binload User's Manual

A Serial Loader for the Motorola 68hc9s12c32 MCU

Introduction

binload is a menu-based Win32 application that supports Motorola's Serial Monitor on a 68hc9s12c32 microcontroller (MCU). binload allows you to read and modify memory, monitor the values of specific memory locations, and download S19 records to the target MCU.

binload can be invoked from a batch file, and many of binload's menu commands can be triggered by command-line options. For example, it is possible to reset the target, erase all flash memory, download an S19 file, and exit binload, all from a single command. This makes binload useful for rapid development cycles or even for limited production use.

Installation

binload is a single Win32 executable file named binload.exe. Installation consists of copying the file to a directory on your execution path or, if appropriate, in your working directory. binload does not modify your registry, it does not use any DLLs other than those common to a basic Win32 app, and it does not have a setup file. If you want to uninstall binload, simply delete the executable file.

Invocation

binload can be invoked in several ways. From the Windows desktop, you can navigate to the program's icon and double-click on it. You will be presented with a DOS-type window, where you will see the binload sign-on message and a command prompt.

You can also invoke binload from a shortcut. Simply create a standard Windows desktop shortcut, pointing to the binload.exe file. Double-clicking the shortcut launches the binload application. You can also set up binload's command-line options in the shortcut, which allows you to trigger several complicated binload operations with one double-click.

You can also embed a call to binload inside a larger batch file. binload supports many command-line options that allow you to perform the most important loading operations without have to use the menu. You can even have binload perform a series of operations and then quit, without ever showing the command menu or command prompt. When binload exits, it reports an error code, allowing your batch file to test for successful completion.

Command-line options

binload supports several command-line options that essentially mimic operations available from binload's command prompt. For example, you can use the /r command-line option to cause binload to reset the target MCU.

Note that any command-line options found are executed IN ORDER. Thus, with careful arrangement of the command-line options, you can perform a complicated series of binload operations without ever entering a binload command, or even displaying the binload window.

The following paragraphs describe the available binload command-line options.

/p define COM port

The /p option allows you to select the COM port that binload uses to communicate with the target MCU. By default, binload uses COM2. To select a different COM port, use a /p option of the form:

/p n

where n is the number of the port to use; 1 selects COM1, etc.

/h display list of interactive commands

The /h option displays a list of interactive commands. These are the menu-based commands available to the user once the binload window appears. Note that these commands are not the command-line options, though many of them are analogous to command-line options. To display the list of interactive commands, use the option:

/h

/r reset target MCU

The /r option causes binload to issue a reset command to the target MCU. After the reset completes, the target will be in its default reset state. To reset the target, use the option:

/r

/e erase target flash memory

The /e option causes binload to issue an erase-all command to the target MCU. After the erase completes, the target's flash memory will be erased to $ff, excepting the serial monitor in the target's upper flash block. To erase the target's flash memory, use the option:

/e

/l load a S19 file into target memory

The /l option causes binload to open a selected file and write all S19 records in that file to the target MCU. The serial monitor in the MCU will automatically select the proper writing procedure for the memory involved.

Note that attempts to load a file into flash memory without first erasing that flash will likely fail. In general, you should first use the /e option above to erase flash, then use the /l option to program the flash. To load the file foo.s19 into the target's flash memory, use the option:

/l foo.s19

Note that you must specify a full path to the S19 file you want to load; this includes any extension, such as .s19.

/q quit the binload application

The /q options quits the binload application immediately. If this option is present on the command-line, binload quits without further processing of command-line options. It also quits without presenting its menu window, so you will not be able to enter any commands to binload manually.

This option is most useful for rapid developement cycles. You can create a shortcut or batch file that invokes binload with a full sequence of commands to erase and program the target MCU. If the last command-line option present is /q, binload exits immediately after programming your target MCU. To quit the binload application from the command-line, use the option:

/q

Menu commands

binload provides several commands, available interactively from its command window. To start binload for interactive operation, simply enter the command:

binload

from a Windows shortcut, batch file, or command-line prompt. If you include command-line options on your command, and provided you don't include the /q option and no options fail, you will also be presented with the command window.

The command window (or menu window) presents a simple command prompt ("> ") and accepts commands. These commands are generally one- or two-letter commands, followed by zero or more arguments. Any arguments on a command line must be separated by at least one space or tab character.

The following paragraphs describe the menu commands available at the command prompt. Note that some of these commands are similar to command-line options described above.

help command

Entering a command of:

help

causes binload to display a list of interactive commands that are available at the command prompt.

quit command

Entering a command of:

quit

causes binload to quit immediately and returns control to the Windows desktop.

reset command

Entering a command of:

reset

causes binload to issue a reset command to the target MCU. binload will report the status of the reset command. If the status is "No error" then the reset completed properly. If any other status is reported, you may need to reset the target manually and retry the reset command.

info command

Entering a command of:

info

causes binload to issue a device info request command to the target MCU and to report the returned values. This command reports the target's Device Code and two-byte HCS12 device ID register.

rb command (Read Byte)

Entering a command of:

rb addr

causes binload to display the byte at the address given as addr. For example, to read the byte at address $4200, use:

rb $4200

rbl command (Read Byte, Looping)

Entering a command of:

rbl addr

causes binload to enter a loop that displays the byte at the address given as addr, delays for a short time, then displays the byte again. You can interrupt this loop by pressing a key on the keyboard. For example, to start a looping read of address $3456, use:

rbl $3456

This command is most useful for real-time monitoring of I/O ports; you can start a looping read on a port, then manipulate the input pins of that port and watch the port value change.

rw command (Read Word)

Entering a command of:

rw addr

causes binload to display the word (16-bit) value at the address given as addr. For example, to display the word at address $8900, use:

rw $8900

rwl command (Read Word, Looping)

Entering a command of:

rwl addr

causes binload to enter a loop that displays the word (16-bit) value at the address given as addr, delays for a short time, then displays the word again. You can interrupt this loop by pressing a key on the keyboard. For example, to start a looping read of address $3456, use:

rwl $3456

This command is most useful for real-time monitoring of I/O ports; you can start a looping read on a 16-bit port or register, then manipulate the input pins of that port and watch the port value change.

rmem command (Read Memory)

Entering a command of:

rmem addr

causes binload to display the contents of a block of memory, starting at the address given as addr. The block of memory to display is fixed at 128 bytes. To display the 128-byte block of memory starting at 0, use:

rmem 0 or
rmem $0

wb command (Write Byte)

Entering a command of:

wb addr b

causes binload to write the byte given as b to the address given as addr. The contents of addr are then read back to verify that the value was properly written. If the write was not successful, binload displays the address, desired byte, and actual byte read. To write the byte $45 to address $4000, use:

wb $4000 $45

ww command (Write Word)

Entering a command of:

ww addr w

causes binload to write the word (16-bit value) given as w to the address given as addr. The contents of addr are then read back to verify that the value was properly written. If the write was not successful, binload displays the address, desired word, and actual word read. To write the word $55aa to address $c600, use:

ww $c600 $55aa

erase command (Erase Memory)

Entering a command of:

erase type

causes binload to erase a specific section of memory, depending on the type specified. Options for the type field are **all** to erase all flash memory, **page** to erase the page of memory currently specified by the target's PPAGE register, or **ee** to erase the device's EEPROM. To erase all of flash memory on the target device, use:

erase all

This operation can take a second or more to complete, depending on the type and amount of memory involved.

Note that the 'c32 device does not support EEPROM. Attempts to erase EEPROM on a 'c32 will generate an error response.

load (Load S19 File)

Entering a command of:

load filepath

causes binload to open the specified file and write all S1 records in that file to the target MCU. S0 and S9 records are read but ignored. To load the file foo.s19, in directory c:\develop, to the target MCU, use:

load c:\develop\foo.s19

Note that the load command assumes that the target device is a ‘c32; the load command is not guaranteed to work with any other HCS12 variant. This is because the load command must compensate for certain features of the ‘c32 device and the Serial Monitor built into the target device.

The ‘c32 has two fixed blocks of 16K flash; one from $4000 to $7fff and the other from $c000 to $fff. The 16K block in between, from $8000 to $bfff, is called the paged block because its contents are controlled by the value written to the paging register, PPAGE.

On the ‘c32, only the lowest bit (bit 0) of PPAGE controls the contents of the paged block of flash. If bit 0 of PPAGE is cleared (is 0), the paged block contains an exact copy of the low fixed block, starting at $4000. If bit 0 of PPAGE is set (is 1), the paged block contains an exact copy of the high fixed block, starting at $c000.

Immediately out of reset, PPAGE defaults to 0, causing the paged block to duplicate the low fixed block.

This means that, out of reset, the ‘c32 appears to have 32K of contiguous flash, starting at $8000 and going up to $ffff. In reality, the lower half of this flash is actually a duplicate of the low fixed block at $4000, but code written for $8000 and stored in the paged block will execute properly.

Unfortunately, early version of the Serial Monitor could not write to the flash in the paged block of a ‘c32. In order to correct for this, the load command automatically checks the target address of each byte it is about to write. If the byte would be written to an address in the paged block, the load command instead writes that byte to the corresponding address in the low fixed block. For example, instead of writing $a9 to address $8090, the load command would write $a9 to address $4090.

This remapping of downloaded bytes works because, following reset, the contents of the low fixed block are duplicated in the paged block, essentially restoring the contents of the flash at $8000.

The bottom line is that you can develop your ‘c32 applications as if you had 32K of contiguous flash, from $8000 to $ffff.

Miscellaneous information

The following sections contain miscellaneous information about binload and its commands.

Numerical notation

If you need to supply a number to binload, such as an argument, you can use a preceding dollar-sign (‘$’) to indicate hexadecimal notation, and binload will interpret the number per ANSI-C sscanf() %x rules. If no dollar-sign is present, binload will interpret the number per ANSI-C sscanf() %d rules.