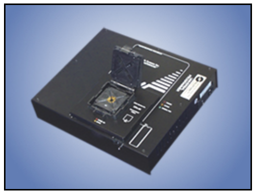
Advin Systems Inc.

PILOT
DEVICE PROGRAMMERS
User's Manual



www.Advin.com

Advin Systems Inc., 556 Weddell Drive, #8, Sunnyvale, CA 94089

(408) 243-7000   FAX: (408) 541-9006

Document Number 21140-V172

Legal Notices

The information contained in this manual has been carefully checked and is believed to be accurate and complete at the time of printing. However, no responsibility is assumed for errors that might appear. Advin Systems Inc. reserves the right to make any changes to the product and/or the manual at any time without notice.

Advin Systems Inc. assumes no liability arising out of the use or application of any of its products.

No part of this document may be copied or reproduced in any form or by any means without prior written consent of Advin Systems Inc.

Life Support Policy

Advin's products are not authorized for use as critical components or programming of critical components in life support devices or systems and the use of such implies that user bears all risk of such use.

Warranty Information

The product is guaranteed against defects in material or workmanship and guaranteed to meet specifications in effect at the time of manufacture for a period of **one year** from date of delivery. If it should become necessary to return a product for service or repair within the warranty period, contact the factory first for return authorization. The returned item will either be repaired or replaced. There are no other warranties, expressed or implied. The liability of Advin Systems Inc. is limited to the purchase price of the product, and does not cover any lost profits, consequential damages, or any claim against the purchaser by any party.

Trademarks:
PILOT, PILOT-xxx, U44-Plus, U84-Plus, U128-Plus, MVP are trademarks of Advin Systems Inc.
IBM, IBM PC are registered trademarks of IBM Corporation
MICROSOFT WINDOWS is a registered trademark of MICROSOFT Corporation.
All rights reserved. Printed in U.S.A.

This manual was last revised on: Nov 1, 2004

\Man\WinMan\User-Manual-0411C-44.doc

TABLE OF CONTENTS

PREFACE

Welcome to the world of PILOT programming instruments. PILOT programmers are designed to be easy to use.

Though it is not really necessary to read the manual in order to use a PILOT programmer, it is advised that you should read through it before you begin any serious work. If you don't, you may miss important information.

This manual applies to the following PILOT programmer models:

PILOT-U128+
PILOT-U84+
PILOT-U44+
PILOT-MVP
PILOT-1600
PILOT-146

PROM

The word "PROM" is used throughout the manual as a convenient way to represent a semiconductor chip, because historically, chip programmers or device programmers were called PROM programmers.

FEEDBACKS

We welcome feedbacks. If you see anything that is not explained well in this document, or if you see mistakes, please do not hesitate to let us know. You can email us at **Support.Advin@Gmail.com**. We show our appreciation to customers who are the first to tell us errors or improvements with nice company ceramic coffee mugs. Please include your address so that we can send one to you.

1.0 GENERAL INFORMATION

PRODUCT OVERVIEW

PILOT programmers are controlled by IBM PCs or compatible computers. The control software that runs on the PCs is called Captain. You can install Captain from the supplied CD or from downloads at www.Advin.com.

SYSTEM REQUIREMENTS

Minimum System Requirements are:

- A PC with a 400MHz or faster processor
- Microsoft Windows 98/NT/2000/ME/XP
- 128MB RAM
- 20 MB of hard disk space
- CD-ROM drive (for installation)
- Printer Port: One parallel printer port

Programming pulse lengths will be within specification limits and is independent of computer speed. However, faster machines will result in reduced over-head and better programming speeds and throughputs.

2.0 INSTALLATION

A PILOT programmer is designed to operate as a slave unit to a PC through a standard parallel printer port. Parallel ports installed as LPT1, LPT2, or LPT3 will all work.

2.1 HARDWARE INSTALLATION

Each PILOT programmer comes with a five-foot 25-pin interface cable. This cable is for connecting the programmer to a parallel port (not a serial port) on the PC. Install hardware in the following sequence:

1. Make sure the programmer power switch is off.

2. There is no need to change AC voltage setting in the programmer. It has a built-in power supply that will automatically sense and adjust to any AC voltages between 110 and 230 volts. That is, you do not have to change anything inside the programmer if you take your programmer to other countries outside the United States.

3. Connect the female end of the 25-pin interface cable to the programmer.

4. Connect the other end of the cable to any parallel printer port on the PC. You can use LPT1, LPT2, or LPT3.

5. Connect power cord from programmer to power source.

6. Install add-on modules, if any, as follows:

   All add-on modules that have 50-pin connectors are installed on top of the programmer by plugging into the 50-pin expansion port that is **closest** to the ZIF socket.

   After a module is installed, its silk-screened legend on the module should look right side up to the user, and not up side down.

7. Power up the system by pressing the end of the power switch marked "1". The red LED shall come on within two seconds.

2.2 SOFTWARE INSTALLATION

The shipped CD comes with four major pieces of software:

Install-Captain98-vXXX.exe
Install-CaptainNT-vXXX.exe
Install-NTdriver.exe
AdvinDOS

For Windows 98 users:

To install the Windows-based software **Captain**, all you need to do is to click on: **Install-Captain98-vXXX.exe**, where XXX is the software version number.

Follow the on-screen instructions to complete installation.

For Windows NT/2000/ME/XP users:

You need to install the NT driver by clicking on:
Install-NTdriver.exe

You need to restart you computer for the NT driver to take effect. Then you need to click on:
Install-CaptainNT-vXXX.exe, where XXX is the software version number.

Follow the on-screen instructions to complete installation.

For DOS users:

The DOS versions of software should only be used by customers who need to program a limited variety of devices. Many newer devices are supported on Windows but not on DOS.

To install the DOS-based software, create a directory such as “AdvinDOS” on your computer, and then copy all files under AdvinDOS to it.

Invoke advin.exe to start the software. You’ll need to select appropriate software modules such as spEE or spEPROM to program different classes of devices such as EE-memories or EPROMs.

To program devices on Gang modules, you’ll need to use sgXX instead of spXX.

2.3 SOFTWARE INVOCATION

The Captain software can be invoked by clicking on Captain.exe or by clicking on short cuts according to standard Windows operations.

After Captain is started, it scans through the three possible parallel port addresses, starting from LPT3, then LPT2, then LPT1, until it finds a programmer that is connected and is powered up. If one is found, a message like this will appear:

```
Programmer found at LPT2.
LPT port selection good.
Machine model is PILOT-xxx.
```

If the programmer is not powered up or if it is not connected properly, you will receive a message that the programmer is not found and to check power switch and parallel cable. In this case, you should make the proper connection or power up the programmer, then manually click on the yellow **Configure Port** icon at the left-most side of the tool bar to have Captain search for the programmer again.

When the **Configure Port** command is invoked or when software is first started, the presence of any add-on module is detected and displayed on the bottom of the Status Display Panel. For example, if the add-on module is a **PX-32**, the software will display: "Machine Model: PILOT-U44+ with **PX-32**"

If you do not see the module name being displayed, please double check and make sure the module is indeed plugged-in properly.

2.4 QUICK START

As a quick start, you can try the commonly used commands via these icons:

1. Select a device by clicking .
2. Place a device into the ZIF socket or adapter. Then read it into the data buffer by clicking .
3. To display or edit the data buffer, click . Here you can make changes to your data.
4. To program a device, click .

5 To select and load a data file, click to ***specify*** your data file and load it.

6. If the data filename is *already specified*, as shown in the Status Panel, you can click on to load the data file.

2.5 IN CASE OF PROBLEMS

Here is a checklist in case of problems:

If the **Configure Port** command cannot recognize the hardware:

1. Make sure the power switch is turned on and the red LED is lit. If the switch is on, but the LED is off, check the power cord connection and the power supply. If both are good, check and replace the fuse as follows:

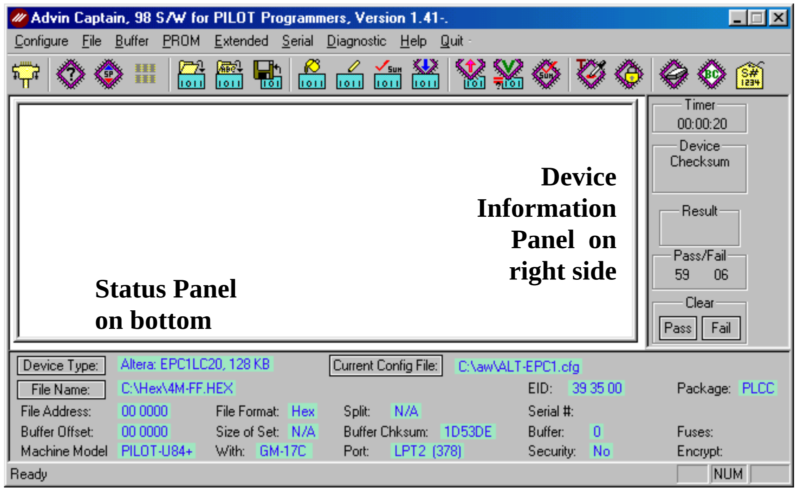
   Disconnect power cable from power source and check the fuse that is located in a HOLDER at the middle of the power entry module (between the IEC plug and the power switch. There is no need to open up the programmer hardware.). This HOLDER can be plied out with a flat screw driver. When this holder is plied out, the fuse will come out with it. In case the fuse is blown, you can find a replacement fuse in a little tray within the HOLDER. (In case both fuses are blown, you can replace them with a one-amp, 250 volt, slow blow fuse.)

2. Check the interface cable. Make sure the cable is connected to the proper connector on the PC and is firmly plugged in. If the cable you're using is not the one supplied, it could have been made incorrectly or be otherwise defective.

3. Is your PILOT programmer connected to an "A/B switch" or "T-switch"? If so, plug the programmer directly into the parallel port at the PC, without going through the switch. Some switches do cause problems.

3.0 SCREEN DISPLAY

Below is a graphical representation of the main window of the Advin Captain Software.



Device Information Panel on right side

Status Panel on bottom

Device Type: Altera: EPC1LC20, 128 KB
Current Config File: C:\aw\ALT-EPC1.cfg
File Name: C:\Hex\4M-FF.HEX
EID: 39 35 00
Package: PLCC
File Address: 00 0000
File Format: Hex
Split: N/A
Serial #:
Buffer Offset: 00 0000
Size of Set: N/A
Buffer Chksum: 1D53DE
Buffer: 0
Fuses:
Machine Model PILOT-U84+
With: GM-17C
Port: LPT2 (378)
Security: No
Encrypt:

The pull-down menus, on the top of the Captain window, let you access all of the functions available.

The toolbar contains icons for the most frequently used functions. This toolbar is a quicker method than that of the pull-down menu. When you place your cursor over any of these icons, the description of the icon will appear.

The lower portion of the Captain window is the status panel. It displays the current status and selections.

The time it took for the device to program, the checksum of the device, and how many of the devices have passed or failed are all displayed on the right side of the Captain window.

The software version number can be found at the upper blue border above the pull-down menu. If you call technical support, you'll need to know the software version you are currently using.

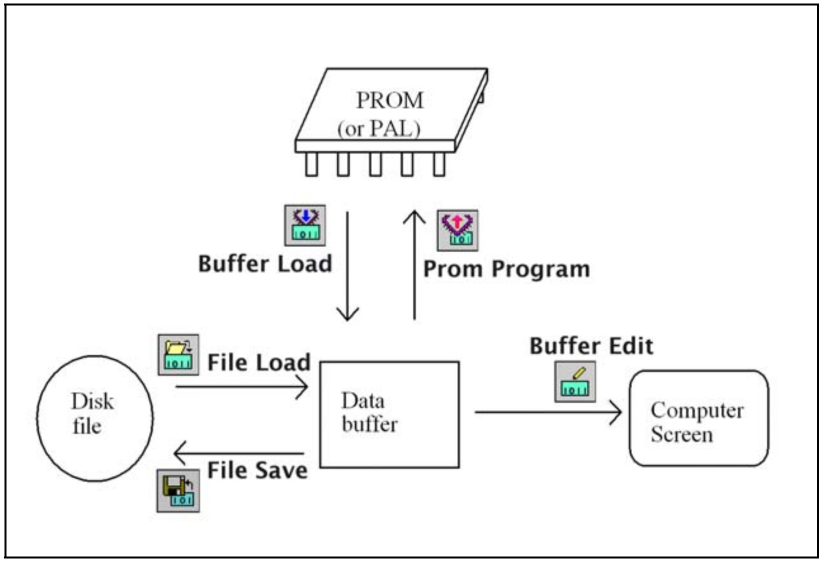
4.0 COMMAND DESCRIPTIONS

COMMAND OVERVIEW

The following diagram shows data flow among the four major components of interest: data file, data buffer, PROM and computer screen.

Shown below are the most commonly used commands. For example, to read a master PROM into the data buffer, use **Buffer Load.** To program a PROM, use **Prom Program.**

The most frequently used commands and directions of data flow:



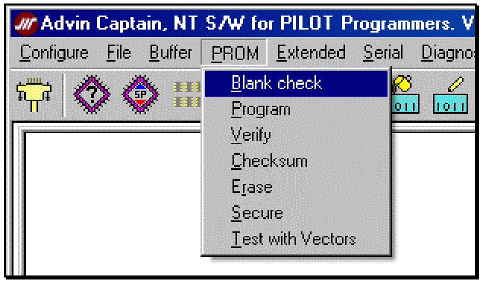
Data-Flow-R288.GIF

Commands can be invoked by one of three methods:

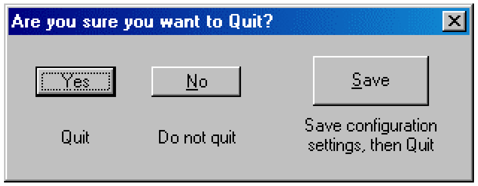
a. By the mouse, via an icon. Most commonly-used commands are represented by an icon.

b. By the mouse, via the pull-down menu. For example, the **Prom Program** command can be invoked by clicking **Prom**, then clicking **Program**.

c. By the keyboard. For example, **[ALT][p]** causes the following pull down menu to pop up. Since the 'P' in **Program** is underlined, hitting **[p]** would invoke the **Program** command.



On the following screen, hitting the **[s]** key is the same as clicking on the "Save" button. Hitting [n] is the same as clicking the "No" button. Since the "Yes" button is highlighted, hitting [space bar] or 'y' would select the "yes" option.


Are you sure you want to Quit?
Yes
No
Save
Quit
Do not quit
Save configuration settings, then Quit

CONFIGURE commands specify system parameters such as device type, set-size, odd/even splitting, operation options, etc. In general, these commands should be used before other commands.

BUFFER commands allow you to see and edit data in the data buffer. The data buffer is a temporary buffer that holds data to be programmed to a device. It also holds data that is read from a device.

The PROM command group consists of erase, blank check, program, verify, and checksum functions. These are commands that operate on the **device** (versus **buffer** commands which operate on the data buffer).

The EXTENDED commands are those that do not fit into any of the above mentioned categories, for example, the **Active Range** command and certain special commands that apply only to certain devices.

The DIAGNOSTIC commands can be used to make sure the programmer hardware is working properly.

4.1 CONFIGURE COMMANDS

4.1.1 CONFIGURE PORT n

The programmer can be connected to the PC through any one of the three parallel printer ports: LPT1, LPT2, or LPT3. When the "**Configure Port n**" command is given, Captain finds what programmer model is connected at LPTn. If n is "**Auto**", Captain automatically scans all three ports, starting with port 3. If a programmer is found, then it stops. If no programmer is found at a port, it resets the port and goes on to the next port. If a programmer is found at a port, this message is displayed:

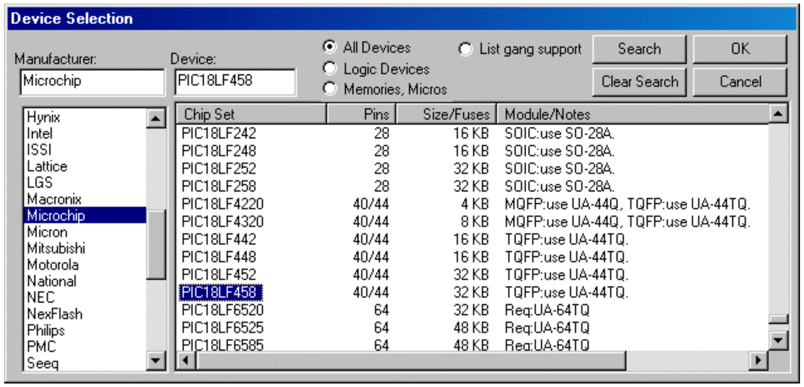
> **Programmer connection found at LPTn.**
> **LPT Port Selection good.**
> **Machine Model is PILOT-xxx.**

The machine model will also be displayed at the lower left hand corner of the screen. If there is an add-on module present, the module name will be displayed along side the machine model. (For example: PILOT-U44+ with UA-44.) For some older modules which do not carry module IDs, their name will not be displayed.

The **Configure Port Auto** command can be selected by clicking .

4.1.2 CONFIGURE DEVICE

This command allows you to specify the manufacturer and the device type that you need to program.



Device Selection

Manufacturer: Microchip

Device: PIC18LF458

● All Devices ○ List gang support
○ Logic Devices
○ Memories, Micros

Hynix
Intel
ISSI
Lattice
LGS
Macronix
Microchip
Micron
Mitsubishi
Motorola
National
NEC
NexFlash
Philips
PMC
Seeq

Chip Set	Pins	Size/Fuses	Module/Notes
PIC18LF242	28	16 KB	SOIC:use SO-28A.
PIC18LF248	28	16 KB	SOIC:use SO-28A.
PIC18LF252	28	32 KB	SOIC:use SO-28A.
PIC18LF258	28	32 KB	SOIC:use SO-28A.
PIC18LF4220	40/44	4 KB	MQFP:use UA-44Q, TQFP:use UA-44TQ.
PIC18LF4320	40/44	8 KB	MQFP:use UA-44Q, TQFP:use UA-44TQ.
PIC18LF442	40/44	16 KB	TQFP:use UA-44TQ.
PIC18LF448	40/44	16 KB	TQFP:use UA-44TQ.
PIC18LF452	40/44	32 KB	TQFP:use UA-44TQ.
PIC18LF458	40/44	32 KB	TQFP:use UA-44TQ.
PIC18LF6520	64	32 KB	Req:UA-64TQ
PIC18LF6525	64	48 KB	Req:UA-64TQ
PIC18LF6585	64	48 KB	Req:UA-64TQ

You can select the manufacturer and device type by using cursor keys or by typing them into the edit boxes.

Each device type includes all speed, temperature and package deviations. In cases where the programming algorithm is different, the device deviations are listed separately.

Under the "number of pins" column, 40/44 means both 40 and 44 pin packages, where 40 is the DIP package and 44 includes PLCC or QFP packages.

Under the "Size" column, Kb means K bits and KB means K bytes.

If no gang module is installed, the devices listed are those supported in single-site mode. If Captain detects the presence of a gang module, the device listed are those supported in gang mode.

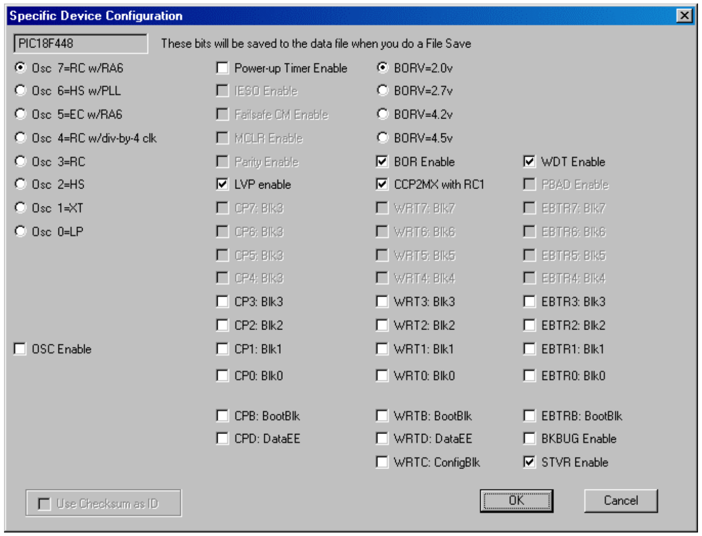
If you want to know what devices are supported in gang mode but do not have a gang module, you can check the "List gang support" button and Captain will list devices that are supported under gang mode.

4.1.3 CONFIGURE SPECIFIC-DEVICE-CONFIGURATION

This command is also reachable via this icon .

Some devices have certain configuration bits or fuses that need to be programmed. The Extended Specific-Device-Configuration command opens up screens designed specifically for these devices.

For example, the screen for a Microchip PIC18F448 looks like:



Specific Device Configuration

PIC18F448 These bits will be saved to the data file when you do a File Save

● Osc 7=RC w/RA6	□ Power-up Timer Enable	● BORV=2.0v	
○ Osc 6=HS w/PLL	□ IESO Enable	○ BORV=2.7v	
○ Osc 5=EC w/RA6	□ Failsafe CM Enable	○ BORV=4.2v	
○ Osc 4=RC w/div-by-4 clk	□ MCLR Enable	○ BORV=4.5v	
○ Osc 3=RC	□ Parity Enable	☑ BOR Enable	☑ WDT Enable
○ Osc 2=HS	☑ LVP enable	☑ CCP2MX with RC1	□ PBAD Enable
○ Osc 1=XT	□ CP7: Blk3	□ WRT7: Blk7	□ EBTR7: Blk7
○ Osc 0=LP	□ CP6: Blk3	□ WRT6: Blk6	□ EBTR6: Blk6
	□ CP5: Blk3	□ WRT5: Blk5	□ EBTR5: Blk5
	□ CP4: Blk3	□ WRT4: Blk4	□ EBTR4: Blk4
	□ CP3: Blk3	□ WRT3: Blk3	□ EBTR3: Blk3
	□ CP2: Blk2	□ WRT2: Blk2	□ EBTR2: Blk2
□ OSC Enable	□ CP1: Blk1	□ WRT1: Blk1	□ EBTR1: Blk1
	□ CP0: Blk0	□ WRT0: Blk0	□ EBTR0: Blk0
	□ CPB: BootBlk	□ WRTB: BootBlk	□ EBTRB: BootBlk
	□ CPD: DataEE	□ WRTD: DataEE	□ BKBUG Enable
		□ WRTC: ConfigBlk	☑ STVR Enable

□ Use Checksum as ID

SP-PIC448-R150.GIF

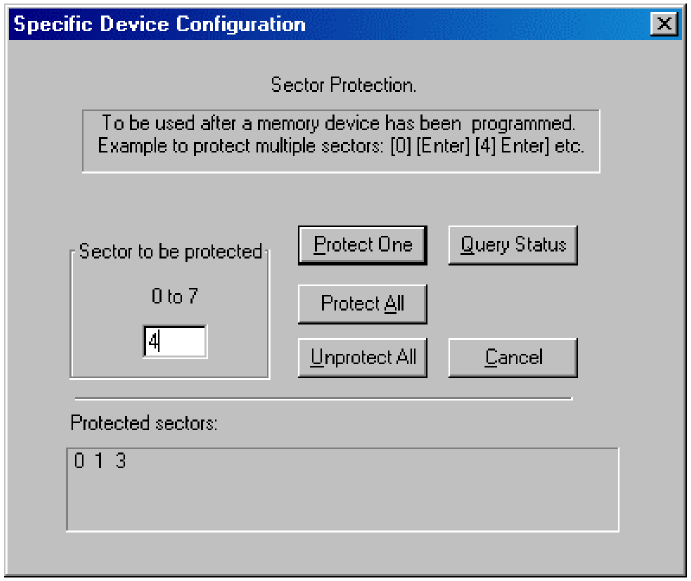
For most devices, these specific device configuration bits are part of the data file. (This is true if the IC manufacturer has guidelines in the programming spec regarding what addresses in the logical data file should be used to store these bits.)

That means, after you load a data file, these configurations bits will be already setup. If your data file does not contain these settings, you can use the SP icon to set these bits. If you save the data file, these bits will be saved as part of the data file.

The above also means that you should NOT load the data file AFTER you do the setup of these specific configuration bits.

Please be aware that for some devices, the device configuration bits are not part of the data file.

The Specific-Device-Configuration screen of an AMD 29F040B device looks like:



Specific Device Configuration

Sector Protection.

To be used after a memory device has been programmed.
Example to protect multiple sectors: [0] [Enter] [4] Enter] etc.

Sector to be protected

0 to 7

4

Protected sectors:

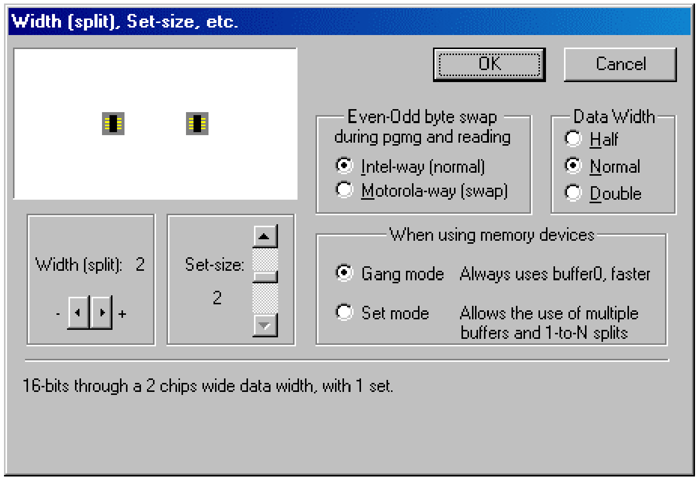
0 1 3

SP-29F040B-R100.GIF

4.1.4 CONFIGURE WIDTH/SET-SIZE

(Applies to EPROMs, EEPROMs and FLASH EPROMs. Does not apply to micros or logic devices)

This command allows you to specify the data bus width of the target processor with which your PROMs are going to be used.



Width (split), Set-size, etc.

Even-Odd byte swap during pgmg and reading
● Intel-way (normal)
○ Motorola-way (swap)

Data Width
○ Half
● Normal
○ Double

Width (split): 2

Set-size: 2

When using memory devices
● Gang mode   Always uses buffer0, faster
○ Set mode   Allows the use of multiple buffers and 1-to-N splits

16-bits through a 2 chips wide data width, with 1 set.

Split-R144.GIF

Width and Set-Size

Width affects only **File Load** and **File Save** commands. All other commands are executed independent of **Width**.

A width of ‘1’ means no split during file load. In technical terms, your data bus is as wide as the data bus for 1 PROM. (For example, your data bus is 8-bit and it feeds into **one** set of byte-wide memory such as 29F010.) During file load, consecutive bytes from the file will be loaded into consecutive bytes in the same buffer.

A width of ‘2’ means a **1-to-2** split during file load. In technical terms, your data bus width is twice the data size of your PROM. (For example, your data bus is 16-bit and it feeds into **two** sets of byte-wide memory such as 29F010. Or, your data bus is 32-bit and it feeds into **two** sets of word-wide memory such as 29F100.) During file load, data will alternatively go to even and odd buffers.
This is what will happen when width=2: If your PROM is byte-wide, consecutive **bytes** will go to alternate PROMs. If your PROM is word-wide, consecutive **words** will go to alternative PROMs.

A width of ‘4’ assumes that your data bus width is four times the data size on a PROM. During a **File Load**, a **1-to-4** split will happen.

For example, if you are using word-wide EPROMs such as 27C210s and you are selecting **Configure Width 4**, your EPROMs will be sitting on a 4x16=64-bit wide data bus.

Even or odd, Intel-way or Motorola-way

In almost all digital control systems, each data **byte** is addressable. That means each 8-bit **byte** has its own address. In the case of a piece of data that is comprised of a 16-bit **word**, i.e. two **bytes**, two addresses will then be involved.

Intel's standard way of locating these two bytes is to put the lower order byte first, i.e. at the lower address. Motorola's way is to put the higher order byte first, i.e., at the higher address.

Therefore, when programming 16-bit wide PROMs (such as a 27C210) for an Intel data bus application, the **even-addressed** bytes in your data file should go to **bits 0-7** of the PROM, and the higher addressed bytes should go to **bits 8-15**.

Conversely, when programming 16-bit wide PROMs for a Motorola application, the **odd-addressed** bytes should go to **bits 0-7** and the even-addressed bytes should go to **bits 8-15**.

Data Width

The **Normal** setting should be used predominately.

Half is used under very special situations where only half the data width is used in programming. During normal programming, a 16-bit memory device is programmed 16-bit at a time. 8-bit programming can be used if and when you mount the memory device on a custom circuit board that uses the 8-bit option of the device. (For example, when the BYTE/ pin of a device is tied to Vih.)

4.1.5 CONFIGURE SAVE CONFIGURATION

This command allows you to save current configuration information so that you do not have to re-enter them every time you use Captain. Examples of configuration information saved are: file format, file name, device selection, and items in the Configure Operations screen.

Configuration items are saved into a file called **default.cfg** in the default Captain directory. The file **default.cfg** is a default file used by Captain when it is first initialized. You can also specify your own file name ending in **.cfg** that can be loaded using the load configuration command described below.

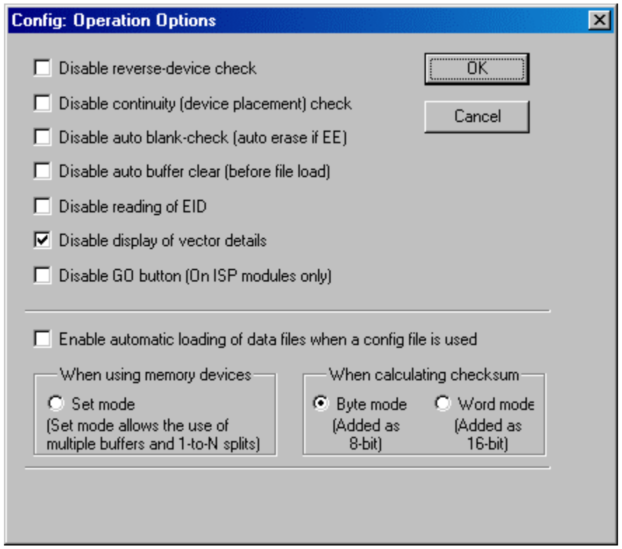
If the **default.cfg** file is not present, a new one will be created. If it is already there, you will be prompted before the software updates it. You can change the configuration information and re-save it as often as you like.

4.1.6 CONFIGURE LOAD CONFIGURATION

This command allows you to load your configuration file that you specify. This command can be used to load different configuration for the different types of devices, data files, file formats, etc…

4.1.7 CONFIGURE OPERATION OPTIONS

This command allows you to change other configuration items that are less frequently used.



□ Disable reverse-device check

□ Disable continuity (device placement) check

□ Disable auto blank-check (auto erase if EE)

□ Disable auto buffer clear (before file load)

□ Disable reading of EID

☑ Disable display of vector details

□ Disable GO button (On ISP modules only)

□ Enable automatic loading of data files when a config file is used

When using memory devices

○ Set mode
(Set mode allows the use of multiple buffers and 1-to-N splits)

When calculating checksum

● Byte mode (Added as 8-bit)   ○ Word mode (Added as 16-bit)

Config-Operation.GIF

Disable (Insertion or) Reverse-Device Check

The software checks to make sure a device is inserted properly into the programming socket. If a device is not inserted at all or if it is inserted with pin 1 facing the opposite direction, the software will issue an error message. This check can be disabled if you want to.

(A word of CAUTION: reverse-device detection works for about 99% of all devices. A few devices cannot be checked for reverse insertion. The detection is there to save your device, only most of the time, when it is inserted in reverse.)

Disable Continuity (Device Placement) Check

Before any operation is done on a device, the software checks to make sure all pins of the device are making good contact with the programming socket, i.e. continuity from device to socket. In case you do not want the software to perform this check (e.g. an IC manufacturer might want to run operations without a device inserted), you can disable it.

Disable Auto-Erase and Auto-Blank-Check

If a device is electrically erasable, it will be erased and blank-checked before programming. If you need to save time (e.g. if you are programming a batch of brand new devices) by avoiding this operation, you can disable it.

Disable Auto Buffer Clear (Before File Load)

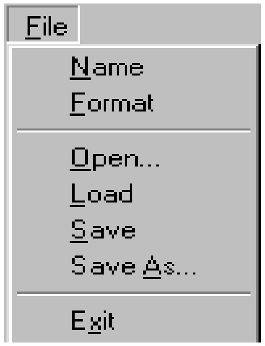
Normally, the data buffer is erased at the beginning of every **File Load** operation. This gives you a pre-defined empty buffer and gives you a predictable checksum if your data file is not a completely full data file.

However, in some applications, a customer may have to load multiple partial data files. In those cases, the Auto Buffer Clear feature can be disabled.

Disable Reading of Electronic ID (EID)

This command should normally not be used. It is reserved for special privileged users such as semiconductor manufacturers who are using our equipment for device testing purposes.

4.2 FILE COMMANDS



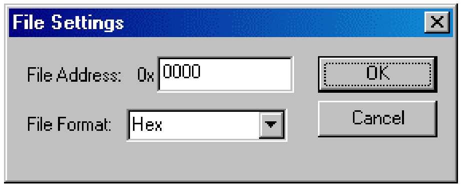
The file command group consists of all file related commands such as selecting a file format, loading and saving a file.

4.2.1 FILE NAME

This command gives the user the ability to specify a filename without actually opening up the file. This an be used with the **File Save** command to save a new file that is in the buffer.

4.2.2 FILE FORMAT

This command allows the user to select the file type (hex, bin, S-record, etc…) and/or the starting file address that will be used to load into the buffer. Below is a graphical representation of the pop-up window and default settings.



File Settings

File Address: 0x 0000

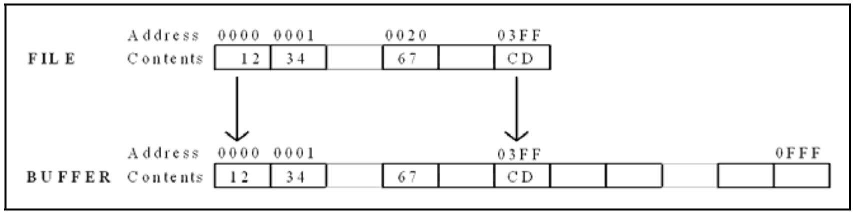
File Format: Hex

The file address window specifies which byte in the file will be loaded into the **beginning** of the buffer. In other words, **File Address= 0x0020** means data from address 20 in the file will go into the first byte of the buffer during a file load command.

In the following examples, device type is assumed to be 2732 (address range 0000-0FFF).

Example 1

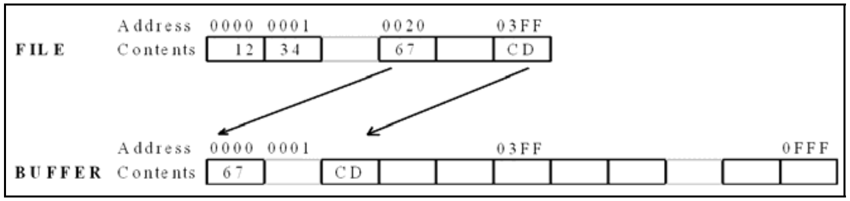
If a **File Address = 0x0000** has been entered, a file load will cause the following: (In this case, the file size is smaller than the buffer size.)



File-Addr-00.GIF

Example 2

If a **File Address = 0x0020**, a file load will cause the following to happen. That is, addresses before 0020 in the file are ignored. Data starting at 0020 of the file will be taken and put into address 0 of the buffer.



File-Addr-20.GIF

4.2.3 FILE OPEN

This command allows the user to select the file that will be load into the buffer. After choosing a file this command will automatically perform a file load to the buffer specified.

4.2.4 FILE LOAD

This command invokes the file loading process and loads data into the specified buffer. The name of the file to be loaded should be already specified by a **File Name** earlier.

If **Configure Width/Set Size** is used earlier to specify split, the file will be loaded over two or more buffers. The user must specify file load in set mode in order for this to work.

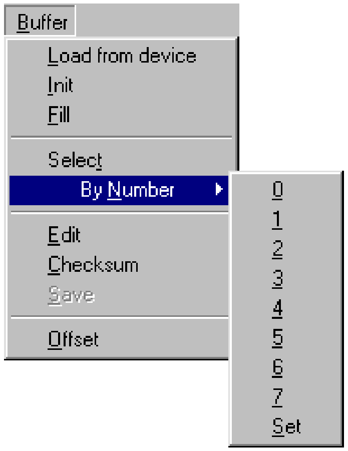
4.2.5 FILE SAVE

Data in the selected buffer or buffers will be saved onto the file specified earlier by the File Name command.

4.2.6 FILE SAVE AS

This command allows the user to save the file and specify a file name.

4.3 BUFFER COMMANDS



The buffer command group consists of all buffer related commands such as selecting a buffer load, buffer initialize, buffer edit, and buffer checksum. To the left is the pull down menu for the buffer commands.

4.3.1 BUFFER LOAD FROM DEVICE n

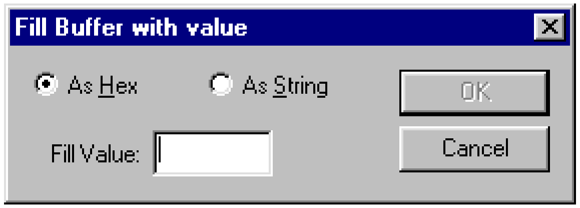
Selected buffers are loaded with data from PROMs in corresponding sockets. Once in buffers, the PROM data can be either saved onto a file, or modified and a new PROM burned.

4.3.2 BUFFER INIT

This command initializes the buffer back to its original state, usually it is all FF's.

4.3.3 BUFFER FILL

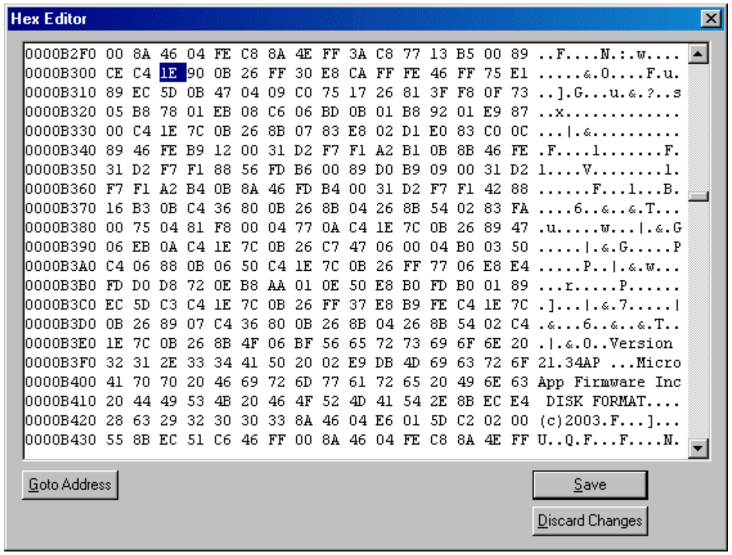
Selected buffers are initialized with a value the user enters, usually a hex value. Below is the pop-up window.



When a device type is selected with the **Configure Device** command, a buffer of proper size will be created and initialized to FFs. Normally, there is no need for you to initialize the data buffers before you do a **File Load**, unless you are using "Disable Auto Buffer Clear Before File Load" and you are using data files that do not completely fill the buffer.

4.3.4 BUFFER EDIT

This command allows data in the selected buffer to be viewed and changed. Below is the pop-up window for this command.



```
0000B2F0 00 8A 46 04 FE C8 8A 4E FF 3A C8 77 13 B5 00 89 ..F....N.:.w....
0000B300 CE C4 1E 90 0B 26 FF 30 E8 CA FF FE 46 FF 75 E1 .....&.0....F.u.
0000B310 89 EC 5D 0B 47 04 09 C0 75 17 26 81 3F F8 0F 73 ..].G...u.&.?..s
0000B320 05 B8 78 01 EB 08 C6 06 BD 0B 01 B8 92 01 E9 87 ..x.............
0000B330 00 C4 1E 7C 0B 26 8B 07 83 E8 02 D1 E0 83 C0 0C ...|.&..........
0000B340 89 46 FE B9 12 00 31 D2 F7 F1 A2 B1 0B 8B 46 FE .F....1.......F.
0000B350 31 D2 F7 F1 88 56 FD B6 00 89 D0 B9 09 00 31 D2 1....V........1.
0000B360 F7 F1 A2 B4 0B 8A 46 FD B4 00 31 D2 F7 F1 42 88 ......F...1...B.
0000B370 16 B3 0B C4 36 80 0B 26 8B 04 26 8B 54 02 83 FA ....6..&..&.T...
0000B380 00 75 04 81 F8 00 04 77 0A C4 1E 7C 0B 26 89 47 .u.....w...|.&.G
0000B390 06 EB 0A C4 1E 7C 0B 26 C7 47 06 00 04 B0 03 50 .....|.&.G.....P
0000B3A0 C4 06 88 0B 06 50 C4 1E 7C 0B 26 FF 77 06 E8 E4 .....P..|.&.w...
0000B3B0 FD D0 D8 72 0E B8 AA 01 0E 50 E8 B0 FD B0 01 89 ...r.....P......
0000B3C0 EC 5D C3 C4 1E 7C 0B 26 FF 37 E8 B9 FE C4 1E 7C .]...|.&.7.....|
0000B3D0 0B 26 89 07 C4 36 80 0B 26 8B 04 26 8B 54 02 C4 .&...6..&..&.T..
0000B3E0 1E 7C 0B 26 8B 4F 06 BF 56 65 72 73 69 6F 6E 20 .|.&.O..Version 
0000B3F0 32 31 2E 33 34 41 50 20 02 E9 DB 4D 69 63 72 6F 21.34AP ...Micro
0000B400 41 70 70 20 46 69 72 6D 77 61 72 65 20 49 6E 63 App Firmware Inc
0000B410 20 44 49 53 4B 20 46 4F 52 4D 41 54 2E 8B EC E4  DISK FORMAT....
0000B420 28 63 29 32 30 30 33 8A 46 04 E6 01 5D C2 02 00 (c)2003.F...]...
0000B430 55 8B EC 51 C6 46 FF 00 8A 46 04 FE C8 8A 4E FF U..Q.F...F....N.
```

Buffer-Edit-R144.GIF

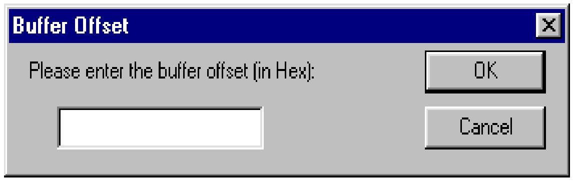
You can make changes by clicking on the hex value and entering a new value or by using the right side of the display for ASCII values. After changes are made, hit the **SAVE** button to keep the changes, or hit the **DISCARD CHANGES** to return to the original data.

4.3.5 BUFFER CHECKSUM

A checksum is calculated by adding up all the bytes in the buffer. Checksums are 16-bit values. That is, they are "word-accumulated".

4.3.6 BUFFER OFFSET

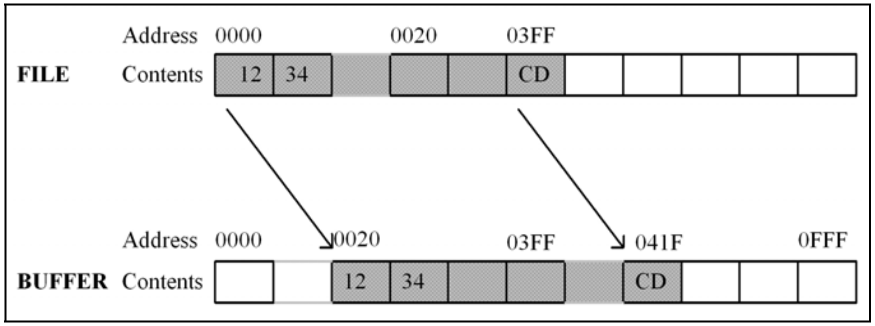
If you can recall from an earlier section describing the **File Address** command: it allows you to select certain parts of a file when loading it into a buffer, and discarding a certain beginning part. In other words, the data is moved **down** to a lower address. If you want to move data **up** to a higher address, you can use the analogous command **Buffer Offset**. Below is the pop-up window.


Buffer Offset

Please enter the buffer offset (in Hex):

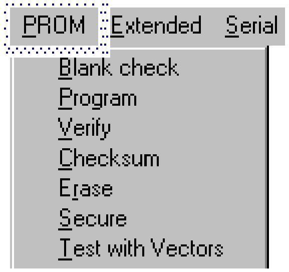
File Address allows you to specify how much to **ignore** at the beginning of the **-file**. **Buffer Offset** allows you to specify how much to **skip** at the beginning of the **buffer**.

For example, if File Address is 0000 and Buffer Offset is 20, a file load results in:



Buffer-Offset-20.GIF

4.4 PROM COMMMANDS



The PROM command group consists of all commands that either program or access the PROMs. (The only exception to this is the command that copies PROM data into buffers. That command is in the buffer command group).

The word **PROM** as used in this manual means "device". This is because our programmers have evolved from programming PROMs to programming a whole new range of other devices such as Flash, PLDs, micro controllers, etc. PROM is still being used here mainly for compatibility between old and new versions of software.

4.4.1 PROM BLANK-CHECK

Checks to see if selected PROMs are blank. As soon as a non-blank byte is detected, the offset address and value of this byte will be displayed.

4.4.2 PROM PROGRAM

Programs a device with data from corresponding buffers. If the selected device is electrically erasable, it will be **automatically erased** first before the programming cycle starts. The entire device is programmed. Since the buffer size is always the same as the PROM size, this means the entire buffer is programmed onto the PROM. (If you want only a certain part of the PROM to be programmed, you can use **Extended Active-Range** as explained later in section **4.5.1**.)

As soon as a byte fails to program, the command is terminated.

At the end of the command, offset locations and values of any defective bytes will be displayed.

4.4.3 PROM VERIFY

This command verifies data in a PROM against data in corresponding buffer. A defective byte terminates the command. The offset address of the error byte will be reported. The byte values in both the buffer and the problem PROM are also reported.

4.4.4 PROM CHECKSUM

Calculates checksums for selected PROMs. The checksum is a 16 bit value calculated by simply adding up all data bytes.

4.4.5 PROM ERASE

Electrically erases a device (when applicable), and then checks to see if it is blank.

4.4.6 PROM SECURE

Secures the PROM from being read and/or re-programmed (when applicable).

For most devices, a secured device would yield all 0s or all Fs when read by the programmer. (e.g. when you do a Buffer Load to read the data into the buffer.)

An electrically-erasable device can be erased and reprogrammed again even after the device has been secured.

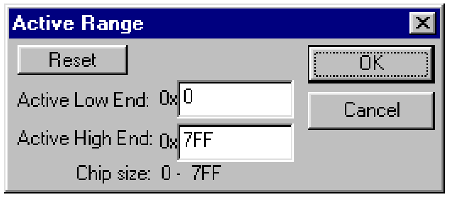
Most UV-erasable device can also be erased and reprogrammed again even after the device has been secured. A major exception to this are the Microchip PIC 16C or 16LC devices.

A very frequently asked question is: "Can I read a device that is secured". The answer is NO. The reason is: if you can read a device that has been secured, then the security feature is useless.

4.5 EXTENDED COMMANDS

The Extended command group consists of all commands that allow you to selectively program a certain part of a device or select device specific functions. To the left is the pull down menu for the Extended commands.

4.5.1 EXTENDED ACTIVE-RANGE



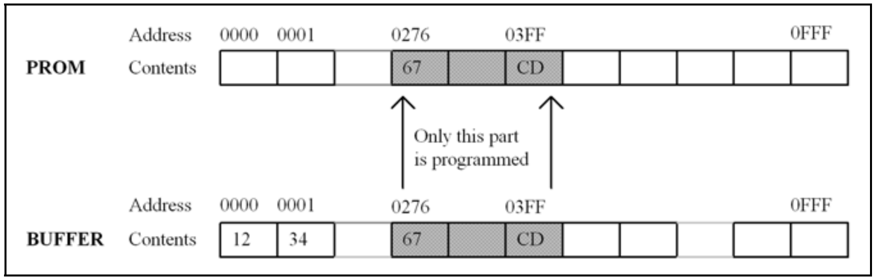
Active Low End: 0x 0

Active High End: 0x 7FF

Chip size: 0 - 7FF

Active range is preset to include the complete address range of a device type during **Configure Device** time. If you want to program only a certain part of a device, you can use the LOW and HIGH windows to select the low and high address boundaries. Then only bytes from LOW to HIGH, inclusive, will be programmed during a **Prom Program** command.

Example: When Active-range Low= 276, Active-range High=3FF, the following will happen in a **PROM PROGRAM**:



Active-Range-R400.GIF

Besides the **Prom Program** command, the active range setting is also effective during **Prom Checksum** and **Buffer Load** commands. You can get the checksum of a certain block of data in the device or you can selectively load a certain part of a device into the buffer using this Active-range facility.

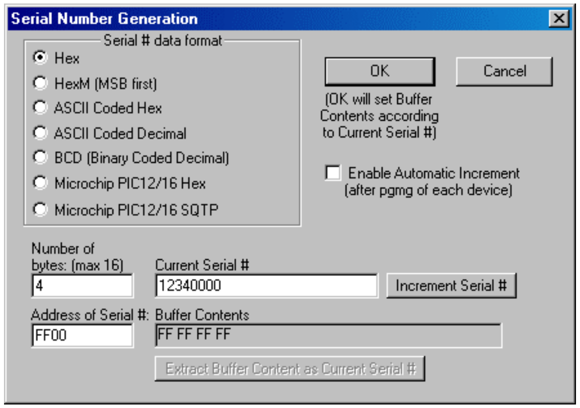
Active-range does not apply to other commands such as File Load and File Save. (It would be very confusing during a file load, if File address, Buffer Offset and Active Range commands were all in operation.) Below is the pop-up window for this command:

4.5.2 EXTENDED SPECIFIC-DEVICE-CONFIGURATION

This is the same command as Configure Specific-Device-Configuration, which is also reachable via the SP icon .

4.6 AUTOMATIC SERIAL NUMBER GENERATION

When the serial number icon is clicked, this dialog appears:



Serial Number Generation

Serial # data format
- ● Hex
- ○ HexM (MSB first)
- ○ ASCII Coded Hex
- ○ ASCII Coded Decimal
- ○ BCD (Binary Coded Decimal)
- ○ Microchip PIC12/16 Hex
- ○ Microchip PIC12/16 SQTP

(OK will set Buffer Contents according to Current Serial #)

□ Enable Automatic Increment (after pgmg of each device)

Number of bytes: (max 16): 4

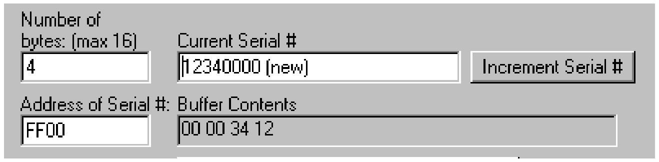
Current Serial #: 12340000

Address of Serial #: FF00

Buffer Contents: FF FF FF FF

If you enter the serial number as above, clicking OK will allow Captain to generate serial numbers at the 4 bytes starting at FF00.

If you click the S# 1234 icon again, you will see that buffer area indeed contains the serial number in the format you have selected, which is the Hex format, with LSB starting first.



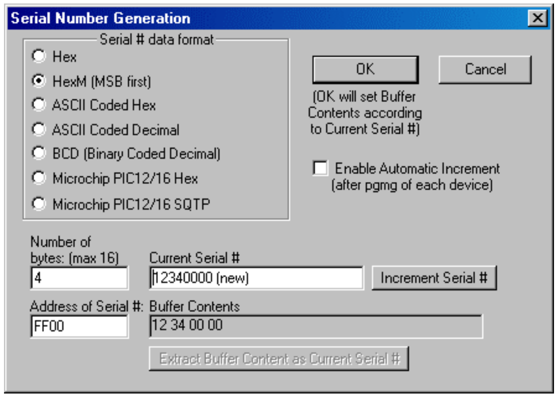
Number of bytes: (max 16): 4

Current Serial #: 12340000 (new)

Address of Serial #: FF00

Buffer Contents: 00 00 34 12

If you have chosen the “Hex (MSB Format)”, then the buffer contents would have been:



Serial Number Generation

Serial # data format
- ○ Hex
- ● HexM (MSB first)
- ○ ASCII Coded Hex
- ○ ASCII Coded Decimal
- ○ BCD (Binary Coded Decimal)
- ○ Microchip PIC12/16 Hex
- ○ Microchip PIC12/16 SQTP

(OK will set Buffer Contents according to Current Serial #)

□ Enable Automatic Increment (after pgmg of each device)

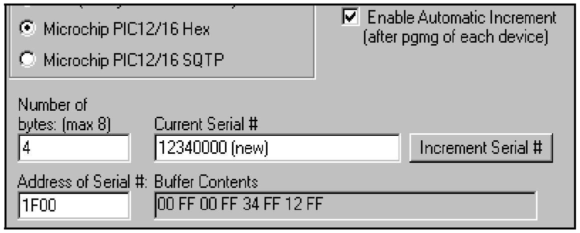
Number of bytes: (max 16): 4

Current Serial #: 12340000 (new)

Address of Serial #: FF00

Buffer Contents: 12 34 00 00

Microchip PIC12 and PIC16 devices have only 14 data bits per word. Therefore serial numbers has to be put into alternating bytes, with odd bytes not holding any serial number information. Therefore, if you use the "PIC12/16 Hex" format, the buffer contents will look like this (the FFs are the original contents in the buffer, not changed by Captain):



- ● Microchip PIC12/16 Hex
- ○ Microchip PIC12/16 SQTP

☑ Enable Automatic Increment (after pgmg of each device)

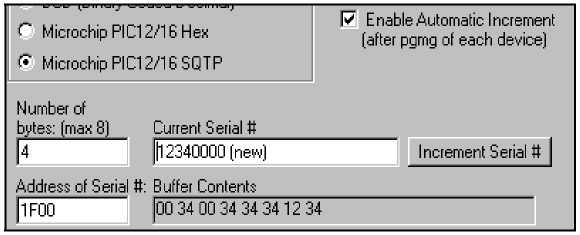
Number of bytes: (max 8): 4

Current Serial #: 12340000 (new)

Address of Serial #: 1F00

Buffer Contents: 00 FF 00 FF 34 FF 12 FF

If you use the "PIC12/16 SQTP" format, Captain automatically inserts the PIC instruction code of "34" into alternating bytes (i.e. the high 6-bits of the PIC instruction word):



- ○ Microchip PIC12/16 Hex
- ● Microchip PIC12/16 SQTP

☑ Enable Automatic Increment (after pgmg of each device)

Number of bytes: (max 8): 4

Current Serial #: 12340000 (new)

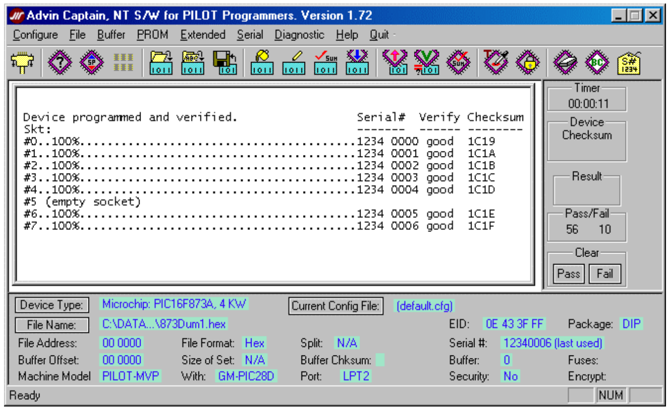
Address of Serial #: 1F00

Buffer Contents: 00 34 00 34 34 34 12 34

The word “new” means that this serial number will be used when you invoke the “Prom Program” command next time. If the word “new” is not there, it means the current device has been programmed with the indicated serial number.

If the "Enable Automatic Increment…." Button is selected, then Captain will automatically increment the serial number after each device has been programmed.

Under gang mode (when using Gang Modules), serial numbers will be automatically increased across all the devices being programmed.



Advin Captain, NT S/W for PILOT Programmers. Version 1.72

```
Device programmed and verified.                  Serial#  Verify Checksum
Skt:                                             -------  ------ --------
#0..100%.........................................1234 0000 good  1C19
#1..100%.........................................1234 0001 good  1C1A
#2..100%.........................................1234 0002 good  1C1B
#3..100%.........................................1234 0003 good  1C1C
#4..100%.........................................1234 0004 good  1C1D
#5 (empty socket)
#6..100%.........................................1234 0005 good  1C1E
#7..100%.........................................1234 0006 good  1C1F
```

Timer: 00:00:11

Pass/Fail: 56 10

Device Type: Microchip: PIC16F873A, 4 KW

Current Config File: (default.cfg)

File Name: C:\DATA...\873Dum1.hex

EID: 0E 43 3F FF

Package: DIP

File Address: 00 0000

File Format: Hex

Split: N/A

Serial #: 12340006 (last used)

Buffer Offset: 00 0000

Size of Set: N/A

Buffer Chksum:

Buffer: 0

Fuses:

Machine Model: PILOT-MVP

With: GM-PIC28D

Port: LPT2

Security: No

Encrypt:

Ready

Notice that serial numbers are automatically skipped over empty sockets.

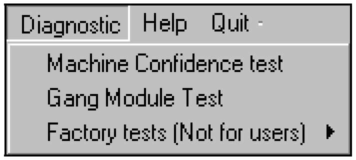
4.7 ISP, IN-SYSTEM PROGRAMMING (IN-CIRCUIT PROGRAMMING)

PILOT programmer hardware and Captain software provide excellent support for ISP programming, especially with Microchip PIC micros and serial PROMs. Since there is continuous development in these areas, with new PICs being released every month, you can find more up-to-date information on ISP programming from our web site at this page:

http://www.advin.com/isp.htm

The ISP programming options dialog can be reached via **Configure ISP Options** command.

4.8 DIAGNOSTIC COMMANDS



Machine Confidence Test

This command tests the programmer hardware. It makes sure the hardware is functioning properly and reports errors otherwise. Before the test is invoked, you should make sure all add-on modules have been removed from the programmer.

Gang Module Test

This command tests only these gang modules:

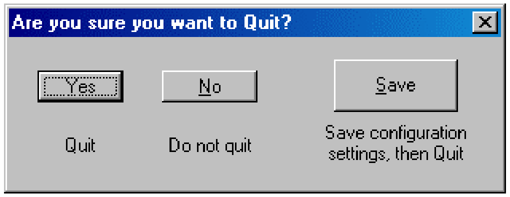
GM-932D and GM-932C.

Either one of these gang modules can be left on the programmer when this test is invoked.

Factory Tests

These tests should not be used.

4.9 EXIT COMMANDS


Are you sure you want to Quit?
Yes
No
Save
Quit
Do not quit
Save configuration settings, then Quit

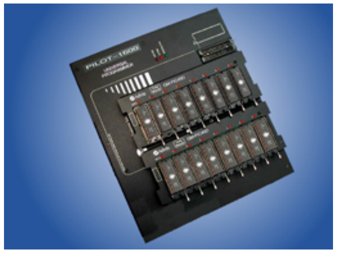
The **Save** option should be used if you want current sections such as device name, filename, etc. to be saved into the default configuration file default.cfg. You can either click on this icon or you can hit the 'S' key on the keyboard to select this option.

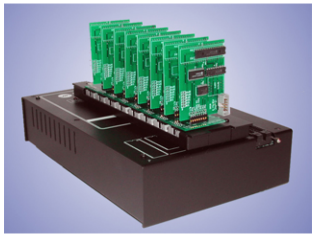
Since the "Yes" button is highlighted, hitting the space bar on your keyboard selects the option that quits without saving the current configuration items into default.cfg.

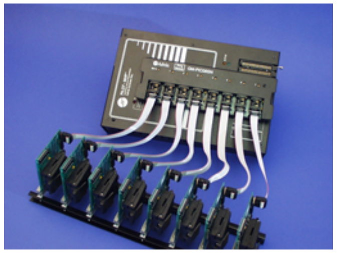
4.10 HELP COMMANDS

The "Help" sub-menu allows most-often-needed information to be displayed.

Expanding Possibilities







Advin Systems Inc.
556 Weddell Drive, #8
Sunnyvale, CA 94089, USA

408-243-7000 Fax: 408-541-9006
www.Advin.com