# HT42B533-x
# USB to SPI Bridge IC

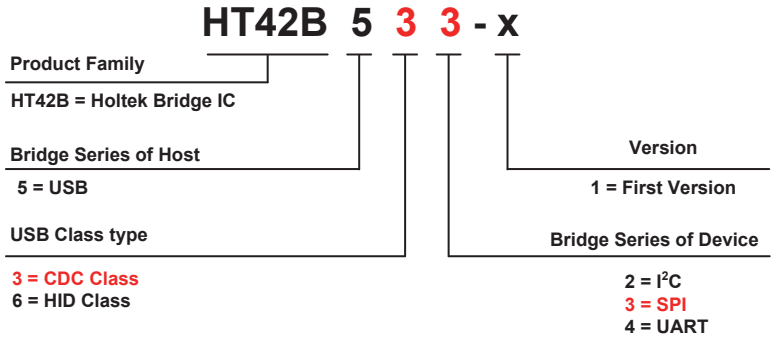
## Features

- Operating Voltage ($V_{DD}$): 3.3V~5.5V
- SPI pins Voltage ($V_{DDIO}$): 1.8V~$V_{DD}$ (Less than $V_{DD}$ voltage)
- Fully integrated internal 12MHz oscillator with 0.25% accuracy for all USB modes which requires no external components
- USB interface
  - USB 2.0 Full Speed compatible
  - Implements USB protocol composite device:
    - Communication Device Class (CDC) for communications and configuration.
    - Human Interface Device (HID) for user configure USB VID, PID and device description strings
  - Integrated an internal 1.5kΩ pull-high resistor on D+ pin
- Serial Interface – SPI
  - Supports clock rate up to 8MHz
  - Supports Master and Slave modes decided by AP command
  - Supports maximum 128 bytes transmit buffer and 128 bytes receive buffer
  - Supports SDI (Master mode) / $\overline{SCS}$ (Slave mode) pin resume signal to request a remote wake-up
  - Supports VDDIO pin for SPI pins power supply
- Support standard Windows® drivers for Virtual Com Port (VCP): Windows XP (SP2), Vista, Windows 7, Windows 8, Windows 8.1 (only an INF file is required) and Windows 10.
- Support Android 4.0 or later version and Mac OS X
- Integrated 256 bytes internal true EEPROM for user memory
- Power down and wake-up functions to reduce power consumption
- Package types: 10-pin MSOP, 16-pin NSOP

## General Description

The HT42B533-x is a high performance USB to SPI bridge controller with fully integrated USB and SPI interface functions, designed for applications that communicate with various types of SPI. The device includes a USB 2.0 full speed compatible interface which is used for PC commication. The device also includes a fully integrated high speed oscillator which is used for USB and SPI clock generator.

## USB Bridge IC Naming Rules

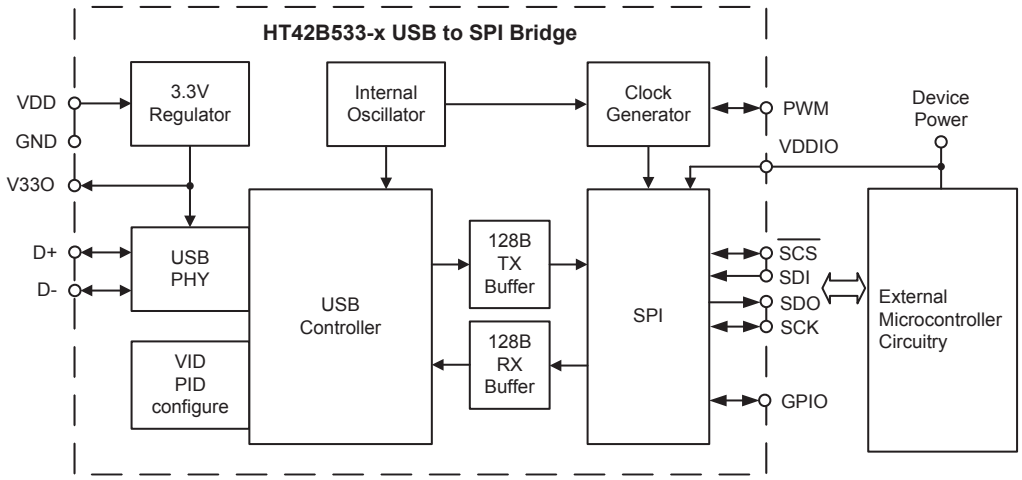


## Selection Table

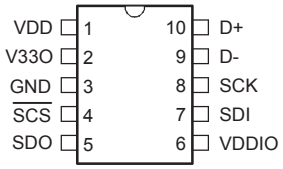
Most features are common to all devices. The following table summarises the main features of each device.

| Part No. | Description | $V_{DD}$ | USB | Virtual COM | HID | FIFO/Buffer | Interface Data Rate | I/O $V_{DD}$ | Package |
|---|---|---|---|---|---|---|---|---|---|
| HT42B532-x | USB to $I^2C$ Bridge | 3.3V~ 5.5V | Full-Speed | √ | – | TX: 62 bytes RX: 62 bytes | Up to 400kHz | √ | 8SOP, 10MSOP |
| HT42B533-x | USB to SPI Bridge | | | √ | – | TX: 128 bytes RX: 128 bytes | Up to 8MHz | √ | 10MSOP, 16NSOP |
| HT42B534-x | USB to UART Bridge | | | √ | – | TX: 128 bytes RX: 128 bytes | Up to 3Mbps Baud | √ | 8/10SOP, 10MSOP, 16NSOP |
| HT42B564-x | USB (HID) to UART Bridge | | | – | √ | TX: 32 bytes RX: 32 bytes | Up to 115.2kbps Baud | √ | 10SOP |

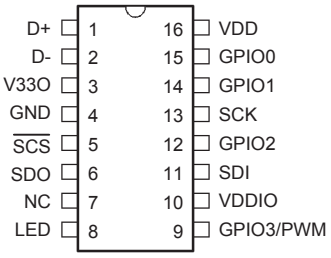
## Block Diagram




## Pin Assignment




**HT42B533-1**
**10 MSOP-A**




**HT42B533-1**
**16 NSOP-A**

| Package type | Marking |
|---|---|
| 10MSOP | B533-x |
| 16NSOP | HT42B533-x |

Note: x=1 for version number.

## Pin Descriptions

As the Pin Description table applies to the package type with the most pins, not all of the listed pins may be present on package types with smaller numbers of pins.

| Pin Name | Type | Description |
|---|---|---|
| D+ | I/O | USB D+ Line |
| D- | I/O | USB D- Line |
| $\overline{SCS}$ | I/O | SPI slave select |
| SDO | O | SPI data output |
| SDI | I | SPI data input |
| SCK | I/O | SPI serial clock |
| LED | O | Data transfer signal LED indication, active low |
| PWM | O | Pulse Width Modulator output (Only for 16NSOP package) |
| GPIO0~3 | I/O | GPIO |
| V33O | O | 3.3V Regulator output |
| VDDIO | PWR | Positive power supply for $\overline{SCS}$, SDO, SDI, SCK pins |
| VDD | PWR | Positive power supply, USB bus power |
| GND | PWR | Negative power supply, ground |

## Absolute Maximum Ratings

Supply Voltage ..... $V_{SS}$-0.3V to $V_{SS}$+6.0V
Input Voltage ..... $V_{SS}$-0.3V to $V_{SS}$+0.3V
Storage Temperature ..... -50°C to 125°C
Operating Temperature ..... -40°C to 85°C

$I_{OH}$ Total ..... -80mA
$I_{OL}$ Total ..... 80mA
Total Power Dissipation ..... 500mW

Note: These are stress ratings only. Stresses exceeding the range specified under "Absolute Maximum Ratings" may cause substantial damage to this device. Functional operation of these devices at other conditions beyond those listed in the specification is not implied and prolonged exposure to extreme conditions may affect device reliability.

## D.C. Characteristics

Ta=25°C

| Symbol | Parameter | Test Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| | | $V_{DD}$ | Conditions | | | | |
| $V_{DD}$ | Operating Voltage | — | — | 3.3 | — | 5.5 | V |
| $V_{DDIO}$ | VDDIO Input Voltage for SPI Pins | — | — | 1.8 | — | $V_{DD}$ | V |
| $I_{DD}$ | Operating Current | 5V | No load | — | 11 | 16 | mA |
| $I_{SUS}$ | Suspend Current (USB) | 5V | Suspend mode, No load, USB on, other peripherals off | — | 360 | 450 | μA |
| $V_{IL}$ | Input Low Voltage for Input Pins | — | — | 0 | — | $0.2V_{DDIO}$ | V |
| $V_{IH}$ | Input High Voltage for Input Pins | — | — | $0.8V_{DDIO}$ | — | $V_{DDIO}$ | V |
| $I_{OL}$ | Sink Current for I/O Pins | 3V | $V_{OL}$ = $0.1V_{DDIO}$ | 4 | 8 | — | mA |
| | | 5V | | 10 | 20 | — | mA |
| $I_{OH}$ | Source Current for I/O Pins | 3V | $V_{OH}$ = $0.9V_{DDIO}$ | -2 | -4 | — | mA |
| | | 5V | | -5 | -10 | — | mA |
| $R_{PH}$ | Pull-high Resistance for I/O Ports | 3V | — | 20 | 60 | 100 | kΩ |
| | | 5V | — | 10 | 30 | 50 | kΩ |

| Symbol | Parameter | Test Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| | | $V_{DD}$ | Conditions | | | | |
| $I_{LEAK}$ | Input Leakage Current | 3V | $V_{IN}$ = $V_{DD}$ or $V_{IN}$ = $V_{SS}$ | — | — | ±1 | μA |
| | | 5V | | — | — | ±1 | μA |
| $V_{V33O}$ | 3.3V Regulator Output Voltage | 5V | $I_{V33O}$ = 70mA | 3.0 | 3.3 | 3.6 | V |
| $R_{UDP1}$ | Pull-high Resistance between D+ and V33O | 3.3V | — | -5% | 1.5 | +5% | kΩ |

## A.C. Characteristics

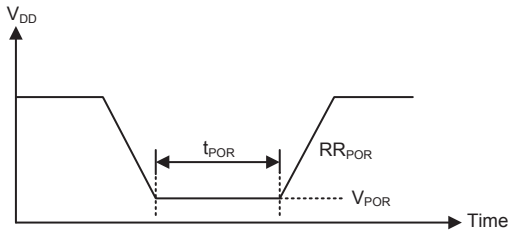
Ta=25°C

| Symbol | Parameter | Test Condition | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| | | $V_{DD}$ | Condition | | | | |
| $f_{HIRC}$ | High Speed Internal RC Oscillator | 3.3V~5.5V | USB mode | -0.25% | 12 | +0.25% | MHz |
| $t_{SST}$ | System Start-up Timer Period | — | SPI pins Wake-up from power down mode SDI(Master mode)/ SCS(Slave mode) | 16 | — | — | $t_{HIRC}$ |
| $t_{RSTD}$ | System Reset Delay Time | — | Power-on reset | 25 | 50 | 100 | ms |

## Power-on Reset Characteristics

Ta=25°C

| Symbol | Parameter | Test Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| | | $V_{DD}$ | Conditions | | | | |
| $V_{POR}$ | $V_{DD}$ Start Voltage to Ensure Power-on Reset | — | — | — | — | 100 | mV |
| $RR_{POR}$ | $V_{DD}$ Rising Rate to Ensure Power-on Reset | — | — | 0.035 | — | — | V/ms |
| $t_{POR}$ | Minimum Time for $V_{DD}$ Stays at $V_{POR}$ to Ensure Power-on Reset | — | — | 1 | — | — | ms |



## USB Interface

The USB interface, being USB 2.0 full-speed compatible, is a 4-wire series bus that allows communication between a host device and up to 127 max peripheral devices on the same bus. A token based protocol method is used by the host device for communication control. Other advantages of the USB bus include live plugging and unplugging and dynamic device configuration. As the complexity of USB data protocol does not permit comprehensive USB operation information to be provided in this datasheet, the reader should therefore consult other external information for a detailed USB understanding. The device includes a USB interface function allowing for the convenient design of USB peripheral products.

### Power Plane

There are two power planes for the device and they are the USB bus power input ($V_{DD}$) and 3.3V regulator output ($V_{33O}$).

For the USB SIE $V_{DD}$, it will supply power for all circuits related to USB SIE and it is sourced from pin "VDD". Once the USB is removed from the USB and there is no power in the USB BUS, the USB SIE circuit is no longer operational.

### USB Interface Operation

To communicate with an external USB host, the internal USB module has the external pins known as D+ and D- along with the 3.3V regulator output pin V33O. A Serial Interface Engine (SIE) decodes the incoming USB data stream and transfers it to the correct endpoint buffer memory known as the FIFO. The USB module has 4 endpoints, EP0 ~ EP3. The endpoint 0 supports the Control transfer while the endpoint 1 ~ endpoint 3 support the Interrupt or Bulk transfer. The HT42B533-x Bridge IC supports the USB Communication Device Class (CDC) for communications and configuration.

| Endpoint | Transfer Type |
|---|---|
| 0 | Control |
| 1 | Interrupt |
| 2 | Bulk Out |
| 3 | Bulk In |

**USB Endpoint Transfer Type**

If there is no signal on the USB bus for over 3ms, the USB device will enter the suspend mode. The device enters the suspend state to meet the requirements of the USB suspend current specification. When the resume signal is asserted by the USB host, the device will be woken up and leave the suspend mode.

As the USB device has a remote wake-up function, the USB device can wake up the USB host by sending a remote wake-up pulse. Once the USB host receives a remote wake-up signal from the USB device, the host will send a resume signal to device.

### USB VID and PID Configure

The device has configured the default Vender ID (VID:0x04D9), Product ID (PID:0xB533) and product description strings of "USB to SPI Bridge". The user can update Vender ID, Product ID, product description strings and remote wake-up setting using their application programs.

This device has been configured to the default USB configuration data as shown in the following table.

| Parameter | Value (Hex) |
|---|---|
| USB Vendor ID (VID) | 0x04D9 |
| USB Product ID (PID) | 0xB533 |
| Remote wake-up | Default disable |
| Manufacturer Name | Holtek |
| Product Description | USB to SPI Bridge |
| Serial Number | 0000 |

## SPI Interface

The HT42B533-x contains an SPI function. The SPI interface is often used to communicate with external peripheral devices such as microcontrollers, sensors, Flash devices, etc. Originally developed by Motorola, the four line SPI interface is a synchronous serial data interface that has a relatively simple communication protocol simplifying the programming requirements when communicating with external hardware devices.

### SPI Interface Operation

The communication is full duplex and operates as a slave/master type, where the devices can be either master or slave. Although the SPI interface specification can control multiple slave devices from a single master, the device provides only one $\overline{SCS}$ pin. If the master needs to control multiple slave devices from a single master, the master can use a GPIO pin to select the slave devices. It is a four line interface with pin names SDI, SDO, SCK and $\overline{SCS}$. Pins SDI and SDO are the Serial Data Input and Serial Data Output lines. The SCK pin is the Serial Clock line and $\overline{SCS}$ is the Slave Select line.

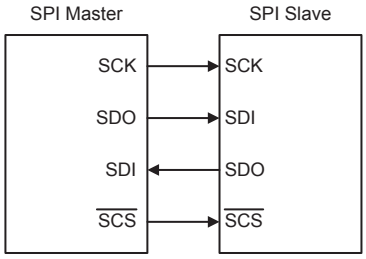
The SPI Serial Interface function includes the following features:

- Full-duplex synchronous data transfer
- Both Master and Slave mode
- Master mode serial clock frequency up to 8MHz
- LSB first or MSB first data transmission modes
- Rising or falling active clock edge
- 128-byte Deep FIFO Transmit Data Buffer
- 128-byte Deep FIFO Receive Data Buffer
- SDI pin (Master mode) or $\overline{SCS}$ pin (Slave mode) wake-up function
- SPI pins power supply by the VDDIO pin

### SPI Communication

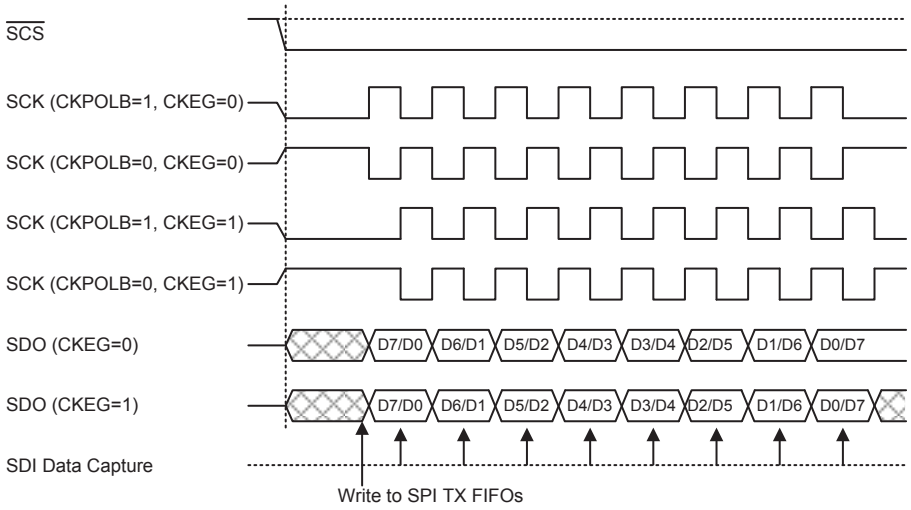
After the SPI interface is enabled using the application program, then in the Master mode, when data is written, transmission/reception will begin simultaneously. In the Slave mode, when the clock signal from the master has been received, any data in the SPI TX FIFOs will be transmitted by the SDO pin and any data on the SDI pin will be shifted into the SPI RX FIFOs.

The master should output a $\overline{SCS}$ signal to enable the slave devices before a clock signal is provided. The slave data to be transferred should be well prepared at the appropriate moment relative to the SCK signal depending upon the configurations of the CKPOLB bit and CKEG bit. The accompanying timing diagram shows the relationship between the slave data and SCK signal for various configurations of the CKPOLB and CKEG bits. The SPI will continue to function if the SPI clock source is active.

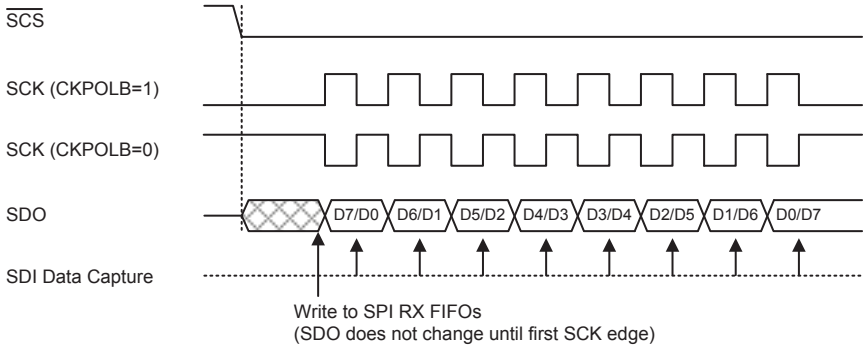



**SPI Master/Slave Connection**

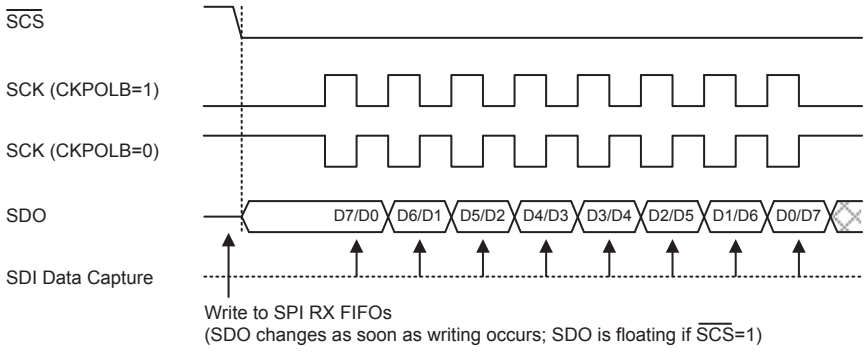
| SPI Timing Setup Mode | CKPOLB | CKEG | Descrition |
|---|---|---|---|
| Mode 3 | 0 | 0 | SCK is high base level when the clock is inactive and data is captured at SCK rising edge |
| Mode 2 | 0 | 1 | SCK is high base level when the clock is inactive and data is captured at SCK falling edg |
| Mode 1 | 1 | 0 | SCK is low base level when the clock is inactive and data is captured at SCK falling edge |
| Mode 0 | 1 | 1 | SCK is low base level when the clock is inactive and data is captured at SCK rising edge |




**SPI Master Mode Timing**




**SPI Slave Mode Timing – CKEG=0**




**SPI Slave Mode Timing – CKEG=1**

### SPI Clock

The SCK pin clock for the Master mode can be set using the Holtek Bridge API command to define the desired value, as shown below.

| SPI Mode | SPI Clock | Value (Hex) |
|---|---|---|
| Master | 8MHz | 02 |
| Master | 6MHz | 03 |
| Master | 4MHz | 04 |
| Master | 3MHz | 05 |
| Master | 1MHz | 06 |
| Master | 750KHz | 07 |
| Master | 250KHz | 08 |
| Slave | — | 09 |

### SPI Power Down and Wake-up

If the USB host sends a suspend signal to the HT42B533-x USB device, it will enter the suspend mode. It is recommended to ensure that the SPI data transmission or reception has been finished before the device enters the suspend mode.

The SPI function contains the SDI pin (master mode) and $\overline{SCS}$ pin (slave mode) wake-up functions. A falling edge on the SDI pin or $\overline{SCS}$ pin will wake up the device from the suspend mode.

## Holtek Bridge DLL User Guideline

### Holtek USB Bridge Program

Holtek has provided the DLL to build the HT42B533-x/HT42B532-x Bridge IC application programs for USB to SPI or USB to $I^2C$ data communication. The API descriptions are described as below.

- **HTB_API BOOL OpenDevice(int nCom);**
  Holtek Bridge operates in the format of Virtual COM Port. Define the COM port number using the parameter.
- **HTB_API void CloseDevice();**
  Used to turn off the Bridge Device.
- **HTB_API BOOL SetIICDataRate(int nDR);**
  Used to configure the $I^2C$ Data Rate (for USB to $I^2C$ Bridge). Refer to HoltekBridgeDLL.h for parameter definition.
- **HTB_API BOOL SetSPIDataRate(int nDR);**
  Used to configure the SPI Data Rate (for USB to SPI Bridge). Refer to HoltekBridgeDLL.h for parameter definition.
- **HTB_API BOOL SetSPIMode(int nMode, int nOrder, int nCSB);**
  - Refer to HoltekBridgeDLL.h for parameter definition.
  - Used to configure SPI mode 0/1/2/3, LSB/MSB, and whether to use CSB (for USB to SPI Bridge).
  - When in the SPI Slave Mode, CSB must be enabled.
- **HTB_API BOOL SetIICMode(int nMode, int nAddr);**
  Used to configure $I^2C$ Master or Slave mode and address.
- **HTB_API BOOL SetIIC_ReceiverEnd(BOOL bAck);**
  - Used to configure when the Master Receiver ends transmission, $I^2C$ returns ACK or NACK.
    bAck: When the data length defined by SetDirection or BRRead has been received, $I^2C$ will return ACK or NACK.
    TRUE: ACK
    FALSE: NACK
  - After the OpenDevice action, this parameter's default status is NACK. The parameter should be configured after OpenDevice and before Read/Write actions.
- **HTB_API BOOL SetIIC_Restart(BOOL bRestart);**
  - Used to define the signal behavior when in the Master Mode.
    bRestart: When the data length defined by SetDirection or BRRead has been received or transmitted, $I^2C$ will generate a STOP signal or Restart signal.
    TRUE: Restart
    FALSE: STOP
  - After the OpenDevice action, this parameter's default status is STOP. The parameter should be configured before Read/Write actions.
- **HTB_API BOOL BRRead(char *p, DWORD nLen, DWORD&BytesRead, DWORD dwTimeOut);**
  - Read
    If the $I^2C$ Receiver returns NACK, the return value is FALSE;
    "BytesRead" indicates the actual byte count being read, if it is not enough, continue call Read function.
    "dwTimeOut" indicates the waiting time for read, unit: ms.
- **HTB_API BOOL BRWrite(char *p, DWORD nLen);**
  - Write
    If the $I^2C$ Receiver returns NACK, the return value is FALSE;

- **HTB_API BOOL Finalize();**
  - To end the current transmission.
  - When in the SPI master mode and CSB is enabled, calling this function will pull CSB high.
  - When in the SPI slave mode, calling this function will reset the bridge to its Receiver default status.
- **HTB_API BOOL ResetDevice();**
  Reset Bridge. This action will clear the contents already stored in the Bridge FIFO.
- **HTB_API BOOL SlaveCSBFalling();**
  When in the SPI Slave Mode, this function is used to detect whether the master has re-enabled CSB. If yes, it means the master will re-transmit commands, in which case call Finalize or ResetDevice to reset the bridge to its Receiver default status.
- **HTB_API BOOL SetDirection(BYTE ucDir, WORD ucLen);**
- **HTB_API BOOL PureRead(char *p, WORD ucLen, WORD &BytesRead,DWORD dwTimeOut);**
- **HTB_API BOOL PureWrite(char *p, WORD nLen);**
  - For both SPI and $I^2C$, before switching bewteen Read and Write, first to set Direction.
  - SetDirection(DIR_READ,LEN) + PureRead = BRRead
  - SetDirection(DIR_WRITE,LEN) + PureWrite = BRWrite
  - The SetDirection function defines the total length "LEN", PureRead or PureWrite supports any length but their total length can not larger than "LEN".
  - "dwTimeOut" indicates the waiting time for read, unit: ms.
- **HTB_API BOOL SetGPIOWakeUp(BYTE);**
  - Used to configure GPIO0/GPIO1/GPIO2/GPIO3 with or without wake up function.
  - This parameter is transmitted in the format of OR. For example, to enable GPIO0 and GPIO2 wake up functions, the parameter is set as shown below:
    SetGPIOWakeup(GPIO0|GPIO2);
- **HTB_API BOOL SetGPIOPullUp(BYTE);**
  - Used to configure GPIO0/GPIO1/GPIO2/GPIO3 with or without pull-high function.
  - This parameter is set in the same way as the previous one.
- **HTB_API BOOL SetGPIOInput(BYTE);**
  - Used to configure GPIO0/GPIO1/GPIO2/GPIO3 input/output direction.
  - SetGPIOInput(GPIO1|GPIO2) indicates GPIO1/GPIO2 are input, GPIO0/GPIO3 are output.
- **HTB_API BOOL SetPWM(BOOL bEnable, PWM *pwm=NULL);**
  - Used to configure the desired PWM value, then the PWM signal will be generated on GPIO3.
  - Refer to the associated Bridge user manual for more details about the setup value.
    The PWM structure is described as below:

| | |
|---|---|
| bPeriod | defines Period width, unit: clock |
| bClock | defines PWM clock frequency |
| bActiveLevel | defines active low or active high |
| bOutputMode | defines PWM signal output mode |
| bOpMode | defines PWM operating mode |
| wDuty | defines Duty width, unit: clock |

## Example

**Setup PWM**

```
PWM pwm;
pwm.bPeriod=PD_1024_CLK;
pwm.bClock=CLK_3M;
pwm.bActiveLevel=ACTIVE_LOW;
pwm.bOutputMode=PWM_OUTPUT;
pwm.bOpMode=PWM_OUTPUT;
pwm.wDuty=0x80;
BOOL bRet = SetPWM(TRUE,&pwm);
```

**Read device which needs a ACK response**

```
DWORD dw=0;
char szBuf[9]={0x10,0,0,0,0,0,0,0,0};
char szRead[16];
BOOL bRet=OpenDevice(3);
bRet=SetIIC_ReceiverEnd(TRUE);
bRet=SetIICDataRate(IIC_200K);
bRet=SetIICMode(IIC_MASTER,0x51);
bRet=BRWrite(szBuf,9);                //Write 8 bytes of 0 to address 0x10
bRet=BRWrite(szBuf,1);                //Write the read address 0x10
bRet=BRRead(szRead,16,dw,50);         //data read back, response with ACK after Read ends
Finalize();
CloseDevice();
```

**Write to/Read from Holtek EEPROM HT24LC0x – using Restart signal**

```
DWORD dw=0;
char szBuf[9]={0x10,0,0,0,0,0,0,0,0};
char szRead[16];
BOOL bRet=OpenDevice(3);
bRet=SetIICDataRate(IIC_200K);
bRet=SetIICMode(IIC_MASTER,0x51);
bRet=BRWrite(szBuf,9);                //Write 8 bytes of 0 to address 0x10
bRet=SetIIC_Restart(TRUE);            //Set before BRWrite
bRet=BRWrite(szBuf,1);                //Write the read address 0x10, generate Restart
                                      //signal after Write ends
bRet=SetIIC_Restart(FALSE);           //Set before BRRead
bRet=BRRead(szRead,16,dw,50);         //data read back, response with NACK after Read
                                      //ends and generate a STOP signal
Finalize();
CloseDevice();
```

**Write to/Read from SPI Flash**

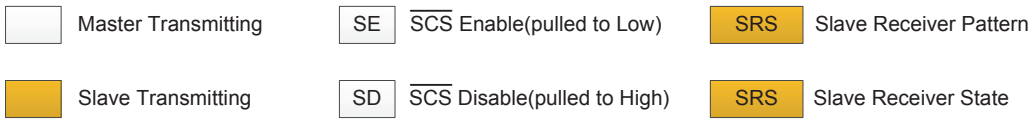
```
DWORD dw=0;
char szID[4]={0x90,0,0,0};
char szCmd[4]={0x03,0,0,0};
char szRead[16];
:
BOOL bRet=OpenDevice(3);
bRet=SetSPIDataRate(SPI_4M);
bRet=SetSPIMode(SPI_MODE0, SPI_MSB,SPI_EN_CSB);
bRet=BRWrite(szID,4);                 //read id
bRet=BRRead(szRead,2,dw,50);
Finalize();
bRet=BRWrite(szCmd,4);
bRet=BRRead(szRead,0x10,dw,50);       //read 0x10 bytes from address 0x00
Finalize();
```

## SPI Slave Mode Setup Description

When the SPI is configured to operate in the slave mode, it is to operate as a Slave Receiver which means to read from the USB host (BULK OUT). The SPI Master terminal should follow the protocol shown in the table for normal communications.

| Master Transmitter (MT) Request | Master Transmitter (MT) Command Code | Slave Receiver (SR) Response |
|---|---|---|
| Wirte data available request | 0xA3 | Buffer Length |
| Write data | 0xA5 | |
| Read data available request | 0xA4 | Buffer Length |
| Read data | 0xA6 | |
| MT $\overline{SCS}$ enable notification | 0xA0 0x7F 0x5A 0xA8 | |
| MT $\overline{SCS}$ disable notification | 0xA0 0x7F 0x5A 0xA8 | |

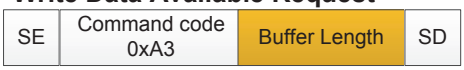
**MT $\overline{SCS}$ Enable Notification**




**MT : Master Transmitter**
**SR : Slave Receiver**
**ST : Slave Transmitter**

**Write data : data from SPI interface to the host (BULK IN)**
**Read data : data from host to SPI interface (BULK OUT)**

**Write Data Available Request**




**Wrtie Data**


**Read Data Available Request**

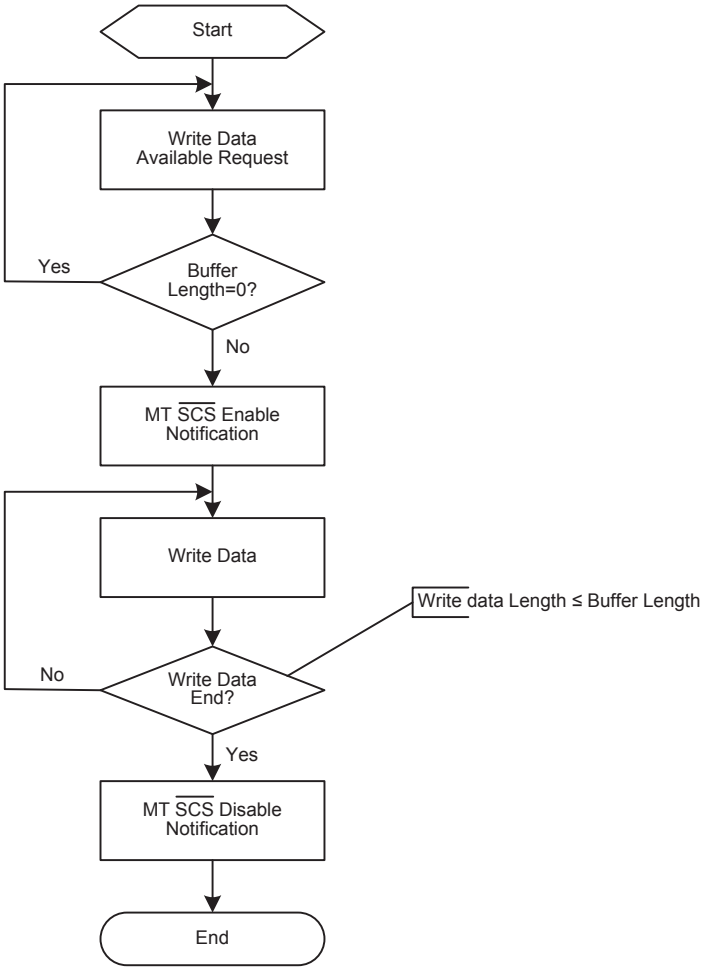

**Read Data**


**MT $\overline{SCS}$ Enable Notification**

**Write Data Flow (Data from SPI Interface to Host: BULK IN)**

**MT : Master Transmitter**
**SR : Slave Receiver**
**ST : Slave Transmitter**

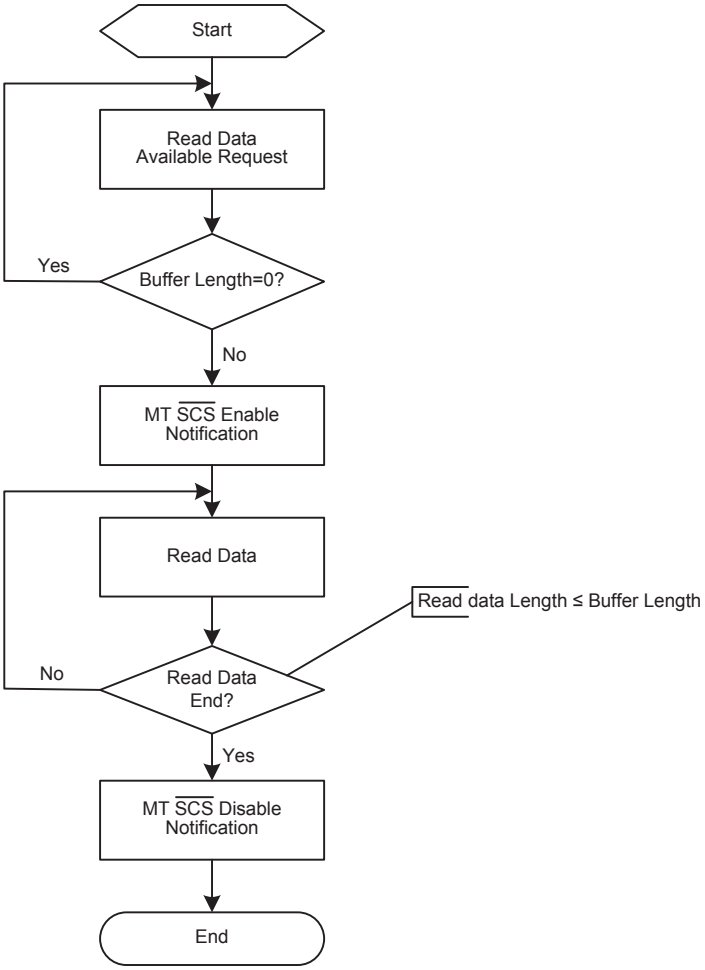
**MT Write Data to SR Flow**



**Read Data Flow (Data from Host to SPI Interface: BULK OUT)**

**MT : Master Transmitter**
**SR : Slave Receiver**
**ST : Slave Transmitter**

**MT Read Data from SR Flow**




**Master Transmitter $\overline{SCS}$ Enable/Disable Notification**

The SPI master enables or disables the data transmission by pulling the $\overline{SCS}$ line low or high respectively.

- MT $\overline{SCS}$ Enable Notification
  The master must send a request to notify the slave receiver before starting to read or write data.
- MT $\overline{SCS}$ Disable Notification:
  The master must send a request to notify the slave receiver before the data read or data write is completed.

  Note that when switching from MT $\overline{SCS}$ Enable Notification to MT $\overline{SCS}$ Disable Notification, the delay time is at least 2ms.

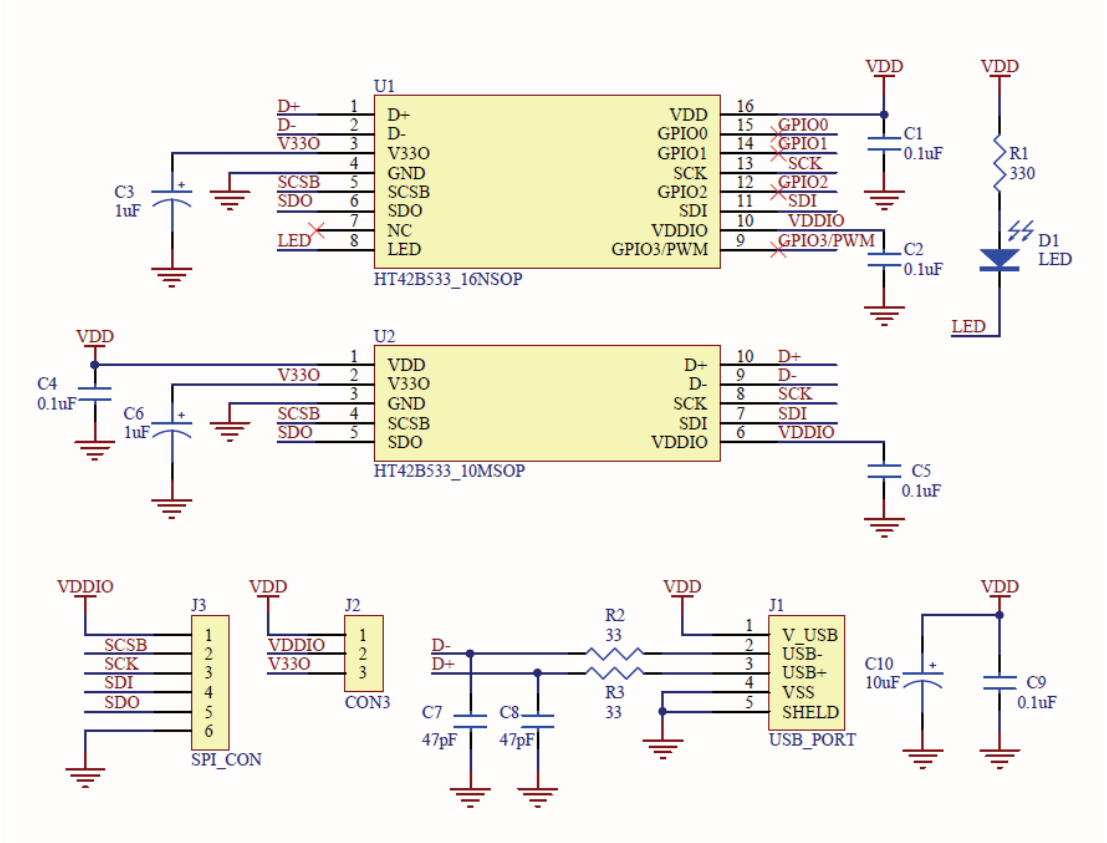
### Product Description Update

To implement product description update first open the Holtek HT42B534-x Bridge IC demo AP, if the USB had been plugged into the host PC, it will show that the USB has been openned successfully on a new window. The user can use the application program to update the customer VID, PID, Manufacturer Name, Product Description, Serial Number and 256 bytes of user memory. It can configure the SPI bridge device hardware flow control and remote wake-up functions. In addition to the definable descriptions, a user memory area which is not used to store parameters is also provided for users to record data.

The configuration descriptor length table is shown as below.

| Parameter | Length |
|---|---|
| USB Vendor ID(VID) | 1 Word (hex) |
| USB Product ID(PID) | 1 Word (hex) |
| Manufacturer Name | Support max. 16 characters |
| Product Description | Support max. 32 characters |
| Serial Number | Support max. 4 words |

## Application Circuits



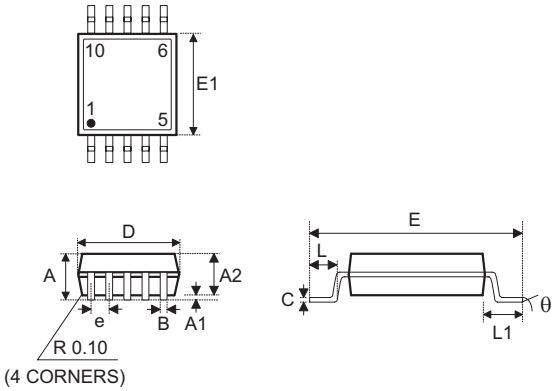
## Package Information

Note that the package information provided here is for consultation purposes only. As this information may be updated at regular intervals users are reminded to consult the Holtek website for the latest version of the Package/Carton Information.

Additional supplementary information with regard to packaging is listed below. Click on the relevant section to be transferred to the relevant website page.

- Package Information (include Outline Dimensions, Product Tape and Reel Specifications)
- The Operation Instruction of Packing Materials
- Carton information

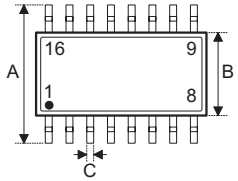
## 10-pin MSOP Outline Dimensions

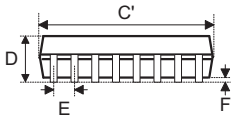


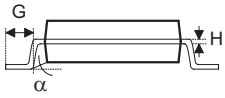

| Symbol | Dimensions in inch | | |
|---|---|---|---|
| | Min. | Nom. | Max. |
| A | — | — | 0.043 |
| A1 | 0.000 | — | 0.006 |
| A2 | 0.030 | 0.033 | 0.037 |
| b | 0.007 | — | 0.013 |
| c | 0.003 | — | 0.009 |
| D | — | 0.118 BSC | — |
| E | — | 0.193 BSC | — |
| E1 | — | 0.118 BSC | — |
| e | — | 0.020 BSC | — |
| L | 0.016 | 0.024 | 0.031 |
| L1 | — | 0.037 BSC | — |
| y | — | 0.004 | — |
| α | 0° | — | 8° |

| Symbol | Dimensions in mm | | |
|---|---|---|---|
| | Min. | Nom. | Max. |
| A | — | — | 1.10 |
| A1 | 0.00 | — | 0.15 |
| A2 | 0.75 | 0.85 | 0.95 |
| b | 0.17 | — | 0.33 |
| c | 0.08 | — | 0.23 |
| D | — | 3 BSC | — |
| E | — | 4.9 BSC | — |
| E1 | — | 3 BSC | — |
| e | — | 0.5 BSC | — |
| L | 0.40 | 0.60 | 0.80 |
| L1 | — | 0.95 BSC | — |
| y | — | 0.1 | — |
| α | 0° | — | 8° |

## 16-pin NSOP (150mil) Outline Dimensions




| Symbol | Dimensions in inch | | |
|---|---|---|---|
| | Min. | Nom. | Max. |
| A | — | 0.236 BSC | — |
| B | — | 0.154 BSC | — |
| C | 0.012 | — | 0.020 |
| C’ | — | 0.390 BSC | — |
| D | — | — | 0.069 |
| E | — | 0.050 BSC | — |
| F | 0.004 | — | 0.010 |
| G | 0.016 | — | 0.050 |
| H | 0.004 | — | 0.010 |
| α | 0° | — | 8° |

| Symbol | Dimensions in mm | | |
|---|---|---|---|
| | Min. | Nom. | Max. |
| A | — | 6 BSC | — |
| B | — | 3.9 BSC | — |
| C | 0.31 | — | 0.51 |
| C’ | — | 9.9 BSC | — |
| D | — | — | 1.75 |
| E | — | 1.27 BSC | — |
| F | 0.10 | — | 0.25 |
| G | 0.40 | — | 1.27 |
| H | 0.10 | — | 0.25 |
| α | 0° | — | 8° |