
SQTPSM File Format Specification

INTRODUCTION

This document shows how a Serial Quick Turn Programming (SQTPSM) file is produced and used by MPLAB[®] IPE Integrated Programming Environment. Engineers can use this information to generate their own SQTP file.

- [Overview](#)
- [Example SQTP File](#)
- [Intel HEX File Format](#)
- [MPLAB IPE SQTP File Generation](#)
- [Customer Generated SQTP Files](#)
- [Programming of Devices](#)
- [Understanding Usage of RETLW in SQTP File for Midrange and Baseline Devices](#)
- [Examples of SQTP Files for Various Memory Regions](#)
- [Differences in SQTP File Behavior Between MPLAB IPE v2.35 \(and Before\) and MPLAB IPE v2.40 \(and Later\)](#)
- [Differences in the SQTP Feature Between MPLAB IDE v8.xx and MPLAB IPE for the Flash Data Memory Region](#)

OVERVIEW

Serialization is a method of programming microcontrollers whereby each chip is programmed with a slightly different code. Typically, all locations are programmed with the same basic code, except for a few contiguous bytes. Those bytes are programmed with a different number (referred to as “key” or “ID number” or “serial number”) in a program region. Typical applications for such programming are where each unit must have a different access code, as in the Internet of Things (IoT), car alarms or garage door openers.

An SQTP file (.num) contains the serial numbers to be used as each device is programmed.

Microchip devices require that the serial number must reside in contiguous locations, with up to 256 locations. The minimum number of bytes in a line of SQTP should be the Word size of the memory. It can be a multiple of the Word size, up to 256 bytes.

SQTP File Format Specification

EXAMPLE SQTP FILE

Note: This section “Example SQTP File” uses material from the Wikipedia article [Intel HEX](#), which is released under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](#) (view authors).

The following is an example of a typical SQTP file. The file is in [Intel HEX File Format](#). As a visual aid, the fields of Intel HEX records are colored as follows:

Start code
 Byte count
 Address
 Record type
 Data
 Checksum

TABLE 1-1: EXAMPLE SQTP FILE TYPICAL CONTENT

SQTP File Records	Record Description
:02000004740086	Extended Linear Address (Intel HEX Record Type = 04) Allows for 32-bit addressing (up to 4GB). The address field is ignored (typically 0000) and the byte count is always 02. The two encoded, big endian data bytes specify the upper 16 bits of the 32 bit absolute address for all subsequent type 00 records; these upper address bits apply until the next 04 record. If no type 04 record precedes a 00 record, the upper 16 address bits default to 0000. The absolute address for a type 00 record is formed by combining the upper 16 address bits of the most recent 04 record with the low 16 address bits of the 00 record.
:04000000BF7087ED59 ↓ :0400000043BD7F3449	Data (Intel HEX Record Type = 00) Contains data and a 16-bit starting address for the data. The byte count specifies the number of data bytes in the record. The first example record has 04 data bytes (BF, 70, 87, ED) located at consecutive addresses beginning at address 0000.
:00000001FF	End of File (Intel HEX Record Type = 01) Must occur exactly once per file in the last line of the file. The data field is empty (thus byte count is 00) and the address field is typically 0000.

INTEL HEX FILE FORMAT

Note: This section “Intel HEX File Format” uses material from the Wikipedia article *Intel HEX*, which is released under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](#) (view authors).

The SQTP file records (lines of text) conform to Intel® HEX file format. Intel HEX consists of lines of ASCII text that are separated by line feed or carriage return characters, or both. Each text line contains hexadecimal characters that encode multiple binary numbers. The binary numbers may represent data, memory addresses, or other values, depending on their position in the line, and the type and length of the line.

File Records

A record is a line of text that consists of six fields. Each field contains a character and multiple digits in the following order, from left to right:

- Field 1.** Start Code – 1 character
This character is an ASCII colon ‘:’.
- Field 2.** Byte Count – 2 hexadecimal digits
These two digits indicate the number of bytes (HEX digit pairs) in the data field. The maximum byte count is 255 (0xFF).
For example, 16 (0x10) and 32 (0x20) byte counts are commonly used.
- Field 3.** Address – 4 hexadecimal digits
These four digits represent the 16-bit beginning memory address offset of the data.
The physical address of the data is computed by adding this offset to a previously established base address. This provides memory addressing beyond the 64-kilobyte limit of 16-bit addresses. The base address, which defaults to zero, can be changed by various types of records.
Base addresses and address offsets are always expressed as big endian values.
- Field 4.** Record Type – 2 hexadecimal digits, 00 to 05
These two digits define the meaning of the data field. For SQTP, only three types are used; 00, 01, and 04 (see [Table 1-1](#)).
- Field 5.** Data – a sequence of n bytes of data
Data is represented by $2n$ hexadecimal digits. Some records omit this field (n equals zero).
The meaning and interpretation of the data bytes depends on the application.
- Field 6.** Checksum – 2 hexadecimal digits
These two digits represent a computed value that is used to verify the record has no errors by ensuring the summation of each of the bytes in the line add up to 0. See [Calculating the Checksum](#).

SQTP File Format Specification

Calculating the Checksum

As mentioned in the format table above, the last two characters represent a checksum of the data in the line. Since the checksum is a two-digit hexadecimal value, it may represent a value of 0 to 255, inclusive.

The checksum is calculated by summing the value of the data on the line, excluding the leading colon and checksum byte itself, and taking its two's complement. For example, consider the line:

```
:0300300002337A1E
```

Breaking this line into its components:

- Record Length: 03 (3 bytes of data)
- Address: 0030 (the 3 bytes will be stored at 0030, 0031, and 0032)
- Record Type: 00 (normal data)
- Data: 02, 33, 7A
- Checksum: 1E

Taking all the data bytes above, we have to calculate the checksum based on the following hexadecimal values:

$$03 + 00 + 30 + 00 + 02 + 33 + 7A = E2$$

The two's complement of E2 is 1E which is, as you can see, the checksum value. The two's complement of a number is the value which must be added to the number to reach the value 256 (decimal). That is to say, $E2 + 1E = 100$.

You may also calculate the two's complement by subtracting the value from 100h. In other words, $100h - E2h = 1Eh$, which is the checksum.

If the value in question is greater than FFh, simply take the part which is less than 100h.

For example, if you want the two's complement of the value 494h, simply drop the leading "4" which leaves you with 94h. The two's complement of 94h is 6Ch.

SQTP File Format Specification

MPLAB IPE SQTP FILE GENERATION

MPLAB IPE can be used to generate an SQTP file.

1. Launch MPLAB IPE.
2. Select *Settings>Advanced Mode*. In the “Advanced Settings” dialog, enter a password to enable Advanced Mode (the factory default is “microchip”). Click **Log on**.
3. Click on the **SQTP** tab on the left side of the window.
4. Set up SQTP file generation as described in [Table 1-2](#).

TABLE 1-2: MPLAB IPE SQTP SETTINGS

Setting	Description	Applies to
Generation Method:		
Random	Select this option to generate unique, random numbers for each part. There is no guarantee that the numbers will be non-repeating. However, the probability of such an occurrence is infinitesimally small for a reasonably large field.	SQTP Field 5. Data
Pseudo Random Seed Value (Hex):	Select this option to generate a pseudo-random set of non-repeating numbers based on the Hex value you enter in the Seed Value field. Pseudo-random sequences are, by definition, non-repeating until all possible values are used.	SQTP Field 5. Data
Sequential Start Value (Hex): Increment (Hex):	Select this option to generate sequential numbers based on the starting value specified and incrementing each number by the amount specified. The least significant digit is in the lowest memory location. The increment value must be between 1 and 255. Numbers are always in hexadecimal format, not in BCD or any other format.	SQTP Field 5. Data
Start Address (Hex)	Enter the starting address (in Hex) for the serial number.	SQTP Field 3. Address
Number of bytes (Dec)	Enter the size of the serial number (in decimal). Make sure a large enough serial number is specified for the number of parts planned to program using this file.	SQTP Field 5. Data
Number of parts (Dec)	Enter the number of parts to be programmed using this file.	SQTP Field 5. Data
Generate	Click Generate to create the SQTP (.num) file.	
Location:		
Program Memory, EEPROM, Auxiliary Memory, User ID Memory, Boot Memory, Flash Data	Select this option to load the SQTP number in the selected memory	Memories to program

SQTP File Format Specification

TABLE 1-2: MPLAB IPE SQTP SETTINGS (CONTINUED)

Access Method:		
RETLW	Select this option to use a series of RETLW (Return Literal W) instructions with the serial number bytes as the literal data. This selection is ignored if not applicable to the device chosen. See Understanding Usage of RETLW in SQTP File for Midrange and Baseline Devices .	8-bit devices
Raw Data	Select this option to use the raw data. This is used for most devices.	All PIC® and dsPIC® devices
Format for PSV	If the Raw Data option is selected, selecting Format for PSV formats SQTP data to make it compatible with PSV (Program Space Visibility).	PIC24, dsPIC devices

CUSTOMER GENERATED SQTP FILES

You can develop and use your own numbering scheme as long as it follows these guidelines:

- The number length should be determined from the native memory size of the device to be programmed and number of contiguous locations (up to 256 locations).
- Your file is in the format shown in [Example SQTP File](#).

For examples of MPLAB IPE generated SQTP files, see [Examples of SQTP Files for Various Memory Regions](#).

SQTP File Format Specification

PROGRAMMING OF DEVICES

Once the SQTP file has been produced, load the file into MPLAB IPE (Figure 1-2). Then MPLAB IPE will program the first device with basic code (including Configuration bits) and the first serial number from the SQTP file. Next it will program the second device with basic code and the second serial number. This will continue until the end of the SQTP file or number of devices to be programmed is reached. Then, depending on the settings under the Settings button, SQTP section (Figure 1-2), programming will halt or continue at the top of the file.

FIGURE 1-1: MPLAB IPE OPERATE - SQTP FILE SELECTION

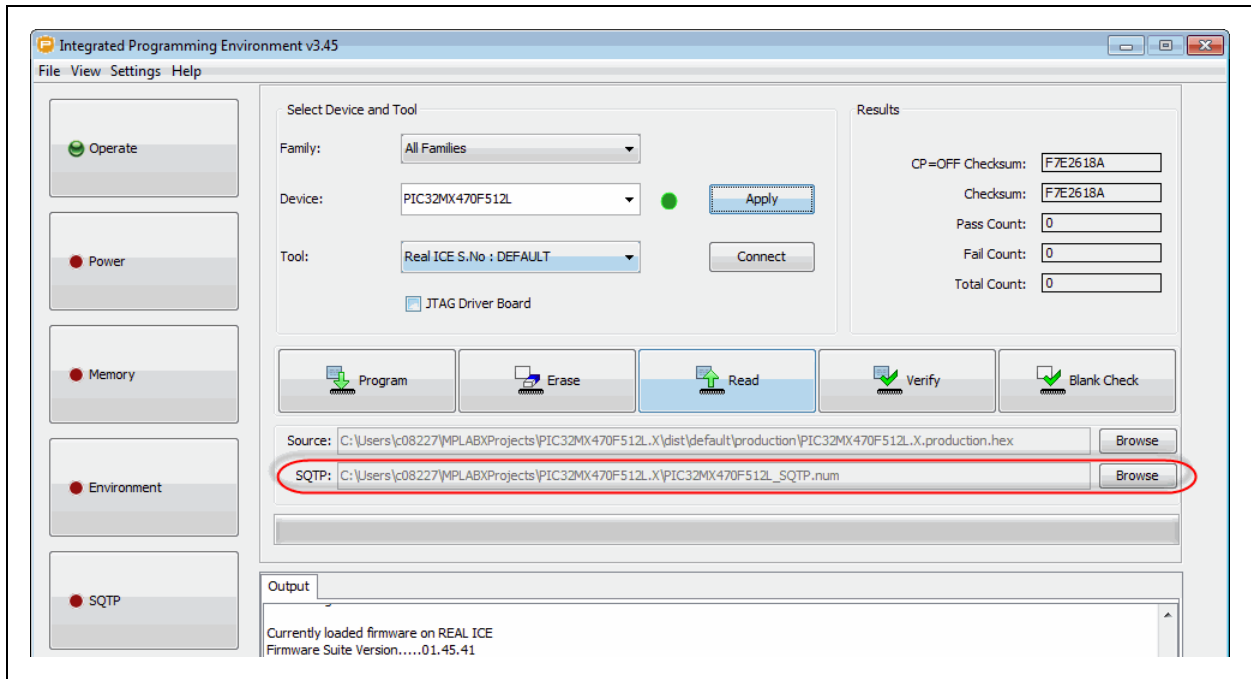
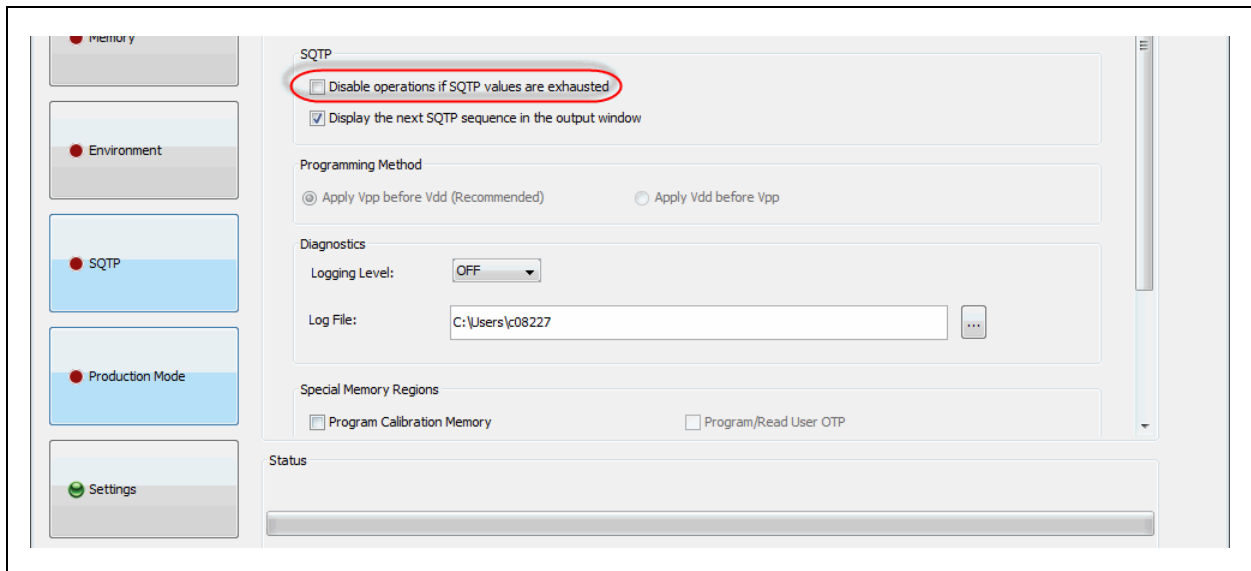


FIGURE 1-2: MPLAB IPE SETTINGS - SQTP SECTION



SQTP File Format Specification

UNDERSTANDING USAGE OF RETLW IN SQTP FILE FOR MIDRANGE AND BASELINE DEVICES

For midrange and baseline families of devices, the serial number must reside in contiguous locations, with up to 256 locations used. These locations must be coded in the finished product as RETLW NN, where NN = 8-bit random code. In MPLAB IPE, you would select “RETLW” under “Access Method” on the SQTP tab.

The first example presented is an implementation of a table in the program memory of baseline devices. Baseline devices use Harvard architecture, in which the program memory is separate from data memory.

All instructions operate on data that is fetched from the register file or data memory. Since there are no instructions to read from, or write to, the program memory, simply storing data Words in program memory is of no use. There is, however, a simple and elegant way to implement constant tables in the program memory by using the RETLW instruction. This instruction returns from a subroutine, as well as loads an 8-bit constant into the W register.

The following example shows how to get a byte of “serial information” from the table stored at location 0h.

In the SQTP file, the record would be (RETLW = 08h):

:16000000 01 08 02 08 03 08 04 08 05 08 06 08 07 08 08 08 86

Programmed into the device with the application:

```
ORG 0          ;store serial numbers
RETLW 01h
RETLW 02h
RETLW 03h
RETLW 04h
RETLW 05h
RETLW 06h
RETLW 07h
RETLW 08h ;end of serial numbers
.
.
main_prog ORG XYZ ;This is main program
.
.
MOVLW byte_num ;byte_num = 0 for 1st byte
CALL get_1byte;
.
.
get_1byte MOVWF PC ;write W to program counter
                ;W = offset = 0 for 1st byte
                ;end of get_1byte subroutine
.
.
END
```


SQTP File Format Specification

The second example shows how a serial number may reside at a location other than 0h (0000).

```
main_prog ORG XYZ ;This is main program
.
.
MOVLW byte_num ;byte_num = 0 for 1st byte
CALL get_1byte;
.
.
get_1byte ADDWFPC ;W = offset
RETLW 01h ;
RETLW 02h ;
RETLW 03h ;
RETLW 04h ;
RETLW 05h ;
RETLW 06h ;
RETLW 07h ;
RETLW 08h ;end of serial numbers
.
.
END
```

SQTP File Format Specification

EXAMPLES OF SQTP FILES FOR VARIOUS MEMORY REGIONS

Examples of various memory regions are provided for different devices and settings.

TABLE 1-3: SQTP FILE EXAMPLES BY MEMORY REGION

Memory Region	Device	SQTP settings	Generated SQTP file
Program Memory	PIC18F1220	Sequential increment: 1 Start Address: 0x0 Number of bytes: 2 Number of parts: 5 Access RETLW (0C): Yes	:0400000000C000CE4 :0400000010C000CE3 :0400000020C000CE2 :0400000030C000CE1 :0400000040C000CE0 :00000001FF
	PIC32MX360F512L	Sequential increment: 1 Start Address: 0x1D000000 Number of bytes: 4 Number of parts: 5 Access RETLW: NA	:02000004740086 :0400000000000000FC :0400000010000000FB :0400000020000000FA :0400000030000000F9 :0400000040000000F8 :00000001FF
User ID	PIC12F1501	Random values: Yes Start Address: 0x8000 Number of bytes: 2 Number of parts: 5 Access RETLW (34): Yes	:020000040001F9 :040000007E34CF340E :040000009034C5348E :040000000B34113468 :04000000F234F334CE :040000001C346834A4 :00000001FF
	PIC32MX360F512L	Sequential increment: 1 Start Address: 0x1FC02FF0 Number of bytes: 4 Number of parts: 5 Access RETLW: NA	:020000047F007B :04BFC000000000007D :04BFC000010000007C :04BFC000020000007B :04BFC000030000007A :04BFC0000400000079 :00000001FF
Auxiliary Memory	dsPIC33EP256MU806	Sequential increment: 1 Start Address: 0x7FC000 Number of bytes: 4 Number of parts: 5 Access RETLW: NA	:0200000401FFFA :0400000000000000FC :0400000010000000FB :0400000020000000FA :0400000030000000F9 :0400000040000000F8 :00000001FF
Boot Memory	PIC32MX110F016B	Random values: Yes Start Address: 0x1FC00000 Number of bytes: 4 Number of parts: 5 Access RETLW: NA	:020000047F007B :0400000039268EC748 :040000001FB777E2CD :04000000031E7E3D20 :04000000D56F64E272 :04000000F993C2A707 :00000001FF

SQTP File Format Specification

TABLE 1-3: SQTP FILE EXAMPLES BY MEMORY REGION (CONTINUED)

Memory Region	Device	SQTP settings	Generated SQTP file
EEPROM	PIC12F1840	Sequential increment: 1 Start Address: 0x0 Number of bytes: 2 Number of parts: 5 Access RETLW: NA	:020000040000FA :02F00000000000E :02F0000001000D :02F0000002000C :02F0000003000B :02F0000004000A :00000001FF
	PIC18F1220	Sequential increment: 1 Start Address: 0x0 Number of bytes: 2 Number of parts: 5 Access RETLW: NA	:0200000400F00A :020000000000FE :020000000100FD :020000000200FC :020000000300FB :020000000400FA :00000001FF

SQTP File Format Specification

APPENDIX A: DIFFERENCES IN SQTP FILE BEHAVIOR BETWEEN MPLAB IPE v2.35 (and Before) AND MPLAB IPE v2.40 (and Later)

In MPLAB IDE v8, the SQTP file generated for PIC32 MCUs correctly follows Intel HEX file format. However, when the file is loaded back into the IDE, the bytes are swapped. This is a bug.

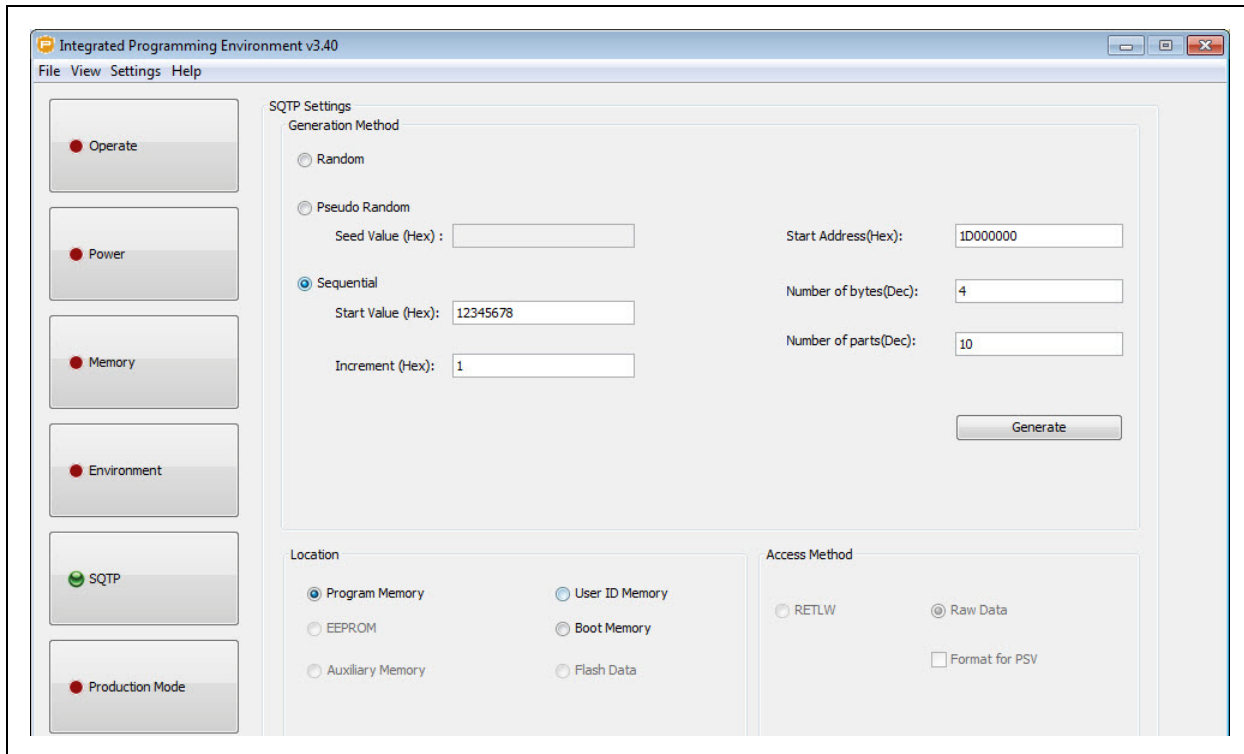
When MPLAB IPE was created (along with the next-generation IDE, MPLAB X IDE), it was decided to keep the behavior of MPLAB IDE v8 in MPLAB IPE as many customers had established work-arounds for this behavior. This continued up to v2.35.

In MPLAB IPE v2.40 it was decided to fix the original bug from MPLAB IDE v8 and offer an option to not swap the bytes. So you may now select 32-bit byte order (12345678) or 16-bit byte order (56781234) for PIC32 MCUs. This behavior will continue for following versions of MPLAB IPE.

Example

An SQTP file for PIC32MX110F016B with the settings shown in [Figure 1](#) is generated.

FIGURE 1: MPLAB® IPE SQTP OPTIONS



SQTP File Format Specification

When the SQTP file is loaded into MPLAB IDE v8, the resulting memory view is shown in [Figure 2](#). When the SQTP file is loaded into MPLAB IPE v2.35 or lower, the memory view is as shown in [Figure 3](#). When the SQTP file is loaded into MPLAB IPE v2.40 or higher, the 32-bit byte order the memory view is shown in [Figure 4](#).

FIGURE 2: MPLAB IDE v8 MEMORY VIEW

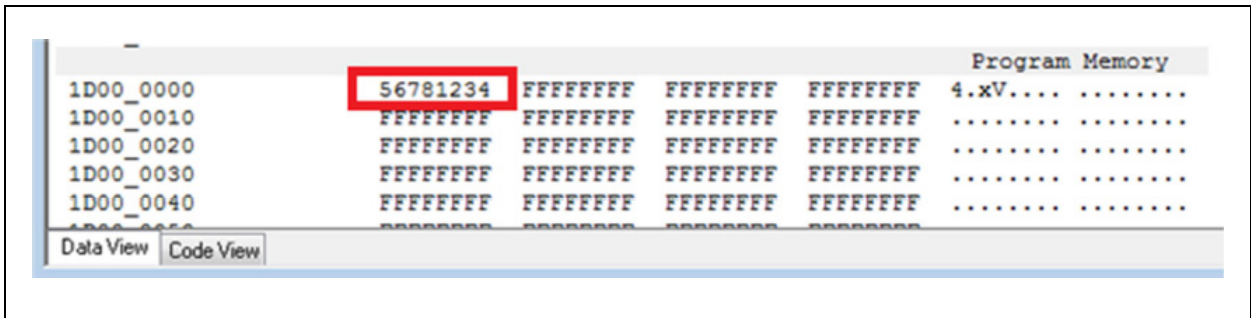


FIGURE 3: MPLAB IPE v2.35 OR BEFORE MEMORY VIEW

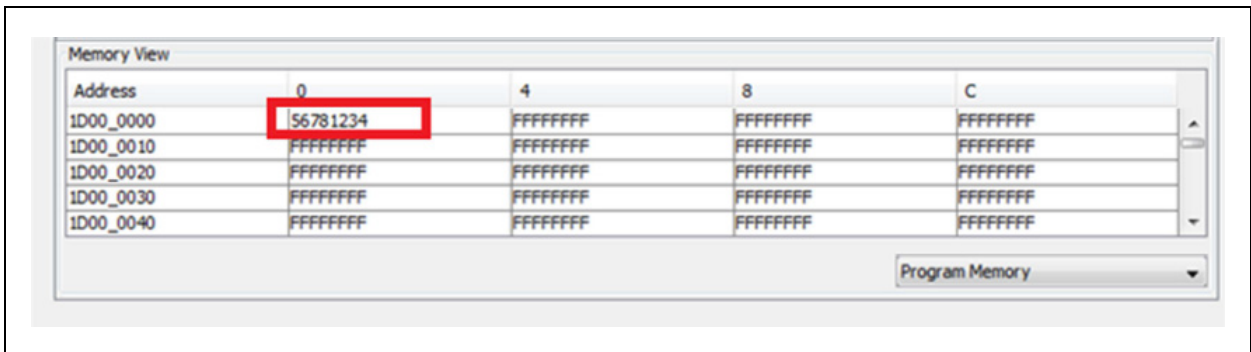
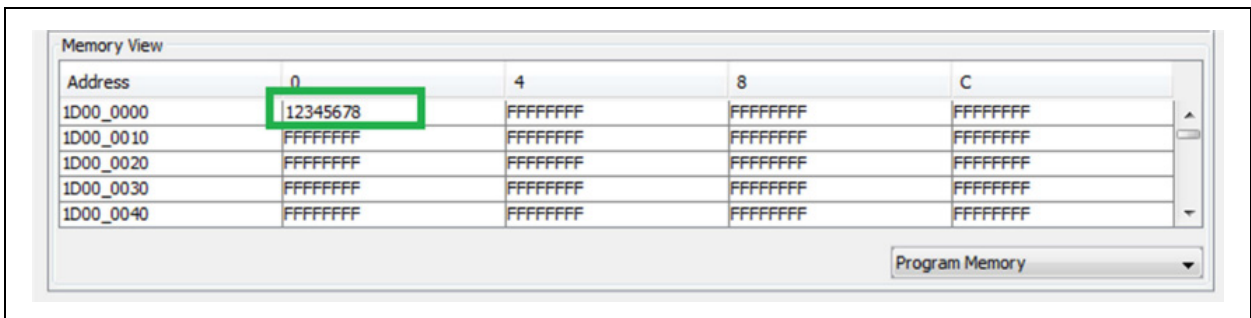


FIGURE 4: MPLAB IPE v2.40 OR LATER MEMORY VIEW



SQTP File Format Specification

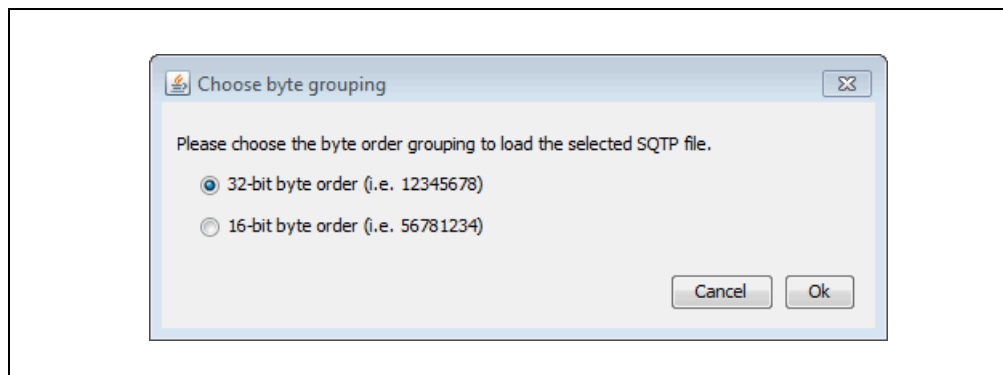
Note to New Users

Do not use the SQTP feature for PIC32 devices in MPLAB IDE v8.

Note to Existing MPLAB IDE Users

Please use MPLAB IPE to generate your SQTP file. You will be able to select the byte order for your file on **Generate**.

FIGURE 5: BYTE ORDER SELECTION



SQTP File Format Specification

APPENDIX B: DIFFERENCES IN THE SQTP FEATURE BETWEEN MPLAB IDE v8.XX AND MPLAB IPE FOR THE FLASH DATA MEMORY REGION

Table 1-4 shows examples of SQTP files created by MPLAB IDE v8.92 and several versions of MPLAB IPE for the PIC12F529T39A Flash Data memory region.

Issues with these files are discussed in the following sections:

- [Issue 1: RETLW Included in SQTP file](#)
- [Issue 2: SQTP Does Not Load Properly](#)
- [Issue 3: MPLAB IDE v8 SQTP File Addressing Not Intel Hex Standard](#)

Table 1-4: SQTP Files for the PIC12F529T39A

Example	IDE	Settings	SQTP File
1	MPLAB® IDE v8.92	Generation Method: Random* Location: Flash Data Access Method: RETLW (Raw Data selection unavailable) Start Address (Hex): 0 Number of Bytes (Dec): 2 Number of Parts (Dec): 5	:020000040000fa :04060000F808D1081D :040600005508B308DE :04060000850852080F :040600009B08EC085F :040600006008840802 :00000001FF
2	MPLAB IPE v2.20 (and before)	Generation Method: Random* Location: Flash Data Access Method: RETLW (Raw Data selection unavailable) Start Address (Hex): 600 Number of Bytes (Dec): 2 Number of Parts (Dec): 5	:020000040000FA :04060000F808D1081D :040600005508B308DE :04060000850852080F :040600009B08EC085F :040600006008840802 :00000001FF
3	MPLAB IPE v2.25	Generation Method: Random* Location: Flash Data Access Method: RETLW (Raw Data selection unavailable) Start Address (Hex): 600 Number of Bytes (Dec): 2 Number of Parts (Dec): 5	:020000040000FA :040C0000F808D10817 :040C00005508B308D8 :040C00008508520809 :040C00009B08EC0859 :040C000060088408FC :00000001FF
4	MPLAB IPE v2.26 (and later)	Generation Method: Random* Location: Flash Data Access Method: Raw Data (RETLW selection unavailable) Start Address (Hex): 600 Number of Bytes (Dec): 2 Number of Parts (Dec): 5	:020000040000FA :020C0000F8D129 :020C000055B3EA :020C000085521B :020C00009BEC6B :020C000060840E :00000001FF

* The same "random" serial numbers are shown in each example for comparison.

Issue 1: RETLW Included in SQTP file

In Examples 1, 2 and 3, the `RETLW` option for the SQTP file should not have been allowed for Flash data because Flash data is equivalent to the EEPROM region which should not contain the `RETLW` instruction.

Example 4 shows the issue fixed and does not have the option for `RETLW` in Flash data memory.

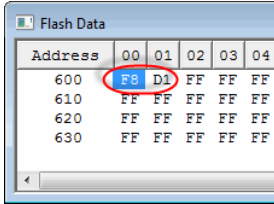
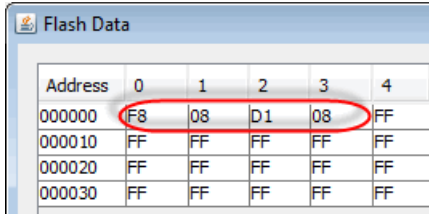
For the examples in Table 1-4, `RETLW` is `08h`.

SQTP File Format Specification

Issue 2: SQTP Does Not Load Properly

For Example 1, the incorrectly-added RETLW instruction in the SQTP file (see [Issue 1: RETLW Included in SQTP file](#)) is omitted when the file is reloaded into IDE memory. For Examples 2 and 3, the SQTP file is reloaded into IPE memory with the RETLW instruction. See [Table 1-5](#).

Table 1-5: SQTP Reloaded into IDE/IPE Memory

From Examples	Flash Data Memory View																														
1 of Table 1-4	 <table border="1"> <thead> <tr> <th>Address</th> <th>00</th> <th>01</th> <th>02</th> <th>03</th> <th>04</th> </tr> </thead> <tbody> <tr> <td>600</td> <td>F8</td> <td>D1</td> <td>FF</td> <td>FF</td> <td>FF</td> </tr> <tr> <td>610</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> </tr> <tr> <td>620</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> </tr> <tr> <td>630</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> </tr> </tbody> </table>	Address	00	01	02	03	04	600	F8	D1	FF	FF	FF	610	FF	FF	FF	FF	FF	620	FF	FF	FF	FF	FF	630	FF	FF	FF	FF	FF
Address	00	01	02	03	04																										
600	F8	D1	FF	FF	FF																										
610	FF	FF	FF	FF	FF																										
620	FF	FF	FF	FF	FF																										
630	FF	FF	FF	FF	FF																										
2* and 3* of Table 1-4	 <table border="1"> <thead> <tr> <th>Address</th> <th>0</th> <th>1</th> <th>2</th> <th>3</th> <th>4</th> </tr> </thead> <tbody> <tr> <td>000000</td> <td>F8</td> <td>08</td> <td>D1</td> <td>08</td> <td>FF</td> </tr> <tr> <td>000010</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> </tr> <tr> <td>000020</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> </tr> <tr> <td>000030</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> <td>FF</td> </tr> </tbody> </table>	Address	0	1	2	3	4	000000	F8	08	D1	08	FF	000010	FF	FF	FF	FF	FF	000020	FF	FF	FF	FF	FF	000030	FF	FF	FF	FF	FF
Address	0	1	2	3	4																										
000000	F8	08	D1	08	FF																										
000010	FF	FF	FF	FF	FF																										
000020	FF	FF	FF	FF	FF																										
000030	FF	FF	FF	FF	FF																										

* Flash Data memory begins at 600h. MPLAB IPE memory window shows first address of this memory region as 0h.

Issue 3: MPLAB IDE v8 SQTP File Addressing Not Intel Hex Standard

MPLAB IDE v8 does not follow the Intel Hex file standard in addressing Flash data. There is no issue for those who generate and use the file in MPLAB IDE v8. However, there is an issue when the SQTP files from MPLAB IDE v8 are used in MPLAB IPE before v2.25. The issue has been fixed in MPLAB IPE v2.25 (and later).

As seen in the examples in [Table 1-4](#), Examples 1 and 2 have an SQTP file address of 0600, whereas Examples 3 and 4 correctly show 0C00.

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.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

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.

**QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
= ISO/TS 16949 =**

Trademarks

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Helder, JukeBlox, KEELOQ, KEELOQ logo, Klear, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KlearNet, KlearNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICKit, PICTail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2016, Microchip Technology Incorporated, All Rights Reserved.
ISBN: 978-1-5224-1024-9



MICROCHIP

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

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
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

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 - Guangzhou
Tel: 86-20-8755-8029

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: 86-21-3326-8000
Fax: 86-21-3326-8021

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: 886-7-213-7830

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

Finland - Espoo
Tel: 358-9-4520-820

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

France - Saint Cloud
Tel: 33-1-30-60-70-00

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-67-3636

Germany - Karlsruhe
Tel: 49-721-625370

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

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Ra'anana
Tel: 972-9-744-7705

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

Italy - Padova
Tel: 39-049-7625286

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

Norway - Trondheim
Tel: 47-7289-7561

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

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

Sweden - Gothenberg
Tel: 46-31-704-60-40

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

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