CPROGACMP User Guide

1 Introduction

CPROGACMP is a Windows command-line version of the PROGACMP software which programs Flash, EEPROM, EPROM, etc. through a PEMicro hardware interface to a supported ARM Cortex processor. The hardware interfaces are available from PEMicro.

Once your interface hardware (Cyclone LC/FX) is properly connected between your PC and target device, you may launch the CPROGACMP executable from the command line. In addition to the executable, multiple command-line parameters must also be passed in order to configure which PEMicro hardware interface CPROGACMP should attempt to connect to, and to configure how that hardware interface will connect to the target device. These parameters include the name of the configuration (.CFG) file, as well as startup commands such as the name of the hardware interface or the port to which the interface is connected.

The .CFG file specifies how to program the target as you intend, and it includes standard programming commands and, optionally, configuration commands. The following chapters will provide a detailed explanation of these commands and parameters.

2 Startup

a. Connect the hardware interface between your PC and the target MCU via the debug ribbon cable.

b. Start the programming software by running it from the Windows Command prompt or by calling the CPROGACMP executable with the correct command line parameters. Allowed command line parameters are:

CPROGACMP **[?] [filename] [/PARAMn=s] [v] [reset_delay n] [bdm_speed n] [hideapp] [freq n] [Interface=x] [port=y] [showports] [/logfile logfilename]**

where:

[filename] A file containing programming commands and comments, default = prog.cfg. See **Section 7 - Example Programming Script File** for an example.

[/PARAMn=s] A command-line parameter which can modify the executing script by replacing special tags (/PARAMn). This can be used to replace any part of the script including programming commands, filenames, and parameters. Valid values of **n** are 0..9. **s** is a string which will replace any occurrence of /PARAMn in the script file. **Section 8 - Using Command-Line Parameters in a Script** has an example for usage.

[INTERFACE=x] Where x is one of the following: (See examples section)

USBMULTILINK (This setting also supports OSBDM)

CYCLONE

TRACELINK

OPENSDA

PARALLEL (Parallel Port or BDM Lightning [Legacy])

[PORT=y] Where the value of y is one of the following (see the showports command-line parameter for a list of connected hardware; always specify the "interface" type as well):

USBx Where x = 1,2,3, or 4. Represents an enumeration number for each piece of hardware starting at 1. Useful if trying to connect to a Cyclone, Tracelink, or Multilink product. If only one piece of hardware is connected, it will always

enumerate as USB1.

An example to select the first Multilink found is:

INTERFACE=USBMULTILINK PORT=USB1

#.#.#.# Ethernet IP address #.#.#.#. Each # symbol represents a decimal number between 0 and 255. Valid for Cyclone and Tracelink interfaces.

Connection is via Ethernet.

INTERFACE=CYCLONE PORT=10.0.1.223

NAME Some products, such as the Cyclone and Tracelink, support assigning a name to the unit, such as "Joe's Max". The Cyclone may be referred to by it's assigned name. If there are any spaces in the name, the whole parameter should be enclosed in double quotes (this is a Windows requirement, not a PEMicro requirement).

Examples:

INTERFACE=CYCLONE PORT=MyCyclone99

INTERFACE=CYCLONE "PORT=Joe's Cyclone"

UNIQUEID USB Multilink products all have a unique serial number assigned to them, such as PE5650030. The Multilink may be referred to this number. This is useful in the case where multiple units are connected to the same PC.

Examples:

INTERFACE=USBMULTILINK PORT=PE5650030

COMx — Where x = 1,2,3, or 4. Represents a COM port number. Valid for Cyclone interfaces.

To connect to a Cyclone on COM1 :

INTERFACE=CYCLONE PORT=COM1

x — Where x = 1,2,3, or 4. Represents a parallel port number

To select a parallel interface on Parallel Port #1 :

INTERFACE=PARALLEL PORT=1

[showports] The command-line programmer outputs all available ports to a text file and then terminates (regardless of other command-line parameters). This information output to the text file includes the parameters needed to contact attached programming hardware as well as a description of the hardware interface. The default output filename is ports.txt and is created in the same folder as CPROG.

The output can also be directed to a different file.

Example: SHOWPORTS=C:\MYPORTS.TXT

This list does not show parallel port or COM port options which are also available. Below is an example of the output for various hardware interfaces connected to the PC (Note that there are different ways to address the same unit; the data for each interface may be followed by a [DUPLICATE] line which shows a different label for the same interface).

Showports Output Example:

INTERFACE=USBMULTILINK PORT=PE5650030

```
    ; USB1 : Multilink Universal FX Rev A (PE5650030)[PortNum=21]
INTERFACE=USBMULTILINK   PORT=USB1
    ; USB1 : Multilink Universal FX Rev A (PE5650030)[PortNum=21][DUPLICATE]

INTERFACE=CYCLONE        PORT=10.0.9.197
            ; 10.0.9.197  :  Public Cyclone [PortNum=61]
INTERFACE=CYCLONE        "PORT=Public Cyclone"
            ; 10.0.9.197  :  Public Cyclone[PortNum=61][DUPLICATE]

INTERFACE=CYCLONE        "PORT=Joe's Cyclone"
             ; USB1 : Cyclone (Joe's)[PortNum=101]
INTERFACE=CYCLONE        PORT=USB1
            ; USB1 : Cyclone (Joe's)[PortNum=101][DUPLICATE]

INTERFACE=TRACELINK      PORT=10.1.5.2
        ; 10.1.5.2  :  MCF52259_TRACE[PortNum=123]
INTERFACE=TRACELINK      PORT=MCF52259_TRACE
        ; 10.1.5.2  :  MCF52259_TRACE[PortNum=123][DUPLICATE]
```

[v] Causes the programmer not to check the range of S-record addresses before programming or verifying. This speeds up the programming process. The option should be used with care as all out of range s-records will be ignored.

[reset_delay n] Specifies a delay after the programmer resets the target that we check to see if the part has properly gone into background debug mode. This is useful if the target has a reset driver which hold the MCU in reset after the programmer releases the reset line. The n value is a delay in milliseconds.

[bdm_speed n] This option allows the user to set the BDM shift clock speed of PEMicro's debug interface. This integer value may be used

to determine the speed of communications according to the following equations:

USB Multilink (includes Universal): (1000000/(N+1)) Hz
USB Multilink Universal FX: (25000000/(N+1)) Hz
Cyclone or Tracelink: (50000000/(2*N+5)) Hz
Cyclone: (50000000/(2*N+5)) Hz

The value n should be between 0 and 31. This shift clock takes effect after the commands in the top of the programming algorithm are executed so that these commands can increase the target frequency and allow a faster shift clock. This clock can't generally exceed a div 4 of the processor bus frequency.

[?] Use the '?' character option to cause the command-line programmer to wait and display the result of programming in the PROGACMP window. If the user does not use a batch file to test errorlevel, this provides a method to display the programming result. This option should be the FIRST command-line option.

[hideapp] This will cause the command-line programmer to not display a visual presence while running with the exception of appearing on the taskbar. 32-bit applications only!

[freq n] By default, the PROGACMP software tries to determine automatically how fast the target is running by loading a delay routine in the processor and timing how long it takes to execute. On some machines, this may yield inconsistent results which may affect algorithms which program flash internal to an MCU. PEMicro provides a command-line mechanism allowing the user to inform the PROGACMP software exactly how fast the target processor is running. In this way, the timing in the algorithms will be precise. On the

command-line, you specify the INTERNAL clock frequency in Hertz following the 'FREQ' identifier. Note that in general if you are using a flash device external to the MCU, this timing parameter is not needed as the flash handles the timing itself.

[/logfile logfilename] This option opens a logfile of the name "logfilename" which will cause any information which is written to the status window to also be written to this file. The "logfilename" should be a full path name such as c:\mydir\mysubdir\mylog.log.

Command Line Examples:

CPROGACMP C:\ENGINE.CFG INTERFACE=USBMULTILINK PORT=PE5650030

Opens CPROGACMP with the following options:

- Run the C:\ENGINE.CFG script
- Interface is first Multilink [FX] with serial number PE5650030
- Autodetect communications frequency (io_delay_cnt not set)

CPROGACMP C:\ENGINE.CFG Interface=CYCLONE Port=209.61.110.251

Opens CPROGACMP with the following options:

- Run the C:\ENGINE.CFG script
- Interface is Cyclone LC/FX via the Ethernet Port with an IP address of 209.61.110.251

CPROGACMP C:\ENGINE.CFG Interface=USBMULTILINK Port=USB1

Opens CPROGACMP with the following options:

- Run the C:\ENGINE.CFG script

- Interface is Multilink [FX], first interface detected.

3 Programming Commands

Programming commands all start with a two character sequence followed by white space (blanks or tabs). Lines starting with characters which are not commands are listed as REMarks. The term *filename* means a full DOS path to a file. Commands use the same two letter codes as used in the interactive programmers PROGACMP. The same .ARP files used by PROGACMP are used to set up for a particular device to be programmed. If a user function is specified for a particular device, its two character command and the meaning or *user_par* are specified in the .ARP file.

Note: The command parameters *starting_addr*, *ending_addr, base_addr, byte, word,* and *user_par* use a default hexadecimal format.

BM	- Blank check module.
BR starting_addr ending_addr	- Blank check range.
CHANGEV n.nn	- (Cyclone only) Change the voltage provided to the target, where n.nn represents a value between 0.00 and 5.00, inclusive. When the command executes the Cyclone will immediately change to that voltage. If the Cyclone relays are off prior to calling this command, then the relays will turn on and set the new voltage value when this command is executed. Note that too low of a voltage value may put the device into low-power mode which can lose debug communication altogether. Make sure the Cyclone's jumper settings are set correctly to send the power to the right ports.
EB starting_addr ending_addr	- Erase byte range.
EW starting_addr ending_addr	- Erase word range.
EM	- Erase module.
PB starting_addr byte ... byte	- Program bytes.
PW starting_addr word ... word	- Program words.

PM	- Program module.
PU	- Program user options specified in selected user options file. See **Section 3.1 - User Options Programming**.
CM filename base_addr	- Choose module .ARP file. Note: Certain modules may require a base address to be specified.
VM	- Verify module.
VR starting_addr ending_addr	- Verify range.
UM filename	- Upload module.
UR starting_addr ending_addr filename	- Upload range.
SS filename	- Specify S record.
SM starting_addr ending_addr	- Show module.
SU filepath	- Specify path to user options file. See **Section 3.1 - User Options Programming**.
RELAYSOFF	- (MultiInk FX & Cyclone only) Turn off the relays that provide power to the target, including a power down delay if specified. Especially useful for users who want to power cycle their board before running tests, allow their bootloader to run, or have the application code run after programming.
RELAYSON	- (MultiInk FX & Cyclone only) Turn on the relays to provide power to the target, including a power up delay if specified. The voltage supplied will be based on the last voltage setting specified. For Cyclone users, the CHANGEV command can change the voltage value. Especially useful for users who want to power cycle their board before running tests, allow their bootloader to run, or have the application code run after programming.

HE	- Help (look at cprog.doc file).
QU	- Quit.
RE	- Resets and halts chip.
GO	- Toggles hardware reset line and starts device running. Can be used as final command if you want the device to run for testing.
DE timeinms	- Delays "timeinms" milliseconds
xx user_par	- Only for user function specified in .ARP file.

3.1 User Options Programming

Some ARM devices have areas of flash memory dedicated to programming user configuration data. As some writes to such areas can be sensitive or permanent, it is important that the developer is able to write these options correctly the first time and avoid mis-programming adjacent options that they wish to leave untouched. As of version 7.78 of CPROGACMP, a set of two new commands has been introduced:

- Specify User Options File (SU)
- Program User Options (PU)

These commands allow the developer to individually program user options with an existing user options (.OPT) file. Information regarding how to create a .OPT file is available in the PROGACMP User Manual or Cyclone LC/FX User Manual. A list of supported devices is available at:

http://www.pemicro.com/blog/index.cfm?post_id=177

3.1.1 Specify User Options File (SU)

With the Specify User Options File (SU) command, the user specifies the path to an existing user options (.OPT) file. For example:

```
SU C:\PEMicro\PROGACMP\supportFiles_ARM\Atmel\SAME\myFile.OPT
```

3.1.2 Program User Options (PU)

Once a file has been specified, the Program User Options (PU) command can be used to write the values specified by the file. For most devices, new option values won't take effect until the device is reset.

4 Configuration Commands For Startup

Configuration commands are all processed before the programmer attempts to contact the target. The whole configuration file is parsed for these commands prior to attempting communications. This section gives an overview of using these configuration commands to do different type of configuration.

In the GUI version (PROGACMP) the user would select a check box to connect via SWD, select a check box to provide power to the target, type power up and power down delays into text boxes, etc. Obviously, this is not possible in the command line version of PROGACMP. In CPROGACMP the user can achieve this with configuration commands.

Configuration commands should be placed at the beginning of the .CFG file. For example, to tell the hardware interface to connect to the target using SWD, instead of checking the SWD box you would place the configuration command **:useswd 1** at the beginning of your configuration (.CFG) file.

Note: The default base for configuration command parameters is decimal.

An overview of the configuration commands is as follows:

:DEVICE xxx

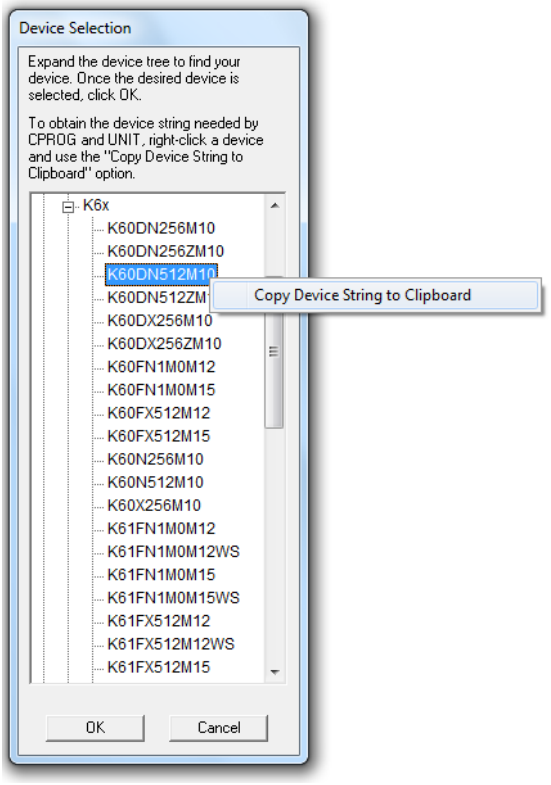
This is a mandatory parameter that specifies the device. It is required because the debug protocol changes between manufacturers and sometimes also between families or devices.

Where:

xxx: represents the device and follows the “VENDOR_FAMILY_DEVICE” format.

The easiest way to obtain the device string (xxx) is from the Device Selection dialog in the PROGACMP software. In the PEMICRO Connection Manager, click “Select New Device” to open the Device Selection dialog. Expand the device tree to find your device, then right-click and select “Copy Device String to Clipboard.”

Figure 4-1: Device Selection Dialog



Example:

:DEVICE Freescale_K6x_K60DN512M10

:CUSTOMTRIMREF nnnnnnnn.nn

Desired internal reference clock frequency for the “PT; Program Trim” command. This frequency overrides the default internal reference clock frequency. Valid values for “n” depend on the particular device being programmed. Please refer to the electrical specifications of your device for valid internal reference frequency clock range. Where:

nnnnnnnn.nn: Frequency in Hertz with two decimal places

:DEVICEPOWER n

For Cyclone (excludes Cyclone MAX). This setting defines the target voltage that will be provided to the target if the source of the voltage is derived from the Cyclone's internal power. Valid values of n are:

0 : 5 Volts, Generated/Switched by Cyclone

2 : 3 Volts, Generated/Switched by Cyclone

4 : 2 Volts, Generated/Switched by Cyclone

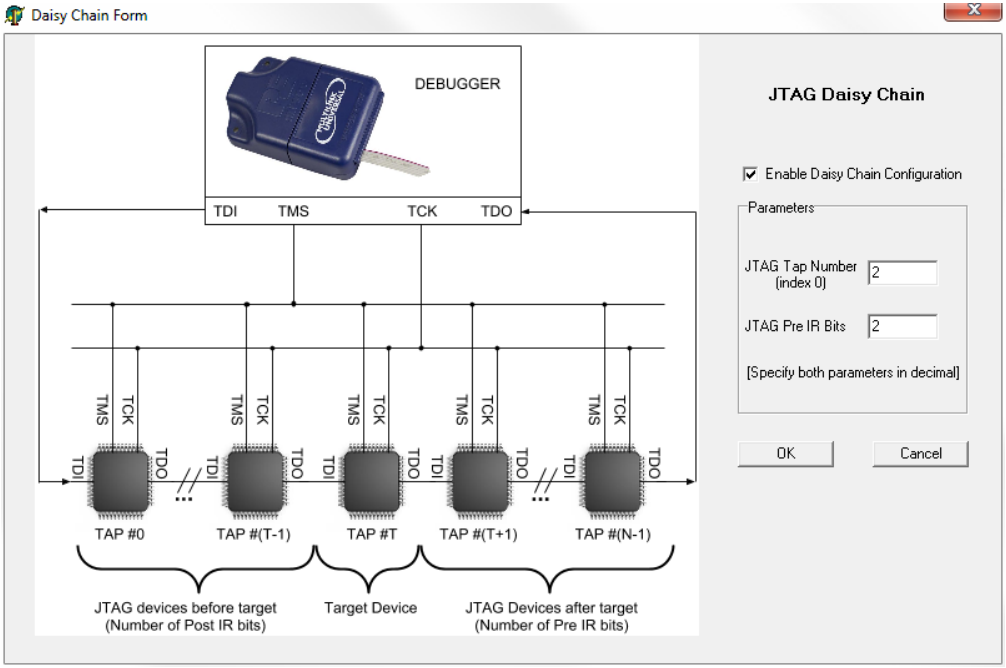
:JTAGPREIR n

This is mandatory for JTAG daisy chain configurations and specifies the total length of the IR registers following the target device. It is important to specify JTAG communications using the :USESWD 0 command.

:JTAGTAPNUM n

This is mandatory for JTAG daisy chain configurations and selects the index of the target device in the daisy chain. Index 0 specifies the first device in the daisy chain. It is important to specify JTAG communications using the :USESWD 0 command.

Figure 4-2: Daisy Chain Setup Dialog



:PROVIDEPOWER n

Determines whether interface should provide power to the target. NOTE: Not all hardware interfaces support this command. Valid values of n are:

0 : Interface does not provide power to target. (default)

1 : Enable Interface provides power to target.

(NOTE: Is the same as legacy option **:USEPRORELAYS n**)

:POWERDOWNDELAY n

Amount of time to delay when the power to the target is turned off for the targets power supply to drop to below 0.1v. n is the time in milliseconds.

:POWERUPDELAY n

Amount of time to delay when the power to the target is turned on OR the target is reset, and before the software attempts to talk to the target. This time can be a combination of power on time and reset time (especially if a reset driver is used). n is the time in milliseconds.

:POWEROFFONEXIT n

Determines whether power provided to the target should be turned off when the CPROGACMP application terminates. NOTE: Not all hardware interfaces support this command. Valid values of n are:

0 : Turn power off upon exit (default)

1 : Keep power on upon exit

:USESWD n

Specifies whether to use SWD or JTAG protocol. Valid values of n are:

0 : use JTAG protocol

1 : use SWD protocol

Example Configuration File

The following example illustrates a .CFG file with 4 configuration commands. This specific example would be for connecting a Multilink FX to a Freescale KL05Z32 board. With the configuration commands used here, the FX would connect to the KL05Z32 using SWD and provide power to the target with a power up delay of 4000 ms and a power down delay of 250 ms.

```
:device Freescale_KL0x_KL05Z32M4
    :useswd 1
    :powerdowndelay 250
    :powerupdelay 4000
    :providepower 1
    CM
    C:\pemicro\progacmp\supportFiles_ARM\freescale_kl05z32m4_1x32x8k_pflash
    .arp
    SS C:\test\kinetis\kl05z.s19
    EM
```

BM

PM

VM

5 Verification Overview

There are several commands available that can be used to verify the contents of the flash on the device after programming it. The most widely used command is “VC ;Verify CRC of Object File to Module”. The “VC” command will instruct CPROGACMP to first calculate a 16-bit CRC value from the chosen object file. CPROGACMP will then load code into the RAM of the device and instruct the device to calculate a 16 bit CRC value from the contents in FLASH of the device. Only valid address ranges in the object file are calculated on the device. Once the 16-bit CRC value from the object file and the device are available, CPROGACMP compares them. An error is thrown if the two values do not match.

Alternatively, the “VM ;Verify Module” command can be used to perform a byte by byte verification between the chosen object file and the device. Typically, the VM command will take longer to perform than VC command since CPROGACMP has to read the contents of FLASH of the device byte by byte. There are also two other commands that can be used for verification. The “SC ;Show Module CRC” instructs CPROGACMP to load code into the RAM of the device and instruct the device to calculate a 16-bit CRC value from the contents of the entire FLASH of the device, which includes blank regions. Once the 16-bit CRC value has been calculated, CPROGACMP will display the value in the status window. The “VV ;Verify Module CRC to Value” command is similar to the “SC” command. The difference is that instead of displaying the calculated 16-bit CRC value, CPROGACMP will compare the calculated value against a 16-bit CRC value given by the user.

6 DOS Error Returns

DOS error returns are provided so they may be tested in .BAT files. The error codes used are:

0 - Program completed with no errors.

1 - Cancelled by user.

2 - Error reading S record file.

3 - Verify error.
4 - Verify cancelled by user.
5 - S record file is not selected.
6 - Starting address is not in module.
7 - Ending address is not in module or is less than starting address.
8 - Unable to open file for uploading.
9 - File write error during upload.
10 - Upload cancelled by user.
11 - Error opening .ARP file.
12 - Error reading .ARP file.
13 - Device did not initialize.
14 - Error loading .ARP file.
15 - Error enabling module just selected.
16 - Specified S record file not found.
17 - Insufficient buffer space specified by .ARP to hold a file S-record.
18 - Error during programming.
19 - Start address does not point into module.
20 - Error during last byte programming.
21 - Programming address no longer in module.
22 - Start address is not on an aligned word boundary.
23 - Error during last word programming.
24 - Module could not be erased.
25 - Module word not erased.
26 - Selected .ARP file does not implement byte checking.
27 - Module byte not erased.
28 - Word erase starting address must be even.
29 - Word erase ending address must be even.
30 - User parameter is not in the range.
31 - Error during .ARP specified function.
32 - Specified port is not available or error opening port.

33 - Command is inactive for this .ARP file.

34 - Cannot enter background mode. Check connections.

35 - Not able to access processor. Try a software reset.

36 - Invalid .ARP file.

37 - Not able to access processor RAM. Try a software reset.

38 - Initialization cancelled by user.

39 - Error converting hexadecimal command number.

40 - Configuration file not specified and file prog.cfg does not exist.

41 - .ARP file does not exist.

42 - Error in io_delay number on command line.

43 - Invalid command line parameter.

44 - Error specifying decimal delay in milliseconds.

47 - Error in script file.

49 - Cable not detected

50 - S-Record file does not contain valid data.

51 - Checksum Verification failure - S-record data does not match MCU memory.

52 - Sorting must be enabled to verify flash checksum.

53 - S-Records not all in range of module. (see "v" command line parameter)

54 - Error detected in settings on command line for port/interface

55 - Missing device parameter in script file

60 - Error calculating device CRC value

61 - Error - Device CRC does not match value given

70 - Error - CPROG is already running

71 - Error - Must specify both the INTERFACE and PORT on the command line

72 – The selected target processor is not supported by the current hardware interface.

7 Example Programming Script File

The programming script file should be a pure ASCII file with one command per line. This is the CFG file in the previous examples.

An example is:

```
RE                                              ;Reset the MCU
CM C:\PEMICRO\Freescale_MK40X256_PFlash_DFlash.ARP
                                                ;Choose Flash Module
EM                                              ;Erase the module
BM                                              ;Blank Check the module
SS C:\PEMICRO\TEST.S19                          ;Specify the S19 to use
PM                                              ;Program the module with the S19
VM                                              ;Verify the module again
```

Note: The path names of files that are relative to the CPROG executable can also be used.

8 Using Command-Line Parameters in a Script

A command-line parameter in the form of /PARAMn=s can be used to insert text into the script file in place of special tags. This can be used to replace any part of the script including programming commands, filenames, and parameters. Valid values of n are 0..9. s is a string which will replace any occurrence of /PARAMn in the script file.

As an example, the following generic script could be used for programming with exactly the same functionality of the example script in **Section 7 - Example Programming Script File**:

```
RE                  ;Reset the MCU
CM /PARAM1          ;Choose Flash Module
EM                  ;Erase the module
BM                  ;Blank Check the module
SS /PARAM2          ;Specify the S19 to use
PM                  ;Program the module with the S19
/PARAM3             ;Verify the module again
```

The following parameters would be added to the CPROG command line:

/PARAM1=C:\PEMICRO\Freescale_MK40X256_PFlash_DFlash.ARP

/PARAM2=C:\PEMICRO\TEST.S19

/PARAM3=VM

NOTE: If a /PARAMn parameter has a space in its value, the entire parameter needs to be enclosed in double quotations. This indicates to Windows that it is a single parameter. For example, if the path in /PARAM2 above contained a space, you would need to specify it on the command line like this:

“/PARAM2=C:\PEMICRO\EXAMPLE FILES\TEST.S19"

So the complete example command line would be (note that this is continuous; no line breaks):

C:\PROJECT\CProgARMCortex INTERFACE=CYCLONE PORT=USB1 BDM_SPEED 1 C:\PROJECT\GENERIC.CFG /PARAM1=C:\PEMICRO\Freescale_MK40X256_PFlash_DFlash.ARP “/PARAM2=C:\PEMICRO\EXAMPLE FILES\TEST.S19” /PARAM3=VM

9 Sample Batch File

Here is an example of calling the command-line programmer and testing its error code return in a simple batch file. Sample batch files are given for both Windows 95/98/XP and Windows 2000/NT/XP/Vista/7/8/10.

Windows NT/2000/Vista/7/8/10:

C:\PROJECT\CPROGACMP C:\PROJECT\ENGINE.CFG INTERFACE=USBMULTILINK PORT=USB1

if errorlevel 1 goto bad

goto good

:bad

ECHO BAD BAD BAD BAD BAD BAD BAD BAD

:good

ECHO done

Windows 95/98/ME/XP:

START /W C:\PROJECT\CPROGACMP C:\PROJECT\ENGINE.CFG INTERFACE=USBMULTILINK PORT=USB1

if errorlevel 1 goto bad

goto good

:bad

ECHO BAD BAD BAD BAD BAD BAD BAD BAD

:good

ECHO done

Note: The path names of files that are relative to the CPROG executable can also be used.

10 Information

For more information on CPROGACMP and PROGACMP please contact us:

P&E Microcomputer Systems, Inc.
98 Galen St.
Watertown, MA 02472-4502
USA

VOICE: (617) 923-0053
FAX: (617) 923-0808
WEB: http://www.pemicro.com

To view our entire library of.ARP modules, go to the Support page of PEMicro's website at www.pemicro.com/support.