QR-Codes

Bithaufen

QR-Codes verstehen und ohne technische Hilfsmittel dekodieren

QR-Codes enthalten zusätzlich zu den Nutzdaten Informationen, aus denen man fehlende Pixel wiederherstellen kann. Wenn der Code mal so stark beschädigt sein sollte, dass selbst eine App ihn nicht mehr erkennt, dann können Sie versuchen, ihn manuell zu entschlüsseln. Wir zeigen Ihnen, wie das geht.

Von Wilhelm Drehling

kompakt

- QR-Codes folgen einem strikten Aufbau, in dem Informationen bestimmten Arealen zugeteilt werden.
- Die Nachricht sowie viele andere Details kommen mehrfach und teilweise auch maskiert im Code vor.
- Mit dem in diesem Artikel erlangten Wissen können Sie einen beschädigten QR-Code per Hand dekodieren.

Smartphone raus, entsperren und die Kamera-App über einen QR-Code halten. Im Bruchteil einer Sekunde wandelt sich der verwitterte Sticker am Laternenmast in einen Strom von Bits um, welchen das Smartphone in eine lesbare Zeichenkette dekodiert. Bereit zum Antippen ploppt die gescannte Information auf dem Bildschirm auf.

QR-Codes sind ein häufig benutztes Mittel für Außenwerbung, weil sie eine Menge Schaden aushalten. Wenn aber mal der Code

am Laternenmast mehr als 30 Prozent beschädigt ist und technische Hilfsmittel nicht weiterhelfen, dann können Sie versuchen, den QR-Code per Hand zu dekodieren. Kleine Codes mit 20 Zeichen sind problemlos in ein paar Minuten geschafft, bei den wirklich großen sollten Sie es sich zweimal überlegen, ob es sich lohnt. Denn diese können bis zu 2953 herkömmliche Zeichen speichern. In der Theorie lassen sich damit sogar die sämtlichen Web-Tipps von Seite 56 in einen QR-Code packen.

Das ist übrigens nicht das erste Mal, dass wir uns QR-Codes genauer anschauen: Wir haben uns vor einiger Zeit schon mal näher mit den quadratischen Codes beschäftigt und Seiten empfohlen, die zuverlässig für Sie QR-Codes erstellen [1]. Dieser Artikel fokussiert sich dagegen auf den Aufbau von QR-Codes. Dazu zerlegen wir den Code wortwörtlich in seine Bestandteile: in seine Felder und Masken. Schritt-für-Schritt zeigen wir die kleinen Finessen, die sich die Erfinder ausgedacht haben. Mit dem dadurch erlangten Wissen können Sie QR-Codes per Hand dechiffrieren. Das ist zwar nicht sonderlich hilfreich im Alltag, macht aber Spaß und ist ganz nebenbei noch lehrreich!

Geschichtsstunde

Als Kopf hinter den QR-Codes gilt der Ingenieur Masahiro Hara, der für die japanische Firma Denso Wave arbeitete. Denso erhielt 1992 den Auftrag, die Lesbarkeit von Barcodes zu verbessern.

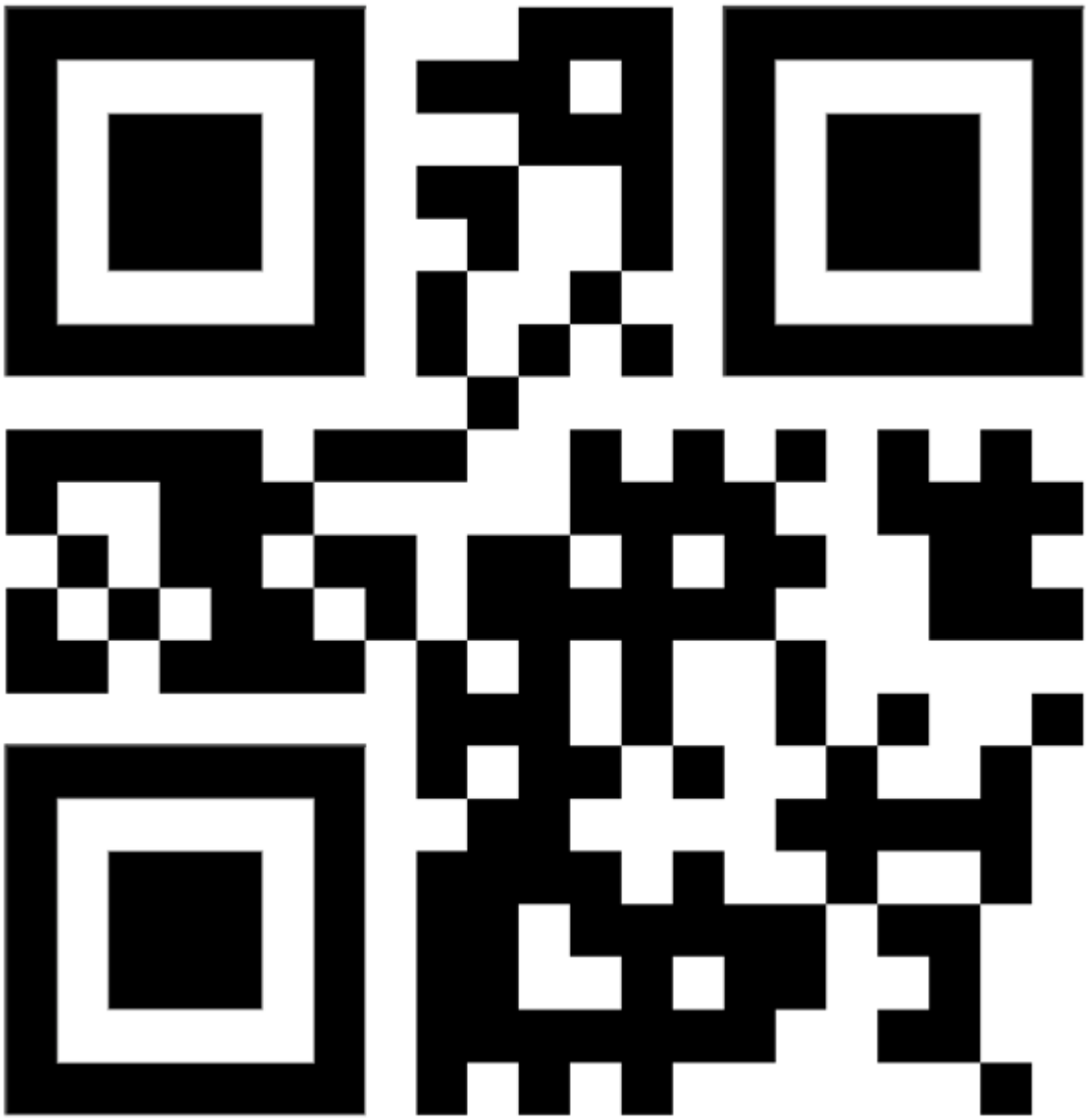
Das Unternehmen hat die Leitung Masahiro Hara übertragen, der sich des Problems prompt annahm und schnell feststellte: Selbst wenn er den Barcodescanner verbessert, die fehlende Unterstützung der Barcodes für das japanische Schriftsystem Kanji und die kleine Speicherkapazität löst das grundlegende Problem der Ineffizienz nicht.

Mit seinem Team entwickelte er in zwei Jahren einen zweidimensionalen Code, mit mehr Kapazität und schnellerer Lesbarkeit: der Quick-Response-Code, kurz QR-Code. Unter ct.de/yvy2 haben wir Ihnen die ausführliche Herkunftsgeschichte verlinkt. Dort geht der Erfinder unter anderem auf die genauen Gründe und Schwierigkeiten ein, die das Team im Laufe der Entwicklung überwinden musste.

Massig

Die Idee von zweidimensionalen Codes ist nicht neu, schon vor dem QR-Code spukten unterschiedliche Entwürfe herum. Zum Beispiel stapelte „Code 16k“ Barcodes aufeinander, um mehr Zeichen abspeichern zu können. Der QR-Code und sein ehemaliger Kontrahent gehören zu den sogenannten Matrix-Codes. Das ist eine Gruppe von Codes, die Informationen zweidimensional abspeichern, im Unterschied zu eindimensionalen Liniencodes wie Barcodes. Der QR-Code ordnet dabei jede Fläche einem Bit zu, Schwarz einer 1 und Weiß einer 0. Der starke Kontrast erlaubt es, auf kleinerem Raum größere Datenmengen abzuspeichern.

Wie viel in einen QR-Code passt, legt die Standardisierung ISO 18004 fest, die aktuelle stammt aus dem Jahr 2015 (siehe ct.de/yvy2). Diese ist gut 115 Seiten lang und definiert fein säuberlich alle Arten von QR-Codes. Das ist nicht wenig, denn es gibt insgesamt 40 Versionen, die mit aufsteigender Versionsnummer immer größer werden. Die kleinste Version 1 nimmt eine Größe von 21 × 21 Pixeln ein, die größte Version 40 177 × 177 Pixel. Die Obergrenze der Speicherkapazität liegt bei 7089 Zahlen (numerisch), 4296 alphanumerischen Zeichen (Großbuchstaben und Ziffern, zzgl. Schrägstrich, Punkt, Komma u. a.), 2958 Bytes mit beliebiger Kodierung wie UTF-8 oder bei respektablen 1817 Zeichen des japanischen Schriftsystems Kanji.



Mithilfe unserer Vorlage dechiffrieren Sie im Laufe des Artikels einen QR-Code der Version 1 mit 7 Prozent Fehlerkorrektur, der als Nachricht „ct.de“ enthält.

Fehler

QR-Codes enthalten darüber hinaus fehlerkorrigierende Bereiche. Sie sorgen dafür, dass der Code trotz Schäden scanbar bleibt. Insgesamt gibt es vier Stufen der Fehlerkorrektur, die niedrigste L sichert gerade mal 7 Prozent ab, während H einen Schaden von bis zu 30 Prozent ausgleichen kann. Für die Erfinder des QR-Codes bei der Firma Denso Wave war dieser Umstand besonders wichtig: Da die QR-Codes für ein Lagersystem gedacht waren, sollten diese von Scannern erkannt werden, selbst wenn Schmutz oder Kratzer den Code verunzieren.

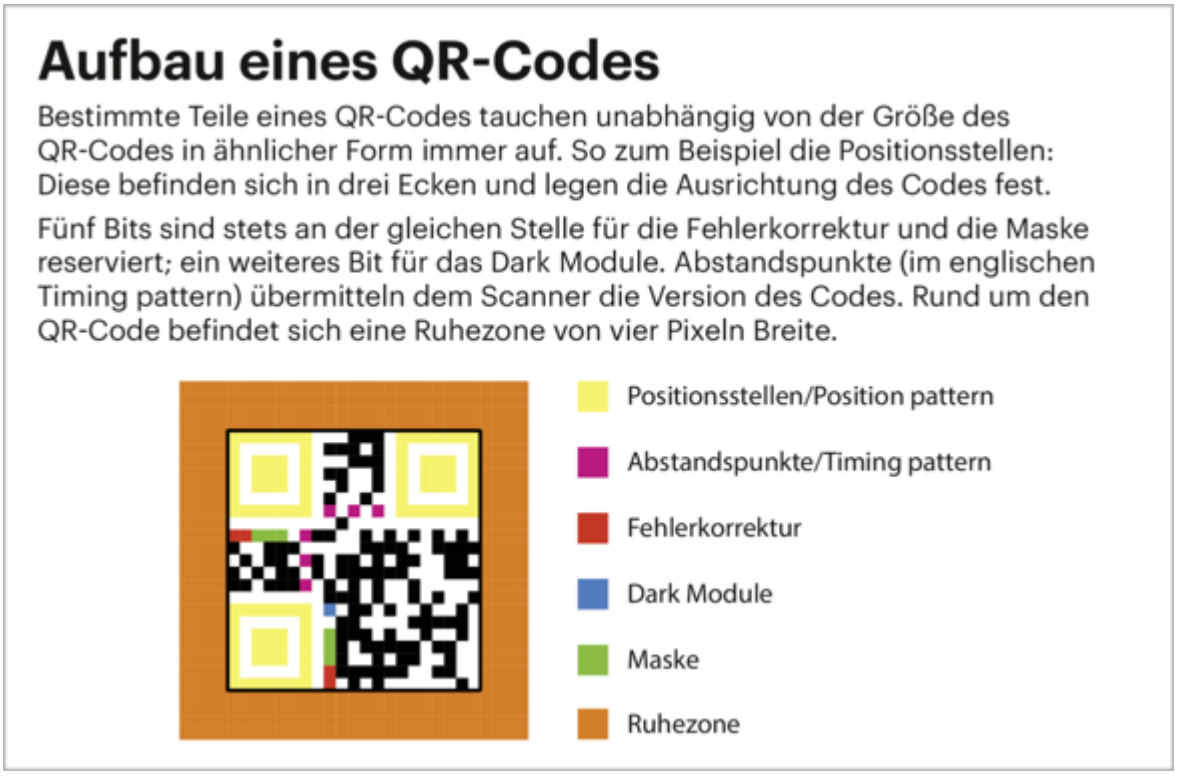
Darum kümmert sich die sogenannte Reed-Solomon-Fehlerkorrektur, benannt nach den Mathematikern und Ingenieuren Irving S. Reed und Gustave Solomon. QR-Codes können damit einen gewissen Grad an Schaden überstehen, weil in dem Code zusätzlich Informationen eingeflochten sind, die eine Wiederherstellung des Codes ermöglichen. Die Methode kommt auch bei CDs oder DVDs zum Tragen, ist aber alles andere als trivial.

Da der Fehlerkorrekturalgorithmus für die Dekodierung des vollständig lesbaren Beispielcodes keine Rolle spielt, gehen wir nachfolgend nicht näher darauf weiter ein. Sie können auch ohne den Algorithmus einen QR-Code dekodieren – das liegt am speziellen Aufbau des Codes, doch dazu im Verlauf des Artikels mehr. Eine ausführliche Erklärung der Methode haben wir unter ct.de/yvy2 verlinkt.

Aufgrund der großen Speicherkapazität und der fehlerkorrigierenden Parts findet der QR-Code in allen möglichen Gebieten Anwendung: beispielsweise im digitalen Impfausweis [2] oder bei der Weitergabe von Transaktionsdaten für Online-Überweisungen [3].

Querschnitt

Das Folgende befasst sich wegen der leichteren Nachvollziehbarkeit mit der kleinsten Version 1, die 21 × 21 Felder misst.



Bestimmte Abschnitte des QR-Codes tauchen unabhängig von Inhalt und Version des Codes immer auf (siehe Schaubild). Einige Elemente sind eher versteckt, wie die Abstandspunkte (im Englischen „Timing pattern“), andere wiederum sehr auffällig, wie die markanten Positionsstellen an den drei Ecken („Position pattern“). Die Konstellation und genaue Größe der Positionsstellen sind kein Zufall: Die Erfinder haben damals herkömmliche Zeitschriften auf besonders selten vorkommende Schwarz-Weiß-Abstände untersucht. Heraus kam ein Verhältnis von 1-1-3-1-1, also einmal Schwarz und Weiß, gefolgt von dreimal Schwarz, und zum Abschluss wieder einmal Weiß und Schwarz. Die direkten Pixel-Nachbarn rund um die Position bleiben abgetrennt weiß.

Bei größeren QR-Code-Versionen gibt es zusätzlich kleinere Positionsstellen an fest definierten Stellen. Der direkte Bereich rund um die Positionsstellen ist dem 15 Bit langen Formatstring vorbehalten (siehe Kasten „Formatstring“), der aus der Maske und dem Level der Fehlerkorrektur besteht.

Es gibt außerdem noch ein Pixel, das immer schwarz ist und in jeder Version an seinem festgeschriebenen Platz verharrt – dieser nennt sich „Dark Module“. Er schreibt vor, welche Farbe

die dunklen Pixel haben sollen, üblicherweise also schwarz.

Maske aufsetzen

In keinem der QR-Codes, den Sie in freier Wildbahn entdecken, entspricht ein schwarzes Pixel immer einer 1 und ein weißer immer einer 0 der kodierten Information. Über ihr liegt nämlich eine Maske, die die schwarzen und weißen Felder visuell so modifiziert, dass die entstehenden Einsen und Nullen möglichst gleichmäßig verteilt sind. Es gibt dadurch keine Anhäufungen von Bereichen, in denen kaum Unterschiede erkennbar sind. Scanner können aufgrund der Maske die Codes leichter erkennen und auslesen, die Wahrscheinlichkeit von Fehlerkennungen sinkt drastisch.

Es kommen acht Masken infrage, die ein QR-Code-Generator auf einen QR-Code anwenden kann. Wenn Sie zum Beispiel auf einer Webseite einen QR-Code zu „ct.de“ erstellen möchten, erzeugt der Generator zunächst einen rohen QR-Code, der jedoch noch nicht optimal lesbar ist. Dann probiert der Generator alle Masken auf dem rohen Code aus und bewertet die Ergebnisse anschließend nach vier Regeln. Welche Maske am Ende der glückliche Gewinner sein darf und auf den QR-Code gelegt wird, entscheidet der sogenannte „Penalty Score“. Das ist eine Art Konto für die Strafpunkte.

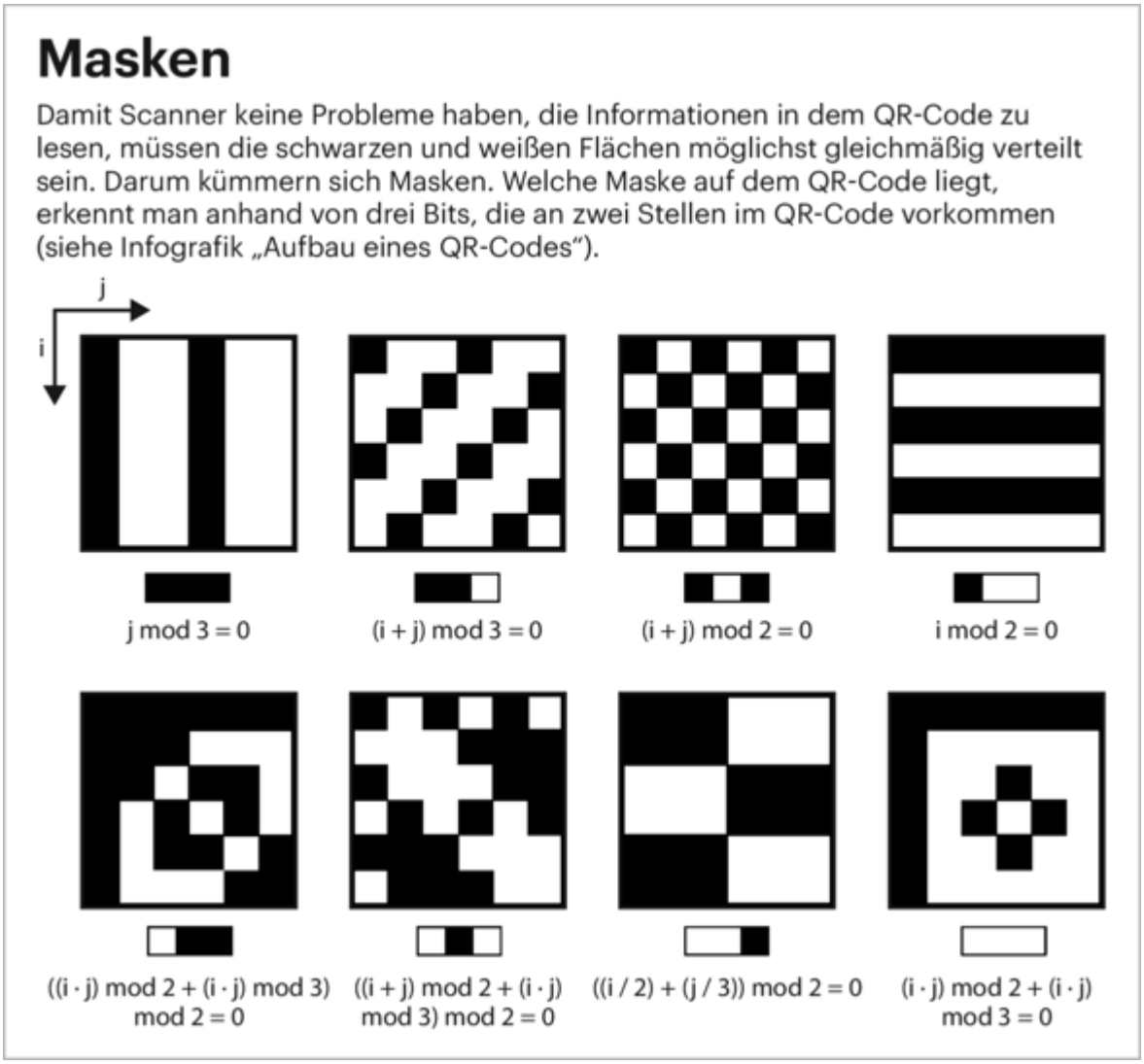
Die erste Regel schreibt vor, dass es Strafpunkte für hintereinander gereihte Pixelketten der gleichen Farbe gibt. Fünf Pixel kosten 3 Strafpunkte, plus 1 für jedes gleichfarbige Pixel, das dahinter folgt. Als Nächstes durchsucht der Generator den Code nach viereckigen Ansammlungen gleichfarbiger Pixel. Für jede gibt es weitere 3 Strafpunkte. Harsche 40 Punkte landen auf dem Konto, wenn der Generator eines der folgenden beiden Muster im Code findet: In binärer Schreibweise lauten die unerwünschten Pixelketten 10111010000 und 00001011101 (0 = weiß, 1 = schwarz). Das entspricht dem Verhältnis der Positionsstellen (1-1-3-1-1), mit vier weißen Pixeln vor oder nach der speziellen Abfolge.

Das letzte Kriterium macht den Kohl nicht fett, zählt aber trotzdem in die Bewertung hinein: das Verhältnis der Anzahl schwarzer Pixel zu der maximalen Pixelanzahl. Ein Version-1-QR-Code enthält 21 · 21 = 441 mögliche Pixel. Wenn beispielsweise 219 davon schwarz sind, entspricht das einem Anteil von gerundet 49,7 Prozent. Der Generator verteilt anschließend geringfügig Strafpunkte, wenn das Verhältnis größer als 5 Prozent zur Mitte ist. Am Ende wählt der Generator die Maske aus, welche die wenigsten Strafpunkte gesammelt hat.

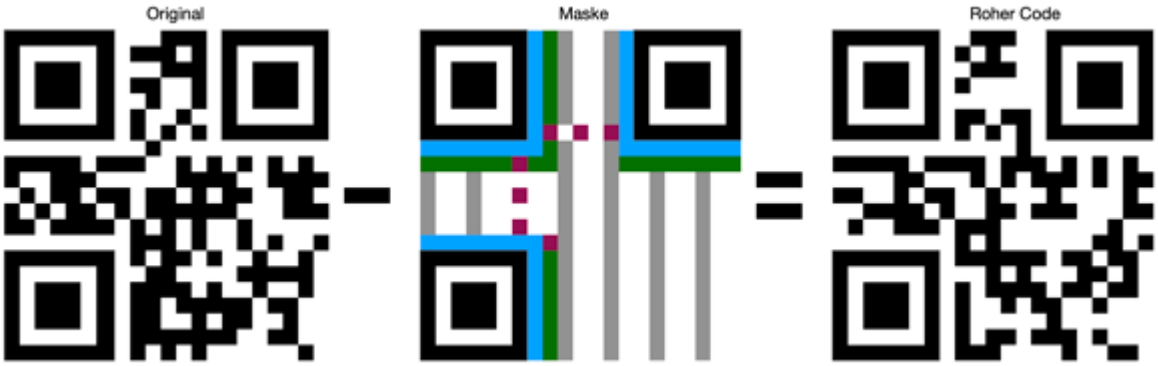
Maske lüften

Als Beispiel dechiffrieren wir im Folgenden einen QR-Code, der die Zeichenkette „ct.de“ enthält. Damit Sie den Schritten leichter folgen können, haben wir für Sie eine Vorlage in Excel vorbereitet, die Sie unter ct.de/yvy2 herunterladen können. Dort befindet sich bereits unser Beispiel-QR-Code, außerdem noch alle acht Masken und weitere Kleinigkeiten, die Ihnen bei der Dekodierung helfen. Die Vorlage eignet sich auch dazu, mal einen anderen QR-Code zu knacken; wie zum Beispiel im Rätsel am Ende des Artikels.

Zurück zu den Masken: Die Information, welche Maske zum Einsatz kam, verbirgt sich im Formatstring (siehe Kasten „Formatstring“). Jede der acht möglichen Masken besitzt eine dreistellige Kombination an Bits, die Sie an zwei Orten im QR-Code ablesen können. Die Infografik „Masken“ schlüsselt für Sie auf, welche Bits für welche Maske stehen.



Für den Beispielcode lauten die Bits 111, was der ersten Maske und der mathematischen Formel j mod 3 = 0 entspricht (j steht für die Spaltennummer, beginnend bei 0, „mod“ liefert den Rest einer Division). Das bedeutet, dass Sie jede dritte senkrechte Zeile invertieren müssen, beginnend mit der ersten. Das können Sie in unserer Vorlage per Hand erledigen, indem Sie die einzelnen Felder umfärben. Dann sieht es ungefähr so aus wie auf dem Bild oben. Obacht: Sie können nicht einfach alle Pixel austauschen, denn es gibt Bereiche, die Sie nicht anfassen dürfen. Das trifft auf die Positionsstellen und direkten Nachbarpixel zu, den Formatstring, die Abstandspunkte und das Dark Module.

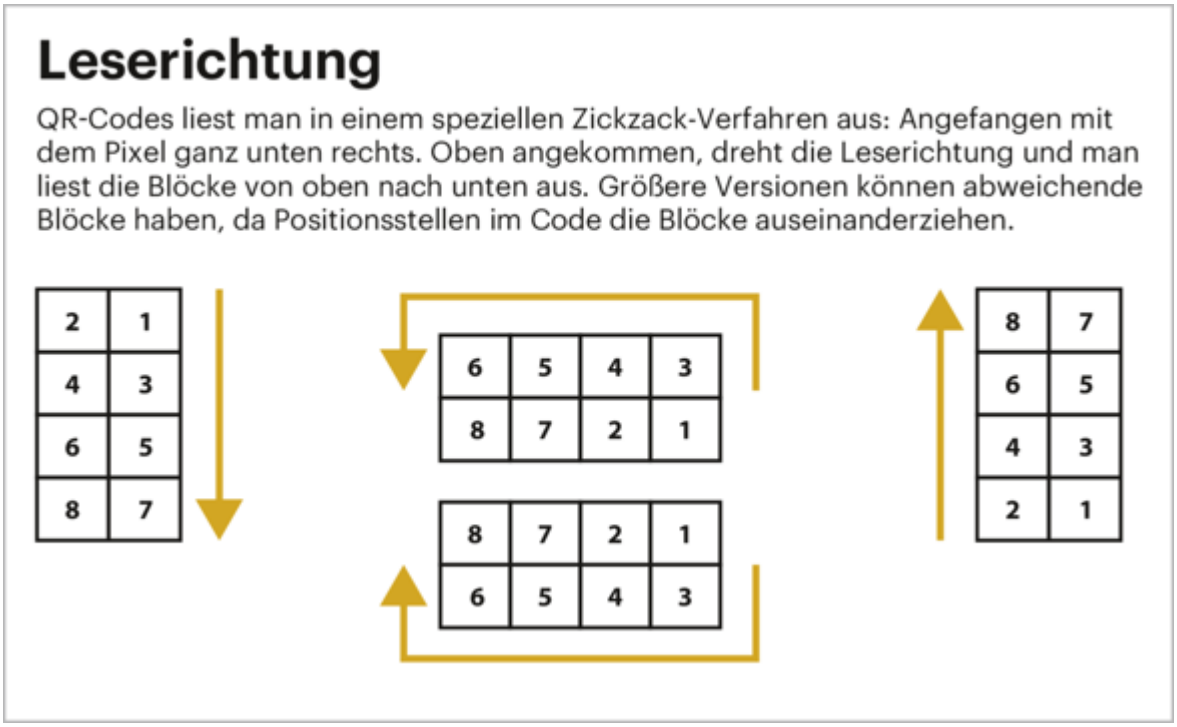


Mit unserer Excel-Vorlage können Sie spielend leicht erkennen, welche Felder Sie bei dem QR-Code invertieren müssen.

Kodierung

Jetzt liegt der QR-Code nackt und unmaskiert dar. Sie benötigen aber noch zwei weitere Informationen, um den Code endlich dekodieren zu können. Zuerst müssen Sie herausbekommen, wie die Nachricht kodiert wurde. Außerdem noch, wie lang die Nachricht ist.

Je nachdem, welches Verfahren der Generator verwendet hat, unterscheidet sich, wie viele Bits hintereinander ein Zeichen darstellen. Denn QR-Codes speichern die Nachricht nicht als Ganzes ab, sondern zerlegen sie in einzelne Zeichen und verpacken die dann beispielsweise binär in einem Block der Größe 2 × 4. Welche Kodierung also der Generator benutzt hat, erkennen Sie an den vier Pixeln ganz unten rechts in der Ecke. Den 2-×-2-Block sowie alle folgenden müssen Sie übrigens ganz speziell auslesen (siehe Infografik „Leserichtung“), angefangen ganz unten rechts. Ein weißes Feld steht für 0, ein schwarzes für eine 1.



Es gibt hierfür acht Möglichkeiten, aber relevant sind nur vier davon: Der numerische Modus (0001) liegt vor, wenn der QR-Code ausschließlich aus den Dezimalzahlen 0 bis 9 besteht. Alphanumerisch (0010) nimmt die Großbuchstaben von A bis Z ohne Umlaute hinzu und ein paar Sonderzeichen wie Dollar, Plus, Minus oder Punkt. Über ct.de/yvy2 finden Sie eine Übersetzungstabelle.

Der am häufigsten vorkommende Encoding-Modus ist 0100 und steht für den Byte-Modus. Damit kann man beliebige Daten in binär kodieren; handelt es sich um Text wie in URLs, dann kommen die verfügbaren Zeichen aus der Zeichentabelle ISO 8859-1. Der Modus bringt das klein geschriebene Alphabet mit, eine große Anzahl an Umlauten und eine ganze Palette an Sonderzeichen. Den letzten Modus treffen Sie vermutlich seltener an, denn hinter 1000 verbirgt sich das japanische Schriftsystem Kanji.

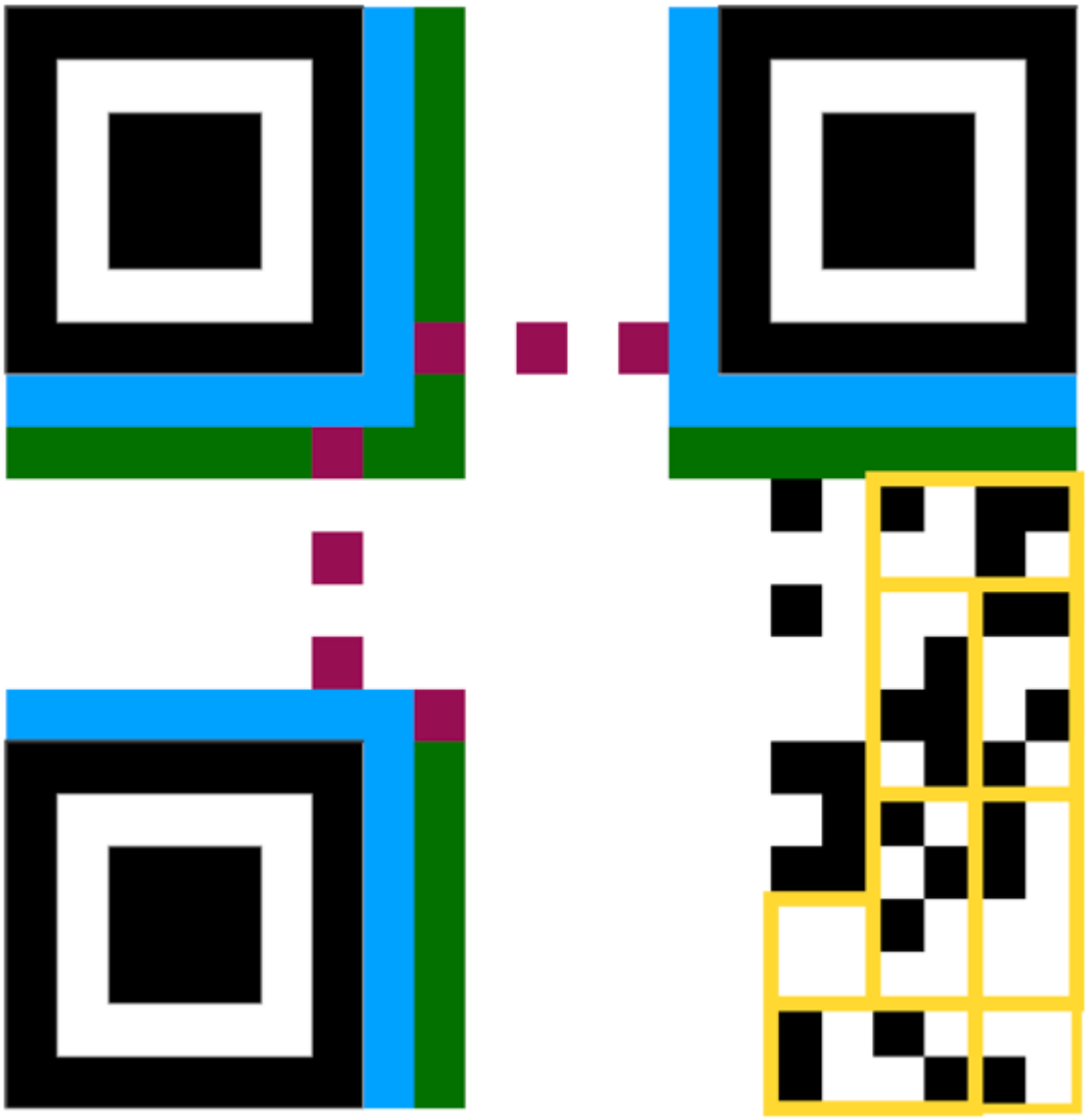
Der Beispiel-QR-Code benutzt den Byte-Modus (0100), was bedeutet, dass jedes Zeichen einem 2-×-4-Block entspricht. Jetzt brauchen Sie nur noch die Länge der Nachricht, um herauszubekommen, wie viele Blöcke Sie hintereinander dechiffrieren müssen. Im Falle des Beispiels wissen Sie die Antwort schon (ct.de = 5 Zeichen), bei einem unbekannten QR-

Code aber nicht. Daher erklären wir noch mal, wo sich die Information befindet und wie sie Sie auslesen.

Die 8 Pixel (2-×-4-Block) direkt über dem 2-×-2-Block mit der Encoding-Information sind für die Länge der Nachricht reserviert. Beachten Sie auch hier wieder die spezielle Leserichtung der Blöcke (siehe Infografik „Leserichtung“)! Sie fangen also wieder unten rechts im Block an, lesen im Zickzack nach oben und schreiben jede neue Zahl rechts dazu. Für die Länge kommt die Bit-Reihenfolge 00000101 heraus. Die Umrechnung können Sie einem entsprechenden binär zu dezimal Rechner überlassen oder es selbst per Hand versuchen. Spoiler: Die Lösung lautet 5; so viele Blöcke müssen Sie dechiffrieren, um an die Botschaft zu gelangen.

Nachricht

Jetzt können Sie, ausgehend von den schon fertigen zwei Blöcken, fünf weitere Blöcke in der korrekten Leserichtung markieren (das sollte ungefähr so aussehen wie auf dem Bild aus unserer Vorlage auf S. 147). Stellen Sie sich vor, als würden Sie von dem allerersten Block unten rechts mit der Encoding-Information eine Zickzacklinie durch alle Blöcke ziehen. Dabei dürfen Sie nicht in die reservierten Bereiche hineintappen. Zu den verbotenen Zonen gehören wie zuvor erklärt der Formatstring, die Positionsmuster, das Dark Module und die Zeilen mit den Abstandspunkten. Ein 2 × 2 Pixel großer weißer Block nach den fünf Blöcken signalisiert das Ende des Klartextes. Alle folgenden Blöcke brauchen Sie nicht zu entschlüsseln, da es sich um die Fehlerkorrektur handelt.



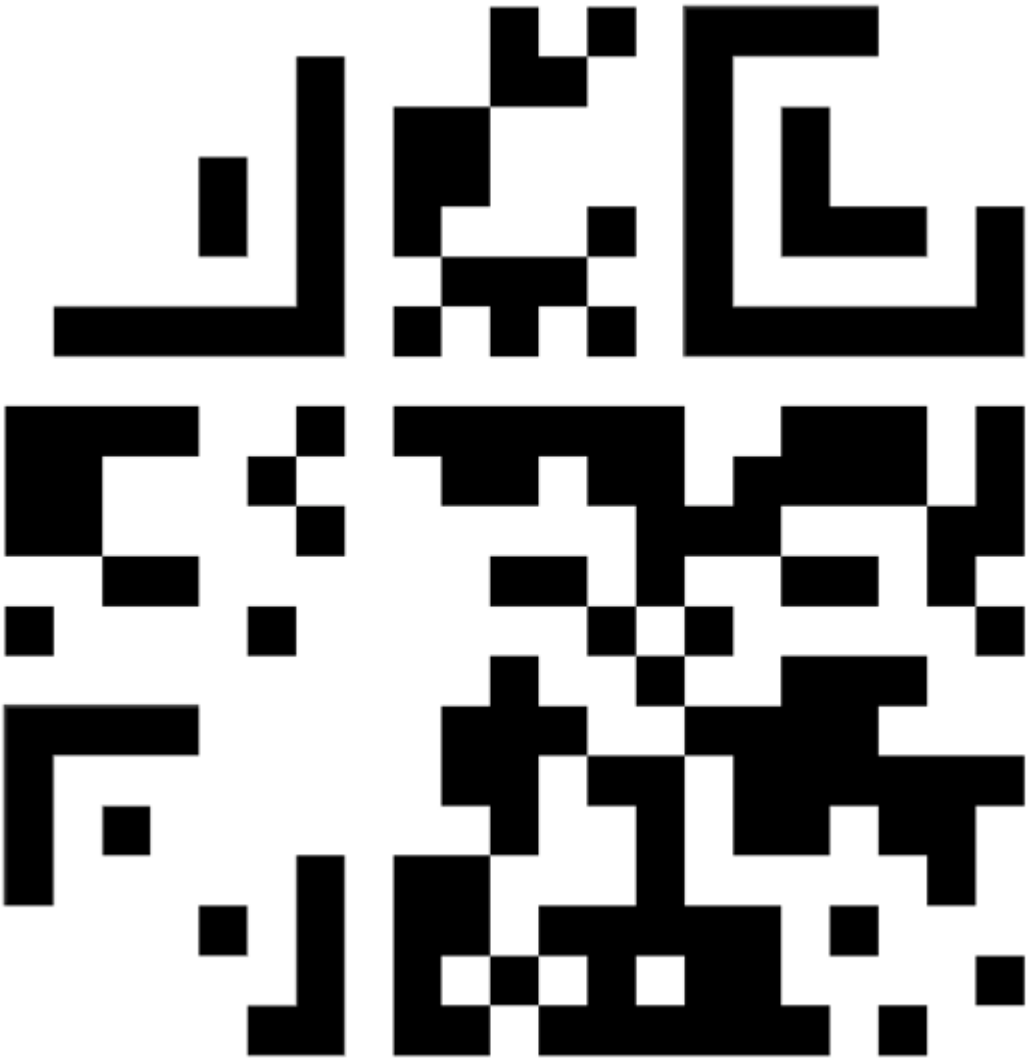
Ganz unten rechts sitzt der Encoding-Block, darüber ein 2-×-4-Block mit der Länge der Nachricht. Daraufhin folgt die Nachricht („ct.de“) aus fünf Blöcken, die bei einem 2 × 2 großen Block aus weißen Pixeln stoppt.
Im ersten Block der Nachricht steht die binäre Zahl 01100011, was in Dezimal umgerechnet 99 ergibt. Ein Blick in die ISO 8859-1 Tabelle verrät (siehe ct.de/yvy2), dass es sich um den Buchstaben „c“ handelt. Der zweite Block liegt seitwärts, passen Sie daher auf, dass Sie den Block korrekt auslesen (siehe Infografik „Leserichtung“). Diesmal kommt binär kodiert 01110100 heraus. Das ist 116 im Dezimalsystem und laut der Tabelle ein „t“.

Bleiben Sie aufmerksam, die Leserichtung ändert sich wieder! Jetzt lesen Sie die nächsten beiden Blöcke von oben nach unten

aus. Für den dritten Block sollten Sie 00101110 herausbekommen; an Stelle 46 in der Tabelle steht dann wie erwartet ein Punkt. Damit haben Sie schon mal mehr als die Hälfte erfolgreich dekodiert. Hinter dem vierten Block verbirgt sich der Buchstabe „d“ (01100100, in dezimal 100), beim letzten Block müssen Sie wieder die geänderte Leserichtung beachten. Für die binäre Darstellung 01100101 (Dezimal: 101) kommt der Buchstabe „e“ heraus. Wenn Sie alles richtig aufgeschrieben haben, sollten Sie gegen einen weißen 2-×-2-Block stoßen, dem Ende des Klartextes. Damit haben Sie die Botschaft „ct.de“ erfolgreich dekodiert.

Letztes Bit

Einen QR-Code per Hand zu entziffern, wird niemals schneller sein als der mobile Begleiter in der Hosentasche. Nichtsdestotrotz lernen Sie auf diese Weise eine Menge über die Funktionsweise der QR-Codes kennen und können sie ohne technische Hilfsmittel dekodieren. Das ist zwar im Alltag nicht wirklich nützlich, hat aber einen hohen Nerd-Faktor! Sudokus kann schließlich jeder lösen, aber QR-Codes?



Rätsel: Diesen QR-Code haben wir absichtlich so weit zerstört, dass er nicht mehr scanbar ist. Schaffen Sie es, die Botschaft im QR-Code zu retten?
Wenn Sie nach der Lektüre nun das Gefühl haben, einen QR-Code per Hand dekodieren zu können, dann probieren Sie Ihr frisch erlangtes Wissen ruhig an unserem kleinen Rätsel aus (der Code befindet sich ebenfalls in der Excel-Vorlage). Wir haben den QR-Code im Bild unten absichtlich so stark beschädigt, dass er nicht mehr scanbar ist. Mit den obigen Schritten können Sie die verlorene Botschaft trotzdem entschlüsseln. Viel Spaß! *(wid@ct.de)*

Formatstring

Bei QR-Codes kommt außer Reed-Solomon ein weiterer Fehlerkorrekturalgorithmus namens Bose-Chaudhuri-Hocquenghem

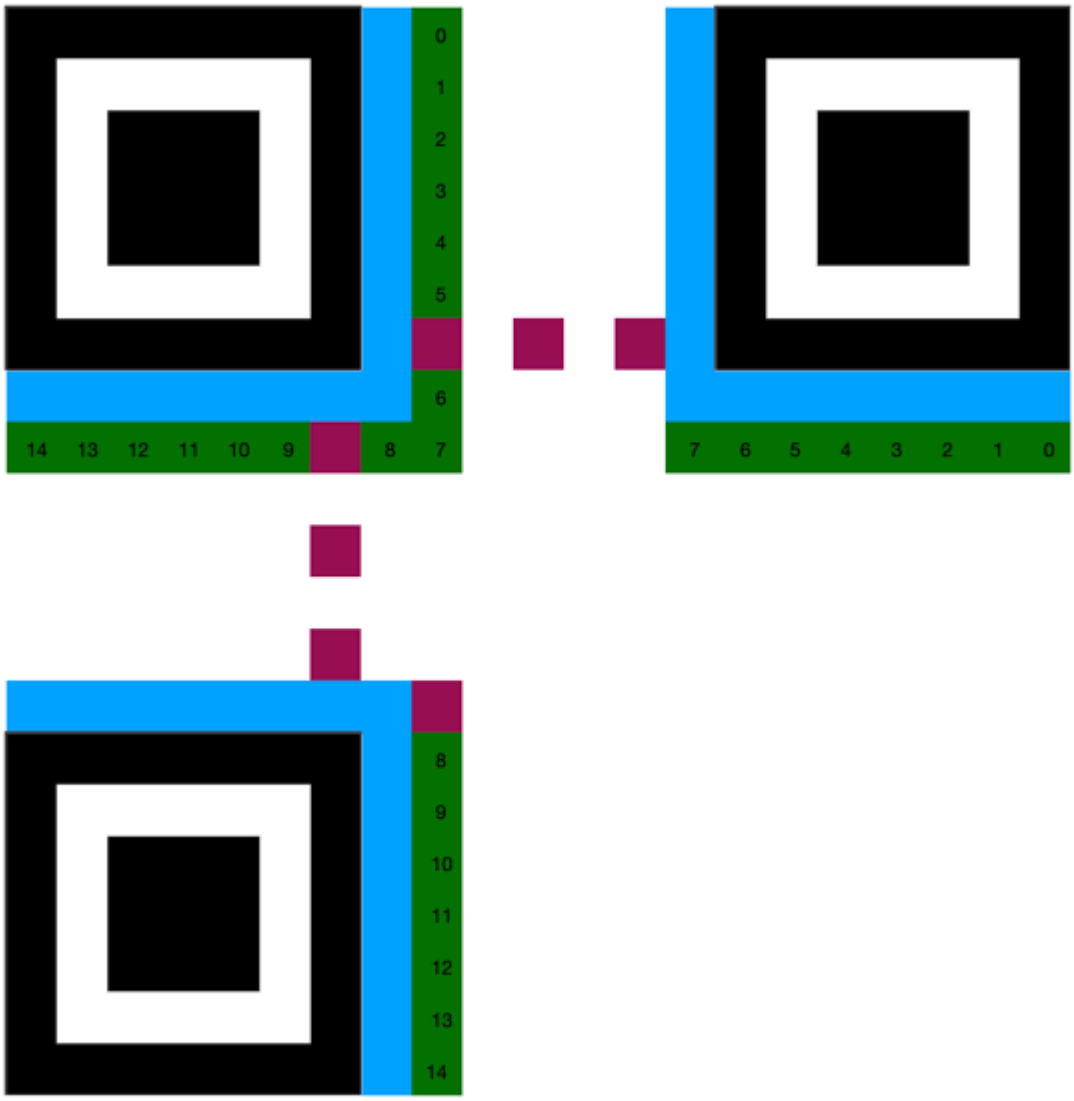
(BCH) zum Einsatz, der den Formatstring schützt. Der String kommt an zwei Orten im QR-Code vor und setzt sich aus insgesamt 15 Bits zusammen: zwei Bits für den Level der Fehlerkorrektur und drei Bits für die Maske. Die restlichen zehn sind für die Fehlerkorrektur. Im Folgenden erklären wir, wie die Fehlerkorrektur mit BCH berechnet wird.

Falls Sie einen Blick in den ISO-Standard werfen, stellen Sie fest, dass sich die Reihenfolge und Nummerierung der Masken von der in unserer Infografik unterscheidet. Das liegt daran, dass auf dem Formatstring eine spezielle Maske aufliegt, die verhindern soll, dass irgendeine Kombination von Level der Fehlerkorrektur und Maske fünf Nullen hintereinander ergibt. Dadurch ergeben sich für die Infografik eine andere Reihenfolge und Kennzeichnung, als es der ISO-Standard vorschreibt. Unsere Infografik berücksichtigt im Grunde also nur die draufgelegte Maske, wodurch sich zwar andere Reihenfolgen ergeben, aber so brauchen Sie nicht den Formatstring lernen, um einen QR-Code zu dekodieren. Für den Formatstring dagegen benötigt man die originale rohe Maskeninformation, und die stammt aus dem ISO-Standard (siehe ct.de/yvy2).

Für die Fehlerkorrektur kommen folgende vier Level infrage: L lässt 7 Prozent Schaden zu (01), M 15 Prozent (00), Q 25 Prozent (11) und H 30 Prozent (10). Je nachdem, wie der rohe QR-Code aussieht, kann sich der Generator für eine von acht Masken entscheiden, die von 000 bis 111 binär durchnummeriert sind (eine Liste finden Sie unter ct.de/yvy2). Der QR-Code mit der Nachricht „ct.de“ verwendet die Fehlerkorrektur L (01) und die Maske 010 (aus dem ISO-Standard), kombiniert ergibt das 01010. Als Nächstes berechnet man aus dem 5-Bit-Original 10 Bit an Fehlerkorrektur. Doch dafür braucht man einen Generator.

Für jede Version eines QR-Codes gibt es eine festgelegte Generator-Gleichung; für Version 1 lautet er $x^{10} + x^8 + x^5 + x^4 +$

$x^2 + x + 1$. Keine Angst, die Gleichung brauchen Sie sich nicht zu merken. Die Gleichung übersetzt man in binär von links nach rechts: Jedes x^n ergibt eine 1, fehlende Exponenten wie x^9, x^7 oder x^6 eine 0. Somit kommt 10100110111 als der Generator heraus.



Gegen Schäden geschützt: An zwei Stellen im QR-Code können Sie den Formatstring ablesen.
Danach bringt man das Original 01010 auf eine Länge von 15 Bits, indem man zehn Nullen anhängt. Danach werden vorne überstehende Nullen entfernt. Der Formatstring lautet damit vorerst 101000000000000, mit einer Länge von 14 Bits. Man verrechnet nun so lange den Formatstring mit dem Generator, bis das Ergebnis zehn Bits oder kürzer ist. Damit man beide Zahlen verrechnen kann, bringt man den Generator ebenfalls auf

eine Länge von 14 Bits, indem man die fehlenden drei Nullen anhängt (10100110111000).

Für die Kalkulation beider Zahlen braucht man den XOR-Operator: Jede Kombination von 0 und 1 ergibt 1, während bei 0 und 0 sowie 1 und 1 eine 0 herauskommt. Das ist bei dem Beispiel gleich beim ersten Versuch erreicht: 10100000000000 XOR 10100110111000 = 00000110111000. Manchmal benötigt man aber mehrere Durchläufe. Schneidet man die Nullen auf der linken Seite alle ab, kommt ein 9 Bit langer String heraus. Da der String aber 10 Bits lang sein soll, muss man eine Null auf der linken Seite übrig lassen. Der rohe Formatstring besteht jetzt aus dem Original 01010 und der gerade berechneten Fehlerkorrektur 0110111000.

Zuletzt legt man die Maske 101010000010010 via XOR auf den rohen Formatstring. Der finale Formatstring lautet also 010100110111000 XOR 101010000010010 = 111110110101010. Das entspricht den exakten 15 Bits, wie Sie sie auf dem Aufmacher oder dem Beispiel-QR-Code sehen.

1. Literatur

2. André Kramer, Quadratisch, praktisch, Code, Erfinderische und praktische Anwendungen für QR-Codes, c’t 7/2013, S. 140
3. Gerald Himmelein, Multipass, Inhalt, Apps und Datenschutz: So funktioniert das digitale Impfzertifikat, c’t 15/2021, S. 34
4. Jan Mahn, Schöner zahlen, Rechnungen schneller überweisen mit QR-Codes, c’t 7/2022, S. 138

QR-Code Linksammlung: ct.de/yvy2