# MCP2221A

## USB 2.0 to I$^2$C/UART Protocol Converter with GPIO

> **Note:** The MCP2221A is identical to the MCP2221 in all aspects except for the maximum supported baud rate of the UART, which has been increased from 115200 (MCP2221) to 460800 (MCP2221A).

## Features:

### Universal Serial Bus (USB)

- Supports Full-Speed USB (12 Mb/s)
- Implements USB Protocol Composite Device:
  - Communication Device Class (CDC) for USB-to-UART conversion
  - Human Interface Device (HID) for I$^2$C device control and configuration
- 448-Byte Buffer to Handle Data Throughput at Any Supported UART Baud Rate:
  - 64-byte transmit
  - 384-byte receive
- Human Interface Device (HID) for Both I$^2$C Communication and Control:
  - 64-byte buffer to handle data throughput at any I$^2$C baud rate
- Fully Configurable VID and PID Assignments and String Descriptors
- Bus-Powered or Self-Powered
- USB 2.0-Compliant: TID# 40001594

### USB Driver and Software Support

- Enumerates as a Composite USB Device (CDC and HID) Using Standard Drivers for Virtual Com Port (VCP) on the following Windows® Operating Systems: XP® (SP3), Vista®, 7, 8, 8.1 and 10
- Configuration Utility for Establishing a Custom Boot-up Configuration
- I$^2$C/SMBus Terminal
- Windows DLL

### CDC and Universal Asynchronous Receiver/Transmitter (UART) Options

- Communications Device Class (CDC) for the USB-to-UART Option
- Responds to `SET LINE CODING` Commands to Dynamically Change Baud Rates
- Supports Baud Rates: 300-460800
- UART UTx and URx Pins Only

- Serial Number Used During the CDC Enumeration can be Enabled by Using the Microchip Provided Configuration Utility or by Calling the Proper API from the Support Libraries for this Device

### I$^2$C/SMBus

- The Device Runs as an I$^2$C Host. The Data to Write/Read on the I$^2$C Bus are Conveyed by the USB Interface.
- I$^2$C Host:
  - Up to 400 kHz clock rate
  - Supports 7-bit or 10-bit addressable devices; 10-bit addressable devices are supported through the PC host library
  - Supports block reads/writes of up to 65,535 bytes long
- SMBus Host:
  - Up to 400 kHz clock rate
  - Supports all of the SMBus transfers
  - SMBus functionality is achieved through a combination of chip and support library processing

### General Purpose Input/Output (GPIO) Pins

- Four General Purpose Input/Output Pins
- All GP Pins can be Assigned to Other Functionalities
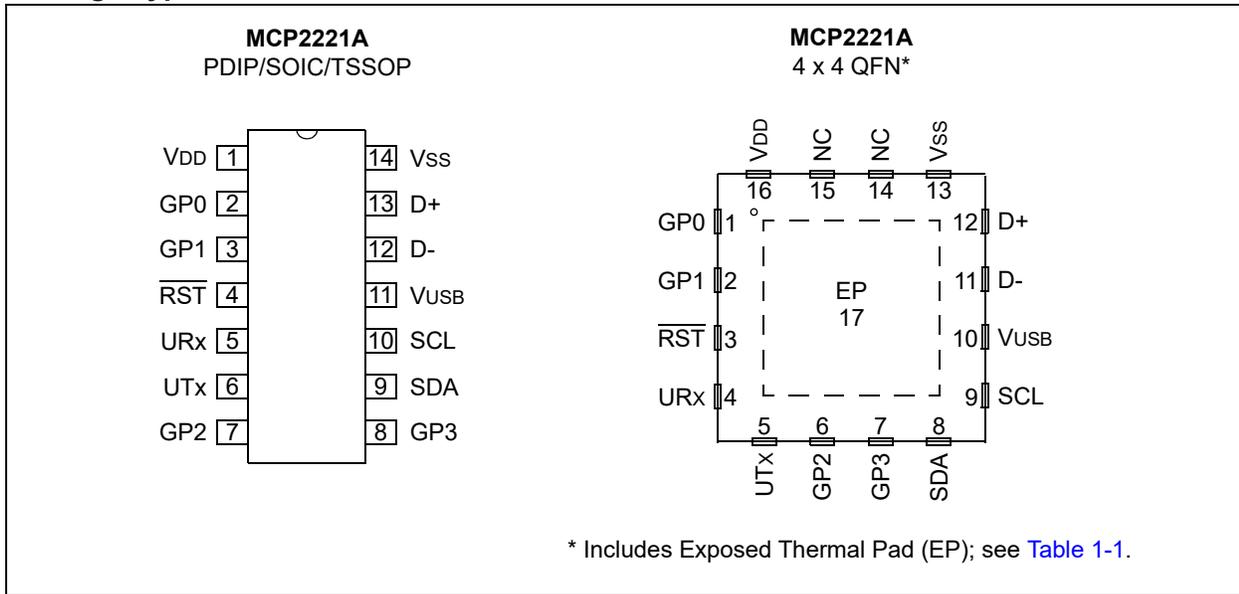
### Other Functionalities

- UART Activity LED Outputs (UTx and URx)
- SSPND Output Pin
- USBCFG Output Pin (indicates when the enumeration is completed)
- Three ADC Inputs
- One DAC with Two Possible Output Options
- Clock Reference Output: 12 MHz or Other Configurable Values
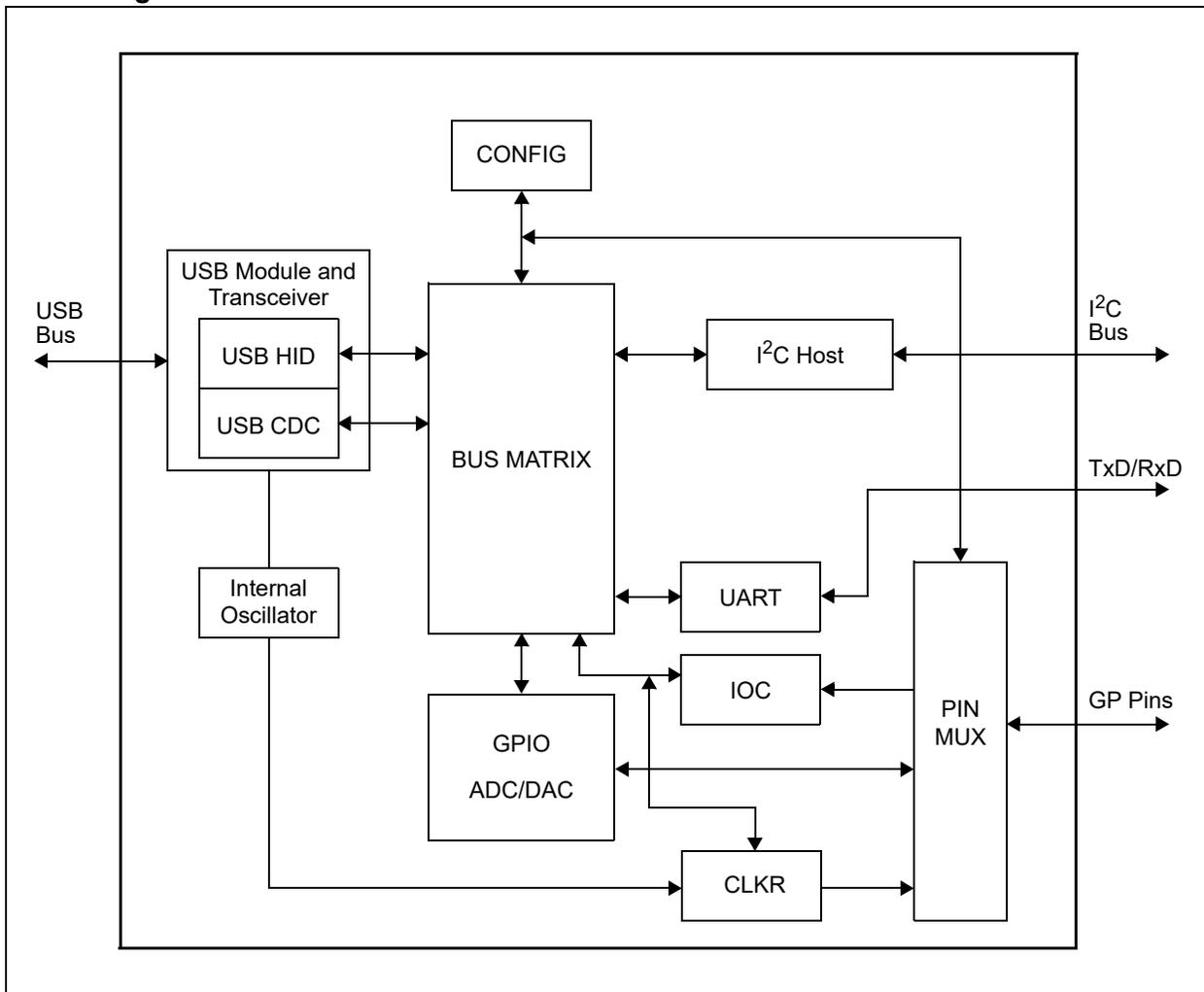- External Interrupt Edge Detection

### Other

- Operating Voltage: 3.0V to 5.5V
- Electrostatic Discharge (ESD) Protection: > 4 kV Human Body Model (HBM)
- Industrial(I) Operating Temperature: -40°C to +85°C
- Automotive AEC-Q100 Qualified

# MCP2221A

## Package Types

**MCP2221A**
PDIP/SOIC/TSSOP

| | | | |
|---|---|---|---|
| VDD | 1 | 14 | VSS |
| GP0 | 2 | 13 | D+ |
| GP1 | 3 | 12 | D- |
| $\overline{RST}$ | 4 | 11 | VUSB |
| URx | 5 | 10 | SCL |
| UTx | 6 | 9 | SDA |
| GP2 | 7 | 8 | GP3 |

**MCP2221A**
4 x 4 QFN*

Top pins: 16 VDD, 15 NC, 14 NC, 13 VSS

| Left | | | Right |
|---|---|---|---|
| GP0 | 1 | 12 | D+ |
| GP1 | 2 | 11 | D- |
| $\overline{RST}$ | 3 | 10 | VUSB |
| URx | 4 | 9 | SCL |

EP 17

Bottom pins: 5 UTx, 6 GP2, 7 GP3, 8 SDA

* Includes Exposed Thermal Pad (EP); see Table 1-1.

## Block Diagram

## 1.0 FUNCTIONAL DESCRIPTION

The MCP2221A is a USB-to-UART serial converter that enables USB connectivity in applications that have UART and/or I$^2$C interfaces. The device reduces external components by integrating the USB termination resistors and the oscillator needed for USB operation.

The MCP2221A has four GP pins for miscellaneous functionalities (including GPIO, USBCFG, SSPND, Clock Output, ADC, DAC and interrupt detector).

See Table 1-1 and **Section 1.7 "Pin MUX Module"** for details about the pin functions.

TABLE 1-1: PINOUT DESCRIPTION

| Pin Name | PDIP, SOIC, SSOP | QFN | Pin Type | Standard Function | Alternate Functions |
|---|---|---|---|---|---|
| GP0 | 2 | 1 | I/O | General purpose I/O or alternate function pin | • SSPND (OUT) – Signals when the host has entered Suspend mode<br>• LED_URx (OUT) – UART Rx LED activity output (factory default) |
| GP1 | 3 | 2 | I/O | General purpose I/O or alternate function pin | • CLKR (OUT) – Clock Reference Output<br>• ADC1 (IN) – ADC Channel 1<br>• LED_UTx (OUT) – UART Tx LED activity output (factory default)<br>• IOC (IN) – External Interrupt Edge Detector |
| RST | 4 | 3 | I | Reset input (with internal pull-up) | N/A |
| URx | 5 | 4 | I | UART Rx pin (input) | N/A |
| UTx | 6 | 5 | O | UART Tx pin (output) | N/A |
| GP2 | 7 | 6 | I/O | General purpose I/O or alternate function pin | • USBCFG (OUT) – USB device-configured status (factory default)<br>• ADC2 (IN) – ADC Channel 2<br>• DAC1 (OUT) – DAC Output 1 |
| GP3 | 8 | 7 | I/O | General purpose I/O or alternate function pin | • LED_I2C (OUT) – USB/I$^2$C traffic indicator (factory default)<br>• ADC3 (IN) – ADC Channel 3<br>• DAC2 (OUT) – DAC Output 2 |
| SDA | 9 | 8 | I/O | I$^2$C Data Line | N/A |
| SCL | 10 | 9 | I/O | I$^2$C Clock Line | N/A |
| V$_{USB}$ | 11 | 10 | USB | USB Power pin (internally connected to 3.3V); should be locally bypassed with a high-quality ceramic capacitor | — |
| D- | 12 | 11 | USB | USB D- | — |
| D+ | 13 | 12 | USB | USB D+ | — |
| V$_{SS}$ | 14 | 13 | P | Ground | — |
| NC | — | 14, 15 | — | Not Connected | — |
| V$_{DD}$ | 1 | 16 | P | Power | — |
| EP | — | 17 | — | Exposed Thermal Pad (EP); do not electrically connect | — |

# MCP2221A

## 1.1 Supported Operating Systems

The following operating systems are supported:

- Windows – XP (SP3), Vista, 7, 8, 8.1 and 10
- Linux® – Any distribution with support for CDC and HID classes
- Mac OS® – All versions beginning with 10.7

### 1.1.1 ENUMERATION

The MCP2221A enumerates as a composite USB device after POR. The device enumerates as both a Human Interface Device (HID) for I$^2$C, GPIO control, and as CDC for the USB-to-UART converter.

#### 1.1.1.1 USB HID

The MCP2221A enumerates as a HID, so the device can be configured, while the I$^2$C and GPIO can be controlled. A DLL package, with example applications and tools, is supplied by Microchip on the device web page, on the Microchip website www.microchip.com.

#### 1.1.1.2 USB CDC

The CDC enumeration implements the USB-to-UART data translation.

## 1.2 Bus Matrix Module

The Bus Matrix module is the heart of the MCP2221A. All other modules are tied together and controlled via the Bus Matrix module. This module manages the data transfers between the USB and the UART, the I$^2$C host module, as well as the command requests generated by the USB host controller, and commands for controlling the function of the UART, GPIO, ADC, DAC and clock output.

### 1.2.1 UART

The control module interfaces to the UART and USB modules.

### 1.2.2 ACCESSING THE DEVICE

The MCP2221A can be accessed for reading and writing via USB host commands. The device cannot be accessed or controlled via the UART interface.

## 1.3 UART Interface

The MCP2221A UART interface consists of the UTx and URx data signals.

The UART is configurable for several baud rates. The available baud rates are listed in Table 1-2.

### 1.3.1 GET/SET LINE CODING

The `GET_LINE_CODING` and `SET_LINE_CODING` commands are used to read and set the UART parameters while in operation. For example, terminal applications (e.g., PuTTY, RealTerm, HyperTerminal®, etc.) send the `SET_LINE_COMMAND` when connecting to the port. The MCP2221A responds by setting the baud rate only.

The other parameters (data bits, parity, Stop bits) remain unchanged.

> **Note:** MCP2221A supports only eight data bits, no parity and one Stop bit.

#### 1.3.1.1 Rounding Errors

Primary baud rate settings (with associated rounding errors) are shown in Table 1-2.

If baud rates other than the ones shown in the table are used, the error percentage can be calculated using Equation 1-1 to find the actual baud rate.

**TABLE 1-2: UART PRIMARY BAUD RATES**

| Desired Rate | Actual rate | % Error |
|---|---|---|
| 300 | 300 | 0.00% |
| 1200 | 1200 | 0.00% |
| 2400 | 2400 | 0.00% |
| 4800 | 4800 | 0.00% |
| 9600 | 9600 | 0.00% |
| 19200 | 19200 | 0.00% |
| 38400 | 38339 | 0.16% |
| 57600 | 57692 | 0.16% |
| 115200 | 115385 | 0.16% |
| 230400 | 230769 | 0.16% |
| 460800 | 461538 | 0.16% |

**EQUATION 1-1: SOLVING FOR ACTUAL BAUD RATE**

$$ActualRate = \frac{12MHz}{int(x)}$$

Where:

$$x = \frac{12MHz}{DesiredBaud}$$

### 1.3.2 CUSTOM BAUD RATES

Custom baud rates are configured by sending the `SET_LINE_CODING` USB command. See **Section 2.0 "USB Enumeration Process"** for more information.

## 1.4 Device Configuration

The MCP2221A keeps all the essential device configuration settings stored in Flash memory.

Device configuration settings affect the way the MCP2221A behaves at run time.

The settings are stored into the Flash memory on the device. Some of the settings are also copied into SRAM at power-up/Reset.

These device configuration settings reside in the following two distinct areas of Flash memory:

- **Chip Settings**

  The Chip settings area stores the key MCP2221A parameters – USB parameters, ADC/DAC reference voltage choice, start-up DAC value, Clock Reference (CLKR) output frequency and duty cycle values.

- **GP Settings**

  The GP settings area stores the GP designation settings. For GP settings that are assigned to GPIO output operation, output values (logic '1' or '0') are also specified.

Even though the MCP2221A places a partial copy of the Chip settings in SRAM, the following Chip settings always reside in Flash:

- USB Manufacturer/Product and Serial Number Descriptors
- USB VID and PID Pair
- USB Options (e.g., the requested amount of current that is presented to the USB host during the USB enumeration process)

### 1.4.1 POWER-UP/RESET DEVICE CONFIGURATION BEHAVIOR

At power-up/Reset, the MCP2221A configures the device options (GP designation, special function pins' parameters and USB enumeration options) according to the Flash settings. Then, the Flash Chip settings and GP settings are loaded into SRAM to allow for their temporary modification at run time.

Chip settings of the device configuration Flash are copied partially into SRAM. Only the run-time modifiable parameters are copied into SRAM.

GP settings of the device configuration Flash (GP settings area) are copied entirely into the SRAM. By copying the GP settings completely into SRAM, the user is allowed to completely change the GP designation at run time.

The SRAM copy of the settings can be altered at run time in order to change certain device behavior, e.g., GP designation (the GPs can be reassigned for a different type of operation than the one assigned at power-up) and special parameters (DAC value, ADC/DAC voltage references, clock output value).

**FIGURE 1-1: CHIP SETTINGS RUN-TIME MANAGEMENT**
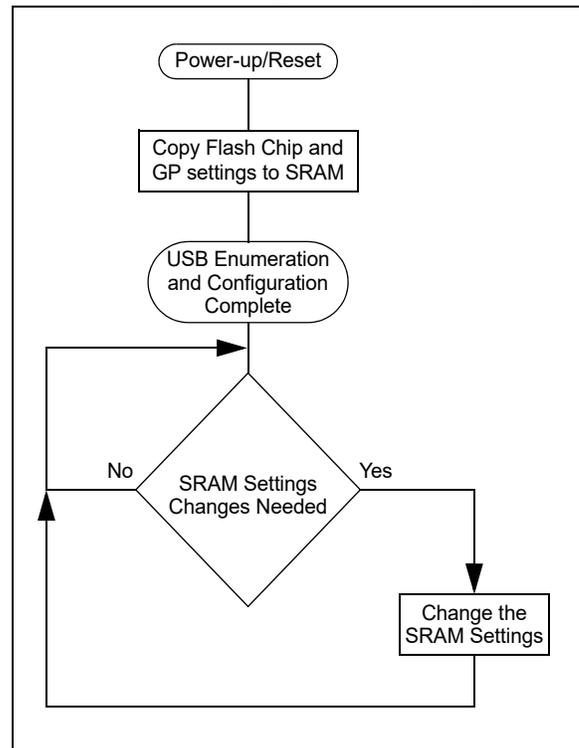


The SRAM settings (GP and partial Chip settings) can be modified through USB HID commands and they will have an effect on the following device features:

- GP pin designation (switch between GPIO, Dedicated or Special Functions modes)
- GPIO direction and output value (only for GPIO outputs) – for the GPs assigned to work in GPIO mode
- Clock output duty cycle and value – if GP1 is assigned for CLKR mode (Clock Reference Output mode), by modifying the SRAM settings, the clock frequency and duty cycle can be changed at run time
- DAC value and voltage reference used – the DAC value setting, as well as the voltage reference used for it, are stored in SRAM settings and they can be changed at run time. Through this mechanism, at run time, the user can change the DAC value, as well as the voltage reference.
- ADC voltage reference value – the voltage reference used for ADC conversions can be changed by altering its corresponding SRAM setting
- Interrupt-on-Change (IOC) detector settings – if GP1 is assigned for IOC mode, the SRAM settings are used for setting up the triggers used for external interrupt detection (positive, negative edge detection or both)

# MCP2221A

## 1.4.2 CHIP SETTINGS MAP

The Chip settings area resides in Flash memory and is copied into SRAM at run time. Not all of the device's settings can be altered at run time. All the fields in the Flash settings can be altered by the user.

**TABLE 1-3: CHIP SETTINGS MAP**

| Byte Index | Register Name | Comments |
|---|---|---|
| 0 | CHIPSETTING0 | Controls the USB CDC serial number enumeration, default state for the GP LED designation, default state for GP dedicated function pins and Chip settings protection level |
| 1 | CHIPSETTING1 | Default clock output divider and duty cycle |
| 2 | CHIPSETTING2 | DAC reference options and default DAC value |
| 3 | CHIPSETTING3 | ADC reference and interrupt detection settings |
| 4 | USBVIDL | USB VID lower byte |
| 5 | USBVIDH | USB VID higher byte |
| 6 | USBPIDL | USB PID lower value |
| 7 | USBPIDH | USB PID higher byte |
| 8 | USBPWRATTR | USB power attributes |
| 9 | USBREQCRT | USB required current |
| 10 | PASS0 | Password Byte 0 |
| 11 | PASS1 | Password Byte 1 |
| 12 | PASS2 | Password Byte 2 |
| 13 | PASS3 | Password Byte 3 |
| 14 | PASS4 | Password Byte 4 |
| 15 | PASS5 | Password Byte 5 |
| 16 | PASS6 | Password Byte 6 |
| 17 | PASS7 | Password Byte 7 |

## REGISTER 1-1: CHIPSETTING0 REGISTER

| R/W-0 | r-1 | r-1 | r-1 | r-1 | r-1 | R/W-0 | R/W-0 |
|-------|-----|-----|-----|-----|-----|-------|-------|
| CDCSNEN | — | — | — | — | — | CHIPPROT1 | CHIPPROT0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | r = Reserved bit | | |
|---------|------------------|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7      **CDCSNEN:** USB CDC Serial Number Enable bit

         1 = USB CDC serial number is enumerated
         0 = No USB CDC serial number enumeration (factory default)

bit 6-2      **Reserved:** Set to '1'

bit 1-0      **CHIPPROT[1:0]:** Chip Settings Protection Level bits

         11 = Reserved
         10 = Permanently locked
         01 = Password protection
         00 = Chip settings unprotected (factory default)

## REGISTER 1-2: CHIPSETTING1 REGISTER

| r-0 | r-0 | r-0 | R/W-1 | R/W-0 | R/W-0 | R/W-1 | R/W-0 |
|-----|-----|-----|-------|-------|-------|-------|-------|
| — | — | — | CLKDC1 | CLKDC0 | CLKDIV2 | CLKDIV1 | CLKDIV0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | r = Reserved bit | | |
|---------|------------------|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-5      **Reserved:** Set to '0'

bit 4-3      **CLKDC[1:0]:** Clock Out Duty Cycle bits

         11 = Duty cycle 75% (75% of one clock period is logic '1' and 25% of one clock period is logic '0')
         10 = Duty cycle 50% (50% of one clock period is logic '1' and 50% of one clock period is logic '0') (factory default)
         01 = Duty cycle 25% (25% of one clock period is logic '1' and 75% of one clock period is logic '0')
         00 = Duty cycle 0% (100% of one clock period is logic '0')

bit 2-0      **CLKDIV[2:0]:** Clock Out Divider Output bits

         111 = 375 kHz clock output
         110 = 750 kHz clock output
         101 = 1.5 MHz clock output
         100 = 3 MHz clock output
         011 = 6 MHz clock output
         010 = 12 MHz clock output (factory default)
         001 = 24 MHz clock output
         000 = Reserved

## REGISTER 1-3: CHIPSETTING2 REGISTER

| R/W-1 | R/W-0 | R/W-0 | R/W-0 | R/W-1 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| DACVRM1 | DACVRM0 | DACREF | DACVAL4 | DACVAL3 | DACVAL2 | DACVAL1 | DACVAL0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | |
|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-6 **DACVRM[1:0]:** DAC Internal Voltage Reference (DAC VRM) Selection bits
11 = VRM voltage is 4.096V (only if V$_{DD}$ is above this voltage)
10 = VRM voltage is 2.048V (factory default)
01 = VRM voltage is 1.024V
00 = VRM is off

bit 5 **DACREF:** DAC Reference Output Selection bit
1 = DAC reference output is DAC VRM voltage selection
0 = DAC reference output is V$_{DD}$ (factory default)

bit 4-0 **DACVAL[4:0]:** Initial DAC Output Value bits
5-bit value for the DAC output (factory default is eight decimal)

## REGISTER 1-4: CHIPSETTING3 REGISTER

| r-0 | R/W-1 | R/W-1 | R/W-0 | R/W-1 | R/W-1 | r-0 | r-0 |
|---|---|---|---|---|---|---|---|
| — | INTDETFEEN | INTDETREEN | ADCVRM1 | ADCVRM0 | ADCREF | — | — |
| bit 7 | | | | | | | bit 0 |

| Legend: | r = Reserved bit | |
|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7 **Reserved:** Set to '0'

bit 6 **INTDETFEEN:** Interrupt Falling Edge Detect Enable bit
1 = Interrupt detector will trigger when a falling edge is detected (factory default)
0 = Falling edges will not trigger the detector

bit 5 **INTDETREEN:** Interrupt Rising Edge Detect Enable bit
1 = Interrupt detector will trigger when a rising edge is detected (factory default)
0 = Rising edges will not trigger the detector

bit 4-3 **ADCVRM[1:0]:** ADC Internal Voltage Reference (ADC VRM) Selection bits
11 = VRM voltage is 4.096V (only if V$_{DD}$ is above this voltage)
10 = VRM voltage is 2.048V
01 = VRM voltage is 1.024V (factory default)
00 = VRM is off

bit 2 **ADCREF:** ADC Reference Output Selection bit
1 = ADC reference output is ADC VRM voltage selection (factory default)
0 = ADC reference output is V$_{DD}$

bit 1-0 **Reserved:** Set to '0'

**REGISTER 1-5: USBVIDL REGISTER**

| R/W-1 | R/W-1 | R/W-0 | R/W-1 | R/W-1 | R/W-0 | R/W-0 | R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| USBVIDL7 | USBVIDL6 | USBVIDL5 | USBVIDL4 | USBVIDL3 | USBVIDL2 | USBVIDL1 | USBVIDL0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | | |
|---------|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-0 **USBVIDL[7:0]:** USB VID Lower Byte (factory default: 0xD8(hex))

**REGISTER 1-6: USBVIDH REGISTER**

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-1 | R/W-0 | R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| USBVIDH7 | USBVIDH6 | USBVIDH5 | USBVIDH4 | USBVIDH3 | USBVIDH2 | USBVIDH1 | USBVIDH0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | | |
|---------|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-0 **USBVIDH[7:0]:** USB VID Higher Byte (factory default: 0x04(hex))

## REGISTER 1-7: USBPIDL REGISTER

| R/W-1 | R/W-1 | R/W-0 | R/W-1 | R/W-1 | R/W-1 | R/W-0 | R/W-1 |
|---|---|---|---|---|---|---|---|
| USBPIDL7 | USBPIDL6 | USBPIDL5 | USBPIDL4 | USBPIDL3 | USBPIDL2 | USBPIDL1 | USBPIDL0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-0 **USBPIDL[7:0]:** USB PID Lower Byte (factory default: 0xDD(hex))

## REGISTER 1-8: USBPIDH REGISTER

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| USBPIDH7 | USBPIDH6 | USBPIDH5 | USBPIDH4 | USBPIDH3 | USBPIDH2 | USBPIDH1 | USBPIDH0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-0 **USBPIDH[7:0]:** USB PID Higher Byte (factory default: 0x00(hex))

## REGISTER 1-9: USBPWRATTR REGISTER

| r-1 | R/W-0 | R/W-0 | r-0 | r-0 | r-0 | r-0 | r-0 |
|---|---|---|---|---|---|---|---|
| — | SELFPWR | REMWKUP | — | — | — | — | — |
| bit 7 | | | | | | | bit 0 |

| Legend: | r = Reserved bit | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7 **Reserved:** Reserved – set to '1' (factory default)

bit 6 **SELFPWR:** USB Self-Powered Attribute bit
1 = Chip will enumerate on the USB bus as being self-powered
0 = Chip will enumerate on the USB bus as being USB bus powered (factory default)

bit 5 **REMWKUP:** USB Remote Wake-up Capability bit
1 = Chip will enumerate on the USB bus as being able to wake up the USB host
0 = Chip will enumerate as not being capable of remote wake-up of the USB host (factory default)

bit 4-0 **Reserved:** Set all bits to '0' (factory default)

## REGISTER 1-10: USBREQCRT REGISTER

| R/W-0 | R/W-0 | R/W-1 | R/W-1 | R/W-0 | R/W-0 | R/W-1 | R/W-0 |
|---|---|---|---|---|---|---|---|
| | | | USBREQCRT[7:0] | | | | |
| bit 7 | | | | | | | bit 0 |

| Legend: | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-0 **USBREQCRT[7:0]:** USB Bus-Powered Required Current Amount bits (in units of 2 mA)

Factory default is 50 (decimal); the USB enumeration interprets this value as a current requirement of 100 mA.

## REGISTER 1-11: PASS0-PASS7 REGISTER

| R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 | R/W-0 |
|---|---|---|---|---|---|---|---|
| PASSx7 | PASSx6 | PASSx5 | PASSx4 | PASSx3 | PASSx2 | PASSx1 | PASSx0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | | |
|---|---|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-0 **PASSx[7:0]:** Password Byte x Value bits (factory default is '0')

## 1.4.3 GP SETTINGS MAP

The GP settings area resides in Flash memory and is copied into SRAM at run time. The user can alter both the Flash and the SRAM GP settings. Any modification in the SRAM copy of the GP settings will have an immediate effect. The GP pins designation changes according to the new content of the SRAM settings. The Flash variant of the settings will affect the power-up behavior of the GP pins.

**TABLE 1-4: GP SETTINGS MAP**

| Byte Index | Register Name | Comments |
|---|---|---|
| 0 | GPSETTING0 | GP0 pin designation and GPIO default output value when GP is set for GPIO output operation |
| 1 | GPSETTING1 | GP1 pin designation and GPIO default output value when GP is set for GPIO output operation |
| 2 | GPSETTING2 | GP2 pin designation and GPIO default output value when GP is set for GPIO output operation |
| 3 | GPSETTING3 | GP3 pin designation and GPIO default output value when GP is set for GPIO output operation |

**REGISTER 1-12: GPSETTING0 REGISTER**

| r-0 | r-0 | r-0 | R/W-1 | R/W-0 | R/W-0 | R/W-1 | R/W-0 |
|---|---|---|---|---|---|---|---|
| — | — | — | GPIOOUTVAL | GPIODIR | GPDES2 | GPDES1 | GPDES0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | |
|---|---|---|
| | r = Reserved bit | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared        x = Bit is unknown |

bit 7-5      **Reserved:** Set to '0'

bit 4        **GPIOOUTVAL:** GPIO Output Value bit (valid only when GP0 is set for GPIO output operation)
      1 = Default output value is logic '1' (factory default)
      0 = Default output value is logic '0'

bit 3        **GPIODIR:** GPIO Direction bit (input or output; valid only when GP0 is set for GPIO operation)
      1 = GPIO input
      0 = GPIO output (factory default)

bit 2-0      **GPDES[2:0]:** GP0 Designation bits
      111 = Reserved
      110 = Reserved
      101 = Reserved
      100 = Reserved
      011 = Reserved
      010 = Alternate Function 0 (LED UART Rx – LED_URx) (factory default)
      001 = Dedicated function operation (SSPND)
      000 = GPIO operation (GPIO0)

**REGISTER 1-13: GPSETTING1 REGISTER**

| r-0 | r-0 | r-0 | R/W-1 | R/W-0 | R/W-0 | R/W-1 | R/W-1 |
|-----|-----|-----|-------|-------|-------|-------|-------|
| — | — | — | GPIOOUTVAL | GPIODIR | GPDES2 | GPDES1 | GPDES0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | r = Reserved bit | | |
|---------|------------------|---|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-5     **Reserved:** Set to '0'

bit 4     **GPIOOUTVAL:** GPIO Output Value bit (valid only when GP1 is set for GPIO output operation)
        1 = Default output value is logic '1' (factory default)
        0 = Default output value is logic '0'

bit 3     **GPIODIR:** GPIO Direction bit (input or output; valid only when GP1 is set for GPIO operation)
        1 = GPIO input
        0 = GPIO output (factory default)

bit 2-0     **GPDES[2:0]:** GP1 Designation bits
        111 = Reserved
        110 = Reserved
        101 = Reserved
        100 = Alternate Function 2 (Interrupt detector)
        011 = Alternate Function 1 (LED UART Tx – LED_UTx) (factory default)
        010 = Alternate Function 0 (ADC1)
        001 = Dedicated function operation (clock output)
        000 = GPIO operation (GPIO1)

**REGISTER 1-14:     GPSETTING2 REGISTER**

| r-0 | r-0 | r-0 | R/W-1 | R/W-0 | R/W-0 | R/W-0 | R/W-1 |
|-----|-----|-----|-------|-------|-------|-------|-------|
| — | — | — | GPIOOUTVAL | GPIODIR | GPDES2 | GPDES1 | GPDES0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | | |
|---------|---|---|---|
| | r = Reserved bit | | |
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-5     **Reserved:** Set to '0'

bit 4     **GPIOOUTVAL:** GPIO Output Value bit (valid only when GP2 is set for GPIO output operation)

  1 =  Default output value is logic '1' (factory default)
  0 =  Default output value is logic '0'

bit 3     **GPIODIR:** GPIO Direction bit (input or output; valid only when GP2 is set for GPIO operation)

  1 =  GPIO input
  0 =  GPIO output (factory default)

bit 2-0     **GPDES[2:0]:** GP2 Designation bits

  111 = Reserved
  110 = Reserved
  101 = Reserved
  100 = Reserved
  011 = Alternate Function 1 (DAC1)
  010 = Alternate Function 0 (ADC2)
  001 = Dedicated function operation (USBCFG) (factory default)
  000 = GPIO operation (GPIO2)

**REGISTER 1-15:    GPSETTING3 REGISTER**

| r-0 | r-0 | r-0 | R/W-1 | R/W-0 | R/W-0 | R/W-0 | R/W-1 |
|-----|-----|-----|-------|-------|-------|-------|-------|
| — | — | — | GPIOOUTVAL | GPIODIR | GPDES2 | GPDES1 | GPDES0 |
| bit 7 | | | | | | | bit 0 |

| Legend: | | r = Reserved bit | |
|---------|---|------------------|---|
| R = Readable bit | W = Writable bit | U = Unimplemented bit, read as '0' | |
| -n = Value at POR | '1' = Bit is set | '0' = Bit is cleared | x = Bit is unknown |

bit 7-5      **Reserved:** Set to '0'

bit 4        **GPIOOUTVAL:** GPIO Output Value bit (valid only when GP3 is set for GPIO output operation)
             1 = Default output value is logic '1' (factory default)
             0 = Default output value is logic '0'

bit 3        **GPIODIR:** GPIO Direction bit (input or output; valid only when GP3 is set for GPIO operation)
             1 = GPIO input
             0 = GPIO output (factory default)

bit 2-0      **GPDES[2:0]:** GP3 Designation bits
             111 = Reserved
             110 = Reserved
             101 = Reserved
             100 = Reserved
             011 = Alternate Function 1 (DAC2)
             010 = Alternate Function 0 (ADC3)
             001 = Dedicated function operation (LED_I2C) (factory default)
             000 = GPIO operation (GPIO3)

# MCP2221A

## 1.5 USB Module (HID, CDC and Transceiver Modules)

The USB HID and CDC modules in the MCP2221A are full-speed USB 2.0-compliant.

- Composite Device (CDC + HID):
  - CDC: USB-to-UART communications
  - HID: $I^2C$ transactions, GPIO control, configuration and miscellaneous operations (ADC, DAC, clock output)
- 448-Byte Buffer to Handle Data Throughput at Any UART Baud Rate:
  - 64-byte transmit
  - 384-byte receive
- Fully Configurable VID and PID Assignments and Descriptors (stored on-chip)
- Bus-Powered or Self-Powered

### 1.5.1 DESCRIPTORS

During configuration, the supplied PC interface stores the descriptors in the MCP2221A.

### 1.5.2 SUSPEND AND RESUME

The USB suspend and resume signals are supported for power management of the MCP2221A. The device enters Suspend mode when "Suspend Signaling" is detected on the bus.

The MCP2221A exits Suspend mode when any of the following events occur:

- "Resume Signaling" is detected or generated
- A USB "Reset" signal is detected
- A device Reset occurs

## 1.6 USB Transceiver

The MCP2221A has a built-in USB 2.0 full-speed transceiver internally connected to the USB module.

The USB transceiver obtains power from the $V_{USB}$ pin, which is internally connected to the 3.3V regulator. The best electrical signal quality is obtained when $V_{USB}$ is locally bypassed with a high-quality ceramic capacitor.

### 1.6.1 INTERNAL PULL-UP RESISTORS

The MCP2221A devices have built-in pull-up resistors designed to meet the requirements for full-speed USB.

### 1.6.2 MCP2221A POWER OPTIONS

The following are the main power options for the MCP2221A:

- USB Bus-Powered (5V)
- 3.3V Self-Powered

#### 1.6.2.1 Internal Power Supply Details

MCP2221A offers various options for power supply. To meet the required USB signaling levels, the MCP2221A incorporates an internal LDO that is used solely by the USB transceiver to present the correct D+/D- voltage levels.

Figure 1-2 shows the internal connections of the USB transceiver LDO in relation to the $V_{DD}$ power supply rail. The output of the USB transceiver LDO is tied to the $V_{USB}$ line. A capacitor connected to the $V_{USB}$ pin is required if the USB transceiver LDO provides the 3.3V supply to the transceiver.

**FIGURE 1-2: MCP2221A INTERNAL POWER SUPPLY DETAILS**
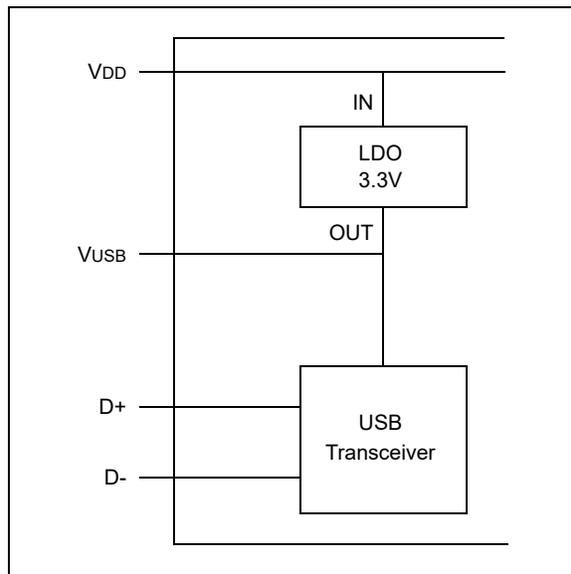


The provided $V_{DD}$ voltage has a direct influence on the voltage levels present on the GPIO and UART Tx/Rx pins. When $V_{DD}$ is 5V, all of these pins will have a logical '1' around 5V with the variations specified in **Section 4.1 "DC Characteristics"**.

For applications that require a 3.3V logical '1' level, $V_{DD}$ must be connected to a power supply providing the 3.3V voltage. In this case, the internal USB transceiver LDO cannot provide the required 3.3V power. It is necessary to also connect the $V_{USB}$ pin of the MCP2221A to the 3.3V power supply rail. This way, the USB transceiver is powered up directly from the 3.3V power supply.

### 1.6.2.2 USB Bus-Powered (5V)

In Bus Power Only mode, all power for the application is drawn from the USB (Figure 1-3). This is effectively the simplest power method for the device.

**FIGURE 1-3:** **BUS POWER ONLY**
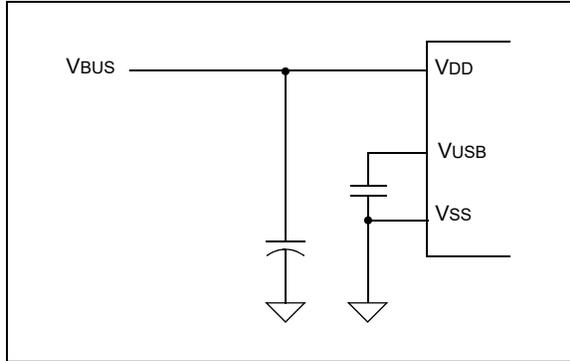
In order to meet the inrush current requirements of the USB 2.0 specifications, the total effective capacitance appearing across VBUS and ground must be no more than 10 μF. If it is more than 10 μF, some kind of inrush limiting is required. For more details on inrush current limiting, search for that subject in the latest "*Universal Serial Bus Specification*".

According to the USB 2.0 specification, all USB devices must also support a Low-Power Suspend mode. In the USB Suspend mode, devices must consume no more than 500 μA (or 2.5 mA for high powered devices that are remote wake-up-capable) from the 5V VBUS line of the USB cable.

The host signals the USB device to enter Suspend mode by stopping all USB traffic to that device for more than 3 ms.

The USB bus provides a 5V voltage. However, the USB transceiver requires 3.3V for the signaling (on D+ and D- lines).

During USB Suspend mode, the D+ or D- pull-up resistor must remain active, which will consume some of the allowed suspend current budget (500 μA/2.5 mA). The VUSB pin requires an external bypass capacitor with a value between 0.22 and 0.47 μF (ceramic cap).

Figure 1-4 shows a circuit where MCP2221A's internal LDO is used to provide 3.3V to the USB transceiver. The voltage on the VDD affects the voltage levels onto the UART and GPIO pins. With VDD at 5V, these pins will have a logic '1' of 5V with the variations specified in **Section 4.1 "DC Characteristics"**.

**FIGURE 1-4:** **TYPICAL POWER SUPPLY OPTION USING THE 5V PROVIDED BY THE USB**
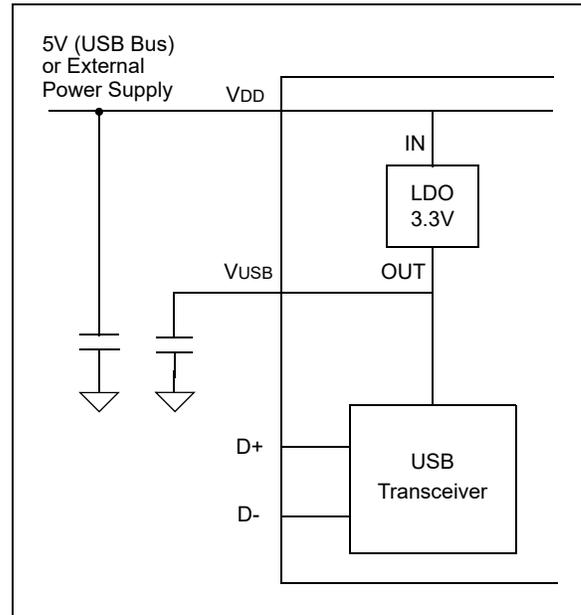
### 1.6.2.3 3.3V Self-Powered

Typically, many embedded applications are using 3.3V power supplies. When such option is available in the target system, MCP2221A can be powered up from the existing 3.3V power supply rail. The typical connections for MCP2221A are shown in Figure 1-5.

In this example, MCP2221A has both VDD and VUSB lines tied to the 3.3V rail. These tied connections disable the internal USB transceiver LDO of the MCP2221A to regulate the power supply on the VUSB pin. Another consequence is that the '1' logical level on the GPIO pins will be at the 3.3V level, in accordance with the variations specified in **Section 4.1 "DC Characteristics"**.

# MCP2221A

## 1.6.2.4 Remote Wake-up Capability

The MCP2221A offers a mechanism for triggering a remote wake-up event for the USB host. The remote wake-up trigger works only with the external interrupt detector.

In order to use this capability, GP1 must be designated for interrupt detection operation. Before the USB host goes into Sleep/Standby mode, the interrupt detector must be set up for detecting positive edges, negative edges or both; also, the detector flag must be cleared. After these conditions are met, the USB host can go into Sleep/Standby mode and it will be awakened whenever an external signal on GP1 triggers the interrupt detector.

**FIGURE 1-5:** **USING AN EXTERNALLY PROVIDED 3.3V POWER SUPPLY**
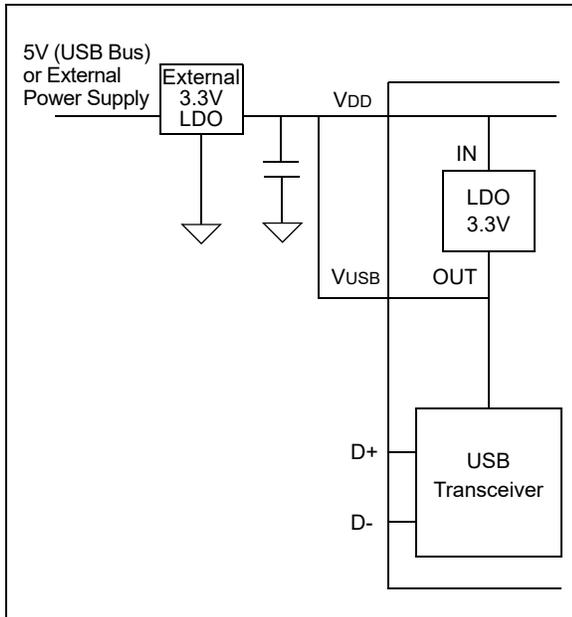


## 1.7 Pin MUX Module

The pin MUX module offers multiple functionalities for the GP pins.

### 1.7.1 CONFIGURABLE PIN FUNCTIONS

The pins can be configured as:

- GPIO – individually configurable general purpose input or output
- SSPND – USB Suspend state
- USBCFG – indicates USB configuration status
- LED_URX – indicates UART receive traffic (when seen from the MCP2221A)
- LED_UTX – indicates UART transmit traffic (when seen from the MCP2221A)
- LED_I2C – indicates $I^2C$ traffic
- ADC1/2/3 – analog inputs connected to the internal 10-bit ADC
- DAC1/2 – analog outputs connected to the same 5-bit DAC
- CLKR – digital clock output (the nominal value is 12 MHz, but other values are possible)
- IOC – external interrupt detector

#### 1.7.1.1 GPIO Pin Function

When the GPIO pin function is enabled for a given GP(n) pin, it will operate as a digital input or an output pin. When configured as a digital output, its value is controlled through the USB HID commands. When configured as a digital input, its logic value is read using USB HID commands.

#### 1.7.1.2 SSPND Pin Function

The SSPND pin (if enabled) reflects the USB state (Suspend/Resume). The pin is active-low when the Suspend state has been issued by the USB host. Likewise, the pin drives high after the Resume state is achieved.

This pin allows the application to go into Low-Power mode when USB communication is suspended and switches to a full active state when USB activity is resumed.

### 1.7.1.3 USBCFG Pin Function

The USBCFG pin (if enabled) starts out low during power-up or after Reset and goes high after the device successfully configures to the USB. The pin will go low when in Suspend mode and high when the USB resumes.

### 1.7.1.4 LED_URX

The "Rx" in this pin name refers to the UART of the MCP2221A. The LED_URX pin is an indicator of UART Rx characters being received.

This pin will pulse low for a period of time (a few milliseconds). This allows the application to provide a visual indication of the UART Rx traffic.

### 1.7.1.5 LED_UTX

The "Tx" in this pin name refers to the UART of the MCP2221A. The LED_UTX pin is an indicator of UART Tx characters being transmitted.

This pin will pulse low for a period of time (a few milliseconds). This allows the application to provide a visual indication of the UART Tx traffic.

### 1.7.1.6 LED_I2C

The "I2C" in this pin name refers to the $I^2C$ module in the MCP2221A. The LED_I2C pin is an indicator of $I^2C$ activity.

This pin will pulse low for a period of time (a few milliseconds). This allows the application to provide a visual indication of the $I^2C$ traffic.

### 1.7.1.7 ADC1/2/3

When GP1/2/3 are configured for ADC operation, they will work as analog input pins and they are tied to the first three channels of the 10-bit ADC in the MCP2221A.

### 1.7.1.8 DAC1/2

When GP2/3 are configured for DAC operation, they will work as analog output pins and they are tied to the output of the MCP2221A's 5-bit DAC.

**TABLE 1-5:    GP DESIGNATION TABLE**

| GP Designation Bits[2:0] | Assignment | GP0 | GP1 | GP2 | GP3 |
|---|---|---|---|---|---|
| 000 | GPIO | GPIO | GPIO | GPIO | GPIO |
| 001 | DEDICATED_FUNC | SSPND | CLK OUT | USBCFG | LED_I2C |
| 010 | ALT_FUNC_0 | LED_URX | ADC1 | ADC2 | ADC3 |
| 011 | ALT_FUNC_1 | — | LED_UTX | DAC1 | DAC2 |
| 100 | ALT_FUNC_2 | — | IOC | — | — |

## 1.8 GPIO/ADC/DAC Module

This module communicates with the USB HID sub-module through the Bus Matrix module. It allows the manipulation of GPIOs, retrieving the ADC data and setting the DAC value.

### 1.8.1 GPIO

When the GPs are configured for GPIO operation, those configured GPs can be used as digital inputs or outputs.

When working as outputs, the GPs output logic levels (logic '0' or '1').

#### 1.8.1.1 VRM

The ADC and DAC sub-modules each have a Voltage Reference (VRM) module. Each VRM can be configured (at power-up and run time) to output one of the four voltage choices available.

> **Note:** When the Set SRAM settings command is used for GPIO control, the reference voltage for VRM is always reinitialized to the default value (VDD) if it is not explicitly set.
>
> For saving t**he** actual VRM value, use the Set GPIO Output Values command.

The VRM can provide the following voltages as a reference:

- VDD – the VRM output is exactly the voltage present at the VDD pin of the MCP2221A. It can take any value from 3.3 to 5V.
- 1.024V – the VRM output of 1.024V is obtained from an internal voltage reference.
- 2.048V – the VRM output of 2.048V is obtained from an internal voltage reference.
- 4.096V – the VRM output of 4.096V is obtained from an internal voltage reference. If the VDD is lower than 4.096V, the VRM output will have the value of VDD.
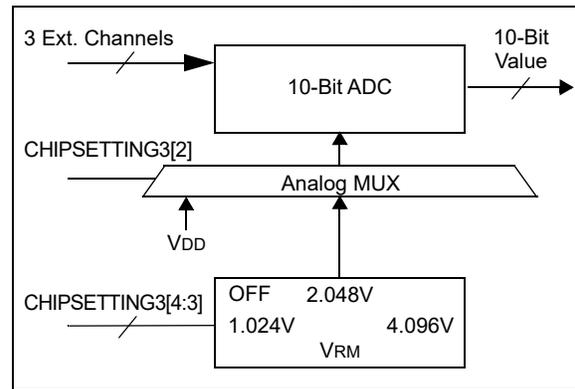
### 1.8.2 ADC CONVERTER

The ADC Converter produces 10-bit values and it uses its own VRM module.

It features three external channels (connected to GP1/2/3 if configured for ADC operation).

The sampling rate of the ADC is around 1000 SPS.

**FIGURE 1-6: ADC SUB-MODULE DETAILS**



### 1.8.3 DAC CONVERTER

The DAC is 5-bit wide, has a single analog output and it uses its own VRM module.

The DAC output voltage can be routed to GP2/3 (if GP2/3 are configured for DAC operation).

If the GP2 and GP3 are configured for DAC operation, they will present the same analog voltage value because they are connected to the same DAC output.

**FIGURE 1-7: DAC SUB-MODULE DETAILS**
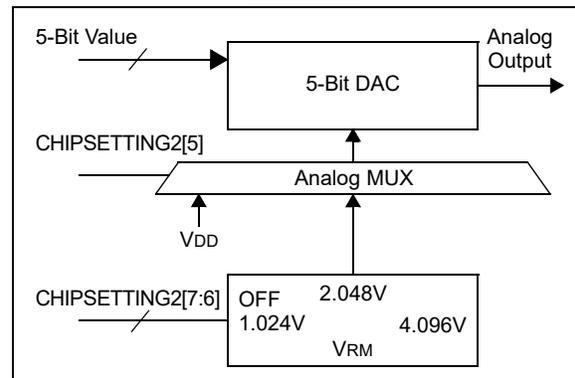


## 1.9 CLKR

When GP1 is configured for clock output operation, the GP1 pin will act as a digital output, providing a clock signal derived from the device's internal clock. The clock's nominal frequency is 12 MHz, ±0.25%. Other clock values and duty cycles are possible by setting different values that are associated with this mode of operation.

## 1.10 IOC

When GP1 is configured for Interrupt-on-Change (IOC) operation, GP1 acts as a digital input that is sensitive to positive and negative edges. Depending on the settings associated with this mode of operation, the GP1 can detect positive, negative or both edges.

## 1.11 RESET/POR

### 1.11.1 RESET PIN

The $\overline{RST}$ pin provides a method for triggering an external Reset of the device. A Reset is generated by holding the pin low. These devices have a noise filter in the Reset path, which detects and ignores small pulses.

### 1.11.2 POR

A POR pulse is generated on-chip whenever VDD rises above a certain threshold. This allows the device to start in the initialized state when VDD is adequate for operation.

To take advantage of the POR circuitry, tie the $\overline{RST}$ pin to VDD through a resistor (1-10 kΩ). This will eliminate external RC components usually needed to create a POR delay.

In the self-powered configuration, it is recommended to tie the $\overline{RST}$ pin to the $V_{BUS}$ line of the USB connector, as in Figure 1-8.

**FIGURE 1-8:** **CONNECTING THE $\overline{RST}$ PIN IN A SELF-POWERED CONFIGURATION (RECOMMENDED)**
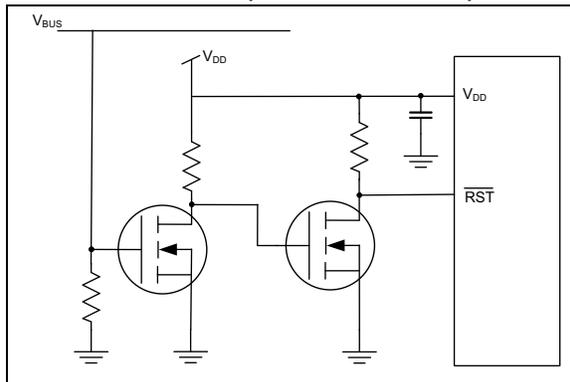


When the device starts normal operation (i.e., exits the Reset condition), device operating parameters (voltage, frequency, temperature, etc.) must be met to ensure operation. If these conditions are not achieved, the device must be held in Reset until the operating conditions are met.

## 1.12 Internal Oscillator

The MCP2221A features an internal oscillator that provides a 12 MHz clock, which is needed for the USB modules (HID and CDC).

Full-speed USB is nominally 12 Mb/s. The clock signal's accuracy is over temperature (2,500 ppm maximum).

The internal clock of the MCP2221A is fed into the CLKR module to provide a clock signal outside of the device.

GP1 can be configured as a clock output pin providing a 12 MHz clock to the rest of the system. Other clock and duty cycle values are possible by using different settings for this module.

## 1.13 I²C Host Module

The I²C host module is responsible for the I²C traffic generation. The module is controlled through the USB HID, through the Bus Matrix module.

The I²C module only implements the functionality of an I²C/SMBus host.

### 1.13.1 I²C/SMBus HOST

The I²C host initiates all the I²C/SMBus transactions (being read or write operations) on the bus.

The I²C/SMBus host module has the following capabilities:

- Sending/receiving data at a multitude of bit rates, up to 400 kbps
- 7-Bit Addressing mode
- Single data transfers of up to 65,535 bytes
- Clock stretching (it allows the slower I²C clients to communicate)

All the user data to be sent/transmitted over the I²C bus are conveyed to the USB host only through the USB HID interface.

## 1.14 Bus Matrix Module

The Bus Matrix module manages the communication between various functional modules, such as USB (HID and CDC), I²C, UART, GPIO/ADC/DAC, Config, IOC, CLKR, MUX pin.

## 1.15 Config Module

The Config module is in charge of the storage of the device settings and also of their management (loading/modifying/access protection). The module uses nonvolatile memory for storing the power-up device settings.

At power-up, the module loads the settings from the nonvolatile storage area into an SRAM location (volatile settings). These settings represent the device's configuration, along with other key parameters (e.g., string descriptors, VID/PID, etc.). After the settings are loaded in SRAM (volatile settings), they can be changed through the USB HID interface.

The user can read/modify/change either settings (nonvolatile or volatile) through the same interface (USB HID).

The Config module contains the relevant power-up settings that are used by the MCP2221A. A few examples of settings are USB descriptors, GP settings, ADC, DAC and CLKR.

# MCP2221A

## 2.0    USB ENUMERATION PROCESS

The MCP2221A implements the CDC class to support the USB-to-UART protocol converter functionality. Using USB-to-UART (CDC class) adapters with personal computers running the Windows Operating System (OS) requires some consideration because of the way the Windows OS responds to their connection.

When a USB-to-UART (CDC class) adapter is connected to the USB port of the PC, Windows searches for a driver. After a suitable driver is found, the system creates an entry in the registry. The entry stores relevant information about the USB-to-UART adapter, its driver and the associated COM port.

The COM port and its number are legacy-type adapters, which are still supported by Windows OS. Historically, the COM ports in a computer are part of the computer's motherboard and are assigned a different index number. With the advent of USB-to-UART adapters, the Windows OS kept the COM port concept and extended it to support the USB adapters.

Whenever a USB-to-UART adapter is first connected to a PC, the system searches the registry for an entry that is suitable for the connected adapter. If one is not found, the system asks for a suitable driver. If this step is completed, it creates a registry entry, and assigns a COM port number as well. Then, whenever the USB-to-UART adapter is connected to that PC, the system checks the registry entry, loads the specified driver and assigns the given COM port number (as found in the registry entry).

During the enumeration process, the device can specify a serial number. If it does, this number is stored in the registry entry and it is used to assign the same COM port number to the adapter in question, no matter which USB port the adapter is connected to.

USB-to-UART adapters have the option to not present a serial number during USB enumeration. In this case, the operating system would not be able to differentiate between two identical devices if neither is providing its serial number. Each time one of these two devices (with no serial number provided during enumeration) is connected to the same USB port, it will have the same COM port number assigned.

Both functionalities (with or without serial numbers) are very useful for different applications.

When the serial number is provided, an adapter using the MCP2221A solution receives the same COM port number from a Windows machine, no matter which USB port it is connected to.

The case with no serial number is useful for test/validation of products using the MCP2221A. The fact that all the tested boards are not supplying a serial number will force Windows to assign them the same COM port number (but only if connected to the same USB port).

The MCP2221A is factory-set to not use a serial number. Later in the process, if a customer wants the benefits provided by using a serial number, the Configuration Utility from Microchip can be used to enable the MCP2221A to enumerate its serial number as well.

The MCP2221A comes with a uniquely provided serial number to be used during the USB enumeration process; however, this can be changed by the user in the Configuration Utility.

The serial number enumeration enable/disable can be changed, as well, using the Configuration Utility.

All the USB-related settings mentioned above are part of the Device Configuration (Chip settings area) and they reside only in Flash. When the Chip settings area ($1^{st}$ area) is being copied into the SRAM (at power-up), the USB settings are skipped (not copied into SRAM).

## 3.0 USB HID COMMUNICATION

Except for the USB CDC and UART modules, all the other modules in the MCP2221A use USB HID protocol for communication.

The USB HID protocol uses 64-byte reports.

A typical command exchange starts with a 64-byte packet that is written by the USB host (i.e., the PC). Afterward, the USB host reads the response from the device as a 64-byte packet.

## 3.1 USB HID Commands/Responses

### 3.1.1 Status/Set Parameters

This command offers many options for this device. It is used to poll for the status of the device. It is also used to establish certain $I^2C$ bus parameters/conditions.

**TABLE 3-1: COMMAND STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0x10 | Status/Set Parameters – command code. |
| 1 | Don't care | Any value | — |
| 2 | Cancel current $I^2C$/SMBus transfer (sub-command) | 0x10 | When this value is put in this field, the device will cancel the current $I^2C$/SMBus transfer and will attempt to free the $I^2C$ bus. This command is very useful since it can cancel a transfer and free the bus. An example would be when trying to communicate with a device using a wrong address. This will cause a time-out to occur. This time-out situation can be read using the "Status/Set Parameters" and the cancellation of the $I^2C$/SMBus transfer can be achieved by this sub-command. |
| | | Any other value | No effect. |
| 3 | Set $I^2C$/SMBus communication speed (sub-command) | 0x20 | When this value is put in this field, the device will take the next command field and interpret it as the system clock divider that will give the $I^2C$/SMBus communication clock. |
| | | Any other value | No effect. |
| 4 | The $I^2C$/SMBus system clock divider that will be used to establish the communication speed[1] | — | The value in this field is being taken into consideration only when the Byte Index 3 contains the code for establishing a new communication speed. In all the other cases, this field's value will not matter. |
| 5-63 | Don't care | Any value | — |

**Note 1:** To compute the desired value for the clock divider, use the following adjusted formula:
Divider = (12 MHz/$I^2C$CLOCK RATE) – 2.

### 3.1.1.1 Responses

**TABLE 3-2: RESPONSE 1 STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0x10 | `Status/Set Parameters` – command code echo. |
| 1 | — | 0x00 | Command completed successfully. |
| 2 | Cancel transfer | 0x00 | No special operation (i.e., cancel current I$^2$C/SMBus transfer). |
| | | 0x10 | The current I$^2$C/SMBus transfer was marked for cancellation. The actual I$^2$C/SMBus transfer cancellation and bus release will need some time (a few hundreds of microseconds, depending on the communication speed initially chosen for the canceled transfer). |
| | | 0x11 | The I$^2$C engine (inside MCP2221A) was already in Idle mode. The cancellation command had no effect. |
| 3 | — | 0x00 | No set I$^2$C/SMBus communication speed was issued. |
| | | 0x20 | The new I$^2$C/SMBus communication speed is now considered. |
| | | 0x21 | The I$^2$C/SMBus communication speed was not set (e.g., I$^2$C transfer in progress). |
| 4 | — | The divider value given at the same index in the command field | Only in the case when the code for establishing a new communication speed is given at Byte Index 3. |
| | | 0x00 | When the communication speed is not being set. |
| 5-7 | Don't care | Any value | — |
| 8 | I$^2$C Communication state | 0 | If the MCP2221A I$^2$C Engine is in Idle Mode, else timeout occurred. |
| 9 | Lower byte (16-bit value) of the requested I$^2$C transfer length | — | — |
| 10 | Higher byte (16-bit value) of the requested I$^2$C transfer length | — | — |
| 11 | Lower byte (16-bit value) of the already transferred (through I$^2$C) number of bytes | — | — |
| 12 | Higher byte (16-bit value) of the already transferred (through I$^2$C) number of bytes | — | — |
| 13 | Internal I$^2$C data buffer counter | — | — |
| 14 | Current I$^2$C communication speed divider value | — | — |
| 15 | Current I$^2$C time-out value | — | — |
| 16 | Lower byte (16-bit value) of the I$^2$C address being used | — | — |
| 17 | Higher byte (16-bit value) of the I$^2$C address being used | — | — |
| 18-19 | Don't care | Any value | — |

**TABLE 3-2: RESPONSE 1 STRUCTURE (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 20 | Bit 7: don't care | — | — |
| | Bit 6: ACK Status | — | If ACK was received from client value is `0`, else `1`. |
| | Bit 5-0: don't care | — | — |
| 18-19 | Don't care | Any value | — |
| 22 | SCL line value – as read from the pin | — | — |
| 23 | SDA line value – as read from the pin | — | — |
| 24 | Interrupt edge detector state | `0 or 1` | — |
| 25 | I²C read pending value | `0, 1 or 2` | This field is used by the USB host to know if the MCP2221A still has to read from a client device. |
| 26-45 | Don't care | Any value | — |
| 46 | MCP2221A Hardware Revision Major ('A') | — | — |
| 47 | MCP2221A Hardware Revision Minor ('6') | — | — |
| 48 | MCP2221A Firmware Revision Major ('1') | — | — |
| 49 | MCP2221A Firmware Revision Minor ('1') | — | — |
| 50-55 | ADC data (16-bit) values | — | 3 x (16-bit) little-endian ADC channel values (CH0 LSB, CH0 MSB, CH1 LSB, CH1 MSB, CH2 LSB, CH2 MSB). |
| 56-63 | Don't care | Any value | — |

# MCP2221A

### 3.1.2 `Read Flash Data`

This command is used to read various important data structures and strings that are stored in Flash memory on the MCP2221A.

**TABLE 3-3: COMMAND STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0xB0 | `Read Flash Data` – command code. |
| 1 | `Read Flash Data` sub-code. The value in this field will instruct the MCP2221A on what Flash data are to be read. | 0x00 | `Read Chip Settings` – reads the MCP2221A Flash settings. |
| | | 0x01 | `Read GP Settings` – reads the MCP2221A Flash GP settings |
| | | 0x02 | `Read USB Manufacturer Descriptor String` – reads the USB Manufacturer String Descriptor used during the USB enumeration. |
| | | 0x03 | `Read USB Product Descriptor String` – reads the USB Product String Descriptor used during the USB enumeration. |
| | | 0x04 | `Read USB Serial Number Descriptor String` – reads the USB Serial Number String Descriptor that is used during USB enumeration. This serial number can be changed by the user through a specific USB HID command. |
| | | 0x05 | `Read Chip Factory Serial Number` – reads the factory set serial number. This serial number cannot be changed. |
| | | Any other value | No meaning. The device will reply with a code for an unsupported command at Byte Index 1 in the Response report. |
| 2-63 | Reserved | 0x00 | — |

#### 3.1.2.1 Responses

**TABLE 3-4: RESPONSE STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0xB0 | `Read Flash Data` – command code. |
| 1 | — | 0x00 | Command completed successfully. |
| | | 0x01 | Command is not supported. |
| 2 | Data structure length or Don't care | — | — |
| 3-63 | Data or Don't care | — | Depends on the issued sub-command or the returned code at Byte Index 1. |

**TABLE 3-5:** **RESPONSE STRUCTURE – `Read Chip Settings` SUB-COMMAND**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0xB0 | `Read Flash Data` – command code echo. |
| 1 | — | 0x00 | Command completed successfully. |
| 2 | Structure length | — | — |
| 3 | Don't care | — | — |
| 4 | Bit 7: CDC serial number enumeration enable | 1 | The USB serial number will be used during the USB enumeration of the CDC interface. |
| | | 0 | No serial number descriptor will be presented during the USB enumeration. |
| | Bits 6-2: Not implemented | — | — |
| | Bits 1-0: Chip configuration security option | 11-10 | Permanently locked. |
| | | 01 | Password-protected. |
| | | 00 | Unsecured. |
| 5 | Bits 7-5 | Don't care | — |
| | Bits 4-0: Clock output divider value | — | If the GP pin (exposing the clock output) is enabled for clock output operation, the divider value will be used on the 48 MHz USB internal clock and its divided output will be sent to this pin. |
| 6 | Bits 7-6: DAC reference voltage option | 11 | Reference voltage is 4.096V (only if $V_{DD}$ is above this voltage). |
| | | 10 | Reference voltage is 2.048V. |
| | | 01 | Reference voltage is 1.024V. |
| | | 00 | Reference voltage is off (this is useful for the case in which the DAC uses another reference other than $V_{RM}$ DAC; e.g., $V_{DD}$). |
| | Bit 5: DAC reference option | 1 | DAC reference is $V_{RM}$ DAC voltage. |
| | | 0 | DAC reference is $V_{DD}$. |
| | Bits 4-0: Power-up DAC value | — | — |
| 7 | Bit 7 | Don't care | — |
| | Bit 6: Interrupt detection – negative edge | — | If set, the interrupt detection flag will be set when a negative edge occurs. |
| | Bit 5: Interrupt detection – positive edge | — | If set, the interrupt detection flag will be set when a positive edge occurs. |
| | Bits 4-3: ADC reference voltage | 11 | Reference voltage is 4.096V (only if $V_{DD}$ is above this voltage). |
| | | 10 | Reference voltage is 2.048V. |
| | | 01 | Reference voltage is 1.024V. |
| | | 00 | Reference voltage is off (this is useful for the case in which the ADC uses another reference other than $V_{RM}$ ADC; e.g., $V_{DD}$). |
| | Bit 2 | 1 | DAC reference is $V_{DD}$ DAC voltage. |
| | | 0 | DAC reference is $V_{RM}$. |
| | Bit 1 | Don't care | — |
| | Bit 0 | Don't care | — |
| 8 | Lower byte of the 16-bit USB VID value | — | — |
| 9 | Higher byte of the 16-bit USB VID value | — | — |

# MCP2221A

**TABLE 3-5:  RESPONSE STRUCTURE – `Read Chip Settings` SUB-COMMAND (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 10 | Lower byte of the 16-bit USB PID value | — | — |
| 11 | Higher byte of the 16-bit USB PID value | — | — |
| 12 | USB power attributes[1] | — | This value will be used by the MCP2221A's USB Configuration Descriptor (power attributes' value) during the USB enumeration. |
| 13 | USB requested number of mA(s)[1] | — | The requested mA value during the USB enumeration will represent the value at this index multiplied by two. |
| 14-63 | Don't care | — | — |

**Note  1:**  Please consult the USB 2.0 specification for details on the correct values for power and attributes.

**TABLE 3-6: RESPONSE STRUCTURE – `Read GP Settings` SUB-COMMAND**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0xB0 | `Read Flash Data` – command code echo. |
| 1 | — | 0x00 | Command completed successfully. |
| 2 | — | — | Structure length. |
| 3 | — | Don't care | — |
| 4 | **GP0 Power-up Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP0 is set as an output GPIO, this value will be present at the GP0 pin at power-up/Reset. |
| | Bit 3: GPIO direction (input/output) – works only when GP0 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP0 designation | 111–011 | Don't care. |
| | | 010 | Alternate Function 0 (LED_URx). |
| | | 001 | Dedicated function operation (SSPND). |
| | | 000 | GPIO operation. |
| 5 | **GP1 Power-up Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP1 is set as an output GPIO, this value will be present at the GP1 pin at power-up/Reset. |
| | Bit 3: GPIO direction (input/output) – works only when GP0 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP1 designation | 111–101 | Don't care. |
| | | 001 | Dedicated function operation (clock output). |
| | | 100 | Alternate Function 2 (interrupt detection). |
| | | 011 | Alternate Function 1 (LED_UTx). |
| | | 010 | Alternate Function 0 (ADC1). |
| | | 000 | GPIO operation. |
| 6 | **GP2 Power-up Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP2 is set as an output GPIO, this value will be present at the GP2 pin at power-up/Reset. |
| | Bit 3: GPIO direction (input/output) – works only when GP2 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP1 designation | 111–100 | Don't care. |
| | | 011 | Alternate Function 1 (DAC1). |
| | | 010 | Alternate Function 0 (ADC2). |
| | | 001 | Dedicated function operation (USB). |
| | | 000 | GPIO operation |

**TABLE 3-6:** RESPONSE STRUCTURE – `Read GP Settings` SUB-COMMAND (CONTINUED)

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 7 | **GP3 Power-up Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP3 is set as an output GPIO, this value will be present at the GP3 pin at Power-up/Reset. |
| | Bit 3: GPIO direction (input/output) – works only when GP3 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP1 designation | 111–100 | Don't care. |
| | | 011 | Alternate Function 1 (DAC2). |
| | | 010 | Alternate Function 0 (ADC3). |
| | | 001 | Dedicated function operation (LED_I2C). |
| | | 000 | GPIO operation. |
| 8-63 | — | Don't care | — |

**TABLE 3-7:** RESPONSE STRUCTURE – `Read USB Manufacturer Descriptor String` SUB-COMMAND

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB0 | `Read Flash Data` – command code echo. |
| 1 | 0x00 | Command completed successfully. |
| 2 | **Note 2** | Number of bytes + 2 in the provided USB Manufacturer Descriptor String. The actual string starts at Byte Index 4. |
| 3 | 0x03 | The value at this index must always be 0x03. |
| 4 + 2 x Unicode_char_number + 0[1] | — | Lower byte of the 16-bit Unicode character. |
| 4 + 2 x Unicode_char_number + 1[1] | — | Higher byte of the 16-bit Unicode character. |
| (4 + 2 x Unicode_char_number + 2) – 63[1] | — | Don't care. Only if the USB String Descriptor is less than 60 bytes long in total. |

**Note 1:** "Unicode_char_number" value starts from 0 to a maximum of 30 (included).

**2:** The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

**TABLE 3-8:** RESPONSE STRUCTURE – `Read USB Product Descriptor String` SUB-COMMAND

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB0 | `Read Flash Data` – command code echo. |
| 1 | 0x00 | Command completed successfully. |
| 2 | **Note 2** | Number of bytes + 2 in the provided USB Product Descriptor String. The actual string starts at Byte Index 4. |
| 3 | 0x03 | The value at this index must always be 0x03. |
| 4 + 2 x Unicode_char_number + 0[1] | — | Lower byte of the 16-bit Unicode character. |
| 4 + 2 x Unicode_char_number + 1[1] | — | Higher byte of the 16-bit Unicode character. |
| (4 + 2 x Unicode_char_number + 2) – 63[1] | — | Don't care. Only if the USB String Descriptor is less than 60 bytes long (in total). |

**Note 1:** "Unicode_char_number" value starts from 0 to a maximum of 30 (included).

**2:** The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

**TABLE 3-9:** **RESPONSE STRUCTURE –** `Read USB Serial Number Descriptor String` **SUB-COMMAND**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB0 | `Read Flash Data` – command code echo. |
| 1 | 0x00 | Command completed successfully. |
| 2 | **Note 2** | The number of bytes + 2 in the provided USB Serial Number Descriptor String. The actual string starts at Byte Index 4. |
| 3 | 0x03 | The value at this index must always be 0x03. |
| 4 + 2 x Unicode_char_number + 0[1] | — | Lower byte of the 16-bit Unicode character |
| 4 + 2 x Unicode_char_number + 1[1] | — | Higher byte of the 16-bit Unicode character |
| (4 + 2 x Unicode_char_number + 2) – 63[1] | — | Don't care. Only if the USB String Descriptor is less than 60 bytes long in total. |

**Note 1:** "Unicode_char_number" value starts from 0 to a maximum of 30 (included).

**2:** The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

**TABLE 3-10:** **RESPONSE STRUCTURE –** `Read Chip Factory Serial Number` **SUB-COMMAND**[1]

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB0 | `Read Flash Data` – command code echo. |
| 1 | 0x00 | Command completed successfully. |
| 2 | — | Structure length. |
| 3 | Don't care | — |
| 4 – (4 + Structure Length – 1) | — | Structure data – Factory Serial Number String. |
| (4 + Structure Length) – 63 | Don't care | — |

**Note 1:** The Chip Serial Number is typically 8 bytes in length.

# MCP2221A

### 3.1.3 `Write Flash Data`

This command is used to write various important data structures and strings into the Flash memory of the device.

**TABLE 3-11: COMMAND STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0xB1 | `Write Flash Data` – command code. |
| 1 | `Write Flash Data` sub-code. The value in this field will instruct the MCP2221A about the particular Flash settings to be altered. | 0x00 | `Write Chip Settings` – writes the MCP2221A Flash settings. |
| | | 0x01 | `Write GP Settings` – writes the MCP2221A Flash GP settings. |
| | | 0x02 | `Write USB Manufacturer Descriptor String` – writes the USB Manufacturer String Descriptor used during the USB enumeration. |
| | | 0x03 | `Write USB Product Descriptor String` – writes the USB Product String Descriptor used during the USB enumeration. |
| | | 0x04 | `Write USB Serial Number Descriptor String` – writes the USB Serial Number String Descriptor used during the USB enumeration. |
| | | Any other value | No meaning. The device will reply with a code for an unsupported command at Byte Index 1 in the response report. |
| 2-63 | Data to be written | — | Data format depends on the `Write Flash Data` sub-code (at Byte Index 1). |

**TABLE 3-12: SUB-COMMAND STRUCTURE – `Write Chip Settings` SUB-COMMAND**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0xB1 | `Write Flash Data` – command code. |
| 1 | — | 0x00 | `Write Chip Settings` – writes the MCP2221A Flash device settings. |
| 2 | Bit 7: CDC serial number enumeration enable | 1 | The USB serial number will be used during the USB enumeration of the CDC interface. |
| | | 0 | No serial number descriptor will be presented during the USB enumeration. |
| | Bits 6-2: Not implemented | — | — |
| | Bits 1-0: Chip configuration security option | 11-10 | Permanently locked. |
| | | 01 | Password-protected. |
| | | 00 | Unsecured. |
| 3 | Bits 7-5 | Don't care | — |
| | Bits 4-0: Clock output divider value | — | If the GP pin (exposing the clock output) is enabled for clock output operation, the divider value will be used on the 48 MHz USB internal clock and its divided output will be sent to this pin. |
| 4 | Bits 7-6: DAC reference voltage option | 11 | Reference voltage is 4.096V (only if VDD is above this voltage). |
| | | 10 | Reference voltage is 2.048V. |
| | | 01 | Reference voltage is 1.024V. |
| | | 00 | Reference voltage is off (this is useful for the case in which the DAC uses another reference other than VRM DAC, i.e., VDD). |
| | Bit 5: DAC reference option | 1 | DAC reference is VDD. |
| | | 0 | DAC reference is VRM DAC voltage. |
| | Bits 4-0: Power-up DAC value | — | — |
| 5 | Bit 7 | Don't care | — |
| | Bit 6: Interrupt detection – negative edge | — | If set, the interrupt detection flag will be set when a negative edge occurs. |
| | Bit 5: Interrupt detection – positive edge | — | If set, the interrupt detection flag will be set when a positive edge occurs. |
| | Bits 4-3: ADC reference voltage | 11 | Reference voltage is 4.096V (only if VDD is above this voltage). |
| | | 10 | Reference voltage is 2.048V. |
| | | 01 | Reference voltage is 1.024V. |
| | | 00 | Reference voltage is off (this is useful for the case in which the ADC uses another reference other than VRM ADC; e.g., VDD). |
| | Bit 2: ADC reference option | 1 | ADC reference voltage is VRM ADC. |
| | | 0 | ADC reference voltage is VDD. |
| | Bit 1 | Don't care | — |
| | Bit 0 | Don't care | — |
| 6 | Lower byte of the 16-bit USB VID value. | — | — |
| 7 | Higher byte of the 16-bit USB VID value. | — | — |

**TABLE 3-12: SUB-COMMAND STRUCTURE – `Write Chip Settings` SUB-COMMAND (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 8 | Lower byte of the 16-bit USB PID value. | — | — |
| 9 | Higher byte of the 16-bit USB PID value. | — | — |
| 10 | USB power attributes | — | This value will be used by the MCP2221A's USB Configuration Descriptor (power attributes value) during the USB enumeration. |
| 11 | USB requested number of mA(s) | — | The requested mA value during the USB enumeration will represent the value at this index multiplied by two. |
| 12-19 | 8-byte password (for Flash modifications protection) | — | — |
| 20-63 | — | Don't care | — |

**TABLE 3-13:** **SUB-COMMAND STRUCTURE – `Write GP Settings` SUB-COMMAND**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0xB1 | `Write Flash Data` – command code. |
| 1 | — | 0x01 | `Write GP Settings` – it will write the MCP2221A Flash GP settings. |
| 2 | **GP0 Power-up Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP0 is set as an output GPIO, this value will be present at the GP0 pin at power-up/Reset. |
| | Bit 3: GPIO direction (input/output) – works only when GP0 is set for GPIO operation | `1` | GPIO Input mode. |
| | | `0` | GPIO Output mode. |
| | Bit 2-0: GP0 designation | `111-011` | Don't care. |
| | | `010` | Dedicated function operation (SSPND). |
| | | `001` | Alternate Function 0 (LED_URx). |
| | | `000` | GPIO operation. |
| 3 | **GP1 Power-up Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP1 is set as an output GPIO, this value will be present at the GP1 pin at power-up/Reset. |
| | Bit 3: GPIO direction (input/output) – works only when GP1 is set for GPIO operation | `1` | GPIO Input mode. |
| | | `0` | GPIO Output mode. |
| | Bits 2-0: GP1 designation | `111-101` | Don't care. |
| | | `100` | Alternate Function 2 (interrupt detection). |
| | | `011` | Alternate Function 1 (LED_UTx). |
| | | `010` | Alternate Function 0 (ADC1). |
| | | `001` | Dedicated function operation (clock output). |
| | | `000` | GPIO operation. |
| 4 | **GP2 Power-up Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP2 is set as an output GPIO, this value will be present at the GP2 pin at power-up/Reset. |
| | Bit 3: GPIO direction (input/output) – works only when GP2 is set for GPIO operation | `1` | GPIO Input mode. |
| | | `0` | GPIO Output mode. |
| | Bits 2-0: GP2 designation | `111-100` | Don't care. |
| | | `011` | Alternate Function1 (DAC1). |
| | | `010` | Alternate Function 0 (ADC2). |
| | | `001` | Dedicated function operation (clock output). |
| | | `000` | GPIO operation. |

**TABLE 3-13: SUB-COMMAND STRUCTURE – `Write GP Settings` SUB-COMMAND (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 5 | **GP3 Power-up Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP3 is set as an output GPIO, this value will be present at the GP2 pin at power-up/Reset. |
| | Bit 3: GPIO direction (input/output) – works only when GP3 is set for GPIO operation. | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP3 designation | 111-100 | Don't care. |
| | | 011 | Alternate Function 1 (DAC2). |
| | | 010 | Alternate function 0 (ADC3). |
| | | 001 | Dedicated function operation (LED_I2C). |
| | | 000 | GPIO operation. |
| 6-63 | | Don't care | — |

**TABLE 3-14: SUB-COMMAND STRUCTURE – `Write USB Manufacturer Descriptor String` SUB-COMMAND**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB1 | `Write Flash Data` – command code. |
| 1 | 0x02 | `Write USB Manufacturer Descriptor String` – writes the USB Manufacturer String Descriptor used during the USB enumeration. |
| 2 | **Note 2** | Number of bytes + 2 in the provided USB Serial Number Descriptor String. |
| 3 | 0x03 | The value at this index must always be 0x03. |
| 4 + 2 x Unicode_char_number + 0[1] | — | Lower byte of the 16-bit Unicode character. |
| 4 + 2 x Unicode_char_number + 1[1] | — | Higher byte of the 16-bit Unicode character. |

**Note 1:** "Unicode_char_number" value starts from 0 to a maximum of 30 (included).

**2:** The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

**TABLE 3-15: SUB-COMMAND STRUCTURE – `Write USB Product Descriptor String` SUB-COMMAND**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB1 | `Write Flash Data` – command code. |
| 1 | 0x03 | `Write USB Product Descriptor String` – writes the USB Product String Descriptor used during the USB enumeration. |
| 2 | **Note 2** | Number of bytes + 2 in the provided USB Serial Number Descriptor String. The actual string starts at Byte Index 4. |
| 3 | 0x03 | The value at this index must always be 0x03. |
| 4 + 2 x Unicode_char_number + 0[1] | — | Lower byte of the 16-bit Unicode character. |
| 4 + 2 x Unicode_char_number + 1[1] | — | Higher byte of the 16-bit Unicode character. |

**Note 1:** "Unicode_char_number" value starts from 0 to a maximum of 30 (included).

**2:** The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

**TABLE 3-16:** **SUB-COMMAND STRUCTURE –** `Write USB Serial Number Descriptor String`
**SUB-COMMAND**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB1 | `Write Flash Data` – command code. |
| 1 | 0x04 | `Write USB Serial Number Descriptor String` – writes the USB Serial Number String Descriptor used during the USB enumeration. |
| 2 | **Note 2** | Number of bytes + 2 in the provided USB Serial Number Descriptor String. The actual string starts at Byte Index 4. |
| 3 | 0x03 | The value at this index must always be 0x03. |
| 4 + 2 x Unicode_char_number + 0[1] | — | Lower byte of the 16-bit Unicode character. |
| 4 + 2 x Unicode_char_number + 1[1] | — | Higher byte of the 16-bit Unicode character. |

**Note 1:** "Unicode_char_number" value starts from 0 to a maximum of 30 (included).

**2:** The value at Byte Index 2 must be 2 + 2 x (number of Unicode characters in the string).

### 3.1.3.1    Responses

**TABLE 3-17:** **RESPONSE STRUCTURE –** `Read Chip Factory Serial Number`
**SUB-COMMAND**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB1 | `Write Flash Data` – command code. |
| 1 | 0x00 | Command completed successfully. |
| | 0x02 | Command not supported. |
| | 0x03 | Command not allowed. |
| 2-63 | Don't care | — |

# MCP2221A

### 3.1.4 `Send Flash Access Password`

This command is used to send a user-supplied password that will be compared to the one stored in the device's Flash when Flash updates (Chip/GP configuration, USB strings) are required and the Flash data are password-protected.

In the case where no protection mechanism is in place or the Flash data have been permanently locked, this command has no meaning.

**TABLE 3-18:    COMMAND STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB2 | `Send Flash Access Password` – command code. |
| 1 | Don't care | — |
| 2 | — | Password Byte 1. |
| 3 | — | Password Byte 2. |
| 4 | — | Password Byte 3. |
| 5 | — | Password Byte 4. |
| 6 | — | Password Byte 5. |
| 7 | — | Password Byte 6. |
| 8 | — | Password Byte 7. |
| 9 | — | Password Byte 8. |
| 10-63 | — | Don't care. |

#### 3.1.4.1    Responses

**TABLE 3-19:    RESPONSE 1 STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0xB2 | `Send Flash Access Password` – command code echo. |
| 1 | 0x00 | Command completed successfully. |
|  | 0x03 | Command not allowed (when the number of failed Flash updates has been reached, no password will be accepted). |
| 2-63 | Don't care | — |

### 3.1.5 I$^2$C Write Data

This command is used to write user-given data to the I$^2$C client device (the speed is specified by the STATUS/SET Parameters command).

The command will have the following effects:

- The I$^2$C engine will send the "Start" condition.
- The selected I$^2$C client address is sent next and the I$^2$C engine will wait for the client to send an Acknowledge bit.

- The user data follow next and the I$^2$C engine awaits for the Acknowledge bit from the client.
- If the requested length is more than 60 bytes, subsequent user bytes will be sent on the bus.
- When the user data length (being sent on the bus) reaches the requested length, the I$^2$C engine will send the "Stop" condition on the bus.

**TABLE 3-20: COMMAND STRUCTURE[1]**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x90 | I$^2$C Write Data – command code. |
| 1 | Low Byte | Requested I$^2$C transfer length – 16-bit value – low byte. |
| 2 | High Byte | Requested I$^2$C transfer length – 16-bit value – high byte. |
| 3 | I$^2$C Client Address | 8-bit value representing the I$^2$C client address to communicate with (even – address to write, odd – address to read) **(Note 2)**. |
| 4-63 | — | User data to be sent to the selected I$^2$C client device. |

**Note 1:** When the requested transfer length is more than 60 bytes, subsequent "I$^2$C Write Data" commands will transport the remainder of the user data (until the requested length).

**2:** The I$^2$C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

#### 3.1.5.1 Responses

**TABLE 3-21: RESPONSE 1 STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x90 | I$^2$C Write Data – command code echo. |
| 1 | 0x00 | Command completed successfully. |
| | 0x01 | I$^2$C engine is busy (command not completed). |
| 2 | — | Reserved. |
| 3-63 | Don't care | — |

# MCP2221A

### 3.1.6 I²C Write Data Repeated-START

This command is used to write user-given data to the I²C client device (the speed is specified by the STATUS/SET Parameters command).

The command will have the following effects:

• The I²C engine will send the "Repeated Start" condition.

• The selected I²C client address is sent next and the I²C engine will wait for the client to send an Acknowledge bit.

• The user data follow next and the I²C engine waits for the Acknowledge bit from the client.

• If the requested length is more than 60 bytes, subsequent user bytes will be sent on the bus.

• When the user data length (being sent on the bus) reaches the requested length, the I²C engine will send the "Stop" condition on the bus.

**TABLE 3-22: COMMAND STRUCTURE [1]**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x92 | I²C Write Data Repeated-START – command code (cI2C_CMD_RSTART_WRDATA7). |
| 1 | Low Byte | Requested I²C transfer length – 16-bit value – low byte. |
| 2 | High Byte | Requested I²C transfer length – 16-bit value – high byte. |
| 3 | I²C Client Address | 8-bit value representing the I²C client address to communicate with (even – address to write, odd – address to read) **(Note 2)**. |
| 4-63 | — | User data to be sent to the selected I²C client device |

**Note 1:** When the requested transfer length is more than 60 bytes, subsequent "I²C Write Data Repeated-START" commands will transport the remainder of the user data (till the requested length).

**2:** The I²C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

#### 3.1.6.1 Responses

**TABLE 3-23: RESPONSE 1 STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x92 | I²C Write Data Repeated-START – command code echo (cI2C_CMD_RSTART_WRDATA7). |
| 1 | 0x00 | Command completed successfully. |
| | 0x01 | I²C engine is busy (command not completed). |
| 2 | 0x10 | I²C engine sent START condition. |
| | 0x40 [1] | I²C engine start the write user data to client procedure |
| 3-63 | Don't care | — |

**Note 1:** This value applies only if Byte Index 1 is 0x00.

### 3.1.7    I$^2$C Write Data No STOP

This command is used to write user-given data to the I$^2$C client device (the speed is specified by the `STATUS/SET Parameters` command).

The command will have the following effects:

- The I$^2$C engine will send the "Start" condition.
- The selected I$^2$C client address is sent next and the I$^2$C engine will wait for the client to send an Acknowledge bit.

- The user data follow next and the I$^2$C engine waits for the Acknowledge bit from the client.
- If the requested length is more than 60 bytes, subsequent user bytes will be sent on the bus.
- When the user data length (being sent on the bus) reaches the requested length, the I$^2$C engine will **not** send the "Stop" condition on the bus.

**TABLE 3-24:    COMMAND STRUCTURE [1]**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x94 | I$^2$C Write Data No STOP – command code. |
| 1 | Low Byte | Requested I$^2$C transfer length – 16-bit value – low byte. |
| 2 | High Byte | Requested I$^2$C transfer length – 16-bit value – high byte. |
| 3 | I$^2$C Client Address | 8-bit value representing the I$^2$C client address to communicate with (even – address to write, odd – address to read) **(Note 2)**. |
| 4-63 | — | User data to be sent to the selected I$^2$C client device. |

**Note 1:**  When the requested transfer length is more than 60 bytes, subsequent "I$^2$C Write Data No STOP" commands will transport the remainder of the user data (till the requested length).

**2:**  The I$^2$C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

### 3.1.7.1    Responses

**TABLE 3-25:    RESPONSE 1 STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x94 | I$^2$C Write Data No STOP – command code echo. |
| 1 | 0x00 | Command completed successfully. |
|  | 0x01 | I$^2$C engine is busy (command not completed). |
| 2 | — | Reserved. |
| 3-63 | Don't care | — |

# MCP2221A

### 3.1.8    I²C Read Data

This command is used to read user-given data to the I²C client device (the speed is specified by the `STATUS/SET Parameters` command).

The command will have the following effects:

• The I²C engine will send the "Start" condition.

• The selected I²C client address is sent next and the I²C engine will wait for the client to send an Acknowledge bit.

• The user data are read next and the I²C engine sends the Acknowledge bit to the client.

• If the requested length is more than 60 bytes, subsequent user bytes will be read from the I²C client on the bus.

• When the user data length (being sent on the bus) reaches the requested length, the I²C engine will send the "Stop" condition on the bus.

**TABLE 3-26:    COMMAND STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x91 | `I²C Read Data` – command code. |
| 1 | Low Byte | Requested I²C transfer length – 16-bit value – low byte. |
| 2 | High Byte | Requested I²C transfer length – 16-bit value – high byte. |
| 3 | I²C Client Address | 8-bit value representing the I²C client address to communicate with (even – address to write, odd – address to read) **(Note 1)**. |
| 4-63 | Don't care | — |

**Note 1:**    The I²C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

#### 3.1.8.1    Responses

**TABLE 3-27:    RESPONSE 1 STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x91 | `I²C Read Data` – command code echo. |
| 1 | 0x00 | Command completed successfully. |
|  | 0x01 | I²C engine is busy (command not completed). |
| 2 | — | Reserved. |
| 3-63 | Don't care | — |

### 3.1.9    I$^2$C Read Data Repeated-START

This command is used to read user-given data to the I$^2$C client device (the speed is specified by the `STATUS/SET Parameters` command).

The command will have the following effect:

- The I$^2$C engine will send the "Repeated-START" condition.
- The selected I$^2$C client address is sent next and the I$^2$C engine will wait for the client to send an Acknowledge bit.

- The user data are read next and the I$^2$C engine sends the Acknowledge bit to the client.
- If the requested length is more than 60 bytes, subsequent user bytes will be read from the I$^2$C client on the bus.
- When the user data length (being sent on the bus) reaches the requested length, the I$^2$C engine will send the "Stop" condition on the bus.

**TABLE 3-28:    COMMAND STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x93 | I$^2$C Read Data Repeated-START – command code. |
| 1 | Low Byte | Requested I$^2$C transfer length – 16-bit value – low byte. |
| 2 | High Byte | Requested I$^2$C transfer length – 16-bit value – high byte. |
| 3 | I$^2$C Client Address | 8-bit value representing the I$^2$C client address to communicate with (even – address to write, odd – address to read) **(Note 1)**. |
| 4-63 | Don't care | — |

**Note  1:**    The I$^2$C client address is represented on 8 bits, with even values for writes and odd for reads. To get the 8-bit address value out of a 7-bit address, the 7-bit value needs to be shifted left by one position. For write operations, use the shifted value; while for reads, add one to the shifted value.

### 3.1.9.1    Responses

**TABLE 3-29:    RESPONSE 1 STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x93 | I$^2$C Read Data Repeated-START – command code echo. |
| 1 | 0x00 | Command completed successfully. |
|  | 0x01 | I$^2$C engine is busy (command not completed). |
| 2 | — | Reserved. |
| 3-63 | Don't care | — |

### 3.1.10 I²C Read Data – Get I²C Data

This command is used to read back the data from the I²C client device.

**TABLE 3-30: COMMAND STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x40 | I²C Read Data – Get I²C Data – command code. |
| 1-63 | Don't care | — |

#### 3.1.10.1 Responses

**TABLE 3-31: RESPONSE 1 STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x40 | I²C Read Data – Get I²C Data – command code echo. |
| 1 | 0x00 | Command completed successfully. |
|  | 0x41 | Error reading the I²C client data from the I²C engine. |
| 2 | — | Reserved. |
| 3 | 0-60 | The number of read-back data bytes to follow in this packet: from 0 to a maximum of 60 bytes of read-back bytes. |
|  | 127 | This value is signaled when an error has occurred and the following data should not be taken into account. |
| 4-63 | — | User data or Don't care. |

### 3.1.11 Set GPIO Output Values

This command is used to change the GPIO output value for those GP pins assigned for GPIO operation (GPIO outputs).

**TABLE 3-32: COMMAND STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0x50 | Set GPIO Output Values – command code. |
| 1 | — | Don't care | — |
| 2 | Alter GP0 output (enable/disable) | 0x00 | Do not modify GP0 output (if GP0 is set as GPIO output). |
| | | Any other value | The next byte (Index 3) will be the value used to set GP0 output (only if GP0 is set for GPIO output). |
| 3 | GP0 output value | 0x00 | GP0 (if set up for GPIO output operation) will take a logical value of '0'. |
| | | Any other value | GP0 (if set up for GPIO output operation) will take a logical value of '1'. |
| 4 | Alter GP0 pin direction (enable/disable) | 0x00 | Leave the GP0 GPIO designation as is (input or output). |
| | | Any other value | The next byte (index 5) will be the value used to set GP0's pin direction (only if GP0 is set for GPIO operation). |
| 5 | GP0 pin direction (input or output) | 0x00 | Set GP0 GPIO as output. |
| | | Any other value | GP0 (if set up for GPIO operation) will be set as a digital input. |
| 6 | Alter GP1 output (enable/disable) | 0x00 | Do not modify GP1 output (if GP1 is set as GPIO output). |
| | | Any other value | The next byte (Index 7) will be the value used to set GP1 output (only if GP1 is set for GPIO output). |
| 7 | GP1 output value | 0x00 | GP1 (if set up for GPIO output operation) will take a logical value of '0'. |
| | | Any other value | GP1 (if set up for GPIO output operation) will take a logical value of '1'. |
| 8 | Alter GP1 pin direction (enable/disable) | 0x00 | Leave the GP1 GPIO designation as is (input or output). |
| | | Any other value | The next byte (Index 9) will be the value used to set GP1's pin direction (only if GP1 is set for GPIO operation). |
| 9 | GP1 pin direction (input or output) | 0x00 | Set GP1 GPIO as output. |
| | | Any other value | GP1 (if set up for GPIO operation) will be set as a digital input. |
| 10 | Alter GP2 output (enable/disable) | 0x00 | Do not modify GP2 output (if GP2 is set as GPIO output). |
| | | Any other value | The next byte (Index 11) will be the value used to set GP2 output (only if GP2 is set for GPIO output). |
| 11 | GP2 output value | 0x00 | GP2 (if GP2 is set up for GPIO output operation) will take a logical value of '0'. |
| | | Any other value | GP2 (if GP2 is set up for GPIO output operation) will take a logical value of '1'. |
| 12 | Alter GP2 pin direction (enable/disable) | 0x00 | Leave the GP2 GPIO designation as is (input or output). |
| | | Any other value | The next byte (Index 13) will be the value used to set GP2's pin direction (only if GP2 is set for GPIO operation). |
| 13 | GP2 pin direction (input or output) | 0x00 | Set GP2 GPIO as output. |
| | | Any other value | GP2 (if set up for GPIO operation) will be set as a digital input. |

**TABLE 3-32: COMMAND STRUCTURE (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 14 | Alter GP3 output (enable/disable) | 0x00 | Do not modify GP3 output (if GP3 is set as GPIO output). |
| | | Any other value | The next byte (Index 11) will be the value used to set GP3 output (only if GP3 is set for GPIO output). |
| 15 | GP3 output value | 0x00 | GP3 (if set up for GPIO output operation) will take a logical value of '0'. |
| | | Any other value | GP3 (if set up for GPIO output operation) will take a logical value of '1'. |
| 16 | Alter GP3 pin direction (enable/disable) | 0x00 | Leave the GP3 GPIO designation as is (input or output). |
| | | Any other value | The next byte (Index 17) will be the value used to set GP3's pin direction (only if GP3 is set for GPIO operation) |
| 17 | GP3 pin direction (input or output) | 0x00 | Set GP3 GPIO as output. |
| | | Any other value | GP3 (if set up for GPIO operation) will be set as a digital input. |
| 18-63 | Reserved | 0x00 | — |

### 3.1.11.1 Responses

**TABLE 3-33: RESPONSE 1 STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0x50 | `Set GPIO Output Values` – command code. |
| 1 | — | 0x00 | Command completed successfully. |
| 2 | Alter GP0 output (enable/disable) status | 0xEE | If GP0 is not set for GPIO operation. |
| | | Any other value | If GP0 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 3 | GP0 output value status | 0xEE | If GP0 is not set for GPIO operation. |
| | | Any other value | If GP0 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 4 | Alter GP0 pin direction (enable/disable) | 0xEE | If GP0 is not set for GPIO operation. |
| | | Any other value | If GP0 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 5 | GP1 pin direction (input or output) | 0xEE | If GP1 is not set for GPIO operation. |
| | | Any other value | If GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 6 | Alter GP1 output (enable/disable) status | 0xEE | If GP1 is not set for GPIO operation. |
| | | Any other value | If GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 7 | GP1 output value status | 0xEE | If GP1 is not set for GPIO operation |
| | | Any other value | If GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |

**TABLE 3-33: RESPONSE 1 STRUCTURE (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 8 | Alter GP1 pin direction (enable/disable) | 0xEE | If GP1 is not set for GPIO operation. |
| | | Any other value | If the GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 9 | GP1 pin direction (input or output) | 0xEE | If GP1 is not set for GPIO operation. |
| | | Any other value | If GP1 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 10 | Alter GP2 output (enable/disable) status | 0xEE | If GP2 is not set for GPIO operation. |
| | | Any other value | If GP2 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 11 | GP2 output value status | 0xEE | If GP2 is not set for GPIO operation. |
| | | Any other value | If GP2 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 12 | Alter GP2 pin direction (enable/disable) | 0xEE | If GP2 is not set for GPIO operation. |
| | | Any other value | If GP2 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 13 | GP2 pin direction (input or output) | 0xEE | If GP2 is not set for GPIO operation. |
| | | Any other value | If GP2 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 14 | Alter GP3 output (enable/disable) status | 0xEE | If GP3 is not set for GPIO operation. |
| | | Any other value | If GP3 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 15 | GP3 output value status | 0xEE | If GP3 is not set for GPIO operation. |
| | | Any other value | If GP3 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 16 | Alter GP3 pin direction (enable/disable) | 0xEE | If GP3 is not set for GPIO operation. |
| | | Any other value | If GP3 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 17 | GP3 pin direction (input or output) | 0xEE | If GP3 is not set for GPIO operation. |
| | | Any other value | If GP3 is already set for GPIO operation, the value will be copied from the same byte index in the command structure. |
| 18-63 | — | Don't care | — |

# MCP2221A

### 3.1.12    Get GPIO Values

This command is used to retrieve the GPIO direction and pin value for those GP pins assigned for GPIO operation (GPIO inputs or outputs).

**TABLE 3-34:    COMMAND STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x51 | `Get GPIO Values` – command code. |
| 1-63 | Don't care | — |

### 3.1.12.1    Responses

**TABLE 3-35:    RESPONSE 1 STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0x51 | `Get GPIO Values` – command code. |
| 1 | — | 0x00 | Command completed successfully. |
| 2 | GP0 pin value | 0xEE | If GP0 is not set for GPIO operation. |
| | | Other values (0x00 or 0x01) | If GP0 is already set for GPIO operation, the value represents the GP0 logic pin value. |
| 3 | GP0 direction value | 0xEF | If GP0 is not set for GPIO operation. |
| | | Other values (0x00 or 0x01) | If GP0 is already set for GPIO operation, the value represents the GP0 pin designation (0x00 for output and 0x01 for input). |
| 4 | GP1 pin value | 0xEE | If GP1 is not set for GPIO operation. |
| | | Other values (0x00 or 0x01) | If GP1 is already set for GPIO operation, the value represents the GP1 logic pin value. |
| 5 | GP1 direction value | 0xEF | If GP1 is not set for GPIO operation. |
| | | Other values (0x00 or 0x01) | If GP1 is already set for GPIO operation, the value represents the GP1 pin designation (0x00 for output and 0x01 for input). |
| 6 | GP2 pin value | 0xEE | If GP2 is not set for GPIO operation. |
| | | Other values (0x00 or 0x01) | If GP2 is already set for GPIO operation, the value represents the GP2 logic pin value. |
| 7 | GP2 direction value | 0xEF | If GP2 is not set for GPIO operation. |
| | | Other values (0x00 or 0x01) | If GP2 is already set for GPIO operation, the value represents the GP2 pin designation (0x00 for output and 0x01 for input). |
| 8 | GP3 pin value | 0xEE | If GP3 is not set for GPIO operation |
| | | Other values (0x00 or 0x01) | If GP3 is already set for GPIO operation, the value represents the GP3 logic pin value. |
| 9 | GP3 direction value | 0xEF | If GP3 is not set for GPIO operation. |
| | | Other values (0x00 or 0x01) | If GP3 is already set for GPIO operation, the value represents the GP3 pin designation (0x00 for output and 0x01 for input). |
| 10-63 | — | Don't care | — |

### 3.1.13 Set SRAM settings

This command is used to alter various run-time Chip settings. The altered settings reside in SRAM memory and they will not affect the Chip's power-up/Reset default settings. These altered settings will be active until the next chip power-up/Reset.

**TABLE 3-36:** **COMMAND STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0x60 | `Set SRAM settings` – command code. |
| 1 | — | Don't care | — |
| 2 | **Clock Output Divider Value** – this allows the user to modify the clock output value on-the-fly at run time | | |
| | Bit 7: Enable loading of a new clock divider | 1 | Bits[4:0] will be loaded into the clock divider. |
| | | 0 | Clock divider value won't be altered. |
| | Bits 6-5 | Don't care | — |
| | Bits 4-3: Duty cycle | 00 | 0% duty cycle. |
| | | 01 | 25% duty cycle. |
| | | 10 | 50% duty cycle. |
| | | 11 | 75% duty cycle. |
| | Bits 2-0: Clock divider value | — | — |
| 3 | **DAC Voltage Reference** – this allows the user to modify the DAC reference voltage | | |
| | Bit 7: Enable loading of a new DAC reference | 1 | Bits[2:0] will be used for DAC reference voltage selection. |
| | | 0 | DAC reference will remain unaltered. |
| | Bits 6-3 | Don't care | — |
| | Bits 2-1: DAC $V_{RM}$ voltage selection; these bits are used to change the DAC $V_{RM}$ voltage | 11 | $V_{RM}$ voltage is 4.096V (only if $V_{DD}$ is higher than this value). |
| | | 10 | $V_{RM}$ voltage is 2.048V. |
| | | 01 | $V_{RM}$ voltage is 1.024V. |
| | | 00 | $V_{RM}$ voltage is off. |
| | Bit 0: This bit is used to change the DAC reference voltage | 1 | DAC voltage reference is the internal DAC voltage reference module (DAC $V_{RM}$). |
| | | 0 | DAC voltage reference is $V_{DD}$. |
| 4 | **Set DAC Output Value** | | |
| | Bit 7: Enable loading of a new DAC value | 1 | Bits[4:0] will be used for DAC reference voltage selection. |
| | | 0 | The current DAC value will remain unaltered. |
| | Bits 6-5 | Don't care | — |
| | Bits 4-0: The new DAC value | — | — |

**TABLE 3-36:    COMMAND STRUCTURE (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 5 | **ADC Voltage Reference** – this allows the user to modify the ADC reference voltage | | |
| | Bit 7: Enable loading of a new ADC reference | 1 | Bits[2:0] will be used for ADC reference voltage selection. |
| | | 0 | ADC reference will remain unaltered. |
| | Bits 6-3 | Don't care | — |
| | Bits 2-1: These bits are used to change the DAC VRM voltage | 11 | VRM voltage is 4.096V (only if VDD is higher than this value). |
| | | 10 | VRM voltage is 2.048V. |
| | | 01 | VRM voltage is 1.024V. |
| | | 00 | VRM voltage is off. |
| | Bit 0: This bit is used to change the DAC reference voltage | 1 | VDD ADC voltage reference is the internal ADC voltage reference module (ADC VRM). |
| | | 0 | ADC voltage reference is VDD. |
| 6 | **Set Up the Interrupt Detection Mechanism and Clear the Detection Flag** – useful for preparing the interrupt detection module to detect a new interrupt condition | | |
| | Bit 7: Enable the modification of the interrupt detection conditions | 1 | The interrupt detection settings and flag will change. |
| | | 0 | The interrupt detection settings and flag will remain unchanged. |
| | Bits 6-5 | Don't care | — |
| | Bit 4 | — | Enable the modification of the positive edge detection. |
| | Bit 3: The new value for the positive edge detector | 1 | Interrupt detection will trigger on positive edges. |
| | | 0 | Interrupt detection will not trigger on positive edges. |
| | Bit 2 | — | Enable the modification of the negative edge detection. |
| | Bit 1: The new value for the negative edge detector | 1 | Interrupt detection will trigger on negative edges. |
| | | 0 | Interrupt detection will not trigger on negative edges. |
| | Bit 0: Clear the interrupt detection flag | 1 | Clear the interrupt detection flag. |
| | | 0 | Leave the interrupt detection flag as is. |
| 7 | Bit 7: Alter GPIO configuration: alters the current GP designation | 1 | Alter the GP designation. The values from Byte Index 8 will be used to load a new set of values into the SRAM GP settings. |
| | | 0 | Do not alter the current GP designation. |
| 8 | **GP0 Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP0 is set as an output GPIO, this value will be present at the GP0 pin |
| | Bit 3: GPIO direction (input/output) – works only when GP0 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP0 designation | 111–011 | Don't care. |
| | | 010 | Alternate Function 0 (LED_URx). |
| | | 001 | Dedicated function operation (SSPND). |
| | | 000 | GPIO operation. |

**TABLE 3-36: COMMAND STRUCTURE (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 9 | **GP1 Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP1 is set as an output GPIO, this value will be present at the GP1 pin. |
| | Bit 3: GPIO direction (input/output) – works only when GP1 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP0 designation | 111–101 | Don't care. |
| | | 100 | Alternate Function 2 (interrupt detection). |
| | | 011 | Alternate Function 1 (LED_UTx). |
| | | 010 | Alternate Function 0 (ADC1). |
| | | 001 | Dedicated function operation (clock output). |
| | | 000 | GPIO operation. |
| 10 | **GP2 Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP2 is set as an output GPIO, this value will be present at the GP2 pin. |
| | Bit 3: GPIO direction (input/output) – works only when GP2 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP2 designation | 111–100 | Don't care. |
| | | 011 | Alternate Function 1 (DAC1). |
| | | 010 | Alternate Function 0 (ADC2). |
| | | 001 | Dedicated function operation (USBCFG). |
| | | 000 | GPIO operation. |
| 11 | **GP3 Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When GP3 is set as an output GPIO, this value will be present at the GP3 pin. |
| | Bit 3: GPIO direction (input/output) – works only when GP3 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP3 Designation | 111–100 | Don't care. |
| | | 011 | Alternate Function 1 (DAC2). |
| | | 010 | Alternate Function 0 (ADC3). |
| | | 001 | Dedicated function operation (LED_I2C). |
| | | 000 | GPIO operation. |
| 12-63 | Reserved | 0x00 | — |

### 3.1.13.1 Responses

**TABLE 3-37: RESPONSE 1 STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x60 | Set SRAM settings – command code echo. |
| 1 | 0x00 | Command completed successfully. |
| 2-63 | Don't care | — |

# MCP2221A

### 3.1.14 Get SRAM Settings

This command is used to retrieve the run-time Chip and
GP settings.

**TABLE 3-38: COMMAND STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x61 | Get SRAM Settings – command code echo. |
| 1-63 | 0x00 | Command completed successfully. |

### 3.1.14.1 Responses

**TABLE 3-39: RESPONSE 1 STRUCTURE**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 0 | — | 0x61 | Get SRAM Settings – command code echo. |
| 1 | — | 0x00 | Command completed successfully. |
| 2 | Length in bytes of the SRAM Chip settings area | — | — |
| 3 | Length in bytes of the SRAM GP settings area | — | — |
| 4 | Bit 7: CDC serial number enumeration enable | 1 | The USB serial number will be used during the USB enumeration of the CDC interface. |
| | | 0 | No serial number descriptor will be presented during the USB enumeration. |
| | Bits 6-2:Not implemented | — | — |
| | Bits 1-0: Chip configuration security option | 10 | Permanently locked. |
| | | 01 | Password-protected. |
| | | 00 | Unsecured. |
| 5 | Bits 7-5 | Don't care | — |
| | Bits 4-0: Clock Output divider value | — | If the GP pin (exposing the clock output) is enabled for clock output operation, the divider value will be used on the 48 MHz USB internal clock and its divided output will be sent to this pin. (Bits[4:3] for duty cycle and bits[2:0] for the clock divider.) |
| 6 | Bits 7-6: DAC reference voltage option | 11 | Reference voltage is 4.096V. |
| | | 10 | Reference voltage is 2.048V. |
| | | 01 | Reference voltage is 1.024V. |
| | | 00 | Reference voltage is off (this is useful for the case in which the DAC uses another reference other than $V_{RM}$ DAC; e.g., $V_{DD}$). |
| | Bit 5: DAC reference option | 1 | DAC reference is $V_{RM}$ DAC voltage. |
| | | 0 | DAC reference is $V_{DD}$. |
| | Bits 4-0: Power-up DAC value | — | — |

**TABLE 3-39:    RESPONSE 1 STRUCTURE (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 7 | Bit 7 | Don't care | — |
| | Bit 6: Interrupt detection – negative edge | — | If set, the interrupt detection flag will be set when a negative edge occurs. |
| | Bit 5: Interrupt detection – positive edge | — | If set, the interrupt detection flag will be set when a positive edge occurs |
| | Bits 4-3: ADC reference voltage | 11 | Reference voltage is 4.096V (only if VDD is above this voltage). |
| | | 10 | Reference voltage is 2.048V. |
| | | 01 | Reference voltage is 1.024V. |
| | | 00 | Reference voltage is off (this is useful for the case in which the ADC uses another reference other than VRM DAC; e.g., VDD). |
| | Bit 2: ADC reference option | 1 | ADC reference is VRM ADC. |
| | | 0 | ADC reference is VDD. |
| | Bit 1 | Don't care | — |
| | Bit 2 | Don't care | — |
| 8 | Lower byte of the 16-bit USB VID value | — | — |
| 9 | Higher byte of the 16-bit USB VID value | — | — |
| 10 | Lower byte of the 16-bit USB PID value | — | — |
| 11 | Higher byte of the 16-bit USB PID value | — | — |
| 12 | USB power attributes | — | This value will be used by the MCP2221A's USB Configuration Descriptor (power attributes value) during the USB enumeration. |
| 13 | USB requested number of mA(s) | — | The requested mA value during the USB enumeration will represent the value at this index multiplied by two. |
| 14 | Current Supplied Password Byte 1 | — | — |
| 15 | Current Supplied Password Byte 2 | — | — |
| 16 | Current Supplied Password Byte 3 | — | — |
| 17 | Current Supplied Password Byte 4 | — | — |
| 18 | Current Supplied Password Byte 5 | — | — |
| 19 | Current Supplied Password Byte 6 | — | — |
| 20 | Current Supplied Password Byte 7 | — | — |
| 21 | Current Supplied Password Byte 8 | — | — |

**TABLE 3-39:    RESPONSE 1 STRUCTURE (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 22 | **GP0 Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO Output value | — | When the GP0 is set as an output GPIO, this value will be present at the GP0 pin. |
| | Bit 3: GPIO Direction (Input/Output) – Works only when GP0 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP0 Designation | 111–011 | Don't care. |
| | | 010 | Alternate Function 0 (LED_URx). |
| | | 001 | Dedicated function operation (SSPND). |
| | | 000 | GPIO operation. |
| 23 | **GP1 Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO Output value | — | When the GP1 is set as an output GPIO, this value will be present at the GP1 pin. |
| | Bit 3: GPIO Direction (Input/Output) – Works only when GP1 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP1 Designation | 111–101 | Don't care. |
| | | 100 | Alternate Function 2 (interrupt detection). |
| | | 011 | Alternate Function 1 (LED_UTx). |
| | | 010 | Alternate Function 0 (ADC1). |
| | | 001 | Dedicated function operation (clock output). |
| | | 000 | GPIO operation. |
| 24 | **GP2 Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When the GP2 is set as an output GPIO, this value will be present at the GP2 pin. |
| | Bit 3: GPIO direction (input/output) – works only when GP2 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP2 Designation | 111–100 | Don't care. |
| | | 011 | Alternate Function 1 (DAC1). |
| | | 010 | Alternate Function 0 (ADC2). |
| | | 001 | Dedicated function operation (USBCFG). |
| | | 000 | GPIO operation. |

**TABLE 3-39: RESPONSE 1 STRUCTURE (CONTINUED)**

| Byte Index | Function Description | Value | Effect |
|---|---|---|---|
| 25 | **GP3 Settings** | | |
| | Bits 7-5 | Don't care | — |
| | Bit 4: GPIO output value | — | When the GP3 is set as an output GPIO, this value will be present at the GP3 pin. |
| | Bit 3: GPIO direction (input/output) – works only when GP3 is set for GPIO operation | 1 | GPIO Input mode. |
| | | 0 | GPIO Output mode. |
| | Bits 2-0: GP3 designation | 111-100 | Don't care. |
| | | 011 | Alternate Function 1 (DAC2). |
| | | 010 | Alternate Function 0 (ADC3). |
| | | 001 | Dedicated function operation (LED_I2C). |
| | | 000 | GPIO operation. |
| 26-63 | | Don't care | — |

## 3.1.15 Reset Chip

This command is used to force a Reset of the MCP2221A device. This command is useful when the Flash memory is updated with new data. The MCP2221A would need to be re-enumerated to see the new data.

> **Note:** This command is the only command that does not expect a response.

**TABLE 3-40: COMMAND STRUCTURE**

| Byte Index | Value | Effect |
|---|---|---|
| 0 | 0x70 | Reset Chip – command code. |
| 1 | 0xAB | — |
| 2 | 0xCD | — |
| 3 | 0xEF | — |
| 4-63 | 0x00 | Reserved. |

# MCP2221A

## 4.0    ELECTRICAL CHARACTERISTICS

### Absolute Maximum Ratings†(1)

Ambient temperature under bias ..................................................................................................-40°C to +85°C

Storage temperature ............................................................................................................... -65°C to +150°C

Voltage on $V_{DD}$ with respect to $V_{SS}$  .......................................................................................... -0.3V to +6.0V

Voltage on $\overline{RST}$ with respect to $V_{SS}$ ................................................................................................-0.3V to +9.0V

Voltage on $V_{USB}$ pin with respect to $V_{SS}$ ........................................................................................ -0.3V to +4.0V

Voltage on D+ and D- pins with respect to $V_{SS}$ ......................................................... -0.3V to ($V_{USB}$ + 0.3V)

Voltage on all other pins with respect to $V_{SS}$  ..............................................................-0.3V to ($V_{DD}$ + 0.3V)

Total power dissipation(2) ......................................................................................................................800 mW

Maximum current out of $V_{SS}$ pin ........................................................................................................ 95 mA

Maximum current into $V_{DD}$ pin ........................................................................................................... 95 mA

Clamp current, $I_K$ ($V_{PIN}$ < 0 or $V_{PIN}$ > $V_{DD}$)......................................................................................±20 mA

Maximum output current sunk by any I/O pin........................................................................................25 mA

Maximum output current sourced by any I/O pin...................................................................................25 mA

Maximum current sunk by all ports........................................................................................................90 mA

Maximum current sourced by all ports ................................................................................................. 90 mA

---

**†   NOTICE:** Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure above maximum rating conditions for extended periods may affect device reliability.

---

**Note  1:**   $V_{USB}$ must always be ≤ $V_{DD}$ + 0.3V.

**2:**   Power dissipation is calculated as follows: $P_{DIS} = V_{DD}$ x {$I_{DD} - \sum I_{OH}$} + $\sum$ {($V_{DD} - V_{OH}$) x $I_{OH}$} + $\sum$($V_{OL}$ x $I_{OL}$).

## 4.1 DC CHARACTERISTICS

| DC Characteristics | | Operating Conditions (unless otherwise indicated): $3.0V \leq V_{DD} \leq 5.5V$ at $-40°C \leq T_A \leq +85°C$ (I-Temp) | | | | | |
|---|---|---|---|---|---|---|---|
| Param. No. | Characteristic | Sym. | Min. | Typ. | Max. | Units | Conditions |
| D001 | Supply Voltage | $V_{DD}$ | 3.0 | — | 5.5 | V | |
| | POR Release Voltage | $V_{POR}$ | — | 1.6 | — | V | |
| | POR Rearm Voltage | — | — | 0.8 | — | V | |
| D003 | $V_{DD}$ Rise Rate to Ensure POR | $SV_{DD}$ | 0.05 | — | — | V/ms | Design guidance only, not tested |
| D004 | Supply Current | $I_{DD}$ | — | — | — | — | |
| | $V_{DD}$ = 3.0V | | — | — | 10 | 12 | mA | $F_{OSC}$ = 12 MHz (330 nF on $V_{USB}$) |
| | $V_{DD}$ = 5.0V | | — | — | 13 | 15 | mA | |
| D005 | Standby Current | $I_{DDS}$ | — | 46 | — | µA | |
| **Input Low Voltage** | | | | | | | |
| D031 | Schmitt Trigger (URx) | $V_{IL}$ | — | — | $0.2 V_{DD}$ | V | $3.0V \leq V_{DD} \leq 5.5V$ |
| | TTL (GP pins) | | — | — | 0.8 | | $4.5V \leq V_{DD} \leq 5.5V$ |
| **Input High Voltage** | | | | | | | |
| D041 | Schmitt Trigger (URx) | $V_{IH}$ | $0.8 V_{DD}$ | — | $V_{DD}$ | V | $3.0V \leq V_{DD} \leq 5.5V$ |
| | TTL (GP pins) | | 2.0 | — | $V_{DD}$ | | $4.5V \leq V_{DD} \leq 5.5V$ |
| **Input Leakage Current** | | | | | | | |
| D060 | GP, URx | $I_{IL}$ | — | ±50 | ±100 | nA | $V_{SS} \leq V_{PIN} \leq V_{DD}$, pin at high-Z |
| **Output Low Voltage** | | | | | | | |
| D080 | GP, UTx | $V_{OL}$ | — | — | 0.6 | V | $I_{OL}$ = 8.0 mA, $V_{DD}$ = 5.0V |
| | | | — | — | 0.6 | | $I_{OL}$ = 6.0 mA, $V_{DD}$ = 3.3V |
| **Output High Voltage** | | | | | | | |
| D090 | GP, UTx | $V_{OH}$ | $V_{DD}$ – 0.7 | — | — | V | $I_{OH}$ = -3.5 mA, $V_{DD}$ = 5.0V |
| | | | $V_{DD}$ – 0.7 | — | — | | $I_{OH}$ = -3.0 mA, $V_{DD}$ = 3.3V |
| **Capacitive Loading Specs on Output Pins** | | | | | | | |
| D102 | GPIO | $C_{IO}$ | — | — | 50 | pF | **Note 1** |

**Note 1:** Characterized only, not 100% tested.

# MCP2221A

**FIGURE 4-1:** POR AND POR REARM WITH SLOW RISING V$_{DD}$



**Note 1:** When NPOR is low, the device is held in Reset.
   **2:** T$_{POR}$ = 1 µs typical.
   **3:** T$_{VLOW}$ = 2.7 µs typical.
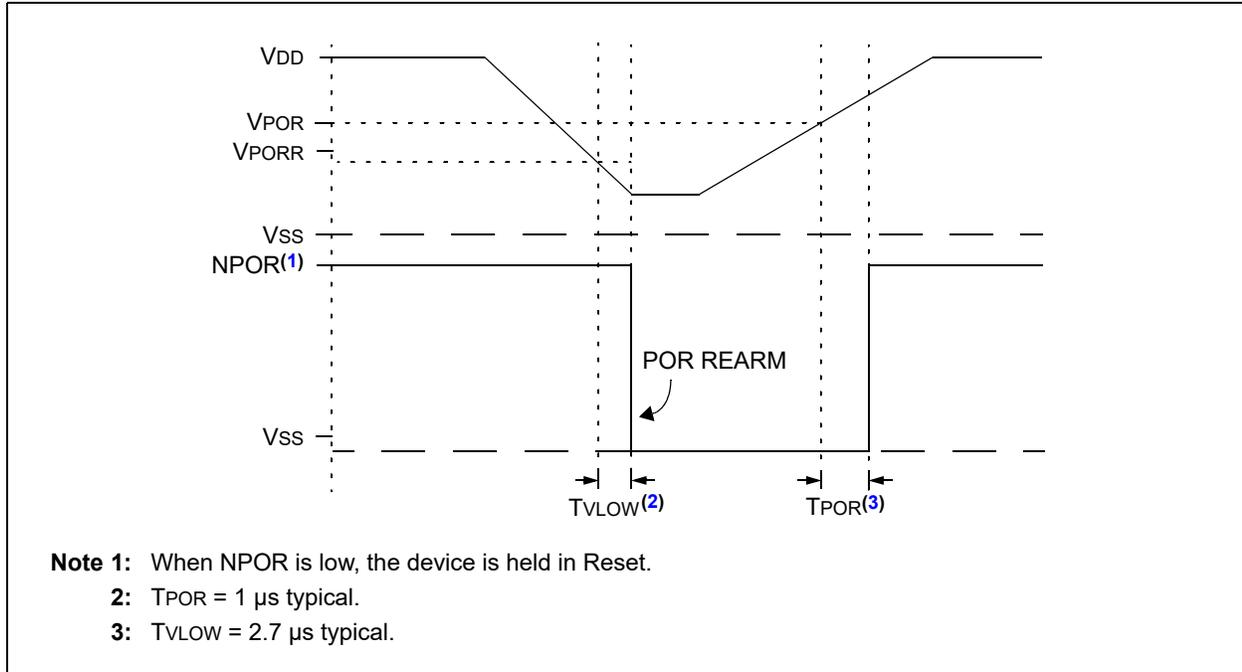
**TABLE 4-1:** USB MODULE SPECIFICATIONS

| DC Characteristics | | Operating Conditions (unless otherwise indicated): 3.0V ≤ V$_{DD}$ ≤ 5.5V at -40°C ≤ T$_A$ ≤ +85°C (I-Temp) | | | | | |
|---|---|---|---|---|---|---|---|
| Param. No. | Characteristic | Sym. | Min. | Typ. | Max. | Units | Conditions |
| D313 | USB Voltage | V$_{USB}$ | 3.0 | — | 3.6 | V | Voltage on the V$_{USB}$ pin must be in this range for proper USB operation |
| D314 | Input Leakage on Pin | I$_{IL}$ | — | — | ±1 | µA | V$_{SS}$ ≤ V$_{PIN}$ ≤ V$_{DD}$ pin at high-impedance |
| D315 | Input Low Voltage for USB Buffer | V$_{ILUSB}$ | — | — | 0.8 | V | For V$_{USB}$ range |
| D316 | Input High Voltage for USB Buffer | V$_{IHUSB}$ | 2.0 | — | — | V | For V$_{USB}$ range |
| D318 | Differential Input Sensitivity | V$_{DIFS}$ | — | — | 0.2 | V | The difference between D+ and D- must exceed this value while V$_{CM}$ is met |
| D319 | Differential Common-Mode Range | V$_{CM}$ | 0.8 | — | 2.5 | V | |
| D320 | Driver Output Impedance[1] | Z$_{OUT}$ | 28 | — | 44 | W | |
| D321 | Voltage Output Low | V$_{OL}$ | 0.0 | — | 0.3 | V | 1.5 kΩ load connected to 3.6V |
| D322 | Voltage Output High | V$_{OH}$ | 2.8 | — | 3.6 | V | 1.5 kΩ load connected to ground |

**Note 1:** The D+ and D- signal lines have been built-in impedance matching resistors. No external resistors, capacitors or magnetic components are necessary on the D+/D- signal paths between the MCP2221A family device and the USB cable.

**TABLE 4-2: A/D CONVERTER (ADC) CHARACTERISTICS**

| Standard Operating Conditions (unless otherwise stated): Operating Temperature Tested at +25°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param No. | Characteristic | Sym. | Min. | Typ. | Max. | Units | Conditions |
| AD01 | Resolution | $N_R$ | — | — | 10 | bit | |
| AD08 | Recommended Impedance of Analog Voltage Source | $Z_{AIN}$ | — | — | 10 | kΩ | Can go higher if external 0.01 µF capacitor is present on input pin |

**TABLE 4-3: DIGITAL-TO-ANALOG CONVERTER (DAC) SPECIFICATIONS**

| Operating Conditions: 3.0V < $V_{DD}$ < 5.5V, -40°C < $T_A$ < +85°C (unless otherwise stated) | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param No. | Characteristic | Sym. | Min. | Typ. | Max. | Units | Conditions |
| DAC01* | Step-Size | $C_{LSB}$ | — | $V_{DD}/32$ | — | V | |
| DAC02* | Absolute Accuracy | $C_{ACC}$ | — | — | ±1/2 | LSb | |
| DAC03* | Unit Resistor Value (R) | $C_R$ | — | 5K | — | Ω | |
| DAC04* | Settling Time[1] | $C_{ST}$ | — | — | 10 | µs | |

    * These parameters are characterized but not tested.

**Note 1:** Settling time measured while the DACVAL[4:0] bits transition from '0000' to '1111'.

**TABLE 4-4: THERMAL CONSIDERATIONS**

| Standard Operating Conditions (unless otherwise stated): Operating Temperature: -40°C ≤ $T_A$ ≤ +85°C (I-Temp) | | | | | |
|---|---|---|---|---|---|
| Param. No. | Characteristic | Sym. | Typ. | Units | Conditions |
| TH01 | Thermal Resistance Junction to Ambient | $\theta_{JA}$ | 70 | °C/W | 14-pin PDIP package |
| | | | 95.3 | °C/W | 14-pin SOIC package |
| | | | 100 | °C/W | 14-pin TSSOP package |
| | | | 45.7 | °C/W | 16-pin QFN 4 x 4 mm package |
| TH02 | Thermal Resistance Junction to Case | $\theta_{JC}$ | 32 | °C/W | 14-pin PDIP package |
| | | | 31 | °C/W | 14-pin SOIC package |
| | | | 24.4 | °C/W | 14-pin TSSOP package |
| | | | 6.3 | °C/W | 16-pin QFN 4 x 4 mm package |
| TH03 | Maximum Junction Temperature | $T_{JMAX}$ | +150 | °C | |
| TH04 | Power Dissipation | PD | — | W | PD = $P_{INTERNAL}$ + $P_{I/O}$ |
| TH05 | Internal Power Dissipation | $P_{INTERNAL}$ | — | W | $P_{INTERNAL}$ = $I_{DD}$ x $V_{DD}$[1] |
| TH06 | I/O Power Dissipation | $P_{I/O}$ | — | W | $P_{I/O}$ = Σ($I_{OL}$ x $V_{OL}$) + Σ($I_{OH}$ x ($V_{DD}$ – $V_{OH}$)) |
| TH07 | Derated Power | $P_{DER}$ | — | W | $P_{DER}$ = $PD_{MAX}$ ($T_J$ – $T_A$)/$\theta_{JA}$[2,3] |

**Note 1:** $I_{DD}$ is the current to run the device alone without driving any load on the output pins.

    **2:** $T_A$ = Ambient Temperature.

    **3:** $T_J$ = Junction Temperature.

## 4.2 AC Characteristics

### 4.2.1 TIMING PARAMETER SYMBOLOGY

The timing parameter symbols have been created in one of the following formats:

| 1. TppS2ppS | | 2. TppS | |
|---|---|---|---|
| T | | | |
| F | Frequency | T | Time |
| E | Error | | |

Lowercase letters (pp) and their meanings:

| pp | | | |
|---|---|---|---|
| io | Input or Output pin | osc | Oscillator |
| rx | Receive | tx | Transmit |
| bitclk | RX/TX BITCLK | RST | Reset |
| drt | Device Reset Timer | | |

Uppercase letters and their meanings:

| S | | | |
|---|---|---|---|
| F | Fall | P | Period |
| H | High | R | Rise |
| I | Invalid (high-impedance) | V | Valid |
| L | Low | Z | High-impedance |

### 4.2.2 TIMING CONDITIONS

The operating temperature and voltage specified in Table 4-5 apply to all timing specifications, unless otherwise noted. Figure 4-2 specifies the load conditions for the timing specifications.

**TABLE 4-5: TEMPERATURE AND VOLTAGE SPECIFICATIONS – AC**

| AC CHARACTERISTICS | **Standard Operating Conditions (unless otherwise stated):**<br>Operating temperature -40°C $\leq$ T$_A$ $\leq$ +85°C<br>Operating voltage V$_{DD}$ range as described in DC spec, **Section 4.1 "DC Characteristics"**. |
|---|---|

**FIGURE 4-2: LOAD CONDITIONS FOR DEVICE TIMING SPECIFICATIONS**
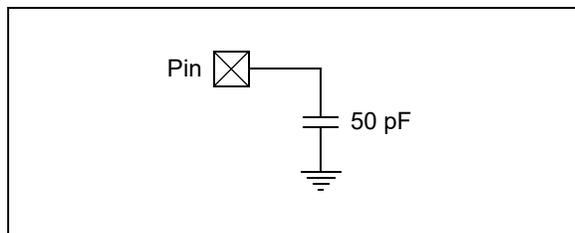
### 4.2.3 TIMING DIAGRAMS AND SPECIFICATIONS

**TABLE 4-6: RESET, OSCILLATOR START-UP TIMER AND POWER-UP TIMER PARAMETERS[1]**

| Standard Operating Conditions (unless otherwise stated): Operating Temperature: -40°C ≤ TA ≤ +85°C | | | | | | | |
|---|---|---|---|---|---|---|---|
| Param No. | Characteristic | Sym. | Min. | Typ.[2] | Max. | Units | Conditions |
| 30 | RST Pulse Width (low) | T<sub>RST</sub> | 2 | — | — | µs | |
| 31 | Power-up Timer | T<sub>PWRT</sub> | 40 | 65 | 140 | ms | |
| 32 | Oscillator Start-up Time | T<sub>OST</sub> | — | 1024 | — | T<sub>OST</sub> | |

**Note 1:** These parameters are characterized but not tested.

**2:** Data in the "Typ." column are at 5V, +25°C unless otherwise stated. These parameters are for design guidance only and are not tested.

# MCP2221A

## 5.0 PACKAGING INFORMATION

### 5.1 Package Marking Information

14-Lead PDIP (.300 in)

Example

XXXXXXXXXXXXXX
XXXXXXXXXXXXXX
YYWWNNN

MCP2221A
P(e3)
1448256
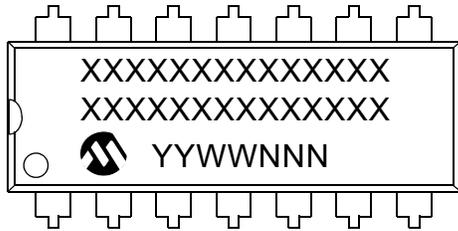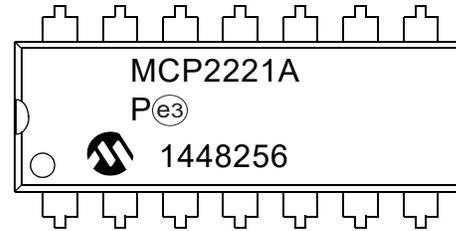
14-Lead SOIC (3.90 mm)

Example

XXXXXXXXXX
XXXXXXXXXX
YYWWNNN

MCP2221A
SL
1448256
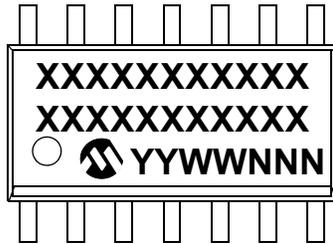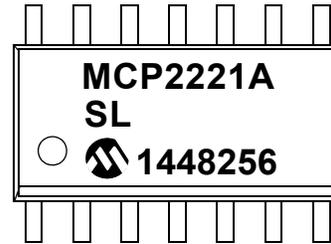
14-Lead TSSOP (4.4 mm)

Example

XXXXXXX
YYWW
NNN

2221AST
1448
256

16-Lead QFN (4x4x0.9 mm)

Example

PIN 1

XXXXX
XXXXX
XXXXX
YWWNNN

PIN 1

2221A
ML(e3)
703256

| **Legend:** | XX...X | Customer-specific information |
| --- | --- | --- |
| | Y | Year code (last digit of calendar year) |
| | YY | Year code (last 2 digits of calendar year) |
| | WW | Week code (week of January 1 is week '01') |
| | NNN | Alphanumeric traceability code |
| | (e3) | Pb-free JEDEC designator for Matte Tin (Sn) |
| | * | This package is Pb-free. The Pb-free JEDEC designator ((e3)) can be found on the outer packaging for this package. |

**Note**: In the event the full Microchip part number cannot be marked on one line, it will be carried over to the next line, thus limiting the number of available characters for customer-specific information.

## 14-Lead Plastic Dual In-Line (P) – 300 mil Body [PDIP]

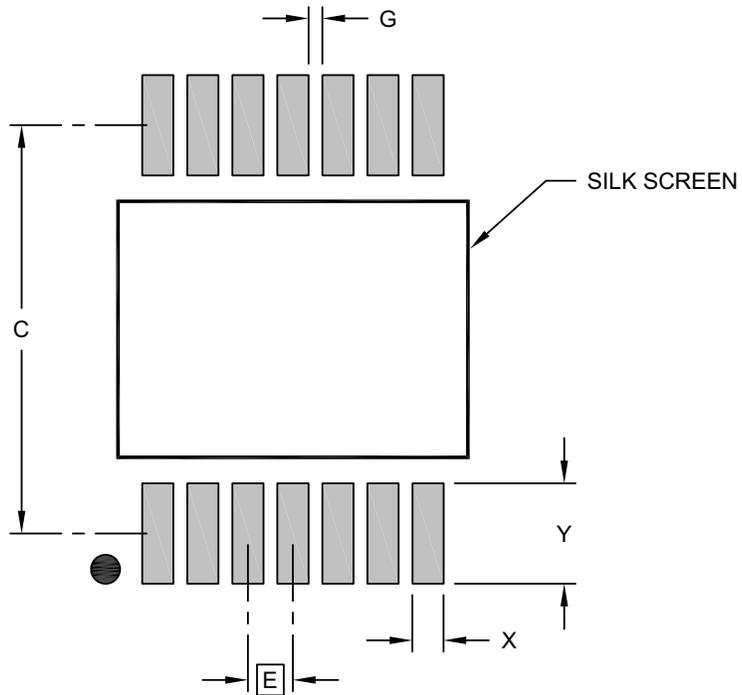**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging
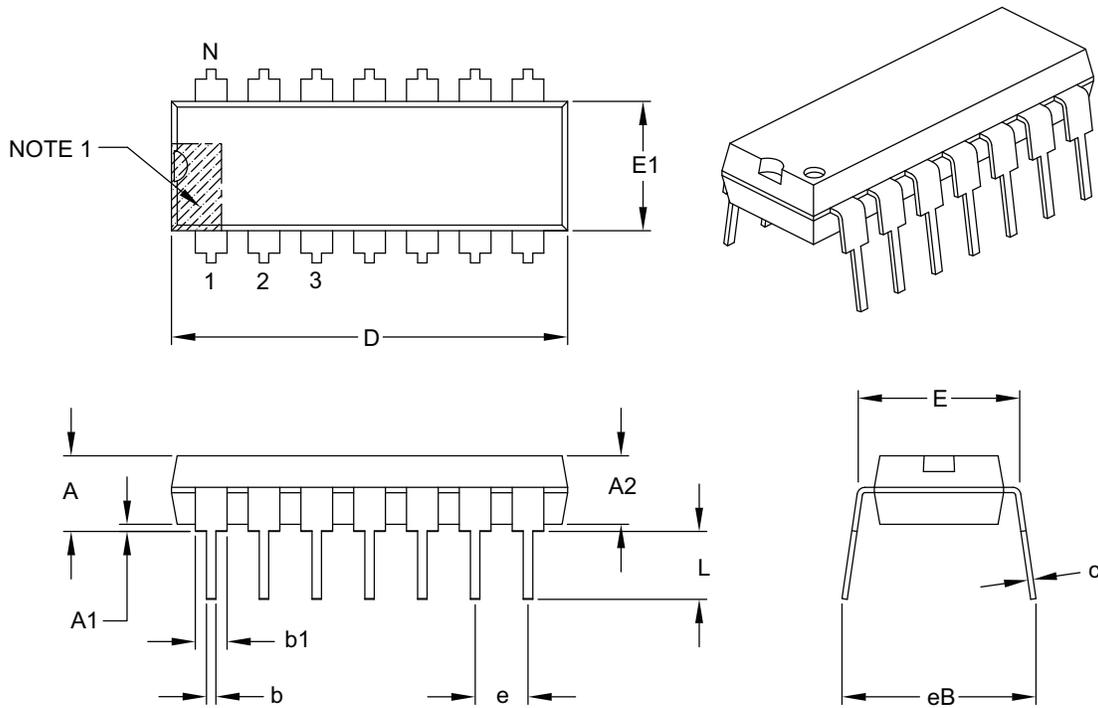


| | Units | | INCHES | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 14 | |
| Pitch | e | | .100 BSC | |
| Top to Seating Plane | A | – | – | .210 |
| Molded Package Thickness | A2 | .115 | .130 | .195 |
| Base to Seating Plane | A1 | .015 | – | – |
| Shoulder to Shoulder Width | E | .290 | .310 | .325 |
| Molded Package Width | E1 | .240 | .250 | .280 |
| Overall Length | D | .735 | .750 | .775 |
| Tip to Seating Plane | L | .115 | .130 | .150 |
| Lead Thickness | c | .008 | .010 | .015 |
| Upper Lead Width | b1 | .045 | .060 | .070 |
| Lower Lead Width | b | .014 | .018 | .022 |
| Overall Row Spacing § | eB | – | – | .430 |

**Notes:**

1. Pin 1 visual index feature may vary, but must be located with the hatched area.
2. § Significant Characteristic.
3. Dimensions D and E1 do not include mold flash or protrusions. Mold flash or protrusions shall not exceed .010" per side.
4. Dimensioning and tolerancing per ASME Y14.5M.

    BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-005B

# MCP2221A

**14-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]**

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging

TOP VIEW

SIDE VIEW

VIEW A–A

VIEW C

Microchip Technology Drawing No. C04-065-SL Rev D Sheet 1 of 2

**14-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]**
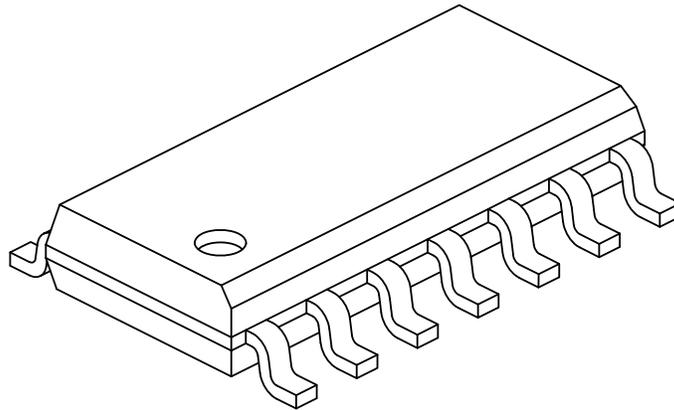
| Note: | For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging |
|---|---|



| | Units | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | | 14 | |
| Pitch | e | | 1.27 BSC | |
| Overall Height | A | - | - | 1.75 |
| Molded Package Thickness | A2 | 1.25 | - | - |
| Standoff        § | A1 | 0.10 | - | 0.25 |
| Overall Width | E | | 6.00 BSC | |
| Molded Package Width | E1 | | 3.90 BSC | |
| Overall Length | D | | 8.65 BSC | |
| Chamfer (Optional) | h | 0.25 | - | 0.50 |
| Foot Length | L | 0.40 | - | 1.27 |
| Footprint | L1 | | 1.04 REF | |
| Lead Angle | $\Theta$ | 0° | - | - |
| Foot Angle | $\varphi$ | 0° | - | 8° |
| Lead Thickness | c | 0.10 | - | 0.25 |
| Lead Width | b | 0.31 | - | 0.51 |
| Mold Draft Angle Top | $\alpha$ | 5° | - | 15° |
| Mold Draft Angle Bottom | $\beta$ | 5° | - | 15° |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. § Significant Characteristic
3. Dimension D does not include mold flash, protrusions or gate burrs, which shall not exceed 0.15 mm per end. Dimension E1 does not include interlead flash or protrusion, which shall not exceed 0.25 mm per side.
4. Dimensioning and tolerancing per ASME Y14.5M
    BSC: Basic Dimension. Theoretically exact value shown without tolerances.
    REF: Reference Dimension, usually without tolerance, for information purposes only.
5. Datums A & B to be determined at Datum H.

Microchip Technology Drawing No. C04-065-SL Rev D Sheet 2 of 2

## 14-Lead Plastic Small Outline (SL) - Narrow, 3.90 mm Body [SOIC]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging

RECOMMENDED LAND PATTERN

| | Units | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 1.27 BSC | |
| Contact Pad Spacing | C | | 5.40 | |
| Contact Pad Width (X14) | X | | | 0.60 |
| Contact Pad Length (X14) | Y | | | 1.55 |

Notes:

1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2065-SL Rev D

**14-Lead Thin Shrink Small Outline Package [ST] 4.4 mm Body [TSSOP]**

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging

TOP VIEW

SIDE VIEW

VIEW A–A

SEE DETAIL B

SEATING PLANE

Microchip Technology Drawing C04-087 Rev D  Sheet 1 of 2

**14-Lead Thin Shrink Small Outline Package [ST] 4.4 mm Body [TSSOP]**

| | |
|---|---|
| **Note:** | For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging |

DETAIL B
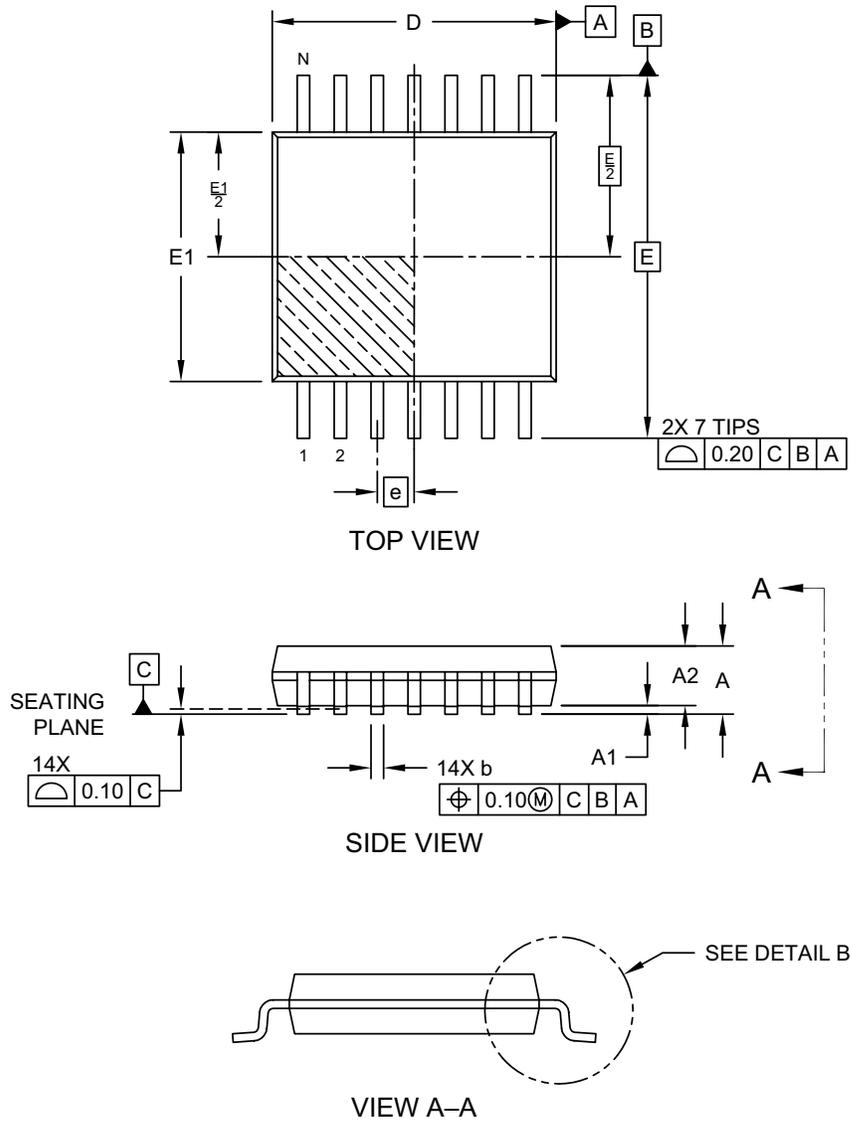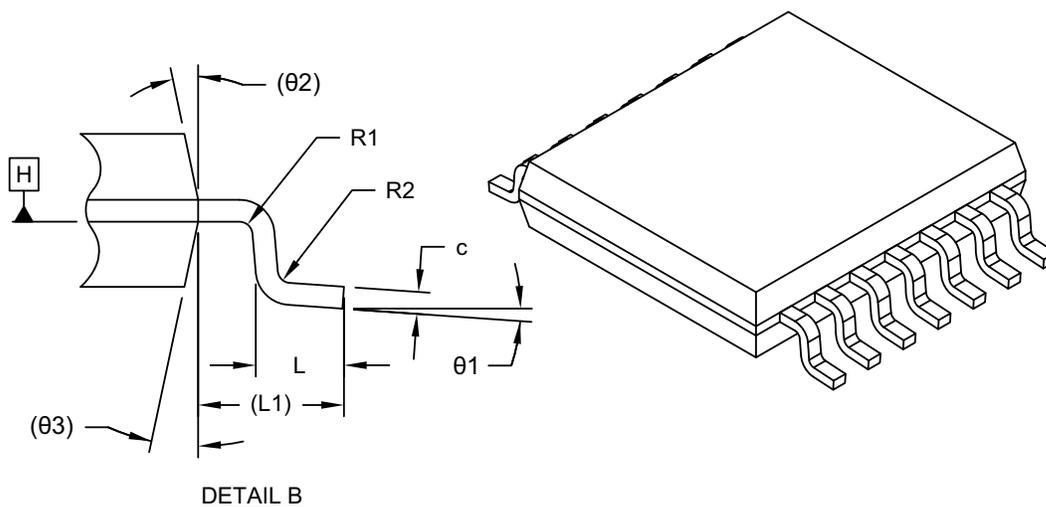
| | Units | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Terminals | N | | 14 | |
| Pitch | e | | 0.65 BSC | |
| Overall Height | A | – | – | 1.20 |
| Standoff | A1 | 0.05 | – | 0.15 |
| Molded Package Thickness | A2 | 0.80 | 1.00 | 1.05 |
| Overall Length | D | 4.90 | 5.00 | 5.10 |
| Overall Width | E | | 6.40 BSC | |
| Molded Package Width | E1 | 4.30 | 4.40 | 4.50 |
| Terminal Width | b | 0.19 | – | 0.30 |
| Terminal Thickness | c | 0.09 | – | 0.20 |
| Terminal Length | L | 0.45 | 0.60 | 0.75 |
| Footprint | L1 | | 1.00 REF | |
| Lead Bend Radius | R1 | 0.09 | – | – |
| Lead Bend Radius | R2 | 0.09 | – | – |
| Foot Angle | θ1 | 0° | – | 8° |
| Mold Draft Angle | θ2 | – | 12° REF | – |
| Mold Draft Angle | θ3 | – | 12° REF | – |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.
2. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.
   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing C04-087 Rev D  Sheet 2 of 2

## 14-Lead Thin Shrink Small Outline Package [ST] 4.4 mm Body [TSSOP]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



## RECOMMENDED LAND PATTERN

| | Units | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | 0.65 BSC | | |
| Contact Pad Spacing | C | | 5.90 | |
| Contact Pad Width (Xnn) | X | | | 0.45 |
| Contact Pad Length (Xnn) | Y | | | 1.45 |
| Contact Pad to Contact Pad (Xnn) | G | 0.20 | | |

Notes:
1. Dimensioning and tolerancing per ASME Y14.5M
   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing C04-2087 Rev D

# MCP2221A

## 16-Lead Plastic Quad Flat, No Lead Package (ML) - 4x4x0.9mm Body [QFN]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging



TOP VIEW

SIDE VIEW

BOTTOM VIEW
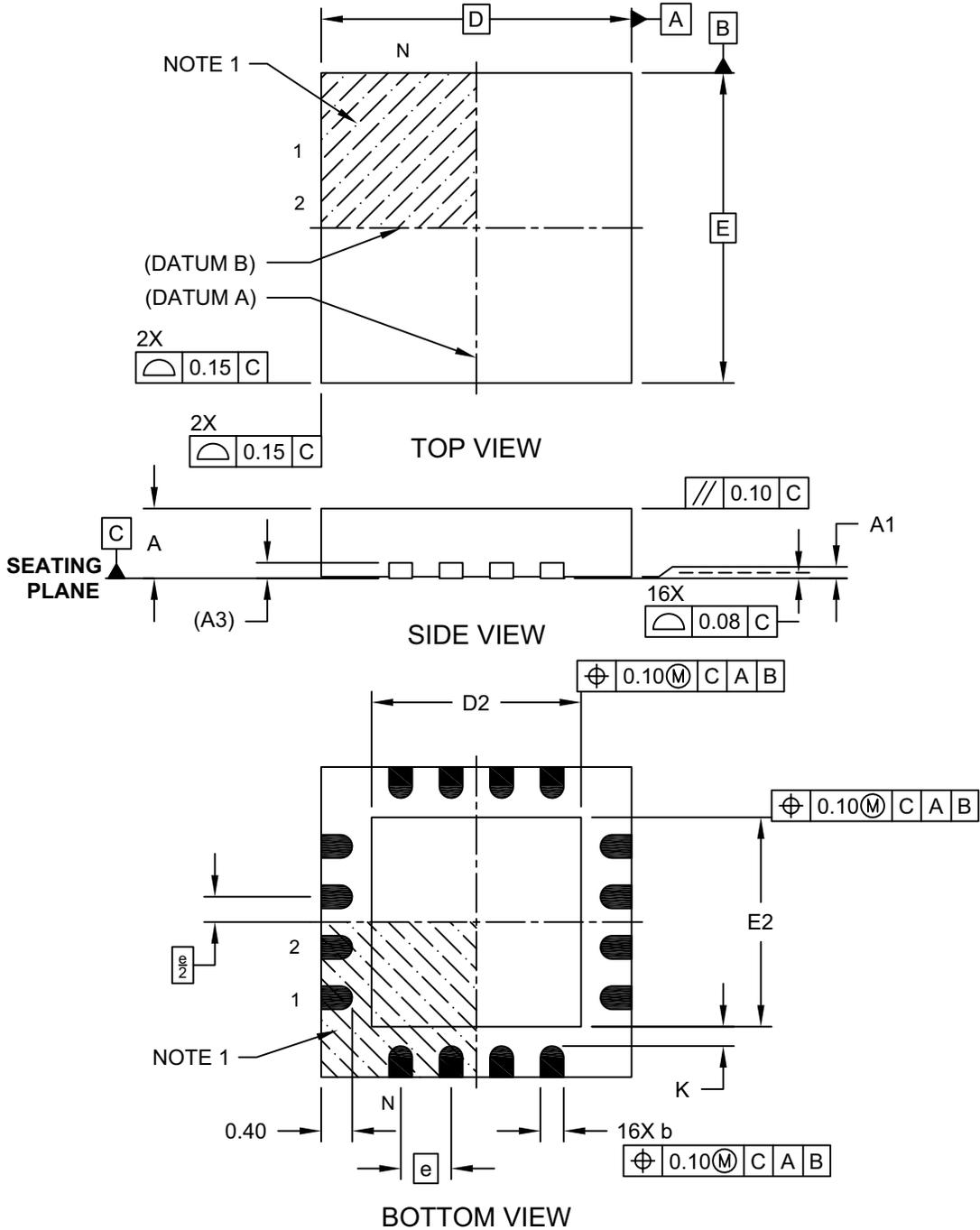
Microchip Technology Drawing  C04-127D Sheet 1 of 2

## 16-Lead Plastic Quad Flat, No Lead Package (ML) - 4x4x0.9mm Body [QFN]

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging
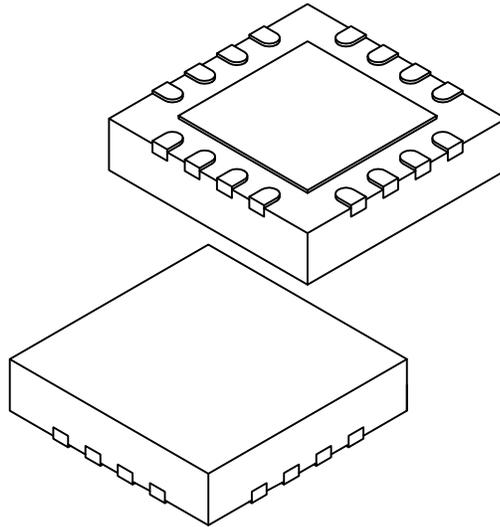


| | Units | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Number of Pins | N | 16 | | |
| Pitch | e | 0.65 BSC | | |
| Overall Height | A | 0.80 | 0.90 | 1.00 |
| Standoff | A1 | 0.00 | 0.02 | 0.05 |
| Contact Thickness | A3 | 0.20 REF | | |
| Overall Width | E | 4.00 BSC | | |
| Exposed Pad Width | E2 | 2.50 | 2.65 | 2.80 |
| Overall Length | D | 4.00 BSC | | |
| Exposed Pad Length | D2 | 2.50 | 2.65 | 2.80 |
| Contact Width | b | 0.25 | 0.30 | 0.35 |
| Contact Length | L | 0.30 | 0.40 | 0.50 |
| Contact-to-Exposed Pad | K | 0.20 | - | - |

Notes:

1. Pin 1 visual index feature may vary, but must be located within the hatched area.

2. Package is saw singulated

3. Dimensioning and tolerancing per ASME Y14.5M

   BSC: Basic Dimension. Theoretically exact value shown without tolerances.

   REF: Reference Dimension, usually without tolerance, for information purposes only.

Microchip Technology Drawing  C04-127D Sheet 2 of 2

# MCP2221A

**16-Lead Plastic Quad Flat, No Lead Package (ML) - 4x4x0.9mm Body [QFN]**

**Note:** For the most current package drawings, please see the Microchip Packaging Specification located at http://www.microchip.com/packaging

RECOMMENDED LAND PATTERN

SILK SCREEN
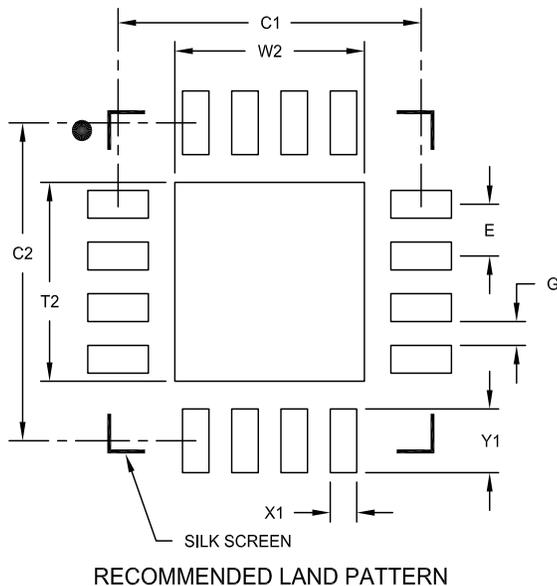
| | Units | MILLIMETERS | | |
|---|---|---|---|---|
| Dimension Limits | | MIN | NOM | MAX |
| Contact Pitch | E | | 0.65 BSC | |
| Optional Center Pad Width | W2 | | | 2.50 |
| Optional Center Pad Length | T2 | | | 2.50 |
| Contact Pad Spacing | C1 | | 4.00 | |
| Contact Pad Spacing | C2 | | 4.00 | |
| Contact Pad Width (X28) | X1 | | | 0.35 |
| Contact Pad Length (X28) | Y1 | | | 0.80 |
| Distance Between Pads | G | 0.30 | | |

Notes:
1. Dimensioning and tolerancing per ASME Y14.5M

    BSC: Basic Dimension. Theoretically exact value shown without tolerances.

Microchip Technology Drawing No. C04-2127A

© 2016-2021 Microchip Technology Inc.

## APPENDIX A:   REVISION HISTORY

### Revision E (September 2021)

The following is the list of modifications:

• Updated **Section 5.0, Packaging Information.**
• Updated **Table 3-2**, **Table 3-5**, **Table 3-12**, **Table 3-23** and **Table 3-39** for clarity.

### Revision D (April 2021)

The following is the list of modifications:

• Added the **Note** in **Section 1.8.1.1**.
• Updated **Table 3-1**.
• The I$^2$C standard uses the terminology "Master" and "Slave". The equivalent Microchip terminology used in this document is "Host" and "Client" respectively.

### Revision C (June 2020)

The following is the list of modifications:

• Updated description of **Register 1-1**.
• Added **Figure 1-8**.
• Clarified **Table 3-2** byte 8 as "don't care".
• Made minor typographical edits.

### Revision B (July 2017)

The following is the list of modifications:

• Added **Table 4-2** and **Table 4-3**.
• Made minor typographical edits.

### Revision A (June 2016)

• Original release of this document.

**NOTES:**

## PRODUCT IDENTIFICATION SYSTEM

To order or obtain information, e.g., on pricing or delivery, contact your local Microchip sales office.

| PART NO. | [X](1) | X | /XX | XXX |
|---|---|---|---|---|
| Device | Tape and Reel Option | Temperature Range | Package | Qualification |

**Device:**
MCP2221A: USB to I²C/UART Protocol Converter
MCP2221AT: USB to I²C/UART Protocol Converter (Tape and Reel)

**Temperature Range:** I = -40°C to +85°C (Industrial)

**Package:**
ML = Plastic Quad Flat, No Lead Package – 4x4x0.9 mm Body (QFN), 16-Lead
P = Plastic Dual In Line, 300 mil. Body (PDIP), 14-Lead
SL = Plastic Small Outline – Narrow, 3.90 mm Body (SOIC), 14-Lead
ST = Plastic Thin Shrink Small Outline – 4.4 mm Body (TSSOP), 14-Lead

**Qualification:**
(Blank) = Standard Qualification
VAO = AEC-Q100 Automotive Qualification

**Note 1:** Tape and Reel identifier only appears in the catalog part number description. This identifier is used for ordering purposes and is not printed on the device package. Check with your Microchip Sales Office for package availability with the Tape and Reel option.

**Examples:**

a) MCP2221A- I/P: Industrial temperature, 14-Lead PDIP package

a) MCP2221A- I/SL: Industrial temperature, 14-Lead SOIC package

b) MCP2221AT- I/SL: Industrial temperature, 14-Lead SOIC package, Tape and Reel

a) MCP2221A- I/ST: Industrial temperature, 14-Lead TSSOP package

b) MCP2221AT- I/ST: Industrial temperature, 14-Lead TSSOP package, Tape and Reel

a) MCP2221A- I/ML: Industrial temperature, 16-Lead 4x4 QFN package

b) MCP2221AT- I/ML: Industrial temperature, 16-Lead 4x4 QFN package, Tape and Reel

a) MCP2221A-I/MLVAO Industrial temperature, 16-Lead 4x4 QFN package, Automotive Qualified

**NOTES:**

# 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

**Australia - Sydney**
Tel: 61-2-9868-6733

**China - Beijing**
Tel: 86-10-8569-7000

**China - Chengdu**
Tel: 86-28-8665-5511

**China - Chongqing**
Tel: 86-23-8980-9588

**China - Dongguan**
Tel: 86-769-8702-9880

**China - Guangzhou**
Tel: 86-20-8755-8029

**China - Hangzhou**
Tel: 86-571-8792-8115

**China - Hong Kong SAR**
Tel: 852-2943-5100

**China - Nanjing**
Tel: 86-25-8473-2460

**China - Qingdao**
Tel: 86-532-8502-7355

**China - Shanghai**
Tel: 86-21-3326-8000

**China - Shenyang**
Tel: 86-24-2334-2829

**China - Shenzhen**
Tel: 86-755-8864-2200

**China - Suzhou**
Tel: 86-186-6233-1526

**China - Wuhan**
Tel: 86-27-5980-5300

**China - Xian**
Tel: 86-29-8833-7252

**China - Xiamen**
Tel: 86-592-2388138

**China - Zhuhai**
Tel: 86-756-3210040

## ASIA/PACIFIC

**India - Bangalore**
Tel: 91-80-3090-4444

**India - New Delhi**
Tel: 91-11-4160-8631

**India - Pune**
Tel: 91-20-4121-0141

**Japan - Osaka**
Tel: 81-6-6152-7160

**Japan - Tokyo**
Tel: 81-3-6880- 3770

**Korea - Daegu**
Tel: 82-53-744-4301

**Korea - Seoul**
Tel: 82-2-554-7200

**Malaysia - Kuala Lumpur**
Tel: 60-3-7651-7906

**Malaysia - Penang**
Tel: 60-4-227-8870

**Philippines - Manila**
Tel: 63-2-634-9065

**Singapore**
Tel: 65-6334-8870

**Taiwan - Hsin Chu**
Tel: 886-3-577-8366

**Taiwan - Kaohsiung**
Tel: 886-7-213-7830

**Taiwan - Taipei**
Tel: 886-2-2508-8600

**Thailand - Bangkok**
Tel: 66-2-694-1351

**Vietnam - Ho Chi Minh**
Tel: 84-28-5448-2100

## EUROPE

**Austria - Wels**
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

**Denmark - Copenhagen**
Tel: 45-4485-5910
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

**Germany - Garching**
Tel: 49-8931-9700

**Germany - Haan**
Tel: 49-2129-3766400

**Germany - Heilbronn**
Tel: 49-7131-72400

**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: 47-7288-4388

**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

09/14/21

**Note the following details of the code protection feature on Microchip products:**

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.

- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.

- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable". Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.