In-System Programming

CP3UB17/CP3CN17
ISP and CRISP User’s Guide

National Semiconductor
The Sight & Sound of Information

This document is the property of National Semiconductor Corporation. No warranty is provided and no liability is assumed by National Semiconductor Corporation with respect to the accuracy of this documentation or the merchantability or fitness of the product for a particular application. No license of any kind is conveyed by National Semiconductor Corporation with respect to its intellectual property or that of others. All information in this document is subject to change without notice.

National Semiconductor Corporation products are not authorized for use in life support systems or under conditions where failure of the product would endanger the life or safety of the user, except when prior written approval is obtained from National Semiconductor Corporation.

CompactRISC™ is a trademark of National Semiconductor Corporation. The National Semiconductor logo is a registered trademark of National Semiconductor Corporation.

All other trademarks mentioned in this document are property of their respective companies.

Preface

Scope of This Document

This document provides user information for the In-System Programming (ISP) interface for the CP3UB17/CP3CN17 Connectivity Processors and the CRISP host-side ISP programming software. This document describes the hardware and software configuration of the ISP interface for both the chip side and the host side of the interface. The protocol for communication over the ISP interface is described in a separate document, the **In System Programming Communication Protocol Specification**.

Related Documentation

CP3CN17/CP3UB17 Evaluation Kit User's Guide—The hardware and software documentation for the CP3UB17/CP3CN17 Evaluation Kit.

In System Programming Communication Protocol Specification—This is the reference for the protocol used over the ISP interface. This document is useful for developing host-side ISP interface software.

CP3UB17/CP3CN17 Data Sheets—These are the full data sheets for the Connectivity Processors. Refer to these documents for information about the on-chip peripheral devices, signal descriptions, package pinout, and electrical specifications.

Preface

Revision History

Revision	Release Date	Summary of Changes
1.5	9/17/03	Original release.

Table of Contents

Table of Contents

Overview

1

The In-System Programming (ISP) interface provides a mechanism for updating the software of a CP3UB17/CP3CN17 device after it has been installed in a system. This may be used to correct software bugs, update software with additional features, customize the software for specific products, etc.

As shown in Figure 1-1, a conventional application downloaded through the Nexus or parallel port interface may use the entire 256K byte flash program memory. If the ISP interface is used, up to 7K bytes of the flash program memory are reserved for the ISP software that handles the communication interface with the host system and loads the application into flash program memory. **BOOTAREA** is the address of the first byte of memory available for application programs downloaded through the ISP.

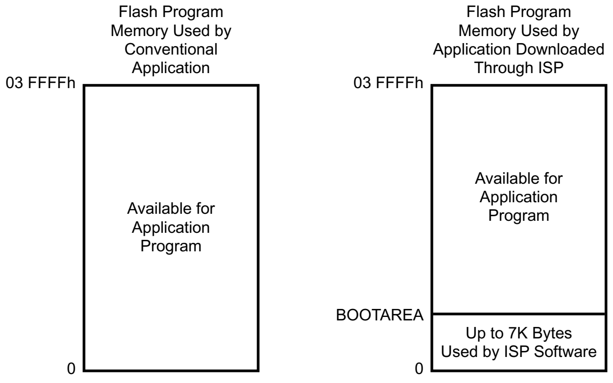

Flash Program Memory Used by Conventional Application
03 FFFFh
Available for Application Program
0
Flash Program Memory Used by Application Downloaded Through ISP
03 FFFFh
Available for Application Program
BOOTAREA
Up to 7K Bytes Used by ISP Software
0

Figure 1-1. Flash Program Memory Map

Overview

The ISP software provided by National Semiconductor implements a protocol for downloading and verifying code over the on-chip UART interface of the CP3UB17/CP3CN17 device. The states sampled from the ENV pins at reset are used to select whether the device executes the ISP software or application software following reset.

The CRISP software provides a user interface for controlling the ISP interface from a host PC. CRISP provides a complete set of functions for downloading, verifying, and uploading code on the target device, as well as loading, saving, and editing code in disk files on the host PC.

The remainder of this User's Guide consists of these chapters:

Chapter 2—Setting Up for ISP Download. Describes setting up the hardware and software configuration for running the procedures in the following chapters.

Chapter 3—Downloading Through the ISP. Describes downloading an example application program through the ISP. This is a very quick procedure to demonstrate the basic ISP functions. The following chapters provide more detailed information and procedures for using the CRISP host-side software and compiling application programs for downloading through the ISP.

Chapter 4—CRISP User Interface. Describes the graphical user interface (GUI) of CRISP.

Chapter 5—Using CRISP. Detailed procedures for establishing a serial connection from the host PC to the target device, loading a disk file into a CRISP buffer, setting the timing registers for programming the flash memory, downloading a CRISP buffer to a memory area in the target device, verifying the programming of the target device against a CRISP buffer, and closing the serial connection.

Chapter 6—Other CRISP Features. Detailed information on examining data in a CRISP buffer, locating mismatches found during a memory verification operation, programming the Protection Word, reading the Protection Word, uploading a CRISP buffer from the target device, saving a CRISP buffer to a disk file, setting up and using the Auto Program mode ("single click" programming of several programming operations), and editing the contents of a CRISP buffer.

Chapter 7—Compiling for ISP Download. Detailed procedure for compiling an application program for downloading through the ISP interface.

Overview

Setting Up for ISP Download

2

2.1 Hardware Configuration

The hardware configuration for using the ISP interface is shown in Figure 2-1. The parallel port cable is used for downloading the software which handles the target side of the ISP interface. To enable the parallel port connection, the LPT / Nexus jumper must be installed on the LPT side of the jumper block.

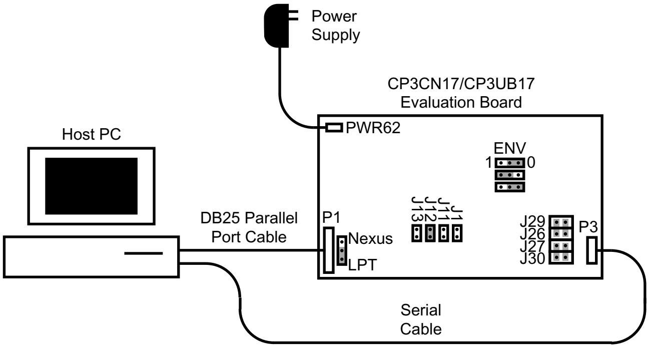


Figure 2-1. System Configuration

2.1.1 Clock Source Jumpers

To run the ISP code, the clock source jumpers must be configured for the crystal network or the oscillator module. To use the crystal network, jumpers J1 and J11 are installed, and jumpers J12 and J13 are empty. To use the oscillator module, jumper J12 is installed and jumpers J1, J11, and J12 are empty.

2.1.2 ENV Jumpers

The operating environment of the CP3UB17/CP3CN17 device is selected by the states sampled from the ENV pins. On the evaluation board, these states are controlled by jumpers J15, J16, and J17. Each of these jumpers has two positions, marked as 0 and 1. Table 2-1 shows the jumper configurations used to select the operating environment. See the CP3UB17/CP3CN17 device data sheet for detailed information about the behavior of the CPU in these modes.

Table 2-1. **Operating Environment Jumper Configurations**

Environ -ment	ENV2 (J17)	ENV1 (J16)	ENV0 (J15)	Description
DEV	0	0	0	Development mode. On-chip flash program memory is disabled. External SRAM program memory is enabled.
IRE	1	1	1	Internal ROM mode. On-chip flash program memory is enabled.
ERE	0	1	1	External ROM mode. Both on-chip flash program memory and external flash program memory are enabled.
ISP	0	1	0	In-System Programming mode. On-chip flash program memory is enabled.

To download the software that runs the ISP interface, the ENV2:0 jumpers are configured for IRE mode (111) or ERE mode (011). To run the ISP software, the ENV2:0 jumpers are configured for ISP mode (010). While the ISP software is running on the evaluation board, the CRISP software can be run on the host PC to download an application program to the evaluation board through the ISP interface. After an application program has been downloaded, the ENV2:0 jumpers are configured for IRE mode (111) or ERE mode (011) to run the program.

Whenever the ENV jumper configuration is changed, the power to the evaluation board should be interrupted by unplugging the power supply from connector PWR62, waiting a few seconds, then plugging it back in.

2.1.3 Serial Port Jumpers

Jumpers are used to configure the signals between the on-chip UART and serial port connector P3 for a crossover or straight-through configuration. The UART configuration jumpers are in two sets of adjacent two-pin headers, as shown in Figure 2-2.

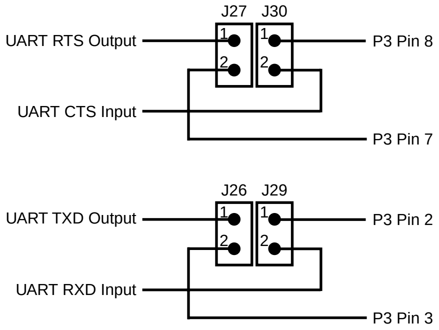


Figure 2-2. UART Configuration Jumpers

Jumpers are installed either between pins on the same header or between the same pin positions on adjacent headers. Table 2-2 shows the jumper configurations.

Table 2-2. UART Jumper Configurations

Crossover	Straight-Through
J26 Pin 1 - J29 Pin 1 J26 Pin 2 - J29 Pin 2 J27 Pin 1 - J30 Pin 1 J27 Pin 2 - J30 Pin 2	J26 Pin 1 - J26 Pin 2 J27 Pin 1 - J27 Pin 2 J29 Pin 1 - J29 Pin 2 J30 Pin 1 - J30 Pin 2

The serial cable provided in the evaluation kit requires a crossover configuration.

2.2 Software Configuration

2.2.1 CRISP Software

The CRISP software used for handling the host side of the ISP interface is installed from the evaluation kit CD-ROM. It may also be distributed as a self-installing **CRISPvX.Y.Z.EXE** file (in which X.Y.Z is the version number of the software release). After installation, the default location for the CRISP application is **C:\Program Files\CRISP\CRISP.exe**.

2.2.2 ISP Software

After installing the evaluation kit CD-ROM, the **C:\National_SEK_X_Y\software\CP3000\example\ISP** folder (in which X.Y is the version number of the software release) contains the files used to download the ISP software to the evaluation board and an example application compiled for downloading through the ISP. The ISP software is provided as an Intel Extended Hex format file, **isp_flash_xx17.hex**. This file must be downloaded with special programming of the Protection Word to allocate space for the ISP code and indicate that it is non-empty. This programming is provided in the winIDEA project files **isp.QRF** and **isp.jrf**. Certain combinations of states in the Protection Word can prevent the CP3UB17/ CP3CN17 device from accepting software downloads (by enabling protection modes that cannot be disabled), so it is safer to use these project files rather than using the Protection Word programming features of winIDEA to create new project files.

Application programs downloaded through the ISP must be compiled for loading above the ISP software. The ISP software is loaded into the region of program flash memory between 0 and **`BOOTAREA-1`**.

The following procedure downloads the ISP software:

1. **Configure the Evaluation Board for Downloading.** The evaluation board must be configured as shown in Figure 2-1, with the ENV jumpers selecting the IRE (111) or ERE (011) operating environment.
2. **Download the Hex File to the Target.** Double-click on the **isp.jrf** file to enter winIDEA and open the ISP project. Select the Debug -> Download command. After successfully downloading the ISP software, exit from winIDEA.

Downloading Through the ISP 3

An application program downloaded through the ISP is loaded starting at the **BOOTAREA** address. For detailed information about specifying the **BOOTAREA** address and compiling application programs for downloading through the ISP, see Chapter 7.

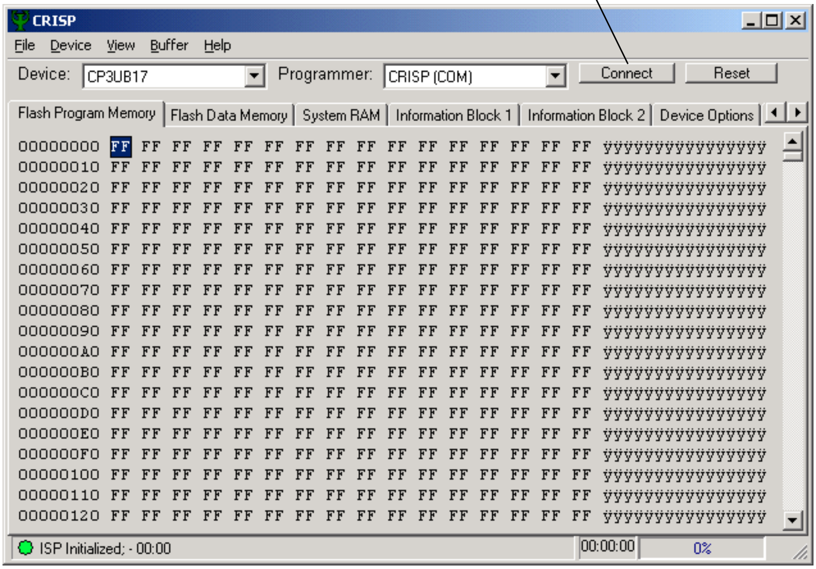
An example application program **LED1_Blink.hex** is provided for downloading through the ISP interface. When executed on the evaluation board, this program flashes LD1.

The following procedure uses CRISP to download the **LED1_Blink.hex** example program:

1. **Configure the ENV Jumpers.** After setting up the hardware and software as described in Chapter 2, configure the ENV jumpers on the evaluation board for ISP mode (010). Then, power-cycle the board by unplugging the power supply from connector PWR62, waiting a few seconds, and then plugging it back in.

2. **Enter CRISP.** From the Start menu, select Programs -> CRISP -> CRISP. After the program starts up, click on the splash image to close it, then click on the Connect button.

Connect Button

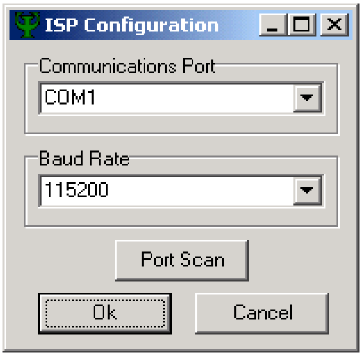


Device: CP3UB17 Programmer: CRISP (COM)

```
00000000 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000010 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000020 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000030 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000100 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000110 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

ISP Initialized; - 00:00 00:00:00

3. **Connect to the Evaluation Board.** A dialog box allows selecting the port and baud rate. Select the port used for the serial connection to the evaluation board. The serial connection must be set for 115200 baud. After clicking the Ok button, CRISP will attempt to connect to the board.

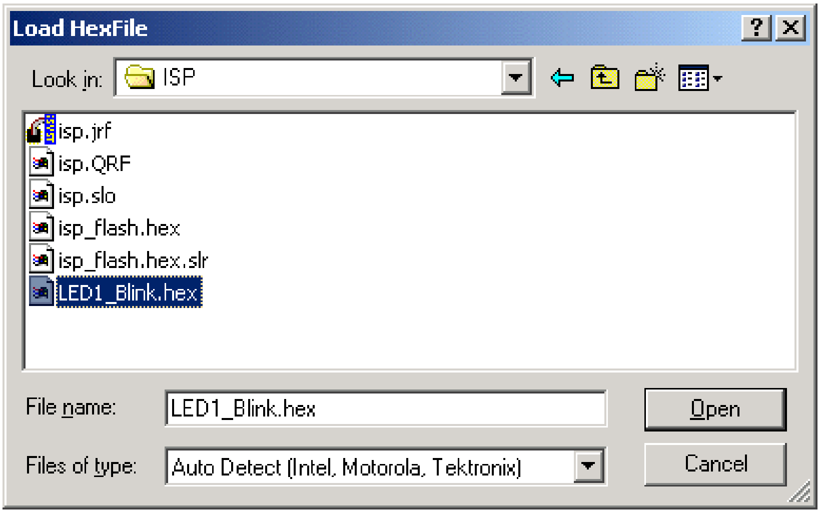


Communications Port: COM1

Baud Rate: 115200

If CRISP fails to connect, power-cycle the board and try again. Also, check that jumpers J26, J27, J29, and J30 for the UART signals are configured for a crossover configuration rather than a straight-through configuration, as described in Chapter 2.

4. **Load the Download File Into CRISP.** Select the File -> Load -> Load All command. This brings up a dialog box for selecting the download file. Browse to the **C:\National_SEK_X_Y\software\CP3000\example\ISP** folder (in which X.Y is the version number of the software release), then select the **LED1_Blink.hex** file and click the Open button.



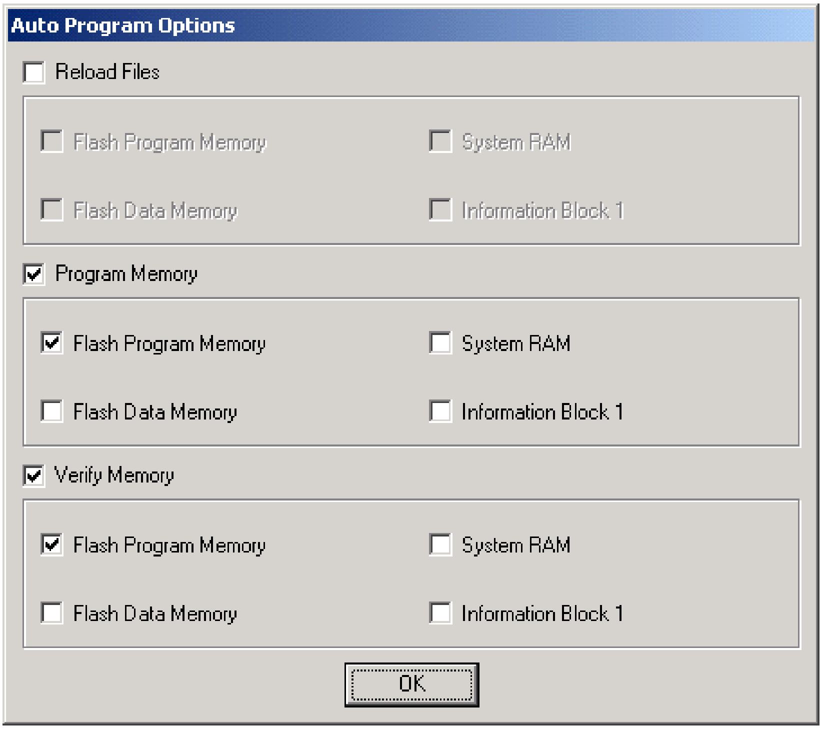
Look in: ISP

isp.jrf
isp.QRF
isp.slo
isp_flash.hex
isp_flash.hex.slr
LED1_Blink.hex

File name: LED1_Blink.hex

Files of type: Auto Detect (Intel, Motorola, Tektronix)

5. **Set the Auto Program Options.** Select the Device -> Auto Program Options command. Check the Program Memory, Flash Program Memory, Verify Memory, and Verify Flash Program Memory options. Then, click the OK button.



☐ Reload Files

☐ Flash Program Memory ☐ System RAM

☐ Flash Data Memory ☐ Information Block 1

☑ Program Memory

☑ Flash Program Memory ☐ System RAM

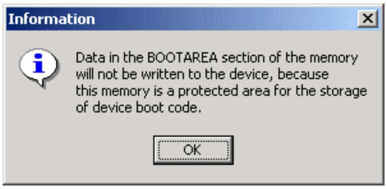
☐ Flash Data Memory ☐ Information Block 1

☑ Verify Memory

☑ Flash Program Memory ☐ System RAM

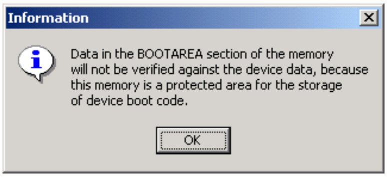
☐ Flash Data Memory ☐ Information Block 1

6. **Download to the Evaluation Board.** Select the Device -> Auto Program command. A dialog box will warn that the BOOTAREA section will not be programmed. Click the OK button to dismiss the dialog box and download the application program.



Data in the BOOTAREA section of the memory will not be written to the device, because this memory is a protected area for the storage of device boot code.

Another warning will appear during the verify phase. Click the OK button to dismiss the dialog box.



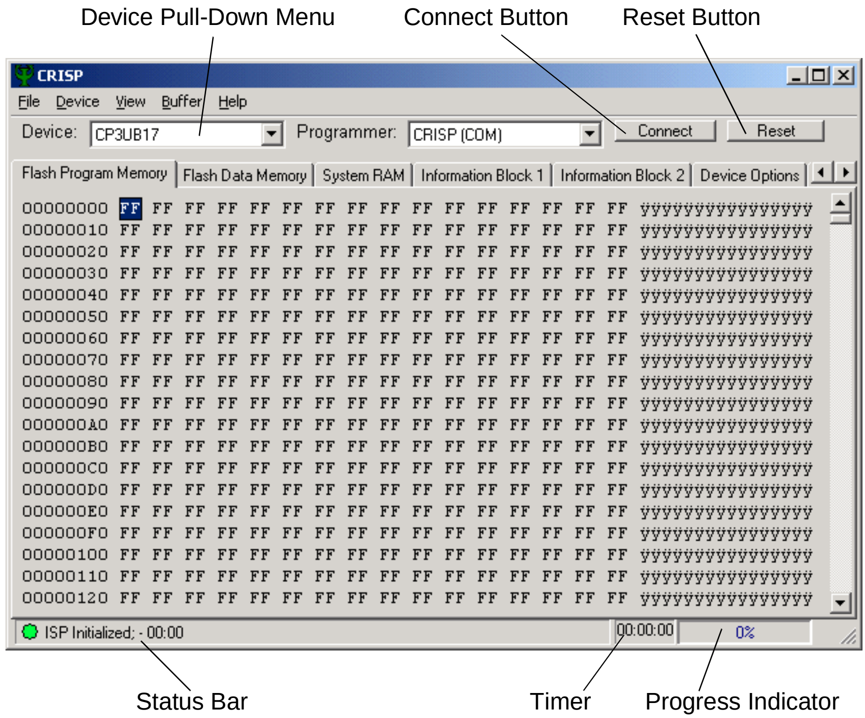
Data in the BOOTAREA section of the memory will not be verified against the device data, because this memory is a protected area for the storage of device boot code.

7. **Run the Program.** To run the downloaded software, change the ENV jumper configuration to IRE (111) or ERE (011) mode. Then, press the Reset button. The evaluation board will execute the application software, as indicated by one of the LEDs flashing at a rate of about 1 Hz.

Downloading Through the ISP

4 CRISP User Interface

The CRISP user interface is straightforward and simple to learn. This chapter describes the user interface and the functions accessed through the user interface. A screen shot of CRISP is shown in Figure 4-1, and each part of the interface is described below.



Device: CP3UB17

Programmer: CRISP (COM)

```
00000000 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000010 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000020 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000030 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000100 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000110 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

● ISP Initialized; - 00:00 | 00:00:00 | 0%

Figure 4-1. CRISP User Interface

- **Device Pull-Down Menu**—use this menu to select the appropriate CP3000 device.
- **Connect Button**—the Connect button is used to initiate the connection process to the target device. Once a device is connected, this becomes the Disconnect button.
- **Reset Button**—the Reset button can be used to reset the connection, and it becomes the Abort button during any ISP operation. Clicking the Abort button will terminate the current ISP operation.
- **Status Bar**—CRISP communicates its status to the user through the status bar. When CRISP is performing programming or verification commands, the progress of the operation will be displayed here. If an error occurs during a ISP process, it will be reported here.
- **Timer**—indicates elapsed time for the current CRISP operation.
- **Progress Indicator**—while a CRISP operation is proceeding, displays a bar graph and a numerical percentage value indicating how much of the current operation has been completed.

The main window of CRISP displays several tabs which provide further information on memory contents, device-specific settings, and CRISP status information.

The first four tabs in Figure 4-1 display the contents of the four main memory buffers of CRISP, which relate to the four main memory storage locations of the CP3UB17/CP3CN17 device. These buffers display the data read from the target device after performing an upload or the data the user wishes to download to these areas after opening a valid Intel Hex File.

The Device Options tab displays various settings relevant to the chosen target device, such as memory protection settings and the current flash timing settings.

The Status tab maintains a continuous list of CRISP activities reflecting the status of the current ISP operation and highlighting any erroneous activity.

5 Using CRISP

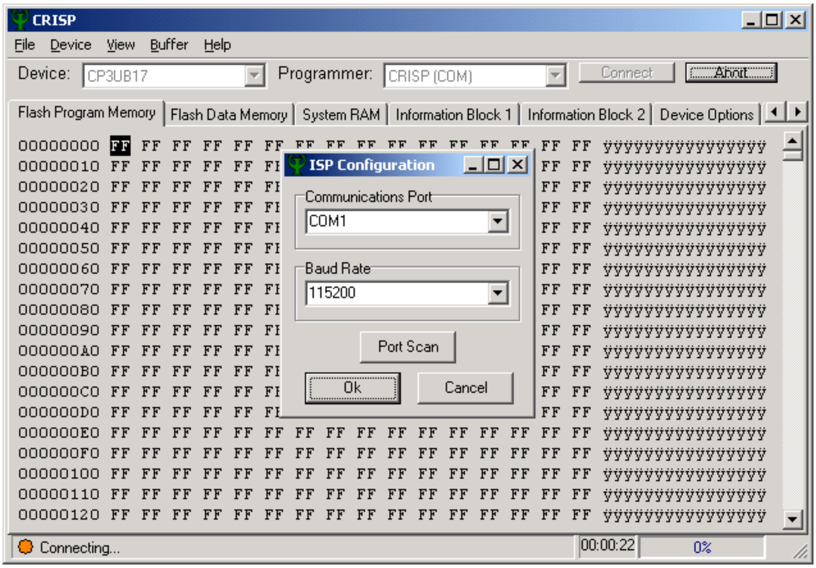
This chapter provides detailed procedures for using CRISP to:

- Make a connection to an ISP-enabled CP3UB17/CP3CN17 device.
- Open an Intel Hex file.
- Program the file into the flash program memory.
- Verify the contents of the flash program memory.
- Terminate the connection with the target device.

5.1 Establishing a Serial Connection

To establish a serial connection, the host PC must be connected to the target device through an available COM port. The target device must be in ISP mode with the hardware configuration described in Chapter 2. The following procedure establishes a serial connection.

1. **Select Port and Baud Rate.** Click the Connect button once. The screen should then look like the screen shot in Figure 5-1.



Device: CP3UB17

Programmer: CRISP (COM)

Communications Port: COM1

Baud Rate: 115200

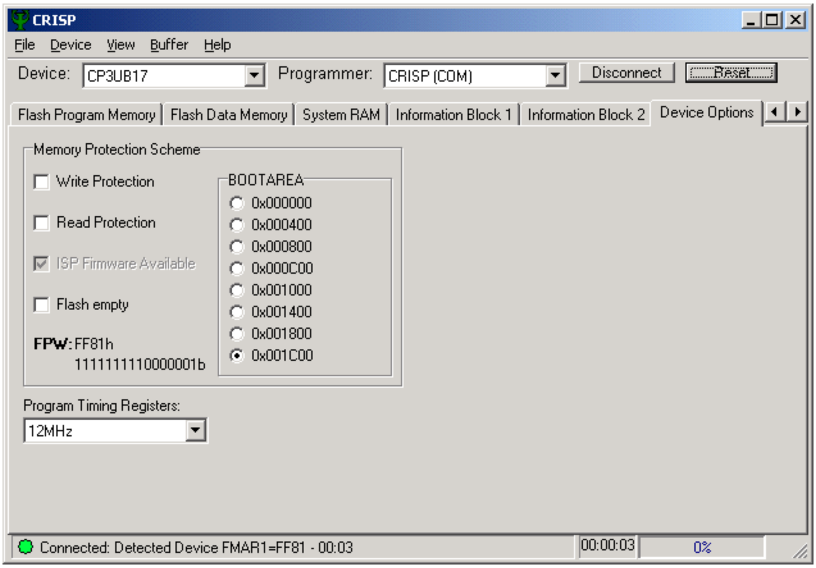
```
00000000 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000010 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000020 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000030 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000040 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000050 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000A0 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000B0 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000C0 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000D0 FF FF FF FF FF FI                          FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000100 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000110 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

Connecting... 00:00:22 0%

Figure 5-1. Screen after Clicking Connect Button

The Communications Port pull-down menu is used to select the COM port used on the PC. The Baud Rate pull-down menu is used to select the connection speed, which must be set for 115200 baud. To begin making the connection, click the Ok button.

2. **Connecting to the Target.** After a few seconds, the connection should be made, and the CRISP application will look like that shown in Figure 5-2.



Device: CP3UB17

Programmer: CRISP (COM)

Memory Protection Scheme

□ Write Protection

□ Read Protection

☑ ISP Firmware Available

□ Flash empty

FPW: FF81h
1111111110000001b

BOOTAREA

○ 0x000000
○ 0x000400
○ 0x000800
○ 0x000C00
○ 0x001000
○ 0x001400
○ 0x001800
● 0x001C00

Program Timing Registers: 12MHz

● Connected: Detected Device FMAR1=FF81 - 00:03 | 00:00:03 | 0%

Figure 5-2. Screen after a Connection has been Successfully Established

The target device flash protection settings and the current flash timing register settings are displayed in the Device Options tab in the main window. The status bar shows a green circle, with a message declaring that the connection was a success.

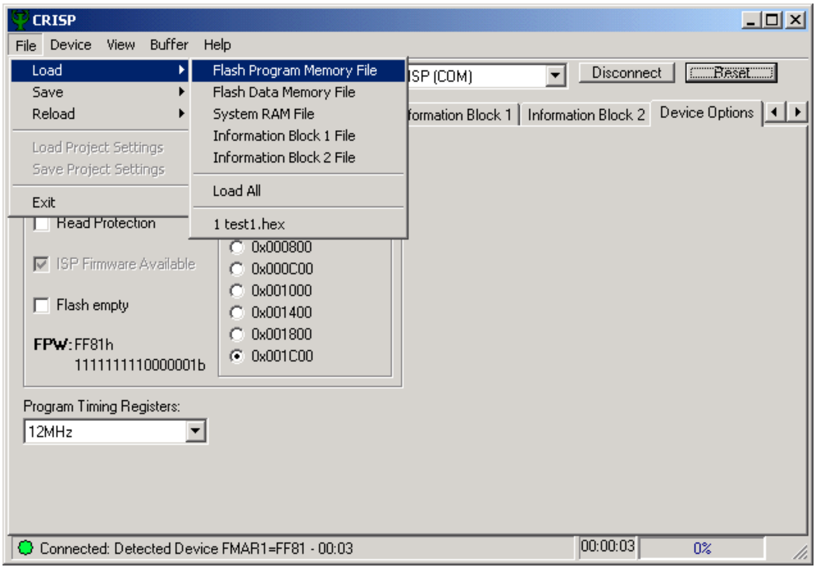
5.2 Loading an Intel Hex File

Disk files can be loaded into buffers maintained by CRISP for each memory area in the target device. The following procedure describes how to load an Intel Hex file into the program memory buffer.

Using CRISP

Loading an Intel Hex File

1. **Browse for a Hex File.** Select the File -> Load -> Flash Program Memory File command to bring up a dialog box which displays hex files in the current folder. The command is shown in Figure 5-3.



□ Read Protection

☑ ISP Firmware Available

□ Flash empty

FPW: FF81h
1111111110000001b

○ 0x000800
○ 0x000C00
○ 0x001000
○ 0x001400
○ 0x001800
● 0x001C00

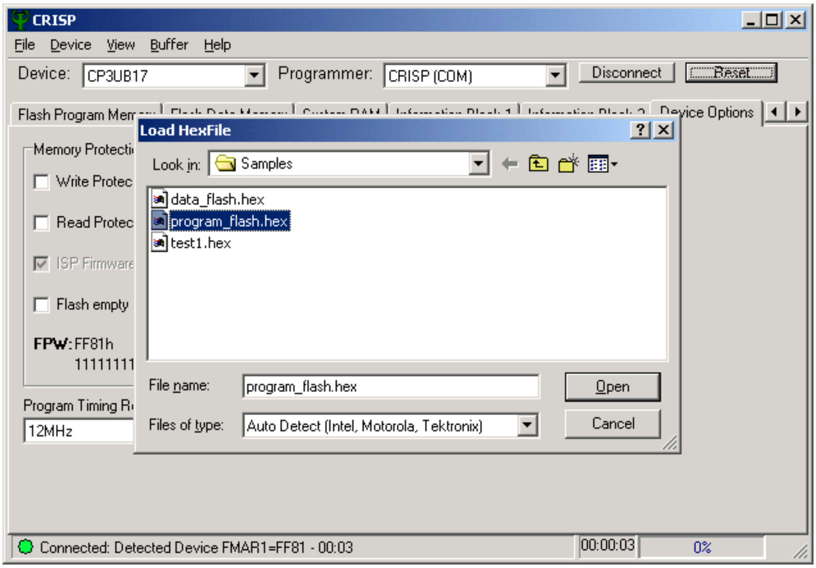
Program Timing Registers: 12MHz

Connected: Detected Device FMAR1=FF81 - 00:03 | 00:00:03 | 0%

Figure 5-3. Selecting the Command to Open a Program Flash File

By default the dialog box displays the contents of the **C:\Program Files\CRISP\Samples** folder. If this is not the case, browse for this folder. The folder should contain a file named **program_flash.hex**.

2. **Open the Hex File.** Select the file, as shown in Figure 5-4, and click the Open button.



Device: CP3UB17

Programmer: CRISP (COM)

FPW: FF81h
11111111

Program Timing R
12MHz

Look in: Samples

data_flash.hex
program_flash.hex
test1.hex

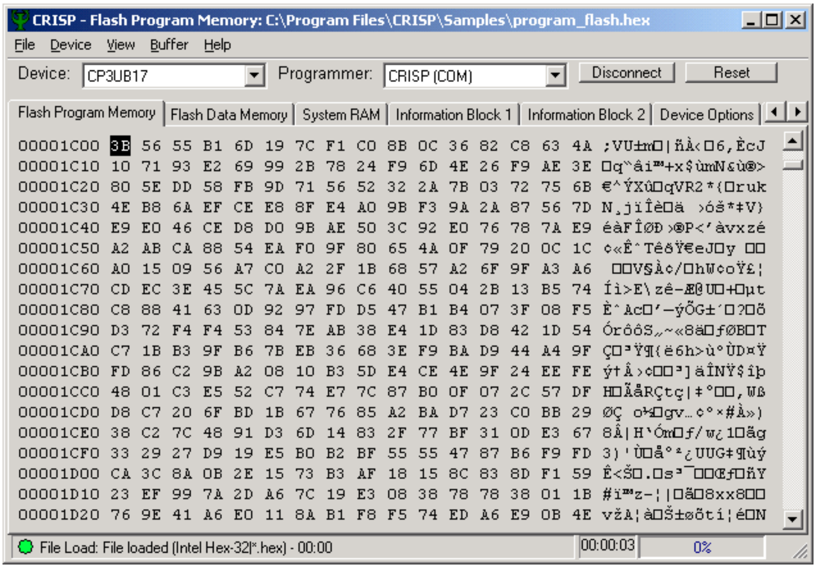
File name: program_flash.hex

Files of type: Auto Detect (Intel, Motorola, Tektronix)

Connected: Detected Device FMAR1=FF81 - 00:03 | 00:00:03 | 0%

Figure 5-4. Opening the program_flash.hex file

3. **Examine the Hex File.** The **program_flash.hex** file starts at memory address 00 1C00h. The contents of the file are displayed in the Flash Program Memory tab in the main window as shown in Figure 5-5.



Device: CP3UB17

Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6‚ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q“âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<’àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«ÊˆTêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 ÈˆAc□’—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ùºÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢º×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H‘Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

File Load: File loaded (Intel Hex-32|*.hex) - 00:00 | 00:00:03 | 0%

Figure 5-5. Contents of the File Displayed in the Main Window

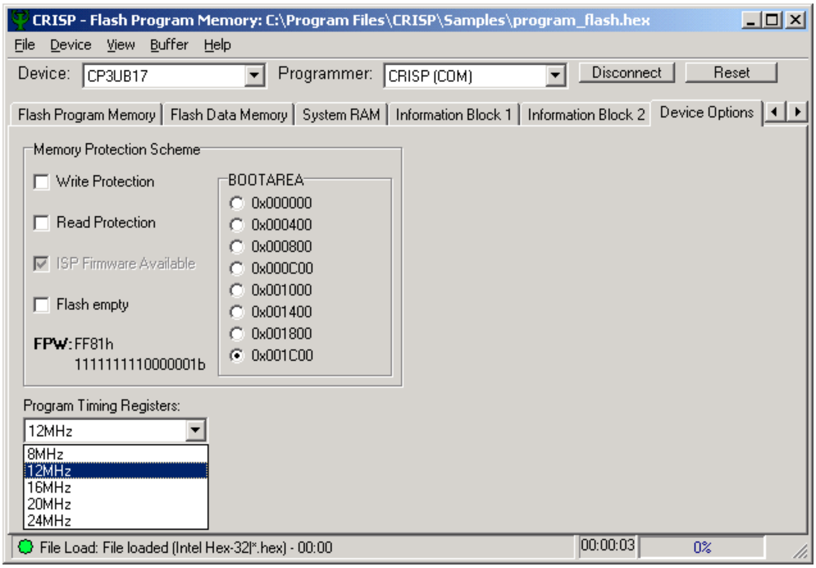
5.3 Setting the Flash Timing

To ensure data retention, the timing used to program the flash memory must be within specifications. The ISP firmware assumes by default that the CP3UB17/CP3CN17 device is driven with a 12-MHz clock source is connected, and the internal CPU clock is also set up for 12-MHz operation[1]. The following procedure programs suitable values into the flash timing registers to support a 12-MHz input clock.

If the flash timing register settings need to be changed for a different device type, the latest version of CRISP will have the register settings stored in its device database. The Device pull-down menu in the upper-left corner of the main window is used to select the device type.

[1] See the device data sheet for detailed information on clock generation.

1. **Select the Clock Frequency.** Click on the Device Options tab in the main CRISP window to bring up the view shown in Figure 5-6. Use the Program Timing Registers pull-down menu in the lower left corner of the window to select the "12 MHz" timing setting.



Device: CP3UB17

Programmer: CRISP (COM)

Memory Protection Scheme

□ Write Protection

□ Read Protection

☑ ISP Firmware Available

□ Flash empty

FPW: FF81h
1111111110000001b

BOOTAREA

○ 0x000000
○ 0x000400
○ 0x000800
○ 0x000C00
○ 0x001000
○ 0x001400
○ 0x001800
● 0x001C00

Program Timing Registers: 12MHz

8MHz
12MHz
16MHz
20MHz
24MHz

File Load: File loaded (Intel Hex-32|*.hex) - 00:00 | 00:00:03 | 0%

Figure 5-6. **Selecting the Clock Frequency**

2. **Program Flash Timing Registers.** Write the flash timing registers in the CP3UB17/CP3CN17 device by selecting the Device -> Program -> Program Timing Registers command, as shown in Figure 5-7.

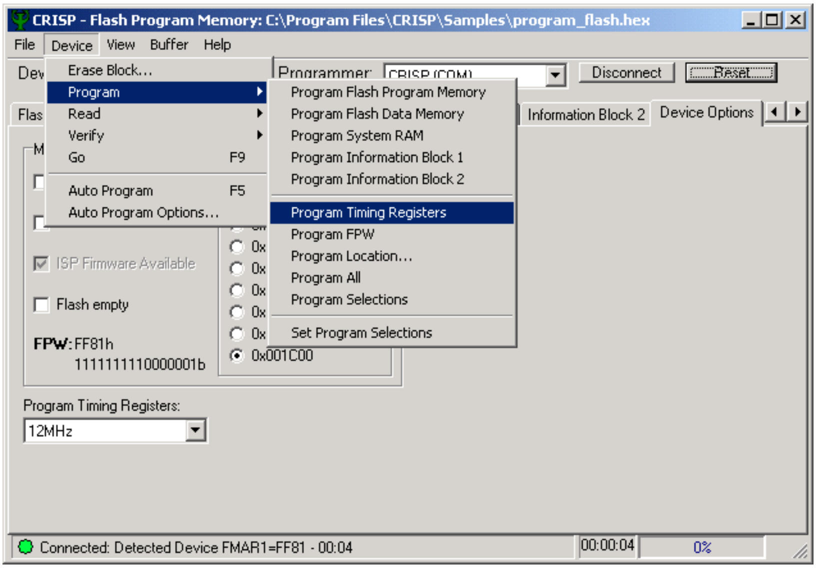
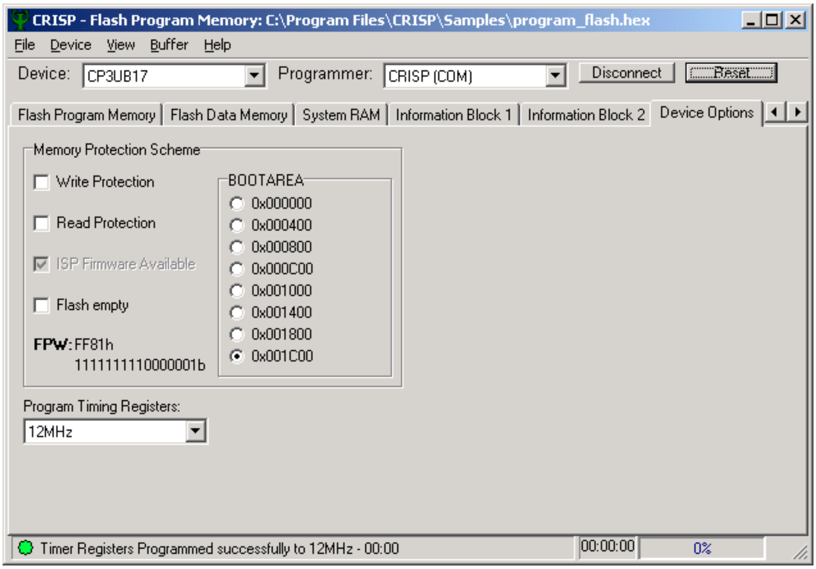


Figure 5-7. Programming the Flash Timing Registers

3. **Flash Timing Programming Complete.** After the timing registers have been successfully programmed, CRISP will display the message "Timing Registers Programmed successfully to 12 Mhz" in the status bar, as shown in Figure 5-8.



CRISP - Flash Program Memory: C:\Program Files\CRISP\Samples\program_flash.hex

Device: CP3UB17

Programmer: CRISP (COM)

□ Write Protection

□ Read Protection

☑ ISP Firmware Available

□ Flash empty

FPW: FF81h
1111111110000001b

○ 0x000000
○ 0x000400
○ 0x000800
○ 0x000C00
○ 0x001000
○ 0x001400
○ 0x001800
● 0x001C00

Program Timing Registers: 12MHz

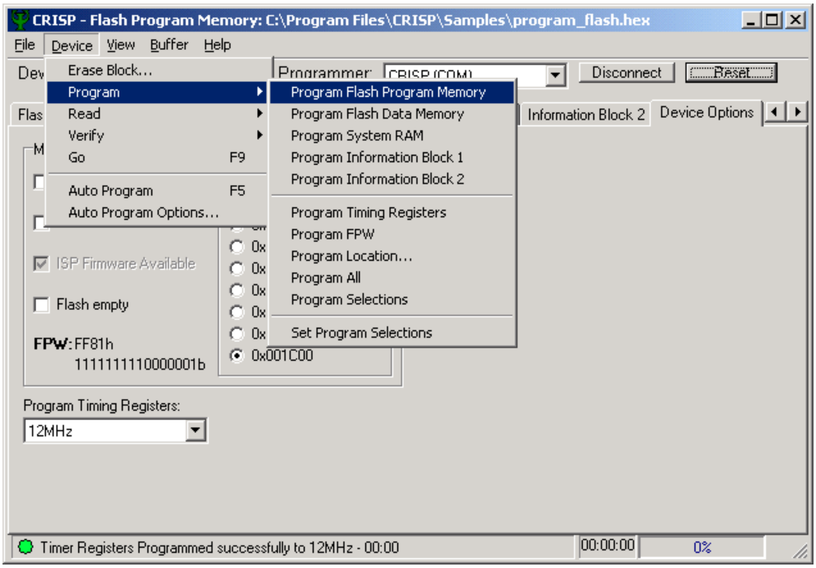
● Timer Registers Programmed successfully to 12MHz - 00:00 | 00:00:00 | 0%

Figure 5-8. Successful Completion of Flash Timing Register Programming

5.4 Programming the Flash Program Memory

Once the hex file has been loaded and the flash timing registers have been programmed, the hex file can be downloaded into the chosen memory area of the target. The following procedure loads the **program_flash.hex** file from CRISP buffer to the target.

1. **Initiate the Download.** Select the Device -> Program -> Program Flash Program Memory command, as shown in Figure 5-9.



CRISP - Flash Program Memory: C:\Program Files\CRISP\Samples\program_flash.hex

Programmer: CRISP (COM)

☑ ISP Firmware Available

□ Flash empty

FPW: FF81h
1111111110000001b

● 0x001C00

Program Timing Registers:
12MHz

Timer Registers Programmed successfully to 12MHz - 00:00

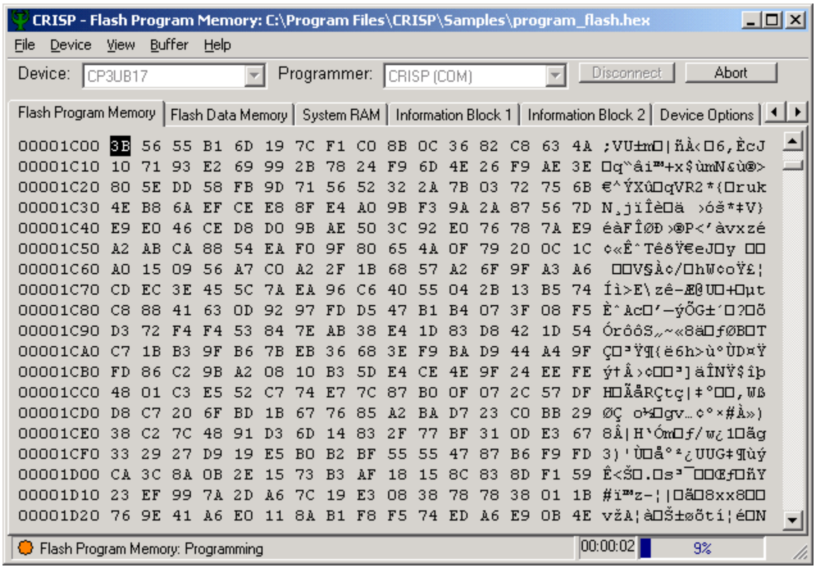
00:00:00

0%

Figure 5-9. Selecting the Program Flash Program Memory Command

The CRISP application will now download the hex file into the program flash memory of the target device.

2. **Monitor CRISP Status.** During this process, the status bar should look as shown in Figure 5-10.



Device: CP3UB17

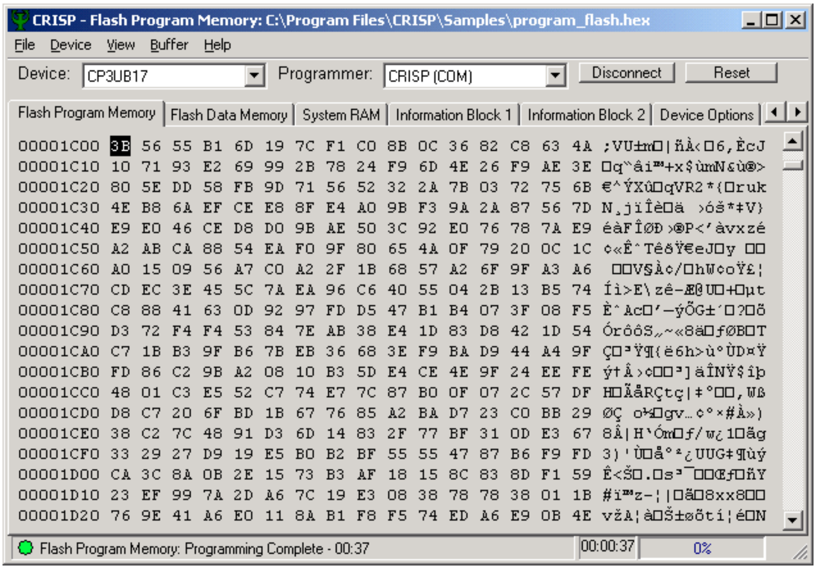
Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6‚ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q“âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<’àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«ÊˆTêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 ÈˆAc□’—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ùºÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢º×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H‘Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

Flash Program Memory: Programming | 00:00:02 | 9%

Figure 5-10. **CRISP Status Bar During Programming**

3. **Programming Complete.** If the download was successful, the status bar's circle should have returned to its green color and display the message "Programming Complete", as shown in Figure 5-11.



CRISP - Flash Program Memory: C:\Program Files\CRISP\Samples\program_flash.hex

Device: CP3UB17

Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6,ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q"âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<'àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«Ê^TêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 È^Ac□'—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ùºÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢º×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H'Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

● Flash Program Memory: Programming Complete - 00:37

00:00:37

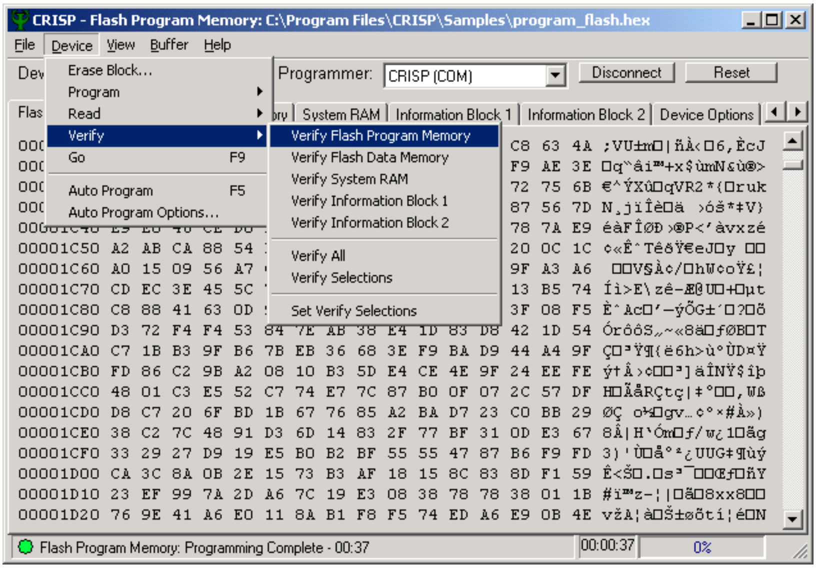
0%

Figure 5-11. Status Bar after a Successful Flash Program

5.5 Verifying the Contents of the Flash Program Memory

During the programming process, each flash page which is programmed is also automatically verified. However, it is also possible to explicitly verify the device programming against the contents of a CRISP buffer. The following procedure verifies the flash program memory.

1. **Initiate the Verification Operation.** Select the Device -> Verify -> Verify Flash Program Memory command. This is shown in Figure 5-12.



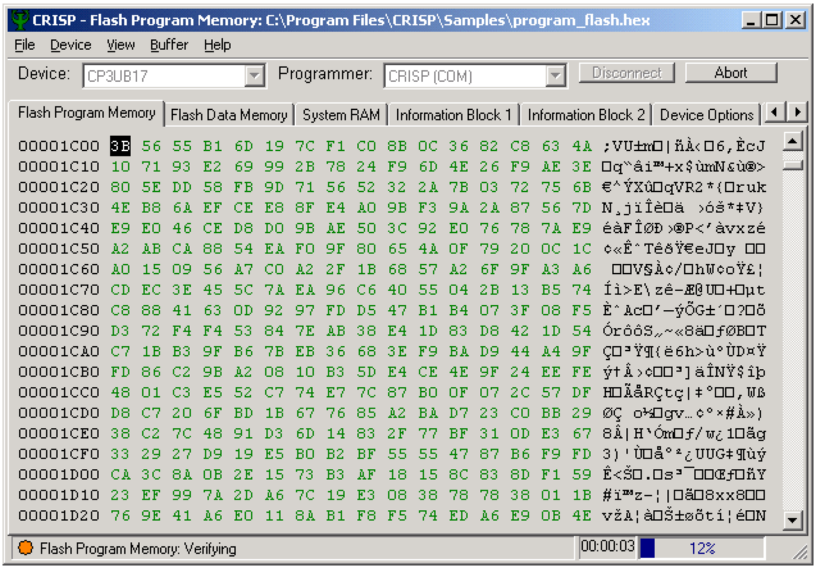
```
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ù°ÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢°×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H‘Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

Flash Program Memory: Programming Complete - 00:37 00:00:37 0%

Figure 5-12. **Selecting the Verify Flash Program Memory Command**

The CRISP application will now use the ISP protocol to upload the contents of the flash program memory and compare it with the hex file loaded into the memory of the PC.

2. **Monitor Verification Status.** During this process the status bar should look as shown in Figure 5-13.



CRISP - Flash Program Memory: C:\Program Files\CRISP\Samples\program_flash.hex

Device: CP3UB17

Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6‚ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q“âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<’àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«ÊˆTêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 ÈˆAc□’—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ù°ÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢°×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H‘Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

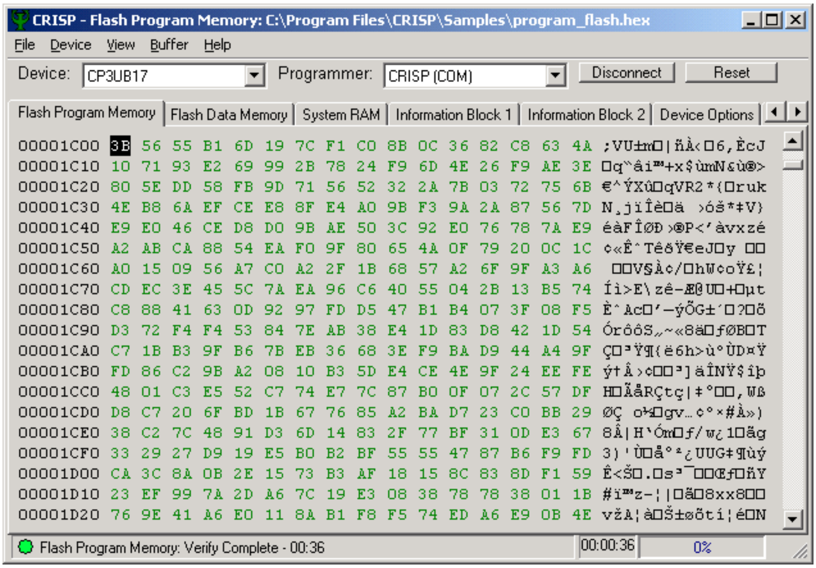
Flash Program Memory: Verifying

00:00:03

12%

Figure 5-13. **CRISP Status Bar During Verification**

3. **Verification Complete.** If the verification was successful, the status bar’s circle should have returned to its green color and display the message "Verify Complete", as shown in Figure 5-14.



Device: CP3UB17

Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6,ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q“âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<’àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«ÊˆTêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 ÈˆAc□’—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ùºÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢º×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H‘Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

● Flash Program Memory: Verify Complete - 00:36

00:00:36

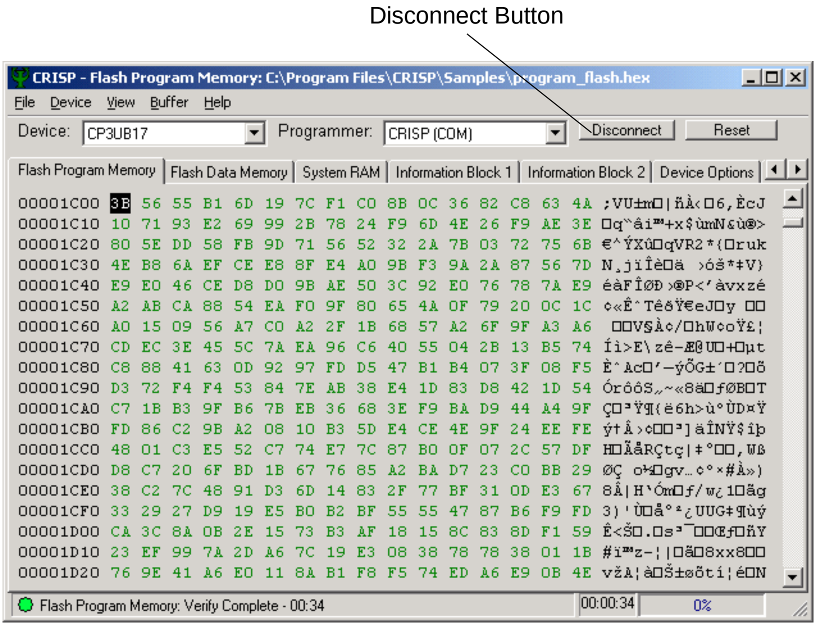
0%

Figure 5-14. Status Bar after the Verification Process

5.6 Closing the Serial Connection

When all ISP operations have been completed, the serial connection from the host PC to the target should be closed to make it available for other applications or to allow CRISP to connect to other target devices. The following procedure closes the connection.

1. **Initiate Closing the Connection.** Click the Disconnect button in the CRISP main window, as shown in Figure 5-15.



Device: CP3UB17

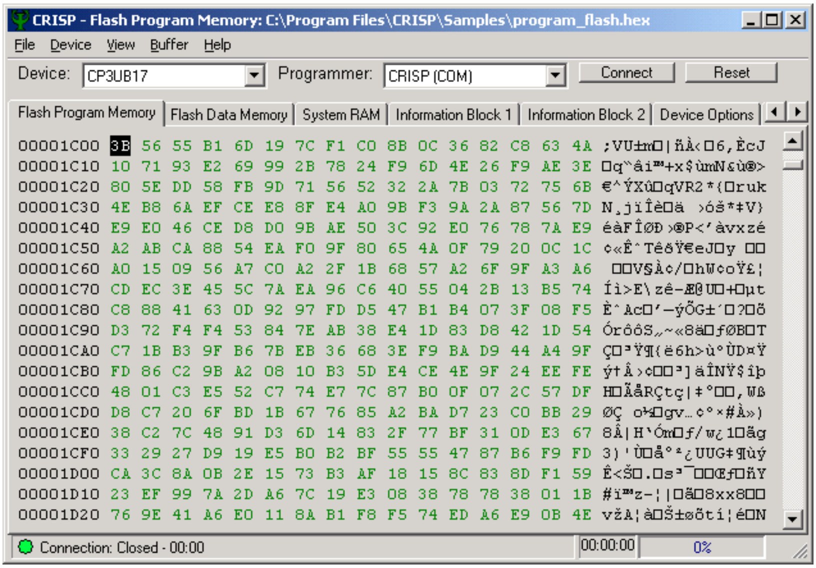
Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6,ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q"âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<'àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«ÊˆTêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 ÈˆAc□'—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ùºÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢º×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H'Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

● Flash Program Memory: Verify Complete - 00:34 | 00:00:34 | 0%

Figure 5-15. Click the Disconnect Button

2. **Connection Closed.** When CRISP has completed the disconnection process, the status bar will display the message "Connection: Closed", as shown in Figure 5-16.



Device: CP3UB17

Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6,ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q"âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<'àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«ÊˆTêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 ÈˆAc□'—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ùºÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢º×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H'Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

● Connection: Closed - 00:00 | 00:00:00 | 0%

Figure 5-16. Status Bar Indicating Connection is Closed

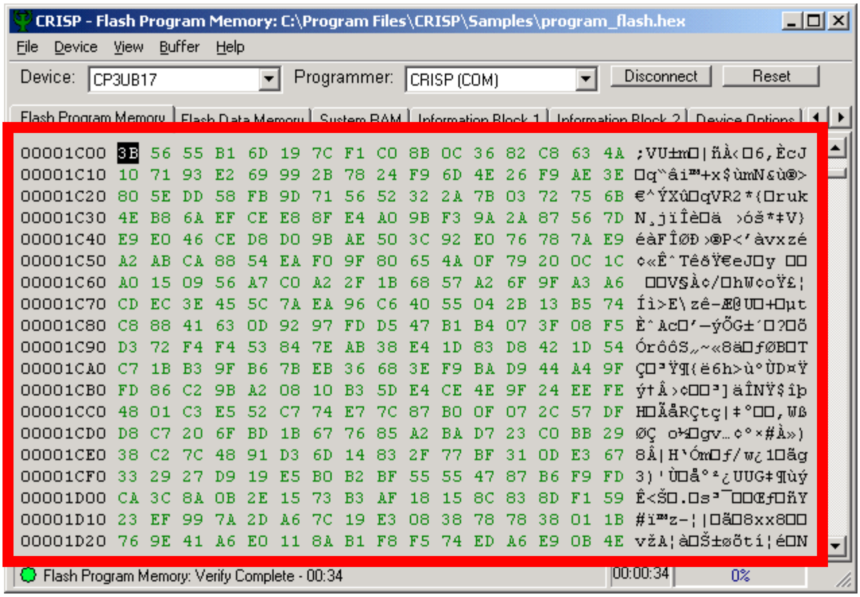
The CRISP application can now be closed, or it can connect to another target device.

6 Other CRISP Features

Chapter 5 described how to download an Intel Hex File into the flash program memory of an ISP-enabled CP3UB17/CP3CN17 device. This chapter looks at some of the other functions supported by CRISP and shows how to get the most out of its features.

6.1 Decoding Hex File Data

After loading a hex file into a CRISP buffer, it is sometimes useful to be able to examine and understand the data it contains. In the main window, CRISP displays the addresses and data in hexadecimal format, with 16 bytes of data per line. On the right side of the main window, the data is decoded into ASCII text, in which each byte is displayed as an ASCII character. The data displayed in the main window is highlighted in red in Figure 6-1.



Device: CP3UB17

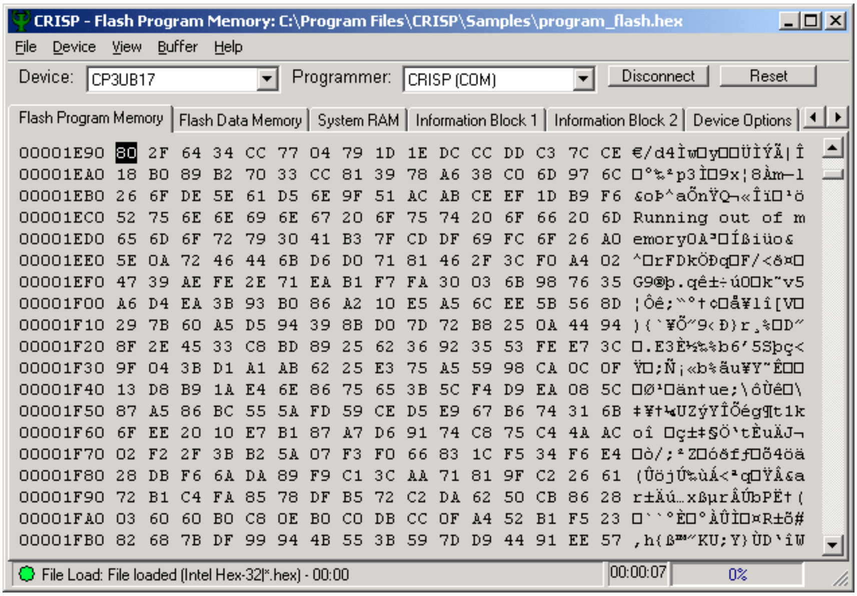
Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6‚ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q“âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<’àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«ÊˆTêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 ÈˆAc□’—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 54 ÓrôôS„~«8ä□ƒØB□T
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ùºÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢º×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H‘Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

Flash Program Memory: Verify Complete - 00:34 | 00:00:34 | 0%

Figure 6-1. Intel Hex File Displayed in CRISP

If ASCII text is embedded in a file, it is easy to locate, as shown in Figure 6-2. Here, the ASCII character R is stored at address 00 1EC0h.



Device: CP3UB17

Programmer: CRISP (COM)

```
00001E90 80 2F 64 34 CC 77 04 79 1D 1E DC CC DD C3 7C CE €/d4Ìw□y□□ÜÌÝÃ|Î
00001EA0 18 B0 89 B2 70 33 CC 81 39 78 A6 38 C0 6D 97 6C □°‰²p3Ì□9x¦8Àm—l
00001EB0 26 6F DE 5E 61 D5 6E 9F 51 AC AB CE EF 1D B9 F6 &oÞ^aÕnŸQ¬«Îï□¹ö
00001EC0 52 75 6E 6E 69 6E 67 20 6F 75 74 20 6F 66 20 6D Running out of m
00001ED0 65 6D 6F 72 79 30 41 B3 7F CD DF 69 FC 6F 26 A0 emory0A³□Íßiüo&
00001EE0 5E 0A 72 46 44 6B D6 D0 71 81 46 2F 3C F0 A4 02 ^□rFDkÖÐq□F/<ð¤□
00001EF0 47 39 AE FE 2E 71 EA B1 F7 FA 30 03 6B 98 76 35 G9®þ.qê±÷ú0□k˜v5
00001F00 A6 D4 EA 3B 93 B0 86 A2 10 E5 A5 6C EE 5B 56 8D ¦Ôê;“°†¢□å¥lî[V□
00001F10 29 7B 60 A5 D5 94 39 8B D0 7D 72 B8 25 0A 44 94 ){`¥Õ”9‹Ð}r¸%□D”
00001F20 8F 2E 45 33 C8 BD 89 25 62 36 92 35 53 FE E7 3C □.E3È½‰%b6’5Sþç<
00001F30 9F 04 3B D1 A1 AB 62 25 E3 75 A5 59 98 CA 0C 0F Ÿ□;Ñ¡«b%ãu¥Y˜Ê□□
00001F40 13 D8 B9 1A E4 6E 86 75 65 3B 5C F4 D9 EA 08 5C □Ø¹□än†ue;\ôÙê□\
00001F50 87 A5 86 BC 55 5A FD 59 CE D5 E9 67 B6 74 31 6B ‡¥†¼UZýYÎÕég¶t1k
00001F60 6F EE 20 10 E7 B1 87 A7 D6 91 74 C8 75 C4 4A AC oî □ç±‡§Ö‘tÈuÄJ¬
00001F70 02 F2 2F 3B B2 5A 07 F3 F0 66 83 1C F5 34 F6 E4 □ò/;²Z□óðfƒ□õ4öä
00001F80 28 DB F6 6A DA 89 F9 C1 3C AA 71 81 9F C2 26 61 (ÛöjÚ‰ùÁ<ªq□ŸÂ&a
00001F90 72 B1 C4 FA 85 78 DF B5 72 C2 DA 62 50 CB 86 28 r±Äú…xßµrÂÚbPË†(
00001FA0 03 60 60 B0 C8 0E B0 C0 DB CC 0F A4 52 B1 F5 23 □``°È□°ÀÛÌ□¤R±õ#
00001FB0 82 68 7B DF 99 94 4B 55 3B 59 7D D9 44 91 EE 57 ‚h{ß™”KU;Y}ÙD‘îW
```

● File Load: File loaded (Intel Hex-32|*.hex) - 00:00 | 00:00:07 | 0%

Figure 6-2. ASCII Representation of Data

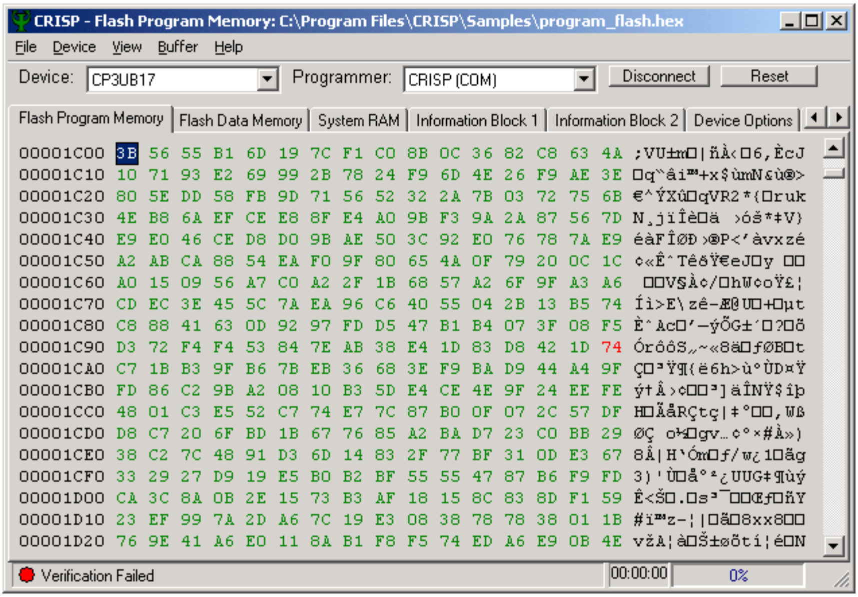
Other CRISP Features

Decoding Hex File Data

If the cursor is held over an individual byte of data in the main window, a pop-up information window is displayed in a yellow box. This window provides detailed information about the individual byte of data, including its address, its value in hexadecimal, decimal, and binary, and, when possible, its equivalent ASCII character.

6.2 Locating Where a Verification Failed

If a byte of data read during a verification operation does not match the data held in the CRISP buffer, "Verification Failed" will be displayed in the status bar, as shown in Figure 6-3.



Device: CP3UB17

Programmer: CRISP (COM)

```
00001C00 3B 56 55 B1 6D 19 7C F1 C0 8B 0C 36 82 C8 63 4A ;VU±m□|ñÀ‹□6,ÈcJ
00001C10 10 71 93 E2 69 99 2B 78 24 F9 6D 4E 26 F9 AE 3E □q"âi™+x$ùmN&ù®>
00001C20 80 5E DD 58 FB 9D 71 56 52 32 2A 7B 03 72 75 6B €^ÝXû□qVR2*{□ruk
00001C30 4E B8 6A EF CE E8 8F E4 A0 9B F3 9A 2A 87 56 7D N¸jïÎè□ä ›óš*‡V}
00001C40 E9 E0 46 CE D8 D0 9B AE 50 3C 92 E0 76 78 7A E9 éàFÎØÐ›®P<’àvxzé
00001C50 A2 AB CA 88 54 EA F0 9F 80 65 4A 0F 79 20 0C 1C ¢«Ê^TêðŸ€eJ□y □□
00001C60 A0 15 09 56 A7 C0 A2 2F 1B 68 57 A2 6F 9F A3 A6  □□V§À¢/□hW¢oŸ£¦
00001C70 CD EC 3E 45 5C 7A EA 96 C6 40 55 04 2B 13 B5 74 Íì>E\zê–Æ@U□+□µt
00001C80 C8 88 41 63 0D 92 97 FD D5 47 B1 B4 07 3F 08 F5 È^Ac□’—ýÕG±´□?□õ
00001C90 D3 72 F4 F4 53 84 7E AB 38 E4 1D 83 D8 42 1D 74 ÓrôôS„~«8ä□ƒØB□t
00001CA0 C7 1B B3 9F B6 7B EB 36 68 3E F9 BA D9 44 A4 9F Ç□³Ÿ¶{ë6h>ùºÙD¤Ÿ
00001CB0 FD 86 C2 9B A2 08 10 B3 5D E4 CE 4E 9F 24 EE FE ý†Â›¢□□³]äÎNŸ$îþ
00001CC0 48 01 C3 E5 52 C7 74 E7 7C 87 B0 0F 07 2C 57 DF H□ÃåRÇtç|‡°□□,Wß
00001CD0 D8 C7 20 6F BD 1B 67 76 85 A2 BA D7 23 C0 BB 29 ØÇ o½□gv…¢º×#À»)
00001CE0 38 C2 7C 48 91 D3 6D 14 83 2F 77 BF 31 0D E3 67 8Â|H‘Óm□ƒ/w¿1□ãg
00001CF0 33 29 27 D9 19 E5 B0 B2 BF 55 55 47 87 B6 F9 FD 3)'Ù□å°²¿UUG‡¶ùý
00001D00 CA 3C 8A 0B 2E 15 73 B3 AF 18 15 8C 83 8D F1 59 Ê<Š□.□s³¯□□Œƒ□ñY
00001D10 23 EF 99 7A 2D A6 7C 19 E3 08 38 78 78 38 01 1B #ï™z-¦|□ã□8xx8□□
00001D20 76 9E 41 A6 E0 11 8A B1 F8 F5 74 ED A6 E9 0B 4E vžA¦à□Š±øõtí¦é□N
```

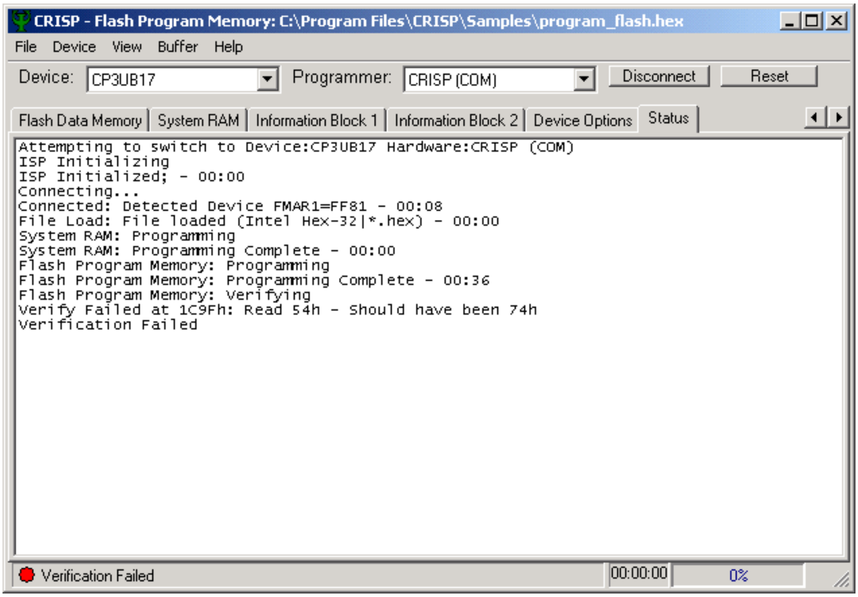
● Verification Failed | 00:00:00 | 0%

Figure 6-3. **Status Bar after a Verification Failure**

Any byte that fails to verify will be highlighted in red in the main window tab for the memory area in which the mismatch was detected. In this case the byte stored at address 00 1C9Fh in the flash program memory has failed verification.

Locating Where a Verification Failed

Selecting the View -> Status command provides more detailed information on the failure. In this case, as shown in Figure 6-4, the byte at address 00 1C9Fh should have been 74h, but actually read as 54h.



```
Attempting to switch to Device:CP3UB17 Hardware:CRISP (COM)
ISP Initializing
ISP Initialized; - 00:00
Connecting...
Connected: Detected Device FMAR1=FF81 - 00:08
File Load: File loaded (Intel Hex-32|*.hex) - 00:00
System RAM: Programming
System RAM: Programming Complete - 00:00
Flash Program Memory: Programming
Flash Program Memory: Programming Complete - 00:36
Flash Program Memory: Verifying
Verify Failed at 1C9Fh: Read 54h - Should have been 74h
Verification Failed
```

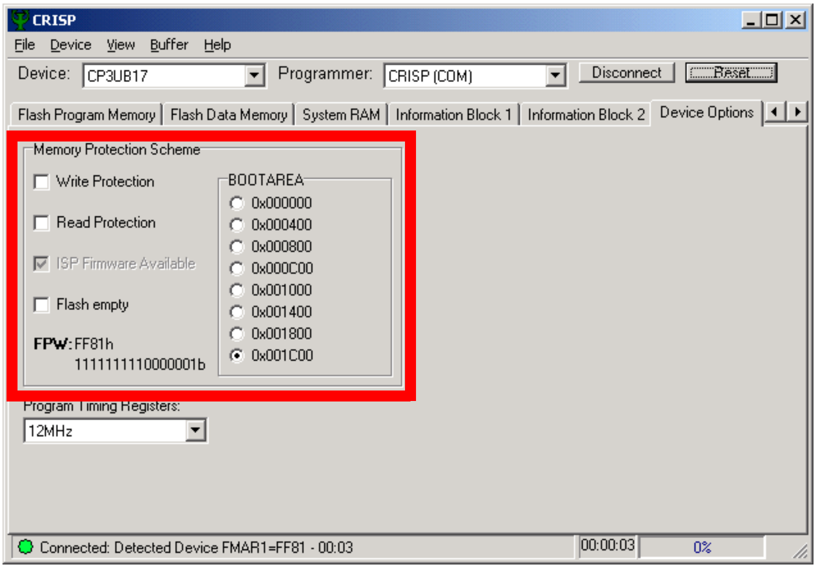
Figure 6-4. Further Details on the Failing Byte in the Status Tab Window

6.3 Programming the Protection Word

The Protection Word is used to enable or disable various flash memory options, such as read or write access to the memory. The following procedure programs the Protection Word.

Before programming the Protection Word, read the device data sheet about the features controlled by the Protection Word. You MUST understand these features before programming the Protection Word, because some settings are irreversible and may make the device unusable. These features are fully discussed in the device data sheet, but not in this document.

1. **View the Memory Protection Scheme.** In the CRISP main window, click on the Device Options tab. On the left, the Memory Protection Scheme settings are displayed, as highlighted in red in Figure 6-5.



Device: CP3UB17

Programmer: CRISP (COM)

Memory Protection Scheme

- □ Write Protection
- □ Read Protection
- ☑ ISP Firmware Available
- □ Flash empty

FPW: FF81h
1111111110000001b

BOOTAREA

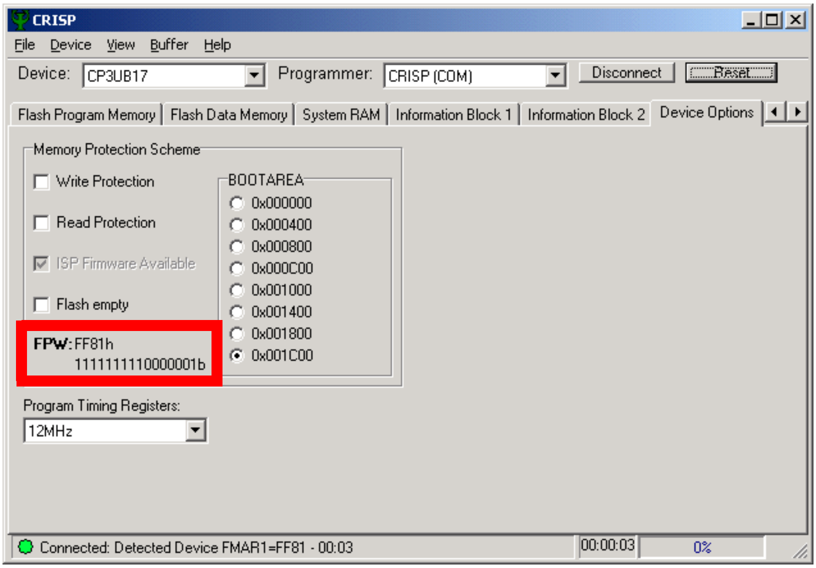
- ○ 0x000000
- ○ 0x000400
- ○ 0x000800
- ○ 0x000C00
- ○ 0x001000
- ○ 0x001400
- ○ 0x001800
- ● 0x001C00

Program Timing Registers: 12MHz

Connected: Detected Device FMAR1=FF81 - 00:03 | 00:00:03 | 0%

Figure 6-5. Memory Protection Scheme Settings

2. **View Flash Protection Word Value.** The BOOTAREA radio buttons, and Write Protection, Read Protection, ISP Firmware Available, and Flash empty check boxes are used to display and change the Flash Protection Word programming. The value which will be programmed into the Flash Protection Word is indicated at the bottom left hand corner of the Memory Protection Scheme area in both binary and hexadecimal, as shown in Figure 6-6.



Device: CP3UB17

Programmer: CRISP (COM)

Memory Protection Scheme

☐ Write Protection

☐ Read Protection

☑ ISP Firmware Available

☐ Flash empty

FPW: FF81h
1111111110000001b

BOOTAREA

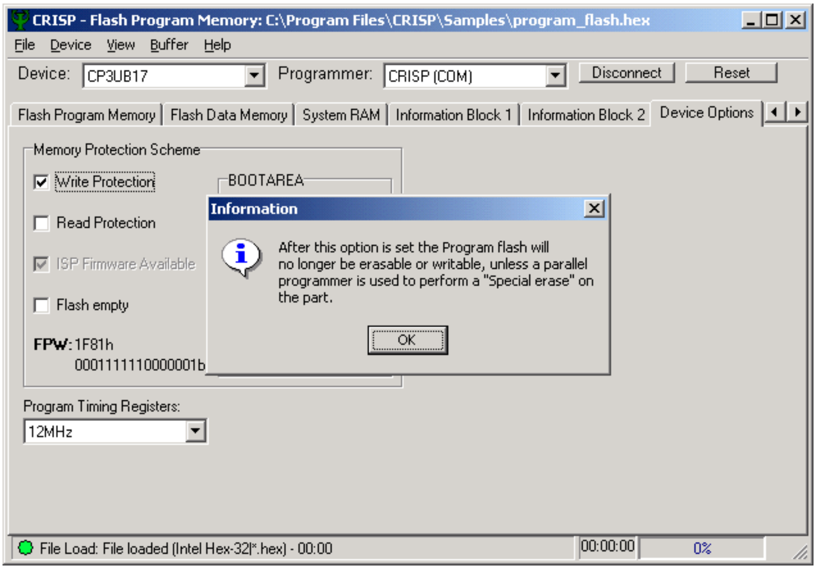
○ 0x000000
○ 0x000400
○ 0x000800
○ 0x000C00
○ 0x001000
○ 0x001400
○ 0x001800
● 0x001C00

Program Timing Registers: 12MHz

Connected: Detected Device FMAR1=FF81 - 00:03 | 00:00:03 | 0%

Figure 6-6. Flash Protection Word Value

3. **Write Protection Warning.** Some settings are irreversible and may make the device permanently unusable. For example, write protection cannot be disabled after it is enabled (unless a special programmer is used). In Figure 6-7, selecting the Write Protection setting has opened a pop-up dialog box to warn the user.



CRISP - Flash Program Memory: C:\Program Files\CRISP\Samples\program_flash.hex

Device: CP3UB17

Programmer: CRISP (COM)

Memory Protection Scheme

☑ Write Protection

□ Read Protection

☑ ISP Firmware Available

□ Flash empty

FPW: 1F81h
0001111110000001b

Program Timing Registers: 12MHz

Information

After this option is set the Program flash will no longer be erasable or writable, unless a parallel programmer is used to perform a "Special erase" on the part.

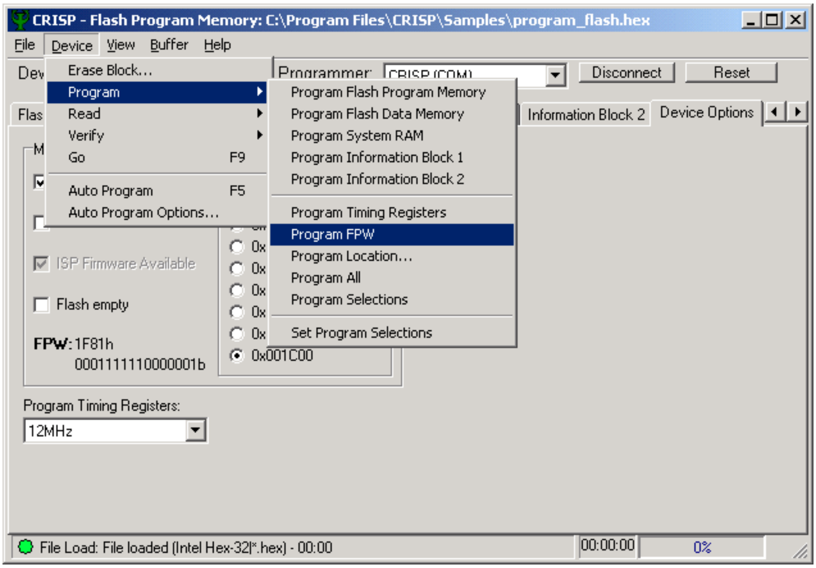
File Load: File loaded (Intel Hex-32|*.hex) - 00:00

00:00:00

0%

Figure 6-7. Write Protection Warning

4. **Initiate Programming the Protection Word.** When the desired settings have been selected, programming can be initiated by selecting the Device -> Program -> Program FPW command, as shown in Figure 6-8.


CRISP - Flash Program Memory: C:\Program Files\CRISP\Samples\program_flash.hex

Programmer: CRISP (COM)

☑ ISP Firmware Available

□ Flash empty

FPW: 1F81h
0001111110000001b

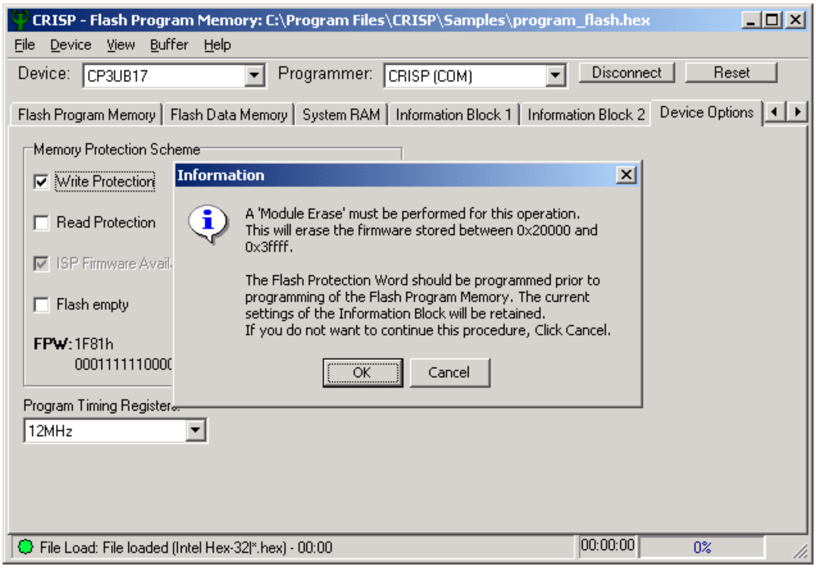
● 0x001C00

Program Timing Registers: 12MHz

File Load: File loaded (Intel Hex-32|*.hex) - 00:00 | 00:00:00 | 0%

Figure 6-8. Selecting the Program FPW Command

5. **Program FPW Warning.** After the Program FPW command has been selected, a further dialog box is opened to confirm the selection. It explains that changing the Protection Word settings also erases the memory from 02 0000h to 03 FFFFh. The dialog box also gives the user the option to continue or cancel the programming the operation, as shown in Figure 6-9. **Note:** The Protection Word settings do not take effect until **after** a device reset or power to the device has been cycled.



Device: CP3UB17
Programmer: CRISP (COM)

☑ Write Protection
□ Read Protection
☑ ISP Firmware Avail
□ Flash empty

FPW: 1F81h
000111111000(

Program Timing Register: 12MHz

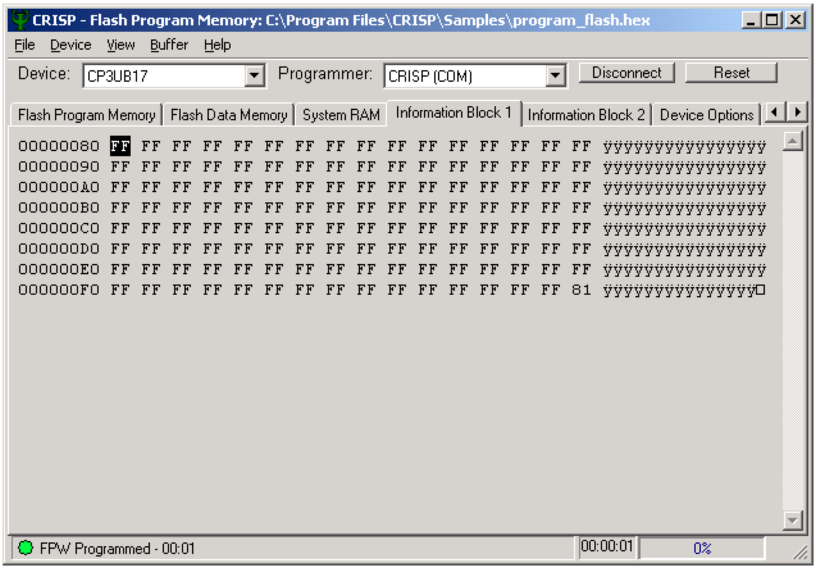
A 'Module Erase' must be performed for this operation.
This will erase the firmware stored between 0x20000 and 0x3ffff.

The Flash Protection Word should be programmed prior to programming of the Flash Program Memory. The current settings of the Information Block will be retained.
If you do not want to continue this procedure, Click Cancel.

File Load: File loaded (Intel Hex-32|*.hex) - 00:00 | 00:00:00 | 0%

Figure 6-9. Protection Word Programming Warning

6. **Confirm Programming.** Click the Ok button to confirm the command to program the new Protection Word settings into the target device. If the operation is successful, the status bar circle will turn green, and the message FPW Programmed will be displayed, as shown in Figure 6-10.



Device: CP3UB17

Programmer: CRISP (COM)

```
00000080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 81 ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ□
```

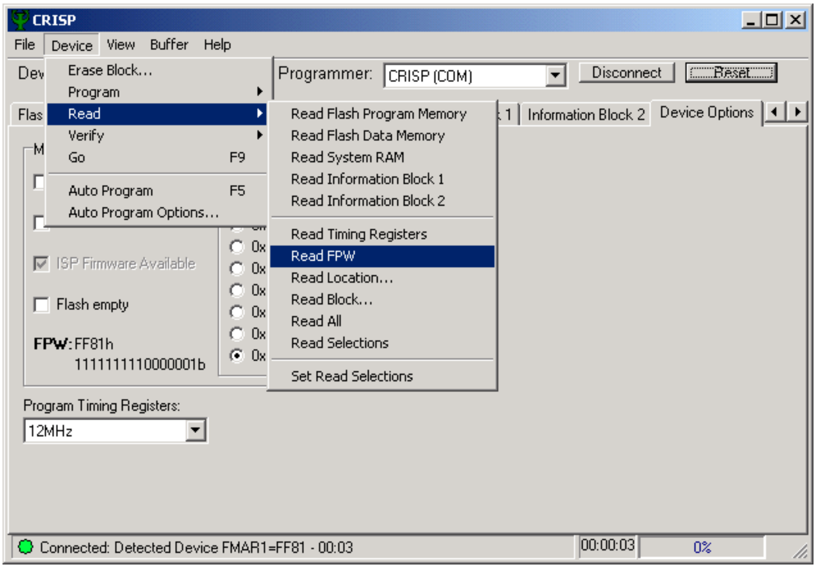
● FPW Programmed - 00:01 | 00:00:01 | 0%

Figure 6-10. Successful Completion of Protection Word Programming

6.4 Reading the Protection Word Settings

The following procedure reads the current Protection Word settings.

1. **Initiate Protection Word Read.** Select the Device -> Read -> Read FPW command, as shown in Figure 6-11.



Programmer: CRISP (COM)

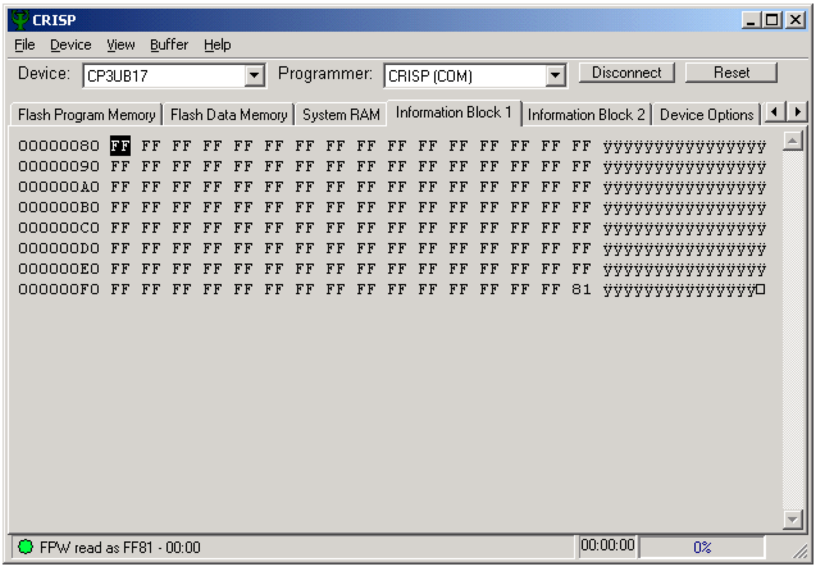
FPW: FF81h
1111111110000001b

Program Timing Registers: 12MHz

Connected: Detected Device FMAR1=FF81 - 00:03 | 00:00:03 | 0%

Figure 6-11. Selecting the Read FPW Command

2. **View the Protection Word Settings.** When the Protection Word has been read, its value is displayed in the status bar, as shown in Figure 6-12.



```
00000080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF 81 ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ□
```

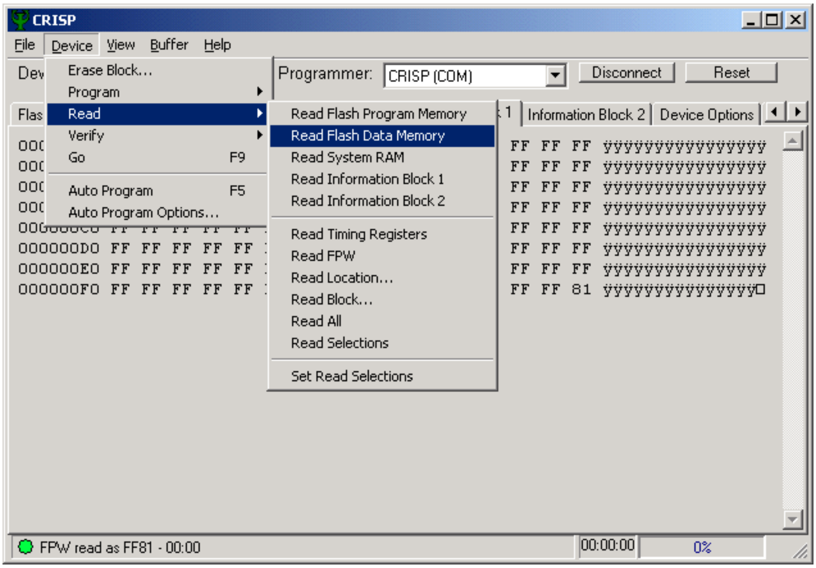
● FPW read as FF81 - 00:00 | 00:00:00 | 0%

Figure 6-12. Protection Word Settings Displayed in Status Bar

6.5 Uploading the Memory Contents

The contents of any of the memory areas can be uploaded into CRISP memory buffers. The following procedure uploads the contents of the flash data memory into a CRISP buffer, then saves the buffer to a disk file.

1. **Initiate Flash Data Memory Upload.** After entering CRISP, the default contents of the Flash Data Memory buffer are empty (all bytes FFh). Select the Device -> Read -> Read Flash Data Memory command initiates an upload from the target device, as shown in Figure 6-13.



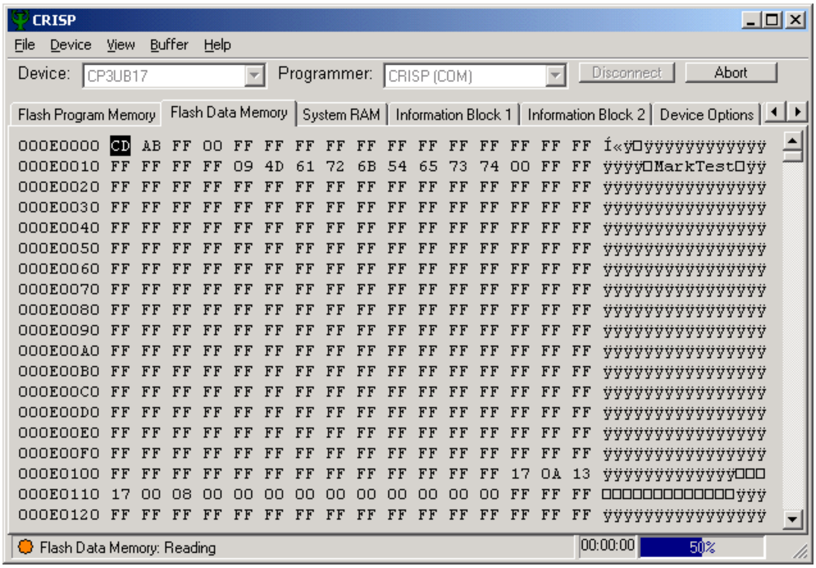
```
000000D0  FF  FF  FF  FF  FF
000000E0  FF  FF  FF  FF  FF
000000F0  FF  FF  FF  FF  FF

FF  FF  FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
FF  FF  FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
FF  FF  FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
FF  FF  FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
FF  FF  FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
FF  FF  FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
FF  FF  FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
FF  FF  81  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ□
```

FPW read as FF81 - 00:00

Figure 6-13. **Initiating a Flash Data Memory Upload**

2. **Monitor the Upload.** After selecting the Read Flash Data Memory command, CRISP will initiate the reading process, uploading from the target device into the Flash Data Memory buffer of CRISP. During this operation, the status bar will look as shown in Figure 6-14.



Device: CP3UB17

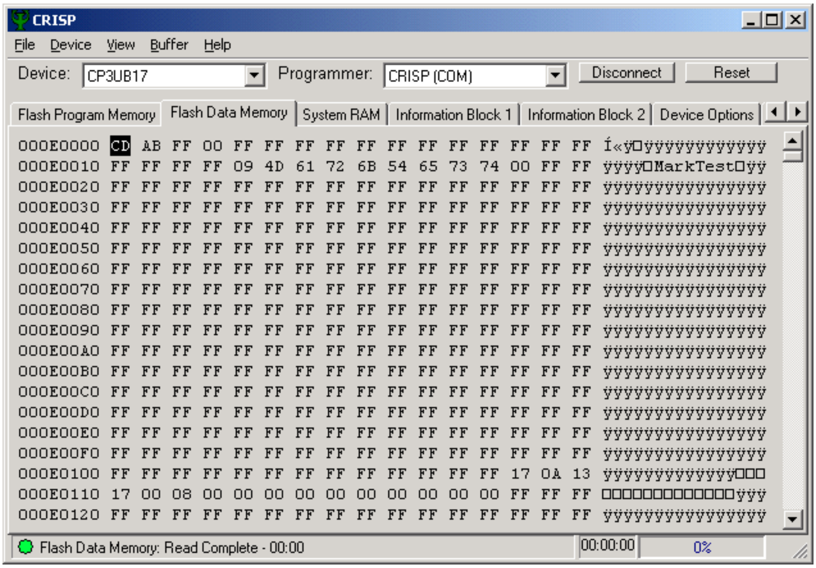
Programmer: CRISP (COM)

```
000E0000 CD AB FF 00 FF FF FF FF FF FF FF FF FF FF FF FF Í«ÿ□ÿÿÿÿÿÿÿÿÿÿÿÿ
000E0010 FF FF FF FF 09 4D 61 72 6B 54 65 73 74 00 FF FF ÿÿÿÿ□MarkTest□ÿÿ
000E0020 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0030 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0100 FF FF FF FF FF FF FF FF FF FF FF FF FF 17 0A 13 ÿÿÿÿÿÿÿÿÿÿÿÿÿ□□□
000E0110 17 00 08 00 00 00 00 00 00 00 00 00 00 FF FF FF □□□□□□□□□□□□□ÿÿÿ
000E0120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

Flash Data Memory: Reading 00:00:00 50%

Figure 6-14. Status Bar During a Memory Upload from the Target Device

3. **Examine CRISP Buffer.** After the memory has been read and uploaded into the CRISP buffer, the contents of the upload are displayed in the main CRISP window. Scroll the window to examine the higher memory locations. The uploaded data is shown in Figure 6-15.



Device: CP3UB17

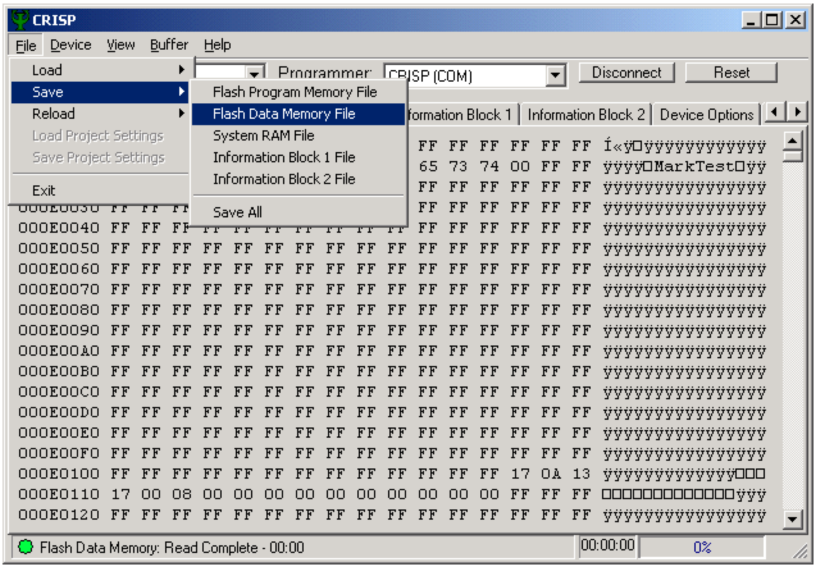
Programmer: CRISP (COM)

```
000E0000 CD AB FF 00 FF FF FF FF FF FF FF FF FF FF FF FF Í«ÿ□ÿÿÿÿÿÿÿÿÿÿÿÿ
000E0010 FF FF FF FF 09 4D 61 72 6B 54 65 73 74 00 FF FF ÿÿÿÿ□MarkTest□ÿÿ
000E0020 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0030 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0100 FF FF FF FF FF FF FF FF FF FF FF FF FF 17 0A 13 ÿÿÿÿÿÿÿÿÿÿÿÿÿ□□□
000E0110 17 00 08 00 00 00 00 00 00 00 00 00 00 FF FF FF □□□□□□□□□□□□□ÿÿÿ
000E0120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

● Flash Data Memory: Read Complete - 00:00 | 00:00:00 | 0%

Figure 6-15. CRISP Window Displaying Uploaded Data

4. **Initiate Saving Buffer to Disk File.** The CRISP buffer can be saved to a disk file by selecting the File -> Save -> Flash Data Memory File command, as shown in Figure 6-16.

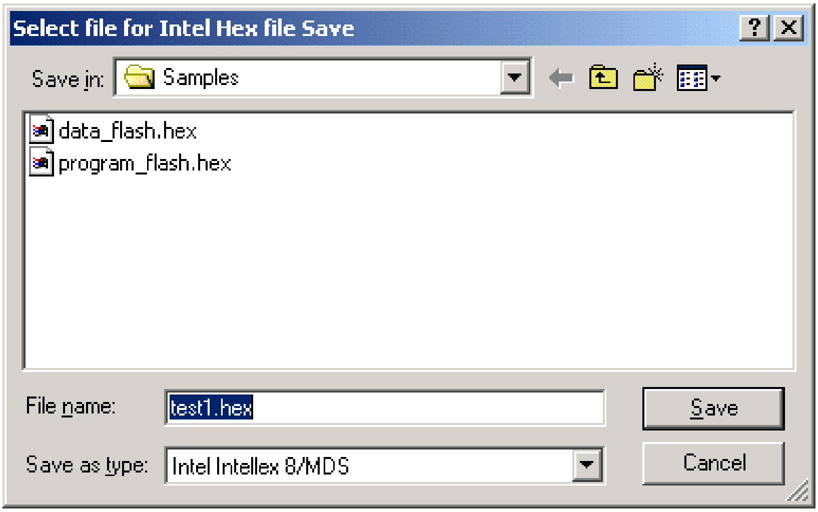


```
                                       FF FF FF FF FF FF Í«ÿ□ÿÿÿÿÿÿÿÿÿÿÿÿ
                                       65 73 74 00 FF FF ÿÿÿÿ□MarkTest□ÿÿ
                                       FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0030 FF FF FF                      FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0100 FF FF FF FF FF FF FF FF FF FF FF FF FF 17 0A 13 ÿÿÿÿÿÿÿÿÿÿÿÿÿ□□□
000E0110 17 00 08 00 00 00 00 00 00 00 00 00 00 FF FF FF □□□□□□□□□□□□□ÿÿÿ
000E0120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

● Flash Data Memory: Read Complete - 00:00 | 00:00:00 | 0%

Figure 6-16. **Selecting the Save Command**

5. **Specifying Disk File Name.** After selecting a buffer under the Save command, a dialog box will open for specifying the folder and file name, as shown in Figure 6-17. Files are stored, by default, in Intel Hex format.



Select file for Intel Hex file Save

Save in: Samples

data_flash.hex
program_flash.hex

File name: test1.hex

Save as type: Intel Intellex 8/MDS

Figure 6-17. Specifying Folder and File for Saving CRISP Buffer

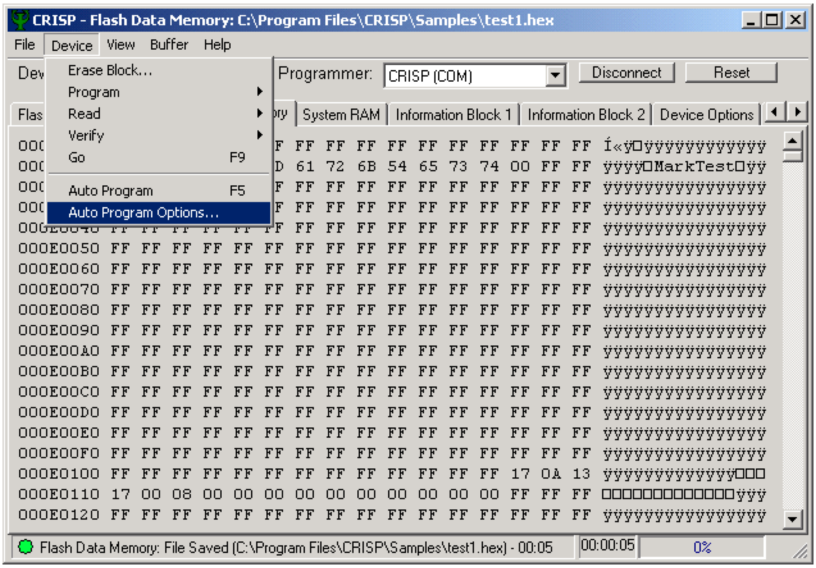
6.6 Using the Auto Program Mode

The Auto Program feature is useful in mass-programming environments, in which many devices are programmed in the same way. This feature can be tailored to the needs of the user, to allow defining a "single click" programming process. CRISP can be set up to automatically program four memory areas: Program Flash Memory, Data Flash Memory, System RAM, and Information Block 1. There are three steps that can be included in the Auto Programming sequence:

- **Reload Files**—selecting this option causes the Intel Hex file for the selected memory buffers to be reloaded into CRISP before each programming sequence. This feature could be used to maintain the latest code release or to program serial numbers into the Data Flash Memory, if a suitable environment were developed.
- **Program Memory**—selecting this option enables the programming part of the sequence and allows the user to specify which memory areas will be programmed.
- **Verify Memory**—selecting this option enables verification of the selected memory areas.

The following example assumes that the Intel Hex file **program_flash.hex** has been loaded into the Flash Program Memory buffer, and the file **data_flash.hex** has been loaded into the Flash Data Memory buffer.

1. **Enter Auto Program Options Mode.** Before Auto Program mode can be used, the operations to occur in this "single click" programming mode must be defined. To define these operations, select the Device -> Auto Program Options command, as shown in Figure 6-18.

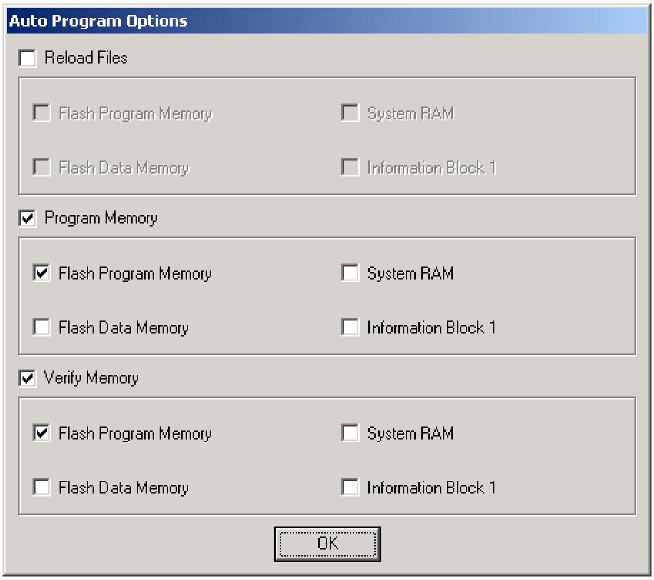


```
000(                    F FF FF FF FF FF FF FF FF FF FF  Í«ÿ□ÿÿÿÿÿÿÿÿÿÿÿÿ
000(                    D 61 72 6B 54 65 73 74 00 FF FF  ÿÿÿÿ□MarkTest□ÿÿ
000(                    F FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000(                    F FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0100 FF FF FF FF FF FF FF FF FF FF FF FF FF 17 0A 13  ÿÿÿÿÿÿÿÿÿÿÿÿÿ□□□
000E0110 17 00 08 00 00 00 00 00 00 00 00 00 00 FF FF FF  □□□□□□□□□□□□□ÿÿÿ
000E0120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

Flash Data Memory: File Saved (C:\Program Files\CRISP\Samples\test1.hex) - 00:05 | 00:00:05 | 0%

Figure 6-18. Selecting the Auto Program Options Command

2. **Review Auto Program Options.** The dialog box that opens shows the current "Auto Program" settings. By default, programming and verifying are enabled for the Program Flash Memory, as shown in Figure 6-19.



Auto Program Options

□ Reload Files

□ Flash Program Memory □ System RAM

□ Flash Data Memory □ Information Block 1

☑ Program Memory

☑ Flash Program Memory □ System RAM

□ Flash Data Memory □ Information Block 1

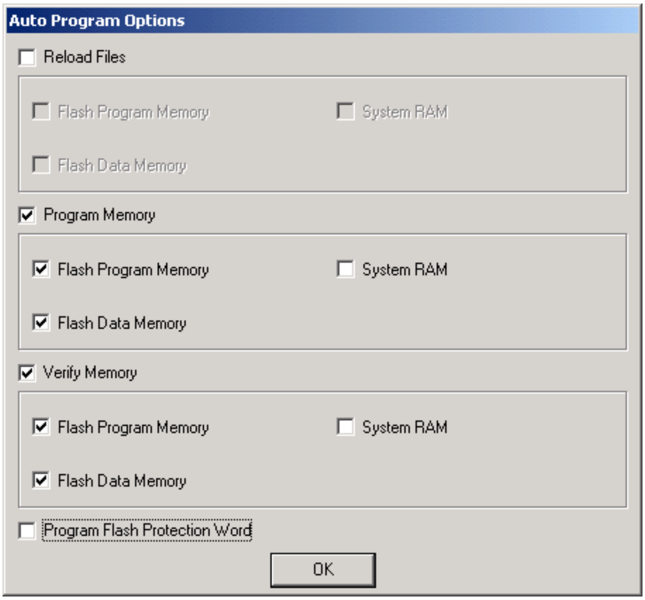
☑ Verify Memory

☑ Flash Program Memory □ System RAM

□ Flash Data Memory □ Information Block 1

Figure 6-19. **Auto Program Options Dialog Box**

3. **Select Auto Program Options.** In this example, there is no need to reload the Intel Hex files each time a device is programmed, so this option is left disabled. However, we do want to program and verify both the Flash Program Memory and the Flash Data Memory memories, so these options must be checked. We do not want to modify the contents of the System RAM or Information Block 1, so these options remain unchecked. The final settings for the Auto Program Options dialog box are shown in Figure 6-20. Click the OK button to proceed.



Auto Program Options

☐ Reload Files

☐ Flash Program Memory ☐ System RAM

☐ Flash Data Memory

☑ Program Memory

☑ Flash Program Memory ☐ System RAM

☑ Flash Data Memory

☑ Verify Memory

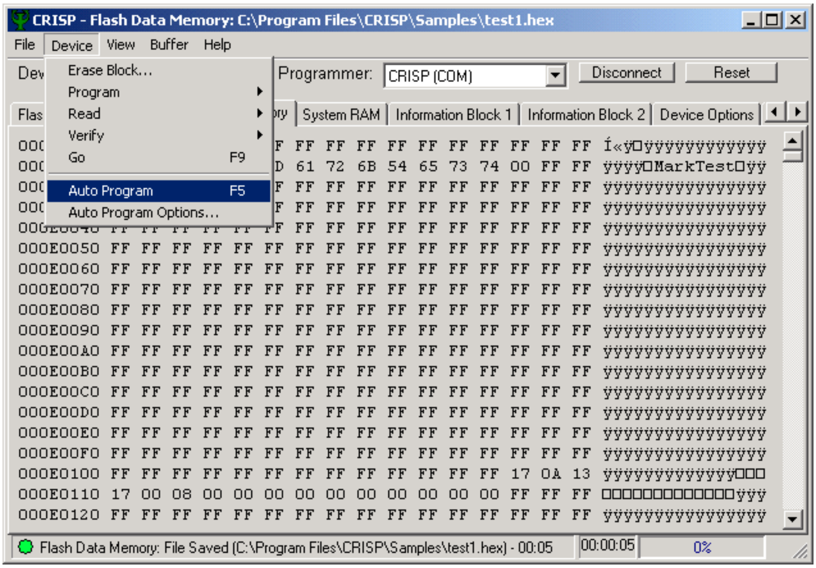
☑ Flash Program Memory ☐ System RAM

☑ Flash Data Memory

☐ Program Flash Protection Word

Figure 6-20. **Auto Program Settings used in this Example**

4. **Invoke "Single Click" Programming.** Select the Device -> Auto Program command, as shown in Figure 6-21.

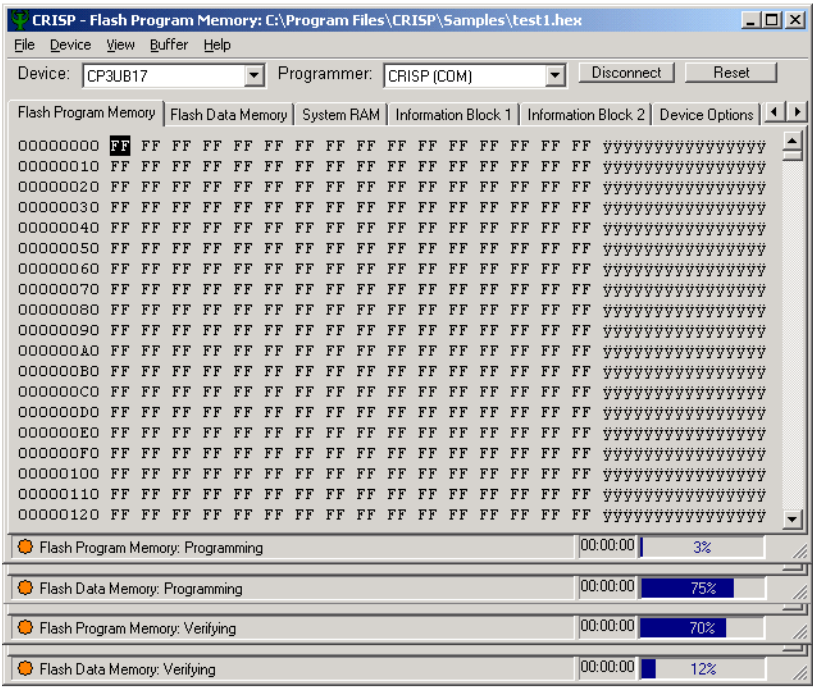


```
00C                   F FF FF FF FF FF FF FF FF FF FF  Í«ÿ□ÿÿÿÿÿÿÿÿÿÿÿÿ
00C                   D 61 72 6B 54 65 73 74 00 FF FF  ÿÿÿÿ□MarkTest□ÿÿ
00C                   F FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00C                   F FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00C                   F FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E00F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000E0100 FF FF FF FF FF FF FF FF FF FF FF FF FF 17 0A 13  ÿÿÿÿÿÿÿÿÿÿÿÿÿ□□□
000E0110 17 00 08 00 00 00 00 00 00 00 00 00 00 FF FF FF  □□□□□□□□□□□□□ÿÿÿ
000E0120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

Flash Data Memory: File Saved (C:\Program Files\CRISP\Samples\test1.hex) - 00:05 | 00:00:05 | 0%

Figure 6-21. Selecting the Auto Program Command

5. **Monitor Programming Operation.** The current status of the programming is indicated in the status bar. Throughout the Auto Program sequence, the operation being executed and the target memory area are displayed, while the progress bar increments to display what percentage of the current operation has been completed, as shown in Figure 6-22.



Device: CP3UB17

Programmer: CRISP (COM)

```
00000000 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000010 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000020 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000030 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000100 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000110 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

● Flash Program Memory: Programming 00:00:00 3%

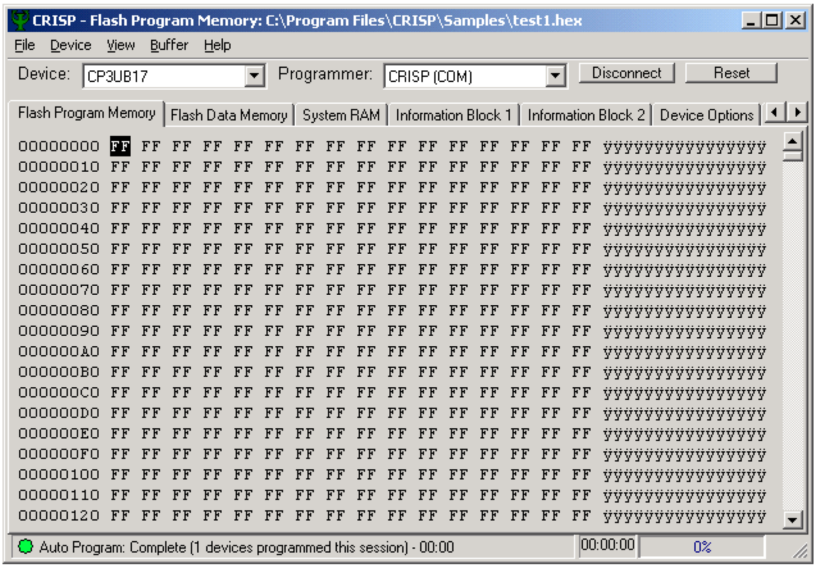
● Flash Data Memory: Programming 00:00:00 75%

● Flash Program Memory: Verifying 00:00:00 70%

● Flash Data Memory: Verifying 00:00:00 12%

Figure 6-22. **Status Bar During Auto Program Mode**

6. **Auto Program Complete.** On successful completion of the Auto Program sequence, the status bar circle turns green and the status bar displays the message Auto Program: Complete. It also displays how many devices have been programmed during this session, as shown in Figure 6-23.



Device: CP3UB17

Programmer: CRISP (COM)

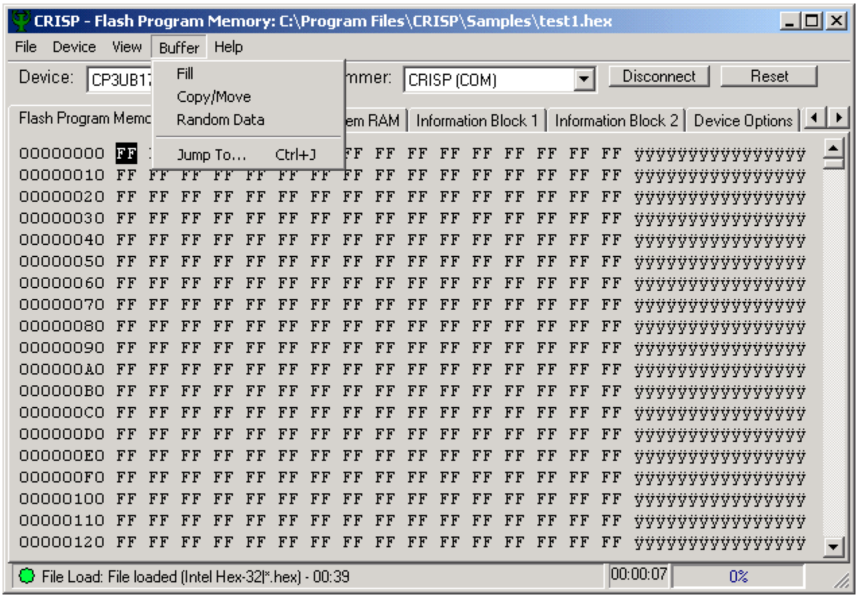
```
00000000 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000010 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000020 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000030 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000100 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000110 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

● Auto Program: Complete (1 devices programmed this session) - 00:00 | 00:00:00 | 0%

Figure 6-23. Successful Auto Programming

6.7 Buffer Editing Functions

CRISP includes some features which allow simple editing operations on the memory buffers. These operations are invoked by the commands on the Buffer menu. These commands are shown in Figure 6-24.



Device: CP3UB17

mmer: CRISP (COM)

```
00000000 FF :                      FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000010 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000020 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000030 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000040 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000060 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000070 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000080 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000090 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000A0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000D0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000E0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
000000F0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000100 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000110 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
00000120 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
```

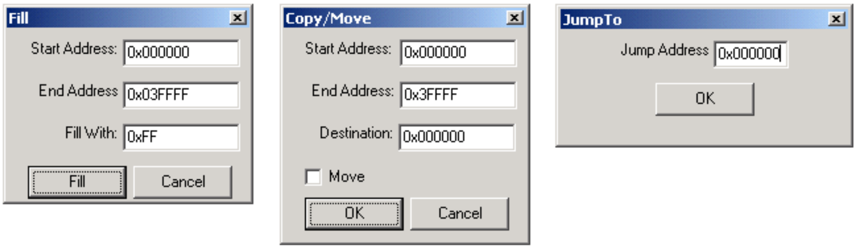
File Load: File loaded (Intel Hex-32|*.hex) - 00:39 | 00:00:07 | 0%

Figure 6-24. Buffer Menu Commands

Individual bytes in a buffer can also be modified by clicking on the chosen byte in the main window to highlight it, and then typing a new value into that location using the keyboard.

The editing operations on the Buffer menu are:

- **Fill**—fills a specified address range with a single hexadecimal value. The start address, end address, and fill value are entered in a dialog box.
- **Copy/Move**—copies or moves a block of data within the current memory buffer. The source start address, source end address, and destination start address are specified in a dialog box. A check box selects between moving or copying the data. The move operation fills the source region with FFh, while the copy operation leaves the source region unchanged.
- **Random Data**— fills the currently selected memory buffer with random data.
- **Jump To**—jumps to a specified address in the memory buffer, as a faster alternative to scrolling over a large address range.



Start Address: 0x000000
End Address 0x03FFFF
Fill With: 0xFF

Start Address: 0x000000
End Address: 0x3FFFF
Destination: 0x000000
□ Move

Jump Address 0x000000

Figure 6-25. **DialogBoxes Used by Buffer Commands**

6.8 Device -> Read Commands

CRISP also supports some other read operations under the Device -> Read commands. These operations are:

- **Read Timing Registers**—this command reads the values in the flash timing registers and compares them against a database of timing values to determine which clock speed is currently selected. The clock speed is displayed in the status bar.
- **Read Location**—this command opens a dialog box in which the user can perform byte or word reads on specific memory locations, including register locations.
- **Read Block**—this command opens a dialog box which allows the user to read a chosen block of data. The data will be read in 32-byte blocks.
- **Set Read Selections**—this command opens a dialog box which allows the user to select and automatically upload Program Flash Memory, Data Flash Memory, System RAM, and Information Block 1.

Other CRISP Features

Device -> Read Commands

7 Compiling for ISP Download

Compiling for download through the ISP interface differs from compiling for download through winIDEA in two ways:

- The application must be loaded beginning at the **BOOTAREA** address, rather than beginning at address 0. The region from 0 to **BOOTAREA** - 1 in program memory space is reserved for the ISP software. The **BOOTAREA** address must be set at 1C00h, to allow enough memory space for the ISP code.
- The output file from the compiler tool chain must be an Intel Extended Hex format file (**.hex**), not a **.dbg** file.

Detailed procedures for compiling programs using the winIDEA integrated development environment are given in the **CP3CN17/ CP3UB17 Evaluation Kit User's Guide**.

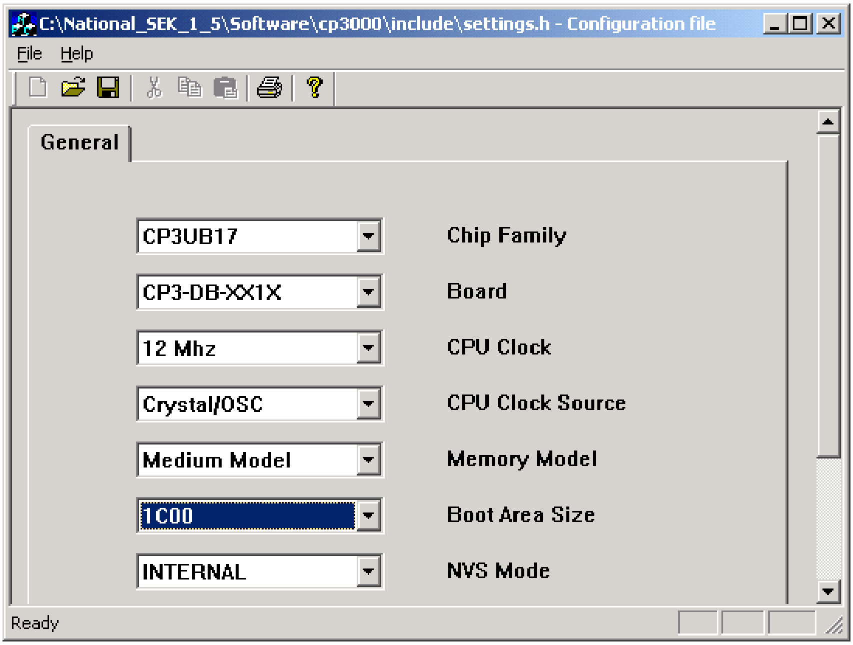
7.1 Reserving the BOOTAREA Space

The Configurator utility (**ConfigTool.exe**) is used to specify the size of the program memory space reserved for the ISP software. This utility is located in the **C:\National_SEK_X_Y\software\tools\bin** folder (in which X.Y is the version number of the software release). This program generates the **settings.h** and **settings.mak** files in the **C:\National_SEK_X_Y\software\cp3000\include** folder.

Compiling for ISP Download

Reserving the BOOTAREA Space

To reserve space for the ISP software, the **ConfigTool.exe** program is executed with the Boot Area Size set to 1C00.



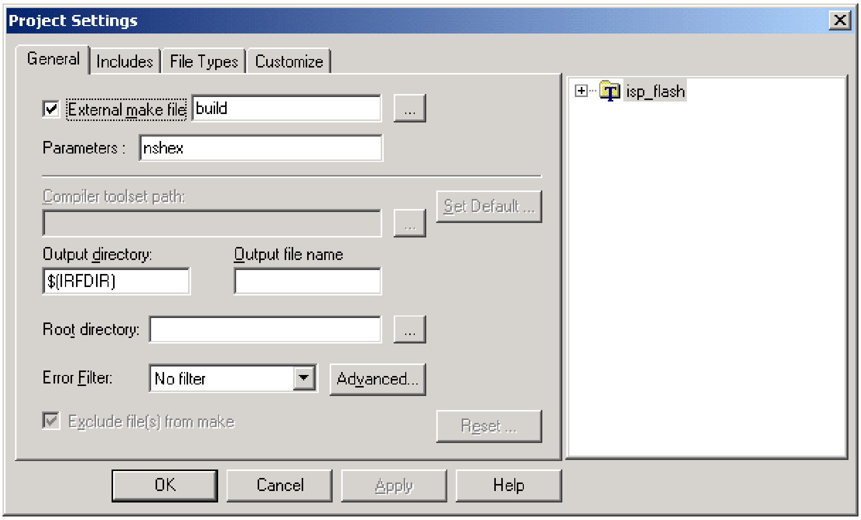
C:\National_SEK_1_5\Software\cp3000\include\settings.h - Configuration file
File
Help
General
Ready

CP3UB17	Chip Family
CP3-DB-XX1X	Board
12 Mhz	CPU Clock
Crystal/OSC	CPU Clock Source
Medium Model	Memory Model
1C00	Boot Area Size
INTERNAL	NVS Mode

After generating the new **settings.h** and **settings.mak** files, any compilations will generate application code that will be loaded above the section reserved for the ISP software.

7.2 Generating a Hex File

The compiled software must be generated in Intel Extended Hex format. To select this format in winIDEA, select the Project -> Settings command. In the Parameters box, enter **nshex** when using the National C compiler tool chain or **iarhex** when using the IAR Systems tool chain.



☑ External make file: build

Parameters : nshex

Compiler toolset path:

Output directory: $(IRFDIR)

Output file name:

Root directory:

Error Filter: No filter

☑ Exclude file(s) from make

When compiling with either of these settings, the output file will be a **.hex** file, rather than the usual **.dbg** file.

Notes

National Semiconductor
2900 Semiconductor Drive
PO Box 58090
Santa Clara, CA 95052

Tel: 1-800-272-9959
Fax: 1-800-737-7018

Visit our Web site at:
www.national.com/appinfo/cp3000

For more information, send Email to:
support@nsc.com

National Semiconductor
Europe

Fax:	+49 (0) 180-530 85 86
Email:	europe.support@nsc.com
Deutsch Tel:	+49 (0) 69 9508 6208
English Tel:	+44 (0) 870 24 0 2171
Francais Tel:	+33 (0) 1 41 91 8790

National Semiconductor
Asia Pacific
Customer Response Group
Tel: 65-254-4466
Fax: 65-250-4466
Email: ap.support@nsc.com

National Semiconductor
Japan Ltd.
Tel: 81-3-5639-7560
Fax: 81-3-5639-7507

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied, and National reserves the right, at any time without notice, to change said circuitry or specifications.