

The Adoption of ISO 10646

Outline

- Background
- What is ISO 10646
- Why adoption of ISO 10646
- Issues to be considered when migrating to ISO 10646 enabled platforms

Background

- HK is a bilingual society, English, Chinese, Cantonese dialect,
- systems:
 - English systems(ASCII based)
 - Chinese system: traditional Chinese(Big5 based and simplified Chinese (GB2312 based)
 - Most Chinese systems are Big5 based
 - Also English with add-on language support

Problems with current systems

- Big5 has limited repertoire
- Problem with Electronic interchange with regions using non-Big5-HKSCS system
- Problem with exchange of non-Big5 systems(Chinese)
- Demand for more information on-line and exchangeable(other than English and Chinese)

Alternatives: ISO 10646 enabled systems

What is ISO 10646

- It is a **coded character set(codeset)** -- a set of characters with each character having a unique code(code point)
 - Code points are computers internal representations of text information
 - Used for text processing and exchange
 - also has transformation forms for storage and conversion
- First published in 1993

Features

- Universal: characters in all major languages for modern use
 - superset of all existing major National and Industrial standards
 - also referred to as the Universal Character Set(UCS)
- Uniform and Efficient: fixed-width encoding, no need to identify the coding length(ASCII, Big5, GB),
- Open Framework: Fix the coding architectures, and characters(code points) can be filled up(assigned) later.
- No ambiguity: No need for codeset identification

Coding Structure(UCS-2)

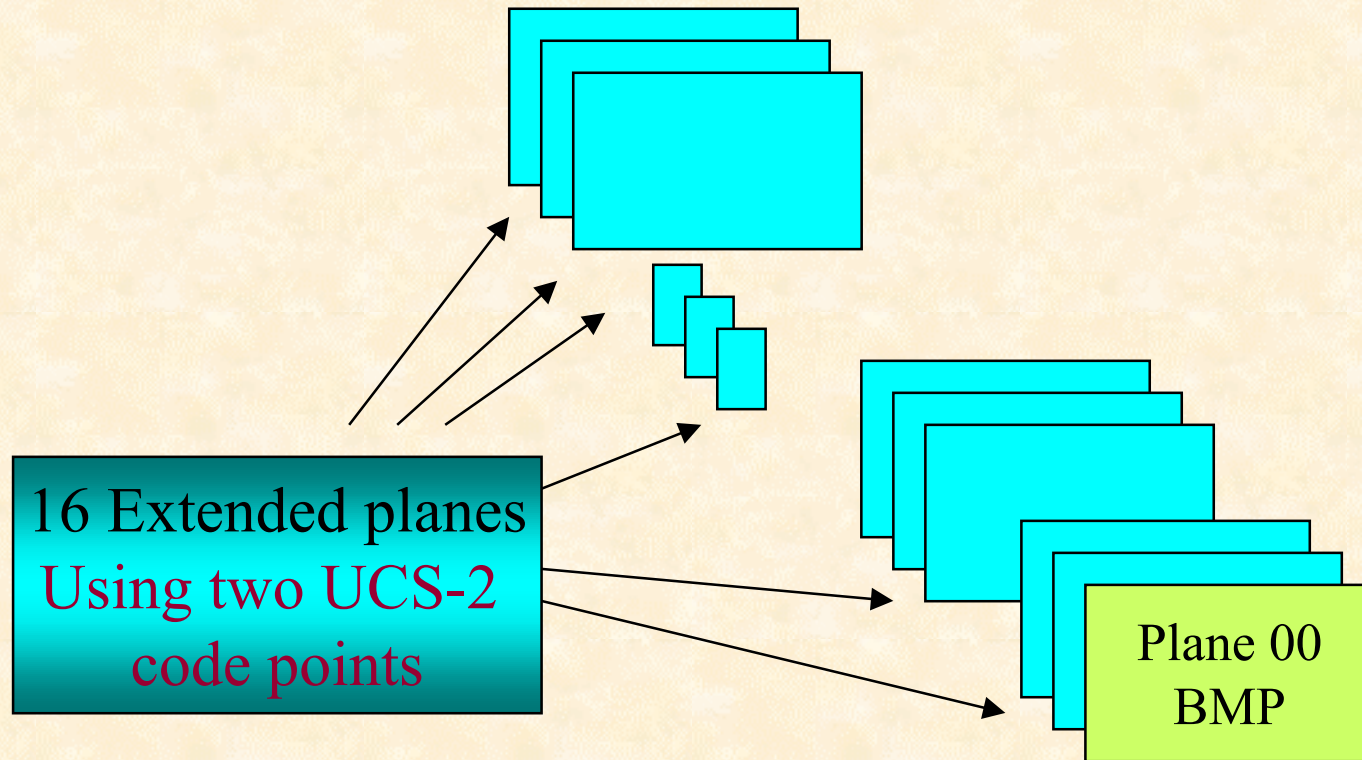
- Two-byte coding assignment

- $00\ 00_{16}$ to $FF\ FF_{16}$

High Byte (total: 256)	Low Byte (total: 256)
---------------------------	--------------------------

- Total code points: $2^{16} = 65,536$
- also referred to as the **Basic Multilingual Plane(BMP)**
- Switching mechanism to use code range of BMP to access another $16 \times 65,536$
- Currently only BMP contains characters

Architecture



Basic Multilingual Plane

A-Zone

Alphabets, Symbols, CJK Misc

I-Zone

CJK ideographs

O-zone

Hangul

S-Zone(Surrogate)

R-Zone

Private Use, Compatibility, Arabic Presentations

Current Allocation

0000-1FFF: General scripts for alphabets: 0-256 are identical to ISO8859-1 (Latin-1) characters

2000-2FFF: punctuation marks and symbols

3000-33FF: CJK phonetics and symbols, and CJK compatibility zone(Kangxi radicals)

4E00-9FFF: for CJK Unified Ideographs(20,992)

A000-D7FF: Korean hangul

D800-DFFF: Surrogate Area

E000 -F8FF: Private use area

F900 - FFFD: Others

FFFE FFFF: are reserved

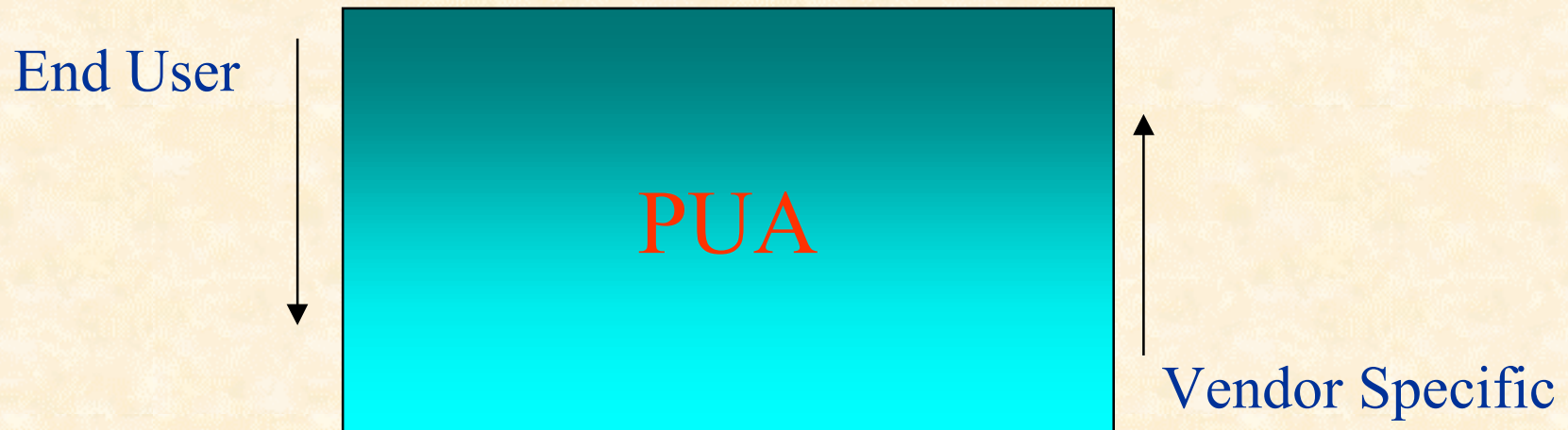
CJK Ideographs

- Han characters from different Asian countries
 - CJK: 20,902 chars(out of 20,992 in the zone)
- Extension A: 6,558 chars(out of 6,656)
 - Available in the next version in BMP
- Extension B: over 40,000
 - Still at editing stage
 - Will be outside of BMP
- Note: Some of the ideographic characters are not used in Chinese
- Example: Korean character

The image shows a single Korean character, '碁' (Go), which is a Hanja used in the Korean word 'Go' (碁). It is presented in a white square box.

Private Use Area(PUA)

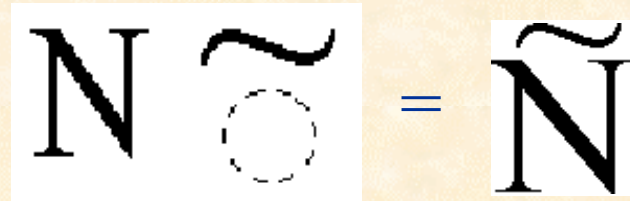
- E000 - F8FF: 6,400 code positions
- Two sub areas(conceptually): End User Sub-area and Corporate Use Sub-area
- Grow in different directions, no clear boundary



Unicode

Implementation of UCS-2

- Use UCS-2 as internal code
- Recognise UCS-2 as character data type
- Defines actions associated with certain characters
 - control characters
 - combining characters



Difficulties

The large set of characters is a good idea for exchange, but headache for system support

- Supported components
 - Output: large font set
 - Input: Many different input methods and possibly for different languages
- Take time to develop ISO 10646 enabled applications
- Compatibility with current software
- Compatibility with existing electronic data

Misconception

- Supporting the framework vs supporting every character
 - having the framework makes it possible to support, say Russian if needed, but we do not need to support Russian script related applications
- Support of characters vs understanding/use of a particular language
- Applications use codeset as representation, specific applications are still language dependent
- Input method is both codeset and specific language dependent: importance of sub-division in ISO 10646

Examples

- Reading both simplified Chinese and traditional Chinese
 - font set is enough
- Composing Chinese in Hong Kong
 - font set as well as input methods
- Teaching Putonghua software
 - input pinyin symbols
 - find the correct Chinese characters
 - output the correct pronunciation

Migration to ISO 10646

- Software developers:
 - System vendors
 - Input and output vendors
 - Software developer
 - Migration software tools: conversion utilities
 - making codesets as transparent to users as possible
- End Users
 - Awareness of ISO 10646
 - Select ISO 10646/Unicode enabled software
 - Look for bundled conversion utilities
 - Planning and archiving