File created: 26-Aug-2024 16:58:36 **{WMEDLEY}<library>UNICODE.;74**

edit by: rmk

changes to: (FNS UNICODE-EXTEND-TRANSLATION?)

previous date: 27-Mar-2024 23:07:42 {WMEDLEY}<library>UNICODE.;73

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
(RPAQQ UNICODECOMS
       ((COMS                                                    ; External formats
              (FNS UTF8.OUTCHARFN UTF8.INCCODEFN UTF8.PEEKCCODEFN \UTF8.BACKCCODEFN)
              (FNS UTF16BE.OUTCHARFN UTF16BE.INCCODEFN UTF16BE.PEEKCCODEFN \UTF16BE.BACKCCODEFN)
              (FNS UTF16LE.OUTCHARFN UTF16LE.INCCODEFN UTF16LE.PEEKCCODEFN \UTF16LE.BACKCCODEFN)
              (FNS READBOM WRITEBOM)
              (INITVARS (EXTERNALEOL 'LF))
              (FNS MAKE-UNICODE-FORMATS)
              (P (MAKE-UNICODE-FORMATS EXTERNALEOL))
              (ADDVARS (*DEFAULT-EXTERNALFORMATS* (UNIX :UTF-8)))
              (FNS UNICODE.UNMAPPED UNICODE-EXTEND-TRANSLATION? UTF8.BINCODE \UTF8.FETCHCODE)
              (FNS UTF8.VALIDATE UTF8-SIZE-FROM-BYTE1 NUTF8-BYTE1-BYTES NUTF8-CODE-BYTES NUTF8-STRING-BYTES)
              (DECLARE%: EVAL@COMPILE DONTCOPY (MACROS UNICODE.TRANSLATE \UTF8.GETBASEBYTE))
              (FNS XTOUCODE UTOXCODE))
        ;;
        (COMS                                                    ; Read Unicode mapping files
              (INITVARS (UNICODEDIRECTORIES NIL))
              (VARS XCCS-CHARSETS)
              (FNS READ-UNICODE-MAPPING-FILENAMES READ-UNICODE-MAPPING))
        [COMS                                                    ; Make translation tables for  UTF external formats
              (FNS MAKE-UNICODE-TRANSLATION-TABLES MERGE-UNICODE-TRANSLATION-TABLES
                   MERGE-UNICODE-TRANSLATION-TABLES1)
              (FNS INVERT-ALL-UNICODE-MAPPINGS ALL-UNICODE-MAPPINGS)
              (INITVARS (*XCCSTOUNICODE*)
                     (*UNICODETOXCCS*)
                     (*INVERTED-UNICODE-MAPPINGS*))
              (GLOBALVARS *XCCSTOUNICODE* *UNICODETOXCCS*)
              [DECLARE%: DONTEVAL@LOAD DOCOPY (P (MAKE-UNICODE-TRANSLATION-TABLES 'DEFAULT]
              (DECLARE%: EVAL@COMPILE DONTCOPY
                     ;; These control the layout of the translation tables.  Since many of the upper panels are sparse, doing it per-panel (128)
                     ;; seems more space-efficient, and residual alists can be shorter
                     (CONSTANTS (TRANSLATION-SEGMENT-SIZE 128)
                            (MAX-ALIST-LENGTH 10)
                            (N-TRANSLATION-SEGMENTS (IQUOTIENT 65536 TRANSLATION-SEGMENT-SIZE))
                            (TRANSLATION-SHIFT (INTEGERLENGTH (SUB1 TRANSLATION-SEGMENT-SIZE)))
                            (TRANSLATION-MASK (SUB1 TRANSLATION-SEGMENT-SIZE]
        ;;
        (COMS                                                    ; Write Unicode mapping files
              (FNS WRITE-UNICODE-MAPPING WRITE-UNICODE-INCLUDED WRITE-UNICODE-MAPPING-HEADER
                   WRITE-UNICODE-MAPPING-FILENAME HEXSTRING)
              (FNS XCCS-UTF8-AFTER-OPEN)
              ;; Automate dumping of a documentation prefix
              [DECLARE%: EVAL@COMPILE DONTCOPY (CONSTANTS (MISSINGCODE (CL:PARSE-INTEGER "FFFE" :RADIX 16))
                                                          (UNDEFINEDCODE (CL:PARSE-INTEGER "FFFF" :RADIX 16]
              (VARS UNICODE-MAPPING-HEADER))
        (FNS UTF8HEXSTRING XTOUSTRING XCCSSTRING)
        (FNS UNHEXSTRING)
        (FNS SHOWCHARS)
        (DECLARE%: EVAL@COMPILE DONTCOPY (FILES (FROM LOADUPS)
                                                EXPORTS.ALL))
        (PROP (FILETYPE)
              UNICODE)))

;; External formats

(DEFINEQ

(UTF8.OUTCHARFN
  [LAMBDA (STREAM CHARCODE RAW)                                  ; Edited 31-Jan-2024 00:32 by rmk
                                                                 ; Edited  8-Aug-2021 13:02 by rmk:
                                                                 ; Edited 17-Aug-2020 08:45 by rmk:
                                                                 ; Edited 30-Jan-2020 23:08 by rmk:
    ;; Perhaps the translation table should already do the mapping for EOL to LF, but that seems to be a separate property of the stream. Also,
    ;; CRLF=2 bytes.

    ;; Print UTF8 sequence for CHARCODE.  Do not do XCCS to Unicode translation if RAW.

    (IF (EQ CHARCODE (CHARCODE EOL))
        THEN (FREPLACE (STREAM CHARPOSITION) OF STREAM WITH 0)
             (\BOUTEOL STREAM)
```

```
      ELSE (CHANGE (FFETCH (STREAM CHARPOSITION) OF STREAM)
                   (IPLUS16 1 DATUM))
           (FOR C INSIDE (CL:IF RAW
                                CHARCODE
                                (UNICODE.TRANSLATE CHARCODE *XCCSTOUNICODE*))
              DO (IF (ILESSP C 128)
                     THEN (\BOUT STREAM C)
                   ELSEIF (ILESSP C 2048)
                     THEN                                          ; x800
                          (\BOUT STREAM (LOGOR (LLSH 3 6)
                                               (LRSH C 6)))
                          (\BOUT STREAM (LOGOR (LLSH 2 6)
                                               (LOADBYTE C 0 6)))
                   ELSEIF (ILESSP C 65536)
                     THEN                                          ; x10000
                          (\BOUT STREAM (LOGOR (LLSH 7 5)
                                               (LRSH C 12)))
                          (\BOUT STREAM (LOGOR (LLSH 2 6)
                                               (LOADBYTE C 6 6)))
                          (\BOUT STREAM (LOGOR (LLSH 2 6)
                                               (LOADBYTE C 0 6)))
                   ELSEIF (ILESSP C 2097152)
                     THEN                                          ; x200000
                          (\BOUT STREAM (LOGOR (LLSH 15 4)
                                               (LRSH C 18)))
                          (\BOUT STREAM (LOGOR (LLSH 2 6)
                                               (LOADBYTE C 12 6)))
                          (\BOUT STREAM (LOGOR (LLSH 2 6)
                                               (LOADBYTE C 6 6)))
                          (\BOUT STREAM (LOGOR (LLSH 2 6)
                                               (LOADBYTE C 0 6)))
                   ELSE (ERROR "CHARCODE too big for UTF8" C])
```

(UTF8.INCCODEFN

```
  [LAMBDA (STREAM COUNTP RAW)                                      ; Edited  2-Feb-2024 11:44 by rmk
                                                                   ; Edited 30-Jan-2024 22:56 by rmk
                                                                   ; Edited  6-Aug-2021 16:02 by rmk:
                                                                   ; Edited  6-Aug-2020 17:13 by rmk:
```

;; Do not do UNICODE to XCSS translation if RAW.

;; Test for smallp because the stream's End-of-file operation may suppress the error

```
(DECLARE (USEDFREE *BYTECOUNTER*))
(LET (BYTE1 BYTE2 BYTE3 BYTE4 CODE (COUNT 1))
     (SETQ BYTE1 (\BIN STREAM))
```

;; Distinguish on header bytes (modulo peculiar EOF behavior--the caller will get whatever ended up in BYTE1

```
     (CL:WHEN (SMALLP BYTE1)
         [SETQ CODE (if (ILEQ BYTE1 127)
                        then ;; Test first:  Ascii is the common case.  EOL requires its own translation
                             (SELCHARQ BYTE1
                                  (CR (SELECTC (fetch (STREAM EOLCONVENTION) of STREAM)
                                               (CR.EOLC          ; Also eq BYTE1
                                                        (CHARCODE EOL))
                                               (CRLF.EOLC (if (EQ (CHARCODE LF)
                                                                  (\PEEKBIN STREAM T))
                                                              then (\BIN STREAM)
                                                                   (CL:WHEN COUNTP (SETQ COUNT 2))
                                                                   (CHARCODE EOL)
                                                            else BYTE1))
                                                      BYTE1))
                                  (LF (CL:IF (EQ LF.EOLC (fetch (STREAM EOLCONVENTION) of STREAM))
                                             (CHARCODE EOL)
                                             BYTE1))
                                  BYTE1)
                      elseif (ILEQ BYTE1 223)
                        then                                       ; 2 bytes
                             (SETQ COUNT 2)
                             (SETQ BYTE2 (\BIN STREAM))
                             (CL:WHEN (OR (NOT (SMALLP BYTE2))
                                          (ILESSP BYTE2 128))
                                 (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2)))
                             (LOGOR (LLSH (LOADBYTE BYTE1 0 5)
                                          6)
                                    (LOADBYTE BYTE2 0 6))
                      elseif (ILEQ BYTE1 239)
                        then                                       ; 3 bytes
                             (SETQ BYTE2 (\BIN STREAM))
                             (CL:WHEN (OR (NOT (SMALLP BYTE2))
                                          (ILESSP BYTE2 128))
                                 (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2)))
                             (SETQ BYTE3 (\BIN STREAM))
                             (CL:WHEN (OR (NOT (SMALLP BYTE3))
                                          (ILESSP BYTE3 128))
                                 (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2 BYTE3)))
                             (SETQ COUNT 3)
```

```
                              (LOGOR (LLSH (LOADBYTE BYTE1 0 4)
                                           12)
                                     (LLSH (LOADBYTE BYTE2 0 6)
                                           6)
                                     (LOADBYTE BYTE3 0 6))
                        else                                          ; 4 bytes
                             (SETQ BYTE2 (\BIN STREAM))
                             (CL:WHEN (OR (NOT (SMALLP BYTE2))
                                          (ILESSP BYTE2 128))
                                 (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2)))
                             (SETQ BYTE3 (\BIN STREAM))
                             (CL:WHEN (OR (NOT (SMALLP BYTE3))
                                          (ILESSP BYTE3 128))
                                 (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2 BYTE3)))
                             (SETQ BYTE4 (\BIN STREAM))
                             (CL:WHEN (OR (NOT (SMALLP BYTE4))
                                          (ILESSP BYTE4 128))
                                 (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2 BYTE3 BYTE4)))
                             (SETQ COUNT 4)
                             (LOGOR (LLSH (LOADBYTE BYTE1 0 3)
                                          18)
                                    (LLSH (LOADBYTE BYTE2 0 6)
                                          12)
                                    (LLSH (LOADBYTE BYTE3 0 6)
                                          6)
                                    (LOADBYTE BYTE4 0 6])
        (CL:UNLESS (OR RAW (NOT (SMALLP CODE)))
            (SETQ CODE (UNICODE.TRANSLATE CODE *UNICODETOXCCS*)))
        (CL:WHEN COUNTP (SETQ *BYTECOUNTER* COUNT))
        CODE])
```

(**UTF8.PEEKCCODEFN**

```
  [LAMBDA (STREAM NOERROR RAW)                                        ; Edited  2-Feb-2024 11:48 by rmk
                                                                      ; Edited 14-Jun-2021 22:53 by rmk:
```

;; Modeled this after \EUCPEEK on LLREAD. In the multi-byte (non-ASCII) case, backs the file pointer to the beginning by the proper number of ;; \BACKFILEPTRs, and returns a count of 0. Returns NIL if NOERROR and either invalid UTF8 or end of file.

;; Could be that the caller takes care of backing up the file position if the number of binned-bytes is returned.

;; Do not do UNICODE to XCCS translation if RAW

```
(PROG (BYTE1 BYTE2 BYTE3 BYTE4 CODE)
      (SETQ BYTE1 (\PEEKBIN STREAM NOERROR))
```

;; Distinguish on header bytex

```
      (CL:UNLESS BYTE1 (RETURN NIL))
      [if (ILEQ BYTE1 127)
          then ;; Test first:  Ascii is the common case.  No need to back up, since we peeked.
               (SETQ CODE BYTE1)
        elseif [ILEQ BYTE1 223                                        ; 2 bytes
                     (BIN STREAM)
                     (SETQ BYTE2 (\PEEKBIN STREAM NOERROR))
                     (\BACKFILEPTR STREAM)
                     (if (AND BYTE2 (IGEQ BYTE2 128))
                         then (SETQ CODE (LOGOR (LLSH (LOADBYTE BYTE1 0 5)
                                                      6)
                                                (LOADBYTE BYTE2 0 6)))
                       elseif NOERROR
                       else (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2]
        elseif (ILEQ BYTE1 239)
          then                                                        ; 3 bytes
               (BIN STREAM)
               (CL:UNLESS (AND (SETQ BYTE2 (\PEEKBIN STREAM NOERROR))
                               (IGEQ BYTE2 128))
                   (\BACKFILEPTR STREAM)
                   (OR NOERROR (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2)))
                   (RETURN CODE))
               (BIN STREAM)
               (SETQ BYTE3 (\PEEKBIN STREAM NOERROR))                 ; PEEK the last, no need to back it up
               (\BACKFILEPTR STREAM)
               (\BACKFILEPTR STREAM)
               (if (AND BYTE3 (IGEQ BYTE3 128))
                   then (SETQ CODE (LOGOR (LLSH (LOADBYTE BYTE1 0 4)
                                                12)
                                          (LLSH (LOADBYTE BYTE2 0 6)
                                                6)
                                          (LOADBYTE BYTE3 0 6)))
                 elseif NOERROR
                 else (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2 BYTE3)))
        else                                                          ; 4 bytes
             (BIN STREAM)
             (CL:UNLESS (AND (SETQ BYTE2 (\PEEKBIN STREAM NOERROR))
                             (IGEQ BYTE2 128))
                 (\BACKFILEPTR STREAM)
                 (OR NOERROR (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2)))
                 (RETURN CODE))
```

```
                (BIN STREAM)
                (CL:UNLESS (AND (SETQ BYTE3 (\PEEKBIN STREAM NOERROR))
                                (IGEQ BYTE3 128))
                    (\BACKFILEPTR STREAM)
                    (\BACKFILEPTR STREAM)
                    (OR NOERROR (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2 BYTE3)))
                    (RETURN CODE))
                (BIN STREAM)
                (SETQ BYTE4 (\PEEKBIN STREAM NOERROR))
                (\BACKFILEPTR STREAM)
                (\BACKFILEPTR STREAM)
                (\BACKFILEPTR STREAM)
                (if (AND BYTE4 (IGEQ BYTE4 128))
                    then (SETQ CODE (LOGOR (LLSH (LOADBYTE BYTE1 0 3)
                                                 18)
                                           (LLSH (LOADBYTE BYTE2 0 6)
                                                 12)
                                           (LLSH (LOADBYTE BYTE3 0 6)
                                                 6)
                                           (LOADBYTE BYTE4 0 6)))
                  elseif NOERROR
                  else (ERROR "INVALID UTF8 SEQUENCE" (LIST BYTE1 BYTE2 BYTE3 BYTE4]
          (CL:WHEN (AND CODE (NOT RAW))
              (SETQ CODE (UNICODE.TRANSLATE CODE *UNICODETOXCCS*)))
          (RETURN CODE])
```

(\UTF8.BACKCCODEFN

```
  [LAMBDA (STREAM COUNTP RAW)                                   ; Edited 19-Jul-2022 15:30 by rmk
                                                                ; Edited  6-Aug-2021 16:04 by rmk:
```

;; \BACKFILEPTR is NIL at beginning of FILE. Presumably a little bit more efficient if we decoded the UTF8 bytes backwards and didn't do the ;; peek, but probably not worth the complexity.

```
    (DECLARE (USEDFREE *BYTECOUNTER*))
    (BIND (C _ 0) WHILE (IF (\BACKFILEPTR STREAM)
                            THEN (ADD C -1)
                                 (EQ 2 (LRSH (\PEEKBIN STREAM)
                                             6))
                            ELSE (CL:WHEN COUNTP (SETQ *BYTECOUNTER* C))
                                 (RETURN NIL))
       REPEATUNTIL (EQ C -4) FINALLY (CL:WHEN COUNTP (SETQ *BYTECOUNTER* C))
                                     (RETURN (UTF8.PEEKCCODEFN STREAM NIL RAW])

)

(DEFINEQ
```

(UTF16BE.OUTCHARFN

```
  [LAMBDA (STREAM CHARCODE RAW)                                 ; Edited 31-Jan-2024 00:32 by rmk
                                                                ; Edited  8-Aug-2021 13:09 by rmk:
                                                                ; Edited 30-Jan-2020 23:08 by rmk:
```

;; PRINT UTF16 sequence for CHARCODE. Do not do XCCS to UNICODE translation if RAW.

;; Not sure about EOL conversion if truly "raw"

```
    (IF (EQ CHARCODE (CHARCODE EOL))
        THEN (FREPLACE (STREAM CHARPOSITION) OF STREAM WITH 0)
      ELSE (CHANGE (FFETCH (STREAM CHARPOSITION) OF STREAM)
                   (IPLUS16 1 DATUM)))
    (FOR C INSIDE (CL:IF RAW
                         CHARCODE
                         (UNICODE.TRANSLATE CHARCODE *XCCSTOUNICODE*))
       DO (\WOUT STREAM C])
```

(UTF16BE.INCCODEFN

```
  [LAMBDA (STREAM COUNTP RAW)                                   ; Edited 10-Mar-2024 12:00 by rmk
                                                                ; Edited  6-Aug-2021 16:05 by rmk:
```

;; Do not do UNICODE to XCCS translation if RAW. Test for SMALLPin case of funky EOF behavior

```
    (DECLARE (USEDFREE *BYTECOUNTER*))
    (LET (CODE BYTE1 BYTE2 COUNT)
         (IF [AND (SMALLP (SETQ BYTE1 (\BIN STREAM)))
                  (SMALLP (SETQ BYTE2 (\BIN STREAM]
             THEN (SETQ COUNT 2)
                  (SETQ CODE (create WORD
                                     HIBYTE _ (\BIN STREAM)
                                     LOBYTE _ (\BIN STREAM)))
                  (CL:UNLESS RAW
                      (SETQ CODE (UNICODE.TRANSLATE CODE *UNICODETOXCCS*)))
                  (CL:WHEN COUNTP (SETQ *BYTECOUNTER* COUNT))
                  CODE
           ELSE (ERROR "ODD NUMBER OF BYTES IN UTF16 FILE" STREAM])
```

(UTF16BE.PEEKCCODEFN

```
  [LAMBDA (STREAM NOERROR RAW)                                  ; Edited 10-Mar-2024 12:01 by rmk
                                                                ; Edited 14-Jun-2021 22:58 by rmk:
```

```
    ;; Could be that the caller takes care of backing up the file position if the number of binned-bytes is returned.
    ;; Do not do UNICODE to XCCS translation if RAW
    (LET (BYTE1 BYTE2 CODE)
         (SETQ BYTE1 (\PEEKBIN STREAM NOERROR))
         (IF BYTE1
             THEN (\BIN STREAM)
                  (SETQ BYTE2 (\PEEKBIN STREAM NOERROR))
                  (\BACKFILEPTR STREAM)
                  (IF BYTE2
                      THEN (SETQ CODE (create WORD
                                              HIBYTE _ BYTE1
                                              LOBYTE _ BYTE2))
                           (CL:IF RAW
                               CODE
                               (UNICODE.TRANSLATE CODE *UNICODETOXCCS*))
                    ELSEIF NOERROR
                      THEN NIL)
           ELSEIF NOERROR
             THEN NIL
           ELSE (ERROR "INVALID UTF16 CHARACTER" (LIST BYTE1 BYTE2])
```

(\UTF16BE.BACKCCODEFN

```
  [LAMBDA (STREAM COUNTP RAW)                                    ; Edited 10-Mar-2024 12:02 by rmk
                                                                 ; Edited 19-Jul-2022 15:14 by rmk
                                                                 ; Edited  6-Aug-2021 16:07 by rmk:
    ;; \BACKFILEPTR is NIL at beginning of FILE, do nothing.
    (DECLARE (USEDFREE *BYTECOUNTER*))
    (CL:WHEN (\BACKFILEPTR STREAM)
        (LET (CODE (BYTE2 (\PEEKBIN STREAM)))
             (IF (\BACKFILEPTR STREAM)
                 THEN (CL:WHEN COUNTP (SETQ *BYTECOUNTER* -2))
                      (SETQ CODE (create WORD
                                         HIBYTE _ (\PEEKBIN STREAM)
                                         LOBYTE _ BYTE2))
                      (CL:IF RAW
                          CODE
                          (UNICODE.TRANSLATE CODE *UNICODETOXCCS*))
               ELSEIF COUNTP
                 THEN (SETQ *BYTECOUNTER* -1)
                      NIL)))])

)

(DEFINEQ
```

(UTF16LE.OUTCHARFN

```
  [LAMBDA (STREAM CHARCODE RAW)                                  ; Edited 10-Mar-2024 11:58 by rmk
                                                                 ; Edited  8-Aug-2021 13:09 by rmk:
                                                                 ; Edited 30-Jan-2020 23:08 by rmk:
    ;; PRINT UTF16 sequence for CHARCODE.  Do not do XCCS to UNICODE translation if RAW.
    ;; Not sure about EOL conversion if truly "raw"
    (IF (EQ CHARCODE (CHARCODE EOL))
        THEN (FREPLACE (STREAM CHARPOSITION) OF STREAM WITH 0)
      ELSE (CHANGE (FFETCH (STREAM CHARPOSITION) OF STREAM)
                   (IPLUS16 1 DATUM)))
    (FOR C INSIDE (CL:IF RAW
                      CHARCODE
                      (UNICODE.TRANSLATE CHARCODE *XCCSTOUNICODE*))
       DO (BOUT STREAM (fetch LOBYTE of CHARCODE))
          (BOUT STREAM (fetch HIBYTE of CHARCODE])
```

(UTF16LE.INCCODEFN

```
  [LAMBDA (STREAM COUNTP RAW)                                    ; Edited 10-Mar-2024 12:03 by rmk
                                                                 ; Edited  6-Aug-2021 16:05 by rmk:
    ;; Do not do UNICODE to XCCS translation if RAW.  Test for SMALLPin case of funky EOF behavior
    (DECLARE (USEDFREE *BYTECOUNTER*))
    (LET (CODE BYTE1 BYTE2 COUNT)
         (IF [AND (SMALLP (SETQ BYTE1 (\BIN STREAM)))
                  (SMALLP (SETQ BYTE2 (\BIN STREAM]
             THEN (SETQ COUNT 2)
                  (SETQ CODE (create WORD
                                     LOBYTE _ (\BIN STREAM)
                                     HIBYTE _ (\BIN STREAM)))
                  (CL:UNLESS RAW
                      (SETQ CODE (UNICODE.TRANSLATE CODE *UNICODETOXCCS*)))
                  (CL:WHEN COUNTP (SETQ *BYTECOUNTER* COUNT))
                  CODE
           ELSE (ERROR "ODD NUMBER OF BYTES IN UTF16 FILE" STREAM])
```

(UTF16LE.PEEKCCODEFN

```
  [LAMBDA (STREAM NOERROR RAW)                                    ; Edited 10-Mar-2024 11:43 by rmk
                                                                  ; Edited 14-Jun-2021 22:58 by rmk:
```

;; Could be that the caller takes care of backing up the file position if the number of binned-bytes is returned.

;; Do not do UNICODE to XCCS translation if RAW

```
(LET (BYTE1 BYTE2 CODE)
     (SETQ BYTE1 (\PEEKBIN STREAM NOERROR))
     (IF BYTE1
         THEN (\BIN STREAM)
              (SETQ BYTE2 (\PEEKBIN STREAM NOERROR))
              (\BACKFILEPTR STREAM)
              (IF BYTE2
                  THEN (SETQ CODE (LOGOR (LLSH BYTE2 8)
                                         BYTE1))
                       (CL:IF RAW
                           CODE
                           (UNICODE.TRANSLATE CODE *UNICODETOXCCS*))
                ELSEIF NOERROR
                  THEN NIL)
       ELSEIF NOERROR
         THEN NIL
       ELSE (ERROR "INVALID UTF16 CHARACTER" (LIST BYTE1 BYTE2])
```

(\UTF16LE.BACKCCODEFN

```
  [LAMBDA (STREAM COUNTP RAW)                                     ; Edited 10-Mar-2024 12:04 by rmk
                                                                  ; Edited 19-Jul-2022 15:14 by rmk
                                                                  ; Edited  6-Aug-2021 16:07 by rmk:
```

;; \BACKFILEPTR is NIL at beginning of FILE, do nothing.

```
(DECLARE (USEDFREE *BYTECOUNTER*))
(CL:WHEN (\BACKFILEPTR STREAM)
    (LET (CODE (BYTE2 (\PEEKBIN STREAM)))
         (IF (\BACKFILEPTR STREAM)
             THEN (CL:WHEN COUNTP (SETQ *BYTECOUNTER* -2))
                  (SETQ CODE (create WORD
                                     HIBYTE _ BYTE2
                                     LOBYTE _ (\PEEKBIN STREAM)))
                  (CL:IF RAW
                      CODE
                      (UNICODE.TRANSLATE CODE *UNICODETOXCCS*))
           ELSEIF COUNTP
             THEN (SETQ *BYTECOUNTER* -1)
                  NIL)))])
```

)

(DEFINEQ

(READBOM

```
  [LAMBDA (STREAM COUNTP)                                         ; Edited 11-Mar-2024 23:53 by rmk
                                                                  ; Edited 10-Mar-2024 13:01 by rmk
```

;; If COUNTP, this must be under a generic \INCCODE that binds *BYTECOUNTER*

;; Reads and decodes the BOM bytes. If BOM ispresent, the stream is left at the first following byte, otherwise the stream is reset to its position on ;; entry (presumably 0).

;; I used the UNHEXTRING constants so that the hex bytes are visible in the code, maybe there's another function that does that?

```
(DECLARE (USEDFREE *BYTECOUNTER*))
(SELECTC (\PEEKBIN STREAM T)
     ((UNHEXSTRING "EF")
          (BIN STREAM)
          (if (EQ (CONSTANT (UNHEXSTRING "BB"))
                  (\PEEKBIN STREAM T))
              then (BIN STREAM)
                   (if (EQ (CONSTANT (UNHEXSTRING "BF"))
                           (\PEEKBIN STREAM T))
                       then (BIN STREAM)
                            (CL:WHEN COUNTP (add *BYTECOUNTER* 3))
                            :UTF-8
                     else (\BACKFILEPTR STREAM))
            else (\BACKFILEPTR STREAM)))
     ((UNHEXSTRING "FE")
          (BIN STREAM)
          (if (EQ (CONSTANT (UNHEXSTRING "FF"))
                  (\PEEKBIN STREAM T))
              then (BIN STREAM)
                   (CL:WHEN COUNTP (add *BYTECOUNTER* 2))
                   :UTF-16BE
            else (\BACKFILEPTR STREAM)))
     ((UNHEXSTRING "FF")
          (BIN STREAM)
          (if (EQ (CONSTANT (UNHEXSTRING "FE"))
                  (\PEEKBIN STREAM T))
              then (BIN STREAM)
                   (CL:WHEN COUNTP (add *BYTECOUNTER* 2))
```

```
                    :UTF-16LE
              else (\BACKFILEPTR STREAM)))
        NIL])
```

(WRITEBOM

```
  [LAMBDA (STREAM FORMAT)                                    ; Edited 16-Mar-2024 20:53 by rmk
                                                             ; Edited 11-Mar-2024 23:53 by rmk
                                                             ; Edited 10-Mar-2024 13:01 by rmk
```

;; Writes a BOM that represents FORMAT (:UTF-8, :UTF16-BE, :UTF16-LE

```
    (SELECTQ FORMAT
         (:UTF-8 (BOUT STREAM (CONSTANT (UNHEXSTRING "EF")))
                 (BOUT STREAM (CONSTANT (UNHEXSTRING "BB")))
                 (BOUT STREAM (CONSTANT (UNHEXSTRING "BF"))))
         (:UTF-16BE (BOUT STREAM (CONSTANT (UNHEXSTRING "FE")))
                    (BOUT STREAM (CONSTANT (UNHEXSTRING "FF"))))
         (:UTF-16LE (BOUT STREAM (CONSTANT (UNHEXSTRING "FF")))
                    (BOUT STREAM (UNHEXSTRING "FE")))
         NIL])

)

(RPAQ? EXTERNALEOL 'LF)

(DEFINEQ
```

(MAKE-UNICODE-FORMATS

```
  [LAMBDA (EXTERNALEOL)                                      ; Edited 10-Mar-2024 11:55 by rmk
                                                             ; Edited  8-Dec-2023 15:19 by rmk
                                                             ; Edited 19-Jul-2022 15:36 by rmk
                                                             ; Edited  6-Aug-2021 16:08 by rmk:
```

;; RAW formats do not do XCCS/Unicode translation, just deal with the byte encoding.

;; The EXTERNALEOL specifies the EOLCONVENTION of the stream, particularly to produce output files with the desired convention. On input ;; the macro \CHECKEOLC (LLREAD) coerces only that coding to the internal EOL, which is a mistake.

```
    (MAKE-EXTERNALFORMAT :UTF-8 (FUNCTION UTF8.INCCODEFN)
           (FUNCTION UTF8.PEEKCCODEFN)
           (FUNCTION \UTF8.BACKCCODEFN)
           (FUNCTION UTF8.OUTCHARFN)
          NIL EXTERNALEOL NIL NIL NIL (FUNCTION NILL))
    (MAKE-EXTERNALFORMAT :UTF-8-RAW [FUNCTION (LAMBDA (STREAM COUNTP)
                                                (UTF8.INCCODEFN STREAM COUNTP T]
           [FUNCTION (LAMBDA (STREAM NOERROR)
                       (UTF8.PEEKCCODEFN STREAM NOERROR T]
           [FUNCTION (LAMBDA (STREAM COUNTP)
                       (\UTF8.BACKCCODEFN STREAM COUNTP T]
           [FUNCTION (LAMBDA (STREAM CHARCODE)
                       (UTF8.OUTCHARFN STREAM CHARCODE T]
          NIL EXTERNALEOL NIL NIL NIL (FUNCTION NILL))
    (MAKE-EXTERNALFORMAT :UTF-16BE (FUNCTION UTF16BE.INCCODEFN)
           (FUNCTION UTF16BE.PEEKCCODEFN)
           (FUNCTION \UTF16BE.BACKCCODEFN)
           (FUNCTION UTF16BE.OUTCHARFN)
          NIL EXTERNALEOL NIL NIL NIL (FUNCTION NILL))
    (MAKE-EXTERNALFORMAT :UTF-16BE-RAW [FUNCTION (LAMBDA (STREAM COUNTP)
                                                   (UTF16BE.INCCODEFN STREAM COUNTP T]
           [FUNCTION (LAMBDA (STREAM NOERROR)
                       (UTF16BE.PEEKCCODEFN STREAM NOERROR T]
           [FUNCTION (LAMBDA (STREAM COUNTP)
                       (\UTF16BE.BACKCCODEFN STREAM COUNTP T]
           [FUNCTION (LAMBDA (STREAM CHARCODE)
                       (UTF16BE.OUTCHARFN STREAM CHARCODE T]
          NIL EXTERNALEOL NIL NIL NIL (FUNCTION NILL))
    (MAKE-EXTERNALFORMAT :UTF-16LE (FUNCTION UTF16LE.INCCODEFN)
           (FUNCTION UTF16LE.PEEKCCODEFN)
           (FUNCTION \UTF16LE.BACKCCODEFN)
           (FUNCTION UTF16LE.OUTCHARFN)
          NIL EXTERNALEOL NIL NIL NIL (FUNCTION NILL))
    (MAKE-EXTERNALFORMAT :UTF-16LE-RAW [FUNCTION (LAMBDA (STREAM COUNTP)
                                                   (UTF16LE.INCCODEFN STREAM COUNTP T]
           [FUNCTION (LAMBDA (STREAM NOERROR)
                       (UTF16LE.PEEKCCODEFN STREAM NOERROR T]
           [FUNCTION (LAMBDA (STREAM COUNTP)
                       (\UTF16LE.BACKCCODEFN STREAM COUNTP T]
           [FUNCTION (LAMBDA (STREAM CHARCODE)
                       (UTF16LE.OUTCHARFN STREAM CHARCODE T]
          NIL EXTERNALEOL NIL NIL NIL (FUNCTION NILL])

)

(MAKE-UNICODE-FORMATS EXTERNALEOL)

(ADDTOVAR *DEFAULT-EXTERNALFORMATS* (UNIX :UTF-8))

(DEFINEQ
```

(UNICODE.UNMAPPED

```
[LAMBDA (CODE TRANSLATION-TABLE ALREADYTRIED)                    ; Edited  2-Feb-2024 23:52 by rmk
                                                                 ; Edited 31-Jan-2024 10:07 by rmk
                                                                 ; Edited 11-Aug-2020 20:23 by rmk:
```

;; This is the slow fall-out when UNICODE.TRANSLATE determines that CODED has no fast mapping in TRANSLATION-TABLE.

;;

;; If we have not already dummied it up, we try to extend the current table by finding and merging the character set that has a mapping for CODE.

;;

;; When a proper mapping is not available we gin up a distinct unused code and put it in the hash array. If CODE has not previously been seen,
;; we allocate a new code in the forward unmapped hasharray and put the inverse in the backward array.

;; ALREADYTRIED suppresses the recursion through the extension attempt.

;; ALREADYTRIED breaks the loop of finding that CODE's character set doesn't have a mapping for CODE.

```
(LET (INVERSE NEXTCODE (FORWARD (CL:SVREF TRANSLATION-TABLE N-TRANSLATION-SEGMENTS)))
     (if (GETHASH CODE (CAR FORWARD))
       elseif (AND (NOT ALREADYTRIED)
                   (UNICODE-EXTEND-TRANSLATION? CODE TRANSLATION-TABLE))
       elseif (AND (ILEQ CODE (CADDR FORWARD))
                   (IGEQ CODE (CADDDR FORWARD)))
         then (ERROR "UNMAPPED CODE IS EITHER XCCS-UNUSED OR UNICODE-PRIVATE" CODE)
       else (SETQ INVERSE (CL:SVREF TRANSLATION-TABLE (ADD1 N-TRANSLATION-SEGMENTS)))
            (SETQ NEXTCODE (ADD (CADR INVERSE)
                                1))
            (CL:WHEN (IGREATERP NEXTCODE (CADDR INVERSE))
                   (ERROR "EXHAUSTED RANGE FOR UNMAPPED CODES" CODE))
            (PUTHASH CODE NEXTCODE (CAR FORWARD))
            (PUTHASH NEXTCODE CODE (CAR INVERSE))
            NEXTCODE])
```

(UNICODE-EXTEND-TRANSLATION?

```
[LAMBDA (CODE TRANSLATION-TABLE)                                 ; Edited 26-Aug-2024 16:49 by rmk
                                                                 ; Edited 27-Mar-2024 23:02 by rmk
                                                                 ; Edited  5-Feb-2024 13:48 by rmk
                                                                 ; Edited  3-Feb-2024 12:40 by rmk
```

;; There is currently no mapping for CODE in TRANSLATION-TABLE, hopefully just because the relevant character-set mapping has not been
;; installed. We infer from TRANSLATION-TABLE whether CODE is an XCCS or UNICODE code and look for the proper mapping table (forward or
;; inverted) for its character set.

```
(LET (MAPPING FILE (INVERTED (EQ TRANSLATION-TABLE *UNICODETOXCCS*)))
     (SETQ FILE (FINDFILE (CL:IF INVERTED
                                 'INVERTED-UNICODE-MAPPINGS.TXT
                                 'UNICODE-MAPPINGS.TXT)
                          T UNICODEDIRECTORIES))
     (CL:WHEN FILE
         [SETQ MAPPING (CL:WITH-OPEN-FILE (STREAM FILE :INPUT)
                                 (CL:WHEN (FFILEPOS (CONCAT "[" (LRSH CODE 8)
                                                            " ")
                                                    STREAM NIL NIL NIL T)
                                          (READ STREAM]
         (CL:WHEN MAPPING
             ;; Merge MAPPING into both tables, respecting the direction indicated by TRANSLATION-TABLE.
             (if INVERTED
                 then (MERGE-UNICODE-TRANSLATION-TABLES MAPPING *UNICODETOXCCS* *XCCSTOUNICODE*)
               else (MERGE-UNICODE-TRANSLATION-TABLES MAPPING *XCCSTOUNICODE* *UNICODETOXCCS*))
             ;; Hopefully we have now installed and can retrieve the mapping for CODE in its translation table.
             (UNICODE.TRANSLATE CODE TRANSLATION-TABLE T)))])
```

(UTF8.BINCODE

```
[LAMBDA (STREAM RAW)                                             ; Edited  4-Feb-2024 01:06 by rmk
                                                                 ; Edited  1-Feb-2024 11:21 by rmk
                                                                 ; Edited 28-Dec-2023 13:32 by rmk
                                                                 ; Edited  6-Aug-2021 16:02 by rmk:
                                                                 ; Edited  6-Aug-2020 17:13 by rmk:
```

;; Decodes a UTF8 character code by binning from STREAM

;; The validity of STREAM is guaranteed by the caller (presumably TEDIT), we aren't testing here for the validity of the trailing bytes.

;; This doesn't do EOL conversion or translation, unlike UTF8.INCCODEFN.

```
(LET ((BYTE1 (BIN STREAM))
      CODE)
     [SETQ CODE (if (ILEQ BYTE1 127)
                    then BYTE1
                  elseif (ILEQ BYTE1 223)
                    then                                         ; 2 bytes
                         (LOGOR (LLSH (LOADBYTE BYTE1 0 5)
                                      6)
                                (LOADBYTE (BIN STREAM)
                                          0 6))
                  elseif (ILEQ BYTE1 239)
                    then                                         ; 3 bytes
```

```
                              (LOGOR (LLSH (LOADBYTE BYTE1 0 4)
                                           12)
                                     (LLSH (LOADBYTE (BIN STREAM)
                                                     0 6)
                                           6)
                                     (LOADBYTE (BIN STREAM)
                                               0 6))
                           else                                          ; 4 bytes
                                (LOGOR (LLSH (LOADBYTE BYTE1 0 3)
                                             18)
                                       (LLSH (LOADBYTE (BIN STREAM)
                                                       0 6)
                                             12)
                                       (LLSH (LOADBYTE (BIN STREAM)
                                                       0 6)
                                             6)
                                       (LOADBYTE (BIN STREAM)
                                                 0 6]
                 (CL:IF RAW
                     CODE
                     (UNICODE.TRANSLATE CODE *UNICODETOXCCS*))])
```

(\UTF8.FETCHCODE

```
  [LAMBDA (CODESIZE BUFFER BYTEOFFSET)                                   ; Edited 28-Dec-2023 13:32 by rmk
                                                                         ; Edited  6-Aug-2021 16:02 by rmk:
                                                                         ; Edited  6-Aug-2020 17:13 by rmk:
```

;; Decodes a UTF8 byte sequence of size CODESIZE in BUFFER starting at BYTEOFFSET.

;; The validity of the thesize, buffer, and offset are guaranteed by the caller.

```
    (LET ((BYTE1 (\GETBASEBYTE BUFFER BYTEOFFSET))
          BYTE2 BYTE3 BYTE4)
         (SELECTQ CODESIZE
             (2 (SETQ BYTE2 (\UTF8.GETBASEBYTE BUFFER (IPLUS 1 BYTEOFFSET)))
                (LOGOR (LLSH (LOADBYTE BYTE1 0 5)
                             6)
                       (LOADBYTE BYTE2 0 6)))
             (3 (SETQ BYTE2 (\UTF8.GETBASEBYTE BUFFER (IPLUS 1 BYTEOFFSET)))
                (SETQ BYTE3 (\UTF8.GETBASEBYTE BUFFER (IPLUS 2 BYTEOFFSET)))
                (LOGOR (LLSH (LOADBYTE BYTE1 0 4)
                             12)
                       (LLSH (LOADBYTE BYTE2 0 6)
                             6)
                       (LOADBYTE BYTE3 0 6)))
             (4 (SETQ BYTE2 (\UTF8.GETBASEBYTE BUFFER (IPLUS 1 BYTEOFFSET)))
                (SETQ BYTE3 (\UTF8.GETBASEBYTE BUFFER (IPLUS 2 BYTEOFFSET)))
                (SETQ BYTE4 (\UTF8.GETBASEBYTE BUFFER (IPLUS 3 BYTEOFFSET)))
                (LOGOR (LLSH (LOADBYTE BYTE1 0 3)
                             18)
                       (LLSH (LOADBYTE BYTE2 0 6)
                             12)
                       (LLSH (LOADBYTE BYTE3 0 6)
                             6)
                       (LOADBYTE BYTE4 0 6)))
             (1 BYTE1)
             (SHOULDNT])

)

(DEFINEQ
```

(UTF8.VALIDATE

```
  [LAMBDA (STREAM BYTE)                                                  ; Edited  2-Feb-2024 12:03 by rmk
                                                                         ; Edited 28-Dec-2023 11:57 by rmk
                                                                         ; Edited  6-Aug-2021 16:02 by rmk:
                                                                         ; Edited  6-Aug-2020 17:13 by rmk:
```

;; Returns the codesize if the bytes starting at STREAM's current position form a valid UTF-8 sequence.

;; If BYTE is provided, it is interpreted as the just-read header byte with the stream is positioned just after it.

;; Test for smallp because the stream's End-of-file operation may suppress the error--otherwise an error will happen if the streams runs out of ;; necessary bytes.

;; For valid sequences, returns the same value as UTF8-SIZE-FROM-BYTE1, but this reads/validates the rest of the bytes. On a non-NILreturn the ;; stream is positioned before the header byte of the next putative code. The stream position is uncertain on a NIL return.

;;

;; Distinguish on the header byte BYTE. Not SMALLP presumably if ENDOFSTREAMOP did something unusual.

```
    (CL:UNLESS BYTE
        (SETQ BYTE (BIN STREAM)))
    (CL:WHEN (SMALLP BYTE)
        (if (ILEQ BYTE 127)
            then 1
          elseif (ILEQ BYTE 223)
            then                                                         ; 2 bytes
                 (CL:UNLESS (OR [NOT (SMALLP (SETQ BYTE (BIN STREAM]
                                (ILESSP BYTE 128))
```

```
                    2)
          elseif (ILEQ BYTE 239)
            then                                                    ; 3 bytes
                 (CL:UNLESS (OR (OR [NOT (SMALLP (SETQ BYTE (BIN STREAM]
                                    (ILESSP BYTE 128))
                                (OR [NOT (SMALLP (SETQ BYTE (BIN STREAM]
                                    (ILESSP BYTE 128)))
                    3)
          else                                                      ; 4 bytes
               (CL:UNLESS (OR (OR [NOT (SMALLP (SETQ BYTE (BIN STREAM]
                                  (ILESSP BYTE 128))
                              (OR [NOT (SMALLP (SETQ BYTE (BIN STREAM]
                                  (ILESSP BYTE 128))
                              (OR [NOT (SMALLP (SETQ BYTE (BIN STREAM]
                                  (ILESSP BYTE 128)))
                         4))))])
```

(UTF8-SIZE-FROM-BYTE1

```
  [LAMBDA (BYTE1)                                                   ; Edited  2-Feb-2024 11:50 by rmk
```

;; Returns the number of bytes of a UTF-8 code, given that BYTE1 is the first (header) byte of the sequence.

```
    (if (ILEQ BYTE1 127)
        then 1
      elseif (ILEQ BYTE1 223)
        then 2
      elseif (ILEQ BYTE1 239)
        then 3
      else 4])
```

(NUTF8-BYTE1-BYTES

```
  [LAMBDA (BYTE1)                                                   ; Edited  3-Feb-2024 15:00 by rmk
                                                                    ; Edited  8-Jan-2024 10:57 by rmk
                                                                    ; Edited 28-Jun-2022 00:02 by rmk
                                                                    ; Edited 10-Aug-2020 12:35 by rmk:
```

;; Returns the number of bytes in a UTF8 code representation whose first byte is BYTEE1.

```
    (IF (ILEQ BYTE1 127)
        THEN 1
      ELSEIF (ILEQ BYTE1 223)
        THEN 2
      ELSEIF (ILEQ BYTE1 239)
        THEN 3
      ELSE 4])
```

(NUTF8-CODE-BYTES

```
  [LAMBDA (CODE)                                                    ; Edited  3-Feb-2024 14:42 by rmk
                                                                    ; Edited  8-Jan-2024 10:57 by rmk
                                                                    ; Edited 28-Jun-2022 00:02 by rmk
                                                                    ; Edited 10-Aug-2020 12:35 by rmk:
```

;; Returns the number of bytes needed to encode in UTF8 a number headed by BYTE.

```
    (IF (ILESSP CODE 128)
        THEN 1
      ELSEIF (ILESSP CODE 2048)
        THEN                                                        ; x800
             2
      ELSEIF (ILESSP CODE 65536)
        THEN                                                        ; x10000
             3
      ELSEIF (ILESSP CODE 2097152)
        THEN                                                        ; x200000
             4
      ELSE (ERROR "INVALID UTF-8 CODE"])
```

(NUTF8-STRING-BYTES

```
  [LAMBDA (STRING RAW)                                              ; Edited  3-Feb-2024 21:32 by rmk
                                                                    ; Edited 10-Aug-2020 09:06 by rmk:
```

;; Returns the number of bytes it would take to represent STRING in UTF8, assuming it is an XCCS string unless RAWFLG.

```
    (FOR I C FROM 1 WHILE (SETQ C (NTHCHARCODE STRING I)) SUM (NUTF8-CODE-BYTES (CL:IF RAW
                                                                                  C
                                                                                  (XTOUCODE C))])

)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(PUTPROPS UNICODE.TRANSLATE MACRO [OPENLAMBDA (CODE TRANSLATION-TABLE ALREADYTRIED)
                                    (LET [(X (CL:SVREF TRANSLATION-TABLE (LRSH CODE TRANSLATION-SHIFT]
                                         (OR [COND
                                                ((LISTP X)
```

```
                                                    (CDR (FASSOC (LOGAND CODE TRANSLATION-MASK)
                                                                 X)))
                                                   (X (CL:SVREF X (LOGAND CODE TRANSLATION-MASK]
                                                  (UNICODE.UNMAPPED CODE TRANSLATION-TABLE ALREADYTRIED])

(PUTPROPS \UTF8.GETBASEBYTE MACRO ((BASE OFFSET ERROR?)          ; Fetches the OFFSET'th byte from BASE, checking for UTF-8
                                                                 ; validity if ERROR?
                                  (IF ERROR?
                                      THEN (LET ((BYTE (\GETBASEBYTE BASE OFFSET)))
                                                (CL:WHEN (ILESSP BYTE 128)
                                                         (ERROR "INVALID UTF8 BYTE" BYTE))
                                                BYTE)
                                    ELSE (\GETBASEBYTE BASE OFFSET))))
)
)

(DEFINEQ

(XTOUCODE
  [LAMBDA (XCCSCODE)                                             ; Edited  9-Aug-2020 09:04 by rmk:
    (UNICODE.TRANSLATE XCCSCODE *XCCSTOUNICODE*])

(UTOXCODE
  [LAMBDA (UNICODE)                                              ; Edited  9-Aug-2020 09:04 by rmk:
    (UNICODE.TRANSLATE UNICODE *UNICODETOXCCS*])

)

;;
;; Read Unicode mapping files

(RPAQ? UNICODEDIRECTORIES NIL)

(RPAQQ XCCS-CHARSETS
       ((LATIN "0")
        (JAPANESE-SYMBOLS1 "41")
        (JAPANESE-SYMBOLS2 "42")
        (EXTENDED-LATIN "43")
        (HIRAGANA "44")
        (KATAKANA "45")
        (GREEK "46")
        (CYRILLIC "47")
        (FORMS "50")
        (JIS "60-166")
        (ARABIC "340")
        (HEBREW "341")
        (IPA "342")
        (HANGUL "343")
        (GEORGIAN-ARMENIAN "344")
        (DEVANAGRI "345")
        (BENGALI "346")
        (GURMUKHI "347")
        (THAI-LAO "350")
        (SYMBOLS2 "356")
        (SYMBOLS1 "357")
        (LIGATURES "360")
        (ACCENTED-LATIN1 "361")
        (MORE-ARABIC "365")
        (GRAPHIC-VARIANTS "375")
        (DEFAULT LATIN ACCENTED-LATIN1 EXTENDED-LATIN SYMBOLS1 SYMBOLS2 FORMS JAPANESE-SYMBOLS1
              JAPANESE-SYMBOLS2)
        (JAPANESE HIRAGANA KATAKANA JIS)))

(DEFINEQ

(READ-UNICODE-MAPPING-FILENAMES
  [LAMBDA (FILESPEC)                                             ; Edited  3-Feb-2024 11:00 by rmk
                                                                 ; Edited 30-Jan-2024 08:45 by rmk
                                                                 ; Edited 26-Jan-2024 14:02 by mth
                                                                 ; Edited  5-Aug-2020 15:59 by kaplan
    ;; FILESPEC can be a file name, character-set name, the name of a collection of character sets, an XCCS character code, or list of those. Maps
    ;; those into the names of files that contain the indicated Unicode mappings.   ; Edited  4-Aug-2020 17:31 by rmk:
    (DECLARE (USEDFREE UNICODEDIRECTORIES XCCS-CHARSETS))
    (FOR F X CSI INSIDE FILESPEC JOIN
                                                                 ;; Last case hopes to pick up all the tables that are grouped together in a subdirectory (e.g. if F is
                                                                 ;; JIS)
                                  [OR (CL:WHEN (CHARCODEP F)     ; An XCCS code can retrieve its character set
                                               (for D FN (FOCTAL _ (OCTALSTRING (LRSH F 8))) inside
                                                                                             UNICODEDIRECTORIES
                                                  when [SETQ FN (FILDIR (PACKFILENAME 'DIRECTORY D 'BODY
                                                                                      (CONCAT 'XCCS- FOCTAL '=*)
                                                                                      'EXTENSION
                                                                                      'TXT]
                                                  do (RETURN FN)))
                                      (MKLIST (FINDFILE (PACKFILENAME 'BODY F 'EXTENSION 'TXT)
```

```
                                                T UNICODEDIRECTORIES))
                                          (for D inside UNICODEDIRECTORIES
                                             when [SETQ $$VAL (OR (FILDIR (PACKFILENAME 'NAME (CONCAT "XCCS-*="
                                                                                                      F)
                                                                                'EXTENSION
                                                                                'TXT
                                                                                'BODY D))
                                                                  (FILDIR (PACKFILENAME 'NAME
                                                                                (CONCAT "XCCS-" F "=*")
                                                                                'EXTENSION
                                                                                'TXT
                                                                                'BODY D]
                                             do (RETURN $$VAL))
                                          (AND (SETQ CSI (ASSOC F XCCS-CHARSETS))
                                               (READ-UNICODE-MAPPING-FILENAMES (CDR CSI)
                                                      UNICODEDIRECTORIES))
                                          (for D inside UNICODEDIRECTORIES
                                             when (DIRECTORYNAMEP (SETQ D (CONCAT D ">" F ">")))
                                             join (FILDIR (CONCAT D ">*.TXT;*"]
        finally (RETURN (CL:REMOVE-DUPLICATES $$VAL :TEST (FUNCTION STREQUAL])
```

(READ-UNICODE-MAPPING

```
  [LAMBDA (FILESPEC NOPRINT NOERROR)                    ; Edited  3-Feb-2024 00:21 by rmk
                                                        ; Edited  5-Jan-2024 12:26 by rmk
                                                        ; Edited  3-Jul-2021 13:37 by rmk:
```

;; Combines the char-mapping tables from FILES coded in the Uncode-CDROM format. Comments prefixed by # and

;; Column 1: Input hex code in the format 0xXXXX

;; Column 2: Corresponding Unicode code-sequence in the format

;; 0xXXXX ... 0xYYYY

;; Column 3: (after #) Character name in some mapping files, utf-8 character

;; for XCCS mapping files

;;

;; Result is a list of (fromcode tocode1 ... tocoden) integer lists (almost always with only a single tocode

```
    (FOR FILE [SEPBITTABLE _ (MAKEBITTABLE (CHARCODE (TAB SPACE] IN (READ-UNICODE-MAPPING-FILENAMES FILESPEC)
       JOIN (CL:WITH-OPEN-FILE (STREAM FILE :DIRECTION :INPUT :EXTERNAL-FORMAT :UTF-8-RAW)
                    (BIND LINE NAME CHARSET START FIRST (CL:UNLESS (FILEPOS "Name:" STREAM NIL NIL NIL T)
                                                           (ERROR "NOT A UNICODE MAPPING FILE" (FULLNAME STREAM)
                                                                  ))
                                                       (SETQ NAME (CL:STRING-TRIM " " (CL:READ-LINE STREAM NIL
                                                                                                    NIL)))
                                                       (SETQ CHARSET (CL:IF
                                                                        (FILEPOS "XCCS charset:" STREAM NIL NIL
                                                                                 NIL T)
                                                                        (CL:STRING-TRIM " "
                                                                                 (CL:READ-LINE STREAM NIL NIL))
                                                                        ""))
                                                       (CL:UNLESS NOPRINT
                                                                  ; Strip off XCCS in front of name
                                                           (PRINTOUT T T CHARSET " "
                                                                  [SUBSTRING NAME (CONSTANT (ADD1 (NCHARS
                                                                                                    "XCCS"]
                                                                  T))
                          WHILE (SETQ LINE (CL:READ-LINE STREAM NIL NIL)) WHEN (SETQ START
                                                                                  (STRPOSL SEPBITTABLE LINE 1 T))
                          UNLESS (EQ (CHARCODE %#)
                                     (NTHCHARCODE LINE START))
                          COLLECT (BIND END WHILE [SETQ END (OR (STRPOSL SEPBITTABLE LINE START)
                                                                (ADD1 (NCHARS LINE]
                                   COLLECT [CHARCODE.DECODE (SUBSTRING LINE START (SUB1 END)
                                                                       (CONSTANT (CONCAT]
                                   REPEATWHILE (AND (SETQ START (STRPOSL SEPBITTABLE LINE END T))
                                                    (NEQ (CHARCODE %#)
                                                         (NTHCHARCODE LINE START])

)

;; Make translation tables for  UTF external formats

(DEFINEQ

(MAKE-UNICODE-TRANSLATION-TABLES
  [LAMBDA (MAPPING REINSTALL)                           ; Edited  3-Feb-2024 00:24 by rmk
                                                        ; Edited 30-Jan-2024 09:54 by rmk
                                                        ; Edited 21-Aug-2021 13:12 by rmk:
                                                        ; Edited 17-Aug-2020 08:46 by rmk:
    (CL:UNLESS [AND (LISTP MAPPING)
                    (FOR PAIR IN MAPPING AS I TO 10 ALWAYS (AND (LISTP PAIR)
                                                                (CHARCODEP (CAR PAIR))
                                                                (CHARCODEP (CADR PAIR]
        (SETQ MAPPING (READ-UNICODE-MAPPING MAPPING T)))
```

;; MAPPING is the list of numeric code correspondence pairs constructed by applying READ-UNICODE-MAPPING to a Unicode mapping file.

```
;; This produces two recoding arrays, one maps left-side codes into right-side codes (e.g. XCCS or ISO8859-1 to Unicode), for printing, the other
;; maps right-side (Unicode) codes to corresponding right-side codes (e.g. XCCS).
;;
;; We assume that the left-to-right mapping into Unicode is functional, so that each left code maps to a unique right (Unicode) code, because
;; Unicode is presumably the most refined coding scheme.  But several Unicode codes may map to the same left code, for logically different codes
;; that happen to have the same glyphs. In that case the heuristic is to map each "from" code to the lowest of the possible "to" codes. This means
;; that round-trip reading/writing or writing/reading from one or both starting points may not always be lossless.
;;
;;  Each recoding array has 256 elements, one for each possible high-order byte of a character code.  An array entry is either NIL, a 256-array of
;; codes indexed by low-order bytes, or an alist of (lower-order-bytes . codes).  The latter is used to save space for sparsely populated character
;; sets.
;;
;; The element 256 of each array contains a hash table for characters that might be encountered in XCCS memory or Unicode files for which there
;; is no mapping.  Element 257 contains the corresponding inverse unmapped hash-array, so that UNICODE.TRANSLATE can update them
;; consistently.
;;
;; UNICODE.TRANSLATE assigns an unmapped Unicode character to a "not used" XCCS code position (from 5,0 to 40,FF, leaving other low
;; not-used sets for other internal uses (TEDIT?).
;;
;; An unmapped XCCS character is assigned a code in the "private use" code blocks 0xE000-F8FF
;;
;; If REINSTALL is T, the new mapping vectors replace the current maps in the *XCCSTOUNICODE* and *UNICODETOXCCS* global variables.
;; Values are also installed if those variables are NIL.
;;
(LET ((LTORARRAY (CL:MAKE-ARRAY (IPLUS 2 N-TRANSLATION-SEGMENTS)
                                :INITIAL-ELEMENT NIL))
      (RTOLARRAY (CL:MAKE-ARRAY (IPLUS 2 N-TRANSLATION-SEGMENTS)
                                :INITIAL-ELEMENT NIL)))
    ;; The left-to-right direction (into Unicode).  We start by distributing the mappings into alists in arrays indexed by the higher-order (charaset
    ;; set byte).  The second loop converts long alists into arrays.
    [FOR M LEFTC RBASE RCODES IN MAPPING EACHTIME (SETQ RCODES (CDR M))
                                                  (SETQ RBASE (CAR RCODES))
       UNLESS (IGEQ RBASE MISSINGCODE) DO (SETQ LEFTC (CAR M))
                                          ;; (CDR RCODES) contains combiners on the base
                                          (CL:PUSH (CONS (LOGAND LEFTC TRANSLATION-MASK)
                                                         (CL:IF (CDR RCODES)
                                                             RCODES
                                                             RBASE))
                                                   (CL:SVREF LTORARRAY (LRSH LEFTC TRANSLATION-SHIFT]
    (FOR I CSA FROM 0 TO (SUB1 N-TRANSLATION-SEGMENTS) WHEN (IGREATERP (LENGTH (CL:SVREF LTORARRAY I))
                                                                       MAX-ALIST-LENGTH)
       DO ;; Leave it alone if the alist is short
          (SETQ CSA (CL:MAKE-ARRAY TRANSLATION-SEGMENT-SIZE :INITIAL-ELEMENT NIL))
          (FOR P IN (CL:SVREF LTORARRAY I) DO (CL:SETF (CL:SVREF CSA (LOGAND (CAR P)
                                                                             TRANSLATION-MASK))
                                                       (CDR P)))
          (CL:SETF (CL:SVREF LTORARRAY I)
                 CSA))
    ;;
    ;; Now the right-to-left direction (from Unicode). Here we have to detect and compensate for ambiguity.
    (FOR M LEFTC RBASE RCOMBINERS PREV IN MAPPING EACHTIME (SETQ RBASE (CADR M))
                                                           (SETQ RCOMBINERS (CDDR M))
       UNLESS (OR (IGEQ RBASE MISSINGCODE)
                  RCOMBINERS)
       DO ;; Have we already seen an explicit mapping from right to left?
          (SETQ LEFTC (CAR M))
          [SETQ PREV (ASSOC (LOGAND RBASE TRANSLATION-MASK)
                            (CL:SVREF RTOLARRAY (LRSH RBASE TRANSLATION-SHIFT]
          (IF (NULL PREV)
              THEN (CL:PUSH (CONS (LOGAND RBASE TRANSLATION-MASK)
                                  LEFTC)
                            (CL:SVREF RTOLARRAY (LRSH RBASE TRANSLATION-SHIFT)))
            ELSEIF (IGREATERP (CDR PREV)
                              LEFTC)
              THEN (RPLACD PREV LEFTC)))
    (FOR I CSA FROM 0 TO (SUB1 N-TRANSLATION-SEGMENTS) WHEN (IGREATERP (LENGTH (CL:SVREF RTOLARRAY I))
                                                                       MAX-ALIST-LENGTH)
       DO ;; Long list, make an array
          (SETQ CSA (CL:MAKE-ARRAY TRANSLATION-SEGMENT-SIZE :INITIAL-ELEMENT NIL))
          (FOR P IN (CL:SVREF RTOLARRAY I) DO (CL:SETF (CL:SVREF CSA (LOGAND (CAR P)
                                                                             TRANSLATION-MASK))
                                                       (CDR P)))
          (CL:SETF (CL:SVREF RTOLARRAY I)
```

```
                    CSA))
        ;;
        ;; Allocate the hash arrays for future out-of-map codes.  We we have to keep track of the next available and last possible codes, as well as
        ;; the first available, for error checking.
        (CL:SETF (CL:SVREF LTORARRAY N-TRANSLATION-SEGMENTS)
               (LIST (HASHARRAY 10)
                     (CHARCODE.DECODE "5,0")
                     (CHARCODE.DECODE "40,0")
                     (CHARCODE.DECODE "5,0")))
        (CL:SETF (CL:SVREF RTOLARRAY N-TRANSLATION-SEGMENTS)
               (LIST (HASHARRAY 10)
                     (CHARCODE.DECODE "U+E000")
                     (CHARCODE.DECODE "U+F8FF")
                     (CHARCODE.DECODE "U+E000")))
        ;; Now put in the inverse unmapped hash arrays
        (CL:SETF (CL:SVREF LTORARRAY (ADD1 N-TRANSLATION-SEGMENTS))
               (CL:SVREF RTOLARRAY N-TRANSLATION-SEGMENTS))
        (CL:SETF (CL:SVREF RTOLARRAY (ADD1 N-TRANSLATION-SEGMENTS))
               (CL:SVREF LTORARRAY N-TRANSLATION-SEGMENTS))
        ;;
        (CL:WHEN [OR REINSTALL (NULL (GETATOMVAL '*XCCSTOUNICODE*]
            (SETQ *XCCSTOUNICODE* LTORARRAY)
            (SETQ *UNICODETOXCCS* RTOLARRAY))
        (LIST LTORARRAY RTOLARRAY])
```

(MERGE-UNICODE-TRANSLATION-TABLES

```
  [LAMBDA (ADDITION TABLE INVERSETABLE)                    ; Edited 27-Mar-2024 12:10 by rmk
                                                           ; Edited  3-Feb-2024 12:46 by rmk
                                                           ; Edited 31-Jan-2024 10:06 by rmk
    ;; ADDITION is a pair containing a mapping array and its inverse, or a list that maps codes in the forward direction.
    ;; The forward ADDITION mappings are merged destructively into TABLE and its inverses are merged into INVERSETABLE.
    (CL:UNLESS (AND (LISTP ADDITION)
                    (CL:ARRAYP (CAR ADDITION))
                    (CL:ARRAYP (CADR ADDITION)))
        ;; Make temporary mapping arrays when ADDTION is a list of corresponding code-pairs.
        (SETQ ADDITION (MAKE-UNICODE-TRANSLATION-TABLES ADDITION)))
    (MERGE-UNICODE-TRANSLATION-TABLES1 (CAR ADDITION)
          TABLE)
    (MERGE-UNICODE-TRANSLATION-TABLES1 (CADR ADDITION)
          INVERSETABLE)
    (LIST TABLE INVERSETABLE])
```

(MERGE-UNICODE-TRANSLATION-TABLES1

```
  [LAMBDA (ADDARRAY TARGETARRAY)                           ; Edited  2-Feb-2024 13:18 by rmk
                                                           ; Edited 31-Jan-2024 00:22 by rmk
    (for I TELT AELT (A _ ADDARRAY) from 0 TO (SUB1 N-TRANSLATION-SEGMENTS) when (SETQ AELT (CL:SVREF A I))
       do (SETQ TELT (CL:SVREF TARGETARRAY I))
          (CL:WHEN (EQ I 97))
          (CL:SETF (CL:SVREF TARGETARRAY I)
                 (if TELT
                     then ;; Have to resolve, union giving priority to AELT.  Have to deal with ALIST vs array cases on both sides.
                          (if (LISTP TELT)
                              then (if (LISTP AELT)
                                       then ;; 2 alists
                                            (SETQ TELT (APPEND AELT (for TE in TELT
                                                                       unless (ASSOC (CAR TE)
                                                                                     AELT)
                                                                       collect TE)))
                                            ;; Make an array if alist is too big
                                            (if (IGREATERP (LENGTH TELT)
                                                           MAX-ALIST-LENGTH)
                                                then (for TE (TARRAY _ (CL:MAKE-ARRAY TRANSLATION-SEGMENT-SIZE
                                                                                      :INITIAL-ELEMENT NIL))
                                                        in TELT do (CL:SETF (CL:SVREF TARRAY (CAR TE))
                                                                            (CDR TE))
                                                        finally (RETURN TARRAY))
                                              else TELT)
                                   else ;; Old Alist with new array.  Copy the TELT's into empty array positions
                                        (for TE TINDEX in TELT eachtime (SETQ TINDEX (CAR TE))
                                           unless (CL:SVREF AELT TINDEX) do (CL:SETF (CL:SVREF AELT TINDEX)
                                                                                     (CDR TE)))
                                        AELT)
                            elseif (LISTP AELT)
                              then ;; Old array with new Alist
```

```
                                    (for AE in AELT do (CL:SETF (CL:SVREF TELT (CAR AE))
                                                               (CDR AE)))
                                    TELT
                                else ;; 2 arrays. Smash AE value into TELT
                                    (for J AE from 0 to (SUB1 TRANSLATION-SEGMENT-SIZE)
                                       when (SETQ AE (CL:SVREF AELT J)) do (CL:SETF (CL:SVREF TELT J)
                                                                                    AE))
                                    TELT)
                            else AELT])

)

(DEFINEQ
```

(INVERT-ALL-UNICODE-MAPPINGS

```
  [LAMBDA (FILE)                                                    ; Edited 27-Mar-2024 14:50 by rmk
                                                                    ; Edited  5-Feb-2024 13:14 by rmk
                                                                    ; Edited  3-Feb-2024 09:16 by rmk
```

;; Reads all the XCCS-to-UNICODE mapping files that we know about, and produces a 2-level index that maps each UNICODE code back to the ;; one or more XCCS corresponding XCCS codes.

;; The first index level groups all the unicode codes that have the same high-ordere byte. The index is sorted by the high-order bytes, the pairs ;; within each group are sorted by their unicode. If a given unicode maps to multiple XCCS codes, the pair with the lowest XCCS code comes first.

;; GIven a UCODE, the lookup for the lowest XCCS is

;; (CADR (ASSOC UCODE (CADR (ASSOC (LRSH UCODE 8) INDEX)))).

;; If FILE is given, the resulting is written to that file.

```
   (LET (INDEX)
        [for M in (READ-UNICODE-MAPPING (for N in XCCS-CHARSETS collect (CAR N))
                                        T)
           do (push [CDR (OR (ASSOC (LRSH (CADR M)
                                          8)
                                    INDEX)
                             (CAR (push INDEX (CONS (LRSH (CADR M)
                                                          8]
                      (LIST (CADR M)
                            (CAR M]
```

;; Sort within groups to get the lowest XCCS code first. But also push the sublists down an extra CONS, so that a subsequent READ will get ;; them all.

```
        [for I in INDEX do (change (CDR I)
                                   (CONS (SORT DATUM (FUNCTION (LAMBDA (M1 M2)
                                                               (OR (ILESSP (CAR M1)
                                                                           (CAR M2))
                                                                   (AND (EQ (CAR M1)
                                                                            (CAR M2))
                                                                        (ILESSP (CADR M1)
                                                                                (CADR M2]
        (SETQ INDEX (SORT INDEX T))                                 ; Sort groups
        (if FILE
            then (CL:WITH-OPEN-FILE (STREAM [PACKFILENAME 'DIRECTORY (CAR (MKLIST UNICODEDIRECTORIES))
                                                          'BODY
                                                          (CL:IF (EQ FILE T)
                                                              'INVERTED-UNICODE-MAPPINGS.TXT
                                                              (PACKFILENAME 'BODY FILE 'EXTENSION 'TXT))]
                                            :DIRECTION :OUTPUT :IF-EXISTS :NEW-VERSION)
                 ;; We can FFILEPOS for "[nnn " then READ. Or just READFILE
                 (for I in INDEX do (PRINTOUT STREAM "[" (CAR I)
                                              " "
                                              (CADR I)
                                              "]" T))
                 (FULLNAME STREAM))
          else INDEX])
```

(ALL-UNICODE-MAPPINGS

```
  [LAMBDA (FILE)                                                    ; Edited 27-Mar-2024 14:48 by rmk
                                                                    ; Edited  5-Feb-2024 13:14 by rmk
                                                                    ; Edited  3-Feb-2024 09:16 by rmk
```

;; Reads all the XCCS-to-UNICODE mapping files that we know about, and iproduces a 2-level index that maps each XCCS code to the ;; corresponding UNICODE.

;; The first index level groups all the XCCS codes in the same character set. The index is sorted by the high-order bytes, the pairs within each ;; group are sorted by their XCCS code.

;; GIven a XCCS code, the lookup for the corresonding Unicode is

;; (CADR (ASSOC XCCSCODE (CADR (ASSOC (LRSH XCCSCODE 8) INDEX)))).

;; If FILE is given, the resulting is written to that file. If FILE is T, the file is UNICODE-MAPPINGS.TXT

```
   (LET (INDEX)
        [for M in (READ-UNICODE-MAPPING (for N in XCCS-CHARSETS collect (CAR N))
                                        T)
           do (push [CDR (OR (ASSOC (LRSH (CAR M)
                                          8)
```

```
                                    INDEX)
                                (CAR (push INDEX (CONS (LRSH (CAR M)
                                                             8]
                            (LIST (CAR M)
                                  (CADR M]
          ;; Push the sublists down an extra CONS, so that a subsequent READ will get them all.
          [for I in INDEX do (change (CDR I)
                                     (CONS (SORT DATUM (FUNCTION (LAMBDA (M1 M2)
                                                                 (OR (ILESSP (CAR M1)
                                                                             (CAR M2))
                                                                     (AND (EQ (CAR M1)
                                                                              (CAR M2))
                                                                          (ILESSP (CADR M1)
                                                                                  (CADR M2]
          (SETQ INDEX (SORT INDEX T))                                    ; Sort groups
          (if FILE
              then (CL:WITH-OPEN-FILE (STREAM [PACKFILENAME 'DIRECTORY (CAR (MKLIST UNICODEDIRECTORIES))
                                                            'BODY
                                                            (CL:IF (EQ FILE T)
                                                                'UNICODE-MAPPINGS.TXT
                                                                (PACKFILENAME 'BODY FILE 'EXTENSION 'TXT))]
                                           :DIRECTION :OUTPUT :IF-EXISTS :NEW-VERSION)
                       ;; We can FFILEPOS for "[nnn " then READ. Or just READFILE
                       (for I in INDEX do (PRINTOUT STREAM "[" (CAR I)
                                                           " "
                                                           (CADR I)
                                                           "]" T))
                       (FULLNAME STREAM))
            else INDEX])

)

(RPAQ? *XCCSTOUNICODE* )

(RPAQ? *UNICODETOXCCS* )

(RPAQ? *INVERTED-UNICODE-MAPPINGS* )

(DECLARE%: DOEVAL@COMPILE DONTCOPY

(GLOBALVARS *XCCSTOUNICODE* *UNICODETOXCCS*)
)

(DECLARE%: DONTEVAL@LOAD DOCOPY

(MAKE-UNICODE-TRANSLATION-TABLES 'DEFAULT)
)

(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(RPAQQ TRANSLATION-SEGMENT-SIZE 128)

(RPAQQ MAX-ALIST-LENGTH 10)

(RPAQ N-TRANSLATION-SEGMENTS (IQUOTIENT 65536 TRANSLATION-SEGMENT-SIZE))

(RPAQ TRANSLATION-SHIFT (INTEGERLENGTH (SUB1 TRANSLATION-SEGMENT-SIZE)))

(RPAQ TRANSLATION-MASK (SUB1 TRANSLATION-SEGMENT-SIZE))

(CONSTANTS (TRANSLATION-SEGMENT-SIZE 128)
       (MAX-ALIST-LENGTH 10)
       (N-TRANSLATION-SEGMENTS (IQUOTIENT 65536 TRANSLATION-SEGMENT-SIZE))
       (TRANSLATION-SHIFT (INTEGERLENGTH (SUB1 TRANSLATION-SEGMENT-SIZE)))
       (TRANSLATION-MASK (SUB1 TRANSLATION-SEGMENT-SIZE)))
)
)


;;
;; Write Unicode mapping files

(DEFINEQ

(WRITE-UNICODE-MAPPING
  [LAMBDA (MAPPING INCLUDECHARSETS FILE EMPTYOK)                 ; Edited  4-Jan-2024 22:44 by rmk
                                                                 ; Edited 16-Aug-2020 16:56 by rmk:
    ;; Writes a symbol unicode mapping file.  Mapping is a list of (XCCS-code Unicode) pairs, which may contain codes in multiple character sets.
    ;; If FILE is NIL, it defaults to a name XCCS- followed by the octal character sets in the mapping, in the unicode/XEROX directory.
    ;; The output lines are of the form x0XXX<tab>x0UUUU<tab># Unicode-char
    ;; If INCLUDECHARSETS=T then the mappings are split up into separate per-character set files.
```

```
;; Otherwise, all and only mappings included in thos charsets are included in a single output file--an implicit subset.
(IF (AND (EQ INCLUDECHARSETS T)
         (NULL FILE))
    THEN (IF MAPPING
             THEN (FOR CSI F IN XCCS-SET-NAMES WHEN (SETQ F (WRITE-UNICODE-MAPPING MAPPING (CAR CSI)
                                                                                   NIL T))
                     COLLECT F)
           ELSE (PRINTOUT T "THERE ARE NO MAPPINGS" T)
                NIL)
  ELSE (LET (IMAPPING CSETINFO RANGES)
            (CL:MULTIPLE-VALUE-SETQ (IMAPPING CSETINFO RANGES)
                  (WRITE-UNICODE-INCLUDED MAPPING INCLUDECHARSETS))
            (IF IMAPPING
                THEN (CL:WITH-OPEN-FILE
                      (STREAM (WRITE-UNICODE-MAPPING-FILENAME FILE CSETINFO RANGES)
                              :DIRECTION :OUTPUT :IF-EXISTS :NEW-VERSION :EXTERNAL-FORMAT :UTF-8-RAW)
                      (WRITE-UNICODE-MAPPING-HEADER STREAM CSETINFO RANGES)
                      (SORT IMAPPING T)
                      (FOR M CSET LEFTC FIRSTRIGHTC CSI IN IMAPPING
                         DO (SETQ LEFTC (CAR M))
                            (SETQ FIRSTRIGHTC (CADR M))
                            (CL:UNLESS (EQ CSET (LRSH LEFTC 8))
                                (SETQ CSET (LRSH LEFTC 8))
                                (SETQ CSI (ASSOC CSET CSETINFO))
                                (PRINTOUT STREAM T "#  " .P2 (CADR CSI)
                                       " "
                                       (CADDR CSI)
                                       T))
                            (PRINTOUT STREAM "0x" (HEXSTRING LEFTC 4)
                                   %#
                                   (FOR RIGHTC IN (CDR M) DO (PRINTOUT NIL "      " "0x" (HEXSTRING
                                                                                          RIGHTC 4))
                                        )
                                   "      #  "
                                   (SELECTC FIRSTRIGHTC
                                        (UNDEFINEDCODE ;; FFFF
                                            "UNDEFINED")
                                        (MISSINGCODE ;; FFFE
                                                "MISSING")
                                        (IF (ILESSP FIRSTRIGHTC 32)
                                            THEN ; Control chars
                                                 [CONCAT "^" (CHARACTER (IPLUS FIRSTRIGHTC (CHARCODE @]
                                          ELSE (CHARACTER FIRSTRIGHTC)))
                                   T))
                      (FULLNAME STREAM))
              ELSEIF (NOT EMPTYOK)
                THEN (PRINTOUT T "THERE ARE NO MAPPINGS")
                     (CL:WHEN INCLUDECHARSETS
                         (PRINTOUT T " FOR " .PPVTL (MKLIST INCLUDECHARSETS)
                                T))
                   NIL])
```

(WRITE-UNICODE-INCLUDED

```
[LAMBDA (MAPPING INCLUDECHARSETS)                          ; Edited  4-Aug-2020 17:47 by rmk:
    ;; CSETINFO is a list of (num string name) for each included character set.
    (LET (CHARSETS CSETINFO RANGES ICSETS IMAPPING)
         ;; Normalize the INCLUDECHARSETS, then reduce MAPPING to the included mappings
         [SETQ ICSETS (FOR C POS KNOWN INSIDE (OR INCLUDECHARSETS (FOR CSI IN XCCS-SET-NAMES
                                                                       COLLECT (CAR CSI)))
                         JOIN [SETQ KNOWN (OR (SASSOC C XCCS-SET-NAMES)
                                              (FIND N IN XCCS-SET-NAMES SUCHTHAT (EQ C (CADR N)))
                                              (HELP "UNKNOWN CHARACTER SET" (OCTALSTRING C]
                              (IF (SETQ POS (STRPOS "-" (CAR KNOWN)))
                                  THEN (FOR I FROM (CL:PARSE-INTEGER (SUBSTRING (CAR KNOWN)
                                                                                1
                                                                                (SUB1 POS))
                                                                     :RADIX 8)
                                         TO (CL:PARSE-INTEGER (SUBSTRING (CAR KNOWN)
                                                                         (ADD1 POS))
                                                              :RADIX 8)
                                         COLLECT (LIST I (OCTALSTRING I)
                                                       (CADR KNOWN)))
                                ELSE (CONS (CONS (CL:PARSE-INTEGER (CAR KNOWN)
                                                                   :RADIX 8)
                                                 KNOWN]
         (SETQ IMAPPING (FOR M CSI IN MAPPING WHEN (SETQ CSI (ASSOC (LRSH (CAR M)
                                                                          8)
                                                                    ICSETS))
                           COLLECT
                                ;; The attested subset of INCLUDED
```

```
                                (CL:UNLESS (MEMB CSI CSETINFO)
                                       (PUSH CSETINFO CSI))
                                M))
```

;; Sort as numbers, not octal strings, then group into consecutive ranges

```
(SETQ CSETINFO (SORT CSETINFO T))
[SETQ RANGES (FOR CTAIL C START END ON (FOR CSI IN CSETINFO COLLECT (CAR CSI)) WHILE CTAIL
                  COLLECT (SETQ START (CAR CTAIL))
                         (SETQ END START)
                         (CONS START (WHILE [AND (CDR CTAIL)
                                                 (EQ END (SUB1 (CADR CTAIL]
                                     COLLECT (SETQ CTAIL (CDR CTAIL))
                                             (SETQ END (CAR CTAIL]
```

;; Split out groups of less than 3. But if a range exhaustively covers a known subset (like JIS), replace by the name

```
[SETQ RANGES (FOR R STR KNOWN LAST IN RANGES
                JOIN (SETQ LAST (CAR (LAST R)))
                     (IF (EQ (CAR R)
                             LAST)
                         THEN (CONS (OCTALSTRING (CAR R)))
                        ELSEIF (SETQ KNOWN (SASSOC (SETQ STR (CONCAT (OCTALSTRING (CAR R))
                                                                     "-"
                                                                     (OCTALSTRING LAST)))
                                                   XCCS-SET-NAMES))
                         THEN (CONS (CADR KNOWN))
                        ELSEIF (CDDR R)
                         THEN (CONS STR)
                        ELSE (LIST (OCTALSTRING (CAR R))
                                   (OCTALSTRING LAST]
(CL:VALUES IMAPPING CSETINFO RANGES])
```

(WRITE-UNICODE-MAPPING-HEADER

```
[LAMBDA (STREAM CSETINFO RANGES)                                    ; Edited  5-Jan-2024 13:24 by rmk
                                                                    ; Edited  4-Aug-2020 17:38 by rmk:
```

;; Writes the standard per-file header information

```
(FOR LINE IN UNICODE-MAPPING-HEADER
   DO (PRINTOUT STREAM "#" 2)
      (SELECTQ LINE
           (XCCSCHARACTERSETS
                (PRINTOUT STREAM "        XCCS charset")
                (IF (CDR CSETINFO)
                    THEN (PRINTOUT STREAM "s:" -4)
                         (FOR R IN RANGES DO (PRINTOUT STREAM R " "))
                  ELSE                                              ; Singleton
                       (PRINTOUT STREAM ": " -4 (CADAR CSETINFO)
                                 " "
                                 (CADDAR CSETINFO)))
                (TERPRI STREAM))
           (DATE (PRINTOUT STREAM "        Date:" -13 (DATE (DATEFORMAT NO.TIME NO.LEADING.SPACES))
                           T))
           (PRINTOUT STREAM LINE T)))
(TERPRI STREAM])
```

(WRITE-UNICODE-MAPPING-FILENAME

```
[LAMBDA (FILE CSETINFO RANGES)                                      ; Edited  4-Aug-2020 19:34 by rmk:
  (PACKFILENAME 'BODY [OR FILE (CONCATLIST (CONS 'XCCS- (IF (CDR CSETINFO)
                                                            THEN (FOR RTAIL R ON RANGES
                                                                    JOIN (SETQ R (CAR RTAIL))
                                                                         (SETQ R (CL:IF (LISTP R)
                                                                                        (LIST (CAR R)
                                                                                              "-"
                                                                                              (CDR R))
                                                                                        (CONS R)))
                                                                         (CL:IF (CDR RTAIL)
                                                                                (NCONC1 R ","))
                                                                         R)
                                                          ELSE (LIST (CADAR CSETINFO)
                                                                     "="
                                                                     (CADDAR CSETINFO]
                      'DIRECTORY
                      (CAR UNICODEDIRECTORIES)
                      'EXTENSION
                      'TXT])
```

(HEXSTRING

```
[LAMBDA (N WIDTH)                                                   ; Edited 23-Jul-2020 08:28 by rmk:
                                                                    ; Edited 20-Dec-93 17:51 by rmk:
```

;; Converts positive numbers to Hex strings, padding on the right with 0 up to WIDTH if given.

```
(CL:UNLESS (FIXP N)
    (SETQ N (CHARCODE.DECODE N)))
(LET [CHAR (STR (ALLOCSTRING [IMAX (OR WIDTH 0)
                                   (FOR I (LEFT _ N) FROM 0 UNTIL (EQ LEFT 0)
```

```
                                        DO (SETQ LEFT (LRSH LEFT 4)) FINALLY (RETURN (MAX I 1]
                          (CHARCODE 0]
          (FOR I FROM -1 BY -1 UNTIL (EQ N 0) DO (SETQ CHAR (LOGAND N 15))
                                                 [RPLCHARCODE STR I (IF (ILESSP CHAR 10)
                                                                        THEN (+ CHAR (CHARCODE 0))
                                                                      ELSE (+ (- CHAR 10)
                                                                              (CHARCODE A]
                                                 (SETQ N (LRSH N 4)))
          STR])

)

(DEFINEQ
```

(XCCS-UTF8-AFTER-OPEN

```
  [LAMBDA (STREAM ACCESS PARAMETERS)                    ; Edited  3-Jan-2024 10:27 by rmk
                                                        ; Edited 13-Aug-2020 11:54 by rmk:
```

;; If added to STREAM-AFTER-OPEN-FNS, causes mapping files to be opened as UTF-8. For development

```
    (CL:WHEN (AND (STRPOS "XCCS-" (U-CASE (FULLNAME STREAM)))
                  [EQ 'TXT (U-CASE (FILENAMEFIELD (FULLNAME STREAM)
                                                  'EXTENSION]
                  (NOT (ASSOC 'EXTERNALFORMAT PARAMETERS)))
             (STREAMPROP STREAM 'EXTERNALFORMAT :UTF-8))])

)
```

;; Automate dumping of a documentation prefix

```
(DECLARE%: EVAL@COMPILE DONTCOPY

(DECLARE%: EVAL@COMPILE

(RPAQ MISSINGCODE (CL:PARSE-INTEGER "FFFE" :RADIX 16))

(RPAQ UNDEFINEDCODE (CL:PARSE-INTEGER "FFFF" :RADIX 16))

(CONSTANTS (MISSINGCODE (CL:PARSE-INTEGER "FFFE" :RADIX 16))
       (UNDEFINEDCODE (CL:PARSE-INTEGER "FFFF" :RADIX 16)))
)
)

(RPAQQ UNICODE-MAPPING-HEADER
       ("" "        Name:             XCCS (Version 2.0) to Unicode" "        Unicode version:  3.0"
           XCCSCHARACTERSETS "        Table version:    0.1" "        Table format:     Format A" DATE "
           Author:           Ron Kaplan <Ron.Kaplan@post.harvard.edu>" "" "This file contains mappings from the
           Xerox Character Code Standard (version" "2.0, 1990) into Unicode 3.0. standard codes.  That is an
           extension of the" "version of XCCS corresponding to the fonts in the Medley system." "" "The format
           of this file conforms to the format of the other Unicode-supplied" "mapping files:" "   Three
           white-space (tab or spaces) separated columns:" "     Column 1 is the XCCS code (as hex 0xXXXX)" "
           Column 2 is the corresponding Unicode (as hex 0xXXXX)" "     Column 3 (after #) is a comment column.
           For convenience, it contains the" "        Unicode character itself and the Unicode character names
           when available." "Unicode FFFF is used for undefined XCCS codes (Column 3 = UNDEFINED" "Unicode FFFE
           is used for XCCS codes that have not yet been filled in." "(Column 3 = MISSING)" "" "This file is
           encoded in UTF-8, so that the Unicode characters" "are properly displayed in Column 3 and can be
           edited by standard" "Unicode-enabled editors (e.g. Mac Textedit)." "" "This file can also be read by
           the function" "READ-UNICODE-MAPPING in the UNICODE Medley library package." "" "The entries are in
           XCCS order and grouped by character sets.  In front of" "the mappings, for convenience, there is a
           line with the octal XCCS" "character set, after #." "" "Note that a given XCCS code might map to
           codes in several different Unicode" "positions, since there are repetitions in the Unicode
           standard." "" "For more details, see the associated README.TXT file." "" "Any comments or problems,
           contact <ron.kaplan@post.harvard.edu>"))

(DEFINEQ
```

(UTF8HEXSTRING

```
  [LAMBDA (CHARCODE)                                    ; Edited 10-Aug-2020 08:33 by rmk:
```

;; Utility to produces the UTF8 hexstring representing CODE

```
    (HEXSTRING (IF (ILESSP CHARCODE 128)
                   THEN CHARCODE
                 ELSEIF (ILESSP CHARCODE 2048)
                   THEN                                 ; x800
                        (LOGOR (LLSH (LOGOR (LLSH 3 6)
                                            (LRSH CHARCODE 6))
                                     8)
                               (LOGOR (LLSH 2 6)
                                      (LOADBYTE CHARCODE 0 6)))
                 ELSEIF (ILESSP CHARCODE 65536)
                   THEN                                 ; x10000
                        (LOGOR (LLSH (LOGOR (LLSH 7 5)
                                            (LRSH CHARCODE 12))
                                     16)
                               (LLSH (LOGOR (LLSH 2 6)
                                            (LOADBYTE CHARCODE 6 6))
                                     8)
                               (LOGOR (LLSH 2 6)
```

```
                                    (LOADBYTE CHARCODE 0 6)))
                    ELSEIF (ILESSP CHARCODE 2097152)
                      THEN                                          ; x200000
                           (LOGOR (LLSH (LOGOR (LLSH 15 4)
                                               (LRSH CHARCODE 18))
                                        24)
                                  (LLSH (LOGOR (LLSH 2 6)
                                               (LOADBYTE CHARCODE 12 6))
                                        16)
                                  (LLSH (LOGOR (LLSH 2 6)
                                               (LOADBYTE CHARCODE 6 6))
                                        8)
                                  (LOGOR (LLSH 2 6)
                                         (LOADBYTE CHARCODE 0 6)))
                    ELSE (ERROR "CHARCODE too big for UTF8" CHARCODE])
```

(XTOUSTRING

```
  [LAMBDA (XCCSSTRING RAWFLG)                                       ; Edited  3-Feb-2024 14:55 by rmk
                                                                    ; Edited 10-Aug-2020 21:42 by rmk:
```

;; Produces a string that contains the UTF8 bytes that represent the characters in XCCSSTRING. Applies the XCCSTOUNICODE translation
;; unless RAWFLG.

;; The resulting string will not be readable inside Medley.

```
    (LET [(USTR (ALLOCSTRING (NUTF8-STRING-BYTES XCCSSTRING RAWFLG]
         (FOR I CHARCODE (SINDEX _ 0) FROM 1 WHILE (SETQ CHARCODE (NTHCHARCODE XCCSSTRING I))
            DO (CL:UNLESS RAWFLG
                     (SETQ CHARCODE (XTOUCODE CHARCODE)))
                (IF (ILESSP CHARCODE 128)
                    THEN (RPLCHARCODE USTR (ADD SINDEX 1)
                                 CHARCODE)
                  ELSEIF (ILESSP CHARCODE 2048)
                    THEN                                            ; x800
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 3 6)
                                       (LRSH CHARCODE 6)))
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 2 6)
                                       (LOADBYTE CHARCODE 0 6)))
                  ELSEIF (ILESSP CHARCODE 65536)
                    THEN                                            ; x10000
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 7 5)
                                       (LRSH CHARCODE 12)))
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 2 6)
                                       (LOADBYTE CHARCODE 6 6)))
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 2 6)
                                       (LOADBYTE CHARCODE 0 6)))
                  ELSEIF (ILESSP CHARCODE 2097152)
                    THEN                                            ; x200000
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 15 4)
                                       (LRSH CHARCODE 18)))
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 2 6)
                                       (LOADBYTE CHARCODE 12 6)))
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 2 6)
                                       (LOADBYTE CHARCODE 6 6)))
                         (RPLCHARCODE USTR (ADD SINDEX 1)
                                (LOGOR (LLSH 2 6)
                                       (LOADBYTE CHARCODE 0 6)))
                  ELSE (SHOULDNT)))
          USTR])
```

(XCCSSTRING

```
  [LAMBDA (CODE)                                                    ; Edited 13-Aug-2020 12:16 by rmk:
```

;; Returns XCCS character representation of string "cset,char"

```
    (CL:UNLESS (FIXP CODE)
        (SETQ CODE (CHCON1 CODE)))
    (CONCAT (OCTALSTRING (LRSH CODE 8))
           ","
           (OCTALSTRING (LOGAND CODE 255])

)

(DEFINEQ
```

(UNHEXSTRING

```
  [LAMBDA (HSTRING)                                                 ; Edited 10-Mar-2024 12:56 by rmk
```

;; Converts a hexstring to its number.

```
    (for I B (N _ 0) from 1 while (SETQ B (NTHCHARCODE HSTRING I))
```

```
       do [SETQ N (IPLUS (LLSH N 4)
                         (if (AND (IGEQ B (CHARCODE 0))
                                  (ILEQ B (CHARCODE 9)))
                             then (IDIFFERENCE B (CHARCODE 0))
                           elseif (AND (IGEQ (SETQ B (UCASECODE B))
                                             (CHARCODE A))
                                       (ILEQ B (CHARCODE F)))
                             then (IPLUS 10 (IDIFFERENCE B (CHARCODE A)))
                           else (ERROR "INVALID HEX CHARACTER" (NTHCHARCODE HSTRING I]
       finally (RETURN N])

)

(DEFINEQ
```

(SHOWCHARS

```
  [LAMBDA (FROMCHAR TOCHAR FONT)                                    ; Edited 26-Jan-2024 14:18 by mth
                                                                    ; Edited  1-Aug-2020 09:27 by rmk:
    (RESETFORM (DSPFONT (OR FONT '(CLASSIC 12))
                        T)
           (CL:WHEN (AND (SMALLP FROMCHAR)
                         (NOT TOCHAR))
               ;; If a small number, assume it's an octal (in decimal) character set, no need for string quotes
               (SETQ TOCHAR (CONCAT FROMCHAR "," 376))
               (SETQ FROMCHAR (CONCAT FROMCHAR "," 41)))
           (CL:UNLESS (SMALLP FROMCHAR)
               (SETQ FROMCHAR (CHARCODE.DECODE FROMCHAR)))
           (CL:UNLESS (SMALLP TOCHAR)
               (SETQ TOCHAR (CL:IF TOCHAR
                                   (CHARCODE.DECODE TOCHAR)
                                   FROMCHAR)))
           (for C from FROMCHAR to TOCHAR unless (AND (IGEQ (LOGAND C 255)
                                                            127)
                                                      (ILEQ (LOGAND C 255)
                                                            (PLUS 128 33)))
              do (PRINTOUT T .P2 (CONCAT (OCTALSTRING (LRSH C 8))
                                         ","
                                         (OCTALSTRING (LOGAND C 255)))
                        10
                        (CHARACTER C)
                        T])

)

(DECLARE%: EVAL@COMPILE DONTCOPY

(FILESLOAD (FROM LOADUPS)
       EXPORTS.ALL)
)

(PUTPROPS UNICODE FILETYPE :TCOMPL)
```

FUNCTION INDEX

VARIABLE INDEX

CONSTANT INDEX

MACRO INDEX

PROPERTY INDEX