eZ80L92 Development Kit

External Flash Loader

PUG001305-0803 Product User Guide

Introduction

The Flash Loader utility integrated within ZiLOG Developer Studio (ZDSII) provides the ability to program Flash memory inside an eZ80® processor on a target eZ80® Development Module via the eZ80® Development Platform. The ZDSII Integrated Development Environment (IDE) provides an intuitive interface to this feature that guides the user through this task using a file developed for the ZDSII IDE and language tools developed for the eZ80L92 device.

The target-based external Flash Loader utility allows the user to reprogram Flash memory inside an eZ80® processor on a stand-alone target system. In this system, neither the ZDSII IDE nor ZPAKII is used. This target-based, external use model is primarily designed for field upgrades. The Flash Loader software runs on the eZ80® device and interacts with the user across a serial port to download a `.hex` file for burning into internal eZ80® Flash memory.

This document discusses the target-based, *external* Flash Loader utility. For a discussion of the ZDSII IDE Flash Loader utility, please see the ZiLOG Developer Studio—eZ80 User Manual (UM0123) on www.zilog.com.

Features

- Hardware support for the eZ80® Development Platform
- Flash support to program the write-protected boot block using ZDS II
- Flash support to program user code via the onboard serial console port
- Ability to easily modify the MAC address in Flash memory using ZDS II

Requirements

Software

- ZiLOG Developer Studio II—Integrated Development Environment includes:
  - Editor
  - C-Compiler
  - Assembler
  - Librarian
  - Debugger

- Flash Loader Utility
- Flash Boot Block Utility
- A terminal emulation program, such as HyperTerminal

Hardware

- eZ80® Development Platform
- One RS232 cable @ 57.6kbps, 8-N-2
- ZPAK II, including:
  - ZDI Target Interface Module (TIM)
  - One 40-pin ribbon cable to connect the TIM to ZPAK II
- One Ethernet cable
- One Ethernet hub with power supply
- One 5V 1000mA power supply for ZPAK II
- One 9V 1200mA power supply for the eZ80® Development Platform

For demonstration purposes, a MAC address is provided on the bottom of the eZ80® Development Platform.

Caution: Always use a grounding strap to prevent damage resulting from electrostatic discharge (ESD).

Settings

Terminal

The eZ80L92 Development Kit requires specific terminal settings for using the Flash Loader utility. Configure these settings with the following brief instruction.

1. Connect the Host PC to the console port (P2) of the eZ80® Development Platform.
2. Set the COM properties to:
   - 57600 baud
   - 8 bits
   - No parity
   - 2 stop bits

Jumpers

The jumper and its settings are listed below.

- J7—Flash Write Enable ($\overline{\text{FlashWE}}$)
  - Installed—the first 16KB boot block is vulnerable to modification
  - Removed—the first 16KB boot is write-protected

If the Write Enable jumper (J7) is not installed, the user cannot program Flash.

Memory Map

The Flash Loader utility resides in the boot block section of Flash memory, in the address range 0000h–3FFFh. This 16KB block is vulnerable to change when jumper J7 is installed, and write-protected when jumper J7 is removed. The primary interrupt vector table and the MAC address reside within this protected block. The next two 8KB parameter blocks, in the address range 4000h–7FFFh, are reserved.

In the block starting at address 8000h, 200h bytes are reserved for the secondary interrupt vector table. The requirement for primary and secondary interrupt vector tables is explained in the Interrupt Vectors and Jump Table Arrangement section on page 4. User code resides in Flash memory beginning at address 8200h, extending until the end of Flash. On the next page, Figure 1 illustrates the memory map for the eZ80® Development Platform before booting into the Flash Loader.

During reset, the eZ80® Development Platform maps Flash memory at address 000000h and places SRAM beyond the first 64KB of memory space. Because this default memory configuration is not optimal for handling interrupts on the eZ80L92 device, the Flash Loader changes the chip select configuration so that SRAM starts at address 000000h and Flash memory starts at address 800000h. These SRAM and Flash memory locations are simply swapped after the user boots into the Flash Loader utility.

Flash and SRAM memories on the eZ80® Development Platform use the following address ranges while using the Flash Loader:

Flash Memory: 800000h–FFFFFFh (8MB, assuming all 8MB is physically present on the eZ80® Development Platform).

SRAM: 000000h–07FFFFh (512KB, limited to available SRAM on the eZ80® Development Platform, Rev B).

Note: The change in the memory locations and ranges does not require the user to build the application to start at address 808200h instead of 8200h. Flash Loader handles this issue internally.

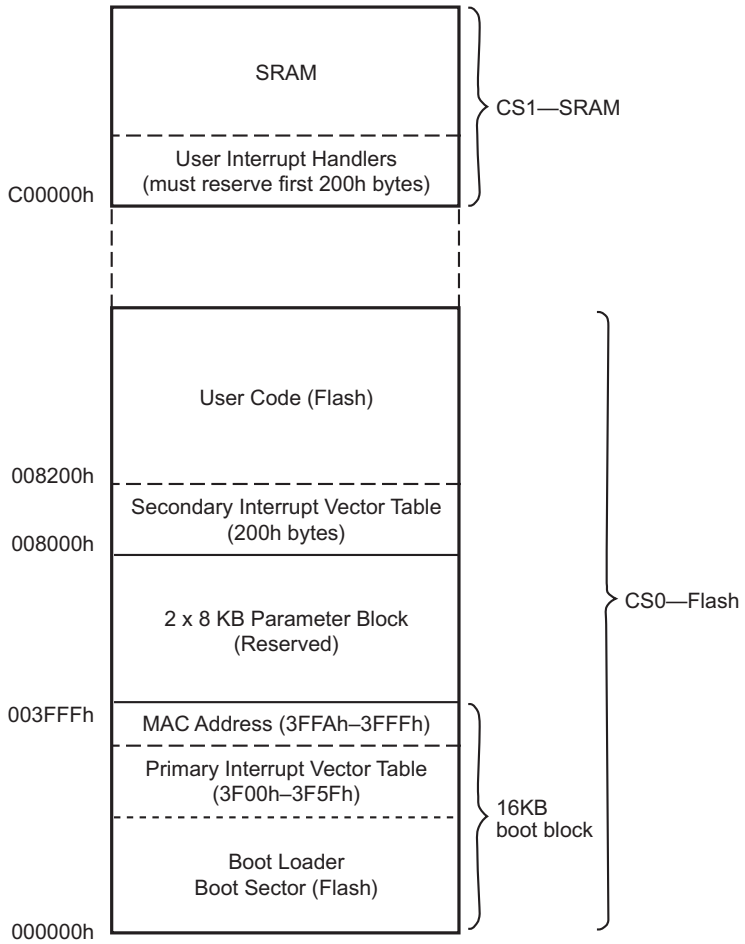

Figure 1. eZ80® Development Platform Memory Map

Interrupt Vectors and Jump Table Arrangement

Figure 1 shows three interrupt vector tables—primary, secondary, and user. This arrangement is necessary for the following reasons:

1. The eZ80® Development Platform does not contain SRAM in the first 64KB of memory.
2. Interrupt register I on the eZ80® Development Platform is only 8 bits; therefore, the interrupt controller cannot jump to user handlers located beyond 64KB.

Additionally, these three tables allow the user to run programs in SRAM before burning Flash memory.

When an interrupt occurs, the control first jumps to the primary interrupt vector table at 3F00h. In turn, a series of jumps can be made to the secondary interrupt vector table starting at 8000h, and to the user vector table located at C00000h, as shown in Figure 2.

For the jump sequence to the interrupt vectors to work, the user application must:

1. Set up the interrupt vector table, starting at C00000h, to manage all of the interrupts.
2. Set the I register value to 3Fh.

Note: The I register value is set to 3Fh by the Boot Loader utility prior to jumping to the user application at 8200h.

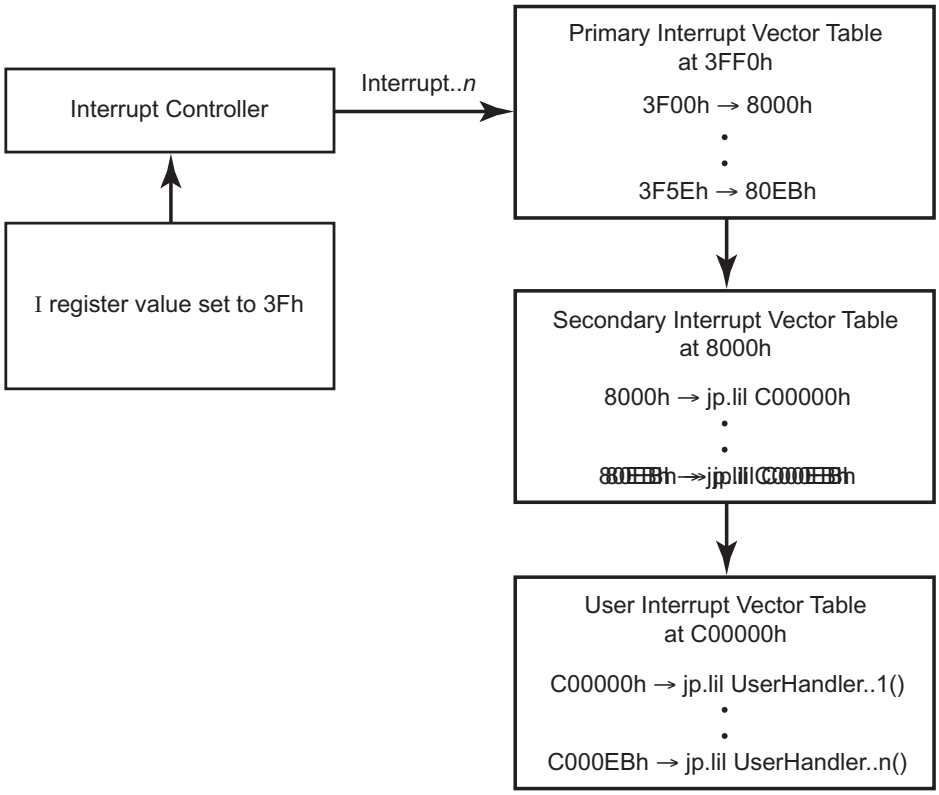

Figure 2. Interrupt-initiated interactions between Vector Tables

Alternatively, the Flash Loader utility allows the user to override the above scheme. The user can set the starting address of the interrupt vector table at 8000h. To develop an application, the user can generate a table containing the required ISR handlers for which the Flash Loader does not flash a secondary interrupt vector table at address 8000h. This alternative is demonstrated in the *2Tables* demonstration application, which can be found in the filepath FlashLoader120/Demo/L92/2Tables. Please see the Demonstration Applications section on page 13 for more information.

Table 1 lists each interrupt vector and its corresponding three jump locations.

Table 1. Interrupt Vectors and Corresponding Jump Locations

Interrupt Vector	Primary Table Location (Hexadecimal)	Secondary Table Location (Hexadecimal)	User Table Location (Hexadecimal)
00	3F00	8000	C00000
02	3F02	8005	C00005
04	3F04	800A	C0000A
06	3F06	800F	C0000F
08	3F08	8014	C00014
0A	3F0A	8019	C00019
0C	3F0C	801E	C0001E
0E	3F0E	8023	C00023
10	3F10	8028	C00028
12	3F12	802D	C0002D
14	3F14	8032	C00032
16	3F16	8037	C00037
18	3F18	803C	C0003C
1A	3F1A	8041	C00041
1C	3F1C	8046	C00046
1E	3F1E	804B	C0004B
20	3F20	8050	C00050
22	3F22	8055	C00055
24	3F24	805A	C0005A
26	3F26	805F	C0005F
28	3F28	8064	C00064

Table 1. Interrupt Vectors and Corresponding Jump Locations (Continued)

Interrupt Vector	Primary Table Location (Hexadecimal)	Secondary Table Location (Hexadecimal)	User Table Location (Hexadecimal)
2A	3F2A	8069	C00069
2C	3F2C	806E	C0006E
2E	3F2E	8073	C00073
30	3F30	8078	C00078
32	3F32	807D	C0007D
34	3F34	8082	C00082
36	3F36	8087	C00087
38	3F38	808C	C0008C
3A	3F3A	8091	C00091
3C	3F3C	8096	C00096
3E	3F3E	809B	C0009B
40	3F40	80A0	C000A0
42	3F42	80A5	C000A5
44	3F44	80AA	C000AA
46	3F46	80AF	C000AF
48	3F48	80B4	C000B4
4A	3F4A	80B9	C000B9
4C	3F4C	80BE	C000BE
4E	3F4E	80C3	C000C3
50	3F50	80C8	C000C8
52	3F52	80CD	C000CD
54	3F54	80D2	C000D2
56	3F56	80D7	C000D7
58	3F58	80DC	C000DC
5A	3F5A	80E1	C000E1
5C	3F5C	80E6	C000E6
5E	3F5E	80EB	C000EB

NMI Handlers

The eZ80® Development Platform also supports a nonmaskable interrupt (NMI) that jumps to address 000066h. This interrupt is routed to SRAM, as shown in Table 2.

Table 2. NMI and Corresponding Jump Locations

Interrupt Vector	Primary Table Location (Hexadecimal)	Secondary Table Location (Hexadecimal)	User Table Location (Hexadecimal)
NMI	0066h	80F0h	C000F0h

RST *n* Handlers

The eZ80L92 device supports eight software Reset instructions. Upon encountering these Reset instructions, the Boot Loader utility calls to defined address locations in SRAM. Table 3 lists these Reset values and their corresponding call addresses and memory locations.

Table 3. Reset Instructions and Calls

Reset Instruction	Call Address	SRAM Location
RST 00h	00h	—
RST 08h	08h	C00104h
RST 10h	10h	C00109h
RST 18h	18h	C0010Eh
RST 20h	20h	C00113h
RST 28h	28h	C00118h
RST 30h	30h	C0011Dh
RST 38h	38h	C00122h

Installing the eZ80L92 Flash Loader Utility

The eZ80® Development Platform is equipped with the Flash Loader utility installed in the boot sector of Flash memory. In addition, the boot sector contains the first jump table (3F00h–3F5Fh) and the MAC address (3FFAh–3FFFh). If it subsequently becomes necessary to reinstall the Flash Loader program, or to update or modify the Flash Loader, follow the steps in this section.

Caution: When reinstallation is complete, reboot the eZ80® with ZPAK II disconnected, and start the Flash Loader utility with the eZ80® CPU operating in STANDALONE mode.

Necessary Hardware

- eZ80® Development Platform
- ZPAK II
- One serial cable
- One Ethernet cable
- One Ethernet hub with power supply
- Two power supplies

Procedure

The following steps demonstrate how to program the Flash Loader utility into the protected Flash boot block.

Always use a grounding strap to prevent damage resulting from electrostatic discharge (ESD).

1. Connect ZPAK II to the eZ80® Development Platform using the TIM and the ribbon cable.
2. Using the Ethernet cable and the hub, connect ZPAK II to the Network Interface Card of the PC running ZDS II.
3. Connect console port P2 of the eZ80® Development Platform to the COM port of the terminal PC.
4. Install the FlashWE jumper (J7).
5. Apply power to ZPAK II and to the eZ80® Development Platform.
6. Launch **HyperTerminal**. In the **Connect To** dialog of **HyperTerminal**, select the COM port that the eZ80® Development Platform is connected to. In the **COM1 Properties** dialog, set the COM port parameters to:
   - 57600 bps
   - 8 data bits
   - No parity
   - 2 stop bits
   - No flow control

7. From the **File** menu in ZDS, choose **Open Project**, and navigate the path `..\L92FlashLoader\L92ZPAKLoader`. Double-click the project file named `L92_Zpac_flash_Loader.pro` .
8. Choose **Project** → **Settings** and click the **Debugger** tab. Choose **ZPAK II** as the driver from the drop-down list and click **Configure ZPAK II**. In the **Configure ZPAK II** dialog box, enter the appropriate IP address and the port number for the ZPAK II hardware. Click **OK** to close the **Configure ZPAK II** dialog box. Click **OK** to close the **Settings** dialog box.
9. In ZDS, choose **Connect** to establish communications with ZPAK II. Choose Download Code to begin downloading code to the eZ80® Development Platform. (Alternatively, choosing the **Reset** command performs the above operations). A progress bar indicates the status of the download process.
10. After the download is complete, choose **Debug** → **Go** from the **Build** menu in ZDS. The program runs and the following output appears in the **HyperTerminal** window:

```
Reset
eZ80 Flash Loader Utility
Version x.xx ZDS (where x.xx denotes the current version)
Enter Mac Address >
```

11. In the **HyperTerminal** window, enter the MAC address located on a label on the bottom of the eZ80® Development Platform. This MAC address is provided for demonstration purposes only.

Note: No backspace characters are allowed when entering the MAC address. In the event of a keyboard input error, enter any string of characters until the program prompts the user. At the prompt, press *M* to cause the `Enter Mac Address >` prompt to reappear.

Example Output

```
Enter Mac Address > 00:90:23:00:00:00
Type 'H' for help
eZ80L92>
```

Note: The user can enter *H* at any time to list the available Flash Loader commands.

12. Press *L* to prepare the eZ80® Development Platform to receive code. If the Write Enable jumper (J7) is not installed, the user cannot program Flash.

Example Output

```
eZ80L92> L
Start sending file via Xmodem protocol...
```

13. Select the `..\L92FlashLoader\L92BootLoader\l92_eZ80flash_loader.hex` file and send this file via the XMODEM protocol from the **Send File** option of the **Transfer** menu in **HyperTerminal**.

    A brief delay ensues before the download begins (approximately 15 seconds). The Flash Loader utility completes loading into the protected boot sector of Flash.

Note: Make sure that the Write Enable jumper is installed when programming the boot sector. This Write Enable jumper is intended only for the boot sector. When the Flash Loader utility is programmed, the user can remove jumper J7 before programming the remainder of Flash memory to ensure that the boot block is protected from being accidentally overwritten.

When the transfer is complete, the user is prompted with the following output:

```
Done
The Interrupt Vector Table @8000h: installing...
eZ80L92>
```

14. Remove the applicable Write Enable jumper.
15. Stop ZDS and disconnect ZPAK II from the eZ80® Development Platform.

The Flash Loader utility now resides in the write-protected boot block of Flash memory, and the user can boot from Flash and load Flash without ZDS.

Loading User Code

After the Boot Loader code is written to Flash memory, it can be used to load user code starting at address `8200h`. For proper execution, the hex file downloaded into Flash must be properly built for the memory configuration of the eZ80® Development Platform, and the user interrupt table must be set up as explained in the Interrupt Vectors and Jump Table Arrangement section on page 4 (only if the user program requires interrupts).

Necessary Hardware

- eZ80® Development Platform

- Power supply
- Serial cable

Procedure

The following steps demonstrate how to load the user code section of Flash memory onto the eZ80® Development Platform.

Caution: Always use a grounding strap to prevent damage resulting from electrostatic discharge (ESD).

1. Connect console port P2 to the COM port of the terminal PC.
2. Ensure that jumper J7 is removed to disable the Flash WRITE in the boot sector.
3. Apply power to the eZ80® Development Platform.
4. Launch **HyperTerminal**. In the **Connect To** dialog of **HyperTerminal**, select the COM port that the eZ80® Development Platform is connected to. In the **COM1 Properties** dialog box, set the COM port parameters to:
   - 57600 bps
   - 8 data bits
   - No parity
   - 2 stop bits
   - No flow control
5. Hold down the space bar of the terminal PC and reset the eZ80® Development Platform.

The Flash Loader utility checks the serial port for a space character when it is reset. If a space character is sampled when reset, the eZ80® boots into the Flash Loader. If no space character is sampled at reset, the eZ80® jumps to the user application at address 8200h.

Example Output

```
Reset
eZ80 Flash Loader Utility
Version x.xx eZ80 (where x.xx denotes the current version)
Type 'H' for help
eZ80L92>
```

In the example above, the Flash Loader utility is booted from the Flash boot block. If ZDS initiates the program execution, ZDS appears following the version number, instead of eZ80.

Note: The user can enter *H* at any time to list the available Flash Loader commands.

6. Press *L* to load the user application into Flash memory.

Example Output

```
eZ80L92> L
Start sending file via Xmodem protocol…
```

Note: Every time the *L* command is issued, the Flash Loader program writes a secondary interrupt vector table in Flash memory. The example output shown above appears after a brief delay.

7. Select the appropriate hex file and send via the XMODEM protocol using the **Send File** option of the **Transfer** menu in **HyperTerminal**. Make sure that jumper J7 is removed when programming user code to protect the boot sector from being accidentally overwritten.

Example Output

```
Done
eZ80L92>
```

The user code section now contains code and is ready to run.

8. Reset the eZ80® Development Platform to run user code starting at address 8200h.

Demonstration Applications

Demonstration applications are included with the Flash Loader utility to demonstrate a number of the functions required to write a user application.

The *2Tables* and *3Tables* demo applications are designed to be used with the eZ80L92 device. The *2Tables* application demonstrates how to set up an interrupt table in Flash memory at address 8000h. *3Tables* demonstrates how to set up an interrupt table in SRAM at address C00000h. Except for this change in the interrupt setup, there is no difference between these two demo applications.

In the *2Tables* demo, the user interrupt table is a part of the user application code. Therefore, the user table is placed at address 8000h during the flashing of the application. The *2Tables* demo application shows how to place the user interrupt table into Flash memory at address 8000h and how to override the Flash Loader utility to place the secondary interrupt table at address 8000h. The `.asm` file associated with this demo features a `.secondtable` section containing a user interrupt table. The entries for the Timer0, UART0, and UART1 interrupts each feature associated ISR handlers. All entries for unused interrupts feature null ISRs.

In the *3Tables* demo, the user interrupt table is not a part of the user application code. The demo application shows how to place the user interrupt table into SRAM to start from address C00000h. As a part of the interrupt initialization routine, the user table is generated at runtime and placed into SRAM. First, the **main** function (see the `main.c` file in the *3Tables* demo) initializes the user interrupt table to display null ISRs for all interrupts (see `interrupt.c`). Next, the timer interrupt initialization (see `time.c`) and UART interrupt initialization (see `stream.c`) routines place their ISR handlers into their respective entries in the table. The Flash Loader utility creates the secondary interrupt tables in Flash memory at address 8000h during the flashing of the *3Tables* demo application.

All of the other demonstrated functions are common to both applications. These common functions are: stack initialization, chip select programming, NMI handling, and bus mode (for the eZ80L92 device only). These implementations are found in their respective `.asm` files. Other functions demonstrated in these demo applications are discussed elsewhere in this document.

The locations for the demo applications are as follows:

```
2Tables: FlashLoader120\Demo\L92\2Tables
```

```
3Tables: FlashLoader120\Demo\L92\3Tables
```

These two directories contain project-specific files only. There are several other source files that are common to both demo applications. The path for these files is:

```
FlashLoader120\Demo\CommonFiles
```

Note: The *2Tables* and *3Tables* demonstration applications predefine the TWOTABLE and THREETABLE symbols, respectively, to conditionally compile the common source files.

This publication is subject to replacement by a later edition. To determine whether a later edition exists, or to request copies of publications, contact:

ZiLOG Worldwide Headquarters
532 Race Street
San Jose, CA 95126
Telephone: 408.558.8500
Fax: 408.558.8300
www.ZiLOG.com

Document Disclaimer