Lösungsblatt

# Aufgabe 5: Kommunikation – Morse-Code – Verschlüsselung

Bei der Lösung der Aufgaben lernen die Schülerinnen und Schüler, wie bei einer Datenübertragung Sender und Empfänger synchronisiert und Daten für die Über­tragung auf einem Kommunikationskanal kodiert und verschlüsselt werden. Dabei erlernen sie zugleich den Umgang mit Strings und indizierten Listen.

Die Schülerinnen und Schüler erhalten neben den Bauanleitungen das Programm-Fragment mit den Alphabeten („Morse\_Code\_Template.ft“).

## Konstruktionsaufgabe

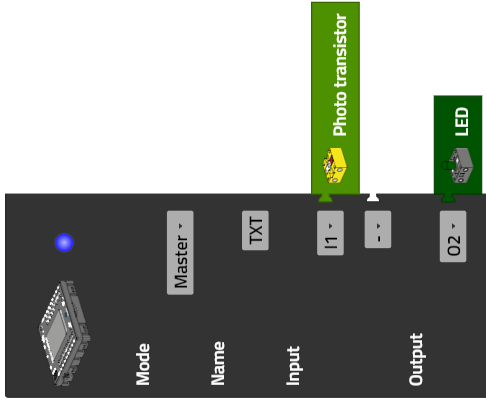
Siehe Bauanleitung.

Die Sende-LED wird über den Morse-Taster mit dem 9V-Ausgang und einem GND-Anschluss des TXT verbunden; der Morse-Taster benötigt den TXT nur als Strom­quelle. Die Empfangs-LED wird an O2 und der Fototransistor an I1 ange­schlossen.

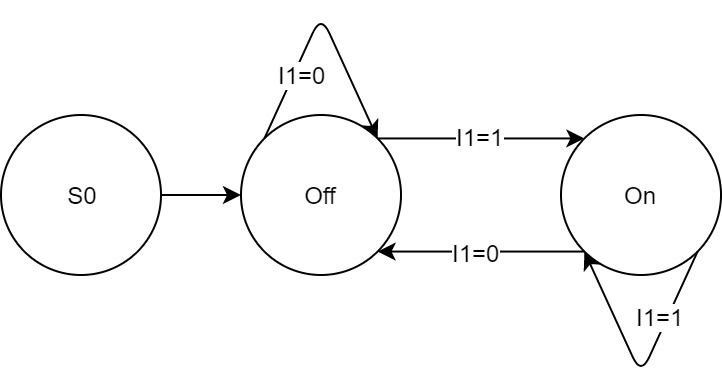
## Programmieraufgaben

**1. Lichtsignale**

Anschlusskonfiguration:

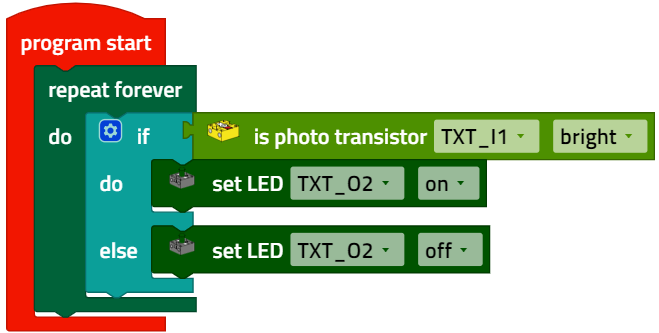


1a. Zustandsübergangsdiagramm des Empfängers:



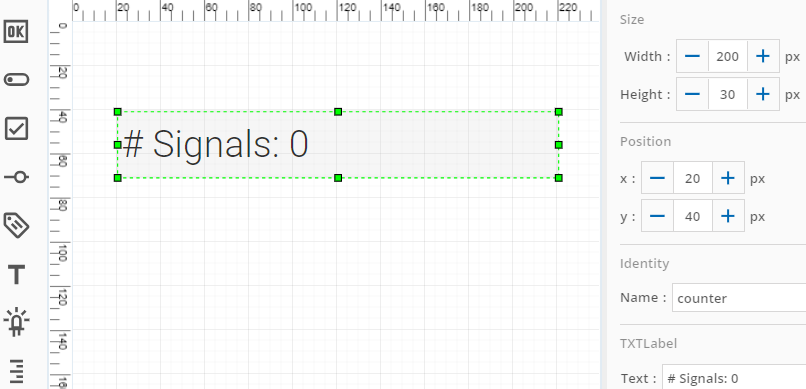
*State-Transition\_Diagram\_Light\_Signal.drawio*

1b. Programm (Beispiel) Empfänger:

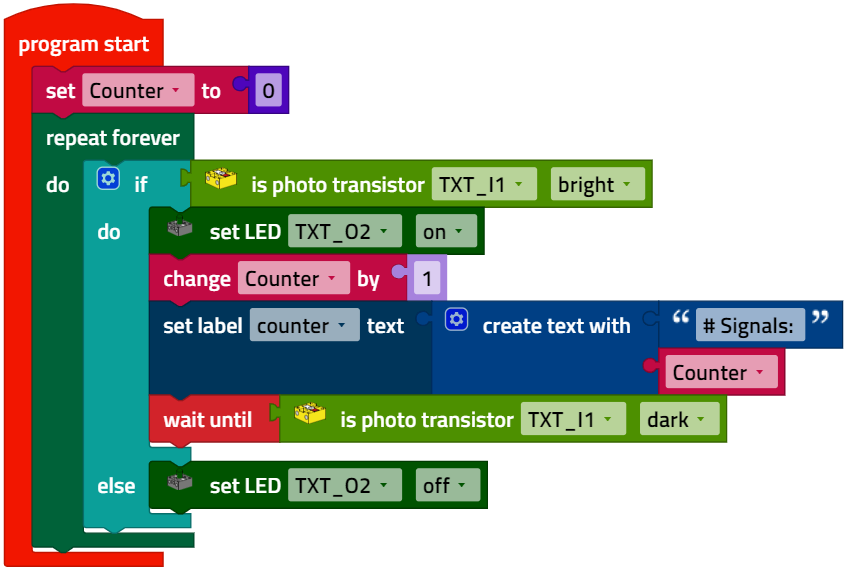


*Light\_Signal.ft*

1c. Konfiguration des Displays:



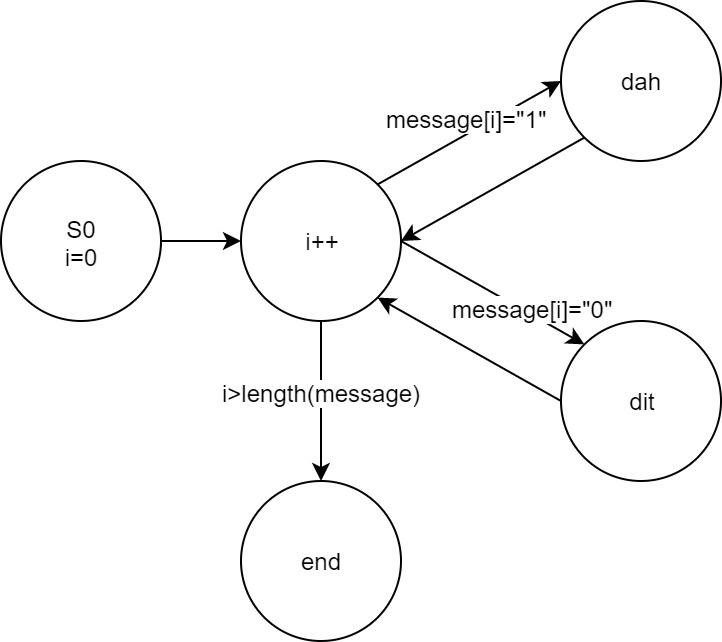
Programm (Beispiel) Empfänger mit Zähler:



*Light\_Signal\_Counter.ft*

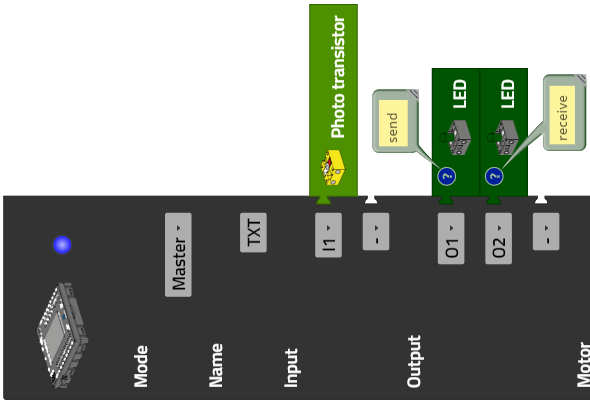
**2. Binärcode-Sender**

2a. Zustandsübergangsdiagramm des Bitfolgen-Senders:

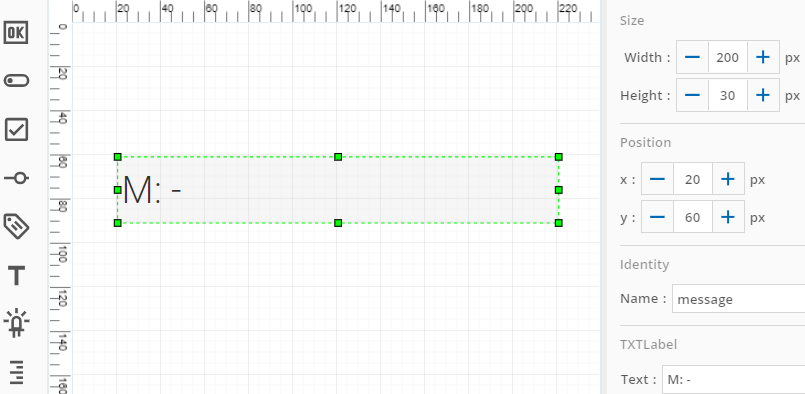


*State-Transition\_Diagram\_Bit\_Sequence\_Sender.drawio*

2b. Anschlusskonfiguration der Sende-LED:



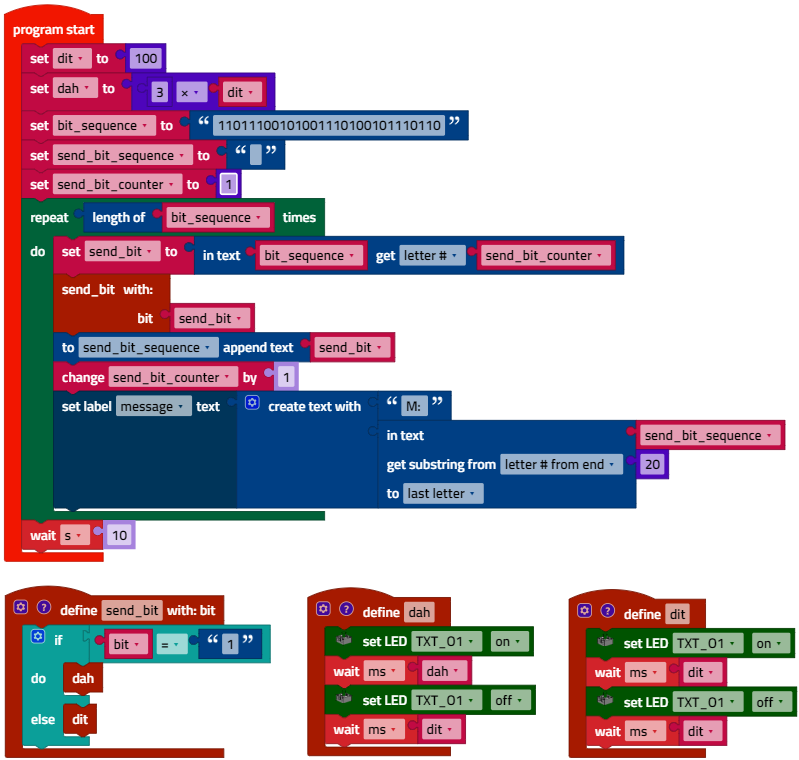
Konfiguration des Display:



Von der gesendeten Bitfolge werden die letzten 20 Zeichen auf dem Display ausgegeben.

Für die Lösung der Aufgabe empfiehlt es sich, die Text-Funktion zur Auswertung des Nachrichten-Textes zu verwenden, sowie Teile des Programms in Funktionen auszu­gliedern, damit das gesamte Programm übersichtlich bleibt.

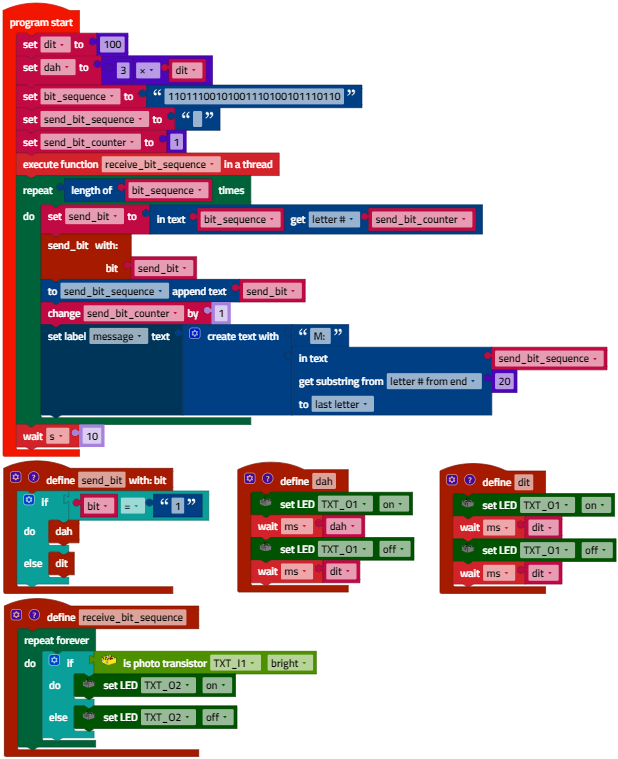
Programm (Beispiel) Bitfolgen-Sender:



*Bit\_Sequence\_Sender.ft*

2c. Die Empfangsroutine („receive\_bit\_sequence“) wird als nebenläufiger Prozess (Thread) gestartet. Sie kann in den Experimentieraufgaben um einen Dekodierer erweitert werden.

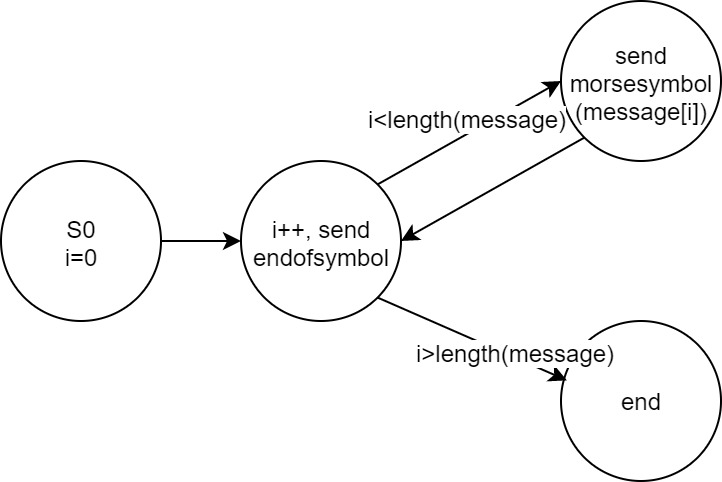
Programm (Beispiel) Bitfolgen-Sender mit Empfangsanzeige (LED):



*Bit\_Sequence\_Sender\_Receive\_LED.ft*

**3. Morse-Zeichen-Sender**

3a. Zustandsübergangsdiagramm des Morsecode-Senders:



*State-Transition\_Diagram\_Morse\_Code\_Sender.drawio*

Das Senden eines Zeichens der Nachricht entspricht dabei dem Senden eines Morse-Symbols – einer „0“/„1“-Folge wie beim Bitfolgen-Sender aus Teilaufgabe 2a.

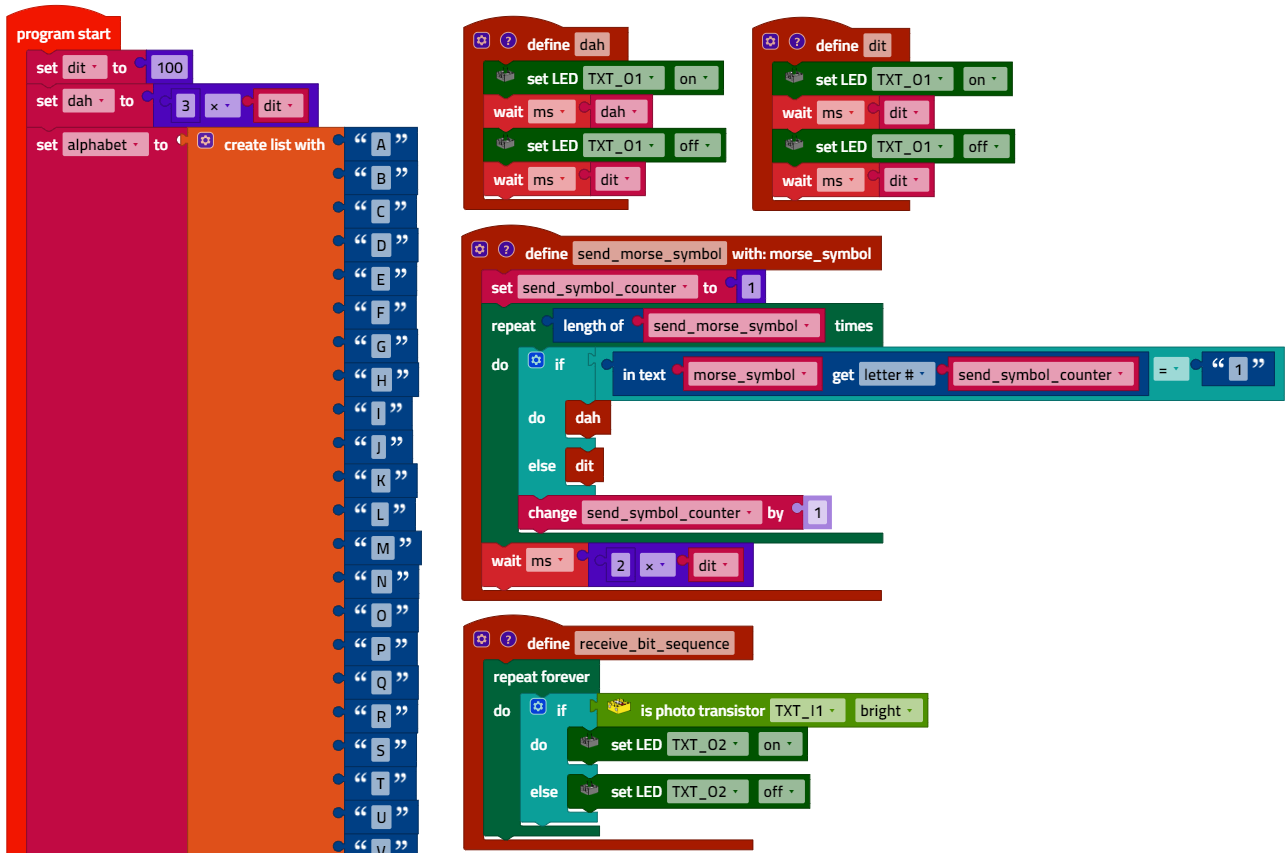
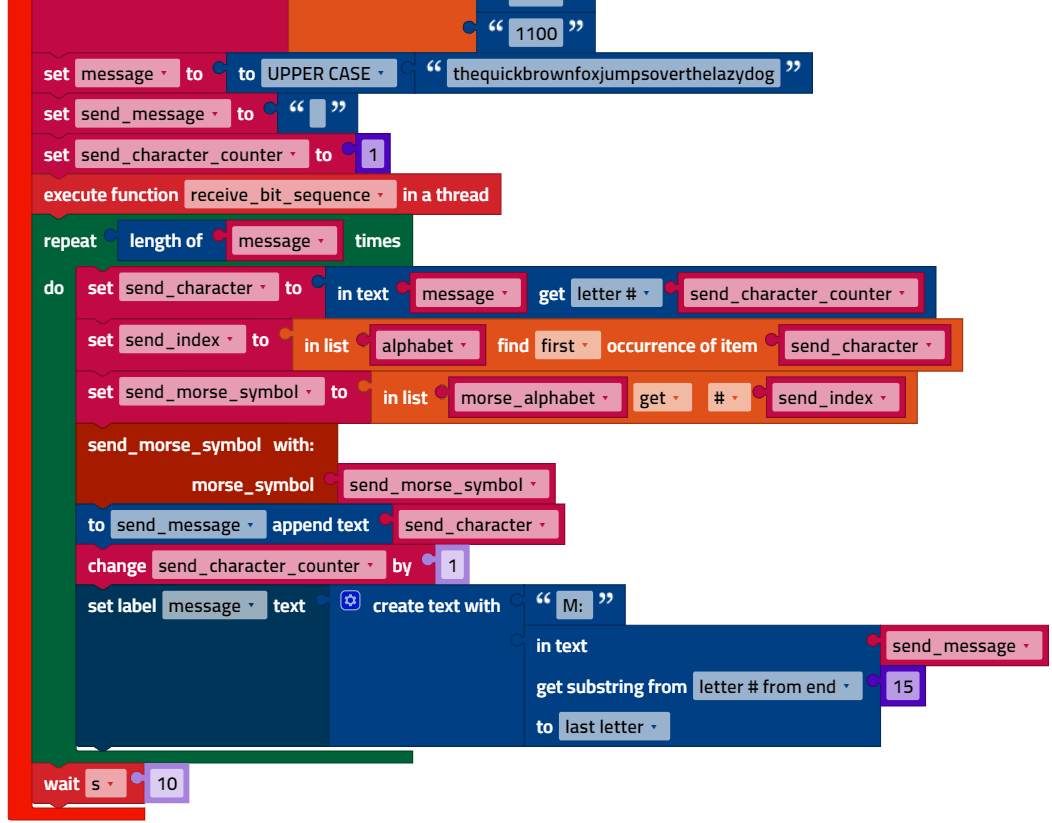
3b. Über die Listen-Funktionen kann der Index eines Listenelements bestimmt und ein Element mit einem bestimmten Index ausgelesen werden. Damit kann die Kodierung des Textes sehr elegant vorgenommen werden.

**Hinweis**: Das Morse-Alphabet unterscheidet nicht zwischen Klein- und Großbuch­staben. Daher muss der Nachrichtentext vor der Kodierung in Großbuchstaben umgewandelt werden.

**Lösungsvariante**: Die Nachricht kann auch über das Eingabefeld des TXT-Touch-Displays eingetippt werden. Im folgenden Lösungsbeispiel wird sie als Variable im Programmcode vorgegeben.

(Die Initialisierung der beiden Alphabete ist in der Abbildung verkürzt widergegeben.)

Programm (Beispiel):

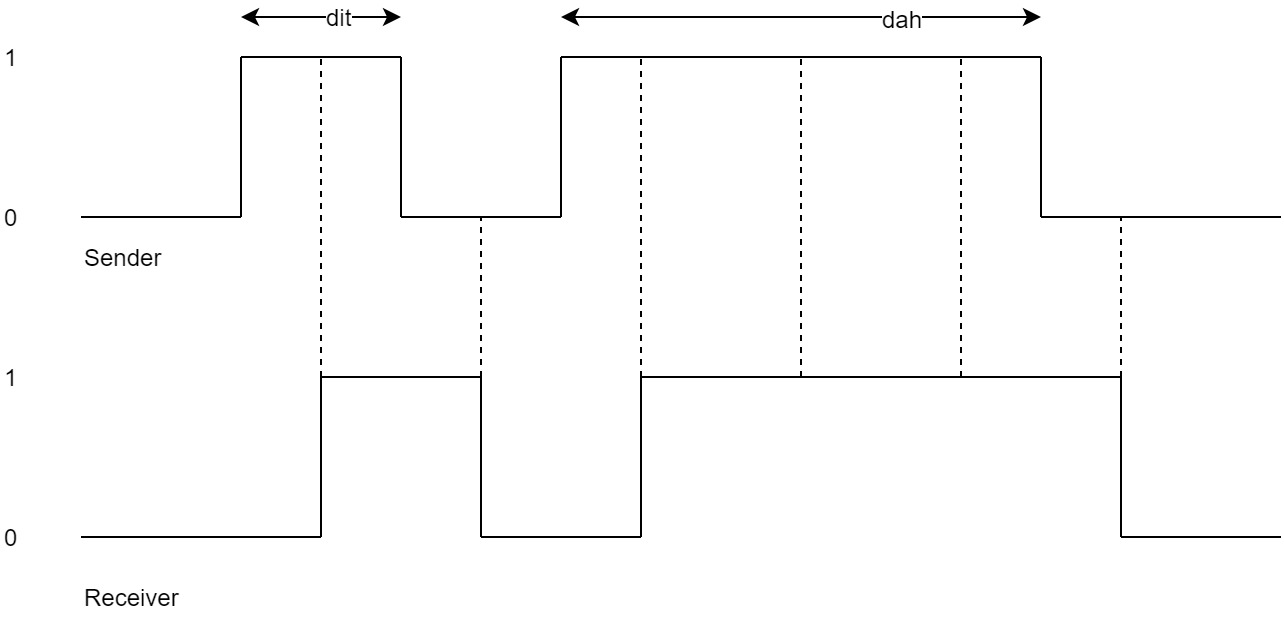
  
…  


*Morse\_Code\_Sender.ft*

## Experimentieraufgaben

**1. Bitfolgen-Empfänger**

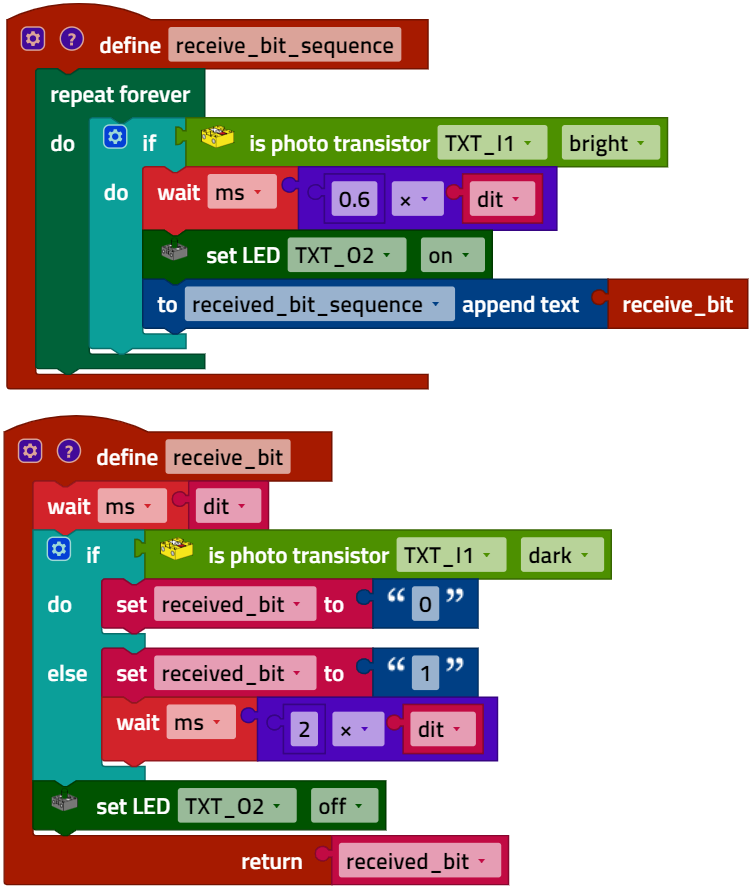
Der Empfänger-Prozess wird vom Fototransistor „getriggert“. Anschließend wartet der Prozess etwa eine halbe Signallänge (0,5 x „dit“), um das Signal anschließend etwa in der Mitte der Signaldauer abzutasten (siehe Skizze).



*Morse\_Signal\_Sampling.drawio*

Im Hauptprogramm wird statt der gesendeten die empfangene Bitfolge („received\_bit\_sequence“) auf dem Display des TXT ausgegeben.

1a. Programmauszug (Beispiel):



*Bit\_Sequence\_Sender\_Receiver.ft*

1b. Die Übermittlung ist auch noch mit einer Länge von „dit“ = 90 ms fehlerfrei möglich. Das entspricht einer Geschwindigkeit von etwa 11 bit/s.

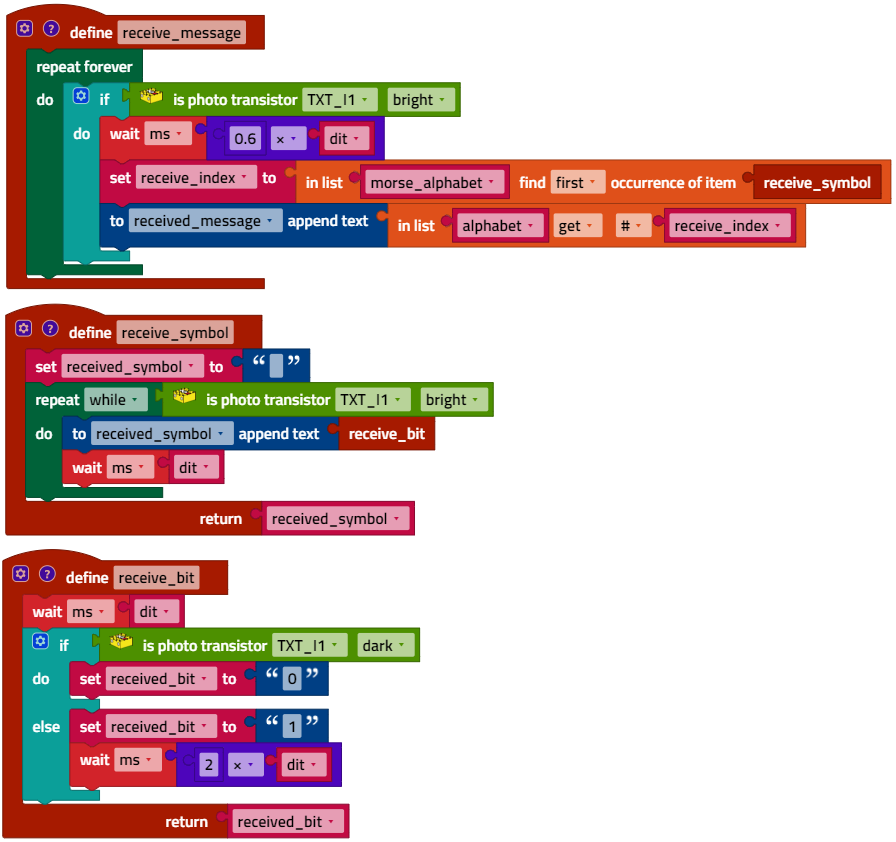
Zum Vergleich: Ein DSL-Anschluss erreicht heute 100 Mbit/s, das ist etwa die 9.000.000-fache Geschwindigkeit.

**2. Morse-Zeichen-Empfänger**

Der Empfang wurde im folgenden Beispielprogramm in drei Teile unterteilt:

* der Empfang eines einzelnen Bit (wie in der Experimentieraufgabe 1)
* der Empfang eines Morse-Zeichens
* der Empfang einer Nachricht (gestartet als Thread)

2a. Programmauszug (Beispiel):



*Morse\_Code\_Sender\_Receiver.ft*

2c. Der Empfang funktioniert im Beispielprogramm noch mit „dit“ = 70 ms zuverlässig. Damit liegt die Übertragungsgeschwindigkeit bei maximal etwa 14 „dit“ pro Sekunde.

2d. Das Referenz-Wort „Paris“ besteht aus 50 „dit“. Somit erreicht unser Morse-Sender für „dit“ = 100 ms 12 WpM. Das entspricht der Morsegeschwindigkeit, die Funkama­teure in ihrer Prüfung nachweisen müssen. Der Rekord liegt bei 88 WpM.

**4. Verschlüsselung – Caesar-Chiffre**

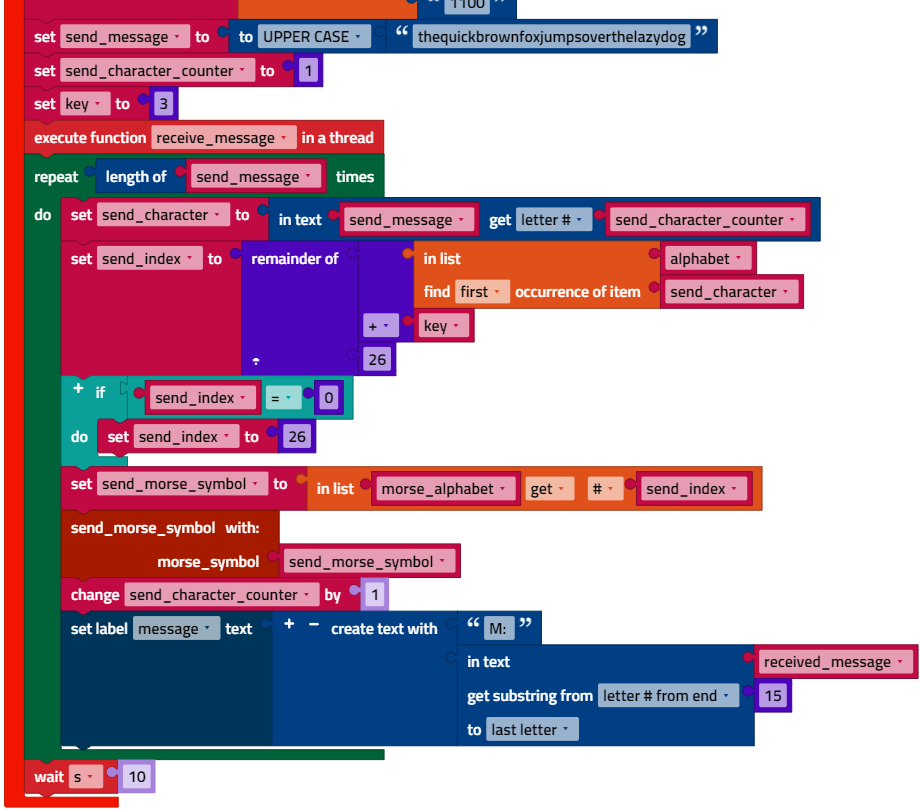
Die Verschlüsselung mit der Caesar-Chiffre lässt sich sehr leicht programmieren, indem die Zeichen des Listenelements „Morse-Alphabet“ um die entsprechende Anzahl Zeichen „verschoben“ werden: beim Verschlüsseln um +3 Zeichen, beim Entschlüsseln um -3 Zeichen.

Achtung: Der Index der Listenelemente des Alphabets und des Morse-Alphabets läuft von 1 bis zur Anzahl der Elemente (also 26). Nach der Addition bzw. Subtraktion des Schlüssels muss der Rest der Division durch 26 bestimmt werden. Ist das Ergebnis (der Index) = 0, muss der Index auf den Maximalwert (26) gesetzt werden.

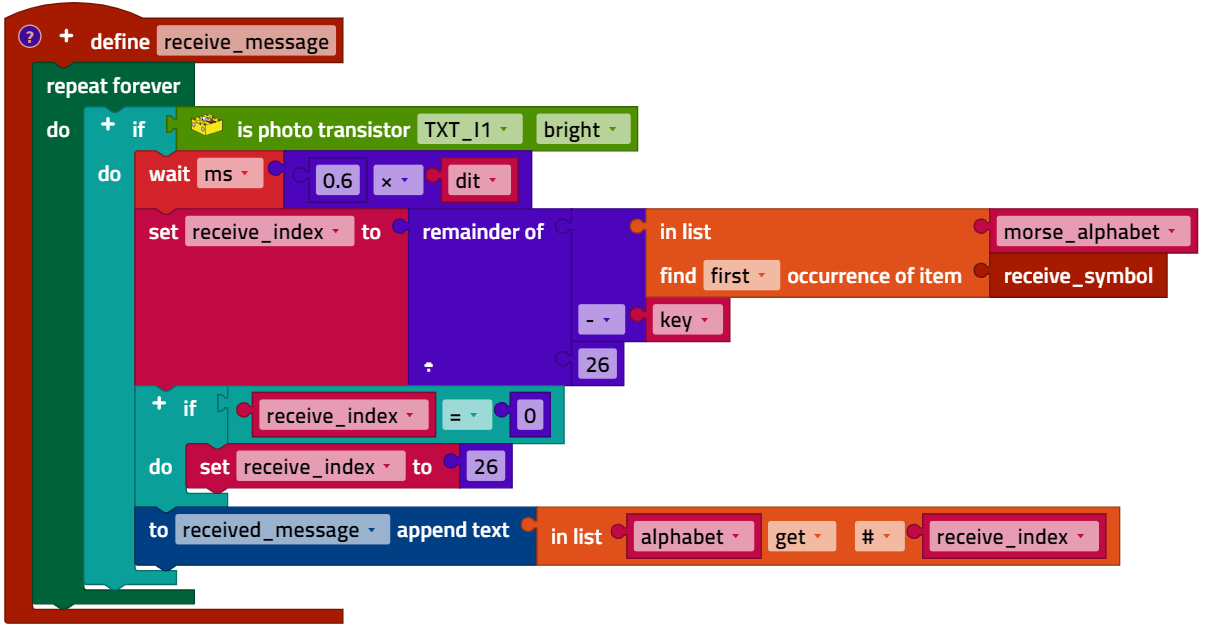
Lösungsvariante: Der Schlüssel (die Zahl der Buchstaben, um die das Alphabet „verschoben“ wird) kann auch variabel über eine Eingabe auf dem Display gewählt werden.

Programmauszüge (Beispiel):

Verschlüsselung:

…  


Entschlüsselung:

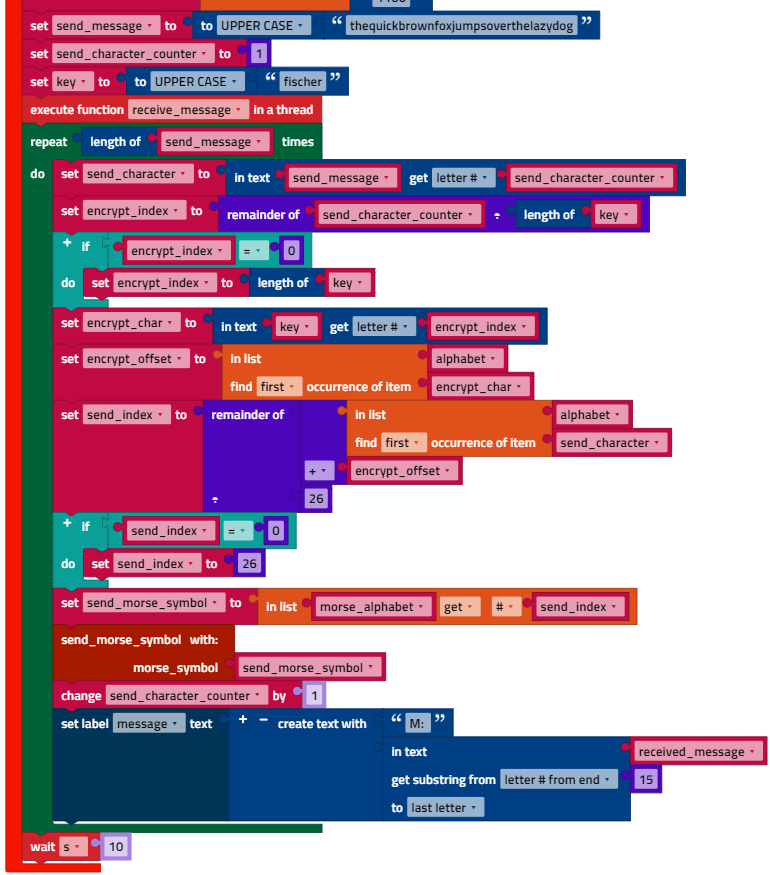


*Morse\_Code\_Sender\_Receiver\_with\_Caesar\_Encryption.ft*

**5. Verschlüsselung – Vigenère-Chiffre**

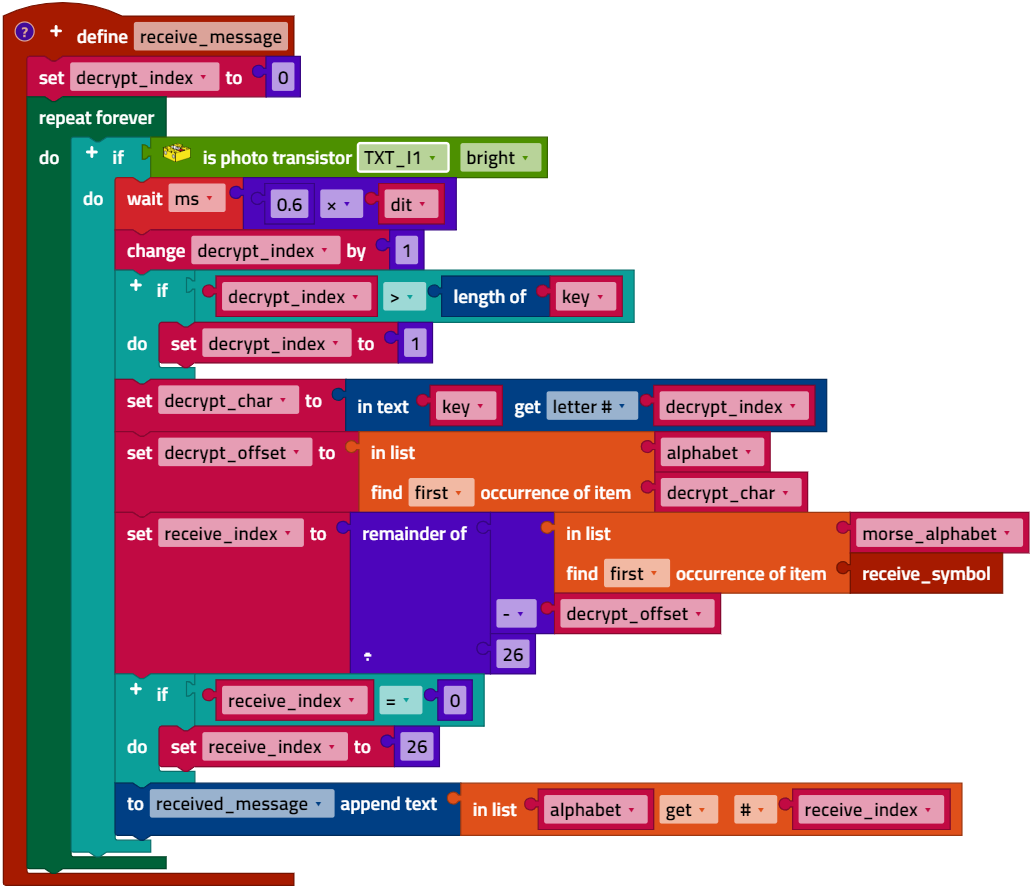
Programmauszüge (Beispiel):

Verschlüsselung:

…  


Für die Entschlüsselung müssen andere Variablen zur Bestimmung des aktuellen Schlüssel-Zeichens verwendet werden, da Ver- und Entschlüsselung nebenläufig erfolgen.

Entschlüsselung:



*Morse\_Code\_Sender\_Receiver\_with\_Vigenere\_Encryption.ft*

Anlagen

# Aufgabe 5: Kommunikation – Morse-Code – Verschlüsselung

## Erforderliches Material

* PC für Programmentwicklung, lokal oder über Web-Schnittstelle.
* USB-Kabel oder BLE- bzw. WLAN-Verbindung für die Übertragung des Programms auf den TXT4.0.
* Programm-Template (für Morsecode): Morse\_Code\_Template.ft

## Weiterführende Informationen

[1] Albrecht Beutelspacher: *Kryptologie: Eine Einführung in die Wissenschaft vom Verschlüsseln, Verbergen und Verheimlichen*. 10. Auflage, Springer Verlag, 2015.

[2] Simon Singh: Codes. *Die Kunst der Verschlüsselung*. Impian, 2021.

[3] Online-Diagrammeditor zur Erstellung von Zustandsübergangsdiagrammen (Format drawio): <https://www.diagrammeditor.de/>