

## I<sup>2</sup>C LIBRARY

I<sup>2</sup>C full master MSSP module is available with a number of PIC MCU models. *mikroC PRO for PIC* provides library which supports the master I<sup>2</sup>C mode.

**Note:** Some MCUs have multiple I<sup>2</sup>C modules. In order to use the desired I<sup>2</sup>C library routine, simply change the number 1 in the prototype with the appropriate module number, i.e. `I2C1_Init(100000);`

### Library Routines

- I2C1\_Init
- I2C1\_Start
- I2C1\_Repeated\_Start
- I2C1\_Is\_Idle
- I2C1\_Rd
- I2C1\_Wr
- I2C1\_Stop

### I2C1\_Init

<b>Prototype</b>	<code>void I2C1_Init(unsigned long clock);</code>
<b>Returns</b>	Nothing.
<b>Description</b>	<p>Initializes I<sup>2</sup>C with desired <code>clock</code> (refer to device data sheet for correct values in respect with Fosc). Needs to be called before using other functions of I<sup>2</sup>C Library.</p> <p>You don't need to configure ports manually for using the module; library will take care of the initialization.</p>
<b>Requires</b>	<p>Library requires MSSP module on PORTB or PORTC.</p> <p><b>Note:</b> Calculation of the I<sup>2</sup>C clock value is carried out by the compiler, as it would produce a relatively large code if performed on the library level. Therefore, compiler needs to know the value of the parameter in the compile time. That is why this parameter needs to be a constant, and not a variable.</p>
<b>Example</b>	<code>I2C1_Init(100000);</code>

## I2C1\_Start

<b>Prototype</b>	<code>unsigned short I2C1_Start(void);</code>
<b>Returns</b>	If there is no error, function returns 0.
<b>Description</b>	Determines if I <sup>2</sup> C bus is free and issues START signal.
<b>Requires</b>	I <sup>2</sup> C must be configured before using this function. See I2C1_Init.
<b>Example</b>	<code>I2C1_Start();</code>

## I2C1\_Repeated\_Start

<b>Prototype</b>	<code>void I2C1_Repeated_Start(void);</code>
<b>Returns</b>	Nothing.
<b>Description</b>	Issues repeated START signal.
<b>Requires</b>	I <sup>2</sup> C must be configured before using this function. See I2C1_Init.
<b>Example</b>	<code>I2C1_Repeated_Start();</code>

## I2C1\_Is\_Idle

<b>Prototype</b>	<code>unsigned short I2C1_Is_Idle(void);</code>
<b>Returns</b>	Returns 1 if I <sup>2</sup> C bus is free, otherwise returns 0.
<b>Description</b>	Tests if I <sup>2</sup> C bus is free.
<b>Requires</b>	I <sup>2</sup> C must be configured before using this function. See I2C1_Init.
<b>Example</b>	<code>if (I2C1_Is_Idle()) { ... }</code>

## I2C1\_Rd

<b>Prototype</b>	<code>unsigned short I2C1_Rd(unsigned short ack);</code>
<b>Returns</b>	Returns one byte from the slave.
<b>Description</b>	Reads one byte from the slave, and sends not <i>acknowledge</i> signal if parameter ack is 0, otherwise it sends <i>acknowledge</i> .
<b>Requires</b>	I <sup>2</sup> C must be configured before using this function. See I2C1_Init. Also, START signal needs to be issued in order to use this function. See I2C1_Start.
<b>Example</b>	Read data and send not acknowledge signal:  <code>unsigned short take; ... take = I2C1_Rd(0);</code>

### I2C1\_Wr

<b>Prototype</b>	<code>unsigned short I2C1_Wr(unsigned short data_);</code>
<b>Returns</b>	Returns 0 if there were no errors.
<b>Description</b>	Sends data byte (parameter data) via I <sup>2</sup> C bus.
<b>Requires</b>	I <sup>2</sup> C must be configured before using this function. See I2C1_Init. Also, START signal needs to be issued in order to use this function. See I2C1_Start.
<b>Example</b>	<code>I2C1_Write(0xA3);</code>

### I2C1\_Stop

<b>Prototype</b>	<code>void I2C1_Stop(void);</code>
<b>Returns</b>	Nothing.
<b>Description</b>	Issues STOP signal.
<b>Requires</b>	I <sup>2</sup> C must be configured before using this function. See I2C1_Init.
<b>Example</b>	<code>I2C1_Stop();</code>

## Library Example

This code demonstrates use of I<sup>2</sup>C library. PIC MCU is connected (SCL, SDA pins) to 24c02 EEPROM. Program sends data to EEPROM (data is written at address 2). Then, we read data via I<sup>2</sup>C from EEPROM and send its value to PORTB, to check if the cycle was successful (see the figure below how to interface 24c02 to PIC).

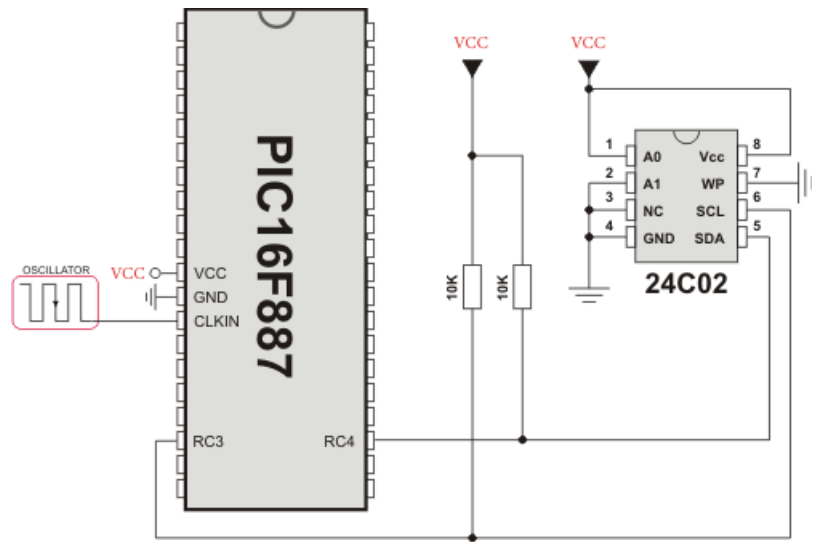
```
void main(){
    ANSEL  = 0;                // Configure AN pins as digital I/O
    ANSELH = 0;
    PORTB  = 0;
    TRISB  = 0;                // Configure PORTB as output

    I2C1_Init(100000);         // initialize I2C communication
    I2C1_Start();               // issue I2C start signal
    I2C1_Wr(0xA2);              // send byte via I2C (device address + W)
    I2C1_Wr(2);                 // send byte (address of EEPROM location)
    I2C1_Wr(0xF0);              // send data (data to be written)
    I2C1_Stop();               // issue I2C stop signal

    Delay_100ms();

    I2C1_Start();               // issue I2C start signal
    I2C1_Wr(0xA2);              // send byte via I2C (device address + W)
    I2C1_Wr(2);                 // send byte (data address)
    I2C1_Repeated_Start();      // issue I2C signal repeated start
    I2C1_Wr(0xA3);              // send byte (device address + R)
    PORTB = I2C1_Rd(0u);        // Read the data (NO acknowledge)
    I2C1_Stop();               // issue I2C stop signal
}
```

HW Connection



Interfacing 24c02 to PIC via I<sup>2</sup>C