The Interactive Disassembler - Hexadecimal file formats

All these formats are line oriented and use only printable ASCII characters except for the carriage return/line feed at the end of each line.

Intel HEX Format (.HEX)

Code of this format is often downloaded from a PC to a development system and run from RAM. Alternatively, the hex file can be converted to a binary file and programmed into an EPROM. This format produces one 8-bit HEX file with a low byte, high byte combination. Each data record begins with a 9 character prefix and ends with a 2 character checksum. Each record has the following format:

```
:BBAAAATTHHHH....HHCC
```

where

BB
: is a two digit hexadecimal byte count representing the number of data bytes that will appear on the line;

AAAA
: is a four digit hexadecimal address representing the starting address of the data record;

TT
: is a two digit record type:

- 00 - Data record
- 01 - End of file record
- 02 - Extended segment address record
- 03 - Start segment address record
- 04 - Extended linear address record
- 05 - Start linear address record

HH
: is a two digit hexadecimal data word, presented in low byte, high byte combinations;

CC
: is a two digit hexadecimal checksum that is the two's compliment of the sum of all preceding bytes in the record including the prefix; (sum of all bytes + checksum = 00).

INTEL Split 8-bit HEX Format INHX8S (.HXL/.HXH or .OBL/.OBH)

The Intellec split 8-bit file format produces two output files: .HXL and .HXH. The format is the same as the normal 8-bit format, except that the low bytes of the data word are stored in the .HXL file, and the high bytes of the data word are stored at the .HXH file

---

INTEL 32-bit HEX Format INHX32 (Intel Extended HEX Format) (.HEX)

The extended 32-bit address HEX format is similar to the Hex 8 format describred above, except that the Intel extended linear address record is output also to establish the upper 16 bits of the data address.

Each data record begins with 9 charcter prefix and ends with a 2 character checksum. Each record has the following format:

```
:BBAAAATTHHHH....HHCC
```

where

BB
: is a two digit hexadecimal byte count representing the number of data bytes that will appear on the line;

AAAA
: is a four digit hexadecimal address representing the starting address of the data record;

TT
: is a two digit record type:

- 00 - Data record
- 01 - End of file record
- 02 - Extended segment address record
- 03 - Start segment address record
- 04 - Extended linear address record
- 05 - Start linear address record

HH
: is a two digit hexadecimal data word, presented in low byte, high byte combinations;

CC
: is a two digit hexadecimal checksum that is the two's compliment of the sum of all preceding bytes in the record including the prefix;

PIC 8-bit HEX Format INHX8M (.OBJ)

This format produces one 8-bit hexadecimal file with a low-byte/high-byte combination. Since each address can only contain 8 bits in this format, all addresses are doubled, low byte first. This format are useful for transferring PIC series object code to the third party EPROM programmers.

PIC 16-bit HEX Format INHX16

The 16-bit word format is basically the same as the 8-bit word format. The difference between this format and INHX8M is the word length and the high/low byte order. INHX8M has 8-bit words (two hexadecimal digits) with the low byte first, rather than 16-bit words (four hexadecimal digits) with the high byte first. The byte count (BB), however, is based on 16-bit words.

S-Records

Ordinary S-Records cannot hold anything but addresses and data. S-Records may come out of order and there is no header.

An s-record looks like:

```
S<type><length><address><data><checksum>
```

where

length
: is the number of bytes following upto the checksum. Note that this is not the number of chars following, since it takes two chars to represent a byte.

type
: is one of:

- 0) header record
- 1) two byte address data record
- 2) three byte address data record
- 3) four byte address data record
- 7) four byte address termination record
- 8) three byte address termination record
- 9) two byte address termination record

address
: is the start address of the data following, or in the case of a termination record, the start address of the image

data
: is the data.

checksum
: is the sum of all the raw byte data in the record, from the length upwards, modulo 256 and subtracted from 255.

Tektronix Hex Format

Tek Hex records can hold symbols and data, but not relocations. Their main application is communication with devices like PROM programmers and ICE equipment. TekHex may come out of order and there is no header. A TekHex record looks like:

```
%<block length><type><checksum><stuff><cr>
```

where

length
: is the number of bytes in the record not including the % sign.

type
: is one of:

- 3) symbol record
- 6) data record
- 8) termination record

The data can come out of order, and may be discontigous.

MOS Technology Hex Object Format

Each line in the file assumes the following format:

```
;NNAAAAHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHHCCCCTT
```

where:

;
: Record Start Character

NN
: Byte Count (hex)

AAAA
: Address of first byte (hex)

HH
: Data Bytes (hex)

CCCC
: Check Sum (hex)

TT
: Line Terminator (carriage return, line feed)

All fields marked 'hex' consist of two or four ASCII hexadecimal digits (0-9, A-F). A maximum of 24 data bytes will be represented on each line.

The last line of the file will be a record with a byte count of zero (';00').

The checksum is defined as:

```
sum      =  byte_count + address_hi + address_lo +
                (sum of all data bytes)
checksum =  (sum & ffffh)
```

Go back to IDA home page
Your feedback is appreciated at mailto:ig@datarescue.com