TB3159

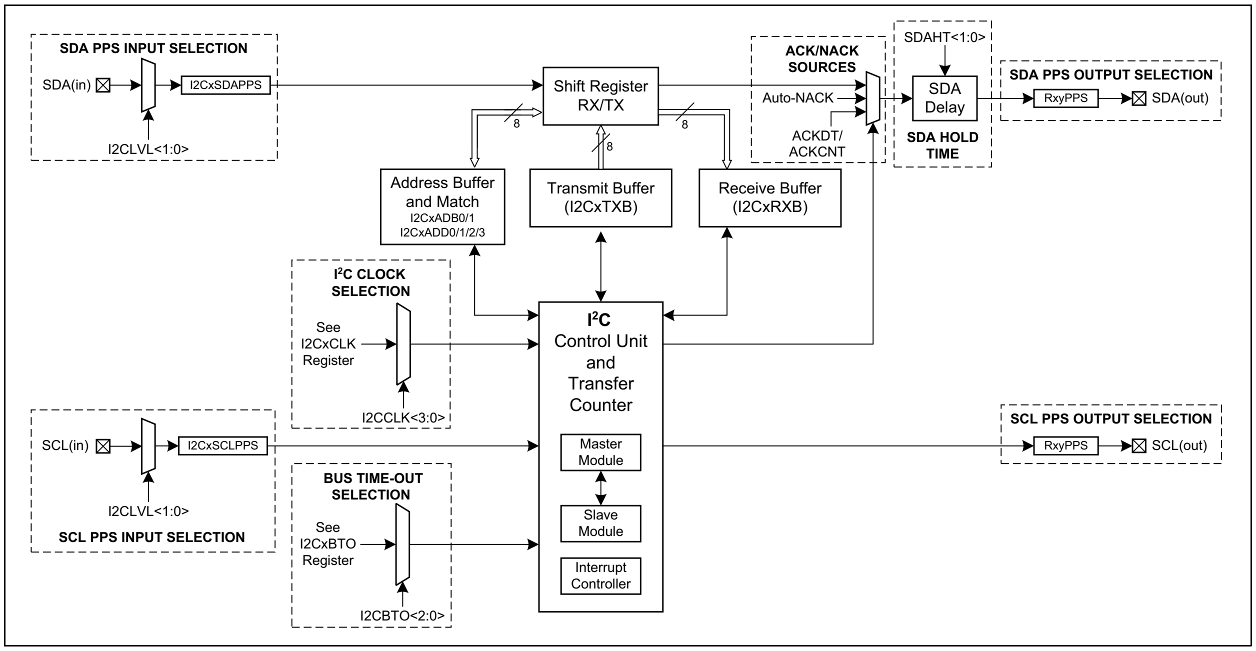
I²C Communications with Hardware Protocol Acceleration on 8-Bit PIC® Microcontrollers

Author: Mary Iva Rosario Salimbao
Microchip Technology Inc.

INTRODUCTION

This technical brief discusses the I²C module, its features and basic functionality. Figure 1 shows the simplified block diagram of the I²C module.

FIGURE 1: SIMPLIFIED I²C MODULE BLOCK DIAGRAM

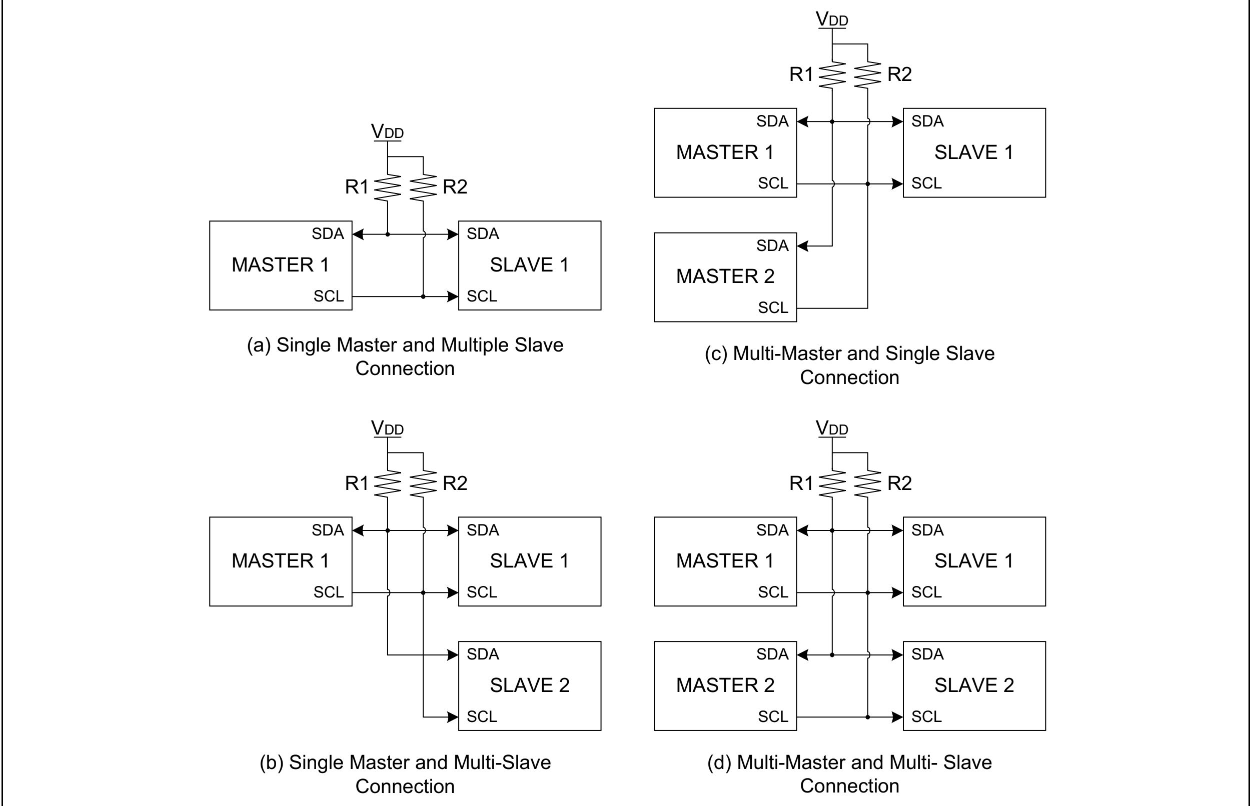


In Figure 1, the Control Unit houses both the master and slave modules for master and slave operation, and the Interrupt Controller for monitoring the state of the module. The Transfer Counter automates the transmission of data through an auto-count feature. Buffers hold a byte of data or address, while another is shifted in or out on the SDA pin by the Shift register. The ACK/NACK sources add a 9th bit to the transmission for Acknowledgment. These nine bits are shifted out in intervals set by the SDA Delay. Clock and Bus Time-out selections control the timing of the SCL and time-out, respectively. For the SDA and SCL pins, these are configured through Peripheral Pin Select (PPS).

I^{2}C PROTOCOL OVERVIEW

The I^{2}C module follows the Phillips® I^{2}C specification. The module provides a bidirectional master/slave synchronous interface between the PIC® microcontroller and other I^{2}C supported devices. These devices are connected via a two-wire serial bus, which allows multiple masters to communicate with multiple slaves. Figure 2 shows the different connection types between master and slave devices.

FIGURE 2: SAMPLE I^{2}C CONNECTION BETWEEN MASTERS AND SLAVES

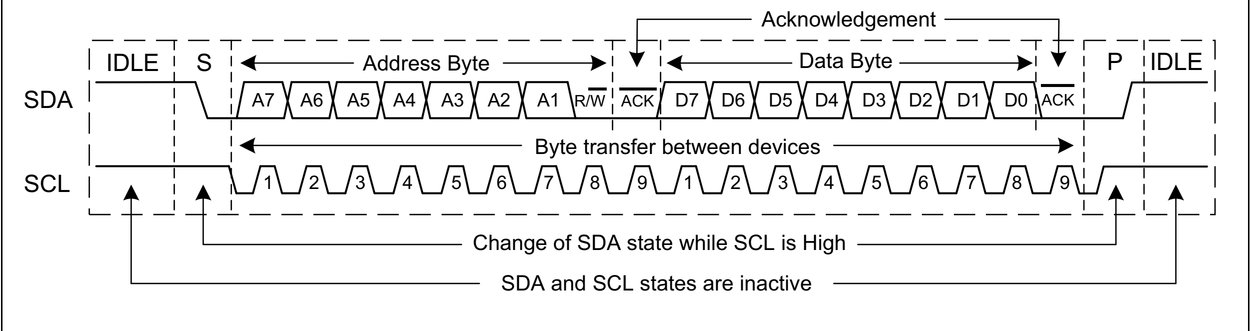

VDD
R1
R2
SDA
SCL
MASTER 1
MASTER 2
SLAVE 1
SLAVE 2
(a) Single Master and Multiple Slave Connection
(c) Multi-Master and Single Slave Connection
(b) Single Master and Multi-Slave Connection
(d) Multi-Master and Multi- Slave Connection

Bus Lines

The I^{2}C bus is comprised of the Serial Clock (SCL) and Serial Data (SDA) lines. The clock signal generated by a master device is sent through the SCL line to control the data transmission. The SDA line transfers the data sent/received by the master/slave. Each signal line requires a pull-up resistor for an open-drain connection. Both lines will initially float high when the bus is Idle and must be driven low in a specific order to initiate communication between I^{2}C devices. The transition of the SDA line is always performed while the SCL line is being held low. Transitions that occur while the SCL line is held high are used to indicate Start and Stop conditions.

Figure 3 shows the states of the SDA and SCL signals in a typical I^{2}C message. The figure also shows the basic components of an I^{2}C message: the Start and Stop conditions, address and data bytes, and the Acknowledgment.

FIGURE 3: TYPICAL I^{2}C MESSAGE

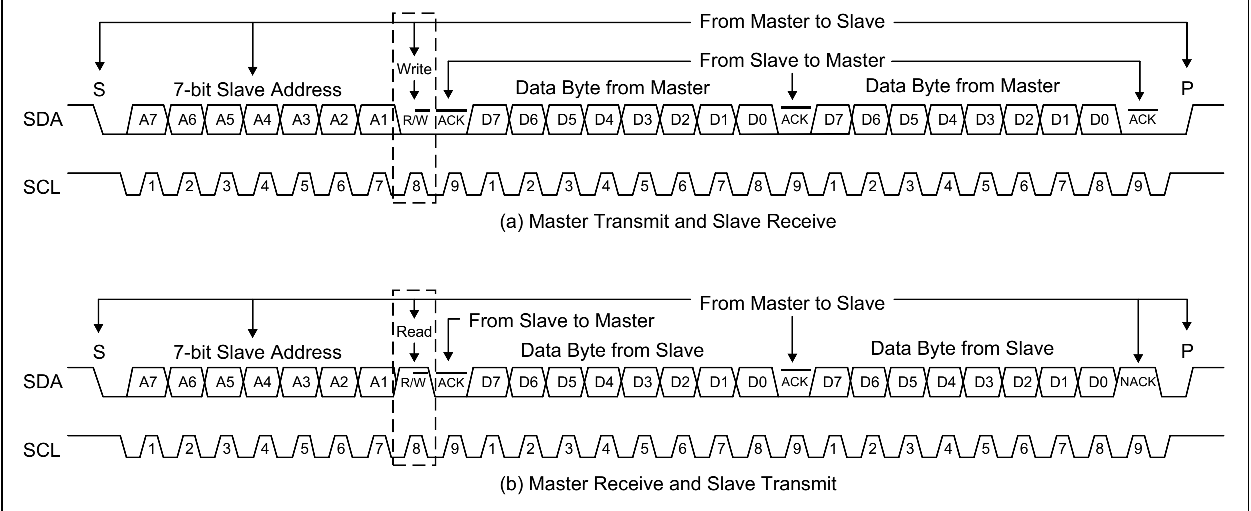

Acknowledgement
IDLE
S
Address Byte
Data Byte
P
IDLE
SDA
A7 A6 A5 A4 A3 A2 A1 R/W ACK D7 D6 D5 D4 D3 D2 D1 D0 ACK
Byte transfer between devices
SCL
1 2 3 4 5 6 7 8 9 1 2 3 4 5 6 7 8 9
Change of SDA state while SCL is High
SDA and SCL states are inactive

I^{2}C Operations

An I^{2}C supported device can operate in Master Transmit/Receive or Slave Transmit/Receive modes. The transmission between the Start and Stop conditions is sent in 9-bit segments, consisting of an address or data byte, followed by an ACK/NACK condition.

Figure 4 shows a simple I^{2}C transmission of two data bytes in transmit and receive operation. The main difference between (a) and (b) is the state of the R/$\overline{W}$ bit. This bit determines the operation in the addressing stage of the transmission.

FIGURE 4: I^{2}C COMMUNICATION BETWEEN A MASTER AND A 7-BIT ADDRESSED SLAVE DEVICE


From Master to Slave
From Slave to Master
Write
S
7-bit Slave Address
Data Byte from Master
Data Byte from Master
P
SDA
A7 A6 A5 A4 A3 A2 A1 R/W ACK D7 D6 D5 D4 D3 D2 D1 D0 ACK D7 D6 D5 D4 D3 D2 D1 D0 ACK
SCL
(a) Master Transmit and Slave Receive
From Master to Slave
Read
From Slave to Master
S
7-bit Slave Address
Data Byte from Slave
Data Byte from Slave
P
SDA
A7 A6 A5 A4 A3 A2 A1 R/W ACK D7 D6 D5 D4 D3 D2 D1 D0 ACK D7 D6 D5 D4 D3 D2 D1 D0 NACK
SCL
(b) Master Receive and Slave Transmit

I^{2}C MODULE FEATURES

Listed below are the features provided by the I^{2}C module.

- Master Mode
- Slave Mode with Byte NACKing
- Multi-Master Mode
- I^{2}C Master and Slave Hardware Support
  - Dedicated Address, Receive and Transmit Buffers
  - Up to Four Slave Addresses Matching[(1)]
  - General Call Address Matching
  - 7-Bit and 10-Bit Addressing with Masking
  - Start, Restart, Stop, Address, Read, Write, ACK and Byte Count Interrupts
  - Bus Time-out, Bus Collision and NACK Detect Error Interrupts
- Hardware-Based Clock Stretching for:
  - RX Buffer Full
  - TX Buffer Empty
  - After Address, Write and ACK
- Auto-Byte Count
- Selectable Clock
- Bus Collision Detection
- Bus Time-out Detection with Programmable Sources
- SDA Hold Time Selection
- Programmable Bus-Free Time Selection
- I^{2}C, SMBus and 1.8V Input Level Selections
- DMA[(2)] and PMD Support

Note 1: Support for four slave addresses matching is only available in 7-bit addressing. Address matching for 10-bit addressing is only up to two addresses.

2: The availability of the Direct Memory Access (DMA) module is device-specific. Refer to the data sheet to check if these features are present in the device.

For SMBus and PMBus™ compatibility, bus time-out allows the module to be reset by a selected time-out in the I2CxBTO register. The module can also be disabled through PMD when not in use. Automation of data handling in the master and slave modules reduces code overhead and enhances the overall performance of the module.

Multi-Master Mode

For the multi-master support feature, the Bus-Free (BFRE) status bit allows the master to determine if the bus is free before it asserts the Start condition. The master waits for a bus-free time of 8 to 64 I^{2}C clock pulses (BFRET<1:0>) before setting the BFRE bit. This prevents the master from attempting to take control of the bus while another master is using it. In cases where both masters assert a Start at the same time, bus arbitration will take place at the addressing phase.

I^{2}C MODULE OPERATION

This section discusses how the I^{2}C module operates in accordance with its features. For a summary of registers and bits associated with the module, see Table A-1. For a more detailed discussion on these registers, refer to the device data sheet.

Setting Up the I^{2}C Bus

The SCL and SDA lines are digital, open-drain and bidirectional. I/O pins assigned to these signals must be configured as such through the ANSELx and ODCONx registers, and PPS, respectively. The PPS enables these pins to function as peripheral inputs (I2CxSCLPPS and I2CxSDAPPS) and outputs (RxyPPS). Input threshold, slew rate and internal pull-up settings are configured in the RxyI2C Control registers. The SCL clock is configured through I^{2}C clock selection, while the SDA hold time can be set to 30/100/300 ns in the SDAHT bits of the I2CxCON2 register. Example 1 shows a sample configuration of the SDA and SCL pins on a K42 device.

EXAMPLE 1: CONFIGURING SCL AND SDA PINS

```
// Configure the pins as digital
ANSELCbits.ANSELC3 = 0;
ANSELCbits.ANSELC4 = 0;

// PPS Unlock Sequence
PPSLOCK = 0x55;
PPSLOCK = 0xAA;
PPSLOCKbits.PPSLOCKED = 0x00;

// Set RC4 for SDA
RC4PPS = 0x22;
I2C1SDAPPS = 0x14;

// Set RC3 for SCL
RC3PPS = 0x21;
I2C1SCLPPS = 0x13;

// PPS Lock Sequence
PPSLOCK = 0x55;
PPSLOCK = 0xAA;
PPSLOCKbits.PPSLOCKED = 0x01;

// Configure the pins as Open-drain
ODCONCbits.ODCC3 = 1;
ODCONCbits.ODCC4 = 1;

// Set the I2C levels
RC3I2Cbits.TH = 1;
RC4I2Cbits.TH = 1;

// Configure the pins as Outputs
TRISCbits.TRISC3 = 0;
TRISCbits.TRISC4 = 0;
```

Clock Selection

The Clock Source (I2CxCLK) register configures the I^{2}C clock for the master device. This register presents a selection of clock sources provided by the internal CPU oscillator and other peripherals. The Fast Mode Enable (FMEN) bit provides an additional configuration by controlling the sampling of the SCL pin before being driven by the hardware. Enabling or disabling Fast mode sets the SCL frequency to FCLK/4 or FCLK/5, respectively.

Mode Selection

The I^{2}C module has a selection of I^{2}C modes defined by the type of addressing. These modes are defined by the Mode Select bits in the I2CxCON0 Control register. The selection includes options for 7-Bit and 10-Bit Master modes, 7-Bit and 10-Bit Slave modes, and 7-Bit Multi-Master modes.

Conditions

The I^{2}C module observes a number of conditions which indicate different events on the bus. Refer to Figure 4 and Figure 5 to see examples of these conditions on the SDA line.

1. Start Condition (S):

   A master device generates the Start condition to indicate the start of a transfer, changing the state of the bus from Idle to Active. The Start condition is a transition of the SDA line from high-to-low while the SCL line is high.

2. Stop Condition (P):

   The master also generates a Stop condition to release control of the bus when the transmission has ended and returns the bus to its Idle state. The SDA line state changes from low-to-high while the SCL line is high.

3. Restart Condition (RS):

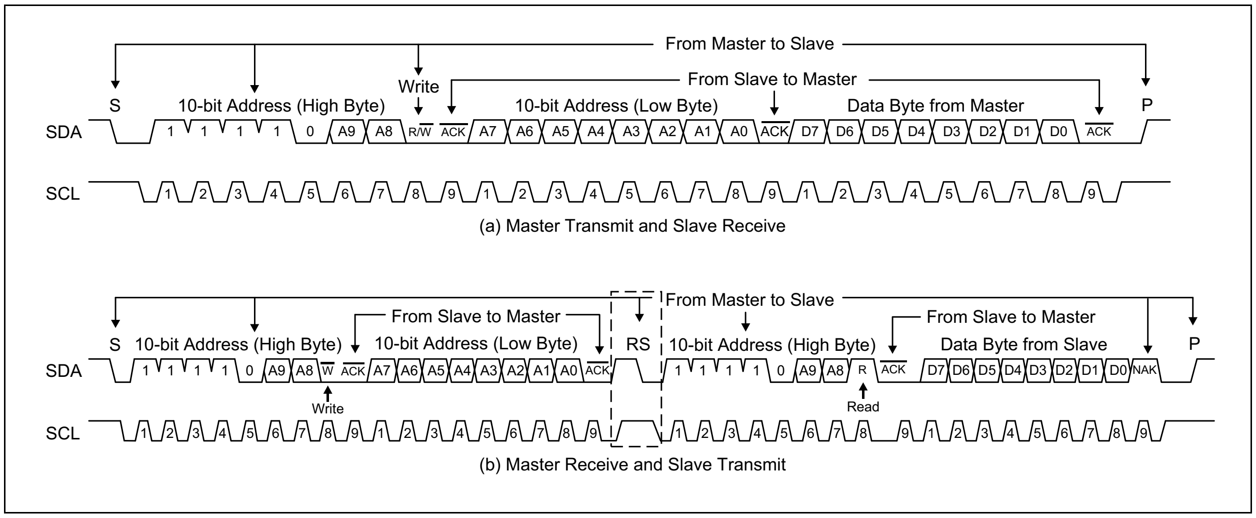
   If the master still needs to control the bus after a transmission, it issues a Restart condition. This prevents other master devices from taking control over the bus between transfers.

4. ACK/NACK Condition:

   The ACK condition Acknowledges the control of a slave device or the success of a data transfer. This condition is sent by the receiver of the transmission by pulling the SDA line low on the 9th clock pulse of the SCL. Otherwise, the receiver will let the SDA line float high, which indicates a NACK condition. A master device also sends a NACK to terminate data reception from the slave.

The Start, Stop and Restart conditions are always generated by the master. The master sends out a Start condition either by setting the Start bit or writing to the transmit buffer while the bus is still Idle. After a Start condition occurs, the master module is now active. The Restart condition is enabled by setting the Restart Enable (RSEN) bit. Once the master module is active and the Start (S) bit is set, the master asserts a Restart if either the byte count is zero or an ACK was not received. A Stop condition occurs either by receiving or sending out a NACK, or when the byte count reaches zero. Interrupt flags are assigned to each of these conditions in the local I^{2}C Peripheral Interrupt Flag register.

FIGURE 5: I^{2}C COMMUNICATION BETWEEN A MASTER AND A 10-BIT ADDRESSED SLAVE DEVICE


From Master to Slave
From Slave to Master
S
Write
10-bit Address (High Byte)
10-bit Address (Low Byte)
Data Byte from Master
P
SDA
SCL
1 1 1 1 0 A9 A8 R/W ACK A7 A6 A5 A4 A3 A2 A1 A0 ACK D7 D6 D5 D4 D3 D2 D1 D0 ACK
(a) Master Transmit and Slave Receive
RS
Data Byte from Slave
W
R
Read
NAK
(b) Master Receive and Slave Transmit

Acknowledge Sequence

The ACK/NACK sequence result is stored in the ACK Status (ACKSTAT) bit. The 9th bit always has the ACK/NACK information. An ACK condition clears the status bit while a NACK sets the bit high. In receive operation, the Data Byte Count register (I2CxCNT), Acknowledge Data bit (ACKDT) and End-of-Count bit (ACKCNT) determine the result of the sequence. If the count still hasn't reached '0', the receiver sends out the ACKDT bit, otherwise, it sends ACKCNT.

In Slave mode, if the Hold Enable bits are set and an address match occurs, clock stretching is initiated. This allows the user to set the ACK value sent back to the transmitter. Other conditions also result in a NACK, such as transmit and receive buffer errors. The master hardware automatically sends a Stop upon detection of a NACK.

Addressing

A Slave device can either have a 7-bit or 10-bit address. In this module, there can be four 7-bit addresses and two 10-bit addresses. After a Start condition, the master always sends out the address byte/s first. The address byte/s holds the address of the slave that the master is attempting to communicate with and the Read/Write ($R/\overline{W}$) bit.

1. 7-Bit Addressing:

   The master sends a single address byte of which the slave address occupies bits 7 to 1. Bit 0 of the address byte holds the $R/\overline{W}$ bit. (See Figure 4.)

2. 10-Bit Addressing:

   The first byte consists of the combination, '11110xx', and the $R/\overline{W}$ bit. 'xx' are the two Most Significant bits (MSbs) of the 10-bit address. The second byte contains the remaining eight bits of the address. (See Figure 5.)

The $R/\overline{W}$ bit determines the direction of the data. If the master wishes to transmit data, the SDA line floats high. For data reception, the SDA line is pulled low. If the slave device is present on the bus, it responds with an ACK. However, if the master fails to connect to the slave, a NACK is returned.

10-bit addressing in master receive and slave transmit operation requires a Restart condition for clocking out data from the slave. The master first sends out the two address bytes for a write operation. If the slave device Acknowledges both address bytes, the master software sets the Start bit for a Restart condition. The master will resend the high address byte for a read operation, wait for an ACK condition and then read out the data from the slave.

The module provides two Address Data Buffers (I2CxADBx) and four Address (I2CxADRx) registers. The Address Data Buffers act as either transmit or receive buffers (see Table 1). In Master modes, the buffers contain the address byte/s to be shifted out to the bus. In Slave modes, the buffers hold the received address byte/s and match the address with the Address registers. The Address registers contain the Slave mode addresses used for matching and masking in Slave mode. The multiple Address registers are required to support SMBus and PMBus communication.

TABLE 1: DIRECTION OF ADDRESS DATA BUFFERS FOR DIFFERENT I^2C MODES

Modes	I2CxADB0	I2CxADB1
Slave (7-bit, with or without masking)	RX	—
Slave (10-bit, with or without masking)	RX	RX
Master (7-bit)	—	TX
Master (10-bit)	TX	TX
Multi-Master (7-bit)	RX	TX

General Call Address Support

The I^2C module supports general call addressing in which a master can address all slave devices and receives an ACK. The general call address is defined as '0x00' and requires the software to enable the General Call Address Enable bit. It is not required to define any of the Address registers with the general call address.

Data Transfer

Communication continues with sending out the data byte/s. If the data byte is properly sent or received, an ACK bit is returned. The I^2C bus is released after a Stop condition is detected and returns to its Idle state until the next Start condition.

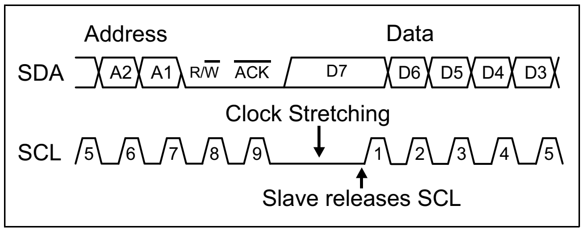
Dedicated Buffers

The I^2C module has two data buffers, one for transmission (TXB) and another for reception (RXB). These buffers can be accessed by software or DMA. The I^2C module requires the user to monitor the state of these buffers through their status (TXBE/RXBF) or interrupt (TXIF/RXIF) flags to ensure proper read and write operation. The user must not load a new byte to the TXB if the buffer is full. Data must not be read from the RXB if it is empty or if it holds old data. Incorrect read and write operations will cause error condition flags to set.

Clock Stretching

If the slave device cannot keep up with the master's data rate, it can delay the transmission by using clock stretching. The slave has the ability to hold the SCL line low at any point of the transfer until it is ready to continue the communication (see Figure 6). When the master detects that the SCL line is held low, it will wait until the slave releases the line to transmit or receive the remaining data.

FIGURE 6: CLOCK STRETCHING



Clock stretching occurs when the Transmit Buffer Empty (TXBE) bit or Receive Buffer Full (RXBF) bit is set and the Count register is not zero. Setting the dedicated Interrupt and Hold bits for address match (ADRIE), data write (WRIE) and Acknowledge status (ACKTIE) also enables clock stretching for these operations. Clock stretching can be disabled by setting the Clock Stretching Disable (CSTRDIS) bit.

Auto-Byte Count

The Byte Count (I2CxCNT) register is a hardware counter which controls the length of the data transfer for the I^{2}C bus. The register holds the number of data bytes to be sent or received, excluding the address bytes. This register automatically decrements for every set of byte+ACK on the bus, on the 9th falling SCL edge of each transmitted byte. Once the register decrements to '0', the Count Interrupt Flag (CNTIF) is set and master hardware will issue a Stop condition. If the number of bytes exceeds 256, the register can be rewritten midway.

Interrupts, Status and Error Detection

The I^{2}C module has multiple ways to monitor the state of its operation. Users can either observe the Status registers (I2CxSTAT0/1) or enable the relevant interrupt sources. Most status bits are read-only and show the current state of the I^{2}C bus and module (Active Master or Active Slave).

The local I^{2}C interrupts contain (I2CxPIR) flags for the I^{2}C bus conditions, clock stretching options and the Byte Count register. Setting any of the flags will set the main I^{2}C Interrupt Flag (I2CxIF).

The I^{2}C Error (I2CxERR) register contains interrupts that are set as a result of a communication error. These errors include bus time-out, bus collision and NACK detection. Detection of any of these errors will also set the main I^{2}C Error Interrupt Flag (I2CxEIF).

The transmit and receive buffers also have their own interrupt enable and flag bits. If these buffer empty (TXIF) and full (RXIF) interrupts are enabled, these will require servicing before the next transmission. For a vectored Interrupt Controller, Interrupt Priority Levels (IPLs) can also be set for these interrupt sources. For a complete list of interrupts, refer to the device data sheet.

SAMPLE I^{2}C MODULE APPLICATION

Master Mode

Example 2 shows the software configuration of the I^{2}C module as an I^{2}C master device. With the 500 kHz clock source and the fast mode enabled, the module communicates on an SCL frequency of 125 kHz. Example 3 and Example 4 show a simple master write and read operation to an I^{2}C slave EEPROM. The address buffer `I2C1ADB1` holds the 7-bit slave address, `EE_SLAVE_ADDRESS`. The Count register `I2C1CNT` holds the total number of bytes to be sent or received.

EXAMPLE 2: I^{2}C MASTER MODE INITIALIZATION

```
// 7bit Master Mode (MODE = 4)
I2C1CON0 = 0x04;

// I2C Clock = MFINTOSC (500kHz)
I2C1CLK = 0x03;

// ACK for every valid byte (ACKDT = 0)
// NACKs to end a Read (ACKCNT = 1)
I2C1CON1 = 0x80;

// Auto-count disabled (ACNT = 0)
// General Call disabled (GCEN = 0)
// Fast mode enabled
// (FME = 1; SCL = I2CCLK/4)
// ADB1 address buffer used (ADB = 0)
// SDA Hold time of 300 ns (SDAHT = 0)
// Bus free time of 16 I2C Clock pulses
// (BFRET = 1)
I2C1CON2 = 0x21;

// Enable I2C module
I2C1CON0bits.EN = 1;
```

Since the I^{2}C bus is still Idle at this point, a write to the transmit buffer initiates the transmission. The module asserts the Start condition and shifts out the slave address with the write bit on the SDA line. After an ACK is received, the module shifts out the first byte of data (EEPROM memory address byte) pointed to by the data pointer. After loading the final byte into the buffer, the module will still take some time to shift the data out and send a stop.

EXAMPLE 3: I^{2}C MASTER WRITE TO AN EEPROM

```
// Write operation (W/R bit = 0)
I2C1ADB1 = (EE_SLAVE_ADDRESS<<1)|0;
totalByteLength = EE_ADDR_BYTE_LENGTH
                +DATA_BYTE_LENGTH;
I2C1CNT = totalByteLength;

for(i = 0; i < totalByteLength; i++){
    // Write to TXB starts communication
    I2C1TXB = *dataBlock;
    dataBlock++;

    // Wait until TXB is empty
    while(!I2C1STAT1bits.TXBE);
}
```

A typical read from an EEPROM comprises of a master write operation for the EEPROM memory address, and a master read operation for the data bytes. After the transmission of the memory address bytes, `I2C1ADB1` and `I2C1CNT` are rewritten for a read operation. The module asserts another Start condition by setting the start bit, and shifts out `EE_SLAVE_ADDRESS` with the read bit. After the slave returns an ACK, the master shifts in the first data byte and stores it in the receive buffer. The master must read the receive buffer through software to clear `RXBF` for the next read. After the final byte is read, the master sends a NACK as configured in Example 2.

EXAMPLE 4: I^{2}C MASTER READ FROM AN EEPROM

```
// Read operation (W/R bit = 1)
I2C1ADB1 = (EE_SLAVE_ADDRESS<<1)|1;
I2C1CNT = READ_BYTE_LENGTH;
I2C1CON0bits.S = 1;

for(i = 0; i < READ_BYTE_LENGTH; i++){
    while (!I2C1STAT1bits.RXBF);
    *readBlock = I2C1RXB;
    readBlock++;
}
```

Slave Mode

Example 5 shows the configuration of the I^{2}C module as slave device with up to four 7-bit addresses. Address registers may hold similar or completely different addresses. If a master device requests for any of the 4 addresses, the slave module will respond with an ACK.

Since an I^{2}C slave waits for an I^{2}C master to initiate the communication, the slave is set to use interrupts and status bits to detect whether data is being sent or received. Interrupts also allow the slave to control its next action through software. These interrupts are enabled before enabling the module to ensure that any immediate communication from an I^{2}C master is serviced in the interrupt routines.

EXAMPLE 5: I^{2}C SLAVE MODE INITIALIZATION

```
// 7bit Slave Mode (MODE = 0)
I2C1CON0 = 0x00;

// Slave Address Match
I2C1ADR0 = 0x98;
I2C1ADR1 = 0x98;
I2C1ADR2 = 0x98;
I2C1ADR3 = 0x98;

// ACK for every valid byte (ACKDT = 0)
// ACK at the end of a Read (ACKCNT = 0)
// Clock stretching enabled (CSTRDIS = 0)
I2C1CON1 = 0x00;

// Auto-count disabled (ACNT = 0)
// General Call disabled (GCEN = 0)
// Fast mode enabled (FME = 1)
// ADB0 address buffer used (ADB = 0)
// SDA Hold time of 30 ns (SDAHT = 2)
// Bus free time of 8 I2C Clock pulses
// (BFRET = 1)
I2C1CON2 = 0x28;

// Clear all I2C flags
PIR3bits.I2C1F = 0;
I2C1PIR = 0x00;

// Enable Global and I2C interrupts
INTCON0bits.IPEN = 1;
INTCON0bits.GIEH = 1;
PIE3bits.I2C1IE = 1;
PIE3bits.I2C1IP = 1;

// Enable local interrupt on ACK Sequence
I2C1PIE = 0x40;

// Enable I2C module
I2C1CON0bits.EN = 1;
```

There are multiple interrupt options for an I^{2}C slave module interrupt service routine. In Example 5, the ACK interrupt hold is enabled. If the interrupt event is triggered, the slave software is allowed to access the data buffers while clock stretching is in progress (see Example 6). Similarly, these actions can also be done in the address hold, data hold, buffer full and empty interrupts. For the hold interrupts, the `ACKDT` bit can also be modified, and clearing `CSTR` releases the clock. For the buffer interrupts, a buffer read or write releases SCL and resumes communication.

EXAMPLE 6: I^{2}C SLAVE INTERRUPT SERVICE

```
void interrupt I2CSLAVE_ISR (void){
    if (PIR3bits.I2C1IF){
        // Clear the I2C interrupt flag
        PIR3bits.I2C1IF = 0;

        // ACK Sequence Interrupt Detected
        if (I2C1PIRbits.ACKTIF){
            // Clear the interrupt flag
            I2C1PIRbits.ACKTIF = 0;

            // For Slave Read/Master Write
            if (!I2C1STAT0bits.R){
                // Data Byte Received
                if (I2C1STAT0bits.D){
                    // Read from RXB
                    *readBlock = I2C1RXB;
                    readBlock++;
                }
            }
            // For Slave Write/Master Read
            else {
                // Write to TXB
                I2C1TXB = *dataBlock;
                dataBlock++;
            }
            // Release SCL
            I2C1CON0bits.CSTR = 0;
        }
     }else {
        // Other Interrupts…
    }
}
```

CONCLUSION

The I^2C module is a two-wire synchronous and bidirectional interface, which supports communication between multiple master and slave devices. Hardware protocol acceleration, through the additional hardware support features, enables the module to operate with minimal software overhead.

Additional hardware support allows the module to function with only one interrupt per data byte. With the addition of DMA, the module only requires a single CPU interrupt per message to be handled.

APPENDIX A: REGISTERS AND BITS ASSOCIATED WITH THE I^{2}C MODULE

TABLE A-1: SUMMARY OF I^{2}C MODULE REGISTERS AND BITS

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CxBTO	—	—	—	—	—	BTO<2:0>		
I2CxCLK	—	—	—	—	—	CLK<2:0>		
I2CxPIE	CNTIE	ACKTIE	—	WRIE	ADRIE	PCIE	RSCIE	SCIE
I2CxPIR	CNTIF	ACKTIF	—	WRIF	ADRIF	PCIF	RSCIF	SCIF
I2CxERR	—	BTOIF	BCLIF	NACKIF	—	BTOIE	BCLIE	NACKIE
I2CxSTAT0	BFRE	SMA	MMA	R	D	—	—	—
I2CxSTAT1	TXWE	—	TXBE	—	RXRE	CLRBF	—	RXBF
I2CxCON0	EN	RSEN	S	CSTR	MDR	MODE<2:0>		
I2CxCON1	ACKCNT	ACKDT	ACKSTAT	ACKT	—	RXOV	TXU	CSD
I2CxCON2	ACNT	GCEN	FME	ADB	SDAHT<3:2>		BFRET<1:0>	
I2CxADR0	ADR<7:0>							
I2CxADR1	ADR<7:1>							—
I2CxADR2	ADR<7:0>							
I2CxADR3	ADR<7:1>							—
I2CxADB0	ADB<7:0>							
I2CxADB1	ADB<7:0>							
I2CxCNT	CNT<7:0>							
I2CxRXB	RXB<7:0>							
I2CxTXB	TXB<7:0>							
I2CxSDAPPS	—	—	—	I2CxSDAPPS<4:0>				
I2CxSCLPPS	—	—	—	I2CxSCLPPS<4:0>				
RxyI2C	—	SLEW	PU<1:0>		—	—	TH<1:0>	

ISBN: 978-1-5224-1872-6

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

11/07/16