Emulation Extension Pak (EEP) and Emulation Header User's Guide

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Trademarks

ISBN: 978-1-63277-835-2

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Table of Contents

Chapter 1. EEP and Emulation Header Overview

1.1 What is an Emulation Extension Pak? 5
1.2 What is an Emulation Header? 5
1.3 Why Would I Want to Use an Emulation Header? 6
1.4 Compare Emulation Header, Debug Header and Device Features 6
1.5 Support Information 8
1.6 Emulation Header Hardware Setup 9
1.7 Emulation Header Setup for MPLAB X IDE 11
1.8 Additional Information 11

Chapter 2. Emulation Header Features

2.1 Introduction 13
2.2 Breakpoint, Runtime Watch, and Trace Resources 14
2.3 Runtime Watches 14
2.4 Real Time Hardware Instruction Trace 15
2.5 Hardware Address/Data Breakpoints 19
2.6 Enhanced Event Breakpoints 22
2.7 Event Combiners 23
2.8 Stopwatch Cycle Counter 25
2.9 Trigger In/Out 25
2.10 View Hardware Stack On Halt 27
2.11 Previous Program Counter 28
2.12 Background Debug 28

Chapter 3. Emulation Header List

3.1 Introduction 29
3.2 AC244055 31
3.3 AC244063 33
3.4 AC244064 35
3.5 AC244065 37
3.6 AC244066 39

Appendix A. -ME2 Silicon Errata

A.1 Introduction 41
A.2 CCP3 Capture 41
A.3 Hardware Breakpoint Issue 41
A.4 Trigger In/Halt during Multi-Cycle Instruction Processing 42

Appendix B. Emulation Header Target Footprints

B.1 Introduction 43
B.2 DIP Device Footprints 43
B.3 TQFP/PLCC Device Footprints 43

Appendix C. Emulation Header Connections

C.1 Introduction 45
C.2 6-Pin Modular Connector 45
C.3 6-Pin SIL Connector 46
C.4 Modular-to-SIL Adapter 47
C.5 Ordering Information 47

Index 51

Worldwide Sales and Service 54

Chapter 1. EEP and Emulation Header Overview

NOTICE TO CUSTOMERS

All documentation becomes dated, and this manual is no exception. Microchip tools and documentation are constantly evolving to meet customer needs, so some actual dialogs and/or tool descriptions may differ from those in this document. Please refer to our web site (www.microchip.com) to obtain the latest documentation available.

Documents are identified with a "DS" number. This number is located on the bottom of each page, in front of the page number. The numbering convention for the DS number is "DSXXXXXA", where "XXXXX" is the document number and "A" is the revision level of the document.

For the most up-to-date information on development tools, see the MPLAB X IDE online Help (Help menu).

This chapter contains the following topics:

- What is an Emulation Extension Pak?
- What is an Emulation Header?
- Why Would I Want to Use an Emulation Header?
- Compare Emulation Header, Debug Header and Device Features (Future)
- Support Information
- Emulation Header Hardware Setup
- Emulation Header Setup for MPLAB X IDE
- Additional Information

1.1 WHAT IS AN EMULATION EXTENSION PAK?

An Emulation Extension Pak (EEP) contains an emulation header; gold, single, in-line pins; a trace cable, and a trace adapter board. An EEP is what you purchase when you want an emulation header.

1.2 WHAT IS AN EMULATION HEADER?

An emulation header is a circuit board that allows a debug tool to debug code for a specific device. A special version of the device (-ME2) with on-board emulation circuitry is located on the header. Connectors on the side of the header allow it to connect directly to, or through, an adapter to the emulator. Connectors on the bottom of the header allow it to connect directly to, or through, a transition socket to a target board.

1.3 WHY WOULD I WANT TO USE AN EMULATION HEADER?

Although some devices have on-board debug circuitry to allow you to debug your code, you often lose device resources to debugging, i.e., debugging requires the use of two I/O lines, plus VDD, VSS and VPP, to communicate with the device. Using an emulation header can free up these resources for your application, and give you new and powerful debugging features not found on debug headers.

For details on emulation header features, see Chapter 2. "Emulation Header Features".

For details on debug header features, see the user's guide or online help file for supported debug tools (listed in Section 1.5 "Support Information"). Also see the *Processor Extension Pak and Debug Header Specification* (DS51292). Find all documentation on the Microchip website (http://www.microchip.com).

For a comparison of emulation versus debug features, see Section 1.4 "Compare Emulation Header, Debug Header and Device Features (Future)".

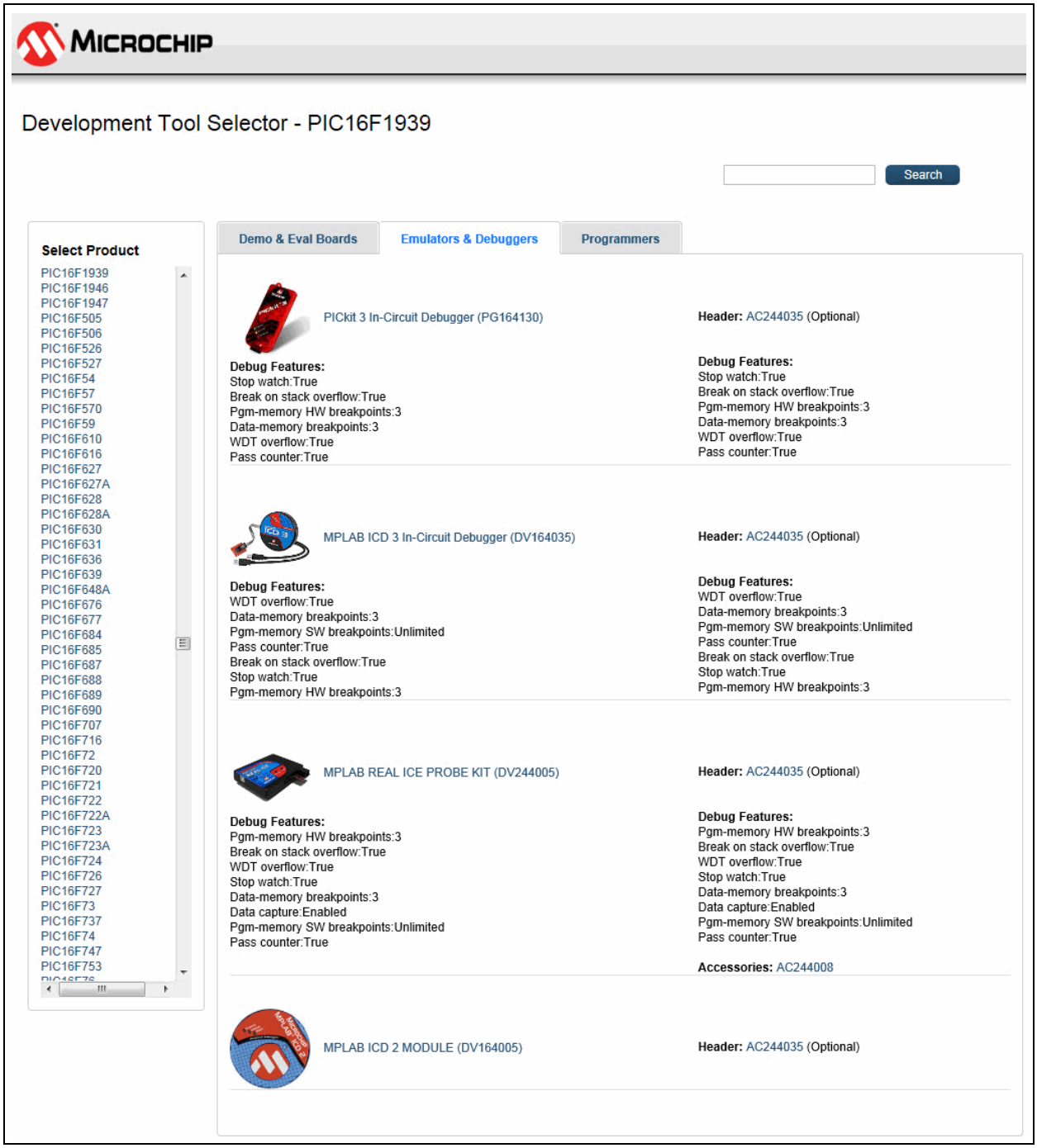
1.4 COMPARE EMULATION HEADER, DEBUG HEADER AND DEVICE FEATURES (FUTURE)

Use the Development Tool Selector (DTS) to compare the extra debug features of an emulation header to a supported device (with on-board debug circuitry) and supported debug header.

To find features by device:

1. In a web browser, go to: http://www.microchip.com/dtsapp/
2. Select your device from the "Select Product" list. Or type and search for the name of your device in the "Search" box and it will appear at the top of the "Select Product" list, where you can select it.
3. Click on the tab "Emulators & Debuggers" to see debug features.

FIGURE 1-1: DTS DEVICE INFORMATION



1.5 SUPPORT INFORMATION

Emulation headers require specific MPLAB X IDE versions and debug tools to operate. Acquire these before purchasing an emulation header in an emulation extension pak (EEP). Available EEPs are listed in Chapter 3. "Emulation Header List".

To proceed with setting up emulation header hardware, see Section 1.6 "Emulation Header Hardware Setup".

Contact Customer Support for issues with emulation headers.

1.5.1 Software Support

Emulation headers are supported on MPLAB X IDE v1.90 and greater.

1.5.2 Tool Support

Emulation headers are supported on the following tools:

- PICkit™ 3 in-circuit debugger
- MPLAB® ICD 3 in-circuit debugger
- MPLAB® REAL ICE® in-circuit emulator

Note:	Not all features are supported on all tools.

1.5.3 Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Technical support is available through the web site at: http://support.microchip.com

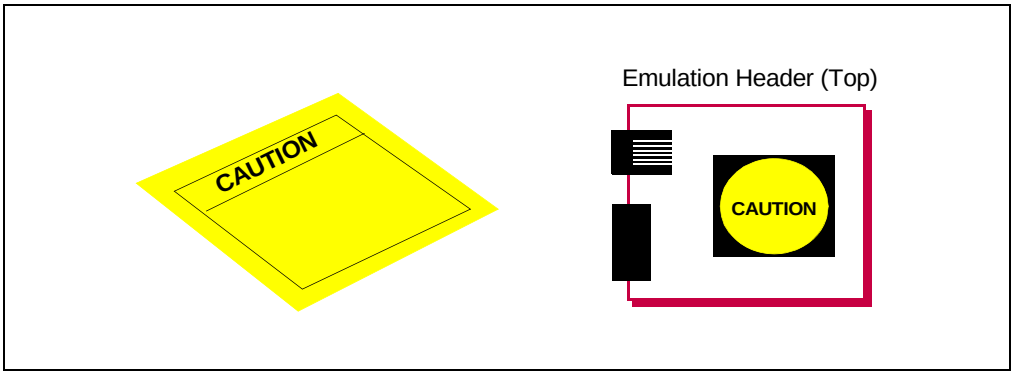
Documentation errors or comments may be sent to: docerrors@microchip.com.

1.6 EMULATION HEADER HARDWARE SETUP

To set up your header, follow these instructions before doing anything else:

1. Check the header box for any paper inserts that specify special operating instructions and the emulation header for any stickers (Figure 1-2).

FIGURE 1-2: SPECIAL HEADER INSTRUCTIONS



2. Set any jumpers or switches on the header to determine device functionality, or selection, as specified for that header. See the section “Emulation Header List” for information on how to set up individual headers.
3. Connect the header to your desired debug tool by consulting the tool documentation for connection options. Example connections are shown in Figure 1-3, Figure 1-4, and Figure 1-5.

FIGURE 1-3: PICkit™ 3 IN-CIRCUIT DEBUGGER CONNECTIONS

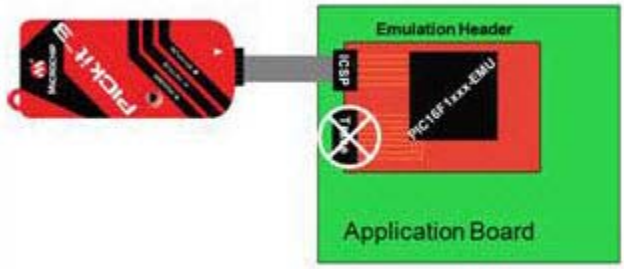


FIGURE 1-4: MPLAB® ICD 3 IN-CIRCUIT DEBUGGER CONNECTIONS

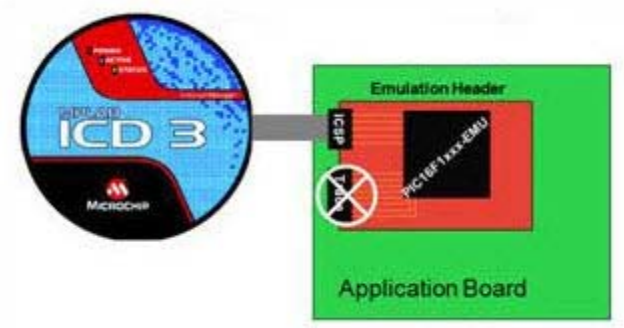
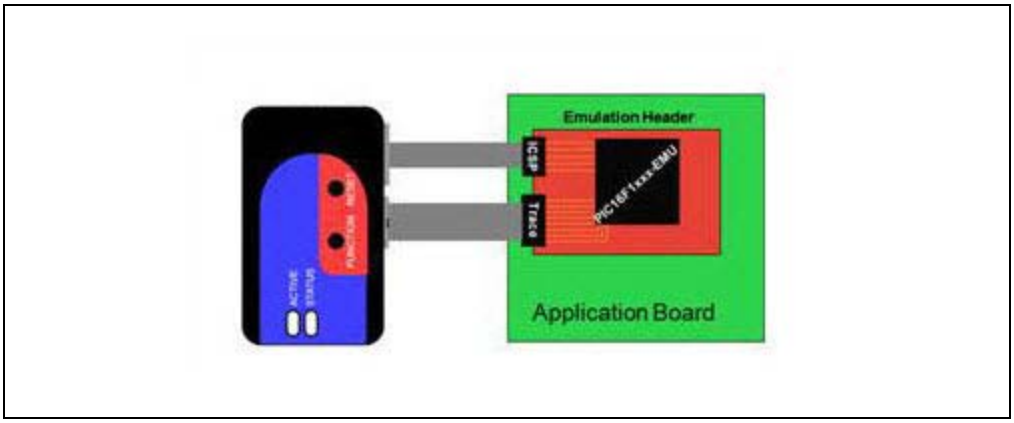


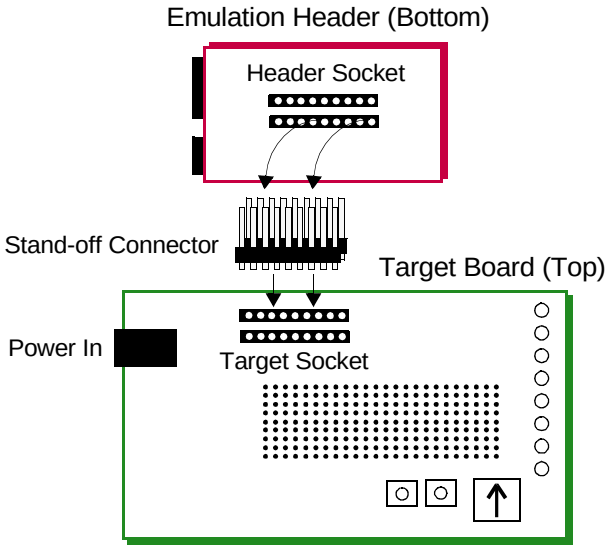
FIGURE 1-5: MPLAB® REAL ICE™ IN-CIRCUIT EMULATOR CONNECTIONS



4. Connect the header to the target board. On the bottom of the header is a socket that is used to connect to the target board. The header can be connected to the target board as follows:
   a) PDIP header socket to PDIP target socket with a stand-off (male-to-male) connector or single in-line pins. An example is shown in Figure 1-6.
   b) Header socket to plug on the target board
   c) Header socket to target socket with a transition socket (see the *Transition Socket Specification*, DS51194)

   The header socket will have the same pin count as your selected device. The -ME2 device on the top of the header usually has a larger pin count because it has additional pins that are dedicated to debug.

FIGURE 1-6: CONNECT HEADER TO TARGET



5. If using a debug tool that can power the target, power that tool now.
6. Power the target, if needed.

1.7 EMULATION HEADER SETUP FOR MPLAB X IDE

Emulation header functionality is supported on MPLAB X IDE 1.90 or greater, but **not** on MPLAB IDE v8. Please use debug headers, if you are still using MPLAB IDE v8.

You need to do the following to use an emulation header on MPLAB X IDE:

1. Set up the emulation header as specified in Section 1.6 “Emulation Header Hardware Setup”.
2. Begin creating a project for a device supported by your emulation header using the Projects wizard (*File>New Project*). See MPLAB X IDE documentation for more on projects.
3. In one step of the wizard you will have an opportunity to specify the emulation header product number (AC######).
4. In another step you will specify the hardware (debug) tool to which the emulation header is attached.
5. Once the wizard is complete, write code for your project.
6. Select *Debug>Debug Project* to run and debug your code.

Note: An emulation header can only be used to debug (Debug menu), not to program (Run menu). See “Section 1.8.1 “Programming Notes””.

1.8 ADDITIONAL INFORMATION

The following additional information is useful when using an emulation header from an Emulation Extension Pak.

1.8.1 Programming Notes

The emulation header is designed to be used with the in-circuit emulator in debugger mode, *Debug>Debug Project,* (not in programmer mode, *Run>Run Project*), in MPLAB X IDE. Any programming of the special -ME2 device on the header is for debug purposes.

To program production (non-special) devices with your debug tool, use the *Universal Programming Module* (AC162049) or design a modular interface connector on the target. See the appropriate specification for connections. For the most up-to-date device programming specifications, see the Microchip website at http://www.microchip.com.

Also, production devices can be programmed with the following tools:

- MPLAB PM3 device programmer
- PICkit 3 development programmer
- MPLAB ICD 3 in-circuit debugger (select as a programmer)
- MPLAB REAL ICE in-circuit emulator (select as a programmer)

1.8.2 Calibration Bits

The calibration bits for the band gap and internal oscillator are always preserved to their factory settings.

1.8.3 Performance Issues

See the MPLAB X IDE help file regarding your debug tool for information on specific device limitations that could affect performance.

NOTES:

Chapter 2. Emulation Header Features

2.1 INTRODUCTION

Emulation header features depend on the debug tool used. The table below shows a list of all available emulation header features, and the features that are supported on each tool.

TABLE 2-1: EMULATION FEATURE SUPPORT BY HARDWARE TOOL

Features	RI	ICD3	PK3
Section 2.3 "Runtime Watches" - see Note	✓	✗	✗
Section 2.4 "Real Time Hardware Instruction Trace" - see Note	✓	✗	✗
Section 2.5 "Hardware Address/Data Breakpoints" - see Note Section 2.5.2 "Range Breakpoints" Section 2.5.3 "Data Value Comparison" Section 2.5.4 "Data Value Mask" Section 2.5.6 "Trigger Out Operation" Section 2.5.7 "Interrupt Context Detection"	✓	✓	✓
Section 2.6 "Enhanced Event Breakpoints" - see Note Section 2.6.1 "Execution Out-of-Bounds Detection" Section 2.6.2 "Break on Trigger In/Emit Trigger Out"	✓	✓	✓
Section 2.7 "Event Combiners"	✓	✓	✓
Section 2.8 "Stopwatch Cycle Counter"	✓	✓	✓
Section 2.9 "Trigger In/Out"	✓	✓	✓
Section 2.10 "View Hardware Stack On Halt"	✓	✓	✓
Section 2.11 "Previous Program Counter"	✓	✓	✓
Section 2.12 "Background Debug"	✓	✓	✓

Legend:
RI = MPLAB® REAL ICE™ In-Circuit Emulation
ICD3 = MPLAB® ICD 3 In-Circuit Debugger
PK3 = PICkit™ 3 In-Circuit Debugger

Note: See also Section 2.2 "Breakpoint, Runtime Watch, and Trace Resources".

2.2 BREAKPOINT, RUNTIME WATCH, AND TRACE RESOURCES

Emulation headers have 32 data capture resources, total, that may be used for breakpoints and runtime watches. For example, if you use a data capture resource for a breakpoint, you will have one less data capture resource for other breakpoints or runtime watches.

Runtime watches and trace are mutually exclusive. Data streaming from the emulation header may be used for either runtime watches or trace. For example, if you want to use trace, you cannot use runtime watches.

Selecting trace in the Project Properties takes precedence over other selections, such as in the Watches window or in any plug-ins.

2.3 RUNTIME WATCHES

In the MPLAB X IDE Watches window (*Window>Debugging>Watches*), you can create a watch for a symbol where the value changes at runtime.

Before you add a runtime watch to the Watches window, you need to set up the clock. Perform the following steps to set up the clock:

1. Right click on the project name and select "Properties".
2. Click on the debug tool name (e.g., Real ICE) and select option category "Clock".
3. Set the runtime instruction speed.

To add a global symbol or SFR as a runtime watch, do one of the following:

- Right click in the Watches window and select "New Runtime Watch" or select *Debug>New Runtime Watch*. Click the selection buttons to see either Global Symbols or SFRs. Click on a name from the list and then click **OK**.
- Select the symbol or SFR name in the Editor window and then select "New Runtime Watch" from the right click menu. The name is populated in the Watches window. Click **OK**.

Debug Run and watch the values change as the program executes. For more information on the Watches window, see MPLAB X IDE documentation.

2.3.1 Symbol/SFR Size

For all devices except PIC32 MCUs, symbols used in a runtime watch must be sized to match the device memory. That is, you need 8-bit symbols when using an 8-bit device.

2.3.2 SFR Caveats

Unless an SFR is explicitly read or written by code, no value change will be visible in the Watches window. As the runtime watch uses the trace mechanism, code must read or cause a change before a trace packet is output. Therefore any SFR value changes, without code explicitly reading it or causing the SFR value change, cannot be directly monitored as a Runtime Watch.

For example, setting up a Runtime Watch on the Analog-to-Digital Converter (ADC) conversion result SFRs ADRESH and ADRESL will not show any runtime changes, even when the input voltage to the ADC is changing. A workaround is code that copies the ADRESH and ADRESL register values to RAM variables (e.g., ADRESH_copy and ADRESL_copy') after an ADC conversion is complete. These variables, added to the Watches window as Runtime Watches, will then reflect the changes in ADC values.

Another example would be monitoring an input pin or an entire port that is configured as inputs. External changes on the input pin(s) will not trigger a trace watch packet; and, therefore, no runtime updates to the port will be displayed. The Runtime Watch display will only update when code reads the input pin or input port.

2.4 REAL TIME HARDWARE INSTRUCTION TRACE

Note: This feature is only supported on the MPLAB REAL ICE in-circuit emulator.

Real Time Hardware Instruction Trace is a real-time dump of the program execution stream that can be captured and analyzed. The functionality that is provided includes the following:

- full instruction execution information, up to 32 MHz
- trace through Reset conditions
- trace buffer with optional stall

Emulation header trace is similar to PIC32 instruction trace, which is a non-intrusive hardware instruction trace. You can use trace to capture every instruction executed by the device. The trace data is output from the device (using the pins TRCLK, TRDAT[6:0], and TRSTALL) to the emulator. The emulator streams this data to a trace buffer, that acts like a rolling first-in/first-out (FIFO) buffer, on the PC.

The amount of trace data is limited only by the size of the trace buffer. This buffer can fill quickly even when set to the maximum size, so it is wise to determine exactly what you need to capture.

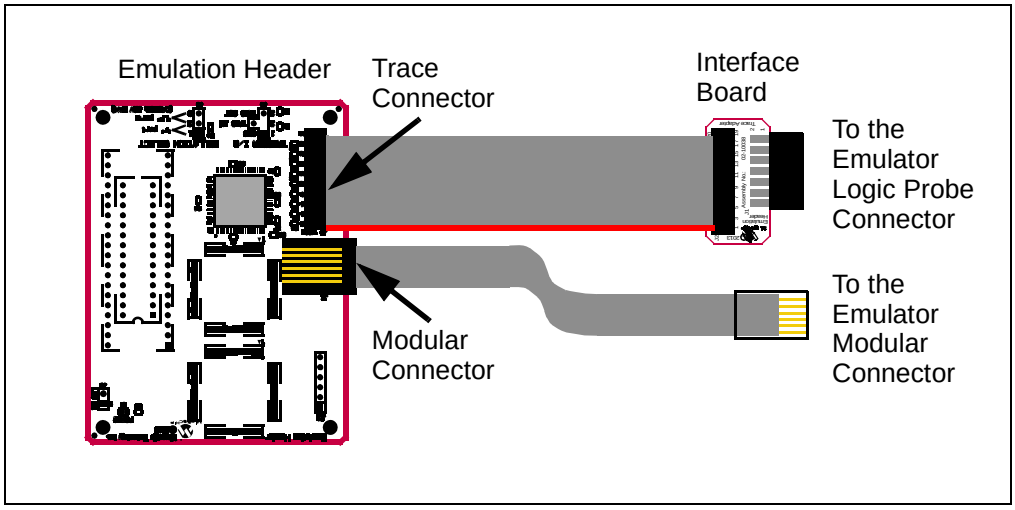
Note: Execution will NOT HALT when this external buffer is full. However, MPLAB X IDE will announce when the trace buffer has overflowed.

Trace requires hardware and software setup before trace data can be viewed in a window.

2.4.1 Set Up Trace Hardware

To use trace, you will need the ribbon cable and emulator interface board that comes with the emulation header. Connect one end to the emulation header (Figure 2-1). Connect the other end to the emulator. For more information on complete hardware setup and connections, refer to Section 1.6 “Emulation Header Hardware Setup”. For details on trace connectors, see Section 2.4.4 “Trace Hardware”.

FIGURE 2-1: EMULATION TRACE CABLE CONNECTED TO HEADER

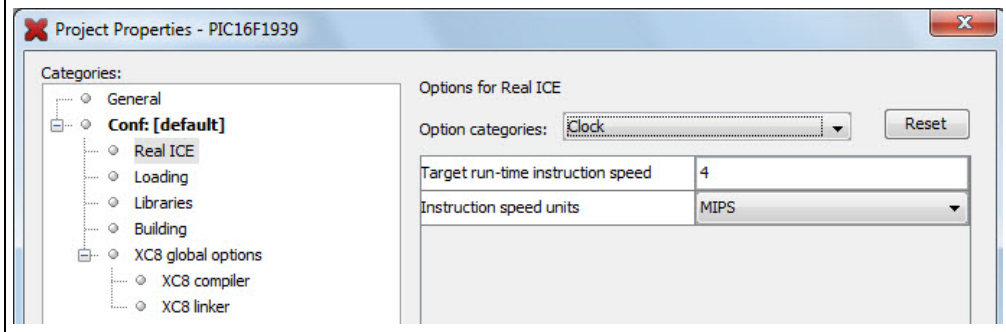


2.4.2 Set Up Trace for MPLAB X IDE

To set up MPLAB X IDE to use trace for the MPLAB REAL ICE in-circuit emulator, perform the following steps:

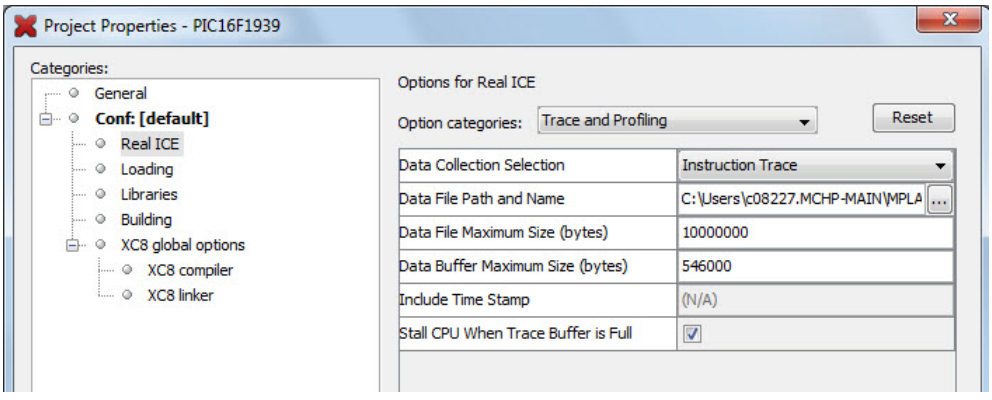
1. Right click on the project name and select "Properties" to open the Project Properties window.
2. Click on "Real ICE" under "Categories".
3. Under "Option categories", select "Clock". Specify the target run-time instruction speed and speed units.

FIGURE 2-2: CLOCK OPTION CATEGORY



4. Under "Option categories", select "Trace and Profiling".
5. Under "Data Collection Selection", choose "Instruction Trace/Profiling".
6. Set up any other trace-related options.

FIGURE 2-3: TRACE AND PROFILING OPTION CATEGORY



7. Click **OK**.

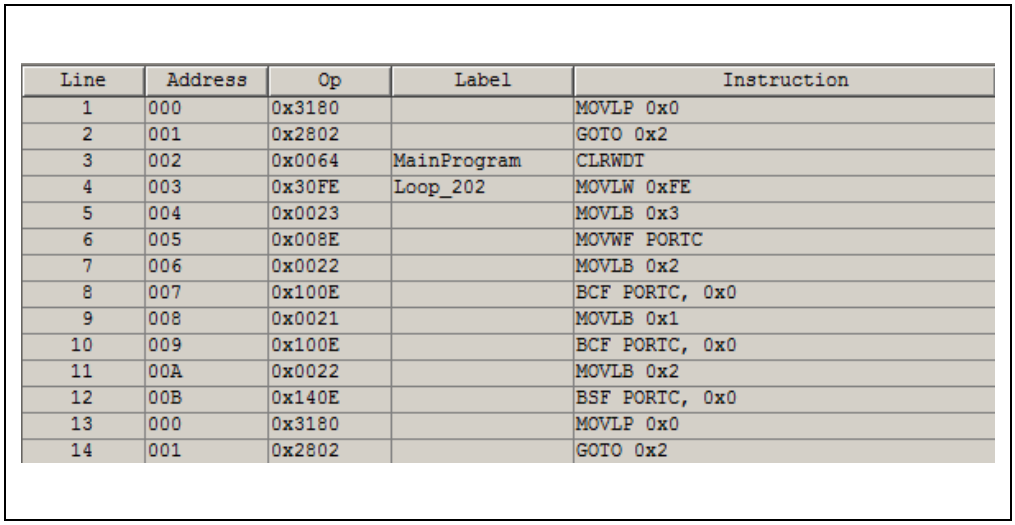
2.4.3 Using Trace

Once trace hardware and MPLAB X IDE are set up, you can begin using trace. On a Debug Run, trace will continue to fill the trace buffer with data, rolling over when the buffer is full, until a program Halt. For more on the trace buffer, see Section 2.4 “Real Time Hardware Instruction Trace”.

2.4.3.1 VIEWING TRACE DATA

When trace is enabled and code is run, trace data will be collected by the emulator. Once the device is halted, trace data will be decoded and displayed in the Trace window (*Window>Debugging>Trace*).

FIGURE 2-4: TRACE WINDOW



Line	Address	Op	Label	Instruction
1	000	0x3180		MOVLP 0x0
2	001	0x2802		GOTO 0x2
3	002	0x0064	MainProgram	CLRWDT
4	003	0x30FE	Loop_202	MOVLW 0xFE
5	004	0x0023		MOVLB 0x3
6	005	0x008E		MOVWF PORTC
7	006	0x0022		MOVLB 0x2
8	007	0x100E		BCF PORTC, 0x0
9	008	0x0021		MOVLB 0x1
10	009	0x100E		BCF PORTC, 0x0
11	00A	0x0022		MOVLB 0x2
12	00B	0x140E		BSF PORTC, 0x0
13	000	0x3180		MOVLP 0x0
14	001	0x2802		GOTO 0x2

2.4.3.2 BANK SELECT AND TRACE

For PIC12F/16F1xxx devices and related headers, Trace can decode instructions resulting in a bank set. Once established, decoding with that bank continues until the next bank switch. Therefore, scrolling up in the window will cause the bank to be indeterminate, where as scrolling down sets the bank.

2.4.3.3 IMPROVING THE TRACE EXPERIENCE

Remove as many USB devices from your PC USB ports as you can. This should improve trace throughput.

2.4.4 Trace Hardware

Hardware details are shown in the following figures.

FIGURE 2-5: TRACE CONNECTOR ON EMULATION HEADER

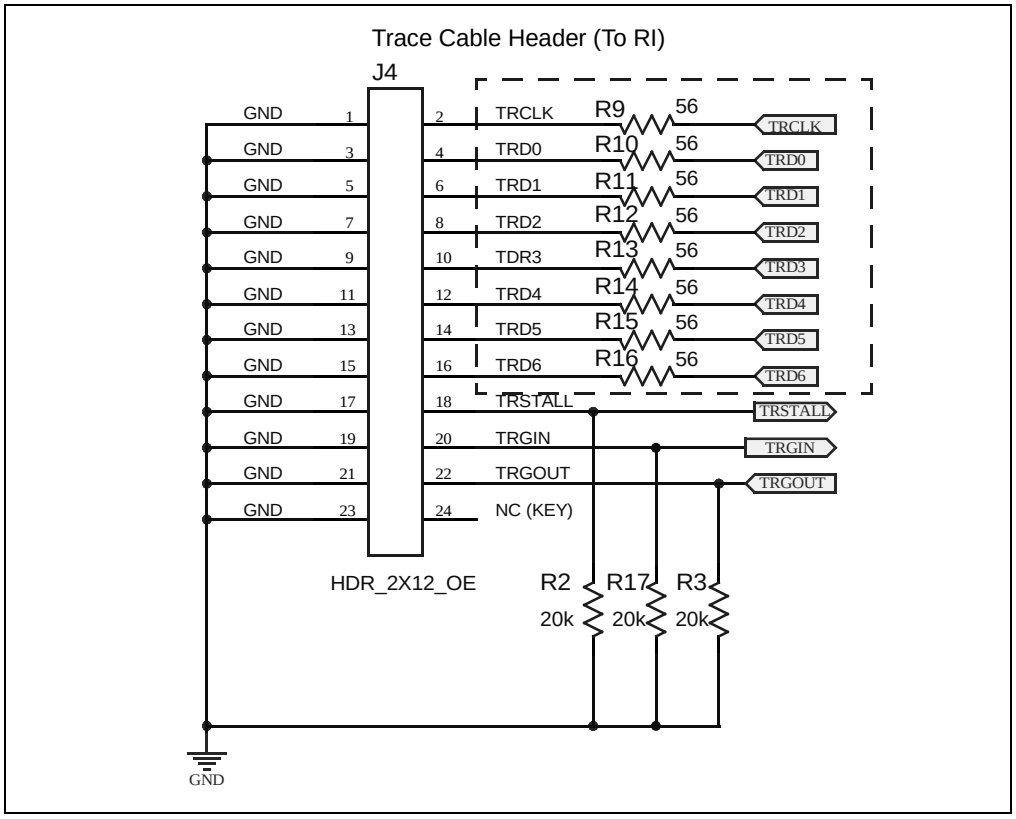
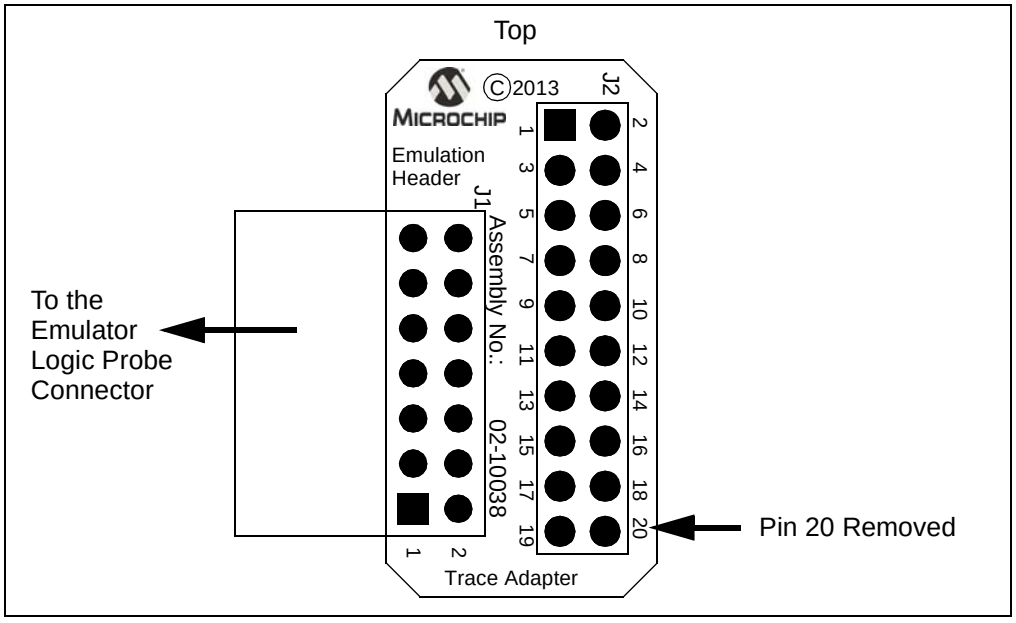


FIGURE 2-6: INTERFACE BOARD



2.5 HARDWARE ADDRESS/DATA BREAKPOINTS

Up to 32 hardware address/data breakpoints are available to use on the emulation header (for details see Section 2.2 “Breakpoint, Runtime Watch, and Trace Resources”.). Software breakpoints are useful, but real hardware breakpoints are incomparable when you need unfettered control of qualifying the breakpoint/event conditions beyond the simple address matching. Consider the addition of a special interrupt contextual qualifier and these hardware breakpoints offer a high degree of configurability.

Emulation header hardware breakpoints can trigger without halting execution and could be used as trigger events to other features.

2.5.1 Breakpoint Setup

Address/Data Breakpoints may be found and set up on the New Breakpoint dialog (*Debug>New Breakpoint*) by choosing either “Address” or “Data” as the “Breakpoint Type”. After the breakpoint is created, it can be edited by right clicking and selecting “Customize”. For details, see the MPLAB X IDE documentation.

2.5.2 Range Breakpoints

A breakpoint may be specified to occur with a range of program or data memory. A start address, and then an end address, are specified for the breakpoint. Then the break condition is selected.

The dialog items used to set up range breakpoints are described below.

TABLE 2-2: ADDRESS/DATA BREAKPOINTS - SETTINGS

Item	Description
Enable Range Address	check to set a range breakpoint
Address (Start)	enter a starting hexadecimal memory address
Address (End)	enter an ending hexadecimal memory address
Breaks on	specifies the conditions of the break Conditions differ for address memory and data memory.

2.5.3 Data Value Comparison

Data memory breakpoints may be set up to compare a “Break on” value to another value before breaking.

The dialog items used to set up data value comparison are described below.

TABLE 2-3: DATA BREAKPOINTS - SETTINGS

Item	Description
Breaks on	specifies the conditions of the break
Value	for the “Breaks on” selection of: • Read Specific Value • Write Specific Value • Read or Write Specific Value Enter a hexadecimal value here.
Value Comparison	compare to “Value” as specified: **= Value**: Equal-to value **!= Value**: Not-equal-to value **> Value**: Greater-than value **< Value**: Less-than value

2.5.4 Data Value Mask

Data memory breakpoints may be set up to compare a masked "Break on" value to another value before breaking.

To set up a value comparison, see Section 2.5.3 "Data Value Comparison".

The dialog items used to set up a data value mask are described below.

TABLE 2-4: DATA BREAKPOINTS - SETTINGS

Item	Description
Data Value Mask	use mask when comparing to "Value" Enter a value in the range 0x00 to 0xhh, where: 0x00: no bits compared 0xhh: all bits compared

2.5.5 Pass Count Operation

Using a pass count allows you to delay breaking until after a specified count.

Event must occur Count times

Count is the number of times you will pass the breakpoint before stopping.

- 0 means execution will stop immediately
- 1 means execution will pass once then stop the next time (stop on second time)
- 10 means execution will pass 10 times and stop the next time (stop on eleventh time)

The dialog items used to set up a pass count operation are found in the "Pass count" section of the dialog and are described below.

TABLE 2-5: ADDRESS/DATA BREAKPOINTS - PASS COUNT

Item	Description
Condition	determines when the break specified under "Breaks on" occurs: **Always Break**: Always break when the "Breaks on" condition is met. **Event must occur Count times**: An event ("Breaks on" condition) must occur Count times before actually breaking.
Count*	according to the condition specified, enters the number of events

2.5.6 Trigger Out Operation

A trigger out pulse can be generated when an address or data breakpoint is reached. For more on triggers, see Section 2.9 "Trigger In/Out".

The dialog item used to set up trigger out operation is described below

TABLE 2-6: ADDRESS/DATA BREAKPOINTS - TRIGGER OUT OPTIONS

Item	Description
Trigger Options	selects when to trigger, either: • do not emit a trigger out pulse when breakpoint is hit • emits a trigger out pulse when breakpoint is hit

2.5.7 Interrupt Context Detection

An address or data breakpoint can be set based on the context of an interrupt. You can set up the breakpoint so it only breaks when it is in the interrupt section of code (ISR), only when it is in main line code, or when it is in either ISR or main code. This can assist when attempting to narrow down issues in code regions.

Address/Data Breakpoints may be found and set up on the New Breakpoint Dialog (*Debug>New Breakpoint*) by choosing either “Address” or “Data” as the “Breakpoint Type”. After the breakpoint is created, it may be edited by right clicking and selecting “Customize”.

The dialog item used to set up interrupt context is described below.

TABLE 2-7: ADDRESS/DATA BREAKPOINTS - INTERRUPT CONTEXT

Item	Description
Interrupt Context	Interrupt Context qualifier for address/data breakpoints select from: • Always break (break in both ISR and main code) • Break in main line (non-interrupt) context only - break in main code only • Break in interrupt context only - break in ISR code only

2.6 ENHANCED EVENT BREAKPOINTS

For a definition of event breakpoints, see the MPLAB X IDE Help, "New Breakpoint Dialog". Additional events for emulation headers are:

- Break on MCLR reset
- Break on execution out of bounds
- Break on trigger in signal

When creating a new breakpoint or customizing an existing breakpoint using an emulation header, additional actions are available for each event breakpoint:

Action	Description
Break	breaks (halts) execution per option specified
Trigger out	emits a trigger out pulse per option specified
Break and trigger out	breaks (halts) execution AND emits a trigger out pulse per option specified

Event breakpoints may be found and set up on the New Breakpoint Dialog (*Debug>New Breakpoint*) by choosing "Event" as the "Breakpoint Type". After the breakpoint is created, it may be edited by right clicking and selecting "Customize".

2.6.1 Execution Out-of-Bounds Detection

An emulation header can be used to detect out-of-bounds execution of code. Out-of-bounds code execution is detected by an event breakpoint that watches for Program Counter (PC) values that exceed the available program memory of the emulated MCU. The out-of-bounds code execution condition is typically caused by a computed GOTO or CALL that erroneously computes the index, or by loading PCLATH with an incorrect value. Once code is halted due to the execution out-of-bounds event breakpoint the 'Previous PC' functionality can be used to identify the offending instruction.

The Out-of-bounds break option may be found and set up on the New Breakpoint Dialog (*Debug>New Breakpoint*) by choosing "Event" as the "Breakpoint Type" and checking "Break on execution out of bounds". After the breakpoint is created, it may be edited by right clicking and selecting "Customize".

2.6.2 Break on Trigger In/Emit Trigger Out

An emulation header can be set up to break on a trigger in. For details on this type of trigger, see Section 2.9.2 "Trigger In Operation".

Also, the emulation header may be set to trigger out, or break and trigger out, when an enhanced even breakpoint is hit. For details on this type of trigger, see Section 2.9.3 "Trigger Out Operation".

See also: Section 2.9.1 "Trigger In/Out Hardware".

2.7 EVENT COMBINERS

An event combiner monitors multiple event inputs (currently breakpoints only) and can generate a halt or a trigger out that is based on combinations and sequences of those inputs.

Emulation headers have four (4) Event Combiners, and each combines eight (8) combinational or sequential events into a single event trigger. Individual breakpoints may be grouped into sequences, logical 'AND' lists, or a nested combination of these for more complex control. The Event Combiners were modeled after the legacy MPLAB® ICE 2000 complex triggers.

Set up the following complex breakpoints through selections in the Breakpoint window (*Window>Debugging>Breakpoints*).

2.7.1 Complex Breakpoint Sequence

A breakpoint sequence is a list of breakpoints that execute but do not halt until the last breakpoint is executed. Sequenced breakpoints can be useful when there are more than one execution path leading to a certain instruction, and you only want to exercise one specific path.

To create a Breakpoint Sequence:

1. Right click on an existing breakpoint or shift click to select a group of existing breakpoints and right click on the group.
2. From the pop-up menu, go to "Complex Breakpoint" and select "Add to New Sequence".
3. Enter a name for your sequence in the dialog box, and click **OK**.
4. The breakpoint(s) will appear under the new sequence.

To add Existing Breakpoints to a Sequence:

1. Right click on an existing breakpoint or shift click to select a group of existing breakpoints and right click on the group.
2. From the pop-up menu, go to "Complex Breakpoint" and select "Move to *Name*", where *Name* is the name of the sequence.

To add a New Breakpoint to a Sequence:

Right click on the sequence and select "New Breakpoint".

For more on setting up a new breakpoint, see Section 2.5 "Hardware Address/Data Breakpoints" and Section 2.6 "Enhanced Event Breakpoints".

To select the Sequence Order:

1. Expand on a sequence to see all items.
2. Right click on an item and select *Complex Breakpoints>Move Up* or *Complex Breakpoints>Move Down*. Sequence execution of breakpoints is bottom-up; the last breakpoint in the sequence occurs first.

To remove Breakpoints from a Sequence:

1. Right click on an existing breakpoint or shift click to select a group of existing breakpoints and right click on the group.
2. From the pop-up menu, go to "Complex Breakpoint" and select "Remove from *Name*", where *Name* is the name of the sequence.

2.7.2 Complex Breakpoint Latched-And

In addition to breakpoint sequences, a Latched-And (hardware AND) is available to AND a list of breakpoints. ANDed breakpoints can be useful when a variable is modified in more than one location and you need to break only when that variable is modified in one particular location.

To create a Breakpoint Latch-And:

1. Right click on an existing breakpoint or shift click to select a group of existing breakpoints and right click on the group.
2. From the pop-up menu, go to "Complex Breakpoint" and select "Add a New Latched-And".
3. Enter a name for your Latched-And in the dialog box and click **OK**.
4. The breakpoint(s) will appear under the new Latched-And.

To add Existing Breakpoints to a Latch-And:

1. Right click on an existing breakpoint or shift click to select a group of existing breakpoints and right click on the group.
2. From the pop-up menu, go to "Complex Breakpoint" and select "Move to *Name*", where *Name* is the name of the Latched-And.

To add a New Breakpoint to a Latch-And:

Right click on the Latched-And and select "New Breakpoint".

To remove Breakpoints from a Latch-And:

1. Right click on an existing breakpoint or shift click to select a group of existing breakpoints and right click on the group.
2. From the pop-up menu, go to "Complex Breakpoint" and select "Remove from *Name*", where *Name* is the name of the Latched-And.

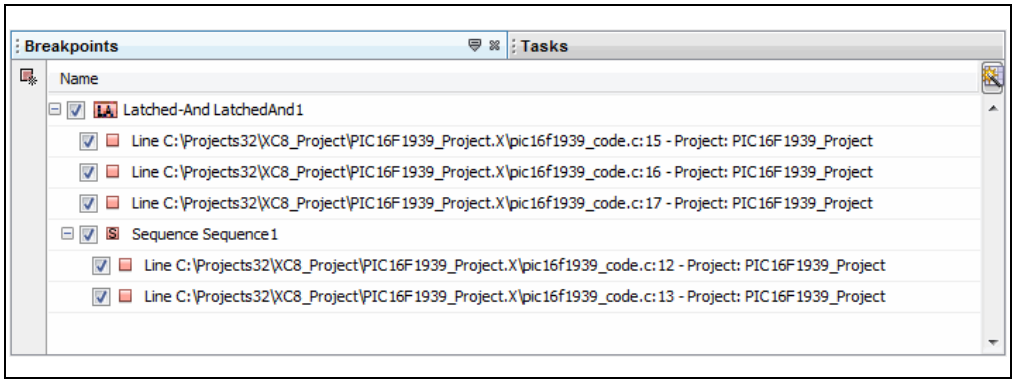
2.7.3 Complex Breakpoint Nesting

Complex breakpoints may be nested to create even more complex breaking schemes.

To nest one group of complex breakpoints into another:

1. Create two groups of complex breakpoints (Sequenced, Latched-And or one of each).
2. Right click on the complex breakpoint group you wish to nest.
3. From the pop-up menu, go to "Complex Breakpoint" and select "Move to *Name*", where *Name* is the name of the other complex breakpoint group.
4. The first group will appear under the second group, thus creating a scheme.

FIGURE 2-7: NESTED COMPLEX BREAKPOINTS



2.8 STOPWATCH CYCLE COUNTER

The Stopwatch Cycle Counter (32-bit-wide instruction cycle counter) has the ability to perform the existing basic instruction cycle counting (all instruction cycles counted) that exists on standard Enhanced Midrange parts..

> **Note:** The count units are in instruction cycles, not in instructions (as not all instructions execute in a single cycle).

The stopwatch is available under *Window>Debugging>Stopwatch*.

The stopwatch uses two (2) breakpoint resources.

2.9 TRIGGER IN/OUT

The emulation header is capable of producing an output pulse for external triggering and detecting an input pulse for internal triggering.

- Trigger In/Out Hardware
- Trigger In Operation
- Trigger Out Operation

2.9.1 Trigger In/Out Hardware

Pins on the emulation header may be used for Trigger In/Out. Pin functions are labeled on the board silkscreen, namely:

- GND – ground
- TRIG IN – used for Trigger In
- TRIG OUT – used for Trigger Out

2.9.2 Trigger In Operation

A pulse on the Trigger In (TRIG IN) pin can be used to generate a trigger condition, halt and/or trigger out signal. Set up Trigger In by selecting *Window>Debugging>Triggers*.

Selection	Description
Polarity	Trigger-in pin pulse polarity: **Positive:** a positive-going pulse **Negative:** a negative-going pulse
Noise Reduction Filter	reduces the noise on the trigger-in pin using a filter, which helps mitigate spurious triggers from halting code Enables or disables this feature
Trigger Trace	Trigger trace when a pulse is received on the trigger-in pin Enables or disables this feature

An Event Breakpoint can be set up to break when a Trigger In pulse is detected. See Section 2.6 “Enhanced Event Breakpoints”.

2.9.3 Trigger Out Operation

A pulse can be output on the Trigger Out (TRIG OUT) pin based on the setting of breakpoint types:

- Line
- Data or Address (see Section 2.5 “Hardware Address/Data Breakpoints”)
- Event (see Section 2.6 “Enhanced Event Breakpoints”).

The breakpoint can be set up so that the pulse is emitted without halting device execution.

Set up Trigger Out by selecting *Window>Debugging>Triggers*.

Selection	Description
Polarity	Trigger-out pin pulse polarity: **Positive:** a positive-going pulse **Negative:** a negative-going pulse
Slew Rate Limiting	Limit the slew rate on the Trigger Out pulse. **Enable:** slew rate is slow and limited **Disable:** Slew rate is as fast as possible
One Shot	The trigger out pulse duration is: **Enable:** a fixed width, regardless of MCU operating frequency **Disable:** the length of the event itself, which is MCU frequency dependent
Force Trigger Out	Click this button to force a Trigger Out pulse.

A handy feature of the trigger out signal is that its duration can last as long as the occurring event. For example, if the customer needs to time the duration of the watchdog timer (whose timeout period is user-programmable); the following code, in conjunction with the trigger out and SLEEP event breakpoint features, can make timing an event very simple without employing the old-school technique of writing a single line of (special, non-production) code to wiggle an I/O pin.

The following code is an example:

```
    ; ------------------------------------------------------
    ;   T R I G G E R   O U T   T E S T :
    ;   T I M I N G   T H E   W A T C H D O G   T I M E R
    ; ------------------------------------------------------

    ; 1) Ensure the watchdog timer configuration bit is enabled.
    ; 2) Set an Event Breakpoint (Break on SLEEP) to initiate a
    :    'Trigger out' action only.
    ; 3) Connect your oscilloscope probe to the TRIGGER OUT pin and
    ;    set up your oscilloscope to trigger on the rising edge.
    ; 4) Run the following code:

    CLRWDT
    NOP
    NOP
    SLEEP

    ; The expiration of the watchdog timer will wake up the MCU from
    : sleep after ~2 seconds. The trigger out pulse high-time
    ; duration is the duration of the watchdog timer: ___---___
    ; Since the default watchdog timer period value is 2 seconds
    : typical, the trigger out pulse measured on the oscilloscope
    ; should be approximately 2 seconds.
    NOP
    NOP
    NOP
Loop_101:
    BRA         Loop_101
    ; ------------------------------------------------------
```

2.10 VIEW HARDWARE STACK ON HALT

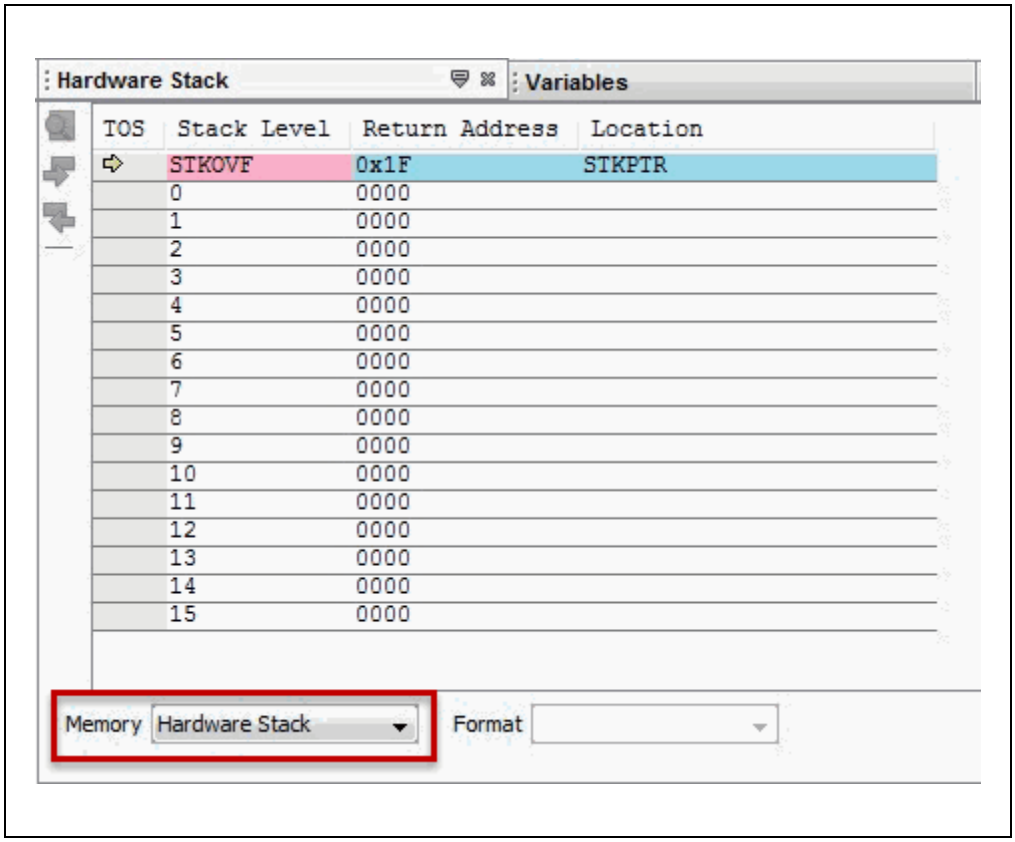
View the contents of the hardware stack on halt.

To open the hardware stack window, do the following:

1. Select *Window>PIC Memory Views* and choose any memory window type (e.g., Program Memory).
2. In the memory window, under the “Memory” drop-down list, select “Hardware Stack”.

The first stack level is denoted by “0”.

FIGURE 2-8: HARDWARE STACK VIEW



2.11 PREVIOUS PROGRAM COUNTER

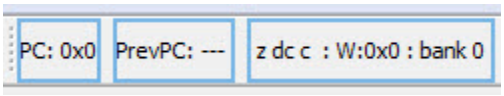
MPLAB X IDE has a Previous Program Counter (PrevPC) display, whose value can be used as discussed below.

The caller of a subroutine or branch can be identified from the PrevPC value if a breakpoint is set on the first instruction of the destination routine (or subroutine). When the part is halted in the debug session, the PrevPC display in MPLAB X IDE will indicate the address of the calling or branch instruction.

Similarly, if a breakpoint is set on the instruction that follows a BTFSC-pair (not the skipped instruction, but the next one), PrevPC will identify whether or not the skip occurred.

Additionally if an 'execution out-of-bounds' halt exception occurs, the PrevPC feature can be used to identify the wild-branch instruction that caused the 'execution out-of-bounds' halt exception.

FIGURE 2-9: PREVIOUS PC VIEW



2.12 BACKGROUND DEBUG

The emulation header's on-board -ME2 device contains a Background Debug control interface that allows you read/write access to RAM memory, SFRs, and emulation registers while your program is running or even sleeping.

Note:	Write access to RAM memory and SFRs while your program is running/sleeping is currently not supported in MPLAB X IDE but is planned in a future release.

Background Debug capability includes the following advantages:

- Allows runtime changes of breakpoints (i.e., runtime address/data/complex/event breakpoints).
- Compared to debug headers (with -ICE or -ICD devices), yields noticeably faster single-stepping speeds at lower MCU operating frequencies.

Chapter 3. Emulation Header List

3.1 INTRODUCTION

Currently available emulation headers and their associated -ME2 devices are shown below, organized by supported device.

TABLE 1: OPTIONAL EMULATION HEADERS - PIC12/16 DEVICES

Device Supported by Emulation Header	Pin Count	EEP* Part Number	-ME2 Device on Emulation Header	VDD Max
PIC12F1612	8	AC244066	PIC16F1619-ME2	5.5V
PIC16F1613	14			
PIC16F1614	14			
PIC16F1615	14			
PIC16F1618	20			
PIC16F1619	20			
PIC12LF1612	8			3.6V
PIC16LF1613	14			
PIC16LF1614	14			
PIC16LF1615	14			
PIC16LF1618	20			
PIC16LF1619	20			
PIC16F1703	14	AC244065	PIC16F1719-ME2	5.5V
PIC16F1704	14			
PIC16F1705	14			
PIC16F1707	20			
PIC16F1708	20			
PIC16F1709	20			
PIC16F1713	28			
PIC16F1716	28			
PIC16F1717	40/44			
PIC16F1718	28			
PIC16F1719	40/44			
PIC16LF1703	14			3.6V
PIC16LF1704	14			
PIC16LF1705	14			
PIC16LF1707	20			
PIC16LF1708	20			
PIC16LF1709	20			
PIC16LF1713	28			
PIC16LF1716	28			
PIC16LF1717	40/44			
PIC16LF1718	28			
PIC16LF1719	40/44			

* See Section 1.1 "What is an Emulation Extension Pak?".

TABLE 1: OPTIONAL EMULATION HEADERS - PIC12/16 DEVICES (CON'T)

Device Supported by Emulation Header	Pin Count	EEP* Part Number	-ME2 Device on Emulation Header	VDD Max
PIC16F1782 PIC16F1783 PIC16F1784 PIC16F1786 PIC16F1787 PIC16F1788 PIC16F1789	28 28 40/44 28 40/44 28 40/44	AC244064	PIC16F1789-ME2	5.5V
PIC16LF1782 PIC16LF1783 PIC16LF1784 PIC16LF1786 PIC16LF1787 PIC16LF1788 PIC16LF1789	28 28 40/44 28 40/44 28 40/44			3.6V
PIC12F1822 PIC12F1840 PIC16F1823 PIC16F1824 PIC16F1825 PIC16F1826 PIC16F1827 PIC16F1828 PIC16F1829 PIC16F1847	8 8 14 14 14 18 18 20 20 18	AC244063	PIC16F1829-ME2	5.5V
PIC12LF1822 PIC12LF1840 PIC16LF1823 PIC16LF1824 PIC16LF1825 PIC16LF1826 PIC16LF1827 PIC16LF1828 PIC16LF1829 PIC16LF1847	8 8 14 14 14 18 18 20 20 18			3.6V
PIC16F1933 PIC16F1934 PIC16F1936 PIC16F1937 PIC16F1938 PIC16F1939	28 40/44 28 40/44 28 40/44	AC244055	PIC16F1939-ME2	5.5V
PIC16LF1933 PIC16LF1934 PIC16LF1936 PIC16LF1937 PIC16LF1938 PIC16LF1939	28 40/44 28 40/44 28 40/44			3.6V

* See Section 1.1 "What is an Emulation Extension Pak?".

3.2 AC244055

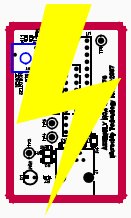
3.2.1 Header Identification

The AC number is used for ordering the Emulation Extension Pak, which contains the emulation header. However, this number is not on the header, as the board may be used for multiple headers by inserting different -ME2 devices. To identify this header, use the following information.

AC Number	-ME2 Device	Board Assembly Number
AC244055	PIC16F1939-ME2	02-10039

3.2.2 Header Setup and Operation

For this header, select your device type using J5.



CAUTION

Header Damage.
Selecting the wrong type of device may cause device damage, especially when V_{DD} is greater than 3.6 V (with respect to V_{SS}).

You can also enable the Power LED by using J3. J6 is not a jumper but a group of pins that you can use for trigger in and/or trigger out signals.

Jumper	Name	Setting	Function	Notes
J3	LED Enable	Open	disable power LED indicator	
		Short	enable power LED indicator	
J5	Emulation Select	pins 1-2	emulate 'F' variants (5.5 V MAX)	Note 1
		pins 2-3	emulate 'LF' variants (3.6 V MAX!)	
J6	Trigger I/O	pin 1	Ground	Note 2
		pin 2	used for Trigger In	
			used for Trigger Out	

Note 1: Do not change this setting when the header is powered. Power down first.

Note 2: See Section 2.9 “Trigger In/Out”.

3.2.3 Header Limitations and Errata

See the “Limitations” section in your debug tool online Help file for details.

For silicon errata, see:

Section A.3 “**Hardware Breakpoint Issue**”

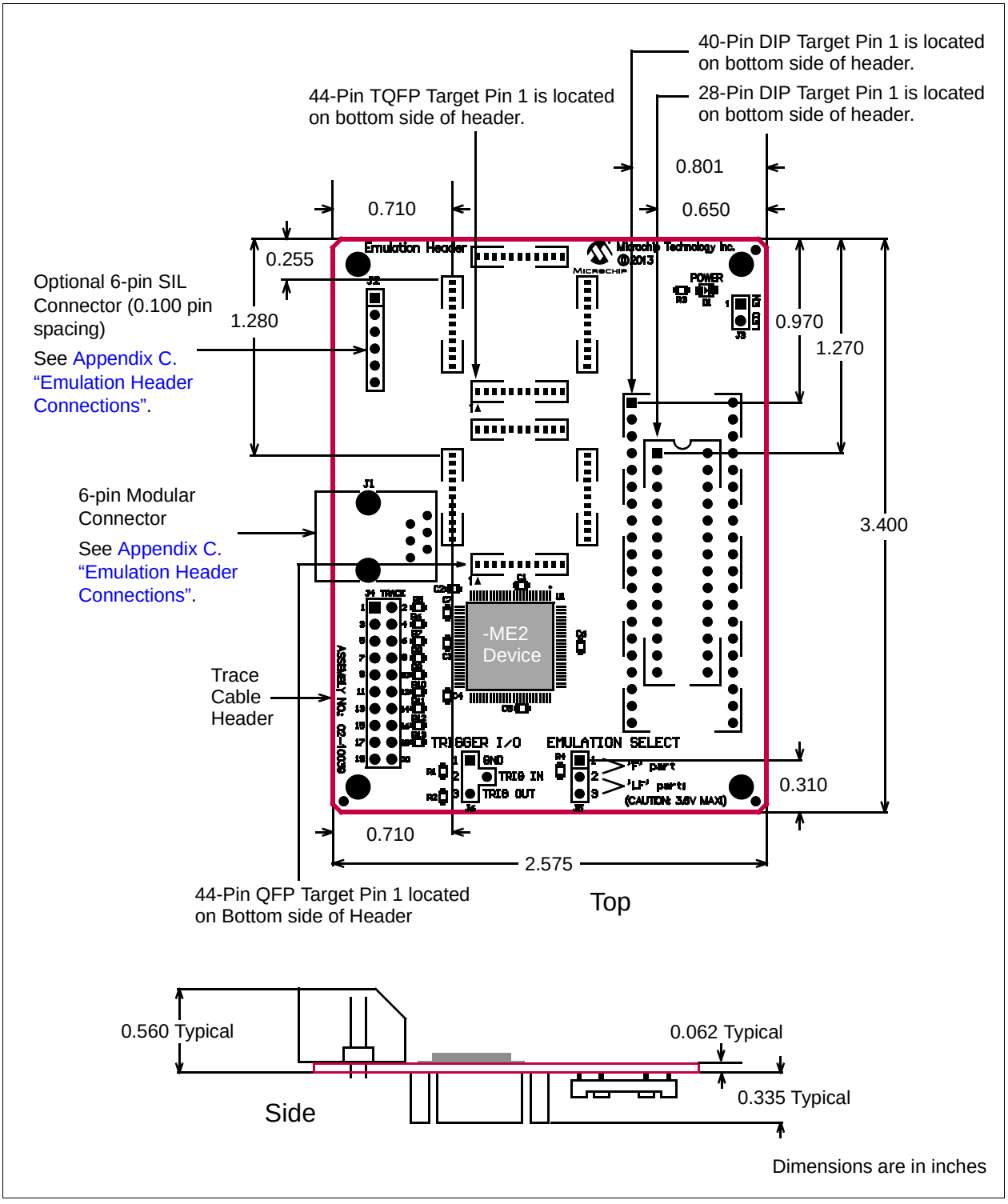
Section A.4 “Trigger In/Halt during Multi-Cycle Instruction Processing”

3.2.4 Header Dimensions

The figure below lists the dimensions for the emulation header. Dimensions are design values in inches.

If the length and/or width of the emulation header is too large a footprint for the target board, consider using stand-offs, single in-line pins, transition sockets or other extenders in the header connection socket to raise the header above the target.

FIGURE 3-1: DIMENSIONS - AC244055



3.3 AC244063

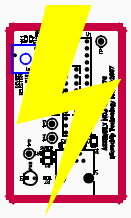
3.3.1 Header Identification

The AC number is used for ordering the Emulation Extension Pak, which contains the emulation header. However, this number is not on the header, as the board may be used for multiple headers by inserting different -ME2 devices. To identify this header, use the following information.

AC Number	-ME2 Device	Board Assembly Number
AC244063	PIC16F1829-ME2	02-10212

3.3.2 Header Setup and Operation

For this header, select your device type using J5.



CAUTION

Header Damage.
Selecting the wrong type of device may cause device damage, especially when Vdd is greater than 3.6 V (with respect to Vss).

You can also enable the Power LED by using J3.

J7, J8, and J9 are a group of pins that you can use for trigger in/trigger out signals.

Jumper	Name	Setting	Function	Notes
J3	LED Enable	Open	disable power LED indicator	
		Short	enable power LED indicator	
J5	Emulation Select	pins 1-2	emulate 'F' variants (5.5 V MAX)	Note 1
		pins 2-3	emulate 'LF' variants (3.6 V MAX!)	
J6	Device Package	pins 1-2	18-pin devices	Note 1
		pins 2-3	8/14/20-pin devices	
J7	Trigger I/O	pin 1	Ground	Note 2
J8		pin 2	used for Trigger In	
J9		pin 3	used for Trigger Out	

Note 1: Do not change this setting when the header is powered. Power down first.

Note 2: See Section 2.9 “Trigger In/Out”.

3.3.3 Header Limitations

See the “Limitations” section in your debug tool online Help file for details.

For silicon errata, see:

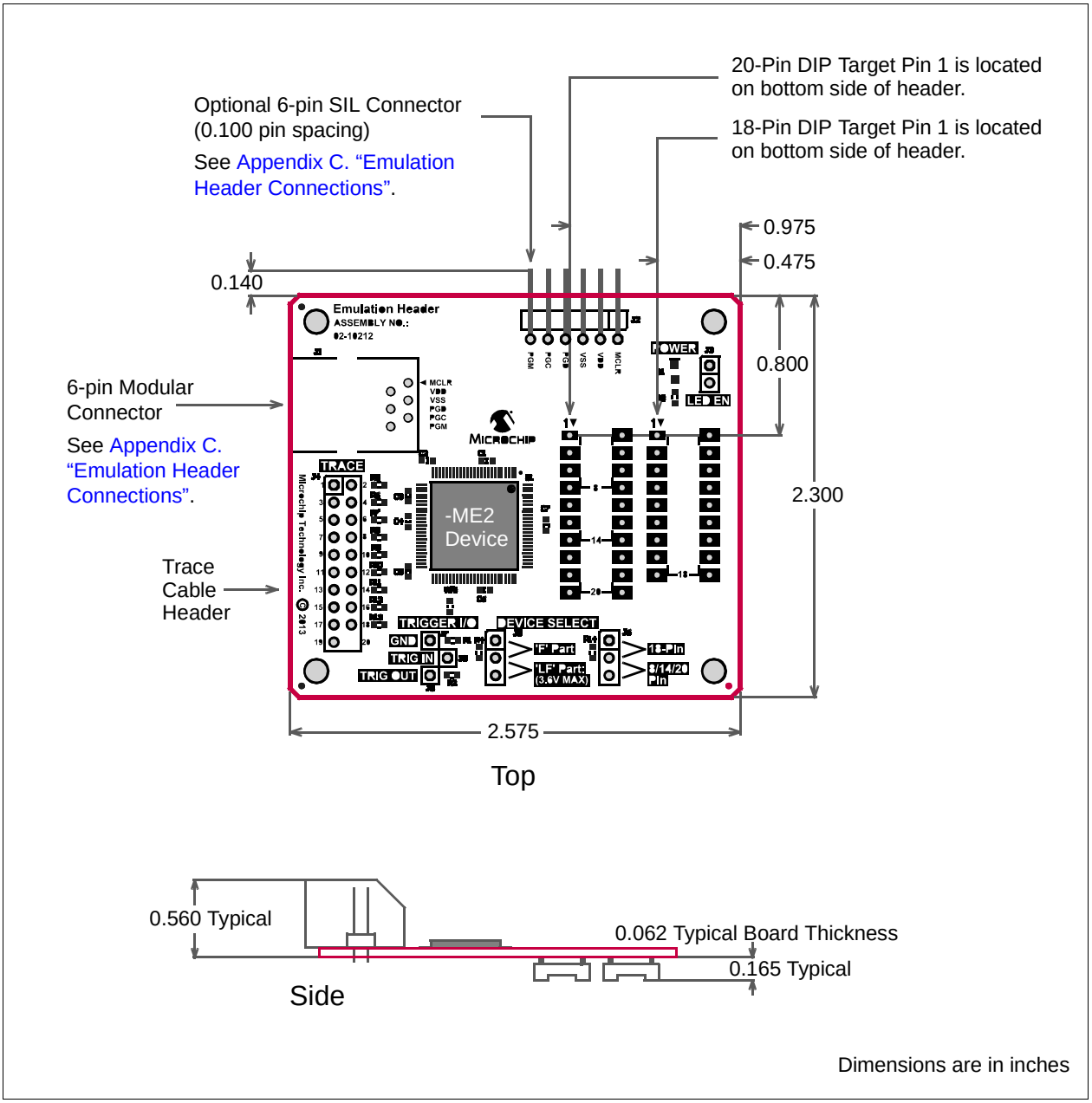
Section A.4 “Trigger In/Halt during Multi-Cycle Instruction Processing”

3.3.4 Header Dimensions

The figure below lists the dimensions for the emulation header. Dimensions are design values in inches.

If the length and/or width of the emulation header is too large a footprint for the target board, consider using stand-offs, single in-line pins, transition sockets or other extenders in the header connection socket to raise the header above the target.

FIGURE 3-2: DIMENSIONS - AC244063



3.4 AC244064

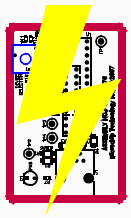
3.4.1 Header Identification

The AC number is used for ordering the Emulation Extension Pak, which contains the emulation header. However, this number is not on the header, as the board may be used for multiple headers by inserting different -ME2 devices. To identify this header, use the following information.

AC Number	-ME2 Device	Board Assembly Number
AC244064	PIC16F1789-ME2	02-10039

3.4.2 Header Setup and Operation

For this header, select your device type using J5.



CAUTION

Header Damage.
Selecting the wrong type of device may cause device damage, especially when V_{DD} is greater than 3.6 V (with respect to V_{SS}).

You can also enable the Power LED by using J3.

J6 is not a jumper but a group of pins that you can use for trigger in/trigger out signals.

Jumper	Name	Setting	Function	Notes
J3	LED Enable	Open	disable power LED indicator	
		Short	enable power LED indicator	
J5	Emulation Select	pins 1-2	emulate 'F' variants (5.5 V MAX)	Note 1
		pins 2-3	emulate 'LF' variants (3.6 V MAX!)	
J6	Trigger I/O	pin 1	Ground	Note 2
		pin 2	used for Trigger In	
		pin 3	used for Trigger Out	

Note 1: Do not change this setting when the header is powered. Power down first.

Note 2: See Section 2.9 “Trigger In/Out”.

3.4.3 Header Limitations and Errata

See the “Limitations” section in your debug tool online Help file for details.

For silicon errata, see:

Section A.2 “**CCP3 Capture**”.

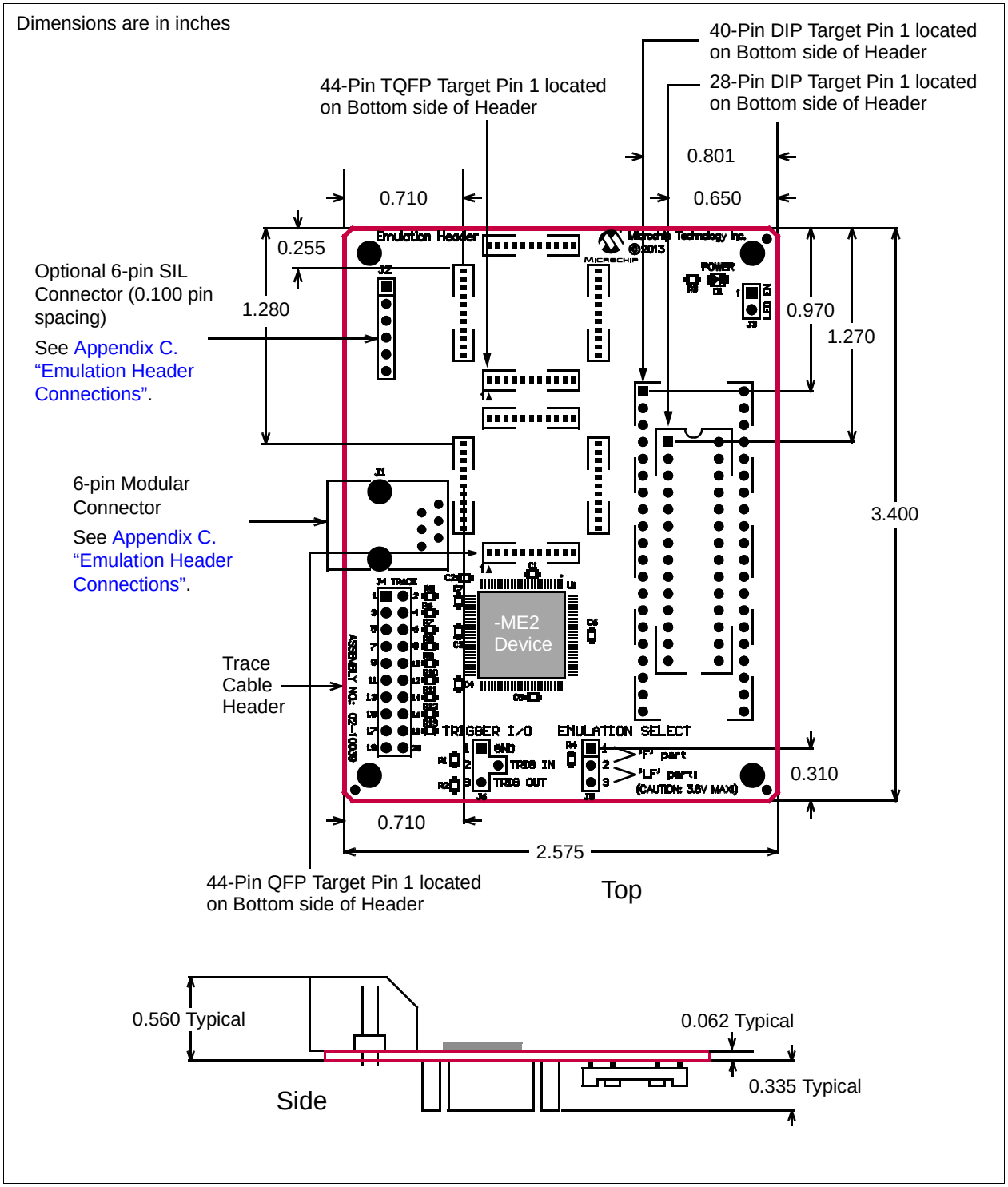
Section A.4 “Trigger In/Halt during Multi-Cycle Instruction Processing”

3.4.4 Header Dimensions

The figure below lists the dimensions for the emulation header. Dimensions are design values in inches.

If the length and/or width of the emulation header is too large a footprint for the target board, consider using stand-offs, single in-line pins, transition sockets or other extenders in the header connection socket to raise the header above the target.

FIGURE 3-3: DIMENSIONS - AC244064



3.5 AC244065

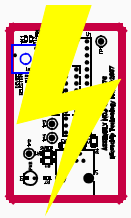
3.5.1 Header Identification

The AC number is used for ordering the Emulation Extension Pak, which contains the emulation header. However, this number is not on the header, as the board may be used for multiple headers by inserting different -ME2 devices. To identify this header, use the following information.

AC Number	-ME2 Device	Board Assembly Number
AC244065	PIC16F1719-ME2	02-10315

3.5.2 Header Setup and Operation

For this header, select your device type using JP2.



CAUTION

Header Damage.
Selecting the wrong type of device may cause device damage, especially when V_{DD} is greater than 3.6 V (with respect to V_{SS}).

You can also enable the Power LED by using JP1. TP1-3 are not jumpers but a group of test points that you can use for trigger in/trigger out signals.

Jumper	Name	Setting	Function	Notes
JP1	LED Enable	Open	disable power LED indicator	
		Short	enable power LED indicator	
JP2	Emulation Select	pins 1-2	emulate 'F' variants (5.5 V MAX)	Note 1
		pins 2-3	emulate 'LF' variants (3.6 V MAX!)	
JP3	Device Package	pins 1-2	28/40/44-pin devices	Note 1
		pins 2-3	14/20-pin devices	
TP1	Trigger I/O		Ground	Note 2
TP2			used for Trigger In	
TP3			used for Trigger Out	

Note 1: Do not change this setting when the header is powered. Power down first.

Note 2: See Section 2.9 “Trigger In/Out”.

3.5.3 Header Limitations and Errata

See the “Limitations” section in your debug tool online Help file for details.

For silicon errata, see:

Section A.2 “**CCP3 Capture**”

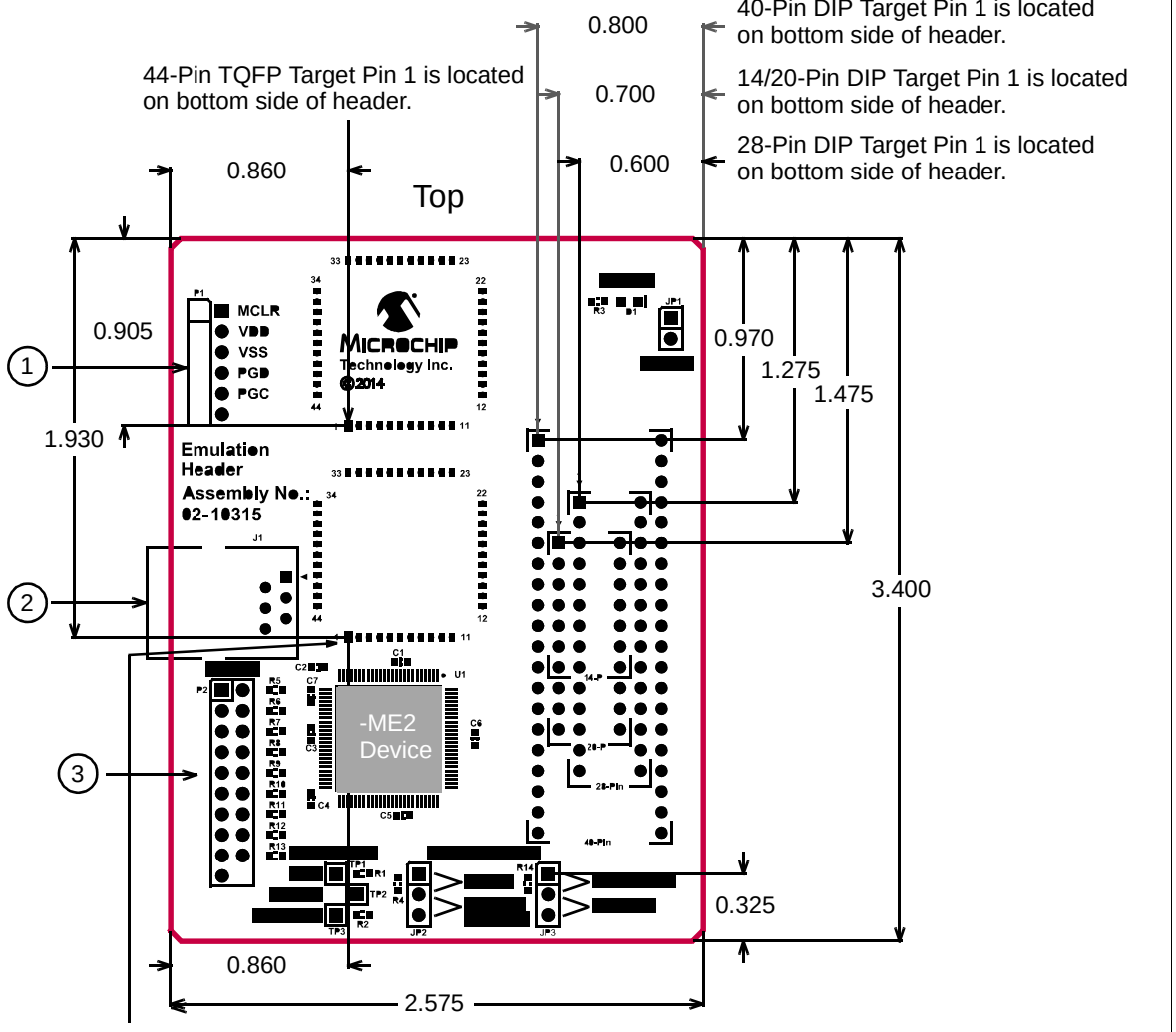
Section A.4 “Trigger In/Halt during Multi-Cycle Instruction Processing”

3.5.4 Header Dimensions

The figure below lists the dimensions for the emulation header. Dimensions are design values in inches.

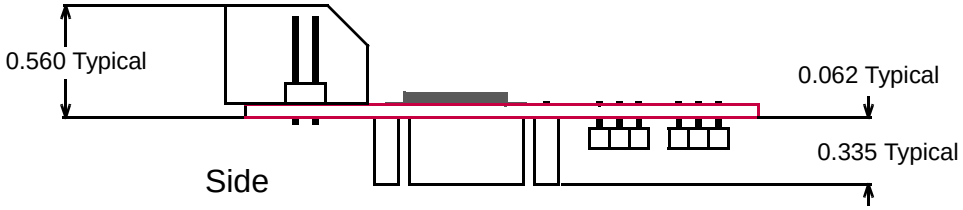
If the length and/or width of the emulation header is too large a footprint for the target board, consider using stand-offs, single in-line pins, transition sockets or other extenders in the header connection socket to raise the header above the target.

FIGURE 3-4: DIMENSIONS - AC244065



44-Pin QFP Target Pin 1 is located on bottom side of header.

1. Optional 6-pin SIL Connector (0.100 pin spacing). See Appendix C. "Emulation Header Connections".
2. 6-pin Modular Connector. See Appendix C. "Emulation Header Connections".
3. Trace Cable Header



Dimensions are in inches

3.6 AC244066

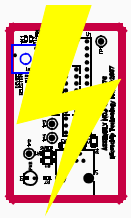
3.6.1 Header Identification

The AC number is used for ordering the Emulation Extension Pak, which contains the emulation header. However, this number is not on the header, as the board may be used for multiple headers by inserting different -ME2 devices. To identify this header, use the following information.

AC Number	-ME2 Device	Board Assembly Number
AC244066	PIC16F1619-ME2	02-10384

3.6.2 Header Setup and Operation

For this header, select your device type using JP2.



CAUTION

Header Damage.
Selecting the wrong type of device may cause device damage, especially when VDD is greater than 3.6 V (with respect to VSS).

You can also enable the Power LED by using JP1.

J7-J9 are not jumpers but a group of test points that you can use for trigger in/trigger out signals.

Jumper	Name	Setting	Function	Notes
JP1	LED Enable	Open	disable power LED indicator	
		Short	enable power LED indicator	
JP2	Emulation Select	pins 1-2	emulate 'F' variants (5.5 V MAX)	Note 1
		pins 2-3	emulate 'LF' variants (3.6 V MAX!)	
J7	Trigger I/O		Ground	Note 2
J8			used for Trigger In	
J9			used for Trigger Out	

Note 1: Do not change this setting when the header is powered. Power down first.

Note 2: See Section 2.9 “Trigger In/Out”.

3.6.3 Header Limitations and Errata

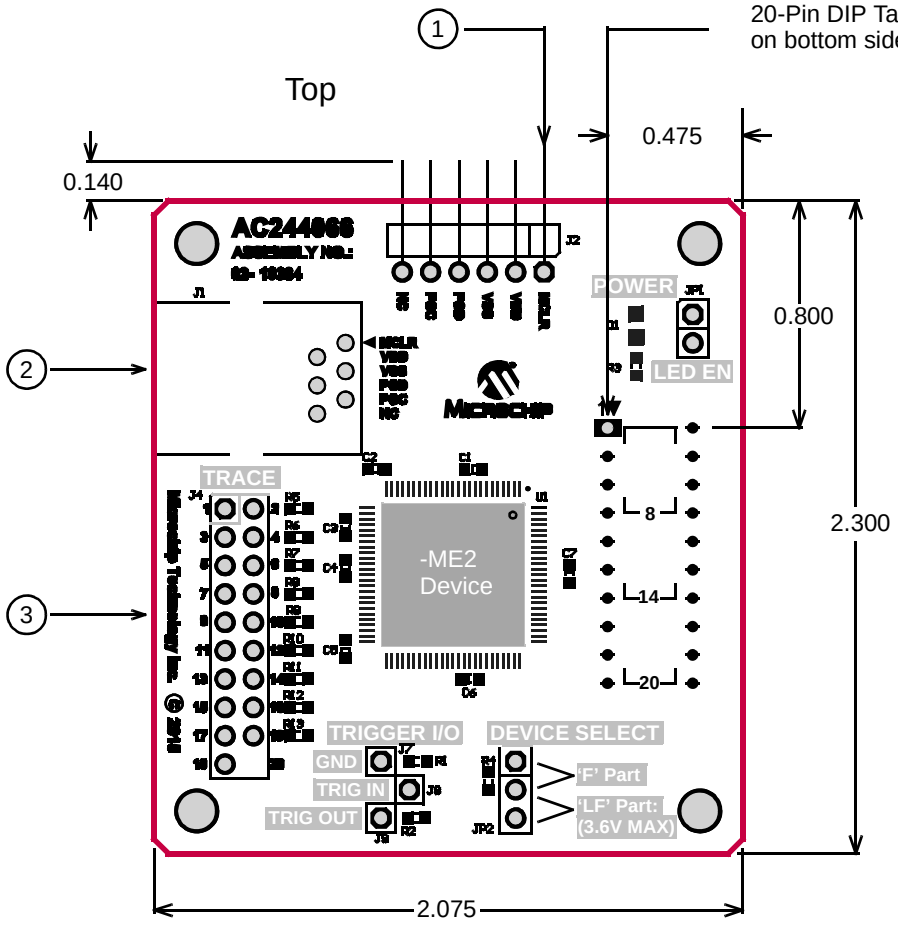
See the “Limitations” section in your debug tool online Help file for details.

3.6.4 Header Dimensions

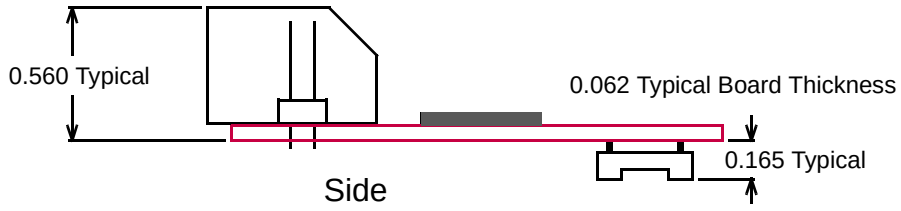
The figure below lists the dimensions for the emulation header. Dimensions are design values in inches.

If the length and/or width of the emulation header is too large a footprint for the target board, consider using stand-offs, single in-line pins, transition sockets or other extenders in the header connection socket to raise the header above the target.

FIGURE 3-5: DIMENSIONS - AC244066



1. Optional 6-pin SIL Connector (0.100 pin spacing). See Appendix C. "Emulation Header Connections".
2. 6-pin Modular Connector. See Appendix C. "Emulation Header Connections".
3. Trace Cable Header



Dimensions are in inches

Appendix A. -ME2 Silicon Errata

A.1 INTRODUCTION

All emulation header boards have a special -ME2 device mounted on them. In some cases the -ME2 silicon has errata, i.e., the device does not operate as expected due to silicon issues.

Known errata are:

- CCP3 Capture
- Hardware Breakpoint Issue
- Trigger In/Halt during Multi-Cycle Instruction Processing

A.2 CCP3 CAPTURE

When the input threshold control for RE0 is configured for TTL, the CCP3 capture input is ignored. This applies to silicon revision C0 of the devices listed below.

PIC16F1789-ME2 (AC244064):

- PIC16(L)F1784
- PIC16(L)F1787

Work-around: use ST Threshold.

A.3 HARDWARE BREAKPOINT ISSUE

When using this emulation header with C language or Assembly language code, hardware breakpoints will not function past memory locations 0x3FF for the emulated devices listed below.

PIC16F1939-ME2 (AC244055):

- PIC16(L)F1936
- PIC16(L)F1937

Workarounds:

1. Software breakpoints for these four parts could be used (with the caveat that software breakpoints do not offer the advanced capability of hardware breakpoints).
2. Any function(s) that need to be debugged with hardware breakpoints could be explicitly (and temporarily) located in the program memory area below 0x400 until the function is properly debugged.

   In C language for the MPLAB XC8 C compiler, this can be done with the `@ address` construct as follows, which will locate `function_1` at program memory base address `0x200`.

```
void function_1(void) @ 0x200
{
    asm("NOP");
    asm("NOP");
    asm("NOP");
}
```

In assembly language for the MPASM assembler, this can be done with the CODE directive as follows, which will locate function_1 at program memory base address 0x200.

```
HW_BRKPT CODE 0x0200

function_1:

    NOP
    NOP
    NOP

    RETURN
```

A.4 TRIGGER IN/HALT DURING MULTI-CYCLE INSTRUCTION PROCESSING

For some -ME2 devices, the arrival of a trigger-in/HALT event during processing of a multi-cycle instruction may cause the emulator to fail to properly execute/complete the instruction before coming to a HALT. This applies to the following -ME2 products:

PIC16(L)F1939-ME2 (AC244055)

PIC16(L)F1829-ME2 (AC244063)

PIC16(L)F1789-ME2 (AC244064)

PIC16(L)F1719-ME2 (AC244065)

Appendix B. Emulation Header Target Footprints

B.1 INTRODUCTION

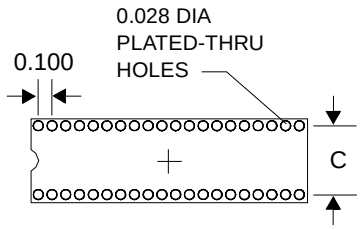
To connect an emulation header directly to a target board (without the use of a transition socket) the following information will be helpful.

- DIP Device Footprints
- TQFP/PLCC Device Footprints

B.2 DIP DEVICE FOOTPRINTS

The DIP device adapter footprint shown will accept adapter plugs like Samtec series APA plugs. These plugs can be soldered into place during development/emulation and eliminate the need for other sockets.

FIGURE B-1: DIP FOOTPRINT



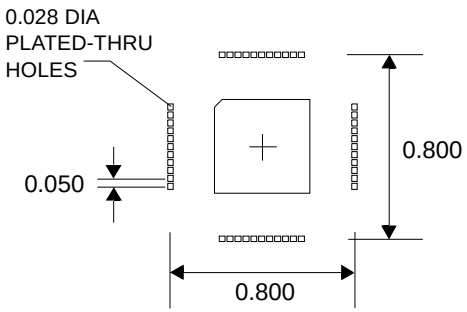
UNLESS OTHERWISE SPECIFIED,
DIMENSIONS ARE IN INCHES.
Drawing of DIP is 40-pin.

DIP	C
8-Pin	0.300
14-Pin	0.300
18-Pin	0.300
20-Pin	0.300
28-Pin	0.300
40-Pin	0.600

B.3 TQFP/PLCC DEVICE FOOTPRINTS

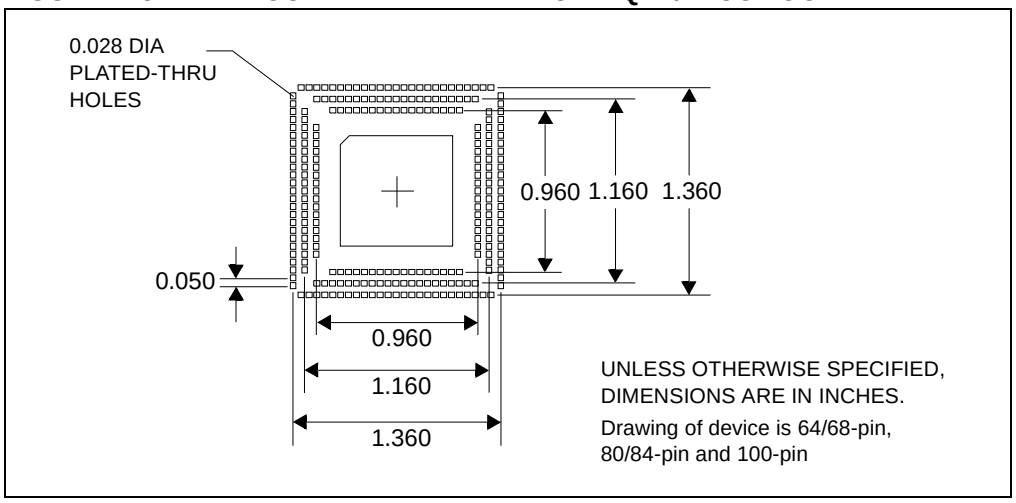
TQFP/PLCC device adapter footprints shown will accept board stackers like Samtec series DWM 0.050 Pitch Stackers. These stackers can be soldered into place during development/emulation and eliminate the need for other sockets.

FIGURE B-2: SINGLE-ROW TQFP/PLCC FOOTPRINT



UNLESS OTHERWISE SPECIFIED,
DIMENSIONS ARE IN INCHES.
Drawing of device is 44-pin TQFP/PLCC.

FIGURE B-3: DOUBLE AND TRIPLE-ROW TQFP/PLCC FOOTPRINT



Header pin-out matches the PLCC package. PLCC will map to TQFP as follows:

Header to 44-pin TQFP – one-to-one mapping.

Appendix C. Emulation Header Connections

C.1 INTRODUCTION

The following types of emulation header connectors are described here. Information on connecting development tools to the header is presented, as well.

The topics discussed here are:

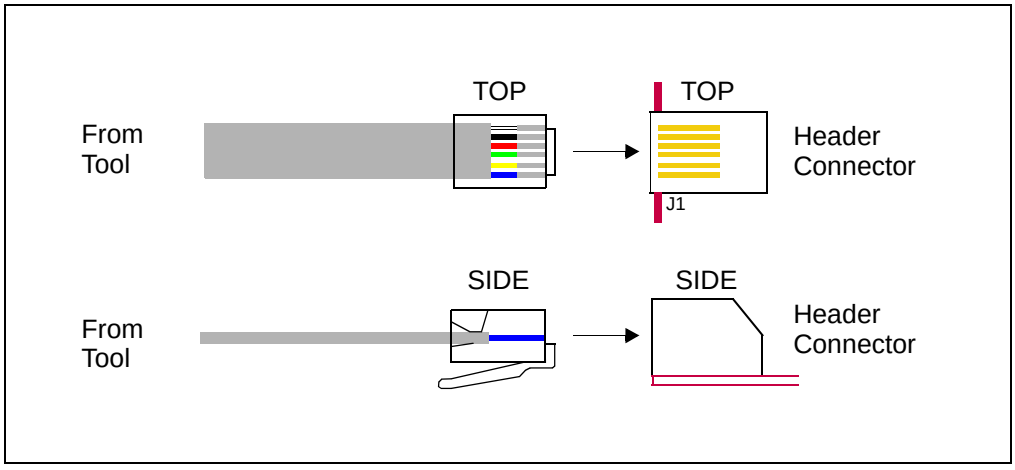
- 6-Pin Modular Connector
- 6-Pin SIL Connector
- Modular-to-SIL Adapter
- Ordering Information

C.2 6-PIN MODULAR CONNECTOR

Emulation headers with 6-pin modular (RJ-11/ICSP) connectors can connect directly to the following tools:

- MPLAB REAL ICE in-circuit emulator (Standard Driver Board)
- MPLAB ICD 2 or 3

FIGURE 1: MODULAR CONNECTION



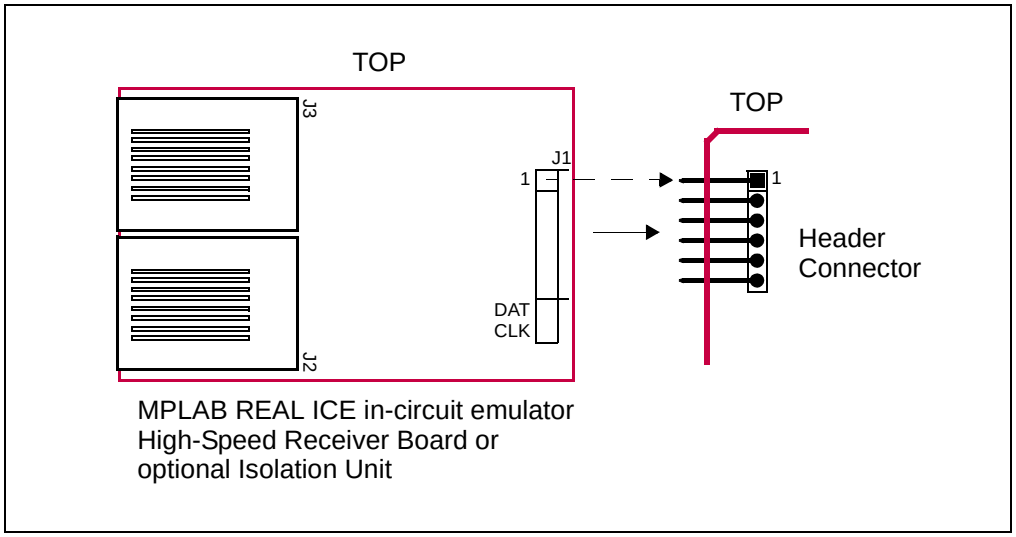
C.3 6-PIN SIL CONNECTOR

Emulation headers with 6-pin SIL (Single In-Line) connectors are compatible with the PICkit 3 but can also be used with the MPLAB REAL ICE in-circuit emulator.

The 6-pin modular cable attached to the Standard Driver Board may be connected to the 6 header pins through the Modular-to-SIL Adapter.

The 8-pin socket of the High Speed Driver Board or optional Isolation Unit may be connected directly to the 6 header pins. Be sure to line up pin 1 on the board with pin 1 on the header.

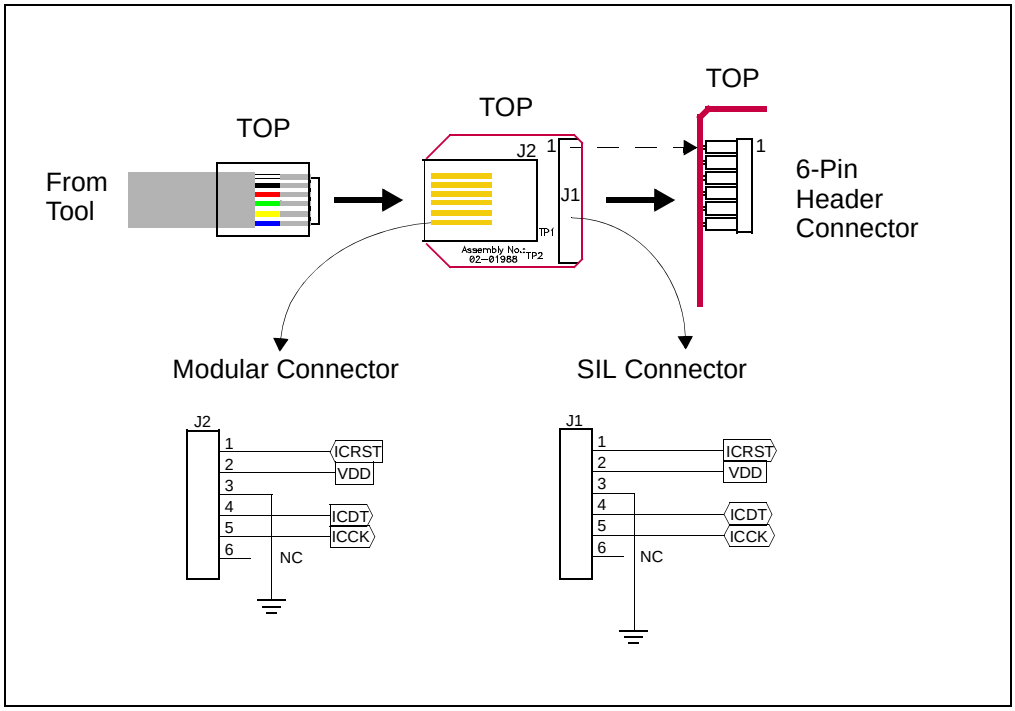
FIGURE C-1: 6-PIN SIL CONNECTION TO AN EMULATOR



C.4 MODULAR-TO-SIL ADAPTER

You can use this adapter for a 6-pin modular connector to an 6-pin SIL connector. Ensure that you line up pin 1 of J1 with pin 1 of the 6-pin header connector.

FIGURE C-2: MODULAR-TO-SIL ADAPTER CONNECTION



C.5 ORDERING INFORMATION

To order the development tools and other hardware shown here, please refer to the table below.

TABLE C-1: MICROCHIP HARDWARE ORDERING NUMBERS

Hardware	Order #
MPLAB REAL ICE in-circuit emulator (Standard Communication)	DV244005
MPLAB REAL ICE in-circuit emulator (High-Speed Communication) – Performance Pak	AC244002
MPLAB REAL ICE Isolation Unit (works with High-Speed Communication)	AC244005
Modular-to-SIL Adapter	AC164110

NOTES:

Appendix D. Revision History

D.1 Revision A (2014)

Release of original document.

D.2 Revision B (October 2015)

- Chapter 1. "EEP and Emulation Header Overview" - reorganized for better information flow.
- Chapter 2. "Emulation Header Features" - reorganized for better information flow. Added new content for tool support table, "Breakpoint, Runtime Watch, and Trace Resources", "Runtime Watches", "View Hardware Stack on Halt", and "Previous Program Counter". Added content to "Hardware Address/Data Breakpoints" and "Enhanced Event Breakpoints."
- Chapter 3. "Emulation Header List" - Added AC244065 and AC244066.
- Appendix A. "-ME2 Silicon Errata" - Added this chapter. Some content has been taken from a previous chapter.

NOTES:

Index

Numerics

6-Pin Modular Connector 73
6-Pin SIL Connector 75
8-Pin SIL Connector 74

A

AC162050 10
AC162052 12
AC162053 14
AC162054 14
AC162055 12
AC162056 12
AC162057 12
AC162058 10
AC162059 16
AC162060 18
AC162061 20
AC162062 39
AC162064 42
AC162065 44
AC162066 22
AC162067 47
AC162070 16
AC162074 47
AC162078 50
AC162079 39
AC162083 24
AC162087 39
AC162088 53
AC162091 39
AC162094 53
AC162096 16
AC244022 44
AC244023 26
AC244024 26
AC244026 56
AC244027 56
AC244028 28
AC244033 59
AC244034 59
AC244035 61
AC244036 61
AC244043 63
AC244044 63
AC244045 30
AC244046 65
AC244047 65
AC244051 32
AC244052 32
AC244053 67
AC244054 67
Additional Information 7

C

Calibration Bits 7

I

ICE vs. ICD 3

J

Jumper Settings10, 12, 18, 22, 28, 39, 47, 50, 53, 56, 57, ... 61

M

Modular Connector 73
Modular-to-SIL Adapter 76
MPLAB ICE 2000 22

O

Ordering Hardware 77

P

PCM16YM0 22
Performance 7
PIC10F200 9
PIC10F202 9
PIC10F204 9
PIC10F206 9
PIC10F220 9
PIC10F222 9
PIC10F320 9
PIC10F322 9
PIC10LF320 9
PIC10LF322 9
PIC12F1501 9
PIC12F1822 ..... 35, 36
PIC12F1840 ..... 35, 36
PIC12F508 9
PIC12F509 9
PIC12F510 9
PIC12F519 9
PIC12F609 ..... 9, 24
PIC12F615 ..... 9, 24
PIC12F617 ..... 9, 24
PIC12F629 ..... 9, 10
PIC12F635 ..... 9, 12
PIC12F675 ..... 9, 10
PIC12F683 ..... 9, 10
PIC12HV609 ..... 9, 24
PIC12HV615 ..... 9, 24
PIC12LF1501 9
PIC16F1454 35
PIC16F1455 35
PIC16F1458 35

PIC16F1459 35
PIC16F1503 10
PIC16F1507 10
PIC16F1508 35
PIC16F1509 35
PIC16F1823 ..... 35, 36
PIC16F1824 ..... 35, 36
PIC16F1825 ..... 35, 36
PIC16F1826 35
PIC16F1827 35
PIC16F1829 ..... 35, 36
PIC16F1847 35
PIC16F1933 36
PIC16F1934 36
PIC16F1936 36
PIC16F1937 36
PIC16F1938 36
PIC16F1939 36
PIC16F505 9
PIC16F506 9
PIC16F526 9
PIC16F610 ..... 9, 24
PIC16F616 ..... 9, 24
PIC16F627A 9, 14
PIC16F628A 9, 14
PIC16F630 ..... 9, 12
PIC16F631 ..... 9, 20
PIC16F636 ..... 9, 12
PIC16F639 ..... 9, 22
PIC16F648A 10, 14
PIC16F676 ..... 10, 12
PIC16F677 ..... 10, 20
PIC16F684 ..... 10, 12
PIC16F685 ..... 10, 20
PIC16F687 ..... 10, 20
PIC16F688 ..... 10, 12
PIC16F689 ..... 10, 20
PIC16F690 ..... 10, 20
PIC16F716 ..... 10, 14
PIC16F722 35
PIC16F723 35
PIC16F724 35
PIC16F726 35
PIC16F727 35
PIC16F785 ..... 10, 18
PIC16HV610 ..... 9, 24
PIC16HV616 ..... 9, 24
PIC16HV785 ..... 10, 18
PIC16LF1454 35
PIC16LF1455 35
PIC16LF1458 35
PIC16LF1459 35
PIC16LF1503 10
PIC16LF1507 10
PIC16LF1508 35
PIC16LF1509 35
PIC16LF1826 35
PIC16LF1827 35
PIC16LF1847 35
PIC16LF1933 36
PIC16LF1934 36
PIC16LF1936 36
PIC16LF1937 36
PIC16LF1938 36
PIC16LF1939 36
PIC16LF722 35
PIC16LF723 35
PIC16LF724 35
PIC16LF726 35
PIC16LF727 35
PIC18F1230 ..... 36, 50
PIC18F1330 ..... 36, 50
PIC18F13K22 36
PIC18F13K50 10
PIC18F14K22 36
PIC18F14K50 10
PIC18F24J10 36
PIC18F25J10 ..... 36, 47
PIC18F44J10 36
PIC18F45J10 ..... 36, 47
PIC18F63J11 37
PIC18F63J90 37
PIC18F64J11 37
PIC18F64J16 37
PIC18F64J90 37
PIC18F64J95 37
PIC18F65J10 37
PIC18F65J11 37
PIC18F65J15 37
PIC18F65J16 37
PIC18F65J50 37
PIC18F65J55 37
PIC18F65J90 37
PIC18F66J10 37
PIC18F66J11 37
PIC18F66J15 37
PIC18F66J16 37
PIC18F66J50 37
PIC18F66J55 37
PIC18F66J60 38
PIC18F66J65 38
PIC18F67J10 37
PIC18F67J11 37
PIC18F67J50 37
PIC18F67J60 38
PIC18F83J11 37
PIC18F83J90 37
PIC18F84J11 37
PIC18F84J16 37
PIC18F84J90 37
PIC18F84J95 37
PIC18F85J10 37
PIC18F85J11 37
PIC18F85J15 37
PIC18F85J16 37
PIC18F85J50 37
PIC18F85J55 37
PIC18F85J90 37
PIC18F86J10 37
PIC18F86J11 37

PIC18F86J15 37
PIC18F86J16 37
PIC18F86J50 37
PIC18F86J55 37
PIC18F86J60 38
PIC18F86J65 38
PIC18F87J10 37
PIC18F87J11 37
PIC18F87J50 37
PIC18F87J60 38
PIC18F96J60 38
PIC18F96J65 38
PIC18F97J60 38
PIC18LF13K22 36
PIC18LF13K50 10
PIC18LF14K22 36
PIC18LF14K50 10
PIC18LF24J10 36
PIC18LF25J10 ..... 36, 47
PIC18LF44J10 36
PIC18LF45J10 ..... 36, 47
PIC24F04KA200 10
PIC24F04KA201 10
PIC24F08KA101 38
PIC24F08KA102 38
PIC24F16KA101 38
PIC24F16KA102 38
PIC24FJ128GA006 38
PIC24FJ128GA008 38
PIC24FJ128GA010 38
PIC24FJ16GA002 38
PIC24FJ16GA004 38
PIC24FJ32GA002 38
PIC24FJ32GA004 38
PIC24FJ48GA002 38
PIC24FJ48GA004 38
PIC24FJ64GA002 38
PIC24FJ64GA004 38
PIC24FJ64GA006 38
PIC24FJ64GA008 38
PIC24FJ64GA010 38
PIC24FJ96GA006 38
PIC24FJ96GA008 38
PIC24FJ96GA010 38
PICDEM HPC Explorer Board 39
Pin Count 9, 35
Programming Non-ICD Devices 4

S

SIL Connector, 6 Pin 75
SIL Connector, 8 Pin 74
Switch Settings 20
Switch Settings, Rotary 24

T

Transition Socket 5

V

Vdd Max 9, 35
Vddcore Max 9, 35

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110

Canada - Toronto
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon

Hong Kong
Tel: 852-2943-5100
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Dongguan
Tel: 86-769-8702-9880

China - Hangzhou
Tel: 86-571-8792-8115
Fax: 86-571-8792-8116

China - Hong Kong SAR
Tel: 852-2943-5100
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8864-2200
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

ASIA/PACIFIC

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-3019-1500

Japan - Osaka
Tel: 81-6-6152-7160
Fax: 81-6-6152-9310

Japan - Tokyo
Tel: 81-3-6880- 3770
Fax: 81-3-6880-3771

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7828

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Dusseldorf
Tel: 49-2129-3766400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Venice
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Poland - Warsaw
Tel: 48-22-3325737

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820

07/14/15