

# I2C™ Master Library Module (Interrupt-driven)

<b>1.</b>	<b>Introduction .....</b>	<b>2</b>
<b>2.</b>	<b>Module Features.....</b>	<b>2</b>
<b>3.</b>	<b>List of Component Modules.....</b>	<b>3</b>
<b>4.</b>	<b>Using the Library Module in a Project.....</b>	<b>3</b>
<b>5.</b>	<b>List of Shared Parameters.....</b>	<b>4</b>
	<i>Shared Data Bytes .....</i>	<i>4</i>
	<i>Shared Functions .....</i>	<i>4</i>
	<i>Shared Macros .....</i>	<i>4</i>
<b>6.</b>	<b>Functions .....</b>	<b>5</b>
<b>7.</b>	<b>Macros.....</b>	<b>8</b>
<b>8.</b>	<b>Error and Status Flags.....</b>	<b>9</b>

## 1. Introduction

The I2CMIInt is a general-purpose library module. It configures the MSSP module in the Master mode and helps in communicating with the I2C<sup>TM</sup> Slave. If this library module is used with a device not having the MSSP module, then the following message is displayed while compiling. "This controller does not have MSSP".

The module code is linkable and relocatable, which provides the user, the facility to use it without modifications.

It provides interrupt-based operation and has its own Tx/Rx buffer, which provides maximum benefit of parallel processing.

By using this Module one can write his application to interact with any of the I2C Slaves like the EEPROM, ADC, Digital Potentiometer, LCD, etc.

The module allows the user to concentrate more on his application's development by providing these library functions.

## 2. Module Features

It supports following features: -

- It provides simple and primitive functions to communicate with the I2C Slave.
- User defined length of the Tx/Rx Buffer.
- Interrupt driven transmission and reception.
- It provides the option to enable the SMBus specific inputs.
- It generates Error flags on the occurrence of an error. All error conditions are passed through the 'I2CMIIntStatus' Register.

### 3. List of Component Modules

<code>I2CMIInt.P16.ex.txt</code>	This is an example file developed to demonstrate the use of the library functions for the PIC16 family.
<code>I2CMIInt.P18.ex.txt</code>	This is an example file developed to demonstrate the use of the library functions for the PIC18 family.
<code>I2CMIInt.asm</code>	This is the I2C Master code implementation file. <u>One needs to include this file in their project.</u>
<code>16I2CMI.asm</code>	This is the I2C Master code implementation file for the PIC16 family. The <code>I2CMIInt.asm</code> file will include this file if the PIC16 family processor is used.
<code>18I2CMI.asm</code>	This is the I2C Master code implementation file for the PIC18 family. The <code>I2CMIInt.asm</code> file will include this file if the PIC18 family processor is used.
<code>I2CMIInt.inc</code>	This file contains the definitions of all the shared parameters and the macros. <u>One needs to include this in the Assembly file</u> where the library functions and macros are called. This file takes care of the definitions of all the Extern Global parameter so one can directly call the library routines in their program.
<code>P16xxx.inc</code>	General purpose processor definition file for the PIC16 family
<code>P18xxx.inc</code>	General purpose processor definition file for the PIC18 family

### 4. Using the Library Module in a Project

Please follow the steps below to use this library module in your project.

1. Use the Application Maestro to configure the module as required.
2. At the 'Generate Files' step, save the output to the directory where your project code resides.
3. Launch the MPLAB, and open the project's workspace.
4. Verify that the Microchip language tool suite is selected (*Project>Select Language Toolsuite*).
5. In the Workspace view, right-click on the "Source Files" node. Select the "Add Files" option.  
Select the file `I2CMIInt.asm` and click **OK**.
6. Now right-click on the "Linker Scripts" node and select "Add Files". Add the appropriate linker file (`.lkr`) for the project's target microcontroller.
7. Add any other files that the project may require. Save and close the project.
8. In your main source (assembler) file, add the `include` directive at the head of the code listing to include the file `I2CMIInt.inc`. By doing so, all files required to make the generated code work in your project will be included by reference when you build the project.
9. To use the module in your application, invoke the functions or the macros as needed.

## 5. List of Shared Parameters

### ***Shared Data Bytes***

<code>vI2CIntStatus</code>	It is the Error/Status register. The details of each bit of this register is explained in "Section 8"
----------------------------	--

### ***Shared Functions***

<code>I2CIntInit</code>	It is used for the Synchronous Serial Port Initialization It initializes the Port according to the options opted through the Application Maestro.
<code>I2CIntStart</code>	It generates START condition on the I2C Bus.
<code>I2CIntReStart</code>	It generates Repeated START condition on the I2C Bus.
<code>I2CIntPut</code>	It is used for transmitting a byte on the I2C Bus.
<code>I2CIntSetGetCount</code>	It is used for specifying the number of bytes to be received.
<code>I2CIntGet</code>	It is used for reading the received byte.
<code>I2CIntStop</code>	It generates the STOP condition on the I2C Bus.
<code>I2CIntISR</code>	It is called from the interrupt handler.
<code>I2CIntDiscardBuffer</code>	It is used for discarding the buffer.

### ***Shared Macros***

<code>mI2CIntDisable</code>	It disables the Synchronous Serial Port.
<code>mSetI2CSIntHighPriority</code>	It sets the interrupt priority of the MSSP as High.
<code>mSetI2CSIntLowPriority</code>	It sets the interrupt priority of the MSSP as Low.

## 6. Functions

Function	<code>I2CIntInit</code>
Pre-conditions	None
Overview	This function is used for initializing the MSSP module. It initializes the module according to the Application Maestro options.
Input	Application Maestro options
Output	None
Side Effects	Bank selection bits and 'W' register are changed
Stack Requirement	1 level deep
Maximum T-Cycles taken	15 Cycles by the PIC16 family 12 Cycles by the PIC18 family

Function	<code>I2CIntStart</code>
Pre-conditions	The function ' <code>I2CIntInit</code> ' should have been called
Overview	This function is used for generating the Start condition on the I2C bus. It initializes the buffer. In the Multi-Master setup it also checks the prior state of the bit ' <code>SSPSTAT&lt;S&gt;</code> '.
Input	None
Output	Single-Master setup: None Multi-Master setup: If the bit ' <code>SSPSTAT&lt;S&gt;</code> ' is set prior to sending the Start bit then, the bit ' <code>I2CIntStatus&lt;I2CErrBusBusy&gt;</code> ' is set.
Side Effects	Bank selection bits and 'W' register are changed
Stack Requirement	1 level deep
Maximum T-Cycles taken	26 Cycles by the PIC16 family 19 Cycles by the PIC18 family

Function	<code>I2CIntReStart</code>
Pre-conditions	The function ' <code>I2CIntPut</code> ' should have been called
Overview	This function is used for generating the repeated Start condition.
Input	None
Output	None
Side Effects	Bank selection bits and 'W' register are changed
Stack Requirement	1 level deep
Maximum T-Cycles taken	17 Cycles by the PIC16 family 13 Cycles by the PIC18 family

Function	I2CMIIntStop
Pre-conditions	The function 'I2CMIIntPut' or the function 'I2CMIIntSetGetCount' should have been called
Overview	This function is used for generating the Stop condition on the I2C bus.
Input	None
Output	None
Side Effects	Bank selection bits and 'W' register are changed
Stack Requirement	1 level deep
Maximum T-Cycles taken	22 Cycles by the PIC16 family 16 Cycles by the PIC18 family

Function	I2CMIIntPut
Pre-conditions	The function 'I2CMIIntStart' should have been called.
Overview	This function sends the byte in the 'W' Reg. over I2C bus, if the bus is free, else saves the byte in the buffer.
Input	'W' Register.
Output	If 'W' is written into the buffer then, the register I2CMIIntTxDataCount is incremented and the bit 'vI2CMIIntStatus<I2CMBufEmpty>' is cleared. If the buffer gets full then, the bit 'vI2CMIIntStatus<I2CMBufFull>' is set.
Side Effects	Bank selection bits and 'W' register are changed
Stack Requirement	1 level deep
Maximum T-Cycles taken	50 Cycles by the PIC16 family 45 Cycles by the PIC18 family

Function	I2CMIIntGet
Pre-conditions	The function 'I2CMIIntSetGetCount' should have been called and the bit 'vI2CMIIntStatus<I2CMDDataReady>' should read '1'.
Overview	This function reads the byte received.
Input	None
Output	'W' Register will have the read byte. The bit 'vI2CMIIntStatus<I2CMBufFull>' is cleared. If the buffer gets empty then, the bit 'vI2CMIIntStatus<I2CMBufEmpty>' is set and the bit 'vI2CMIIntStatus<I2CMDDataReady>' is cleared.
Side Effects	Bank selection bits and 'W' register are changed
Stack Requirement	1 level deep
Maximum T-Cycles taken	49 Cycles by the PIC16 family 42 Cycles by the PIC18 family

Function	I2CMIIntSetGetCount
Pre-conditions	The function 'I2CMIIntStart' should have been called.
Overview	This function sets the number of bytes to be received.
Input	'W' Register.
Output	I2CMIIntRxDataCount is added with 'W'.
Side Effects	Bank selection bits are changed
Stack Requirement	1 level deep
Maximum T-Cycles taken	27 Cycles by the PIC16 family 19 Cycles by the PIC18 family

  

Function	I2CMIIntISR
Pre-conditions	This function should be called from the interrupt handler.
Overview	This is the interrupt service routine for SSPIF and BCLIF. This handles the transmission and reception of bytes.
Input	None
Output	While transmission is going on, if the buffer is full then, the bit vI2CMIIntStatus<I2CMBufFull> is set. If there is no-acknowledge then, the bit vI2CMIIntStatus<I2CMErrNoAck> is set. If the bus collision occurs then, the bit vI2CMIIntStatus<I2CMErrBusCollision> is set. While reception is going on, if the buffer is empty then, the bit vI2CMIIntStatus<I2CMBufEmpty> is set, else the bit vI2CMIIntStatus<I2CMDDataReady> is set.
Side Effects	Bank selection bits and 'W' register are changed
Stack Requirement	2 level deep
Maximum T-Cycles taken	79 Cycles by the PIC16 family 70 Cycles by the PIC18 family

  

Function	I2CMIIntDiscardBuf
Pre-conditions	The function 'I2CMIIntPut' is called or the function 'I2CMIIntSetGetCount' is called and the bit 'vI2CMIIntStatus<I2CMDDataReady>' is set.
Overview	This function flushes the buffer.
Input	None
Output	None.
Side Effects	Bank selection bits are changed
Stack Requirement	1 level deep
Maximum T-Cycles taken	11 Cycles by the PIC16 family 10 Cycles by the PIC18 family

## 7. Macros

Macro	<code>mI2CIntDisable</code>
Overview	Pre-conditions- The function 'I2CIntStop' should have been called and the bit 'I2CIntStatus<I2CMBusy>' is '0'. Disables the MSSP module.
Input	None
Output	None
Side Effects	Bank selection bits are changed.
Stack Requirement	None
Maximum T-Cycles taken	3 Cycles by the PIC16 family 1 Cycle by the PIC18 family

Macro	<code>mSetI2CIntHighPriority</code> (Valid only for the PIC18 family devices).
Overview	Pre-conditions- The bit RCON<IPEN> should be set. This sets the interrupt priority of the MSSP as High.
Input	None
Output	None
Side Effects	Bank selection bits are changed.
Stack Requirement	None
Maximum T-Cycles taken	2 Cycles by the PIC18 family

Macro	<code>mSetI2CIntLowPriority</code> (Valid only for the PIC18 family devices).
Overview	Pre-conditions- The bit RCON<IPEN> should be set. This sets the interrupt priority of the MSSP as Low.
Input	None
Output	None
Side Effects	Bank selection bits are changed.
Stack Requirement	None
Maximum T-Cycles taken	2 Cycles by the PIC18 family



## 8. Error and Status Flags

All errors/statuses are set as a content of the register 'vI2CMIntStatus'. The individual errors/statuses are unique. Please refer the list below for the information.

I2CMErrNoAck	This indicates that, No-Acknowledge is received from the Slave after transmitting the byte.
I2CMErrBusBusy	This indicates that, some other Master is already using the bus. This can occur only in the Multi-Master setup.
I2CMErrBusCollision	This indicates that, the I2C Bus Collision has occurred. This can occur only in the Multi-Master setup.
I2CMBusy	This indicates that, the I2C Module is busy doing some pre-assigned work.
I2CMDDataReady	This indicates that, the data is received and is ready to be read.
I2CMBufFull	This indicates that, the buffer is full.
I2CMBufEmpty	This indicates that, the buffer is empty.
I2CMBufOverFlow	This indicates that, the buffer is over flowing.