Case Number: **01431152**

Subject: **PIC18F47Q43 silicon bug PUSHL opcode**

Case Number 01431152	IDE Name
Original Case Number	IDE Version
Case Reason Hardware/Firmware Support	Compiler
Customer Subject PIC18F47Q43 silicon bug PUSHL opcode	Compiler Version
Date/Time Opened 3/30/2024 2:00 PM	Compiler Edition
Status Resolution Proposed	HPA Activation Key
Urgency Critical	Stack / Library Name / App Note
Primary Target Device PIC18F47Q43	Programmer / Debugger
Additional Microchip Devices	Case Category Microcontrollers and Processors
Non-Microchip Devices Involved	Case Sub-Category 8-bit Microcontrollers
Peripheral/Component	Disti Client Account Name
Check All Devices?	IsWorkingDirectlyCloudPartner
	OtherCloudServices

Description

The extended instruction set opcode PUSHL fails using some literal value:

PUSHL 0xEF
PUSHL 0xEE
PUSHL 0xED
PUSHL 0xEC
PUSHL 0xEB
PUSHL 0xE7
PUSHL 0xE6
PUSHL 0xE5
PUSHL 0xE4
PUSHL 0xE3
PUSHL 0xDE
PUSHL 0xDC

PUSHL 0xDB

All of these corrupt the FSR2 register, some fail to push the literal value.

Exactly how the opcode fails depends on what value is in the WREG and what bank is selected.

Our unit test can be found here:

https://github.com/dsoze1138/MPLABX_pic-as_examples/tree/master/18F47Q43_UART_picas_v615.X

Design Stage
Design
Application Details
This is a execution time dependent subsystem in a image processing test system. Selection of the controller vendor depends on the resolution of this issue.
Phone Access Number
0030054192
Customer Can Call in From:
3/30/2024 6:00 PM
Note
All times shown are based on your Time Zone settings.
View our Support Phone Numbers Here
Contact Name
Dan Soze
Case Origin
Community Portal

Proposed Resolution

Hi Dan,

Thank you for reaching out to Microchip!

By taking a look in the SFR, at these addresses that you have mentioned in your comment are the indirect operands PREINC, POSTINC, POSTDEC, PLUSW. Here is an explanation of what each of them does:

10.4.3.2 FSR Registers and POSTINC, POSTDEC, PREINC and PLUSW

In addition to the INDF operand, each FSR register pair also has four additional indirect operands. Like INDF, these are "virtual" registers that cannot be directly read or written. Accessing these registers actually accesses the location to which the associated FSR register pair points, and also performs a specific action on the FSR value. They are:

- POSTDEC: accesses the location to which the FSR points, then automatically decrements the FSR by 1 afterwards
- POSTINC: accesses the location to which the FSR points, then automatically increments the FSR by 1 afterwards
- PREINC: automatically increments the FSR by one, then uses the location to which the FSR points in the operation
- PLUSW: adds the signed value of the W register (range of -127 to 128) to that of the FSR and uses the location to which the result points in the operation.

Using the PUSHL instruction with one of these operands will also execute the respective operation, so that is why you may see unexpected behavior.

For instance, when you write PUSHL 0xDC, it will execute the PREINC function, which increments the FSR by one and then writes the respective literal value to the address pointed by FSR.

Thank you,

Kind regards,

Ana

Created by: Dan Soze (4/1/2024, 10:15)

Hi Ana,
The proposed resolution does not address the issue I have raised.

The PUSHL opcode is documented to write a LITERAL to memory addressed by FSR2 then decrement FSR2.

What your response describes is what the implementation in the PIC18F47Q43 silicon actually does. This is not what is described in the data sheet.

The issue that Microchip must answer is why the data sheet or errata for the PIC18F47Q43 fails to contain any information about why the PUSHL opcode does not act as documented in the data sheet.