2.2 Specification

The FSS-UTF encodes UCS values in the range [0,0x7FFFFFFF] using multi-byte characters of lengths 1, 2, 3, 4, 5 and 6 bytes.

For all encodings of more than one byte, the initial byte specifies the number of bytes used by setting 1 in the equivalent number of high-order bits. The next most significant bit is always 0. For example, a 2-byte sequence starts with 110 and a 6-byte sequence starts with 1111110.

The following table shows the format of the first byte of a character; the free bits available for coding the character are indicated by X.

Single byte	0XXXXXXX	seven free bits
First of two bytes	110XXXXX	five free bits
First of three bytes	1110XXXX	four free bits
First of four bytes	11110XXX	three free bits
First of five bytes	111110XX	two free bits
First of six bytes	1111110X	one free bit

All subsequent bytes in multi-byte characters have the following format:

10XXXXXX six free bits

For all subsequent bytes the two most significant bits are set to 10. Therefore, every byte that does not start with 10 is the start of a UCS character sequence.

Figure 2-1 illustrates the FSS-UTF:

No. of bytes in char.	No. of bits available for coding	Byte Sequence in Binary
1	7	0vvvvvvv
2	11	110vvvvv 10vvvvvv
3	16	1110vvvv 10vvvvvv 10vvvvvv
4	21	11110vvv 10vvvvvv 10vvvvvv 10vvvvvv
5	26	111110vv 10vvvvvv 10vvvvvv 10vvvvvv 10vvvvvv
6	31	1111110v 10vvvvvv 10vvvvvv 10vvvvvv 10vvvvvv 10vvvvvv

Figure 2-1 Transformation Format

The UCS value is the concatenation of the v bits in the multi-byte encoding. When there are multiple ways to encode a value, for example, UCS 0, only the shortest encoding is legal. The range of values possible with each type of byte sequence is shown in the following table.

No. of bytes in sequence	Hexadecimal Value Minimum	Maximum
1	00000000	0000007F
2	00000080	000007FF
3	00000800	0000FFFF
4	00010000	001FFFFF
5	00200000	03FFFFFF
6	04000000	7FFFFFFF

2.3 Example

The example below implements the ISO C standard *wctomb*() and *mbtowc*() functions, to demonstrate the algorithms for converting from UCS to FSS-UTF and converting from FSS-UTF to UCS. The example includes error checks, some of which may not be necessary for conformance:

```
typedef
struct
{
    int      cmask;
    int      cval;
    int      shift;
    long     lmask;
    long     lval;
} Tab;

static
Tab    tab[] =
{
    0x80,   0x00,   0*6,   0x7F,          0,          /* 1 byte sequence */
    0xE0,   0xC0,   1*6,   0x7FF,         0x80,       /* 2 byte sequence */
    0xF0,   0xE0,   2*6,   0xFFFF,        0x800,      /* 3 byte sequence */
    0xF8,   0xF0,   3*6,   0x1FFFFF,      0x10000,    /* 4 byte sequence */
    0xFC,   0xF8,   4*6,   0x3FFFFFF,     0x200000,   /* 5 byte sequence */
    0xFE,   0xFC,   5*6,   0x7FFFFFFF,    0x4000000,  /* 6 byte sequence */
    0,                                                /* end of table */
};

int
mbtowc(wchar_t *p, char *s, size_t n)
{
    long l;
    int c0, c, nc;
    Tab *t;

    if(s == 0)
        return 0;

    nc = 0;
    if(n <= nc)
        return -1;
    c0 = *s & 0xff;
    l = c0;
    for(t=tab; t->cmask; t++) {
        nc++;
        if((c0 & t->cmask) == t->cval) {
            l &= t->lmask;
            if(l < t->lval)
                return -1;
            *p = l;
            return nc;
        }
```

```
        if(n <= nc)
            return -1;
        s++;
        c = (*s ^ 0x80) & 0xFF;
        if(c & 0xC0)
            return -1;
        l = (l<<6) | c;
    }
    return -1;
}

int
wctomb(char *s, wchar_t wc)
{
    long l;
    int c, nc;
    Tab *t;

    if(s == 0)
        return 0;

    l = wc;
    nc = 0;
    for(t=tab; t->cmask; t++) {
        nc++;
        if(l <= t->lmask) {
            c = t->shift;
            *s = t->cval | (l>>c);
            while(c > 0) {
                c -= 6;
                s++;
                *s = 0x80 | ((l>>c) & 0x3F);
            }
            return nc;
        }
    }
    return -1;
}
```