The enigma behind the Enigma machine

Dekkers, I.

Inge M. Dekkers

The enigma behind the Enigma machine

Master thesis

date: 03.08.2023

Thesis supervisor: Prof. Dr. F.M. Spieksma

Leiden University
Mathematical Institute

Contents

1 Introduction

This master thesis is meant to be a collection of many aspects revolving around Enigma, a brand which includes various encipherment machines. The concepts for the machine were all invented and used before and during World War II. Although there is much information on the subject, the information seems to be very fragmented. Many bits and pieces of information seem to be spread over many articles, publications, books, and websites. Therefore, this thesis aims to collect the main concepts, information and stories about Enigma supported by worked-out examples and extensive explanations.

Starting in 1917, the main concept of the Enigma machine, which is the rotor construction, was invented by four different men from four different countries within three years. However, the only one who was financially successful, was the German Arthur Scherbius. Together with E. Richard Ritter, he founded the company Scherbius and Ritter. After their concept was denied by the Imperial German Navy and the Foreign Office, they transferred their patent rights for the rotor concept to the Gewerkschaft Securitas. They made their first machine in 1923. Other than the Enigma machines used during World War II, this Enigma machine printed out the outcome resulting in a heavy and big machine. The machine also had some other flaws, which made it less dependable. From there on, the Enigma machine was innovated multiple times. Over the years, this resulted in a lot of different versions of the machine. We will discuss this in Section 4. The initial versions were not made for the purpose of war; they were made for commercial purposes.

However, the collection of information described in this thesis is not near to completion. As already mentioned, and as will be discussed in Section 4, there are many different types and versions of Enigma. Therefore, for the main concepts, this thesis focusses on Enigma I, which is the machine used by the German Army and German Air Force during World War II.

For this version of Enigma, Sections 2 and 3 will elaborately explain the mechanism inside the machine, supported by examples. To fully understand all these processes, we also have programmed this version of the machine in the programming language Python, which can be found in Appendix B.

A strength of the concept of Enigma, is the security of the mechanism. There are many variables influencing this. This is reflected by the number of settings a machine can start in. Section 5 contains a formula for finding the number of settings based on the variables that can be chosen. In addition, a program in Python is provided that gives the amount of starting settings based on the entered variables.

Lastly, this thesis will not only focus on the complexity of the concept of the Enigma, but will also focus on several events and techniques used to break the code. These will be discussed in Section 6 and the corresponding appendices.

When it comes to sources, many were used. Especially, where it came to getting an understanding of the mechanism of the Enigma I machine. To prevent the text from continuous interruptions, the text will not contain direct citations or references to sources. Therefore, all sources are listed at the end of every chapter.

Sources: [9], [10], [18], [22], [17].

2 The machine

When one looks at an Enigma machine, one can distinguish four different components. In Figure 1, they are marked in red, green, orange and blue:

Figure 1: Enigma machine

Source: https://drenigma.org/2018/03/26/how-many-enigma-machines-are-there-left/

Despite of the fact that it looks like a typewriter, it is nothing like that. An Enigma machine can encipher and decipher messages. In this chapter, we are going to track and explain the path a certain letter will travel until it becomes enciphered. The **keyboard** is marked in orange and works just as a normal keyboard, containing all 26 letters of the alphabet. Before a letter is enciphered, it needs to pass through a couple of stations, like the **plugboard** (marked in blue) and the **rotors** (marked in red). The enciphered letter will eventually light up on the **lampboard** (marked in green). A letter will pass the stations in the following order:

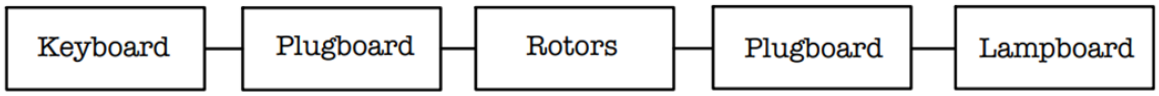


Figure 2: Stations of the Enigma machine

What happens at these stations, will be explained in the following paragraphs.

2.1 Plugboard

A station that the letter passes twice, is the plugboard. The plugboard looks as follows:

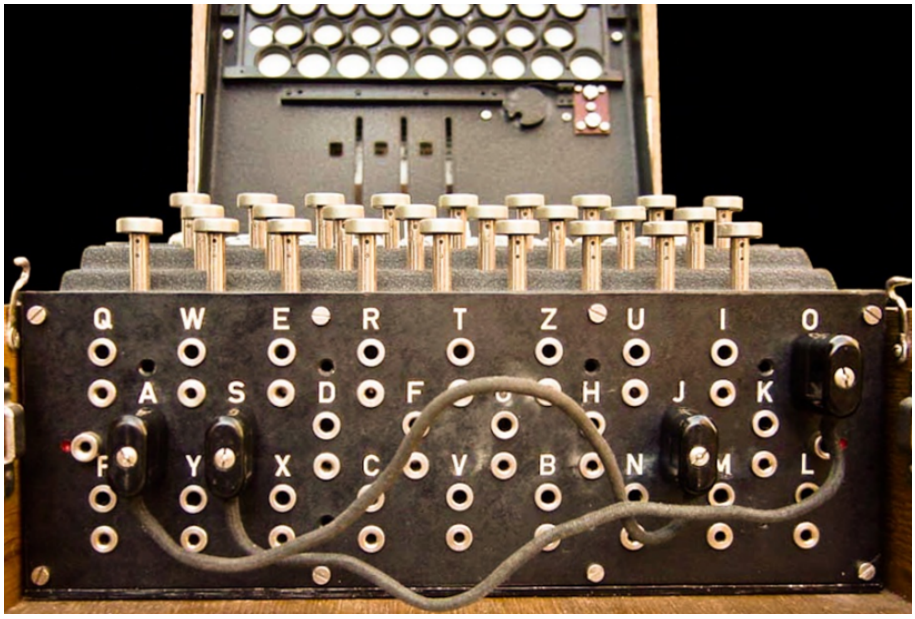


Figure 3: Enigma plugboard
Source: https://nl.m.wikipedia.org/wiki/Bestand:Enigma-plugboard.jpg

The plugboard has a plug for every letter of the alphabet, which can be connected to each other by a cable. In this way one may create pairs of letters, as each letter can only be connected to exactly one other letter. If a letter is connected to another letter on the plugboard, the incoming letter gets switched to the letter that it is connected to. Say, for example, we connect A and B with a cable, then if an A enters the plugboard, it comes out as a B. Following the same argument, if a B enters the plugboard, it becomes an A. Another possibility is that a letter is not connected on the plugboard, which means that the letter will remain itself after passing through the plugboard.

2.2 Rotors

After passing through the plugboard, we arrive at the rotors. If we open the machine, we will see the following:

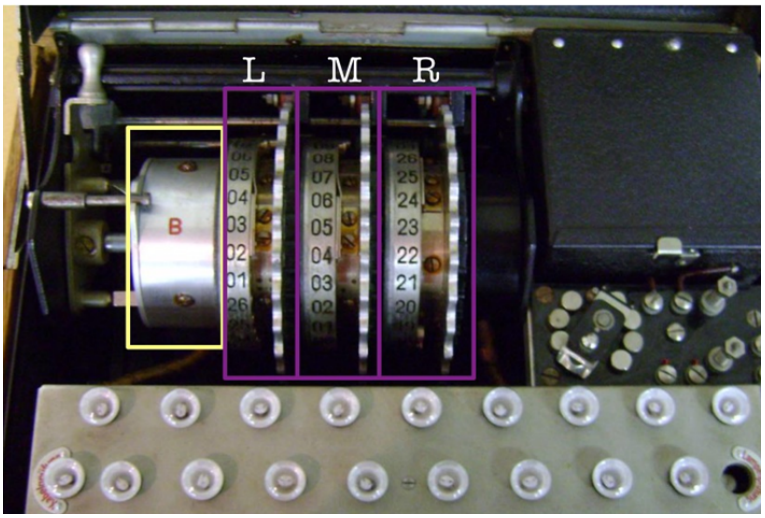


Figure 4: Enigma rotors
Source: https://www.ciphermachinesandcryptology.com/nl/enigmatech.htm

On the left, marked in yellow, we see what is called the **reflector**. On the right we see three rotors next to each other, which are marked in purple. In this machine, there is only place for three rotors. For simplicity, we refer to the left space where a rotor can be placed as L, to the middle space as M, and to the right space as R.

2.2.1 Rotor

When taking a rotor out of the machine, it will look as follows:

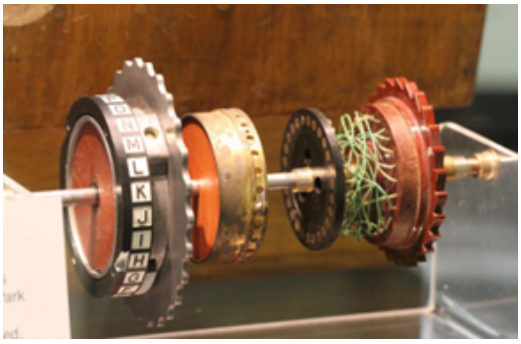


Figure 5: Inside of a rotor 1
Source:
https://www.flickr.com/photos/duanewessels/7063367271

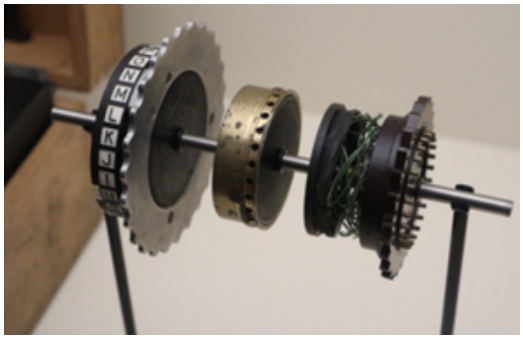


Figure 6: Inside of a rotor 2
Source: https://hackaday.com/2019/08/06/espionage-on-display-as-gchq-hosts-a-temporary-exhibit/topsecret-enigma-rotor/

Each rotor has 26 pin contacts and 26 plate contacts. When two rotors are next to each other, every pin contact of the left rotor touches a plate contact of the right rotor. The plate contacts of the most left rotor will be connected to the pin contacts of the reflector. The pin contacts of the most right rotor will be connected to the plate contacts of what is called the **entry disc**. The entry disc only is the entry towards the rotor construction. It is fixed in the machine and has no mixed up wiring. The reflector and entry disc will respectively look as follows:

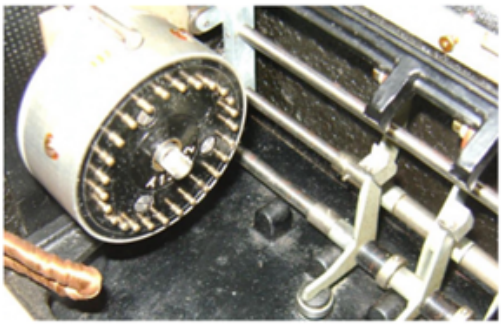


Figure 7: Reflector

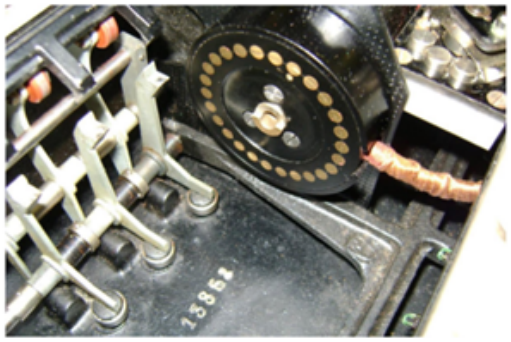


Figure 8: Entry disc

Source: https://www.ciphermachinesandcryptology.com/en/enigmatech.htm

How a letter is passed on to the next rotor is as follows. Within the rotor there are 26 cables that randomly connect a pin contact to a plate contact. This can equivalently be interpreted as one letter being randomly connected to another letter, which in fact can also be itself. This internal wiring, which is fixed, is different for every rotor. Lastly, all rotors have what is called an alphabet ring. As the rotors can be rotated by the user, the alphabet ring is meant for showing the user in which configuration the rotors are as one letter can be seen through the rotor window (marked in red in Figure 1). The alphabet ring does also influence the stepping mechanism described in Section 2.3. One can rotate the alphabet ring separately from the internal wiring, so the alphabet ring can be set as well. This will be more elaborately discussed in Section 2.3 and 2.6.

2.2.2 Reflector

Within the reflector, which is fixed inside of the machine, all 26 pin contacts are connected in pairs. When the electric current leaves the rotor that is in the position most left, it enters the reflector through a pin contact. Through the internal wiring, it gets sent to the connected pin contact. From there on, the electric current goes back through the three rotors. Within the reflector, it is not possible for a pin contact to be connected to itself. This would result in a letter that gets enciphered as itself, which may result in complete messages being enciphered as itself. Beside, it will not be electrically possible, as the two electric currents cannot pass through one wire in opposite directions.

2.3 The Stepping Mechanism

One of the elements which makes the Enigma machine so complicated, is the stepping mechanism. The stepping mechanism is completely mechanical, so no electricity is needed. The stepping mechanism makes the rotors rotate during every key press, which results in every letter being enciphered in a different setting. Before we discuss further details of the stepping of the rotors, we have to take a look at the rotors. Every rotor has an **alphabet ring**, which shows you in which configuration the rotors are if you look through the rotor windows. The alphabet ring has another ring that is attached to it which has a small **notch**, called the **notched ring**. The notched ring and alphabet ring can not rotate separately as they are stuck together. However, these two attached rings can be rotated separately from the internal wiring of the rotor. A rotor looks as follows:

Figure 9: Notch

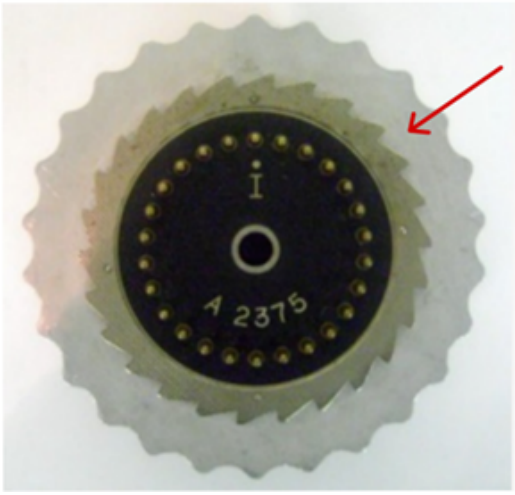


Figure 10: Ratchet

Source: https://www.ciphermachinesandcryptology.com/en/enigmatech.htm

Circled in blue, you see the **notch**. The red arrow points at what is called the **ratchet**. When all the rotors are out of the Enigma machine, it looks as follows:

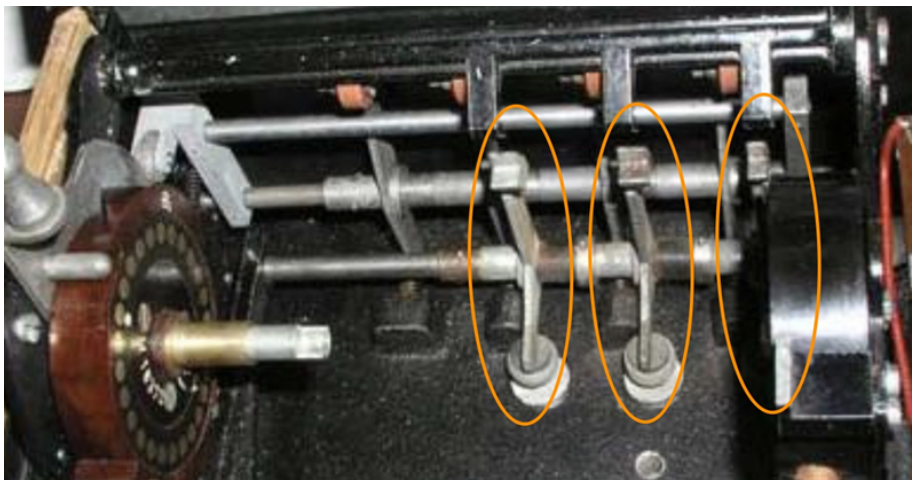


Figure 11: Enigma pawls

Source: https://commons.wikimedia.org/wiki/File:EnigmaStepping.jpg

Circled in orange, we see what are called the **pawls**. In Figure 11, the three pawls can be seen (the third one on the right is not as visible as the left and the middle pawl). When you press a key, the pawls will lift up. The end of the pawl will fit into the ratchet, which will cause the rotor to make one step. However, this does not

happen with all the rotors. Say we look at three rotors next to each other. So, one rotor is in space L, one is in space M, and the last rotor is in space R. The pawl will always reach the ratchet of the rotor in space R, which means that this rotor will always step. This is different for the rotors in spaces L and M. The notched rings of the rotors in spaces M and R will withhold the pawl from fitting into the ratchets of rotors in spaces L and M respectively, and therefore respectively withhold the rotors in spaces L and M from stepping. However, there is a way that the rotors in spaces L and M can step. The rotor in space M can step when the notch of the rotor in space R is perfectly lined up with the middle paw. Then the pawl can fit in the ratchet of the rotor in space M, which makes this rotor step. This is analogous for the rotor in space L.

The position of the notch of a rotor is mostly indicated by the letter that is presented in the rotor window. For example, if the notch of a rotor in space R is on E, this makes the rotor in space M step if the rotor in space R steps from E to F. It is also important to know, that the rotors turn before enciphering a letter. So, the first letter is not enciphered in the start settings.

2.3.1 Double Stepping

Occasionally it occurs that the rotor in the middle steps two consecutive times, this is called **double stepping**. This happens when a certain combination of events occur. Suppose, the rotor in space R (Figure 4) is in notch position, meaning that during the next key press the rotor in space M will advance. Suppose, after that key press, the rotor in space M is in notch position. The pawl will fit in the ratchet of the rotor in space L, but also fits in the notch of the rotor in space M. As the pawl lifts, the rotor in space L advances and the pawl also pushes the rotor in space M, as a result of which the latter advances one step. Now, the rotor in space M has stepped two consecutive times. In other words, it made a double step. So, the rotor in space L advances one step at the same time the rotor in space M steps the second step of the double step.

To clarify double stepping, we give an example. Say we place a rotor called III in space L, a rotor called II in space M, and a rotor called I in space R. Say we put the rotors in the setting IDN. Rotor II has a notch on E, so rotor III can only advance when rotor II steps from E to F. Say that rotor I has a notch on Q, so rotor II advances when rotor I steps from Q to R. We get the following settings, when pressing the keys:

$$\text{I D N} \rightarrow \text{I D O} \rightarrow \text{I D P} \rightarrow \text{I D Q} \rightarrow \text{I E R} \rightarrow \text{J F S} \rightarrow \text{J F T} \rightarrow \cdots$$

We see that after the fourth and fifth key press, the middle rotor will step. Before the fourth key press rotor I is in setting Q, which means rotor II will advance during the next key press. After the fourth key press rotor II is in setting E, which means that rotor III will advance during the next key press, as the notch of rotor II is on E.

2.4 No letter can be encoded as itself and letter pairs

A special and necessary characteristic of the Enigma machine is that no letter can ever be enciphered as itself. This is established by the reflector, which, as we already know, always connects one pin contact to another pin contact and never to same contact. The reflector sends the letter that arrived to another letter and after that, it travels back through the three rotors. It is not possible that the path back crosses the path we traveled to the reflector. Otherwise we would not have ended up in the reflector where we originally did. This results in the fact that in a certain setting, two letters will always form a pair. Say that, for example, in a certain setting we press A and B lights up. In the same setting, if we press B, A will light up, as we travel the same way from A to B but backwards. So, when one encodes the letter I, and it is for example encoded as the letter J, when we encode J in the exact same settings, we get the letter I back.

2.5 Example without ring setting

We will now consider a simple example with only 6 letters in an alphabet. We will look at the reflector and rotors in two dimensions, as if they are folded out. We get the following rotors, which in Figure 12 are all in position A (which is displayed in the little square, functioning as the rotor window), and reflector:

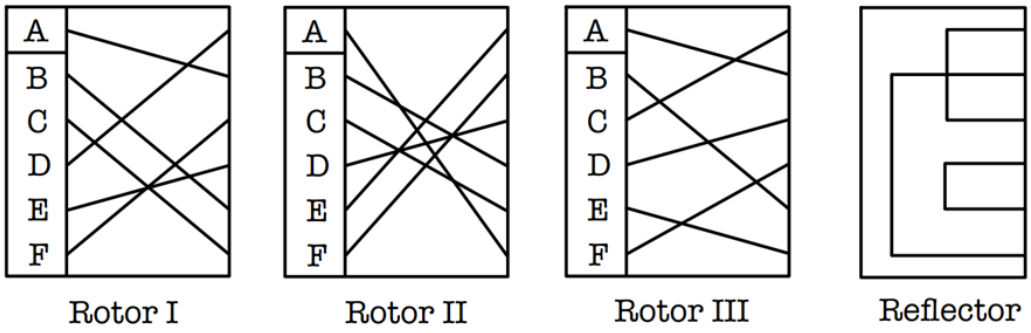


Figure 12: Example rotors

Suppose the rotors have the notch on the following position:

- Rotor I on position C (but this is not relevant),
- Rotor II on position E, and
- Rotor III on position A.

We thus have the following settings:

- Rotor order (from left to right): I - II - III
- Rotor settings: A D F
- Plugboard settings: AB/CD/EF

Before any key is pressed, we have the following starting position:

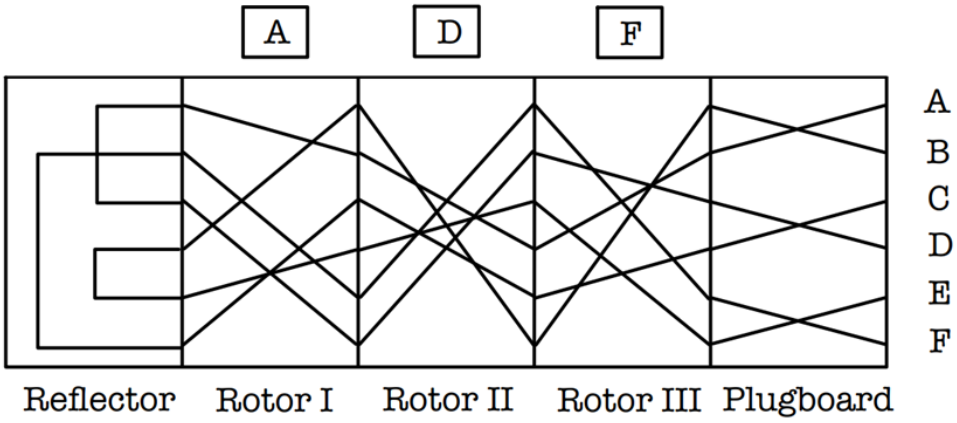


Figure 13: Example starting position

The first letter will be enciphered following this position:

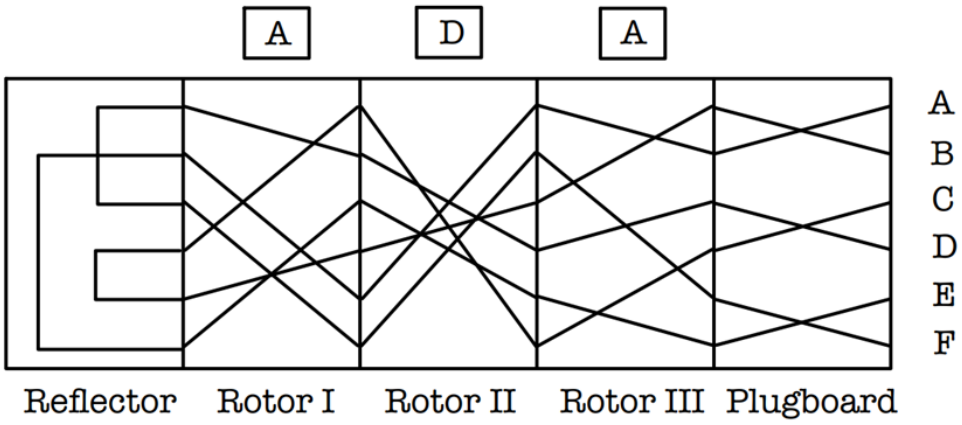


Figure 14: Example settings for enciphering the first letter

The second letter will be enciphered following this position:

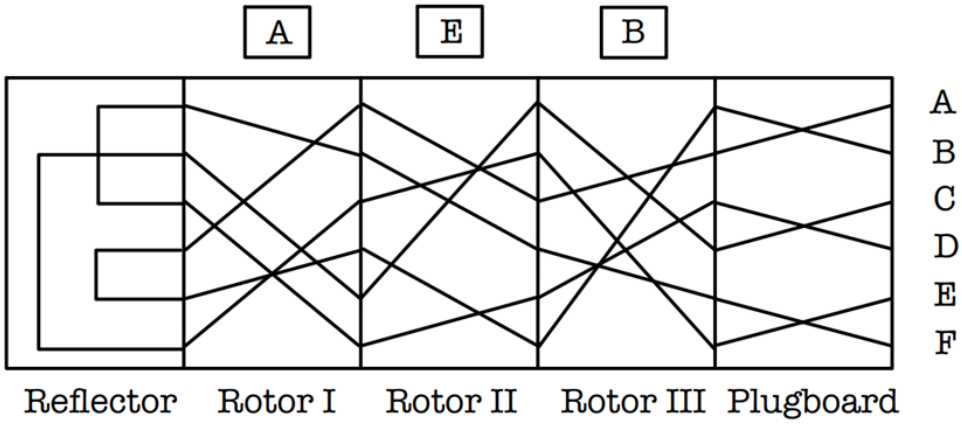


Figure 15: Example settings for enciphering the second letter

The third letter will be enciphered following this position:

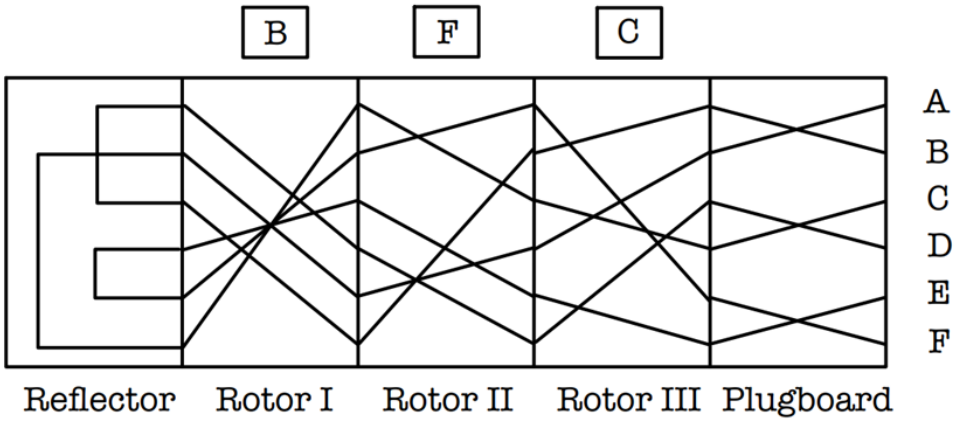


Figure 16: Example settings for enciphering the third letter

In this way, the message A B D will be enciphered to E A B. In this example, you can clearly see that in a certain setting, a letter can not be enciphered as itself, and that all the letters are divided into pairs of two in a certain setting. We also see an example of double stepping.

2.6 Ring settings

One thing we have not discussed yet, is the **ring setting** of a rotor. Up until now, we assumed that the internal wiring of a rotor was stuck to the alphabet ring. However, it is possible to rotate the internal wiring relative to the alphabet ring when setting the rotors in the given settings. When the encoding starts, the internal wiring and the alphabet ring rotate as a whole; they are stuck together. If we look at the rotor, we see the following:

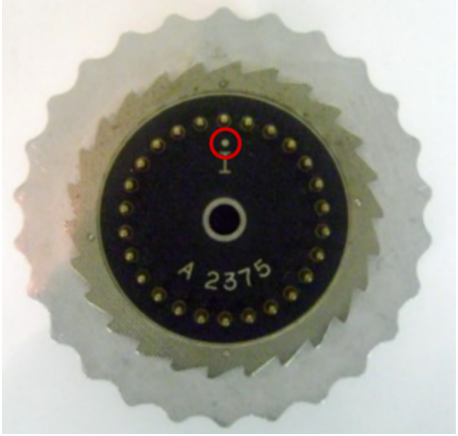


Figure 17: Enigma dot on rotor

Source: https://www.ciphermachinesandcryptology.com/en/enigmatech.htm

Circled in red is a little dot. How this dot is lined up with the alphabet ring, will determine the ring setting. Say the dot is lined up with A on the alphabet ring, then the ring setting for this rotor will be 1 (or A). If the dot lines up with Q, the ring setting will be 17 (or Q). So in the example discussed in Section 2.5, all ring settings were 1.

To make it more clearly, we take a look at rotor I from the previous example:

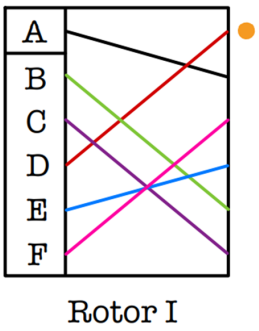


Figure 18: Example rotor with dot

The orange dot will indicate the ring setting, so now the rotor is in ring setting 1. Say we want the rotor to be in ring setting 4 (or D), then we line the orange dot up with D on the alphabet ring:

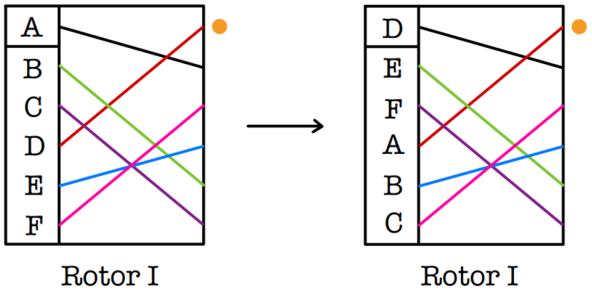


Figure 19: Example rotor in ring setting 4 (D)

If we set the rotor to setting A in the rotor window, we will get the following:

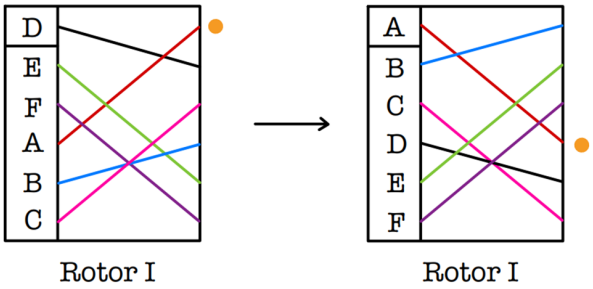


Figure 20: Example rotor in ring setting 4 (D) and setting A

Due to the colors given to the wires, it is easy to see that the internal wiring is completely different than the wiring we started with. Although, the window setting is the same, namely A. This adds to the complexity of the Enigma machine.

In the next paragraph we will show an example which includes ring settings.

2.7 Example with ring setting

We will use the same rotors as in the previous example. We will get the following situation:

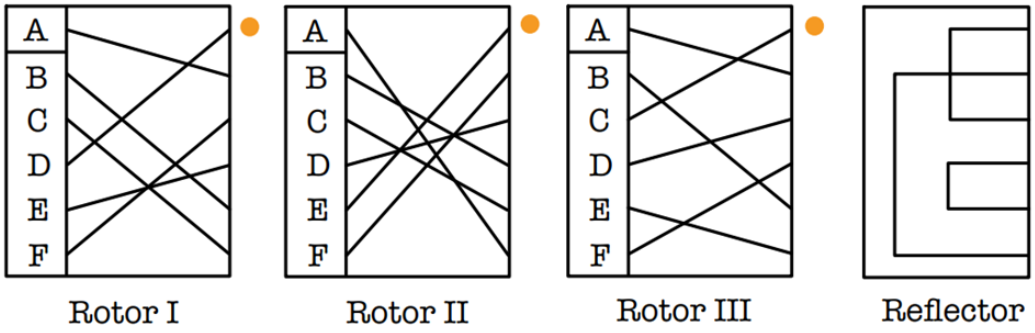


Figure 21: Example rotors including a dot

The rotors will have the notch on the following position:

- Rotor I on position C (but this is not relevant),
- Rotor II on position E, and
- Rotor III on position A.

We have the following settings:

- Rotor order (from left to right): I - II - III
- Rotor settings: A D F
- Plugboard settings: AB/CD/EF
- Ring settings: 3 - 5 - 2

When put in the the right ring settings, the rotors look as follows:

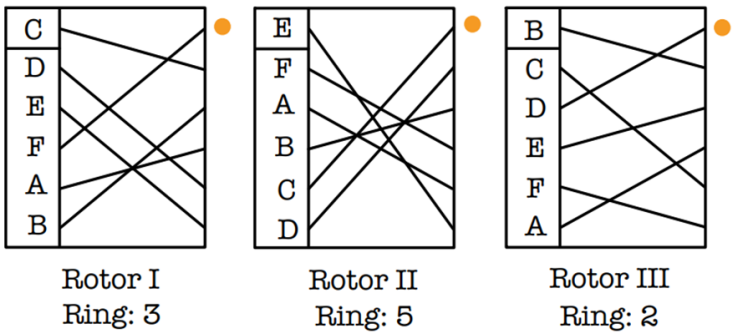


Figure 22: Example rotors in given ring setting

Then we put them in the right rotor setting:

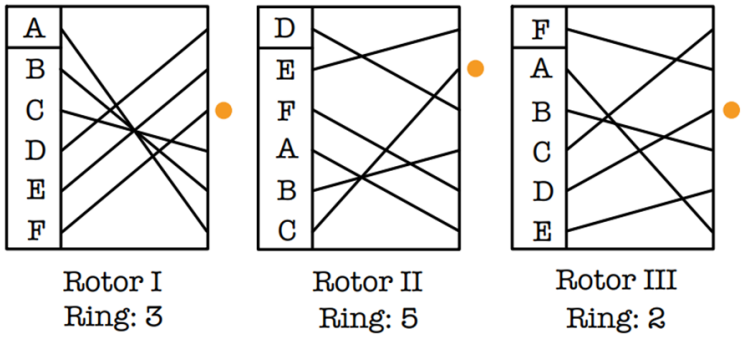


Figure 23: Example rotors in correct rotor setting and ring setting

Before any key is pressed, we have the following starting position:

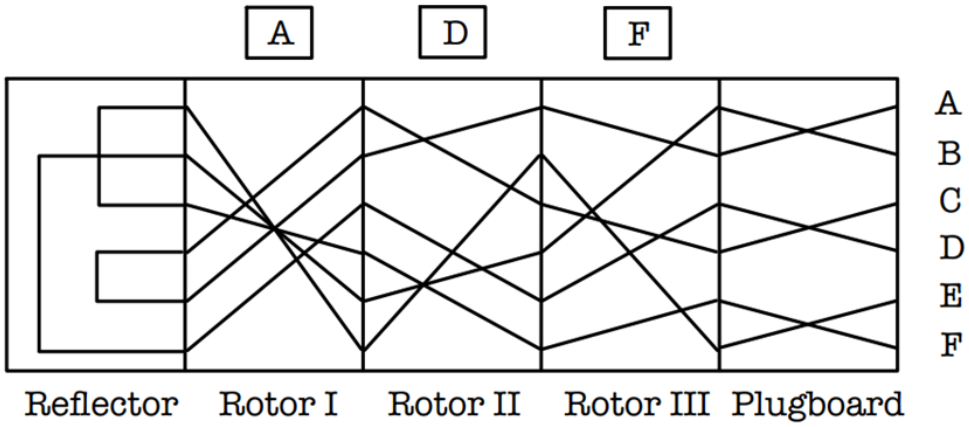


Figure 24: Starting position

The first letter will be enciphered following this position:

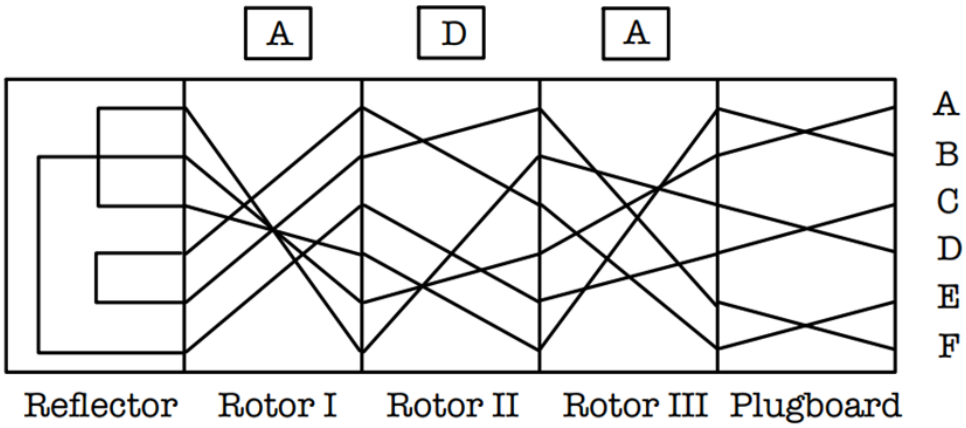


Figure 25: Setting for enciphering the first letter

The second letter will be enciphered following this position:

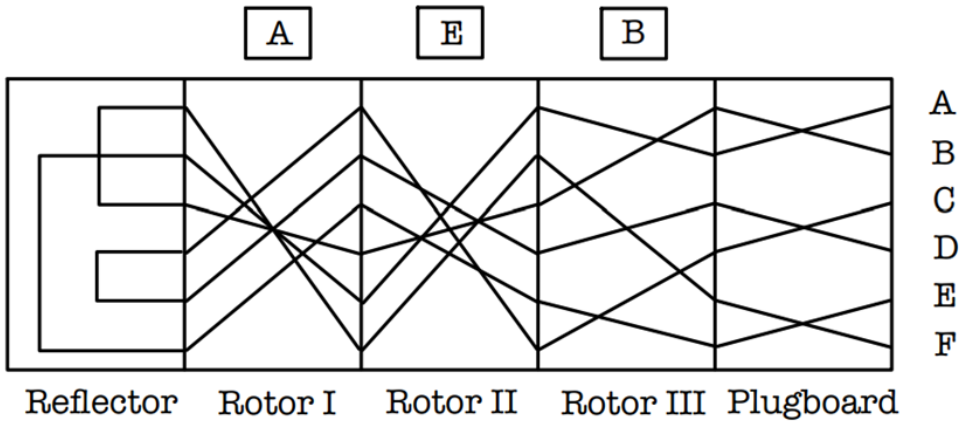


Figure 26: Setting for enciphering the second letter

The third letter will be enciphered following this position:

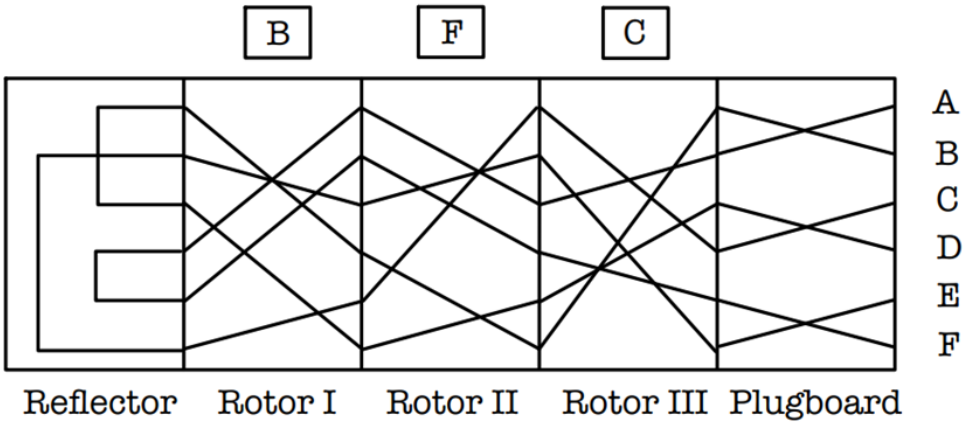


Figure 27: Setting for enciphering the third letter

In this way, the message A B D will be enciphered to C F B.

Sources: [30], [27], [29], [5], [15], [14], [13], [20], [28], [19], [24]

3 Enigma expressed in permutations

In Section 2, we discussed the entire machine. However, since an encoding is nothing but a permutation of the alphabet, we can express this process through permutations.

If we look at the Enigma machine itself, we know that a letter that enters the machine through a tap on the keyboard, always comes out of the machine as another letter. If this were not the case, this would have been a security risk, as then it would be statistically possible to encipher every letter of a message to itself. When we encipher a letter, it is enciphered in the settings we get after pressing the key on the keyboard. So, the rotors turn before enciphering a letter.

As the Enigma machine can completely be expressed through permutations, it is also bijective function as all permutations are bijective. The entire alphabet is mapped back to the alphabet and exists completely out of pairs, since no letter is mapped to no or two or more letters. So, this results in a permutation that completely exists out of cycles of length two.

For simplicity, we replace the letters by numbers. So A corresponds to 0, B corresponds to 1, C to 2 and so on. We use the notation $\mathcal{A} = \{0, 1, 2, \ldots, 24, 25\}$ for the set of letters.

So, if we were to formally express the Enigma machine through a permutation, we have a permutation

$$E : \mathcal{A} \to \mathcal{A}$$

such that

$$\forall x \in \mathcal{A}\ \exists y \in \mathcal{A},\ y \neq x \text{ with } E(x) = y \wedge E(y) = x.$$

However, as already mentioned in Section 2, the Enigma machine exists of three main elements: the plugboard, the rotors, and the reflector. These elements can all be individually expressed through permutations, after which they can be composed to equal the permutation E, representing the entire Enigma machine. All elements, also referred to as stations in Section 2, will be discussed independently.

3.1 Plugboard

In the plugboard, two letters get switched when they are connected by a cable. So, within the plugboard, either two letters are mapped to each other or a letter is mapped to itself. Thus, the plugboard can be defined as a bijective function with the possibility of letters being mapped to themselves. The number of letters being mapped to themselves, depends on the number of cables used. Say, we use $0 \leq c \leq 13$ cables, then $2c$ letters are part of a pair and $26 - 2c$ letters are mapped to themselves.

So, if we were to formally express the plugboard through a permutation, we have a permutation

$$P : \mathcal{A} \to \mathcal{A}$$

of the form

$$(b_1 b_2)(b_3 b_4) \cdots (b_{2c-1} b_{2c})(a_1)(a_2) \cdots (a_{26-2c}).$$

It holds that $P^2 = I$ with I the identity map $I = (1)(2) \cdots (25)(26)$. P is fixed during encoding.

3.2 Reflector

After passing through the rotors, a letter arrives at the reflector. The concept is the same as that of the plugboard, but the letters are permanently connected and a letter can not be mapped to itself. So, the reflector can also be seen as a bijective function.

So, if we were to formally express the reflector through a permutation, we have a permutation

$$F : \mathcal{A} \to \mathcal{A}$$

of the form

$$(h_1 h_2)(h_3 h_4) \cdots (h_{25} h_{26}).$$

Consequently, $F^2 = I$ with I the identity map $I = (1)(2) \cdots (25)(26)$. F is fixed during encoding.

3.3 Rotors

The permutations associated with the rotors are a bit more complicated than the ones before, since we have to take the initial setting and the ring setting into account. Also, the rotors step, and double stepping occurs.

As we saw in Figure 4, we have three spaces where a rotor can be placed: L, M and R. Say T is a finite collection of rotors. Let $\{\delta_L, \delta_M, \delta_R\} \subset T$ a choice of three different rotors. Rotor δ_L is placed in space L, rotor δ_M is placed in space M, and rotor δ_R is placed in space R.

We have to define the rotation of the rotors, as we have to rotate them in the correct position and as they rotate every time a letter on the keyboard is tapped. If we, for example, look at Figures 24 and 25, we see that the rotors rotate upwards. In other words, they rotate to the back. So, say d represents the rotation, then

$$d = \begin{pmatrix} 0 & 1 & 2 & \cdots & 23 & 24 & 25 \\ 1 & 2 & 3 & \cdots & 24 & 25 & 0 \end{pmatrix} = (0\ 1\ 2\ \ldots\ 24\ 25).$$

Now we have defined a one step rotation, we look at the internal wiring of the rotors. Every rotor has internal wiring that connects 26 plates to 26 contacts. One can see this as connecting all 26 letters to the alphabet to the 26 letters of the alphabet. In other words, the internal wiring of a rotor is a permutation. We will define the internal wiring of any rotor as follows:

Definition 1. *For a rotor δ, with $\delta \in T$, the internal wiring of δ is represented by a permutation*

$$f_\delta : \mathcal{A} \to \mathcal{A}$$

which maps the alphabet to the alphabet.

If we rotate rotor δ n times, we would get the permutation

$$d^{-n} \circ f_\delta \circ d^n.$$

Notice that since the rotor has made one full rotation after 26 steps, we have to work with modulo 26. For $x \in \mathbb{Z}$ it holds that $(x \mod 26) \in \mathcal{A}$. This corresponds to the numbers we replaced the letters with. So, we get

$$d^{-(n \mod 26)} \circ f_\delta \circ d^{(n \mod 26)}.$$

3.3.1 Setting the rotors

Next, we have to set rotor δ in the correct ring setting and initial setting. We define the settings below.

Definition 2. *Say s_δ represents the given ring setting of rotor $\delta \in \{\delta_L, \delta_M, \delta_R\}$ with $s_\delta \in \mathcal{A}$.*

However, this is not the amount of steps a rotor needs to advance to be in the given ring setting.

Lemma 1. *We define r_δ to be the number of steps rotor $\delta \in \{\delta_L, \delta_M, \delta_R\}$ has to advance to be in the given ring setting. It holds that*

$$r_\delta = 26 - s_\delta.$$

Proof. Since the rotors rotate to the back, and the ring setting represents the number of advancements of the internal wiring to the front, rotor δ has to step $26 - s_\delta$ times to be in the correct ring setting. In this way, we line up the orange dot, that has been introduced in previous examples, with the correct letter on the alphabet ring. □

To visualize this proof, we will consider the following example

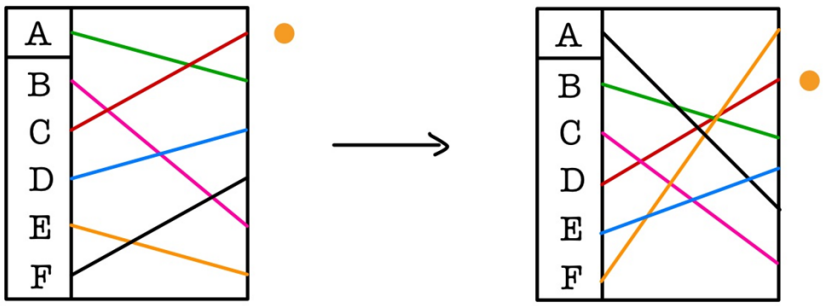


Figure 28: Example ring setting

In Figure 28 a rotor is being set from ring setting 0 (corresponding to A) to ring setting 1 (corresponding to B). One can see that the colored internal wiring has stepped on step downwards. However, the rotors rotate upwards instead of downwards. Therefore, in this example we have to step upwards five times, which corresponds to stepping $6 - 1$ times. If we translate this to the complete 26 letter alphabet, we will get that the number of steps that the rotor needs to make to be in the given ring setting, is equal to $r_\delta = 26 - s_\delta$ with $\delta \in \{\delta_L, \delta_M, \delta_R\}$.

So, after setting rotor δ in the given ring setting, $\delta \in \{\delta_L, \delta_M, \delta_R\}$, we get the following permutation

$$d^{-(r_\delta \mod 26)} \circ f_\delta \circ d^{(r_\delta \mod 26)}.$$

Next, we have to set the rotor in the given initial setting. To make the given letter show through the window at the top of the machine, the rotor has to rotate as a whole.

Definition 3. *We define q_δ to be the given initial position of rotor $\delta \in \{\delta_L, \delta_M, \delta_R\}$ with $q_\delta \in \mathcal{A}$.*

So, rotor δ as a whole has to step q_δ times to be in the given initial position.

Thus, setting rotor δ, with $\delta \in \{\delta_L, \delta_M, \delta_R\}$, in the given ring setting and initial position, gives the following permutation:

$$d^{-((r_\delta + q_\delta) \mod 26)} \circ f_\delta \circ d^{((r_\delta + q_\delta) \mod 26)}.$$

3.3.2 Including the stepping mechanism

Next, we will consider the stepping mechanism of the rotors. Other than the regular stepping of the rotors, we also have to take double stepping of rotor δ_M into account.

We first start with the rotor δ_R. This rotor steps with every key press. Say we press n keys on the keyboard, then the right rotor will rotate n times. If we were to express this through a permutation, we get

$$R(n) := d^{-((r_{\delta_R} + q_{\delta_R} + n) \mod 26)} \circ f_{\delta_R} \circ d^{((r_{\delta_R} + q_{\delta_R} + n) \mod 26)} \tag{1}$$

Now, we have to consider the more complicated cases. To do so, we first have to define some things.

Definition 4. *The **notch letter** is the letter X such that the rotor on its left advances if the rotor steps from X to the next letter in the window.*

The notch letter of a rotor is known. Its position determines the number of steps it takes till the rotor on the left steps for the first time.

Definition 5. *Define by γ_δ the number of steps rotor δ, with $\delta \in \{\delta_L, \delta_M, \delta_R\}$, has to advance from the initially set letter to the letter after the notch letter.*

Clearly, $1 \leq \gamma_\delta \leq 26$. Note that after γ_{δ_R} steps, rotor δ_M has stepped due to the position of the notch of rotor δ_R for the first time. Also note that the notch position of rotor δ_L is irrelevant, since there is no rotor on its left.

We will first look at rotor δ_M.

Theorem 2. *The number of steps that rotor δ_M has advanced after $n \geq 1$ key presses, is equal to*

$$k_2(n) = \begin{cases} 0, & n < \gamma_{\delta_R} \text{ and } \gamma_{\delta_M} \neq 1 \\ 1, & n < \gamma_{\delta_R} \text{ and } \gamma_{\delta_M} = 1 \\ 1 + \left\lfloor \frac{n-\gamma_{\delta_R}}{26} \right\rfloor + \left\lceil \frac{n-x+1}{650} \right\rceil, & 1 \leq \gamma_{\delta_R} \leq n \text{ and } \gamma_{\delta_M} \neq 1 \\ \left\lfloor \frac{n-1}{26} \right\rfloor + \left\lceil \frac{n}{650} \right\rceil, & \gamma_{\delta_R} = 1 \text{ and } \gamma_{\delta_M} = 1 \\ 3 + \left\lfloor \frac{n-\gamma_{\delta_R}}{26} \right\rfloor + \left\lfloor \frac{n-\gamma_{\delta_R}-625}{650} \right\rfloor, & 2 \leq \gamma_{\delta_R} \leq n \text{ and } \gamma_{\delta_M} = 1 \end{cases} \tag{2}$$

with

$$x = \begin{cases} 2, & \text{if } \gamma_{\delta_M} = 2 \text{ and } \gamma_{\delta_R} = 1 \\ \gamma_{\delta_R} + 1, & \text{if } \gamma_{\delta_M} = 2 \text{ and } \gamma_{\delta_R} \neq 1 \\ \gamma_{\delta_R} + 26 \cdot \gamma_{\delta_M} - 51, & \text{if } \gamma_{\delta_M} \geq 3 \end{cases}. \tag{3}$$

Proof. Notice that $n \geq 1$ and that $1 \leq \gamma_\delta \leq 26$. Also keep into account that, as described before, rotor δ_M steps as a consequence of two different situations. One situation being that rotor δ_R is in notch position (so when δ_R advances one step from this position, then δ_M advances one step). The other situation is that the notch of rotor δ_M is lined up for the rotor δ_L to advance one step, thus causing δ_M to step simultaneously with δ_L. In other words, if rotor δ_M is in notch position. We will consider five different cases.

1. $n < \gamma_{\delta_R}$ and $\gamma_{\delta_M} \neq 1$

In this case, δ_M has not stepped due to δ_R as $n < \gamma_{\delta_R}$. Also, because of that, δ_M cannot get into the position to step due to its own notch being in the correct position. In addition, δ_M is not in notch position because it holds that $\gamma_{\delta_M} \neq 1$. So, in this case, rotor δ_M has advanced zero times after n key presses.

2. $n < \gamma_{\delta_R}$ and $\gamma_{\delta_M} = 1$

As $\gamma_{\delta_M} = 1$, rotor δ_M advances one step as we start in the middle of double stepping and $n \geq 1$. So, δ_M steps due to itself being in notch position. However, it is not possible for δ_M to step another time as a consequence of δ_R being in notch position, as $n < \gamma_{\delta_R}$. So, rotor δ_M only steps once.

3. $1 \leq \gamma_{\delta_R} \leq n$ and $\gamma_{\delta_M} \neq 1$

For this case, we have that $\gamma_{\delta_M} \neq 1$. When $\gamma_{\delta_M} = 1$, we have special cases that we will discuss later on. We know that, as rotor δ_R has advanced at least γ_{δ_R} steps, that rotor δ_M has at least advanced one step. Also, we know that in the remaining $(n - \gamma_{\delta_R})$ steps, rotor δ_M steps every 26 times. So the amount of steps rotor δ_M has advanced due to rotation of rotor δ_R, is

$$1 + \left\lfloor \frac{n - \gamma_{\delta_R}}{26} \right\rfloor.$$

However, we did not take double stepping into account. As explained before, this may occur as a consequence of the notch position of rotor δ_M, causing rotor δ_L to step, which in turn forces rotor δ_M to step a second time. Double stepping occurs once every 650 steps. We explain this by showing a simplified example with the first six letters of the alphabet.

Say the notch letters of both rotors δ_M and δ_R are A. So, the rotor on its left advances if the rotor steps from A to B in the window. Say we start with letters A, F, and A showing through the windows from left to right. We get the following:

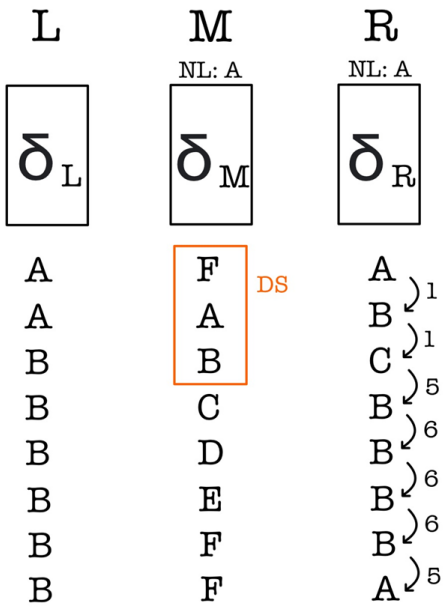


Figure 29: Example stepping mechanism

After two key presses, we immediately see that rotor δ_M stepped two consecutive times. After these two steps, rotor δ_R has to step five times to end up at the letter after the notch letter. Then, we let rotor δ_R rotate enough complete cycles to make F shows up in the middle again. We know that with these settings, double stepping occurs when F shows in the middle and A shows on the right. So, we have to take five extra steps to end up in this situation. We can easily translate this to 26 letters. Then, we get $2 + 25 + 23 \times 26 + 25 = 650$. So a double step of the middle rotor happens every 650 key presses.

However, in Figure 29 we start at the situation that double stepping occurs after two key presses. This is not always the case. So, we denote by x the number of key presses up to and including rotor δ_M stepping for the first time due to its notch being in the position to make rotor δ_L advance.

So, the number of times rotor δ_M advances due to its notch being in the position to make rotor δ_L advance, is equal to

$$\left\lceil \frac{n - x + 1}{650} \right\rceil.$$

As we just mentioned, x is the number of key presses one has to do up to and including the first time rotor δ_M has advanced two consecutive times. So, when $n \geq x$, $(n - x)$ is the amount of steps left after double stepping occurred for the first time. We divide by 650 as double stepping occurs every 650 key presses. So, for every 650 extra key presses after double stepping occurred once, one will be added because we are rounding up. However, if $n = x$, the number of steps rotor δ_M advances due to itself being in notch position would be equal to zero, but it should be equal to one. So, therefore, in the denominator we have to add $+1$. As the first double stepping always occurs in the first 650 steps, it holds that $x \leq 650$ and thus $\left\lceil \frac{n-x+1}{650} \right\rceil \geq 0$. So, when $n < x$, it holds that $\left\lceil \frac{n-x+1}{650} \right\rceil = 0$.

So, the number of steps of rotor δ_M makes when $1 \leq \gamma_{\delta_R} \leq n$ and $\gamma_{\delta_M} \neq 1$ is equal to

$$1 + \left\lfloor \frac{n - \gamma_{\delta_R}}{26} \right\rfloor + \left\lceil \frac{n - x + 1}{650} \right\rceil.$$

For the specification of x, we have to consider four cases. We again use a simplified example with six letters to support the different cases described.

1. $\gamma_{\delta_M} = 2$ **and** $\gamma_{\delta_R} = 1$

When $\gamma_{\delta_M} = 2$ and $\gamma_{\delta_R} = 1$, we are in the position for rotor δ_M to step twice. In the simplified example, for rotor δ_M the letter F should show through the window. For rotor δ_R it would mean that the letter A shows through the window. To make sure the double stepping occurred, we have to step two times. Therefore, in this case, $x = 2$.

2. $\gamma_{\delta_M} = 2$ **and** $\gamma_{\delta_R} \neq 1$

When $\gamma_{\delta_M} = 2$ and $\gamma_{\delta_R} \neq 1$, rotor δ_M is in the correct position to double step, but rotor δ_R is not yet in the correct position. The right position of rotor δ_R is obtained when the notch letter is in the right position, so after $(\gamma_{\delta_R} - 1)$ steps. Then we are in the correct position to double step, which takes two more steps of the rotor on the right. Therefore, in this case, $x = (\gamma_{\delta_R} - 1) + 2 = \gamma_{\delta_R} + 1$. The following simplified example shows what occurs.

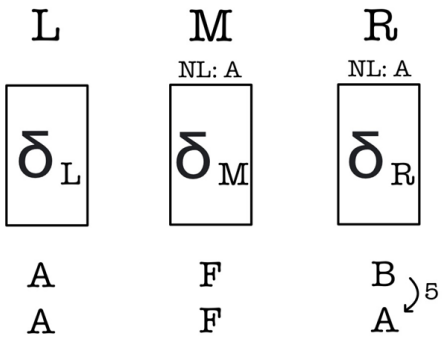


Figure 30: Example stepping mechanism

3. $\gamma_{\delta_M} \geq 3$

We first have to determine how many steps the rotor δ_R on the right has to step to get the correct situation for double stepping to occur. We explain this again on the basis of the simplified example we have been using. Say we have the following:

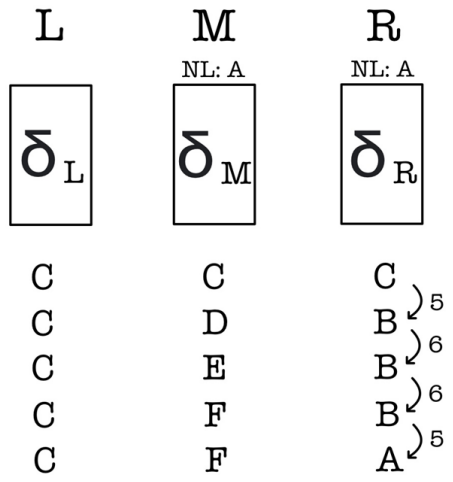


Figure 31: Example stepping mechanism

We start with C, C, C in the windows. First, we step such that rotor δ_R ends up at the letter after the notch letter. Stepping to this, rotor δ_M also has advanced one step. After that, we have to make sure rotor δ_M steps to the correct position to double step. So, in this case rotor δ_M has stepped once and we have to end up two letters before the letter after the notch letter. In other words, in the example rotor δ_M has stepped from C to

D, and for double stepping to occur, rotor δ_M has to end up at F. To do so, rotor δ_M has to step $(\gamma_{\delta_M} - 3)$ times. To do so, $(\gamma_{\delta_M} - 3) \cdot 6$ keys have to be pressed on the keyboard. Now rotor δ_M is in the correct position to double step. However, rotor δ_R is on B, which is the letter after the notch letter. So we have to step 5 more times to end up at the notch letter, and thus the correct position to double step. After that, we have to double step, through two key presses. If we translate this to 26 letters, we get

$$x = \gamma_{\delta_R} + (\gamma_{\delta_M} - 3) \cdot 26 + 25 + 2 = \gamma_{\delta_R} + 26 \cdot \gamma_{\delta_M} - 51,$$

which is positive as $\gamma_{\delta_M} \geq 3$.

So, we obtain Equation 3 for x.

4. $\boldsymbol{\gamma_{\delta_R} = 1}$ and $\boldsymbol{\gamma_{\delta_M} = 1}$

This case is a special case, which can only occur as a starting position. In this case, both rotors δ_M and δ_R are in notch position. When we press a key at the keyboard, all three rotors will simultaneously rotate. As δ_R always rotate, we say that the rotation of rotor δ_M was due to it being in notch position and not due to the step of rotor δ_R. So, rotor δ_M stepping due to rotor δ_R every 26 from now on, and due to the notch position of itself every 650 steps. So we get

$$\left\lfloor \frac{n-1}{26} \right\rfloor + \left\lceil \frac{n}{650} \right\rceil.$$

The first step of δ_M is taken into account in $\left\lceil \frac{n}{650} \right\rceil$, as for $n = 1$ it holds that $\left\lceil \frac{n}{650} \right\rceil = 1$.

5. $\boldsymbol{2 \leq \gamma_{\delta_R} \leq n}$ and $\boldsymbol{\gamma_{\delta_M} = 1}$

First, we know that since $2 \leq \gamma_{\delta_R} \leq n$, that rotor δ_M has stepped at least one time due to rotor δ_R being in notch position. After these γ_{δ_R} steps, rotor δ_M advances every 26 times due to rotor δ_R being in the notch position. So, as in a previous case, the number of steps rotor δ_M advances due to rotor δ_R being in the notch position, is equal to

$$1 + \left\lfloor \frac{n - \gamma_{\delta_R}}{26} \right\rfloor. \tag{4}$$

We also have to consider the steps δ_M advances due to itself being in notch position. In this case, we know that δ_M steps after one key press since $\gamma_{\delta_M} = 1$. This step is not due to δ_R being in notch position. Therefore, we officially are not in a case of double stepping, but it was the way the settings were chosen, that made δ_M step. Therefore, the first case of double stepping is not in 650 key presses. So, we have to find the number of steps we need to advance to reach our first official case of double stepping. To explain this, we make use of a simplified example that we used before:

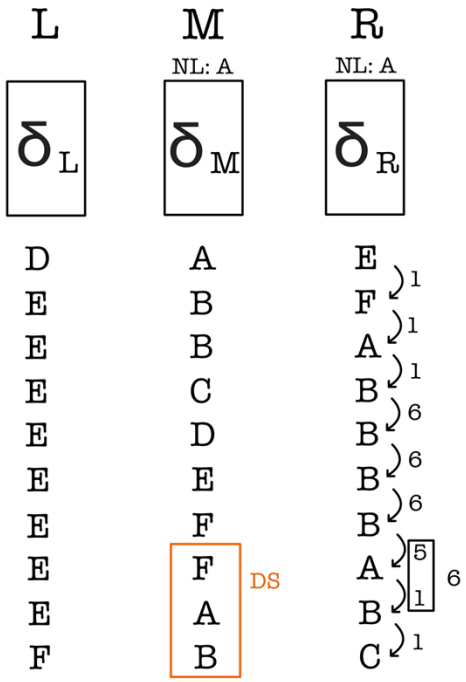


Figure 32: Example stepping mechanism

In this example, we have $\gamma_{\delta_R} = 3$ and $\gamma_{\delta_M} = 1$. We see that in the first step, both rotors δ_M and δ_L advance. Then, the the next time that rotor δ_M advances because of its notch being in the correct position, is after $\gamma_{\delta_R} + 4 \cdot 6 + 1$ key presses. If we translate this to the 26 letter alphabet, we get a case of double stepping after $\gamma_{\delta_R} + 24 \cdot 26 + 1 = \gamma_{\delta_R} + 625$ key presses. So, during this process, rotor δ_M stepped two times due to its own notch begin in the correct position. After the first case of double stepping, double stepping occurs every 650 key presses. So, the number of steps rotor δ_M advances due to its own notch being in notch position, is equal to

$$2 + \left\lfloor \frac{n - (\gamma_{\delta_R} + 625)}{650} \right\rfloor$$

which is equal to

$$2 + \left\lfloor \frac{n - \gamma_{\delta_R} - 625}{650} \right\rfloor. \tag{5}$$

Adding up Equations 4 and 5, we get

$$1 + \left\lfloor \frac{n - \gamma_{\delta_R}}{26} \right\rfloor + 2 + \left\lfloor \frac{n - \gamma_{\delta_R} - 625}{650} \right\rfloor = 3 + \left\lfloor \frac{n - \gamma_{\delta_R}}{26} \right\rfloor + \left\lfloor \frac{n - \gamma_{\delta_R} - 625}{650} \right\rfloor \tag{6}$$

So, we find the δ_M advanced after n key presses, is as in Equation 2.

□

Therefore, we can express the rotation of rotor δ_M, through the following permutation

$$M(n) := d^{-((r_{\delta_M} + q_{\delta_M} + k_2(n)) \mod 26)} \circ f_{\delta_M} \circ d^{((r_{\delta_M} + q_{\delta_M} + k_2(n)) \mod 26)} \tag{7}$$

Finally, the number of steps the rotor δ_L steps is derived analogously. For every case we distinguished, when rotor δ_M stepped due to itself being in the notch position. Every time this happens, rotor δ_L steps as well since the pawl will fit the ratchet of both rotors δ_L and δ_M. The number of steps δ_M advances due to its own notch being in notch position, has been described for every case. So we get the following:

Corollary 2.1. *The number of steps that the left rotor advances after $n \geq 1$ key presses, is equal to*

$$k_2(n) = \begin{cases} 0, & n < \gamma_{\delta_R} \text{ and } \gamma_{\delta_M} \neq 1 \\ 1, & n < \gamma_{\delta_R} \text{ and } \gamma_{\delta_M} = 1 \\ \left\lceil \frac{n-x+1}{650} \right\rceil, & 1 \leq \gamma_{\delta_R} \leq n \text{ and } \gamma_{\delta_M} \neq 1 \\ \left\lceil \frac{n}{650} \right\rceil, & \gamma_{\delta_R} = 1 \text{ and } \gamma_{\delta_M} = 1 \\ 2 + \left\lfloor \frac{n-\gamma_{\delta_R}-625}{650} \right\rfloor, & 2 \leq \gamma_{\delta_R} \leq n \text{ and } \gamma_{\delta_M} = 1 \end{cases} \tag{8}$$

with

$$x = \begin{cases} 2, & \text{if } \gamma_{\delta_M} = 2 \text{ and } \gamma_{\delta_R} = 1 \\ \gamma_{\delta_R} + 1, & \text{if } \gamma_{\delta_M} = 2 \text{ and } \gamma_{\delta_R} \neq 1 \\ \gamma_{\delta_R} + 26 \cdot \gamma_{\delta_M} - 51, & \text{if } \gamma_{\delta_M} \geq 3 \end{cases}.$$

Proof. The number of steps rotor δ_L advances is equal to the number of times rotor δ_M steps due to itself being in the notch position. This has been indicated for every case in the proof of Theorem 2. So, we find Equation 8 with the same x (Equation 3) as defined in Theorem 2.

□

As a consequence, we can express the rotation of rotor δ_L through the following permutation

$$L(n) := d^{-((r_{\delta_L}+q_{\delta_L}+k_1(n)) \mod 26)} \circ f_{\delta_L} \circ d^{((r_{\delta_L}+q_{\delta_L}+k_1(n)) \mod 26)} \tag{9}$$

So, if we express going through all three rotors in one permutation, we get

$$\begin{aligned} Q(n) &= L(n) \circ M(n) \circ R(n) \\ &= d^{-((r_{\delta_L}+q_{\delta_L}+k_1(n)) \mod 26)} \circ f_{\delta_L} \circ d^{((r_{\delta_L}+q_{\delta_L}+k_1(n)-r_{\delta_M}-q_{\delta_M}-k_2(n)) \mod 26)} \circ f_{\delta_M} \\ &\circ d^{((r_{\delta_M}+q_{\delta_M}+k_2(n)-r_{\delta_R}-q_{\delta_R}-n) \mod 26)} \circ f_{\delta_R} \circ d^{((r_{\delta_R}+q_{\delta_R}+n) \mod 26)}. \end{aligned}$$

For the permutation representing the Enigma I machine, we may conclude the following

Theorem 3. *Let r_{δ_i}, q_{δ_i} with $i \in \{L, M, R\}$ be given, as well as γ_{δ_i}, $i \in \{M, R\}$. Then*

$$E(n) = P^{-1} \circ Q^{-1}(n) \circ F \circ Q(n) \circ P \tag{10}$$

with

$$Q(n) = L(n) \circ M(n) \circ R(n)$$

where $R(n)$, $M(n)$, and $L(n)$ are given by Equations 1, 7, and 9 respectively.

4 Different versions of Enigma

There is not one type of Enigma machine. Over time, many versions and types have been created and used. Sometimes, only minor changes were made, as a result of which a new version of Enigma machine would be similar to the previous version. However, other times there would be major changes in design and concept.

It all started with Die Handelsmaschine, produced by the company Scherbius and Ritter in 1923. It was the first ever cipher machine being sold under the Enigma brand. Other than the more advanced versions of the machine, this version had a built in typewriter. This made the machine very big and heavy. Besides, the machine had four rotors that were permanently placed in the machine, and lacked a plugboard and reflector. Because of lacking a reflector, the current was not returned and went through the wheels only once in a specific direction. Therefore, to decrypt messages, the current had to be reversed. This could be done by a handle on the machine. However, the machine turned out to not be reliable at all times and had to be improved. This is when Die Schreibende Enigma came out. This version was very similar to Die Handelsmaschine, and it still printed the result directly. Its successor, Enigma H29, was the last version of Enigma that printed on paper.

During the process of improving the printing Enigma machines, Enigma A was produced and introduced in 1924. This was the first ever version having a lampboard instead of a printing mechanism, which made the machine much smaller and more affordable. Enigma B was the successor of Enigma A. In comparison to Enigma A, it had automatic advancing rotors, three cipher wheels that could move and also could be removed, a fixed reflector, a letter ring that could be moved for ring settings, and some other small details. After Enigma B, Enigma C was developed, which was very similar to Enigma B. Enigma C had a special version, called Funkschlüssel C. Other than the Enigma C, which just had 26 letters, Funkschlüssel C had 29 letters, the 26 letters of the alphabet and Ä, Ö and Ü. However, it is not possible to have an odd amount of letters. Therefore, the letter X stayed unencrypted at all times. Some other small changes were made as well.

After Enigma C, the much more improved Enigma D came out. In the Enigma D machine the rotors could be removed and the order of the rotors could be changed. Also, the reflector became settable. It could be rotated into 26 different positions. Because of the settable reflector, an extra window was added on the left of the rotor windows. Through this window, the setting of the reflector could be seen. About a year after Enigma D was introduced, Enigma K was developed in 1927. However, Enigma K was more or less identical to Enigma D, since there were no major differences between the two types.

Then, around 1927-1929, Enigma I was developed, the Enigma machine the German Army and Air Force used. It was based on the Enigma D version, but with a fixed reflector and the first ever plugboard. The machine first had three different rotors one could choose from, but to improve the strength of the machine, two more rotors were supplied in 1938. In 1934 the German Navy introduced a similar machine known as M3. A difference with Enigma I is, that the numbers on the rotor rings were replaced by letters. To be able to communicate with the German Army and Air Force, this machine had the same five rotors as Enigma I. In 1939 three extra rotors were added, which were exclusively used for the German Navy. Then, in 1942, the German Navy decided to start using Enigma M4, produced mainly for the U-boats. This machine had place for an extra rotor between the reflector and the most left rotor. This fourth rotor could not be interchanged with other rotors and did not rotate while enciphering.

The types of Enigma discussed above are just a selection of many more types that existed and were used. However, only Enigma I is being discussed in this thesis.

Sources: [9], [18], [10], [11], [12], [1], [2], [3], [4], [6], [7], [8]

5 Number of starting settings

One thing that makes Enigma so complicated to break, is the number of different settings it has. Therefore, in this paragraph we discuss the amount of starting settings in which the machine can be set up. The number of settings consists of the combination of different elements of the machine.

First of all, there is a specific amount of rotors to choose from, say n. Three need to be chosen, since the machine fits three rotors. Furthermore, the order of the rotors matters and a particular rotor can only be chosen once. Therefore, if there are n rotors to choose from, there are $\frac{n!}{(n-3)!}$ possible rotor orders.

Second, the plugboard settings need to be considered. At the front of the machine, there are 26 sockets, one for every letter. A choice would be to not use any cables and thus not to pair up any letters. However, it is also possible to pair up every letter by using 13 cables, as every cable connects two letters and every letter can only be connected once on the plugboard. Say c is the amount of cables that we use, then $2 \cdot c$ letters will be paired up, resulting in c pairs. So, there are $\binom{26}{2 \cdot c} = \frac{26!}{(26-2 \cdot c)! \cdot (2 \cdot c)!}$ possibilities for choosing $2 \cdot c$ out of 26 letters. Now, these $2 \cdot c$ letters need to be paired up. Start by plugging the first cable in a random socket. Then there are $2 \cdot c - 1$ options left to put the other end of the cable in. For the next cable this process can be repeated. One end of a cable is plugged in a random socket that is not used yet, which leaves $2 \cdot c - 3$ options to put the other end of the cable in, since one cable is fully plugged in and from one cable just one end is plugged in. This can be repeated until two unused plugs are left, which we obviously connect. So we get:

$$(2 \cdot c - 1) \times (2 \cdot c - 3) \times (2 \cdot c - 5) \times \cdots \times 3 \times 1 = \frac{(2 \cdot c)!}{(2 \cdot c) \times (2 \cdot c - 2) \times (2 \cdot c - 4) \times \cdots \times 4 \times 2} = \frac{2 \cdot c}{c! \times 2^c}.$$

Third, there are the settings of the individual rotors, which are a fixed given in this machine. For every rotor, there are 26 ways to rotate the internal wiring of the rotor relative to the machine. Since the machine fits three rotors, this gives $26^3 = 17,576$ possible combinations. However, the outer ring of the rotor also needs to be considered. Even though the outer ring has nothing to do with the internal wiring of the rotor, the ring does have a notch. It does matter where these notches are positioned because this affects when the rotors will advance a step. However, the position of the notch only matters for the middle rotor and the rotor on the right. So, for the middle and left rotor, each has 26 ways to set the ring position. This gives $26 \times 26 = 676$ possibilities to set the rings.

Lastly, we have to combine all these possibilities by multiplication. We get:

$$\frac{n!}{(n-3)!} \times \frac{26!}{(26 - 2 \cdot c)! \cdot (2 \cdot c)!} \times (2 \cdot c - 1) \times (2 \cdot c - 3) \times \cdots \times 3 \times 1 \times 17576 \times 676.$$

This can also be found by the following Python program in Appendix A, where the variables can be changed. In Appendix B one can find a programmed version of the Enigma I machine.

Sources: [16], [23], [25]

6 How Enigma was broken

Enigma has a long history. It played a significant role in World War II. Although much physical fighting was going on, there also was a fight that was not to be known or seen by many. It was the fight to break the Enigma.

6.1 Providing secret information — November 1931

It all started on 1 November 1931. A man called Hans Thilo Schmidt handed over the Enigma manuals to the French Secret Service. He first had sent a letter to the French Deuxième Bureau, a security service in France, saying that he had access to manuals for coding machines that the Germans had been using since June 1930. Schmidt worked at the Cipher Office. At this office, ciphers for the German Armed Forces were created. To earn extra money, Schmidt wanted to sell the documents to Lemoine, who was a secret agent at Deuxième Bureau.

On 8 November 1931, they decided to meet again. During this meeting, an extra person was attending. Next to Schmidt and Lemoine, coding expert Bertrand also attended the meeting. Schmidt brought the manuals that were used by the Germans, which explained how to use the Enigma machine. These manuals were top secret. To not make it obvious that the documents were missing from the office, they were photographed by Bertrand. After this meeting, Schmidt became a spy for the Deuxième Bureau.

After looking into the documents that Schmidt provided, the French and British concluded that the documents did not make it possible to break Enigma. To try and break the cipher, Bertrand wanted to share the documents with the Poles. Long before the first meeting between Schmidt and the French Secret Service, the Poles had mentioned that they were unable to read the code used by the German Army.

6.2 A mistake made by the Germans — January 1929

However, about two years before Bertrand went to Warsaw with the documents, on the last Sunday of January 1929, a box had been sent to Poland by the Germans. The German embassy wanted it back as soon as possible, since it had been sent to Poland by accident. However, the Poles decided to open the box and found a commercial Enigma machine. The Polish General Staff's Cipher Bureau called two engineers to examine it. Danilewicz and Palluth took the Saturday night and Sunday to do so. On Monday, the Enigma machine was returned to Germany. As far as known, no one ever realized that it had been opened. This finding probably led the Poles to attempt to break Enigma. However, the Poles were not successful in breaking the cipher either. What the Poles did not know, was that the internal wiring of the rotors of the commercial Enigma were not the same as the internal wiring of the rotors for the Enigma machine used by the German Army. This was the reason that the code could not be broken at that moment.

In January 1929, Ciężki, head of the German section of the Cipher Bureau, set up a cryptology course and interviewed students to follow the course. Rejewski, Zygalski and Różycki were employed by the Cipher Bureau. By December 1931, when Bertrand came to Warsaw with the photographs, students were already working on simple ciphers at the Poznań army command post.

6.3 A new attempt — December 1931 - December 1932

Major Gwido Langer, Head of the Cipher Bureau Warsaw, was handed the photographs by Bertrand. This was of value to the Poles, as it revealed that the German Army had adapted the commercial Enigma (which they already inspected when the Germans accidentally sent it to them). However, it was not enough to read Enigma messages. They could not even be read, when setting instructions were sent to Warsaw. Three lists of Enigma settings were handed over, and yet the Poles were unable to reconstruct an Enigma machine.

In 1938, the Deuxième bureau staff concluded that the documents that Schmidt had provided would not lead to breaking Enigma, so they came up with a plan. They wanted to make the Germans believe that they had broken Enigma. They hoped that the Germans might be tempted to use a different cipher that was easier to break. When the plan was told to Langer, he persuaded Bertrand not to go on with the plan. He said that the Poles would soon be able to share more about their research. However, in fact, the Poles had already broken the Enigma and had already been reading messages for five years, that is since December 1932.

They managed to do so, after Rejewski had been asked by Ciężki to put in extra hours in secret. He worked on the Enigma cipher and was given the documents from Schmidt. Rejewski first had to construct a replica of the Enigma machine. He did this through formulas that could work out the internal wiring, but he needed the settings used to do so. To his luck, the settings of October and September 1932 were delivered by Schmidt. However, the formulas developed to determine the internal wiring were not producing the correct answer. This was because Rejewski assumed that the wiring within the entry disc was random, as in the commercial Enigma. After he found out that this was not the case, he figured out the wiring of two rotors. After having determined these settings, he could figure out the wiring of the third wheel and reflector. Beside his own efforts, Rejewski also had help from the Germans themselves. They included an example in the Enigma manuals delivered by Schmidt. This example consisted of a sample message and the enciphering of the message, using given settings that were also included. Which formulas were used is explained in Appendix C.

6.4 Unexpected help — 1933 - 1937

After the Poles found out how the Enigma machine worked, replicas of the machine needed to be made. When these were ready, Różycki, Zygalski, and Rejewski tried to read the messages. They were, unexpectedly, helped by the Germans themselves. The **message setting**, which are the letters showing through the rotor windows, had to be enciphered two consecutive times and then sent at the beginning of each message. However, this turned out to be a huge mistake of the Germans. The sending procedure went as follows:

German sending procedure:

1. Set the machine in the correct settings according to the setting list of that day.
2. The person sending, chooses a message setting. Say that this is IMD.
3. Turn the wheels into the **initial position**, also known as the Grundstellung, which was indicated on the setting list. Say for a particular day it is ZIF.
4. Encipher the message setting twice, say the letters AEG LXO light up. These six letters are known as the **indicator**.
5. Put the indicator at the beginning of the message.
6. Set the wheels to the message setting, so IMD.
7. Now you can encipher the message.

The fact that the message setting had to be enciphered twice, enabled Różycki, Zygalski, and Rejewski to find new patterns, which in their turn enabled them to break Enigma. The patterns could be seen as fingerprints. A certain fingerprint could only be produced if the Enigma machine was set in a particular way. All the codebreakers had to do, was to produce a list of all possible fingerprints together with wheel positions which could produce that particular fingerprint. This method was called the **Characteristics method** and is further explained and analyzed in Appendix D.1.

6.5 A new procedure and decoding tools — 1938

These patterns also helped with creating new methods to break the Enigma cipher, when the operating procedure of Enigma itself was changed. On 15 September 1938, there was a crucial change in the enciphering procedure. Now, senders had to choose their own initial position, which they had to send unenciphered before double encipherment of the message setting. Within weeks, there were two new tools to decode messages: the Bomby (plural for Bomba) and the Perforated sheets. Both worked because the double encipherment of the message setting was still part of the sending procedure. The Bomby are further explained in Appendix D.2.

After coming up with the concept of the Bomby, six Bomby were made available for the cryptographers. In November 1938, Enigma messages were broken. It took less than two hours to find the settings. However, the Bomby only worked when not too many plugs were used. This was still the case in the autumn of 1938 when only five up to eight cables were used. After a while, the Germans increased the number of cables. The Bomby became redundant at that moment and had to be adapted.

The second tool created, the so-called perforated sheets, was designed by Zygalski. He exploited the fact that there was a limited number of indicators with special characteristics. To be more specific, indicators that had

the first and fourth, the second and fifth or the third and sixth letters in common. These were referred to as **females**. Females could not occur for a substantial number of initial settings. If a female indicator was spotted, certain settings could be ruled out. The more females were found, the more initial settings one could eliminate. They found females until enough settings were eliminated and only a few possible settings were left. These remaining were tested manually.

To perform this procedure, sheets were used, which functioned as a catalogue. If a female indicator was possible at a certain initial setting, a hole was punched. The relevant sheets regarding a certain day's female indicators, were stacked on top of each other and placed on a table that lit up. Any place light came through represented a wheel position, ring settings and wheel order that could have produced the female indicators. These settings had to be tested manually. The plugboard settings were ignored by the Poles who were making the sheets. If a female indicator was produced without the plugboard cables in a certain setting, it could also be produced with the plugboard cables set. The only difference would be that the female letters were different.

However, the production of the sheets was slow. Then, on 15 December 1938, the Germans added two rotors. The amount of possible wheel orders went from six to sixty. This made the production of the sheets impossible on a short term and there was no time to quickly adjust the Bomby.

6.6 Additional rotors and cables — December 1938

Not even a month later, at the beginning of 1939, the Germans changed their operating procedure again. Now 7 up until 10 cables had to be used on the plugboard. This made the Bomby less effective. However, the wiring of the two new wheels fortunately were figured out quickly. Nevertheless, messages could only be read when the new rotors were not used.

6.7 The Poles share information with the English and French — 1939

On 24/25 July 1939, Dilly Knox, a senior British cryptographer, Alastair Denniston, head of Britain's Government Code and Cipher School, Gustave Bertrand and Henri Braquenié, who were French cipher experts, were told that the Poles had broken the Enigma cipher. The British and the French both received a replica of the Enigma machine and were explained how to produce sheets.

But the general Enigma cipher was not the only one used. A twenty-year-old undergraduate from St. John's College Cambridge, named Harry Hinsley, was the chosen one to solve one of the most important keys used during the war: the Enigma cipher used by the German Navy. To help break the Enigma cipher, and to get students to join the school's new office at Bletchley Park, (under)graduates were approached by Denniston to join. Hinsley was invited to an interview at the Foreign Office but was not directly told what his future job consisted of. He was offered a civilian post at the Foreign Office during the interview. Hinsley saw this as an opportunity and accepted the offer. Only later, he was told by a tutor what his work really was going to consist of. Hinsley worked in the Naval intelligence hut, also referred to as Hut 4. He was asked to help and discover as much on the messages as possible. To do so, he had access to the following information about the messages: the date, the time of origin, the time of interception, the radio frequency used, and sometimes they knew where the message came from.

Although the British cryptographers were struggling to break the Enigma, there was a breakthrough. Peter Twinn, a twenty-three-year-old Oxford graduate, was given an enciphered Enigma message with matching German text and the corresponding settings upon his arrival at Broadway, the location of the Government Code and Cipher School before it moved to Bletchley Park, in 1939. An obstacle that was holding up Twinn, who worked together with Knox, was that he could not manage to reconstruct the Enigma machine. However, this changed in July 1939. In Poland, the Poles shared their information on the Enigma machine, which included the wiring of multiple parts of the machine. Knox was a part of this meeting, and when he arrived home, he shared this information with Twinn. Since Twinn had an enciphered message, together with the matching German text and the corresponding settings, a few messages could be read. However, the two were not yet able to work out new settings, which meant that no recent messages could be read. In fact, no one was able to until the set of perforated sheets was complete and produced.

In the middle of December 1939, the first set of perforated sheets was ready. However, there was a problem. The Poles made a mistake when they passed on the information to the British codebreakers when it came to

the turnover positions of the two new wheels.

In the beginning of 1940, Polish codebreakers were not allowed to travel to London, so their British colleagues had to visit them in France whenever they wanted to meet. Alan Turing, a British codebreaker did, and he visited the Poles to discuss the Enigma machine. However, at the time of the visit, Turing had already invented something to break the cipher. He invented an electro-mechanical machine, referred to as the Bombe, which could determine the wheel setting and plugboard settings used to encipher a message.

6.8 A remastered invention — 1940

The British Bombe consisted of multiple Enigma machines that were wired together. The machine was able to go through each wheel setting and test whether it could be rejected or not. It differed from the Polish Bomba, as it was not based on a twice enciphered initial message setting. For creating the Bombe, Turing tried to find a systematic way to conclude whether a certain setting was possible or not. The few that were not rejected, could be tested manually. A more detailed description of the Bombe is given in Appendix E.

However, the Bombe needed a crib to make it function. A **crib** is a piece of text that was guessed to match a piece of cipher text. This created pairs of letters. The Bombe was then able to test for which settings it would be possible or impossible to create these pairs of letters. Since the British cryptographers were reading the messages since January 1940, due to the success of the perforated sheets, coming up with cribs did not take much effort.

On 18 March 1940, the first Bombe was installed at Bletchley Park. However, the results were disappointing. It failed to do its job. Then Gordon Welchman came up with a suggestion that would make Turing's prototype at least twice as powerful. Also, Turing himself had another improvement. The Bombe was updated and became known under a new name: the Spider. The Spider was used to break Air Force Enigma.

Richard Pendered was asked to check whether the Spider worked. He started in the summer of 1940. The Spider was checking for possible solutions. When it stopped, the corresponding settings had to be tested manually on a replica Enigma machine. They knew the settings were correct, when German text started to appear on paper. After days of trying and failing, the Spider worked. The first one was installed at Bletchley Park on August 8, 1940.

6.9 Naval Enigma breakthrough — 1940

In February 1940, there was the first breakthrough when it came to making an attempt to read Naval Enigma. However, this was not due to the cryptographers at Bletchley Park. It had something to do with the U-boat U-33. It carried information about the ciphers on board. U-33 was on the surface when a ship passed it. Although they thought they had not been noticed, a crew member of the British HMS Gleaner thought he heard something that sounded like a diesel engine. They decided to investigate it and started to approach the U-boat. In a final attempt to escape, the U-boat started a rapidly decline. However, while doing this, the depth charges started exploding. This made the U-boat dive as deep as possible and hit the see bed. This caused a leak in the U-boat, and water started coming in. Therefore, the crew decided to blow up the tanks which made the U-boat starting to float again. When they finally reached the surface, everyone was ordered to abandon the U-boat, after which they blew up the U-boat from the inside. However, when it came to Enigma material on board, not everything was destroyed by the explosion. A couple of crew members had Enigma wheels in their pockets, which they needed to throw into the sea as soon as they left the U-boat. One crew member failed to do so and, once the crew was rescued, the British found the wheels in his pockets and took them. In total, there were three wheels of which two were only used in the Naval Enigma machine.

These three wheels gave the British cryptographers, including Alan Turing, an opportunity to work on the Naval Enigma machine. This machine had the feature to choose three out of eight wheels, instead of choosing three out of five wheels, which was the case in the Air Force and Army Enigma. After 1 May 1937, the Naval Enigma stopped with this procedure. From then on, the operators could no longer choose their own message setting. Now, they had to choose a three-letter group from a book, say for example IOP. They then needed to encipher this on the Enigma machine that was put into the Grundstellung, which is also known as the initial setting, of the day. Say that the initial setting would be ERT and say the letters lighting up on the machine were MLD. Then MLD would be the message setting for the message. So, the machine had to be set in the message setting (MLD) before starting to encipher the message. This was safer since people would not use

the same or predictable message setting every day. A more detailed description of Naval Enigma is given in Appendix F.

However, the sender had to communicate the settings to the receiver without it being picked up by the enemy. The Germans came up with the following procedure. Other than picking the three-letter group (also referred to as a **trigram**) IOP out of the book, another trigram also had to be selected by the operator, say QAZ. These had to be written above each other. The first trigram had a random letter that was added in the front, and the second had a random letter added in the back. It would for example look like this:

C Q A Z

I O P B

The next step was to pair letters from the same column, which were referred to as **bigrams**. These bigrams were transformed through something called a bigram table. In a bigram table, every possible pair of letters was matched to a replacement of another pair of letters. So maybe CI corresponded with UX in the bigram table, and therefore replaced by UX. The same was done for all four bigrams. One could get something like this:

U N W V

X F S Y

Then they would systematically change the position of the letters as follows:

U X N F

W S V Y

This was put in the beginning of the message without being enciphered. To find the settings, the receiver would do the same steps but in reversed order:

C Q A Z

I O P B

Then the receiver would look at the first three letters of the second line. He would put his machine in the initial setting that was stated the codebook, in this case ERT, and would then type out IOP. He would then get the letters MLD, he puts his wheels in this order, and started to decipher the message.

6.10 Mistakes — 1939 - 1940

Alan Turing thought he figured out the system by the end of 1939. It was based on message settings and indicators of only seven messages that were given to the British cryptographers by the Poles in July 1939. The Poles were already able to read some of the messages of May 1937, after the indicating system had been changed. They were able to do so, because the Germans made two mistakes.

The first mistake the Germans made was that they did not change the wheel orders and ring settings. The second mistake was that one of the torpedo boats was still allowed to use the old system, which was already broken by the codebreakers.

However, although Turing did understand the system, he could not manage to break the messages. He needed the bigram tables the Germans were using to do so. The only way to obtain these, was to capture them. On 26 April 1940, at the Norwegian coast, Enigma documents were captured from a German trawler. This ship was sailing under the Dutch flag and pretended to be a fisher boat. The crew from the British destroyer boarded the boat and hoped to find the Enigma documents, but the Germans managed to throw two bags with the most important codebooks overboard. However, one of the bags did not have enough weight to sink and kept floating. The British got the bag out of the ocean and the documents turned out to be invaluable. Because of this action, the Naval Enigma cipher was broken between 22 and 27 April 1940. The first Naval Enigma message was read on 11 May 1940. As can be noticed by looking at the dates, there was a delay in decrypting the messages. This was the case because the plugboard settings for 23 and 24 April were overlooked in the beginning. When they were found and the codes for these days were broken, their paired days (22 April /25 April) were broken as well since wheel orders and ring settings were only changed every two days. However, the delay meant that the messages were not up to date anymore. But breaking the messages was helpful for the

cryptographers. For example, it made Alan Turing realize how the indicator system worked.

This gave the British the opportunity to find a way to break recent Naval Enigma messages. After guessing the way the indicator system worked, Turing invented a procedure called **Banburismus**. This method made use of one of the weaknesses of the Naval Enigma system. This was that the trigrams were all typed out in the same wheel position, namely the initial position. Banburismus was meant to eliminate many possible wheel orders. After performing the procedure, only a manageable number of wheel orders should remain. These could be tested on the Bombe.

However, the bigram tables were needed to use Banburismus, and these were not available to the codebreakers. Fortunately, there was a way to figure these out without them having to be captured. After an Enigma setting for a message was worked out, open spaces in the bigram tables could be worked out. The message settings were worked out by using an so-called **Eins catalogue** which is explained further in Appendix F. There were nine bigram tables. To reconstruct even one, one needed around three days of message settings. The codebreakers almost completed the message settings of April 1940 and because of this they were able to try to use Banburismus for the first time. However, it failed the first time.

6.11 New bigram tables — July 1940

After the cryptographers kept trying, in November 1940 Banburismus finally helped to break Naval Enigma settings. But they were very unlucky: the Germans introduced a new set of bigram tables on July 1. So, for all messages after July 1, new bigram tables had to be constructed. They basically had to start all over. The only difference was that they knew they could break messages, if codebooks could be captured.

In the beginning of May 1940, another change had occurred. The Germans stopped enciphering their message setting twice when it came to the Army and Air Force Enigma cipher. The change the Germans made, removed a weak spot in the encipherment. The double enciphering of the message setting was the reason the British codebreakers were able to read many messages that were sent over the last seven years. So, this was another major setback.

Rees, a math pupil, and Herivel, a mathematician, were ordered to test possible solutions that remained after using the perforated sheets. After that, they had to figure out the plugboard settings. This process took exceptionally long, and Herivel felt that there had to be a quicker way. In fact, there was. The idea he came up with, was revolutionary for breaking Enigma.

6.12 Techniques and mistakes

Herivel thought of the procedure that took place even before enciphering a text. Before using an Enigma machine, the wheels and ring settings had to be set in the settings of that day. Herivel imagined that an operator would first put the wheels in the Enigma machine, and after that would set the wheels in the correct ring settings. So, for each wheel, someone would do the following. Say the ring setting for example would be ERT: for the left wheel, the ring setting, in this case E, had to be lined up with the red dot on the catch. The catch served as a marker but was mainly meant to fix the ring relative to the core position. Herivel thought that fixing the settings would be done easiest when the catch and the ring were near the top. If the operator was lazy, he would not change the wheels anymore. Then they would use the letters showing through the windows as the initial position for the first message. Then, the initial position, which was given unenciphered at the beginning of a message, would be the same or nearly the same as the ring settings. If this was the case for all the first messages from each operator each day, the initial settings were expected to be quite similar. If many senders would in fact do this, then the British cryptographers could graph them and look for clusters. This did not work immediately, but it was one of the only hopes the cryptographers had after the indicator system was changed. Eventually, one day the method started to work. The technique became known as **the Herivel Tip**.

However, the Herivel Tip was not enough to break Enigma quickly. Fortunately, the Germans made other mistakes as well. They used message settings that were easy or obvious, which are known as **cillis**. Cillis are named after the first obvious message setting found by the British. They helped to determine the wheel order of the Enigma machine. Cillis are further explained in Appendix G.

May 22 1940 was the first time since the Germans changed their indicator systems, that the British were able to read an Air Force Enigma message. The message read was a message from 20 May. The Enigma network that

they broke, was referred to as 'Red'. Breaking into Red was particularly important. The codebreakers wanted to use the Bombe to break the Enigma messages. To do so, they needed cribs to use the Bombe. However, cribs were usually found when they were able to read Enigma messages. The cillis and Herivel Tip made sure that Air Force Enigma messages could be read during the installation of the Bombes and contributed to being able to read the messages after the machines were installed. Therefore, these two factors contributed to breaking Red Enigma. The messages remained broken from 22 May 1940 until the end of the war.

It was important to break Naval Enigma code. Only then the cryptographers would know the location of all the U-boats in the Atlantic. They then could redirect the convoys to prevent them to be attacked by wolf packs, which were groups of U-boats that worked together.

In August 1940, the Spider Bombe was invented by Alan Turing. It was used to break Air Force Enigma, and the codebreakers hoped it could also break Naval Enigma. Another concept that might break Naval Enigma, was to use something called the **Wild Cat Scheme**. The cryptographers would send out something random in Morse code on the frequency which the U-boats were using. By sending these messages, they tried to provoke a reply from the Germans. If this reply would be predictable, they would have a possible crib.

On 12 March 1941, Enigma documents for Home Waters network were handed to Turing and Peter Twinn. It contained the inner and outer settings, and the plugboard settings. When the documents reached the British cryptographers, they were overdue, as they were for February, and it was already March. However, the documents were extremely helpful. On the same day the cryptographers received the documents, they managed to read a message from February. This helped to reconstruct the bigram tables. Thus, they could take another go when it came to the Banburismus procedure.

So, by the end of March, also due to delays, the bigram tables were almost complete. Now the cryptographers had another chance to try out the Banburismus procedure. But it did not work immediately. This was partially due to dummy messages consisting of strings of consonants. This resulted in no March messages being broken and only eight days of April, but these were only read by 10 May.

6.13 Breaking the code — 1941

After the Krebs', which was a German trawler, codebooks were captured in March 1941, Enigma messages were read. They were given to the codebreakers at the Naval Intelligence at Bletchley Park. The cryptographers also investigated the message traffic and thought it did not reveal anything at all. But the Germans were about to invade Norway, which caused there to be more wireless activity than before. Hinsley also found that the German Navy had two radio networks: one for communication with crew in the Baltic and one for communicating with crew outside the Baltic. In addition, he noticed that messages to the people at the Baltic were also repeated on other frequencies. This indicated a shift in areas, in particular a shift from the Baltic Sea to Skagerrak.

The Enigma messages were broken too slowly to have an impact on the battles fought at sea, but the decrypts did show something else. There were trawlers that had to go to isolated spots north of Iceland to see what the weather looked like. These ships had to have a Naval Enigma machine and codebooks onboard. These were not used to report the weather; however, they did receive messages confirming that the weather reports came through. So, the British decided to go after these trawlers. One day, when they approached the ship called München, they started to fire at them hoping the German crew would panic and leave the boat without destroying the codebooks. The Germans did as they were told and threw the machine and current codebooks overboard in a weighted bag. However, from the captain's cabin, containing the safe with the codebooks the Germans were not currently using, they were able to get a couple of codebooks, which turned out to be important. The documents arrived at Bletchley Park three days later. They contained information the British cryptographers needed. The documents contained the inner settings for June 1941, and two sheets of outer settings. This meant that the messages sent in June could be broken.

Around 9 May, two warships had a U-boat in sight, namely U-110. The crew of the warships decided to start firing their guns at the U-boat. This had the wished effect as the crew of the U-boat started to leap overboard as the U-boat was sinking. Since the German crew was panicking and leapt overboard as quickly as possible, they did not get rid of the Enigma machine and codebooks. However, unlike what the German crew was thinking, the U-boat was in fact not sinking at all.

After the German crew leapt overboard, the British crew had to board the U-boat to capture the codebooks and other useful documents. Beside the documents, they also found an Enigma machine. This operation be-

came known as Operation Primrose and probably was one of the most important events since some extremely important documents were captured. The Naval Enigma settings for June 1941 were most important because it made reading the current U-boat messages possible for the first time. This was an important improvement since before this capture, this took approximately ten days. After capturing the June settings, this was reduced to around six hours. The documents captured from the U-110 also were important for being able to read Offizier messages, which was important for being able to read the Naval Enigma messages.

However, the Germans got suspicious when none of the prisoners could tell them, whether the U-boat had sunk. Fortunately, the codes used by the German prisoners was already broken, so the warning of the Germans about the fact that there was a chance that the British captured Enigma documents, was probably intercepted by the British. However, after a while, prisoners stated they did see the U-boat sink. The secret of Enigma therefore was not compromised.

6.14 The introduction of new bigram tables — June 1941

Nevertheless, the capture of the codebooks from U-110 and München did not break the Naval Enigma entirely. On 15 June 1941, the Germans started to use a new set of bigram tables. The cryptographers could still read the messages of June, as they still had the documents from München, but this was not the case for the messages after June. Therefore, they needed a month of settings to construct the new bigram tables. To do this, there were two options. The British either had to capture the settings for July or capture the bigram tables.

The idea was to capture another weathership, but the British feared this might raise suspicion with the Germans. If they got suspicious, the Germans might change their enciphering procedure. This would be risky as Air Force Enigma would also be in danger. The only hope was that, with the cribs of June, the new bigram tables could be reconstructed. However, they decided to risk it, and went after Lauenburg. Their strategy was the same as with prior captured. They tried to scare the crew to make sure they would abandon the ship. It worked, because after firing at the boat, the crew immediately left the boat. However, the British did not find codebooks as they were probably burnt or thrown overboard. However, they found charts, signaling papers, and three sheets of paper. These turned out to be the plugboard settings and inner settings, so the wheel order and settings. So, throughout July, the British cryptographers could read the Naval Enigma messages within hours again.

6.15 Finally broken — August 1941

For August and September, the plan was to also capture documents, but this was not successful. However, after the first week of August 1941, the codebreakers did manage to read all the Naval Enigma messages until the end of the war, except for two days. After July, it took around fifty hours to read the Naval Enigma messages.

To read the messages, two strategies could be used. The strategy that cryptographers preferred was using a combination of Banburismus and the Bombe. Banburismus ruled out wheel orders, and then, based on a menu derived from a crib or information that was gathered during the Banburismus procedure, these could be tested.

There was also the possibility that Banburismus would not work. In that case, the cryptographers could use a crib in combination with the Bombe. Then the menu that was produced by a crib was run on all possible wheel orders. This took longer, but could have been worse since the Germans made a mistake again. They created rules when it came to the wheel order. The first rule was that wheels 6, 7 or 8 always had to be one of the wheels installed in the Enigma machine, which already eliminated sixty possible wheel settings. The second rule stated that a wheel could not stay at the same place in the Enigma machine on two consecutive days. These rules reduced the amount of wheel orders that needed to be tested to 105, which would take the Bombe 35 hours.

The cribs cryptographers used to try to break Naval Enigma originated from messages reporting the weather. These were always on the same time and on the same frequency. Besides, they also included a particular sign at the beginning of the message which usually existed out of three letters. If stereotypical words were used in the beginning of the message, it was easy to come up with a matching crib. However, the Germans sometimes added dummy letters to confuse the people that were trying to break the code. In that case, the cribster had to find out how to line up the crib with the cipher text.

To do this, the cryptographers made use of a weakness of Enigma again. A letter could never be enciphered as itself, so if a letter in the crib and the enciphered message lined up above each other, it would be incorrect. For

example:

Crib: G E **R** M A N

Cipher text: K G **R** X S T H I M

Two letters lining up was called a **crash**. To prevent this, the crib would be slid to the right until there were no crashes. For the example this would be two letters to the right:

Crib: G E R M A N

Cipher text: K G R X S T H I M

So, this would likely be the correct position. If they found the correct position, cryptographers would produce the menu, which would make it possible for the Bombe to give the wheel settings and the plugboard settings.

There also was a variant of Naval Enigma, which was only used by officers. The messages would be enciphered twice. These kinds of messages were broken only once, when the settings were captured from U-110 in June 1941. However, in July 1941, new settings were introduced, and the British cryptographers had to start all over again. To break what was called Offizier Naval Enigma, they needed a crib and then use the Bombe. However, the cryptographers feared that the messages could not be broken regularly, since it was unlikely that there were stereotypical messages to work out a crib with. But at the end, they managed to find a crib and break the Offizier Naval Enigma.

1 June 1941 was a turning point. The codebreakers could break Naval Enigma almost as fast as the Germans could read it. However, they had to pay attention to the fact that they could not sink every ship they knew the whereabouts of. If they would do so, this probably would raise suspicions with the Germans.

When capturing U-boat U-570, the British found documents. They contained plain German texts with their matching enciphered texts and a couple of settings. Germany was worried that their code might be compromised, although they were still very much convinced of the safety oftheir system. This was also the case, because they had a procedure which they referred to as **the keyword procedure**. They thought that this procedure made it almost impossible for the enemy to read their messages.

The keyword procedure worked as follows: say the key is INDIGO.

The first letter, in this case I, the ninth letter of the alphabet, corresponded to the addition to the wheel order. Say the initial wheel order was 1/8/5, then nine was added every setting. Since there were only eight rotors available, we would get the wheel order 2/1/6 $((1 \to 2 \to 3 \to 4 \to 5 \to 6 \to 7 \to 8 \to 1 \to 2)/(8 \to 1 \to 2 \to 3 \to 4 \to 5 \to 6 \to 7 \to 8 \to 1)/(5 \to 6 \to 7 \to 8 \to 1 \to 2 \to 3 \to 4 \to 5 \to 6))$.

The second, third, and fourth letter, in this example N, D, and I, indicated the addition to the ring settings. So, the corresponding numbers to the letters, in this case 14, 4 and 9, were added to the given ring settings. So ZUS for instance became NYB.

The fifth letter, in this case G, was for the addition the plugboard settings. So, plugboard setting K-L became R-S.

This trick would only have worked if the British cryptographers would have used trial and error. However, when a correct crib was found and the Bombe was used, this was not a problem at all.

The Germans sometimes were remarkably close to finding out that their Naval Enigma was compromised. However, Maertens, the head of the German Navy's Communication Service, could not believe this.

The sinking of two ships raised suspicion in the Germans. Admiral Kurt Fick thought that it must be treason or that the cipher was not secret enough. However, no one was to blame, since everyone followed the rules while enciphering the messages. In Fick's words he cipher was superior compared to other countries. Besides, Fick also indicated that there was no sign in the messages of the British that the Germans could read, since they had enciphered the British cipher.

6.16 Other forms of encipherment

Smaller ships and U-boats without Enigma machines and small German ships sailing along the Norwegian coasts used another form of encipherment called **Reservehandfahren (RHV)**. The British cryptographers also tried to break this cipher, since this cipher was particularly important for Naval Enigma.

RHV was broken in June 1941 for the first time due to the codebooks that were captured from U-110. However, this was not the only thing that was the cause of the break. The Germans made another error. If you want to encipher following RHV, the plain text had to be fitted in a cage, which is a shape drawn on paper. This shape changed every month. A cage made sure every letter of the unenciphered text would become part of a vertical bigram after which these bigrams were replaced by other bigrams based on bigram tables for RHV. The bigram tables and shape of the cage were changed occasionally, but not monthly. The changes were staggered so one of the elements was always known to the cryptographers. This helped to regularly break the cipher. The content of the messages was important, since messages sent with Naval Enigma were often repeated over RHV. From these texts, the cryptographers could get cribs to help breaking Naval Enigma.

The same held for Werftschlüssel, which was broken in 1940 and the broken messages were used as cribs.

6.17 Another flaw in the system — 1941

The Germans tried to be as secure as possible, also when it came to the bigram tables. There were nine sets of bigram tables in total, and each day one set was chosen. The chosen set of tables was communicated to the crews by lists which indicated which set to use. The codebreakers at Bletchley did not have these lists, so they had to figure everything out themselves.

In May 1941, the bigram tables and Kenngruppenbuch were captured from U-110. Joan Clarke, one of the cryptographers working on the Naval Enigma, and the rest of the Hut 8 crew found out something useful. The Germans were supposed to make up random message indicators. However, it turned out that they were not. The Germans tended to choose the trigrams in the Kenngruppenbuch that were at the top or near the top of the columns. There was also preference for certain columns. There were 733 columns, which obviously did not fit on one page. The telegraphists preferred to choose from the column in the middle of the book. So, if the cryptographers at Bletchley looked at the different bigram tables, they looked at which table transformed the indicator letters to one of the favorite trigrams. This set of tables was usually the bigram table that was used. The Germans must have realized that their telegraphists were not as random and therefore enforced a new rule. One could only use one trigram once ever. The trick found by Joan Clarke could not be used anymore, but the German system had flaws that were spotted by new recruits.

Jack Good, a mathematics scholar from Cambridge, found a flaw in the German system. The Germans had to add a dummy letter to the trigrams, and he wondered whether this choice was random, or whether there was a certain preference towards certain letters. From broken messages, he found out that certain letters were used more frequently than others. If this was the case, the same trick as Clarke had invented, could be used. They applied all the bigram tables and found out which letters were popular dummy letters. Later, this became an important part of the Banburismus procedure.

On 29 November 1941, the Germans replaced the bigram tables. This meant that there was not enough information to carry out the Banburismus procedure. However, the cryptographers were still able to use cribs and the Bombe. Banburismus took around 300 messages to work, where the Bombe only took one if the crib matched. But Bombes took a long time. Luckily, the cryptographers at Bletchley had access to 15 Bombes, so this did not take this long in practice.

The staff of Naval Enigma Hut 8 also wanted to recreate the bigram tables again. To do so, they wanted to have access to codebooks. Therefore, the British captured another trawler, called Föhn, and found Enigma settings, a set of bigram tables, five Enigma wheels and an Enigma machine. They could start breaking the codes again. Due to the capture of these documents, the Naval Enigma used by surface ships and U-boats in the Arctic was broken every day for the rest of the war. This saved countless of lives.

The inner settings of the Naval Enigma machine took at least 48 hours to break. After they were broken, the messages from two days could be read quickly, as the inner settings were only changed every two days. If the inner settings were known, the rest of the settings were deduced quickly. To find these inner settings, Banburismus was used. This reduced the amount of wheel orders that had to be tested on the Bombe.

6.18 The fourth wheel — February 1942

On 1 February 1942, the Germans introduced a new Enigma wheel for the Enigma machines used on the U-boats in the Arctic. The Germans called it Triton, while the British called it Shark. This fourth wheel was placed left of the other three wheels in the machine and could be set into 26 positions. Unlike the other three wheels, this wheel could not turn and was not interchangeable with the other wheels. This was advantageous for the British codebreakers as there were only 26 times as many options to check. If it would have been interchangeable, there would be 3,024 ($8 \times 7 \times 6 \times 5$) options to check. However, breaking the codes would take a lot more time. If the cryptographers wanted to work out the settings in 24 hours, they would have needed 85 Bombes, Bletchley only had 12 at the end of 1941 and the Naval Enigma department had to share these with the Army and Air Force departments.

The wiring of the fourth wheel was found out easily. The wheel was already part of the Enigma machine since December 1941, but they put it in position A with ringsetting Z. So, in this way the machine functioned as a regular three-wheel Enigma machine, but the Germans made mistakes again. One German telegraphist put the ring in setting B by accident. After that, he communicated to the receiver of the message that he put the ringsetting of the fourth wheel in position B by accident. This gave the British cryptographers the opportunity to find out the wiring of the extra wheel.

The cribs used for breaking messages was found with the help of weather reports and weather codebooks. However, this codebook was replaced by a newer version on 20 January 1942. The cryptographers found cribs here and there, but it was impossible to use Banburismus because this only worked if there were pairs of messages which had the same settings. Since the fourth wheel of the Enigma machine could be placed in so many ways, this would hardly be the case, so there were not enough pairs to perform Banburismus.

The British worried over the fact that the Germans might get suspicious because suddenly, after they changed their systems, the Allies got less successful. However, Naval Cipher Number 3, which was a cipher used by the British, American and Canadian Navies in the Atlantic communication, was broken by the Germans in February 1942. Many more U-boats were successful in sinking Allied ships, so the Germans thought this was the case because they had broken the code. In this way, the Germans did not get suspicious when their success grew right after changing their Enigma cipher machine.

In the meantime, the Americans were trying to make a Bombe that could break the Naval Enigma messages that were enciphered on a four-wheel Enigma machine. In May 1943, the first two prototypes were ready. To break Enigma, documents from U-559 were recovered. These were important documents, as they were an updated version of the weather codebook, which were used to create cribs. However, the documents did not result in breaking the Enigma messages. It was only useful for cribs, which was not enough for four-wheel Enigma messages unless there would be a substantial number of three-wheel Bombes or if there would be a new four-wheel Bombe model.

On 13 December 1942, there was a breakthrough when it came to the four-wheel Enigma cipher. It was broken because of another mistake made by the Germans. When the British cryptographers finally managed to break a four-wheel Enigma message, they found out that for short messages the Germans would still use the three-wheel Enigma machine. This meant that the cribs from the weather reports could indeed be used to break the codes, and that this could be done by three-wheel Bombes. For other longer messages, the setting of the fourth wheel was quite easy to work out if the other settings were already known.

On 17 February 1943, documents from U-205 were captured. The British crew found books, including Naval Enigma bigram tables and the U-boat's short signal book. However, both had already been in British hands because of earlier captures.

6.19 Delays in the process

On 10 March 1943, there was an Enigma black out. The Germans introduced a new short weather report codebook, so the codebreakers were not able to get cribs out of the weather code books, which was the technique they had been using since mid-December 1942 to break Shark. On 19 March, the British codebreakers managed to break into Shark again by using the short signal code books. They created a new method for coming up with cribs which could be used in combination with the Bombes again. This procedure went as follows: say a short message was intercepted. The cryptographers would investigate the information available, so the position,

course, speed and messages from the previous day to guess what the commander was trying to say. They could translate this information into the short code, which would then function as a crib. The first half of 1943, the Bombe took about 24-72 hours to break the code.

There was also a little setback when it came to breaking Dolphin, also called Home Waters Naval Enigma. On 1 March 1943, the Germans changed the bigram tables again which made the Banburismus procedure impossible to use. However, by this time, Bletchley had access to 70 Bombes, and this was enough to use the crib and Bombe method. After three weeks the new bigram tables were constructed again.

But there was still a delay in breaking messages, in particular the Offizier messages were hard to break since the entire messages were enciphered twice. The settings for this procedure were as follows: the wheel order and ring settings were the same as the regular settings but the plugboard settings were different. These settings were chosen from a certain list that would change every month. There were 26 settings on the list and each of them had a letter of the alphabet corresponding to it. The Germans communicated these settings to the receiver by sending them a name of which the first letter was the same as the letters corresponding to the chosen settings. So, say that, for example, the settings corresponded to the letter P were chosen. Then the message would be enciphered in the settings corresponding to P and for instance "Offizier Paula", which was not enciphered, was put in front of the message. Then this enciphered message was enciphered again, but then in the 'normal' settings of that day.

But the Offizier settings were not broken regularly yet. The capture from U-110 did show the British codebreakers the system the Germans used, but this was not enough to break the cipher. Besides, the settings of June, which were captured, were running out as June passed. As mentioned before, cribs could be used to break Offizier Enigma with the Bombe, but the alignment of the cribs was exceedingly difficult. Although the wheel order was known, it took a lot of time on the Bombe. This was especially a problem in 1941 and1942, when the codebreakers did not have access to many Bombes. So, they had to find a faster way.

6.20 The influence of having information

The French Deuxième Bureau had a profound influence on breaking Enigma. They basically helped to let the Allies master breaking Enigma for at least the first two years of war. Towards the end of the war, it became known that the British had been reading the German messages. Nevertheless, the German Naval Communications Department was still in denial and said this could not be the case.

When it came to reading the Shark messages, Dolphin had to be broken first, as these would deliver cribs for Shark to be broken. So, it was important Dolphin was broken as quickly as possible. At the end of June 1943, the codebreakers decided not to use the long Banburismus procedure anymore, but solely work with available cribs and Bombes. But there was another setback. On 1 July 1943, a new fourth wheel, Gamma, and a new reflector, Caesar, were introduced by the Germans for Shark. These could replace the original fourth wheel, Beta, and reflector, Bruno. The messages could not be broken, but Pendered came to the rescue. He used a version of the rodding procedure to find out the wiring of the new fourth wheel and reflector. A windfall was that the Germans only changed the combination of fourth wheel and reflector once a month, so only once per month there would be a delay in breaking the messages. At the same time, the Germans still believed that someone needed all the codebooks to break Enigma.

The British had also managed to capture documents from U-505, which arrived at Bletchley Park on 20 June 1944. The documents held information on Offizier and regular Enigma settings for June, short weather codebooks, new versions of the bigram tables and short signal books, and some sort of address book to give their (U-boat) position on the grid in code. Because of these documents, after breaking the messages, the British codebreakers also almost directly understood where the U-boats were on the grid.

The Americans also worked out strategies and information that made breaking Shark and Dolphin faster. There were not as many wheel orders to check since Germans did not use the same wheel order in six months. Also, the Americans found out that when the wheel on the left was for instance wheel 2, then there would be an eighty percent chance the next day wheel 6, 7 or 8 was in this place. Besides, more Bombes became available. In the beginning of 1944, the British had access to 182 three- and four-wheel Bombes in Britain and America combined. This was an excessively significant difference with 1941, when only twelve were available.

Breaking the Enigma shortened the war by approximately two years.

Source: [26]

Bibliography

[1] *Die Handelsmaschine, Printing Enigma machine, 1923.* https://www.cryptomuseum.com/crypto/enigma/pe23/index.htm. Accessed: 30-4-2023.

[2] *Die schreibende Enigma, Printing Enigma machine, 1924-1926.* https://www.cryptomuseum.com/crypto/enigma/pe26/index.htm. Accessed: 30-4-2023.

[3] *Enigma A, Glow lamp Enigma machine, 1924.* https://www.cryptomuseum.com/crypto/enigma/a/index.htm. Accessed: 30-4-2023.

[4] *Enigma B, Glow lamp Enigma machine, 1924-1925.* https://www.cryptomuseum.com/crypto/enigma/b/index.htm. Accessed: 30-4-2023.

[5] *Enigma bericht procedures.* https://www.ciphermachinesandcryptology.com/nl/enigmaproc.htm. Accessed: between 28/4/2022 and 11/8/2022.

[6] *Enigma C, Glow lamp Enigma machine, 1925.* https://www.cryptomuseum.com/crypto/enigma/c/index.htm. Accessed: 30-4-2023.

[7] *Enigma D, Commercial Enigma A26, 1926.* https://www.cryptomuseum.com/crypto/enigma/d/index.htm. Accessed: 30-4-2023.

[8] *Enigma D, Commercial Enigma A27, 1927.* https://www.cryptomuseum.com/crypto/enigma/k/index.htm. Accessed: 30-4-2023.

[9] *Enigma, Enigma cipher machines.* https://www.cryptomuseum.com/crypto/enigma/index.htm. Accessed: 30-4-2023.

[10] *Enigma I, The Service Enigma Machine.* https://www.cryptomuseum.com/crypto/enigma/i/. Accessed: 30-4-2023.

[11] *Enigma M1, M2 and M3, 3-wheel Naval Enigma.* https://www.cryptomuseum.com/crypto/enigma/m3/index.htm. Accessed: 30-4-2023.

[12] *Enigma M4, Naval 4-wheel Enigma.* https://www.cryptomuseum.com/crypto/enigma/m4/index.htm. Accessed: 30-4-2023.

[13] *Enigma machine.* https://www.cs.mcgill.ca/~rwest/wikispeedia/wpcd/wp/e/Enigma_machine.htm. Accessed: between 28/4/2022 and 11/8/2022.

[14] *Enigma wiring.* https://www.cryptomuseum.com/crypto/enigma/wiring.htm. Accessed: between 28/4/2022 and 11/8/2022.

[15] *Enigma's Secrets, How it Worked and How the Code was Broken.* https://www.mpoweruk.com/enigma.htm. Accessed: between 28/4/2022 and 11/8/2022.

[16] *Exploring the Enigma.* https://plus.maths.org/content/exploring-enigma#question3. Accessed: 30-8-2022.

[17] James Grime. *Maths from the talk "Alan Turing and the Enigma Machine"*. 2013.

[18] *History of the Enigma, The rotor-based cipher machines.* https://www.cryptomuseum.com/crypto/enigma/hist.htm. Accessed: 30-4-2023.

[19] A. Hodges and D. Hofstadter. *Alan Turing: The Enigma: The Book That Inspired the Film The Imitation Game - Updated Edition*. Princeton University Press, 2014. ISBN: 9781400865123. URL: https://books.google.nl/books?id=QnUPBAAAQBAJ.

[20] *How does an Enigma machine work?* https://www.cryptomuseum.com/crypto/enigma/working.htm. Accessed: between 28/4/2022 and 11/8/2022.

[21] W. Kozaczuk. *Enigma: How the German Machine Cipher was Broken, and how it was Read by the Allies in World War Two*. Foreign intelligence book series. University Publications of America, 1984. ISBN: 9780890935477. URL: https://books.google.nl/books?id=bryEAAAAIAAJ.

[22] Louis Kruh and Cipher Deavours. "The commercial Enigma: Beginnings of machine cryptography". In: *Cryptologia* 26.1 (2002), pp. 1–16.

[23] *Military Use of the Enigma.* https://www.codesandciphers.org.uk/enigma/enigma3.htm. Accessed: 30-8-2022.

[24] Jared Owen. *How did the Enigma Machine work?* YouTube. URL: https://www.youtube.com/watch?v=ybkkiGtJmkM.

[25] Kalika Prasad and Munesh Kumari. "A review on mathematical strength and analysis of Enigma". In: *arXiv preprint arXiv:2004.09982* (2020).

[26] H. Sebag-Montefiore. *Enigma: The Battle For The Code*. Orion Publishing Group, 2018. ISBN: 9781474608329. URL: https://books.google.nl/books?id=O6xItAEACAAJ.

[27] *Technical Details of the Enigma Machine*. https://www.ciphermachinesandcryptology.com/en/enigmatech.htm. Accessed: between 28/4/2022 and 11/8/2022.

[28] *The Enigma Machine*. http://stanford.edu/class/archive/cs/cs106a/cs106a.1164/handouts/29-TheEnigmaMachine.pdf. Accessed: between 28/4/2022 and 11/8/2022.

[29] *The Enigma Machine, Its Construction, Operation and Complexity*. http://www.ellsbury.com/enigma3.htm. Accessed: between 28/4/2022 and 11/8/2022.

[30] *The German cipher machine Enigma*. https://www.matematiksider.dk/enigma_eng.html. Accessed: between 28/4/2022 and 11/8/2022.

A Program to find the number of starting settings

```
# This code is written by Inge Dekkers.
# In this Python program, we look at the amount of possible starting settings of an Enigma I machine.
# This is the enigma machine used by the German Army and Air Force.

from math import factorial
import sys

# AMOUNT OF ROTOR ORDERS

# Input of the number of rotors
number_of_rotors = int(input("The amount of rotors we can choose from: "))
n = number_of_rotors

# The machine needs at least 3 inserted rotors, so at least three rotors are needed to choose from.
if number_of_rotors < 3:
    print("This input is not valid.")
    sys.exit()

# Calculation of the amount of orders three rotors can be put in.
def rotor_order(x):
    order_amount = x * (x-1) * (x-2)
    return order_amount

print ("The amount of rotor orders is: ", rotor_order(n))

# AMOUNT OF PLUGBOARD SETTINGS

# Input of the number of rotors
number_of_cables = int(input("The amount of plugboard cables we use: "))
c = number_of_cables

# We cannot have more than 13 cables, since we have 26 letters.
# We do not have to use any cables.
if (number_of_cables < 0) or (number_of_cables > 13):
    print("This input is not valid")
    sys.exit()

def plugboard_settings(x):
    # We choose 2*x sockets of 26 to connect in pairs.
    sockets = int(factorial(26)/(factorial(26 - 2*x) * factorial(2*x)))
    i = 2*x - 1 # There are 2*x partners available for the first socket.
    pairs = 1
    while i > 0:
        pairs = pairs * i # Multiplying combinations
        i = i - 2 # In every step there are two options less.
    amount_plugboard = sockets * pairs # Combined by multiplication
    return amount_plugboard

print("The amount of rotor settings is: ", plugboard_settings(c))

# AMOUNT OF SETTINGS OF A SINGLE ROTOR

wiring_setting = 26*26*26 # Internal wiring
notch_setting = 26*26 # Notch position
rotor_setting = wiring_setting * notch_setting # Combined by multiplication
print("The amount of settings of the rotors is: ", rotor_setting)


# FINAL AMOUNT OF SETTINGS

settings = rotor_order(n) * plugboard_settings(c) * rotor_setting # Combining all elements
```

```
62
63 print("The amount of settings is: ", rotor_order(n), "*", plugboard_settings(c), "*",
       rotor_setting, "=", f'{settings:,}')
```

B Programmed version of Enigma I

In the code below, an Enigma I machine is programmed in the programming language Python.

```
# This code is written by Inge Dekkers.
# This code represents Enigma I, which is the enigma machine used by the German Army and Air Force.
# The program has five rotors to choose from.

#The input of the plugboardsettings
plug1, plug2 = input("The first plugboard setting/pair with spaces and capitals: ").split()
plug3, plug4 = input("The second plugboard setting/pair with spaces and capitals: ").split()
plug5, plug6 = input("The third plugboard setting/pair with spaces and capitals: ").split()
plug7, plug8 = input("The fourth plugboard setting/pair with spaces and capitals: ").split()
plug9, plug10 = input("The fifth plugboard setting/pair with spaces and capitals: ").split()
plug11, plug12 = input("The sixth plugboard setting/pair with spaces and capitals: ").split()
plug13, plug14 = input("The seventh plugboard setting/pair with spaces and capitals: ").split()
plug15, plug16 = input("The eighth plugboard setting/pair with spaces and capitals: ").split()
plug17, plug18 = input("The ninth plugboard setting/pair with spaces and capitals: ").split()
plug19, plug20 = input("The tenth plugboard setting/pair with spaces and capitals: ").split()

#This makes an array to save the matching ascii numbers of the plugboard settings.
plug = (ord(plug1), ord(plug2), ord(plug3), ord(plug4), ord(plug5), ord(plug6), ord(plug7), ord(plug8), ord(plug9),
ord(plug10), ord(plug11), ord(plug12), ord(plug13), ord(plug14), ord(plug15), ord(plug16), ord(plug17), ord(plug18),
ord(plug19), ord(plug20))

#This is the plugboard function, where two connected letters are switched.
def plugboard(p):
    for i in range(0, 19, 2):
        if p == plug[i]:
            p = plug[i + 1]
        elif p == plug[i + 1]:
            p = plug[i]
    return p

#Array differences for reflector pairs AY, BR, CU, DH, EQ, FS, GL, IP, JX, KN, MO, TZ and VW
difference_reflector = (24, 16, 18, 4, 12, 13, 5, -4, 7, 14, 3, -5, 2, -3, -2, -7, -12, -16, -13, 6, -18, 1, -1, -14, -24, -6)

#Reflector function, two letters are switched based on differences between the ascii values.
def reflector(r):
    for i in range(65, 91): #This ends at 91 because range goes from 65 to 90 in this case, by definition of range.
        if r == i:
            r = r + difference_reflector[(i - 65)]
            break
    return r

#Here the user chooses the rotors, their order and their ringsettings.
input_X = int(input("Most left rotor (in numbers):  "))
input_Y = int(input("Middle rotor (in numbers):  "))
input_Z = int(input("Most right rotor (in numbers):  "))
setting_X = int(input("Setting of the most left rotor:  "))
```

```
setting_Y = int(input("Setting of the middel rotor:  "))
setting_Z = int(input("Setting of the most right rotor:  "))
ring_X = int(input("Ring setting of the most left rotor:  "))
ring_Y = int(input("Ring setting of the middle rotor:  "))
ring_Z = int(input("Ring setting of the most right rotor:  "))


#Here values are assigned to the variables we need in the rotors.
s = [0, 0, 0, 0, 0] #settings (showing through the window)
rs = [0, 0, 0, 0, 0] #ringsettings

#The settings are put into an array.
for i in range(1, 6):
    if input_X == i:
        s[i - 1] = setting_X #(i-1) is used as an array counts from zero
        rs[i - 1] = ring_X
    if input_Y == i:
        s[i - 1] = setting_Y
        rs[i - 1] = ring_Y
    if input_Z == i:
        s[i - 1] = setting_Z
        rs[i - 1] = ring_Z

s1, s2, s3, s4, s5 = s
rs1, rs2, rs3, rs4, rs5 = rs

#Here the rotation construction starts.

def position_X():
    for i in range(1,6): #goes from 1 up until 5 by definition of range
        if input_X == i:
            return s[i-1]

def position_Y():
    for i in range(1,6): #goes from 1 up until 5 by definition of range
        if input_Y == i:
            return s[i-1]

def position_Z():
    for i in range(1,6): #goes from 1 up until 5 by definition of range
        if input_Z == i:
            return s[i-1]

#We only need positions Y and Z because X is the most left rotor.

#Notch list I: Q->R, II: E->F, III: V->W, IV: J->K, V: Z->A
notch_list = [18, 6, 23, 11, 1]

#The notches are matched with the rotors.
def notch_Y():
    for i in range(1,6):
        if input_Y == i:
            n = notch_list[i-1]
            return n

def notch_Z():
    for i in range(1,6):
        if input_Z == i:
            n = notch_list[i-1]
            return n

rot_z = position_Z()
rot_y = position_Y()
rot_x = position_X()
n_z = notch_Z()
n_y = notch_Y()
```

```
113
114 variable_list = []
115 variable_list.extend((s1,s2,s3,s4,s5))
116
117 #This is the definition for the rotation mechanism.
118 def rotation(rot_x, rot_y, rot_z, n_y, n_z, input_X, input_Y, input_Z):
119     rot_z = rot_z + 1 #Rotor Z always advances one step.
120     if rot_z > 26:
121         rot_z = rot_z - 26
122     if rot_y == n_y - 1:
123         if rot_z == n_z:
124             rot_y = rot_y + 1
125             if rot_y > 26:
126                 rot_y = rot_y - 26
127             rot_x = rot_x + 1
128             if rot_x > 26:
129                 rot_x = rot_x - 26
130         else:
131             rot_y = rot_y + 1
132             if rot_y > 26:
133                 rot_y = rot_y - 26
134             rot_x = rot_x + 1
135             if rot_x > 26:
136                 rot_x = rot_x - 26
137     elif rot_z == n_z:
138         rot_y = rot_y + 1
139         if rot_y > 26:
140             rot_y = rot_y - 26
141     for i in range(1,6):
142         if input_X == i:
143             variable_list[i - 1] = rot_x
144     for i in range(1,6):
145         if input_Y == i:
146             variable_list[i - 1] = rot_y
147     for i in range(1,6):
148         if input_Z == i:
149             variable_list[i - 1] = rot_z
150     s1, s2, s3, s4, s5 = variable_list
151     return(s1, s2, s3, s4, s5, rot_x, rot_y, rot_z)
152
153 #Here we end the rotation construction
154
155 #The rotors
156
157 #Rotor I
158
159 #Differences AE,BK,CM,DF,EL,FG,GD,HQ,IV,JZ,KN,LT,MO,NW,OY,PH,QX,RU,SS,TP,UA,VI,WB,XR,YC
    ,ZJ (from right to left)
160 rotor_I_list = [4, 9, 10, 2, 7, 1, -3, 9, 13, 16, 3, 8, 2, 9, 10, -8, 7, 3, 0, -4, -20,
     -13, -21, -6, -22, -16]
161
162 def rotor_I(q1, s1):
163     q1 = q1 + (s1 - 1) - (rs1 - 1) #entry letter (q1) entering the rotor considering
        setting and ringsetting
164     if q1 < 65:
165         q1 = q1 + 26
166     elif q1 > 90:
167         q1 = q1 - 26
168     for i in range(65, 91):
169         if q1 == i:
170             q1 = q1 - (s1 - 1) + (rs1 - 1) + rotor_I_list[(i - 65)] #entry letter going
                through rotor
171             if q1 < 39: # To keep the alphabet
172                 q1 = q1 + 52
173             elif q1 < 65:
174                 q1 = q1 + 26
```

```
            if q1 > 116:
                q1 = q1 - 52
            elif q1 > 90:
                q1 = q1 - 26
            break
    return q1

#Differences AU,BW,CY,DG,EA,FD,GF,HP,IV,JZ,KB,LE,MC,NK,OM,PT,QH,RX,SS,TL,UR,VI,WN,XQ,YO,ZJ (from left to right)
rotor_I_reversed_list = [20, 21, 22, 3, -4, -2, -1, 8, 13, 16, -9, -7, -10, -3, -2, 4, -9, 6, 0, -8, -3, -13, -9, -7, -10, -16]

def rotor_I_reversed(p1, s1):
    p1 = p1 + (s1 - 1) - (rs1 - 1)
    if p1 < 65:
        p1 = p1 + 26
    elif p1 > 90:
        p1 = p1 - 26
    for i in range(65, 91):
        if p1 == i:
            p1 = p1 - (s1 - 1) + (rs1 - 1) + rotor_I_reversed_list[(i - 65)]
            if p1 < 39:
                p1 = p1 + 52
            elif p1 < 65:
                p1 = p1 + 26
            if p1 > 116:
                p1 = p1 - 52
            elif p1 > 90:
                p1 = p1 - 26
            break
    return p1

#Differences AA,BJ,CD,CK,ES,FI,GR,HU,IX,JB,KL,LH,MW,NT,OM,PC,QQ,RG,SZ,TN,UP,VY,WF,XV,YO,ZE (from right to left)
rotor_II_list = [0, 8, 1, 7, 14, 3, 11, 13, 15, -8, 1, -4, 10, 6, -2, -13, 0, -11, 7, -6, -5, 3, -17, -2, -10, -21]

#Rotor II

def rotor_II(q2, s2):
    q2 = q2 + (s2 - 1) - (rs2 - 1)
    if q2 < 65:
        q2 = q2 + 26
    elif q2 > 90:
        q2 = q2 - 26
    for i in range(65, 91):
        if q2 == i:
            q2 = q2 - (s2 - 1) + (rs2 - 1) + rotor_II_list[(i - 65)]
            if q2 < 39:
                q2 = q2 + 52
            elif q2 < 65:
                q2 = q2 + 26
            if q2 > 116:
                q2 = q2 - 52
            elif q2 > 90:
                q2 = q2 - 26
            break
    return q2

#Differences AA,BJ,CP,DC,EZ,FW,GR,HL,IF,JB,KD,LK,MO,NT,OY,PU,QQ,RG,SE,TN,UH,VX,WM,XI,YV,ZS (from left to right)
rotor_II_reversed_list = [0, 8, 13, -1, 21, 17, 11, 4, -3, -8, -7, -1, 2, 6, 10, 5, 0, -11, -14, -6, -13, 2, -10, -15, -3, -7]

def rotor_II_reversed(p2, s2):
    p2 = p2 + (s2 - 1) - (rs2 - 1)
```

```
    if p2 < 65:
        p2 = p2 + 26
    elif p2 > 90:
        p2 = p2 - 26
    for i in range(65, 91):
        if p2 == i:
            p2 = p2 - (s2 - 1) + (rs2 - 1) + rotor_II_reversed_list[(i - 65)]
            if p2 < 39:
                p2 = p2 + 52
            elif p2 < 65:
                p2 = p2 + 26
            if p2 > 116:
                p2 = p2 - 52
            elif p2 > 90:
                p2 = p2 - 26
            break
    return p2

#Differences AB,BD,CF,DH,EJ,FL,GC,HP,IR,JT,KX,LV,MZ,NN,OY,PE,QI,RW,SG,TA,UK,VM,WU,XS,YQ,ZO (from right to left)
rotor_III_list = [1, 2, 3, 4, 5, 6, -4, 8, 9, 10, 13, 10, 13, 0, 10, -11, -8, 5, -12, -19, -10, -9, -2, -5, -8, -11]

#Rotor III

def rotor_III(q3, s3):
    q3 = q3 + (s3 - 1) - (rs3 - 1)
    if q3 < 65:
        q3 = q3 + 26
    elif q3 > 90:
        q3 = q3 - 26
    for i in range(65, 91):
        if q3 == i:
            q3 = q3 - (s3 - 1) + (rs3 - 1) + rotor_III_list[(i - 65)]
            if q3 < 39:
                q3 = q3 + 52
            elif q3 < 65:
                q3 = q3 + 26
            if q3 > 116:
                q3 = q3 - 52
            elif q3 > 90:
                q3 = q3 - 26
            break
    return q3

#Differences AT,BA,CG,DB,EP,FC,GS,HD,IQ,JE,KU,LF,MV,NN,OZ,PH,QY,RI,SX,TJ,UW,VL,WR,XK,YO,ZM (from left to right)
rotor_III_reversed_list = [19, -1, 4, -2, 11, -3, 12, -4, 8, -5, 10, -6, 9, 0, 11, -8, 8, -9, 5, -10, 2, -10, -5, -13, -10, -13]

def rotor_III_reversed(p3, s3):
    p3 = p3 + (s3 - 1) - (rs3 - 1)
    if p3 < 65:
        p3 = p3 + 26
    elif p3 > 90:
        p3 = p3 - 26
    for i in range(65, 91):
        if p3 == i:
            p3 = p3 - (s3 - 1) + (rs3 - 1) + rotor_III_reversed_list[(i - 65)]
            if p3 < 39:
                p3 = p3 + 52
            elif p3 < 65:
                p3 = p3 + 26
            if p3 > 116:
                p3 = p3 - 52
            elif p3 > 90:
```

```
                p3 = p3 - 26
            break
    return p3

#Differences AE,BS,CO,DV,EP,FZ,GJ,HA,IY,JQ,KU,LI,MR,NH,OX,PL,QN,RF,ST,TG,UK,VD,WC,XM,YW,ZB (from right to left)
rotor_IV_list = [4, 17, 12, 18, 11, 20, 3, -7, 16, 7, 10, -3, 5, -6, 9, -4, -3, -12, 1, -13, -10, -18, -20, -11, -2, -24]

#Rotor IV

def rotor_IV(q4, s4):
    q4 = q4 + (s4 - 1) - (rs4 - 1)
    if q4 < 65:
        q4 = q4 + 26
    elif q4 > 90:
        q4 = q4 - 26
    for i in range(65, 91):
        if q4 == i:
            q4 = q4 - (s4 - 1) + (rs4 - 1) + rotor_IV_list[(i - 65)]
            if q4 < 39:
                q4 = q4 + 52
            elif q4 < 65:
                q4 = q4 + 26
            if q4 > 116:
                q4 = q4 - 52
            elif q4 > 90:
                q4 = q4 - 26
            break
    return q4

#Differences AH,BZ,CW,DV,EA,FR,GT,HN,IL,JG,KU,LP,MX,NQ,OC,PE,QJ,RM,SB,TS,UK,VD,WY,XO,YI,ZF (from left to right)
rotor_IV_reversed_list = [7, 24, 20, 18, -4, 12, 13, 6, 3, -3, 10, 4, 11, 3, -12, -11, -7, -5, -17, -1, -10, -18, 2, -9, -16, -20]

def rotor_IV_reversed(p4, s4):
    p4 = p4 + (s4 - 1) - (rs4 - 1)
    if p4 < 65:
        p4 = p4 + 26
    elif p4 > 90:
        p4 = p4 - 26
    for i in range(65, 91):
        if p4 == i:
            p4 = p4 - (s4 - 1) + (rs4 - 1) + rotor_IV_reversed_list[(i - 65)]
            if p4 < 39:
                p4 = p4 + 52
            elif p4 < 65:
                p4 = p4 + 26
            if p4 > 116:
                p4 = p4 - 52
            elif p4 > 90:
                p4 = p4 - 26
            break
    return p4

#Differences AV,BZ,CB,DR,EG,FI,GT,HY,IU,JP,KS,LD,MN,NH,OL,PX,QA,RW,SM,TJ,UQVO,WF,XE,YC,ZK (from right to left)
rotor_V_list = [21, 24, -1, 14, 2, 3, 13, 17, 12, 6, 8, -8, 1, -6, -3, 8, -16, 5, -6, -10, -4, -7, -17, -19, -22, -15]

#Rotor V

def rotor_V(q5, s5):
    q5 = q5 + (s5 - 1) - (rs5 - 1)
    if q5 < 65:
```

```
        q5 = q5 + 26
    elif q5 > 90:
        q5 = q5 - 26
    for i in range(65, 91):
        if q5 == i:
            q5 = q5 - (s5 - 1) + (rs5 - 1) + rotor_V_list[(i - 65)]
            if q5 < 39:
                q5 = q5 + 52
            elif q5 < 65:
                q5 = q5 + 26
            if q5 > 116:
                q5 = q5 - 52
            elif q5 > 90:
                q5 = q5 - 26
            break
    return q5

#Difference AQ,BC,CY,DL,EX,FW,GE,HN,IF,JT,KZ,LO,MS,NM,OV,PJ,QU,RD,SK,TG,UI,A,WR,XP,YH,ZB
rotor_V_reversed_list = [16, 1, 22, 8, 19, 17, -2, 6, -3, 10, 15, 3, 6, -1, 7, -6, 4, -14, -8, -13, -12, -21, -5, -8, -17, -24]

def rotor_V_reversed(p5, s5):
    p5 = p5 + (s5 - 1) - (rs5 - 1)
    if p5 < 65:
        p5 = p5 + 26
    elif p5 > 90:
        p5 = p5 - 26
    for i in range(65, 91):
        if p5 == i:
            p5 = p5 - (s5 - 1) + (rs5 - 1) + rotor_V_reversed_list[(i - 65)]
            if p5 < 39:
                p5 = p5 + 52
            elif p5 < 65:
                p5 = p5 + 26
            if p5 > 116:
                p5 = p5 - 52
            elif p5 > 90:
                p5 = p5 - 26
            break
    return p5

#The next six functions are for the rotor order

def rotor_X(x1, rot_x):
    test1 = input_X
    if test1 == 1:
        return rotor_I(x1, rot_x)
    elif test1 == 2:
        return rotor_II(x1, rot_x)
    elif test1 == 3:
        return rotor_III(x1, rot_x)
    elif test1 == 4:
        return rotor_IV(x1, rot_x)
    elif test1 == 5:
        return rotor_V(x1, rot_x)

def rotor_X_reversed(x2, rot_x):
    test2 = input_X
    if test2 == 1:
        return rotor_I_reversed(x2, rot_x)
    elif test2 == 2:
        return rotor_II_reversed(x2, rot_x)
    elif test2 == 3:
        return rotor_III_reversed(x2, rot_x)
    elif test2 == 4:
```

```
        return rotor_IV_reversed(x2, rot_x)
    elif test2 == 5:
        return rotor_V_reversed(x2, rot_x)


def rotor_Y(x3, rot_y):
    test3 = input_Y
    if test3 == 1:
        return rotor_I(x3, rot_y)
    elif test3 == 2:
        return rotor_II(x3, rot_y)
    elif test3 == 3:
        return rotor_III(x3, rot_y)
    elif test3 == 4:
        return rotor_IV(x3, rot_y)
    elif test3 == 5:
        return rotor_V(x3, rot_y)

def rotor_Y_reversed(x4, rot_y):
    test4 = input_Y
    if test4 == 1:
        return rotor_I_reversed(x4, rot_y)
    elif test4 == 2:
        return rotor_II_reversed(x4, rot_y)
    elif test4 == 3:
        return rotor_III_reversed(x4, rot_y)
    elif test4 == 4:
        return rotor_IV_reversed(x4, rot_y)
    elif test4 == 5:
        return rotor_V_reversed(x4, rot_y)


def rotor_Z(x5, rot_z):
    test5 = input_Z
    if test5 == 1:
        return rotor_I(x5, rot_z)
    elif test5 == 2:
        return rotor_II(x5, rot_z)
    elif test5 == 3:
        return rotor_III(x5, rot_z)
    elif test5 == 4:
        return rotor_IV(x5, rot_z)
    elif test5 == 5:
        return rotor_V(x5, rot_z)

def rotor_Z_reversed(x6, rot_z):
    test6 = input_Z
    if test6 == 1:
        return rotor_I_reversed(x6, rot_z)
    elif test6 == 2:
        return rotor_II_reversed(x6, rot_z)
    elif test6 == 3:
        return rotor_III_reversed(x6, rot_z)
    elif test6 == 4:
        return rotor_IV_reversed(x6, rot_z)
    elif test6 == 5:
        return rotor_V_reversed(x6, rot_z)


#Input the text you want to code.
x = input("The text I want to code is (in capitals and without spaces): ")

#All the stations combined.
for i in range(len(x)): #len(x) gives length of the string of text
    rotation(rot_x, rot_y, rot_z, n_y, n_z, input_X, input_Y, input_Z)
    s1, s2, s3, s4, s5, rot_x, rot_y, rot_z = rotation(rot_x, rot_y, rot_z, n_y, n_z,
```

```
            input_X, input_Y, input_Z)
    l = ord(x[i])
    k = plugboard(l)
    s = rotor_Z(k, rot_z)
    w = rotor_Y(s, rot_y)
    v = rotor_X(w, rot_x)
    m = reflector(v)
    z = rotor_X_reversed(m, rot_x)
    o = rotor_Y_reversed(z, rot_y)
    h = rotor_Z_reversed(o, rot_z)
    g = plugboard(h)
    print(chr(g), end = " ")
```

C Wheel wiring

For the Poles to construct a replica of the Enigma machine, it was necessary to find out the wiring of the wheels. They did this using permutations. At the beginning of the war, the Germans enciphered their message key two consecutive times and put this before the enciphered message. This resulted in pairs of letters from which was known that the letters before enciphering were the same. These six letters, representing the doubly enciphered message key, could also be expressed via six different permutations, which we name ϕ_1 up until ϕ_6.

We often use the same notation as used in Section 3. So, we have:

$$\begin{aligned} E(n) &= \text{the enigma machine} \\ P &= \text{the plugboard} \\ f_{\delta_L} &= \text{wiring left wheel in the Enigma machine} \\ f_{\delta_M} &= \text{wiring middle wheel in the Enigma machine} \\ f_{\delta_R} &= \text{wiring right wheel in the Enigma machine} \\ F &= \text{the reflector} \\ H &= \text{entry disc} \end{aligned}$$

As already explained in Sections 2 and 3, we get the following path:

$$P^{-1} \circ H^{-1} \circ f_{\delta_R}^{-1} \circ f_{\delta_M}^{-1} \circ f_{\delta_L}^{-1} \circ F \circ f_{\delta_L} \circ f_{\delta_M} \circ f_{\delta_R} \circ H \circ P.$$

However, for simplicity, we denote this like

$$P\ H\ f_{\delta_R}\ f_{\delta_M}\ f_{\delta_L}\ F\ f_{\delta_L}^{-1}\ f_{\delta_M}^{-1}\ f_{\delta_R}^{-1}\ H^{-1}\ P^{-1}.$$

We defined d to be the rotation. So, we have

$$d = \begin{pmatrix} 0 & 1 & 2 & \cdots & 23 & 24 & 25 \\ 1 & 2 & 3 & \cdots & 24 & 25 & 0 \end{pmatrix} = (0\ 1\ 2\ \ldots\ 24\ 25).$$

So, for every letter of the six letters representing the enciphering of the message key we have:

$$\begin{aligned} \phi_1 &= P\ H\ d\ f_{\delta_R}\ d^{-1}\ f_{\delta_M}\ f_{\delta_L}\ F\ f_{\delta_L}^{-1}\ f_{\delta_M}^{-1}\ d\ f_{\delta_R}^{-1}\ d^{-1}\ H^{-1}\ P^{-1} \\ \phi_2 &= P\ H\ d^2\ f_{\delta_R}\ d^{-2}\ f_{\delta_M}\ f_{\delta_L}\ F\ f_{\delta_L}^{-1}\ f_{\delta_M}^{-1}\ d^2\ f_{\delta_R}^{-1}\ d^{-2}\ H^{-1}\ P^{-1} \\ &\vdots \\ \phi_6 &= P\ H\ d^6\ f_{\delta_R}\ d^{-6}\ f_{\delta_M}\ f_{\delta_L}\ F\ f_{\delta_L}^{-1}\ f_{\delta_M}^{-1}\ d^6\ f_{\delta_R}^{-1}\ d^{-6}\ H^{-1}\ P^{-1} \end{aligned}$$

Assumption 1. *We assume that only the right rotor advances and the middle and left rotor do not.*

We can quite safely assume this, as in only twenty-one out of twenty-six cases there will not be an advancement of the next rotor. So, for all six permutations, there is a reoccurring sequence. For simplicity, we refer to this reoccurring sequence as S. We now can rewrite the permutations:

$$\begin{aligned} \phi_1 &= P\ H\ d\ f_{\delta_R}\ d^{-1}\ S\ d\ f_{\delta_R}^{-1}\ d^{-1}\ H^{-1}\ P^{-1} \\ \phi_2 &= P\ H\ d^2\ f_{\delta_R}\ d^{-2}\ S\ d^2\ f_{\delta_R}^{-1}\ d^{-2}\ H^{-1}\ P^{-1} \\ &\vdots \\ \phi_6 &= P\ H\ d^6\ f_{\delta_R}\ d^{-6}\ S\ d^6\ f_{\delta_R}^{-1}\ d^{-6}\ H^{-1}\ P^{-1} \end{aligned}$$

This leaves us with six equations containing four unknown permutations: P, H, f_{δ_R}, and S. First Kozaczuk thought H was known, as he thought this would be the same as in the commercial and military Enigmas. It turned out this was not the case. However, he thought so and thus he was left with only three unknowns. In addition, on 9 December 1932, Kozaczuk received photos from the daily keys of September and October 1932.

These contained the plugboard settings. So, in this case P was known. Since P and H were known, they were transformed to the left:

$$
\begin{aligned}
H^{-1}\ P^{-1}\ \phi_1\ P\ H &= d\ f_{\delta_R}\ d^{-1}\ S\ d\ f_{\delta_R}^{-1}\ d^{-1} \\
H^{-1}\ P^{-1}\ \phi_2\ P\ H &= d^{\,2}\ f_{\delta_R}\ d^{-2}\ S\ d^{\,2}\ f_{\delta_R}^{-1}\ d^{-2} \\
&\vdots \\
H^{-1}\ P^{-1}\ \phi_6\ P\ H &= d^{\,6}\ f_{\delta_R}\ d^{-6}\ S\ d^{\,6}\ f_{\delta_R}^{-1}\ d^{-6}
\end{aligned}
$$

So, on the left the permutations are known, on the right f_{δ_R} and S were unknown. As d was also known, these could also be transformed to the left. For simplicity we also rename:

$$
\begin{aligned}
\mathrm{U} &= d^{-1}\ P^{-1}\ \phi_1\ P\ H\ d = f_{\delta_R}\ d^{-1}\ S\ d\ f_{\delta_R}^{-1} \\
\mathrm{V} &= d^{-2}\ P^{-1}\ \phi_1\ P\ H\ d^{\,2} = f_{\delta_R}\ d^{-2}\ S\ d^{\,2}\ f_{\delta_R}^{-1} \\
&\vdots \\
\mathrm{Z} &= d^{-6}\ P^{-1}\ \phi_1\ P\ H\ d^{\,6} = f_{\delta_R}\ d^{-6}\ S\ d^{\,6}\ f_{\delta_R}^{-1}
\end{aligned}
$$

When we multiply, we get:

$$
\begin{aligned}
\mathrm{UV} &= f_{\delta_R}\ d^{-1}(\ S\ d^{-1}\ S\ d)\ d\ f_{\delta_R}^{-1} \\
\mathrm{VW} &= f_{\delta_R}\ d^{-2}(\ S\ d^{-1}\ S\ d)\ d^{\,2}\ f_{\delta_R}^{-1} \\
&\vdots \\
\mathrm{YZ} &= f_{\delta_R}\ d^{-5}(\ S\ d^{-1}\ S\ d)\ d^{\,5}\ f_{\delta_R}^{-1}
\end{aligned}
$$

When we substitute, we get:

$$
\begin{aligned}
\mathrm{VW} &= f_{\delta_R}\ d^{-1}\ f_{\delta_R}^{-1}\ (\mathrm{UV})\ f_{\delta_R}\ d\ f_{\delta_R}^{-1} \\
\mathrm{WX} &= f_{\delta_R}\ d^{-1}\ f_{\delta_R}^{-1}\ (\mathrm{VW})\ f_{\delta_R}\ d\ f_{\delta_R}^{-1} \\
\mathrm{XY} &= f_{\delta_R}\ d^{-1}\ f_{\delta_R}^{-1}\ (\mathrm{WX})\ f_{\delta_R}\ d\ f_{\delta_R}^{-1} \\
\mathrm{YZ} &= f_{\delta_R}\ d^{-1}\ f_{\delta_R}^{-1}\ (\mathrm{XY})\ f_{\delta_R}\ d\ f_{\delta_R}^{-1}
\end{aligned}
$$

The question that remains, is how these expressions can be useful for finding the wiring within the wheels. We explain by showing an example. We do not work out the beginning of the example as it is tedious. We can find products $\phi_1\phi_4$, $\phi_2\phi_5$, and $\phi_3\phi_6$ by looking for cycles in the six letter key indicators of a certain day. From these expressions, we can find ϕ_1 up until ϕ_6, and with P, d, and all powers of d known, we can find U up until Z. Then by multiplying these permutations we can find UV up until YZ, from which we will only need UV, VW and WX.

The only unknown remaining is $f_{\delta_R}\ d\ f_{\delta_R}^{-1}$, which we need to find. Say the following UV, VW and WX are found:

UV	=	(	i	b	d	a	j	)	(	e	f	g	h	c	)
VW	=	(	e	a	h	f	b	)	(	c	g	j	i	d	)
WX	=	(	i	e	f	c	g	)	(	b	h	j	a	d	)

The first step will be to line permutations UV and VW, and permutations VW an WX up, such that the vertical letter pairs will be identical in both pairings. Since the permutations UV, VW and WX are already known, only $f_{\delta_R}\ d\ f_{\delta_R}^{-1}$ is still unknown. $f_{\delta_R}\ d\ f_{\delta_R}^{-1}$ sends a letter from for instance UV to VW, but also from VW to WX. Both these 'steps' need to be identical. Therefore, the two sets of permutations need to line up such that all pairings are identical. As an example, we look at a ten letter alphabet. Rewriting and lining up UV, VW, and WX gives the following:

UV	=	(	a	j	i	b	d	)	(	c	e	f	g	h	)
VW	=	(	d	c	g	j	i	)	(	e	a	h	f	b	)

VW	=	(	d	c	g	j	i	)	(	e	a	h	f	b	)
WX	=	(	i	e	f	c	g	)	(	a	d	b	h	j	)

Now, all the lined-up pairs in both situations are identical. So, $f_{\delta_R}\ d\ f_{\delta_R}^{-1}$ can be found. For example, if in VW we go from D to C, $f_{\delta_R}\ d^{-1}\ f_{\delta_R}^{-1}$ sends D to A, then UV will send A to J and $f_{\delta_R}\ d\ f_{\delta_R}^{-1}$ sends J to C. So, the connections in $f_{\delta_R}\ d\ f_{\delta_R}^{-1}$ will include (AD) and (JC). This process will be repeated until $f_{\delta_R}\ d\ f_{\delta_R}^{-1}$ is complete. In this example the following is obtained:

$$f_{\delta_R}\ d\ f_{\delta_R}^{-1} = (\text{a d i g f h b j c e})$$

Now the same can be done to find f_{δ_R}, since this is the exact same situation as described above. The permutation d can be placed underneath $f_{\delta_R}\ d\ f_{\delta_R}^{-1}$ and rearranging can start. The only difference is that there are no other pairings to compare with. So, looking at the example, there would be ten possible permutations for f_{δ_R}. Which permutation is the correct one, depends on factors which can only be known when looking at the reconstruction of the wiring in all rotors.

Source: [21]

D Codebreaking methods invented by the Poles

D.1 The Characteristic Method

In 1936, a characteristic method for breaking the code was invented. It was used for breaking the Enigma cipher which was used by the Armed Forces. However, it became redundant as soon as Germany changed its indicator procedure on 15 September 1938.

The characteristic method made use of the way the Germans communicated their indicator to the receiver of the message. The **message settings**, which were the letters showing through the windows just before starting enciphering the message, were enciphered two consecutive times. So, every message started with six letters, which were used to communicate the message settings. From these six letters, chains could be created from the following letter pairs: first and fourth, second and fifth, and third and sixth. This could give chains of various lengths, as the length depends on the wheel order and the **initial position**, which were the letters showing through the windows just before starting enciphering the message setting.

For every wheel order, there would be a specific length of a chain that would be possible. The Poles reported this length for every wheel order in a catalogue, which would help the British cryptographers to break the code. So, after receiving a message, the codebreakers would construct the chains from the intercepted indicators, and they would determine the length of it. After that, they would search in the catalogue to find out which wheel orders could be possible with the found chain lengths. After discovering the wheel order and wheel settings, other settings could be worked out.

The way the chains or cycles were constructed is explained in the following example:

Say the intercepted indicators were as follows:

Indicator 1: D M Q V B N

Indicator 2: V O N P U Y

The chain that could be constructed out of these indicators, would then be DVP (from the first letter from indicator 1, to the fourth letter of indicator 1, to the first letter of indicator 2, to the fourth letter of indicator 2). If more indicators were intercepted, this chain might be expanded until no link could be found anymore. If a chain stopped, another could be started. So, it is possible that there might be more chains for the letter pairs formed by the first and fourth letters. The same could be done for the pair formed by the second and fifth letters and pair forms by the third and sixth pair. For every wheel setting, around sixty indicators were needed to create the series of chains. These were also known as **characteristic cycles**.

D.2 The Polish Bomby

The Bomby, which is plural for Bomba, were used to find out which wheel order and settings could have produced a certain indicator. Sometimes, a situation like the following situation would occur:

	Indicator	Initial position
Message 1	W A H W I K	A A A
Message 2	D W R M W R	A A G
Message 3	R A W K T W	A A L

There are three indicators from which the pairs all have the same reoccurring letter. The initial positions on the right were known, as the receiver sent these unenciphered at the beginning of the message. The initial position does not give everything away, since the core position of the wheel is not known, the initial position only shows what is shown through the windows. The Bomby were invented to find these wheel core settings.

A Bomba consisted out of six replica Enigma machines that were wired together. However, there were no rings on the wheels and there was no plugboard. It was solely meant to test the wheel orders, which could only be one wheel order at the time.

For example, say we have a specific wheel order from left to right: 2/3/1. In the six Enigma machines the wheels were set into the exact same order. Then the wheels were individually set in such a way that the 'distance' between the six sets of wheels, were identical to the 'distance' between the wheel positions that produced the six W's. This distance could be figured out by looking at the initial positions corresponding to the indicators.

To define distance, we follow an example corresponding to the table above. For example, AAG is six places removed from AAA, because the right-hand wheel has to make six steps to go from AAA to AAG in the window. Therefore, the distance between these two positions is six.

After everything was set up and installed, the process could begin. The Bomba would test if the initial positions could have produced the indicators. If this would be the case, there would be a chance that the Bomba had found the correct wheel order and positions for the message setting of that day. If this was not the case, then all the right wheels in all six Enigma machines had to be rotated one place further, so that the correct distance was kept maintained and the same test would have been executed again. This was done until all positions were tested or until the correct settings were found.

Source: [26]

E The British Bombe

Too many settings are possible on the Enigma machine to test manually. To try and reduce the number of settings that needed to be tested manually, Alan Turing invented the Bombe. This machine was used to rule out as many plugboard settings and wheel orders as possible. To do so, the machine made logical deductions by making use of contradictions. To start with, assumptions were made, which were tested. If a deduction was impossible, one of the assumptions was incorrect. Then the Bombe would test the next set of assumptions until impossible deductions could not be found. These sets would be tested manually on a replica Enigma machine.

Example

Say we have the following assumptions:

1. The wheel order from left to right 1/2/3.
2. One of the plugboard connections is B-L.
3. The cipher text and matching plain text are in the table below. We assume AAA is the wheel setting on the first letter of the crib, AAB the wheel setting for the second letter of the crib etc.

Relative wheel position	1	2	3	4	5	6	7
Cipher text	Z	Q	L	R	X	S	L
Crib	D	E	K	K	E	R	S

Deduction 1

The first deduction will be based on the third pair of letters. In this case, the cipher letter is L and the crib letter is K. We know that for this pair, the wheel setting is AAC. We have the following situation:

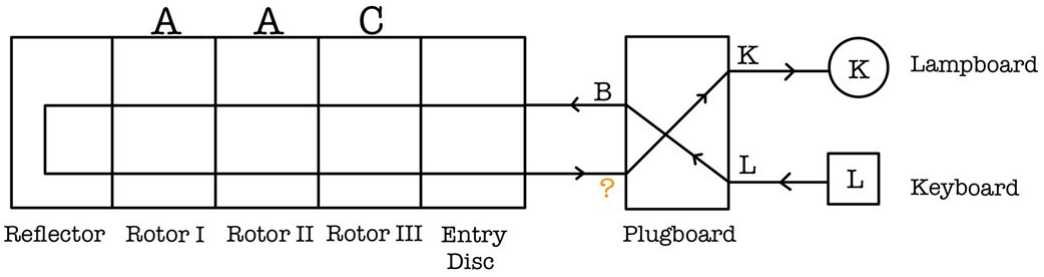


Figure 33: Deduction 1 with an unknown

Now we can find out which letter is the one with the orange question mark. This can be done by putting a replica Enigma machine in the given settings, but without a plugboard. If we type out the letter B, the letter that will replace the orange question mark will light up. Say, this is the letter F. Then we have the following:

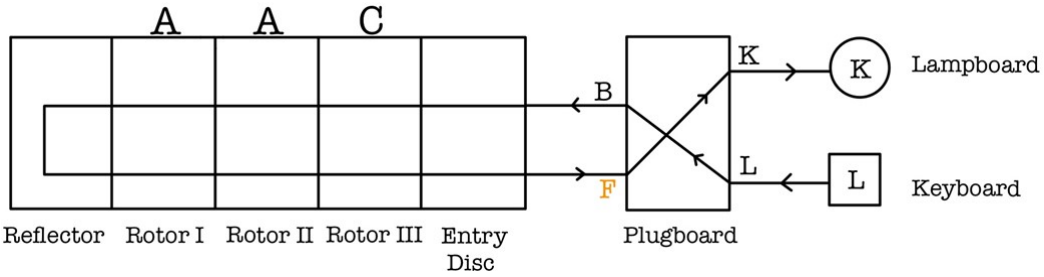


Figure 34: Deduction 1

We can see that F-K is a new plugboard setting, given the assumptions above, which is our first deduction.

Deduction 2

The second deduction will be based on the fourth pair of letters. In this case, the cipher letter is R and the crib letter is K. We know that for this pair, the wheel setting is AAD. We have the following situation:

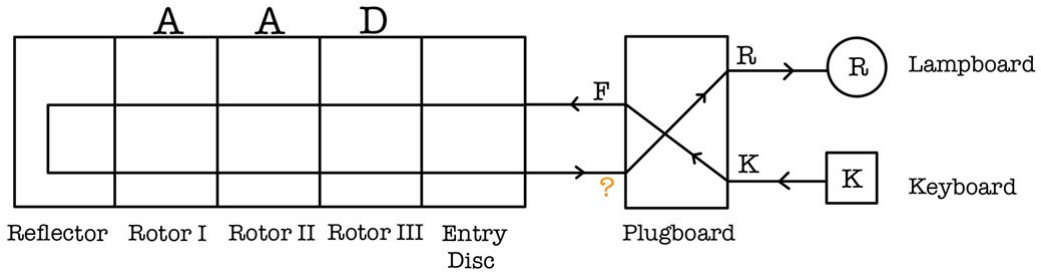


Figure 35: Deduction 2 with an unknown

Again, we can deduce that if we type out the letter F, the letter that will replace the orange question mark will light up. Say, this is the letter H. Then we have the following:

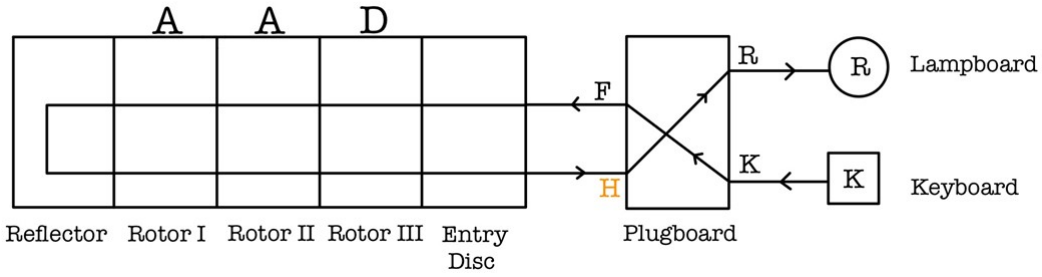


Figure 36: Deduction 2

We can see that H-R is a new plugboard setting, given the assumptions above, which is our second deduction.

Deduction 3

The third deduction will be based on the sixth pair of letters. In this case, the cipher letter is S and the crib letter is R. We know that for this pair, the wheel setting is AAF. We have the following situation:

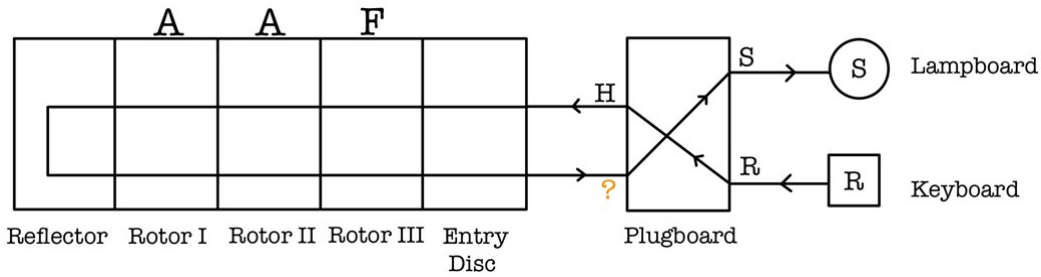


Figure 37: Deduction 3 with an unknown

Again, we can deduce that if we type out the letter H, the letter that will replace the orange question mark will light up. Say, this is the letter I. Then we have the following:

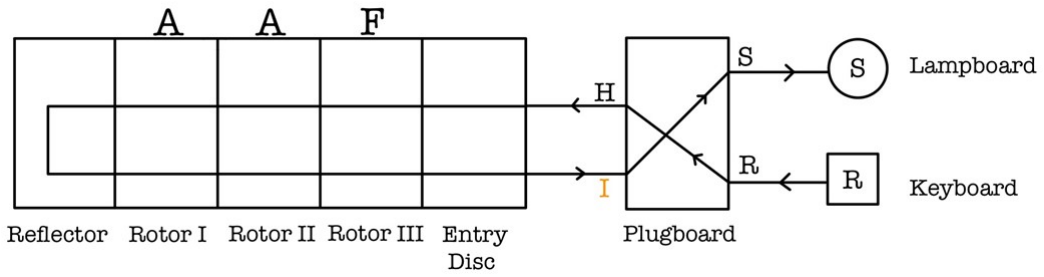


Figure 38: Deduction 3

We can see that I-S is a new plugboard setting, given the assumptions above, which is our third deduction.

Deduction 4

The fourth deduction will be based on the seventh pair of letters. In this case, the cipher letter is L and the crib letter is S. We know that for this pair, the wheel setting is AAG. We have the following situation:

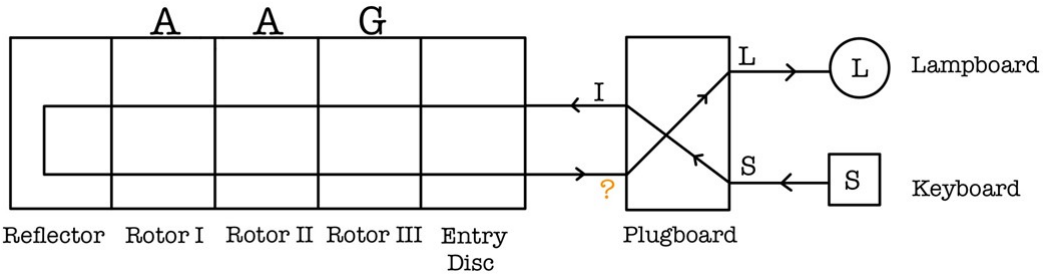


Figure 39: Deduction 4 with an unknown

At this point, the cryptographers were able to figure out whether the assumptions above were correct. The same procedure as before could be executed. The replica Enigma would be set in the given settings without a plugboard. In this case, the letter lighting up will tell if the assumptions are correct. This is the case if the letter B would light up. We assumed plugboard setting B-L, so if the letter B lights up, this is not in contradiction with the assumptions. In the case that B does not light up, it is in contradiction with the assumptions. In this case, this set of assumptions can be ruled out and the next set of assumptions will be tested.

End example

As presented in the example, this is how the cryptographers were able to test whether a set of assumptions was correct or not. The Bombe executed a similar procedure, but faster. It existed out of replica Enigma machines which were wired together. However, the plugboards were left out. The machines were wired in such a way that if a letter came out of one of the Enigma machines, it also went into the next Enigma machine as this letter. The wheel were then set in the starting positions, corresponding to the letters out of the chain, which is the following:

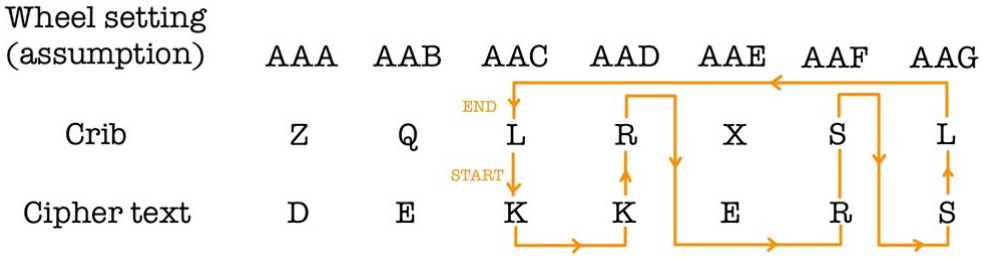


Figure 40: The chain of letters

So, as mentioned before, the distance between these positions had to be correct. The assumptions were then tested. After the test was done, the wheels of all Enigma machines that were part of the Bombe, had to be advanced one step to maintain the distance between the positions. Then the assumptions were tested again. This was repeated for all 17,576 possible wheel settings. Afterwards, the settings that were not ruled out by the Bombe, were tested manually. Other than shown in the picture above, cryptographers would visualize the chain differently:

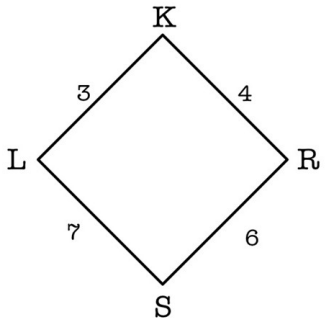


Figure 41: Chain of letters as visualized by cryptographers

They referred to this as a **menu**. This was used to display the way the wheels of the Bombe had to be set. Two letters are connected with each other in the menu if the Bombe can make a deduction based on the sockets to which these two letters of the menu are connected to. The numbers in the menu represent the relative position in which the wheels are set. AAA is relative position 1, so 3, for example, represents relative position AAC.

However, the Bombe only worked when the cribs led to **closures**. Closures are chains that have the same starting and ending letter. When using the Bombe, longer cribs were required than used for the example above, but some did not break the cipher. So, Gordon Welchman came up with a alteration for the Bombe to make it make more deductions given a cipher text and crib. Also, the cribs had not be as long as before.

The alteration of the Bombe made use of the fact that plugboard connections were two way connections. It was not the case that if you had the plugboard connection A-B, that the current only could flow from A to B. In fact, it could also flow from B to A. So if you had found a new plugboard connection due to a deduction, for both letters of this connection you could look to see if they appear in your cipher text or crib. If both do appear in either one, based on this you could make two other deductions, instead of one as we have shown in the example above.

After Welchman suggested the alteration and it was applied, Turing also came up with an idea to alter the new version of the Bombe. The procedure he came up with was referred to as **simultaneous scanning**. Where beforehand one had to run two separate tests to see if A was connected to B or C on the plugboard, with simultaneous scanning it could be tested whether A was connected to any letter on the plugboard in one test. The test even kept going when a set of assumptions was not correct. If the Bombe found that for instance A was connected to many other letters on the plugboard, this suggested an incorrect wheel order or wheel setting. This ruled out many sets of assumptions at the same time, which made this new procedure very important. It could also occur that the Bombe indicated that a certain letter, say J, was plugboard connected to every other letter except for one letter. This suggested that the wheels were correctly set, but the assumed plugboard connection was incorrect. What also could be assumed was that one point which according to the Bombe did not connect to J on the plugboard, was in fact the letter to which J was connected on the plugboard. It was also possible that the Bombe indicated that J was connected to just one other letter on the plugboard. This indicated that the assumptions about the plugboard connection, wheel order and settings were correct.

Source: [26]

F Naval Enigma

In April 1940, the British found a document explaining the indicating system of Naval Enigma. However, it took the codebreakers up until March 1941 to master the entire system. To break the cipher, the bigram tables had to be reconstructed, which made sure that the Banburismus procedure could be performed. This method was the preferred method to break Home Waters Enigma until July 1943.

To send a message with the Naval Enigma cipher, the sender had to take a couple of steps. First, the person encrypting the messages had to pick two three letter groups, also known as **trigrams**, out of a book. This book was called the Kenngruppenbuch. Say that the sender picked CRI and UEY. After that, the trigrams were extended with an extra letter. The first trigram got one extra letter in the beginning, say H, and the second trigram got a letter added at the end, say D. This resulted in the following:

H C R I

U E Y D

The second four letter group was also known as **Verfahrenkenngruppe**.

Second, the wheels would be put in the initial position which was given in the setting list. Say this was JKL. The sender of the message then took the first three letters of the second four-letter group, in this case UEY, and typed this out on the Enigma machine. Say this lit up HCB. Then HCB would be the message setting.

Third, the Enigma machine wheels would be put into the message setting, in this case HCB, and would encipher the message. Before communicating the enciphered message, the sender had to make it clear to the receiver how to decrypt the message without the enemy finding out. To do this, the four vertical letter pairs, also referred to as bigrams, mentioned above had to be transformed. So, the first of the bigrams above were transformed. How these bigrams needed to be transformed, was described by the bigram tables that the German cryptographers had access to. So, the following

H C R I

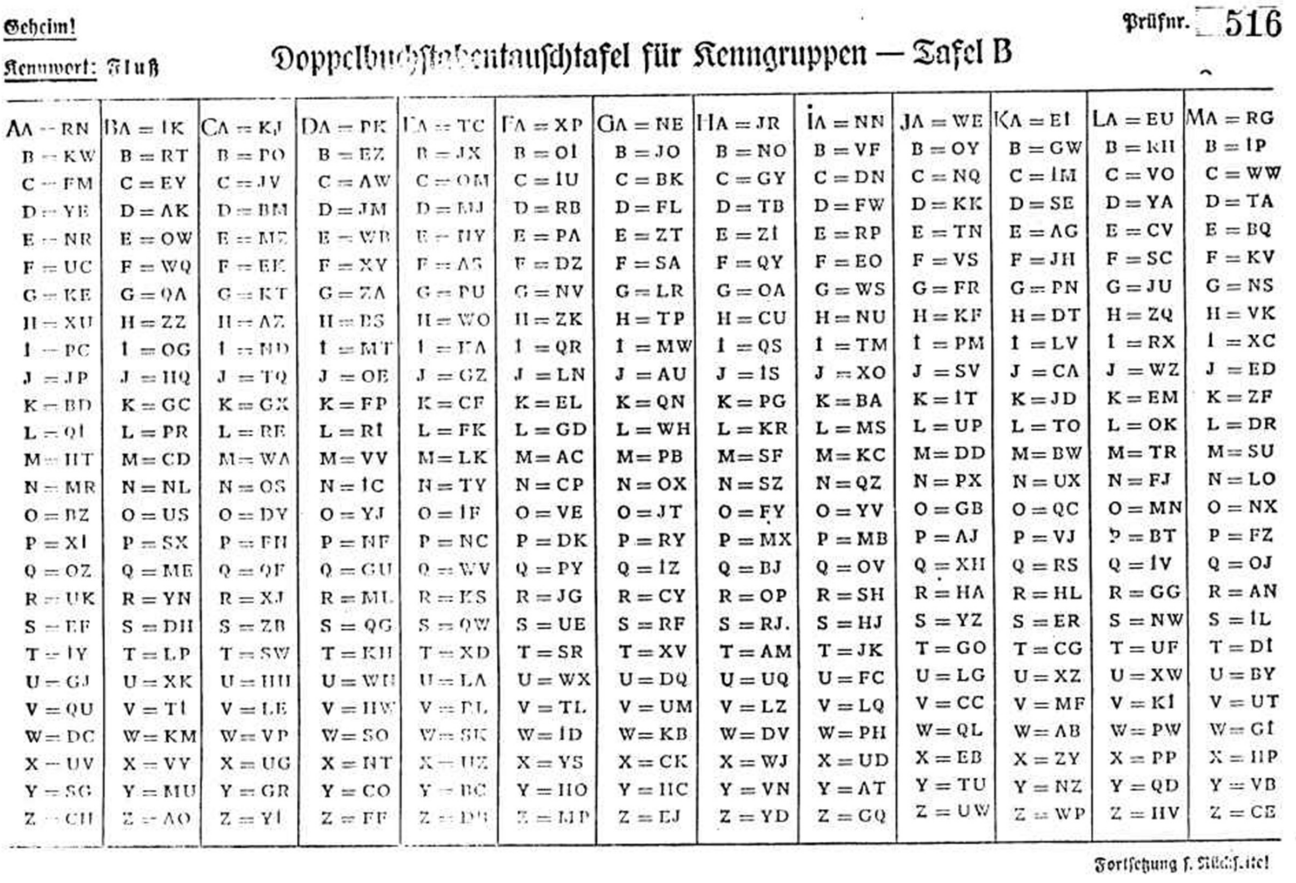
U E Y D

became for instance this:

U M G F

Q Z P W

with the following table:



Geheim! Prüfnr. 516

Kennwort: Fluß

Doppelbuchstabentauschtafel für Kenngruppen — Tafel B

AA = RN	BA = IK	CA = KJ	DA = PK	EA = TC	FA = XP	GA = NE	HA = JR	IA = NN	JA = WE	KA = EI	LA = EU	MA = RG
B = KW	B = RT	B = PO	B = EZ	B = JX	B = OI	B = JO	B = NO	B = VF	B = OY	B = GW	B = KH	B = IP
C = FM	C = EY	C = JV	C = AW	C = QM	C = IU	C = BK	C = GY	C = DN	C = NQ	C = IM	C = VO	C = WW
D = YE	D = AK	D = BM	D = JM	D = MJ	D = RB	D = FL	D = TB	D = FW	D = KK	D = SE	D = YA	D = TA
E = NR	E = OW	E = MZ	E = WB	E = HY	E = PA	E = ZT	E = ZI	E = RP	E = TN	E = AG	E = CV	E = BQ
F = UC	F = WQ	F = EK	F = XY	F = AS	F = DZ	F = SA	F = QY	F = EO	F = VS	F = JH	F = SC	F = KV
G = KE	G = QA	G = KT	G = ZA	G = PU	G = NV	G = LR	G = OA	G = WS	G = FR	G = PN	G = JU	G = NS
H = XU	H = ZZ	H = AZ	H = BS	H = WO	H = ZK	H = TP	H = CU	H = NU	H = KF	H = DT	H = ZQ	H = VK
I = PC	I = OG	I = HD	I = MT	I = IA	I = QR	I = MW	I = QS	I = TM	I = PM	I = LV	I = RX	I = XC
J = JP	J = HQ	J = TQ	J = OE	J = GZ	J = LN	J = AU	J = IS	J = XO	J = SV	J = CA	J = WZ	J = ED
K = BD	K = GC	K = GX	K = FP	K = CF	K = EL	K = QN	K = PG	K = BA	K = IT	K = JD	K = EM	K = ZF
L = QI	L = PR	L = RE	L = RI	L = FK	L = GD	L = WH	L = KR	L = MS	L = UP	L = TO	L = OK	L = DR
M = HT	M = CD	M = WA	M = VV	M = LK	M = AC	M = PB	M = SF	M = KC	M = DD	M = BW	M = TR	M = SU
N = MR	N = NL	N = OS	N = IC	N = TY	N = CP	N = OX	N = SZ	N = QZ	N = PX	N = UX	N = FJ	N = LO
O = BZ	O = US	O = DY	O = YJ	O = IF	O = VE	O = JT	O = FY	O = YV	O = GB	O = QC	O = MN	O = NX
P = XI	P = SX	P = FH	P = NF	P = NC	P = DK	P = RY	P = MX	P = MB	P = AJ	P = VJ	P = BT	P = FZ
Q = OZ	Q = ME	Q = QF	Q = GU	Q = WV	Q = PY	Q = IZ	Q = BJ	Q = OV	Q = XH	Q = RS	Q = IV	Q = OJ
R = UK	R = YN	R = XJ	R = ML	R = ES	R = JG	R = CY	R = OP	R = SH	R = HA	R = HL	R = GG	R = AN
S = EF	S = DH	S = ZB	S = QG	S = QW	S = UE	S = RF	S = RJ.	S = HJ	S = YZ	S = ER	S = NW	S = IL
T = IY	T = LP	T = SW	T = KH	T = XD	T = SR	T = XV	T = AM	T = JK	T = GO	T = CG	T = UF	T = DI
U = GJ	U = XK	U = HH	U = WH	U = LA	U = WX	U = DQ	U = UQ	U = FC	U = LG	U = XZ	U = XW	U = BY
V = QU	V = TI	V = LE	V = HW	V = DL	V = TL	V = UM	V = LZ	V = LQ	V = CC	V = MF	V = KI	V = UT
W = DC	W = KM	W = VP	W = SO	W = SK	W = ID	W = KB	W = DV	W = PH	W = QL	W = AB	W = PW	W = GI
X = UV	X = VY	X = UG	X = NT	X = UZ	X = YS	X = CK	X = WJ	X = UD	X = EB	X = ZY	X = PP	X = HP
Y = SG	Y = MU	Y = GR	Y = CO	Y = BC	Y = HO	Y = HC	Y = VN	Y = AT	Y = TU	Y = NZ	Y = QD	Y = VB
Z = CH	Z = AO	Z = YI	Z = FF	Z = DB	Z = MP	Z = EJ	Z = YD	Z = GQ	Z = UW	Z = WP	Z = HV	Z = CE

Fortsetzung s. Rückseite!

Figure 42: Bigram Tables

Source: https://www.ciphermachinesandcryptology.com/en/enigmaproc.htm

Lastly, the letters were repositioned as follows:

U G Q P

M F Z W

This would be the letters sent at the beginning of a message, so these letters were not enciphered.

To decode the message, the steps were basically carried out backwards; the letters were scrambled back and were transformed with the bigram tables, then the receiver would know the initial position of the wheel, in this case UEY. Then, he could find the message settings, in this case HCB, and decipher the enciphered text.

Reconstruction of the bigram tables

To reconstruct the bigram tables, first the cryptographers needed to know the settings of the day, which could be found out in two ways. The first possibility would be that the settings of that day were captured, the second option was that the cryptographers used cribs and Bombes to break the settings. The Bombe only worked out the plugboard settings and the wheel order. The ring settings, initial position and message setting were worked out manually by the codebreakers.

The reconstruction of the bigram tables was done in multiple steps. These are described below.

1. For simplicity, the same settings as used in the previous example will be used. First, the wheels would be set in the initial position, JKL, and then the message setting, HCB, was tapped out. This would light up UEY. The following was known:

   - - - -

   U E Y -

2. The cryptographers also looked at the indicator at the beginning of the message. They knew to rearrange

   U G Q P

   M F Z W

   to this

   U M G F

   Q Z P W

3. So, there were three entries that were partially known:

   ?U & UQ / ?E & MZ / ?Y & GP

   If other bigrams or partial bigrams were available, the missing information could be filled in. This was also made possible because the bigrams were reversible. It was important that these tables were reconstructed, as these were necessary to carry out the Banburismus procedure.

But for the bigram tables to be reconstructed, all the settings had to be known. This also included the message setting, which was not found by the Bombe and therefore had to be found manually. To do this, the codebreakers used something called the **Eins Catalogue**.

The catalogue consisted of $17,576$ ($= 26 \times 26 \times 26$) times the word EINS enciphered in each wheel position. These were all alphabetically ordered in a catalogue. So, say EINS enciphers as MAUD with wheels set to JKL. If you would see MAUD in an enciphered text, it could be possible that the sender tapped EINS with the wheels set to JKL.

So, for every four-letter group the cryptographers had to see if it appeared in the catalogue. For example, for the enciphered text LMAUDYHJ, first LMAU was looked up in the catalogue. If the cryptographers could not find this, they would check the next four letters: MAUD. MAUD did appear in the catalogue with wheel settings JKL. To produce the M the wheels had to be set to JKL. So, to produce the L, the wheels had to be set to JKK. Afterwards, the cryptographers checked if this was the right setting, by setting the wheels to JKK and typing out the message to see if German plain text came out.

Banburismus

The Banburismus procedure was invoked to rule out wheel orders. There were 336 possible wheel orders, and it took the Bombe a lot of time to go through all these options. It would be way faster if certain orders were already ruled out and speed was important when not many Bombes were available.

In the first stage of the Banburismus procedure, the distance between message settings had to be determined. This worked as follows:

A message always started with eight letters that were part of the indicating system. To start, the cryptographers chose two messages and paired them up if they had the first two, or at least the first letter op the Verfahrenkenngruppe in common. So, for example a message with Verfahrenkenngruppe HPA and a message with Verfahrenkenngruppe HPC were paired up. The cryptographers did not know the message settings of these messages yet. However, if two Verfahrenkenngruppe had the two first letters in common, the message settings also had the first two letters in common. This was the case because HP had to be enciphered in the initial position to find the message setting and the initial position was the same for everyone. So, HP would in both cases be enciphered to the same combination of letters. Because of this, the cryptographers also knew the third letter of the Verfahrenkenngruppe was different as these were also decrypted in the same settings and two different letters could never encrypt to the same letter. If this were the case, this would be in contradiction with the mechanism of Enigma, as the letters in the Enigma machine were basically paired. What the cryptographers did know, was that the distance between the two third letters could not be more than twenty-five.

To find the distance between these two third letters, the codebreakers first investigated the first enciphered text. For every letter, they punched a hole in something called a **Banbury Sheet**. This sheet had the alphabet on the left vertical axis. If V was the first letter of the cipher text, in the first column a hole was punched in row V. Say the second letter was K, then a hole was punched in the second column in row K. The cryptographers did the same thing for the cipher text of the second message, but on a different Banbury Sheet.

After the two sheets were finished, they were laid on top of each other. The sheet of the first message on the bottom and the sheet of the second message on top. Now, the starting positions of the two messages were on top of each other. Then, the top sheet was slid to the right one step at the time. For every step the sheet was slid, a test was performed that will be explained later. Codebreakers did this until they could carry out the test at each position. The positions they tested were twenty-five places to the left and right.

For every position, the cryptographers would write down the amount of lined up pairs. There was a pair if the letter on top was the same as the letter underneath. This was easy to see as two holes in the sheets would align in this case. If there were a lot of pairs when the sheets were in a certain position, this could mean that the matching plain text was enciphered in the same position. Because when you type out lines on Enigma machines that have the same wheel settings, these cipher texts tend to have more letters in common compared to when the wheel settings are different. When the sheets had many letters in common, the message would be lined up '**in depth**'.

But finding holes that lined up, was not enough to conclude that two messages were in depth. Therefore, Turing introduced a certain scoring system. If, for instance, five consecutive holes lined up, this scored more points than when you had fewer consecutive holes lined up. Also, the higher the percentage of holes of the columns lining up, the higher the score.

So, when they found the position with the best score, the distance could be written down for every pair of messages. This gave the distance between the third letters for each pair of message settings there was. The cryptographers could draw a chain of this. For example, say the message settings with Verfahrenkenngruppe HPA and HPC had a distance of two. Then we would get A.C, where the dot indicates the distance between these third two letters. Say the message settings for Verfahrenkenngruppe JKC and JKJ was eleven places, you would get A.C.........J . This might be extended by other pairs the cryptographers found, to U....Y....A.C.........J..U for example.

The second stage of the procedure was to determine the message settings based on the distance between the message settings. To do this, cryptographers would make an assumption. Say that, for example, A was the third letter of the message setting and the third letter of the corresponding Verfahrenkenngruppe was U. In the example, U is the first letter of the chain. This would result in the following:

Third letter Verfahrenkenngruppe	U*					Y					A		C									J			
Third letter message setting	A*	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Then, the following was assumed. Say the Verfahrenkenngruppe was PTU. If the U was tapped, the wheel turned two places from the initial position, also referred to as +2 from initial position, and this lit up A. This would be the third letter of the message setting.

Because of how Enigma works, you have pairs, and you can fill in some information, which is underlined in the next table.

Third letter Verfahrenkenngruppe	K/U*			M			Y				X	A		C								A				J	F	
Third letter message setting	A*	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		

But looking at the letters, the assumption that was marked with a star could not be correct, as A has two letters paired with it which is not possible.

When one assumption did not work, another one was tested. Say the same situation occurred, but it was assumed that B lit up instead of A. If this was incorrect, the next letters of the alphabet were tried until the cryptographers tried them all. Say they all gave a contradiction except for one and, for example, G lit up instead of A. The chain U....Y....A.C.....J..U would then be processed as follows:

Third letter Verfahrenkenngruppe	Q		S	J			U			D		Y					A		C		G					L		
Third letter message setting	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		

Say, the cryptographers also found other information from other pairs of messages:

Third letter Verfahrenkenngruppe	Q		S	J			U			D		Y					A	T	C	R	G					L		
Third letter message setting	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		

In the third, and final, stage of the Banburismus procedure, the cryptographers would try to find out which wheels could be ruled out based on the turnover position of each wheel.

The information in the table above was also used to find out which one of the eight wheels was used in the wheel positions most right in the Enigma machine. To find this out, the cryptographers had to look at the turnover position of each of the Enigma wheels, which would be the letter on the ring showing through the window for the neighboring wheel on its left to turn over. The following table includes this information:

Third letter Verfahrenkenngruppe	Q		S	J			U			D		Y					A	T	C	R	G					L		
Third letter message setting	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		
					2					4			6				1					3				5		
													7													6		
													8													7		
																										8		

When we look at the chain we formed above, the J was eleven places from C. Based on this, the codebreakers could conclude that the turnover induced by the right wheel did not happen. In these eleven spaces, the middle letter of the Verfahrenkenngruppe did not change. So, between S and D there was no turnover of the middle wheel. Therefore, wheels 3, 5, 6, 7 and 8 could not have been in the position on the right. With the same principle, other wheels could be ruled out. The same could be done for the wheel in de middle, which were able to turnover the wheel on the left.

In this way, many wheel orders could be ruled out. Besides, cryptographers always knew one of the wheels 6, 7 or 8 was used. So, if you knew ?/2/3, then the question mark always had to be 6, 7 or 8.

Now, the Bombe could figure out the correct wheel order, because of the information in the table directly above. This was as useful as finding a crib and working this out, the only difference was that in table directly above no German plain text was represented in neither of the two lines. However, this did not matter for the Bombe. It only needed what letter lit up when you tapped a certain key.

Source: [26]

G Cillis

During the war, the Germans made many mistakes when it came to Enigma. Some of these mistakes were called **cillis**. Cillis helped the cryptographers break Air Force Enigma on 22 May 1940 and the time after this date.

For now, we refer to the letters showing through the windows when the first letter of a message was enciphered, as the message setting. We refer to the letters showing through the windows when the message setting was enciphered, as the initial setting, even though the positions of the wheel cores were not known yet.

German messages were not allowed to be longer than 250 words. Therefore, if there was a message that was longer than this maximum amount of words, it was mandatory to it split up in several parts. At one point, it started to stand out that, say IBL was the setting of the rotors when the first part of the message was enciphered, IBL was used as the initial position for the next part of the message. So, the cryptographers only had to look at the initial position that was used for the second part of the message, which was stated unenciphered at the beginning of the message, to know in what setting the last letter of the prior part was enciphered. Then, they could try to work out the message setting used for the first part of the message.

So, to find out the message setting, all the cryptographers had to do was to turn the wheels x places back, with x being the number of letters of the message. However, the wheels and wheel order had a substantial impact on this, given that the turnover positions were different for most wheels. Therefore, the message setting could not be exactly determined, only a selection of possibilities could be made.

To find the correct message setting, the Germans helped the cryptographers by making another mistake. Knox noticed that the Germans were using message settings that were obvious or lazy. They for instance broke up famous sayings or used the diagonals on the keyboard, for example TGB or UJM. Choosing a message setting like this was stated to be forbidden.

Say that the cryptographers would look at the possible message settings and found that TGB was one of the possible message settings for the first message. Then they would look at the rotors who could make it possible to go x places back to get TGB. This, for instance, could only be the case when rotor 1 was in the middle and rotor 3 was on the right. The cryptographers might also have found that UJM was a possible message setting for the second part of the message and could be established with rotors 1 and 3 on the same places as mentioned before. Now the cryptographers were able to guess that the German sender was using the diagonals of the Enigma keyboard. After that, the last rotor was found by trial and error.

So, cillis were used to find the wheel order and message settings. The name cillis is a combination of CIL, the first message setting found this way, and the word silly, which is what the cryptographers thought of the mistakes the Germans made.

Source: [26]