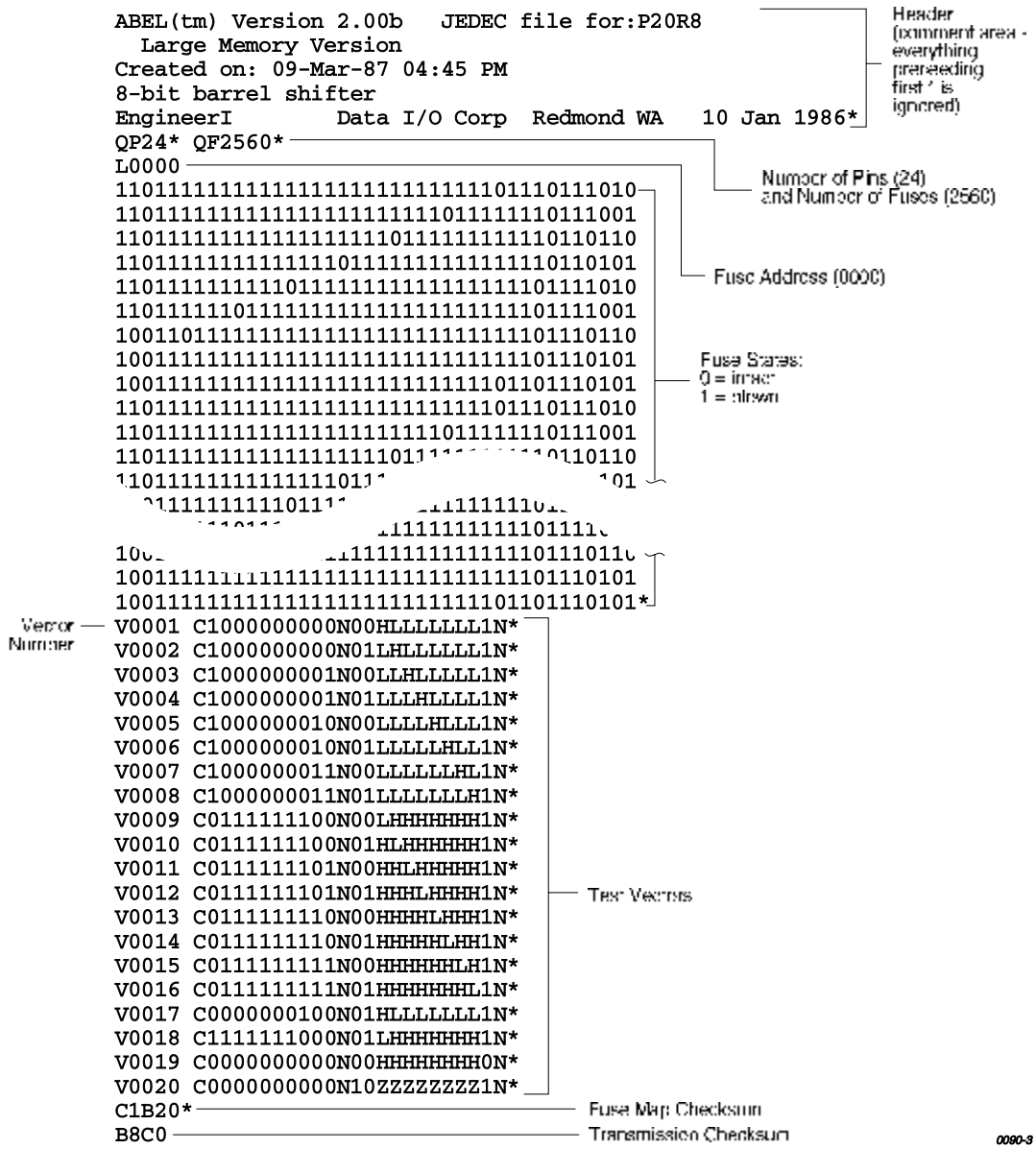
JEDEC Full Format, Code 91

The full JEDEC format consists of a start-of-text character (STX), various fields, an end-of-text character (ETX), and a transmission checksum. A sample JEDEC transmission sent in the full format is shown in Figure 1-18. Each of the fields is described on the following pages.

Figure 1-18
An Example of JEDEC Full Format



JEDEC Field Syntax

<field> ::= [<delimiter>]<field identifier>{<field character>}'*'

<field identifier>::= 'A' | 'C' | 'D' | 'F' | 'G' | 'K' | 'L' | 'N' | 'P' | 'Q' | 'R' | 'S' | 'T' | 'V' | 'X'

<reserved identifier>::= 'B' | 'E' | 'H' | 'I' | 'J' | 'M' | 'O' | 'U' | 'W' | 'Y' | 'Z'

Following the design specification field in a JEDEC transmission can be any number of information fields. Each of the JEDEC fields begins with a character that identifies what type of field it is. Fields are terminated with an asterisk character. Multiple character identifiers can be used to create sub-fields (i.e., A1, A$, or AB3). Although they are not required, you may use carriage returns (CR) and line feeds (LF) to improve readability of the data.

Field Identifiers

Field identifiers which are currently used in JEDEC transmissions are shown above on the "field identifiers" line. The "reserved identifier" line indicates characters not currently used (reserved for future use as field identifiers). JEDEC field identifiers are defined as follows:

A	Access time	N	Note field
B	*	O	*
C	Checksum field	P	Pin sequence
D	Device type	Q	Value field
E	*	R	Resulting vector field
F	Default fuse state field	S	Starting vector
G	Security fuse field	T	Test cycles
H	*	U	*
I	*	V	Test vector field
J	*	W	*
K	Fuse list field (hex format)	X	Default test condition
L	Fuse list field	Y	*
M	*	Z	*

** Reserved for future use*

Device Field (D)

Device selection by this field is not supported by the programmer. It has been replaced by the QF and QP fields and manual selection of devices.

Fuse Information Fields (L, K, F, C)

<fuse information> :: = [<default state>] <fuse list> {<fuse list>} [<fuse checksum>]

<fuse list> : = 'L' <number> <delimiter> {<binary-digit> [<delimiter>]} ' * '

<fuse list> :: = 'K' <number> <delimiter> {<hex-digit> [<delimiter>]} '*'

<default state> :: = 'F' <binary-digit> '* '

<fuse checksum> :: = 'C' <hex-digit>:4 '* '

Each fuse of a device is assigned a decimal number and has two possible states: zero, specifying a low-resistance link, or one, specifying a high resistance link. The state of each fuse in the device is given by three fields: the fuse list (L field or K field), the default state (F field), and the fuse checksum (C field).

Fuse states are explicitly defined by either the L field or the K field. The character L begins the L field and is followed by the decimal number of the first fuse for which this field defines a state. The first fuse number is followed by a list of binary values indicating the fuse states.

The information in the K field is the same as that of the L field except that the information is represented by hex characters instead of binary values. This allows more compact representation of the fusemap data. The character K begins the K field and is followed by the decimal number of the first fuse. The fuse data follow the fuse number and are represented by hex characters. Each bit of each hex character represents the state of one fuse, so each hex character represents four fuses. The most significant bit of the first hex character following the fuse number corresponds to the state of that fuse number. The next most significant bit corresponds to the state of the next fuse number, etc. The least significant bit of the first hex character corresponds to the state of the fuse at the location specified by the fuse number plus three.

The K field supports download operations only. The K field is not part of the JEDEC standard, but is supported by Data I/O for fast data transfer. The L and K fields can be any length desired, and any number of L or K fields can be specified. If the state of a fuse is specified more than once, the last state specified replaces all previous ones for that fuse. The F field defines the states of fuses that are not explicitly defined in the L or K fields. If no F field is specified, all fuse states must be defined by L or K fields.

The C field, the fuse information checksum field, is used to detect transmitting and receiving errors. The field contains a 16-bit sum (modulus 65535) computed by adding 8-bit words containing the fuse states for the entire device. The 8-bit words are formed as shown in the following figure. Unused bits in the final 8-bit word are set to zero before the checksum is calculated.

Word 00 Fuse No.	msb 7	6	5	4	3	2	1	lsb 0
Word 01 Fuse No.	msb 15	14	13	12	11	10	9	lsb 8
Word 62 Fuse No.	msb 503	-	-	-	499	498	497	lsb 496

Following is an example of full specification of the L, C, and F fields:

```
F0*L0 01010101* L0008 01010111* L1000 0101*C019E*
```

Following is an alternate way of defining the same fuse states using the K field:

```
F0*K0 55* K0008 57* K1000 5* C019E*
```

Another example, where F and C are not specified:

```
L0200     01101010101010101011
0101110101101000100100100 10*
```

The Security Fuse Field (G)

<security fuse>::='G'<binary-digit>'*'

The JEDEC G field is used to enable the security fuse of some logic devices. To enable the fuse, send a 1 in the G field:

```
G1*
```

The Note Field (N)

<note>::='N'<field characters>'*'

The note field is used in JEDEC transmission to insert notes or comments. The programmer will ignore this field; it will not be interpreted as data. An example of a note field would be:

N Test Preload*

The Value Fields (QF, QP, and QV)

JEDEC value fields define values or limits for the data file, such as number of fuses. The QF subfield defines the number of fuses in the device. All of the value fields must occur before any device programming or testing fields appear in the data file. Files with ONLY testing fields do not require the QF field, and fields with ONLY programming data do not require the QP and QV fields.

The QF subfield tells the programmer how much memory to reserve for fuse data, the number of fuses to set to the default condition, and the number of fuses to include in the fuse checksum. The QP subfield defines the number of pins or test conditions in the test vector, and the QV subfield defines the maximum number of test vectors.

The P Field

The P field remaps the device pinout and is used with the V (test vector) field. An asterisk terminates the field. The syntax of the field is as follows:

<pin list>::='P'<pin number>:N'*'

<pin number>::=<delimiter><number>

The following example shows a P field, V field, and the resulting application:

```
P 1 2 3 4 5 6 14 15 16 17 7 8 9 10 11 12 13 18 19 20 *
V0001 111000HLHHNNNNNNNNNN*
V0002 100000HHHLNNNNNNNNNN*
```

The result of applying the above P and V fields is that vector 1 will apply 111000 to pins 1 through 6, and HLHH to pins 14 through 17. Pins 7 through 13 and 18 through 20 will not be tested.

JEDEC U and E Fields

As of Version 2.5, the programmer supports the optional JEDEC U (user data) and E (electrical data) fields. The U and E fields are described below.

Note: Implementation of the JEDEC U and E fields is not part of the JEDEC-3C (JESD3-C) standard.

User Data (U Field)

The U field allows user data fuses that do not affect the logical or electrical functionality of the device to be specified in JEDEC files. For instance, the U field can be used to specify the User Data Signature fuse available in some types of PLD devices because this fuse contains information only (it has no logical or electrical functionality).

Note: To have the JEDEC U field processed correctly, you must select the device before downloading the JEDEC file.

The following guidelines apply to the U field:

- The U field must be included for devices with U fuses.
- Each U-field cell must be explicitly provided if the U field is present.
- The F (default fuse state) field does not affect U fuses.
- There can only be one U field in a JEDEC file.
- The U field fuses must be listed in the order they appear in the device.
- The U field must be listed after the L field and E field (if used), and before the V (test vector) field (if used).
- The U field is specified using binary numbers, since the full number of U-field cells is otherwise unknown.
- The number of cells specified in the U field is not included in the QF (number of fuses) field.

- The U-field cells are not included in the C (fuse checksum) field.
- The U field reads left to right to be consistent with the L (fuse list) and E fields.

The syntax for the U field is as follows:

```
<User Data Fuse List>::'U'<binary-digit(s)>'*'
```

The character U begins the U field and is followed by one binary digit for each U fuse. Each binary digit indicates one of two possible states (zero, specifying a low-resistance link, or one, specifying a high-resistance link) for each fuse.

For example,

```
QF24*
L0000
10101100000000000000000*
E10100111*
C011A*
U10110110*
```

Electrical Data (E field)

The E field allows special feature fuses that do not affect the logic function of the device to be specified in JEDEC files.

The following guidelines apply to the E field:

- The E-field cell must be explicitly provided if the E field is present.
- The F (default fuse state) field does not affect E fuses.
- There can only be one E field in a JEDEC file.
- The E field fuses must be listed in the order they appear in the device.
- The E field must be listed before the C (checksum) field. If the U field is used, the E field must come before the U (user data) field.
- The E field is specified using binary numbers, since the full number of E-field cells is otherwise unknown.
- The number of cells specified in the E field is not included in the QF (number of fuses) field.
- The E-field cells are included in the C (fuse checksum) field.
- The E field reads left to right for the purpose of checksum calculation.

The syntax for the E field is as follows:

```
<Electrical Data Fuse List>::'E'<binary digit(s)>'*'
```

The character E begins the E field and is followed by one binary digit for each E fuse. Each binary digit indicates one of two possible states (zero, specifying a low-resistance link, or one, specifying a high-resistance link) for each fuse. For example,

```
QF24*
L0000
10101100000000000000000*
E10100111*
C011A*
U10110110*
```

Test Field (V field)

<function test> :: = [<pin list>] <test vector> {<test vector>}

<pin number> :: = <delimiter> <number>

N :: = number of pins on device

<test vector> :: = 'V' <number> <delimiter> < test condition> :N '* '

<test condition> :: = <digit> 'B' | 'C' | 'D' | 'F' | 'H' | 'K' | 'L' | 'N' | 'P' | 'U' | 'X' | 'Z'

<reserved condition> :: = 'A' | 'E' | 'G' | 'I' | 'J' | 'M' | 'O' | 'Q' | 'R' | 'S' | 'T' | 'V' | 'W' | 'Y' | 'Z'

Functional test information is specified by test vectors containing test conditions for each device pin. Each test vector contains *n* test conditions, where n is the number of pins on the device. The following table lists the conditions that can be specified for device pins.

When using structured test vectors to check your logic design, do NOT use 101 or 010 transitions as tests for clock pins: use C, K, U, or D instead.

Test Conditions

0	Drive input low
1	Drive input high
2-9	Drive input to supervoltage #2-9
B	Buried register preload (not supported)
C	Drive input low, high, low
D	Drive input low, fast slew
F	Float input or output
H	Test output high
K	Drive input high, low, high
L	Verifies that the specified output pin is low
N	Power pins and outputs not tested
P	Preload registers
U	Drive input high, fast slew
X	Output not tested, input default level
Z	Test input or output for high impedance

Note: C, K, U, and D are clocking functions that allow for setup time.

The C, K, U, and D driving signals are presented after the other inputs are stable. The L, H, and Z tests are performed after all inputs have stabilized, including C, K, U, and D.

Test vectors are numbered by following the V character with a number. The vectors are applied in numerical order. If the same numbered vector is specified more than one time, the data in the last vector replace any data contained in previous vectors with that number.

The following example uses the V field to specify functional test information for a device:

```
V0001C01010101NHLLLHHLHLN*
V0002C01011111NHLLHLLLHLN*
V0003C10010111NZZZZZZZZZN*
V0004C01010100NFLHHLFFLLN*
```