AN-16 (Single Byte Character Set Font File Format)

February 24, 2006

The printer will accept fonts via download. Each font consists of a header followed by a series of graphic bit maps of each character, in ASCII order (e.g. the first character might be a space, followed by an exclamation point, followed by double quotes etc), or whatever mapping is represented by that font. The file must be in binary. There are several versions of .FON files - each version uses a slightly different header.

Each character of glyph is represented by its own bitmap. Each row of each character represented contains an integer number of bytes even if all bits within that byte are not used. A "one" will turn the paper dark; a "zero" will leave the paper white. There must be a bit mapped entry for each character within the sequence to be represented. If the font is proportional, then an extra 2 bytes is added to the beginning of the bitmap for each glyph. Those 2 bytes contain the width in bits of that particular character.

Each font has a unique one character and five character name (e.g. MF226, PT10B, FONT1). The one character name is used in the line printer mode to select the font via the ESC w command, and as an abbreviated method to refer to the font in Easy Print protocol. The five character name is used in Easy Print protocol only. You may select any one and five character names, but they MUST BE UNIQUE. If they are NOT unique, then the first font downloaded with that name will be selectable and the rest will be unusable.

To speed finding fonts, each five character name has a MOD 256 summation of that name in the header. To calculate this value, ADD the ASCII value of each character (in HEX) and use the lower order byte. For example, the characters in the name PT10B have ASCII values 50H, 54H, 31H, 30H, and 42H which add to 147H. Using just the lower order byte, we would enter 47H in the table where it says "MOD 256 summation..."

Version 1.0 HEADER (thermal only):

The version 1.0 header fonts are used in the MF2, MF3, and 2t/4t printers with IrDA

The 54 bytes (ALL bytes must be present) within the header are as follows:

4 BYTES	May be anything, rewritten internally
3 BYTES	Font version number (must be "1.0")
1 BYTE	Mod 256 summation of the five character name
5 BYTES	Five character name for this font
1 BYTE	One character name for this font
1 BYTE	Must be 00 for a monospace font Must be 05 for a proportional font
2 BYTES, LSB 1st	Number of dots wide for a monospace font 0xFFFF for a proportional font
2 BYTES, LSB 1st	Number of dots high
1 BYTE	Number of bytes in each row
2 BYTES	Number of bytes in each character
1 BYTE	First ASCII character represented in this font
1 BYTE	Last ASCII character represented in this font
1 BYTE	Reserved
1 BYTE	USER version number
8 BYTE	USER creation date
20 BYTES	USER description

Version 1.1 HEADER (impact only):

The version 1.1 header fonts are used in the 8i Flash 180 CPS impact printers

The 55 bytes (ALL bytes must be present) within the header are as follows:

4 BYTES	May be anything, rewritten internally
3 BYTES	Font version number (must be "1.1")
1 BYTE	Mod 256 summation of the five character name
5 BYTES	Five character name for this font
1 BYTE	One character name for this font
1 BYTE	Must be 00 for monospace font Must be 05 for proportional font
2 BYTES, LSB 1st	Number of dots wide for monospace font 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots high
1 BYTE	Number of bytes in each row
2 BYTES	Number of bytes in each character
1 BYTE	Number of added spaces for compressed (impact)
1 BYTE	First ASCII character represented in this font
1 BYTE	Last ASCII character represented in this font
1 BYTE	Dotline to place underline (impact)
1 BYTE	USER version number
8 BYTE	USER creation date
20 BYTES	USER description

Version 1.3 HEADER (thermal and impact):

The 71 bytes (ALL bytes must be present) within the header are as follows:

4 BYTES	May be anything, rewritten internally
4 BYTES	Font version number (must be "1.3",0x00 terminating NUL)
1 BYTE	Mod 256 summation of the five character name
6 BYTES	Five character name for this font with 0x00 terminating NUL

Impact:

1 BYTE	One character name for this font
1 BYTE	Name of PICA pitch font in this set (impact)
1 BYTE	Name of ELITE pitch font in this set (impact)
1 BYTE	Name of ITALIC PICA pitch font in this set (impact)
1 BYTE	Name of ITALIC ELITE pitch font in this set

OR Thermal:

1 BYTE	One character name for this font
1 BYTE	One character name for this font
1 BYTE	One character name for this font
1 BYTE	One character name for this font
1 BYTE	One character name for this font

Impact AND Thermal:

1 BYTE	Must be 00 for monospace font Must be 05 for proportional font
1 BYTE	Must be 01 if this font is to appear on the self test printout Must be 00 if this font is NOT to appear on the self test printout

Impact:

2 BYTES, LSB 1st	Actual number of dots wide 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots wide, monospace font, PICA pitch 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots wide for monospace font, PICA condensed 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots wide for monospace font, ELITE pitch 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots wide for monospace font, ELITE condensed 0xFFFF for proportional font

OR Thermal:

2 BYTES, LSB 1st	Number of dots wide for monospace font 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots wide for monospace font 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots wide for monospace font 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots wide for monospace font 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots wide for monospace font 0xFFFF for proportional font
2 BYTES, LSB 1st	Number of dots high
1 BYTE	Number of bytes in each row
2 BYTES	Number of bytes in each character
1 BYTE	First ASCII character represented in this font
1 BYTE	Last ASCII character represented in this font
1 BYTE	Dotline to place underline (impact)
1 BYTE	USER version number
9 BYTE	USER creation date (8 bytes plus terminating NUL)
21 BYTES	USER description (20 bytes plus terminating NUL)

EACH CHARACTER:

Characters are represented with a bitmap within a "cell". The cell represents a character position on the printed line. The cell must extend upwards to the top of the highest character *and* down to the bottom of the lowest character. Characters are left justified within this cell; white space between characters can be adjusted but exist on the right side within this cell.

The "cell" for each font may be any width, but the downloaded image width must be a multiple of 8 wide (since 8 is the number of bits within a byte), and may be any height. Each character then contains a series of bytes; one or more bytes constitute each dot line row of the character; a sequence of these byte(s), then will build the entire character one dot line row at a time.

Each dot within the cell is 1/200 inch tall and 1/200 inch wide. To build a character approximately 0.07 inches tall and .06 inches wide, we would use 14 dotlines high and 12 dots wide. Since the 12 dot width of the character would need to be represented by two bytes, we could specify a character width of 12 dots for a zero dot spacing between characters, 13 dots for a one dot spacing, on up to 16 dots for a 4 dot spacing. This size character would probably look best with a 2 dot spacing, so we would specify a width of 14 dots total in the font header.

If the font is a proportionally spaced font, then two bytes are added to the beginning of EACH bitmap that represents a character. The two bytes are the width of that particular character, LSB first.

The letter "A" then might be built as follows (note that even though the character is only 12 dots wide, we must represent it with two full bytes, or 16 dots):

CELL EXAMPLE #1:

```
0000011000000000   ->    006h & 000h
0000011000000000   ->    006h & 000h
0000111100000000   ->    00fh & 000h
0000111100000000   ->    00fh & 000h
0001111110000000   ->    01fh & 080h
0001100110000000   ->    019h & 080h
0011100111000000   ->    039h & 0c0h
0011111111000000   ->    03fh & 0c0h
0111111111100000   ->    07fh & 0e0h
0110000001100000   ->    060h & 060h
1110000001110000   ->    0e0h & 070h
1100000000110000   ->    0c0h & 030h
1100000000110000   ->    0c0h & 030h
1100000000110000   ->    0c0h & 030h
```

The "cell" for this character is 14 dots high and 12 dots wide. If, in the same font, we wanted to also represent the lower case "j", with descenders (that part of the character below an imaginary line along the bottom of all upper case letters) we would need to increase the cell size to 20 high from 14 high:

CELL EXAMPLE #2:

```
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000000 00000000  ->    000h & 000h
00000000 00000000  ->    000h & 000h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
00000001 10000000  ->    001h & 080h
01100001 10000000  ->    061h & 080h
01110011 10000000  ->    073h & 080h
00111111 00000000  ->    03fh & 000h
00011110 00000000  ->    01eh & 000h
```

The letter "A" would need to allow for the increased cell size, and would become:

```
0000011000000000   ->    006h & 000h
0000011000000000   ->    006h & 000h
0000111100000000   ->    00fh & 000h
0000111100000000   ->    00fh & 000h
0001111110000000   ->    01fh & 080h
0001100110000000   ->    019h & 080h
0011100111000000   ->    039h & 0c0h
0011111111000000   ->    03fh & 0c0h
0111111111100000   ->    07fh & 0e0h
0110000001100000   ->    060h & 060h
1110000001110000   ->    0e0h & 070h
1100000000110000   ->    0c0h & 030h
1100000000110000   ->    0c0h & 030h
1100000000110000   ->    0c0h & 030h
0000000000000000   ->    000h & 000h
0000000000000000   ->    000h & 000h
0000000000000000   ->    000h & 000h
0000000000000000   ->    000h & 000h
0000000000000000   ->    000h & 000h
0000000000000000   ->    000h & 000h
```

FONT FILE EXAMPLE – VERSION 1.0 MONOSPACE:

Since the file must be in binary, it is often easiest to create using an assembler. The following example shows the header, and two of the characters in a form ready to assemble. The object code output, then, can be sent to the printer using the Oneil WINDOWS configuration program, or a special command.

```
dl      PT10B_LINK      ; Value is reset internally and can be any 4 byte number
db      "1.0"           ; font version number (1.0 flags this header and respresentation)
db      047H            ; code resulting from mod 256 summation of name of font below
db      "PT10B"         ; name of this font
db      'E'             ; single letter used as an alternate to refer to this font
db      00d             ; use 00 for straight text
dw      14d             ; (ABSOLUTE = 12) number of dots wide for this font
dw      20d             ; number of dots high for this font
db      02d             ; number of bytes in each dot row of each character
dw      40d             ; number of bytes in each character represented in this font
db      'A'             ; The letter "A" is first ASCII character represented
db      'B'             ; The letter "B" is last ASCII character represented
db      00d             ; reserved
db      '1'             ; USER font version number
db      "04/30/96"      ; USER font creation date
db      "2 CHARS EXAMPLE FONT"      ; USER 20 character description
```

```
; BIT MAPPED PATTERN OF LETTER "A":
        db      006h,000h
        db      006h,000h
        db      00fh,000h
        db      00fh,000h
        db      01fh,080h
        db      019h,080h
        db      039h,0c0h
        db      03fh,0c0h
        db      07fh,0e0h
        db      060h,060h
        db      0e0h,070h
        db      0c0h,030h
        db      0c0h,030h
        db      0c0h,030h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h

; BIT MAPPED PATTERN OF LETTER "B":
        db      0ffh,0e0h
        db      0ffh,0f0h
        db      0c0h,070h
        db      0c0h,030h
        db      0c0h,030h
        db      0c0h,070h
        db      0ffh,0e0h
        db      0ffh,0e0h
        db      0c0h,070h
        db      0c0h,030h
        db      0c0h,030h
        db      0c0h,070h
        db      0ffh,0f0h
        db      0ffh,0e0h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h

PT10B_LINK:
        END
```

FONT FILE EXAMPLE – VERSION 1.0 PROPORTIONAL:

Since the file must be in binary, it is often easiest to create using an assembler. The following example shows the header, and two of the characters in a form ready to assemble. The object code output, then, can be sent to the printer using the Oneil WINDOWS configuration program, or a special command.

```
        dl      PT10B_LINK      ; Value is reset internally and can be any 4 byte number
        db      "1.0"           ; font version number (1.0 flags this header and respresentation)
        db      047H            ; code resulting from mod 256 summation of name of font below
        db      "PT10B"         ; name of this font
        db      'E'             ; single letter used as an alternate to refer to this font
        db      05d             ; use 05 for proportional text
        dw      0FFFFH          ; (ABSOLUTE = 12) number of dots wide for this font
        dw      20d             ; number of dots high for this font
        db      02d             ; number of bytes in each dot row of each character
        dw      40d             ; number of bytes in each character represented in this font
        db      'A'             ; The letter "A" is first ASCII character represented
        db      'B'             ; The letter "B" is last ASCII character represented
        db      00d             ; reserved
        db      '1'             ; USER font version number
        db      "04/30/96"      ; USER font creation date
        db      "2 CHARS EXAMPLE FONT"      ; USER 20 character description

; BIT MAPPED PATTERN OF LETTER "A":
        dw      13d             ; character is 11 wide plus 2 whitespace
        db      006h,000h
        db      006h,000h
        db      00fh,000h
        db      00fh,000h
        db      01fh,080h
        db      019h,080h
        db      039h,0c0h
        db      03fh,0c0h
        db      07fh,0e0h
        db      060h,060h
        db      0e0h,070h
        db      0c0h,030h
        db      0c0h,030h
        db      0c0h,030h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
```

```
; BIT MAPPED PATTERN OF LETTER "B":
        dw      14d             ; character is 12 wide plus 2 whitespace
        db      0ffh,0e0h
        db      0ffh,0f0h
        db      0c0h,070h
        db      0c0h,030h
        db      0c0h,030h
        db      0c0h,070h
        db      0ffh,0e0h
        db      0ffh,0e0h
        db      0c0h,070h
        db      0c0h,030h
        db      0c0h,030h
        db      0c0h,070h
        db      0ffh,0f0h
        db      0ffh,0e0h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h
        db      000h,000h

PT10B_LINK:
        END
```