SECTION 16: PROGRAM LOADING

INTRODUCTION

Program loading is performed to initialize the contents of NV RAM and to configure the microcontroller. Loading is done using a Bootstrap ROM Loader built into all members of the Secure Microcontroller family. When this Bootstrap Loader is invoked, the user's NV RAM appears as data memory to the ROM and can therefore be initialized. Once loading is complete, the Bootstrap ROM then becomes transparent. It has no effect on the user's memory map and is completely invisible. Bootstrap Loading is normally done for the initial program loading. There are no restrictions on using it for upgrades or reprogramming, though it is possible using the memory mapping features, to perform partial reloads without invoking the loader.

The Bootstrap Loader is primarily used to initialize memory, but it is capable of several other functions. It can change the memory map configuration, dump or verify the contents of memory, perform a CRC check, fill a block with a constant, manipulate the security features and the I/O ports. It can not display or edit the Scratchpad RAM (128 bytes) or SFRs since it uses these resources for its own operation. Note that the Scratchpad RAM will be erased by the loader. Therefore the loader should not be invoked while critical data is stored in this area. The MOVX RAM area will not be altered by the loader unless a user requests that this be done.

Each version of microcontroller has different loader modes and different commands. All versions are capable of being programmed via the serial port and this is the preferred method. The DS5000 series is also capable of a parallel programming technique similar to an EPROM–based 8051. The DS5001/DS5002 series supports an alternate parallel load mode that is for use by a host microprocessor using the 8042–type RPC mode. The following discussions explain the methods of invoking the Bootstrap Loader. Then the different programming modes are described. The remaining sections give a detailed description of the commands and syntax used by the Bootstrap ROM.

The DS5000 series [DS5000(T), DS2250T, and DS5000FP] and the DS5001/2 series [DS5001FP, DS2251T, DS5002FP, DS2251T] have different program loading modes available. The DS5000 series can be loaded via its serial port or in a parallel fashion like an EPROM type 87C51. The serial mode allows the DS5000 to be programmed in a fixture or while installed in the end system. The parallel method requires a super–voltage and is normally done in a fixture only. The DS5001 series has a similar serial mode with the same benefits. The parallel mode is entirely different. The DS5001FP or DS5002FP, using its RPC slave interface, can be loaded in a parallel manner by a host microprocessor. This is also an in–system technique but could be performed in a fixture. It requires no super–voltage pulses. Note that this mode is a high–speed loader and bears no resemblance to an 87C51 load mode.

Note: Dallas Semiconductor highly recommends that serial load capability be designed into the target system. This provides substantial flexibility to upgrade and troubleshoot the system. Using in–system serial loading allows a product to take full advantage of the Secure Microcontroller's features.

INVOKING THE BOOTSTRAP LOADER

The Secure Microcontroller defaults to normal operating (non–loader) mode without external hardware. If the loader mode is desired, it can be invoked at any time using the methods described later in this section. Once the Bootstrap Loader session is complete, the device will perform a hardware reset and begin operation. This will be identical to an external reset, except that the bootstrap loader, as part of normal operation, will modify various locations in scratchpad RAM. The following table shows which areas of scratchpad RAM are guaranteed destroyed, guaranteed preserved, or will be of indeterminate state after exiting the bootstrap loader.

	DS5000FP	DS5001/2FP
Guaranteed Preserved	None	70h–7Fh
Indeterminate	None	38h–6Fh
Guaranteed Destroyed	00h–7Fh	0h–37h

The guaranteed preserved locations are areas in scratchpad RAM that will not be changed by the bootstrap loader. These locations as useful for storing data such as serial numbers, which should be retained regardless of the software. Similarly, the guaranteed destroyed locations have all been overwritten during bootstrap loader execution with indeterminate data. These locations can be used to store security–sensitive data, because it will be erased by the loader before another program can read it out.

The indeterminate area contains various stacks and buffers used by the loader, and a given byte in this area may or may not be modified by the loader. As such the user should not rely on the bootstrap loader preserving any data in this area. In a like manner, because not all locations are used, the indeterminate area of scratchpad RAM should not be used for storing security–sensitive data.

The methods of invoking the loader vary between the DS5000 series and DS5001/DS5002 series, and are described below.

DS5000 Series

The DS5000 is placed in its Program Load configuration by simultaneously applying a logic 1 to the RST pin and forcing the $\overline{PSEN}$ line to a logic 0 level. Immediately following this action, the DS5000 will look for a serial ASCII carriage return (0DH) character received at 57600, 19200, 9600, 2400, 1200, or 300 bps over the serial port or a Parallel Program Load pulse. For whichever type is first detected, the DS5000 will place itself in the associated Program Load mode. If an ASCII <CR> character is detected first, then the DS5000 will place itself in the Serial Program Load mode and will ignore any Parallel Program Strobe pulses. Conversely, if a Parallel Program Strobe pulse is first detected, then the DS5000 will be placed in the Parallel Program Load mode and all incoming data on the serial port will be ignored. The selected program load mode will remain in effect until the next time power is removed from the device or when the Program Load configuration (RST=1, $\overline{PSEN}$=0) is removed.

In normal programming of the DS5000, problems with selection of the incorrect Program Load mode will not be encountered. When the DS5000 is placed in an 8751–compatible programming system, no serial data will be applied on the RXD pin for the serial port. As a result, there is almost no chance of random activity on this pin being interpreted as a <CR> character at a valid baud rate. Similarly, serial program loading will most often be performed in the end system. Consequently, there is again almost no chance of a valid Parallel Program Strobe pulse (with V_{PP} voltage at 13V) being interpreted as a signal to invoke the Parallel Program Load mode when the Serial Program Load mode is desired.

If the Program Load configuration is removed such that RST=0 and $\overline{PSEN}$=1 with power still applied at V_{CC}, the device will undergo an internal hardware reset and will begin executing code from the reset vector at 0000h in Program Memory.

DS5001/DS5002 Series

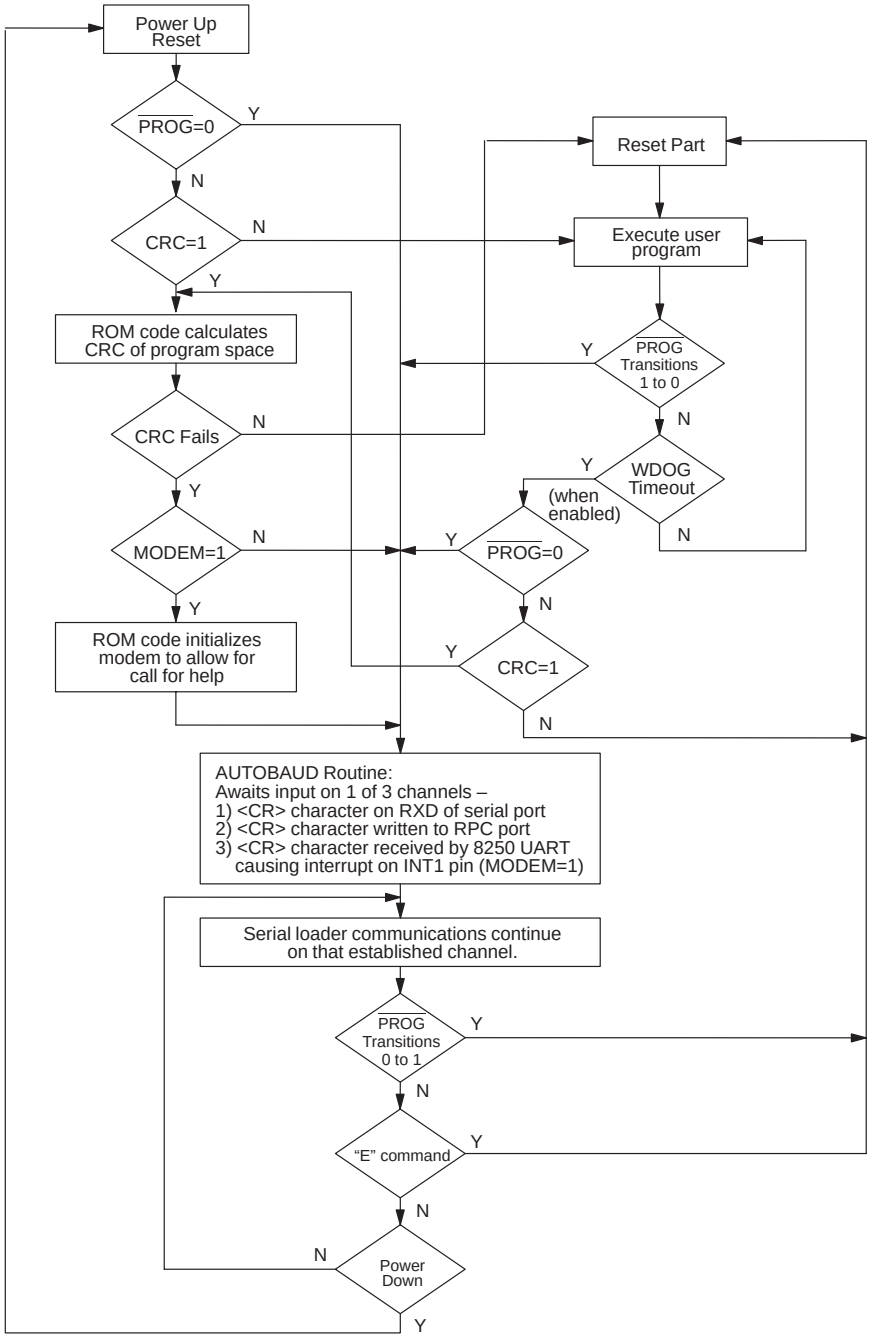
The DS5001 and DS5002 microcontrollers use an identical method to invoke the bootstrap loader. A falling edge on the $\overline{PROG}$ pin will invoke the loader with a single device pin. Note that the $\overline{PROG}$ pin must remain low for 48 oscillator clocks to be certain of recognition. Taking the $\overline{PROG}$ pin to a logic 1 will remove the loader and cause the DS5001 to perform a reset. Note also that the $\overline{PROG}$ pin must be high for as much as 48 clocks before the CPU is guaranteed to exit the loader. This constitutes the "pseudo–edge" detection of the program pin. Note also that pulling RST=1 and $\overline{PSEN}$=0 will also cause the loader to be invoked.

Once the loader mode stimulus has been detected, the DS5001 will begin looking for an ASCII carriage return (0Dh). It will look at the serial port and the RPC (8042) port. For serial reception, the loader will auto–baud at 57600, 19200, 9600, 2400, 1200 and 300 bps. For RPC mode, the 0Dh value must be written into the Data In buffer as described in the RPC section under Parallel I/O. When either of these conditions is detected, the loader will place itself in that loader mode. Activity on the other port will be ignored. This condition will remain until the loader is exited or power is cycled. When the loader stimulus is removed, the processor will perform a hardware reset and begin execution at location 0000h.

EXITING THE LOADER

The normal method of leaving the loader is to remove the stimulus that invoked it. In the DS5000 series, the RST pin, must be driven low or allowed to float and the $\overline{PSEN}$ signal should be allowed to float. The RST pin has an internal pull–down. The $\overline{PSEN}$ is an output and will drive itself. Note that both of these conditions must occur or the loader will not be exited. For the DS5001, there are several options. If the RST and $\overline{PSEN}$ option is used, the they must be removed as described above. If $\overline{PROG}$ is pulled low, it can either be returned high or the Exit "E" command can be issued. Since the loader is edge activated, this will restart the user's code even while the $\overline{PROG}$ pin is low. If power were to cycle while the $\overline{PROG}$ pin were low, the loader would be invoked on power up. The flow of these conditions is shown in Figure 16–1.

INVOKING AND EXITING THE LOADER ON THE DS5001/DS5002 SERIES Figure 16–1



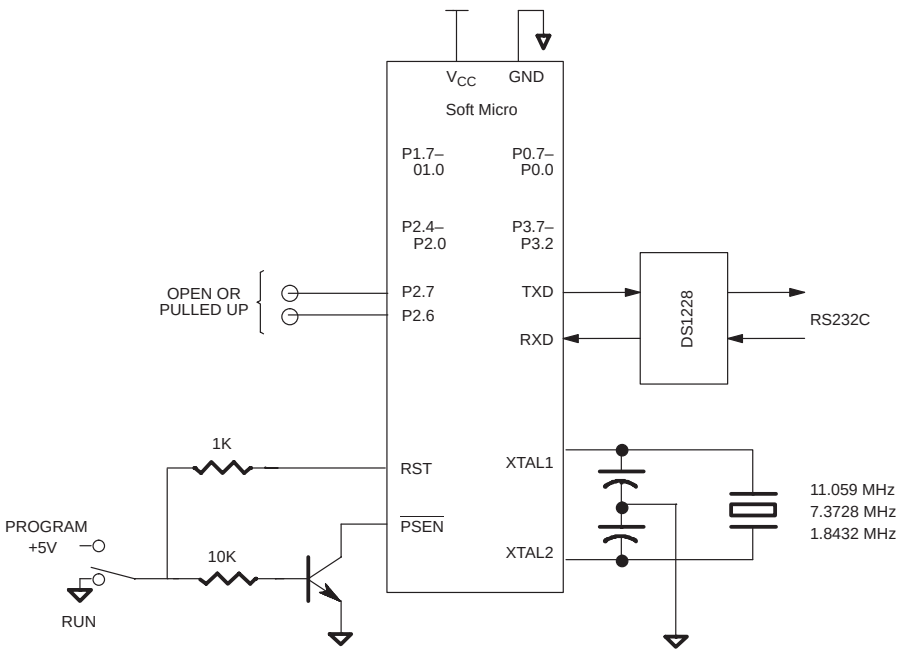
SERIAL PROGRAM LOAD MODE

The Serial Bootstrap Loader provides the easiest method of initially loading application software into the nonvolatile RAM. Communication can be performed over a standard asynchronous serial communications port using a terminal emulator program with 8–N–1 (8 data bits, no parity, 1 stop bit) protocol settings. A typical application would use a simple RS232C serial interface to program the device as part of a final production procedure.

The hardware configuration which is required for the Serial Program Load mode is illustrated in Figure 16–2. Note that as shown, it is important to have P2.7 and P2.6 either open or pulled up during serial programming of a DS5000 series device. Failure to do this results in a parallel load operation. A variety of crystals can be used to produce standard baud rates. Tables 16–1 and 16–2 show the baud rates which are supported using a variety of popular crystal frequencies. The serial loader is designed to operate across a 3–wire interface from a standard UART. The receive, transmit, and ground wires are all that are necessary to establish communication with the device.

The Serial Loader implements an easy–to–use command line interface which allows an application program in an Intel Hex representation to be loaded and read back from the device. Intel Hex is the typical format which existing 8051 cross–assemblers output.

SERIAL LOAD CONFIGURATION Figure 16–2



AUTO–BAUD RATE DETECTION

The Serial Bootstrap Loader has the capability of determining which of the six supported baud rate frequencies is being used for communication and initializing its internal hardware for communication at that frequency. When the Program Load mode is first invoked, the device will watch for activity on the serial port. If a <CR> character is received at one of the six supported baud rates, then operation in the Serial Program Load mode will be established. The loader expects to talk asynchronously at 300, 1200, 2400, 9600, 19200, or 57600 baud using eight data bits, no parity, and one stop bit in full duplex. A break signal followed by a carriage return will cause a re–determination of baud rate. Although an 11.0592 MHz crystal is standard for generating baud rates, the auto–baud rate detector allows a variety of crystals to be used. If a crystal frequency other than 11.0592 MHz is used, then the baud rate frequencies which will be recognized by the serial loader are shown in Table 16–1. Other crystals will generate non–standard band rates.

SERIAL LOADER BAUD RATES FOR DIFFERENT CRYSTAL FREQUENCIES Table 16–1

	BAUD RATE					
CRYSTAL FREQ (MHz)	**300**	**1200**	**2400**	**9600**	**19200**	**57600**
14.7456		Y	Y	Y	Y	
11.0592	Y	Y	Y	Y	Y	Y
9.21600	Y	Y	Y	Y		
7.37280	Y	Y	Y	Y		
5.52960	Y	Y	Y	Y		
1.84320	Y	Y	Y	Y		

BOOTSTRAP LOADER INITIALIZATION

When loader mode is invoked, the device will await an incoming <CR> character at a valid baud rate through either the serial port (in Serial Program Load mode) or via the parallel interface (in Parallel Program Load mode). At this point, the bootstrap loader will transmit a banner to the host to indicate that it has been invoked. The banner will appear similar to the one shown below, but will vary between specific members of the Secure Microcontroller family and between revision levels. The banner will be followed by a ">" prompt which indicates the device is ready to receive a command.

DS500X LOADER VERSION X.X COPYRIGHT (C) XXX DALLAS SEMICONDUCTOR
>

COMMAND LINE INTERFACE

The Secure Microcontroller family implements an easy-to–use command line interface which is very similar to those found in debugging environments. The Serial Bootstrap Loader responds to single character alphabetic commands which are summarized below. There are differences between versions as noted. A detailed description of each command follows.

COMMAND	FUNCTION	VERSION
C	CRC–16 of RAM	All
D	Dump Intel hex file	All
E	Exit loader	DS5001FP/DS5002FP
F	Fill RAM with a constant	All
G	Get value from ports	All
I	Include CRC	DS5001FP/DS2251T
K	Load 40–bit key	DS5000FP/DS2250T/DS5000(T)
L	Load Intel Hex file	All
N	New – invoke Freshness	DS5001 series
P	Put a value to the ports	All (DS5000 after Rev. D4)
R	Read configuration	All
T	Trace (echo) incoming data	All
U	Unlock security	All
V	Verify RAM against incoming Hex	All
W	Write register(s)	All
Z	Lock	All
^C	Reset loader	All
Xon/Xoff	Flow control of serial transmission	All

Selected commands require arguments and some commands have optional arguments. In all cases, arguments are expected to be hexadecimal numbers. In addition, an ASCII control–C character (^C) will cause the Serial Loader to terminate any function currently being executed and display the command line prompt. An incoming break character (defined as a received null character (00H) with the stop bit = 0) will cause the Serial Bootstrap Loader to be restarted and the baud rate redetermined.

COMMAND LINE SYNTAX

Single–letter ASCII characters are recognized as commands. Arguments are represented by hexadecimal numbers. A hexadecimal number is any sequence of hexadecimal characters. A hexadecimal character may be a digit, 0 through 9, or one of the letters A through F. A byte will always be the right–most two digits of a hexadecimal number. For example, the following hexadecimal numbers will result in the following bytes:

A $\rightarrow$ 0AH
AB $\rightarrow$ 0ABH
ABC $\rightarrow$ 0BCH
ABCD $\rightarrow$ 0CDH

An address will always be the right–most four digits of a hexadecimal number. For example, the following hexadecimal numbers will result in the following addresses:

A → 000AH
AB → 00ABH
ABC → 0ABCH
ABCD → 0ABCDH
ABCDE → 0BCDEH

The D and F commands allow optional addresses to be entered. The syntax [Begin–Address [End–Address]] is used to convey the following meanings:

a) No arguments: Begin–Address is set to 0 and End-Address is set to the Range.

b) One argument: Begin–Address is set to the argument and End–Address is set to the Range.

c) Two arguments: Begin–Address is set to the first argument and End–Address is set to the second argument. This second address may exceed the address value specified by the Range.

In cases b and c, the End Address may not be less than the Begin Address, either implicitly or explicitly. RAM will be addressed from 0 to 1FFF for 8K RAM and from 0 to 7FFF for 32K RAM. This maximum value is determined by the Range.

Error messages will be printed as soon as errors are detected. All messages are preceded by the two characters 'E:', and followed by a mnemonic description.

Commands will not be processed until an entire command line is entered and terminated with a <CR>. No command line may be greater than 16 bytes, which is the maximum number of characters in the K command. Since a command line is not processed until a <CR> is entered, it may be edited with the delete key which will do a destructive delete to the screen. Lines longer than 16 characters will cause an error message to be displayed and no action to be taken on the command line.

Only legal characters will be echoed back to the screen. The legal characters are: 0123456789 <:>, <space>, ABCDEFGHIJKLMNOPQRSTUVWXYZ, and <delete>. Backspace characters (<BS>) are converted to delete characters. The horizontal tab character is converted to space. Lower case alphabetic characters are converted to upper case alphabetic.

The <delete> character is executed as a <BS> <space> <BS> when possible in command mode. This will cause the character to be overprinted on a hardcopy device. The <CR> character generates a <CR> <LF> pair.

The Serial Bootstrap Loader will respond to XOFF characters by stopping transmission as soon as the character is received. A control–C or an XON will resume serial transmission. The Serial Bootstrap Loader will not transmit a XOFF character. The program is able to keep up with input as long as the receiver can keep up with its output. The receiver should be programmed to quit transmission after it sends an XOFF and transmit anything before sending an XON.

Intel Hex data is not echoed unless the Trace mode is toggled on.

COMMAND SUMMARIES

^C

Interrupt whatever is going on, clear all the buffers, put up a prompt and wait for the next command. Anything in the type–ahead buffer is removed. All output is stopped. If trace had been on before, it is cleared. If XOFF had been in effect, it is cleared.

C [begin–address [end–address]]

Return the CRC–16 (cyclic redundancy check) of the NV RAM. This computation is performed over the Range unless optional start and end addresses are given. The CRC–16 algorithm is commonly used in data communications.

D [begin–address [end–address]]

Dump memory in Intel Hex Format. An optional address range may be specified. Each record will contain up to 32 data bytes. The last line printed is the end–of–data record.

E

The serial loader is exited. This works if a negative edge on $\overline{\text{PROG}}$ was used to invoke it, or a CRC check failed.

F byte [begin–address [end–address]]
Fill memory with the value of the specified byte. An optional address range may be specified.

G
Data is read from ports 0, 1, 2 and 3 and is printed as four pairs of hexadecimal digits.

I
A CRC–16 is computed from 0 to CRC_RANGE minus 2 and the computed CRC is put into CRC_RANGE minus one and CRC_RANGE. This is used when power–on CRC checking is desired. This command will print DONE when it finishes.

K byte–1 byte–2 byte–3 byte–4 byte–5
Load the encryption key word. The five bytes are echoed before they are put into the registers.

L
Load standard Intel Hex formatted data into memory. Only record types 00 and 01 are processed. Each line of the file is treated the same way. All characters are discarded before the header character ':' is read. The rest of the record, defined by the length byte, is then processed. Control returns to the command prompt after an Intel End Record is encountered. Each byte put to memory, is read back to verify it is there. If the byte read back is different, an error is reported. All errors are reported immediately after the character is received which caused the error. The program will then read characters until a colon is found and then attempt to process the data input from the command line. Note that all bytes are put to memory as they are encountered. This means that if a bad checksum is found, an error will be reported, but all the bytes on the line will have been put to memory.

M
Toggle the status of the modem available bit (MDM in the CRC register). This will display either AVAILABLE or UNAVAILABLE. This command is only available with the DS5001FP or DS2251T.

N
The Newness command, N, will place the device into freshness mode if V_{CC} is removed following execution of the command. After typing "N" the device will prompt CONFIRM:. At this point the host system must reply FRESH, followed by a carriage return, to complete the process. If completed successfully, the message POWER DOWN TO MAINTAIN FRESHNESS will be returned. Deviation from this sequence will display the message DID NOT CONFIRM and return to the loader prompt. This command is only available with the DS5001FP, DS5002FP or modules based on these parts. This command may not be executed when talking through the modem to the serial loader.

P
DS5000:
P <P0 value> <P1 value> <P2 value> <P3 value>
Writes *values* to all ports simultaneously.
DS5001/DS5002:
P <port> <value>
Writes *value* to the requested port.

R
DS5000:
Displays the value of the MCON register.
DS5001/DS5002:
Displays the values of the MCON register, RPCTL register, MSL bit, and CRC registers in the following form: MCON:XX RPCTL:XX MSL:XX CRC:XX.
The CALIB register is no longer supported and should be ignored. CRC is not displayed on DS5002.

T
Trace by echoing the incoming Intel Hex data. This is a toggle command and will display the state after toggling. Initially the state of the toggle is OFF.

U
Clear the Security Lock. The Range is set to 32K and the partitioning is set to all program memory. Note that unlocking the Security Lock clears the encryption registers, Vector RAM, and selects $\overline{CE1}$ for the RAM. The U and Z commands are the only commands that may be executed when the chip is locked.

V
Verify current contents of memory with the Intel Hex coming in. This command operates similar to the Load command, except that it does not write to RAM; it simply compares the byte in memory to the byte in the data stream. A message is reported if an error is detected.

W byte
DS5000:
Writes *byte* to the MCON register to configure the Partition, Range, and ECE2 bits. The PAA and SL bits are

unaffected by this command.
DS5001/DS5002:

W [CRC/MCON/MSL/RPCTL] byte

Writes *byte* to the requested register. The SL bit is unaffected by this command. This command is discussed in greater detail later in this section.

Z

Set the Security Lock. Only the U and Z commands may be given after the Security Lock is set.

^C

Restarts most operations. N cannot be restarted.

Xon/Xoff

These two characters provide flow control to the serial loader. The serial loader will never issue them, but will respond to both. Xoff (control–S, DS3, 0x13) requests that character transmission stop. Xon (control–Q, DC1, 0x11) requests the resumption of transmission.

NOTES ON SELECTED DS5001FP AND DS5002FP COMMANDS

When using the Include command 'I', certain precautions are needed, The CRCBIT (bit 0) of the CRC SFR (C1h) must be set or the error message E:NOCRCB is printed. The MSL bit must not be cleared (via loader) or the error message E:NOTCOD is printed. If the PM bit (bit 1) of the MCON SFR (C6h) is one, then CRC_RANGE must be less than or equal to the program Range, or else the error message E:BADRNG is printed.

When storing an end system, it may not be desirable to lithium–back RAM or a real–time clock. The Newness command 'N' will accomplish this. When 'N' is issued, the loader will prompt with CONFIRM: after the N is entered. The user must type FRESH without any spaces or deletes and terminate it with a carriage return to put the part into freshness. The message DID NOT CONFIRM is printed if a mistake is made while entering FRESH; otherwise the message POWER DOWN TO MAINTAIN FRESHNESS is printed. The 'N' command should be the last function that is executed. After this, the system should be powered down for storage. At this time, the V_{CCO} will be pulled low, removing power from RAM or clock. The newness command may not be executed when talking through the modem to the serial loader. If it is attempted, the error message E:ILLCMD is printed.

The DS5001FP and DS5002FP provide loader commands to assist in system checkout. These are 'G" Get and 'P' Put. Get will read the values of all four I/O ports. Put is used to write a value to a port. This allows a measure of hardware control while the device is effectively in a reset state. If a port number other than 0, 1, 2, or 3 is used, the error message E:BADREG is printed. Ports 0 and 2 may not be altered when talking to the loader through the RPC interface. The error message E:BADREG is printed if it is attempted. Bits 0 and 1 of port 3 will always be written as ones when port 3 is altered (serial port).

The DS5001FP/DS5002FP write command has more options than the DS5000 version. Any of the following registers/functions can be initialized using the loader. *VAL* is written to the requested register. If an illegal register name is entered, the error message E:BADREG is printed.

CRC	This register selects the range over which a power–on CRC will be performed, and enables the process. Only bits 0, 4, 5, 6 and 7 of the CRC register may be altered. Bits 1, 2, and 3 are left alone.
MCON	Controls range, partition, and peripheral selects, All but bit 0 of the MCON register may be altered.
MSL	This bit allows the data space in a fixed memory (non–partitionable) system to be loaded using the loader software. MSL only uses the low order bit of value to change the MSL bit. In fixed partition and 128K mode, if MSL equals 0, program loads/verifies will go to data space. Upon entry, MSL will be = 1. MSL has no effect in user mode.
RPCTL	RPCTL only uses the low order bit of VAL to change bit 0 of the RPCTL register. The LSB of the RPCTL is also used to determine the Range.

Only record types 00 (data) and 01 (end) may be loaded. Other record types will cause the error message E:BADREC to be printed out, but loading will continue with the next valid record. The last two bytes of each record contain a checksum. This checksum is

compared to the computed value for the record, and if different, the error message E:BADCKS is printed out. Unfortunately, the data bytes for this record will have been put to memory already. End of Data records (01) do not check for valid checksums. After a byte is put to memory, it is read back immediately to see if it is the same. If not, the error message E:MEMVER is printed.

ERROR MESSAGES

E:ARGREQ
An argument or arguments is required for this command.

E:BADCKS
The checksum found at the end of an Intel Hex record loaded into RAM is not the same as the same as the value calculated. The last 2 bytes of each record contain a checksum. End of data records (01) do not check for valid checksums.

E:BADCMD
An invalid command letter was entered.

E:BADREC
A record type other than 00 (data) or 01 (end) was encountered while reading an Intel Hex data record. Loading will continue with the next valid record.

E:BADREG
A port number other than 0, 1, 2, or 3 was used as an argument for a P command. This message is also printed if register other than CALIB, CRC, MCON, MSL, or RPCTL was used as an argument for a W command.

E:BADRNG
The CRC_RANGE must be less than or equal to the Program Range if the PM bit is set. The CRC_RANGE must be less than or equal to the Program Range and the partition if the PM bit is cleared.

E:EXTARG
Extra data was encountered on the command line when it wasn't needed. Reenter the command.

E:ILLCMD
An illegal command was entered via the modem. Reenter the command.

E:ILLOPT
The optional parameters given were in error. If the start address is greater than the given address, either implicitly or explicitly, then an error is printed. The range bit implicitly determines the maximum range.

E:LOCKED
The requested operation cannot be performed because the device is locked.

E:MEMVER
An error was encountered while programming or verifying a byte in RAM. Reload the record or file. Repeated error messages may indicate a bad device.

E:NOCRCB
The CRC bit (bit 0) of the CRC SFR must be set when using the Include command. The Include command is not supported on the DS5002FP/DS2252T.

E:NOTCOD
The Include command was entered while the MSL bit was cleared. The MSL bit must be set (via loader) when using the Include command.

E:NOTHEX
A non–hexadecimal character was found when expected.

E:VERIFY
The byte which was sent did not verify with the corresponding one in RAM.

INTEL HEX FILE FORMAT

8051–compatible assemblers produce an absolute output file in Intel Hex format. These files are composed of a series of records. Records in an Intel Hex file have the following format:

<Header><Hex Information><Record Terminator>

The specific record elements are detailed as follows:

: II aaaa tt dddddd ... dd xx
Where:
: Indicates a record beginning
II Indicates the record length
aaaa Indicates the 16 bit load address
tt Indicates the record type
dd Indicates hex data
xx Indicates the checksum = (2's complement (II+aa+a+tt+dd+dd+...dd)

Record type 00 indicates a data record and type 01 indicates an end record. An end record will appear as :00 00000 01 FF. These are the only valid record types for a NIL hex file. Spaces are provided for clarity.

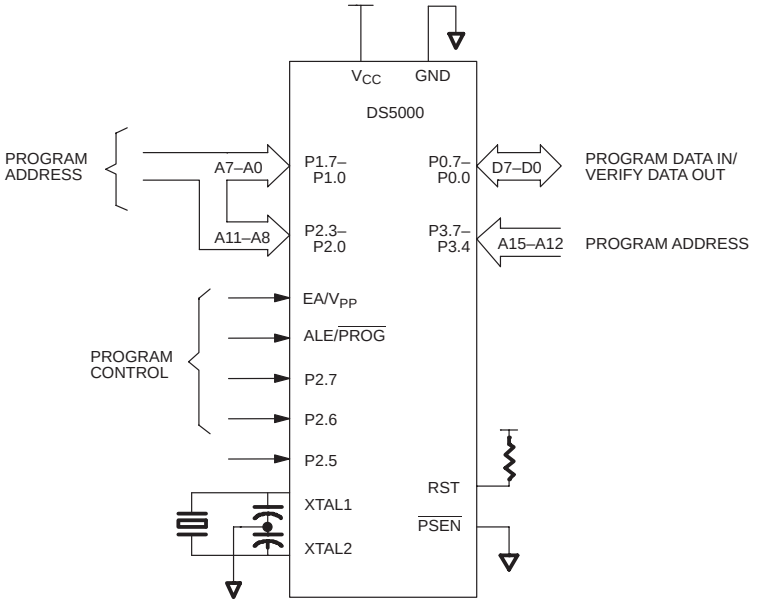
The following is a short Intel hex file. The data bytes begin at 01 and count up to 2F. Notice the records length, beginning address, and record type at the start of each line and the checksum at the end.

```
:20000000010203040506070809 0A0B0C0D0E0F10111213141516171819 1A1B1C1D1E1F20D0
:0F00200021222324252627282 92A2B2C2D2E2F79
:00000001FF
```

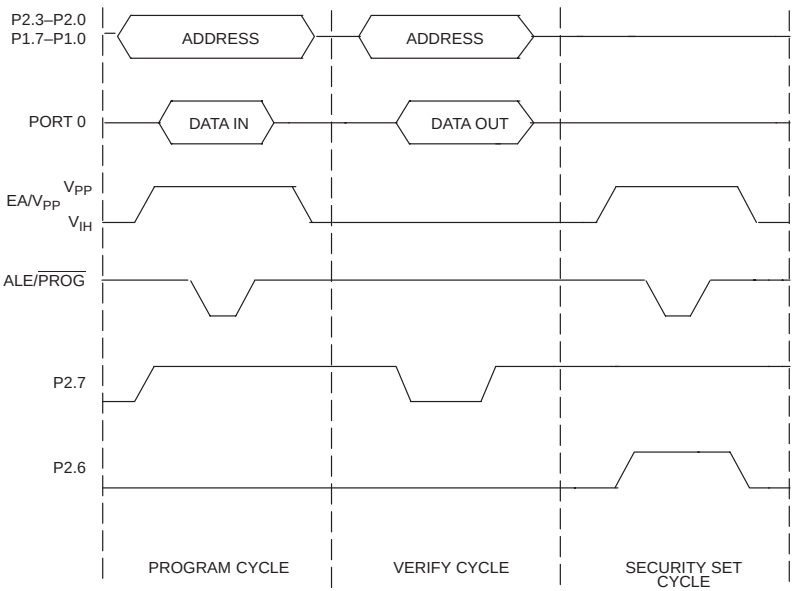
PARALLEL PROGRAM LOAD OPERATION

The DS5000 Parallel Program Load mode is compatible with the Program mode of the 87C51. The hardware configuration used for this mode of operation is shown in Figure 16–3. Dallas Semiconductor recommends the use of the serial Program Load Mode over the Parallel Program Load Mode because of the ease of implementation and simpler hardware interface.

PARALLEL PROGRAM LOAD CONFIGURATION Figure 16–3



PARALLEL PROGRAM LOAD CYCLES Figure 16–4



PARALLEL PROGRAM LOAD MODE

Table 16–3 summarizes the selection of the available Parallel Program Load cycles. Figure 16–4 illustrates the timing associated with these cycles.

8751–COMPATIBLE PROGRAM LOAD CYCLES Table 16–3

MODE	RST	$\overline{\text{PSEN}}$	$\overline{\text{PROG}}$	$\overline{\text{EA}}$	P2.7	P2.6	P2.5
Program	1	0	0	V_{PP}	1	0	X
Security Set	1	0	0	V_{PP}	1	1	X
Verify	1	X	X	1	0	0	X
Prog Expanded	1	0	0	V_{PP}	0	1	0
Verify Expanded	1	0	1	1	0	1	0
Prog MCON or Key	1	0	0	V_{PP}	0	1	1
Verify MCON	1	0	1	1	0	1	1

The Program cycle is used to load a byte of data into a register or memory location within the DS5000. The Verify cycle is used to read this byte back for comparison with the originally loaded value to verify proper loading. The Security Set cycle may be used to enable the Software Security feature of the DS5000. One may also enter bytes for the MCON register or the Encryption Key using the Program MCON cycle. When using this cycle, the absolute register address must be presented at Port 1 and 2 as is the normal Program cycle (Port 2 should be 00H). The MCON contents can be likewise verified using the Verify MCON cycle.

When the DS5000 first detects a Parallel Program Strobe pulse or a Security Set Strobe pulse while in the Program Load mode following a Power–On Reset, the internal hardware of the DS5000 is initialized so that an existing 4K byte 8751 program can be programmed into a DS5000 with little or no modification. This initialization automatically sets the Range Address for 8K bytes and maps the lower 4K byte bank of Embedded RAM as Program Memory. The top 4K bytes of Embedded RAM are mapped as Data Memory. In order to program code (and thereby use the DS5000–enhanced capability), the Program/Verify Expanded cycles can be used. Up to 32K bytes of program code can be entered and verified. Note that the expanded 32K byte Program/Verify cycles take much longer than the standard cycles.

A typical parallel loading session would follow this procedure. First, set the contents of the MCON register with the correct range and partition (if using expanded programming). Next, the Encryption Key can be loaded if desired. Then, program the DS5000 using either standard or expanded program cycles and verify. Last, turn on the security lock using a Security Set cycle.

The Security Set strobe pulse from an 8751–compatible programming system can be used to enable the Software Security feature of the DS5000. To explain this operation on the DS5000, it is useful to review how this function works with the 8751. The Security Set Strobe pulse is used to program the EPROM Security Lock bit on an 8751. The programmed bit disables the on–chip EPROM memory from being read back during a Verify cycle. The bit can only be erased by UV light when the rest of the program is erased.

With the DS5000, the Security Set Strobe pulse serves a similar function for its NV RAM–based Security Lock which when set disables the NV RAM from beginning read either through a Verify cycle in the Parallel Program mode or back through the serial port in the Serial Port mode. When a Security Set Strobe pulse is received by the DS5000, the current state of the Security Lock bit is checked. If it is currently a 0, it will be set to 1. The Security Lock can be cleared by clearing the LSB of the MCON register.

PARALLEL PROGRAMMING CONCERNS

Dallas Semiconductor highly recommends using the serial load mode for programming the DS5000. It has proven highly reliable and easy to use. In the event that parallel programming is still desirable to some users, several incompatibilities have been discovered in conventional device programmers. The following is a summary of these incompatibilities:

1. The DS5000 is a fully CMOS device, and was actually designed to be pin–compatible with the 80C51/87C51 as opposed to the 8051/8751. As a result there is a subtle difference between these two devices. This has to do with the oscillator input pins, XTAL1 and XTAL2. On the CMOS devices, XTAL1 is the pin which is driven in the external drive configuration. On NMOS devices, XTAL2 is driven for the external clock configuration. This difference has no effect when a crystal is tied to the pins for an external time base. However, many programming systems use the external drive configuration in order to maintain the ability to program multiple types of devices in a single 40–pin socket. For this reason, the DS5000 will not operate correctly in a 8751–compatible socket which uses the external clock mode.
2. The 87C51 data sheet specifies a "fast" programming timing algorithm for programming the locations in its on–chip EPROM memory. This algorithm is identical to the 8751 Program mode specification except for the number and duration of ALE low pulses during a "Program Byte" state. There are 25 pulses specified, each with a low time of 90 to 110 µs following by a minimum high time of 10 µs. Since the Parallel Load mode is partially implemented using internal ROM firmware, the 87C51 fast programming algorithm is incompatible with the DS5000. Programming systems which implement this algorithm will not correctly program a DS5000.
3. Also since the Parallel Program mode is partially firmware based, a minimum recovery time is required between back–to–back Program Byte strobes and between a Program Byte strobe followed by a Verify strobe.
4. Many programming systems apply V_{CC} voltage during programming and remove it when programming is completed. This operation is compatible with the DS5000 if the Power–On Reset time spec (t_{POR}) is met before programming begins. Since there is no similar specification on the 8751 or on the 87C51, some programming systems may not meet the DS5000's requirements and Program strobe pulses may not be recognized by the DS5000.
5. The DS5000 is compatible with either the 21V V_{PP} of the 8751 or the 12V V_{PP} of the 87C51. However, some programming systems sample the current that is drawn during programming on the V_{CC} pin and/or on the V_{PP} pin. An 8751 is specified to draw a maximum of 30 mA of I_{PP} current during programming, while an 87C51 is specified for a maximum of 50 mA. A DS5000 will draw a maximum of only 15 mA of I_{PP} current during programming. As a result, these programming systems may erroneously report that the device is incorrectly installed in the socket.

Because of the limitations cited above, Dallas Semiconductor recommends that the Serial Bootstrap Loader be used for initial program loading of the DS5000.

RPC PROGRAM MODE OPERATION

The DS5001FP and DS5002FP series offer high–speed programming mode with many of the benefits of the Serial Loader. Like the Serial mode, it is primarily intended as an in–system technique but can be used in a fixture. This mode uses the RPC (8042) slave interface to perform a high speed parallel load. When the $\overline{PROG}$ pin is pulled to a logic 0, the Bootstrap ROM will begin looking for an ASCII carriage return. This can come in via the serial port or the RPC port. The RPC port is accessed as shown in the section on Parallel I/O. If the RPC buffer is written with a 0Dh, this will cause the loader to respond with its banner and prompt using this same interface. An external microprocessor is assumed to have written and read these values. The RPC loader implements the same command interface and syntax as the Serial Loader. The only difference is the speed at which data can be written, and the lack of a baud rate consideration. As bytes are written into the buffer, they will be acted upon. Handshaking will be used as described in the Parallel I/O section.

The RPC mode requires no super–voltage pulses. The $\overline{CS}$, $\overline{RD}$, and $\overline{WR}$ strobes control the transfer of data between the DS5001 and the host. This protocol makes the DS5001 ideal for PC based applications, but any host processor can perform the loading.