ASCII Space Hex, Code 50

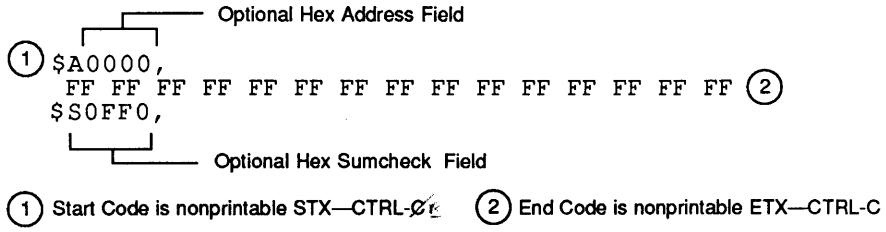
This format begins with a nonprintable STX (CTRL-B) start code and ends with a nonprintable ETX (CTRL-C) end code and can also contain an optional address field and sumcheck field. The illustration shows data bytes formatted in ASCII Space Hex. Data in this format is organized in sequential bytes separated by the execute character (space). Characters immediately preceding the execute character are interpreted as data. This format expresses 8-bit data by 2 hexadecimal characters. Line feeds, carriage returns and other characters may be included in the data stream as long as a data byte directly precedes each execute character.

Although each data byte has an address, most are implied. Data bytes are addressed sequentially unless an explicit address is included in the data stream. This address is preceded by a "$" and an "A," must contain 2 to 8 hexadecimal characters, and must be followed by a comma. The programmer skips to the new address to store the next data byte; succeeding bytes are again stored sequentially.

This format has an end code which terminates input operations. However, if a new start code follows within 16 characters of an end code, input will continue uninterrupted.

After receiving the final end code following an input operation, the programmer calculates a sumcheck of all incoming data. Optionally, a sumcheck can also be entered immediately following the end code. The sumcheck field consists of 2 to 4 hexadecimal characters, sandwiched between a "$" and a comma. The programmer compares this sumcheck with its own calculated sumcheck. If they match, the programmer will display the sumcheck, if not, a sumcheck error will be displayed. It is optional to include the sumcheck in data being input to the programmer, but a sumcheck is always included with data output from the programmer.

Output is begun by invoking an output, or upload, operation. The programmer divides the output data into 8-line blocks. Data transmission is begun with the start code, a nonprintable STX. Data blocks follow, each one prefaced by an address for the first data byte in the block. The end of transmission is signalled by the end code, a nonprintable ETX. Directly following the end code is a sumcheck of the transferred data.



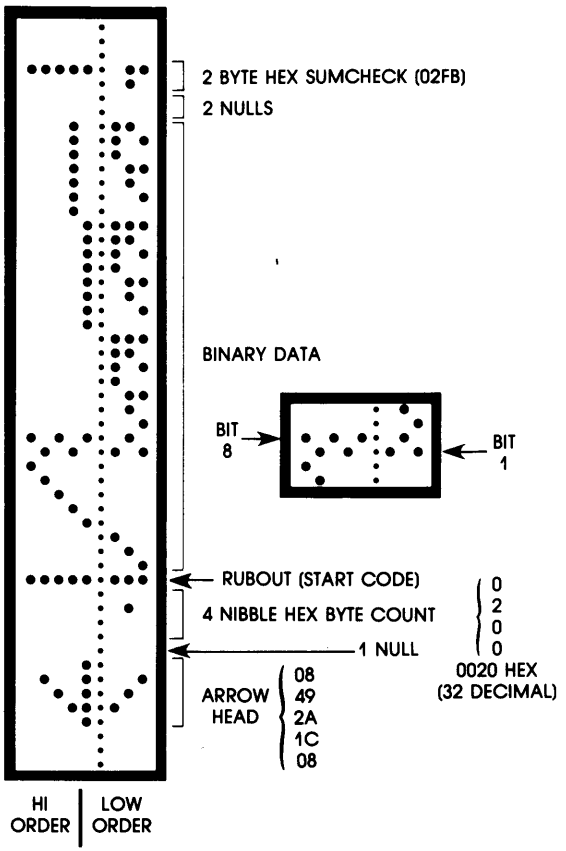
Binary Transfer, Code 10

Data transfer in the Binary format consists of a stream of 8-bit data words preceded by a byte count and followed by a sumcheck. The Binary format does not have addresses.

A paper tape generated by a programmer will contain a 5-byte, arrow-shaped header followed by a null and a 4-nibble byte count. The start code, an 8-bit rubout, follows the byte count. The end of data is signalled by 2 nulls and a 2-byte sumcheck of the data field. Refer to the illustration.

The programmer stores incoming binary data upon receipt of the start character. Data is stored in RAM starting at the first RAM address and ending at the last incoming data byte.

The standard Binary Transfer format (with the arrow header, byte count and sumcheck) can only be used for data in block sizes of 64K or less because a larger byte count could not fit into the 4-nibble byte count field.

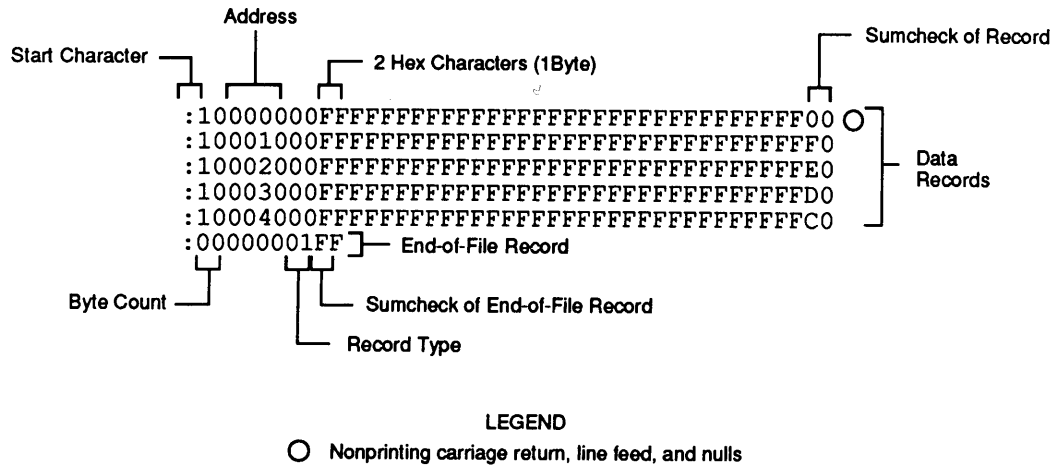


Intel Intellec 8/MDS Format, Code 83

Intel data records begin with a 9-character prefix, followed by data bytes, and end with a 2-character suffix. The illustration shows a valid Intel Intellec data file.

Each record begins with a colon, which is followed by a 2-character byte count. The 4 digits following the byte count give the address of the first data byte. The last 2 digits of the 9-character prefix are the record type. The record type is "00" for data records. Data bytes follow the 9-character prefix. Each data byte is represented by 2 hexadecimal digits ("FF" in the illustration); the number of data bytes in each record must equal the byte count. Following the data bytes of each record is the sumcheck (the binary two's complement of the preceding bytes, including the byte count, address and data bytes) expressed in hexadecimal.

The end-of-file record consists of the "colon" start character, the byte count "00," the address, the record type, "01," and the sumcheck of the record.



Intel MCS-86 Hexadecimal Object, Code 88

The Intel 16-bit Hexadecimal Object file record format has a 9-character (4-field) prefix that defines the start of record, byte count, record address, and record type and a 2-character hexadecimal sumcheck suffix. The illustration shows a valid data file of this format.

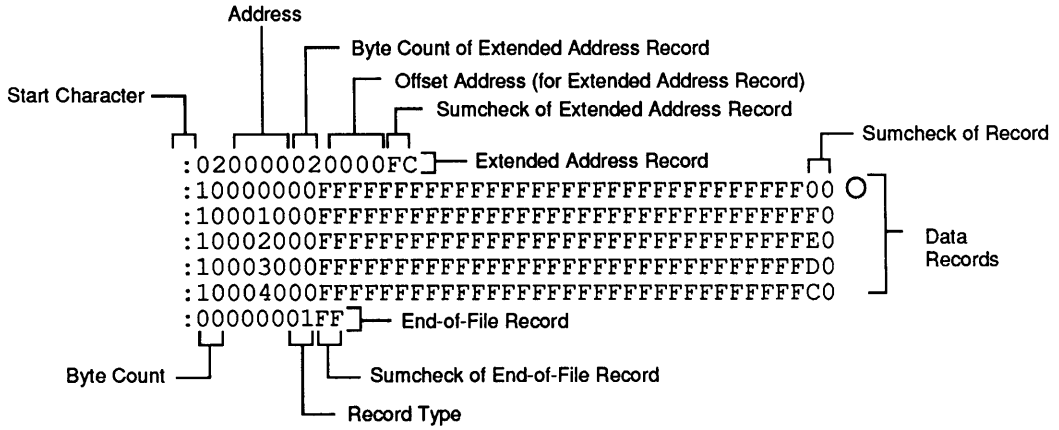
The four record types are:

00 = data record
01 = end record (signals end of file)
02 = extended address record (added to the offset to determine the absolute destination address)
03 = start record (ignored during input and not sent during output by Data I/O translator firmware)

Record type 00, the data record, begins with the colon start character. The colon is followed by the 2-character byte count (in hexadecimal notation), the 4-character address of the first data byte, and the record type, "00." This 9-character prefix is followed by the data bytes. The sumcheck follows the data bytes and is the binary two's complement of the preceding bytes in the record (including the byte count, address and data bytes) expressed in hexadecimal.

Record type 01, the end-of-file record, also begins with the colon start character. The colon is followed by the byte count, "00," the address, "0000," the record type, "01," and the sumcheck, "FF."

Record type 02, the extended address record, defines bits 4 to 19 of the record address. It can appear randomly anywhere within the data file and in any order; i.e., it can be defined such that the data bytes at high addresses are sent before the bytes at lower addresses. The example following the illustration shows how the extended address is used to determine a byte address.



LEGEND

O Nonprinting carriage return, line feed, and nulls

Example

Problem: Find the address for the first data byte for the following file.

```
: 02 0000 02 1230 BA
: 10 0045 00 55AA FF .....BC
```

Solution: Step 1: Find the record address for the byte. The first data byte is 55. Its record address is 0045 from above.

Step 2: Find the offset address. The offset address is 1230 from above.

Step 3: Shift the offset address one place left, then add it to the record address, like this:

offset address	1230	(upper 16 bits)
+ record address	0045	(lower 16 bits)
	12345	(20-bit address)

The address for the first data byte is therefore 12345.

NOTE

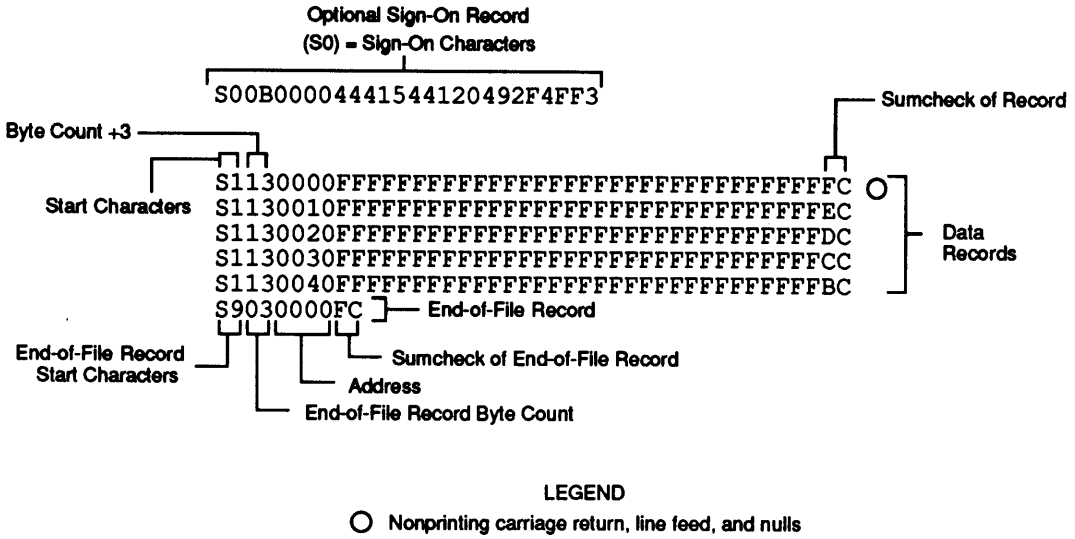
Always specify the address offset when using this format, even when the offset is zero.

Motorola Exorciser (S1) Format, Code 82

Motorola Exorciser data files may begin with an optional sign-on record, which is initiated by the start characters "S0." Valid data records start with an 8-character prefix and end with a 2-character suffix. The illustration shows a series of valid Motorola data records.

Each data record begins with the 2 start characters "S1." The third and fourth characters represent the byte count, which expresses the number of data bytes in the record (in hexadecimal) plus 3 (Two for address plus one for checksum). The last 4 characters of the prefix represent the address of the first data byte in the record . Data bytes follow the address; each data byte is represented by 2 hexadecimal characters ("FF" in the illustration). The suffix is a 2-character sumcheck, which equals the binary one's complement of the summation of the byte count, address and data bytes, expressed in hexadecimal.

The end-of-file record consists of the 2 start characters "S9," the byte count, "03," the address (in hexadecimal) and a sumcheck.

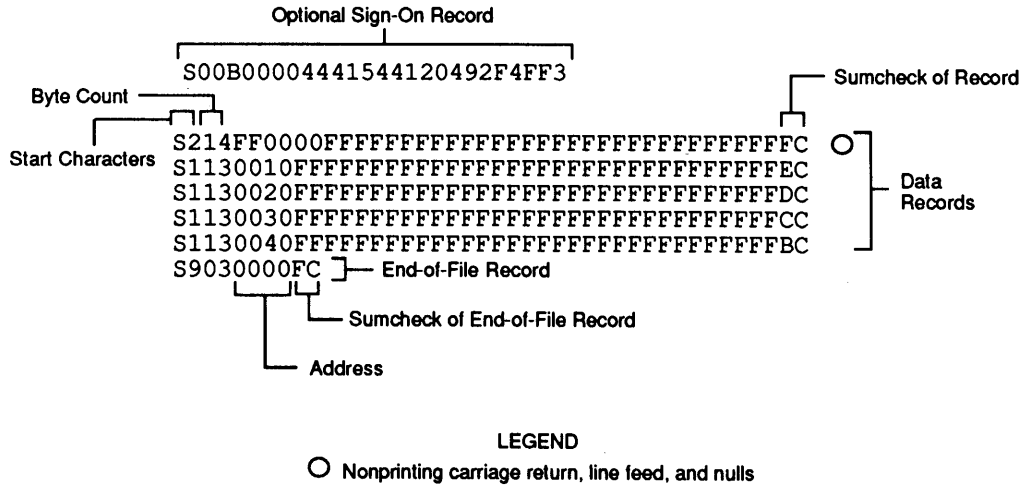


Motorola Exormax (S1 and S2) Format, Code 87

Motorola data files may begin with an optional sign-on record, initiated by the 2 start characters "S0." Data records start with an 8- or 10-character prefix and end with a 2-character suffix. The illustration shows a series of valid Motorola Exormax data records.

Each data record begins with 2 start characters, "S1" or "S2;" "S1" if the address field has 4 characters, "S2" if it has 6 characters. The third and fourth characters of the prefix represent the byte count, which expresses the number of data bytes plus 3, if the start characters were "S1," or plus 4, if the start characters were "S2." The address of the first data byte in the record is expressed by the last 4 characters of the prefix, if the address is "FFFF" or below, or the last 6 characters of the prefix, if the addresses is above "FFFF" (64K). Data bytes follow the prefix; each data byte is represented by 2 hexadecimal characters ("FF" in the illustration). The suffix is a 2-character sumcheck, which equals the binary one's complement of the summation of the preceding bytes in the record (including the byte count, address and data bytes) expressed in hexadecimal.

The end-of-file record begins with either the start characters "S8" or "S9." The start characters must be "S9" if the previous data record started with an "S1;" otherwise, either "S8" or "S9" may be used. Following the start characters are the byte count, "03," the address, "0000," and a sumcheck.

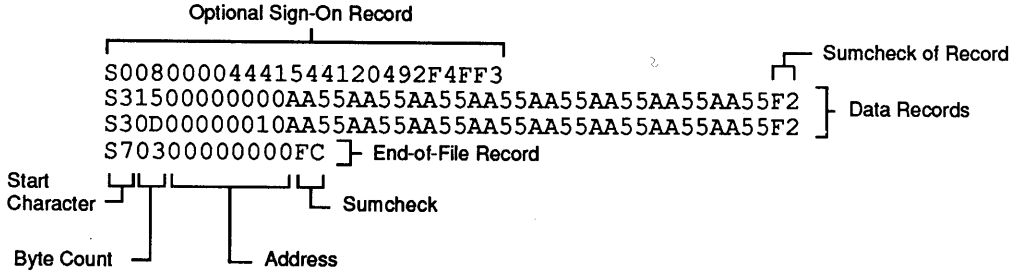


Motorola 32-Bit (S3) Format, Code 95

The Motorola 32-bit (S3) format closely resembles the Motorola Exormax format — the main difference being the addition of the "S3" and "S7" start characters. Motorola data files may begin with an optional sign-on record, initiated by the start characters "S0" (zero). "S1" and "S2" record types are allowed using this format (see the Motorola Exormax Format section for an explanation of these record types). The following paragraphs describe the "S3" format. A sample transmission of the "S3" format is shown below.

The "S3" character is used to begin a data record containing a 4-byte address. The third and fourth characters of the record represent the byte count (which expresses the number of data bytes plus 5), followed by the address and data bytes in the record. The address of the first data byte in the record is expressed by the last 8 characters of the prefix. Data bytes follow the prefix; each data byte is represented by 2 hexadecimal characters. The number of data bytes occurring must be 5 less than the byte count. The suffix is a 2-character sumcheck — the one's complement (in binary) of the preceding bytes in the record, including the byte count, address and data bytes.

The "S7" character starts a termination record for a block of "S3" records. The address field for an "S7" record may optionally contain the 4-byte instruction address that identifies where control is to be passed.



Tektronix Hexadecimal Format, Code 86

Tektronix Hexadecimal records begin with a 9-character (4-field) prefix followed by data and end with a 2-character suffix. The figure illustrates a valid Tektronix data file.

Each data record begins with a slash start character. The next 4 characters of the prefix following the start character express the address of the first data byte. The address is followed by a 2-character byte count, which represents the number of data bytes in the record, and by a 2-character sumcheck of the address and byte count. Data bytes follow the sumcheck, represented by pairs of hexadecimal characters ("FF" in the illustration). Succeeding the data bytes is their sumcheck, an 8-bit sum, modulo 256, of the 4-bit hexadecimal values of the digits making up the data bytes. The sumcheck is expressed as a 2-character hexadecimal number. All records are followed by a carriage return.

The end-of-file record consists of a start character (slash), followed by the transfer address, the byte count, "00," and the sumcheck of the transfer address and byte count.

An optional abort record contains 2 start characters (slashes), followed by an arbitrary string of ASCII characters.