L2/01-206

CEN TECHNICAL REPORT **Draft 3 for CEN Trnnnn:1999**

1999-07-23

Descriptors: Data processing, information interchange, text processing, text communication, graphic characters, character sets, representation of characters, coded character sets, architecture

Information Technology -

Guide to the use of character set standards in Europe

This CEN Technical Report has been drawn up by CEN/TC 304

This CEN Technical Report was established by TC 304 in one official version (English). A version in any other language made by translation under the responsibility of a CEN member into its own language and notified to the Central Secretariat has the same status as the official version.

CEN members are the national bodies of Austria, Belgium, the Czech Republic, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland, and the United Kingdom.

CEN
European Committee for Standardization
Comité Européen de Normalisation
Europäisches Komitee für Normung

Central Secretariat: rue de Stassart 36, B-1050 Brussels

© CEN 1999 Copyright reserved to all CEN members

Ref.No. TR xxxx:1999 E

FOREWORD

This report was produced by a CEN/TC 304 Project Team, set up in June, 1998, as one of several to carry out the funded work program of TC 304 (documented in CEN/TC 304 N 666 R2). A first draft was discussed at the TC meeting in Brussels in November, 1998. A revised draft was circulated for comments within the TC and thereafter discussed at the TC plenary meeting in April, 1999. This revised version is based upon comments received during and after that meeting and is circulated for written ballot within the TC. If approved, the report will then be sent to the CEN BT for approval.

TABLE OF CONTENTS

Guide to the use of character set standards in Europe

1 Introduction

There exist today a large number of standards and related specifications concerning character repertoires and their coding in the form of official as well as manufacturer standards and intended for a wide range of applications and uses. Furthermore, there are character set standards for data communication and there are standards developed specifically for telecommunications applications. The situation can be very confusing to the non-expert user and to people involved in procurement.

The user of IT systems normally does not have to be concerned with these types of standards. However, there may be situations where the user has to be able to express working needs for certain character repertoires. It may also happen that the user, when involved in work together with other parties using other systems, needs to be able to interpret other people's specifications given in the form of reference to standards.

The procurer of IT systems should be able to specify requirements in the form of reference to established standards.

A particular purpose of the report is to give guidance for public procurement in Europe. Since there is an EC directive and a council decision for such procurement requiring the use of official European standards above certain procurement amounts, the report concentrates on such standards. There may be future editions, in which case more attention will be given other types of standards. (See also section 7.)

The main purpose of this report is to give guidance to users and procurers by explaining the purposes and relationships of the official standards in the domain of data communication. Explicit guidance is given in paragraphs marked with ‣.

The text is presented on two levels. The first level, contained in the body of the report, provides a general coverage of character repertoires, coding and uses. The second level, contained in the two annexes, provides much more detailed, tutorial information. The reader who finds the level of technical detail to deep may be better served by the "Manual: Standards for the electronic interchange of personal data: Part 5 – Character sets" (see References).

Further information on character sets and their standardization can be found in the document "Language automation world-wide: The development of character set standards" and on the Letter Database web site (see References).

2 Executive summary

The main body of this report is aimed primarily at the non-technical person who needs to become familiar with use of character set standards in Europe for various purposes in an IT environment. This audience will include managers/decision makers and their advisors; administrators (for procurement purposes); technicians (for programming and system development purposes); standardisers; perhaps also journalists.

The concepts of characters and their coding is introduced in section 5, and a conceptual model on the use of coded character sets is provided in section 6. The guide concentrates on official character set standards. However, there is a range of other standards for character sets that are not official, and there are also specifications concerning associated topics such as rules for ordering character strings. Section 7 goes on to place the official standards in the wider context of these other standards. Sections 8 and 9 describe a range of official character set standards with an international and a European scope respectively. Section 10 introduces a number of procurement issues, and section 11 provides sample text that may be used as the basis for inclusion in (public) procurement specifications for IT systems and software.

In addition, the guide has two annexes which contain a much more technical description of official character set standards.

The activities of CEN/TC304, the committee responsible for the promulgation of character set and related specifications in Europe, are described in section 12, and finally pointers for further reading and research are given in section 13.

3 Scope and field of application

The technical scope of this guide is primarily limited to official character set standards promulgated by ISO/IEC and CEN, as opposed to official telecommunications standards and manufacturer standards. However, an overview of all types of standards is given in section 7. The guide furthermore concentrates on European issues; thus character set standards for non-European languages are not covered.

The guide is mainly intended as an introduction for people who need to familiarise themselves with the concept of character sets and their coding; e.g. managers/decision makers and their advisors; administrators (for procurement purposes); technicians (for programming and system development purposes); standardisers; perhaps also journalists. Particular emphasis is placed on its use by procurers.

4 Definitions

The following terms are used in the body of this report and the official definitions are given here where they exist. They are taken from the standards ISO/IEC 9541:1991 and ISO/IEC 10646-1:1993, except when denoted by an *.

(character) **repertoire**: A specified set of characters that are each represented in a *coded character set*.

control function: An action that affects the recording, processing, transmission, or interpretation of data, and that has a coded representation containing one or more bit combinations.

***Note –** A bit combination in this context is a 7- or (more commonly) 8-bit byte.

control character: A *control function* the coded representation of which consists of a single bit combination.

***Note –** A control character is not strictly spoken a “character” but is called that way because its coded representation is of the same type as that of a coded graphical character.

coded character set (character set): A set of unambiguous rules that establishes a character set and the one-to-one relationship between the characters of the set and their coded representation.

***code table:** A tabular representation of a *coded character set*, showing also the coded representations.

***code page**: Synonym for *code table*, used in the IBM environment.

***code space**: The numeric domain occupied by all bit combinations used for the coding of a *coded character set*.

transliteration: The process which consists of representing the characters of an alphabetical or syllable writing system by the characters of a conversion alphabet.

Note – In principle, a transliteration should be a one-to-one conversion.

***fall-back**: A non-reversible transformation consisting of the substitution of an output character which cannot be represented on the output device by one or more characters which can.

combining character: A member of an identified subset of a coded character set, intended for combination with the preceding or following graphic character, or with a sequence of combining characters preceded or followed by a non-combining character

***diacritic, diacritic mark**: A mark intended for the association with a letter (e.g. acute accent).

glyph: A recognisable abstract graphic symbol which is independent of any specific design.

5 Characters and their coding

5.1 Characters, glyphs and languages

For the presentation of written text we use letters, digits and punctuation marks. Often we also use special symbols such as currency signs. All of these are called characters, and the collection of characters for a specific purpose, such as the presentation of text in a specific language, is called in the standardisation context a *character repertoire*. The most common type of repertoire is of course the alphabet of a language, complemented by the ten digits and a set of special characters.

A character is represented in printed form or on a display surface; hence it must have an agreed shape. Of course, a character may be represented by many variations of its basic shape (or shapes, as with g and g) depending on the font in use (e.g. Times Roman or Arial). No matter how many such variations may be used to represent a character, the basic shape is always recognisable to the human eye. This inherent shape of a character, which is independent of font, is known as a *glyph*. However, it should be recognised that this concept is less straightforward than it first might appear. Thus one and the same glyph may represent, in different contexts, different characters (e.g. the Latin character B is not the same as the Cyrillic character B).

Although the glyph concept is important for the definition of character repertoires, it is not central to the theme of this guide. The reader who wishes to obtain more information about glyphs is referred to ISO/IEC TR 15285, *An operational model for characters and glyphs* (see References).

Almost every language has its own character repertoire. However, the fact that many European languages have a large number of characters in common naturally facilitates the work on defining character repertoires for Europe. In CEN/TC 304 there is a separate activity on providing a catalogue of the alphabets of indigenous languages.

5.2 Coding

In IT systems a character is represented by a 7- or 8-bit combination, usually expressed as a numeric code. A character repertoire with its corresponding set of codes is called a *coded character set* or just *character set*. Such a set is often represented graphically in the form of a *code table* (Figure 1), which also illustrates the principles of the distribution of the codes, the *code structure*. Furthermore, the totality of the bit combinations used for a coded character set is called its *code space*.

	0	1	2	3	4	5	6	7
0			SP	0	@	P	`	p
1			!	1	A	Q	a	q
2			”	2	B	R	b	r
3			#	3	C	S	c	s

Figure 2 – Code table. The first four rows (out of 16) of a 7-bit code table. The row number translated into binary form gives the four least significant bits of the bit combination; the column number gives the three most significant bits.

Coded character sets are used for different purposes in computer systems, and the code structures may therefore vary. For instance, a coded character set used for interchange purposes often needs codes to be reserved for control characters, so that these may be included in the interchange data stream. However, a coded character set used for processing purposes may not need such reserved areas, which instead are often only used to represent more graphic characters. Examples of the latter are the manufacturer standards known as *PC code pages*.

5.2.1 Proliferation of codes; standardization

Early IT systems had severe size limitations. Therefore, the character codes had to be kept small. The earliest codes occupied 5 and 6 bits; later 7 and 8 bits have been used. These provide a coding capacity for 32, 64, 128 and 256 characters respectively. However, even with an 8-bit representation it is not possible to support all European languages in a single coded character set. Thus coded characters sets proliferated. As long as an application (and the character set it used) was restricted in use to a sin-

gle country or geographical region, this proliferation did not create problems, since a character set could be chosen to support the limited number of languages for that region. However, due to the requirements of international trade and the increase in travel, the limitation in the number of characters in one coded set has caused great problems of application interoperability.

In order to avoid a very large number of private character set specifications, many with overlapping scope and leading to interoperability problems, standardisation was needed. It was carried out both by the official standardization organisations and by the manufacturers, most notably by IBM, Apple and Microsoft.

Modern IT systems no longer have the earlier restrictions in size, and a solution is now available which uses a code space sufficient to accommodate the characters of every language in the world in one and the same coded character set. However, since the old solutions seem likely to continue to exist until perhaps 2025, the old problem may remain acute for some time. There will be the added complication of using the old and the new systems together as well as how to migrate, in an orderly fashion, to the new system.

5.3 Control functions and control characters

For IT processing purposes, it is necessary to indicate within a data stream where some action is required, e.g. a carriage return or new line. Such actions are performed through *control functions*, which do not have graphical representations. Over 160 control functions have been standardized. Some of them, such as the carriage return, are represented by a single *control character*, which, even though it does not have a graphical representation, has a coded representation of that type and can therefore be included in a code table. Others are represented by a sequence of characters with a special introducing control character at the beginning of the sequence.

6 The character handling model

Figure 1 below illustrates the character handling model. It represents a simplified IT scenario which consists of two computer systems connected by a communications link. The purpose is to show the different aspects of the handling of characters by users and computer systems and thus introduce basic concepts that will be used in the following sections of the report. It is also intended to help differentiate between the roles of the user(s) and the procurer in the context of this guide.

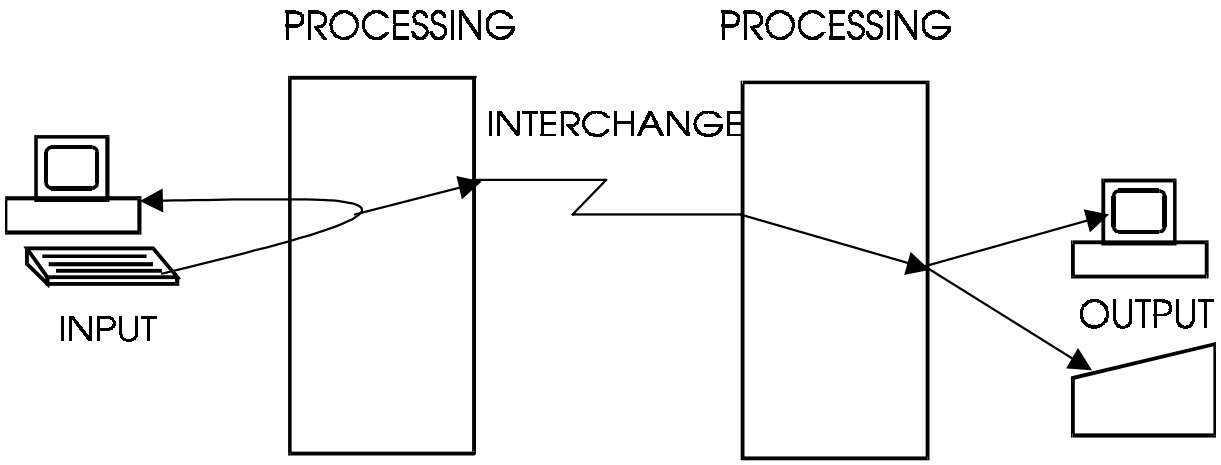

Figure 2 – Character Handling Model

Note – In parallel with the development of this guide, a separate CEN Technical Report was produced entitled "Character Repertoire and Coding Transformations – General model for graphic character transformations". The model described in this guide is entirely consistent with the more general model.

6.1 The input function

The input function provides for the entering of data into a computer system. Figure 2 uses a keyboard for input, but any device capable of entering character data may be used.

- For the user, the main issue is whether or not there is available in the computer system a character repertoire for input which is sufficient for the requirements. For the procurer, the main issue is to produce a procurement specification which satisfies the input needs of all intended users of the product.

Note that the representation of the input text on the monitor screen is a result of both the processing function, e.g. a word processor, and an output function (to the screen).

6.1.1 Keyboards

The main keyboard standard is ISO/IEC 9995, *Keyboard layouts for text and office systems*. National keyboard standards have, in general, been promulgated based upon this international standard. Keyboard standards are related to character set standards but are not central to the theme of this guide. CEN/TC304 has a separate activity on European keyboard standardisation.

6.2 The processing function

The processing function provides for the manipulation of data according to the needs of an application.

Once input, the data is expressed in some internal computer system code. In addition, other information may be associated with each character such as colour, emphasis level and font. Such information is usually intended for some document processing function. Thus the system internal code structure may be quite complex. However, at its heart is the character code itself; document handling and processing is outside the scope of this guide.

Most commercially available computer systems do not use standardised character sets for internal representation of character data, but proprietary character sets or manufacturer specifications.

- The user needs to be able to have all input characters processed, while, again, the procurer needs to produce the appropriate procurement specification.

6.2.1 Ordering

A particularly common requirement on the processing function is that it be able to order character based data. The main ordering standard is ISO/IEC 14651, *International string ordering – Method for comparing character strings and description of a default tailorable ordering*. Standards for ordering, while related to character set standards, are not central to the theme of this guide. In CEN/TC 304, there is a separate activity on European standardisation of ordering.

6.3 The interchange function

The interchange function provides for the interchange of data between computer systems. Since the character sets for processing are generally defined by manufacturer specifications, they are likely to be different in two different computer systems between which data are to be exchanged. Thus a character set for interchange is needed which will have to be different from one or both of the processing character sets. This is where character set standards become very relevant to reduce the number of interchange character sets – potentially one for every possible pair combination of different computer systems.

The main problem here is that there may not be a one-to-one correspondence between the characters in the character set for processing and those in the character set for interchange. In such cases less than trivial transformation functions are required at the interfaces between processing and interchange; see clause 10.2. Such transformations may incur a loss of information.

- The user requirements will relate to the functionality of the interchange (e.g. what loss of information, if any, may be accepted).

Such requirements may also include policy decisions of a more technical nature, e.g. as to what code structure to use. As before, the procurer's role is to transform the requirements into technical specifications.

6.4 The output function

The output function is the process of converting the internal coded representations of the characters to a visual representation on a display or hard copy device. The output character sets may be different depending of the output medium.

The handling of output to physical devices is usually an internal computer system function. Application programs, such as word processing packages, normally have the ability to control also the rendition of the output. This includes the use of fonts, both type and size, and also the use of various levels of emphasis and colour. In some cases, information which specifies particular values of these attributes is carried with the individual character codes right from the time of input. As already stated, these features are outside the scope of this guide.

The main problem is when the output character set is smaller than the character set for processing. The computer system software has to substitute one or more characters for those which cannot be represented by the output function (see clause 10.2). Again, such transformations may incur a loss of information.

‣ The user requirements will relate to the functionality of the output (e.g. what loss of information, if any, may be accepted). As before, the procurer's role is to transform the requirements into technical specifications.

6.5 Cultural issues

Each country or region in Europe (and elsewhere) has cultural conventions which affect the manner in which character sets are used by application programs. Such conventions include, but are not restricted to, ordering (already mentioned), numeric formatting, monetary formatting, date and time conventions, affirmative and negative answers, the use of special characters and personal name rules.

Cultural conventions primarily affect the processing function but may also impact the other functions. They are related to the use of character set standards but are not central to the theme of this guide. In CEN/TC304 there is a separate activity on cultural conventions which is developing and maintaining a registry of such conventions.

7 Official standards, manufacturer standards, and related standards

There are four main categories of character set standards:

- *Official standards for the computer and data communications industry*, promulgated by international standards organisations such as ISO/IEC and CEN, but also by national standards organisations. They are primarily intended for input (mainly for the specification of character repertoires) and for the interchange function in general applications.

- *Official standards for the telecommunications industry*, promulgated by ITU-T (formerly CCITT; the standards are called Recommendations) and ETSI, mainly intended for the interchange function in telematics applications.

- *Manufacturer standards for the computer and data communications industry*, developed by industry groupings such as the UNICODE consortium and individual companies such as IBM and Microsoft. They are mainly used for the input and processing functions but also for the interchange function in internal (proprietary) networks.

- *Related standards*, which involve character sets such as keyboard standards, ordering standards and transformation standards.

As already explained, this guide concentrates on the first of these categories (but does not cover national standards). In the remainder of this section further information is given on the other categories.

7.1 Telecommunication standards

Character set standards promulgated by ITU-T and by ETSI support applications primarily

defined and promoted by the telecommunications companies. Thus for Teletex, Recommendation T.61 was developed. It was subsequently replaced by T.51, which is also used as the basis for character set support in other telematic applications such as the X.400 Message Handling Service and the X.500 Directory Service. There are some links, however, with ISO/IEC standards. T.51 is equivalent to ISO/IEC 6937 (see section 8); and there is also T.50, equivalent to ISO/IEC 646 (see section 8). This alignment is due to collaboration between the two standards organisations. Nevertheless, ISO/IEC 6937 is used primarily for telematic applications and services.

ETSI has developed character set standards in support of radio paging (ERMES), GSM and RDS, but these standards are not aligned at all with ISO/IEC standards.

7.2 Manufacturer standards

Manufacturer standards can be grouped into IBM EBCDIC variants and those in support of personal computers such as the PC and the Macintosh. All use 8-bit codes. The EBCDIC code pages have their own defined structure together with an invariant set of characters and code positions. This is similar in concept to the parts of ISO/IEC 8859 (see section 8). Indeed some EBCDIC code pages have the same repertoire as parts of ISO/IEC 8859, which makes it straightforward to perform mappings between the two environments. Other EBCDIC code pages are designed to support specific countries or regions. More information on this may be found in the IBM document “Character Data Representation Architecture” (see References).

Both IBM and Microsoft have designed PC code pages, and each page is aimed at supporting specific countries or regions. PC code pages do not in general comply with the 8-bit code structure specified in ISO/IEC 2022 (see section 8), and mappings between the two environments may result in the loss of information.

IBM EBCDIC and PC code pages are described in Appendix I of “IBM National Language Support Reference Manual Volume 2” (see References).

The document “Comparisons of Standardized Character Sets for Europe” provides mappings of the differences between various official and manufacturer standards (see References).

7.3 Related Standards

In addition to standards on the definition of character repertoires and coding, there are standards relating to the handling of character sets and other character set issues.

Keyboard standards are concerned with the input to computer systems of character information. They cover such topics as keyboard layouts. There is one international standard plus a range of national standards, most of which are based on the international standard.

Ordering standards are concerned with sorting character strings or sets of strings into some order. There is an international standard, and work is under way in Europe on the ordering of European repertoires. A related activity concerns the matching of character strings, for use in, for instance, search engines.

Transformation standards are concerned with rules for mapping the characters from one repertoire onto another, and from one code system onto another. They are sometimes needed when character information from one computer system needs to be transferred into a different computer system due to a mismatch of functionality. An example is the mapping of an official standardised character set onto EBCDIC, so that character based information may be processed in an IBM machine. There are a number of different types of transformation needed for various scenarios. See also clause 10.3.

8 International character sets

This section describes internationally standardised character sets (i.e. both repertoires and coding) used primarily for input (mainly for the specification of repertoires) and interchange. For processing, most IT systems use proprietary manufacturer standards.

Because of the need to combine different standards and also in order to make the standardisation itself more consistent and effective, a common platform standardising the principles

for code structure, code extension, implementation and registration has been established. The standards which define that platform may be called framework standards.

The international character set standards in this area may be classified in the following way:

- 7- and 8-bit framework standards
- 7- and 8-bit character set standards
- The universal character set (UCS) standard (multiple octet)
- The control function standard

8.1 Framework standards for 7- and 8-bit environments

See also Figure 3 below.

- ISO/IEC 2022:1994, *Information technology – Character code structure and extension techniques*, is the basic framework standard. It provides the framework for the definition of all the 7- and 8-bit coded character sets and their use. In particular, it defines a complex code extension facility which allows considerably more characters to be used in a specific environment than would otherwise be the case with a single 7- or 8-bit code space. It also partitions the 7- and 8-bit code spaces between the representation of graphic characters and control characters. See also Annex A.

- ISO/IEC 4873:1991, *Information technology – ISO 8-bit code for information interchange – Structure and rules for implementation*, is a refinement of ISO/IEC 2022 which concentrates specifically on the implementation and use of the 8-bit coding. It lays the formal basis for the specification of two 8-bit character set standards – the various parts of ISO/IEC 8859 and ISO/IEC 10367 (see section 9).

- ISO/IEC ISP 12070:1996, *Information technology – International Standardised Profiles FCSnnn – Character set 8-bit code structure based on ISO/IEC 2022 – Part 1: FCS111 – 2022 Option 1*, is a profile of both ISO/IEC 2022:1994 and ISO/IEC 4873:1991. It goes into considerable detail concerning the use of character sets within Open System Interconnection (OSI) protocols and, in particular, concentrates on conformance requirements (in the interests of interoperability). It is particularly the latter which makes this standard relevant in this context, whereas its OSI aspects are less interesting nowadays.

Note – A *profile* is a selection of options from one or more standards, constituting a more narrow form of specification than the base standard(s) it refers to.

- ISO 2375:1985, *Data processing – Procedure for the registration of escape sequences*, governs the registration of coded character sets. In the ISO/IEC 2022:1994 framework, the extension mechanisms use escape sequences to control the use of more than one registered coded character set. Such an escape sequence consists of the Escape control character followed by a variable number of octets. In this way, the sending implementation in an interchange can inform the receiving implementation about the elements of the code structure that will be used in the interchange.

A registration authority allocates values for the escape sequences to be used with the registered coded character set. The registration authority is the Information Processing Society of Japan/Information Technology Standards Commission of Japan (IPSJ/ITSCJ), and the register may be accessed on-line at http://www.itscj.ipsj.or.jp/ISO-IR/. All standardised coded character sets are registered, but not all registered coded character sets are standardised. There are currently over 200 registrations. A coded character set may be identified in text, such as a procurement specification, by the sequence ISO-IR nnn, where nnn is the registration number.

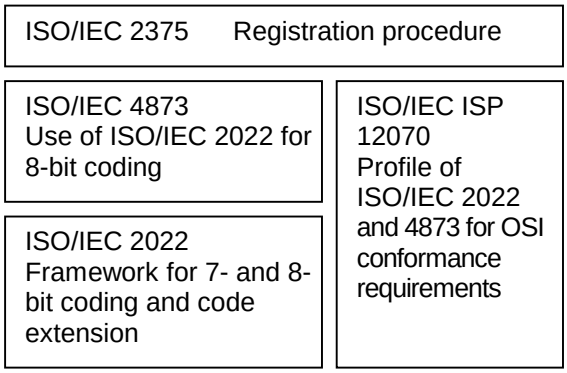

Figure 3 – Overview of framework standards.

‣ Most users and procurers need not concern themselves overmuch with ISO/IEC 2022:1994 or ISO/IEC 4873:1991, since their provisions are usually called up by the referring standards. For this reason, they need not be referred to directly in procurement specifications. However, the sophisticated user with a complex requirement, who may be mixing and matching the use of registered character sets rather than standardised character sets, will probably need to become familiar with them. In this case, the procurer will also need to make reference ISO/IEC ISP 12070-1 in order to ensure the necessary conformance. This ISP is also relevant in procurement situations which include the proposed use of products using OSI protocols.

‣ Neither should most users and procurers need to refer to the registration standard ISO 2375. However, they may well need to know the identities of one or more registered character sets.

8.2 7- and 8-bit character set standards

• ISO/IEC 646:1991, *ISO 7-bit coded character set for information interchange*, is perhaps the oldest internationally standardised character set in existence. It has a very widespread use. A particular characteristic is that a number of code positions have an optional specification, which allows national variants to be specified. In the interests of application interoperability, however, this practice is now deprecated.

‣ Users and procurers are advised not to use national variants of ISO/IEC 646:1991 .

The set of characters which only includes the non-optional characters is called the Invariant Set. There are default allocations of characters to the optional code positions, and when those are used, the character set is called the International Reference Version (IRV). The IRV is registered as ISO-IR 6. It was changed in the 1991 edition of the standard with the currency sign being replaced by the dollar sign.

The standard also defines a default set of control characters to be used where applicable. The IRV:1991 together with the default control character set is identical to ASCII (American Standard Code for Information Interchange), a name in common usage in the industry. This character set does not contain any letters with accents or diacritical marks. It is therefore unsuitable for use in representing most European languages.

• ISO/IEC 8859:nnnn, *Information technology – 8-bit single-byte coded graphic character set*, is a multi-part standard whose parts have been promulgated at different times. Each part defines an 8-bit character set with a specific language coverage. Although language coverage has been the driving force behind the proliferation of parts, each part is designed for widespread general use such as normal commercial and administrative applications. They have not been designed for specialist minority applications.

Each part of ISO/IEC 8859 contains a code table made up of two halves (Figure 4). The left half is identical to the graphic characters of the IRV of ISO/IEC 646:1991 coded in 8 bits. The right half is specific to the part in question. Together, the two halves define a self contained 8-bit coded character set of up to 191 characters. There are currently 15 parts to ISO/IEC 8859, falling into two categories. In the first category, containing 9 parts, the second table contains extra Latin characters. The title of each of these parts is Latin Alphabet No. n where n lies in the range 1 to 9. In the second category, the second table contains characters from a non-Latin script. The title of each of these parts is Latin/xxx where xxx is the name of the second script (e.g. Latin/Greek). Each part of ISO/IEC 8859 supports a range of lan-

guages. They are listed in Annex A in the section on ISO/IEC 8859.

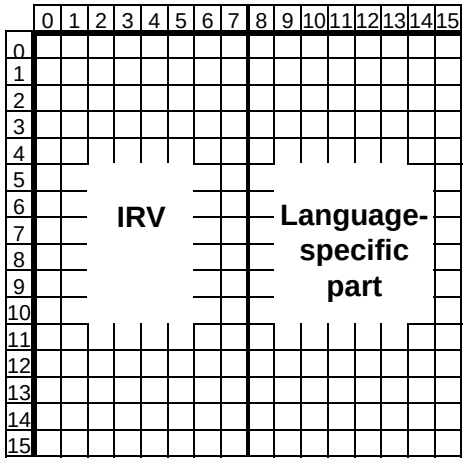

Figure 4 – The two halves of the ISO/IEC 8859 series.

Note – In products supporting ISO/IEC 8859, no code extension is used. However, since the right halves of the code tables – the ones specific to each part of the standard – are all registered, any one of them can form part of larger coded character sets with the use of code extension techniques. For such implementations, reference must be made to the registration number of the code table and not to the corresponding part of the standard.

- ISO/IEC 6937:1994, *Information technology – Coded graphic character set for text communication – Latin alphabet*, is a standard primarily used in ITU-T telematic applications. It is equivalent to the ITU-T recommendation T.51. It overcomes the restrictions placed by the 8-bit code space by using two octets for characters with accents or diacritic marks: a non-spacing diacritic mark followed by the character for which the diacritic mark is to apply. The standard specifies a repertoire of 333 characters. However, it has not found favour outside the world of telecommunications. It is therefore not recommended for use by European IT applications.

- ISO/IEC 10367:1991, *Information technology – Standardised coded graphic character sets for use in 8-bit codes*, attempts to overcome the 191 character limit of the parts of ISO/IEC 8859:nnnn by specifying the code tables of ISO/IEC 8859:nnnn parts for use with the code extension facilities of ISO/IEC 2022:1994, qualified by ISO/IEC 4873:1991. This effectively allows coded character sets of 382 characters to be defined and used. The standard also defines how ISO/IEC 6937:1994 may be used in conjunction with the non-Latin script code tables defined in the right half of the various parts of ISO/IEC 8859:nnnn. A problem with this standard is that it is now out of date, since it refers only to the first 9 parts of ISO/IEC 8859:nnnn whilst there will soon be 15 parts. However, in principle the specification used in ISO/IEC 10367:1991 may of course be applied as well to the remaining parts of ISO/IEC 8859:nnnn.

The merit of this standard is that it specifies a functionality which goes somewhat beyond that of EN 1923 (see clause 9.1) while still using the “old” 8-bit technology.

8.3 The universal character set (UCS) standard

It is obvious that the limitations of both 7- and 8-bit coding create problems in a world where increasingly texts in very different languages need to be communicated between computer systems. Some years ago, the radical step was taken to create a standard with enough code capacity to allow the coding of all alphabets in the world within one framework.

- ISO/IEC 10646-1:1993, *Information technology – Universal Multiple-Octet Coded Character Set (UCS), Part 1: Architecture and Basic Multilingual Plane*, is a relatively new standard which aims to overcome all the shortcomings of the 7- and 8-bit character set standards.

8.3.1 The Basic Multilingual Plane

ISO/IEC 10646 defines a character set and a code structure of up to 4 octets. The main aim was to provide sufficient code capacity so that the alphabets of all the known languages in the world, together with a large range of special characters, could be accommodated. It has to be said that four octets is an overkill, but this was kept to cater for any possible future requirement. It was decided to concentrate at first on populating the lowest two octets of the code space, and this space is known as the Basic Multilingual Plane (BMP).

Since publication, the standard has been subject

to a number of amendments, most of which add further language support to the BMP.

It is planned that the BMP will contain all characters, including combining characters, needed to write all the known living languages of the world, with the exception of some of the Chinese and Japanese ideographs. It should be noted that over 27,000 Han characters for Japanese and Chinese are assigned to the BMP, so there is significant support for the languages of those countries which only require a few thousand characters for the large proportion of applications. The BMP certainly already contains all the characters needed for the vast majority of European languages. For this reason, a two octet as well as a four octet representation is specified for use when the application environment only makes use of the BMP. Other coded representations are specified to cater for situations where more than the BMP is used but where better efficiency than use of the four octet form is needed. More detailed technical information may be found in Annex B.

‣ The UCS represents the future direction for coded character sets and is being implemented by a wide range of suppliers. For this reason, users and procurers are recommended to consider seriously its use for new systems in preference to 7- and 8-bit coded character sets. It should be noted that, whereas 7- and 8-bit character sets are primarily used for interchange, there is evidence that the UCS will be used for the processing function as well. However, 7- and 8-bit systems will continue to exist for some considerable time, possibly for up to 25 years. If interoperability with such systems is required, it is important for the procurer to make sure that suppliers of new products can provide this before there is a commitment to use or purchase.

8.3.2 Subsets

A procedure has been agreed for the submission of proposals for the addition of subsets to the base standard. There are several reasons for this. Many systems have quite modest character set requirements, which are application dependent. There are severe restrictions in small implementations such as for mobile telephones, mobile digital radio systems and set top boxes for digital television. Furthermore, there is in Europe a need to identify a character set specifically to support all European languages. Submissions are normally channelled through the national standards organisations.

8.3.3 Unicode

An equivalent but not identical specification to this standard is also published by an industry consortium called UNICODE. Although the two are separately published, every attempt is made to ensure that they are kept in line. In particular, both publications define the characters contained in the Basic Multilingual Plane (BMP).

The UNICODE standard plays an important role from the point of view of procurement. As already noted there have been a number of amendments to ISO/IEC 10646-1, and over the past few years the appearance of amendments has been continuous. The standard changes every time an amendment is ratified. This means that it is not reasonable for suppliers' implementations to be kept in synchronisation. The UNICODE standard, on the other hand is not changed very often and so suppliers can keep in synchronisation with it. Version 2 of the UNICODE standard is identical position for position with the first edition of ISO/IEC 10646-1 plus its first 7 amendments. Version 3 of the UNICODE standard, due for publication in 1999, will be position for position identical with the second edition of the international standard, which will incorporate the first 31 amendments. These publications represent significant points for the industry to synchronise its products.

8.4 Control functions

- ISO/IEC 6429:1992, *Information technology – Control functions for coded character sets*, defines a comprehensive set of control functions to be used with graphical character sets. It includes the definition of control characters whose codes are positioned in the control character partitions of the 7- and 8-bit code space specified by ISO/IEC 2022:1994. The remainder are coded as sequences of octets to be preceded by an introducer character. Most of them are concerned with document handling. The document handling control functions, however, are rarely implemented in commercial products, and the importance of this standard lies in its specification of single control characters.

This standard was originally defined for use in the 7- and 8-bit code structures. However, ISO/IEC 10646-1 has provisions which allow control functions to be used in the UCS code structure.

9 European character sets

In order to provide specifications particularly suited for European applications, CEN has standardised implementations of ISO/IEC standards, including subsets of the UCS standard. CEN promulgates ENs (European Standards) and CWAs (CEN Workshop Agreements). An EN is promulgated by a full CEN Technical Committee and is ratified by a ballot involving the national standard organisations (NSOs) belonging to CEN. A CWA does not carry the authority of a full EN and is promulgated by a Workshop whose membership is open to all comers. A CWA is not subject to NSO ballot; its authority rests on the fact that it is an agreement – and commitment – based on a wide marketplace consensus.

9.1 8-bit character sets

• EN 1923:1998, *European Character Repertoires and their Coding – 8 bit single byte coding*, specifies seven repertoires and their 8-bit coding for use in Europe. This standard is capable of supporting a large proportion of European language applications. The repertoires are:

- four repertoires for the Latin script
- a repertoire for the Greek script
- a repertoire for the Cyrillic script
- a repertoire for box drawing

The four repertoires for the Latin script are:

- The *Invariant-Latin* repertoire (IVL), based upon the invariant set of ISO/IEC 646:1991 – 83 characters. Since the IVL does not support accented characters, it is not much use for representing non-English European languages. It is therefore deprecated for general use in Europe.

- The *Initial-Latin* repertoire (IL), based upon the IRV of ISO/IEC 646:1991 – 95 characters. For the same reason as for the IVL, this repertoire is also deprecated for general use in Europe. However, it is used for EDIFACT messaging.

- The *Basic-Latin* repertoire (BL), based on ISO/IEC 8859-1:1998 – 191 characters. The Latin characters in BL are primarily intended for use in official languages and several minority languages in a contiguous block of Western European countries.

- The *Large-Latin-8* repertoire (LL8), based on the Latin characters coded in ISO/IEC 10367:1990 – 334 characters. This is the same as the repertoire of ISO/IEC 6937: 1994 plus the LATIN CAPITAL ETH (Icelandic) character. The Latin characters in LL8 are primarily intended for use in official languages and several minority languages in a contiguous block of European countries to the west of the Russian Federation, Belarus and the Ukraine, and also in Turkey.

EN 1923 specifies how these repertoires are coded using the extension mechanisms of ISO/IEC 2022:1994, and how they can be combined. A set of combination options is given. (Not all possible combinations are allowed.)

9.2 16-bit character sets – The multilingual European subsets

The *CEN Workshop Agreement (CWA) nnn* specifies three subsets of ISO/IEC 10646-1:1993. These are the so called Multilingual European Subsets (MES). The purpose of the CWA is to identify subsets of the BMP for specific use in Europe so that collection identifiers may be assigned by ISO/IEC JTC1/SC2 with respect to ISO/IEC 10646-1.

Note: at the time of development of this guide, the CWA had not yet been approved and therefore its publication in that form is not guaranteed. Hence the collection identifiers have not yet been assigned. However, the work towards this particular goal will continue.

• *MES-1* contains the 333 characters of the repertoire of ISO/IEC 6937:1994 plus LATIN CAPITAL LETTER ETH (Icelandic) and the EURO SIGN. It was created for reasons of

compatibility with LL8 and provides a straightforward migration path for older applications. The characters in this subset are primarily intended for use in official languages and several minority languages in a contiguous block of European countries to the west of the Russian Federation, Belarus and the Ukraine, and also in Turkey.

- *MES-2* is much larger than MES-1 and is a superset to that repertoire. It contains additional characters from the Latin script, characters from the Greek and Cyrillic scripts and the characters from a large number of commonly used coded character sets. Its characters are primarily intended for use in official languages and several minority languages in a contiguous block of European countries to the west of the Caspian Sea, excluding Georgia (which uses Georgian script), and Armenia (which uses Armenian script). MES-2 aims to satisfy user requirements in general purpose data and text processing applications in typical office environments in Europe. In addition to a collection of Latin, Greek, and Cyrillic letters, it specifies a wide range of digits, punctuation and other symbols. It is broadly similar to the repertoires in EN 1923, in ISO/IEC 10367 (except for Hebrew and Arabic characters in ISO/IEC 10367), and in the various parts of ISO/IEC 8859 which use European scripts. It is also broadly similar to the repertoire used in the "WGL4.0 Character Set" defined in the increasingly used OpenType specification. It does not, however, contain all the characters of WGL4. MES-2 furthermore includes the full range of polytonic Greek characters, used widely in Greece until 1982 and still enjoying usage there, whether in ordinary contexts or in the context of the study of Ancient Greek.

- *MES-3* is much larger than MES-2 and is a superset to that repertoire. It includes all the collections of ISO/IEC 10646-1 which contain the characters of the Latin, Greek, Cyrillic, Armenian, and Georgian scripts, together with those collections of symbols used academically, commercially, and scientifically in Europe. By including combining characters and phonetic characters of the Latin alphabet (including the International Phonetic Alphabet, I.P.A.), MES-3 also provides for the needs of transliteration and transcription of many of the world's languages into the Latin script, as well as the needs of European publishers (of dictionaries, encyclopaedias, phrase books, and textbooks) who routinely make use of the I.P.A. It also aims to provide all characters used in several more existing European and International character set standards, as well as some proprietary character sets. It provides a European specification for the needs of more specialised groups in government, industry, publishing, academia, and the private sector, than MES-1 and MES-2 do.

MES-3 is defined in two forms: One is a *non-fixed* subset (MES-3A), which permits automatic inclusion in it of any new characters added to the BMP whose coding positions fall within the collections which define this subset. The other is the *fixed* subset (MES-3B), which is invariant over time.

9.3 The EURO SIGN

In order to support the move to EMU in the Euro zone of the EU, the EURO SIGN has now been included in the following of 8-bit code tables.

- ISO/IEC 8859-15:1998 – Latin-9
- ISO-IR 203 – European Supplementary Latin Set ("Latin 9")
- ISO-IR 204 – Supplementary set for Latin-1 alternative with EURO SIGN
- ISO-IR 205 – Supplementary set for Latin-4 alternative with EURO SIGN
- ISO-IR 206 – Supplementary set for Latin-7 alternative with EURO SIGN

In all of these, the EURO SIGN is in the code position usually occupied by the currency sign. The EURO SIGN is already included in ISO/IEC 10646-1 by amendment 18 and is also included in all three Multilingual European Subsets (MES). Furthermore, EN 1923 is currently under review with the intent of revising it to include the EURO SIGN.

10 Procurement issues

The procurement issues described here are related to specific character handling functions (see section 6).

10.1 Repertoires and code structures

10.1.1 User issues

The user does not need to be concerned with the *coded representation* of the input character set since a character, once input, has a code defined by the internal processing. What may be an issue here is the specification of the input character repertoire, which must be able to support the user requirements.

For the same reason, neither does the user need to be concerned with the coded representation of the output character set. Again, the repertoire needs to be sufficient to support the requirements. In particular, there should be font support for that repertoire. Character transformation may be an issue if the product cannot support the full output repertoire needed; see clause 10.2.

10.1.2 Procurement issues: Code structure for the interchange function

The procurement issues will centre around the interchange character set. As in the other cases, the repertoire needs to be sufficient to support the requirements of the user. The main issue will be the code structure. The industry is in a state of transition. The 8-bit code structure of ISO/IEC 2022:1994 has been around for some considerable time and is the safe, but limited, option. The future lies with the multi-octet code structure of ISO/IEC 10646-1:1993, but it will not for some time be available in all receiving systems. Still, this guide recommends that new procurements should specify the multi-octet code structure wherever possible. However, there may be cases where the new system has to operate in an environment which overwhelmingly uses 8-bit coding. If so, that code structure should be used. Increasingly, however, there will be situations where the new system has to operate with both code structures, and then the availability of dual support has to be considered.

10.1.3 Procurement issues: Repertoire for the interchange function

If the 8-bit code structure is chosen for the interchange function, this guide recommends that, in Europe, the repertoire(s) are chosen from those identified in EN 1923:1998.

- For *Latin script applications*, there is a choice of four repertoires. It is unlikely that the IVL or IL repertoires will be sufficient for European operation and a choice of these alone is deprecated, except IL for use in EDIFACT.

- For *Western European language application*, the BL repertoire should be sufficient.

- If *Eastern European languages* have to be taken into account, then the LL8 repertoire is more applicable. The Greek (BG) and Cyrillic (BC) repertoires can be added where appropriate, but they cannot be combined with LL8 unless the code extension facilities of ISO/IEC 2022 are used. It should be noted, however, that there may be significant limitations in these repertoires for some minority language operation and for some specialist technical applications. For such cases, it is recommended to seek specialist advice in the choice of coded character sets outside EN 1923:1998.

- If the *multi-octet code structure* is chosen for the interchange function, this guide recommends that one of the Multilingual European Subsets (MES) be selected for the interchange repertoire. MES-1 is almost identical to LL8. MES-2 provides a much more comprehensive coverage for European languages, whilst MES-3 guarantees support for nearly all indigenous European languages, specifically including minority languages.

The repertoire selection indicated above is a minimum requirement. The supplier may go further.

10.2 Transformation and fall-back

The choices of repertoires for the four character handling functions described in the model should not be made independently of each other. For instance, a choice of a large processing repertoire such as MES-3 may not be sensible if the interchange repertoire is BL within the 8-bit code structure. It is recommended that, if possible, the same repertoire is used throughout.

If this is not possible, the procurer must specify appropriate transformation, based on the user requirements. This may be done by analysing

the interfaces between the character handling functions in the model.

10.2.1 Interfaces

- **Input to Processing**. This is unlikely to be a problem since a computer system would never be configurable to allow an input repertoire to have characters outside of the processing repertoire. However, it is perfectly possible for an application to handle character information which contains characters outside the input or processing repertoires. In this case, the application will then operate with the processing repertoire, but will use a transformation mechanism to allow the user to identify extra characters for input. (An example of would be when someone in the UK needed to communicate with someone in Greece in Greek and the UK system did not support the Greek script.) Thus the application takes care of the problems, and no specific action by the procurer is required.

- **Processing to Output**. This may well become a common problem when the multi-byte UCS is used as the processing repertoire and the data stream received and processed makes extensive use of that repertoire. In the short term, there may be font restrictions which do not allow certain characters to be rendered on output. Here a transformation mechanism (see below) is needed which transforms unsupported characters into an output form that can be recognised

- **Processing to Interchange**. Here, either an a priori agreement between the communicating systems will be in place concerning the interchange repertoire and code structure, or some protocol negotiation will take place. The processing to interchange transformation requirement needs to be examined at two levels. First, there may be a *mismatch of the repertoires*. In this case, some repertoire transformation function will be needed, and the user requirement must then state whether this should be reversible or not. Secondly, a *code transformation* function may be needed since the processing code structure is likely to be based on some manufacturer specification whilst the interchange code structure is likely to be standardised. See clause 10.4.

- **Interchange to Processing**. This is a complement of the preceding case.

10.2.2 Transformation functions

When a character in one repertoire is not contained in the other, a *character transformation* is required. The requirements on such a transformation function may vary. For instance, the requirement may be that it is reversible (i.e. it must be possible to reconstruct the original character from its converted form). An example of this would be where the character é is replaced by the HTML representation &eacute. In another case, a non-reversible transformation may be all that is required. Here, characters that are defined in both character sets are passed through whilst other characters may be substituted by some common substitute character, or some approximation may be made (e.g. by replacing é by e and so on).

A particular type of transformation is caused by the situation when the output characters are represented in a different script from the input characters. *Transliteration* is then required. It is dependent not only on the scripts concerned but also on the languages. Thus the rules for transliteration from Russian is different if the target language is German than if it is French.

In Europe there are five scripts in current use for the indigenous languages: Latin, Greek, Cyrillic, Armenian and Georgian. In CEN/TC 304 there is not at this time any activity devoted to transliteration. Standards in this area are developed by ISO/TC 46 (Information and Documentation).

If the transformation causes loss of information (i.e. it is not reversible), the procedure is called *fall-back*. The computer system may provide means by which the lost information can be traced (for instance with a “reveal codes” function), but this requires some indication on the output device that such a substitution has actually taken place (e.g. by the use of a specific substitute character, a special level of emphasis or colour).

Another method is a one to many character mapping from which the identity of the original character can be deduced. As an example, élève may be transformed to e/le\ve. However, such a

technique can cause problems with tabular formatting of information.

Transformation is related to character set standards but is not central to the theme of this guide. In CEN/TC304 there is separate activity on transformation which is developing a model and is examining various transformation techniques..

10.4 Code structure interoperability

A more fundamental issue of interoperability concerns the use of the code structures themselves. Such issues arise at the processing/interchange and interchange/output interfaces. For both the 8-bit code structure and the UCS code structure, many options have to be defined for the receiver to be able to interpret the incoming data stream correctly. For example, in the 8-bit code structure, how will the receiver know which 8-bit character set is used? And in the UCS code structure, how will the receiver know which coding form is used? As a final example, how will the receiver know which code structure is used – 8-bit or UCS?

10.4.1 8-bit interchange code structure

If the 8-bit code structure is being used during interchange either an a priori agreement has to be in place on which character sets and code structure options to use, or the code extension facilities of ISO/IEC 2022:1994 must be used in the exchange to establish such an understanding. In the former case, a higher level protocol may determine the agreement before the interchange of character data takes place – an example is Internet Mail. In the latter case, the detailed conformance requirements on the use of ISO/IEC 2022:1994, as given in ISO/IEC 12070:1996, could be used. However, as has already been indicated, products rarely support the code extension facilities of ISO/IEC 2022:1994, so in most cases the user has to rely on the establishment of a priori agreements (which may be automated).

10.4.2 UCS interchange code structure

For the multi-byte UCS code structure, the following information is needed by the receiving system:

- the format of the code exchanged – e.g. 2 byte form (UCS-2), 4 byte form (UCS-4) or one of the transformation formats (UTF-8 or UTF-16).
- the subset of the UCS to be used in the interchange
- the order in which the elements of the multi-octet codes are sent
- the implementation level used (basically to determine whether combining character sequences are to be used)

Again, either an a priori agreement is needed, or the designation, identification and signature facilities specified in ISO/IEC 10646-1:1993 must be used. It is believed that the supply industry favours the use of *signatures* for the exchange of plain text files. (A signature is a sequence of octets sent at the start of an interchange to signal what code format will be used and what octet ordering is going to be used for the transmission of each multi-octet code representing a character.) The other two items in the list can be signalled with the use of escape sequences but are less important for interoperability.

‣ It is believed that most UCS products will include mechanisms using an a priori agreement or some higher level protocol exchange to determine these parameters automatically. The procurer must choose between such a product and a more general purpose UCS application, in which case broader solutions to this problem must be sought.

11 Procurement clauses

This section contains sample text that may be used within procurement specifications for products that need to support character set operation. The clauses given apply to general purpose products intended for commercial and administrative applications. They may be tailored to the needs of the specific procurement. If the procurement is for specialist applications which may have unusual requirements or may have restricted capabilities, it is recommended that the procurement officer seek expert advice.

Note – This section is primarily geared at public procurement. In the EU and EES member states, there are laws for such procurement which under certain circumstances stipulate the reference to official standards – in the first instance European Standards. For this reason,

this section only covers the European character sets specified in section 9. For other procurement purposes, manufacturer standards may well be used but are not referenced here. For such cases, the reader is referred to, e.g., the "IBM National Language Support Reference Manual Volume 2 – National Language Design Guide" and to the "Comparisons of Standardized Character Sets for Europe" (see References). Note also that for telecommunication applications, the reader is referred to corresponding ETSI and ITU-T standards.

11.1 Structure

The section is structured after the model described in section 6. Thus if the requirements are identified in terms of standard specifications in section 9, and the procurement specification is structured in accordance with the model, then the relevant procurement clauses can easily be looked up.

11.2 Input character repertoire

11.2.1 8-bit character repertoires currently not including the EURO SIGN: EN 1923:1997

The following clause should be included in the procurement specification in order to specify input repertoire(s) from EN 1923:1997.

> "The product shall support the input repertoire(s) xxxx specified in EN 1923:1998."

where xxxx is one or more of the following:

BL, LL8, BG, BC, BL & BG, BL & BC, BL & BG & BC.

(This guide deprecates the use of the repertoires IVL and IL.)

Note – EN 1923:1997 may be revised to include the EURO SIGN.

11.2.2 8-bit character repertoires which include the EURO SIGN

The following clause should be included in the procurement specification in order to specify input repertoire(s) including the EURO SIGN.

> "The product shall support the input repertoire(s) specified xxxx."

where xxxx is one or more of ISO/IEC 8859-15:1998, ISO IR 203, ISO IR 204, ISO IR 205, and ISO IR 206.

11.2.3 16-bit character repertoires: UCS collection identifiers

The following clause should be included in the procurement specification in order to specify input repertoire(s) using collection identifiers in ISO/IEC 10646-1.

> "The product shall support the input repertoire(s) identified in ISO/IEC 10646-1 collection identifiers nnnn."

Where nnnn is one or more of CId(MES-1), CIdMES-2, CId(MES-3A) and CId(MES-3B).

Note – If more than one input repertoire is chosen, the following clause should be included in the procurement specification.

> "The product shall be configured for use of the specified input repertoires together with applications as follows:
>
> Repertoires xxxx together with application(s) zzzz."

where xxxx are the repertoires as specified in preceding clauses and zzzz is (are) the name(s) of application(s) as appropriate.

11.3 Output character repertoire

11.3.1 8-bit character repertoires currently not including the EURO SIGN: EN 1923:1997

The following clause should be included in the procurement specification in order to specify output repertoire(s) from EN 1923:1997

> "The product shall support the output repertoire(s) xxxx specified in EN 1923:1998."

where xxxx is one or more of the following:

BL, LL8, BG, BC, BL & BG, BL & BC, BL & BG & BC.

(This guide deprecates the use of the repertoires IVL and IL.)

Note – EN 1923:1997 may be revised to include the EURO SIGN.

11.3.2 8-bit character repertoires which include the EURO SIGN

The following clause should be included in the procurement specification in order to specify output repertoire(s) including the EURO SIGN.

> "The product shall support the output repertoire(s) specified xxxx."

where xxxx is one or more of ISO/IEC 8859-15:1998, ISO IR 203, ISO IR 204, ISO IR 205, and ISO IR 206.

11.3.3 16-bit character repertoires: UCS collection identifiers

The following clause should be included in a procurement specification in order to select output repertoire (s) using collection identifiers in ISO/IEC 10646-1.

> "The product shall support the output repertoire(s) identified in ISO/IEC 10646-1 collection identifiers nnnn."

where nnnn is one or more of CId(MES-1), CIdMES-2, CId(MES-3A) and CId(MES-3B).

Note – If more than one output repertoire is chosen, the following clause should be included in the procurement specification.

> "The product shall be configured for use of the specified output repertoires together with applications as follows:
>
> Repertoires xxxx together with application(s) zzzz."

Where xxxx are the repertoires as specified in preceding clauses and zzzz is (are) the name(s) of application(s) as appropriate.

11.3.4 Fall-back and other output transformation functions

There may be cases where the processing repertoire contains characters that cannot be rendered by the output repertoire. If any particular fall-back functions or other output transformation functions are requested, one of the following clauses should be included.

> "If the processing repertoire contains characters that cannot be rendered by the output repertoire, each such character should be represented in such a way as to indicate that it is not the original character."

or

> "If the processing repertoire contains characters that cannot be rendered by the output repertoire, each such character should be represented in such a way as to indicate that it is not the original character and also in such a way as to make it possible for the end user to identify that original character, e.g. by way of identifying its coded representation."

11.4 Processing character repertoire

11.4.1 8-bit character repertoires currently not including the EURO SIGN: EN 1923:1997

The following clause should be included in the procurement specification in order to specify processing repertoire(s) from EN 1923:1997

> "The product shall support the processing repertoire(s) xxxx specified in EN 1923: 1998."

where xxxx is one or more of the following:

BL, LL8, BG, BC, BL & BG, BL & BC, BL & BG & BC.

(This guide deprecates the use of the repertoires IVL and IL.)

Note – EN 1923:1997 may be revised to include the EURO SIGN.

11.4.2 8-bit character repertoires which include the EURO SIGN

The following clause should be included in the procurement specification in order to specify processing repertoire(s) including the EURO SIGN.

> "The product shall support the processing repertoire(s) specified xxxx."

where xxxx is one or more of ISO/IEC 8859-15:1998, ISO IR 203, ISO IR 204, ISO IR 205, and ISO IR 206.

11.4.3 16-bit character repertoires: UCS collection identifiers

The following clause should be included in the procurement specification in order to specify processing repertoire(s) using collection identifiers in ISO/IEC 10646-1.

> "The product shall support the processing repertoire identified in ISO/IEC 10646-1 collection identifiers nnnn."

where nnnn is one or more of CId(MES-1), CIdMES-2, CId(MES-3A) and CId(MES-3B).

Note – If more than one processing repertoire is chosen, the following clause should be included in the procurement specification.

> "The product shall be configured for use of the specified processing repertoires together with applications as follows:
>
> Repertoires xxxx together with application(s) zzzz."

where xxxx are the repertoires as specified in preceding clauses and zzzz is (are) the name(s) of application(s) as appropriate.

11.5 Interchange character repertoire

11.5.1 8-bit character repertoires currently not including the EURO SIGN: EN 1923:1997

The following clause should be included in the procurement specification in order to specify interchange repertoire(s) from EN 1923:1997.

> "The product shall support the interchange repertoire(s) xxxx specified in EN 1923:1998."

where xxxx is one or more of the following:

BL, LL8, BG, BC, BL & BG, BL & BC, BL & BG & BC.

(This guide deprecates the use of the repertoires IVL and IL.)

Note – EN 1923:1997 may be revised to include the EURO SIGN.

11.5.2 8-bit character repertoires which include the EURO SIGN

The following clause should be included in the procurement specification in order to specify interchange repertoire(s) including the EURO SIGN.

> "The product shall support the interchange repertoire(s) specified xxxx."

where xxxx is one or more of ISO/IEC 8859-15:1998, ISO IR 203, ISO IR 204, ISO IR 205, and ISO IR 206.

11.5.3 16-bit character repertoires: UCS collection identifiers

The following clause should be included in the procurement specification in order to specify interchange repertoire(s) using collection identifiers in ISO/IEC 10646-1.

> "The product shall support the interchange repertoire(s) identified in ISO/IEC 10646-1 collection identifiers nnnn."

where nnnn is one or more of CId(MES-1), CIdMES-2, CId(MES-3A) and CId(MES-3B).

Note – If more than one interchange repertoire is chosen, the following clause should be included in the procurement specification.

> "The product shall be configured for use of the specified interchange repertoires together with applications as follows:
>
> Repertoires xxxx together with application(s) zzzz."

where xxxx are the repertoires as specified in preceding clauses and zzzz is (are) the name(s) of application(s) as appropriate.

11.5.4 Fall-back and other transformation functions

There may be cases where the processing repertoire in the product when sending contains characters that are not contained in the interchange repertoire. If any particular fall-back or other transformation functions are requested,

one of the following clauses should be included.

> "If the processing repertoire in the product when sending contains characters that are not contained in the interchange repertoire, each such character should be represented during interchange in such a way as to indicate to the receiving system that it is not the original character."

or

> "If the processing data in the product when sending contain characters that are not contained in the interchange repertoire, each such character should be represented during interchange in such a way as to indicate to the receiving system that it is not the original character and also in such a way as to make it possible for the end user to identify that original character, e.g. by way of identifying its coded representation."

There may also be cases where the interchange repertoire contains characters that are not contained in the processing repertoire in the product when receiving. If any particular fall-back or other transformation functions are requested, one of the following clauses should be included.

> "If the interchange repertoire contains characters that are not contained in the processing repertoire in the product when receiving, each such character should be represented in the latter repertoire in such a way as to indicate to the processing function that it is not the original character."

or

> "If the interchange repertoire contains characters that are not contained in the processing repertoire in the product when receiving, each such character should be represented in the latter repertoire in such a way as to indicate to the processing function that it is not the original character and also in such a way as to make it possible for the end user to identify that original character, e.g. by way of identifying its coded representation."

11.6 Additional requirements when using the 8-bit code structure for interchange

If an interchange repertoire has been selected from EN 1923:1997 and the 8-bit code structure is required for interchange, the following clause should be included in the procurement specification.

> "For the specified interchange repertoires, the product shall support the 8-bit code structure requirements as specified in EN 1923:1998."

If the user is particularly concerned about the achievement of interoperability in the 8-bit code structure environment, the following clause should be included in the procurement specification.

> "The product shall satisfy the conformance requirements of ISO/IEC ISP 12070-1:1996 for operation of the 8-bit code structure."

11.7 Additional requirements when using the multi-byte UCS code structure for interchange

The following requirements should be specified in order to create an environment which maximises interoperability where a general purpose implementation is sought. In many cases interoperability is achieved through the procurement of specific applications which take care of this issue. For such cases, these additional requirements do not apply.

The approach taken is to ask for a minimum requirement on senders and a maximum requirement on receivers.

11.7.1 Levels and coding form

If MES-1 or MES-2 is chosen as the interchange repertoire, the following clause should be added to the procurement specification.

> "For sending, the product shall support at least the level-1 operation using at least the UCS-2 form as specified in ISO/IEC 10646-1:1993."

If the *MES-3A or MES-3B* is chosen as the interchange repertoire, the following clause

should be added to the procurement specification.

> "For sending, the product shall support the level-3 operation using at least the UCS-2 form as specified in ISO/IEC 10646-1: 1993."

Regardless of which repertoire is specified, the following clause should be included.

> "For receiving, the product shall support the level-3 operation using at least the UCS-2 form, the UCS-4 form and the UTF-8 transformation format as specified in ISO/IEC 10646-1:1993."

11.7.2 Ordering of octets

In the UCS code structure, characters are generally represented by multiple octets. The order in which those octets are sent affects interoperability. The following clauses should be included in the procurement specification.

> "For sending, the product shall support at least the normal ordering of octets for each character sent as specified in ISO/IEC 10646-1:1993."

> "For receiving, the product shall support both the normal ordering and the reverse ordering of octets for each character sent as specified in ISO/IEC 10646-1:1993."

11.7.3 Signatures

For sending unstructured text information, the appropriate signatures should be used so that the receiver may understand what format is being used and in which order character octets are being sent. The following clause should be included in the procurement specification.

> "For sending, the product shall support the use of the appropriate signatures as specified in ISO/IEC 10646-1:1993."

For receiving unstructured text information, all the appropriate signatures should be accepted so that the receiver may understand what format is being used and in which order character octets are being sent. The following clause should be included in the procurement specification.

> "For receiving, the product shall support the use of all signatures as specified in ISO/IEC 10646-1:1993."

12 CEN and CEN/TC 304

CEN (Comité Européen de Normalisation) is the organisation responsible for standardisation in Europe. Details of CEN and its operations may be found at http://www.cenorm.be. TC 304 is a technical committee within CEN responsible for European Localisation Requirements to give its proper title.

The work of TC304 is standardisation in the field of information and communication technologies as applied to character sets and related cultural elements (such as date and time conventions, numeric formatting), to ensure that European localisation requirements are satisfied. The work concerns the areas of identification, manipulation and coded representation of character data and its input, interchange and rendition by electronic means.

Within this scope, TC304 is currently active in the following areas:

- Coordinating the introduction of the EURO SIGN character into a range of standards in support of the EMU.
- Developing character repertoires and coded character sets for use in Europe by profiling existing international standards.
- Developing a catalogue of the character repertoires used for indigenous European languages.
- Developing a model for repertoire and code transformations together with rules for specific transformations such as fall-back requirements.
- Providing guidance on the application of international keyboard standards in national standards in a European context.
- Developing ordering rules for European repertoires by profiling existing international standards.

- Providing guidance on character string matching in a European context (for instance for use in search engines).

- Developing a check list for European specific requirements for the localisation of ITC products and services for use in Europe.

- Identifying and registering, for different European communities, cultural elements such as string representations reflecting date and time conventions, currency notations and other culture dependent aspects of IT in Europe.

It is anticipated that, in the near future, TC304 will become active in the area of Language Technology by encouraging the promotion of specifications resulting from European Language Technology research and development projects. Some of these specifications may develop into European or even international standards.

More information concerning TC304 may be found at http://www.stri.is/tc304.

13 References

At the time of publication, the editions of standards indicated were valid. All standards are subject to revision. National Standards Organisations maintain registers of currently valid standards.

13.1 ISO/IEC standards

ISO/IEC 646:1991, *ISO 7-bit coded character set for information interchange*

ISO/IEC 2022:1994, *Information technology – Character code structure and extension techniques*

ISO 2375:1985, *Data processing – Procedure for the registration of escape sequences*

ISO/IEC 4873:1991, *Information technology – ISO 8-bit code for information interchange – Structure and rules for implementation*, is a refinement of ISO/IEC 2022

ISO/IEC 6429:1992, *Information technology – Control functions for coded character sets*

ISO/IEC 6937:1994, *Information technology – Coded graphic character set for text communication – Latin alphabet*

ISO/IEC 8859:nnnn, *Information technology – 8-bit single-byte coded graphic character set*

ISO/IEC 8859-1:1998, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 1: Latin alphabet No. 1*
ISO/IEC 8859-2:1999, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 2: Latin alphabet No. 2*

ISO/IEC 8859-3:1999; *Information technology -- 8-bit single-byte coded graphic character sets -- Part 3: Latin alphabet No. 3*

ISO/IEC 8859-4:1998, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 4: Latin alphabet No. 4*

ISO/IEC 8859-5:1999, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 5: Latin/Cyrillic alphabet*

ISO/IEC 8859-6:1999, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 6: Latin/Arabic alphabet*

ISO 8859-7:1987, *Information processing -- 8-bit single-byte coded graphic character sets -- Part 7: Latin/Greek alphabet*

ISO/IEC 8859-8:1999, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 8: Latin/Hebrew alphabet*

ISO/IEC 8859-9:1999, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 9: Latin alphabet No. 5*

ISO/IEC 8859-10:1998, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 10: Latin alphabet No. 6*

ISO/IEC DIS 8859-11, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 11: Latin/Thai character set*

ISO/IEC 8859-13:1998, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 13: Latin alphabet No. 7*

ISO/IEC 8859-14:1998, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 14: Latin alphabet No. 8 (Celtic)*

ISO/IEC 8859-15:1999, *Information technology -- 8-bit single-byte coded graphic character sets -- Part 15: Latin alphabet No. 9*
ISO/IEC 10367:1991, *Information technology – Standardised coded graphic character sets for use in 8-bit codes*

ISO/IEC 10646-1:1993, *Information technology – Universal Multiple-Octet Coded Character Set (UCS), Part 1: Architecture and Basic Multilingual Plane*

ISO/IEC ISP 12070:1996, *Information technology – International Standardised Profiles FCSnnn – Character set 8-bit code structure based on ISO/IEC 2022 – Part 1: FCS111 – 2022 Option 1*

ISO/IEC TR 15285:1998, *Information technology -- An operational model for characters and glyphs*

13.2 European standards

EN 1923:1998, *European Character Repertoires and their Coding – 8 bit single byte coding*

13.3 ITU-T Standards

ITU-T Recommendation X.400/F.400 (07/96) – *Message handling system and service overview*

ITU-T Recommendation X.500 (08/97) – *Information technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services*

ITU-T Recommendation T.50 (09/92) – *Information technology – 7-bit coded character set for information interchange*

ITU-T Recommendation T.51 (09/92) – *Latin based coded character sets for telematic services*

Amendment 1 to ITU-T Recommendation T.51 (08/95) – *Latin based coded character sets for telematic services*

ITU-T Recommendation T.52 (03/93) – *Non-Latin coded character sets for telematic services*

Amendment 1 (10/96) to ITU-T Recommendation T.52 – *Non-Latin coded character sets for telematic services*

Summary of Amendment 1 (10/96) to ITU-T Recommendation T.52

ITU-T Recommendation T.53 (04/94) – *Character coded control functions for telematic services*

13.4 ETSI standards

13.4.1 ERMES

ETS 300 133-1 ed.1 (1992-07) – 7 parts – *Paging Systems (PS); Enhanced Radio Message System (ERMES); Part 1: General aspects*

13.4.2 GSM

There are very many GSM standards. The pertinent one concerning character sets is:

GTS GSM 03.38 V5.3.0 (1996-07) – *Digital cellular telecommunications system (Phase 2+); Alphabets and language-specific information*

13.5 Unicode

The Unicode Standard, Version 2.0
Addison-Wesley Developers Press, 1996
ISBN 0-0201-48345-9

Note – This standard defines a character code for code identical to that contained in ISO/IEC 10646-1:1993 as amended by the amendments 1 through 7.

The Unicode Standard, Version 2.1
The Unicode Consortium, 1998
Available at http://www.unicode.org/unicode/reports/tr8.html.

Version 2.1 (defined by the published book as

the Unicode Technical Report #8, available on-line) contains two more characters from Amendment 18 to 10646-1: U+20AC EURO SIGN and U+FFFC OBJECT REPLACEMENT CHARACTER, both of which are key, required characters for vendors and for the software that procuring agencies would want to acquire.

13.6 Other non-standard references

Manual: Standards for the electronic interchange of personal data: Part 5. Character sets (1995)
Author: J W van Wingen
ISBN 90-5414-019-4
Published by the Directorate of Departmental Relations and Provision of Information
Directorate-General for Public Information
Ministry of the Interior
P.O. Box 20011
250 EA The Hague
Netherlands

This manual was produced for the Ministry of the Interior in the Netherlands. It provides the reader with an overview of character sets and their standards which in some respects is more extensive than this guide. It also makes certain recommendations targeted at the use of character sets for administrative purposes within the public sector in the Netherlands. It provides a view of some of the issues on the use of character sets and further detailed technical information.

Comparisons of Standardized Character Sets for Europe (1996)
Author: K I Larsson
ISBN 91-7220-275-0
Published by Statskontoret
The Swedish Agency for Administrative Development
Box 2280, 103 17 Stockholm
Sweden
Orders may be directed by e-mail to publikations.service@statskontoret.se

This document was produced for Statskontoret in Sweden. It provides in tabular form a detailed comparison of character sets, both standardised and manufacturer defined. The character sets are mainly 8-bit coded sets and the coverage is comprehensive covering most of the Latin based sets likely to be encountered in Europe. It provides a useful source of code table/code page specification, especially of the manufacturer code pages.

Language automation world-wide: The development of character set standards (19xx).
ISBN: 1-870095-0-4
Author: John Clews
Publisher: SESAME Computer Projects
8 Avenue Rd
Harrogate HG2 7PG
United Kingdom
Telephone: +44 (0) 1423 888 432
Email: European@sesame.demon.co.uk

This document covers all scripts in use world-wide, with a considerable emphasis on character sets for European scripts.

IBM Character Data Representation Library Character Data Representation Architecture Level 2 – Reference (1993)
Publication Number SC09-1390-01

To quote: “The overall objective of CDRA is to define a method of assigning and preserving the meaning and rendering of coded graphic characters through various stages of processing and interchange.” This is a comprehensive architectural document which provides a basis for consistent implementation of character set support across a range of IBM platforms. It is very detailed but will reward the dedicated reader who is intent on researching character set issues in depth from an implementation point of view.

IBM National Language Support Reference Manual Volume 2 – National Language Design Guide (1994)
Publication Number SE09-8002-03

To quote: “This guide is directed at developers, planners, and vendors of computer products intended for international markets”. Appendix I contains details of IBM code pages, both for EBCDIC and for PCs.

Letter Database
Indrek Hein
http://www.eki.ee/letter/

This website contains a comprehensive set of information relating to character sets. Some of this is accessed by internal links and the remainder by search engine.

Annex A

8-Bit Character Sets

This Annex to the Guide to the Use of Character Sets in Europe provides more detailed information about 8-bit character set standards than is found in the main body of the Guide. Annex B deals in more detail with the Universal Multi-octet Coded Character Set (UCS) specified in ISO/IEC 10646-1.

The need to represent characters by bit combinations (binary numbers) is central to the storage and processing of data by computer systems and the interchange of data between such systems. This annex gives guidance on the many standards and other specifications which have been developed to address the issues that arise from this need up and until the advent of the multi-octet code structure embodied in ISO/IEC 10646-1:1993.

Table of Contents

1 More about this annex

The requirement for compatibility between newer and older equipment has led to the standards of the present day containing legacies from decisions taken many years ago. The reasons behind those decisions are often no longer relevant and their present day legacies may appear merely as unnecessary oddities and complexities. This annex includes some historical background which however is not necessary for an understanding of the remainder of the text.

As work on character sets has developed, there has been a gradual refinement of the concepts involved. This has led to character set standards and other literature making use of technical terms that can be a barrier to the reader. It may be helpful to read section 1.2, Concepts and terminology, before exploring the remaining sections in detail.

This gradual evolution of character set standards has led to technical innovations designed to increase the capabilites of coded character sets while remaining backwardly compatible with what has gone before. Within this evolved framework it is now possible to support a wide range of languages. The wider the range that it is required to support simultaneously, however, the more complex is the technical innovation required. For further information see section 3.1, Language support.

Not all the technical innovations are compatible with all the ways that character data may be used by applications. Section 5.1, Application Environments, provides guidance on these limitations.

Other sections provide greater detail on particular issues.

2 Limitations of this annex

This annex does not cover, or only briefly covers, the following topics:

- The new multiple-octet coded character set standard ISO/IEC 10646;
- The use of coded character sets in Open Systems Interconnection (OSI) standards;
- The presentation of character data on display devices or as printed images;
- Transformations between different coding methods;
- Sorting (collating) of coded character set data.

3 User Requirements

The ultimate user of equipment that makes use of coded character sets is concerned with such matters as which languages it supports, whether page layout information can be preserved during interchange of documents, and other matters of a similar nature. This section is concerned with the facilities available in character set standards to meet such requirements.

3.1 Language Support

One of the prime requirements in the use of character sets is to be able to support the languages of concern to the user. A number of different International Standards have been developed to provide multilingual support. In addition, other languages are supported by the character sets of the International Register of Coded Character Sets to be used with Escape Sequences.

The following is an index to the various sections containing information on the languages supported by such standards and register entries.

- For variants of the ASCII 7-bit character set designed primarily to support one individual language in a Latin script, see 6.1.1, ISO/IEC 646. See the historical introduction in 4.1 to find out more about the origins of ASCII.

- For multilingual support obtained by supplementing ASCII to create an 8-bit code, see 6.1.8, ISO/IEC 8859. This standard permits simultaneous support either for a range of languages in the Latin script, or for basic Latin letters (A-Z and a-z, but not þ, ø, etc. or accented letters) together with an alphabet in any one of the Greek, Cyrillic and other scripts.

- For the greatest support of languages in the Latin script that can be achieved with a single 8-bit code, see 6.1.6, ISO/IEC 6937. This is achieved by the use of non-spacing diacritical marks, so permitting more characters to be represented than there are positions in the code table. This complication has its price; see the guidance on application environments in 5.1.

- To achieve the level of support of Latin languages that is provided by ISO/IEC 6937 but without the use of non-spacing diacritical marks, see 6.1.9, ISO/IEC 10367. This is achieved by the use of locking shifts. This complication also has its price; again see the guidance on application environments in 5.1.

- To support up to three of the four scripts Greek, Cyrillic, Hebrew and Arabic simultaneously with basic use of the Latin alphabet, or up to two of them simultaneously with a wide range of Latin languages, see 6.1.9, ISO/IEC 10367. This is achieved by the use of both locking shifts and non-spacing diacritical marks. The provisos mentioned in both the preceding entries in this list then apply.

- To support Chinese, Japanese and Korean ideographic scripts, see 6.2.1 on the International Register.

It may be helpful to read the introduction to concepts and terminology in 4.2 before following some of the above references.

3.2 Page and Display Formats

Text that is communicated by the use of coded character sets is usually intended ultimately for presentation on a screen or printed page. There is a need to be able to communicate, with that data, information concerning the layout of the text on the screen or page.

Such layout information may be either

- embedded in the communicated text, or

- separated from the text as elements of some communication protocol.

Two standards are available that provide coded control functions for embedding layout information in communicated text.

- Control functions applicable to character-imaging devices in general are specified in ISO/IEC 6429; see 6.1.5. This standard includes facilities that permit texts in different scripts to he presented in opposite directions, such as mixed Latin and Arabic, or Latin and Hebrew.

- Control functions intended specifically for the control of page layout are specified in ISO/IEC 10538; see 6.1.10. This standard includes facilities both for fixed format and for

automatically reformattable text. The former assumes that both sender and receiver have the same fonts available. The latter permits reformatting where sender and receiver have access to different fonts.

3.3 European Requirements

There are various published sources of guidance on the use of character set standards in Europe. These include:

- Guidance on European character repertoires for use with 8-bit single-byte coding, given in EN 1923; see 6.3.2.
- Guidance on the interchange of character data by means of the Telex network in Europe, given in EN 1922; see 6.3.1.
- Guidance on the use of character sets in Europe in connection with OSI Abstract Syntax Notation One (ASN.1), given in ISO/IEC ISP 12070; see 6.1.12.

4 Introduction to character sets

Formal specifications for coded character sets are often couched in a language of escape sequences, ISO-IR numbers, CL and GL areas, C0 and G0 code elements and more. There are 7-bit and 8-bit codes and subtle distinctions between 94-character and 96-character sets. This annex cannot escape from the use of such language but this introduction provides an explanation for the novice.

4.1 Historical background

The assignment of specific bit combinations to a particular set of characters constitutes a coded character set, or more concisely, a code. The larger the set of characters, the greater the number of bits required for the coding. Any increase in the number of bits causes a corresponding increase in cost in the systems that use the resulting code. This need not be a monetary cost, it may instead be a cost in terms of resources, but it is nevertheless real. The historical development of coded character sets is a story of balancing the desire for more characters against that for the fewest possible number of bits. The decisions that were made have had consequences that have lasted long after the pressures that led to them have eased. This section gives some account of that history.

4.1.1 The first binary codes

4.1.1.1 *The legacy of Baudot*

The first binary coded character set was a 5-bit code patented by Jean-Maurice-Émile Baudot (1845-1903) in 1874 in connection with his invention of a precursor of the teleprinter. Since the device was operated by electromechanical means, even one further bit would have added significantly to the complexity of the equipment. In 1932 the CCITT (Comité Consultatif International Télégraphique et Téléphonique) standardized a 5-bit code for teleprinters, based on that of Baudot, which is the code of the international telex (teleprinter) network to the present day. This is known as the International Telegraphic Alphabet No.2, also as CCITT code No.2 or simply as the Baudot code. It was last re-issued as ITU-T Recommendation S.1 (1993).

4.1.1.2 *Locking shifts*

A 5-bit code has room for 32 characters, which is not enough even for the 26 letters A to Z and the ten digits. To get round this, a teleprinter operated by the Baudot code has a shift lock in the manner of a typewriter. This locks 26 “keys”, i.e. bit combinations, into one of two modes. In the alphabetic mode they print the letters A to Z. In the numeric mode some of these bit combinations print the ten

digits 0 to 9 and various punctuation marks while the remainder operate certain functions such as line feed, carriage return and sounding a bell. The effect of the remaining 6 bit combinations is not affected by the shift lock. Two of these six are used to switch the shift lock between the two modes. In this way the 5-bit code conveys 58 different signals (26 times 2, plus 6).

4.1.1.3 *National variants*

Right from the beginning it was recognised that different countries had different needs. Although 58 character positions does not give much room for flexibility, the 1932 standard for the Baudot code filled only 55 of the available positions. The remaining three character positions were then available for national use.

4.1.2 ASCII

4.1.2.1 *A 7-bit code*

In the late 1950's the American Standards Association (now the American National Standards Institute) set about the development of a new code for the communications and data processing industries. By then, there was a need for further character positions to be available, both for printing and control purposes. To avoid the need for shift operations, it was agreed to develop a 7-bit code. This has 128 bit combinations available. The code was to be known as the American Standard Code for Information Interchange, or ASCII.

4.1.2.2 *The legacy of paper tape*

By that time it was common to store data on punched paper tape, for input to data processing systems or automated communication equipment. A “1” bit was represented by a hole and a “0” bit by the absence of a hole. Since a row of 7 zero bits would be indistinguishable from blank tape, the coding “0000000” would have to represent a NULL character (absence of any effect).

Since holes, once punched, could not be erased but an erroneous character could always be converted into “1111111”, this bit pattern was adopted as a DELETE character. When received, or otherwise processed, it again was to have no effect but it could be punched on top of any other character to erase that character's effects.

4.1.2.3 *94 characters*

A design decision was taken to reserve the first 32 bit combinations (i.e. with the two most significant bits being 0) for control characters. This range includes the NULL character but not the DELETE character, so it leaves 95 bit combinations for printing characters. The printing characters (including SPACE) were to be arranged in an order that could be used for sorting purposes. The SPACE character is normally sorted before any other printing character and so was allocated the first position among the printing characters. There are then 94 contiguous positions for printing characters between “1100000” (SPACE) and “1111111” (DELETE).

This division of the code positions into 32 for control starting with NULL, followed by 94 printing characters lying between SPACE and DELETE, has dominated the structure of coded character sets right to the present day; see 4.1.4 below.

The first ASCII standard was published in 1963, but at that time it left many bit combinations unallocated. It included capital letters but not small letters. The ASCII standard as we know it today dates from 1968.

4.1.2.4 *Built-in extendability*

Although the use of a 7-bit code for ASCII was designed to avoid the use of the locking shifts of the Baudot code, a pair of locking shift codes SHIFT IN (SI) and SHIFT OUT (SO) were included in the

set of control characters to allow for future extension. An ESCAPE character was also included, to act as a non-locking route to extension.

4.1.2.5 *International adoption*

The later stages of the ASCII story were joint developments with the ISO subcommittee ISO/TC97/SC2. This led to the publication in 1967 of ISO Recommendation 646 (ISO had Recommendations rather than Standards at that time). Just as with the Baudot code, the need for national variants was recognised. With the greater freedom offered by 94 printing characters, 10 positions were reserved for national use. To ensure maximum consistency when these 10 positions were not all required, the recommendation provided a default assignment of characters to these positions. The version in which all the default assignments were used was known as the International Reference Version (IRV).

The most recent edition of this standard is the third edition, ISO/IEC 646:1991, which superseded the second edition of 1983. In these editions there are still 10 positions for national use and in addition a further two that have two alternative graphics assigned (number sign versus pound sign, dollar sign versus a generic currency sign). Both these and the 10 national use positions have specific assignments in the IRV. However, it is important to be aware that the IRV of ISO/IEC 646 was changed between the 1983 and 1991 editions. To conform with *de facto* usage, the 1991 edition recognised ASCII as the new IRV. The IRV of the 1983 edition specified the generic currency sign alternative in the choice between that and the dollar sign.

4.1.3 The world after ASCII

4.1.3.1 *7-bit codes in an 8-bit world*

The 7-bit codes that followed ASCII, such as for other scripts (Greek, Arabic, etc.), followed the basic structure of ASCII. They kept the 32 control characters, SPACE and DELETE and changed only the remaining 94 printing characters. When 7-bit codes were used in the 8-bit environment provided by most computers, the most significant bit was set to “0”. This leaves the NULL character being “00000000” which is consistent with its original design intention but it codes the DELETE character as “01111111”, so no longer having all “1” bits.

4.1.3.2 *8-bit codes*

This approach led to a natural method of extension to accomodate more characters: use a second 94-character 7-bit code and distinguish it by setting the most significant bit to “1”. Such an extended code can be transmitted through 7-bit communication channels by use of the SI and SO locking shifts of ASCII. There is no need for second SPACE and DELETE characters, so these positions are unassigned in the second code. When 7-bit codes came to be designed specifically for use in this extended area, they could use the full 96 printing positions.

This design for 8-bit codes has some immediate consequences. Viewed in binary sequence the control characters are no longer contiguous; there are two control areas “000xxxxx” and “100xxxxx” and two separated areas for printing characters, a lower area with the most significant bit set to “0” and an upper area with it set to “1”. Between them they can accommodate 190 printing characters (94 plus 96) excluding SPACE.

4.1.3.3 *Locking shifts again*

The 8-bit codes described above have room for, say, accented letters or Greek letters in the upper half, but not both. The need to cater for both at once gave rise to an obvious further extension: locking shifts for 8-bit codes. As with any locking shift mechanism, it is only suitable for communication rather than for data processing, but this route was followed. A second set of two 7-bit codes could then be accommodated, to be shifted as required into the lower and upper areas for printing

characters. This gives a total of four 7-bit codes in use simultaneously. There is no need to restrict usage to one alternative code for the lower area and another for the upper, so mechanisms were set up to shift (invoke) any of the four 7-bit codes into the lower area and, independently, any other into the upper area.

4.1.3.4 *The International Register*

The limit of four 7-bit codes is in fact arbitrary; it would be possible to have even more 7-bit codes "on standby" and even more shift mechanisms for invoking them. But four is enough for most needs, and once this is exceeded there seems no particular reason to stop at any other number. The next stage in code extension was therefore to permit the choice of the four 7-bit codes to be changed by means of control functions. This is like sending a telephone message to a user of an interchangeable typehead ("golfball") typewriter to change the typehead.

This is more difficult as there has to be some method of identifying the new choice of code to the remote party to the communication. In effect there has to be a catalogue of typeheads from which one can choose, with catalogue numbers that one can communicate. This was achieved by means of an International Register of such codes, established by ISO. Each registered code is assigned a number and is referenced as ISO-IR xx. In this register, for example, the IRV of ISO 646:1983 is ISO-IR 2 and that of ISO/IEC 646:1991 (ASCII) is ISO-IR 6. This register enables new codes to be selected (designated) and subsequently shifted into use (invoked), all through the use of control functions.

The structure of character codes and their extension techniques that is described above is formalised in the international standard ISO/IEC 2022. The International Register is maintained according to procedures laid down in ISO 2375 and is published by Japanese standards institution (JISC) under the authority of ISO. The register may be accessed on-line. It currently contains over 200 registered coded character sets.

4.1.3.5 *Limits on expansion*

Not all implementations of communications protocols may be able to cope with all the possibilities of this complex system of code extension by designation and invocation. There needs to be a way of notifying the remote user of the intention to use only a selection of the available facilities. This was achieved by means of further control functions, known as announcers. These make it possible, for example, for a system to announce that it will only use a 7-bit code (or an 8-bit code) with no code extension facilities.

The summary of this section on the world after ASCII has skipped over a number of difficulties that arise in these code extension techniques. In particular, attention has been concentrated on the printing characters. The control characters also have their extension problems. An account with greater precision is given in section 4.2.

4.1.4 The future is 16-bit

With the growing processing power of computers and the increasing bandwidth of communications channels, the pressure to squeeze an ever increasing number of characters into an 8-bit code structure has diminished. A need has arisen for a simpler structure at the expense of more bits. This need has given rise to a complete rethinking of code structure for a world of 16-bit and even 32-bit processing and communication. From it has risen a new international standard, ISO/IEC 10646, the Universal Multiple-Octet Coded Character Set.

It is interesting to note that even this "ultimate" standard retains some past legacies. Control functions are coded according to ISO/IEC 2022, although the code extension functions of that standard are forbidden. The first 32 bit combinations are therefore reserved for control purposes. The next 95 bit combinations contain the printing characters of ASCII including SPACE. This brings one to the bit

combination “00...001111111” (the dots denote enough zeroes to fill either 16 or 32 bits, as the case may be). The legacy of paper tape survives. This is still reserved for the DELETE character!

It is the intention that ISO/IEC 10646 will be, in some sense, the last character set standard. It is planned as a multi-part standard, of which part 1 was published in 1993. Future parts will add to the code, and since it has the potential to fill a 32-bit code space, it has the capacity to be extended to meet all foreseeable future needs. It has the ultimate aim of including all characters that have ever been used for communication. The coding of ancient runes has already been standardized, that of Egyptian hieroglyphics is for future study.

More detailed information may be found in Annex B, which deals specifically with the UCS code structure.

4.2 Concepts and terminology

Many of the 7-bit and 8-bit coded character sets in use today share a common structure. This structure, together with notation and terminology for referring to its various elements, is laid down in ISO/IEC 2022. Some familiarity with the main features of that structure is needed to read this annex. These are summarised in this section.

The Universal Multiple-Octet Coded Character Set (UCS) that has been developed more recently, specified in ISO/IEC 10646-1, lies outside the ISO/IEC 2022 structure. Detailed guidance on the structure of the UCS is found in Annex B.

4.2.1 Basic principles of ISO/IEC 2022

The construction of a character code according to ISO/IEC 2022 is most simply explained by a mechanical analogy. It is like a typewriter that takes interchangeable typeheads (“golfballs”). A typewriter without a typehead can't actually type anything but its mechanisms are all in place. The non-printing keys, such as the space bar and backspace, still operate. It is only the printing characters that are missing.

The typehead itself is an inert object, but once placed on the typewriter then each key on the typewriter will print the character that is at a specific position on the typehead. Change the typehead and the typewriter prints different characters, but the relationship between keys and character positions does not change.

The role of the typewriter is taken in ISO/IEC 2022 by a code table. There is one code table for 7-bit codes and another for 8-bit codes. Each code table provides a linkage between character positions and bit combinations. Certain of these positions are already assigned, for the SPACE, DELETE and ESCAPE characters, but the vast majority of character positions are empty. The table is waiting for its equivalent of a typehead.

The role of the typehead is taken in ISO/IEC 2022 by a code element of graphic characters. Such a code element contains a pattern of graphic characters that can be overlaid on (part of) the empty code table. Once overlaid, it provides a graphic character at each of the overlaid positions. The combination of code table and code element completes (part of) the code; the character at a particular position is coded by the bit combination assigned to that position.

The next few paragraphs expand on this model in a more precise way.

4.2.2 Code tables

4.2.2.1 *Layout and notation*

ISO/IEC 2022 defines the structure of a code table separately for 7-bit and for 8-bit codes. A 7-bit code table consists of 128 positions arranged in 8 columns and 16 rows. An 8-bit code table consists of 256 positions arranged in 16 columns and 16 rows. The rows and columns are numbered starting from 0, and by convention a leading zero is included where necessary to make all row and column numbers have two (decimal) digits. A diagram to illustrate the 8-bit case is given below in figure 1.

The notation *xx*/*yy*, e.g. 01/15, is used to label the table position that is in column *xx* and row *yy*. The same notation is used to identify a bit combination, with *yy* being the decimal number whose binary form consists of the least significant four bits of the bit combination and *xx* being similarly related to the most significant four (for an 8-bit code) or three (for a 7-bit code) bits. This notation provides a natural correspondence between positions in the code table and bit combinations of the code.

4.2.2.2 *Structure*

The 8-bit code table is divided into four named areas:

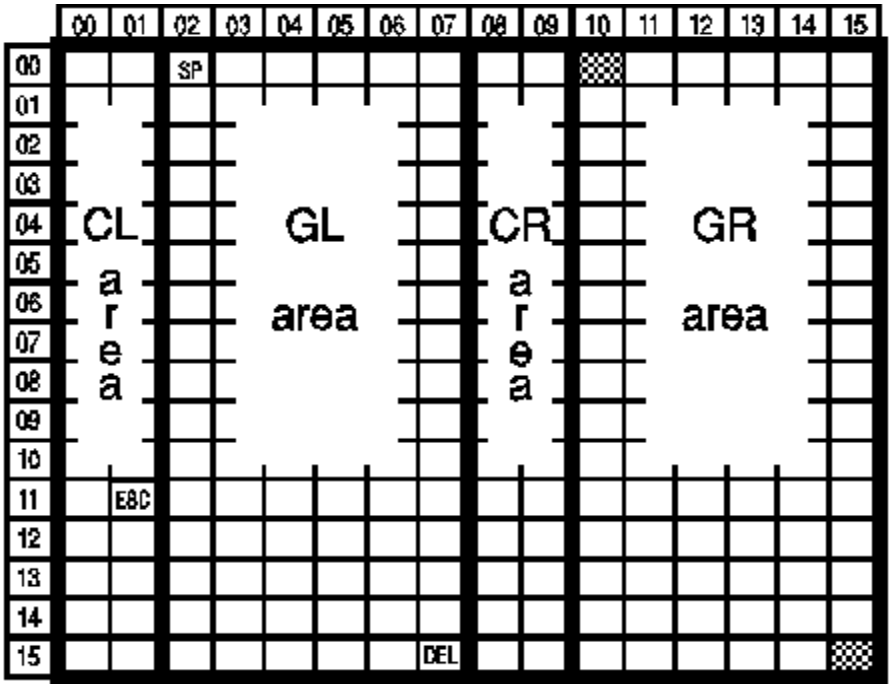

Figure 1: Structure of an 8-bit code table

- Columns 00 and 01 are called the CL area;
- Columns 02 to 07 are called the GL area;
- Columns 08 and 09 are called the CR area;
- Columns 10 to 15 are called the GR area.

The 7-bit code table is similarly divided, but it only has CL and GL areas. The 8-bit code table is illustrated in figure 1.

ISO/IEC 2022 requires that the bit combinations in the CL and CR areas shall be used to represent control functions or be left unused. Only those in the GL and GR areas may be used to represent graphic (printing) characters.

Certain characters have fixed fixed assignments in both the 7-bit and 8-bit code tables as follows:

- 01/11 is assigned to the ESCAPE character (acronym: ESC);
- 02/00 is assigned to the SPACE character (acronym: SP);
- 07/15 is assigned to the DELETE character (acronym: DEL).

These are also shown in the figure. The reasons behind the assignments for SPACE and DELETE are described in 4.1.2.

4.2.2.3 *Escape sequences*

The third of the characters with fixed assignments is the ESCAPE character. This is also in the position in which it was put during the design of ASCII. However, the reason that the ESCAPE character is so important as to require permanent assignment is more recent.

As the development of coded character sets has progressed, there has become an increasing need to be able to code control information that contains parameters. To achieve this, the concept of a control character such as CARRIAGE RETURN (CR) has given way to the more general concept of a control function. The coding of a control function is introduced by a distinctive bit combination, but it is followed by further bit combinations that pass parameter information in coded form. The syntax of these further combinations ensures that they are self-delimiting, i.e. that the end of the coding of the control function may be identified by a suitable algorithm. The ESCAPE character is one such introducer. The complete sequence of bit combinations that represents a control function coded in this manner is known as an escape sequence.

More details of the coding of escape sequences are given in section 5.3.

4.2.3 Code elements

ISO/IEC 2022 constructs a complete code from a selection of the following code elements:

- Four code elements named G0, G1, G2 and G3 containing graphic characters, arranged as possible overlays for the GL and GR areas of the 8-bit code table;
- Two code elements named C0 and C1 containing control characters, arranged as possible overlays for the CL and CR areas of the 8-bit code table;
- A number, possibly zero, of other control functions.

All of these elements may be present in either a 7-bit or an 8-bit code. Each of these types of code element will now be considered in more detail.

4.2.3.1 *Code elements G0, G1, G2 and G3 of graphic characters*

The code elements G1, G2 and G3 may each provide assignments for either 94 or 96 character positions. A set with 94 positions would provide assignments for positions 2/1 to 7/14 of the GL area or 10/1 to 15/14 of the GR area, i.e. excluding the positions assigned to SP and DEL in the GL area and the two corresponding shaded positions in the diagram above for the GR area. A set with 96 positions would provide assignments for all the 96 positions of either the GL or GR area. The code element G0 is similar but only the 94-position option is permitted.

Below is a diagram of a 94-position code element that is suitable for use as any of G0 to G3. It is in fact the ASCII character set:

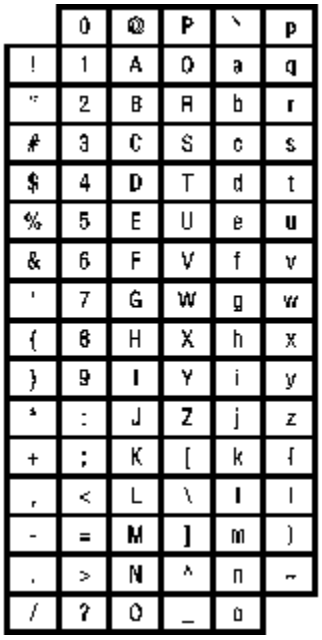

Figure 2: A 94-position code element

The process described above as overlaying is known technically, in ISO/IEC 2022 terminology, as invocation. After being invoked, the code element concerned is said to have GL or GR shift status, as the case may be. Clearly at most two of the code elements G0 to G3 may be invoked simultaneously in an 8-bit code, one in each of the GL and GR areas, and at most one in a 7-bit code where there is no GR area. The mechanism of invocation is described in more detail below.

For the code element illustrated, when it has GL shift status the character "A" is represented by the bit combination 04/01. When it has GR shift status it is represented by the bit combination 12/01.

More details of the use of the code elements of graphic characters are given elsewhere in this annex.

4.2.3.2 *Code elements C0 and C1 of control characters*

Control characters have a name and an identifying acronym, but no graphic representation. Examples of control characters are BACKSPACE (BS), BELL (BEL), START OF HEADING (SOH), SINGLE-SHIFT 2 (SS2) and ESCAPE (ESC). They are a special case of a more general concept, the control function, as explained above concerning escape sequences.

The code elements C0 and C1 each provide assignments to control characters for the 32 character positions of either the CL or CR area. If the code has a C0 code element then this is permanently invoked in the CL area. A C0 code element is required to have the ESCAPE character in position 01/11 so that its invocation does not affect the availability or coding of this control character.

If an 8-bit code has a C1 code element, it would normally be permanently invoked in the CR area. This is not possible for a 7-bit code since there is no CR area in the code table. Instead, the characters of the C1 code element are represented in a 7-bit code by means of an escape sequence. This representation is also permitted for an 8-bit code, as an alternative to invocation in the CR area. In a

particular code, only one of the two alternatives is permitted. The choice should form part of the specification of an 8-bit code.

4.2.3.3 *Other control functions*

A code with a full range of Cn and Gn code elements requires access to a substantial number of control functions. ISO/IEC 2022 makes provision for control functions to meet the following needs, and others besides:

- locking shifts for the invocation of the Gn code elements;
- single shifts to simplify the selection of isolated characters from the code elements that are not invoked;
- dynamical designation of the character sets used for the Gn and Cn code elements, to permit these code elements to be changed during the course of a single communication;
- announcement of code structure facilities.

Announcement of facilities permits the choice of particular options to be notified to the remote party to the communication, such as whether the characters of the C1 code element are to be coded in an 8-bit code by means of escape sequences or by invocation of the element to the CR area. Further information about control functions of each of the above types is given in 5.3.4.

Although some of these control functions may be coded by a single control character in either the C0 or C1 code elements, many of them require the use of an escape sequence; see 5.3.3. The set of available control functions constitutes the final element of a code specification.

The presence of the ESCAPE character in both the 7-bit and 8-bit code tables, before any other code elements are designated, permits all of the Cn and Gn code elements to be designated dynamically. Since Cn code elements do not require separate invocation, the control functions they provide are immediately available for use. These in turn can be used to invoke the Gn code elements as required. The combined effect of all available facilities is to permit a complete code specification to be communicated to a remote party if required.

4.2.4 Repertoire of a code

It is sometimes convenient to be able to refer to the set of characters that can be represented by a code, in a manner abstracted from the details of that representation. This set of characters is known as the *repertoire* of the code.

The concept of a repertoire is more subtle than it may seem to be at first. Certain character set standards permit two or more characters to be combined in specified ways to create new characters that belong to the repertoire but which are not themselves represented *in* the code. It is this distinction between representation *in*, and representation *by*, a code that causes the subtlety.

One means of combining two characters is by use of the BACKSPACE control character to superpose two images. This is still permitted in the 7-bit code of ISO/IEC 646, but not in more recent character set standards. A more recent technique is to specify that certain characters of a code are non-spacing, so that superposition may be achieved without the use of BACKSPACE. The most significant use of non-spacing characters is that of ISO/IEC 6937. Non-spacing characters are in fact only one example of the more general concept of a *combining character*.

4.2.5 Formal definitions

To ensure precision, the character set standards provide formal definitions of the terms that they use. The following extract from ISO/IEC 2022 gives the definitions of terms used above in this discussion of concepts and terminology.

bit combination: An ordered set of bits used for the representation of characters.

byte: A bit string that is operated upon as a unit. [Note that this definition permits 7-bit, 8-bit and even 16-bit bytes, although common parlance uses the term exclusively for 8 bits. Character set standards use the term "octet" when a restriction to 8 bits is intended.]

coded character set; code: A set of unambiguous rules that establishes a character set and the one-to-one relationship between the characters of the set and their bit combinations.

code table: A table showing the character allocated to each bit combination in a code.

code extension: The techniques for the encoding of characters that are not included in the character set of a given code.

combining character: A member of an identified subset of a coded character set, intended for combination with the preceding or following graphic character, or with a sequence of combining characters preceded or followed by a non-combining character.

control character: A control function the coded representation of which consists of a single bit combination.

control function: An action that affects the recording, processing, transmission or interpretation of data, and that has a coded representation consisting of one or more bit combinations.

to designate: To identify a set of characters that are to be represented, in some cases immediately and in others on the occurrence of a further control function, in a prescribed manner.

escape sequence: A string of bit combinations that is used for control purposes in code extension procedures. The first of these bit combinations represents the control function ESCAPE.

graphic character: A character, other than a control function, that has a visual representation normally handwritten, printed or displayed, and that has a coded representation consisting of one or more bit combinations.

to invoke: To cause a designated set of characters to be represented by one or more bit combinations of a coded character set.

repertoire: A specified set of characters that are each represented by one or more bit combinations of a coded character set.

5 Technical Guidance

This section provides a description of the properties of character sets that may be useful, or which should be avoided, in various application environments. It also provides a more detailed account of the construction of graphic character sets and control functions than is provided in section 4 in the Introduction.

The chapter on graphic characters describes the subtle differences in use between character sets with 94 and 96 positions. It explains how both types of set can make use of either single-byte or multiple-byte coding, the latter providing access to the large character sets required for Chinese, Japanese and

Korean ideographic scripts. It also explains how, even without the use of multiple-byte coding, use of combining characters permits a coded character set to represent more characters than it has code positions.

The chapter on control functions describes the two main sources of such functions:

- the ISO 2375 Register, more properly known as the *International Register of Coded Character Sets to be used with Escape Sequences*, and
- the large repertoire of additional control functions that has been standardized in ISO/IEC 6429.

It explains the use of such control functions, with particular emphasis on those control functions that are used for code extension.

5.1 Application Environments

One of the most important distinctions between different application environments is whether the coded data is required for sequential access or for random access. Both forms of access may occur within a single application, e.g. data may be read sequentially from a storage medium into random access memory. Normally the encoded binary data would be read directly from the storage medium to the random access memory in such circumstances. It is possible, however, to transform the data from one character code to another during the transfer process. If this facility is available, different codes may be chosen which optimise the benefits for each form of access.

5.1.1 Features of sequential access

Sequential access permits the use of control functions that change the mapping between bit combinations and characters. This is known as code extension. The simplest such control function is the use of a locking shift mechanism (as on a typewriter) to switch between two such mappings. Use of locking shifts dates back to the earliest teleprinter codes; see section 4.1 for more information. Modern code extension techniques permit both locking shifts and single shifts to be used, the latter affecting only the immediately following character.

When a 16-bit, or even a 32-bit, code may be used then there is no need for such code extension techniques. When there are reasons (such as compatibility with existing equipment) why an 8-bit, or even a 7-bit, code is to be used then user requirements concerning the character repertoire may compel the use of such techniques.

5.1.2 Features of random access

Random access requires each unit of data to be complete in itself, so that it can be interpreted without reference to anything that may precede or follow it. It normally also requires that the boundaries between units of data must be fixed. For example, in byte-oriented storage of data with a code that uses two bytes per character, it may be required that each character code starts at an even address. No such algorithm is possible if the code uses a variable number of bytes in the representation of its characters. However, it may be acceptable to use such a code if, for example, examination of a fixed number of bytes at any point in the data permits the boundaries of the character representations to be determined. This property will here be called auto-resynchronization. Whether or not this is acceptable will depend on the application concerned.

The need for each unit of data to be complete in itself prevents the use of code extension by means of locking shifts. A code which is extended by means only of single shifts is, in effect, a code that uses a variable number of bytes. The coded representation of the single-shift control function may be considered as part of the representation of the character. Such a code is also auto-resynchronizing

provided that the coded representation of the single-shift function is a single byte that cannot occur in the data stream for any other reason.

5.1.3 Use of code extension techniques

A comprehensive set of techniques for code extension with 7-bit and 8-bit codes is given in ISO/IEC 2022. An introduction to these facilities is given in section 4.2.

These code extension techniques permit up to four 7-bit codes to be selected and then brought into use by means of shift mechanisms. For use with a 7-bit code, only one may be shifted into use at any time but for use with an 8-bit code, two may be brought into use simultaneously. There are mechanisms for communicating between the users which 7-bit codes have been selected, and even for changing this selection during the flow of data. Various levels of implementation are defined, each permitting the use of only a specified selection of the code extension techniques.

For 8-bit codes, greater consistency in the use of code extension techniques may be obtained by requiring conformance to ISO/IEC 4873; see 6.1.4. This fixes the left-hand half of the code table permanently as the ASCII character set and restricts the right-hand half to be a single-byte code. It therefore excludes the 7-bit two-byte codes for Chinese, Japanese and Korean that are permitted under ISO/IEC 2022 itself (more information about these codes is given in 5.2.2.3). It still permits the selection of any three such 7-bit codes for mapping by shift mechanisms into the right-hand half of the table (one of the four 7-bit sets of ISO/IEC 2022 is now permanently the ASCII set). It again specifies various levels of implementation.

Even greater consistency, at the expense of even less flexibility, may be obtained by requiring conformance to ISO/IEC 10367; see 6.1.9. This requires the three 7-bit codes of ISO/IEC 4873 to be chosen from 12 such codes that are specified in the standard.

5.1.4 Restriction to subrepertoires

There is, of course, nothing that compels any user of a coded character set to make use of all the characters that can be represented by that set. However, a recipient of coded data will not normally know that the originator of that data was not going to use all these characters. For many purposes this is unimportant. But if it is required to change the coding to that of a different coded character set, it may be desirable to know that the data will only contain characters that are contained in both repertoires.

The control functions specified in ISO/IEC 6429 include one known as IDENTIFY GRAPHIC SUBREPERTOIRE (IGS). This is provided solely for the purpose of indicating that data coded in accordance with ISO/IEC 10367 is in fact being restricted to a subrepertoire of the full repertoire of that standard. The subrepertoire concerned is identified by its number in an International Register. The manner in which this is coded is described in 5.3.5. Procedures for the registration of subrepertoires of ISO/IEC 10367 are laid down in ISO/IEC 7350; see 6.1.7.

5.2 Graphic Characters

Section 4.2 introduced the four code elements G0, G1, G2 and G3 and explained how these each make available 94 or 96 positions for the allocation of characters. This section explains the facilities available through use of these code elements. It shows in particular how they may provide for the representation of more characters than there are code positions.

5.2.1 94 and 96 position character sets

ISO/IEC 2022 provides for sets of graphic characters to make use of either 94 or 96 code positions. It also prohibits the characters SPACE and DELETE from being assigned in any such set. When these sets are invoked,

1. a 94 position set in the GL area provides assignments for bit combinations 02/01 to 7/14;
2. a 94 position set in the GR area provides assignments for bit combinations 10/01 to 15/14;
3. a 96 position set in the GL area provides assignments for bit combinations 02/00 to 7/15;
4. a 96 position set in the GR area provides assignments for bit combinations 10/00 to 15/15.

All four possibilities are permitted. When a 96 position set is invoked in the GL area it overlays the positions 02/00 and 07/15 that are otherwise assigned to the SPACE and DELETE characters. The characters SPACE and DELETE are therefore not available in this situation. When a 94 position set is invoked in the GR area, the bit combinations 10/00 and 15/15 shall not be used.

ISO/IEC 2022 permits the G1, G2 and G3 code elements to be sets of either 94 or 96 positions but the G0 set is required to have only 94 positions. It also permits the G1, G2 and G3 code elements to be invoked into either the GL area or the GR area of the code table but the G0 code element is only permitted to be invoked into the GL area.

5.2.2 Single-byte and multiple-byte character sets

5.2.2.1 *Nesting of character sets*

ISO/IEC 2022 provides for two alternative types of allocation to the code positions of a 94 or 96 position set:

1. Each position may either have a character assigned to it or be left unused, or
2. Each position may either have a further 94 or 96 position set assigned to it or be left unused.

In the second case a 94 position set may only have its positions allocated to further 94 position sets, and similarly a 96 position set may only have its positions allocated to further 96 position sets. Nesting of sets within sets is permitted to any depth.

5.2.2.2 *Coding of nested sets*

When a nested set is invoked, more than one bit combination (byte) is required to represent an individual character. A sequence of bytes is used that may be processed by the following algorithm:

1. Take the next byte in the sequence (which initially will be the first byte). It identifies either a character or a character set at the code position referenced by that byte in the currently invoked set. If it identifies a character set, go to step 2. If it identifies a character, go to step 3.
2. The identified character set is invoked, for processing the next byte only, into the same area (GL or GR) of the code table as the set currently being processed, therefore replacing it. Processing is then repeated from step 1.
3. The identified character is the character represented by the byte sequence. Processing is complete.

The effect of this algorithm is that the characters of a nested set may be represented by a sequence of one or more bytes with the following properties:

- every character of the set is represented by the same number of bytes;
- every byte of the sequence is in the range appropriate to the status of the set as a 94 or 96 position set invoked into the GL or GR area as described above.

A character set that is nested in this way is called a *multiple-byte set*. A set that is not so nested is called a *single-byte set.*

As an illustration of the effect of the coding algorithm, if a character would be represented by the sequence 03/01 05/04 when a particular two-byte set is invoked in the GL area then it would be represented by 11/01 13/04 if the same set were invoked into the GR area.

5.2.2.3 *Chinese, Japanese and Korean national standards*

Two-byte coded character sets have been registered in the ISO 2375 Register to permit Japanese, Chinese and Korean ideographic scripts to be coded within the ISO/IEC 2022 code structure. These sets are taken from corresponding national standards. They are in fact very comprehensive character sets that provide multilingual facilities; they are not confined to the ideographic characters of the languages concerned. Particular examples are as follows:

- ISO-IR 87 (ESC 02/04 xx 04/02) : Japanese standard JIS C 6226:1983

  This 94-position two-byte set contains 6877 graphic characters that include 147 symbols, digits 0-9, Latin letters A-Z and a-z, Hiragana, Katakana, 24 Greek and 33 Cyrillic letters in both capital and small forms, Japanese Kanji, and 32 line drawing characters. There remain 1959 unallocated byte pairs that shall not be used.

- ISO-IR 168 (ESC 02/06 04/00 ESC 02/04 xx 04/02) : Japanese standard JIS X 0208:1990.

  This is a revision of ISO-IR 87 and is designated by the same escape sequences, preceded by the escape sequence that identifies a first revision. The revision introduces two additional characters. More information about the identification of revised registrations is given under escape sequences with intermediate bytes in the section on control functions.

- ISO-IR 159 (ESC 02/04 gg 04/04) : Japanese standard JIS X 0212:1990.

  This 94-position two-byte set contains 6067 characters that supplement those of ISO-IR 87 or ISO-IR 168. It provides 21 additional symbols, 27 additional Latin letters such as ø and þ, 171 Latin letters with diacritical marks, 21 Greek letters (final sigma and 20 letters with diacritical marks), 26 additional Cyrillic letters and 5801 additional Japanese Kanji characters.

- ISO-IR 149 (ESC 02/04 gg 04/03) : Korean standard KS C 5601:1987.

  This 94-position two-byte set contains 8224 characters that include 276 symbols, digits in both Arabic (0,1,...) and Roman (i,ii,... and I,II,...) forms, the Korean Hangul alphabet, Latin letters A-Z and a-z together with 11 additional capital letters and 16 additional small letters, 24 Greek and 33 Cyrillic letters in both capital and small forms, 68 line drawing characters, Japanese Hiragana and Katakana, 2350 Korean Hangul characters, 4888 Korean Hanja characters, and miscellaneous other characters such as vulgar fractions, superscripts and subscripts.

- ISO-IR 171 (ESC 02/04 gg 04/07) : Chinese standard CNS 11643:1986, Set 1.

  This 94-position two-byte set contains 6085 characters that include 234 symbols, digits in Arabic (0,1,...), Roman (i,ii,... and I,II,...) and Chinese forms, Latin letters A-Z and a-z, 24

Greek letters in both capital and small forms, 42 Mandarin phonetic symbols, 213 Chinese character radicals, 33 control code symbols such as “ESC” and “DEL” each as a single graphic, and 5401 of the most frequently used Chinese characters.

- ISO-IR 172 (ESC 02/04 gg 04/08) : Chinese standard CNS 11643:1986, Set 2.

  This 94-position two-byte set contains 7650 of the less frequently used Chinese characters.

In these escape sequences, replacement of “gg” by 02/08, 02/09, 02/10 or 02/11 specifies designation as a G0, G1, G2 or G3 code element respectively. Where “xx” has been used in place of “gg”, it denotes an exception to the current coding rules of ISO/IEC 2022 in that this bit combination is absent in the designation as a G0 code element. It is still replaced by 02/09, 02/10 or 02/11 to specify designation as a G1, G2 or G3 code element.

The Intermediate Bytes in these escape sequences identify designation of a 94-position two-byte character set as the code element concerned; see 5.3.4.4.

5.2.2.4 *Variable-length coding*

The existence of multiple-byte character sets leads to the possibility of variable-length coding. This may occur for two different reasons:

- the GL and GR code areas may contain character sets with differing coding lengths;
- if a 94 position set is invoked into the GL area then the characters SPACE and DELETE remain represented by the single bytes 02/00 and 07/15 even if the set invoked is a multiple-byte set.

When a character set is designated dynamically as the G0, G1, G2 or G3 element of a code by means of an escape sequence, the general syntax of such sequences allows the receiver to identify:

- whether the set is a 94 or 96 position set;
- whether it is a single-byte or multiple-byte set;
- if it is multiple-byte then how many bytes are required for each character.

This is described in more detail in section 5.3.

5.2.3 Combining characters

Another means of extending the repertoire of a character set beyond the number of available code positions is by means of combining characters. The original use of combining characters was to specify that certain characters of a code were to be non-spacing. When implemented on a receiving device such as a teleprinter, this had the effect of superposing the following character (a letter, say) on top of the non-spacing character (such as an accent) to produce a new character (in this example, an accented letter). A single non-spacing accent could therefore increase the repertoire of a code by many accented letters.

Although a non-spacing accent is classified as a graphic character in its own right, its coded representation cannot be used on its own to represent the accent concerned. It has to be followed by a SPACE character; superposition of the non-spacing accent on a non-printing space results in a normal (spacing) accent. This rule is stated explicitly in ISO/IEC 6937, which is the most well-known standard that uses non-spacing characters.

A non-spacing character is a combining character that combines with the following character. Now that the need to implement combining characters within electromechanical devices such as teleprinters has receded, it has become possible also to specify (and implement) characters that combine with the preceding character. It is perhaps more natural, for example, to describe "é" as a small letter E with an acute accent above it than as an acute accent with a small letter E below it. This approach has been adopted for the new multiple-octet coded character set of ISO/IEC 10646. It is permitted also in the 7-bit and 8-bit code structure of ISO/IEC 2022 but has not, in fact, been used.

The use of combining characters brings variable-length coding into use even within a single code element.

5.3 Control Functions

The concept of a control function is an extension of that of a control character, such as CARRIAGE RETURN (CR), LINE FEED (LF), SHIFT IN (SI) and SHIFT OUT (SO), that has been present since the earliest development of character sets. A control character is simply a control function that is coded as a single bit combination. It is conventional for control functions to have both a name and an identifying acronym.

The code structure of ISO/IEC 2022, as described in section 4.2, includes two code elements C0 and C1 containing control functions. This section describes the standardized sources for these code elements and gives a brief account of the control functions that are available through their use. This account is aimed primarily at the use of control functions for code extension purposes.

The multiple-octet coded character set of ISO/IEC 10646 has a code structure that differs from that of ISO/IEC 2022. It includes its own specification of a code structure for graphic characters, but control functions are incorporated by a provision for the use of control functions encoded according to ISO/IEC 2022. Much of this section is therefore equally relevant to both ISO/IEC 2022 and ISO/IEC 10646 code structures.

5.3.1 Primary sets of control functions

The C0 code element of a code is known as its primary set of control functions. One specific C0 set is specified in ISO/IEC 6429. A code is not required by ISO/IEC 2022 to use this as its primary set, but if a primary set includes any of the control functions from the C0 set of ISO/IEC 6429 then it is required to have the same coding as in that standard. Alternative C0 sets are specified in the ISO 2375 Register.

The C0 set of ISO/IEC 6429 has its historical origin in the control characters of the ASCII character set. For this reason, 10 of the control functions of that set are transmission control functions such as START OF HEADER (SOH) that are not relevant to modern communications protocols. The semantics of those functions are specified in ISO/IEC 6429 by reference to a very old standard, ISO 1745, last revised in 1975.

The control functions of the C0 set of ISO/IEC 6429 are each represented by a single control character, i.e. they are coded by a single bit combination. With one exception the actions of these control functions are fully determined by that single control character. The exception is the ESCAPE (ESC) character, which is a control function whose semantic description in ISO/IEC 6429 is as follows:

- ESC causes the meanings of a limited number of bit combinations following it in the data stream to be changed.

The ESCAPE character together with this following sequence of bit combinations is known as an "escape sequence". The use of escape sequences is reserved by ISO/IEC 2022 to be for code

extension purposes; see 5.3.3 below for more details. All primary sets are required by ISO/IEC 2022 to have the ESCAPE character at position 01/11.

5.3.2 Supplementary sets of control functions

The C1 code element of a code is known as its supplementary set of control functions. One specific C1 set is specified in ISO/IEC 6429. A code is not required by ISO/IEC 2022 to use this as its supplementary set, but a supplementary set is not permitted to include the ESCAPE character or any of the 10 transmission control characters of ISO 1745 described above concerning primary sets. Alternative C1 sets are specified in the ISO 2375 Register.

If the C1 code element is invoked into the CR area of an 8-bit code then its control functions are represented by a single control character, as for the C0 code element. Otherwise, and always for a 7-bit code, the control functions of the C1 code element are represented by an escape sequence.

The C1 set of ISO/IEC 6429 includes its own means of extension, similar to that provided by the ESCAPE character in the C0 set. The control function CONTROL SEQUENCE INTRODUCER (CSI) is followed by one or more bit combinations that together constitute a "control sequence". The permitted control sequences, and the functions they represent, are specified in ISO/IEC 6429 itself. This contrasts with escape sequences, whose use is specified by ISO/IEC 2022. Control sequences are primarily used for the control of devices for the display and presentation of character data.

The C1 set of ISO/IEC 6429 also includes provision for control strings, which are distinguished from escape and control sequences by having both an opening and a closing delimiter. The semantics of control strings is not standardized. They are used only where there is prior agreement between the sender and recipient of the data.

5.3.3 Escape sequences

5.3.3.1 *General construction*

The simplest coding of control functions by more than one bit combination is by means of an escape sequence. The general construction of an escape sequence is laid down in ISO/IEC 2022 and is as follows:

- An escape sequence consists of two or more bytes, of which the first is the ESCAPE character (coded 01/11) and the last, known as the Final Byte, is from columns 03 to 07 of the code table (but excluding the DELETE character 07/15).

- Any bytes between the ESCAPE and the Final Byte are known as Intermediate Bytes and are from column 02 of the code table.

This syntax ensures that an escape sequence can be delimited without any further knowledge of its syntax.

All standardized escape sequences are either defined in ISO/IEC 2022 or are specified in the International Register that is administered in accordance with ISO 2375. This International Register is the primary source of coded character sets for use as code elements in accordance with ISO/IEC 2022.

Escape sequences are further classified by the total number of bytes (bit combinations), including the ESCAPE character, that they involve.

5.3.3.2 *Two-byte escape sequences*

The two-byte escape sequences (those with no Intermediate Bytes) are classified into various types. The differing types are distinguished by the column of the code table that contains the Final Byte, as follows:

- Column 03 is reserved for private use where the control functions represented are agreed between the communicating parties.
- Columns 04 and 05 are reserved for representation of the functions of the C1 code element by escape sequences.
- Columns 06 and 07 are reserved for standardized control functions such as locking shifts that are registered, with their coded representations, in accordance with ISO 2375.

Always in a 7-bit code, and optionally in an 8-bit code, the control functions of a C1 code element are represented by two-byte escape sequences. The Final Byte is obtained by overlaying columns 04 and 05 of the code table with the C1 code element, as if the C1 code element were being temporarily invoked into these columns. For more detail of the use of the standardized control functions, see 5.3.4.

5.3.3.3 *Escape sequences with Intermediate Bytes*

Escape sequences with more than two bytes (those with Intermediate Bytes) are also classified into various types. The differing types are distinguished by the position within column 02 of the code table that contains the first Intermediate Byte. All these types are described below in more detail:

- Position 02/00 is reserved for Announcer Functions;
- Positions 02/01 and 02/02 are used for the designation of C0 and C1 sets of control functions;
- Position 02/03 is reserved for further control functions registered in accordance with ISO 2375 or for private use control functions;
- Position 02/04 is used for the designation of multiple byte graphic character sets;
- Position 02/05 is used to denote an escape from ISO/IEC 2022 code structure to a designated other coding system, such as the multiple-octet coded character set ISO/IEC 10646;
- Position 02/06 is used to identify revision of a registration under ISO 2375. It is followed by a final byte that identifies the revision number, with 04/00, 04/01, ... identifying the first, second, ... revision. Revision of a registration is only permitted under special circumstances laid down in ISO 2375, but see designation of C0 and C1 sets below for an example;
- Positions 02/07 and 02/12 are reserved for future standardization.
- Positions 02/08 to 02/11 and 02/13 to 02/15 are used for the designation of single-byte graphic character sets.

5.3.4 Code extension

5.3.4.1 *Locking shifts*

The primary means of invocation of the G0, G1, G2 and G3 code elements into the GL and GR areas of the code table is by means of locking shifts. Seven such locking shifts are required, since the G0 set cannot be invoked into the GR area. Two of these are included in the C0 set of ISO/IEC 6429 and are therefore required to have the same coding in every C0 set that includes them:

- LOCKING-SHIFT ZERO (LS0), which invokes the G0 set into the GL area, is coded as 00/15;

- LOCKING-SHIFT ONE (LS1), which invokes the G1 set into the GL area, is coded as 00/14.

For historical reasons, when used with a 7-bit code these are known instead as SHIFT-IN (SI) and SHIFT-OUT (SO) respectively.

The remaining five locking shifts are represented by standardized escape sequences. Together with their registration numbers in the ISO 2375 Register, they are:

- LOCKING-SHIFT TWO (LS2), which invokes the G2 set into the GL area, which is ISO-IR 62 and coded as ESC 06/14;

- LOCKING-SHIFT THREE (LS3), which invokes the G3 set into the GL area, which is ISO-IR 63 and coded as ESC 06/15;

- LOCKING-SHIFT ONE RIGHT (LS1R), which invokes the G1 set into the GR area, which is ISO-IR 66 and coded as ESC 07/14;

- LOCKING-SHIFT TWO RIGHT (LS2R), which invokes the G2 set into the GR area, which is ISO-IR 65 and coded as ESC 07/13;

- LOCKING-SHIFT THREE RIGHT (LS3R), which invokes the G3 set into the GR area, which is ISO-IR 64 and coded as ESC 07/12.

5.3.4.2 *Single shifts*

The C1 set of ISO/IEC 6429 includes non-locking shifts SINGLE-SHIFT TWO (SS2) and SINGLE-SHIFT THREE (SS3) that are used to invoke the G2 and G3 code elements for the next graphic character only. It is a matter for prior agreement as to whether these sets are invoked into the GL or GR areas by these single shifts. The area selected is known as the single-shift area. The announcer functions of ISO/IEC 2022 may be used to form this agreement.

It is permitted by ISO/IEC 2022 to include these non-locking shifts in a primary (C0) set of control functions. One C0 set that includes them is the set ISO-IR 106, the Teletex primary set of Control Functions of CCITT Recommendation T.61, which is contained in the ISO 2375 register.

5.3.4.3 *Designation of sets of control functions*

Besides the C0 and C1 sets of ISO/IEC 6429, other standardized sets of control functions are specified in the ISO 2375 register. Although this is nominally a register of standardized escape sequences, where these escape sequences are used to designate coded character sets as elements of a 7-bit or 8-bit code then the register includes the specification of that code element. Escape sequences commencing ESC 02/01 and ESC 02/02 designate specific sets of control functions as the C0 and C1 element respectively. As examples:

- The C0 set of ISO/IEC 6429 is registered as ISO-IR 1 and designated by ESC 02/01 04/00;

- The C1 set of ISO/IEC 6429 is registered as ISO-IR 77 and designated by ESC 02/02 04/03. On its own this designates the C1 set of the 1983 (second) edition of ISO/IEC 6429. To identify the C1 set of the 1992 (third) edition, this escape sequence must be immediately preceded by ESC 02/06 04/00 to identify the first revision.

5.3.4.4 *Designation of sets of graphic characters*

By far the largest part of the ISO 2375 register is the specification of sets of graphic characters that may be designated by means of escape sequences. More information on these sets is given in section 5.2. Individual sets are designated as the G0, G1, G2 or G3 code element by an escape sequence that describes, by means of Intermediate Bytes as specified in ISO/IEC 2022, the nature of the character set and the code element to which it is being invoked. The Final Byte identifies the actual character set concerned.

For single-byte character sets, one Intermediate Byte identifies the code element as follows:

- Bytes 02/08 to 02/11 designate a set of 94 positions (i.e. 02/01 to 07/14 if invoked in the GL area) as the G0, G1, G2 or G3 code element respectively.

- Bytes 02/13 to 02/15 designate a set of 96 positions (i.e. 02/00 to 07/15 if invoked in the GL area) as the G1, G2 or G3 code element respectively. Such a set is not permitted as a G0 code element.

For multiple-byte character sets, the first Intermediate Byte is 02/04 and the second Intermediate Byte identifies the code element as follows:

- Bytes 02/08 to 02/11 designate a multi-byte 94-set as the G0, G1, G2 or G3 code element respectively.

- Bytes 02/13 to 02/15 designate a a multibyte 96-set as the G1, G2 or G3 code element respectively. Such a set is not permitted as a G0 code element.

- Exceptionally, for three registrations only, byte 02/04 is followed directly by a Final Byte to designate a two-byte set of 94 positions as a G0 code element. The Final Bytes permitted for this exceptional case are 04/00, 04/01 and 04/02. The exception dates from an early edition of ISO/IEC 2022 in which such a set could be invoked as a G0 code element but not as a G1, G2 or G3 code element.

Further Intermediate Bytes may also be present in the escape sequence. They are used, for example, to identify the number of bytes per character in a multiple-byte character set. A receiving implementation is therefore able to parse a received data stream into characters without the need for detailed knowledge of the contents of the ISO 2375 Register.

5.3.4.5 *Announcement functions*

Provision is made in ISO/IEC 2022 for the announcement, by means of escape sequences, of a wide range of options permitted by that standard. All these escape sequences consist of ESC 02/00 followed by a Final Byte. Examples are:

- ESC 02/00 04/06 to announce that, in an 8-bit code, the C1 code element shall not be invoked into the CR area. Its control functions are therefore coded as escape sequences;

- ESC 02/00 04/07 to announce that, in an 8-bit code, the C1 code element shall be invoked into the CR area. Its control functions are therefore coded by single bytes;

- ESC 02/00 05/12 to announce that, in an 8-bit code, the single-shift area shall be the GR area.

5.3.5 Control sequences

Control sequences are defined in ISO/IEC 6429 and are used to represent many of the control functions that are specified in that standard. The general construction of a control sequence is similar

to that of an escape sequence but it contains a refinement to permit the representation of control functions that require parameters:

- A control sequence consists of the control function CONTROL FUNCTION INTRODUCER (CSI) followed by one or more bytes, of which the last, known as the Final Byte, is from columns 04 to 07 of the code table (but excluding the DELETE character 07/15).

- The bytes, if any, between the CSI and the Final Byte consist of zero or more Parameter Bytes followed by zero or more Intermediate Bytes. Parameter Bytes are from column 03 of the code table and Intermediate Bytes are from column 02.

The CSI control function is present in the C1 set of ISO/IEC 6429 and is coded either by the single bit combination 09/11 or by the escape sequence ESC 05/11 (see 5.3.2 above). Note that the position of CSI in the C1 set corresponds precisely to the position of ESC in the C0 set.

The function represented by a control sequence is determined by the Final Byte together with any Intermediate Bytes that may be present. The Parameter Bytes act solely as parameters of the function so determined. The syntax of the Parameter Bytes is as follows:

- If the first Parameter Byte is in the range 03/00 to 03/11 then the entire sequence of Parameter Bytes is required to consist of bytes from this range. It represents the coding, according to the ISO-IR 6 (ASCII) character set, of a sequence composed of digits ZERO to NINE together with COLON and SEMICOLON. A SEMICOLON separates parameters when the control function takes more than one parameter. Each parameter is expressed as a decimal number in which a COLON is used to separate the integer and fractional parts.

- If the first Parameter Byte is in the range 03/12 to 03/15 then there is no standardized interpretation for the sequence of Parameter Bytes.

An example of a control function that takes a single numeric parameter is:

- IDENTIFY GRAPHIC SUBREPERTOIRE (IGS), coded as CSI nn 02/00 04/13.

This control function identifies a subrepertoire of the graphic characters of ISO/IEC 10367 which is registered in accordance with ISO/IEC 7350. In the coded representation, “nn” represents the registration number of the repertoire in the ISO/IEC 7350 Register.

5.3.6 Control strings

The C1 set of ISO/IEC 6429 includes provision for control strings that have no standardized meaning but which can be used by private agreement for various control purposes. Each control string has an opening delimiter, contained in the C1 set, that indicates the general nature of the control purpose. The available opening delimiters are:

- Application Program Command (APC)

- Device Control String (DCS)

- Operating System Command (OSC)

- Privacy Message (PM)

- Start of String

All control strings are terminated by a common closing delimiter from the C1 set, namely:

- String Terminator (ST)

Between the two delimiters there may be any sequence of bit combinations other than those representing the delimiters SOS and ST.

5.3.7 Control functions for text communication

The control functions specified in ISO/IEC 6429 contain many functions primarily intended for the control of devices for the display and presentation of character data. These can be used for communicating page layout, either in a fixed format or in a form to allow automatic reformatting when the sender and receiver use different fonts. However, the specifications of the control functions need refinement to allow this to be achieved most satisfactorily. A specification of control functions from ISO/IEC 6429 customised for use in page image communication is given in ISO/IEC 10538.

6 Guides to standards

All the standards described in this annex have a separate description within this section.

6.1 International Standards

6.1.1 ISO/IEC 646

6.1.1.1 *Current edition*

ISO/IEC 646:1991, Information technology – ISO 7-bit coded character set for information interchange (third edition).

6.1.1.2 *Description*

This standard contains the specification of a G0 set of 94 graphic characters for use in the GL area of a 7-bit code. Use of the code requires in addition a C0 set of control functions to be invoked in the CL area. The standard requires this set of control functions to be a subset of the C0 set of ISO/IEC 6429.

The specification contains a number of options. Of the 94 code positions for graphic characters, 10 have no specific character allocated to them. These positions are available for national or application-oriented use. A further two positions have two alternative allocations.

6.1.1.3 *Tutorial guidance*

This standard introduces the concept of a *version* of ISO/IEC 646. A version is obtained by:

- specifying the C0 set of control functions;
- allocating a specific graphic character to each of the 10 unallocated code positions, or declaring that position to be unused;
- making a specific choice between the alternatives NUMBER SIGN and POUND SIGN for position 02/03 and between DOLLAR SIGN and CURRENCY SIGN for position 02/04.

The standard specifies one version itself. This is known as the International Reference Version (IRV) of ISO/IEC 646. Its C0 set is the C0 set of ISO/IEC 6429 and its G0 set is that registered as ISO-IR 6 in the ISO 2375 Register. This is the set commonly known as ASCII. In positions 02/03 and 02/04 it specifies the NUMBER SIGN and DOLLAR SIGN respectively. It is important to note that this is a change from the IRV of the second edition ISO 646:1983, which specified the CURRENCY SIGN in position 02/04 and was registered as ISO-IR 2.

The G0 sets of a number of other versions of ISO/IEC 646 are also registered in accordance with ISO 2375. The register entries of a selection of these, together with the escape sequences that designate them as a G0, G1, G2 or G3 code element, are as follows:

- ISO-IR 2 (ESC gg 04/00) : International Reference Version of ISO 646:1983
- ISO-IR 4 (ESC gg 04/01) : British version
- ISO-IR 6 (ESC gg 04/02) : International Reference Version of ISO/IEC 646:1991 (also USA version)
- ISO-IR 10 (ESC gg 04/07) : Swedish version
- ISO-IR 11 (ESC gg 04/08) : Swedish version for names
- ISO-IR 14 (ESC gg 04/10) : Japanese version for Roman Characters
- ISO-IR 15 (ESC gg 05/09) : Italian version
- ISO-IR 16 (ESC gg 04/12) : Portuguese version, ECMA (Olivetti)
- ISO-IR 17 (ESC gg 05/10) : Spanish version, ECMA (Olivetti)
- ISO-IR 21 (ESC gg 04/11) : German version
- ISO-IR 60 (ESC gg 06/00) : Norwegian version
- ISO-IR 69 (ESC gg 06/06) : French version
- ISO-IR 84 (ESC gg 06/07) : Portuguese version, ECMA (IBM)
- ISO-IR 85 (ESC gg 06/08) : Spanish version, ECMA (IBM)
- ISO-IR 86 (ESC gg 06/09) : Hungarian version
- ISO-IR 141 (ESC gg 07/10) : Serbocroatian and Slovenian version
- ISO-IR 170 (ESC gg 02/01 04/02) : Invariant characters of ISO/IEC 646 (82 characters)

In these escape sequences, replacement of “gg” by 02/08, 02/09, 02/10 or 02/11 specifies designation as a G0, G1, G2 or G3 code element respectively. These Intermediate Bytes specify designation of a 94-position single-byte character set as the code element concerned; see designation of graphic character sets in the section on control functions.

For reasons of backward compatibility with previous versions, ISO/IEC 646 permits the use of BACKSPACE (BS) and CARRIAGE RETURN (CR) control functions to create composite characters. In particular, it specifies that the sequence of a letter character, BACKSPACE and one of QUOTATION MARK, APOSTROPHE or COMMA should be interpreted as that letter bearing a diaeresis, acute accent or cedilla respectively. The character set separately includes GRAVE ACCENT, CIRCUMFLEX ACCENT and TILDE which may be combined similarly to produce letters with other diacritical marks. More recent character set standards that permit characters to be combined, such as ISO/IEC 6937, make use of specific combining characters as described in the section on graphic characters, so avoiding the use of control functions.

6.1.2 ISO/IEC 2022

6.1.2.1 *Current edition*

ISO/IEC 2022:1994, Information technology – Character code structure and extension techniques (fourth edition).

6.1.2.2 *Description*

This standard specifies a structure for 7-bit and 8-bit codes that is adopted by all such codes produced under the auspices of ISO/IEC JTC1/SC2. This is the subcommittee entrusted jointly by ISO and IEC with the development of character set coding matters.

This standard also specifies means by which the correspondence between bit combinations and characters may be changed during a particular instance of information interchange. This is known as code extension. It makes use of control functions that are themselves represented by bit combinations within the original code.

6.1.2.3 *Tutorial guidance*

This annex contains introductions to the features of ISO/IEC 2022 at various levels:

- The basic ideas of ISO/IEC 2022 are described in section 4.2, Concepts and terminology.
- Rurther details can be found in section 5.2, Graphic character sets, and 5.3, Control Functions.

The facilities of ISO/IEC 2022 are of a powerful and general nature. A simplified structure for 8-bit codes and code extension is specified in ISO/IEC 4873. The structure of ISO/IEC 4873 does not permit multiple-byte coded character sets to be invoked and therefore excludes, for example, the coding of Japanese, Chinese and Korean ideographic characters as described in 5.2.

6.1.3 ISO 2375

6.1.3.1 *Current edition*

ISO 2375:1985, Data processing – Procedure for registration of escape sequences (third edition).

6.1.3.2 *Description*

The structure of 7-bit and 8-bit codes for the representation of character sets, and code extension techniques for use with such character sets, are specified in ISO/IEC 2022. These code extension techniques make use of escape sequences, a concept defined in ISO/IEC 2022. That standard defines classes of escape sequences but does not assign particular meanings to individual escape sequences. ISO 2375 specifies the procedures to be followed in preparing and maintaining a register of specific escape sequence meanings.

6.1.3.3 *Tutorial guidance*

The procedures of ISO 2375 allow for the registration of an escape sequence, for the withdrawal of a registration by the authority that sponsored it, and in exceptional circumstances for the revision of a registration.

The registration authority for ISO 2375 is:

- IPSJ/ITSCJ (Information Processing Society of Japan/Information Technology Standards Commission of Japan) Room 308-3, Kikai-Shinko-Kaikan Bldg., 3-5-8, Shiba-Koen, Minato-ku, Tokyo 105 JAPAN

The register used to be available free of charge in paper form. However, the register is now available in a much more convenient form electronically on the Web. See 6.2.1 on the ISO 2375 Register for more details of its contents.

More detail of the general classification of escape sequences is given in section 5.3.

6.1.4 ISO/IEC 4873

6.1.4.1 *Current edition*

ISO/IEC 4873:1991, Information technology – ISO 8-bit code for information interchange – Structure and rules for implementation (third edition).

6.1.4.2 *Description*

This standard specifies a structure for 8-bit codes that builds on the general structure for such codes laid down in ISO/IEC 2022. In particular the content of the GL area of the code table is fully specified and the content of the GR area is restricted to be a character set that makes use of single-byte coding (and so contains at most 96 characters). The fixed content for the GL area is the set registered in the ISO 2375 Register as ISO-IR 6. This set is also the International Reference Version (IRV) of ISO/IEC 646:1991 and is more commonly known as the ASCII character set.

The code extension techniques permitted by ISO/IEC 4873 are only a selection of those specified by ISO/IEC 2022.

6.1.4.3 *Tutorial guidance*

ISO/IEC 4873 specifies three levels of implementation:

- Level 1 requires a C0 set in addition to the fixed (ASCII) G0 set and it permits a C1 set and a G1 set to be present. It does not permit the use of a G2 or G3 set. The C1 and G1 sets, if present, are invoked in the CR and GR areas.

- Level 2 includes the facilities of level 1 but permits, in addition, a G2 and G3 set that may be invoked by means of the single-shift control functions SS2 and SS3. The C1 set is required to be present and to contain at least the SS2 and SS3 control functions at positions 08/14 and 08/15 respectively. A C1 set containing only these two control functions is registered as ISO-IR 105. The use of locking shifts is not permitted.

- Level 3 includes the facilities of level 2 but permits, in addition, the use of the locking shifts LS1R, LS2R and LS3R.

The coding of single shifts and locking shifts is given in section 5.3.

A collection of coded graphic character sets suitable for use within the structure of ISO/IEC 4873 has been standardized in ISO/IEC 10367.

6.1.5 ISO/IEC 6429

6.1.5.1 *Current edition*

ISO/IEC 6429:1992, Information technology – Control functions for coded character sets (third edition).

6.1.5.2 *Description*

This standard specifies a repertoire of a large number of control functions, giving both their definitions and their coded representations. It includes a C0 set and a C1 set that may be designated

for use with any 7-bit or 8-bit code that conforms to the code structure laid down in ISO/IEC 2022, or with the universal multiple-octet coded character set of ISO/IEC 10646. The coded representation of individual control functions consists either of:

- a single bit combination from either the C0 or the C1 set, or
- a sequence of bit combinations that starts with one from either the C0 or C1 set and which is constructed according to specific rules that allow the end of the sequence to be determined without detailed knowledge of the syntax of the particular function represented.

6.1.5.3 *Tutorial guidance*

The control functions defined in ISO/IEC 6429 fall into a number of distinct categories. They are primarily concerned with:

- code extension;
- transmission control for coded data;
- formatting, editing and presentation of character data;
- control of ancillary devices.

More detail of the coding methods used, and of the control functions available for code extension, is given in 5.3 on control functions.

The control functions for formatting, editing and presentation of character data are suitable for use in communicating page layout, either in a fixed format or in a form to allow automatic reformatting when the sender and receiver use different fonts. More specific definitions of these functions, tailored to these particular uses, are given in ISO/IEC 10538; see 6.1.10.

6.1.6 ISO/IEC 6937

6.1.6.1 *Current edition*

ISO/IEC 6937:1994, Information technology – Coded graphic character set for text communication – Latin alphabet (second edition).

6.1.6.2 *Description*

This standard contains the specification of a set of graphic characters for the GL and GR areas of an 8-bit code table. Provision is also included for it to be used as a 7-bit code by making use of code extension techniques from ISO/IEC 2022.

The GL area contains the G0 set of ISO/IEC 4873, namely ISO-IR 6, the ASCII character set. The GR area contains characters in 86 of the 96 available code positions, the remaining 10 positions being excluded from use. The 13 characters in column 12, there being three unassigned positions in this column, are non-spacing diacritical marks (combining characters). The characters in all other columns are spacing (non-combining) characters.

Due to its use of non-spacing diacritical marks, the code can represent more characters than there are code positions. The standard includes a normative specification of the 333 graphic characters, including the SPACE character, that it is permitted to represent by use of the 181 bit combinations that are assigned in the code. This set of 333 graphic characters constitutes the repertoire of the standard.

6.1.6.3 *Tutorial guidance*

ISO/IEC 6937 specifies a character set that is primarily intended for information interchange using the Latin script. Characters with diacritical marks (accents) are transmitted by sending a non-spacing accent character (the combining character) followed by the underlying letter character. The available diacritical marks are:

- grave, acute and circumflex accents;
- tilde, macron, breve, diaeresis, cedilla, ogonek and caron;
- ring above and dot above;
- double acute accent.

The approach adopted by this standard originates with electromechanical devices such as teleprinters. In such devices it is not difficult to prevent certain characters from operating the escapement mechanism that moves to the next printing position. It is an approach that is unsuitable for most data and text processing applications. It does have advantages, however, for sorting (collating) purposes since the underlying letter is easily identified.

The repertoire is suitable for use with a wide range of languages. In particular it contains all the characters of all of the codes known as Latin Alphabets Nos. 1 to 6 that are defined in parts of ISO/IEC 8859, and some additional characters as well. However, it does not contain some of the characters of the codes known as Latin Alphabets Nos. 7 to 9 that are defined in other parts of ISO/IEC 8859. There are no plans to add these missing characters, in particular, the EURO SIGN. It states that it covers the following languages:

- Afrikaans, Albanian, Basque, Breton, Catalan, Croat, Czech, Danish, Dutch. English, Esperanto, Estonian, Faroese, Finnish, French, Frisian, Galician, German, Greenlandic, Hungarian, Icelandic, Irish, Italian, Lapp (Sami), Latvian, Lithuanian, Maltese, Norwegian, Occitan, Polish, Portuguese, Rhaeto-Romanic, Roumanian, Scots Gaelic, Slovak, Slovene, Sorbian, Spanish, Swedish, Turkish and Welsh

However, it also provides an informative note that it does not cover the full repertoire required for Welsh. The missing characters are W and w with acute and grave accents and with diaeresis, and Y and y with grave accents. These characters are all included in ISO-IR 182, which may be used with ISO-IR 6 to form a Welsh variant of Latin Alphabet No.1. They are also included in the newer Latin Alphabet No. 8 defined in ISO/IEC 8859-14 together with missing characters needed for other Celtic languages. See 6.1.8, the guide to ISO/IEC 8859, for more details.

6.1.7 ISO/IEC 7350

6.1.7.1 *Current edition*

ISO/IEC 7350:1991, Information technology – Registration of repertoires of graphic characters from ISO/IEC 10367 (second edition).

6.1.7.2 *Description*

ISO/IEC 7350 specifies the procedures to be followed in preparing, publishing and maintaining a register of graphic character repertoires which are composed entirely of graphic characters from ISO/IEC 10367. The coded representation of the characters of such repertoires is not prescribed by the entries in the register.

6.1.7.3 *Tutorial guidance*

A repertoire registered in accordance with ISO/IEC 7350 is required to consist of either:

- the characters of the G0 set of ISO/IEC 10367, and
- the characters of any number of its supplementary sets except that of ISO/IEC 6937

or:

- the characters of the G0 set of ISO/IEC 10367, and
- the characters of any number of the supplementary sets chosen from the Arabic, Cyrillic, Greek, Hebrew or Basic Box Drawing supplementary sets, and
- a subrepertoire of the repertoire associated with the supplementary set of ISO/IEC 6937.

ISO/IEC 7350 requires a numeric identifier to be assigned to each registered repertoire. This identifier is intended for use with the control function IDENTIFY GRAPHIC SUBREPERTOIRE (IGS) that is defined in ISO/IEC 6429. The coding of this control function and its integer parameter by a control sequence is described in section 5.3.

6.1.8 ISO/IEC 8859

6.1.8.1 *Current edition*

This is a multi-part standard.

ISO 8859, Information processing – 8-bit single-byte coded graphic character sets

This is a multi-part standard.

ISO 8859, Information processing - 8-bit single-byte coded graphic character sets

Part 1: Latin alphabet No. 1 (1998)

Part 2: Latin alphabet No. 2 (1999)

Part 3: Latin alphabet No. 3 (1999)

Part 4: Latin alphabet No. 4 (1998)

Part 5: Ltin/Cyrillic alphabet (1999)

Part 6: Latin/Arabic alphabet (1999)

Part 7: Latin/Greek alphabet (1987)

Part 8: Latin/Hebrew alphabet (1999)

Part 9: Latin alphabet No. 5 (1999)

Part 10: Latin alphabet No. 6 (1998)

Part 11: Latin/Thai character set (DIS)

Part 13: Latin alphabet No. 7 (1998)

Part 14: Latin alphabet No. 8 (Celtic) (1998)

Part 15: Latin alphabet No. 9 (1999)

6.1.8.2 *Description*

Each part of this standard contains the specification of a set of graphic characters for the GL and GR areas of an 8-bit code table. For each part, the GL area contains the G0 set of ISO/IEC 4873 (ISO-IR 6, the ASCII character set), so that only the 96 character positions of the GR area vary between the parts.

The GR area makes use of single-byte coding and contains no non-spacing diacritical marks (or other combining characters). The repertoire of the code therefore comprises at most 191 graphic characters including the SPACE character, one for each of the 191 available code positions.

6.1.8.3 *Tutorial guidance*

Each part of ISO/IEC 8859 specifies a character set that is suitable both for data and text processing applications and for information interchange.

The GR area of each of the Latin Alphabets includes a selection of accented Latin letters, and possibly also additional Latin letters such as Icelandic letters Þ (capital letter thorn), þ (small letter thorn) and ð (small letter eth). Each of these parts is suitable for multiple-language applications using the Latin script. The remaining five parts contain characters from a non-latin script in the GR area, as indicated by their title.

Each part specifying a Latin Alphabet lists the languages for which it has been designed. These are:

Latin Alphabet No.1

> Albanian, Basque, Breton, Catalan, Danish, Dutch, English, Faroese, Finnish, French (with restrictions), Frisian, Galician, German, Greenlandic, Icelandic, Irish Gaelic (new orthography), Italian, Latin, Luxemburgish, Norwegian, Portuguese, Rhjaeto-Romanic, Scotish Gaelic, Spanish and Swedish.

Latin Alphabet No.2

> Albanian, Croat, Czech, English, German, Hungarian, Latin, Polish, Romanian, Slovak, Slovene and Sorbian.

Latin Alphabet No.3

> Esperanto and Maltese, and if needed in conjunction with these, English, French (with restrictions), German, Italian, Latin and Portuguese. Coding of Turkish characters is deprecated in this code.

Latin Alphabet No.4

> Danish, English, Estonian, Finnish, German, Greenlandic, Latin, Latvian, Lithuanian, Norwegian, Sámi (with restrictions), Slovene and Swedish.

Latin Alphabet No.5

> Albanian, Basque, Breton, Catalan, Danish, Dutch, English, Faroese, Finnish, French (with restrictions), Frisian, Galician, German, Greenlandic, Irish Gaelic (new orthography), Italian, Latin, Luxemburgish, Norwegian, Portuguese, Spanish, Rhaeto-Romanic, Scottish Gaelic, Spanish, Swedish and Turkish.

Latin Alphabet No.6

Danish, English, Estonian, Faroese, Finnish, German, Greenlandic, Icelandic, Irish Gaelic (new orthography), Latin, Lithuanian, Norwegian, Sámi (with restrictions), Slovene and Swedish.

Latin Alphabet No.7

Danish, English, Estonian, Finnish, German, Latin, Latvian, Lithuanian, Norwegian, Polish, Slovene and Swedish.

Latin Alphabet No.8

Albanian, Basque, Breton, Catalan, Cornish, Danish, Dutch, English, French (with restrictions), Frisian, Galician, German, Greenlandic, Irish Gaelic (old and new orthographies), Ialian, Latin, Luxemburgish, Manx Gaelic, Norwegian, Portuguese, Rhaeto-Romanic, Scottish Gaelic, Spanish, Swedish and Welsh.

Latin Alphabet No.9

Albanian, Basque, Breton, Catalan, Danish, Dutch, English, Estonian, Faroese, Finnish, French, Frisian, Galician, German, Greenlandic, Icelandic, Irish Gaelic (new orthography), Italian, Latin, Luxemburgish, Norwegian, Portuguese, Rhaeto-Romanic, Scotish Gaelic, Spanish and Swedish

For writing French, three characters not included in Latin Alphabets 1, 3, 5 and 8, are also needed. These are included in Latin Alphabet No.9.

The Skolt Sámi dialect, and older Sámi orthography, require certain additional characters. These have been registered in ISO-IR 158 and ISO-IR 197 in the ISO 2375 Register. ISO/IEC 8859-10 recommends the use of that character set as a G2 or G3 code element together with the GL and GR sets of ISO/IEC 8859-10 as G0 and G1 code elements when these characters are required.

ISO-IR 182 was registered to be an alternative GR set for ISO/IEC 8859-1 to cover the Welsh language. Use of Latin Alphabet No.8 is now recommended for that purpose.

The coded character sets of the GR areas of each part of ISO/IEC 8859 are included in the ISO 2375 Register. With the exception of parts 10 to 15 they are also included in ISO/IEC 10367. For the registration number and escape sequences assigned to these sets, see the guide to ISO/IEC 10367 in 6.1.9.

6.1.9 ISO/IEC 10367

6.1.9.1 *Current edition*

ISO/IEC 10367:1991, Information technology – Standardized coded graphic character sets for use in 8-bit codes (first edition).

6.1.9.2 *Description*

ISO/IEC 10367 specifies a collection of coded graphic character sets suitable for use within the structure of an 8-bit code as laid down in ISO/IEC 4873. These sets are all suitable for use as any of the code elements G1, G2 and G3 in a version of ISO/IEC 4873 at any of its three levels of implementation. The G0 code element of ISO/IEC 4873 is prescribed by that standard but is repeated for information in ISO/IEC 10367. ISO/IEC 10367 does not specify the sets C0 and C1 of control functions that may be used in a version of ISO/IEC 4873 that conforms to ISO/IEC 10367.

6.1.9.3 *Tutorial guidance*

The coded graphic character sets specified in ISO/IEC 10367 are all contained in the ISO 2375 Register. Their register entries, together with the escape sequences that designate them as a G1, G2 or G3 code element, are as follows.

- For the G0 code element:
    - ISO-IR 6 (ESC 02/08 04/02) : International Reference Version of ISO/IEC 646:1991
- For the G1, G2 and G3 code elements:
    - ISO-IR 100 (ESC gg 04/01) : Latin Alphabet No.1, Supplementary Set (GR area of ISO 8859-1)
    - ISO-IR 101 (ESC gg 04/02) : Latin Alphabet No.2, Supplementary Set (GR area of ISO 8859-2)
    - ISO-IR 109 (ESC gg 04/03) : Latin Alphabet No.3, Supplementary Set (GR area of ISO 8859-3)
    - ISO-IR 110 (ESC gg 04/04) : Latin Alphabet No.4, Supplementary Set (GR area of ISO 8859-4)
    - ISO-IR 148 (ESC gg 04/13) : Latin Alphabet No.5, Supplementary Set (GR area of ISO/IEC 8859-9)
    - ISO-IR 144 (ESC gg 04/12) : Cyrillic Supplementary Set (GR area of ISO/IEC 8859-5)
    - ISO-IR 127 (ESC gg 04/07) : Arabic Supplementary Set (GR area of ISO 8859- 6)
    - ISO-IR 126 (ESC gg 04/06) : Greek Supplementary Set (GR area of ISO 8859- 7)
    - ISO-IR 138 (ESC gg 04/08) : Hebrew Supplementary Set (GR area of ISO 8859- 8)
    - ISO-IR 154 (ESC gg 05/00) : Supplementary Set for Latin Alphabets No.1 or No.5, and No.2
    - ISO-IR 155 (ESC gg 05/01) : Basic Box Drawing Set
    - ISO-IR 156 (ESC gg 05/02) : Supplementary Set of ISO/IEC 6937

Since the publication of ISO/IEC 10367, other character sets have been registered that are also intended for use as G1, G2 or G3 code elements in a version of ISO/IEC 4873. Some of these have also been standardised in further parts of ISO/IEC 8859. Although these are not part of ISO/IEC 10367, they are listed here for completeness:

- ISO-IR 157 (ESC gg 05/06) : Latin Alphabet No.6, Supplementary Set (GR area of ISO/IEC 8859-10)
- ISO-IR 166 (ESC gg 05/04) : Thai Supplementary Set (GR area of ISO/IEC 8859-11)
- ISO-IR 179 (ESC gg 05/09) : Latin Alphabet No.7, Baltic Rim Supplementary Set (GR area of ISO/IEC 8859-13)

- ISO/IR 199 (ESC gg 05/15) : Latin Alphabet No.8, Celtic Suplementary Set (GR area of ISO/IEC 8859-14)

- ISO-IR 203 (ESC gg 06/02) : Latin Alphabet No.9, European Supplementary Set (GR Area of ISO/IEC 8859-15)

- ISO-IR 158 (ESC gg 05/08) : Supplementary Set for Sami (Lappish) to complement Latin Alphabet No.6 (from annex A of ISO/IEC 8859-10)

- ISO-IR 197 (ESC gg 05/13) : Supplementary Set for Sami to complement Latin Alphabet No.6 (from annex A of ISO/IEC 8859-10)

- ISO-IR 182 (ESC gg 05/12) : Welsh variant of Latin Alphabet No.1, Supplementary Set

In these escape sequences, replacement of "gg" by 02/13, 02/14 or 02/15 specifies designation as a G1, G2 or G3 code element respectively. These Intermediate Bytes specify designation of a 96-position single-byte character set as the code element concerned; see 5.3.4.4. on the designation of graphic character sets.

For more details of the character sets taken from ISO/IEC 646, ISO/IEC 8859 and ISO/IEC 6937, including the languages for which they are suitable, see the entries for those standards in this annex. The entry for ISO/IEC 8859 also covers the additional sets listed above.

There is a requirement concerning the Supplementary Set of ISO/IEC 6937 (ISO- IR 156) that it shall not be used in conjunction with any of the Latin Alphabet Supplementary Sets. However, it may be used in conjunction with any two of the supplementary sets for Greek, Cyrillic, Arabic and Hebrew as G2 and G3 code elements, to provide support for these scripts in addition to a wide range of languages in the Latin script.

Use of ISO-IR 156 requires the support of non-spacing diacritical marks and so results in a code with variable-length coding. All the 333 characters (including SPACE) that are in the repertoire of ISO/IEC 6937 can be represented with single-byte coding by selecting ISO-IR 100 or 148 as the G1 code element, ISO-IR 101 as the G2 code element and ISO-IR 154 as the G3 code element.

There may be a need in an instance of communication to be able to identify a subrepertoire of the full repertoire of characters present in the character sets of ISO/IEC 10367. Procedures for the registration of such subrepertoires are specified in ISO/IEC 7350.

6.1.10 ISO/IEC 10538

6.1.10.1 *Current edition*

ISO/IEC 10538:1991, Information technology – Control functions for text communication (first edition).

6.1.10.2 *Description*

ISO/IEC 10538 defines the control functions, and their coded representations, needed for use in text communication. The coded representations are intended for use when the control functions concerned are embedded in the communicated text, not when they are separated from the text as elements of a communication protocol.

6.1.10.3 *Tutorial guidance*

ISO/IEC 10538 is divided into three sections. The first section provides a general introduction. The second section specifies control functions for text in a page-image format in which the sender's and recipient's pages are intended to be identical. The third section specifies control functions for text that

may be either in a formatted or a reformattable form, suitable for use where the sender's and recipient's fonts differ.

With two exceptions, the control functions of ISO/IEC 10538 have been taken from ISO/IEC 6429 but they have been given more specific definitions than in that standard. The exceptions are the functions PAGE TERMINATOR (PT) and DOCUMENT TERMINATOR (DT). These are alternative names assigned by ISO/IEC 10538 to the control functions INFORMATION SEPARATOR THREE (IS3) and INFORMATION SEPARATOR FOUR (IS4) of ISO/IEC 6429, to represent their correspondingly more specific definitions.

6.1.11 ISO/IEC 10646

6.1.11.1 *Current edition*

ISO/IEC 10646-1:1993, Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane (first edition).

6.1.11.2 *Description*

ISO/IEC 10646 specifies a multiple-octet coded character set, the UCS, that is applicable to the representation, transmission, interchange, processing, storage, input and presentation of the written form of the languages of the world as well as additional symbols. It uses a coding system different from that specified in ISO/IEC 2022, but it provides mechanisms in accordance with ISO/IEC 2022:

- to designate the use of the UCS from ISO/IEC 2022, and
- to designate return to ISO/IEC 2022 from the UCS.

6.1.11.3 *Tutorial guidance*

ISO/IEC 10646 is planned as a standard in multiple parts, of which only part 1 has so far been published. In its full form a four-octet coding (32 bits) will be required, but a two-octet (16 bit) coding is specified that covers use of the Basic Multilingual Plane given in Part 1.

Annex B contains a detailed description to the architecture and content of ISO/IEC 10646 (UCS).

6.1.12 ISO/IEC ISP 12070

6.1.12.1 *Current edition*

ISO/IEC ISP 12070-1:1996, Information technology – International Standardized Profiles FCSnnn – Character Set – Code Structure based on ISO/IEC 2022 – Part 1: FCS111 – 2022 Option 1 (first edition).

6.1.12.2 *Description*

The profile ISO/IEC ISP 12070 forms part of the documentation of Open Systems Interconnection (OSI), for which the Basic Reference Model is specified in ISO/IEC 7498-1:1994 (second edition).

The profile ISO/IEC ISP 12070 was planned as an International Standardized Profile in multiple parts. Each part would provide requirements that may be used to ensure a consistent approach when specifying the use of coded character sets in functional standards. Since activity on OSI has now come to a halt internationally, it is unlikely that that the remaining parts will be produced.

Within the set of character set standards there are two generic code structures, that defined by ISO/IEC 2022 for 7-bit and 8-bit transport mechanisms and that defined by the new ISO/IEC 10646 for a multiple-octet transport mechanism. It was the intention to cover both of these generic code structures as future parts were prepared. Part 1 is concerned with the ISO/IEC 2022 code structure.

The requirements that it specifies apply specifically to Western Europe but they may be applicable also to other regions of the world.

6.1.12.3 *Tutorial guidance*

Part 1 of ISO/IEC ISP 12070 is applicable to use of ISO/IEC 2022 code structure in the following cases:

- in the ASN.1 type GeneralString;
- in the ASN.1 type GraphicString;
- in the ASN.1 type TeletexString;
- outside of the ASN.1 character string types, such as when a character string is embedded in the ASN.1 OCTET STRING type.

Abstract Syntax Notation One (ASN.1) forms part of the Presentation Layer of the OSI model. It is specified in the four-part standard ISO/IEC 8824, the current edition of which was published in 1995, replacing the second edition of 1990 that was a one-part standard. The ASN.1 standard defines a number of character string types, of which those listed above are a selection.

The last case in the list is particularly relevant for character based data streams and for this reason ISO/IEC ISP 12070-1 is included in this annex. It contains a number of specific conformance requirements which go a long way towards supporting implementation interoperability. These could be used as a checklist for developers but also as requirements to be referenced during the procurement process.

6.2 International Registers

6.2.1 ISO 2375 Register (International Register of Coded Character Sets to be used with Escape Sequences)

6.2.1.1 *Current edition*

ISO 2375 Register of International Register of Coded Character Sets to be used with Escape Sequences.

This register is updated as additional entries are approved in accordance with the procedures of ISO 2375. It does not have discrete editions. It is available from the Registration Authority:

> IPSJ/ITSCJ (Information Processing Society of Japan/Information Technology Standards Commission of Japan), Room 308-3, Kikai-Shinko-Kaikan Bldg., 3-5-8, Shiba-Koen, Minato-ku, Tokyo 105 JAPAN

The register used to be available free of charge in paper form. However, the register is now available in a much more convenient form electronically on the Web.

6.2.1.2 *Description*

This register contains the specifications of all coded character sets and control functions that can be brought into use by means of escape sequences. Many, but not all, of the coded character sets specified in the register are taken from national and international standards.

Each register entry is assigned a registration number that provides an unambiguous method for referencing that entry. It is specified in ISO 2375 that the entry with registration number *nn* should be referenced as ISO-IR *nn*.

6.2.1.3 *Tutorial guidance*

The ISO 2375 Register is divided into the following sections and subsections:

- Graphic character sets
    - 94-character sets, i.e. 94-position single-byte sets
    - 96-character sets, i.e. 96-position single-byte sets
    - Multiple-byte sets, which may be either 94-position or 96-position sets
- Control character sets
    - C0 sets of control characters
    - C1 sets of control characters
- Single control functions
- Coding systems different from that of ISO/IEC 2022
    - Coding systems with a standard return
    - Coding systems without a standard return

The registration numbers and escape sequences for a number of register entries are given elsewhere in this annex. These entries may be found as follows:

- For 94-character single-byte sets that provide national and other versions of ISO/IEC 646 (variants of ASCII), see 6.1.1, ISO/IEC 646.
- For 96-character single-byte sets that supplement ISO-IR 6 (ASCII) to provide characters of the Greek, Cyrillic, Arabic and Hebrew scripts and additional Latin characters, see the guide to ISO/IEC 10367 in 6.1.9. This reference includes the character sets of ISO/IEC 6937 and ISO/IEC 8859.
- For two-byte sets from Chinese, Japanese and Korean national standards that provide coding for ideographic scripts, see 5.2.2.3.
- For single control functions for code extension, see 5.3.4.1.

Register entries which specify C0 and C1 sets of control functions include:

- C0 sets:
    - ISO-IR 1 (ESC 02/01 04/00) : C0 set of ISO/IEC 6429
    - ISO-IR 104 (ESC 02/01 04/07) : Minimum C0 set for ISO/IEC 4873
      This set comprises only the character ESCAPE (ESC).
    - ISO-IR 106 (ESC 02/01 04/05) : Teletex primary set of Control Functions of CCITT Recommendation T.61.
      This set comprises the control characters BACKSPACE (BS), LINE FEED (LF), FORM FEED (FF), CARRIAGE RETURN (CR), LOCKING SHIFT ZERO (LS0),

LOCKING SHIFT ONE (LS1), SINGLE-SHIFT TWO (SS2), SINGLE-SHIFT THREE (SS3), SUBSTITUTE CHARACTER (SUB) and ESCAPE (ESC) only.

- C1 sets:
  - ISO-IR 77 (ESC 02/02 04/03) : C1 set of ISO 6429:1983

    The C1 set of ISO/IEC 6429:1992 is designated as the first revision of ISO-IR 77 by means of the escape sequences ESC 02/06 04/00 ESC 02/02 04/03; see designation of sets of control functions in the section on control functions.
  - ISO-IR 105 (ESC 02/02 04/07): Minimum C1 set for ISO/IEC 4873

    This set comprises only the control characters SINGLE-SHIFT TWO (SS2) and SINGLE-SHIFT THREE (SS3).
  - ISO-IR 107 (ESC 02/02 04/08) : Teletex supplementary set of Control Functions of CCITT Recommendation T.61.

    This set comprises the control functions PARTIAL LINE UP (PLU), PARTIAL LINE DOWN (PLD) and CONTROL SEQUENCE INTRODUCER (CSI) only.

The escape sequences listed designate these sets as the C0 or C1 code element, as appropriate.

6.3 European Standards

6.3.1 EN 1922

6.3.1.1 *Current edition*

EN 1922:1998, Information technology – Character repertoire and coding for interworking with telex services (first edition) .

6.3.1.2 *Description*

European Standard EN 1922 is intended to be used with, and identified within, other European functional standards that specify strings of coded characters for interchange of coded information between information processing systems. It describes the graphic character repertoire and control functions relevant for information interchange via Telex network equipment.

European Standard EN 1922 is a revision of European Pre-standard ENV 41504:1990.

6.3.1.3 *Tutorial guidance*

EN 1922 specifies two repertoires, one each for the Latin and Greek scripts. The Latin repertoire is suitable for coding by means of the 5-bit Baudot code described in the historical background in this annex and which is more properly described as CCITT International Telegraphic Alphabet No.2. The Greek repertoire is similarly suitable for coding by means of the 5-bit code specified in the Hellenic national standard ELOT 1095:1989.

EN 1922 also specifies a transformation procedure for interconverting between data coded according to these 5-bit codes and data coded in an 8-bit code according to one of the options of EN 1923.

6.3.2 EN 1923

6.3.2.1 *Current edition*

EN 1923 (to be published), European character repertoires and their coding – 8 bit single byte coding (first edition).

6.3.2.2 *Description*

European Standard EN 1923 specifies the graphic character repertoires, and their coding, which are available for use for information interchange between information processing systems and for use within such systems, in the scripts that are commonly used by the members of CEN and the institutions of the European Union and the European Free Trade Association.

EN 1923 is a successor to three European Pre-standards, namely ENV 41503, ENV 41505 and ENV 41508.

6.3.2.3 *Tutorial guidance*

EN 1923 specifies a number of repertoires of characters selected from the Latin, Greek and Cyrillic scripts and a further repertoire of symbols. Each repertoire is assigned a name and a mnemonic. The names and mnemonics are:

- The Invariant-Latin repertoire (IVL), which is a subset of:
- The Initial-Latin repertoire (IL), which is a subset of:
- The Basic-Latin repertoire (BL), which is a subset of:
- The Full-Latin-8 repertoire (FL8).
- The Basic-Greek repertoire (BG).
- The Basic-Cyrillic repertoire (BC).
- The Symbols-1 repertoire (BS).

A number of options are specified within the standard, each of which identifies a requirement to support one or more of the above repertoires. When 8-bit single-byte coding is used to support of any of the identified options, it is required to conform to ISO/IEC 4873 at one of its three levels of implementation.

Annex B

THE UNIVERSAL CHARACTER SET (UCS)

This Annex to the Guide to the Use of Character Sets in Europe provides more detailed information about the Universal Multi-octet Coded Character Set (UCS) specified in ISO/IEC 10646-1 than is found in the main body of the Guide. Annex A deals in more detail with 8-bit character set standards.

Table of Contents

1 Introduction

1.1 Origins and aims of the UCS

The Universal Multiple-Octet Coded Character Set, more simply known as the UCS, is intended to provide a single coded character set for the encoding of the written forms of all the languages of the world and of a wide range of additional symbols that may be used in conjunction with such languages. It is intended not only to cover languages in current use, but also languages of the past and such additions as may be required in the future.

The coding provided by the UCS is applicable to the representation, transmission, interchange, processing, storage, input and presentation of the written forms of the languages.

To achieve these aims, the UCS is a multi-part standard under continuous development. The first edition of part 1 was published in 1993 as:

- ISO/IEC 10646-1:1993, Information technology – Universal Multiple-Octet Coded Character Set (UCS) – Part 1: Architecture and Basic Multilingual Plane.

At the time of writing, two Technical Corrigenda and Amendments 1 to 9 (Cor.1-2, AMD.1-9) have been published. Amendments 10 to 27 are in preparation. This guide covers both the base standard and the latest available texts of all these corrigenda and amendments.

The Basic Multilingual Plane (BMP) referred to in this title is a subset of the full UCS that may be encoded in 16 bits, so providing for a total of 65,536 character positions of which so far a large proportion have been allocated. The full UCS allows for 31-bit coding (there is a 32nd bit that is constrained to be zero) and so provides for over two thousand million characters. It should therefore have ample space to fulfill its intention of covering all languages.

For many applications of the UCS, the characters of the BMP are all that will be required. It would be very wasteful of resources if a 32-bit coding was imposed on applications that required only a subset that could be encoded in 16 bits. The UCS therefore specifies more than one form of coding for its characters, in particular providing for encoding of the BMP in a 16-bit form.

The UCS standard will be extended in future by the publication of further parts and of further editions of the existing part 1. Future editions incorporate all published corrigenda and amendments issued prior to their publication. They may in addition include further changes that have not been published separately in this way. It is the declared intention that all such extensions of the UCS will be upwardly compatible, i.e. that they will add the coding of additional characters but that once included, no character will be withdrawn or have its coding changed. The scope of the standard is, however, so wide that such an intention is difficult to maintain. It has, indeed, already been broken in published corrigenda and amendments. Nevertheless it is hoped that it will not be necessary in future to make any further exceptions to this important feature.

1.2 The UCS and UNICODE

The UCS has been developed under the auspices of Joint Technical Committee 1 (JTC 1) of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). ISO maintains a World Wide Web site, which includes its catalogue and ordering information, at the URL http://www.iso.ch.

The JTC 1 subcommittee responsible for the UCS is SC 2, which maintains an official information service at the URL http://www.dkuug.dk/JTC1/SC2.

The UCS is closely related to a commercial character encoding called UNICODE™, prepared by The Unicode Consortium (e-mail: unicode-inc@unicode.org) and published as *The Unicode Standard,*

Worldwide Character Encoding which is now at Version 2.1. Information concerning UNICODE™ is available at the URL http://www.unicode.org.

Roughly speaking, UNICODE™ can be regarded as being the 16-bit coding of the BMP of the UCS. There is effective cooperation between the Unicode Consortium and ISO/IEC JTC 1/SC 2 which should ensure that this compatibility is maintained in future enhancements to the BMP. However, UNICODE™ is not simply the BMP of the UCS as it includes guidelines for usage that are not present in the equivalent ISO standard.

The restriction of UNICODE™ to containing only the BMP of the UCS increases the significance of the positioning of characters in future additions to the UCS. More details of the organization of the BMP are given in section 4 of this guide.

2 Nature of character data

2.1 Characters, character names and glyphs

To understand the role of the UCS in the electronic representation of character data, we first need to consider what is meant, in this context, by a character. The instinctive view of a character, which must be our starting point, is that it is the basic element of some writing system, such as a letter of an alphabet or an ideograph of an ideographic writing system. But this view needs refinement in the context of such an ambitious project as the coding of all the languages of the world.

Characters are identified in their written form by their shape, which is an imprecise concept arising from the ability of the human brain to recognize that two distinct and non-identical objects have the same "shape" It is this ability that enables us to read handwriting, different typefaces, etc. It is a learned ability; most Western people have difficulty in telling whether two similar written Chinese ideographs are in fact "the same character". But it exists and we have to accept that there is an abstract concept of "shape" that underlies the entire nature of written language.

Subtleties enter when we realize that there is context dependence to the recognition of written characters. There are letters with the same shape in the Latin and Greek alphabets, for example, but we do not think of them as the same character. The shape for a Latin capital letter A is recognized as a Greek capital letter alpha when it appears in Greek text. A hyphen is interpreted as a minus sign when it appears in mathematical expressions. Are Greek capital letter omega ( ) and the Ohm sign (symbol for electrical unit of resistance), the same character or not? Historically the Greek letter was adopted as the Ohm sign, but it is a question of opinion as to whether it has by usage now become a symbol in its own right. The viewpoint of the UCS is that they are now distinct characters.

There are also subtleties in the opposite direction. The Greek language uses two distinct written forms for the Greek small letter sigma, depending on whether it is ( ), or is not ( ), the final letter of a word. Printed text often makes use of ligatures (joined letters) for reasons of appearance that have no linguistic basis. For example, printed text in the Latin alphabet often combines a small letter F followed by a small letter I into an ligature:

f + i =

This creates a recognizably distinct shape but it is interpreted as two distinct letters when it is read. These are examples where the shape that represents the character or characters is affected by the context in which the character appears.

Which of these subtleties is important, for the purposes of the electronic encoding of data, depends substantially on the use to which the coding is to be put. A particular application of encoded data is normally concerned either with the visual appearance of encoded symbols, e.g. for printing applications, or with the semantics of the encoded symbols, e.g. for data processing. This has given

rise to two distinct concepts arising from our first idea of a character as the basic element of some writing system. Elements of written data that are distinguished from one another by visual appearance are known as *glyphs*. The term *character* has become specialized to mean elements of written data that are distinguished from one another by semantic interpretation. The formal definitions are as follows:

character: A member of a set of elements used for the organisation, control, or representation of data *(taken from ISO/IEC 10646-1:1993)*.

glyph: A recognizable abstract graphic symbol which is independent of any specific design *(taken from ISO/IEC 9541-1:1991)*.

Characters are distinguished from one another by name, not by form or shape. ISO standards for coded character sets normally include tables that show a representative printed form for each character represented. These printed forms are purely illustrative and are not necessarily distinctive; the same shape (glyph) may be used for more than one character in a table. It is the name, such as LATIN CAPITAL LETTER A, that identifies the character being encoded in each code position. It is a convention adopted by the UCS that the names of characters are composed only from Latin capital letters A to Z, digits 0 to 9, space and hyphen. There are restrictions on the use of digits in names, in particular they may only be used in the names of ideographic characters.

With this distinction in place, we can say that the UCS is a standard that specifies an encoding of characters. The standard shows a representative printed form (glyph image) for each encoded character, but these are not all distinct from one another.

2.2 Graphic characters and control characters

The characters described in the preceding section are all graphic characters, i.e. characters that have a visual representation. Character data also includes characters present for control purposes, such as CARRIAGE RETURN or LINE FEED. These particular control characters have names that originate with the use of electromechanical teleprinters, but they are still used today for the characters used to control paragraph separation in modern text processing systems. They are just two examples of many such non-printing characters that may be required to control the systems used for the display or printing of coded character data.

When data is encoded directly as a sequence of characters, such control characters will appear interspersed in the sequence of graphic characters. They must therefore be assigned code positions along with the graphic characters of the code. Nowadays character data is often transmitted or otherwise processed by means of protocols that separate the control data from the character data. One such protocol is Abstract Syntax Notation One (ASN.1). When such protocols are used, it is not necessary to keep code positions for control characters within the code used for graphic characters as the separation is achieved by other means. However, the UCS does reserve code positions for the use of control characters, to permit use in systems where a single sequence of intermixed graphic and control characters is required.

2.3 Alphabetic, syllabic and ideographic scripts

The world's languages whose characters are encoded in the UCS differ substantially from one another in the extent to which the written forms of the languages can be broken down into constituent elements. The scripts used for written languages fall, for this purpose, into three distinct classes:

- alphabetic scripts, in which the number of distinct letters used in writing is limited and at most a few hundred;

- syllabic scripts, where written symbols each represent a language syllable, in which the number of distinct syllables is limited but may run into several thousand; and

- ideographic scripts, where there is in principle no limit on the number of different ideographs that may be used in writing, other than that imposed by the vocabulary of the language.

In these descriptions the meaning of "limited" is that the number will not increase as further words are added to the language, either in the future or to cover language usage in the past.

These different classes of script have very different requirements in terms of the number of code positions required to represent them in a coded character set. All the alphabetic scripts of the world, taken together, require fewer code positions than does the Chinese ideographic script on its own. The UCS sets aside somewhat over one quarter of its code space in the BMP, a total of 20992 code positions, for the East Asian ideographic scripts of the Chinese, Japanese and Korean languages taken together. A further 11172 code positions are occupied by the Korean Hangul syllabic script. This leaves somewhat over one half of the code space of the BMP for all other scripts of all the other languages of the world that are in current use. This is likely to be more than adequate. The effect of the limitation of space on the encoding of the ideographic scripts is described section 4.4.

2.4 Sequence order and writing mode

The written form of a language is composed of a sequence of script elements. This is true whether the script is alphabetic, syllabic or ideographic. But languages differ from one another in the arrangement of the sequence on paper (or other writing surface). Three arrangements are in common use. The succession of script elements may be written left-to-right (e.g. Latin, Cyrillic and Greek scripts and horizontal Japanese Kanji) or right-to-left (e.g. Hebrew and Arabic scripts), with successive rows being written top-to-bottom, or the script elements may be written top-to-bottom (e.g. vertical Japanese Kanji) with successive rows being written right-to-left.

The sequence order of the characters in an encoding of any script is that of the logical succession of characters, regardless of the writing mode. If the encoding is to be used to create a written presentation of the encoded material, it is up to the application to observe the correct writing mode for the script in use. This is so even for encoded data that intermixes two or more scripts with different writing modes, e.g. text in Latin script containing Hebrew quotations. Where it is required to encode the intended writing mode along with the character data, the control functions SELECT PRESENTATION DIRECTIONS and START REVERSED STRING may be used. Their coding, which makes use of control characters, is defined in ISO/IEC 6429:1992. The first of these functions is used to set the writing mode of the main text. The second is used to reverse the direction temporarily, as in the example of Hebrew quotations within a predominantly Latin script.

Certain characters have semantics that depend on writing direction. The symbols "(" and ">" represent an opening parenthesis and a greater-than sign when they occur in a script written from left to right, but in a script written from right to left they represent a closing parenthesis and a less-than sign respectively. There are provisions within ISO/IEC 10646-1 for such characters to be presented in mirrored form, ")" and "<" in this example, when used with a script written from right to left. However, such mirroring should not be performed automatically since there are separate characters which have these glyphs as their normal form. Specific rules governing such forms of presentation that are given in annexes C and D of ISO/IEC 10646-1.

2.5 Precomposed and decomposed characters

Even within alphabetic scripts, there is ambiguity as to what are the constituent elements of the script. Many scripts use diacritical marks, such as accents and tone marks, as modifiers of basic letters. At what point does one cease the decomposition? Is ê (e circumflex) a letter in its own right or a composite of two separate elements, a letter (e) and an accent (circumflex)? If it is to be regarded as a composite on the grounds that the letter and accent are separated from one another, then what about i (small letter I)? A dotless i is a letter in its own right in the Turkish language. And what about ø (o with stroke), which is a superposition that is not visually in two distinct parts?

The UCS has adopted the view that basic letters and diacritical marks should be assigned encodings as separate graphic characters, but that the composites that are in normal use in current languages should also be encoded as graphic characters in their own right. A character such as a diacritical mark, intended only to be used in conjunction with a base letter, is said in the UCS to be a *combining character*. A composite formed from a base letter and one or more combining characters is called a *composite sequence*.

A composite sequence is not a character, as it is not a member of the set of elements that form the UCS; it is a sequence of such elements. But both graphic characters and composite sequences have visual representations as glyphs, and the same glyph may be the visual representation both of a graphic character of the UCS and of a composite sequence. The glyph é (e acute) is the visual representation both of the character LATIN SMALL LETTER E WITH ACUTE and the composite sequence LATIN SMALL LETTER E followed by COMBINING ACUTE ACCENT.

3 Coding of character data

3.1 Fixed and variable length codes

A coded character set provides an unambiguous relationship between the characters of a specified set and sequences of binary digits (bits) that are used to represent these characters. One of the most well known coded character sets is ASCII, the *American Standard Code for Information Interchange*. This represents 128 characters and uses all possible combinations of 7 bits. But there is no reason other than convenience why each character should be represented by the same number of bits, provided that the structure of the code permits the boundaries between one character coding and the next to be distinguished.

Modern codes used for the interchange and processing of information encode each character by one or more octets, an octet being a sequence of 8 bits. A code is fixed length if each character of the code is represented by the same number of octets and is of variable length if this is not the case. One standardized code of variable length for the Latin script is that specified in

- ISO/IEC 6937, Information technology – Coded graphic character set for text communication – Latin alphabet.

In this code, letters without diacritical marks, and other symbols, are encoded by a single octet. Letters with diacritical marks are considered to be characters in their own right but they are encoded by a sequence of two octets. The first octet identifies the diacritical mark and the second identifies the base letter. This ordering originates with electromechanical printing equipment in which diacritical marks are non-spacing characters. The following character is then superposed on the diacritical mark to form an accented character.

The use of non-spacing diacritical marks in the variable length encoding of ISO/IEC 6937 differs in principle from the use of combining characters in the UCS, but the practical effect is similar. In ISO/IEC 6937 the non-spacing diacritical marks are not considered to be characters in their own right. The octet that represents a particular non-spacing diacritical mark is not a valid encoding on its own, instead it carries an implicit signal that it is to be followed by a second octet in order to complete the encoding.

In contrast to ISO/IEC 6937, the combining characters of the UCS are characters in their own right but they do not have a visual representation on their own. A glyph, giving a visual representation, is only associated with a complete composite sequence in which combining characters form only a part. A code with such a structure is capable of representing more glyphs than there are characters in the code.

3.2 Inadequacy of single-octet codes

The ASCII 7-bit code reserved the first 32 of its 128 code positions for control characters. Of the remaining 96 positions, one was used for the SPACE character and another for a DELETE character, so only 94 positions were left for graphic characters.

Due to the influence of ASCII on the development of coded character sets, early 8-bit codes were structured as two 7-bit codes, conceptually with a left-hand and a right-hand code table distinguished from one another by the eighth bit. Each of the 7-bit codes had the first 32 code positions reserved for control characters, but SPACE and DELETE were not required a second time, leaving 96 positions in the right-hand code table for graphic characters.

Such an 8-bit code is very limiting, as even with the use of combining characters it does not contain sufficient space for the base letters of more than one or two alphabetic scripts. A number of single-octet codes with this construction are defined in the multi-part standard ISO/IEC 8859 and further parts are still under development. Each part contains the specification of a single-octet code that includes the graphic characters of ASCII and which makes no use of combining characters. Although an improvement on the 7-bit ASCII code, each part covers the characters required for only a small selection of the world's languages.

3.3 Limitations of two-octet codes

As there is no possibility of using a single-octet code for an ideographic script, the 8-bit code structure with its ASCII inheritance was extended in the simplest manner that would permit the coding of such scripts. A sequence of two or more octets, each corresponding to a code position for a graphic character from the same half (left-hand or right-hand) of the 8-bit code table, could be taken together to encode a character. A sequence of two octets would then permit 94 x 94 (i.e. 8836), or 96 x 96 (i.e. 9216), characters to be encoded instead of the 94 or 96 permitted by single octet coding. The structure for such codes, together with various code extension techniques, is specified in

- ISO/IEC 2022, Information technology – Character code structure and extension techniques

This provides sufficient space for the encoding of the most commonly used Chinese characters in a single code for use in either the left-hand or right-hand area of such a code structure. The full requirements of Chinese, however, are well illustrated by the Chinese Standard Interchange Code CNS 11643 (1992). This defines 7 such character sets which between them provide for the coding of 48027 Chinese characters. The code extension techniques of ISO/IEC 2022 include code switching mechanisms which permit all such tables to be used in conjunction with one another.

By breaking away from the ISO/IEC 2022 code structure, the full 65536 code positions of a two-octet code become available. There is still the need to provide for the coding of control characters, but this is minimal in comparison with the available space. However, if 48027 of these code positions are required for Chinese characters, a single two-octet code becomes inadequate to cover all the languages of the world.

3.4 The four-octet structure of the UCS

Since the intention of the UCS is to have the capability of covering all characters of all languages, a four-octet structure has been adopted. The most significant bit of the most significant octet is constrained to be zero, which permits its use for private internal purposes in a data processing system. The remaining 31 bits allow for over two thousand million code positions, which should be more than enough for all future needs. For reference the four octets are named, in order from the most significant to the least significant,

- the Group-octet, G-octet or simply G;
- the Plane-octet, P-octet or simply P;

- the Row-octet, R-octet or simply R;
- the Cell-octet, C-octet or simply C.

The entire code space is correspondingly viewed as a four-dimensional structure composed of

- 128 groups, each specified by a value for G;
- 256 planes in each group, each plane specified by a value for P;
- 256 rows in each plane, each row specified by a value for R;
- 256 cells in each row, each cell specified by a value for C.

The values of any octet are specified by two hexadecimal digits 0-9, A-F, in which A through F correspond to the decimal values 11 through 15 respectively. The value of G is restricted to the range 00-7F.

A cell within a plane may be described by four hexadecimal digits giving its R and C values, thus 1234 corresponds to R=12, C=34. In every plane the cells FFFE and FFFF are left unused; FFFE has a special use in signatures for coding identification (see the chapter on serial transmission of the UCS) and FFFF is available whenever a value is required that is guaranteed not to be a valid character code.

The plane with G = 00, P = 00 is known as the Basic Multilingual Plane (BMP). The 34 planes P = 0F, 10, E0-FF in Group 00 and the whole of the 32 groups G = 60-7F are designated as available for private use, outside the scope of standardization. Planes P = 0F, 10 in Group 00 were added by Amendment 1 to those reserved for private use, to enable two private use planes to be accessed by the UTF-16 coding methods specified in that Amendment. There is also a block reserved for private use that lies within the BMP; see section 4.

The UCS is one of two major codes that have developed outside of the constraints of ISO/IEC 2022. The other is the commercially-developed UNICODE™, which was developed strictly as a two-octet code. In the interests of compatibility, ISO and the Unicode Consortium cooperated during the development of both codes to ensure that the BMP coincides with UNICODE™ code table. In addition the UCS standard specifies more than one form for the coded representation of characters. One of these is the two-octet BMP form which, as its name implies, provides for the encoding of the characters of the BMP by the values for their R and C octets alone. This coding is identical to that provided by UNICODE™.

The pressure to ensure that all languages in current use can be represented by UNICODE™, so requiring coding within the BMP, has led to compromises in design that would not have been necessary in a pure four-octet code. Once again it is the space required by the ideographic scripts that is the source of the difficulties. This is explained in more detail in section 4.

4 Basic Multilingual Plane (BMP)

4.1 Relationship to 8-bit codes

For reasons of compatibility, the row of the BMP with R = 00 has been given the structure of an 8-bit code according to ISO/IEC 2022. This requires that

- the code positions 0000-001F and 0080-009F are reserved for the coding of control functions (prior to Amendment 3, only 0000-001F was available for the coding of control functions as 0080-009F was reserved for future standardization);
- the code position 007F is reserved for the DELETE character (for historic reasons that have long since ceased to be relevant);

- the code position 0020 is allocated to the SPACE character.

This enables the coded representation of a control function to be obtained by a simple algorithm from its coded representation in an 8-bit code in accordance with ISO/IEC 2022. The algorithm is described elsewhere in this annex.

The graphic characters in the remaining 190 code positions of row 00 are allocated in accordance with the 8-bit code specified in

- ISO/IEC 8859-1:1997, Information technology – 8-bit single-byte coded graphic character sets – Part 1: Latin Alphabet No.1.

That code, and therefore row 00 of the BMP, contains graphic characters used for general purpose applications in typical office environments in at least the following languages:

- Albanian, Basque, Breton, Catalan, Danish, Dutch, English, Faroese, Finnish, French (with restrictions), Frisian, Gaelic, Galician, German, Greenlandic, Icelandic, Irish Gaelic (new orthography), Italian, Latin, Luxemburgish, Norwegian, Portuguese, Rhaeto-Romanic, Spanish and Swedish.

This incorporation of ISO/IEC 8859-1 in particular makes the cells 21-7E of row 00 have the same allocations as the graphic characters of ASCII, which in its internationally standardized form is also known as the International Reference Version (IRV) of:

- ISO/IEC 646:1991, Information technology – ISO 7-bit single-byte coded character set for information interchange.

4.2 The 5 zones of the BMP

To aid its interpretation and development, the Basic Multilingual Plane is divided into five zones corresponding to the following code positions:

- A-zone: code positions 0000-4DFF but excluding the positions 0000-001F and 0080-009F reserved for control characters and 007F reserved for the DELETE character (leaving 19903 positions)
- I-zone: code positions 4E00-9FFF (20992 positions)
- O-zone: code positions A000-D7FF (14336 positions)
- S-zone: code positions D800-DFFF (2048 positions)
- R-zone: code positions E000-FFFD (8190 positions)

The R-zone terminates at FFFD as positions FFFE and FFFF are reserved; see the section of this guide on the 4-octet code structure of the UCS.

Each zone has a distinctive use:

- the A-zone is used for alphabetic and syllabic scripts together with various symbols;
- the I-zone is used for Chinese/Japanese/Korean (CJK) unified ideographs;
- the O-zone is used for the Korean Hangul syllabic script, and for various other scripts;
- the S-zone is reserved for use with transformation format UTF-16;

- the R-zone is known as the restricted use zone and contains sets of graphic characters for various uses including private use that is outside the scope of standardization.

The transformation format UTF-16 was introduced by Amendment 1 to the first edition of ISO/IEC 10646-1, which also created the S-zone by a splitting of the O-zone. Prior to that amendment the O-zone extended to code position DFFF. UTF-16 extends the two-octet coding of the BMP into a variable-length coding. In that coding the characters of all zones of the BMP (P=00) other than the S-zone are encoded in two octets while in addition characters of any of the fifteen planes P=01 to P=10 (remember that 10 here is a hexadecimal value) are encoded in four octets.

4.3 Alphabetic and syllabic scripts of the A-zone

The A-zone is structured into named blocks, each consisting of a consecutive range of cells. Each block is allocated to a related set of characters, although a block may contain individual cells that are currently unallocated. The characters in the UCS from a particular script may be grouped together in a single block (such as BENGALI) or they may be divided among several blocks (such as BASIC ARABIC and ARABIC EXTENDED). The characters of the Latin script occupy the first four named blocks BASIC LATIN, LATIN-1-SUPPLEMENT, LATIN EXTENDED-A, LATIN EXTENDED-B but in addition there is one further block of Latin characters, LATIN EXTENDED ADDITIONAL, which occurs further into the code table.

Separate from the block structure, but closely related to it, is the concept of a collection of characters. A collection is the subset of characters allocated to a specified range of cells. The difference between a block and a collection is that the cells of a collection need not be consecutive and two collections may overlap. Collections are assigned both a name and a number. Blocks divide the code space into separate areas that are allocated for a coherent purpose. Collections put blocks and/or individual characters together to form subsets of practical significance. A user may then put several collections together to form a subset meeting a particular need, such as communication in English and Hebrew.

Table 1 shows the blocks and collections of the first nine rows of the A-zone, comprising cells 0000-08FF. It gives both the name and the range of cells that comprise the block. With the exception of the collection HEBREW EXTENDED, which is formed from two blocks, there is a one-to-one correspondence between blocks and collections for the characters in these seven rows. The table also gives the number assigned to the collection in the first column; the collection name is the same as that of the block.

Table 1 : Blocks and Collections of rows 00-08 of the UCS

(collection = block, except for collection 13; *,† = contains combining characters; see the section below on combining characters for the significance of these markings)

1	BASIC LATIN	0020-007E
2	LATIN-1-SUPPLEMENT	00A0-00FF
3	LATIN EXTENDED-A	0100-017F
4	LATIN EXTENDED-B	0180-024F
5	IPA EXTENSIONS	0250-02AF
6	SPACING MODIFIER LETTERS	02B0-02FF
7†	COMBINING DIACRITICAL MARKS	0300-036F

8	BASIC GREEK	0370-03CF
9	GREEK SYMBOLS AND COPTIC	03D0-03FF
10†	CYRILLIC	0400-04FF
	(Reserved for future standardization)	0500-052F
11	ARMENIAN	0530-058F
	HEBREW EXTENDED-A (31 further Hebrew characters have been allocated to previously reserved cells in this block by Amd. 7)	0590-05CF
12	BASIC HEBREW	05D0-05EA
	HEBREW EXTENDED-B	05EB-05FF
13*	HEBREW EXTENDED (This collection comprises the two blocks HEBREW EXTENDED-A and HEBREW EXTENDED-B)	
14*	BASIC ARABIC	0600-065F
15*	ARABIC EXTENDED	0660-06FF
85	SYRIAC (added by Amd.27, hence the out-of sequence number)	0700-074F
	(Reserved for future standardization)	0750-077F
86*	THAANA (added by Amd.24, hence the out-of sequence number)	0780-07BF
	(Reserved for future standardization)	07C0-08FF

Certain characters in the blocks LATIN-1-SUPPLEMENT AND LATIN-EXTENDED-B have had their names changed by Technical Corrigendum 1 (1996) since the publication of the first edition of the standard in 1993. In the first of these blocks the characters affected are:

- LATIN CAPITAL LIGATURE AE, renamed to
    - LATIN CAPITAL LETTER AE (ash);
- LATIN SMALL LIGATURE AE, renamed to
    - LATIN SMALL LETTER AE (ash).

In the other block the affected characters are these same characters with added diacritical marks MACRON or ACUTE. The same name changes will be made in the next editions of the parts of ISO/IEC 8859 in which these characters appear.

The next five rows, 09-0D, are allocated to scripts that require the two special characters

- ZERO WIDTH NON-JOINER (code position 200C)
- ZERO WIDTH JOINER (code position 200D)

in the coding of languages written in those scripts. As with rows 00-06, there is a collection corresponding to each block, but for these rows the collection consists of the characters allocated to that block together with these two special characters.

Table 2 shows the blocks and collections of rows 09-0D of the A-zone, comprising cells 0900-0DFF. It gives both the name and the range of cells that comprise the block. The table also gives the number assigned to the collection that consists of the characters allocated to the block together with the additional characters at positions 200C and 200D. The collection name is the same as that of the block on which it is based.

Table 2 : Blocks and Collections of Rows 09-0D of the UCS

(collection = block + 200C + 200D; * = contains combining characters)

16*	DEVANAGARI	0900-097F
17*	BENGALI	0980-09FF
18*	GURMUKHI	0A00-0A7F
19*	GUJARATI	0A80-0AFF
20*	ORIYA	0B00-0B7F
21*	TAMIL	0B80-0BFF
22*	TELUGU	0C00-0C7F
23*	KANNADA	0C80-0CFF
24*	MALAYALAM	0D00-0D7F
84*	SINHALA (added by Amd.21, hence the out-of sequence number)	0D80-0DFF

The remainder of the first 32 rows, namely rows 0E-1F, are either reserved or allocated to further scripts that correspond to collections on a one-to-one basis without additional characters. These are shown in Table 3.

Table 3 : Blocks and Collections of Rows 0E-1F

(collection = block; * = contains combining characters)

25*	THAI	0E00-0E7F
26*	LAO	0E80-0EFF
72*	BASIC TIBETAN (added by Amd.6, hence the out-of sequence number)	0F00-0FBF

	(Reserved for future standardization)	0FC0-109F
28	GEORGIAN EXTENDED (note that the collection number is out of sequence)	10A0-10CF
27	BASIC GEORGIAN	10D0-10FF
29	HANGUL JAMO	1100-11FF
73	ETHIOPIC (added by Amd.10, hence the out-of sequence number)	1200-137F
	(Reserved for future standardization)	1380-139F
75	CHEROKEE (added by Amd.12, hence the out-of sequence number)	13A0-13FF
74	UNIFIED CANADIAN ABORIGINAL SYLLABICS (added by Amd.11, hence the out-of sequence number)	1400-167F
82	OGHAM (added by Amd.20, hence the out-of sequence number)	1680-169F
83	RUNIC (added by Amd.19, hence the out-of sequence number)	16A0-16FF
87*	BURMESE (added by Amd.26, hence the out-of sequence number)	1700-177F
88*	KHMER (added by Amd.25, hence the out-of sequence number)	1780-17FF
	(Reserved for future standardization)	1800-1DFF
30	LATIN EXTENDED ADDITIONAL (one additional Latin character has been allocated to a previously reserved cell in this block by Amd.7.)	1E00-1EFF
31	GREEK EXTENDED	1F00-1FFF

The next eight rows of the A-zone contains symbols of various sorts and for various scripts, including technical and special purpose symbols. These take up rows 20-28 and they are followed by a further seven rows that are at present unallocated. This area of the A-zone is structured as in Table 4.

Table 4 : Blocks and Collections of Rows 20-2F

(collection = block; † = contains combining characters)

32	GENERAL PUNCTUATION	2000-206F

33	SUPERSCRIPTS AND SUBSCRIPTS	2070-209F
34	CURRENCY SYMBOLS	20A0-20CF
35†	COMBINING DIACRITICAL MARKS FOR SYMBOLS	20D0-20FF
36	LETTERLIKE SYMBOLS	2100-214F
37	NUMBER FORMS	2150-218F
38	ARROWS	2190-21FF
39	MATHEMATICAL OPERATORS	2200-22FF
40	MISCELLANEOUS TECHNICAL	2300-23FF
41	CONTROL PICTURES	2400-243F
42	OPTICAL CHARACTER RECOGNITION	2440-245F
43	ENCLOSED ALPHANUMERICS	2460-24FF
44	BOX DRAWING	2500-257F
45	BLOCK ELEMENTS	2580-259F
46	GEOMETRIC SHAPES	25A0-25FF
47	MISCELLANEOUS SYMBOLS	2600-26FF
48	DINGBATS	2700-27BF
	(Reserved for future standardization)	27C0-27FF
80	BRAILLE PATTERNS (added by Amd.16)	2800-28FF
	(7 more rows reserved for future standardization)	2900-2FFF

The next 30 rows contain alphabetic scripts and symbols that are used by languages that also make use of ideographic scripts. The reference to CJK in the titles of some of the blocks of these rows is to unified Chinese/Japanese/Korean characters; see the section on ideographic scripts for more information. The blocks and collections of these rows are given in Table 5.

Table 5 : Blocks and Collections of Rows 30-4D

(collection = block; * = contains combining characters)

49*	CJK SYMBOLS AND PUNCTUATION	3000-303F
50*	HIRAGANA	3040-309F

51	KATAKANA	30A0-30FF
52	BOPOMOFO	3100-312F
53	HANGUL COMPATIBILITY JAMO	3130-318F
54	CJK MISCELLANEOUS	3190-319F
55	ENCLOSED CJK LETTERS AND MONTHS	3200-32FF
56	CJK COMPATIBILITY	3300-33FF
81	CJK UNIFIED IDEOGRAPHS EXTENSION A (Amd.17)	3400-4DBF
	(Reserved for future standardization)	4DC0-4DFF

The CJK COMPATIBILITY block includes many symbols for scientific units that have been coded in Chinese national standards as if they were ideographs. Examples, together with their coding, are

- mm^3 (cubic millimetres)

  SQUARE MM CUBED (coded at 33A3)

- μs (microsecond)

  SQUARE MU S (coded at 33B2)

- rad/s^2 (radians per second per second, a unit of angular acceleration)

  SQUARE RAD OVER S SQUARED (coded at 33AF)

The last 26 rows 34-4D of the A-Zone, now contain CJK Unified Ideographs Extension A (Amendment 17). However, these rows were allocated in the first edition of ISO/IEC 10646-1 to the Hangul syllabic script, divided into three blocks and corresponding collections numbered 57-59. Amendment 5 to this first edition deleted these allocations and created instead an allocation for a substantially larger set of Hangul syllabic characters in the O-zone. This was accepted as a violation of the principle that published allocations would not be changed, but there were compelling reasons to adopt this change. It will not be taken as a precedent for future changes of a similar nature.

4.4 Unified ideographs of the I-zone

The I-zone of the BMP is allocated as a single block to Chinese/Japanese/Korean unified ideographs, and it correspondingly forms a single collection. For completeness this is shown in the following table:

Table 6 : The one Block and Collection of the I-zone

60	CJK UNIFIED IDEOGRAPHS	4E00-9FFF

An informative annex S has been added to ISO/IEC 10646-1 by Amendment 8 which describes the unification procedure. This section of the guide is based on that annex.

The I-zone contains 20992 code positions, of which 20902 are currently allocated to specific ideographs. These ideographs were derived from over 54000 ideographs which are found in various different national and regional standards for coded character sets. A process of unification was applied in which single ideographs from two or more of the source standards were associated together and assigned to a single code position in the I-zone. The ideographs that are thus associated are described, for the purposes of the UCS, as unified. To preserve data integrity, any ideographs that are separately encoded in any one of the source standards were not unified. Also ideographs that are unrelated in historical derivation are not unified. However, some ideographs encoded in two different standards for the same language may have been unified.

The unification process is based on the shapes of the ideographs, analyzed according to a systematic procedure. Any ideograph is composed of geometric elements which may themselves be composite structures and possibly ideographs in their own right. This enables the structure of an ideograph to be described by a component tree, where the top node is the ideograph itself and the bottom nodes are primitive elements. When two ideographs are compared, their component trees are compared to see if they agree in all of the following aspects:

- the number of components;
- the relative position of the components in each complete ideograph;
- the structure of the corresponding components.

If all of these aspects agree then the ideographs are considered to have the same abstract shape and are therefore unified. Annex S to ISO/IEC 10646-1 contains a listing of pairs or triples of ideographs that would have been unified under these rules except for the criteria concerning historical derivation or separate encoding in an existing standard.

Unified ideographs are named and listed in the code pages of ISO/IEC 10646-1 in a manner separate from that used for other scripts. For each unified ideograph, the listing reproduces all (which may only be one) of the graphic symbols (source ideographs) that have been unified into that code position. For each graphic symbol it specifies the source standard from which the graphic symbol is taken and the coded representation of the symbol in that standard. The name assigned to each unified ideograph is algorithmically generated by appending their two-octet coded representation to “CJK UNIFIED IDEOGRAPH-”, for example CJK UNIFIED IDEOGRAPH-4E00.

The information concerning CJK unified ideographs has now been replaced by Amd.13.

4.5 The Hangul syllabics of the O-zone and Yi

Amendment 5 to the first edition of ISO/IEC 10646-1 specified a change in the encoding of Hangul syllabic script. Prior to that Amendment, the last 26 rows of the A-zone (row numbers 34-4D) were allocated to the Hangul syllabic script and the entire O-zone was reserved for future standardization. Due to a major revision of the corresponding Korean national standard shortly after the final text of the first edition was agreed, it became necessary to accommodate substantially more syllabic characters into the UCS. To include these additional characters, the total space required would be almost 44 rows.

It was decided that this was sufficient of an exceptional circumstance to merit violating the principle that code positions, once allocated, should not be changed. The Hangul syllabic characters already encoded would be moved from the A-zone to the I-zone, where there was sufficient space to include both the original and the additional characters in a single block, with a corresponding single collection. The amendment contains the statement that this change is not intended to be regarded as a precedent for other changes of allocation in future editions. This statement will itself be incorporated into future editions.

Amendment 14 has added the syllables and radicals of the Yi script to the O-Zone.

Following these amendments, the O-zone has the structure shown in the following table:

Table 7 : The Blocks and Collections of the O-zone

76	YI SYLLABLES	A000-A48F
77	YI RADICALS	A490-A4CF
	(Reserved for future standardization)	A4D0-ABFF
71	HANGUL EXTENDED	AC00-D7A3
	(Reserved for future standardization)	D7A4-D7FF

Amendment 5 contains a mapping table giving the correspondence between the code positions before and after this amendment for the characters originally allocated to rows 34-4D.

The Hangul syllabic characters are assigned names that follow the naming rules used for alphabetic scripts, e.g. HANGUL SYLLABLE GEOLH (KEOLH) rather than the algorithmic name structure used for the CJK unified ideographs of the O-zone.

4.6 The restricted use R-zone

The R-zone is distinguished from the remainder of the BMP in that its code positions are allocated for use only in special circumstances. There are three distinct uses for the R-zone:

- Private use characters

  These may be specific user-defined characters or may be dynamically-redefinable characters. In either case an agreement is necessary between sender and recipient, outside the scope of ISO/IEC 10646, if these are to be exchanged meaningfully between two communicating parties.

- Presentation forms of characters

  A presentation form is an alternative form, for use in a particular context, to the nominal form of a character or sequence of characters from the other zones of graphic characters. The transformation from the nominal form to the presentation forms may involve substitution, superimposition or combination. The rules for such transformations are outside the scope of ISO/IEC 10646. Presentation forms are not normally intended to be used as a substitute for the nominal forms, but specific applications may use them in this way for particular purposes such as compatibility with existing devices. The specification of presentation forms within ISO/IEC 10646, an example of which is LATIN SMALL LIGATURE FI at code position FB01, blurs the distinction between characters and glyphs discussed elsewhere in this annex.

- Compatibility characters

  Compatibility characters are included in the UCS primarily for compatibility with existing coded character sets to allow two-way code conversion without loss of information.

As with the other zones, it is divided into blocks and collections but the block for private use consists, by its very nature, only of unallocated code positions. The structure of this zone is as follows:

Table 8 : The Blocks and Collections of the R-zone

(collection = block; *,† = contains combining characters)

61	PRIVATE USE AREA	E000-F8FF
62	CJK COMPATIBILITY IDEOGRAPHS	F900-FAFF
63*	ALPHABETIC PRESENTATION FORMS	FB00-FB4F
64	ARABIC PRESENTATION FORMS-A	FB50-FDFF
	(Reserved for future standardization)	FE00-FE1F
65†	COMBINING HALF MARKS	FE20-FE2F
66	CJK COMPATIBILITY FORMS	FE30-FE4F
67	SMALL FORM VARIANTS	FE50-FE6F
68	ARABIC PRESENTATION FORMS-B	FE70-FEFE
	(The single character at code position FEFF is not in any of the blocks into which the BMP is divided. Its significance is explained in the chapter of this guide on Serial Transmission of the UCS)	FEFF
69	HALFWIDTH AND FULLWIDTH FORMS	FF00-FFEF
70	SPECIALS	FFF0-FFFD

Recall that the final two positions FFFE, FFFF are required to be left unused in every plane of the UCS. The collection numbered 200 is one of a number of special-purpose collections that have been assigned numbers in the range 200-299. See section 7 of this guide on repertoires and subsets for more information.

5 Visual representation of characters

5.1 Combining and non-combining characters

The concept of a combining character was discussed above in the section of this guide on precomposed and decomposed characters. This section describes their use in more detail.

The definition given in ISO/IEC 10646-1 is:

> **combining character**: A member of an identified subset of the coded character set of ISO/IEC 10646 intended for combination with the preceding non-combining character, or with a sequence of combining characters preceded by a non-combining character.

The fact that combining characters are coded following the base non-combining character separates them in principle from the non-spacing diacritical marks used for coding purposes in the variable-length code of ISO/IEC 6937. The nature of a non-spacing symbol, based on the capabilities of electromechanical printing devices, requires that it is received and printed by such a device before the base character with which it is combined. Combining characters in the UCS behave in a more intuitive way; it is more natural to apply diacritical marks to a character that is already known or written than to one that is to be notified later on.

Combining characters are not an essential part of the coding of the Latin script. They are modifiers of letters and all combinations of base letter and diacritical mark in normal use in languages written in the Latin script are separately coded as precomposed characters. Some examples which illustrate the level to which the coding of precomposed Latin characters has been taken are:

- 0149 LATIN SMALL LETTER N PRECEDED BY APOSTROPHE
- 0171 LATIN SMALL LETTER U WITH DOUBLE ACUTE
- 01E0 LATIN CAPITAL LETTER A WITH DOT ABOVE AND MACRON
- 1E1C LATIN CAPITAL LETTER E WITH CEDILLA AND BREVE

However, diacritical marks are also used in the Latin script for reasons other than as part of normal spelling, for example to indicate stress positions in pronunciation. This may require combinations of letter and mark that are not used in normal language. The entire range of diacritical marks used with Latin (and Greek) scripts is available in collection 7, COMBINING DIACRITICAL MARKS.

A similar situation holds for the Greek script, both for its monotonic and polytonic forms. However, many other scripts such as Arabic and the Indic scripts (Devanagari, Gujarati, etc.) are written in such a manner that combining characters are an essential part of coding. The Indic scripts, for example, represent vowels by combining marks. The Cyrillic script falls between these two situations; whether combining characters are or are not essential depends on the language being represented.

The use of combining characters can add significantly to the difficulties of processing encoded text. For this reason, ISO/IEC 10646-1 defines three distinct levels of implementation in which either none (level 1), or some (level 2), or all (level 3) of the combining characters of the UCS are permitted to be encoded. These levels are described in more detail elsewhere in this annex. At any specified level of implementation, the defined character collections of the UCS have to be interpreted as if any combining characters whose use is not permitted at that level have been removed.

These differences between scripts lead to three distinct situations regarding the character collections of the UCS:

- Collections not containing combining characters at any level of implementation

  These are the collections not marked with either * or † in the tables given above. They include all the collections of graphic characters for the Latin and Greek scripts and also the Basic Hebrew collection. Text requiring only these collections can be encoded at implementation level 1.

- Collections containing combining characters when used at level 3 but not at levels 1 or 2

  These are the collections marked with † in the tables given above. At present there are four collections of this nature, but three of them (those containing the word COMBINING in their name) are empty at levels 1 and 2. The only collection of this nature available for use at levels 1 and 2 is the CYRILLIC collection. Much text written in the Cyrillic script does not make use of the combining characters and so can be encoded at implementation level 1. When they are required, it is necessary to use level 3.

- Collections containing combining characters when used at either of levels 2 and 3

  These are the collections marked with * in the tables given above. They include the HEBREW EXTENDED collection and all the collections for the Indic scripts. Much text requiring these collections can be encoded at level 2 but use of level 1 is not normally practicable.

It is the general intention of the UCS that for most purposes it will not be necessary to use a level 3 implementation but that the choice between levels 1 and 2 will depend on the character collections to be used.

5.2 Composite sequences

The definitions clause of ISO/IEC 10646-1 includes the following terms:

> **composite sequence:** A sequence of graphic characters consisting of a non-combining character followed by one or more combining characters.
>
> **graphic symbol:** The visual representation of a graphic character or of a composite sequence.
>
> **repertoire:** A specified set of characters that are represented in a coded character set.

It also contains the following notes:

- A graphic symbol for a composite sequence generally consists of the combination of the graphic symbols of each character in the sequence.
- A composite sequence is not a character and therefore is not a member of the repertoire of ISO/IEC 10646.

The term glyph is not used in ISO/IEC 10646 but its use enables us to divide the formation of a graphic symbol into two stages:

1. The non-combining character or composite sequence is mapped to a glyph, which is an abstract graphic symbol;
2. An image of the glyph is then formed according to some presentation process, to form a real graphic symbol.

This division separates all aspects such as the font to be used, its size and emphasis, *etc*. from the selection of the appropriate glyph. The process of creating the image of the appropriate glyph will in general be very complex, but it is entirely separate from the process of selecting the appropriate glyph to represent the character data. The mapping from the character data (single character or composite sequence) to the glyph is from one abstract entity to another, devoid of all complications arising from the actual presentation process.

The definitions quoted above give rise to the following features of the mapping from character data to glyphs:

- A single non-combining character (which is in the repertoire of the UCS) and a composite sequence (which is not in the repertoire) may map to the same glyph, *e.g.* the character LATIN SMALL LETTER E WITH ACUTE and the composite sequence LATIN SMALL LETTER E followed by COMBINING ACUTE ACCENT.
- A composite sequence may map to a glyph which represents a character that is not present in the UCS, *e.g.* LATIN SMALL LETTER G followed by COMBINING GRAVE ACCENT; there is no character LATIN SMALL LETTER G WITH GRAVE in the UCS, although LATIN SMALL LETTER G WITH ACUTE is present. Characters which can be represented in this way by the UCS, but which are not in the UCS, are not considered to be part of the repertoire of the UCS.
- A given piece of text may have many equally valid representations by a string of characters, depending on whether precomposed or decomposed characters are used.

- As there are no restrictions on the use of combining characters (in the levels of implementation at which they are permitted at all), many composite sequences will not map to any meaningful glyph. This applies in particular to composite sequences in which non-combining characters from one script are followed from combining characters from another script. This is not forbidden but it is unlikely to be meaningful.

These features make the UCS very different from ISO/IEC 6937, despite the similarity at first sight between the combining characters of the UCS and the non-spacing diacritical marks of ISO/IEC 6937. This latter standard has the following features which may be compared with the list given above for the UCS:

- The letters which can be formed from a base letter preceded by a non-spacing diacritical mark are part of the repertoire of ISO/IEC 6937.

- A given piece of text has a unique coding in terms of ISO/IEC 6937 since accented characters are available only in decomposed form (*i.e.* coded with the aid of non-spacing diacritical marks).

- There is a normative list of the characters comprising the repertoire of the standard, so constructions that have no meaningful glyph, such as NON-SPACING GRAVE ACCENT followed by POUND SIGN, are not conformant to ISO/IEC 6937. Unfortunately this has resulted in certain combinations of grave and acute accents, and of diaeresis, with the letters W and Y, used in the Welsh language, being absent from the repertoire of that standard even though they have natural representations in terms of the available non-spacing diacritical marks.

5.3 Use of multiple combining characters

The UCS permits more than one combining character to be applied to a single non-combining character. When this occurs, it lays down rules for their interpretation. In outline, these are as follows:

1. When two combining characters can interact, then by default they are to be positioned in the sequence in which they are received, working from the base character outward. For example, if a combining macron is followed by a combining diaeresis then the diaeresis would appear above the macron, but if the order were reversed then the macron would be above the diaeresis.

2. Some specific combining characters override the default behaviour by being positioned horizontally, rather than vertically, in relation to one another or by forming a ligature with an adjacent combining character. In this case they are positioned in the sequence in which they are received, working in the same direction (left-to-right or right-to-left) as the script is written.

3. When two combining characters do not interact, such as when one is positioned above the base character and the other below it, then the order in which they are coded does not affect the resulting glyph.

6 Referencing of characters

6.1 Identification of characters for migration to the UCS

Since a character is an abstract concept, the question of whether two characters are or are not “the same” is not a trivial one. If we recall the definition:

> **character:** A member of a set of elements used for the organisation, control, or representation of data (taken from ISO/IEC 10646-1:1993)

then there is no problem within a single coded character set, since any such set must clearly specify its members. But frequently, in the transmission or processing of character data, that data needs to be converted from one coded representation to another. This is a particular problem in the migration of existing applications from other character set codes to the UCS. The question then arises of identifying "the same character" in two different character sets.

One cannot just look at the characters, in a visual representation, since distinct characters may have the same glyph. The question should be whether they are both used in the same way in the organisation, control or representation of data. But the specification of a coded character set does not specify how its characters should be used; that is outside of its scope. It merely makes the characters available for use. The "sameness" of characters from different coded character sets is therefore ultimately a matter of convention or of definition.

One of the largest resources of coded character sets is the *International Register of Coded Character Sets for use with Escape Sequences*, maintained and published by the Registration Authority for ISO 2375 in accordance with the procedures of that standard. Those procedures specify how to compare two coded character sets, as follows. Two sets are deemed to be identical if

- the number of characters is the same;
- the names of the characters are the same according to the terminology of the Registration Authority;
- the same positions are used for the same characters;
- both sets are of the same type, in particular both a C0 or a C1 set;
- the definitions of control characters are functionally equivalent (a more restricted definition is not considered equivalent);
- graphic characters have the same geometric shape apart from aesthetic variations between fonts;
- any non-spacing graphic characters are in the same positions.

If we abstract from this those aspects which compare individual characters, rather than their code positions or overall aspects of the complete set, we see that two graphic characters are regarded as identical if

- they have the same name according to the terminology of the Registration Authority;
- they may be represented by the same glyph;
- for characters intended for combining with other characters then the rules for creating combinations are the same (in ISO 2375 the only recognized form of combination is the use of non-spacing characters).

The first of these requirements permits the Registration Authority to change the name (for example, from that used in the source standard whose code is being registered) to bring it into a standardized form. It is the policy of SC2, the ISO/IEC JTC1 sub-committee responsible for coded character set standards, to align the names of characters in its published standards with those used in ISO/IEC 10646. When necessary, renaming will take place when standards are next revised. Such renaming will ensure that two characters are given distinct names if they have distinct glyphs or distinct combining procedures. It follows that

- two graphic characters, from different coded character sets, should be regarded as the same if they have the same name according to the character naming guidelines of ISO/IEC 10646.

6.2 Naming guidelines of the UCS

The naming guidelines of the UCS are given in annex K of ISO/IEC 10646. They include the following:

- By convention, only Latin capital letters A to Z, space, and hyphen shall be used for writing the names of characters.

  NOTE – Names of ideographic characters may also include digits 0 to 9 provided that a digit is not the first character in a word.

- In some cases, the name of a character can be followed by an additional explanatory statements not part of the name. These statements shall be in parentheses and not in capital Latin letters except the initials of the word where required.

- The name of a character shall wherever possible denote its customary meaning, for examples PLUS SIGN. Where this is not possible, names should describe shapes, not usage; for example: UPWARDS ARROW.

- The names shall be constructed from an appropriate set of the applicable terms of the following grid and ordered in the sequence of this grid

  1. Script (e.g. Latin, Cyrillic, Arabic – letters that are elements of more than one script are considered different even if their shape is the same. This is not necessarily true of non-letter characters such as punctuation marks, even where the usage differs between scripts. In such cases the name reflects the most customary use, with alternative names in parentheses)

  2. Case (e.g. capital, small).

  3. Type (e.g. letter, ligature, digit).

  4. Language (e.g. Ukrainian – only included to remove ambiguity, such as between CYRILLIC CAPITAL LETTER I and CYRILLIC CAPITAL LETTER BYELORUSSIAN-UKRAINIAN I, which are distinguished by having different glyphs).

  5. Attribute (e.g. final, sharp, subscript, vulgar).

  6. Designation (e.g. customary name, name of letter – the names of letters of all scripts other than Latin are represented by their transcription in the language of the first published International Standard).

  7. Mark(s) (e.g. acute, ogonek, ring above, diaeresis).

  8. Qualifier (e.g. sign, symbol).

- There are exceptions to the above rules. Traditional names such as APOSTROPHE and COMMERCIAL AT, shall be acceptable as names and alphanumeric identifiers shall be used for ideographic characters.

Not all of the eight terms in this numbered list need be present. Examples of character names, with term numbers added after each name element, are

- LATIN (1) SMALL (2) LETTER (3) DOTLESS (5) J (6) WITH STROKE (7)

- DIGIT (3) FIVE (6)
- LEFT (5) CURLY (5) BRACKET (6)
- COMBINING (5) ACUTE (7) ACCENT (8)

These guidelines are sufficiently clear that there are very few cases in which it is unclear whether two characters from different coded character sets should have the same name under them. Here are some examples of naming problems.

- The changes in Technical Corrigendum 1 concerning Æ were said to be changes of name, for example from LATIN CAPITAL LIGATURE AE to LATIN CAPITAL LETTER AE.

  Both names are constructed according to the naming guidelines and they differ in one name element, namely Type. They therefore are names for two different characters with the same glyph. The Technical Corrigendum changed the characters allocated to the six code positions affected, it did not rename six characters. This was, however, a correction of an editorial error and not a change of coding from the intentions of the first edition.

- The second edition (1994) of ISO/IEC 6937 contains a character MUSIC NOTE (with a musical quaver as the illustrative glyph). This is not present in the UCS, but that does contain the two characters QUARTER NOTE and EIGHTH NOTE, which have more specific names and illustrative glyphs of a musical crotchet and quaver respectively. Comparison of the glyphs in the two standards shows that MUSIC NOTE should be treated as the character EIGHTH NOTE, not as QUARTER NOTE.

- The first edition (1983) of ISO/IEC 6937, which preceded current naming guidelines, contained a character with the name “small letter g with acute accent”. In 8.3 of the second edition it states that this character has been renamed as LATIN SMALL LETTER G WITH CEDILLA in order to align with ISO/IEC 10367 (the cedilla being placed above the g for presentation purposes). However, the UCS contains both LATIN SMALL LETTER G WITH CEDILLA (in the collection LATIN EXTENDED-A) and LATIN SMALL LETTER G WITH ACUTE (in the collection LATIN EXTENDED-B). The justification for the name change is that the original name was in error; the character concerned was always intended to be the small letter corresponding to “capital letter g with cedilla” but was named erroneously due to the positioning of the diacritical mark.

6.3 Linguistic translation of character names

Because of the significance of the names of characters in constructing correspondences between the UCS and other coded character sets, it has been controversial within the relevant sub-committee ISO/IEC JTC1/SC2 as to whether the names of characters may be translated when the text of ISO/IEC 10646-1 is translated into another language. It has recently been agreed that the names of characters may be translated.

One effect of this decision is that names will no longer serve as language-independent unique identifiers of characters. They retain their central role in determining whether characters from different coded character sets are or are not the same, but the comparison of names must take place in a common language.

6.4 Unique identifiers for characters

If names of characters are to be translatable, there becomes a need for some other form of unique identifier for characters that is language independent. Since the aim of the UCS is to include all the world's characters, this enables the coding of a character in the UCS to be used as an identifier of that character in all situations, including in the specification of other coded character sets. Such a scheme

would solve, for the future, the problem of comparing characters from different coded character sets. However, in order to add such identifiers to existing character sets as they are revised, it is first necessary to create a correspondence between the set concerned and the UCS by means of names as described above.

Amendment 9 to ISO/IEC 10646-1 proposes several alternative forms for unique identifiers constructed from UCS code positions. These have the following constructions, in which hhhhhhhh represents the eight hexadecimal digits that represent the code position in the UCS and kkkk represents the last four of these digits for characters of the Basic Multilingual Plane (BMP):

- hhhhhhhh or -hhhhhhhh or T-hhhhhhhh or U-hhhhhhhh;
- kkkk or +kkkk or T+kkkk or U+kkkk.

The significance of the optional prefixes is as follows:

- a minus sign indicates that the numeric form is the eight-digit form, a plus sign indicates that it is the four-digit form;
- a letter T indicates that the identifier refers to the character at the specified code position before the application of Amendment 5 to the first edition of ISO/IEC 10646-1, this amendment being the reallocation of Hangul syllabic characters from the A-zone to the O-zone;
- a letter U indicates that the identifier refers to the character at the specified code position after this application of Amendment 5.

If there is no prefix letter then the relevant amendment level is unspecified. The three forms (no prefix letter, T prefix, U prefix) coincide unless hhhhhhhh lies in the range 00003400 to 00004DFF inclusive. For this range, the correspondence between the T and U forms is given by the mapping table in the annex to Amd.5. As an example:

- T+340F and U+AC19 identify the same character.

The prefix letters, and the letters A to F used as hexadecimal letters, may be written either as capital letters or as small letters.

6.5 Unique identifiers for glyphs

The unique identifiers described above for characters are based on the International Standard ISO/IEC 10646. There is also an internationally agreed assignment of unique identifiers to glyphs, but this is instead based on an International Registration Authority. The registrar is the Association for Font Information Interchange and the register operates under procedures laid down in ISO/IEC 10036.

Glyphs registered under ISO/IEC 10036 are assigned an identifier by the Registration Authority that is a hexadecimal number in the range from 0 to FFFFFFFF. This is the same range of values as that used for identifiers of characters in accordance with ISO/IEC 10646. For the characters of ASCII the same value has been assigned to one possible glyph for each character as is assigned to the character in the ASCII code, and therefore as also in the UCS. For example, the character LATIN CAPITAL LETTER A has the character identifier U+0041 and is represented by the glyph “A” which has the glyph identifier 41 (hexadecimal). However, certain characters of the ASCII code have had their interpretation refined as coded character sets have developed over time. This has led to departures from a strict correspondence even for the ASCII code. In particular:

- U+0060 is the character GRAVE ACCENT but glyph identifier 0060 is a left single quotation mark (the glyph identifier for a grave accent is 00C1).

The use of code positions 27 (U+0027 is APOSTROPHE) and 60 for right and left single quotation marks was an allowed alternative in the original ASCII code. The glyph for a right single quotation mark is acceptable also for an apostrophe, but that for a left single quotation mark is not acceptable as a grave accent. These ASCII alternatives are still present in the registration entry under ISO 2375 for the ASCII code, namely ISO IR-6 in the *International Register of Coded Character Sets to be used with Escape Sequences*, as this entry dates from 1975. Register entries, once made, cannot be revised (other than in exceptional circumstances and if the possibility of revision was stated in the original entry). However, these alternatives are not present in the international standard equivalent to ASCII, namely the International Reference Version (IRV) of ISO/IEC 646:1991. Nevertheless, that standard states explicitly that its IRV may be identified as ISO IR-6.

For use in a wider context, ISO/IEC 9541-1 specifies a structured-name form for the identification of glyphs registered under ISO/IEC 10036. These have the form

- ISO/IEC 10036/RA/Glyphs::*nnnn*

where *nnnn* is a sequence of decimal digits, beginning with a non-zero digit, which represents the hexadecimal value of the glyph identifier assigned by the Registration Authority. The concept of a structured-name is specified normatively in ISO/IEC 9541-2, which gives both ASN.1 and SGML forms for such names.

7 Repertoires and subsets

7.1 The concept of repertoire

There is really only one concept of a repertoire, namely a repertoire is a specified set of characters. However, the concept is defined slightly differently in different character set standards and it is interpreted in ways that may differ from one's expectations. Two particular definitions are

> **repertoire:** A specified set of characters that are represented by one or more bit combinations of a coded character set. [ISO/IEC 6937]
>
> **repertoire:** A specified set of characters that are represented in a coded character set. [ISO/IEC 10646-1]

For completeness, a coded character set also has a formal definition

> **coded character set:** A set of unambiguous rules that establishes a character set and the one-to-one relationship between the characters of the set and their bit combinations. [ISO/IEC 6937]
>
> **coded character set:** A set of unambiguous rules that establishes a character set and the relationship between the characters of the set and their coded representation. [ISO/IEC 10646-1]

It is instructive to see how these two standards differ in their use of the concept of repertoire. Recall that ISO/IEC 6937 is a standard that bases a variable-length encoding of characters from the Latin script on forming combinations of non-spacing diacritical marks with unaccented letters. It is based on two separate 7-bit coded character sets that are separately registered under ISO 2375. The *primary set of* ISO/IEC 6937 is the left-hand set, coded in an 8-bit code as 20 to 7E This is precisely the ASCII set registered as ISO-IR 6. The *supplementary set of ISO/IEC 6937* is the right-hand set, coded as A0 to FF, which contains both the non-spacing diacritical marks and other (spacing) characters.

The repertoire of ISO/IEC 6937 is specified separately, as a list of characters together with their (variable length) coded representations. It consists of 333 characters, including SPACE. Its characters include the accented characters that are coded by two octets, the first representing a non-

spacing diacritical mark from the supplementary set and the second representing an unaccented letter from the primary set. The repertoire of ISO/IEC 6937 does not include the non-spacing diacritical marks as characters in their own right.

This is entirely consistent with the definition of a repertoire. The repertoire of ISO/IEC 6937 is established by that standard as a specific list of characters, each of which is represented by one or more bit combinations. It is quite separate from the union of the repertoires of the primary and supplementary sets of ISO/IEC 6937, which consists of the 191 characters, including SPACE, each coded by one octet. That repertoire does include, say, NON-SPACING ACUTE ACCENT, but it does not include LATIN SMALL LETTER E WITH ACUTE, while the repertoire of ISO/IEC 6937 includes the latter character but not the former one.

The concept of repertoire as used in ISO/IEC 10646 corresponds in the context of ISO/IEC 6937 to that of the union of its primary and supplementary sets, not to that of ISO/IEC 6937 itself. The repertoire of ISO/IEC 10646 consists of the characters that are assigned to code positions within the 31-bit coding space of the UCS. It therefore includes combining characters (which are the nearest equivalent in ISO/IEC 10646 to the non-spacing diacritical marks of ISO/IEC 6937) but does not include either composite sequences or characters, such as LATIN SMALL LETTER G WITH GRAVE, which have glyphs that can be represented by composite sequences.

There is a faint indication of this difference in the definitions given in these two standards. In ISO/IEC 6937 the definition refers to characters *represented by* bit combinations; in ISO/IEC 10646-1 it refers to characters *represented in* the coded character set. There is no conflict, since it is the definition of a coded character set that is crucial. A coded character set is first required to establish a character set, before it assigns coding. That character set is then the repertoire of the coded character set. A repertoire, composed of characters, is therefore whatever the relevant standard says it is. It is, in principle, quite distinct from the set of glyphs that may be represented by the characters of the repertoire. For many purposes it is this set of glyphs that is relevant, not the set of characters used to represent them. But describing or specifying this set of glyphs is outside of the scope of standards for coded character sets.

7.2 Levels of implementation of the UCS

There are three levels of implementation specified in ISO/IEC 10646, distinguished from one another by limitations on the characters that may be encoded at the level concerned. They are as follows:

- Implementation level 1

  At level 1, the prohibited characters are those from the HANGUL JAMO block and all combining characters.

- Implementation level 2

  At level 2, the prohibited characters are those from the HANGUL JAMO block and a specific subset of combining characters listed in annex B of the standard.

- Implementation level 3

  At level 3, there are no restrictions on the characters that may be used.

Hangul Jamo characters are used in the Hangul syllable composition method. A sequence of two or three Hangul Jamo characters has a glyph that represents a syllable. Hangul syllables also have precomposed coding in the HANGUL EXTENDED block of the I-zone of the BMP. The relationship between coding in terms of Hangul Jamo and that as a single syllabic character is similar to that between the precomposed and decomposed forms of Latin characters with diacritical marks. However, there is no distinction for the Hangul Jamo characters corresponding to that between the non-combining and combining characters of a composite sequence. No Hangul Jamo characters have

a meaning in isolation within the Hangul script. For this reason it is specifically stated that the characters of the HANGUL JAMO block are *not combining characters*. Note that the Hangul syllabic characters of the HANGUL EXTENDED block are permitted at all levels of implementation.

The chapter on visual representation of characters gives more information about the scripts that can be represented at the different levels of implementation.

7.3 Collections and subsets

A collection of characters consists of the characters of the UCS that are allocated to code positions lying within one of the ranges specified for this purpose in annex A of ISO/IEC 10646-1. Each collection is assigned both a number and a name. There is a collection associated with, and frequently identical to, each block into which the BMP is divided. These collections, together with their names and numbers, are listed in the chapter of this guide on the Basic Multilingual Plane (BMP). It should be noted that, as a collection is defined by a range, it may include code positions which have not been assigned characters. An amendment to the standard may allocate characters to such code points. Thus the repertoire defined in a collection may change over time. This is not always desirable, so the notion of a fixed collection was introduced in Corrigendum No.2. As a consequence the definition of a fixed collection has to be much more precise in that no range can contain unassigned code points.

Two different collections of characters may overlap, but of those associated with specific blocks the only overlap is that two of the four characters comprising the collection ZERO-WIDTH BOUNDARY INDICATORS are also present in collections for a number of specific scripts. A number of other specialized collections are defined in annex A which put together selections of characters that are also present in other collections. These consist of script-specific formatting characters and alternate forms. There are also two collections related to the permitted levels of implementation. One consists of all combining characters and the other of those combining characters that are not permitted in an implementation at level 2. Finally there are five large collections (two of which are fixed collections) defined as follows:

Table 9 : Large collections of the UCS

299	BMP FIRST EDITION Note: a fixed collection containing only characters contained in the first edition prior to any amendments.	See ISO/IEC 10646-1 A.3
300	BMP	0000-D7FF, E000-FFFD
301	BMP-AMD.7 Note: a fixed collection containing those characters of the first edition as amended by amendments 1 to 7.	See ISO/IEC 10646-1 A.3
400	PRIVATE USE PLANES	G=00, P=0F, 10, E0-FF
500	PRIVATE USE GROUPS	G=60-7F

The specifications of collections 300 and 400 were changed by Amendment 1 consequent on the introduction of the S-zone and its reservation for the use of UCS Transformation Format 16.

A subset is a more general term that refers to any identified set of characters from the entire repertoire of the UCS. Two alternative means of specifying subsets are recognized within ISO/IEC 10646-1:

- Limited subset

  A limited subset is specified by giving explicitly a list of the graphic characters in the subset. They may be listed by their names or their code positions in the UCS.

- Selected subset

  A selected subset is specified as a list of collections. A selected subset shall always include the BASIC LATIN collection.

A selected subset is more restricted than a limited subset in its permitted content, but it has two great advantages. It is much more concise to list collections rather than individual characters. Also, annex M of ISO/IEC 10646-1 specifies by algorithm an ASN.1 object identifier that may be used to identify a selected subset of the UCS within any context in which OSI protocols are used.

A limited subset may be assigned an ASN.1 object identifier, but only by means outside the scope of ISO/IEC 10646-1. The following European pre-standard:

- ENV 1973:1995 Information technology – European Subsets of ISO/IEC 10646-1

contains the definition of a limited subset (the Minimum European Subset) and assigns an ASN.1 object identifier to it. It also describes a selected subset (the Extended European Subset) that has an ASN.1 object identifier assigned in accordance with the algorithm of ISO/IEC 10646-1.

7.4 Significance of subsets for conformance to the UCS

Because of the size and open-ended nature of the repertoire of the UCS, conformance to ISO/IEC 10646-1 does not require the ability to handle all of the characters in the repertoire. Instead, a claim of conformance for information interchange is required to identify:

- a specific method of coding (see the chapter on coding methods of the UCS);
- a specific subset of characters (see above);
- a specific level of implementation (see above);

A separate definition of conformance is given for conformance of a device. For this purpose a device is a component of information processing equipment which can transmit and/or receive coded information, such as an input/output device, an application program or a gateway function. A claim of conformance for a device is required to specify the above three items and in addition

- a specific selection of control functions that are used in conjunction with the UCS (see the chapter on control functions).

The precise meaning of conformance to ISO/IEC is specified in ISO/IEC 10646 and will not be reproduced here. The important aspect here is that conformance only requires support of the UCS within the limits determined by these specified items.

7.5 Subsets as an aid to migration from 8-bit codes

The ability to conform to ISO/IEC 10646 while supporting only a subset of its characters is a great aid to migration from other coded character sets. In particular it permits support to be developed collection by collection. It is only in a few cases that there is a direct correspondence between the collections defined in ISO/IEC 10646-1 and the repertoires of other standardized coded character sets.

However, expansion of support one collection at a time eases substantially the effort required, such as the development glyphs for additional characters.

The assignment by ISO/IEC 10646 of an ASN.1 object identifier for any selected subset provides a means within OSI protocols for an application to notify its peer, in any communication, of the collections that it supports. The Extended European Subset (EES) specified in ENV 1973 consists of the collections numbered 1-11, 27-28, 30-48, 63, 65 and 70. These contain 4013 code positions, of which 3095 are currently assigned to characters. These are all the collections that contain characters of the Latin, Greek, Cyrillic, Armenian and Georgian scripts together with other characters of the International Phonetic Alphabet and a wide range of symbols used for academic, commercial and scientific purposes within Europe. This subset is defined as guidance for product developers, but it in no way restricts the ability of any developer to extend support to either a smaller or a larger range of collections than that of the EES.

8 Coding methods of the UCS

8.1 The coding alternatives

At present there are four coding methods specified within ISO/IEC 10646-1, any one of which can be specified in a claim of conformance to that standard. These methods have been assigned acronyms for easy reference, as follows:

- UCS-2 is the Two-octet BMP form of coding;
- UCS-4 is the Four-octet canonical form of coding;
- UTF-16 is UCS Transformation Format 16, which was added to ISO/IEC 10646-1 by Amendment 1;
- UTF-8 is UCS Transformation Format 8, which was added to ISO/IEC 10646-1 by Amendment 2.

The first edition of ISO/IEC 10646-1 contained a specification of a transformation format UTF-1 but this was deleted from the standard by Amendment 4 and is not available as a coding method in a claim of conformance to ISO/IEC 10646-1.

8.2 UCS-2: Two-octet BMP form

The two-octet BMP form of coding permits the use of characters from the BMP with each character represented by two octets. The BMP is specified by the G-octet and P-octet both being 00. In this form of coding a character is represented by the R-octet and C-octet of its code position. When expressed as a four-digit hexadecimal number the R-octet gives the most significant two digits and the C-octet gives the least significant two digits. UCS-2 provides a fixed-length coding for all the characters of the BMP.

8.3 UCS-4: Four-octet canonical form

The four-octet canonical form permits the use of all characters of ISO/IEC 10646 with each character represented by the G-octet, P-octet, R-octet and C-octet of its code position. These are taken in decreasing order of significance in the expression of a code position as an eight-digit hexadecimal number. UCS-4 provides a fixed-length coding for all the characters of the UCS.

8.4 UTF-16: UCS Transformation format 16

Once characters start to be allocated outside of the BMP, it will no longer be possible to use UCS-2 to encode all the allocations that have been made. However, a transition to UCS-4 instantly halves the

rate of transfer of data through a communication link or the amount that can be stored on a given storage medium. This effect occurs even if the transition to UCS-4 has been made in order to accommodate only very occasional characters coded outside of the BMP.

The transformation format UTF-16 has been designed to avoid this halving of capacity, by means of a variable-length coding. It provides a means of coding any character within the first 17 planes P=00-10 of Group 00 such that the coding of any character within the BMP (Plane 00) is unchanged from its UCS-2 form. This multiplies the number of available code positions by 17 when compared with the BMP, but the number of octets used for coding is increased only for the (occasional) characters that are allocated to the planes outside the BMP. The capacity of a transmission link or storage device will therefore be little affected.

This has been achieved by reserving the S-zone of the BMP, consisting of the 8 rows D8-DF, for the exclusive use of UTF-16. These R-octet values can therefore never occur in an encoding within UCS-2. They are used instead to provide an escape mechanism into the 16 planes G=00, P=01-10. Amendment 1, which specifies UTF-16, also amended the planes of Group 00 which are reserved for private use. It added planes P=0F, 10 to the planes P=E0-FF which were already reserved for this purpose in the first edition of ISO/IEC 10646-1. The effect of this change is to include two private use planes in the 16 additional planes accessible by use of UTF-16.

The UTF-16 coding for a character coded within Planes 00-10 of Group 00 is constructed as follows:

- If P=00 then the coding is in two octets and is as for UCS-2, i.e. the R-octet followed (i.e. with lower significance) by the C-octet;

- If P=01-10 then the coding is in four octets constructed from the UCS-2 coding of two code positions from the S-zone. The first (most significant) two octets are from the range D800-DBFF, the second (least significant) two octets are from the range DC00-DFFF. The code space P=01-10 is divided into blocks of 400 (hexadecimal value) cells for the purpose of determining the coding. The first two octets determine in which block the code position lies. The second two octets determine the position within the block.

In more detail the correspondences between the UCS code position and the pair of S-zone positions is as follows:

- The first two octets are D800 if P=01, R=00-3F; D801 if P=01, R=40-7F, …, D804 if P=02, R=00-3F, …, DBFF if P=10, R=C0-FF.

- The second two octets run from DC00 to DFFF as the position within the block of cells runs from the first to the last position.

The UTF-16 encoding D800 DC00 and the UCS-4 encoding 00010000 therefore represent the same character.

8.5 UTF-8: UCS Transformation format 8

The aim of UTF-8 is entirely different from that of UTF-16. The transformation format UTF-8 is intended for the transmission of data through communication systems which treat the data stream as a sequence of octets from a coding system conforming to the 8-bit code structure laid down in ISO/IEC 4873. This code structure is specific as to the interpretation of octet values in the range 00-7F but octets in the range 80-FF have a variable interpretation that requires agreement between the communicating parties. A communication channel expecting data to conform to ISO/IEC 4873 may therefore only presume to know the interpretation of octets 00-7F.

In particular the communication system may interpret any octet in the range 00-1F as a control character as specified in ISO/IEC 4873, any octet in the range 20-7E as the ASCII character with this coding, and octet 7F as the DELETE character. To comply with this, UTF-16 encodes BMP code

positions 0000 – 007F inclusive by means of their final octet only. This range of positions includes those reserved for control characters and the DELETE function and it relies on the positioning of the ASCII graphic character set in positions 0020-007E of the BMP.

All other code positions in the UCS are represented in UTF-8 by a sequence of 2, 3, 4, 5 or 6 octets. The first octet of such a sequence is in the range C0-FD. Continuing octets are in the range 80-BF. Octets FE and FF are not used.

There is no concept of most significant and least significant octets in UTF-8 encoding. It is a conversion of UCS characters into an ordered sequence of octets, for transmission or other processing in this form. The terms *first octet* and *continuing octets* refer to the order in which the octets occur in the sequence. This order must be maintained, even in transmission systems which serialize 16-bit words as octets by sending the least significant octet before the most significant octet.

The details of the transformation from UCS-4 coding to UTF-8 coding are complex and are not given here in detail. The transformation is such that a code position within the BMP takes at most 3 octets and a code position in planes P=01-1F of Group 00 takes at most 4 octets. These positions that take a maximum of 4 octets to encode therefore include, and exceed, those that can be encoded within UTF-16.

9 Serial transmission of the UCS

9.1 Octet ordering

For many purposes, character data encoded according to any of the encoding methods of the UCS will need to be transmitted in serial form as a sequence of octets. Each of the encoding methods of the UCS, other than UTF-8, specifies its encodings in terms of octets ranked from a most significant octet to a least significant octet. This corresponds to the representation of the code value as a hexadecimal number of 2, 4, 6, 8 or more hexadecimal digits. This description in terms of orders of significance is entirely separate from the order in which the octets are transmitted down a serial communication channel. UTF-8 coding is distinct in that it is directly a coding of UCS data as an ordered sequence of octets.

It is laid down in 6.3 of ISO/IEC 10646-1 that in any transmission method the sequence order of octets from most to least significant shall be preserved and the most significant and least significant ends of the sequence must be identifiable. Furthermore, when character data is serialized as octets then a more significant octet shall precede a less significant octet. When not serialized as octets, the order of octets is a matter of agreement between sender and recipient. For example, if UCS-4 encoding is transmitted along a 16-bit data path then the most and least significant 16-bit words must be composed respectively of the G and P octets and of the R and C octets but it is a matter of private agreement as to the ordering of the two octets within a 16-bit word. The use of signatures for coding identification, explained below, enables a receiving implementation to determine the octet ordering within a 16-bit word in such cases.

A serial data stream of octets may be transmitted through a data path that is 8 bits wide, but it may also be transmitted in turn as a serial stream of single bits. Such representation of the octet stream as a stream of single bits is outside the scope of ISO/IEC 10646-1, so there is no requirement placed by that standard on the order in which the individual bits of an octet are transmitted. The requirements concerning octet order apply at a higher level of protocol at which the data exists in the form of complete octets.

9.2 Signatures for coding identification

There is one character of the BMP that is not part of any of the blocks into which the BMP is otherwise divided. This is the character

- ZERO WIDTH NO-BREAK SPACE (U-0000FEFF)

It is not normally required for linguistic purposes and is not present in any of the collections associated with particular scripts. It is given a special significance, by a convention laid down in annex F of ISO/IEC 10646-1, that enables a receiving implementation to determine without prior knowledge what octet ordering is being adopted by the originating implementation. The coded form of this character, in each of the coding methods of the UCS, is known as the *signature* of that coding method.

Under this convention, an originating implementation sends the character ZERO WIDTH NO-BREAK SPACE at the beginning of a stream of characters. A receiving implementation that is unaware of the convention may simply ignore this character as it has no semantic meaning when sent as an initial character. But an implementation aware of the convention may use the received coded form to determine the ordering of octets being used by the sender.

The coded forms that may be received are:

- UCS-2 signature: FEFF
- UCS-4 signature: 0000 FEFF
- UTF-16 signature: FEFF
- UTF-8 signature: EF BB BF

The UTF-8 coding method is a special case, in that this method in itself specifies an octet ordering. The octets of the UTF-8 signature should therefore never be received in any other order. For the other three coded forms, if the data received is interpreted as containing the 16-bit word FFFE then it indicates that the order of octets within the word should be reversed. Recall that FFFE is prohibited from allocation to a character within any plane of the UCS, specifically to allow this method of signatures to operate without any possible ambiguity. If the word value FEFF (or FFFE) is preceded by the word value 0000 then it also serves to indicate that UCS-4 coding is being used. It is not possible to distinguish between UCS-2 and UTF-16 coding by signature alone, as they give identical encodings for every character of the BMP.

10 Use of control functions with the UCS

10.1 The coding of control functions in 7-bit and 8-bit codes

The general structure of 7-bit and 8-bit codes for character sets is given in:

- ISO/IEC 2022 Information technology – Character code structure and extension techniques.

In particular, this standard provides the overall rules governing the use of control functions with such character sets. When control functions are used in conjunction with the UCS, their coded representation is algorithmically derived from their ISO/IEC 2022 representations. In the first edition of ISO/IEC 10646-1 this algorithm was based exclusively on the ISO/IEC 2022 representation for a 7-bit code, but Amendment 3 changed this to permit also coded representation of control functions based on their 8-bit code representation.

The following definitions are taken from ISO/IEC 2022:

> **control character:** A control function the coded representation of which consists of a single bit combination.

control function: An action that affects the recording, processing, transmission or interpretation of data, and that has a coded representation consisting of one or more bit combinations.

In this context a bit combination is either a 7-bit or an 8-bit byte. In a 7-bit code the hexadecimal values 00-1F are reserved for control characters; this is known as the CL area of the code table. In an 8-bit code there are two such reserved areas, the CL area comprising values 00-1F and the CR area comprising 80-9F.

The coding of any control function either consists of a single control character from either the CL or CR areas, or such a character followed by one or more bit combinations. These following bit combinations are unrestricted in number and value. In particular, they may be (and usually are) values that on their own would be coded representations of graphic characters. The syntax for the use of any control character must provide a way of determining the end of the coding of each control function coded with its use.

ISO/IEC 2022 specifies the assignment of only one control character:

- ESCAPE (acronym ESC) is required to be coded at 1B in the CL area.

The semantics specified for this control character are simple – it causes the meaning of a limited number of bytes following it to be changed. A sequence consisting of the ESCAPE character and such following bytes is known as an *escape sequence.*

ISO/IEC 2022 requires an escape sequence to have the following structure:

- The first character is the ESCAPE character represented by byte value 1B;
- the last character is a Final Byte with value in the range 30-7E;
- between the first and last character there may be zero, one or more Intermediate Bytes, each with a value in the range 20-2F.

This structure enables the escape sequence to be delimited without knowledge of the semantics of the control functions represented by such sequences. ISO/IEC 2022 also specifies a more detailed structure for the use of escape sequences. These uses are all for code extension purposes, such as changing the sets of control characters or graphic characters that are in use, but that standard does not specify the semantics of individual escape sequences. These are subject to registration in accordance with:

- ISO 2375 Data processing – Procedure for registration of escape sequences.

The register set up in accordance with ISO 2375 is:

- *International Register of Coded Character Sets to be used with Escape Sequences*.

10.2 C0 and C1 sets of control characters

The ISO 2375 register includes escape sequences that invoke individual control functions, but it also includes escape sequences that invoke sets of control characters. Each such set is either a C0-set or a C1-set. One set of each type may be invoked simultaneously in either a 7-bit or an 8-bit code. In both cases a C0-set is mapped into the CL area of the code table. A C1-set may be mapped into the CR area of an 8-bit code, but an alternative coding is necessary in a 7-bit code and this alternative is also available for use with 8-bit codes.

This alternative coding of a C1-set is by means of escape sequences. Control characters that would be coded as values in the CR area are represented by a two-byte escape sequence consisting of ESCAPE

followed by a Final Byte in the range 40-5F. These values 40-5F correspond sequentially to the values 80-9F of the CR area. This coding is known as the *ESC Fe* representation of the control character. This term is used in the first edition of ISO/IEC 10646-1 in its explanation of the coding of control functions, but the permitted coding of control functions was extended by Amendment 3 and references to ESC Fe sequences have been deleted.

Control functions for purposes other than code extension are specified in

- ISO/IEC 6429 Information technology – Control functions for coded character sets.

This standard provides a repertoire of control functions from which particular functions may be selected to meet particular needs. Conformance to ISO/IEC 6429 does not require any particular control functions to be supported. It does, however, require that when any of its control functions are used then they shall be encoded as specified in that standard, and that encodings so specified shall not be used for any other purpose.

A particular type of coding specified in ISO/IEC 6429 is a *control sequence*. Control sequences are similar in nature to the escape sequences of ISO/IEC 2022 but they are less restricted in their use. In particular they permit the coding of control functions that take one or more numeric values as parameters. Control sequences start with the C1 control character

- CONTROL SEQUENCE INTRODUCER (acronym CSI)

which may be coded either as 9B in the CR area or as the ESC Fe escape sequence ESC 5B. This character occupies the same relative location in a C1 set as does the ESCAPE character (1B) in a C0 set. A control sequence has the following structure:

- It starts with the CSI function in either of its two coded representations;
- the last character is a Final Byte with value in the range 40-7E;
- between the first and last character there may be zero, one or more Parameter Bytes each with a value in the range 30-3F, followed by zero, one or more Intermediate Bytes, each with a value in the range 20-2F.

The Parameter Bytes used to represent a sequence of numeric parameters consist of the decimal representation of those numbers, with digits 0-9 coded by hexadecimal values 30-39, any separator symbol within a parameter (e.g. decimal point) coded as 3A and with distinct parameters separated from one another by value 3B.

The Final Byte value 6D is reserved in ISO/IEC 6429 for use by ISO/IEC 10646-1, in which it is used to encode the control function IDENTIFY UNIVERSAL CHARACTER SUBSET.

10.3 The use of control functions with the UCS

The coded representation of a control function for use with the UCS is obtained very simply from its coded representation for use with an 8-bit code. For an 8-bit code the coded representation consists of a sequence of one or more 8-bit bytes, of which the first is in one of the ranges 00-1F or 80-9F. Each of these bytes is converted to a UCS code position by taking the byte value as the C-octet value and setting the G-octet, P-octet and R-octet all to zero. Each UCS code position is then encoded according to the adopted coding method of the UCS, namely one of UCS-2, UCS-4, UTF-8 and UTF-16. Since code positions 0000-001F and 0080-009F of the BMP are reserved for the use of control characters, this gives an unambiguous coded representation of any control function without conflicting with the coding of graphic characters in the UCS.

In the first edition of ISO/IEC 10646-1 there was an additional requirement that control characters from a C1-set should be represented as ESC Fe escape sequences before this algorithm is applied.

This requirement was removed by Amendment 3. As an example the control function CONTROL SEQUENCE INTRODUCER, which has coded representation 9B in the CR area of an 8-bit code, would have coded representations in a UCS-2 coding of

- 001B 005B prior to Amendment 3;
- either 001B 005B or more simply 009B after Amendment 3.

10.4 Identification of UCS subsets by use of control functions

One particular control function is defined within ISO/IEC 10646-1 for identifying selected subsets of the UCS. This is coded by means of an ISO/IEC 6429 control sequence, using a Final Byte value reserved in that standard for the use of ISO/IEC 10646-1. The control function is

- IDENTIFY UNIVERSAL CHARACTER SUBSET (acronym IUCS)

It takes as parameters the collection numbers of the collections that comprise the selected subset. The ISO/IEC 6429 coded representation of this control function consists of :

- the control function CONTROL SEQUENCE INTRODUCER (CSI)
- followed by parameter bytes that encode the sequence of collection numbers
- followed by the Intermediate Byte 20
- followed by the (reserved) Final Byte 6E.

This coded representation is then converted to the form appropriate to the UCS coding method in use, following the transformation rules explained above.

As an example,

- the subset comprising the BASIC LATIN and LATIN-1 SUPPLEMENT collections
- which are collections numbered 1 and 2
- may be identified by the control function IUCS 1, 2
- which has ISO/IEC 6429 coded representation CSI 31 3B 32 20 6E
- and which has UCS-2 encoding 001B 005B 0031 003B 0032 0020 006E.

This UCS-2 coding uses the ESC Fe coding form for CSI; following Amendment 3 to ISO/IEC 10646-1 the code values 001B 005B can be replaced by the single code value 009B.

10.5 Invocation of the UCS from an 8-bit code

The character code structure and extension techniques specified in ISO/IEC 2022 includes provision for control functions that

- designate and invoke an identified coding system different from that of ISO/IEC 2022, and
- return to the coding system of ISO/IEC 2022 following such an invocation.

The coded representation of such a control function is by means of an escape sequence in which the first Intermediate Byte has value 25. If there is a second Intermediate Byte with value 2F, it signifies that the new coding system either has no means of return to ISO/IEC 2022 coding or that it provides

its own means for so doing. If the second Intermediate Byte is either absent or has any other value then it signifies that the new coding system uses the escape sequence

- ESC 25 40

to return to the coding system of ISO/IEC 2022 and that on such return, the state of the coding system (which sets of control and graphic characters are invoked, etc.) is restored to the state at the time that the other coding system was invoked. This is known as the *standard return*.

When a new coding system is designated and invoked by means of an escape sequence that signifies that the invocation is without standard return, then any return to the coding system of ISO/IEC 2022 is (by 15.4.2 of ISO/IEC 2022) a return to that coding system in an unspecified state. No announcements, designations and invocations of sets of control and graphic characters, or of other features of ISO/IEC 2022, survive from the state in which the new coding system was invoked. This is true even if the new coding system in fact specifies that return shall be by means of the escape sequence ESC 25 40; it is the escape sequence used to *invoke* the new coding method, not that used to *return* from it, that determines the state of the ISO/IEC 2022 coding system upon return. This has particular significance for the UCS, for reasons that will be seen below.

The assignment of particular escape sequences of these forms to particular coding methods is subject to registration in accordance with ISO 2375. The escape sequences so allocated are then published in the *International Register of Coded Character Sets to be used with Escape Sequences*. The following escape sequences have been registered to designate and invoke the coding methods of the UCS. They are given along with their registration numbers:

- ISO-IR 162 allocates ESC 25 2F 40 to designate and invoke UCS-2 coding at implementation level 1 without standard return;
- ISO-IR 163 allocates ESC 25 2F 41 to designate and invoke UCS-4 coding at implementation level 1 without standard return;
- ISO-IR 174 allocates ESC 25 2F 43 to designate and invoke UCS-2 coding at implementation level 2 without standard return;
- ISO-IR 175 allocates ESC 25 2F 44 to designate and invoke UCS-4 coding at implementation level 2 without standard return;
- ISO-IR 176 allocates ESC 25 2F 45 to designate and invoke UCS-2 coding at implementation level 3 without standard return;
- ISO-IR 177 allocates ESC 25 2F 46 to designate and invoke UCS-4 coding at implementation level 3 without standard return;
- ISO-IR 178 allocates ESC 25 42 to designate and invoke UTF-1 coding (now withdrawn from ISO/IEC 10646-1 by Amendment 4) with standard return – the register entry includes the complete specification of UTF-1;
- ISO-IR 190 allocates ESC 25 2F 47 to designate and invoke UTF-8 coding at implementation level 1 without standard return;
- ISO-IR 191 allocates ESC 25 2F 48 to designate and invoke UTF-8 coding at implementation level 2 without standard return;
- ISO-IR 192 allocates ESC 25 2F 49 to designate and invoke UTF-8 coding at implementation level 3 without standard return;

- ISO-IR 193 allocates ESC 25 2F 4A to designate and invoke UTF-16 coding at implementation level 1 without standard return;

- ISO-IR 194 allocates ESC 25 2F 4B to designate and invoke UTF-16 coding at implementation level 2 without standard return;

- ISO-IR 195 allocates ESC 25 2F 4C to designate and invoke UTF-16 coding at implementation level 3 without standard return;

- ISO-IR 196 allocates ESC 25 47 to designate and invoke UTF-8 coding at an unspecified implementation level but with standard return.

Note that of the coding methods defined for the UCS, only UTF-8 and (the now withdrawn) UTF-1 have the ability to make use of the standard return, since they are the only coding methods in which the escape sequence for the standard return is not transformed when used with the coding method concerned.

ISO/IEC 10646-1 does specify a means to return to the ISO/IEC 2022 coding system when it has been invoked without standard return. The specified method is by means of the escape sequence ESC 25 40 *transformed to the appropriate coding method of the UCS* as described above, e.g. 001B 0025 0040 designates a return from UCS-2. This is the same escape sequence as that used for a standard return, but except when used with UTF-8 its coded representation will include additional padding octets with value zero. Even when used with UTF-8, when that coding method has been invoked by means of ISO-IR 190, 191 or 192, it is interpreted for the purposes of ISO/IEC 2022 as a non-standard return, with consequent loss of the state of that coding system as described above.