Construcción del QR

Juan Carlos Cruz González Cruz

25 de abril de 2023

Índice

1 Introducción **1**

2 Historia e información **1**

3 Análisis de datos **1**
3.1 Tipos de modos . . . 1
3.2 Como escoger el modo más eficiente . . . 2

4 Codificación de datos **3**
4.1 Escoger el nivel de error de corrección de datos . . . 3
4.2 Determinar la versión más pequeña de los datos . . . 4
4.3 Agregar el indicador de modo . . . 4
4.4 Agregar el indicador de conteo de caracteres . . . 4
4.5 Codificar utilizando el modo seleccionado . . . 5
4.6 Dividir en palabras clave de 8 bits y agregar bytes de pad si es necesario . . . 7

5 Codificación de corrección errores **9**
5.1 Divida las palabras clave de datos en bloques si es necesario . . . 9
5.2 Comprender la división de polinomios . . . 11
5.3 Comprender el campo de Galois . . . 12
5.4 Comprender la aritmética del campo de Galois . . . 12
5.5 Genere potencias de 2 utilizando byte a byte módulo 100011101 . . . 13
5.6 Comprender la multiplicación con logaritmos y antilogaritmos . . . 13
5.7 Comprender el polinomio generador . . . 14
5.8 Polinomio generador para las palabras clave de corrección de errores . . . 14
5.9 Generación de palabras clave de corrección de errores . . . 15
5.10 Divida el polinomio del mensaje por el polinomio generador . . . 17

6 Estructura final del mensaje **18**
6.1 Determinar cuántos bloques y palabras clave de corrección de errores se requieren . . . 18
6.2 Intercalar bloques . . . 20
6.3 Convierta a binario . . . 23
6.4 Agregue bits restantes si es necesario . . . 23

7 Ubicación del módulo en Matrix **24**
7.1 Agregar los patrones del busqueda . . . 25
7.2 Agregar los separadores . . . 26
7.3 Agregue los patrones de alineación . . . 27
7.4 Agregar los patrones de tiempo . . . 29
7.5 Agregar el módulo oscuro y las áreas reservadas . . . 30
7.6 Colocar los bits de datos . . . 31

8 Enmascaramiento de datos **34**

9 Información de formato y versión **39**
9.1 Cadena de formato 40
9.2 Información de versión 43

10 Anexo **46**
10.1 Modos (numérico, alfanumérico y byte) 46
10.2 Capacidades de caracteres por versión, modo y corrección de errores 50
10.3 Palabras de código de corrección de errores e información de bloque 53
10.4 Tabla Log y Antilog para el campo de Galois **GF(256)** 57
10.5 Lista de versiones y bits restantes necesarios 62
10.6 Tabla de ubicaciones de patrones de alineación 63
10.7 Patrones de máscara QR explicados 64
10.8 Información de formato y versión 64

1. Introducción

Un código QR es un tipo especial de código de barras que puede codificar información como números, letras y caracteres Kanji. Este tutorial esta dirigido a toda persona que quiera aprender a codificar un código QR. El proceso de codificación es complicado , especialmente durante el paso en el que se genera palabras de código de corrección de errores. Intentaré explicar todo el proceso en términos simples. Asumiré que el lector tienen conocimientos simples de programación (en particular Python) y matemáticas puras.

2. Historia e información

El código QR fue creado en 1994 por la compañía japonesa Denso-Wase, el cual es una subsidiaria de Toyota que fabrica componentes de autos. El estandar se define en ISO/IEC 18004:2006. El uso de códigos QR está libre de licencia. Los códigos QR más pequeños son de 21×21 píxeles y los más grandes son de 177×177 píxeles. Los tamaños se llaman **versiones**:

(1) Versión 1: 21×21 píxeles.
(2) Versión 2: 25×25 píxeles.
.
.
.
(40)Versión 40: 177×177 píxeles.

En otras palabras, la versión i, es dado por el tamaño $(17+4i) \times (17+4i)$ píxeles. Además, los códigos QR incluyen corrección de errores: cuando codifica el código QR, también crea algunos datos redundantes que ayudarán a un lector de QR a leer el código con precisión, incluso si parte de el es ilegible. **Hay 4 niveles de corrección de errores entre los que puede elegir**:

(L) Proporciona 7 % de corrección de errores.
(M) Proporciona 15 % de corrección de errores.
(Q) Proporciona 25 % de corrección de errores.
(H) Proporciona 30 % de corrección de errores.

Es decir, el porcentaje nos indica que se puede leer el código aunque este tenga el 7 %, 15 %, 25 % o 30 % ilegible.

La capacidad de un código QR determinado depende de la versión y el nivel de corrección de errores, así como el tipo de datos que este codificando. Hay cuatro modos de datos que un código QR puede codificar: **numérico, alfanumérico, binario o Kanji**. La lista de versiones QR del sitio web de Denso-Wave incluye información sobre cuantos bits de datos puede codificar en cada versión.

3. Análisis de datos

Un código QR codifica una cadena de texto. El QR estándar tiene cuatro modos para codificar texto: numérico, alfanumérico, byte y Kanji. Cada modo codifica el texto como una cadena de bits (1 y 0), pero cada modo utilizar un método diferente para convertir el texto en bits, y cada método de codificación esta optimizado para codificar los datos con la cadena de bits más corta posible. En esta sección explicaré como identificar que modo usar.

3.1. Tipos de modos

Los cuatro modos de codificación incluyen los siguientes caracteres:

- **Numérico:** Este modo es para los dígitos decimales del 0 al 9. Ver cuadro 6
- **Alfanumérico:** Ver cuadro 7 para ver que caracteres corresponden a este modo.

- **Byte:** El modo byte, de forma predeterminada, es para caracteres del juego de caracteres ISO-8859-1. Sin embargo, algunos escáneres de códigos QR pueden detectar automáticamente si se usa UFT-8 en modo byte. Ver cuadros 8.
- **Kanji:** El modo Kanji es para caracteres de doble byte del juego de caracteres Shift JIS. Si bien UFT-8 puede codificar caracteres Kanji, debe usar tres o cuatro bytes para hacerlos. Shift JIS, por otro lado, usa solo dos bytes para codificar cada carácter Kanji de manera más eficiente. Si toda la cadena consta de caracteres en el rango de doble byte de Shift JIS, use el modo kanji.También es posible usar múltiples modos dentro del mismo código QR, como se describe más adelante.
- **ECI:** El modo Interpretación de canal extendida o Extended Channel Interpretation (ECI) especifica el conjunto de caracteres (Por ejemplo, UTF-8) directamente. Sin embargo, algunos lectores de códigos QR no son compatibles con el modo ECI y no entenderán los códigos QR que lo utilizan.
- **Anexo estructurado:** El modo anexo estructurado (Structured Append mode) codifica datos a través de múltiples códigos QR, hasta un máximo de 16 códigos QR, no discutiré este modo es este tutorial.
- **FCN1:** El modo FNC1 permite que el código QR funcione como un código de barras GS1, tampoco discutiremos este modo en este texto.

Una nota sobre el modo Kanji

Algunos lectores de códigos QR pueden reconocer cuando se usa UTF-8 en modo byte. Dado que todos los caracteres Shift JIS tienen representaciones en UTF-8, es posible usar el modo de byte para Kanji con codidicación UTF-8.

Sin embargo, los kanji en UTF-8 están codificados con tres bytes (o cuatro, en casos excepcionales), mientrasn que los caracteres Shift JIS están codificados con dos a un byte. En otras palabras, no será posible incluir tantos caracteres en el código QR si usa UTF-8 en modo byte para Kanji. El uso del modo Kanji para Shift JIS Kanji brinda la mayor capacidad.

Porlo tanto, depende de usted usar el modo Kanji para Kanji o no, según las necesidades de sus usuarios.

Una nota sobre UTF-8

Algunos lectores de códigos QR detectan automaticamente si se usa UTF-8 en modo byte, pero aquello que no lo hacen pueden mostrar caracteres incorrectos si se usan UFT-8 en modo byte. Para solucionar esto, es posible utilizar el modo ECI que como se mencionó anteriormente, permite especificar un conjunto de caracteres diferente del conjunto de caracteres ISO-8859-1 predeterminando en el modo bytes. Desafortunadamente, no todos los lectores de códigos QR son compatibles con el mado ECI.

Otra opción es colocar la marca de orden de bytes (BOM) UTF-8 antes del texto de entrada. Algunos lectores de códigos QR leerán la marca de orden de bytes y comprenderán que el texto está codificado en UTF-8. No todos los lectores de códigos QR pueden interpretar esto correctamente. La marca de orden de bytes para UTF-8 es un conjunto de tres números, que se muestra aquí en hexadecimal: 0xEF 0XBB, 0XBF.

3.2. Como escoger el modo más eficiente

Para seleccionar el modo más eficiente para el código QR, examine los caracteres en la cadena de entrada y verifique las siguientes condiciones.

(1) Si la cadena de entrada solo consta de dígitos decimales (0 a 9), utilice el modo numérico.

(2) Si el modo numérico no es aplicable, y si todos los caracteres de la cadena de entrada se puede encontrar en el conjunto de los caracteres del modo alfanumérico, entonces utilice el modo alfanumérico. Las letras minúsculas NO PUEDEN codificarse en modo alfanumérico; solo mayúsculas.

(3) Si hay un carácter que no está en la tabla alfanumérica pero que puede codificarse en ISO 8859-1, use el modo byte. Como se mencionó anteriormente, los lectores de códigos QR pueden reconocer UTF-8 en modo byte.

(4) Si todos los caracteres están en el juego de caracteres Shift JIS, use el modo Kanji. Los caracteres Shift JIS se pueden codificar en UTF-8, por lo que es posible usar el modo byte para Kanji, pero generalmente es más eficiente usar Shift JIS y usar el modo Kanji para caracteres Kanji.

Modos de mezcla y optimización: Es posible usar múltiples modos en un solo código QR al incluir el indicador de modo antes de cada sección de bytes que usa ese modo. La especificación del código QR explica cómo cambiar de modo de la manera más óptima. No discutiré esto en el tutorial. Este texto asumirá que no mezclará modos en sus códigos QR.

Modos de mezcla y optimización: Al examinar los caracteres en el texto de entrada, es posible elegir el modo más óptimo para codificar las limitaciones de los lectores de códigos QR al elegir un modo y tenga en cuenta que no todos los lectores de códigos QR al elegir un modo y tenga en cuenta que no todos los lectores de códigos QR se adhieren al estándar. Además, considere las necesidades de sus usuarios cuando decida si usar el modo Kanji o si usurará UTF-8 en modo byte en su lugar.

Resumen: Al examinar los caracteres en el texto de entrada, es posible elegir el modo más óptimo para codificar ese texto. Asegúrese de considerar las limitaciones de los lectores de códigos QR al elegir un modo y tenga en cuenta que no todos los lectores de códigos QR se adhieren al estándar. Ádemas, considere las necesidades de sus usuarios cuando decida si usar el modo Kanji o si usará UTF-8 en modo byte en su lugar.

4. Codificación de datos

Cada modo de codificación esta diseñado para crear la cadena de bits más corta posible para los caracteres que se utilizar en ese modo. Cada modo utiliza un modo método diferente para convertir el texto de entrada en una cadena de bits. En esta sección explicaremos todo el paso de codificación de datos.

4.1. Escoger el nivel de error de corrección de datos

Antes de codificar los datos, seleccione un nivel de corrección de errores. Como se mencionó en la introducción, los códigos QR utilizan la corrección de errores Reed-Salomon. Este proceso crea palabras de código de corrección de errores (bytes) basados en los datos codificados. Un lector de códigos QR puede usar estos bytes de corrección de errores para determinar si no leyó los datos correctamente, y las palabras clave de corrección de errores se pueden usar para corregir esos errores. Hay cuatro niveles de corrección de errores: L,M,Q,H. La siguiente tabla enumera los niveles y sus capacidades de corrección de errores.

Corrección de errores Reed-Solomon	
Nivel de corrección de errores	Capacidad de corrección de errores
L	Recupera el 7 % de datos
M	Recupera el 15 % de datos
Q	Recupera el 25 % de datos
H	Recupera el 30 % de datos

Tenga en cuenta que los niveles más altos de corrección de errores requieren más bytes, por lo que cuanto mayor sea el nivel de corrección de errores, más grande tendrá que ser el código QR.

4.2. Determinar la versión más pequeña de los datos

Los diferentes tamaños de los códigos QR se denominan versiones. Hay cuarenta versiones disponibles, la versión más pequeña es la versión 1 y tiene un tamaño de 21 píxeles por 21 píxeles. La versión 2 es de 25 píxeles por 25 píxeles. La versión más grande es la versión 40 y tiene un tamaño de 177 por 177 píxeles. Cada versión es 4 píxeles más grande que la versión anterior.

Cada versión tiene una capacidad máxima, dependiendo del modo de uso. Además, el nivel de corrección de errores restringe aún más la capacidad. Existe una tabla de capacidades de caracteres que enumera las capacidades de todas las versiones de QR para un modo de codificación y un nivel de corrección de errores determinados (Ver en anexo; tabla de capacidades 9).

En este punto, cuente la cantidad de caracteres que se codificarán y determine cuál es la versión más pequeña que puede contener esa cantidad de caracteres para el modo de codificación y el nivel de corrección de errores deseado.

Por ejemplo, la frase HOLA MUNDO tiene 10 caracteres. Si se codifica con corrección de errores nivel Q, la tabla de capacidades de caracteres dice que un código de la versión 1 que utiliza la corrección de errores de nivel Q puede contener 16 caracteres en modo alfanumérico, por lo que la versión 1 es la versión más pequeña que puede contener esta cantidad de caracteres. Si la frase tuviera más de 16 caracteres, como HOLA MUNDO DESCONOCIDO (que tiene 22 caracteres), la versión 2 sería la versión más pequeña.

Limites superiores: El código QR de mayor capacidad es 40-L (versión 40, nivel de corrección de errores L). La siguiente tabla se muestra una tabla que enumera la capacidad de un código QR de 40-L para los cuatro modos de codificación. El cual es el numero máximo posible de caracteres que puede contener un solo código QR.

Modo de codificación	Número máximo de caracteres para 40-L
Numérico	7089 caracteres
Alfanumérico	4296 caracteres
Byte	2953 caracteres
Knji	1817 caracteres

4.3. Agregar el indicador de modo

Cada modo de codificación tiene un indicador de modo de cuatro bits que lo identifica. Los datos codificados deben comenzar con el indicador de modo apropiado que especifica el modo que se usa para los bits que vienen después. La siguiente tabla enumera los indicadores de modo para cada modo.

Por ejemplo si codifica HOLA MUNDO en modo alfanumérico, el indicador de modo es 0010.

Nombre del modo	Indicador de modo
Modo numérico	0001
Modo Alfanumérico	0010
Modo Byte	0100
Modo Kanji	1000
Modo ECI	0111

4.4. Agregar el indicador de conteo de caracteres

El indicador de conteo de caracteres es una cadena de bits que representa el número de caracteres que se codifican. El indicador de conteo de caracteres debe colocarse despues del indicador de modo. Además, el indicador de conteo de caracteres debe tener una cierta cantidad de bits, dependiendo de la versión QR.

Cuente la cantidad de caracteres en el texto de entrada original, luego convierta ese número en binario. La longitud del indicador de conteo de caracteres depende del modo de codificación y la versión de código QR que se usará. Para hacer que la cadena binaria tenga la longitud adecuada, rellénela a la izquierda con ceros.

Las siguientes listas contienen los tamaños de los indicadores de conteo de caracteres para cada modo y versión. Por ejemplo si se codifica HOLA MUNDO en un código QR versión 1 en modo alfanumérico, el indicador de conteo de caracteres debe tener 9 bits de longitud. El conteo de caracteres de HOLA MUNDO es 10. En binario, 10 es 1010. Rellénelo a la izquierda para que tenga 9 bits de largo: 000001010. Coloque esto despues del indicador de modo del paso 3 para obtener la siguiente cadena de bits: 0010000001010

Modos	Versiones 1 a 9	Versiones 10 a 26	Versiones 27 a 40
Modo numérico	10 bits	12bits	14bits
Modo Alfanumérico	9 bits	11 bits	13 bits
Modo Byte	8bits	16 bits	16 bits
Modo Kanji	8 bits	10 bits	12 bits

4.5. Codificar utilizando el modo seleccionado

En la sección anterior, análisis de datos, explicamos cómo seleccionar el modo de codificación adecuado para una cadena determinada. Ahora, explicaremos el proceso de codificación para cada modo:

- **Modo numérico:** Para ilustrar la codificación en modo numérico, consideramos la cadena 8675309, codificando en una versión 1 del código QR.

  (1) Divida la cadena en grupos de tres.

  Para codificar una cadena de dígitos en modo numérico, primero divida la cadena en grupos de tres dígitos. Si la longitud de la cadena no es un múltiplo de tres, el grupo final de dígitos tendrá que tener solo uno o dos números.

  Después de dividirse en grupos de 3: 867 530 9

  (2) Convertir cada grupo en binario.

  Ahora trate cada grupo de dígitos como un número de tres dígitos (o menos de tres, si el grupo final tiene dos o uno). Convierta ese número de tres dígitos en 10 bits binarios. Si un grupo comienza con un cero, debe interpretarse como un número de dos dígitos y debe convertirlo a 7 bits binarios, y si hay dos ceros al comienzo del un grupo, debe interpretarse como un número de un dígito y debe convertirlo a cuatro bits binarios. De manera similar, si el grupo final consta de solo dos dígitos, debe convertirlo a 7 bits binarios, y si el grupo final consta de un solo dígito, debe convertirlo a cuatro bits binarios.

$$867 \longrightarrow 1101100011$$
$$530 \longrightarrow 1000010010$$
$$9 \longrightarrow 1001$$

- **Modo alfanumérico:** Para la codificación en modo alfanumérico, usaré la entrada de ejemplo HOLA MUNDO. Para este ejemplo usaré la versión 1. Recuerde, el modo alfanumérico solo puede codificar letras mayúsculas, no minúsculas.

  (1) Primero parte la cadena en pares de caracteres: HO, LA, M, UN, DO.

(2) Crea un número binario para cada par.

Para el modo alfanumérico, cada carácter alfanumérico esta representado por un número. Consulte el cuadro 7 para encontrar estos números. Para cada par de caracteres, obtenga la representación numérica del primer carácter y multiplíquelo por 45. Luego sume ese número a la representación numérica de segundo carácter. En nuestro ejemplo, tendremos los siguiente:

$$\begin{aligned} H &\longrightarrow 17 \\ O &\longrightarrow 24 \\ L &\longrightarrow 21 \\ A &\longrightarrow 10 \\ (SPACE) &\longrightarrow 36 \\ M &\longrightarrow 22 \\ U &\longrightarrow 30 \\ N &\longrightarrow 23 \\ D &\longrightarrow 13 \\ O &\longrightarrow 24 \end{aligned}$$

Siguiendo los pasos de párrafo anterior, multiplica el primer número por 45 y luego súmalo al segundo número:

$$\begin{aligned} HO &\longrightarrow 45 * 17 + 24 = 789 \\ LA &\longrightarrow 45 * 21 + 10 = 955 \\ (SPACE)M &\longrightarrow 45 * 36 + 22 = 1642 \\ UN &\longrightarrow 45 * 30 + 23 = 1373 \\ DO &\longrightarrow 45 * 13 + 24 = 609 \end{aligned}$$

Ahora, convierta ese número en una cadena binaria de 11 bits, rellenando a la izquierda si es necesario.

$$\begin{aligned} HO &\longrightarrow 789 \longrightarrow 01100010101 \\ LA &\longrightarrow 955 \longrightarrow 01110111011 \\ (SPACE)M &\longrightarrow 1642 \longrightarrow 11001101010 \\ UN &\longrightarrow 1373 \longrightarrow 10101011101 \\ DO &\longrightarrow 609 \longrightarrow 01001100001 \end{aligned}$$

Si esta codificando un número impar de caracteres, tome la representación numérica del carácter final y conviértalo en una cadena binaria de 6 bits.

- **Modo byte:** En modo byte, la cadena de datos consiste en el indicador de modo, el indicador de conteo de caracteres y luego los bytes sin formato del texto de entrada.

(1) Convierte la cadena a ISO 8859-1 o UTF-8. El conjunto de caracteres predeterminado para el modo byte es ISO 8859-1 y cuando sea posible debe convertir su texto de entrada a este conjunto de caracteres. Las especificaciones del código QR analiza el modo ECI, que le permite especificar un juego de caracteres diferente al de ISO 8850-1, pero algunos lectores de códigos QR no entienden las secuencias de escape ECI.

Si hay caracteres en la cadena que no se pueden codificar con ISO-8859-1, es posible que pueda codificarlo en UTF-8, ya que algunos lectores de códigos QR pueden detectar y mostrar

correctamente la codificación UTF-8 en el modo byte sin necesidad de secuencias de escape ECI.

Para combatir este problema, es posible que desee probar diferentes lectores de códigos QR para averiguar como tratan los caracteres que no son ISO 8859-1 en modo byte, o pedirles a los usuarios que proporcionen comentarios sobre lectores de códigos QR que utilizan.

(2) Divida la cadena en bytes de 8 bits. Después de convertir su cadena de entrada a ISO-8859-1 o UTF-8 si sus usuarios tienen lectores de códigos QR que pueden reconocerla en modo byte, debe dividir la cadena en bytes de 8 bits.

Por ejemplo, usaremos la cadena de entrada ¡Hola, mundo! para crear un código QR versión 1. Dado que contiene letras minúsculas, una coma y un signo de exclamación, no se puede codificar con el modo alfanumérico, que no incluye letras minúsculas, comas ni signos de exclamación.
Cadena convertida en una cadena binaria de 8 bits usando los cuadros 8. Rellene a la izquierda con 0 si es necesario para que cada uno tenga longitud de 8 bits.

$$
\begin{aligned}
¡ &\longrightarrow 161 \longrightarrow 10100001 \\
H &\longrightarrow 72 \longrightarrow 01001000 \\
o &\longrightarrow 111 \longrightarrow 01101111 \\
l &\longrightarrow 108 \longrightarrow 01101100 \\
a &\longrightarrow 97 \longrightarrow 01100001 \\
, &\longrightarrow 44 \longrightarrow 00101100 \\
(SPACE) &\longrightarrow 32 \longrightarrow 00100000 \\
m &\longrightarrow 109 \longrightarrow 01101101 \\
u &\longrightarrow 117 \longrightarrow 01110101 \\
n &\longrightarrow 110 \longrightarrow 01101110 \\
d &\longrightarrow 100 \longrightarrow 01100100 \\
o &\longrightarrow 111 \longrightarrow 01101111 \\
! &\longrightarrow 33 \longrightarrow 00100001
\end{aligned}
$$

- **Modo kanji:** No trataremos este caso, pues estamos en México y no es muy relevante el sistema Kanji.

Continuando con el ejemplo HOLA MUNDO (Codificado en el modo alfanumérico), la cadena de bits hasta ahora es:

Indicador de modo	Indicador de conteo de caracteres	Datos codificados
0010	000001010	0110 0010 101 0111 0111 011 1100 1101 010 1010 1011 101 0100 1100 001

4.6. Dividir en palabras clave de 8 bits y agregar bytes de pad si es necesario

Después de obtener una cadena de bits que consiste en el indicador de mod, en indicador de conteo de caracteres y los bits de datos, como se describe en los pasos previos, puede ser agregar 0's y pad bytes, porque la especificación del código QR requiere que la cadena de bits debe llenar completamente la capacidad total del código QR. Las siguientes secciones explicaremos el proceso de agregar 0's y pad bytes de relleno a la cadena de bits.

- **Determine el número de bits necesarios para este código QR:** Para determinar cuántos bits de datos se requieren para un código QR en particular, consulte la tabla 10. Encuentre la versión y el nivel de corrección de errores que esta en uso para el código QR que se esta codificando, y busque el número en la columna que está etiquetada como "Numero total de palabras clave de datos para esta versión y nivel de CE". Multiplique este número por 8 para obtener el número total de bits de datos requeridos para esta versión y nivel de corrección de errores.

  Por ejemplo, según los cuadros, un código de versión 1-Q tiene 13 palabras de código de datos total. Por lo tanto, el número total de bits necesarios para este código QR es $13 * 8 = 104$ bits.

- **Agregue un terminador de 0's si es necesario:** Si la cadena de bits es más corta que el número total de bits requeridos, se debe agregar un terminador de hasta cuatro 0 al lado derecho de la cadena. Si la cadena de bits es más de cuatro bits más corta que la cantidad requerida de bits, agregue 4 0's al final. Si la cadena de bits es menos de cuatro bits menos corta, agregue solo la cantidad de 0's que se necesitan para alcanzar la cantidad requerida de bits.

  Por ejemplo, si se codifica HOLA MUNDO en un código QR versión 1-Q, la cantidad total de bits necesarios, como se menciona en la sección anterior, es de 104 bits. La cadena de bits de datos que se muestran en el cuadro anterior tiene una longitud de 68 bits. El terminador solo debe tener una longitud máxima de 4 bits, así que agregue cuatro ceros a la derecha de la cadena. La cadena resultante aún es demasiado corta para llenar la capacidad de 104 bits, pero las especificaciones del código QR requiere que el terminador tenga una longitud máxima de cuatro ceros. Si la cadena hubiera sido de 102 bits, el terminador solo tendría 2 bits de longitud.

  Aquí está el ejemplo de cadena HOLA MUNDO con terminador agregado:

Indicador de modo	Indicador de conteo de caracteres	Datos codificados	Terminador
0010	000001010	0110 0010 101 0111 0111 011 1100 1101 010 1010 1011 101 0100 1100 001	0000

- **Agregue más 0's para hacer que la longitud sea un múltiplo de 8:** Después de agregar el terminador, si el número de bits en la cadena no es un múltiplo de 8, primero complete la cadena de la derecha con 0's para que la longitud de la cadena sea un múltiplo de 8.

  Por ejemplo, después de agregar el terminador a la cadena HOLA MUNDO, la longitud paso a 72 bits. Es un múltiplo de ocho, pero en caso de que no lo sea solo hay que agregar ceros al final para convertir la última parte en un byte binario de 8 bits. La cadena de bits se muestra enseguida dividida en bytes binarios de 8 bits:

(1)	00100000	(2)	01010011	(3)	00010101
(4)	01110111	(5)	01111001	(6)	10101010
(7)	10101110	(8)	10100110	(9)	00010000

- **Agregue bytes de pad si la cadena aún es demasiado corta:** Si la cadena aún no es lo suficientemente larga para llenar la capacidad máxima, agregue los siguientes bytes al final de la cadena, repitiendo hasta que la cadena haya alcanzado la longitud máxima:

  11101100 00010001

Estos bytes equivalen a 236 y 17, respectivamente. La especificación del código QR requiere específicamente que se agreguen si la cadena de bits es demasiado corta en esta etapa.

Por ejemplo, la cadena HOLA MUNDO anterior tiene una longitud de 72 bits. La capacidad requerida para un código 1-Q es de 104 bits como lo mencionamos previamente. El número de bits que se deben agregar para llenar la capacidad restante es 104-72= 32 bits, divida estopor 8: 32/8=4. Por lo tanto, se deben agregar cuatro bytes de relleno al final de la cadena de datos. Esto se muestra a continuación:

(1)	00100000	(2)	01010011	(3)	00010101
(4)	01110111	(5)	01111001	(6)	10101010
(7)	10101110	(8)	10100110	(9)	00010000
(10)	11101100	(11)	00010001	(12)	11101100
(13)	00010001				

Hemos terminado con el procedimiento de obtención de los datos sin procesar, el siguiente paso es generar las palabras clave de corrección de errores para los datos.

5. Codificación de corrección errores

Las palabras clave de corrección de errores permiten a los lectores de códigos QR detectar y corregir errores en los códigos QR. En esta sección explicaremos como crear estas palabras claves de corrección de errores después de codificar los datos.

5.1. Divida las palabras clave de datos en bloques si es necesario

Antes de generar palabras clave de corrección de errores, puede ser necesario dividir las palabras clave de datos en bloques más pequeños si el código QR es más grande que la versión 2. Por ejemplo, si crea un código **5-Q**, la tabla de corrección de errores dice que un código **5-Q** tiene 62 palabras de códigos de datos (cadenas de números binarios de 8 bits). Por ejemplo:

La tabla de corrección de errores menciona **grupo 1** y **grupo 2**, así como **números de bloques**. Esto significa que las palabras de código de datos deben dividirse en hasta dos grupos, y dentro de cada grupo, las palabras de código de datos pueden dividirse aún más en bloques. Las palabras de código de datos se dividen secuencialmente (es decir, comenzando con la palabra de código 1, luego la palabra de código 2 y así sucesivamente).

Para un código **5-Q**, dice que hay dos grupos, el primero de los cuales debe dividirse en dos bloques que contengan 15 palabras de código de datos cada uno, el segundo grupo debe dividirse en 2 bloques que contengan 16 datos de palabras clave cada uno. Note que 15+15+16+16 = 62, pues el número total de palabras de código de datos. A continuación, se encuentran las palabras clave de la tabla 1, divididas en los grupos y bloques correctos:

Cuadro 1: Ejemplo 5-Q

Palabra clave 1	01000011	Palabra clave 2	01010101	Palabra clave 3	01000110
Palabra clave 4	10000110	Palabra clave 5	01010111	Palabra clave 6	00100110
Palabra clave 7	01010101	Palabra clave 8	11000010	Palabra clave 9	01110111
Palabra clave 10	00110010	Palabra clave 11	00000110	Palabra clave 12	00010010
Palabra clave 13	00000110	Palabra clave 14	01100111	Palabra clave 15	00100110
Palabra clave 16	11110110	Palabra clave 17	11110110	Palabra clave 18	01000010
Palabra clave 19	00000111	Palabra clave 20	01110110	Palabra clave 21	10000110
Palabra clave 22	11110010	Palabra clave 23	00000111	Palabra clave 24	00100110
Palabra clave 25	01010110	Palabra clave 26	00010110	Palabra clave 27	11000110
Palabra clave 28	11000111	Palabra clave 29	10010010	Palabra clave 30	00000110
Palabra clave 31	10110110	Palabra clave 32	11100110	Palabra clave 33	11110111
Palabra clave 34	01110111	Palabra clave 35	00110010	Palabra clave 36	00000111
Palabra clave 37	01110110	Palabra clave 38	10000110	Palabra clave 39	01010111
Palabra clave 40	00100110	Palabra clave 41	01010010	Palabra clave 42	00000110
Palabra clave 43	10000110	Palabra clave 44	10010111	Palabra clave 45	00110010
Palabra clave 46	00000111	Palabra clave 47	01000110	Palabra clave 48	11110111
Palabra clave 49	01110110	Palabra clave 50	01010110	Palabra clave 51	11000010
Palabra clave 52	00000110	Palabra clave 53	10010111	Palabra clave 54	00110010
Palabra clave 55	11100000	Palabra clave 56	11101100	Palabra clave 57	00010001
Palabra clave 58	11101100	Palabra clave 59	00010001	Palabra clave 60	11101100
Palabra clave 61	00010001	Palabra clave 62	11101100		

Grupos	Bloques	Palabras claves
Grupo 1	Bloque 1	(Palabra clave 1) 01000011 (Palabra clave 2) 01010101 (Palabra clave 3) 01000110 (Palabra clave 4) 10000110 (Palabra clave 5) 01010111 (Palabra clave 6) 00100110 (Palabra clave 7) 01010101 (Palabra clave 8) 11000010 (Palabra clave 9) 01110111 (Palabra clave 10) 00110010 (Palabra clave 11) 00000110 (Palabra clave 12) 00010010 (Palabra clave 13) 00000110 (Palabra clave 14) 01100111 (Palabra clave 15) 00100110
	Bloque 2	(Palabra clave 16) 11110110 (Palabra clave 17) 11110110 (Palabra clave 18) 01000010 (Palabra clave 19) 00000111 (Palabra clave 20) 01110110 (Palabra clave 21) 10000110 (Palabra clave 22) 11110010 (Palabra clave 23) 00000111 (Palabra clave 24) 00100110 (Palabra clave 25) 01010110 (Palabra clave 26) 00010110 (Palabra clave 27) 11000110 (Palabra clave 28) 11000111 (Palabra clave 29) 10010010 (Palabra clave 30) 00000110

Grupos	Bloques	Palabras claves
Grupo 2	Bloque 1	(Palabra clave 31) 10110110
		(Palabra clave 32) 11100110
		(Palabra clave 33) 11110111
		(Palabra clave 34) 01110111
		(Palabra clave 35) 00110010
		(Palabra clave 36) 00000111
		(Palabra clave 37) 01110110
		(Palabra clave 38) 10000110
		(Palabra clave 39) 01010111
		(Palabra clave 40) 00100110
		(Palabra clave 41) 01010010
		(Palabra clave 42) 00000110
		(Palabra clave 43) 10000110
		(Palabra clave 44) 10010111
		(Palabra clave 45) 00110010
		(Palabra clave 46) 00000111
	Bloque 2	(Palabra clave 47) 01000110
		(Palabra clave 48) 11110111
		(Palabra clave 49) 01110110
		(Palabra clave 50) 01010110
		(Palabra clave 51) 11000010
		(Palabra clave 52) 00000110
		(Palabra clave 53) 10010111
		(Palabra clave 54) 00110010
		(Palabra clave 55) 11100000
		(Palabra clave 56) 11101100
		(Palabra clave 57) 00010001
		(Palabra clave 58) 11101100
		(Palabra clave 59) 00010001
		(Palabra clave 60) 11101100
		(Palabra clave 61) 00010001
		(Palabra clave 62) 11101100

En la tabla, observe que los bloques del grupo 1 constan de 15 palabras de código de datos, y los bloques del grupo 2 constan de 16 palabras de códigos de datos, como se especifica en la tabla de corrección de errores. Tenga en cuenta también que se divide secuencialmente, Es decir, están en el mismo orden que estaban antes de ser separados en bloques.

La tabla de corrección de errores también dice que pasa un código $5-Q$, hay 18 palabras de código de corrección de errores por bloques. En este ejemplo, hay cuatro bloques, por lo que habrá cuatro conjuntos de 18 palabras de código de corrección de errores, para un total de 72 palabras de código de errores.

Como se muestra en la tabla de corrección de errores, los códigos QR más pequeños no requieren que las palabras de código de datos se dividan en absoluto, por lo que, por un código 1-M (por ejemplo), las 16 palabras de código de datos se usaría como un solo bloque, y solo sería necesario generar 10 palabras clave de corrección de errores.

Antes de continuar con el ejemplo de $5-Q$, es necesario expliucar los pasos de la corrección de errores de Reed-Solomon.

5.2. Comprender la división de polinomios

Las palabras clave de corrección de errores se generarán mediante un método denominado corrección de errores de Reed-Solomon. Parte del proceso es realizar una división de polinomios. Es decir, dividir

un polinomio por otro polinomio. Claramente asumiremos que se posee una comprensión basta de como hacer dicha operación, no obstante, enseguida tenemos un ejemplo del mismo.

■ **Ejemplo 5.1.** *Dividir* $3x^2 + x - 1$ *por* $x + 1$

$$\begin{array}{rl|l} 3x^2 & +x-1 & x+1 \\ \cline{3-3} -\,3x^2 & -3x & 3x-2 \\ \cline{1-2} & -\,2x-1 & \\ & \;\;2x+2 & \\ \cline{2-2} & \;\;\;\;1 & \end{array}$$

En general, los pasos para realizar una división de polinomios son:

(1) Encuentra el término apropiado para multiplicarlo con el divisor de tal manera que el resultado de la multiplicación debe tener el mismo primer término que el dividendo (en el primer paso de la división) o el resto (en todos los pasos posteriores de la multiplicación).

(2) Reste el resultado al dividendo (en el primer paso de multiplicación) o del resto (en todos los pasos de multiplicación subsiguiente).

(3) Repite los pasos 1 y 2 hasta que ya no sea posible multiplicar por un polinomio, es decir, sea necesario multiplicar por una fracción. El número en la parte inferior es el resto.

La división polinómica necesaria para la corrección de errores de Reed-Solomon es más simple en algunos aspectos que en este ejemplo, porque no será necesario tratar con los exponentes de los términos polinómicos.

5.3. Comprender el campo de Galois

Como se mencionó en la sección anterior, para generar palabras clave de corrección de errores, este proceso utiliza un método llamado corrección de errores Reed-Solomon. Junto con la división de polinomios, este método usa un campo de Galois, que es esencialmente un conjunto de números, así como algunas operaciones matemáticas que crean números que todavía están en ese conjunto.

El código QR estándar dice que se use la aritmética de módulo 2 bit a bit y la aritmética de módulo 100011101 byte a byte. Esto significa usar el campo de Galois $GF(2^8) = GF(256)$, pues el 100011101 representa el polinomio irreducible $x^8 + x^4 + x^3 + x^2 + 1$ en $\mathbb{F}_2$.

Los números en $GF(2^8)$ estarán todos en el rango de 0 a 255. Observesé que este es el mismo número de rango de números que se puede representar con un byte de ocho bits (el byte de ocho bits más grande posible es 11111111, que equivale a 255).

Esto significa que todas las operaciones matemáticas en $GF(256)$ darán como resultado números que se pueden representar como bytes de ocho bits.

5.4. Comprender la aritmética del campo de Galois

Como se mencionó anteriormente, $GF(256)$ contiene los números del 0 al 255. Las operaciones matemáticas en $GF(256)$ son de naturaleza cíclica, lo que significa que si se realiza una operación matemática dentro de $GF(256)$ que da como resultado un número mayor que 255, será necesario usar la operación módulo para obtener un número que todavía este en el campo de Galois.

En el campo de Galois, los números negativos tienen el mismo valor que los números positivos, por lo que $n = -n$. En otras palabras, siempre use el valor absoluto de los números involucrados en la aritmetica del campo de Galois.

Esto significa que la suma y la resta en el campo de Galois son lo mismo. La suma y la resta en el campo de Galois se realizan sumando y restando normalmente, pero luego realizar la operación de módulo.

Y dado que estamos usando aritmética de módulo 2 bit a bit (como se menciona en la especificación del código QR), esto es lo mismo que realizar la operación **XOR**. Por ejemplo,

$$1 + 1 = 2 \mod 2 = 0 \text{ (o } 1 \wedge 1 = 0)$$
$$0 + 1 = 1 \mod 2 = 1 \text{ (o } 0 \wedge 1 = 1)$$

Con el fin de codigicar un código QR, todas las sumas y restas en $GL(256)$ se realizan mediante la operación **XOR** de los dos números.

5.5. Genere potencias de 2 utilizando byte a byte módulo 100011101

Todos los números en $GF(256)$ se pueden representar como una potencia de 2. Específicamente, todos los números en $GF(256)$ se pueden representar como 2^n, donde 2^8 parecerá demasiado grande para el campo de Galois ya que es igual a 256.

La potencias de 2 de 0 a 8 son:

$$2^0 = 1 \quad 2^1 = 2 \quad 2^2 = 4 \quad 2^3 = 8 \quad 2^4 = 16 \quad 2^5 = 32 \quad 2^6 = 64 \quad 2^7 = 128 \quad 2^2 = 256$$

La especificación del código QR dice que se use la aritmética del módulo 100011101 en bytes (donde 100011101 es un número binario que equivale a 285 en decimal). Esto significa que cuando un número es 256 o mayor, debemos hacer XOR con 285, esto conduce a valores inesperados para 2^8 y mayores. En otras palabras:

$$2^8 = 256 \wedge 285 = 29$$

Tenga en cuenta que al continuar con 29, no tomamos su valor habitual de 512 y XOR con 285 (lo que daría como resultado un número demasiado grande de todos modos). En cambio, dado que $2^9 = 2^8 * 2$, use el valor de 28 que se calculo en el paso anterior. En otras palabras:

$$2^9 = 2^8 * 2 = 29 * 2 = 58$$

Continuando usando la potencia de 2 anterior para crear las siguientes potencias de 2:

$$2^{10} = 2^9 * 2 = 58 * 2 = 116$$
$$2^{11} = 2^{10} * 2 = 116 * 2 = 232$$

Siempre que se obtenga un valor mayor o igual, a 256, nuevamente XOR con 285:

$$2^{12} = 2^{11} * 2 = 232 * 2 = 456 \wedge 285 = 205$$

En general, los valores siempre son iguales o 2 veces la potencia anterior, y si ese valor es 256 o mayor, se hace XOR con 285. Usando este procedimiento, todos los números en $GF(256)$ se pueden representar con 2^n, donde n es un número natural.

5.6. Comprender la multiplicación con logaritmos y antilogaritmos

Debido a que todos lo valores se pueden representar como 2^n como se explico anteriormente, es posible hacer logaritmos y antilogaritmos para simplificar la multiplicación en $GF(256)$. (Dicho sea de paso, esto es lo que hacen las reglas de cálculo, aunque no están restringidas a $GF(256)$). La simplificación es posible porque, en general (es decir, no solo en campos de Galois), uno puede multiplicar dos números p y q con la siguiente operación:

$$b^{\log_b(p)+\log_b(q)}$$

En este caso, la base 2 es la que usamos, por lo que la operación anterior se convierte en:

$$2^{\log_2(p)+\log_2(q)}$$

Y la operación $\log_2(x)$ nos dice cuantas veces hay que multiplicar 2 para obtener una respuesta de x. En otras palabras, determinar n, donde $2^n = n$. Esto proporciona un atajo para multiplicar números en

$GF(256)$. Al multiplicar dos números que tienen la misma base, como $2^2 * 2^8$ esto equivale a sumar los dos exponentes así.

$$2^2 * 2^8 = 2^{2+8} = 2^10$$

Esto significa que dado que todos los valores de $GF(256)$ se han expresado como potencias de dos (explicado en la sección anterior), todas las multiplicaciones en $GF(256)$ se pueden realizar como sumas de exponentes en base 2. Por ejemplo, para multiplicar $16 * 32$, esto es lo mismo que multiplicar 2^4 por 2^5. Como se explicó en el párrafo anterior, esto es lo mismo que $2(4+5)$ o 2^9. El valor 2^9 ya estaba calculado, por lo tanto en $GF(256)$ con módulo de bytes 285, $16 * 32$ es 2^9, que se determinó en la sección anterior como igual a 58. Al sumar exponentes, si el exponente se vuelve mayor o igual a 256, simplemente aplique el módulo 255. En otras palabras:

$$2^{170} * 2^{164} = 2^{(170+164)} = 2^{334} \longrightarrow 2^{334 \mod 255} = 2^{79}$$

Por lo tanto, todo lo que se necesita para realizar la multiplicación en $GF(256)$ es generar todas las potencias de 2. Estos valores ya se calcularon y se encuentran en la tabla logarítmica antilogarítmica 11. La tabla usa notación alfa, donde $\alpha = 2$. La especificación del Código QR tambien usa notación alfa en el Anexo A.

5.7. Comprender el polinomio generador

Hemos avanzado mucho en la generación de palabras clave de corrección de errores, pero aún no hemos llegado ahí. El siguiente paso es comprender los polinomios generadores. Como se mencionó anteriormente la codificación de corrección de errores utiliza la división polinomial. Para hacer eso se necesita dos polinomios. El primer polinomio que se utiliza se denomina **polinomio de mensaje**. El polinomio de mensaje usa las palabras de código de datos del paso de codificación de datos como sus coeficientes. Por ejemplo, si la palabra de código de datos, convertida a número enteros, fueran 25,218 y 35, el polinomio del mensaje sería $25x^2+218x+35$. En la practica, los polinomios de mensajes reales para códigos QR estándar son muchos más largos, pero esto es solo un ejemplo. El polinomio del menaje se dividirá por un polinomio generador. **El polinomio generador** es un polinomio que se crea multiplicando $(x - \alpha^0) \cdots (x - \alpha^{n-1})$ donde n es el número de palabras del código de corrección de errores que se debe generar (consultar la tabla 10). Como se menciono en la sección anterior $\alpha = 2$.

La especificación del código QR enumera los polinomios generadores en el Anexo A, comenzando con 2 y terminando con 68. Aunque los códigos QR estándar siempre requerirán más de 2 palabras clave de corrección de errores por bloque, esta pagina mostrará cómo calcular el polinomio generador para 2 palabras clave de corrección de errores. Porque también ilustraremos el proceso de cálculo del resto de los polinomios generadores.

5.8. Polinomio generador para las palabras clave de corrección de errores

- **Polinomio generador para 2 palabras clave de corrección de errores:** Primero, multiplicamos $x - \alpha^0$ por $x - \alpha^1$. Ya que el coeficiente de x es 1 y $\alpha^1 = 1$, esto lo podemos escribir como

  $$(\alpha^0 x - \alpha^0)(\alpha^0 x - \alpha^1)$$

  Multiplicando cada término de la primera parte por cada término de la segunda parte para obtener esto:

  $$(\alpha^0 x^1 * \alpha^0 x^1) + (\alpha^0 x^0 * \alpha^0 x^1) + (\alpha^0 x^1 * \alpha^1 x^0) + (\alpha^0 x^0 * \alpha^1 x^0)$$

  Tome en cuenta que la suma de los exponentes se puede usar aquí para realizar las multiplicaciones:

  $$\alpha^{0+0} x^{1+1} + \alpha^{0+0} x^{0+1} + \alpha^{0+1} x^{1+0} + \alpha^{0+1} x^{0+0}$$

  El resultado:

  $$\alpha^0 x^2 + (\alpha^0 + \alpha^1) x^1 + \alpha^1 x^0$$

Recuerde que la suma en $GF(256)$ se realiza mediante XOR. Convierta las alfas a sus contra partes enteras usando la tabla 11, luego realice XOR.

$$x^2 + (1 \wedge 2)x^1 + 2x^0 = x^2 + 3x^1 + 2x^0$$

Regresando en notación alpha, esto es:

$$\alpha^0 x^2 + \alpha^{25} x^1 + \alpha^1 x^0$$

Este es el polinomio generador para 2 palabras de corrección de errores.

- **Polinomio generador para 3 palabras clave de corrección de errores:** Todos los demás polinomios generadores se pueden crear de la misma manera, utilizando el polinomio generador de cada paso anterior. Comienza con el polinomio generador y multiplícalo por el siguiente factor, en este caso $x - \alpha^2$.

$$\begin{aligned}\left(\alpha^0 x^2 + \alpha^{25} x^1 + \alpha^1 x^0\right)\left(\alpha^0 x^1 + \alpha^2 x^0\right) &= (\alpha^0 x^2) * (\alpha^0 x^1) + (\alpha^0 x^2) * (\alpha^2 x^0) \\ &+ (\alpha^{25} x^1) * (\alpha^0 x^1) + (\alpha^{25} x^1) * (\alpha^2 x^0) \\ &+ (\alpha^1 x^0) * (\alpha^0 x^1) + (\alpha^1 x^0) * (\alpha^2 x^0)\end{aligned}$$

Para multiplicar sumas los exponentes así:

$$\begin{aligned}&(\alpha^{0+0} x^{2+1}) + (\alpha^{0+2} x^{2+0}) + (\alpha^{25+0} x^{1+1}) \\ &+ (\alpha^{25+2} x^{1+0}) + (\alpha^{1+0} x^{0+1}) + (\alpha^{1+2} x^{0+0})\end{aligned}$$

Después de sumar los exponentes este es el resultado:

$$\alpha^0 x^3 + \alpha^2 x^2 + \alpha^{25} x^2 + \alpha^{27} x^1 + \alpha^1 x^1 + \alpha^3 x^0$$

Ahora, combina términos semejantes:

$$\begin{aligned}&\alpha^0 x^3 + (\alpha^2 \wedge \alpha^{25}) x^2 + (\alpha^{27} \wedge \alpha^1) x^1 + \alpha^3 x^0 \\ &= \alpha^0 x^3 + (4 \wedge 3) x^2 + (12 \wedge 2) x^1 + \alpha^3 x^0 \\ &= \alpha^0 x^3 + 7x^2 + 14x^1 + \alpha^3 x^0 \\ &= \alpha^0 x^3 + \alpha^{198} x^2 + \alpha^{199} x^1 + \alpha^3 x^0\end{aligned}$$

Este es el polinomio generador de tres palabras clave de corrección de errores.

- **Cuando los exponentes son mayores o iguales a 256:** Al realizar la multiplicación sumando exponentes, a veces esto puede dar como resultado un exponente mayor o igual a 256. En este caso, aplique el módulo 255 ANTES de combinar tñerminos semejantes. Por ejemplo:

$$\alpha^{257} x^4 = \alpha^{257 \mod 255} x^4 = \alpha^2 x^4.$$

- **Otros polinomios generadores:** En general, es posible crear todos los polinomios generadores $g(x)$ como se ilustra a continuación: ($g(x)$ para $n-1$ palabras clave de corrección de errores)$*(x - \alpha^{n-1})$ $= g(x)$ para n palabras clave de corrección de errores (¡HAY QUE CODIFICAR ESTO!).

5.9. Generación de palabras clave de corrección de errores

Ahora, es el momento de comenzar a generar palabras clave de corrección de errores. Esta parte utiliza las palabras de código del ejemplo HOLA MUNDO en codificación alfanumérico. Estas palabras de código se utilizarán como coeficientes del polinomio del mensaje. Este ejemplo utiliza el nivel de corrección de errores Q y la versión 1 para crear un código 1-Q. Qué una vez haciendo el procedimiento previo sabemos que solo tenemos el grupo 1 con un solo bloque y que este último tiene 13 palabras clave.

El paso de codificación de datos resultó en las siguientes palabras de código de datos para HOLA MUNDO como código 1-Q

(1)	00100000
(2)	01010011
(3)	00010101
(4)	01110111
(5)	01111001
(6)	10101010
(7)	10101110
(8)	10100110
(9)	00010000
(10)	11101100
(11)	00010001
(12)	11101100
(13)	00010001

Convierta estos números binarios en decimal:

$$32, 83, 21, 119, 121, 170, 174, 166, 8, 236, 17, 236, 17$$

Estos números son los coeficientes del polinomio del mensaje. En otras palabras:

$$\begin{aligned} &32x^{12} + 83x^{11} + 21x^{10} + 119x^9 + 121x^8 + 170x^7 \\ &+ 174x^6 + 166x^5 + 8x^4 + 236x^3 + 17x^2 + 236x + 17 \end{aligned}$$

Ahora, obtengamos el polinomio generador. Dado que este es un código 1-Q, la tabla de corrección de errores 10 dice que se creen 13 palabras clave de corrección de errores. Por lo tanto, utilizaremos el siguiente polinomio generador:

$$\begin{aligned} &x^{13} + \alpha^{74}x^{12} + \alpha^{152}x^{11} + \alpha^{176}x^{10} + \alpha^{100}x^9 + \alpha^{86}x^8 + \alpha^{100}x^7 \\ &+ \alpha^{106}x^6 + \alpha^{104}x^5 + \alpha^{130}x^4 + \alpha^{218}x^3 + \alpha^{206}x^2 + \alpha^{140}x + \alpha^{78} \end{aligned}$$

Ahora es el momento de dividir el polinomio del mensaje por el polinomio generador. Esto se hace de la misma manera que la división de polinomios que se discutió previamente. Estos son los pasos de la división, actualizados para tener en cuenta la aritmética del campo de Galois:

- Encuentre el término apropiado para multiplicar el polinomio generador por. El resultado de la multiplicación debe tener el mismo primer término que el polinomio del mensaje (en el primer paso de multiplicación) o resto (en todos los pasos de multiplicación subsiguientes).
- XOR el resultado con el mensaje polinomio (en el primer paso de multiplicación) o resto (en todos los pasos de multiplicación subsiguientes).
- Realice estos pasos n veces, donde n es el número de palabras de código de datos.

Observe las diferencias entre estos pasos y los de la división larga de polinomios normales. En lugar de restar después del paso de multiplicación, realizamos un XOR (que, en $GF(256)$, es lo mismo).

Más importante aún, después de dividir los dos polinomios, quedará un resto. Los coeficientes de este resto son las palabras de código de corrección de errores.

La división se ilustrará paso a paso en la siguiente sección.

5.10. Divida el polinomio del mensaje por el polinomio generador

El primer paso para la división es preparar el polinomio del mensaje para la división. Para asegurarse de que el exponente del término principal no sea demasiado pequeño durante la división, multiplique el polinomio del mensaje por x^n, donde n es el número de palabras clave de corrección de errores que se necesitan. En este caso, $n = 13$, para 13 palabras de código de corrección de errores, así que multiplique el polinomio del mensaje por x^{10}, lo que nos da:

$$32x^{25} + 83x^{24} + 21x^{23} + 119x^{22} + 121x^{21} + 170x^{20} \\ + 174x^{19} + 166x^{18} + 8x^{17} + 236x^{16} + 17x^{15} + 236x^{14} + 17x^{13}$$

Ahora es posible realizar los pasos de división. El número de pasos en la división debe ser igual al número de términos en el polinomio del mensaje. En este caso, la división tomará 13 pasos para completarse. Esto dará como resultado un RESTO que tiene 13 términos. Estos términos serán las 13 palabras clave de corrección de errores que se requieren. Iniciemos con los pasos:

(1a) **Multiplique el polinomio generador por un termino que iguale al término principal del polinomio del mensaje:** El primer paso es multiplicar el polinomio generador por el término que nos iguale al término principal del polinomio del mensaje. El término principal en este caso es $32x^{25}$. Entonces debemos multiplicar por $32x^{12}$ al polinomio generador para igualarlo al término principal del polinomio del mensaje. Ahora, dado que la notación alfa facilita la realización de la multiplicación, se recomienda convertir 32 a notación alfa. De acuerdo con la tabla log antilog 11, para el valor entero 32, el exponente alfa es 5. Por lo tanto, $32 = \alpha^5$. Multiplica el polinomio generador por $\alpha^5 x^{12}$:

Los exponentes de las alfas y las x's se suman (En caso caso de que uno de los exponentes de las alfas resultantes sea mayor que 255, le aplicamos mód 255). El resultado es:

$$\alpha^5 x^{25} + \alpha^{79} x^{24} + \alpha^{157} x^{23} + \alpha^{181} x^{22} + \alpha^{105} x^{21} + \alpha^{91} x^{20} + \alpha^{105} x^{19} \\ + \alpha^{111} x^{18} + \alpha^{109} x^{17} + \alpha^{135} x^{16} + \alpha^{223} x^{15} + \alpha^{211} x^{14} + \alpha^{145} x^{13} + \alpha^{83} x^{12}$$

Ahora, convierta esto a notación entera, usando la tabla Log - Antlog 11:

$$32x^{25} + 240x^{24} + 213x^{23} + 49x^{22} + 26x^{21} + 163x^{20} + 25x^{19} \\ + 206x^{18} + 189x^{17} + 79x^{16} + 9x^{15} + 178x^{14} + 77x^{13} + 187x^{12}$$

(1b) Hacer XOR el resultado con el mensaje polinomio: Dado que este es el primer paso de la división, XOR el resultado de 1a con el polinomio de mensaje.

$$(32 \oplus 32)x^{25} + (83 \oplus 240)x^{24} + (21 \oplus 213)x^{23} + (119 \oplus 49)x^{22} \\ + (121 \oplus 26)x^{21} + (170 \oplus 163)x^{20} + (174 \oplus 25)x^{19} + (166 \oplus 206)x^{18} \\ + (8 \oplus 189)x^{17} + (236 \oplus 79)x^{16} + (17 \oplus 9)x^{15} + (234 \oplus 178)x^{14} \\ + (17 \oplus 77)x^{13} + (0 \oplus 187)x^{12}$$

El resultado es (ya eliminando el termino cero):

$$163x^{24} + 192x^{23} + 70x^{22} + 99x^{21} \\ + 9x^{20} + 180x^{19} + 104x^{18} + 173x^{17} \\ + 69x^{16} + 24x^{15} + 94x^{14} + 92x^{13} \\ + 187x^{12}$$

- Para los siguientes pasos $(2a), (2b) - (13a), (13b)$ es el mismo procedimiento, solo considerar que los coeficientes están en el campo de galois $GF(256)$. El polinomio generador es:

$$x^{13} + \alpha^{74}x^{12} + \alpha^{152}x^{11} + \alpha^{176}x^{10} + \alpha^{100}x^9 + \alpha^{86}x^8 + \alpha^{100}x^7 + \alpha^{106}x^6 + \alpha^{104}x^5 + \alpha^{130}x^4 + \alpha^{218}x^3 + \alpha^{206}x^2 + \alpha^{140}x + \alpha^{78}$$

  Y el polinomio del mensaje (ajustando su potencia por x^{13}) es:

$$32x^{25} + 83x^{24} + 21x^{23} + 119x^{22} + 121x^{21} + 170x^{20} + 174x^{19} + 166x^{18} + 8x^{17} + 236x^{16} + 17x^{15} + 236x^{14} + 17x^{13}$$

  Y considerar que ya hemos hecho el primer paso de la división.

Utilizaremos los términos del **RESTO** como palabras clave de corrección de errores. La división se ha realizado 13 veces, que es el número de términos del polinomio mensaje. Esto significa que la división está completa y los términos del polinomio anterior son las palabras clave de corrección de errores que se usarán para el polinomio del mensaje original, el polinomio RESIDUO es:

$$29x^{12} + 46x^{11} + 237x^{10} + 25x^9 + 126x^8 + 84x^7 + 181x^6 + 11x^5 + 168x^4 + 99x^3 + 8x^2 + 188x + 246$$

Las palabras de código de datos y las palabras de código de corrección de errores ya se han generado. Para códigos QR más grandes, la especificación del código QR requiere que las palabras clave se intercalen de acuerdo con un patrón particular. En la siguiente sección, **estructura del mensaje final**, explicaremos ese proceso de entrelazado.

6. Estructura final del mensaje

En la sección de codificación de corrección de errores, se explicó el proceso de corrección de errores Reed-Solomon. Ahora debería tener las palabras clave de datos y sus correspondientes palabras clave de corrección de errores. Como se mencionó en el paso anterior, los códigos QR más grandes requieren que divida las palabras clave de datos en bloques más pequeños y genere palabras clave de corrección de errores por separado para cada bloque. Cuando este sea el caso, los bloques de datos y las palabras clave de corrección de errores deben intercalarse de acuerdo con la especificación del código QR. Esta página explica el proceso de entrelazado en detalle.

6.1. Determinar cuántos bloques y palabras clave de corrección de errores se requieren

La tabla de corrección 10 de errores muestra cuántos bloques de datos y palabras de código de corrección de errores por bloque se requieren para cada versión y nivel de corrección de errores.

Tenga en cuenta que los códigos QR más pequeños solo constan de un bloque de palabras clave de datos, con un conjunto de palabras clave de corrección de errores para ese bloque. En este caso, no es necesario intercalar. Simplemente coloque las palabras clave de corrección de errores después de las palabras clave de datos y avance al siguiente paso, **la ubicación del módulo en la matriz**.

Dividir códigos QR más grandes: En la sección de codificación de corrección de errores, mostré un ejemplo 2 usando un código 5-Q. La tabla de corrección de errores dice que el primer grupo de un código 5-Q consta de 2 bloques, con 15 palabras de código de datos por bloque, y el segundo grupo consta de 2 bloques, con 16 palabras de código de datos por bloque. Entonces, Aquí nuevamente está la tabla pero ahora considerando las palabras clave de corrección de errores. Observe que la tabla de corrección de errores dice que para cada bloque, un código 5-Q debe tener 18 palabras de código de corrección de errores. Ahora que se ha explicado la codificación de corrección de errores, se pueden calcular las palabras de código de corrección de errores para los datos anteriores.

Hay cuatro bloques, por lo que se deben generar cuatro conjuntos de 18 palabras de código de corrección de errores. La siguiente es una tabla actualizada que muestra las palabras clave de corrección de datos de cada bloque convertidas en una lista de números enteros y las 18 palabras clave de corrección de errores generadas para cada bloque.

Grupos	Bloques	Palabras claves	Palabras clave de datos como enteros Palabras claves de CE como enteros
Grupo 1	Bloque 1	(Palabra clave 1) 01000011 (Palabra clave 2) 01010101 (Palabra clave 3) 01000110 (Palabra clave 4) 10000110 (Palabra clave 5) 01010111 (Palabra clave 6) 00100110 (Palabra clave 7) 01010101 (Palabra clave 8) 11000010 (Palabra clave 9) 01110111 (Palabra clave 10) 00110010 (Palabra clave 11) 00000110 (Palabra clave 12) 00010010 (Palabra clave 13) 00000110 (Palabra clave 14) 01100111 (Palabra clave 15) 00100110	Palabras clave de datos: 67,85,70,134,87,38,85,194,119,50, 6,18,6,103,38 Palabras claves de CE: 213 199 11 45 115 247 241 223 229 248 154 117 154 111 86 161 111 39
	Bloque 2	(Palabra clave 16) 11110110 (Palabra clave 17) 11110110 (Palabra clave 18) 01000010 (Palabra clave 19) 00000111 (Palabra clave 20) 01110110 (Palabra clave 21) 10000110 (Palabra clave 22) 11110010 (Palabra clave 23) 00000111 (Palabra clave 24) 00100110 (Palabra clave 25) 01010110 (Palabra clave 26) 00010110 (Palabra clave 27) 11000110 (Palabra clave 28) 11000111 (Palabra clave 29) 10010010 (Palabra clave 30) 00000110	Palabras clave de datos: 246,246,66,7,118,134,242,7,38,86, 22,198,199,146,6 Palabras claves de CE: 87 204 96 60 202 182 124 157 200 134 27 129 209 17 163 163 120 133
Grupo 2	Bloque 1	(Palabra clave 31) 10110110 (Palabra clave 32) 11100110 (Palabra clave 33) 11110111 (Palabra clave 34) 01110111 (Palabra clave 35) 00110010 (Palabra clave 36) 00000111 (Palabra clave 37) 01110110 (Palabra clave 38) 10000110 (Palabra clave 39) 01010111 (Palabra clave 40) 00100110 (Palabra clave 41) 01010010 (Palabra clave 42) 00000110 (Palabra clave 43) 10000110 (Palabra clave 44) 10010111 (Palabra clave 45) 00110010 (Palabra clave 46) 00000111	Palabras clave de datos: 182,230,247,119,50,7,118,134,87,38, 82,6,134,151,50,7 Palabras claves de CE: 148 116 177 212 76 133 75 242 238 76 195 230 189 10 108 240 192 141
	Bloque 2	(Palabra clave 47) 01000110 (Palabra clave 48) 11110111 (Palabra clave 49) 01110110	Palabras clave de datos: 70,247,118,86,194,6,151,50,16,236,

Bloque 2

Grupos	Bloques	Palabras claves	Palabras clave de datos como enteros Palabras claves de CE como enteros
		(Palabra clave 50) 01010110 (Palabra clave 51) 11000010 (Palabra clave 52) 00000110 (Palabra clave 53) 10010111 (Palabra clave 54) 00110010 (Palabra clave 55) 11100000 (Palabra clave 56) 11101100 (Palabra clave 57) 00010001 (Palabra clave 58) 11101100 (Palabra clave 59) 00010001 (Palabra clave 60) 11101100 (Palabra clave 61) 00010001 (Palabra clave 62) 11101100	17,236,17,236,17,236 Palabras claves de CE: 235 159 5 173 24 147 59 33 106 40 255 172 82 2 131 32 178 236

Ahora explicamos en seguida el proceso de intercalación de los bloques de datos y las palabras clave de corrección de errores.

6.2. Intercalar bloques

Los bloques se intercalan haciendo lo siguiente:

(1) Tomar la primera palabra de código de datos del primer bloque

(2) Seguido de la primera palabra de código de datos del segundo bloque

(3) seguido de la primera palabra de código de datos del tercer bloque

(4) seguido de la primera palabra de código de datos del cuarto bloque

(5) seguido de la segunda palabra de código de datos del primer bloque

(6) Y así sucesivamente.

Este patrón se repite, recorriendo los bloques, hasta que todas las palabras de código de datos se han intercalado.

Después de eso, haz lo siguiente:

(1) tomar la primera palabra clave de corrección de errores del primer bloque

(2) seguido de la primera palabra clave de corrección de errores del segundo bloque

(3) seguido de la primera palabra clave de corrección de errores del tercer bloque

(4) seguido de la primera palabra clave de corrección de errores del cuarto bloque

(5) seguido de la segunda palabra clave de corrección de errores del primer bloque

(6) y así

Haga esto hasta que se hayan agotado todas las palabras clave de corrección de errores. Este proceso se detalla a continuación.

Intercalar las palabras clave de datos: El proceso de entrelazado de las palabras de código de datos se ilustra con la tabla y la descripción a continuación.

	Bloque 1	Bloque 2	Bloque 3	Bloque 4
Fila 1	67	246	182	70
Fila 2	85	246	230	247
Fila 3	70	66	247	118
Fila 4	134	7	119	86
Fila 5	87	118	50	194
Fila 6	38	134	7	6
Fila 7	85	242	118	151
Fila 8	194	7	134	50
Fila 9	119	38	87	16
Fila 10	50	86	38	236
Fila 11	6	22	82	17
Fila 12	18	198	6	236
Fila 13	6	199	134	17
Fila 14	103	146	151	236
Fila 15	38	6	50	17
Fila 16			7	236

Como se describió anteriormente, tome la primera palabra de código de datos del primer bloque, seguida de la primera palabra de código de datos del segundo bloque, seguida de la primera palabra de código de datos del tercer bloque, seguida de la primera palabra de código de datos del cuarto bloque.

O, como se muestra en la tabla anterior, tome las palabras de código de datos de la fila 1, comenzando con el bloque 1 y terminando con el bloque 4. Los números de la fila 1 son 67, 246, 182 y 70.

Datos intercalados hasta ahora:

$$67, 246, 182, 70$$

Después de eso, coloque los números de la fila 2, comenzando con el bloque 1 y terminando con el bloque 4.

$$67, 246, 182, 70, 85, 246, 230, 247$$

Hacemos lo mismo con la fila 3. Datos intercalados hasta ahora:

$$67, 246, 182, 70, 85, 246, 230, 247, 70, 66, 247, 118$$

Repita el proceso, yendo fila por fila. Aquí está el resultado después de agregar la columna 15. Datos intercalados después de agregar la columna 15:

$$67, 246, 182, 70, 85, 246, 230, 247, 70, 66, 247, 118, 134, 7, 119, 86, 87, 118, 50, 194,$$
$$38, 134, 7, 6, 85, 242, 118, 151, 194, 7, 134, 50, 119, 38, 87, 16, 50, 86, 38, 236, 6, 22,$$
$$82, 17, 18, 198, 6, 236, 6, 199, 134, 17, 103, 146, 151, 236, 38, 6, 50, 17$$

La fila 16 solo tiene dos números, como puede ver en la tabla anterior. Simplemente ponga estos dos números al final de los datos intercalados así:

$$67, 246, 182, 70, 85, 246, 230, 247, 70, 66, 247, 118, 134, 7, 119, 86, 87, 118, 50, 194,$$
$$38, 134, 7, 6, 85, 242, 118, 151, 194, 7, 134, 50, 119, 38, 87, 16, 50, 86, 38, 236, 6, 22,$$
$$82, 17, 18, 198, 6, 236, 6, 199, 134, 17, 103, 146, 151, 236, 38, 6, 50, 17, 7, 236$$

Intercalar las palabras clave de corrección de errores: Ahora que se han intercalado las palabras de código de datos, se deben intercalar las palabras de código de corrección de errores. Este proceso se ilustra con la tabla y la descripción a continuación.

	Bloque 1	Bloque 2	Bloque 3	Bloque 4
Fila 1	213	87	148	235
Fila 2	199	204	116	159
Fila 3	11	96	177	5
Fila 4	45	60	212	173
Fila 5	115	202	76	24
Fila 6	247	182	133	147
Fila 7	241	124	75	59
Fila 8	223	157	242	33
Fila 9	229	200	238	106
Fila 10	248	134	76	40
Fila 11	154	27	195	255
Fila 12	117	129	230	172
Fila 13	154	209	189	82
Fila 14	111	17	10	2
Fila 15	86	163	108	131
Fila 16	161	163	240	32
Fila 17	111	120	192	178
Fila 18	39	133	141	236

Como se describió anteriormente, tome la primera palabra clave de corrección de errores del primer bloque, seguida de la primera palabra clave de corrección de errores del segundo bloque, seguida de la primera palabra clave de corrección de errores del tercer bloque, seguida de la primera palabra clave de corrección de errores del cuarto bloque. . O, como se muestra en la tabla anterior, tome las palabras clave de corrección de errores de la fila 1, comenzando con el bloque 1 y terminando con el bloque 4. Los números de la fila 1 son 213, 87, 148 y 235.

Códigos de corrección de errores intercalados hasta ahora:

$$213, 87, 148, 235$$

Después de eso, coloque los números de la fila 2, comenzando con el bloque 1 y terminando con el bloque 4. Códigos de corrección de errores intercalados hasta ahora:

$$213, 87, 148, 235, 199, 204, 116, 159$$

Haz lo mismo con la fila 3. Códigos de corrección de errores intercalados hasta ahora:

$$213, 87, 148, 235, 199, 204, 116, 159, 11, 96, 177, 5$$

Repita el proceso, yendo fila por fila. Aquí esta el resultado final. Códigos de corrección de errores intercalados:

$$213, 87, 148, 235, 199, 204, 116, 159, 11, 96, 177, 5, 45, 60, 212, 173, 115, 202, 76, 24,$$
$$247, 182, 133, 147, 241, 124, 75, 59, 223, 157, 242, 33, 229, 200, 238, 106, 248, 134,$$
$$76, 40, 154, 27, 195, 255, 117, 129, 230, 172, 154, 209, 189, 82, 111, 17, 10, 2, 86, 163,$$
$$108, 131, 161, 163, 240, 32, 111, 120, 192, 178, 39, 133, 141, 236$$

Colocar palabras de código de corrección de errores intercaladas después de palabras de código de datos intercalados: El mensaje final consta de las palabras de código de datos intercaladas seguidas de las palabras de código de corrección de errores intercaladas. Mensaje final:

67, 246, 182, 70, 85, 246, 230, 247, 70, 66, 247, 118, 134, 7, 119, 86, 87, 118, 50, 194,
38, 134, 7, 6, 85, 242, 118, 151, 194, 7, 134, 50, 119, 38, 87, 16, 50, 86, 38, 236, 6, 22,
82, 17, 18, 198, 6, 236, 6, 199, 134, 17, 103, 146, 151, 236, 38, 6, 50, 17, 7, 236, 213,
87, 148, 235, 199, 204, 116, 159, 11, 96, 177, 5, 45, 60, 212, 173, 115, 202, 76, 24, 247,
182, 133, 147, 241, 124, 75, 59, 223, 157, 242, 33, 229, 200, 238, 106, 248, 134, 76,
40, 154, 27, 195, 255, 117, 129, 230, 172, 154, 209, 189, 82, 111, 17, 10, 2, 86, 163,
108, 131, 161, 163, 240, 32, 111, 120, 192, 178, 39, 133, 141, 236

6.3. Convierta a binario

El mensaje final se convierte a binario de 8 bits. En nuestro ejemplo, las primeras cuatro palabras clave son 67, 246, 182 y 70. Convertidas a binario, estas son:

$$67 = 01000011$$
$$246 = 11110110$$
$$182 = 10110110$$
$$70 = 01000110$$

Los números binarios de 8 bits se juntan en una cadena continua. Usando las primeras cuatro palabras clave de arriba, los primeros 32 bits del mensaje final son:

01000011111101101011011001000110

Hacemos el proceso previo para tener el mensaje final completo en binario como sigue:

01000011111101101011011001000110010101011111011011100110111101110100011
00100001011110111011101101000011000000111011101110101011001010111011101
10001100101100001000100110100001100000011100000110010101011111001001110
11010010111110000100000011110000110001100100111011100100110010101110001
00000011001001010110001001101110110000000110000101100101001000010001000
10010110001100000011011101100000001101100011110000110000100010110011110
01001010010111111011000010011000000110001100100001000100000111111011001
10101010101011110010100111010111100011111001100011101001001111100001011
01100000101100010000010100101101001111001101010010101101011100111100101
00100110000011000111101111011011010000101100100111111000101111100010010
11001110111101111110011101111100100010000111100101110010001110111001101
01011111000100001100100110000101000100110100001101111000011111111110111
01011000000111100110101011001001101011010001101111010101001001101111000
10001000010100000001001010110101000110110110010000011101000011010001111
11000000100000011011110111100011000000101100100010011110000101100011011
1101100

6.4. Agregue bits restantes si es necesario

Para algunas versiones QR, el mensaje binario final no es lo suficientemente largo para completar la cantidad requerida de bits. En este caso, es necesario agregar un cierto número de 0 al final del mensaje final para que tenga la longitud correcta. Estos 0 adicionales se denominan bits restantes. Un código QR de la versión 5, como el de este ejemplo, debe tener 7 bits restantes agregados al final.

Los 8 bits finales de la cadena en el cuadro de texto de arriba son:

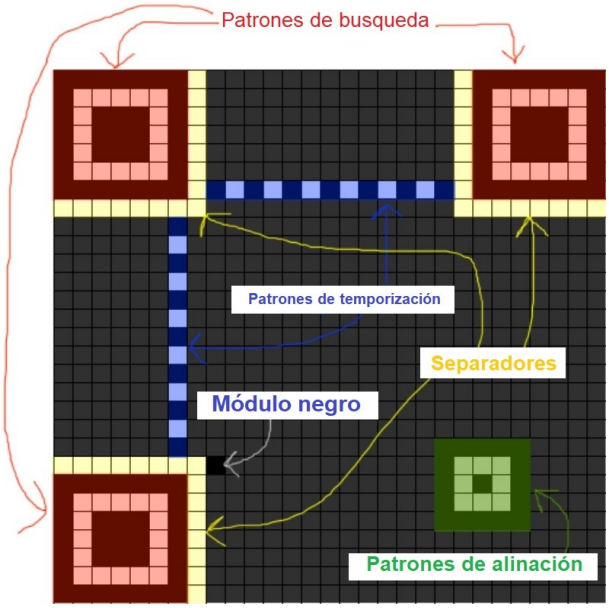
11101100

Con los 7 bits restantes agregados al final, los bits finales de la cadena son:

111011000000000

Para completar, el mensaje final completo en binario, con los 7 bits restantes agregados es:

01000011111101101011011001000110010101011111011011100110111101110100011
00100001011110111011101101000011000000111011101110101011001010111011101
10001100101100001000100110100001100000011100000110010101011111001001110
11010010111110000100000011110000110001100100111011100100110010101110001
00000011001001010110001001101110110000000110000101100101001000010001000
10010110001100000011011101100000001101100011110000110000100010110011110
01001010010111111011000010011000000110001100100001000100000111111011001
10101010101011110010100111010111100011111001100011101001001111100001011
01100000101100010000010100101101001111001101010010101101011100111100101
00100110000011000111101111011011010000101100100111111000101111100010010
11001110111101111110011101111100100010000111100101110010001110111001101
01011111000100001100100110000101000100110100001101111000011111111110111
01011000000111100110101011001001101011010001101111010101001001101111000
10001000010100000001001010110101000110110110010000011101000011010001111
11000000100000011011110111100011000000101100100010011110000101100011011
1101100000000000

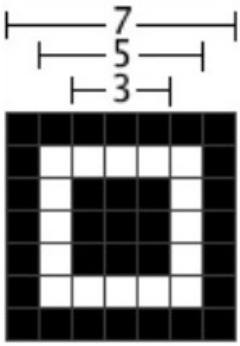


- **Los patrones de búsqueda** son los tres bloques en las esquinas del código QR en la parte superior izquierda, superior derecha e inferior izquierda.
- **Los separadores** son áreas de espacios en blanco junto a los patrones del buscador.
- **Los patrones de alineación** son similares a los patrones de búsqueda, pero más pequeños, y se colocan en todo el código. Se utilizan en las versiones 2 y superiores, y sus posiciones dependen de la versión del código QR.
- **Los patrones de temporización** son líneas de puntos que conectan los patrones del buscador.
- **El módulo oscuro** es un único módulo negro que siempre se coloca al lado del patrón del buscador inferior izquierdo.

En lo que sigue explicaremos con mayor detalle cómo colocar los patrones de función.

7.1. Agregar los patrones del busqueda

Primero, coloque los patrones del buscador en la matriz. El patrón del buscador (que se muestra a continuación) consta de un cuadrado negro exterior de 7 módulos por 7 módulos, un cuadrado blanco interior de 5 módulos por 5 módulos y un cuadrado negro sólido en el centro de 3 módulos por 3 módulos.



El patrón de búsqueda está diseñado para ser un patrón que es poco probable que aparezca dentro de las otras secciones del código QR. Los anchos de los módulos del patrón del buscador tienen una relación de 1:1:3:1:1. Los escáneres de códigos QR pueden buscar esta proporción de módulos claros y oscuros para detectar los patrones del buscador y orientar correctamente el código QR para la decodificación.

Los patrones del buscador siempre se colocan en las esquinas superior izquierda, superior derecha e inferior izquierda del código QR, sin importar qué versión esté en uso.

Para ilustrar esto, las siguientes imágenes muestran las ubicaciones de los patrones del buscador. La primera imagen es un código de la versión 1 y la segunda imagen es un código de la versión 18.

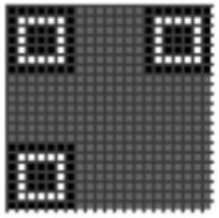


Figura 1: Versión 1

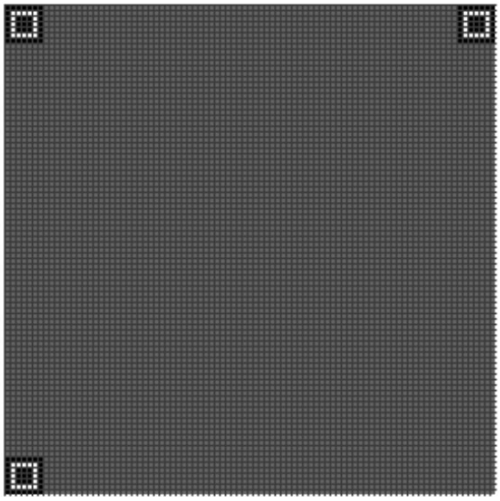


Figura 2: Versión 18

El tamaño de un código QR se puede calcular con la fórmula $(((V-1)*4)+21)$, donde V es la versión del código QR. Por ejemplo, la versión 32 es $(((32-1)*4)+21)$ o 145 módulos por 145 módulos. Por lo tanto, las posiciones de los patrones del buscador se pueden generalizar de la siguiente manera:

- La esquina superior izquierda del patrón del buscador superior izquierdo siempre se coloca en $(0, 0)$.
- La esquina superior izquierda del patrón del buscador superior derecho siempre se coloca en

$$([(((V-1)*4)+21)-7], 0)$$

- La esquina superior izquierda del patrón del buscador inferior izquierdo siempre se coloca en

$$(0, [(((V-1)*4)+21)-7])$$

7.2. Agregar los separadores

Los separadores son líneas de módulos blancos, de un módulo de ancho, que se colocan junto a los patrones del buscador para separarlos del resto del código QR. Los separadores solo se colocan al lado de los bordes de los patrones del buscador que tocan el interior del código QR. Por ejemplo:

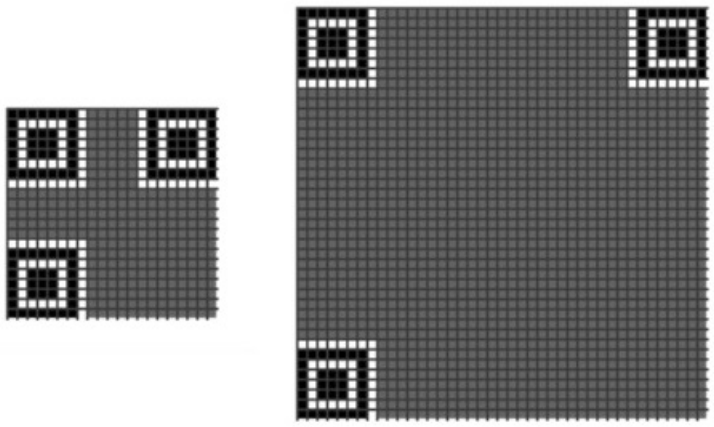


Figura 3: Separadores

7.3. Agregue los patrones de alineación

Los códigos QR de la versión 2 y mayores deben tener patrones de alineación. Un patrón de alineación, que se muestra a continuación, consta de un cuadrado negro de 5 módulos por 5 módulos, un cuadrado interior blanco de 3 módulos por 3 módulos y un único módulo negro en el centro.

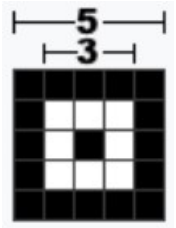


Figura 4: Patrón de alineación

Las ubicaciones en las que se deben colocar los patrones de alineación se definen en la tabla de ubicaciones de patrones de alineación. Los números deben usarse como coordenadas de fila y columna. Por ejemplo, la Versión 2 tiene los números 6 y 18. Esto significa que los MÓDULOS CENTRALES de los patrones de alineación deben colocarse en (6, 6), (6, 18), (18, 6) y (18, 18) .

SIN EMBARGO, los patrones de alineación deben colocarse en la matriz DESPUÉS de que se hayan colocado los patrones de búsqueda y los separadores, y los patrones de alineación NO DEBEN superponerse a los patrones de búsqueda o separadores. Las siguientes imágenes muestran un código de la versión 2, que se describió en el párrafo anterior con patrones de alineación centrados en (6, 6), (6, 18), (18, 6) y (18, 18). Sin embargo, como muestra la imagen de la izquierda, los patrones de alineación resaltados en rojo no deben colocarse en la matriz porque se superponen a los patrones de búsqueda y separadores. Los patrones de alineación que se superponen a los patrones buscadores o separadores simplemente se omiten de la matriz.

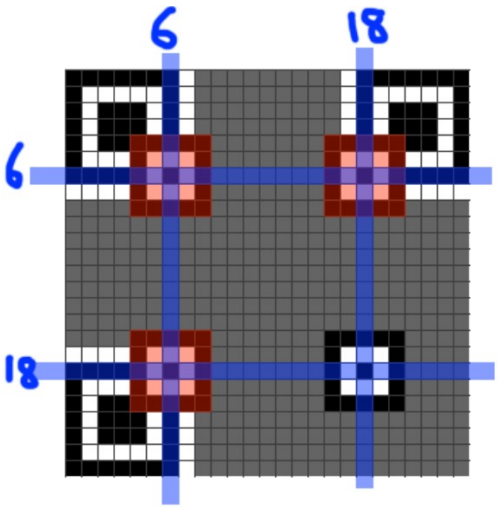


Figura 5: Incorrecto

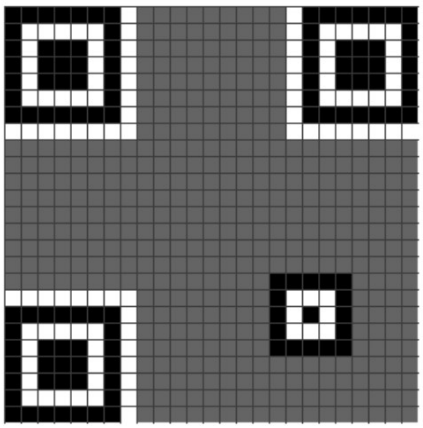


Figura 6: Correcto

Para ilustrar aún más la ubicación del patrón de alineación, la siguiente imagen muestra patrones de alineación para un código QR de la versión 8. La **tabla de ubicaciones de patrones** 13 de alineación enumera 6, 24 y 42 como ubicaciones de patrones de alineación para la versión 8. Todas las combinaciones de estos tres números se utilizan como coordenadas para los patrones de alineación.

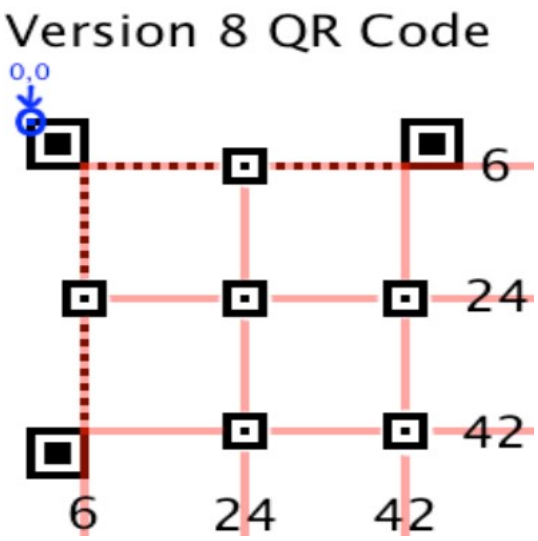


Figura 7: Versión 8

7.4. Agregar los patrones de tiempo

Los patrones de temporización son dos líneas, una horizontal y otra vertical, de módulos oscuros y claros alternados. El patrón de tiempo horizontal se coloca en la sexta fila del código QR entre los separadores. El patrón de tiempo vertical se coloca en la sexta columna del código QR entre los separadores. Los patrones de tiempo siempre comienzan y terminan con un módulo oscuro. Los patrones de alineación pueden superponerse a los patrones de temporización porque sus módulos claros y oscuros siempre coinciden con los módulos claros y oscuros de los patrones de temporización, como se puede ver en la imagen de la derecha. Las siguientes imágenes muestran patrones de tiempo en diferentes versiones de códigos QR.

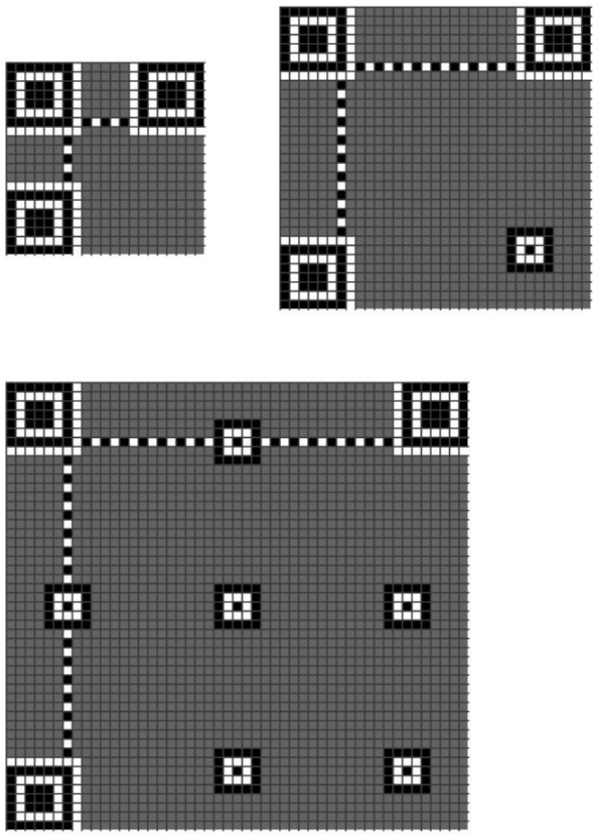


Figura 8: Ejemplos de patrones de tiempo

7.5. Agregar el módulo oscuro y las áreas reservadas

Es casi el momento de agregar los bits de datos a la matriz del código QR. Sin embargo, antes de hacer eso, se debe agregar el módulo oscuro, y hay áreas de la matriz que se deben reservar para los bits de formato y versión, que se agregarán más adelante.

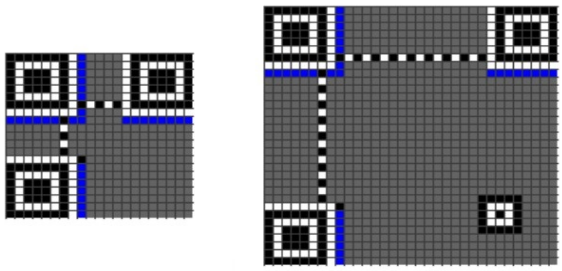
- **Módulo oscuro:** Como se mencionó anteriormente en esta página, todos los códigos QR tienen un módulo oscuro al lado del patrón del buscador en la parte inferior izquierda. Más específicamente, el módulo oscuro siempre se ubica en la coordenada $([(4 * V) + 9], 8)$ donde V es la versión del código QR.
- **Reserve el Área de Información del Formato:** Se debe reservar una franja de módulos al lado de los separadores para el área de información de formato de la siguiente manera:

  (1) Cerca del patrón del buscador superior izquierdo, se debe reservar una franja de un módulo debajo y a la derecha del separador.

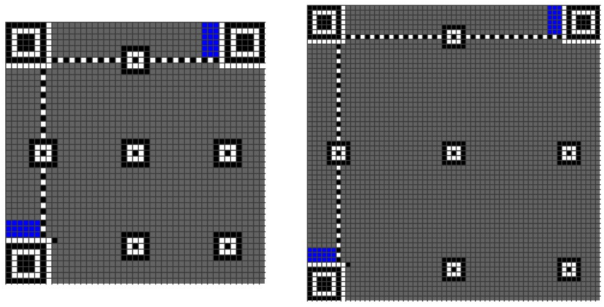
  (2) Cerca del patrón del buscador superior derecho, se debe reservar una franja de un módulo debajo del separador.

  (3) Cerca del patrón del buscador inferior izquierdo, se debe reservar una franja de un módulo a la derecha del separador.

  Las siguientes imágenes muestran las áreas reservadas en azul. Siempre se colocan a lo largo de los separadores, sin importar la versión del código QR.



- **Reserve el Área de Información de la Versión:** Los códigos QR de las versiones 7 y superiores deben contener dos áreas donde se colocan los bits de información de la versión. Las áreas son un bloque de 6x3 sobre el patrón del buscador inferior izquierdo y un bloque de 3x6 a la izquierda del patrón del buscador superior derecho. Las siguientes imágenes muestran las ubicaciones de las áreas reservadas en azul.

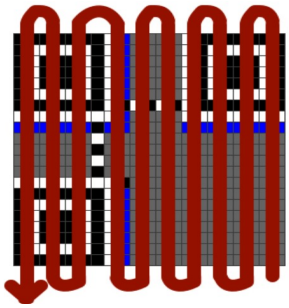


7.6. Colocar los bits de datos

Ahora es el momento de agregar los bits de datos a la matriz QR. Los bits se colocan en un patrón particular.

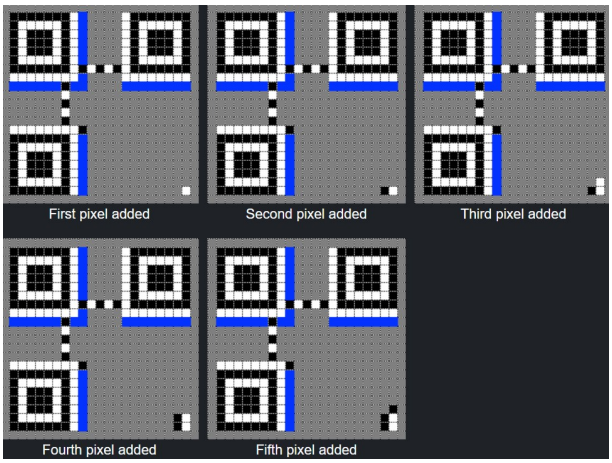
Los bits de datos se colocan comenzando en la parte inferior derecha de la matriz y avanzando hacia arriba en una columna de 2 módulos de ancho. Use píxeles blancos para 0 y píxeles negros para 1. Cuando la columna llega a la parte superior, la siguiente columna de 2 módulos comienza inmediatamente a la izquierda de la columna anterior y continúa hacia abajo. Cada vez que la columna actual alcance el borde de la matriz, pase a la siguiente columna de 2 módulos y cambie de dirección. Si se encuentra un patrón de función o un área reservada, el bit de datos se coloca en el siguiente módulo no utilizado.

La siguiente imagen muestra el patrón de colocación de los bits de datos en el código QR. Observe que cuando se alcanza el patrón de tiempo vertical, la siguiente columna comienza a la izquierda.

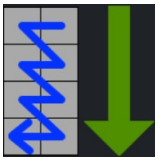


- **Colocación hacia arriba:** La siguiente imagen muestra el orden en el que colocar los bits de datos cuando la columna sube.

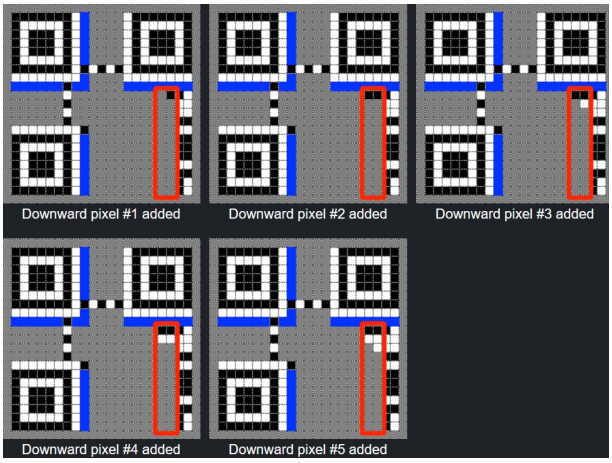
El siguiente ejemplo ilustra la ubicación de los bits en la primera columna hacia arriba.



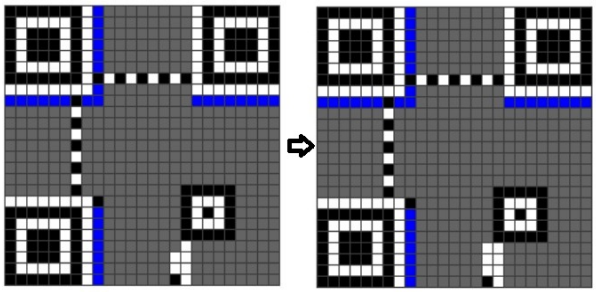
- **Colocación hacia abajo:** La siguiente imagen muestra el orden en el que colocar los bits de datos cuando la columna va hacia abajo.



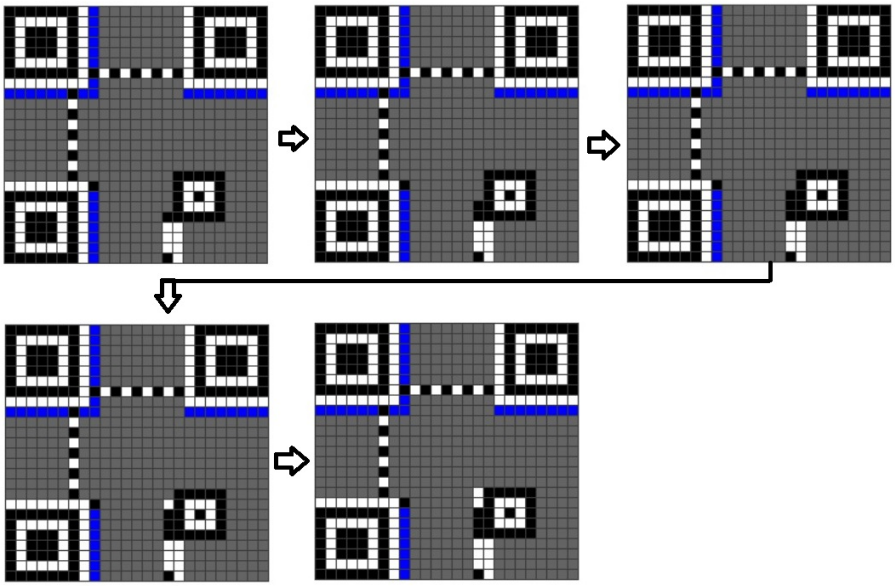
El siguiente ejemplo muestra bits de datos que se colocan en una columna hacia abajo.



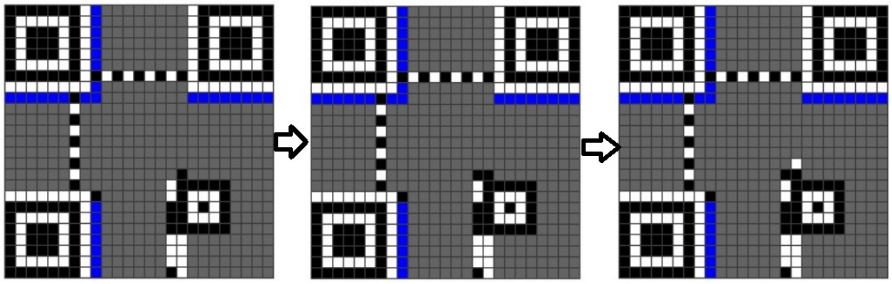
- **Saltar patrones de función:** Cuando encuentre un patrón de función, omita los módulos ocupados hasta llegar al siguiente módulo sin usar. En las siguientes imágenes, los bits de datos debajo del patrón de alineación avanzan en una columna ascendente. Observe que la columna se superpone al patrón de alineación.



Cuando se alcance el patrón de alineación, simplemente omita los módulos que forman parte del patrón de alineación y continúe hacia arriba.

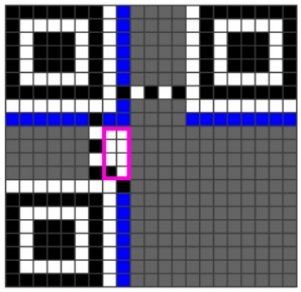


Luego continúe agregando módulos a la columna de la forma habitual.

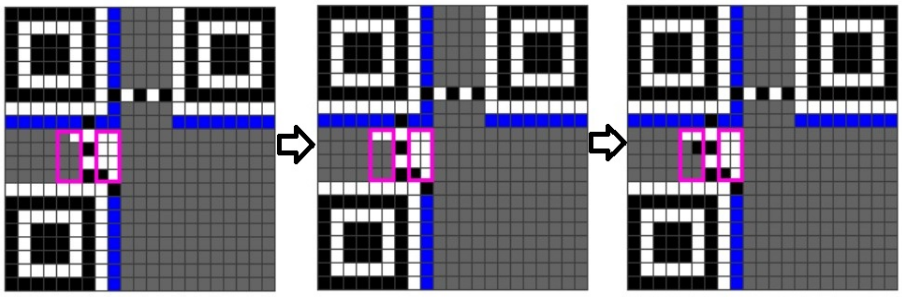


En resumen, proceda siempre normalmente a lo largo de las columnas, omitiendo cualquier módulo que se utilice por patrones de funciones o áreas reservadas. La única excepción es el patrón de temporización vertical, como se explica a continuación.

- **Excepción: patrón de sincronización vertical.** El patrón de tiempo vertical es la única excepción a esta regla. Cuando se alcanza el patrón de tiempo vertical, siempre comience la siguiente columna a la izquierda. Ninguna columna debe superponerse nunca al patrón de tiempo vertical. En la siguiente imagen, la columna anterior (hacia arriba) está resaltada. El resto de los módulos por encima de esta columna están ocupados por patrones de función y áreas reservadas, por lo que se debe iniciar la siguiente columna, yendo hacia abajo.



Como se muestra en las siguientes imágenes, la siguiente columna comienza a la izquierda del patrón de tiempo vertical. La columna no se superpone al patrón de tiempo vertical.



Después de colocar los bits de corrección de errores y datos en la matriz, la especificación del código QR requiere que se aplique un patrón de máscara a los bits de corrección de errores y datos. El propósito de este paso es reducir el número de patrones difíciles de leer en la matriz. Continúe con la siguiente sección para obtener información sobre el enmascaramiento de datos.

8. Enmascaramiento de datos

Ahora que los módulos se han colocado en la matriz, se debe determinar el mejor patrón de máscara. Un patrón de máscara cambia qué módulos son oscuros y cuáles son claros de acuerdo con una regla particular. El propósito de este paso es modificar el código QR para que sea tan fácil de escanear para un lector de códigos QR como sea posible.

Terminología: Enmascaramiento

Si un módulo en el código QR está .[en]mascarado", esto simplemente significa que si es un módulo claro, debe cambiarse a un módulo oscuro, y si es un módulo oscuro, debe cambiarse a un módulo claro. En otras palabras, enmascarar simplemente significa alternar el color del módulo.

Descripción general de los patrones de máscara:

La especificación del código QR define ocho patrones de máscara que se pueden aplicar al código QR. Por ejemplo, para el patrón de máscara #1, se enmascaran todas las filas pares de la matriz QR, y para el patrón de máscara #2, se enmascara una de cada tres columnas de la matriz QR. Consulte el anexo de **patrones de máscara** para obtener más detalles sobre los ocho patrones de máscara.

¿Qué enmascarar?:

Los patrones de máscara SOLO se deben aplicar a los módulos de datos y módulos de corrección de errores. En otras palabras: No enmascare patrones de función (patrones de búsqueda, patrones de temporización, separadores, patrones de alineación) No enmascare áreas reservadas (área de información de formato, área de información de versión)

Determinar la mejor máscara:

Una vez que se ha aplicado un patrón de máscara a la matriz QR, se le otorga una puntuación de penalización basada en cuatro condiciones de evaluación que se definen en la especificación del código QR. Un codificador de código QR debe aplicar los ocho patrones de máscara y evaluar cada uno. El patrón de máscara que resulte en la puntuación de penalización más baja es el patrón de máscara que se debe usar para la salida final.

¿Cómo evaluar áreas reservadas?

Tenga en cuenta que se evalúa toda la matriz (incluidos los patrones de función y las áreas reservadas), aunque el enmascaramiento solo se aplica a los módulos de datos y corrección de errores.

Las cuatro reglas de penalización: Las cuatro reglas de penalización se pueden resumir de la siguiente manera:

(1) La primera regla penaliza el código QR por cada grupo de cinco o más módulos del mismo color en una fila (o columna).

(2) La segunda regla penaliza el código QR por cada área de 2x2 de módulos del mismo color en la matriz.

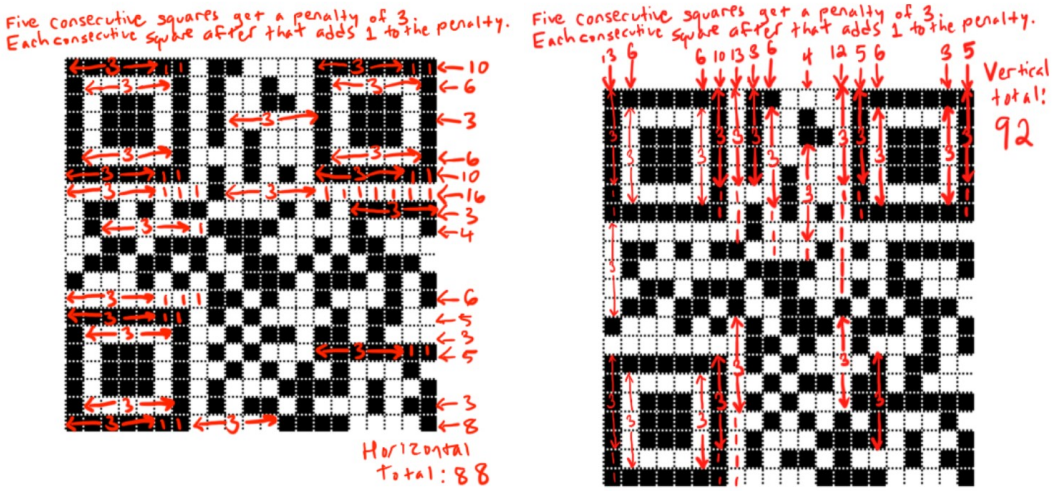
(3) La tercera regla le da al código QR una gran penalización si hay patrones que se parecen a los patrones del buscador.

(4) La cuarta regla penaliza el código QR si más de la mitad de los módulos son oscuros o claros, con una penalización mayor para una diferencia mayor.

Evalué la condición 1

Para la primera condición de evaluación, verifique cada fila una por una. Si hay cinco módulos consecutivos del mismo color, suma 3 a la penalización. Si hay más módulos del mismo color después de los primeros cinco, agregue 1 por cada módulo adicional del mismo color. Luego, verifique cada columna una por una, verificando la misma condición. Sume el total horizontal y vertical para obtener el puntaje de penalización número 1.

Las siguientes imágenes ilustran el proceso de evaluación del código QR de esta manera. En este ejemplo, la penalización horizontal es 88 y la penalización vertical es 92. Por lo tanto, la puntuación de penalización número 1 es 92 + 88 = 180.

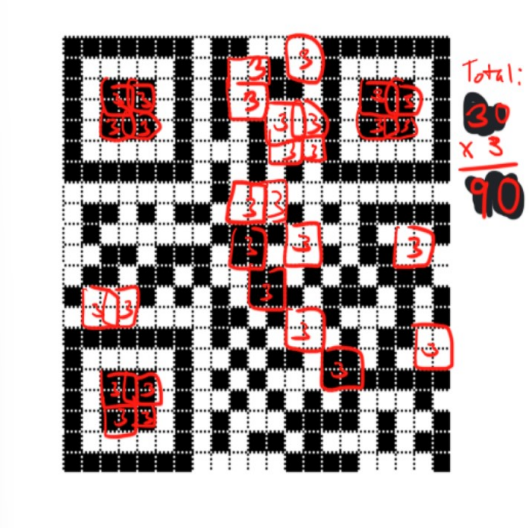


Evalué la condición 2

Para la segunda condición de evaluación, busque áreas del mismo color que tengan al menos 2×2 módulos o más. La especificación del código QR dice que para un bloque de color sólido de tamaño $m \times n$, la puntuación de penalización es $3 \times (m-1) \times (n-1)$. Sin embargo, la especificación del código QR no especifica cómo calcular la penalización cuando hay varias formas de dividir los bloques de colores sólidos.

Por lo tanto, en lugar de buscar bloques de colores sólidos de más de 2×2, simplemente agregue 3 al puntaje de penalización por cada bloque de 2×2 del mismo color en el código QR, asegurándose de contar los bloques de 2×2 superpuestos. Por ejemplo, un bloque de 3×2 del mismo color debe contarse como dos bloques de 2×2, uno superpuesto al otro.

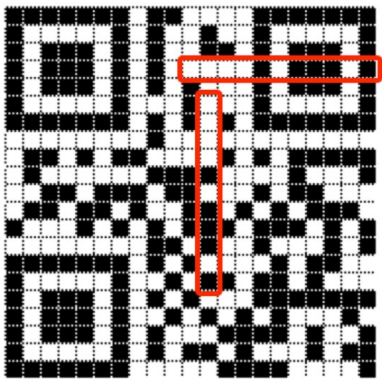
La siguiente imagen ilustra cómo calcular la regla de penalización número 2.



Evalué la condición 3

La tercera regla de penalización busca patrones de oscuro-claro-oscuro-oscuro-oscuro-claro-oscuro que tienen cuatro módulos de luz a cada lado. En otras palabras, busca cualquiera de los siguientes dos patrones:

Cada vez que encuentre este patrón, agregue 40 al puntaje de penalización. En el siguiente ejemplo, hay dos patrones de este tipo. Por lo tanto, la puntuación de penalización número 3 es 80.



Evalué la condición 4

La condición de evaluación final se basa en la relación de módulos claros a módulos oscuros. Para calcular esta regla de penalización, siga los siguientes pasos:

(1) Cuente el número total de módulos en la matriz.

(2) Cuente cuántos módulos oscuros hay en la matriz.

(3) Calcule el porcentaje de módulos en la matriz que están oscuros:

$$\left(\frac{\text{Módulos oscuros}}{\text{Módulos totales}}\right)(100)$$

(4) Determine el múltiplo anterior y siguiente de cinco de este porcentaje. Por ejemplo, para el 43 por ciento, el múltiplo anterior de cinco es 40 y el siguiente múltiplo de cinco es 45.

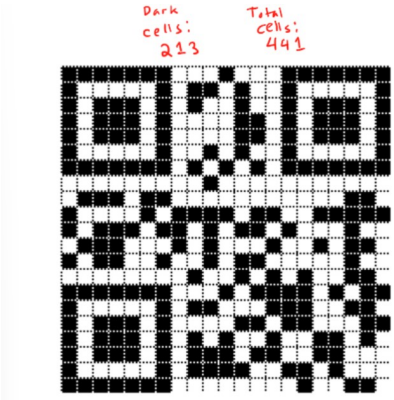
(5) Resta 50 de cada uno de estos múltiplos de cinco y toma el valor absoluto del resultado. Por ejemplo, $|40-50| = |-10| = 10$ y $|45-50| = |-5| = 5$.

(5) Divide cada uno de estos por cinco. Por ejemplo, $\frac{10}{5} = 2$ y $\frac{5}{5} = 1$

(1) Finalmente, tome el menor de los dos números y multiplíquelo por 10. En este ejemplo, el número menor es 1, por lo que el resultado es 10. Este es el puntaje de penalización número 4.

.

Para otro ejemplo, en la imagen a continuación, la cantidad total de módulos es 441 y la cantidad total de módulos oscuros es 213.



El porcentaje de módulos oscuros es $100\left(\frac{213}{441}\right) \approx 48{,}2993$. El múltiplo de 5 previo es 45 y el siguiente múltiplo de cinco es 50. Restando 50 a ambos números y tomando el valor absoluto de cada uno, obtenemos

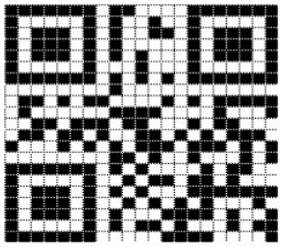
que $|45 - 50| = 5$ y $|50 - 50| = 0$, dividimos casa uno entre 5: $\frac{5}{5} = 1$ y $\frac{0}{5} = 0$. El menor de esos dos números es 0. Multiplicado por 10, sigue siendo 0. Por lo tanto, la puntuación de penalización número 4 es 0 en este ejemplo.

Suma las cuatro puntuaciones de penalización

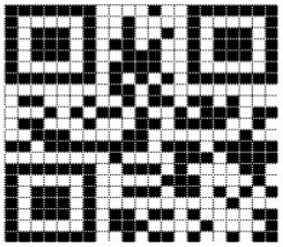
Para completar la evaluación de un código QR, agregue los cuatro puntajes de penalización. El total es el puntaje general de penalización del código QR.

Elija la puntuación de penalización más baja para los ocho patrones de máscara

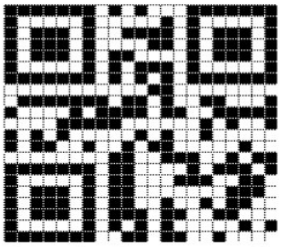
Las siguientes imágenes muestran ocho códigos QR, uno para cada patrón de máscara. Los ocho códigos QR de este ejemplo codifican los mismos datos. Como se muestra, el patrón de máscara con el puntaje de penalización más bajo es el patrón de máscara 0. Por lo tanto, en este ejemplo, el codificador QR debe usar el patrón de máscara 0 al generar el código QR final.



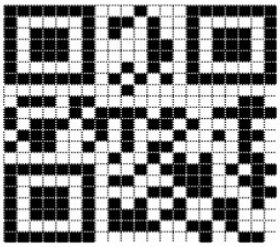
Mask Pattern 0
Penalty 1: 180
Penalty 2: 90
Penalty 3: 80
Penalty 4: 0
Total: 350



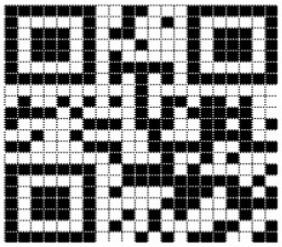
Mask Pattern 1
Penalty 1: 172
Penalty 2: 129
Penalty 3: 120
Penalty 4: 0
Total: 421



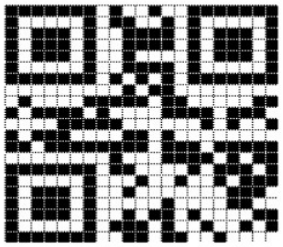
Mask Pattern 2
Penalty 1: 206
Penalty 2: 141
Penalty 3: 160
Penalty 4: 0
Total: 507



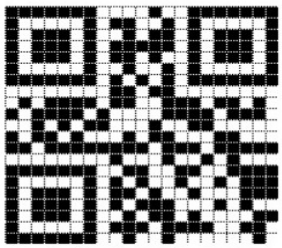
Mask Pattern 3
Penalty 1: 180
Penalty 2: 141
Penalty 3: 120
Penalty 4: 2
Total: 443



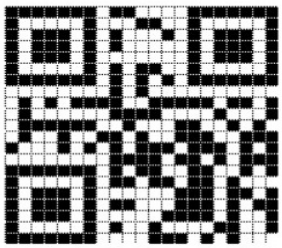
Mask Pattern 4
Penalty 1: 195
Penalty 2: 138
Penalty 3: 200
Penalty 4: 0
Total: 553



Mask Pattern 5
Penalty 1: 189
Penalty 2: 156
Penalty 3: 200
Penalty 4: 2
Total: 547



Mask Pattern 6
Penalty 1: 171
Penalty 2: 102
Penalty 3: 80
Penalty 4: 4
Total: 357



Mask Pattern 7
Penalty 1: 197
Penalty 2: 123
Penalty 3: 200
Penalty 4: 0
Total: 520

9. Información de formato y versión

El paso final para crear un código QR es crear las cadenas de formato y versión, luego colocarlas en las ubicaciones correctas en el código QR. Esta página explica cómo generar las cadenas de formato y versión, e ilustra dónde se colocan en el código QR.

9.1. Cadena de formato

La cadena de información de formato codifica qué nivel de corrección de errores y qué patrón de máscara está en uso en el código QR actual. Dado que hay cuatro niveles de corrección de errores posibles (L, M, Q y H) y ocho patrones de máscara posibles, hay 32 (4 veces 8) cadenas de información de formato posibles. La siguiente sección explica cómo generar estas cadenas de formato. Para obtener una lista completa de las 32 cadenas de formato, consulte la tabla de cadenas de formato 10.8.

Generar la cadena de formato

La cadena de formato siempre tiene una longitud de 15 bits. Para crear la cadena, primero crea una cadena de cinco bits que codifica el nivel de corrección de errores y el patrón de máscara en uso en este código QR. Luego usa esos cinco bits para generar diez bits de corrección de errores. Los quince bits resultantes se someten a XOR con el patrón de máscara 101010000010010. Este proceso se explica en detalle a continuación.

Los bits de corrección de errores

El primer paso para crear la cadena de formato es obtener los dos bits que especifican el nivel de corrección de errores en uso en el código QR. La siguiente tabla muestra las secuencias de bits para cada nivel de corrección de errores.

Nivel CE	Bits	Entero
L	01	1
M	00	0
Q	11	3
H	10	2

Note que los números no van en orden de 0, 1, 2, 3 en la tabla.

Los bits del patrón de máscara

Para los patrones de máscara, consulte la página de patrones de máscara con código QR para encontrar el número de máscara que corresponde a cada patrón. Convierta el número en una cadena binaria de tres bits.

Ejemplo de cadena de formato de cinco bits

Por ejemplo, si hemos utilizado el nivel de corrección de errores L y el patrón de máscara 4, la secuencia binaria de cinco bits se crea así:

- 01 (indicador de nivel de corrección de errores L)
- 100 (binario para 4, es decir, patrón de máscara 4)

Resultado: 01100

Generar bits de corrección de errores para cadena de formato

Ahora que tenemos cinco bits para la cadena de formato, debemos usarla para generar diez bits de corrección de errores. Este paso usa la corrección de errores de Reed-Solomon, pero es un poco más fácil porque en este caso, los polinomios no contienen más de quince términos y sus coeficientes son todos 1 o 0. A continuación desglozamos los pasos:

(1) **Obtenga el polinomio generador:** Al generar las palabras clave de corrección de errores de la cadena de formato, la especificación del código QR dice que se use el siguiente polinomio generador:

$$x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$$

Podemos convertir esto en una cadena binaria tomando solo los coeficientes de cada monomio (incluyendo los que no aparecen). En otras palabras, la cadena binaria que representa el polinomio generador para este paso es:

$$10100110111.$$

(2) **Calcular los bits de corrección de errores**: El siguiente paso es dividir los bits de cadena de formato 01100 (del ejemplo) por el polinomio generador (10100110111 del paso anterior). Para hacer esto, primero cree una cadena de 15 bits colocando diez ceros a la derecha de la cadena de formato, así:

$$01100 \longrightarrow 011000000000000$$

Ahora realizamos la división. Los pasos son:

(*i*) Rellene la cadena del polinomio generador de la derecha con ceros para que tenga la misma longitud que la cadena de formato actual.

(*ii*) XOR la cadena de polinomio generador rellenada con la cadena de formato actual.

(*iii*) Quite los 0 del lado izquierdo del resultado.

Debemos dividir los polinomios hasta que la cadena de formato resultante tenga 10 bits o menos. Por lo tanto, antes de cada paso de división, asegúrese de que la cadena de formato actual tenga 11 bits o más (11 bits es la longitud del polinomio generador). Nuestra cadena actual, 11000000000000, tiene una longitud de 14 bits. Para ejemplificar esto, usaremos el ejemplo previo:

Primera división: Primero, rellena la cadena polinomial del generador a la derecha con 0 para que tenga la misma longitud que la cadena de formato actual:

$$\textbf{Cadena de formato} = 01100 \Longrightarrow 011000000000000 \Longrightarrow 11000000000000$$
$$\textbf{Polinomio generador} = 10100110111 \Longrightarrow 10100110111000$$

Ahora, XOR la cadena de formato con el polinomio generador:

$$11000000000000 \text{ XOR } 10100110111000 = 01100110111000$$

Y elimine los 0 del lado izquierdo del resultado:

$$01100110111000 \Longrightarrow 1100110111000$$

Segunda división: La cadena resultante 1100110111000 tiene una longitud de 13 bits, por lo que como tiene 11 bits o más, podemos continuar. Rellene el polinomio generador a la derecha para que tenga la misma longitud que la cadena de formato actual:

$$10100110111 \Longrightarrow 1010011011100$$

XOR la cadena de formato actual con el polinomio generador rellenado:

$$1100110111000 \text{ XOR } 1010011011100 = 110101100100$$

Tercera división: La cadena resultante 110101100100 tiene una longitud de 12 bits, por lo que como tiene 11 bits o más, podemos continuar. Rellene el polinomio generador a la derecha para que tenga la misma longitud que la cadena de formato actual:

$$10100110111 \Longrightarrow 101001101110$$

XOR la cadena de formato actual con el polinomio generador rellenado:

$$110101100100 \text{ XOR } 101001101110 = 011100001010$$

Y elimine los 0 del lado izquierdo del resultado:

$$011100001010 \Longrightarrow 11100001010$$

Cuarta división: La cadena resultante 11100001010 tiene una longitud de 11 bits, por lo que como tiene 11 bits o más, podemos continuar. No es necesario rellenar el polinomio generador, porque la cadena de formato ahora tiene la misma longitud. XOR la cadena de formato actual con el polinomio generador:

$$11100001010 \text{ XOR } 10100110111 = 1000111101$$

El resultado tiene una longitud de 10 bits, por lo que hemos terminado con el paso de división. **Si el resultado fuera menor de 10 bits, lo rellenaríamos a la IZQUIERDA con 0 para que tenga 10 bits de largo**.

Juntar los bits de formato y corrección de errores

Cree una cadena utilizando los cinco bits originales de la información de formato (el indicador de nivel de corrección de errores y el indicador de patrón de máscara), seguidos de los bits de corrección de errores que acabamos de generar.

- La cadena de formato de cinco bits: 01100
- La cadena de corrección de errores de diez bits del paso de división: 1000111101
- La cadena combinada: 011001000111101

XOR con la cadena de máscara

La especificación del código QR dice XOR el resultado con la siguiente cadena binaria: 101010000010010

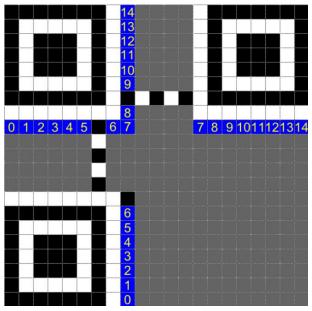
$$011001000111101 \text{ XOR } 101010000010010 = 110011000101111$$

La cadena de formato final para un código con nivel de corrección de errores L y patrón de máscara 4 es 110011000101111. Para obtener una lista de las 32 cadenas de formato, consulte la tabla de cadenas de formato en el Anexo 10.8.

Coloque la cadena de formato en el código QR

La cadena de información de formato se coloca debajo de los patrones de búsqueda superiores y a la derecha de los patrones de búsqueda situados más a la izquierda, como se muestra en la imagen a continuación. El número 0 en la imagen se refiere al bit más significativo de la cadena de formato y el número 14 se refiere al bit menos significativo. En otras palabras, usando la cadena de formato 110011000101111 del ejemplo anterior, los números en la imagen a continuación corresponden a los bits de cadena de formato de la siguiente manera:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	0	0	1	1	0	0	0	1	0	1	1	1	1



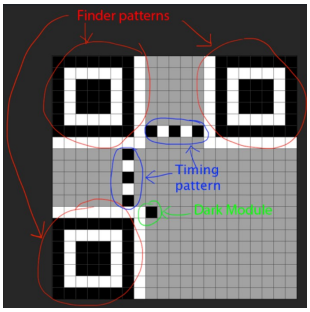
La siguiente imagen muestra la cadena de formato de ejemplo, 110011000101111, en un código QR versión 1.

Módulo oscuro

Cada código QR debe tener un píxel oscuro, también conocido como módulo oscuro, en las coordenadas (8, 4*versión + 9). Es decir, la coordenada y del módulo oscuro es:

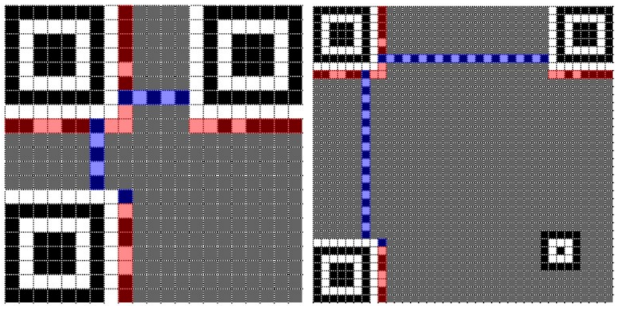
- versión 1: 4*1 + 9 = 13
- versión 2: 4*2 + 9 = 17
- versión 3: 4*3 + 9 = 21
- Y así sucesivamente.

Esto significa que el módulo oscuro siempre está a la derecha de la esquina superior derecha del patrón del buscador inferior izquierdo. Esto se muestra en la imagen de abajo.



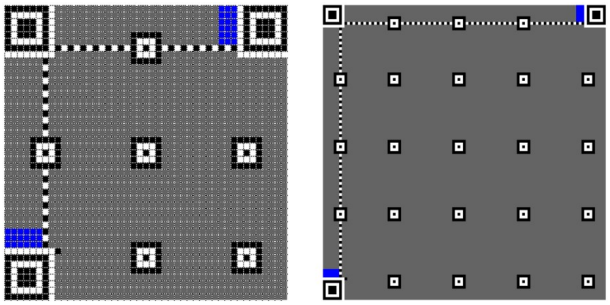
Para códigos más grandes

Independientemente del tamaño del código QR, los bits de cadena de formato se colocan debajo de los patrones de búsqueda superiores ya la derecha de los patrones de búsqueda situados más a la izquierda, como se ilustra a continuación. Tenga en cuenta que ambas imágenes tienen la misma cadena de formato, 110011000101111, del ejemplo anterior en esta página. La cadena de formato se resalta en rojo y el patrón de tiempo y el módulo oscuro se resaltan en azul.



9.2. Información de versión

Si el código QR es la versión 7 o superior, debe incluir una cadena de información de la versión de 18 bits en las esquinas inferior izquierda y superior derecha del código QR. (Para obtener una lista completa de todas las cadenas de información de versión posibles, consulte las tablas de formato y versión en el anexo 10.8). Las áreas de información de versión son los rectángulos azules de 6x3 que se muestran en las imágenes a continuación. La información de la versión se coloca junto a los patrones del buscador sin importar el tamaño del código QR. La imagen de la izquierda es un código de la versión 7 y la imagen de la derecha es un código QR de la versión 22.



Generación de la cadena de información de la versión

La especificación del Código QR dice que se use el código Golay (18, 6) para la cadena de información de la versión. Como tal, la cadena de información de la versión es una cadena de 18 bits que consta de una cadena binaria de seis bits que codifica la versión QR, seguida de una cadena de 12 bits de corrección de errores. La cadena completa tiene una longitud de 18 bits.

Obtenga el polinomio generador

La especificación del código QR dice que se use el siguiente polinomio generador para este paso:

$$x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^5 + x^2 + 1$$

Como se explicó en la sección de cadena de formato anteriormente en esta página, podemos representar este polinomio generador con la siguiente cadena binaria: 1111100100101

Realiza la División

A partir de aquí, podemos seguir los mismos pasos de división que usamos para generar la cadena de información de formato, excepto que en este caso rellenamos las cadenas iniciales para que tengan 18 bits en lugar de 15, y nos detenemos cuando la cadena de bits actual tiene 12 o menos. bits de largo, en lugar de 10 o menos. Comience creando una cadena binaria de seis bits que represente el número de versión. Por ejemplo, para un código de la versión 7, el equivalente binario de seis bits de 7 es: 000111 Convierta esto en una cadena de 18 bits rellenando a la derecha con 0:

000111000000000000

Y quita los 0 del lado izquierdo:

111000000000000

Ahora rellena el polinomio generador de la derecha con ceros para que tenga la misma longitud.

$$1111100100101 \Longrightarrow 111110010010100$$

XOR la cadena de versión y el polinomio generador acolchado:

1110000000000 XOR 111110010010100 = 110010010100

Esta cadena ya tiene la longitud requerida de 12, por lo que no se requieren más divisiones. Al igual que con la cadena de información de formato, **si el resultado es menor que 12, debe rellenarse a la IZQUIERDA con 0 para que tenga una longitud de 12 bits**. Finalmente, coloque la cadena de versión original de seis bits a la izquierda del resultado del último paso.

- cadena de versión: 000111
- cadena de corrección de errores desde arriba: 110010010100
- cadena de información de la versión final: 000111110010010100

Para obtener una lista completa de todas las cadenas de información de versión posibles, consulte las tablas de formato y versión.

Coloque la cadena de versión en el código QR

Hay dos áreas rectangulares donde se debe colocar la cadena de información de la versión: una en la parte inferior izquierda y otra en la parte superior derecha. Enumere la cadena de la siguiente manera, usaré la cadena de información de la versión 7, 000111110010010100, para ejemplificar la enumeración:

17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0	0	0	1	1	1	1	1	0	0	1	0	0	1	0	1	0	0

Bloque de información de la versión inferior izquierda

El bloque de información de la versión inferior izquierda tiene 3 píxeles de alto y 6 píxeles de ancho. La siguiente tabla explica cómo organizar los bits de la cadena de información de la versión en el área de información de la versión inferior izquierda. El 0 representa el bit más a la DERECHA (menos significativo) de la cadena de información de versión, y el 17 representa el bit más a la IZQUIERDA (más significativo) de la cadena de información de versión.

00	03	06	09	12	15
01	04	07	10	13	16
02	05	08	11	14	17

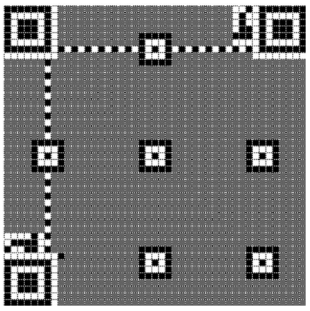
Bloque de información de la versión superior derecha

El bloque de información de la versión superior derecha tiene 3 píxeles de ancho y 6 píxeles de alto. La siguiente tabla explica cómo organizar los bits de la cadena de información de la versión en el área de información de la versión superior derecha. El 0 representa el bit más a la DERECHA (menos significativo) de la cadena de información de versión, y el 17 representa el bit más a la IZQUIERDA (más significativo) de la cadena de información de versión.

00	01	02
03	04	05
06	07	08
09	10	11
12	13	14
15	16	17

Ejemplo de cadena de información de la versión 7

La siguiente imagen es un código QR de la versión 7, que usa la cadena de información de la versión 000111110010010100. Tenga en cuenta que las áreas de información de la versión se completan de acuerdo con el patrón descrito en las tablas anteriores.



Salida de la matriz final

El paso final, después de agregar información de formato y versión a la matriz QR, es generar la matriz final, utilizando el patrón de máscara que se determinó que tiene el puntaje de penalización más bajo.

Agrega la zona tranquila

Tenga en cuenta que la especificación del código QR requiere que la matriz QR esté rodeada por una zona silenciosa: un área de módulos de luz de 4 módulos de ancho

Salida final

La siguiente imagen es un código 1-Q de HOLA MUNDO(En ingles), codificado en modo alfanumérico.

Conclusión

Esto concluye el tutorial del código QR. En resumen, los códigos QR se generan con los siguientes pasos:

(1) Determinar qué modo de codificación usar.

(2) Codificar los datos.

(3) Generar palabras clave de corrección de errores.

(4) Intercalar bloques si es necesario.

(5) Coloque los bits de datos y corrección de errores en la matriz.

(6) Aplique los patrones de máscara y determine cuál resulta en la penalización más baja.

(7) Agregar información de formato y versión.

10. Anexo

En esta sección podrá encontrar todas las tablas que usaremos en este archivo.

10.1. Modos (numérico, alfanumérico y byte)

Cuadro 6: Modo numérico

Carácter	0	1	2	3	4	5	6	7	8	9
Codificación	0	1	2	3	4	5	6	7	8	9

Cuadro 7: Modo alfanumérico

Carácter	Codificación	Carácter	Codificación
0	0	N	23
1	1	O	24
2	2	P	25
3	3	Q	26
4	4	R	27
5	5	S	28
6	6	T	29
7	7	U	30
8	8	V	31
9	9	W	32
A	10	X	33
B	11	Y	34
C	12	Z	35
D	13	(SPACE)	36
E	14	$	37
F	15	%	38
G	16	*	39
H	17	+	40
I	18	-	41
J	19	.	42
K	20	/	43
L	21	:	44
M	22		

Cuadro 8: Modo byte

Carácter	Codificación	Carácter	Codificación	Carácter	Codificación
^@	0	+	43	V	86
^A	1	,	44	W	87
^B	2	-	45	X	88
^C	3	.	46	Y	89
^D	4	/	47	Z	90
^E	5	0	48	[	91
^F	6	1	49	\	92
^G	7	2	50	]	93
^H	8	3	51	^	94
^I	9	4	52	_	95
^J	10	5	53	‘	96
^K	11	6	54	a	97
^L	12	7	55	b	98
^M	13	8	56	c	99
^N	14	9	57	d	100
^O	15	:	58	e	101
^P	16	;	59	f	102
^Q	17	<	60	g	103
^R	18	=	61	h	104
^S	19	>	62	i	105
^T	20	?	63	j	106
^U	21	@	64	k	107
^V	22	A	65	l	108
^W	23	B	66	m	109

Carácter	Codificación	Carácter	Codificación	Carácter	Codificación
^X	24	C	67	n	110
^Y	25	D	68	o	111
^Z	26	E	29	p	112
^[	27	F	70	q	113
^\	28	G	71	r	114
^]	29	H	72	s	115
^^	30	I	73	t	116
^_	31	J	74	u	117
(SPACE)	32	K	75	v	118
!	33	L	76	w	119
"	34	M	77	x	120
#	35	N	78	y	121
$	36	O	79	z	122
%	37	P	80	{	123
&	38	Q	81	|	124
’	39	R	82	}	125
(	40	S	83	∼	126
)	41	T	84	^?	127
*	42	U	85		128
	129	¬	172	×	215
	130		173	Ø	216
	131	®	174	Ù	217
	132	¯	175	Ú	218
	133	○	176	Û	219
	134	±	177	Ü	220
	135	^{2}	178	Ý	221
	136	^{3}	179	Þ	222
	137	´	180	ß	223
	138	μ	181	à	224
	139	,	182	á	225
	140	·	183	â	226
	141	¶	184	ã	227
	142	^{1}	185	ä	228
	143	º	186	å	229
	144	»	187	æ	230
	145	1/4	188	ç	231
	146	1/2	189	è	232
	147	3/4	190	é	233
	148	¿	191	ê	234
	149	À	192	ë	235
	150	Á	193	ì	236
	151	Â	194	í	237
	152	Ã	195	î	238
	153	Ä	196	ï	239
	154	Å	197	ð	240
	155	Æ	198	ñ	241
	156	Ç	199	ò	242
	157	È	200	ó	243
	158	É	201	ô	244
	159	Ê	202	õ	245
	160	Ë	203	ö	246
¡	161	Ì	204	÷	247
¢	162	Í	205	ø	248

Carácter	Codificación	Carácter	Codificación	Carácter	Codificación
£	163	Î	206	ù	249
¤	164	Ï	207	ú	250
¥	165	Ð	208	û	251
¦	166	Ñ	209	ü	252
§	167	Ò	210	ý	253
¨	168	Ó	211	þ	254
©	169	Ô	212	ÿ	255
ª	170	Õ	213		
«	171	Ö	214		

10.2. Capacidades de caracteres por versión, modo y corrección de errores

Cuadro 9: Tabla de capacidades

Versión	Nivel de corrección	Modo numérico	Modo alfanumérico	Modo byte	Modo Kanji
1	L	41	25	17	10
	M	34	20	14	8
	Q	27	16	11	7
	H	17	10	7	4
2	L	77	47	32	20
	M	63	38	26	16
	Q	48	29	20	12
	H	34	20	14	8
3	L	127	77	53	32
	M	101	61	42	26
	Q	77	47	32	20
	H	58	35	24	15
4	L	187	114	78	48
	M	149	90	62	38
	Q	111	67	46	28
	H	82	50	34	21
5	L	255	154	106	65
	M	202	122	84	52
	Q	144	87	60	37
	H	106	64	44	27
6	L	322	195	134	82
	M	255	154	106	65
	Q	178	108	74	45
	H	139	84	58	36
7	L	370	224	154	95
	M	293	178	122	75
	Q	207	125	86	53
	H	154	93	64	39
8	L	461	279	192	118
	M	365	221	152	93
	Q	259	157	108	66
	H	202	122	84	52
9	L	552	335	230	141
	M	432	262	180	111
	Q	312	189	130	80
	H	235	143	98	60
10	L	652	395	271	167
	M	513	311	213	131
	Q	364	221	151	93
	H	288	174	119	74
11	L	772	468	321	198
	M	604	266	251	155
	Q	427	259	177	109
	H	331	200	137	85
12	L	883	535	367	226
	M	691	419	287	177
	Q	489	296	203	125
	H	374	227	155	96
13	L	1022	619	425	262
	M	796	483	331	204

Versión	Nivel de corrección	Modo numérico	Modo alfanumérico	Modo byte	Modo Kanji
	Q	580	352	241	149
	H	427	259	177	109
14	L	1101	667	458	282
	M	871	528	362	223
	Q	621	376	258	159
	H	468	283	194	120
15	L	1250	758	520	320
	M	991	600	412	254
	Q	703	426	292	180
	H	530	321	220	136
16	L	1408	854	586	361
	M	1082	656	450	277
	Q	775	470	322	198
	H	602	365	250	154
17	L	1548	938	644	397
	M	1212	734	504	310
	Q	876	531	364	224
	H	674	408	280	173
18	L	1725	1046	718	442
	M	1346	816	560	345
	Q	948	574	394	243
	H	746	452	310	191
19	L	1903	1153	792	488
	M	1500	909	624	384
	Q	1063	644	442	272
	H	813	493	338	208
20	L	2061	1249	858	528
	M	1600	970	666	410
	Q	1159	702	482	297
	H	919	557	382	235
21	L	2232	1352	929	572
	M	1708	1035	711	438
	Q	1224	742	509	314
	H	969	587	403	248
22	L	2409	1460	1003	618
	M	1872	1134	779	480
	Q	1358	823	565	348
	H	1056	640	439	270
23	L	2620	1588	1091	672
	M	2059	1248	857	528
	Q	1468	890	611	376
	H	1108	672	461	284
24	L	2812	1704	1171	721
	M	2188	1326	911	561
	Q	1588	963	661	407
	H	1228	744	511	315
25	L	3057	1853	1273	784
	M	2395	1451	997	614
	Q	1718	1041	715	440
	H	1286	779	535	330
26	L	3283	1990	1367	842
	M	2544	1542	1059	652
	Q	1804	1094	751	462
	H	1425	864	593	365

Versión	Nivel de corrección	Modo numérico	Modo alfanumérico	Modo byte	Modo Kanji
27	L	3517	2132	1465	902
	M	2701	1637	1125	692
	Q	1933	1172	805	496
	H	1501	910	625	385
28	L	3669	2223	1528	940
	M	2857	1732	1190	732
	Q	2085	1263	868	534
	H	1581	958	658	405
29	L	3909	2369	1628	1002
	M	3035	1839	1264	778
	Q	2181	1322	908	556
	H	1677	1016	698	430
30	L	4158	2520	1732	1066
	M	3289	1994	1370	843
	Q	2358	1429	982	604
	H	1782	1080	742	457
31	L	4417	2677	1840	1132
	M	3486	2113	1452	894
	Q	2473	1499	1030	634
	H	1897	1150	790	486
32	L	4686	2840	1952	1201
	M	3693	2238	1538	947
	Q	2670	1618	1112	684
	H	2022	1226	842	518
33	L	4965	3009	2068	1273
	M	3909	2369	1628	1002
	Q	2805	1700	1168	719
	H	2157	1307	898	553
34	L	5253	3183	2188	1347
	M	4134	2506	1722	1060
	Q	2949	1787	1228	756
	H	2301	1394	958	590
35	L	5529	3351	2303	1417
	M	4343	2632	1809	1113
	Q	3081	1867	1283	790
	H	2361	1431	983	605
36	L	5836	3537	2431	1496
	M	4588	2780	1911	1176
	Q	3244	1966	1351	832
	H	2524	1530	1051	647
37	L	6153	3729	2563	1577
	M	4775	2894	1989	1224
	Q	3417	2071	1423	876
	H	2625	1591	1093	673
38	L	6479	3927	2699	1661
	M	5039	3054	2099	1292
	Q	3599	2181	1499	923
	H	2735	1658	1139	701
39	L	6743	4087	2809	1729
	M	5313	3220	2213	1362
	Q	3791	2298	1579	972
	H	2927	1774	1219	750
40	L	7089	4296	2953	1817
	M	5596	3391	2331	1435

Versión	Nivel de corrección	Modo numérico	Modo alfanumérico	Modo byte	Modo Kanji
	Q	3993	2420	1663	1024
	H	3057	1852	1273	784

10.3. Palabras de código de corrección de errores e información de bloque

La siguiente tabla enumera la cantidad de palabras del código de corrección de errores que debe generar para cada versión y nivel de corrección de errores del código QR. Estos valores se pueden usar para determinar cuántos bytes de datos y de corrección de errores se requieren para un bloque Reed-Solomon dado.

Cuadro 10: Tabla de corrección de errores

Versión y nivel de CE	Número total de palabras clave de datos para esta versión y nivel CE	Palabras de código CE por bloque	Números de bloques en el grupo 1	Número de palabras de código de datos en cada un de los bloques del grupo 1	Números de bloques en el grupo 2	Número de palabras de códigos de datos en cada uno de los bloques del grupo 2	Palabras clave de datos totales
1-L	19	7	1	19			(19*1) = 19
1-M	16	10	1	16			(16*1) = 16
1-Q	13	13	1	13			(13*1) = 13
1-H	9	17	1	9			(9*1) = 9
2-L	34	10	1	34			(34*1)=34
2-M	28	16	1	28			(28*1)=28
2-Q	22	22	1	22			(22*1)=22
2-H	16	28	1	16			(16*1)=16
3-L	55	15	1	55			(55*1)=55
3-M	44	26	1	44			(44*1)=44
3-Q	34	18	2	17			(17*2)=34
3-H	26	22	2	13			(13*2)=26
4-L	80	20	1	80			(80*1)=80
4-M	64	18	2	32			(32*2)=64
4-Q	48	26	2	24			(24*2)=48
4-H	36	16	4	9			(9*4)=36
5-L	108	26	1	108			(108*1)=108
5-M	86	24	2	43			(43*2) = 86
5-Q	62	18	2	15	2	16	(15*2)+(16*2)=62
5-H	46	22	2	11	2	12	(11*2)+(12*2)=46
6-L	136	18	2	68			(68*2)=136
6-M	108	16	4	27			(27*4) = 108
6-Q	76	24	4	19			(19*4) = 76
6-H	60	28	4	15			(15*4) = 60
7-L	156	20	2	78			(78*2)=156
7-M	124	18	4	31			(31*4)=124
7-Q	88	18	2	14	4	15	(31*4)+(15*4)=88
7-H	66	26	4	13	1	14	(13*4)+(14*1)=66
8-L	194	24	2	97			(97*2) = 194
8-M	154	22	2	38	2	39	(38*2) + (39*2) = 154

Versión y nivel de CE	Número total de palabras clave de datos para esta versión y nivel CE	Palabras de código CE por bloque	Números de bloques en el grupo 1	Número de palabras de código de datos en cada un de los bloques del grupo 1	Números de bloques en el grupo 2	Número de palabras de códigos de datos en cada uno de los bloques del grupo 2	Palabras clave de datos totales
8-Q	110	22	4	18	2	19	(18*4) + (19*2) = 110
8-H	86	26	4	14	2	15	(14*4) + (15*2) = 86
9-L	232	30	2	116			(116*2)=232
9-M	182	22	3	36	2	37	(36*3)+(37*2)=182
9-Q	132	20	4	16	4	17	(16*4)+(17*4)=132
9-H	100	24	4	12	4	13	(12*4)+(13*4)=100
10-L	274	18	2	68	2	69	(68*2)+(69*2)=274
10-M	216	26	4	43	1	44	(43*4)+(69*2)=216
10-Q	154	24	6	19	2	20	(19*6)+(20*2)=154
10-H	122	28	6	15	2	16	(15*6)+(16*2)=122
11-L	324	20	4	81			(81*4) = 324
11-M	254	30	1	50	4	51	(50*1)+(51*4)=254
11-Q	180	28	4	22	4	23	(22*4)+(23*4)=180
11-H	140	24	3	12	8	13	(12*3)+(13*4)=140
12-L	370	24	2	92	2	93	(92*2)+(93*2)=370
12-M	290	22	6	36	2	37	(36*6)+(37*2)=290
12-Q	206	26	4	20	6	21	(20*4)+(21*6)=206
12-H	158	28	7	14	4	15	(14*7)+(15*4)=158
13-L	428	26	4	107			(107*4)=428
13-M	334	22	8	37	1	38	(37*8)+(38*1)=334
13-Q	244	24	8	20	4	21	(20*8)+(21*4)=244
13-H	180	22	12	11	4	12	(11*12)+(12*4)=180
14-L	461	30	3	115	1	116	(115*3)+(116*1)=461
14-M	365	24	4	40	5	41	(40*4)+(41*5)=365
14-Q	261	20	11	16	5	17	(16*11)+(17*5)=261
14-H	197	24	11	12	5	13	(12*11)+(13*5)=197
15-L	523	22	5	87	1	88	(87*5)+(88*1)=523
15-M	415	24	5	41	5	42	(41*5)+(42*5)=415
15-Q	295	30	5	24	7	25	(24*5)+(25*7)=295
15-H	223	24	11	12	7	13	(12*11)+(13*7)=223
16-L	589	24	5	98	1	99	(98*5)+(99*1)=589
16-M	453	28	7	45	3	46	(45*7)+(46*3)=453
16-Q	325	24	15	19	2	20	(19*15)+(20*2)=325
16-H	253	30	3	15	13	16	(15*3)+(16*13)=253
17-L	647	28	1	107	5	108	(107*1)+(108*5)=647
17-M	507	28	10	46	1	47	(46*10)+(47*1)=507
17-Q	367	28	1	22	15	23	(22*1)+(23*15)=367
17-H	283	28	2	14	17	15	(14*2)+(15*17)=283
18-L	721	30	5	120	1	121	(120*5)+(121*1)=721
18-M	563	26	9	43	4	44	(43*9)+(44*4)=563
18-Q	397	28	17	22	1	23	(22*17)+(23*1)=397
18-H	313	28	2	14	19	15	(14*2)+(15*19)=313

Versión y nivel de CE	Número total de palabras clave de datos para esta versión y nivel CE	Palabras de código CE por bloque	Números de bloques en el grupo 1	Número de palabras de código de datos en cada un de los bloques del grupo 1	Números de bloques en el grupo 2	Número de palabras de códigos de datos en cada uno de los bloques del grupo 2	Palabras clave de datos totales
19-L	795	28	3	113	4	114	(113*3)+(114*4)=795
19-M	627	26	3	44	11	45	(44*3)+(45*11)=627
19-Q	445	26	17	21	4	22	(21*17)+(22*4)=445
19-H	341	26	9	13	16	12	(13*9)+(14*16)=341
20-L	861	28	3	107	5	108	(107*3)+(108*5)=861
20-M	669	26	3	41	13	42	(41*3)+(42*13)=669
20-Q	485	30	15	24	5	25	(24*15)+(25*5)=485
20-H	385	28	15	15	10	16	(15*15)+(16*10)=385
21-L	932	28	4	116	4	117	(116*4)+(117*4)=932
21-M	714	26	17	42			(42*17)=714
21-Q	512	28	17	22	6	23	(22*17)+(23*6)=512
21-H	406	30	19	16	6	17	(16*19)+(17*6)=406
22-L	1006	28	2	111	7	112	(111*2)+(112*7)=1009
22-M	782	28	17	46			(46*17)=782
22-Q	568	30	7	24	16	25	(24*7)+(25*16)=568
22-H	442	24	34	13			(13*34) = 442
23-L	1094	30	4	121	5	122	(121*4)+(122*5)=1094
23-M	860	28	4	47	14	48	(47*4)+(48*14)=860
23-Q	614	30	11	24	14	25	(24*11)+(25*14)=614
23-H	464	30	16	15	14	16	(15*16)+(16*14)=464
24-L	1174	30	6	117	4	118	(117*6)+(118*4)=1174
24-M	914	28	6	45	14	46	(45*6)+(46*14)=914
24-Q	664	30	11	24	16	25	(24*11)+(25*16)=664
24-H	514	30	30	16	2	17	(16*30)+(17*2)=514
25-L	1276	26	8	106	4	107	(106*8)+(17*2)=514
25-M	1000	28	8	47	13	48	(106*8)+(107*4)=1276
25-Q	718	30	7	24	22	25	(47*8)+(48*13)=1000
25-H	538	30	22	15	13	16	(15*22)+(16*13)=538
26-L	1370	28	10	114	2	115	(114*10)+(115*2)=1370
26-M	1062	28	19	46	4	47	(46*19)+(47*4)=1062
26-Q	754	28	28	22	6	23	(22*28)+(23*6)=754
26-H	596	30	33	16	4	17	(16*33)+(17*4)=596
27-L	1468	30	8	122	4	123	(122*8)+(123*4)=1468
27-M	1128	28	22	45	3	46	(45*22)+(46*3)=1128
27-Q	808	30	8	23	26	24	(23*8)+(24*26)=808
27-H	628	30	12	15	28	16	(15*12)+(16*28)=628
28-L	1531	30	3	117	10	118	(117*3)+(118*10)=1531
28-M	1193	28	3	45	23	46	(45*3)+(46*23)=1193
28-Q	871	30	4	24	31	25	(24*4)+(25*31)=871
28-H	661	30	11	15	31	16	(15*11)+(16*31)=661
29-L	1631	30	7	116	7	117	(116*7)+(117*7)=1631
29-M	1267	28	21	45	7	46	(45*21)+(46*7)=1267

Versión y nivel de CE	Número total de palabras clave de datos para esta versión y nivel CE	Palabras de código CE por bloque	Números de bloques en el grupo 1	Número de palabras de código de datos en cada un de los bloques del grupo 1	Números de bloques en el grupo 2	Número de palabras de códigos de datos en cada uno de los bloques del grupo 2	Palabras clave de datos totales
29-Q	911	30	1	23	37	24	(23*1)+(24*37)=911
29-H	701	30	19	15	26	16	(15*19)+(16*26)=701
30-L	1735	30	5	115	10	116	(115*5)+(116*10)=1735
30-M	1373	28	19	47	10	48	(47*19)+(48*10)=1373
30-Q	985	30	15	24	25	25	(24*15)+(25*25)=985
30-H	745	30	23	15	25	16	(15*23)+(16*25)=745
31-L	1843	30	13	115	3	116	(115*3)+(116*3)=1843
31-M	1455	28	2	46	29	47	(46*2)+(47*29)=1455
31-Q	1033	30	42	24	1	25	(24*42)+(25*1)=1033
31-H	793	30	23	15	28	16	(15*23)+(16*28)=793
32-L	1955	30	17	115			(115*17)=1955
32-M	1541	28	10	46	23	47	(46*10)+(47*23)=1541
32-Q	1115	30	10	24	35	25	(24*10)+(25*35)=1115
32-H	845	30	19	15	35	16	(15*19)+(16*35)=845
33-L	2071	30	17	115	1	116	(115*17)+(116*1)=2071
33-M	1631	28	14	46	21	47	(46*14)+(47*21)=1631
33-Q	1171	30	29	24	19	25	(24*29)+(25*19)=1171
33-H	901	30	11	15	46	16	(15*11)+(16*46)=901
34-L	2191	30	13	115	6	116	(115*13)+(116*6)=2191
34-M	1725	28	14	46	23	47	(46*14)+(47*23)=1725
34-Q	1231	30	44	24	7	25	(24*44)+(25*7)=1231
34-H	961	30	59	16	1	17	(16*59)+(17*1)=961
35-L	2306	30	12	121	7	122	(121*12)+(122*7)=2306
35-M	1812	28	12	47	26	48	(47*12)+(48*26)=1812
35-Q	1286	30	39	24	14	25	(24*39)+(25*14)=1286
35-H	986	30	22	15	41	16	(15*22)+(16*41)=986
36-L	2434	30	6	121	14	122	(121*6)+(122*14)=2434
36-M	1914	28	6	47	34	48	(47*6)+(48*34)=1914
36-Q	1354	30	46	24	10	25	(24*46)+(25*10)=1354
36-H	1054	30	2	15	64	16	(15*2)+(16*64)=1054
37-L	2566	30	17	122	4	123	(122*17)+(123*4)=2566
37-M	1992	28	29	46	14	47	(46*29)+(47*14)=1992
37-Q	1426	30	49	24	10	25	(24*49)+(25*10)=1426
37-H	1096	30	24	15	46	16	(15*24)+(16*46)=1096
38-L	2702	30	4	122	18	123	(122*4)+(123*18)=2702
38-M	2102	28	13	46	32	47	(46*13)+(47*32)=2102
38-Q	1502	30	48	24	14	25	(24*48)+(25*14)=1502
38-H	1142	30	42	15	32	16	(15*42)+(16*32)=1142
39-L	2812	30	20	117	4	118	(117*20)+(118*4)=2812
39-M	2216	28	40	47	7	48	(47*40)+(48*7)=2216
39-Q	1582	30	43	24	22	25	(24*43)+(25*22)=1582
39-H	1222	30	10	15	67	16	(15*10)+(16*67)=1222

Versión y nivel de CE	Número total de palabras clave de datos para esta versión y nivel CE	Palabras de código CE por bloque	Números de bloques en el grupo 1	Número de palabras de código de datos en cada un de los bloques del grupo 1	Números de bloques en el grupo 2	Número de palabras de códigos de datos en cada uno de los bloques del grupo 2	Palabras clave de datos totales
40-L	2956	30	19	118	6	119	(118*19)+(119*6)=2956
40-M	2334	28	18	47	31	48	(47*18)+(48*31)=2334
40-Q	1666	30	34	24	34	25	(24*34)+(25*34)=1666
40-H	1276	30	20	15	61	16	(15*20)+(16*61)=1276

10.4. Tabla Log y Antilog para el campo de Galois GF(256)

La siguiente tabla contiene los valores logarítmicos y antilogarítmicos que se utilizan en la aritmética GF(256), que se utiliza para generar los códigos de corrección de errores necesarios para los códigos QR.

Cuadro 11: Tabla Log y Antilog para el campo GF(256)

Log		Antilog	
Exponente de α	Entero	Entero	Exponente de α
0			
1	2	1	0
2	4	2	1
3	8	3	25
4	16	4	2
5	32	5	50
6	64	6	26
7	128	7	198
8	29	8	3
9	58	9	223
10	116	10	51
11	232	11	238
12	205	11	238
13	135	13	104
14	19	14	199
15	38	15	75
16	76	16	4
17	152	17	100
18	45	18	224
19	90	19	14
20	180	20	52
21	117	21	52
22	234	22	239
23	201	23	129
24	143	24	28
25	3	25	193
26	6	26	105
27	12	27	248
28	24	28	200
29	48	29	8

Log		Antilog	
Exponente de α	Entero	Entero	Exponente de α
30	96	30	76
31	192	31	113
32	157	32	5
33	39	33	138
34	78	34	101
35	156	35	47
36	37	36	225
37	74	37	36
38	148	38	15
39	53	39	33
40	106	40	53
41	212	41	147
42	181	42	142
43	119	43	218
44	238	44	240
45	193	45	18
46	159	46	130
47	35	47	69
48	70	48	29
49	140	49	181
50	5	50	194
51	10	51	125
52	20	52	106
53	40	53	39
54	80	54	246
55	160	55	185
56	93	56	201
57	186	57	154
58	105	58	9
59	210	59	120
60	185	60	77
61	111	61	228
62	222	62	114
63	161	63	166
64	95	64	6
65	190	65	191
66	97	66	139
67	194	67	98
68	153	68	102
69	47	69	221
70	94	70	48
71	188	71	253
72	101	72	226
73	202	73	152
74	137	74	37
75	15	75	179
76	30	76	16
77	60	77	145
78	120	78	34
79	240	79	136
80	253	80	54
81	231	81	208

Log		Antilog	
Exponente de α	Entero	Entero	Exponente de α
82	211	82	148
83	187	83	206
84	107	84	143
85	214	85	150
86	177	86	219
87	127	87	189
88	254	88	241
89	225	89	210
90	223	90	19
91	163	91	92
92	91	92	131
93	182	93	56
94	113	94	70
95	226	95	64
96	217	96	30
97	175	97	66
98	67	98	182
99	134	99	163
100	17	100	195
101	34	101	72
102	68	102	126
103	136	103	110
104	13	104	107
105	26	105	58
106	52	106	40
107	104	107	84
108	208	108	250
109	189	109	133
110	103	110	186
111	206	111	61
112	129	112	202
113	31	113	94
114	62	114	155
115	124	115	159
116	248	116	10
117	237	117	21
118	199	118	121
119	147	119	43
120	59	120	78
121	118	121	212
122	236	122	229
123	197	123	172
124	151	124	115
125	51	125	243
126	102	126	167
127	204	127	87
128	133	128	7
129	23	129	112
130	46	130	192
131	92	131	247
132	184	132	140
133	109	133	128

Log		Antilog	
Exponente de α	Entero	Entero	Exponente de α
134	218	134	99
135	169	135	13
136	79	136	103
137	158	137	74
138	33	138	222
139	66	139	237
140	132	140	49
141	21	141	197
142	42	142	254
143	84	143	24
144	168	144	227
145	77	145	165
146	154	146	153
147	41	147	119
148	82	148	38
149	164	149	184
150	85	150	180
151	170	151	124
152	73	152	17
153	146	153	68
154	57	154	146
155	114	155	217
156	228	156	35
157	213	157	32
158	183	158	137
159	115	159	46
160	230	160	55
161	209	161	63
162	191	162	209
163	99	163	91
164	198	164	149
165	145	165	188
166	63	166	207
167	126	167	205
168	252	168	144
169	229	169	135
170	215	170	151
171	179	171	178
172	123	172	220
173	246	173	252
174	241	174	190
175	255	175	97
176	227	176	242
177	219	177	86
178	171	178	211
179	75	179	171
180	150	180	20
181	49	181	42
182	98	182	93
183	196	183	158
184	149	184	132
185	55	185	60

Log		Antilog	
Exponente de α	Entero	Entero	Exponente de α
186	110	186	57
187	220	187	83
188	165	188	71
189	87	189	109
190	174	190	65
191	65	191	162
192	130	192	31
193	25	193	45
194	50	194	67
195	100	195	216
196	200	196	183
197	141	197	123
198	7	198	164
199	14	199	118
200	28	200	196
201	56	201	23
202	112	202	73
203	224	203	236
204	221	204	127
205	167	205	12
206	83	206	111
207	166	207	246
208	81	208	108
209	162	209	161
210	89	210	59
211	178	211	82
212	121	212	41
213	242	213	157
214	249	214	85
215	239	215	170
216	195	216	251
217	155	217	96
218	43	218	134
219	86	219	177
220	172	220	187
221	69	221	204
222	138	222	62
223	9	223	90
224	18	224	203
225	36	225	89
226	72	226	95
227	144	227	176
228	61	228	156
229	122	229	169
230	244	230	160
231	245	231	81
232	247	232	11
233	243	233	245
234	251	234	22
235	235	235	235
236	203	236	122
237	139	237	117

Log		Antilog	
Exponente de α	Entero	Entero	Exponente de α
238	11	238	44
239	22	239	215
240	44	240	79
241	88	241	174
242	176	242	213
243	125	243	233
244	250	244	230
245	233	245	231
246	207	246	173
247	131	247	232
248	27	248	116
249	54	249	214
250	108	250	244
251	216	251	234
252	173	252	168
253	71	253	80
254	142	254	88
255	1	255	175

10.5. Lista de versiones y bits restantes necesarios

La siguiente tabla enumera las 40 versiones QR y la cantidad de bits restantes que se deben agregar al final de la cadena de mensaje final. Tenga en cuenta que los bits restantes especificados para cada versión son necesarios sin importar qué nivel de corrección de errores esté en uso. También tenga en cuenta que algunas versiones, como las versiones 7 a 13, no requieren que se agreguen bits restantes.

Versión QR	Bits restantes requeridos
1	0
2	7
3	7
4	7
5	7
6	7
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	3
15	3
16	3
17	3
18	3
19	3
20	3
21	4
22	4
23	4
24	4
25	4
26	4

Versión QR	Bits restantes requeridos
27	4
28	3
29	3
30	3
31	3
32	3
33	3
34	3
35	0
36	0
37	0
38	0
39	0
40	0

10.6. Tabla de ubicaciones de patrones de alineación

Los patrones de alineación se explican en detalle en la ubicación del módulo en la página de matriz. Los números en el lado derecho de esta tabla deben usarse como coordenadas de fila y columna. Por ejemplo, la Versión 2 tiene los números 6 y 18. Esto significa que los patrones de alineación deben colocarse en (6, 6), (6, 18), (18, 6) y (18, 18). Sin embargo, no coloque patrones de alineación encima de patrones de búsqueda o separadores.

Versión QR	Fila y columna del módulo central
1	(0)
2	(6,18)
3	(6,22)
4	(6,26)
5	(6,30)
6	(6,34)
7	(6,22,38)
8	(6,24,42)
9	(6,26,46)
10	(6,28,50)
11	(6,30,54)
12	(6,32,58)
13	(6,34,62)
14	(6,26,46,66)
15	(6,26,48,70)
16	(6,26,50,74)
17	(6,30,54,78)
18	(6,30,56,82)
19	(6,30,58,86)
20	(6,34,62,90)
21	(6,28,50,72,94)
22	(6,26,50,74,98)
23	(6,30,54,78,102)
24	(6,28,54,80,106)
25	(6,32,58,84,110)
26	(6,30,58,86,114)
27	(6,34,62,90,118)
28	(6,26,50,74,98,122)

Versión QR	Fila y columna del módulo central
29	(6,30,54,78,102,126
30	(6,26,52,78,104,130)
31	(6,30,56,82,108,134)
32	(6,34,60,86,112,138)
33	(6,30,58,86,114,142)
34	(6,34,62,90,118,146)
35	(6,30,54,78,102,126,150)
36	(6,24,50,76,102,128,154)
37	(6,28,54,80,106,132,158)
38	(6,32,58,84,110,136,162)
39	(6,26,54,82,110,138,166)
40	(6,30,58,86,114,142,170)

10.7. Patrones de máscara QR explicados

Al codificar un código QR, hay ocho patrones de máscara que puede usar para cambiar la matriz de salida. Cada patrón de máscara cambia los bits según sus coordenadas en la matriz QR. El propósito de un patrón de máscara es hacer que el código QR sea más fácil de leer para un escáner QR.

Los patrones de máscara: Cada patrón de máscara utiliza una fórmula para determinar si se cambia o no el color del bit actual. Pones las coordenadas del bit actual en la fórmula, y si el resultado es 0, usas el bit opuesto en esa coordenada. Por ejemplo, si el bit para la coordenada (0,3) es 1 y la fórmula es igual a 0 para esa coordenada, entonces pones un 0 en (0,3) en lugar de un 1.

Aquí está la lista de fórmulas de patrón de máscara. Tenga en cuenta que algunas versiones del código QR estándar han tenido errores en la sección sobre patrones de máscara. La siguiente información ha sido corregida. Nota: mod es la abreviatura de Modulo.

Cuadro 14: Patrones de mascaras

Números de mascaras	Si la fórmula a continuación es verdadera para una determinada coordenada de fila/columna, cambie el bit en esa coordenada
0	(fila + columna) mód 2 == 0
1	(fila) mód 2 == 0
2	(columna) mód 3 ==0
3	(fila + columna) mód 3 == 0
4	(piso(fila/2) + piso(columna/3)) mód 2 == 0
5	((fila*columna) mód 2) + ((fila*columna) mód 3) == 0
6	(((fila*columna) mód 2) + ((fila*columna) mód 3)) mód 2 == 0
7	(((fila + columna) mód 2) + ((fila*columna) mód 3)) mód 2 == 0

10.8. Información de formato y versión

Un código QR utiliza codificación de corrección de errores y patrones de máscara. El tamaño del código QR está representado por un número, llamado número de versión. Para garantizar que un escáner de códigos QR decodifique con precisión lo que escanea, la especificación del código QR requiere que cada código incluya una cadena de información de formato, que le dice al escáner de códigos QR qué nivel de corrección de errores y patrón de máscara está usando el código QR. Además, para la versión 7 y posteriores, la especificación del código QR requiere que cada código incluya una cadena de información de la versión, que le dice al escáner de códigos QR qué versión es el código. Este anexo enumera todas las cadenas de formato y versión posibles.

Sobre las cadenas de información de formato: Los códigos QR usan codificación de corrección de errores. Esta es una forma de generar datos redundantes que los escáneres de códigos QR pueden usar para detectar y corregir errores en el código escaneado. Los códigos QR también usan patrones de máscara. Un patrón de máscara es un algoritmo para cambiar el color (oscuro a claro o claro a oscuro) de un determinado patrón de píxeles en el código para facilitar la lectura precisa de los escáneres. Los códigos QR deben incluir una cadena de formato que contenga la información sobre qué nivel de codificación de corrección de errores y qué patrón de máscara están en uso en el código. Esta página enumera las 32 cadenas de formato posibles. Para obtener una explicación detallada de cómo se generan estas cadenas.

Cuadro 15: Lista de todas las cadenas de información de formato

Nivel de CE	Patrón de máscara	Tipo de información en Bits
L	0	111011111000100
L	1	111001011110011
L	2	111110110101010
L	3	111100010011101
L	4	110011000101111
L	5	110001100011000
L	6	110110001000001
L	7	110100101110110
M	0	101010000010010
M	1	101000100100101
M	2	101111001111100
M	3	101101101001011
M	4	100010111111001
M	5	100000011001110
M	6	100111110010111
M	7	100101010100000
Q	0	011010101011111
Q	1	011000001101000
Q	2	011111100110001
Q	3	011101000000110
Q	4	010010010110100
Q	5	010000110000011
Q	6	010111011011010
Q	7	010101111101101
H	0	001011010001001
H	1	001001110111110
H	2	001110011100111
H	3	001100111010000
H	4	000011101100010
H	5	000001001010101
H	6	000110100001100
H	7	000100000111011

Sobre las cadenas de información de versión: El tamaño de un código QR está representado por un número llamado número de versión. Los códigos de la versión 7 y mayores deben incluir dos bloques rectangulares de 6x3 que contienen la cadena de información de la versión. Para obtener detalles sobre cómo se calculan estas cadenas de información de versión y dónde deben colocarse en el código QR, consulte la página Información de formato y versión.

Cuadro 16: Lista de todas las cadenas de información de versión

Nivel	Cadena de información de versión
7	000111110010010100
8	001000010110111100
9	001001101010011001
10	001010010011010011
11	001011101111110110
12	001100011101100010
13	001101100001000111
14	001110011000001101
15	001111100100101000
16	010000101101111000
17	010001010001011101
18	010010101000010111
19	010011010100110010
20	010100100110100110
21	010101011010000011
22	010110100011001001
23	010111011111101100
24	011000111011000100
25	011001000111100001
26	011010111110101011
27	011011000010001110
28	011100110000011010
29	011101001100111111
30	011110110101110101
31	011111001001010000
32	100000100111010101
33	100001011011110000
34	100010100010111010
35	100011011110011111
36	100100101100001011
37	100101010000101110
38	100110101001100100
39	100111010101000001
40	101000110001101001