was an universal computer in the sense of Turing. He also shows, that the conditional branch is not a necessary instruction for a universal computer.

This is a detailed analysis of a Konrad Zuses machines Z1 and Z3 with many new details. It also contains the report of the fight of Konrad Zuse for his patents from 1938 till 1967. Konrad Zuse lost this fight in 1967.


In this book, among others, many details related to the Plankalkül can be found.


In this paper Konrad Zuse applies a patent, which he did not get. At this time he were in a disunion. On the one side the fight for the patent would be very expensive, on the other side he wanted to build a machine. He decided to build the machines. He also made the first time statements that a memory can store many different things, like statements, names, reference numbers, ranks, data, instructions, messages, conclusions, etc.

/ZUSE37a/ Zuse, Konrad: Einführung in die allgemeine Dyadik., 1937.
In this paper Konrad Zuse mentioned very clear, that a program can also be stored in the memory.

In this paper Konrad Zuse describes very clearly the architectures of his machines.

This is a very complete work about the programming of computers with the Plankalkül. There Konrad Zuse demonstrates the powerfulness of the Plankalkül with arithmetic problems to be solved in engineer bureaus, and he demonstrates chess programs on more than 60 pages.
Konrad Zuse describes his ideas of the computer development in the future. His idea is to use logistical machines, which can be used to solve the mathematical problems of engineers and scientists in the opposite to the algebraic machines, like the Z4 or Mark I of Aiken. The logistical machines should be programmed with the Plankalkül.