ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ГОСТ Р ИСО/МЭК 24778— 2010**

Информационные технологии

# ТЕХНОЛОГИИ АВТОМАТИЧЕСКОЙ ИДЕНТИФИКАЦИИ И СБОРА ДАННЫХ

## Спецификация символики штрихового кода Aztec Code

ISO/IEC 24778:2008
Information technology — Automatic identification and data capture techniques — Aztec Code bar code symbology specification
(IDT)

Издание официальное

БЗ 12—2009/933

Москва
Стандартинформ
2010

# Предисловие

Цели и принципы стандартизации в Российской Федерации установлены Федеральным законом от 27 декабря 2002 г. № 184-ФЗ «О техническом регулировании», а правила применения национальных стандартов Российской Федерации — ГОСТ Р 1.0 — 2004 «Стандартизация в Российской Федерации. Основные положения»

**Сведения о стандарте**

1 ПОДГОТОВЛЕН Обществом с ограниченной ответственностью «РИТ СЕРВИС» совместно с Ассоциацией автоматической идентификации «ЮНИСКАН/ГС1 РУС» на основе собственного аутентичного перевода на русский язык стандарта, указанного в пункте 4, выполненного ООО «РИТ СЕРВИС»

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 355 «Технологии автоматической идентификации и сбора данных и биометрия»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 14 мая 2010 г. № 79-ст

4 Настоящий стандарт идентичен международному стандарту ИСО/МЭК 24778:2008 «Информационные технологии. Технологии автоматической идентификации и сбора данных. Спецификация символики штрихового кода Aztec Code» (ISO/IEC 24778:2008 «Information technology — Automatic identification and data capture techniques — Aztec Code bar code symbology specification»)

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных стандартов соответствующие им национальные стандарты Российской Федерации, сведения о которых приведены в дополнительном приложении ДА

5 ВВЕДЕН ВПЕРВЫЕ

6 Следует обратить внимание на то, что некоторые элементы настоящего стандарта могут быть объектом патентного права. ИСО и МЭК не несут ответственность за установление подлинности каких-либо или всех таких патентных прав

*Информация об изменениях к настоящему стандарту публикуются в ежегодно издаваемом информационном указателе «Национальные стандарты», а текст изменений и поправок — в ежемесячно издаваемых информационных указателях «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ежемесячно издаваемом информационном указателе «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет*

## Содержание

# Введение

Aztec Code является двумерной матричной символикой, символы которой имеют квадратную форму и состоят из квадратных модулей, расположенных на сетке с квадратными ячейками, в центре которой располагается шаблон поиска «мишень», состоящий из концентрических квадратов. Символы Aztec Code позволяют кодировать как малые, так и большие объемы данных с уровнем исправления ошибок, выбираемым пользователем.

Настоящий стандарт рекомендован для применения изготовителями оборудования для работы со штриховыми кодами и пользователями данной технологии, обеспечивая совместимость программных и аппаратных средств, предназначенных для формирования и считывания символов Aztec Code.

Сноски в тексте стандарта, выделенные курсивом, приведены для пояснения текста стандарта.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

# Информационные технологии
# ТЕХНОЛОГИИ АВТОМАТИЧЕСКОЙ ИДЕНТИФИКАЦИИ И СБОРА ДАННЫХ
# Спецификация символики штрихового кода Aztec Code

Information technologies. Automatic identification and data capture techniques. Aztec Code bar code symbology specification

**Дата введения — 2011—07—01**

## 1 Область применения

Настоящий стандарт устанавливает требования к символике Aztec Code[1)], включая метод кодирования знаков данных, правила кодирования и исправления ошибок, структуру графических символов, требования к размерам символов и качеству их печати, рекомендуемый алгоритм декодирования, а также прикладные параметры для применения, задаваемые пользователем.

## 2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты и другие нормативные документы, которые необходимо учитывать при использовании настоящего стандарта. В случае ссылок на документы, у которых указана дата утверждения, необходимо пользоваться только указанной редакцией. В случае, когда дата утверждения не приведена, следует пользоваться последней редакцией ссылочных документов, включая любые поправки и изменения к ним:

ИСО/МЭК 646:1991 Информационные технологии. Набор 7-битовых кодированных знаков ИСО для обмена информацией (ISO/IEC 646:1991 Information technology — ISO 7-bit coded character set for information interchange)

ИСО/МЭК 15415:2004 Информационные технологии. Технологии автоматической идентификации и сбора данных. Спецификация испытаний качества печати символов штрихового кода. Двумерные символы (ISO/IEC 15415:2004 Information technology — Automatic identification and data capture techniques — Bar code print quality test specification — Two-dimensional symbols)

ИСО/МЭК 15424 Информационные технологии. Технологии автоматической идентификации и сбора данных. Идентификаторы носителей данных (включая идентификаторы символик) (ISO/IEC 15424 Information technology — Automatic identification and data capture techniques — Data Carrier Identifiers (including Symbology Identifiers))

ИСО/МЭК 19762 (все части) Информационные технологии. Технологии автоматической идентификации и сбора данных. Гармонизированный словарь. (ISO/IEC 19762 (all parts) Information technology — Automatic identification and data capture (AIDC) techniques — Harmonized vocabulary)

AIM Inc. Международная техническая спецификация: Интерпретации в расширенном канале (AIM Inc. International Technical Specification: Extended Channel Interpretations):

- часть 1. Схемы идентификации и протокол (Part 1: Identification Schemes and Protocol);
- часть 2. Процедура регистрации наборов кодированных знаков и других форматов данных. Регистр наборов знаков (Part 2: Registration Procedure for Coded Character Sets and Other Data Formats. Character Set Register)

---

[1)] *Рекомендуемое наименование символики на русском языке — «Ацтек код».*

**Издание официальное**

# 3 Термины, определения, условные обозначения и функции

## 3.1 Термины и определения

В настоящем стандарте применены термины и определения, установленные в ИСО/МЭК 19762 (все части), а также следующие:

3.1.1 **шаблон поиска «мишень»** (bullseye): Группа концентрических квадратов, используемая в качестве шаблона поиска в Aztec Code.

3.1.2 **контрольное слово** (checkword): Кодовое слово, включаемое в символ Aztec Code с целью исправления и/или обнаружения ошибок.

3.1.3 **кодовое слово данных** (dataword): Кодовое слово, являющееся частью данных сообщения, закодированного в символе.

3.1.4 **блок типа домино** (domino): Двухмодульная подструктура знака символа в символе Aztec Code, являющаяся элементарной структурной единицей при кодировании символа в графическом виде.

3.1.5 **служебное сообщение** (Mode Message): Короткое служебное сообщение фиксированной длины, кодируемое в символе Aztec Code, содержащее информацию о размере символа и длине сообщения с данными, а также кодовые слова исправления ошибок.

## 3.2 Условные обозначения и функции

### 3.2.1 Математические обозначения

В настоящем стандарте использованы следующие математические обозначения:

B — число битов в каждом кодовом слове;

$C_b$ — емкость символа в битах;

$C_w$ — емкость символа в кодовых словах;

D — число кодовых слов данных (сообщений) в символе;

K — число кодовых слов исправления ошибок в символе, равное $C_w$ — D;

L — число слоев данных в символе (от 1 до 32), определяющее его размер;

m — знак-модификатор идентификатора символики;

X — размер X или номинальный шаг квадратной сетки;

x — обобщенная переменная, используемая для представления полиномов исправления ошибок;

($x$, $y$) — декартовы координаты на сетке модулей.

### 3.2.2 Математические функции и операции

В настоящем стандарте применены следующие математические функции и операторы:

abs() — функция абсолютной величины (модуль);

div — оператор целочисленного деления;

max(a, b) — большее из чисел a и b;

mod — остаток после целочисленного деления.

# 4 Параметры символики

## 4.1 Основные параметры

Aztec Code представляет собой двумерную матричную символику, имеющую следующие основные параметры:

а) кодируемый набор знаков:

1) поддерживает кодирование любых 8-битовых значений. По умолчанию используют следующее представление:

а) знаки набора ASCII (версии КОИ-7) по ИСО/МЭК 646 согласно национальной версии США[1)] (далее — знаки ASCII (КОИ-7)) для знаков с десятичными значениями от 0 до 127 знаков кодового набора.

Примечание — Указанная версия ASCII (КОИ-7) состоит из набора знаков G0 по ИСО/МЭК 646 и набора знаков C0 по ИСО/МЭК 6429, в котором знаки с десятичными значениями от 28 до 31 соответствуют знакам FS, GS, RS и US соответственно;

---

[1)] *Набор знаков по ANSI INCITS 4-1986 (R2007) Information Systems — Coded Character Sets — 7-Bit American National Standard Code for Information Interchange (7-Bit ASCII) (Информационные системы — Кодированные наборы знаков — 7-битовый американский национальный стандартный код для обмена информацией (7- битовый ASCII).*

b) знаки расширенного набора ASCII (версия КОИ-8) по ИСО/МЭК 8859-1 (далее — знаки расширенного набора ASCII (КОИ-8)) с десятичными значениями от 128 до 255.

Данное представление соответствует интерпретации в расширенном канале (ECI) — ECI 000003;

2) поддерживает кодирование двух знаков, не являющихся знаками данных: знак FNC1 для совместимости с некоторыми действующими приложениями и знак управляющей последовательности ECI для типового кодирования сведений об интерпретации сообщения;

b) представление данных: темный модуль соответствует двоичной единице, светлый — двоичному нулю;

c) размер символа:

1) символ Aztec Code наименьшего размера представляет собой квадрат размером 15 × 15 модулей, а наибольшего — квадрат размером 151 × 151 модулей;

2) наличие свободной зоны за границами символа не обязательно;

d) объем данных (для рекомендуемого уровня исправления ошибок):

1) символ Aztec Code наименьшего размера обеспечивает кодирование до 13 цифровых или 12 алфавитных знаков или 6 байтов данных;

2) символ Aztec Code наибольшего размера обеспечивает кодирование до 3832 цифровых или 3067 алфавитных символов или 1914 байтов данных;

e) задаваемый уровень исправления ошибок:

1) устанавливается пользователем в размере от 5 % до 95 % объема данных, но не менее трех кодовых слов;

2) рекомендуемый уровень — 23 % емкости символа с добавлением трех кодовых слов;

f) тип кодирования: матричная символика;

g) независимость от ориентации: присутствует.

### 4.2 Дополнительные свойства

Символика Aztec Code обладает следующими неотъемлемыми и необязательными (задаваемыми по выбору) свойствами:

a) цветовая инверсия (неотъемлемое свойство). В настоящем стандарте при представлении и описании символов Aztec Code центр шаблона поиска всегда представлен модулем темного цвета, при этом двоичные единицы кодируют темными модулями; символы с заменой цветов модулей на противоположный легко различаются и декодируются типовым устройством считывания;

b) зеркальное отображение (неотъемлемое свойство). Изображения, содержащие символы Aztec Code в зеркальном отображении, полученные с помощью отраженного светового потока или при обратном направлении сканирования, или считанные с обратной стороны прозрачной подложки, автоматически распознаются и декодируются типовым устройством считывания;

c) интерпретация в расширенном канале (ECI) (необязательное свойство). Механизм интерпретации в расширенном канале (ECI) обеспечивает представление знаков различных наборов (например, арабского, кириллицы, греческого, иврита) и других интерпретаций данных или учет специфических требований для конкретного применения;

d) структурированное соединение (необязательное свойство). Обеспечивает возможность логического последовательного представления файлов данных с использованием до 26 символов Aztec Code. Исходные данные правильно восстанавливаются при любой последовательности сканирования символов;

e) наличие символов для инициализации устройства считывания (необязательное свойство). Применяется специальный формат символов Aztec Code как опция в меню штрихового кода для инициализации устройства считывания. Данные в сообщении, закодированные в таких специальных символах, не подлежат передаче;

f) наличие структуры символов Aztec Runes[1)] (необязательное свойство). Применяется набор из 256 небольших символов (графических объектов), пригодных для машинного считывания, совместимых с символами Aztec Code и предназначенных для специального применения (приложение А).

## 5 Описание символов

Символы Aztec Code имеют номинально квадратную форму и состоят из квадратных модулей, расположенных на условной сетке с ячейками квадратной формы, в центре которой располагается шаблон поиска «мишень». На рисунке 1 изображены два типовых символа Aztec Code. Слева расположен небольшой однослойный символ, в котором закодировано 12 цифр с уровнем исправления

---

[1)] *Наименование символов на русском языке — «Руны Ацтек».*

ошибок 47 %, а справа — более крупный шестислойный символ, в котором закодировано 168 цифр с уровнем исправления ошибок 30 %.

Рисунок 1 — Типовые символы Aztec Code

Символы, приведенные на рисунке 1, иллюстрируют примеры двух основных форматов символов Aztec Code. Слева расположен компактный символ Aztec Code, который визуально отличают по шаблону поиска «мишень», состоящему из двух концентрических квадратов. Компактные символы наилучшим образом подходят для эффективного кодирования коротких сообщений. Справа расположен полноразмерный символ Aztec Code, шаблон поиска «мишень» которого состоит из трех концентрических квадратов. Полноразмерные символы имеют размеры больше, чем компактные, и в них можно кодировать сообщения большего объема. Поскольку кодирующие устройства могут автоматически выбирать, а декодирующие устройства автоматически различать эти два формата, то обеспечивается охват прикладных применений символов Aztec Code.

### 5.1 Структура символа

Основная структура компактного символа Aztec Code приведена на рисунке 2, а основная структура полноразмерного символа Aztec Code — на рисунке 3. В любом символе Aztec Code в его центре расположено ядро символа, со всех четырех сторон окруженное полями данных.

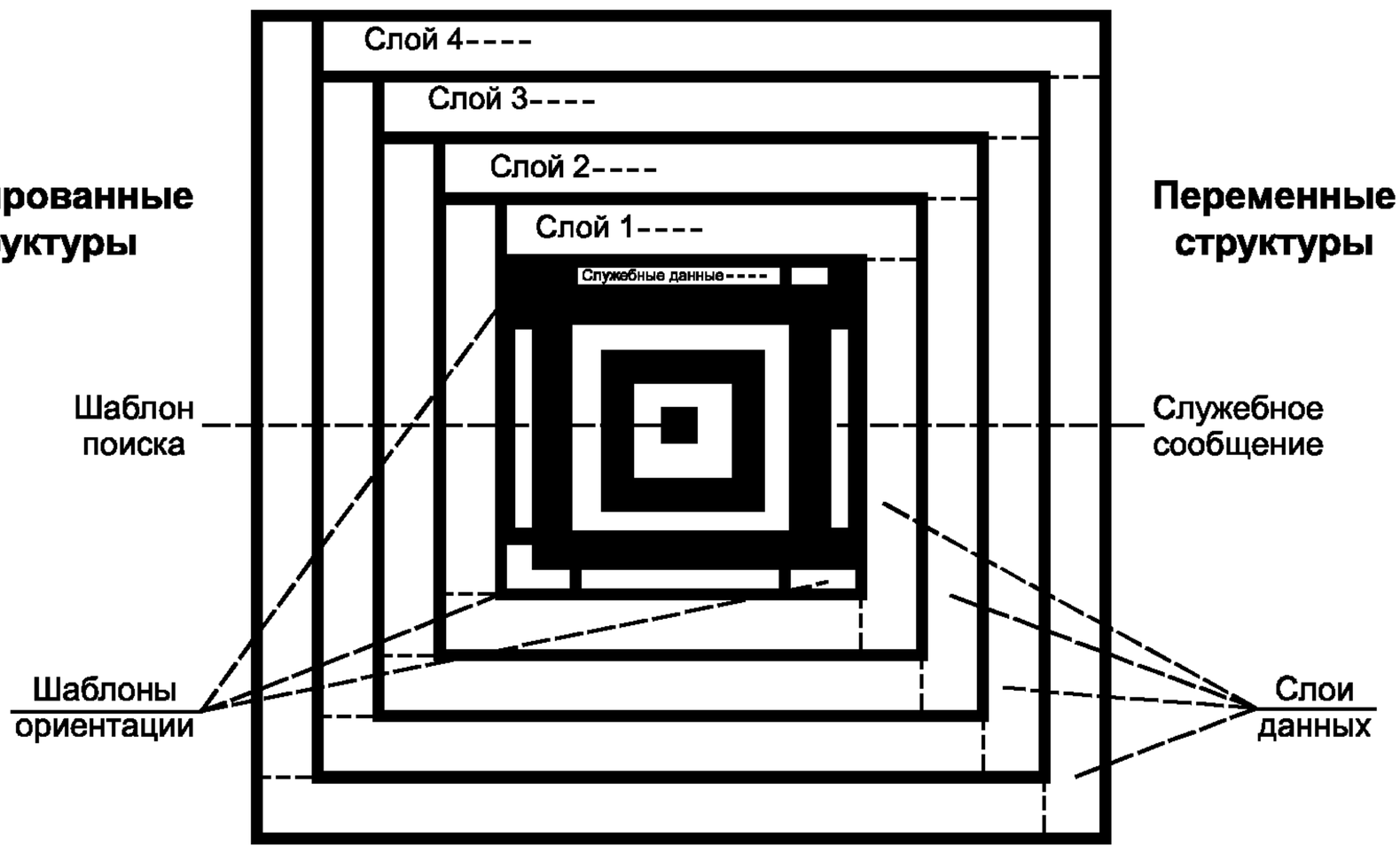


Рисунок 2 — Структура компактного символа Aztec Code

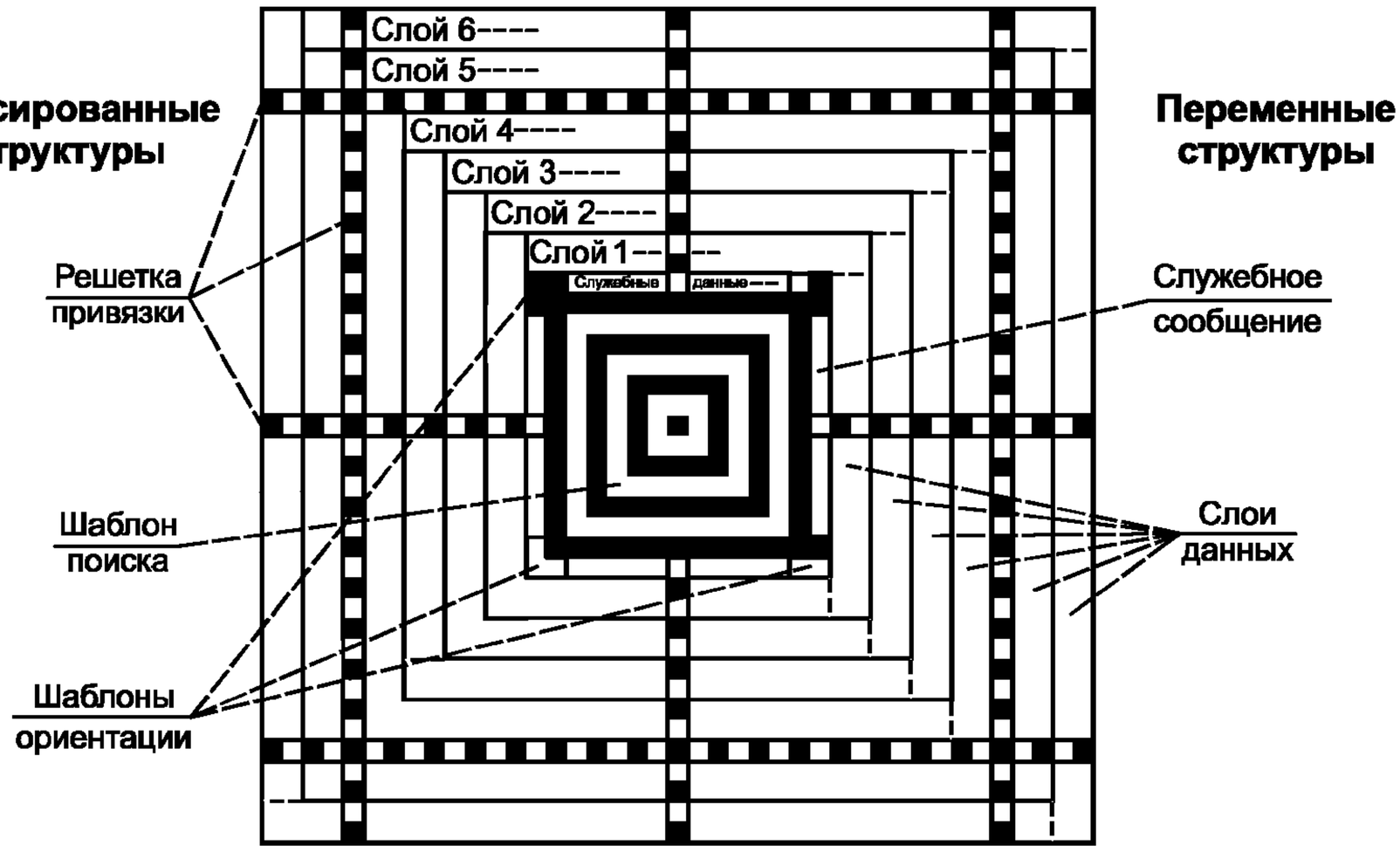


Рисунок 3 — Структура полноразмерного символа Aztec Code

**5.1.1 Ядро символа**

Ядро символа имеет квадратную форму, располагается точно в центре символа Aztec Code и состоит из шаблона поиска, шаблонов ориентации и служебного сообщения. В компактных символах ядро занимает область размером 11×11 модулей, а в полноразмерных символах — 15×15 модулей. Декодирование окружающих полей данных возможно только после успешного обнаружения и декодирования ядра символа.

5.1.1.1 Шаблон поиска

Шаблон поиска в Aztec Code представляет собой группу концентрических квадратов. В центре находится одиночный темный модуль, заключенный в квадрат из светлых модулей, который окружен следующим квадратом из темных модулей и т.д. вплоть до второго темного квадрата размером 9×9 модулей в компактных символах или третьего темного квадрата размером 13×13 модулей в полноразмерных символах.

5.1.1.2 Шаблоны ориентации

По углам шаблона поиска расположены четыре трехмодульных шаблона ориентации, имеющих форму шеврона. Левый верхний шаблон ориентации состоит из темных модулей, а левый нижний — из светлых модулей. Правый верхний шаблон содержит два темных модуля[1)], а правый нижний — один темный модуль[2)] (рисунки 2 и 3).

5.1.1.3 Служебное сообщение

Слой данных, состоящий из модулей, представляющих биты служебной информации, прилегающих к шаблону поиска, за исключением модулей шаблонов ориентации (в полноразмерных символах также за исключением центрального модуля на каждой стороне, являющегося частью решетки привязки), содержит служебное сообщение и кодовые слова исправления ошибок. Указанные модули размещают в данном слое в направлении по часовой стрелке, начиная с его левого верхнего угла. В служебном сообщении в явном виде кодируют как число слоев данных символа (следовательно, и его размер), так и общее число кодовых слов данных в этих слоях. Остальные кодовые слова являются контрольными словами исправления ошибок в сообщении. Подробное описание структуры и процедуры кодирования служебного сообщения приведено в 7.2.

**5.1.2 Поля данных**

Поля данных расположены симметрично вокруг ядра символа и состоят из одного или нескольких слоев данных. В полноразмерных символах по полям данных проходит решетка привязки.

---

[1)] *Темные модули расположены по вертикальной стороне шаблона ориентации.*

[2)] *Темный модуль расположен в правой верхней части шаблона ориентации.*

5.1.2.1 Решетка привязки

Каждая линия решетки привязки (рисунок 3) состоит из последовательности темных и светлых модулей, начинающихся от шаблона поиска и заканчивающихся на границах полей данных[1)]. Линии решетки привязки проходят по осям симметрии символа, а также по каждым 16 строкам и столбцам условной сетки. Периодическая структура линий решетки привязки обеспечивает создание контрольных точек, необходимых для более точного определения месторасположения полей данных в полноразмерных символах Aztec Code. В компактных символах Aztec Code, имеющих ограниченный размер, решетка привязки отсутствует.

5.1.2.2 Слои данных

Кодовые слова данных сообщения вместе с соответствующими кодовыми словами исправления ошибок располагают по слоям данных, каждый из которых имеет толщину размером два модуля и расположен вокруг ядра символа, при этом данные в каждом слое записывают, начиная с левого верхнего угла каждого слоя в направлении по часовой стрелке с последующим переходом на следующий внешний (по отношению к предыдущему) слой данных (по «спирали»). В полноразмерных символах Aztec Code при записи данных пропускают позиции модулей, занятых решеткой привязки. Компактные символы Aztec Code содержат от одного до четырех слоев данных; полноразмерные символы Aztec Code могут содержать от одного до 32 слоев. Подробная информация о кодировании данных сообщения, формировании кодовых слов, кодировании контрольных слов исправления ошибок и итоговом размещении кодовых слов по слоям данных приведены в 7.3.

**5.2 Структура знаков символа и последовательность их расположения**

Для повышения эффективности исправления ошибок с применением кодов Рида-Соломона кодовые слова и знаки символа имеют переменные размеры от 6 до 12 битов в зависимости от общего размера символа в соответствии с рисунком 4.

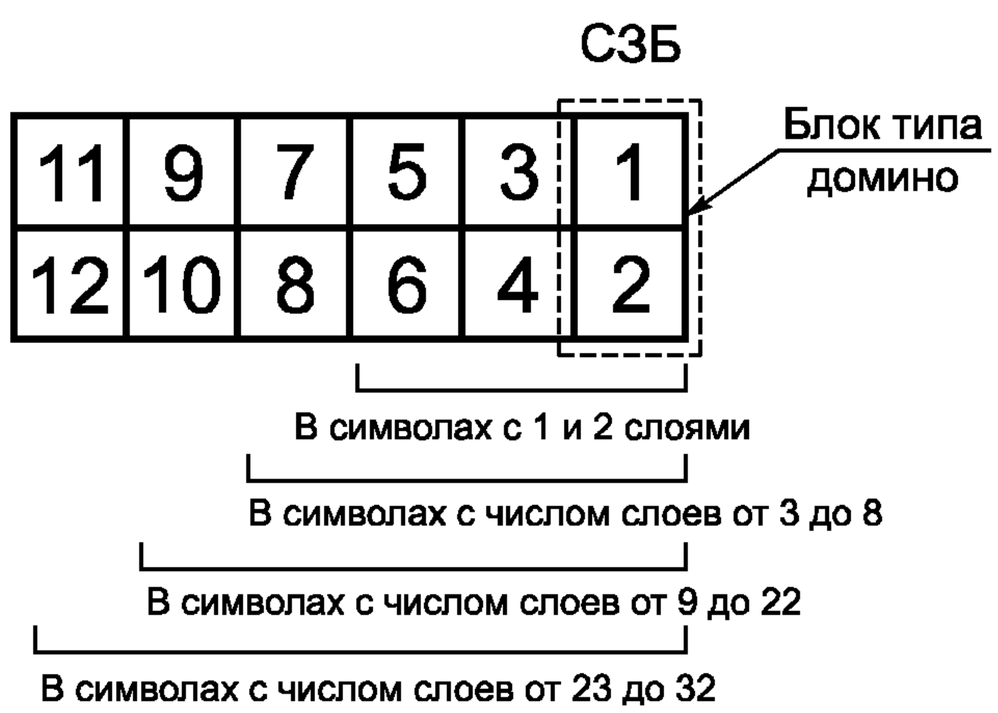


Рисунок 4 — Типовая структура знака в символе[2)]

В результате огибания ядра символа по спирали, поворотов на углах и из-за пропуска элементов решетки привязки фактические формы знаков символа заметно отличаются от общей типовой формы знака символа, характерной для конкретного символа. Если рассматривать каждый знак символа как последовательность блоков типа домино, состоящих из двух модулей в высоту и одного модуля в ширину, то последовательность знаков символа превращается в последовательность блоков типа домино. Размещение блоков типа домино в полях данных систематизировано, что упрощает их размещение в процессе кодирования и идентификацию в процессе декодирования. Расположение последовательности блоков типа домино с поворотами на углах символа и при переходах между слоями данных приведено на рисунке 5.

---

1) *Имеется в виду внешняя граница внешнего поля данных.*

2) *Аббревиатура СЗБ, приведенная на рисунке, означает «старший значащий бит».*

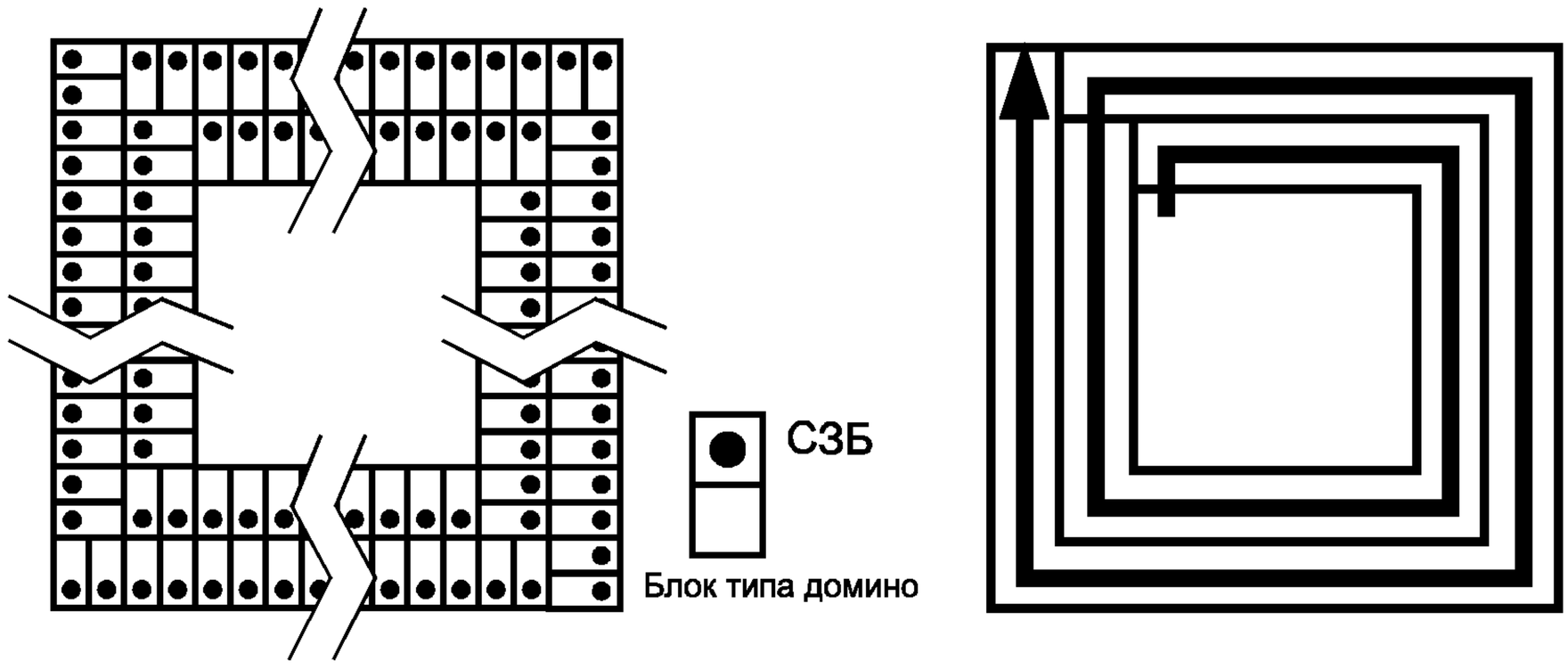


Рисунок 5 — Расположение и порядок следования блоков типа домино

Кодовые слова, расположенные по спирали наружу в направлении от ядра символа, формируют в следующем порядке (обратном естественному) — первое кодовое слово, записанное на первой позиции первого слоя, является последним контрольным словом в коде Рида-Соломона, второе кодовое слово является предпоследним контрольным словом в коде Рида-Соломона и т.д. За контрольными словами следуют кодовые слова данных сообщения, вплоть до последнего кодового слова в последнем (внешнем) слое данных, являющегося первым кодовым словом закодированных данных сообщения. Данное расположение кодовых слов в символе Aztec Code повышает эффективность исправления ошибок, поскольку кодовые слова данных, в которых выявляется ошибка «стирание» (кодовые слова данных, состоящие из одних нулей или единиц, которые считают недействительными), расположены вблизи от периметра символа, где вероятность появления ошибки стирания наиболее высока.

### 5.3 Размер и информационная емкость символа

В таблице 1 приведены основные характеристики и информационная емкость (объем информации, которую можно записать в символах Aztec Code) в зависимости от размеров символа.

Приведенные значения информационной емкости символов Aztec Code, зависят от рекомендуемых задаваемых уровней исправления ошибок. Указанные в таблице 1 значения информационной емкости устанавливают приблизительные максимальные значения, при этом эффективность кодирования данных зависит от их содержимого и возможности пользователя задавать уровень исправления ошибок.

Таблица 1 — Размеры и информационная емкость символов Aztec Code

| Число слоев данных | Размер символа (размер X) | Кодовые слова (число кодовых слов × размер каждого слова), биты | Информационная емкость символа, биты | Информационная емкость символа для различных типов данных | | |
|---|---|---|---|---|---|---|
| | | | | Цифры | Текст (знаки) | Байты |
| 1* | 15 × 15 | 17 × 6 | 102 | 13 | 12 | 6 |
| 1 | 19 × 19 | 21 × 6 | 126 | 18 | 15 | 8 |
| 2* | 19 × 19 | 40 × 6 | 240 | 40 | 33 | 19 |
| 2 | 23 × 23 | 48 × 6 | 288 | 49 | 40 | 24 |
| 3* | 23 × 23 | 51 × 8 | 408 | 70 | 57 | 33 |
| 3 | 27 × 27 | 60 × 8 | 480 | 84 | 68 | 40 |
| 4* | 27 × 27 | 76 × 8 | 608 | 110 | 89 | 53 |
| 4 | 31 × 31 | 88 × 8 | 704 | 128 | 104 | 62 |
| 5 | 37 × 37 | 120 × 8 | 960 | 178 | 144 | 87 |

*Окончание таблицы 1*

| Число слоев данных | Размер символа (размер X) | Кодовые слова (число кодовых слов × размер каждого слова), биты | Информационная емкость символа, биты | Информационная емкость символа для различных типов данных | | |
|---|---|---|---|---|---|---|
| | | | | Цифры | Текст (знаки) | Байты |
| 6 | 41 × 41 | 156 × 8 | 1248 | 232 | 187 | 114 |
| 7 | 45 × 45 | 196 × 8 | 1568 | 294 | 236 | 145 |
| 8 | 49 × 49 | 240 × 8 | 1920 | 362 | 291 | 179 |
| 9 | 53 × 53 | 230 × 10 | 2300 | 433 | 348 | 214 |
| 10 | 57 × 57 | 272 × 10 | 2720 | 516 | 414 | 256 |
| 11 | 61 × 61 | 316 × 10 | 3160 | 601 | 482 | 298 |
| 12 | 67 × 67 | 364 × 10 | 3640 | 691 | 554 | 343 |
| 13 | 71 × 71 | 416 × 10 | 4160 | 793 | 636 | 394 |
| 14 | 75 × 75 | 470 × 10 | 4700 | 896 | 718 | 446 |
| 15 | 79 × 79 | 528 ×10 | 5280 | 1008 | 808 | 502 |
| 16 | 83 × 83 | 588 × 10 | 5880 | 1123 | 900 | 559 |
| 17 | 87 × 87 | 652 × 10 | 6520 | 1246 | 998 | 621 |
| 18 | 91 × 91 | 720 × 10 | 7200 | 1378 | 1104 | 687 |
| 19 | 95 × 95 | 790 × 10 | 7900 | 1511 | 1210 | 753 |
| 20 | 101 × 101 | 864 × 10 | 8640 | 1653 | 1324 | 824 |
| 21 | 105 × 105 | 940 × 10 | 9400 | 1801 | 1442 | 898 |
| 22 | 109 × 109 | 1020 × 10 | 10200 | 1956 | 1566 | 976 |
| 23 | 113 × 113 | 920 × 12 | 11040 | 2116 | 1694 | 1056 |
| 24 | 117 × 117 | 992 × 12 | 11904 | 2281 | 1826 | 1138 |
| 25 | 121 × 121 | 1066 × 12 | 12792 | 2452 | 1963 | 1224 |
| 26 | 125 × 125 | 1144 × 12 | 13728 | 2632 | 2107 | 1314 |
| 27 | 131 × 131 | 1224 × 12 | 14688 | 2818 | 2256 | 1407 |
| 28 | 135 × 135 | 1306 × 12 | 15672 | 3007 | 2407 | 1501 |
| 29 | 139 × 139 | 1392 × 12 | 16704 | 3205 | 2565 | 1600 |
| 30 | 143 × 143 | 1480 × 12 | 17760 | 3409 | 2728 | 1702 |
| 31 | 147 × 147 | 1570 × 12 | 18840 | 3616 | 2894 | 1806 |
| 32 | 151 × 151 | 1664 × 12 | 19968 | 3832 | 3067 | 1914 |

Знаком «*» обозначены строки компактных символов; все остальные строки содержат данные для полноразмерных символов.

Примечание — Полноразмерные символы с 1, 2 и 3 слоями используют, как правило, только для инициализации устройства считывания.

## 6 Общий порядок кодирования

В настоящем разделе установлена последовательность преобразования исходных данных в кодированную форму (в виде символа Aztec Code). В следующих разделах настоящего стандарта установлены правила и порядок выполнения данного преобразования. Пример кодирования приведен в приложении G.

1) Aztec Code позволяет кодировать данные с помощью кодового набора, содержащего 256 знаков. Входные данные сообщения представляют в виде двоичного потока данных, считываемых слева направо. Специальные управляющие знаки FNC1 и ECI могут быть произвольно расположены в данном двоичном потоке данных;

2) каждый знак данных сообщения преобразуют в четыре, пять или восемь битов, перед которыми, в случае необходимости, размещают управляющий знак (состоящий из четырех или пяти битов) регистра или фиксации кодового набора. Таким образом формируют непрерывный двоичный поток данных;

3) минимальное число битов, подлежащих кодированию, определяют по суммарной длине двоичного потока данных и числу битов, составляющих контрольные слова исправления ошибок, которые определяют уровень исправления ошибок (заданный по умолчанию или пользователем). По числу кодируемых битов, приведенных в таблице 1, определяют формат и минимальный размер символа (число слоев данных L). По числу слоев данных определяют размер кодового слова B в битах и общую информационную емкость символа в кодовых словах $C_w$;

4) двоичный поток данных преобразовывают в кодовые слова, не допуская при этом формирования кодовых слов, содержащих только 0 или только 1. Таким образом формируют D кодовых слов сообщения;

5) число контрольных кодовых слов (K) определяют как разность между общим числом кодовых слов $C_w$ и числом кодовых слов D, содержащих исходные данные. В результате применения циклического кодирования Рида-Соломона над полем Галуа $GF(2^B)$ с порождающим многочленом порядка K генерируются K контрольных кодовых слов, которые присоединяют к последовательности кодовых слов;

6) из двоичных значений L и D формируется служебное сообщение, при этом с помощью процедуры циклического кодирования Рида-Соломона над полем GF(16) генерируют дополнительные контрольные биты;

7) изображение символа, содержащего L слоев, формируют следующим образом. Сначала размещают модули фиксированных структур — шаблона поиска, шаблонов ориентации и (для полноразмерных символов) решетки привязки. Затем вокруг шаблона поиска размещают служебное сообщение, после которого следуют спиральные слои блоков типа домино, образующие последовательность кодовых слов данных и контрольных кодовых слов, расположенных в обратном порядке.

# 7 Структура символа

## 7.1 Структура фиксированных шаблонов

Символ Aztec Code содержит три типа фиксированных шаблонов — шаблон поиска, шаблон ориентации и (в полноразмерных символах) решетку привязки (рисунки 2 и 3). Для удобства их определения в настоящем стандарте условную сетку символа следует рассматривать в декартовых прямоугольных координатах на плоскости символа с точкой начала в центре символа (координаты 0, 0) с положительными направлениями значений по оси *x* — вправо, по оси *y* — вверх.

### 7.1.1 Шаблон поиска

Шаблон поиска представляет собой группу концентрических квадратов (типа «мишень»), расположенных в ядре символа. Угловые модули шаблона поиска расположены в точках (-F, -F), (-F, F), (F, F) и (F, -F), при этом F соответствует четырем модулям для компактных символов и шести модулям — для полноразмерных символов. Светлому модулю присваивают значение «0», темному — «1», при этом цвет любого модуля в шаблоне поиска определяют по формуле

$$((\max(\mathrm{abs}(x),\mathrm{abs}(y))) + 1) \bmod 2.$$

### 7.1.2 Модули шаблона ориентации

Шаблон ориентации состоит из четырех трехмодульных групп, имеющих форму шеврона и присоединенных с внешней стороны к углам шаблона поиска. Шесть темных модулей, входящих в указанные трехмодульные группы, располагают в точках (-F-1, F), (-F-1, F+1), (-F, F+1), (F+1, F+1), (F+1, F) и (F+1, -F), а шесть светлых модулей — в точках (F, F+1), (F+1, -F-1), (F, -F-1), (-F, -F-1), (-F-1, -F-1) и (-F-1, -F).

### 7.1.3 Решетка привязки

В полноразмерном символе Aztec Code линии решетки привязки проходят по позициям условной сетки, координаты *x* или *y* которых кратны 16, т.е. ((*x*mod16) = 0) или ((*y*mod16) = 0). Если светлому

модулю присваивают значение «0», а темному — «1», то все модули решетки привязки определяют по следующей формуле

$$(x + y + 1)\text{mod}2.$$

В позициях, где элементы решетки привязки накладываются на элементы шаблона поиска, модули имеют одинаковые значения.

### 7.2 Кодирование и структура служебного сообщения

Служебное сообщение представляет собой последовательность битов, представленных соответствующими модулями, расположенными вокруг шаблона поиска. Последовательность битов служебного сообщения: сначала следуют биты с информацией о размере символа, затем — биты, определяющие длину сообщения, и, в завершении, контрольные биты, обеспечивающие необходимую избыточность для исправления ошибок в служебном сообщении.

#### 7.2.1 Указатель размера символа

Указатель размера символа содержит сведения в битах о числе слоев данных на единицу меньше, чем L, т.е. L – 1. Компактный символ содержит два бита, двоичные значения которых (соответствующие десятичные значения — от 0 до 3) соответствуют символам, содержащим от одного до четырех слоев данных. Полноразмерный символ содержит пять битов, двоичные значения которых (соответствующие десятичные значения — от 0 до 31) соответствуют символам, содержащим от одного до 32 слоев данных. В свою очередь, указанные биты определяют общие размеры символа и его емкость $C_w$, а также число битов B в каждом кодовом слове данных в соответствии с таблицей 1.

#### 7.2.2 Указатель длины сообщения

Указатель длины сообщения содержит информацию (в двоичном представлении) о числе кодовых слов данных (с фактическими данными) D минус 1. Остальные кодовые слова используют для исправления ошибок. В компактном символе указатель представлен шестью битами, а закодированные значения от 0 до 63 соответствуют числу кодовых слов данных от 1 до 64. В полноразмерном символе указатель представлен 11 битами, а значения от 0 до 2047 соответствуют числу кодовых слов данных от 1 до 2048. Логическая максимальная длина сообщения зависит от емкости символа в кодовых словах $C_w$, а рекомендуемая максимальная длина сообщения, обеспечивающая разумную защиту от ошибок, равна 0,77 $C_w$ – 3.

В специальных символах инициализации устройства считывания (раздел 9) старший значащий бит обозначения длины сообщения задают равным 1. Таким образом обозначают длину сообщения, превышающую $C_w$.

#### 7.2.3 Исправление ошибок в служебном сообщении

Информацию о размере символа и длине сообщения представляют в двоичном виде в 4-битовых кодовых словах, которые являются «словами данных» служебного сообщения. К ним добавляют несколько «контрольных слов», вычисленных с помощью циклического кода Рида-Соломона над полем Галуа GF(16) для примитивного многочлена $x^4 + x + 1$ (для десятичного представления — 19).

В компактных символах Aztec Code данные служебного сообщения, записанные в виде «ssmmmmmm» (где «ss» обозначает размер символа, а «mm...» — длину сообщения) образуют два 4-битовых слова данных, к которым добавляют пять контрольных слов с использованием порождающего многочлена $(x – 2^1)...(x – 2^5)$, соответствующего

$$x^5 + 11x^4 + 4x^3 + 6x^2 + 2x + 1.$$

В общей сложности два слова данных с пятью контрольными словами, вычисленными как коэффициенты остатка от операций умножения многочлена слов данных на $x^5$ с последующим длинным делением на порождающий многочлен над полем GF(16), образуют 28-битовое служебное сообщение для компактного символа Aztec Code.

В полноразмерных символах Aztec Code данные служебного сообщения, записанные в виде «sssssmmmmmmmmmmm» (где «ss» обозначает размер символа, а «mm...» — длину сообщения) образуют четыре 4-битовых слова данных, к которым добавляют шесть контрольных слов с использованием порождающего многочлена $(x – 2^1)...(x – 2^6)$, соответствующего

$$x^6 + 7x^5 + 9x^4 + 3x^3 + 12x^2 + 10x + 12.$$

В общей сложности четыре слова данных с шестью контрольными словами образуют 40-битовое служебное сообщение для полноразмерного символа Aztec Code.

**7.2.4 Размещение модулей в служебном сообщении**

При графическом представлении битов служебного сообщения темным модулям соответствует значение «1», а светлым — «0».

Цепочку битов служебного сообщения, начиная со старшего значащего бита обозначения размера символа и заканчивая младшим значащим битом значения последнего контрольного слова, кодируют в одномодульном слое вокруг шаблона поиска, начиная от его левого верхнего угла по направлению часовой стрелке, при этом позиции, занятые элементами шаблона ориентации и решетки привязки, пропускают.

В компактном символе 28 битов фактически преобразуются в четыре 7-разрядные цепочки. Первая часть закодирована вдоль верхней стороны шаблона поиска в направлении слева направо, вторая — вдоль правой стороны шаблона поиска в направлении сверху вниз, третья — вдоль нижней стороны шаблона поиска в направлении справа налево, четвертая — вдоль левой стороны шаблона поиска в направлении снизу вверх. В полноразмерном символе 40 битов служебного сообщения преобразованы в восемь пятиразрядных цепочек. Первые две цепочки закодированы вдоль верхней стороны шаблона поиска в направлении слева направо, и т.д.

**7.3 Кодирование и структура данных в сообщении**

Сообщение данных представляет собой последовательность $C_w$ кодовых слов, каждое размером B битов, кодируемых от ядра символа по «спирали» по часовой стрелке в наружном направлении. Эти кодовые слова включают в себя D кодовых слов, точно по числу необходимых для кодирования исходных данных в сообщении, к которым добавляют все контрольные кодовые слова, необходимые для заполнения символа. Указанные кодовые слова следуют в обратном порядке (см. 7.3.3).

**7.3.1 Кодирование исходного сообщения**

Преобразование последовательности знаков исходных данных в последовательность кодовых слов выполняют в два этапа. На первом этапе все знаки исходных данных последовательно преобразуют в соответствующие двоичные значения, при этом (при необходимости) используют кодовые значения знаков регистра (Shift) и/или знаков фиксации (Latch). Таким образом получают протяженный двоичный поток (строку) входных данных. На втором этапе указанную строку данных преобразуют в последовательность кодовых слов из B битов каждое с соблюдением правила исключения, запрещающего вхождение кодовых слов, состоящих только из 0 или только из 1.

7.3.1.1 Преобразование в двоичный поток данных

В таблице 2 приведены значения, присвоенные знакам ASCII в нескольких кодовых наборах, а также управляющим знакам регистра и фиксации для переключения между кодовыми наборами. Кодовые наборы Upper (прописные буквы) и Lower (строчные буквы) содержат прописные и строчные алфавитные буквы соответственно. Кодовый набор Mixed (смешанные знаки) состоит из управляющих знаков ASCII и специальных графических знаков. Кодовый набор Punctuation (знаки пунктуации) состоит из специальных графических знаков и их комбинаций. Кодовый набор Digit (цифры) содержит цифры и некоторые специальные графические знаки. За исключением знака «FLG(n)» представленного ниже, все знаки в таблице 2, не имеющие соответствующих десятичных значений ASCII, представляют функции сдвига регистра или фиксации для перехода к другим кодовым наборам.

Кодирование сообщения начинают в кодовом наборе Upper, а затем, по мере необходимости, переводят в другие кодовые наборы с использованием знаков фиксации (фиксация кодовых наборов с помощью знаков U/L, L/L, M/L, P/L или D/L) или сдвига регистра (переход в другой кодовый набор только для одного знака с помощью знаков U/S или P/S). В кодовых наборах Upper, Lower, Mixed и Punctuation каждый знак, знаки фиксации или регистра представлены в двоичном потоке данных 5-битовыми значениями, в то время как в кодовым наборе Digit каждый знак, знак фиксации или регистра представлены в кодовом наборе 4-битовыми значениями. Во всех случаях двоичные значения последовательно соединяют друг с другом, начиная со старшего значащего бита.

Режим Byte Shift (B/S) представляет собой особый случай переключения регистра в текстовую строку 8-битовых байтов с динамической кодировкой длины. За знаком B/S следует 5-битовое значение: если оно отличается от нуля, то в нем кодируют число последующих байтов от 1 до 31; если оно равно нулю, то следующие 11 битов кодируют число байтов минус 31. Таким образом, режим Byte Shift позволяет кодировать как изолированные знаки расширенного набора ASCII 8-битовых знаков и управляющие знаки, так и протяженные строки данных в байтах, возможно, кодируемые во всем символе. По завершении строки байтов кодирование возвращается к тому кодовому набору, из которого был активизирован режим B/S.

Таблица 2 — Высокоуровневое кодирование сообщений в символике Aztec Code

| Значение | Кодовый набор Upper[1)] | | Кодовый набор Lower | | Кодовый набор Mixed | | Кодовый набор Punctuation | | Кодовый набор Digit | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Знак | ASCII | Знак | ASCII | Знак | ASCII | Знак | ASCII | Знак | ASCII |
| 0 | P/S | | P/S | | P/S | | FLG(n) | | P/S | |
| 1 | SP | 32 | SP | 32 | SP | 32 | CR | 13 | SP | 32 |
| 2 | A | 65 | a | 97 | SOH | 1 | CR LF | 13, 10 | 0 | 48 |
| 3 | B | 66 | b | 98 | STX | 2 | . SP | 46, 32 | 1 | 49 |
| 4 | C | 67 | c | 99 | ETX | 3 | , SP | 44, 32 | 2 | 50 |
| 5 | D | 68 | d | 100 | EOT | 4 | : SP | 58, 32 | 3 | 51 |
| 6 | E | 69 | e | 101 | ENQ | 5 | ! | 33 | 4 | 52 |
| 7 | F | 70 | f | 102 | ACK | 6 | « | 34 | 5 | 53 |
| 8 | G | 71 | g | 103 | BEL | 7 | # | 35 | 6 | 54 |
| 9 | H | 72 | h | 104 | BS | 8 | $ | 36 | 7 | 55 |
| 10 | I | 73 | i | 105 | HT | 9 | % | 37 | 8 | 56 |
| 11 | J | 74 | j | 106 | LF | 10 | & | 38 | 9 | 57 |
| 12 | K | 75 | k | 107 | VT | 11 | ' | 39 | , | 44 |
| 13 | L | 76 | l | 108 | FF | 12 | ( | 40 | . | 46 |
| 14 | M | 77 | m | 109 | CR | 13 | ) | 41 | U/L | |
| 15 | N | 78 | n | 110 | ESC | 27 | * | 42 | U/S | |
| 16 | O | 79 | o | 111 | FS | 28 | + | 43 | | |
| 17 | P | 80 | p | 112 | GS | 29 | , | 44 | | |
| 18 | Q | 81 | q | 113 | RS | 30 | - | 45 | | |
| 19 | R | 82 | r | 114 | uS | 31 | . | 46 | | |
| 20 | S | 83 | s | 115 | @ | 64 | / | 47 | | |
| 21 | T | 84 | t | 116 | \ | 92 | : | 58 | | |
| 22 | U | 85 | u | 117 | ^ | 94 | ; | 59 | | |
| 23 | V | 86 | v | 118 | _ | 95 | < | 60 | | |
| 24 | W | 87 | w | 119 | ` | 96 | = | 61 | | |
| 25 | X | 88 | x | 120 | \| | 124 | > | 62 | | |
| 26 | Y | 89 | y | 121 | ~ | 126 | ? | 63 | | |
| 27 | Z | 90 | z | 122 | DEL | 127 | [ | 91 | | |
| 28 | L/L | | U/S | | L/L | | ] | 93 | | |
| 29 | M/L | | M/L | | U/L | | { | 123 | | |
| 30 | D/L | | D/L | | P/L | | } | 125 | | |
| 31 | B/S | | B/S | | B/S | | U/L | | | |

[1)] *Кодовый набор Upper — набор, содержащий прописные буквы латинского алфавита.*

*Кодовый набор Lower — набор, содержащий строчные буквы латинского алфавита.*

*Кодовый набор Mixed — набор, содержащий управляющие знаки ASCII и специальные графические знаки.*

*Кодовый набор Punctuation — набор, содержащий управляющие знаки ASCII, специальные графические знаки и их комбинации, а также знаки пунктуации.*

*Кодовый набор Digit — набор, содержащий цифры и некоторые специальные графические знаки.*

Знак-указатель FLG(n) в графе «Punctuation» таблицы 2 представляет собой специальный флаг с активизацией «на месте», который используют для представления различных знаков, не относящихся к данным, поддерживаемых многими стандартизованными символиками. В двоичном потоке за значением знака FLG(n) следуют три дополнительных бита, кодирующих аргумент «n» в двоичном виде. Таким образом, значение n находится в диапазоне от нуля до шести (значение семь является недействительным).

Знак FLG(0) представляет собой знак FNC1 — не относящийся к данным флаг, присутствующий в символике Code 128. Если знак FNC1 предшествует первому знаку данных в сообщении, он указывает на структурирование данных в соответствии с правилами GS1 с использованием идентификаторов применения и влияет на значение знака — модификатора идентификатора символики. Если знак FNC1 следует непосредственно за одной прописной или строчной буквой или двумя цифрами в начале сообщения, то в этом случае он сигнализирует о применении другого стандартного отраслевого формата, определяемого предшествующими данными, и тоже влияет на значение знака — модификатора идентификатора символики. Если знак FNC1 находится в любой другой позиции данных, то он выполняет функцию разделителя полей, а его место в выходной строке занимает управляющий знак ASCII (<GS>) с десятичным значением 29.

Знаки FLG(1) — FLG(6) представляют собой флаг интерпретации в расширенном канале ECI (Extended Channel Interpretation), при обнаружении которого устройство считывания, настроенное на использование интерпретации в расширенном канале, вставляет в выходной поток данных последовательность «\nnnnnn», т.е. знак «ОБРАТНАЯ ДРОБНАЯ ЧЕРТА», за которым следует число из шести цифр (указанное устройство считывания также дублирует знак «ОБРАТНАЯ ДРОБНАЯ ЧЕРТА» в закодированных данных и задает соответствующее значение знаку — модификатора идентификатора символики). Аргумент n в данном случае указывает, сколько из шести цифр явно закодировано в символе с использованием кодового набора Digit. При этом предполагают, что остальными цифрами являются начальные нули. Например, ECI 000123 кодируют в форме FLG(3)123, после чего кодирование возвращают к кодовому набору, из которого был активизирован знак FLG(n).

В приложении H приведен алгоритм определения последовательности кодовых наборов, знаков фиксации и регистра, обеспечивающей минимальное число битов для кодирования входной последовательности байтов и, в итоге, устанавливающей минимальный размер символа с максимальной избыточностью для исправления ошибок.

7.3.1.2 Формирование кодовых слов данных

На втором этапе кодирования входной поток данных после выбора общего размера символа и определения числа битов B в кодовом слове двоичного потока данных в сообщении преобразуют в последовательность кодовых слов данных в сообщении с числом битов B, где B равно 6, 8, 10 или 12. В общем случае преобразование начинают со старшего значащего бита первого кодового слова с двумя принципиальными исключениями: каждый раз, когда первые B – 1 битов, помещенных в кодовое слово, состоят из одних «0», в младший значащий бит этого кодового слова вставляют фиктивную «1» и следующий бит данных в сообщении начинает следующее кодовое слово. Аналогично в младший значащий бит кодовых слов данных в сообщении, начинающихся с B – 1 битов, равных «1», вставляют «0». Таким образом, кодовые слова данных в сообщении, состоящие из одних «0» или «1», являются недействительными и идентифицируются как ошибка стирания в ходе декодирования.

В итоге границы знаков и байтов в исходных данных в сообщении не имеют четкого соответствия с границами кодовых слов. В последнем кодовом слове данных сообщения до B – 1 битов могут оставаться незаполненными. В этом случае эти биты заполняют «1» (при необходимости с добавлением завершающего фиктивного «0») для устранения любых неоднозначностей.

Примечание — Поскольку шаблон-заполнитель может находиться в начале последовательности со знаком регистра Binary Shift, устройство считывания должно обеспечить завершение интерпретации сообщения на последнем кодовом слове данных.

### 7.3.2 Кодирование кодовых слов исправления ошибок в данных сообщения

В результате кодирования исходного потока формируют D кодовых слов. К ним присоединяют K контрольных слов, число которых определяют как $C_w$ – D, вычисляемых с использованием алгоритма циклического кодирования Рида-Соломона над полем Галуа $GF(2^B)$ для соответствующего примитивного многочлена (таблица 3). Значения контрольных слов соответствуют коэффициентам многочлена, представляющих собой остаток от операций умножения многочлена, образованного кодовыми словами данных в сообщении, на $x^K$ с последующим длинным делением на порождающий многочлен $(x – 2^1)...(x – 2^K)$ (приложение B, раздел B.1).

Таблица 3 — Размеры кодовых слов данных и примитивные многочлены

| Число слоев в символе | Кодовые слова | Поле Галуа | Примитивный многочлен | Представление в двоичном виде | Представление в десятичном виде |
|---|---|---|---|---|---|
| От 1 до 2 | 6-битовые | GF(64) | $x^6 + x + 1$ | 1000011 | 67 |
| От 3 до 8 | 8-битовые | GF(256) | $x^8 + x^5 + x^3 + x^2 + 1$ | 100101101 | 301 |
| От 9 до 22 | 10-битовые | GF(1024) | $x^{10} + x^3 + 1$ | 10000001001 | 1033 |
| От 23 до 32 | 12-битовые | GF(4096) | $x^{12} + x^6 + x^5 + x^3 + 1$ | 1000001101001 | 4201 |

### 7.3.3 Размещение модулей данных сообщения в символе

В графическом представлении битов данных в сообщении темные модули представляют «1», а светлые модули — «0».

Для размещения в символе Aztec Code последовательность D + K кодовых слов в обратном порядке укладывают в L слоев, толщина каждого из которых равна двум модулям. Размещение модулей начинают над левым верхним углом ядра символа по спирали по часовой стрелке в наружном направлении. Чтобы лучше понять точную структуру размещения отдельных модулей, сначала представляют каждое кодовое слово в виде «кирпича» высотой 2 модуля (рисунок 4), а затем разбивают этот «кирпич» на B/2 домино шириной 1 модуль. В каждом домино бит старшего разряда расположен над битом младшего разряда. Затем полученные домино размещают в слоях данных, начиная с буквы «С» надписи «Слой 1 ...» на рисунках 2 или 3, а затем поворачивают по часовой стрелке. Позиции, занимаемые решеткой привязки, при этом пропускают. Бит младшего разряда каждого домино всегда расположен ближе к центру символа. Таким образом, домино, размещаемые в нижней части символа, кажутся перевернутыми.

На рисунке 5 приведена последовательность размещения и ориентация домино при повороте на углах в любом слое и при переходах между слоями. Знаки символов часто имеют нерегулярную форму на четырех углах или содержат разрывы при пропускании решетки привязки, но, несмотря на это, они представляют собой целостные двоичные значения кодовых слов. В слоях 12 и 27 полноразмерных символов сами домино разделены надвое ячейками решетки привязки, но даже в этом случае бит старшего разряда каждого домино должен быть расположен непосредственно над битом младшего разряда (по линии, проведенной наружу от ядра символа) с соблюдением ориентации (рисунок 5).

В граничном внешнем слое каждого символа на конце спирали могут остаться несколько неиспользованных домино, у которых оба модуля остаются светлыми («0»).

## 8 Структурированное соединение

При недостатке места для размещения символа Aztec Code, а также для обработки данных в сообщении большого объема, представление которых одним символом неэффективно, данные сообщения могут быть распределены по нескольким символам Aztec Code. Данная задача может быть решена с применением структуры заголовков, определяемой пользователем, однако далее приведен стандартный метод, который должен поддерживаться устройствами считывания Aztec Code.

Во всех символах, входящих в последовательность структурированного соединения, кодирование данных сообщения начинается с последовательности

M/L U/L [пробел I...D пробел] M N ...,

содержащей флаговую последовательность структурированного соединения «M/L U/L», где M/L и U/L — знаки фиксации, «I...D» — необязательное поле идентификатора сообщения, отделенное пробелами, а M N — два знака поля порядка следования символов «M-из-N». После указанного заголовка следует кодируемый сегмент данных в сообщении.

Устройство считывания символа Aztec Code, обнаружив знак Upper Latch перед какими-либо сохраненными данными, должно либо интерпретировать заголовок и сохранить сегмент данных в сообщении для его последующей передачи в составе полностью восстановленного сообщения, либо выдать сигнал о том, что символ является частью упорядоченного соединения, и передать соответствующую информацию о порядке его следования. Протокол интерпретации в расширенном канале (ECI) предоставляет собой механизм передачи этой информации, не зависящий от символики, для применений, использующих

ECI. С другой стороны, устройство считывания символов Aztec Code может сигнализировать о символе структурированного соединения установкой соответствующего значения знака — модификатора идентификатора символики, с последующей передачей неизмененного заголовка и сегмента данных в сообщении.

Необязательное поле идентификатора сообщения содержит произвольную последовательность знаков данных, которая начинается и завершается пробелом и должна быть одинаковой во всех символах, составляющих сообщение. Хотя строка «I...D» может состоять из произвольных знаков (за исключением дополнительных пробелов), самое эффективное кодирование обеспечивается в том случае, если она является строкой прописных букв.

Поле порядка следования символов состоит из двух прописных букв: первая кодирует порядковый номер символа в последовательности, а вторая — общее число символов в последовательности, где «A» = 1, «B» = 2, и т.д. Например в серии из четырех символов следует использовать порядковые поля «AD», «BD», «CD» и «DD». Использование протокола структурированного соединения позволяет связать до 26 символов.

## 9 Символы для инициализации устройства считывания

Часто бывает полезно иметь символы, единственной функцией которых является передача данных для выполнения инициализации и настройки конфигурации устройства считывания. Компактный символ Aztec Code с одним слоем, или полноразмерный символ, содержащий от 1 до 22 слоев, может быть закодирован для выполнения инициализации устройства считывания искусственной установкой старшего значащего бита указателя длины данных в сообщении. При этом младшие биты кодируют фактическую длину данных в сообщении. Поскольку общая емкость такого символа меньше обозначенной длины данных в сообщении, некорректная установка сигнализирует о том, что закодированные данные в сообщении не предназначены для передачи, а будут использоваться для настройки устройства считывания. Формат и значение закодированных данных определяются изготовителем устройства считывания.

## 10 Интерпретация в расширенном канале

Протокол интерпретации в расширенном канале (ECI) позволяет интерпретировать выходной поток данных способом, отличающимся от кодирования по умолчанию. Протокол ECI имеет согласованное определение в нескольких символиках. Поддерживаются четыре широкие категории интерпретаций:

a) международные наборы знаков (или кодовые наборы);

b) интерпретации общего назначения (например, для целей шифрования и уплотнения);

c) интерпретации, определяемые пользователем для закрытых систем;

d) управляющая информация структурированного соединения в небуферизованном режиме.

Протокол ECI полностью установлен в документе AIM Inc. «Международная техническая спецификация: Интерпретация в расширенном канале, часть 1» (International Technical Specification: Extended Channel Interpretations Part 1). Протокол обеспечивает унифицированный метод для определения установленных интерпретаций значений в байтах перед выводом на печать и после декодирования.

Интерпретация в расширенном канале (ECI) определяется числом, состоящим из шести цифр, закодированным в символе Aztec Code знаком ECI, за которым следуют от одного до шести 4-битовых значений, представляющих цифры. Конкретные интерпретации приведены в документе AIM Inc. «Регистр наборов знаков интерпретации в расширенном канале» (Extended Channel Interpretations Character Set Register).

Интерпретация в расширенном канале (ECI) может использоваться только с устройствами считывания, обеспечивающими передачу идентификаторов символик. Устройства считывания, у которых передача идентификатора символики отсутствует или не используется, не должны передавать данные из любых символов, содержащих интерпретацию в расширенном канале (ECI). Исключение допустимо в том случае, если интерпретация в расширенном канале (ECI) полностью обрабатываются устройством считывания.

### 10.1 Кодирование интерпретаций в расширенном канале в символах Aztec Code

Назначение интерпретаций в расширенном канале (ECI) активизируют путем кодирования знака FLG(n) из кодового набора Punctuation, за которым следуют три бита, кодирующих значение «n» от одного до шести в двоичном виде. За знаками FLG(1-6) следуют от одного до шести 4-битовых значений, которые представляют собой номер ECI с использованием кодового набора Digit. Начальные нули, пропущенные в процессе кодирования, снова вставляют в выходные данные сообщения при декодировании, дополняя номер ECI до шести битов.

### 10.2 Кодовые наборы и интерпретации в расширенном канале

Используемые кодовые наборы определяются исключительно кодируемыми 8-битовыми значениями данных и не зависят от действующей интерпретации в расширенном канале (ECI). Например последовательность значений в диапазоне от 48 до 57 десятичных значений ASCII наиболее эффективно кодируют в кодовом наборе Digit, даже если эта последовательность не будет интерпретироваться как цифры.

### 10.3 Интерпретации в расширенном канале и структурированное соединение

Интерпретации в расширенном канале (ECI) могут находиться в любой позиции данных в сообщении, закодированном одним символом или комплектом символов Aztec Code структурированного соединения. Активизируемая интерпретация ECI действует до конца закодированных данных или до обнаружения других интерпретаций в расширенном канале (ECI). Таким образом интерпретация ECI может распространяться на два и более символа.

### 10.4 Протокол постдекодирования

Протокол передачи данных в ECI должен соответствовать приведенному в 16.3. При использовании ECI идентификаторы символики (16.4) должны быть реализованы полностью, а соответствующий идентификатор символики следует передавать в качестве преамбулы к передаваемым данным.

## 11 Возможности пользователей

Потенциальные пользователи символики Aztec Code могут принять решения относительно кодируемых данных, необходимого уровня исправления ошибок и возможности разделения данных в сообщении на несколько символов.

### 11.1 Выбор пользователем кодируемых данных в сообщении

Во многих применениях штрихового кода кодируемые данные в сообщении жестко определены заранее, на стадии планирования может существовать определенная свобода выбора форматов данных, оптимизированных для эффективного кодирования. В общем случае максимальная эффективность кодирования достигается тогда, когда данные в сообщении состоят из длинной последовательности знаков, принадлежащих к одному кодовому набору (таблица 2). Текст, состоящий только из прописных букв латинского алфавита, только цифровых данных или данных, представленных в виде байтов, характеризует три типа данных в сообщении, наиболее эффективно кодируемых в символах Aztec Code.

### 11.2 Выбор пользователем минимального уровня исправления ошибок

С технической точки зрения символика Aztec Code допускает, чтобы символ не включал в себя кодовых слов исправления ошибок или чтобы их число достигало 99 %, хотя оба этих предельных значения практически не применяют. Рекомендуемый уровень исправления ошибок для обычных применений составляет 23 % информационной емкости символа плюс 3 кодовых слова. Таким образом 5-слойный символ, содержащий 120 кодовых слов, обычно содержит по меньшей мере 28 плюс 3 контрольных кодовых слова, а для кодирования данных сообщения остается до 89 кодовых слов. Программы печати должны обеспечивать данный уровень исправления ошибок по умолчанию.

Если пользователь предъявляет менее жесткие требования к надежности приложения или считает надежность приложения недостаточной, он может выбрать другой минимальный процент исправления ошибок (менее 10 %) или более 50 % с обязательным использованием трех дополнительных контрольных слов для обеспечения безопасности данных. Процент назван «минимальным» потому, что в зависимости от длины данных в сообщении символика требует добавления дополнительных контрольных слов выше этого минимума для заполнения символа.

Некоторые применения лучше всего работают при фиксированном размере (числе слоев данных) всех символов Aztec Code независимо от содержащихся в них данных. В этом случае пользователь должен задать размер, включающий в себя соответствующий уровень исправления ошибок для самого длинного предполагаемого сообщения. Тогда более короткие сообщения будут кодироваться с избыточным уровнем исправления ошибок, а символы в данном применении будут более надежными.

### 11.3 Выбор пользователем структурированного соединения

Структурированное соединение позволяет распределить данные в сообщении в нескольких символах Aztec Code, а затем восстановить их после того, как все символы были считаны в произвольном порядке. Структурированное соединение обеспечивает возможность нанесения символов Aztec Code в областях, форма которых отличается от квадратной. Оно также доступно для кодирования данных сообщения, длина которых превышает практическую емкость одного символа Aztec Code.

### 11.4 Выбор пользователем необязательных форматов символов

В соответствии с таблицей 1 программы печати обычно печатают символ наименьшего размера для заданного набора данных и процента исправления ошибок в последовательности слоев данных 1*, 2*, 3*, 4*, 4, 5, 6, ... , 32. Для символов инициализации устройства считывания программа печати обычно печатает символ наименьшего размера в последовательности слоев данных 1*, 1, 2, 3, 4, 5, ... , 22. В другом случае пользователь может задать размер и формат нужного символа, переустанавливая настройки по умолчанию.

Пользователи должны явно задать режим печати символов Aztec Runes для применений, работающих с небольшими наборами данных.

## 12 Размеры

Основным размером символов Aztec Code является размер X, который соответствует расстоянию между центрами элементов сетки с номинально квадратными ячейками, а также номинально соответствует высоте и ширине квадратных модулей этой сетки.

Спецификация символики не ограничивает значения размера X. Тем не менее, значения горизонтального и вертикального шагов сетки должны оставаться постоянными в пределах символа и отличаться не более чем на 10 % друг от друга. Более того, ширина каждого темного модуля должна быть равна горизонтальному шагу сетки ± 20 %, а его высота — вертикальному шагу сетки ± 20 %.

## 13 Рекомендации для пользователей

### 13.1 Интерпретация для визуального чтения

Интерпретация для визуального чтения данных, закодированных в сообщении, может сопровождать символ Aztec Code в любой области за пределами внешнего слоя данных (рисунок G.2). Размер и тип шрифта в настоящем стандарте не установлены.

### 13.2 Возможность автораспознавания

Символ Aztec Code может быть считан декодерами штриховых кодов, запрограммированными соответствующим образом и спроектированными так, чтобы автоматически отличать его от других символик.

Для обеспечения максимальной надежности считывания действующий набор символик декодера должен ограничиваться теми символиками, которые необходимы для данного применения.

### 13.3 Параметры применения, определяемые пользователем

Стандарты по применению должны определять следующие параметры символов Aztec Code, установленные в настоящем стандарте, как переменные:

a) данные, подлежащие кодированию;

b) размер X, используемый при печати символа;

c) размещение символа на этикетке, объекте или документе;

d) требуемый уровень качества печати символа.

Также допускается переопределение (переустановка) уровня исправления ошибок по умолчанию, для чего указывают либо другой минимальный процент исправления ошибок, либо фиксированный формат и размер символа.

## 14 Рекомендуемый алгоритм декодирования

Рекомендуемый алгоритм декодирования находит символы в изображении и декодирует их. Именно этот алгоритм декодирования используют при определении качества печати символов. Он также может использоваться практическими реализациями устройств считывания.

Декодирование символа Aztec Code из сохраненного изображения состоит из следующих этапов:

1) определение положения символа на изображении;

2) обработка шаблона поиска «мишень» с целью определения его центра, основных осей символа и номинальных расстояний между модулями;

3) декодирование ядра символа: сначала алгоритм определяет, не инвертированы ли черный/белый цвета, потом определяет формат символа, его ориентацию, а затем проводит идентификацию, декодирование и проверку служебного сообщения;

4) определение структуры, декодирование и проверка данных сообщения;

5) преобразование кодовых слов данных в выходной поток знаков.

Предполагается, что алгоритм декодирования работает с изображением, представленном в двоичном виде, в котором каждому пикселю присваивают значение «0» для темного модуля или «1» — для светлого (не следует путать данное правило с правилами присваивания значений модулям.) Если изображение хранится в оттенках серого, то сначала необходимо использовать пороговую методику для получения соответствующего изображения в двоичном виде. Для оценки качества печати символа выбирают глобальное пороговое значение, которое представляет собой среднюю величину между максимальным и минимальным значениями коэффициента отражения в изображении, после чего каждому пикселю присваивают значение «1» при превышении указанного порога или «0» — в противном случае. На практике устройство считывания может использовать при обработке неглобальный порог, учитывающий неравномерную освещенность и условия просмотра, и даже применять межпиксельную интерполяцию для построения изображения в двоичном виде с более высоким пространственным разрешением, чем у изображения в оттенках серого. Это минимизирует потери информационного содержимого на этапе предварительной обработки.

### 14.1 Поиск символов-кандидатов

Шаблон поиска «мишень» в символе Aztec Code обладает двумя четко выраженными характеристиками, которые отличают его от других объектов в изображении. Первая: на топологическом уровне его центральный модуль «изолирован» от остальных частей символа, как остров на озере, которое находится на другом острове на озере, и т.д. Это позволяет легко распознавать его на отсканированном изображении, на котором идентифицируются связанные области. В приложении С описан метод сканирования для данной топологии.

Каждая горизонтальная строка топологического алгоритма поиска, содержащая секцию с тремя и более участками постоянно возрастающего шаблона, за которыми следуют три и более участка постоянно убывающего шаблона (строка 9 на рисунке С.7), может рассматриваться как возможный кандидат при обнаружении шаблона поиска «мишень». Наибольшее значение в этой секции становится кандидатом на роль шаблона поиска «мишень» (значение пикселя 4 в строке 9 на рисунке С.7). Описанный критерий является рекомендуемым методом верификации.

В соответствии с этим сканирование, отслеживающее границы на изображении, обнаруживает шаблон поиска «мишень» как одну границу, полностью включенную внутрь другой границы, которая в свою очередь полностью включена внутрь следующей границы, и т.д. Каждый из этих методов обеспечивает надежное обнаружение центров символов-кандидатов, даже в сильно искаженных изображениях, при условии, что их шаблоны поиска остались неповрежденными.

Вторая отличительная характеристика шаблона поиска «мишень», которую можно использовать в работе устройств считывания: шаблон поиска «мишень» состоит из выровненных прямых углов, направленных от центрального модуля наружу. Сканирование изображения, обнаруживающее скопление таких углов, особенно соединенных с длинными отрезками или другими углами, должно обнаружить шаблон поиска «мишень» даже при его частичном повреждении. При таком подходе шаблон поиска «мишень», стертый на 25 %, но сохранивший, по меньшей мере, две группы углов, будет надежно обнаружен.

### 14.2 Обработка изображения шаблона поиска «мишень»

Изображение шаблона поиска обрабатывают с целью нахождения как можно большего числа углов и их сопоставления с другими углами того же концентрического квадрата шаблона поиска «мишень». Позицию центрального модуля определяют как среднюю позицию между противостоящими парами углов. Две главные оси символа определяют как направления между смежными парами углов. Расстояния между модулями вдоль этих осей определяют расстояниями между смежными углами.

Пять критических начальных параметров (точный центр символа, направление двух его главных осей и номинальные расстояния между модулями вдоль этих осей) определяют несколькими способами по особенностям квадратного шаблона поиска «мишень», поэтому они могут определяться устройствами считывания даже в том случае, если шаблон поиска содержит незначительные повреждения.

### 14.3 Декодирование ядра символа

Декодирование ядра символа начинают с определения центров всех модулей квадратной области, центр которой совпадает с центром шаблона поиска «мишень», и определения того, являются они светлыми или темными. По полученной информации сначала определяют визуальную полярность изображения символа (нормальное или инвертированное) и его формат (компактный или полноразмерный), а затем его

ориентацию (направление к верхнему краю символа) и возможные зеркальные отображения, после чего осуществляют сборку, проверку и обработку служебного сообщения.

**14.3.1 Составление схемы и дискретизация центров модулей**

Рекомендуемый метод составления схемы центров модулей в символах Aztec Code может быть в общем виде описан как «выращивание двумерного кристалла». Начиная с позиции центрального модуля, формируют первый слой из центров окружающих его восьми модулей. Эти позиции устанавливают так, чтобы они были равномерно распределены в пределах внутреннего (обычно светлого) кольца шаблона поиска «мишень», после чего формируют следующий слой из центров 16 модулей, подбираемых так, чтобы они попали в пределы следующего (обычно темного) кольца, и т.д. Процесс составления и корректирования позиций центров модулей по отношению к окружающим границам может быть выполнен по спирали в наружном направлении, толщиной один модуль, пока не будет составлена схема всего ядра символа. Характеристика центра каждого модуля в двоичном виде определяется и сохраняется в битовой карте символа.

**14.3.2 Определение инверсии изображения и формата символа**

Если среди восьми модулей, окружающих центр шаблона поиска, преобладают светлые, то в ходе процесса декодирования темные области обычно считают эквивалентными двоичной «1». В противном случае считают, что изображение было инвертировано и светлые области представляют «1».

Пятое кольцо модулей, окружающих центр шаблона поиска (квадрат 11 × 11), используют для определения формата символа: если оно содержит четыре и более двоичных «1» (обычно темных модулей), то символ считают компактным символом Aztec Code и составление схемы ядра символа на этом завершают. В противном случае символ Aztec Code считают полноразмерным и составление схемы ядра символа продолжают еще на два слоя модулей в наружном направлении.

**14.3.3 Определение ориентации и зеркального отображения символа**

Во внешнем слое ядра символа проводят двоичное сравнение 12 битов, представляющих собой модули шаблона ориентации, находящихся в углах, с заданным шаблоном для каждого из четырех возможных способов ориентации и для четырех возможных зеркальных отображений этих шаблонов. Если в любом из восьми проверенных случаев выявляется правильное совпадение от 9 до 12 битов, то такую ориентацию считают истинной. В противном случае считают, что декодирование завершилось неудачей.

**14.3.4 Декодирование служебного сообщения**

При наличии сведений о формате и ориентации символа модули служебного сообщения преобразуют в биты, разбивают на 4-битовые пакеты и проводят в них исправление ошибок над полем GF(16). В процессе исправления ошибок могут использоваться все значения синдромов, поскольку даже ничтожная вероятность необнаруженной ошибки в служебном сообщении в итоге приведет не к невозможности считывания, а к неверному считыванию символа. Если исправление ошибок завершается неудачей, то принимают решение о невозможности декодирования. В противном случае из служебного сообщения извлекают информацию о числе слоев данных L и числе слов данных D. В особом случае, когда старший значащий бит для D установлен и D превышает емкость L слоев, данных символ идентифицируют как символ инициализации устройства считывания, а старший бит в D сбрасывают.

**14.4 Декодирование данных сообщения**

Декодирование данных сообщения начинают с составления схемы и выбора центров для всех оставшихся модулей в границах символа с последующей сборкой и проверкой спиральной последовательности кодовых слов.

**14.4.1 Составление схемы слоев данных**

Для символов Aztec Code, содержащих четыре и менее слоев данных, в том числе для всех компактных символов Aztec Code, «выращивание двумерного кристалла» продолжают в наружном направлении для составления схемы всех L слоев областей данных.

В полноразмерных символах, имеющих больший размер, используют более точный метод составления схемы позиций центров решетки привязки. Центры модулей решетки привязки определяют по алгоритму линейного выращивания кристалла, приведенному в приложении D. После составления схемы решетка привязки обеспечивает набор локализованных эталонных координат, по которым вычисляют и определяют позиции центров всех модулей данных.

Конечным результатом применения обоих методов — «выращивания двумерного кристалла» и составления схемы решетки привязки — является битовая карта, распространяющаяся на всю область символа.

### 14.4.2 Сборка кодовых слов

Размер и число кодовых слов в символе определяют значением L — числом слоев данных, закодированным в служебном сообщении (таблица 1). Сборку кодовых слов проводят в обратном порядке, начиная от последнего контрольного кодового слова с постепенным продвижением к первому кодовому слову данных, по битовой карте символа. Спираль начинается из левого верхнего угла ядра символа и продолжается в наружном направлении по часовой стрелке. Простая циклическая процедура, которая параллельно обрабатывает два слоя модулей, а для полноразмерных символов пропускает все модули, у которых координата *x* или *y* кратна 16, позволяет собрать полную последовательность кодовых слов для символа Aztec Code любого размера.

### 14.4.3 Проверка кодовых слов

Последовательность $C_w$ кодовых слов, из которых D кодовых слов содержат данные, а K кодовых слов (кодовые слова исправления ошибок, вычисляемых по формуле $K = C_w — D$) — контрольную информацию (таблица 3), проверяют и корректируют с использованием процедур Месси-Берлекэмпа/Чена/Форни (Massey-Berlekamp/Chien/Forney), усовершенствованных для эффективной обработки стираний над соответствующим полем Галуа $GF(2^B)$ (приложение В, раздел В.3). Все кодовые слова данных, содержащие только «0» или «1», а также все кодовые слова с модулями, выходящими за пределы изображения, считают стираниями. Процедура Месси-Берлекэмпа начинается с вычисления значений K синдромов и, следовательно, легко адаптируется к числу контрольных кодовых слов, содержащихся в символе. Два (более двух в случае преобладания ошибок — см. раздел В.2) контрольных кодовых слова резервируют для обнаружения ошибок, так что для исправления ошибок используют K-2 (или менее) контрольных кодовых слов. Если выполнить исправление ошибок не удается, декодирование считают несостоявшимся.

## 14.5 Преобразование кодовых слов данных

В ходе преобразования кодовых слов данных (процедура, обратная по отношению к кодированию данных сообщения) слова сначала преобразуют в двоичный поток, а затем последовательные сегменты этого потока преобразуют в поток выходных знаков данных.

### 14.5.1 Формирование двоичного потока данных

Сборка двоичного потока начинается со старшего значащего бита первого кодового слова данных и завершается младшим битом последнего кодового слова данных. При этом пропускают младшие значащие биты всех слов, у которых все наиболее значащие биты содержат только «0» или только «1».

### 14.5.2 Преобразование двоичного потока

Преобразование двоичного потока в знаки данных осуществляют в соответствии с таблицей 2, начиная с кодового набора Upper.

Двоичный поток разбивают на последовательные 5-битовые, 4-битовые, иногда на 11- и 8-битовые значения (с использованием знака Byte Shift) с фиксацией и сдвигом регистра между различными кодовыми наборами. Завершающие биты данных последнего кодового слова, состоящие из одних единиц, игнорируют. Выходная последовательность знаков данных завершается окончанием двоичного потока.

Если в самом начале интерпретации до первого знака данных закодирован знак фиксации кодового набора Upper Latch, то символ идентифицируют как содержащий заголовок структурированного соединения. Далее он либо обрабатывается устройством считывания, либо передается в неизменном виде с включением соответствующего идентификатора символики.

# 15 Качество печати символа

Оценку качества печати символов Aztec Code проводят в соответствии с рекомендациями по качеству печати двумерных матричных символов, установленными в ИСО/МЭК 15415 со следующими дополнениями и изменениями.

## 15.1 Параметры качества печати символа

### 15.1.1 Повреждение фиксированного шаблона

Основные принципы измерения и оценки повреждений фиксированных шаблонов установлены в приложении Е.

### 15.1.2 Осевая неоднородность

Символы Aztec Code имеют номинально квадратную форму. Следовательно, для оценки осевой неоднородности AN (Axial Nonuniformity) необходимо использовать общую высоту H и ширину W символа

(вместо среднего расстояния между центрами модулей) как определено при рекомендуемом алгоритме декодирования. AN вычисляют по формуле

$$AN = abs(H - W)/((H + W)/2).$$

Оценку параметра осуществляют в точном соответствии с пунктом 7.8.6 ИСО/МЭК 15415:2004.

**15.1.3 Неиспользованное исправление ошибок**

Поля, содержащие код Рида-Соломона в служебном сообщении и в данных сообщения, оценивают независимо друг от друга в соответствии с пунктом 7.8.8 ИСО/МЭК 15415:2004 с использованием p = 0 для служебного сообщения и p = 2 для данных сообщения. При оценке неиспользованного исправления ошибок должен быть выбран меньший из двух полученных классов.

**15.1.4 Приращение при печати**

Оценка приращения при печати основана на проверке того, что «коэффициент заполнения» линий, проходящих через шаблон поиска и вдоль решетки привязки (если она имеется), обычно равный 50 %, находится в диапазоне от 30 % до 70 % включительно.

Линии, проходящие через центр символа и выровненные по каждой из его основных осей, перебирают слева направо, а затем сверху вниз. В процессе перебора суммируют число обнаруженных светлых ($N_L$) и темных ($N_D$) пикселей. В полноразмерных символах обработку проводят по всей ширине или высоте символа, а в компактных символах обработку ограничивают шаблоном поиска. Число пикселей нормализуют по числу светлых или темных модулей, которые эти линии должны были пересечь. В результате получают значения $N_L'$ и $N_D'$. Итоговая оценка приращения при печати по каждому направлению составляет

$$D = N_D' / (N_L' + N_D').$$

Полученное значение оценивают по критерию $D_{nom} = 0{,}50$ с пороговыми значениями $D_{min} = 0{,}30$ и $D_{max} = 0{,}70$:

$$D' \begin{cases} = (D - D_{nom}) / (D_{max} - D_{nom}), \text{ если } D > D_{nom}, \\ = (D - D_{nom}) / (D_{nom} - D_{min}) - \text{в противном случае.} \end{cases}$$

Соответствующий класс качества печати определяют следующим образом:

A (4,0), если — $0{,}50 \le D' \le 0{,}50$;

B (3,0), если — $0{,}70 \le D' \le 0{,}70$;

C (2,0), если — $0{,}85 \le D' \le 0{,}85$;

D (1,0), если — $1{,}00 \le D' \le 1{,}00$;

F (0,0), если $D' < -1{,}00$ или $D' > 1{,}00$.

Класс приращения при печати должен быть равен меньшему из двух значений, определенных по горизонтали и по вертикали.

**15.2 Оценка качества печати символа**

**15.2.1 Класс качества изображения**

Классом качества печати для каждого считанного изображения считают наименьший класс, который выбирают из классов параметров, вычисленных по ИСО/МЭК 15415, и классов приращения при печати, вычисленных в соответствии с 15.1.4 настоящего стандарта. Полученный класс соответствует классу качества изображения.

**15.2.2 Класс качества символа**

Для каждого символа определяют пять или более классов качества изображений при повороте символа на 360° вокруг центральной точки шаблона поиска. Углы поворота для всех изображений должны быть приблизительно равными.

Класс качества символа вычисляют как среднее арифметическое значение классов качества изображений в соответствии с 15.2.1.

**15.3 Измерения в процессе технологического контроля**

Существуют ряд средств и методов проведения измерений с целью мониторинга и контроля за процессом создания символов Aztec Code. К их числу относятся:

1) значения контраста символов, полученные с помощью верификатора линейного штрихового кода;

2) специальные эталонные символы, упрощающие визуальную проверку неровностей сетки и измерение приращения при печати с помощью верификатора линейного штрихового кода;

3) определение осевой неоднородности путем физического измерения;

4) визуальная проверка шаблона поиска для выявления критических дефектов.

Эти средства и методы описаны в приложении I.

# 16 Передаваемые данные

## 16.1 Основная интерпретация

Символы, предназначенные для инициализации устройства считывания, не должны содержать никаких данных. Данные символов последовательности структурированного соединения либо сохраняются в буфере для последующей передачи в составе полностью собранных данных сообщения, либо передаются с соответствующим знаком-модификатором идентификатора символики и порядковым заголовком.

Для всех остальных символов Aztec Code должна передаваться полная строка данных сообщения, полученная в результате декодирования.

Более сложные интерпретации рассмотрены далее.

## 16.2 Протокол для знака FNC1

Если первому знаку данных в сообщении предшествует знак FNC1, то он указывает, что закодированные данные в сообщении соответствуют типовому формату с использованием идентификаторов применения GS1. Передача идентификаторов символики должна быть разрешена, а знак FNC1 может отсутствовать в передаваемых данных. Присутствие знака FNC1 должно быть отмечено использованием соответствующего задаваемого значения в идентификаторе символики (приложение F).

Если знак FNC1 следует сразу же за одной буквой верхнего или нижнего регистра или двумя цифрами в начале сообщения, он сигнализирует о соответствии закодированных данных в сообщении конкретному нормативному документу (например, отраслевому или промышленному стандарту). Передача идентификаторов символики должна быть разрешена, а знак FNC1 может отсутствовать в передаваемых данных. Присутствие знака FNC1 должно быть отмечено использованием соответствующего задаваемого значения в идентификаторе символики. Начальный(ые) знак(и) данных сообщения передаются в составе закодированных данных сообщения.

В любой последующей позиции знак FNC1 выполняет функции разделителя полей, а в передаваемых данных сообщения представляется знаком набора ASCII <GS> с десятичным значением 29.

## 16.3 Протокол для интерпретации в расширенном канале

В системах с поддержкой интерпретации в расширенном канале (ECI) при каждой передаче должен использоваться префикс идентификатора символики. Все знаки ECI (FLG(n)), встречающиеся в потоке, будут передаваться в виде управляющего знака с десятичным значением 92 (или $5C_{HEX}$ в шестнадцатеричном представлении) в стандартной интерпретации представляющего знак «\» (ОБРАТНАЯ ДРОБНАЯ ЧЕРТА). Следующие «n» закодированных цифр дополняют начальными нулями до последовательности из шести цифр, которая передается в формате соответствующих знаков набора ASCII (с десятичными значениями от 48 до 57).

Прикладное программное обеспечение, распознающее конструкции вида «\nnnnnn», должны интерпретировать все последующие символы как принадлежащие ECI, определяемой данной последовательностью из шести цифр. Такая интерпретация продолжает действовать до конца закодированных данных или до обнаружения другой последовательности ECI.

Если в закодированных данных встречается байт с десятичным значением 92 (независимо от того, представляет он знак «\» или нет), то в потоке передаются два байта с этим значением. Таким образом, однократное вхождение всегда сигнализирует о начале управляющей последовательности ECI, а двукратное вхождение обозначает реальные данные.

***Пример — Закодированные данные: A\\B\C.***
***Передача: A\\\\B\\C.***

Использование соответствующего идентификатора символики гарантирует, что в данном применении может быть правильно интерпретирован управляющий знак.

## 16.4 Идентификатор символики

После идентификации структуры данных (включая использование каких-либо последовательностей ECI) декодер добавляет соответствующий идентификатор символики в виде преамбулы к передаваемым данным. Идентификатор символики необходим в том случае, если в передаваемых данных присутствуют ECI, если знак FNC1 используют в соответствии с 16.2 или если используют небуферизованное структурированное соединение (приложение F).

## 16.5 Пример передаваемых данных

В этом примере в символе Aztec Code кодируют последовательность двух знаков «¶ Ж». Знак «¶» представлен байтом с десятичным значением 182 в кодовом наборе по умолчанию Aztec Code (ECI 000003, что соответствует ИСО/МЭК 8859-1). Буква кириллицы «Ж» отсутствует в ECI 000003, но может быть пред-

ставлена в наборе знаков по ИСО/МЭК 8859-5 (соответствует ECI 000007) с тем же байтом с десятичным значением 182. Таким образом, для представления всех данных сообщения после первого знака вставляют знак перехода на ECI 000007, как описано ниже.

Символ кодирует данные сообщения <¶> <переключение на ECI 000007> <Ж> с использованием следующей последовательности знаков Aztec Code:

[B/S(1)],[182],[P/S],[FLG(1)],[«7»],[B/S(1)],[182],

которым соответствует следующий двоичный поток (повторяющий приведенную выше форму записи):

[11111(00001)],[10110110],[00000],[00000(001)],[1001],[11111(00001)],[10110110].

Декодер передает следующие байты (включая префикс идентификатора символики с задаваемым значением 3, указывающим на применение протокола ECI):

93, 122, 51, 182, 92, 48, 48, 48, 48, 48, 55, 182.

В интерпретации по умолчанию эти байты будут иметь следующее графическое представление:

]z3¶\000007¶.

Следует обратить внимание на то, что декодер обеспечивает выдачу сигнала о переключении на ECI 000007, но не интерпретацию результата.

Программное обеспечение, поддерживающее ECI в приложении-получателе, удалит управляющую последовательность \000007, а буква кириллицы «Ж» будет представлена способом, зависящим от системы (например, путем изменения шрифта в системе подготовки текста). Конечный результат будет соответствовать исходным данным сообщения «¶Ж».

**Приложение А**
**(обязательное)**

## Символы Aztec Runes

Символы Aztec Runes представляют собой серию небольших, но хорошо различимых графических объектов, пригодных для машинного считывания, спроектированных с расчетом на графическую совместимость с символами Aztec Code. В сущности они являются ядром символа компактных символов Aztec Code с различными служебными сообщениями, которые в этом случае передают восемь битов фактических данных. Таким образом, символы Aztec Runes размером 11×11 модулей включают в себя 256 графических объектов, которые обнаруживаются и считываются устройствами считывания Aztec Code.

### А.1 Описание символа


Рисунок А.1 — Типовые символы Aztec Runes

На рисунке А.1 изображены некоторые типовые символы Aztec Runes. Базовая структура символов Aztec Runes приведена на рисунке А.2. Каждый символ Aztec Rune состоит из шаблона поиска «мишень», шаблонов ориентации и данных сообщения, закодированных по алгоритму Рида-Соломона.

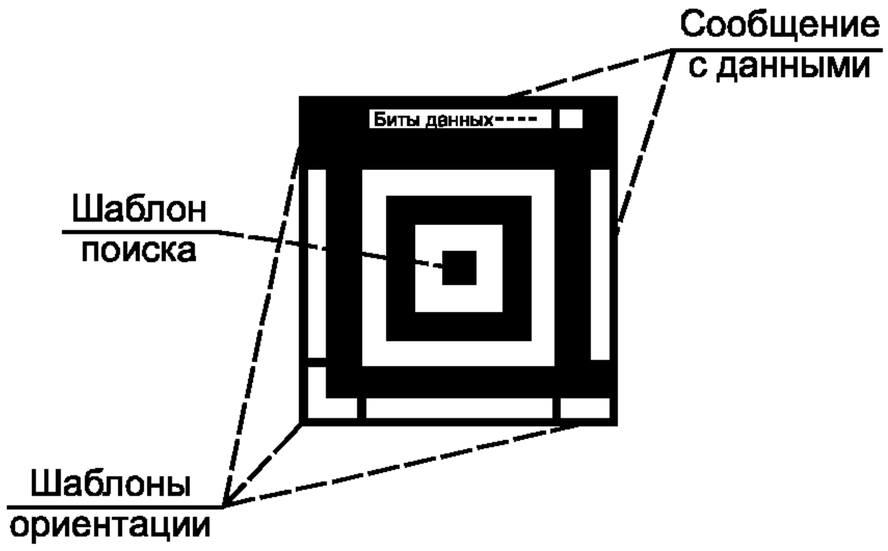


Рисунок А.2 — Структура символа Aztec Rune

Шаблоны поиска и ориентации в символе Aztec Rune совпадают с определяемыми для компактных символов Aztec Code (5.1.1). Модули с данными сообщения заполняют один слой, прилегающий к шаблону поиска (по аналогии со служебным сообщением в 5.1.1.3). Как и для символов Aztec Code, наличие окружающей свободной зоны не обязательно.

### А.2 Кодирование данных сообщения

28 битов, составляющих данные сообщения символа Aztec Rune, подразделяют на семь 4-битовых слов, два из которых содержат данные, а остальные пять являются контрольными кодовыми словами кода Рида-Соломона, вычисленными над полем GF(16), в точном соответствии с определением для служебных сообщений в компактных символах Aztec Code (7.2.3). Однако для того чтобы они отличались от обычных служебных сообщений, каждый бит инвертируется на графическом уровне, что эквивалентно выполнению операции «исключающего ИЛИ» значения каждого слова с двоичным шаблоном «1010».

Таким образом, при кодировании 8-битовое кодируемое значение разбивается на два 4-битовых слова, которые затем дополняются пятью контрольными кодовыми словами с использованием процедур для служебных сообщений компактных символов, после чего значения каждого из семи кодовых слов объединяют операцией «исключающего ИЛИ» со значением «1010» и размещают по часовой стрелке в одном слое, прилегающем к шаблону поиска. При этом темные модули представляют двоичные «1», а светлые модули — двоичные «0».

**А.3 Декодирование данных сообщения**

Рекомендуемый алгоритм декодирования Aztec Code (раздел 14) также подходит для поиска символов Aztec Runes. При этом полностью применимы все методы выявления инверсии изображения, различения символов компактного и полноразмерного форматов, определения ориентации изображения и зеркального отображения по шаблонам ориентации. Декодирование символов Aztec Runes проводят при неудачном исправлении ошибок в служебном сообщении компактного символа следующим образом:

1) каждое из семи 4-битовых слов «служебного сообщения» объединяют операцией «исключающего ИЛИ» со значением «1010»;

2) выполняют повторную попытку декодирования по алгоритму Рида-Соломона.

Декодирование по алгоритму Рида-Соломона завершено успешно, если старшие восемь битов данных в сообщении в данной форме (с инвертированием чередующихся битов по отношению к графическому представлению) содержат закодированное значение символа Aztec Rune.

**А.4 Передаваемые данные**

Значение данных, закодированное в символе Aztec Rune, передают в виде 3-разрядного десятичного числа, дополненного начальными нулями в случае необходимости.

Идентификатор символики, если он используется, должен соответствовать записи «]zC», где знак-модификатор «С» является признаком декодирования символа Aztec Rune вместо обычного символа Aztec Code.

**Приложение В**
**(обязательное)**

## Обнаружение и исправление ошибок

Кодовые слова исправления ошибок в символах Aztec Code вычисляют с использованием схемы помехоустойчивого кодирования Рида-Соломона. Размер всех кодовых слов равен В битам, а полиномиальные арифметические операции над полем GF($2^B$) проводят с использованием поразрядных арифметических операций по модулю 2 и арифметических операций по модулю Р над кодовыми словами (где Р — эквивалентное значение примитивного многочлена). Значения В и Р для служебного сообщения определяют в соответствии с 7.2.3, а для данных сообщения — в соответствии с 7.3.2.

### В.1 Построение кодовых слов исправления ошибок

Кодовые слова исправления ошибок представляют собой остаток после операций умножения многочлена, образованного из «nd» кодовых слов данных, на $x^{nc}$ с последующим длинным делением на порождающий многочлен порядка «nc». Нецелесообразно хранить списки всех возможных порождающих многочленов, используемых с данными сообщениями в Aztec Code. Вместо этого они генерируются непосредственно соответствующим программным обеспечением.

Функции языка С, представленные на рисунке В.1, обеспечивают необходимое кодирование. Функция ReedSolmon(), которая работает с «nd» значениями кодовых слов данных, хранящимися в целочисленном массиве wd[], генерирует логарифмические и антилогарифмические таблицы для поля Галуа размером «gf» с простым модулем «pp», затем использует их в функции prod() — сначала для вычисления коэффициентов порождающего многочлена порядка «nc», затем — для вычисления «nc» дополнительных контрольных кодовых слов, размещаемых после данных в wd[ ]. Функция ReedSolomon() работает не только с данными сообщениями, но и со служебными сообщениями.

### В.2 Возможность исправления ошибок

Кодовые слова исправления ошибок могут исправлять два типа ошибок — стирания (ошибочные кодовые слова в известных позициях) и ошибки (ошибочные кодовые слова в неизвестных позициях). Стирание представляет собой неотсканированный или недекодируемый знак символа, а ошибка — неверно декодированный знак символа. Число исправляемых стираний и ошибок вычисляют по следующей формуле

$$e + 2t < d - p,$$

где e — число стираний;
t — число ошибок;
d — число кодовых слов исправления ошибок;
p — число кодовых слов исправления ошибок, зарезервированных для обнаружения ошибок.

Для служебных сообщений Aztec Code p равно 0 (для обеспечения максимальной возможности продолжения декодирования).

Для данных сообщения значение p обычно равно двум. Тем не менее, если большая часть способности исправления ошибок используется для исправления стираний, вероятность наличия необнаруженных ошибок возрастает. Если ошибок меньше десяти, а число стираний составляет более половины от числа кодовых слов исправления ошибок, p увеличивается до четырех.

В противном случае символ невозможно декодировать без риска ошибочного декодирования.

```
/* «prod(x,y,log,alog,gf)» returns the product «x» times «y» */[1)]
int prod(int x, int y, int *log, int *alog, int gf) {
  if (!x || !y) return 0;
   else return alog[(log[x] + log[y]) % (gf–1)];
}
/* «ReedSolomon(wd,nd,nc,gf,pp)» takes «nd» data codeword values in */[2)]
/* wd[] and adds on «nc» check codewords, all within GF (gf) where «gf» */
/* is a power of 2 and «pp» is the value of its prime modulus polynomial */
void ReedSolomon(int *wd, int nd, int nc, int gf, int pp) {
  int i, j, k, *log, *alog, *c;
```

Рисунок В.1 — Функция языка С, генерирующая контрольные слова кода Рида-Соломона, лист 1

---

[1)] *Функция “prod(x,y,log,alog,gf)” возвращает произведение “x” и “y”.*

[2)] *Функция “ReedSolomon(wd,nd,nc,gf,pp)” получает “nd” кодовых слов данных в массиве wd[] и добавляет “nc” контрольных кодовых слов над полем GF(gf), где “gf” — степень 2, а “pp” — значение примитивного многочлена.*

```
/* allocate, then generate the log & antilog arrays: */[3)]
  log = malloc(sizeof(int) * gf);
  alog = malloc(sizeof(int) * gf);
  log[0] = 1–gf; alog[0] = 1;
  for (i = 1; i < gf; i++) {
       alog[i] = alog[i–1] * 2;
       if (alog[i] >= gf) alog[i] ^= pp;
       log[alog[i]] = i;
  }
/* allocate, then generate the generator polynomial coefficients: */[4)]
  c = malloc(sizeof(int) * (nc)+1);
  for (i=1; i<=nc; i++) c[i] = 0; c[0] = 1;
  for (i=1; i<=nc; i++) {
       c[i] = c[i–1];
       for (j=i–1; j>=1; j- -) {
          c[j] = c[j–1] ^ prod(c[j],alog[i],log,alog,gf);
       }
       c[0] = prod(c[0],alog[i],log,alog,gf);
  }
/* clear, then generate «nc» checkwords in the array wd[]: */[5)]
  for (i=nd; i<=(nd+nc); i++) wd[i] = 0;
  for (i=0; i<nd; i++) {
       k = wd[nd] ^ wd[i];
       for  (j=0;  j<nc;  j++) {
          wd[nd+j] = wd[nd+j+1] ^ prod(k,c[nc–j–1],log,alog,gf);
       }
  }
  free(c);
  free(alog);
  free(log);
}
```

Рисунок В.1, лист 2

### В.3 Метод исправления ошибок

Если общее число стираний e меньше или равно способности исправления ошибок, то активизируется схема восстановления данных. Стирания заменяются нулями. Позиция q-го неизвестного кодового слова равна $j_q$ для q = 1,2,...,e.

Создается многочлен знаков символа $C(x) = C_{n-1}x^{n-1} + C_{n-2}x^{n-2} + ... + C_1x^1 + C_0$, где n коэффициентов определяются по считанным сканером значениям знаков символа ($C_{n-1}$ — первый знак символа, а n — общее число знаков символа). Далее вычисляются значения i синдромов от $S_0$ до $S_{i-1}$, для чего вычисляется C(x) при $x = 2^k$ для k от 1 до i (i — число знаков исправления ошибок в символе). Схема генерирования значений синдромов приведена на рисунке В.2.

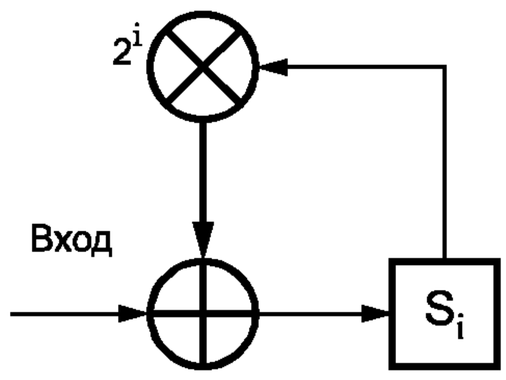


Рисунок В.2 — Схема генерирования значения синдромов

---

[3)] *Выделение памяти и построение массивов логарифмов и антилогарифмов.*
[4)] *Выделение памяти и построение коэффициентов порождающего многочлена.*
[5)] *Очистка и построение “nc” контрольных слов в массиве wd[].*

Поскольку позиции стираний известны ($j_q$ для q = 1, 2,...,e), многочлен позиций ошибок для этих известных позиций может быть вычислен по формуле

$$\Lambda(x) = (1 - xX_1)(1 - xX_2)...(1 - xX_e) = 1 + \Lambda_1 x + ... + \Lambda_e x^e,$$

где $X_q = 2^{j_q}$.

Многочлен позиций ошибок $\Lambda(x)$ может быть обновлен для включения позиций ошибок. Обновление проводят по алгоритму Берлекэмпа-Месси, обеспечивающему декодирование ошибок и стираний в кодах БЧХ в соответствии с «Theory and Practice of Error Control Codes», Richard E. Blahut (Addison Wesley, 1983). На этой стадии следует убедиться в том, что число стираний и ошибок не выходит за пределы соответствующей формулы способности исправления ошибок (В.2).

Результат решения уравнения $\Lambda(x) = 0$ дает информацию о наличии t ошибок, если t > 0; либо об их отсутствии, если t = 0. После этого вычисляют значение ошибки $Y_{j_q}$ [1)] для позиции $j_q$, где q = 1,...,e + t. Для вычисления значений ошибок требуется дополнительный $\Omega$-многочлен, определяемый по формуле

$$\Omega(x) = 1 + (s_1 + \Lambda_1)x + (s_2 + \Lambda_1 s_1 + \Lambda_2)x^2 + ... + (s_\eta + \Lambda_1 s_{\eta-1} + \Lambda_2 s_{\eta-2}{}^{2)} + ... + \Lambda_{\eta-1} s_1 + \Lambda_\eta)x^\eta,$$

где $\eta = e + t$.

Таким образом, значение ошибки в позиции $j_q$ вычисляют по формуле

$$Y_{j_q} = \Omega(X_q - 1) / \left( X_q \prod_{i=1,\, i \neq q}^{\eta} (1 - X_i / X_q) \right).$$

После успешного вычисления значения ошибок их двоичные дополнения суммируют с соответствующими позициями ошибочных кодовых слов.

---

[1)] *В оригинале ИСО/МЭК 24778 ошибочно указано* $Y_{jP}$.

[2)] *В оригинале ИСО/МЭК 24778 ошибочно указано* $\Lambda 2 s_{\eta-2}$.

**Приложение С**
**(обязательное)**

## Топологический алгоритм обнаружения шаблона поиска «мишень»

Данный метод сканирования применяют для выделения топологии шаблона поиска «мишень» в двоичном изображении. Он основан на измерении степени «изолированности» каждого пикселя от белой границы изображения, т.е. числа черных/белых и белых/черных переходов, пересекаемых на пути к каждой границе. Очевидно, что центр шаблона поиска «мишень» является областью с высокой изолированностью.

Полностью формальное определение изолированности пикселя для произвольной геометрии (в худшем случае может оказаться, что центр длинной спирали соединяется с краем изображения) требует многонаправленного сканирования, которое может оказаться достаточно длительным. В данном приложении описан сокращенный однопроходный метод, приемлемый для выделения геометрии шаблона поиска «мишень».

В качестве примера рассмотрена последовательность действий сканирования части изображения, представленного в двоичном виде размером 24×21 пикселей (рисунок С.1). Суть алгоритма заключается в создании вторичного изображения с теми же размерами, значение которого в каждой позиции пикселя обозначает его изолированность от краев. Это также можно сделать последовательным перебором строк изображения сверху вниз с использованием одномерного целочисленного массива, длина которого совпадает с шириной изображения. Тем не менее, на рисунках С.1 — С.8 значения изолированности будут выводиться в позициях пикселей на двумерном массиве.

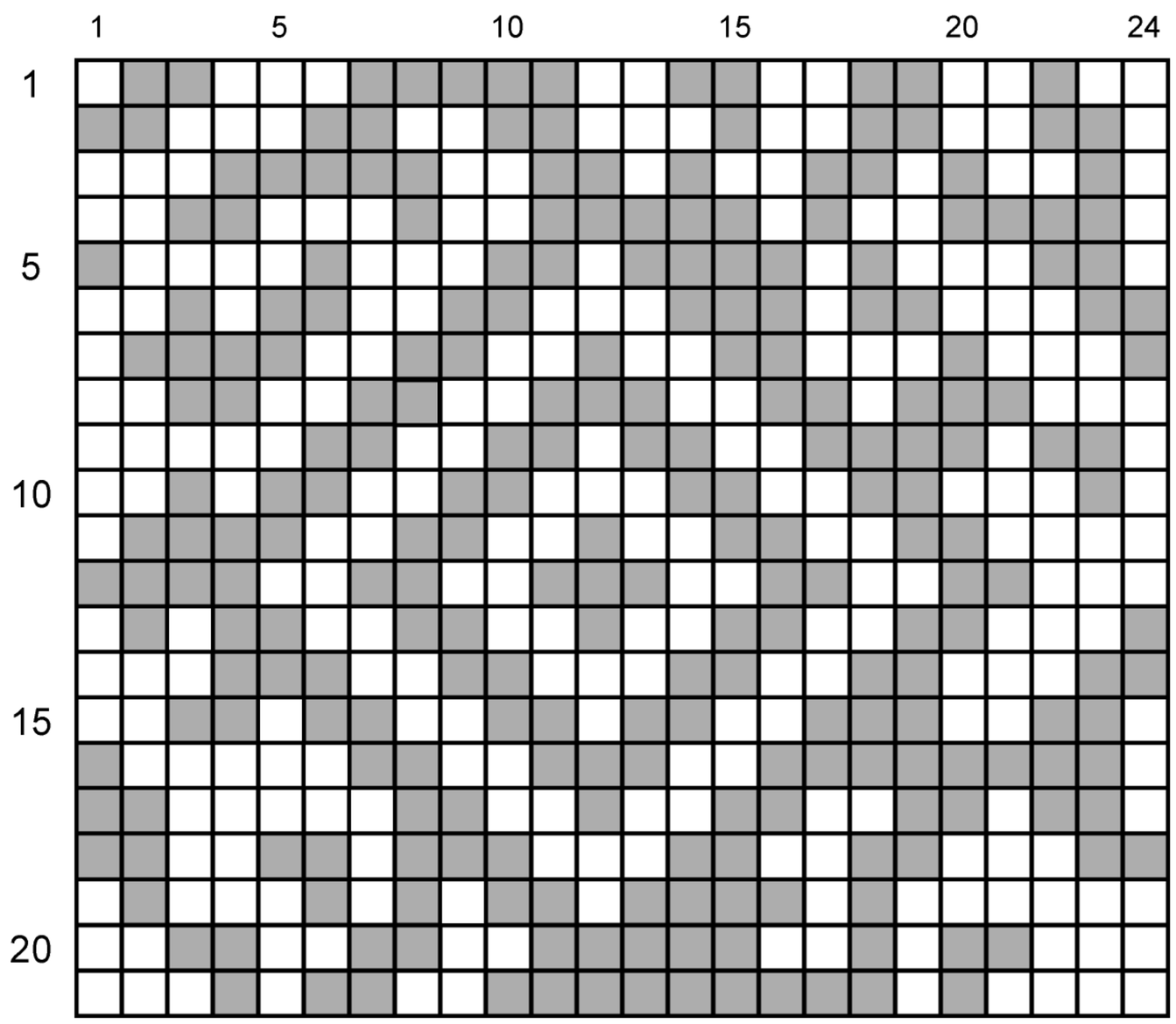


Рисунок С.1 — Изображение в двоичном виде, в котором осуществляют поиск

Основное топологическое правило заключается в том, что значения степени изолированности всегда должны быть четными для светлых пикселей и нечетными для темных пикселей. На начальной стадии целочисленный массив заполняют значениями «0» для светлых пикселей и «1» — для темных пикселей в верхней строке изображения (рисунок С.2).

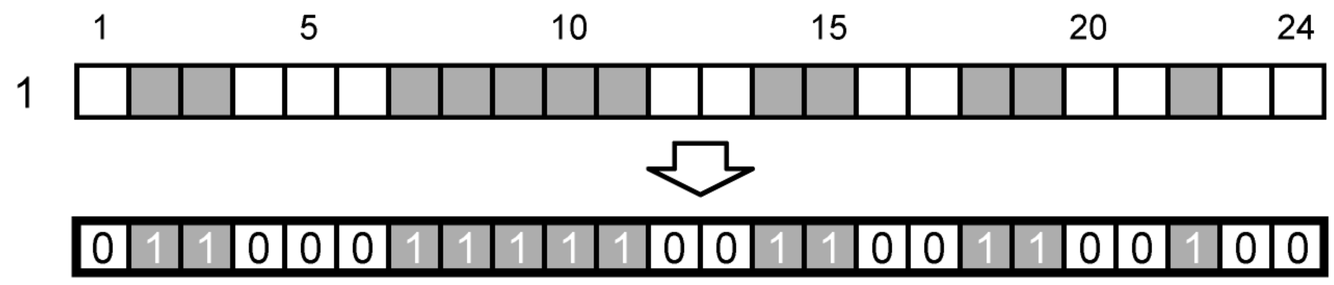


Рисунок С.2 — Исходное содержимое целочисленного массива

Затем каждую из последующих строк сканируют в три этапа. Сначала для новой строки в целочисленном массиве обновляют все элементы, у которых изменился цвет пикселя. Например, во второй строке рисунка С.3 несколько «0» превращаются в «1», а несколько «1» — в «2». Затем крайнее левое значение в массиве преобразуют в «0» или «1», после чего элементы массива перебираются слева направо, и все элементы, более чем на единицу превышающие значение своего соседа слева, последовательно уменьшаются на два до выполнения данного соотношения. Например, в средней изолированной строке значение «2» в столбце 14 уменьшилось до «0». Наконец, крайний правый элемент массива преобразуют в «0» или «1», после чего массив перебирают справа налево, и все элементы, более чем на единицу превышающие рядом расположенные значения справа, уменьшают на величину, кратную двум, чтобы их значение стало меньше либо равно значению предыдущего пикселя плюс один. Так, в нижней изолированной строке значение «2» в столбце 3 тоже было уменьшено до «0».

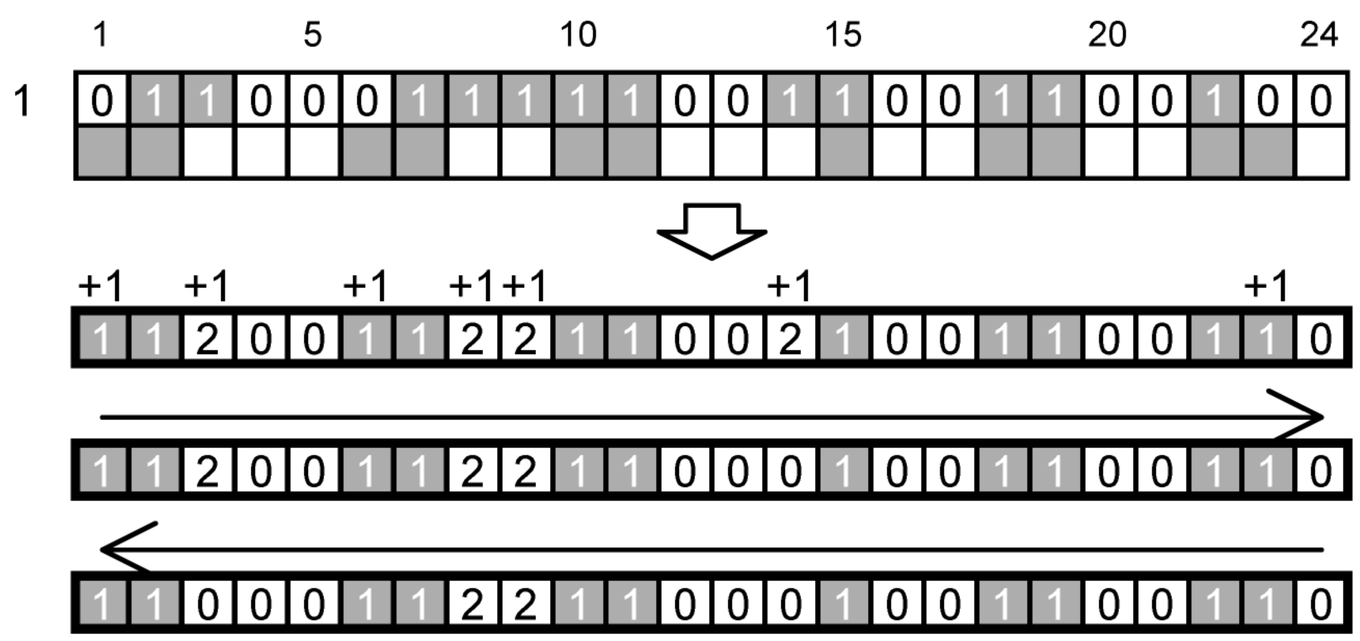


Рисунок С.3 — Последовательность и результаты сканирования строки 2

На рисунках С.4 — С.6 показаны построчные промежуточные результаты сканирования.

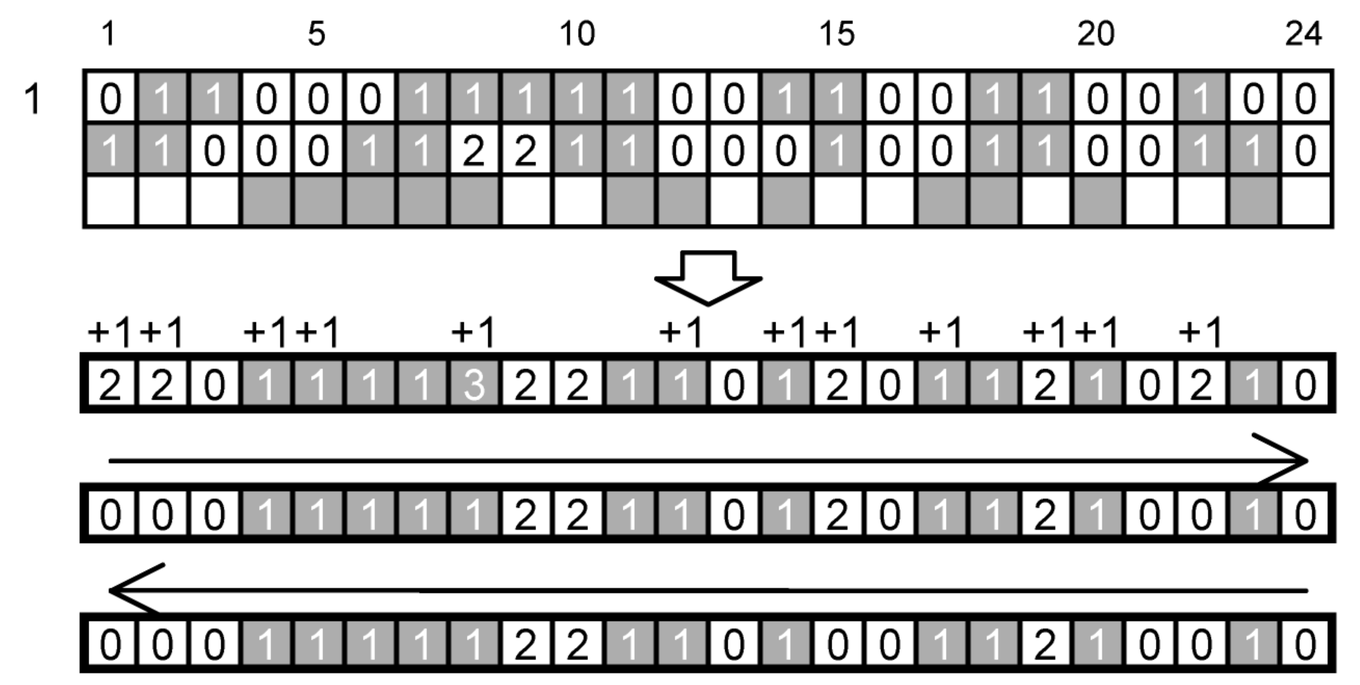


Рисунок С.4 — Последовательность и результаты сканирования строки 3

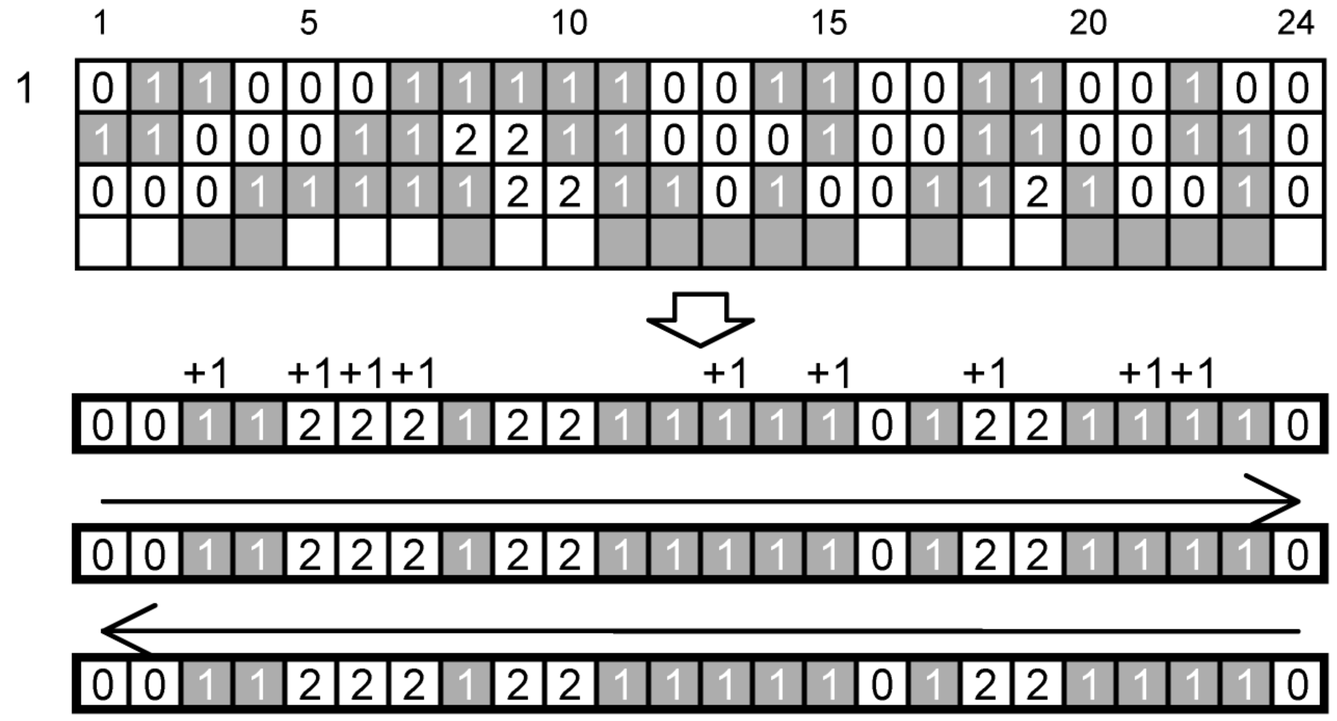


Рисунок С.5 — Последовательность и результаты сканирования строки 4

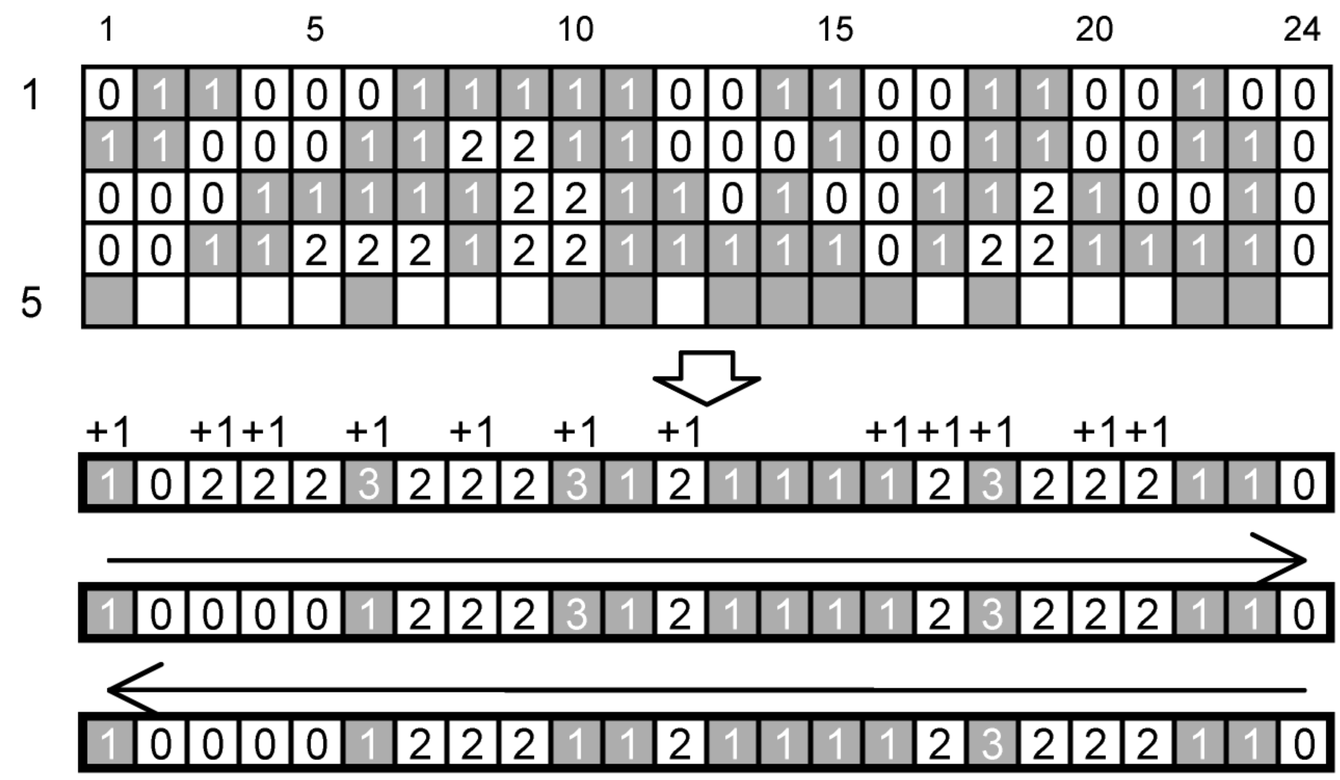


Рисунок С.6 — Последовательность и результаты сканирования строки 5

В процессе обработки строки 3 значение «3» ненадолго появляется в столбце восемь, но затем сбрасывается в ходе перебора слева направо. Следует обратить внимание на то, что начальное значение «2» обнуляется в строке 3.

В процессе обработки строки 5 значение «3» ненадолго появляется в столбце 6, но затем сбрасывается после сброса значения «2», расположенного слева от него. С другой стороны, значение «3» в столбце 18 сохраняется.

На рисунке С.7 показано содержимое целочисленного массива после шести следующих итераций. Топология шаблона поиска «мишень» сформировалась в центре в виде области с относительно высокими значениями изолированности. Хотя наибольшие значения степени изолированности определяют позицию центра, самым надежным показателем «высоты» шаблона поиска «мишень» является число этапов постоянного восхождения к пику и последующего спуска с него. Конечный автомат, отслеживающий последовательные целочисленные значения в ходе завершающего перебора строки справа налево, может выявлять потенциальные центры шаблона поиска «мишень» и измерять их высоту.

| | 1 | | | | 5 | | | | | 10 | | | | | 15 | | | | | 20 | | | | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 2 | 1 | 0 | 0 | 1 | 0 |
| | 0 | 0 | 1 | 1 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 0 |
| 5 | 1 | 0 | 0 | 0 | 0 | 1 | 2 | 2 | 2 | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 2 | 3 | 2 | 2 | 2 | 1 | 1 | 0 |
| | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 1 | 1 | 1 | 2 | 3 | 3 | 2 | 2 | 2 | 1 | 1 |
| | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 3 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 3 | 2 | 2 | 2 | 1 |
| | 0 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 3 | 3 | 3 | 2 | 2 | 1 | 1 | 2 | 1 | 1 | 1 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 0 |
| 10 | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 1 | 0 |
| | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 4 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 0 |

Рисунок С.7 — Результаты сканирования после строки 11

На рисунке С.8 показано содержимое целочисленного массива после шести следующих итераций. Пик (центр шаблона поиска «мишень») расположен на пересечении столбцов 11–13 со строками 11–13, хотя в других частях изображения обнаруживаются меньшие локальные «возвышения» (например, возле столбца 20 в строке 7). Не исключено, что в результате сканирования полного изображения могут быть обнаружены центры нескольких потенциальных шаблонов поиска «мишень», и для их идентификации как шаблона поиска Aztec Code потребуется выполнить дальнейший двумерный графический анализ.

| | 1 | | | | 5 | | | | | 10 | | | | | 15 | | | | | 20 | | | | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 2 | 1 | 0 | 0 | 1 | 0 |
| | 0 | 0 | 1 | 1 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 2 | 2 | 1 | 1 | 1 | 1 | 0 |
| 5 | 1 | 0 | 0 | 0 | 0 | 1 | 2 | 2 | 2 | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 2 | 3 | 2 | 2 | 2 | 1 | 1 | 0 |
| | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 1 | 1 | 1 | 2 | 3 | 3 | 2 | 2 | 2 | 1 | 1 |
| | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 3 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 3 | 2 | 2 | 2 | 1 |
| | 0 | 0 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 3 | 3 | 3 | 2 | 2 | 1 | 1 | 2 | 1 | 1 | 1 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 1 | 1 | 2 | 1 | 1 | 0 |
| 10 | 0 | 0 | 1 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 2 | 2 | 2 | 1 | 0 |
| | 0 | 1 | 1 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 4 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 0 |
| | 1 | 1 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 5 | 4 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 0 | 0 | 0 |
| | 0 | 1 | 2 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 4 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 1 |
| | 0 | 0 | 0 | 1 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |
| 15 | 0 | 0 | 1 | 1 | 2 | 1 | 1 | 2 | 2 | 3 | 3 | 4 | 3 | 3 | 2 | 2 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 3 | 3 | 3 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 2 | 3 | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 1 | 1 | 0 |

Рисунок С.8 — Результаты сканирования после строки 17

Рисунок С.8 также иллюстрирует одну из причин появления наименования символики — «Aztec Code». Квадратный центральный шаблон поиска «мишень» символа Aztec Code выступает на фоне изображения (в числовом виде) как древние ацтекские пирамиды возвышались над окружающей растительностью[1)]. Данный алгоритм также может применяться для обнаружения структуры шаблона поиска «мишень» ряда других матричных символов, хотя для этих символов применение данного метода поиска не рекомендуется.

---

[1)] *В оригинале ИСО/МЭК 24778 приведено данное описание происхождения наименования символики штрихового кода «Aztec Code».*

**Приложение D**
**(обязательное)**

## Алгоритм линейного выращивания кристалла

Измеренные значения вертикального и горизонтального шагов сетки, полученные на основе анализа шаблона поиска «мишень», задают ориентировочную оценку вертикального и горизонтального размеров X. По направлению вертикальных и горизонтальных границ шаблона поиска «мишень» проводят начальную оценку направлений вертикальных и горизонтальных осей.

Для каждого из четырех сегментов решетки привязки, начинающихся в центре шаблона поиска «мишень», строят позиции центральных точек модулей сетки, для чего используют метод «выращивания кристалла» от центра шаблона поиска «мишень» к краям символа.

Сначала строят прогнозируемый центр следующего модуля на расстоянии одного соответствующего размера X от центра шаблона поиска «мишень» вдоль соответствующей оси. Далее строят четыре пробные линии длиной 1X в направлениях, соответствующих румбам компаса, от прогнозируемого центра модуля параллельно осям. Далее регулируют центр по вертикали и горизонтали с использованием следующего метода для каждой оси.

Вдоль двух линий, параллельных оси:

a) если ни одна из линий не пересекает границу цвета, соответствующего номинальному цвету модуля (например, переход от темного цвета к светлому для темного модуля), положение центра вдоль этой оси не изменяется;

b) если только одна линия пересекает соответствующую границу цвета, центр смещается на 5 % расстояния от его исходной позиции до точки, находящейся в 0,5 X от границы;

c) если обе линии пересекают соответствующую границу цвета, центр смещается на 5 % расстояния от исходной позиции до средней точки между двумя границами.

После вычисления позиций всех 16 модулей в подсекции решетки привязки происходит повторное вычисление направлений осей и размера X с использованием полученных центров модулей, после которого алгоритм переходит к следующей подсекции с использованием ранее вычисленных позиций.

Начиная с точки, положение которой было откорректировано, новую точку строят на расстоянии 1X вдоль оси, а ее положение регулируют описанным выше методом. Процесс повторяется для построения центральных точек всех модулей.

Для модулей сегментов решетки привязки, не проходящих через шаблон поиска «мишень», позиции центров строят от модуля, пересекающего соответствующий изначально построенный сегмент (один из четырех).

Сначала пересматривают направление оси и размер X вдоль оси, которая выровнена по точкам центров ближайшей построенной подсекции решетки привязки. Затем, начиная с предполагаемого центра модуля пересечения, «выращивание кристалла» повторяют в обоих направлениях вдоль новой решетки привязки до внешних границ символа.

**Приложение Е**
**(обязательное)**

# Оценка повреждений фиксированных шаблонов

## Е.1 Свойства, подлежащие оценке

К фиксированным шаблонам в компактных символах Aztec Code относят шаблон поиска «мишень», включающий в себя два темных концентрических квадрата, и 3-модульные шаблоны ориентации, расположенные в каждом углу на внешней стороне шаблона поиска «мишень». Данные фиксированные шаблоны расположены в центре символа и содержат 93 модуля. В полноразмерных символах Aztec Code к фиксированным шаблонам относят шаблон поиска «мишень», включающий в себя три темных концентрических квадрата и шаблоны ориентации, расположенные в каждом углу на внешней стороне шаблона поиска «мишень». Данные фиксированные шаблоны расположены в центре символа и содержат 181 модуль. Кроме того, в полноразмерном символе к фиксированному шаблону относят также решетку привязки, проходящую по областям данным со смещением на 16 модулей.

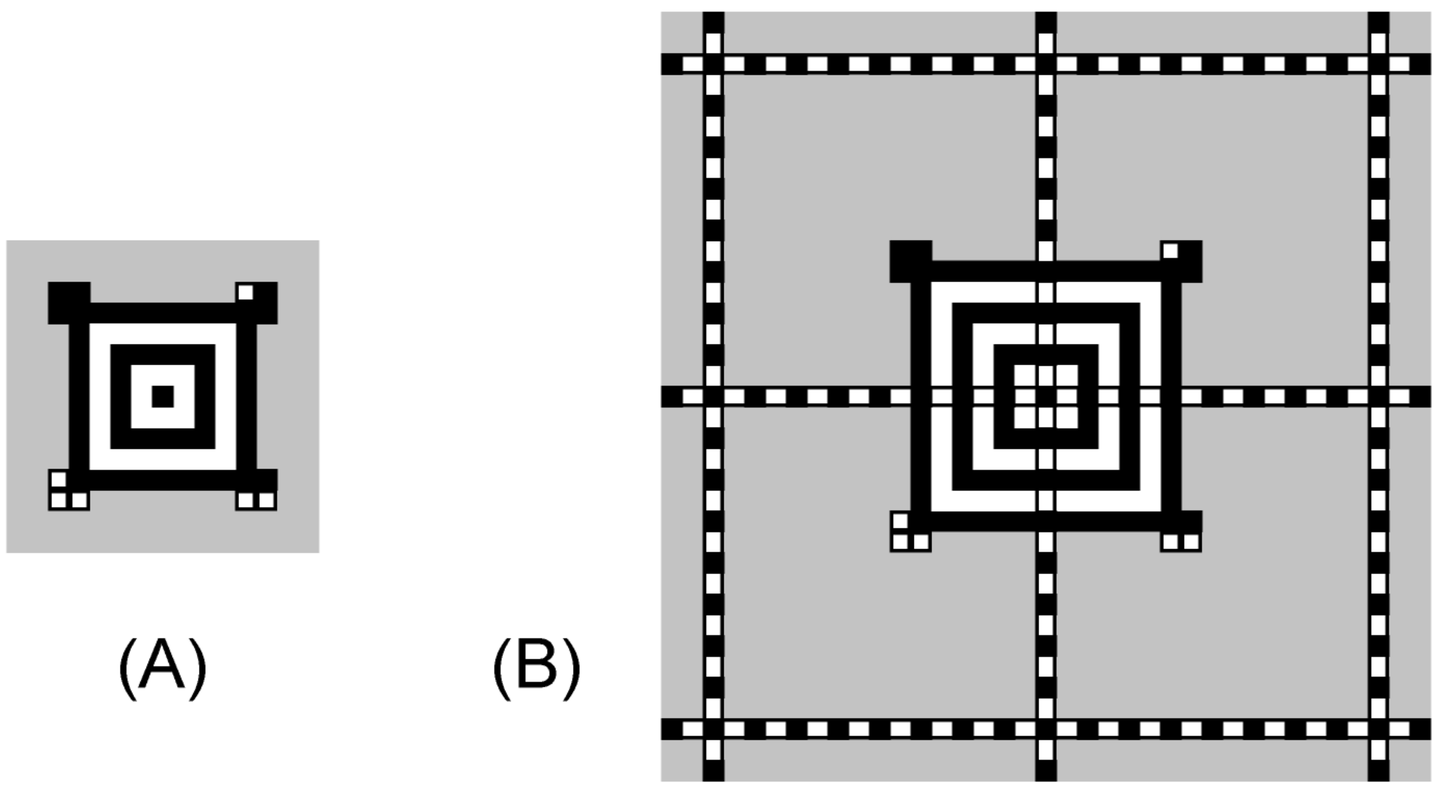


Рисунок Е.1 — Фиксированные шаблоны в компактном (А)
и полноразмерном (В) символах Aztec Code

В обоих случаях шаблон поиска «мишень» и шаблоны ориентации совместно образуют «сегмент А». Каждый из светлых и темных концентрических квадратов шаблона поиска «мишень», а также ее центральный модуль, рассматривают как отдельные подсегменты, при этом шаблоны ориентации не являются частью какого-либо подсегмента.

В полноразмерных символах решетку привязки (включая модули, находящиеся внутри шаблона поиска «мишень») разбивают на отдельные вертикальные и горизонтальные линии (с чередованием темных и светлых модулей), каждую из которых рассматривают как «сегмент В». Модули в точках пересечения принадлежат каждой из линий. На рисунке Е.1 (В) приведены шесть таких линий (или сегментов).

## Е.2 Критерии оценки

Оценку проводят на основании числа ошибочных (имеющих неверный цвет) модулей в каждом сегменте или подсегменте по следующим правилам.

Для оценки сегмента А (шаблон поиска «мишень» и шаблоны ориентации) установлены следующие классы:

A (4), если 0 (ноль) ошибочных модулей;

B (3), если 1 ошибочный модуль;

C (2), если 2 ошибочных модуля, находящихся в одном подсегменте;

D (1), если 3 ошибочных модуля, находящихся в одном подсегменте;

F (0), если 4 и более ошибочных модулей или ошибки встречаются более чем в одном подсегменте.

Для оценки каждого сегмента В (в пределах решетки привязки) установлены следующие классы:

A (4), если 0 (ноль) ошибочных модулей;

B (3), если 7 % и менее ошибочных модулей;

C (2), если 11 % и менее ошибочных модулей;

D (1), если 14 % и менее ошибочных модулей;

F (0)[1)].

Общую оценку повреждений фиксированных шаблонов определяют по наименьшему классу по всем сегментам.

---

*[1)] В оригинале ИСО/МЭК 24778 пояснение к классу F(0) отсутствует. На практике этот класс соответствует браку.*

**Приложение F**
**(обязательное)**

## Идентификаторы символики

Стандартная процедура передачи информации о символике считанного символа, а также вариантов обработки для настройки декодера и другие особенности, встречающиеся в символе, приведены в ИСО/МЭК 15424.

Идентификатор символики Aztec Code имеет вид

] z m,

где ] — знак флага идентификатора символики (знак ASCII с десятичным значением 93);
z — знак кода идентификатора символики для Aztec Code (знак ASCII с десятичным значением 122);
m — знак-модификатор с одним из значений, установленных в таблице F.1.

Таблица F.1 — Значения знаков-модификаторов, установленных для Aztec Code

| Значение | Версия обработки |
|---|---|
| 0 | Без специальных режимов |
| 1 | FNC1 предшествует первому знаку данных сообщения |
| 2 | FNC1 следует за начальной буквой или парой цифр |
| 3 | Реализован протокол ECI |
| 4 | FNC1 предшествует первому знаку данных сообщения; реализован протокол ECI |
| 5 | FNC1 следует за начальной буквой или парой цифр; реализован протокол ECI |
| 6 | Включен заголовок структурированного соединения |
| 7 | Включен заголовок структурированного соединения; FNC1 предшествует первому знаку данных сообщения |
| 8 | Включен заголовок структурированного соединения; FNC1 следует за начальной буквой или парой цифр |
| 9 | Включен заголовок структурированного соединения; реализован протокол ECI |
| A | Включен заголовок структурированного соединения; FNC1 предшествует первому знаку данных сообщения, реализован протокол ECI |
| B | Включен заголовок структурированного соединения; FNC1 следует за начальной буквой или парой цифр, реализован протокол ECI |
| C | Декодирован символ Aztec Rune |
| (Допустимые значения «m»: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C) | |

Устройство считывания Aztec Code может поддерживать альтернативный режим, в котором символы с знаком FNC1 в начале данных сообщения и при отсутствии структурированного соединения или протокола ECI дают команду устройству на проведение эмуляции данных сообщения как представленного в символике GS1-128 путем вывода идентификатора символики «]C1».

**Приложение G**
**(справочное)**

# Пример кодирования символа Aztec Code

В настоящем приложении приведена последовательность кодирования данных сообщения «Code 2D!» в символ Aztec Code.

## G.1 Создание двоичного потока данных сообщения

Входные данные сообщения последовательно обрабатывают по одному знаку или байту и кодируют в соответствии с таблицей 2. Последовательность кодирования данных сообщения «Code 2D!» приведена в таблице G.1.

Т а б л и ц а G.1 — Кодирование данных сообщения «Code 2D!»

| Знаки входных данных сообщения | Десятичное значение (по ASCII) | Кодовый набор | Значение в кодовом наборе | Значение в двоичном потоке |
|---|---|---|---|---|
| C | (67) | Upper | 4 | 00100 |
| [Lower Latch] | | Upper | 28 | 11100 |
| o | (111) | Lower | 16 | 10000 |
| d | (100) | Lower | 5 | 00101 |
| e | (101) | Lower | 6 | 00110 |
| [Digit Latch] | | Lower | 30 | 11110 |
| Пробел | (32) | Digit | 1 | 0001 |
| 2 | (50) | Digit | 4 | 0100 |
| [Upper Shift] | | Digit | 15 | 1111 |
| D | (68) | Upper | 5 | 00101 |
| [Punctuation Shift] | | Digit | 0 | 0000 |
| ! | (33) | Punctuation | 6 | 00110 |

Кодирование начинают в кодовом наборе Upper. Поскольку начальная буква «C» принадлежит кодовому набору Upper, ее кодируют напрямую значением 4 (представление в двоичном виде 00100). Двоичные значения всех элементов во всех кодовых наборах, за исключением кодового набора Digit, состоят из 5 битов. В кодовом наборе Digit они состоят из 4 битов.

Следующий знак данных «o» и несколько последующих знаков данных принадлежат к кодовому набору Lower, поэтому в двоичный поток вставляют знак фиксации Lower Latch (значение 28, представление в двоичном виде 11100), затем знаки «o» (значение 16, представление в двоичном виде 10000), «d» (значение 5, представление в двоичном виде 00101) и «e» (значение 6, представление в двоичном виде 00110).

Следующий знак «пробел» присутствует в нескольких кодовых наборах. Поскольку следующий знак «2» принадлежит к кодовому набору Digit, более эффективно сначала вставить знак фиксации Digit Latch (значение 30, представление в двоичном виде 11110), а затем закодировать пробел (значение 1, представление в двоичном виде 0001) и следующий за ним знак «2» (значение 4, представление в двоичном виде 0100), состоящие из четырех битов каждый.

Далее следует изолированный знак «D» кодового набора Upper, за которым следует изолированный (и последний) знак «!» кодового набора Punctuation. Такую комбинацию наиболее эффективно кодируют 4-битовыми знаками регистра и кодовом набором Digit. Сначала вставляют знак Upper Shift (значение 15, представление в двоичном виде 1111), за которым следует знак «D» в кодовом наборе Upper (значение 5, представление в двоичном виде 00101). Кодирование данных сообщения автоматически возвращается к кодовому набору Digit. Далее вставляют знак Punctuation Shift (значение 0, представление в двоичном виде 0000) и знак «!» (значение 6, представление в двоичном виде 00110) из кодового набора Punctuation.

**G.2 Выбор формата и размера символа**

Весь двоичный поток данных сообщения состоит из 56 битов. Для достижения корректирующей избыточности в E процентов с C дополнительными знаками число битов данных в сообщении умножают на 100/(100 – E), после чего увеличивают на C×100/(100 – E) знаков. По умолчанию минимальный уровень исправления ошибок составляет 23 % при трех дополнительных знаках. Таким образом, информационная емкость символа для кодирования «Code 2D!» должна быть

$C_W$[1)] > 56 × 100/(100 – 23) битов + 3 × 100/(100 – 23) знаков = 73 бита + 4 знака.

Указанные данные сообщения помещают в однослойном компактном символе Aztec Code (таблица 1). Размер символа составляет 15×15 модулей, а размер кодового слова равен 6 битам.

**G.3 Создание потока кодовых слов данных**

Двоичный поток данных сообщения разбивают на 6-битовые кодовые слова. При этом процедура кодирования особенно следит за тем, чтобы эти слова не состояли из одних «0» или одних «1». Полученная последовательность кодовых слов приведена в таблице G.2.

Т а б л и ц а G.2 — Формирование потока кодовых слов данных

| 001001 | 110010 | 00000(1) | 101001 | 101111 | 000010 | 100111 | 100101 | 00000(1) | 011011 |
|---|---|---|---|---|---|---|---|---|---|
| 9 | 50 | 1 | 41 | 47 | 2 | 39 | 37 | 1 | 27 |

Следует обратить внимание на то, что две «1» в круглых скобках представляют собой фиктивные биты, вставленные для того, чтобы значение слова данных отличалось от 0 (независимо от того, равен ли следующий фактический бит данных «1» или нет), а две подчеркнутых «1» в конце добавлены для заполнения последнего кодового слова. В итоге для кодирования потребуются десять 6-битовых кодовых слов.

**G.4 Добавление контрольных кодовых слов кода Рида-Соломона**

Арифметические действия для вычисления контрольных кодовых слов кода Рида-Соломона для поля данных в однослойном символе Aztec Code проводят над полем GF(64), как указано в таблице 3, при этом их результаты не могут быть проверены в обычной десятичной арифметике. Вычисления осуществляют следующим образом:

10 кодовых слов данных определяют коэффициенты многочлена

$$9x^9 + 50x^8 + x^7 + 41x^6 + 47x^5 + 2x^4 + 39x^3 + 37x^2 + x + 27,$$

который умножают на $x^7$, а затем делят на порождающий многочлен $(x – 2^1)...(x – 2^7)$, равный

$$x^7 + 59x^6 + 5x^5 + 25x^4 + 26x^3 + 17x^2 + 29x + 28.$$

Остаток от деления представляет собой многочлен

$$38x^6 + 50x^5 + 8x^4 + 16x^3 + 10x^2 + 20x + 40.$$

Его коэффициенты определяют 7 контрольных кодовых слов, которые добавляют к 10 исходным кодовым словам данных.

**G.5 Построение служебного сообщения**

В этом компактном символе два бита, обозначающие единственный слой данных, кодируют значением (1 – 1 = 0) «00», а шесть битов, обозначающих длину данных сообщения (10 кодовых слов данных), кодируют значением (10 – 1 = 9) «001001».

Арифметические действия для вычисления контрольных кодовых слов кода Рида-Соломона для служебного сообщения в символе Aztec Code проводят над полем GF(16) (пункт 7.2.3), при этом их результаты не могут быть проверены в обычной десятичной арифметике. Вычисления осуществляют следующим образом:

8 битов служебного сообщения разбивают на два 4-битовых слова

0000 1001.

Их значения определяют коэффициенты многочлена

$$0x + 9,$$

который умножают на $x^5$, а затем делят на порождающий многочлен $(x – 2^1)...(x – 2^5)$, равный

$$x^5 + 11x^4 + 4x^3 + 6x^2 + 2x + 1.$$

Остаток от деления представляет собой многочлен

$$12x^4 + 2x^3 + 3x^2 + x + 9.$$

---

[1)] *В оригинале ИСО/МЭК 24778 ошибочно указано «CW».*

Его коэффициенты определяют 5 контрольных кодовых слов, которые добавляют к двум исходным кодовым словам данных.

**G.6 Графическое построение символа**

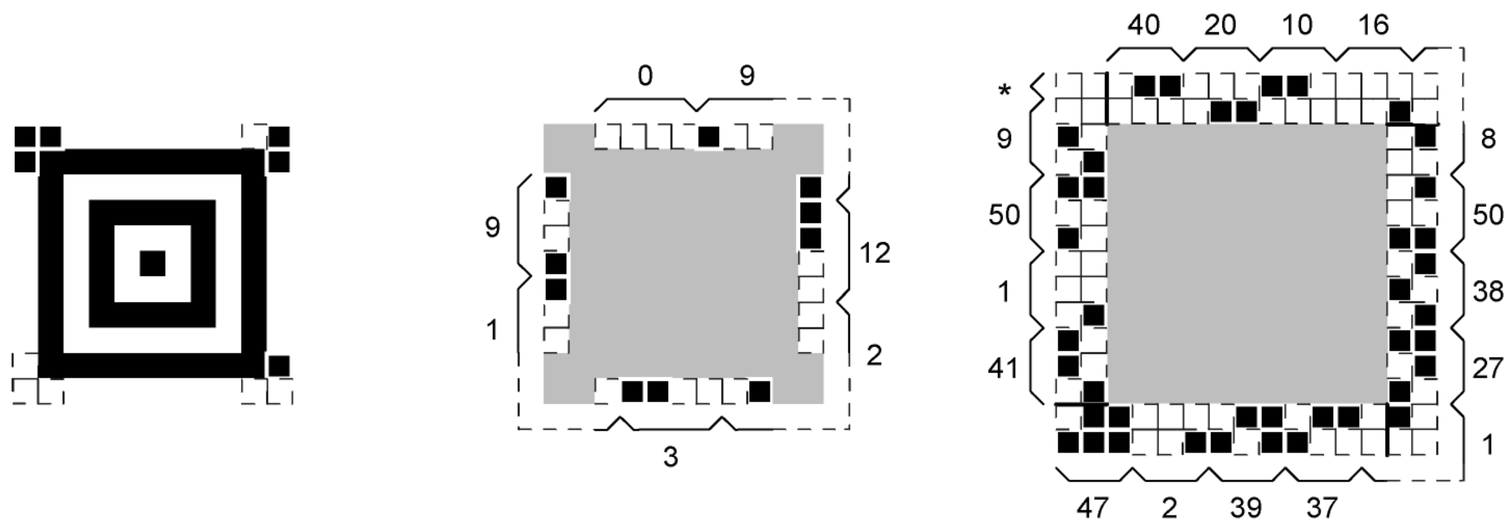


Рисунок G.1 — Последовательность построения символа Aztec Code

Итоговое графическое представление символа строят в три этапа (рисунок G.1):

1) построение начинают с фиксированных шаблонов компактного символа Aztec Code;

2) сначала биты служебного сообщения размещают по часовой стрелке вокруг шаблона поиска (двоичные «1» представлены темными модулями);

3) затем добавляют слои данных, размещаемые по часовой стрелке вокруг ядра символа, но с обратным порядком следования кодовых слов (звездочкой помечен неиспользованный последний блок типа домино, у которого оба модуля светлые).

Визуально кажется, что итоговое графическое изображение (рисунок G.2) содержит беспорядочное множество отдельных модулей, представляющих биты, и структур (выделяется только шаблон поиска). Однако это графическое представления является символом Aztec Code, кодирующим данные сообщения «Code 2D!», который может быть прочитан устройством считывания. Так как наличие свободной зоны не обязательно, вблизи от границ символа может быть размещена дополнительная необязательная интерпретация для визуального чтения.

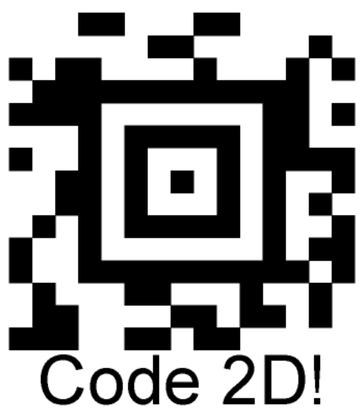


Рисунок G.2 — Символ Aztec Code, кодирующий данные сообщения «Code 2D!»

**Приложение Н**
**(справочное)**

## Обеспечение минимального размера символа

В настоящем приложении приведен метод нахождения последовательности знаков регистра и фиксации, обеспечивающих кодирование любых заданных данных сообщения в строку битов минимальной длины для представления в виде символа Aztec Code. Входную строку знаков обрабатывают последовательно по одному знаку с сохранением информации о возможных вариантах кодирования в формате усеченного дерева или решетки. В итоге определяют последовательность, требующую минимального числа битов.

Перед обработкой каждого знака последовательность кодирования может находиться в относительно небольшом числе возможных «состояний» (таблица 2)[1)]:

1) Upper («U»).
2) Lower («L»).
3) Mixed («M»).
4) Punctuation («P»).
5) Digit («D»).
6) Binary («B»).

Необходимо создать несколько таблиц с константами. Для начала потребуется матрица 6×6 LatLen[From][To], которая содержит размер знаков фиксации, т.е. числа битов, необходимых для перехода из одного состояния в другое с помощью знака фиксации.

Размер знака фиксации (LatLen) для перехода из одного состояния в другое, в битах:

| Из какого состояния (From) | В какое состояние (To) | U | L | M | P | D | B |
|---|---|---|---|---|---|---|---|
| U | | 0 | 5 | 5 | 10 | 5 | 10 |
| L | | 10 | 0 | 5 | 10 | 5 | 10 |
| M | | 5 | 5 | 0 | 5 | 10 | 10 |
| P | | 5 | 10 | 10 | 0 | 10 | 15 |
| D | | 4 | 9 | 9 | 14 | 0 | 14 |
| B | | 0* | 0* | 0* | 0* | 0* | 0* |

* В данном контексте знак регистра Binary Shift выполняет функцию фиксации, потому что кодирование может оставаться в том же состоянии на протяжении многих знаков. Возврат из состояния Binary не требует дополнительных битов, но приводит только к тому состоянию, из которого произошел переход в состояние Binary.

Матрица 5×5 знаков регистра ShftLen[From][To] содержит размер регистровых переходов, т.е. числа битов, необходимых для переключения из одного состояния в другое для одного знака. Большинство элементов матрицы будет пустым («E»), что указывает на невозможность прямого переключения регистра. В этом и в последующих вариантах использования для «E» присвоено заведомо большое значение (например, 10 000).

Размер знака регистра (ShftLen), в битах:

| Из какого состояния (From) | В какое состояние (To) | U | L | M | P | D |
|---|---|---|---|---|---|---|
| U | | E | E | E | 5 | E |
| L | | 5 | E | E | 5 | E |
| M | | E | E | E | 5 | E |
| P | | E | E | E | E | E |
| D | | 4 | E | E | 4 | E |

В дополнительной матрице 6×1 CharSiz[State] содержатся размеры знака, т.е. число битов, необходимых для представления одного знака данных в каждом состоянии:

Размер знака (CharSiz), в битах:

| В каком состоянии (In) | U | L | M | P | D | B |
|---|---|---|---|---|---|---|
| | 5 | 5 | 5 | 5 | 4 | 8 |

[1)] *В оригинале ИСО/МЭК 24778 ошибочно указано «таблица 1».*

Также необходимо определить несколько именованных переменных. Два целочисленных массива с шестью элементами CurLen[State] и NxtLen[State] содержат накапливаемые длины в битах закодированных строк, завершающихся в состоянии «State». Два соответствующих строковых массива CurSeq[State] и NxtSeq[State] содержат информацию о закодированных строках, представленных соответствующим значением CurLen и NxtLen. (Одна из возможных схем: знаки фиксации обозначают прописными буквами, за ними следуют числа, определяющее число кодов знаков, которые были напрямую закодированы в этом кодовом наборе, между ними вставляют отдельные строчные буквы для переключения регистра на один знак). В дополнительной переменной BackTo хранится информация о состоянии, из которого произошел переход, вызванный знаком регистра Binary Shift.

В исходном состоянии элемент CurLen[U] равен нулю, так как состояние Upper является исходным состоянием по умолчанию, а все остальные элементы CurLen равны «E» — признаку пустого значения (например, такими большими значениями, как 10000). Строки CurSeq изначально находятся в неопределенном состоянии (null), а переменной BackTo задано значение U.

Далее входную строку последовательно обрабатывают по одному знаку, при этом используют следующий алгоритм:

1) Сначала следует проверить, нельзя ли уменьшить какие-либо элементы из CurLen за счет фиксации из другого состояния. Для этого перебирают все возможные комбинации состояний X и Y:

ЕСЛИ (CurLen[X] + LatLen[X][Y]) МЕНЬШЕ CurLen[Y], ТО:

(a) уменьшить CurLen[Y] до (CurLen[X] + LatLen[Y][X]) и

(b) заменить CurSeq[Y] на (CurSeq[X] плюс признак фиксации в Y).

Если «фиксация» находится в состоянии Binary, то переменная BackTo сохраняет состояние X. Соответственно возврат из фиксации в состоянии Binary может произойти только в состояние, сохраненное в переменной BackTo. Следует обратить внимание на то, что на этой стадии ни один из элементов CurLen уже не содержит «пустого» значения.

2) Всем элементам NxtLen присваивают признак «пустого» значения.

3) Получаем следующий входной знак и находим по таблице 2[1)] все кодовые наборы, в которых этот знак может кодироваться напрямую (как правило такой кодовый набор один, хотя в отдельных случаях таких наборов также может быть либо ноль, либо два и более, плюс всегда знак Binary (кроме знаков, не являющихся знаками данных)). Для каждого такого состояния X:

a) выполнить непосредственное расширение этого состояния:

ЕСЛИ (CurLen[X] + CharSiz[X]) МЕНЬШЕ NxtLen[X], ТО:

(a) уменьшить NxtLen[X] до (CurLen[X] + CharSiz[X]) и

(b) заменить NxtSeq[X] на (CurSeq[X] плюс признак кодирования еще одного знака);

b) также проверить все возможные прямые переключения регистров из всех состояний Y, отличающихся от X:

ЕСЛИ (CurLen[Y] + ShftLen[Y][X] + CharSiz[X]) МЕНЬШЕ NxtLen[Y], ТО:

(a) уменьшить NxtLen[Y] до (CurLen[Y] + ShftLen[Y][X] + CharSiz[X]) и

(b) заменить NxtSeq[Y] на (CurSeq[Y] плюс признак регистрового перехода в состояние X для одного знака).

4) В особом случае, когда сам знак и его предшественник образуют одну из двузначных последовательностей, представляемых в кодовом наборе Punctuation:

ЕСЛИ CurLen[Punct] МЕНЬШЕ NxtLen[Punct], ТО

(a) уменьшить NxtLen[Punct] до CurLen[Punct] и

(b) заменить NxtSeq[Punct] на CurSeq[Punct].

5) Если NxtSeq[Binary] указывает на то, что с последнего перехода, вызванного знаком регистра Binary Shift, было закодировано ровно 32 байта, следует увеличить значение NxtLen[Binary] на 11.

6) В итоге все элементы массива NxtLen должны быть перемещены в соответствующие элементы массива CurLen, а все элементы массива NxtSeq — в соответствующие элементы массива CurSeq.

Примечание — На этой стадии некоторые элементы массива CurLen снова могут содержать признак «пустого» значения.

7) Если во входном потоке остаются необработанные знаки, следует вернуться к шагу 1.

Когда все входные данные будут обработаны, следует определить, какое состояние обладает наименьшим значением массива CurLen. Соответствующий элемент массива CurSeq описывает путь кодирования, обеспечивающий минимальную длину двоичного потока.

---

[1)] *В оригинале ИСО/МЭК 24778 ошибочно указано «таблица 1».*

**Приложение I**
**(справочное)**

# Рекомендуемые методы измерений в процессе технологического контроля

В данном приложении приведены некоторые рекомендации и методы, используемые для мониторинга и контроля за процессом создания символов Aztec Code, пригодных для считывания. Проверка качества печати в процессе изготовления символов к их числу не относится (для проверки качества печати необходимо использовать метод, указанный в разделе 15 и приложении Е). Данные рекомендации позволяют (как по отдельности, так и в совокупности) достаточно точно оценить практическую пригодность символов, полученных в результате печати.

## I.1 Контраст символа

Многие верификаторы линейного штрихового кода поддерживают режим рефлектометра или режим построения профилей отражения при сканировании и/или получении информации о контрасте символа по ИСО/МЭК 15416 при сканировании без последующего декодирования. За исключением символов, требующих особых условий освещенности, показания контраста, полученные с использованием апертуры 6 или 10 мил (0,150 или 0,250 мм) при длине волны излучения 660 нм (фиксируемое значение контраста символа, полный интервал в профиле отражения при сканировании или разность между пиковыми показаниями рефлектометра) достаточно хорошо коррелируют с контрастом символа, определяемого по изображению. В частности, по этим значениям можно проверить, что контраст символа остается выше минимального, установленного для предполагаемого класса качества печати символа.

## I.2 Специальные эталонные символы

Для контроля рекомендуется использовать символы Aztec Code, кодирующие длинные строки повторяющихся чисел, представляющие собой шаблоны двоичных данных. На рисунке I.1 приведены два примера: на левом символе закодировано 400 цифр «1», а на правом — 400 цифр «3».

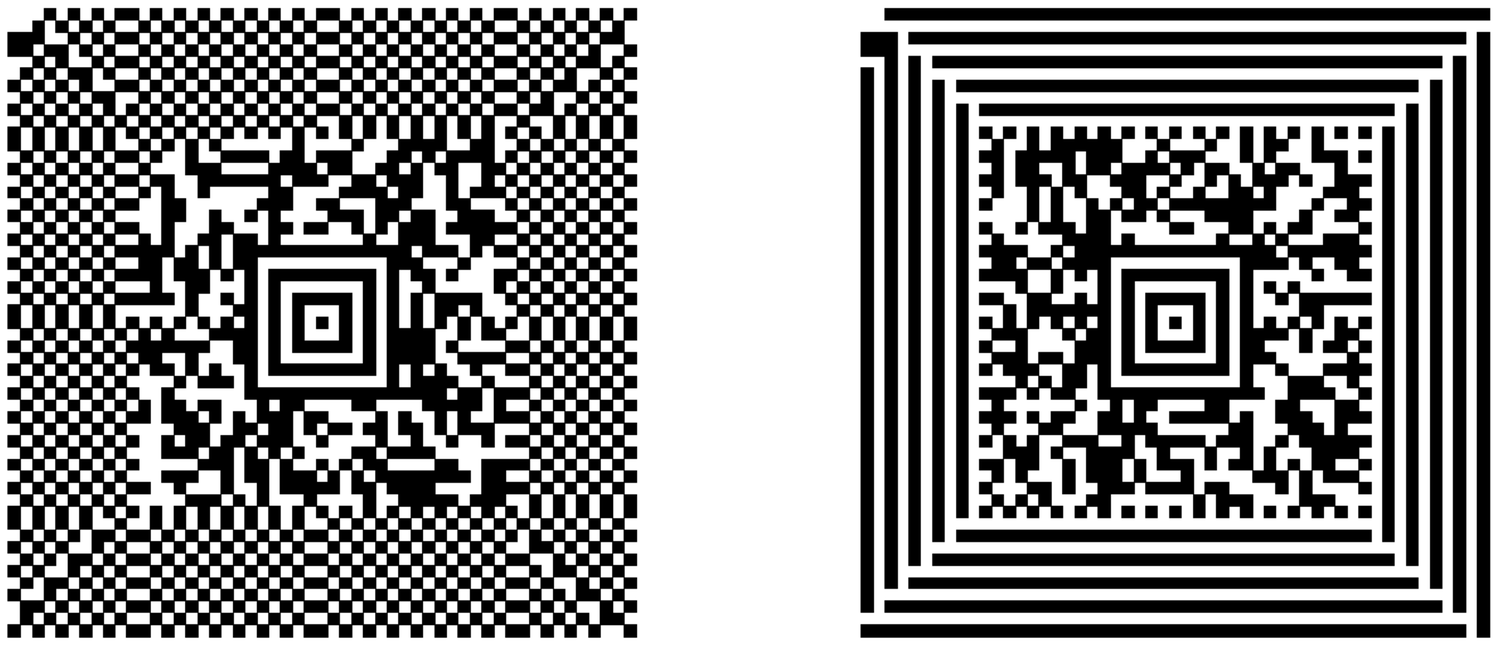


Рисунок I.1 — Тестовые символы Aztec Code

Левый символ предоставляет полезную визуальную информацию, относящуюся как к регулярности сетки, так и к приращению при печати. Области данных с шахматным узором должны быть визуально однородными и приблизительно на 50 % темными. Нерегулярность структуры сетки отображается в виде строк и столбцов неквадратных модулей или в виде нарушений порядка при просмотре по диагонали. Избыточное приращение при печати проявляется в виде уменьшения размеров светлых модулей относительно шага сетки.

Правый символ упрощает измерение отклонения ширины штриха по ИСО/МЭК 15416 (также называемого «приращение при печати») по обеим осям с использованием верификатора линейного штрихового кода, программируемого для перечня значений ширины элемента при сканировании без последующего декодирования. Как при номинально горизонтальном, так и при номинально вертикальном тестовом сканировании на границах образуются серии из 5 штрихов и 4 пробелов, которые должны иметь одинаковую ширину. При независимой

обработке в каждом направлении вычисляют среднюю ширину штриха $Z_B$ и среднюю ширину пробела $Z_S$, после чего нормализованный размер приращения при печати вычисляют по формуле

$$Z_B/(Z_B + Z_S).$$

Некоторые методы печати обычно дают существенно различающиеся значения приращения при печати по горизонтальной и вертикальной осям. Тем не менее, для символов Aztec Code этот показатель, номинально равный 50 %, должен быть в пределах от 35 % до 65 % для каждой оси.

**I.3 Оценка осевой неоднородности**

Изначально каждый символ Aztec Code имеет квадратную форму. Простые измерения общей ширины и высоты символа дают средние значения $X_{avg}$ и $Y_{avg}$, которые могут использоваться в формуле (см. 5.1.2) для оценки осевой неоднородности.

**I.4 Визуальная проверка на наличие дефектов**

Периодическая визуальная проверка шаблона поиска «мишень» в тестовых символах позволяет выявить систематические дефекты, которые могут оказать критическое воздействие на возможность считывания, т.е. чередующиеся концентрические квадраты в шаблоне поиска «мишень» всегда должны полностью находиться в состояниях с противоположными значениями коэффициента отражения. Сбои механизма печати, приводящие к появлению дефектов в форме светлых или темных полос, хорошо заметны и подлежат исправлению.

**Приложение ДА**
**(справочное)**

## Сведения о соответствии ссылочных международных стандартов национальным стандартам Российской Федерации

Таблица ДА

| Обозначение ссылочного международного стандарта | Степень соответствия | Обозначение и наименование соответствующего национального стандарта |
|---|---|---|
| ИСО/МЭК 646:1991 | NEQ | ГОСТ 27463 — 87 Система обработки информации. 7-битные кодированные наборы символов |
| ИСО/МЭК 15415:2004 | — | * |
| ИСО/МЭК 15424 | NEQ | ГОСТ Р 51294.1 — 99 Автоматическая идентификация. Кодирование штриховое. Идентификаторы символик |
| ИСО/МЭК 19762 (все части) | — | * |

* Соответствующий национальный стандарт отсутствует. До его утверждения рекомендуется использовать перевод на русский язык данного международного стандарта. Перевод данного международного стандарта находится в Федеральном информационном фонде технических регламентов и стандартов.

Примечание — В настоящей таблице использованы следующие условные обозначения степени соответствия стандартов:

- NEQ — неэквивалентные стандарты.

## Библиография


[1] Richard E. Blahut. Theory and Practice of Error Control Codes, Addison-Wesley, 1983-4 *(Блейхут Р. Теория и практика кодов, контролирующих ошибки, М.:Мир, 1986, 576 с.)*

[2] Lin and Costello. Error Control Coding: Fundamentals and Applications, Prentice Hall, 1983 *(Лин и Костелло. Корректирующие коды: основные положения и применения)*

[3] C. Britton Rorbaugh. Error Coding Cookbook, McGraw Hill, 1996 *(К. Бриттон Рорбо. Коды с исправлением ошибок: сборник рецептов)*

[4] ISO/IEC 6429 Information technology — Control functions for coded character sets
*(ИСО/МЭК 6429)* *(Информационные технологии. 7-битные и 8-битные кодированные наборы знаков. Управляющие функции)*

[5] ISO/IEC 8859-1 Information technology — 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No. 1
*(ИСО/МЭК 8859-1)* *(Информационные технологии. Информационные технологии. 8-битовые однобайтные кодированные наборы графических знаков. Часть 1. Латинский алфавит № 1)*

[6] ISO/IEC 8859-5 Information technology — 8-bit single-byte coded graphic character sets — Part 5: Latin/Cyrillic alphabet
*(ИСО/МЭК 8859-5)* *(Информационные технологии. Информационные технологии. 8-битовые однобайтные кодированные наборы графических знаков. Часть 5. Латинский/кирилловский алфавит)*

[7] ISO/IEC 15416 Information technology — Automatic identification and data capture techniques — Bar code print quality test specification — Linear symbols
*(ИСО/МЭК 15416)* *(Информационные технологии. Технологии автоматической идентификации и сбора данных. Спецификация испытаний качества печати штриховых кодов. Линейные символы)*

УДК 003.295.8:004.932.2(083.74):006.354 ОКС 35.040 П85

Ключевые слова: информационные технологии, технологии автоматической идентификации и сбора данных, автоматическая идентификация, штриховой код, спецификация символики Aztec Code, матричная символика

Редактор *Т. А. Леонова*
Технический редактор *В. Н. Прусакова*
Корректор *Н. И. Гаврищук*
Компьютерная верстка *Т. Ф. Кузнецовой*

Сдано в набор 18.08.2010. Подписано в печать 19.10.2010. Формат 60×84 1/8. Бумага офсетная. Гарнитура Ариал. Печать офсетная. Усл. печ. л. 6,05. Уч.-изд. л. 4,60. Тираж 100 экз. Зак. 1239.

---

ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru
Набрано и отпечатано в Калужской типографии стандартов, 248021 Калуга, ул. Московская, 256.