

## Table of Contents

<b>Chapter 1</b>	<b>Introduction.....</b>	<b>2</b>
<b>Chapter 2</b>	<b>Validation Platform Setup.....</b>	<b>3</b>
2.1	AP33772S Sink Controller EVB.....	3
2.2	Raspberry Pi 5 .....	4
2.3	Validation Platform Connection and Power up .....	5
<b>Chapter 3</b>	<b>Raspberry Pi Software Setup.....</b>	<b>6</b>
3.1	Raspberry Pi OS .....	6
3.1.1	Download OS Image and Prepare SD Card.....	6
3.1.2	Raspberry PI OS Installation.....	6
3.2	Setup of Required Features.....	7
3.2.1	Raspberry Pi Configuration .....	7
3.2.2	I2C-Tools Installation.....	7
<b>Chapter 4</b>	<b>Basic Command Examples.....</b>	<b>8</b>
4.1	I2C-Tools Command Examples.....	8
4.1.1	Detect all devices attached to I2C - i2cdetect .....	9
4.1.2	Read 1-Byte Length Commands .....	9
4.1.3	Read 2-Bytes Length Commands .....	10
4.1.4	Write 1-Byte Length Commands .....	10
4.1.5	Write 2-Bytes Length Commands .....	11
<b>Chapter 5</b>	<b>Practical Examples .....</b>	<b>12</b>
5.1	Example 1: Bash I2C-Tools Example: ap33772s_getpdo.sh .....	12
5.1.1	Code Details .....	12
5.1.2	Code Execution and Outputs .....	14
5.2	Example 2: Bash I2C-Tools Example: ap33772s_req.sh.....	15
5.2.1	Code Details .....	15
5.2.2	Code Execution and Outputs .....	17
5.3	Example Code Download.....	18
5.3.1	List of Example Codes .....	18
<b>Chapter 6</b>	<b>References .....</b>	<b>19</b>

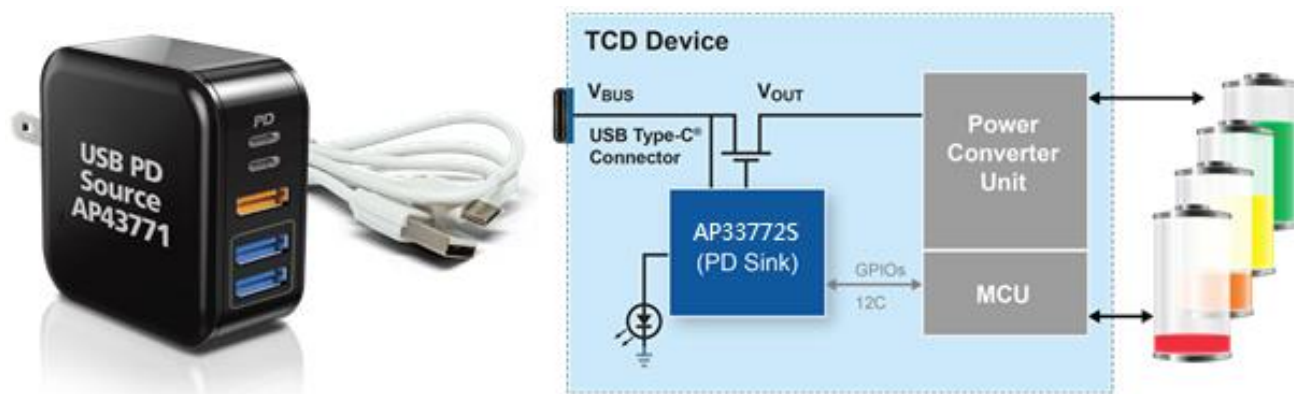
## Chapter 1 Introduction

The AP33772S Sink Controller, working as the protocol device for USB PD3.1 Type C connector-equipped devices (**TCD**, Energy Sink), requests the proper Power Data Object (PDO) from the USB PD3.1 Type C connector-equipped PD3.1 compliance charger (**PDC**, Energy Source).

Figure 1 illustrates a TCD, which is embedded with a PD3.1 Sink controller IC (AP33772S), physically connected to a PDC, which is embedded with a USB PD3.1 decoder (AP43771), through a Type C-to-Type C cable. Based on built-in USB PD3.1 compliant firmware, the AP33772S and AP43771 pair follows the USB PD3.1 standard attachment procedure to establish a suitable PD3.1 charging state.

The AP33772S Sink Controller EVB provides ease of use and great versatility for system designers to request PDOs from USB Power Delivery Chargers by sending the AP33772S's built-in commands through an I2C interface. Typical system design requires MCU programming, which needs specific software (e.g. IDE) setup and can be a time-consuming development process. In contrast, Raspberry Pi (RPI), a single-board computer (SBC) running on a user-friendly Linux OS and equipped with flexible GPIO pins, provides a straightforward way to validate the AP33772S Sink EVB to work with a PD Charger. The goal of this guide is to provide system designers with an effective platform to quickly complete software validation on RPI and port the development to any desirable MCU, meeting rapid turnaround market requirements.

As a supplemental document to the AP33772S EVB User Guide, this User Guide illustrates an easy way to control the AP33772S EVB with an RPI SBC through an I2C Interface. This User Guide covers register definitions and usage information as examples. For the latest and most complete information, please refer to the AP33772S EVB User Guide (see Reference 2).



**Figure 1 – A typical TCD uses the AP33772S PD Sink Controller with an I2C Interface to request power from a USB Type-C PD3.1 Compliance Source Adapter**

## Chapter 2 Validation Platform Setup

### 2.1 AP33772S Sink Controller EVB

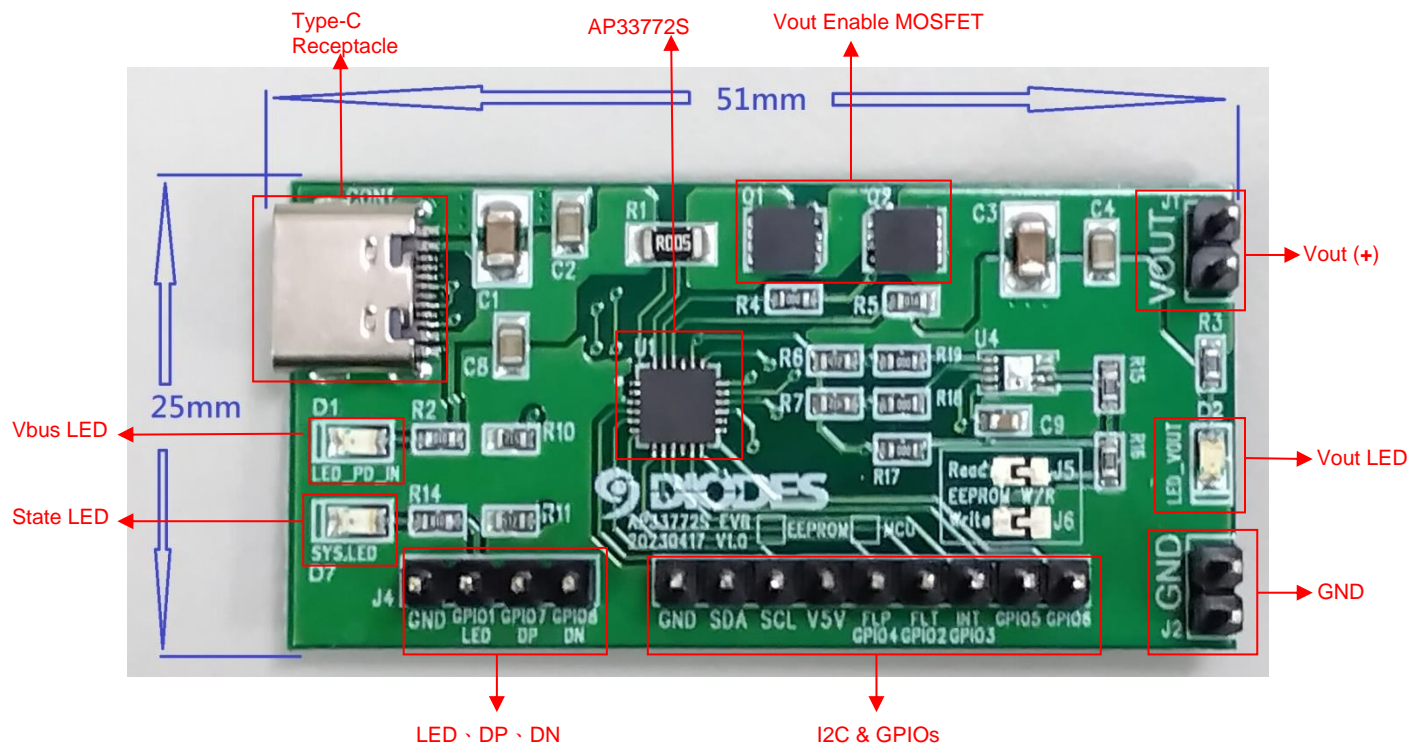


Figure 2 – AP33772S Sink Controller EVB

## 2.2 Raspberry Pi 5

Any modern version of the RPI can control the AP33772S Sink Controller EVB through I2C pins. This User Guide utilizes the Raspberry Pi 5 (RPI 5) for demonstration. WiFi and Bluetooth are also integrated to connect wirelessly without requiring additional components.

Please check the Raspberry Pi official website for additional information: <https://www.raspberrypi.com/products/raspberry-pi-5/>

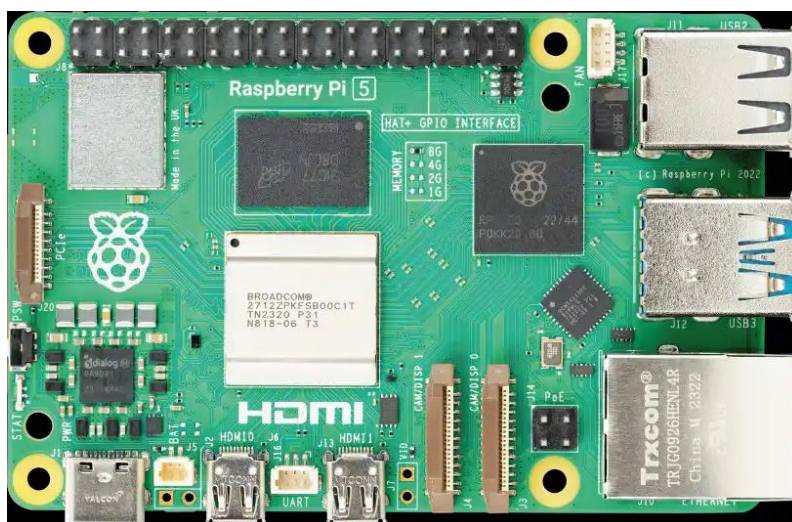


Figure 3 – Raspberry Pi 5 (RPI 5)

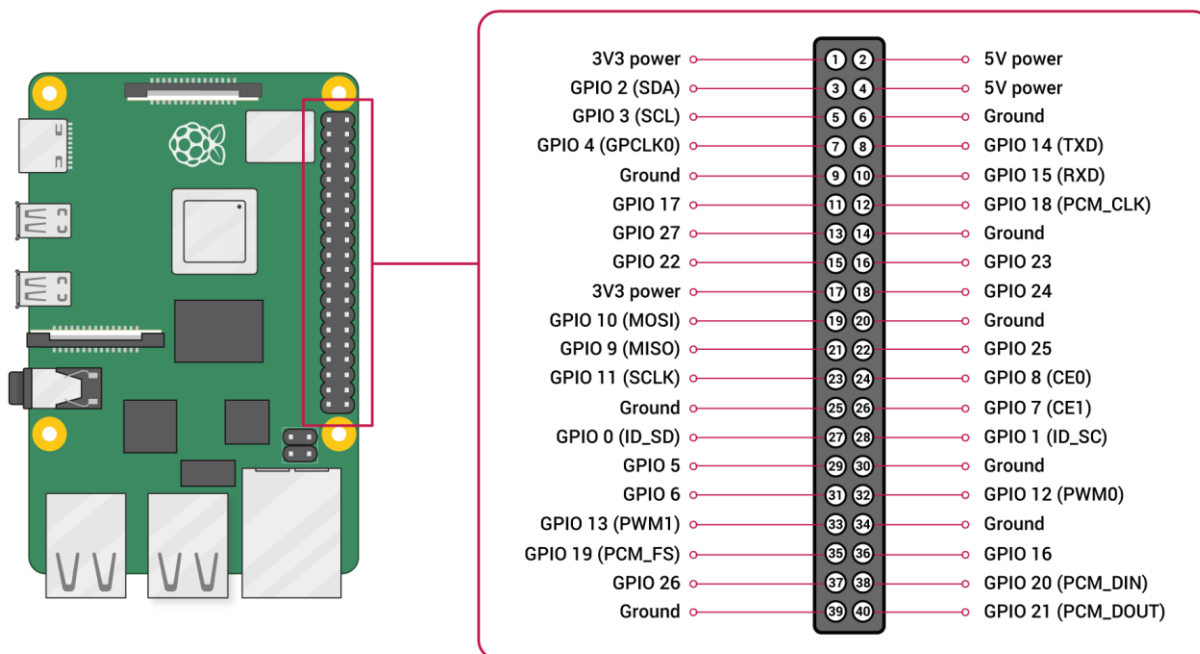


Figure 4 – Raspberry Pi 5 Pinout Diagram

## 2.3 Validation Platform Connection and Power up

Figure 5 shows the complete connection and setup of the Validation Platform. The user should follow these steps below:

1. Connect SCL, SDA, and GND pins between RPI 5 and AP33772S EVB.
2. Connect PD Source and AP33772S EVB with USB Type-C cable.
3. Power up RPI 5 and PD Source.

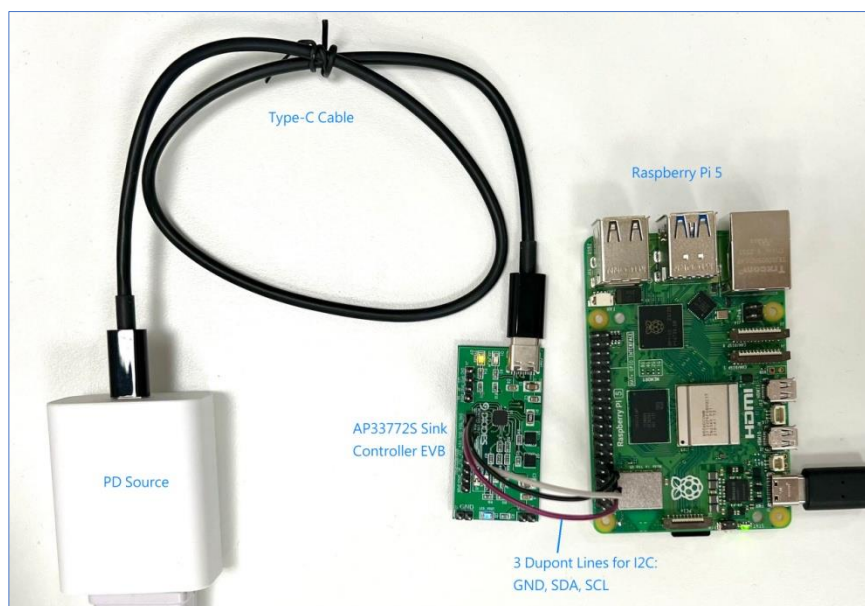


Figure 5 – Complete Setup of the Validation Platform



## Chapter 3 Raspberry Pi Software Setup

### 3.1 Raspberry Pi OS

There are many different operating systems that support RPI, but among these, RPI's official guidelines recommend the widely used Raspberry Pi OS.

#### 3.1.1 Download OS Image and Prepare SD Card

Download then install the Raspberry Pi Imager tool to your selected PC (<https://www.raspberrypi.com/software/>).

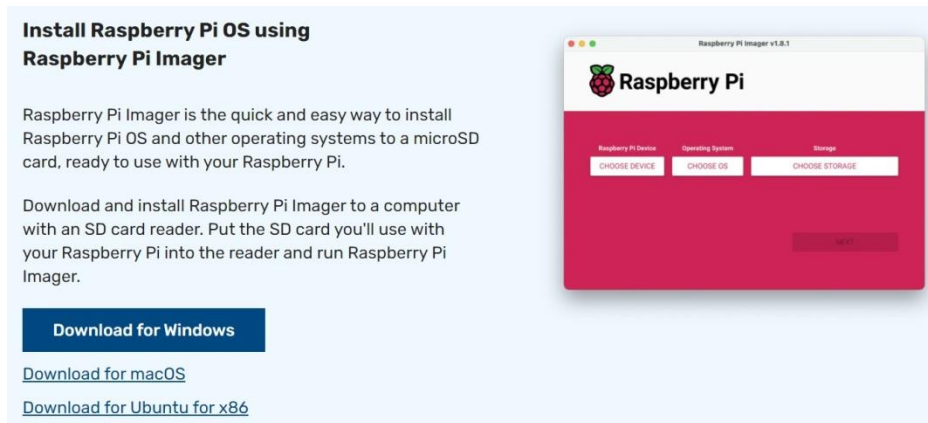


Figure 6 –Download Raspberry Pi Imager Tool

Launch the Raspberry Pi Imager tool and follow the instructions to prepare the Micro-SD for loading the correct OS image (<https://youtu.be/ntaXWS8Lk34/>). Please note that a Micro-SD card of 32GB or greater is recommended.

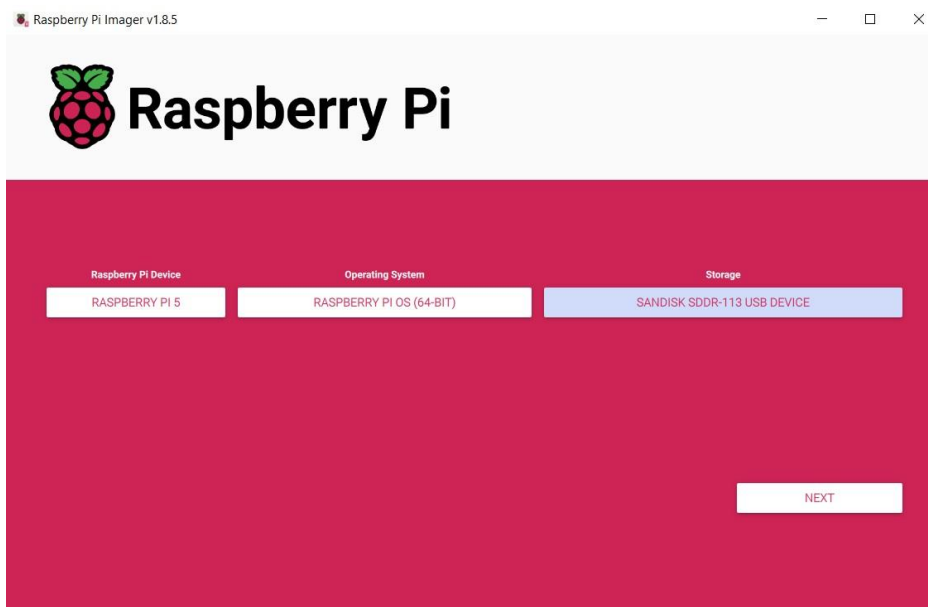


Figure 7 – Raspberry Pi Imager for OS Image Load and Preparation

#### 3.1.2 Raspberry PI OS Installation

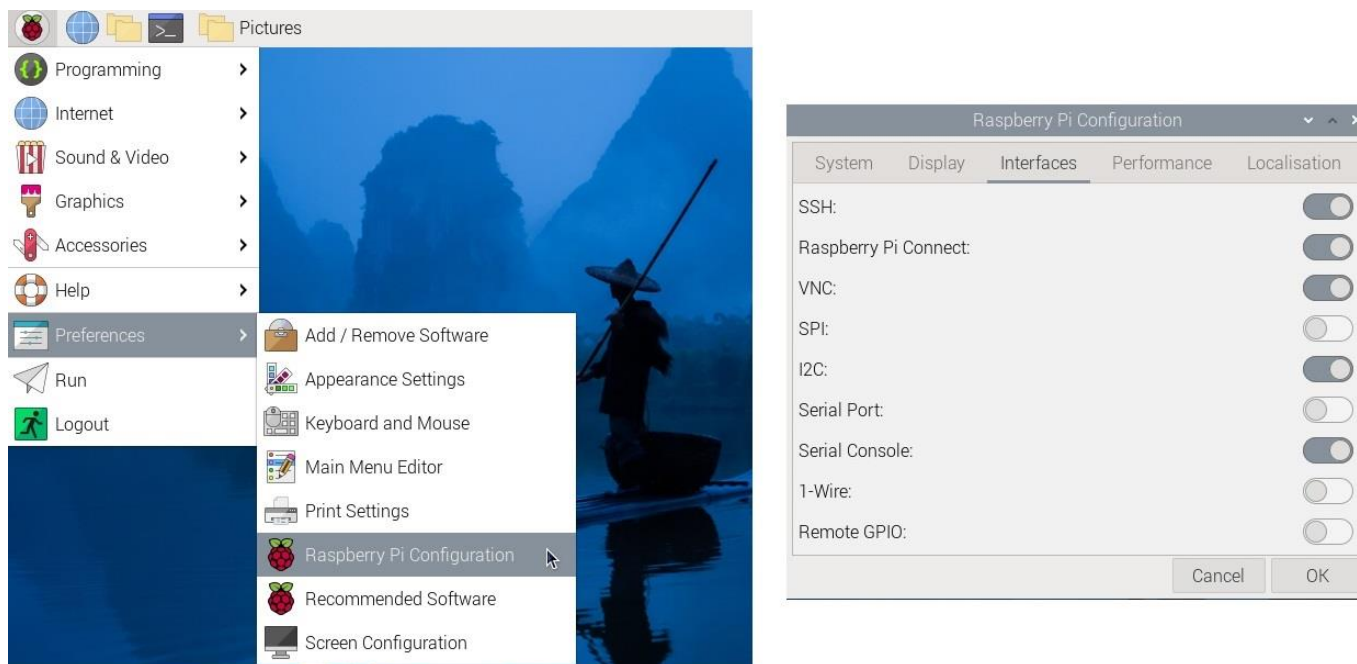
Insert the Micro-SD card that was loaded with the imager into RPI's Micro-SD slot. Connect the power adapter, mouse/keyboard, and HDMI monitor. Power on the RPI and follow the instructions to complete OS installation and basic setup. Make sure the latest updates are included in the OS.

### 3.2 Setup of Required Features

The SSH, VNC, and I2C features must be configured to run the I2C interface on RPI successfully.

#### 3.2.1 Raspberry Pi Configuration

After RPI boot-up, open the “Raspberry Pi Configuration” utility settings and ensure SSH, VNC, and I2C features are turned on.



**Figure 8 – Enable SSH, VNC, and I2C in Raspberry Pi Configuration**

#### 3.2.2 I2C-Tools Installation

I2C-Tools is a toolset that provides simple commands and runs on the command line under Raspberry Pi OS. Install I2C-Tools on the OS by running the following command:

***sudo apt install i2c-tools***

### Chapter 4 Basic Command Examples

This User Guide demonstrates how to operate the I2C interface on RPI. The basic commands are introduced in this section.

#### 4.1 I2C-Tools Command Examples

The I2C-Tools utility package provides i2cdetect, i2cget, and i2cset commands. Simplified usages are described in the examples under this section. For complete information about I2C-Tools' utility, please refer to <https://linuxhint.com/i2c-linux-utilities/>.

Table 1 shows the AP33772S register summary and command usage for user reference. For complete register information, please refer to the AP33772S Sink Controller EVB User Guide.

Register	Command	Length	Attribute	Pwr-on	Description
STATUS	0x01	1	RC	00h	Status
MASK	0x02	1	RW	03h	Interrupt enable mask
OPMODE	0x03	1	RO	00h	Operation mode
CONFIG	0x04	1	RW	F8h	System configuration options
PDCONFIG	0x05	1	RW	03h	PD mode configuration options
SYSTEM	0x06	1	RO/RW	10h	System control and information
TR25	0x0C	2	RW	2710h	Thermal Resistance @25°C, Unit: Ω
TR50	0x0D	2	RW	1041h	Thermal Resistance @50°C, Unit: Ω
TR75	0x0E	2	RW	0788h	Thermal Resistance @75°C, Unit: Ω
TR100	0x0F	2	RW	03CEh	Thermal Resistance @100°C, Unit: Ω
VOLTAGE	0x11	2	RO	0000h	The VOUT Voltage, LSB 80mV
CURRENT	0x12	1	RO	00h	The VOUT Current, LSB 24mA
TEMP	0x13	1	RO	19h	Temperature, Unit: °C The default value is 19h (25°C).
VREQ	0x14	2	RO	0000h	The latest requested voltage negotiated with the source, LSB 50mV
IREQ	0x15	2	RO	0000h	The latest requested current negotiated with the source, LSB 10mA
VSELMIN	0x16	1	RW	19h	The Minimum Selection Voltage, LSB 200mV. The default value is 19h (5000mV).
UVPTHR	0x17	1	RW	01h	UVP Threshold, percentage (%) of VREQ. The default value is 01h (80%).
OVPTHR	0x18	1	RW	19h	OVP Threshold, offset from VREQ. LSB 80mV. The default value is 19h (2000mV).
OCPTHR	0x19	1	RW	00h	OCP Threshold, LSB 50mA
OTPTHR	0x1A	1	RW	78h	OTP Threshold, Unit: °C The default value is 78h (120°C).
DRTHR	0x1B	1	RW	78h	De-Rating Threshold, Unit: °C The default value is 78h (120°C).
SRCPDO	0x20	26	RO	All 00h	Get all PD Source Power Capabilities (SRC_SPR_PDO1 ~ SRC_EPR_PDO13)
SRC_SPR_PDO1	0x21	2	RO	0000h	Source SPR PDO1
SRC_SPR_PDO2	0x22	2	RO	0000h	Source SPR PDO2
SRC_SPR_PDO3	0x23	2	RO	0000h	Source SPR PDO3
SRC_SPR_PDO4	0x24	2	RO	0000h	Source SPR PDO4
SRC_SPR_PDO5	0x25	2	RO	0000h	Source SPR PDO5
SRC_SPR_PDO6	0x26	2	RO	0000h	Source SPR PDO6
SRC_SPR_PDO7	0x27	2	RO	0000h	Source SPR PDO7
SRC_EPR_PDO8	0x28	2	RO	0000h	Source EPR PDO8
SRC_EPR_PDO9	0x29	2	RO	0000h	Source EPR PDO9
SRC_EPR_PDO10	0x2A	2	RO	0000h	Source EPR PDO10
SRC_EPR_PDO11	0x2B	2	RO	0000h	Source EPR PDO11
SRC_EPR_PDO12	0x2C	2	RO	0000h	Source EPR PDO12
SRC_EPR_PDO13	0x2D	2	RO	0000h	Source EPR PDO13
PD_REQMSG	0x31	2	WO	0000h	Send request message with selected voltage, current and PDO index
PD_CMDMSG	0x32	1	WO	00h	Send specific PD command message
PD_MSGRLT	0x33	1	RO	00h	Result and status of PD request or command message
GPIO	0x52	1	RO/RW	00h	GPIO

Table 1 – AP33772S Register Summary



### 4.1.1 Detect all devices attached to I2C - i2cdetect

To display all I2C devices currently attached to I2C-1 bus, type the following command under the command prompt:

***i2cdetect -y 1***

If the AP33772S Sink Controller EVB is attached as shown in Figure 5, the user should see device is attached at address 0x52 (Figure 9).

```
canyon@raspberrypi:~ $ i2cdetect -y 1
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
10:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
20:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
30:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
40:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
50:  -- -- 52 -- -- -- -- -- -- -- -- -- --
60:  -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: 70 -- -- -- -- -- -- -- -- -- -- -- -- -- --
canyon@raspberrypi:~ $
```

Figure 9 – i2cdetect command

### 4.1.2 Read 1-Byte Length Commands

To display the results of a 1-Byte length command with the register attribute including 'R', type the following command format under the command prompt:

***i2cget -y 1 0x52 Command b***

Register	Command	Length	Attribute	Pwr-on
STATUS	0x01	1	RC	00h
MASK	0x02	1	RW	03h
OPMODE	0x03	1	RO	00h
CONFIG	0x04	1	RW	F8h
PDCONFIG	0x05	1	RW	03h
SYSTEM	0x06	1	RO/RW	10h
CURRENT	0x12	1	RO	00h
TEMP	0x13	1	RO	19h
VSELMIN	0x16	1	RW	19h
UVPTHR	0x17	1	RW	01h
OVPTHR	0x18	1	RW	19h
OCPTHR	0x19	1	RW	00h
OTPTHR	0x1A	1	RW	78h
DRTHR	0x1B	1	RW	78h
PD_MSGRLT	0x33	1	RO	00h
GPIO	0x52	1	RO/RW	00h

Table 2 – AP33772S command Summary of 1-Byte length with attribute 'R'

For example, the STATUS Register which Command Address is 0x01. To display the status information of the Sink Controller, type the following under command prompt:

***i2cget -y 1 0x52 0x01 b***

### 4.1.3 Read 2-Bytes Length Commands

To display the results of 2-Bytes length command with the register attribute including 'R', type the following command format under command prompt:

***i2cget -y 1 0x52 Command w***

Register	Command	Length	Attribute	Pwr-on
TR25	0x0C	2	RW	2710h
TR50	0x0D	2	RW	1041h
TR75	0x0E	2	RW	0788h
TR100	0x0F	2	RW	03CEh
VOLTAGE	0x11	2	RO	0000h
VREQ	0x14	2	RO	0000h
IREQ	0x15	2	RO	0000h
SRC_SPR_PDO1	0x21	2	RO	0000h
SRC_SPR_PDO2	0x22	2	RO	0000h
SRC_SPR_PDO3	0x23	2	RO	0000h
SRC_SPR_PDO4	0x24	2	RO	0000h
SRC_SPR_PDO5	0x25	2	RO	0000h
SRC_SPR_PDO6	0x26	2	RO	0000h
SRC_SPR_PDO7	0x27	2	RO	0000h
SRC_EPR_PDO8	0x28	2	RO	0000h
SRC_EPR_PDO9	0x29	2	RO	0000h
SRC_EPR_PDO10	0x2A	2	RO	0000h
SRC_EPR_PDO11	0x2B	2	RO	0000h
SRC_EPR_PDO12	0x2C	2	RO	0000h
SRC_EPR_PDO13	0x2D	2	RO	0000h

**Table 3 – AP33772S command Summary of 2-Bytes length with attribute 'R'**

For example, the VOLTAGE Register's (LSB 80mV) Command Address is 0x11. To report the voltage measured by the Sink Controller, type the following command under the command prompt:

***i2cget -y 1 0x52 0x11 w***

### 4.1.4 Write 1-Byte Length Commands

To set the values of the 1-Byte length command with the register attribute including 'W', type the following command format under the command prompt:

***i2cset -y 1 0x52 Command Data b***

Register	Command	Length	Attribute	Pwr-on
MASK	0x02	1	RW	03h
CONFIG	0x04	1	RW	F8h
PDCONFIG	0x05	1	RW	03h
SYSTEM	0x06	1	RO/RW	10h
VSELMIN	0x16	1	RW	19h
UVPTHR	0x17	1	RW	01h
OVPTHR	0x18	1	RW	19h
OCPTHR	0x19	1	RW	00h
OTPTHR	0x1A	1	RW	78h
DRTHR	0x1B	1	RW	78h
PD_CMDMSG	0x32	1	WO	00h
GPIO	0x52	1	RO/RW	00h

**Table 4 – AP33772S command Summary of 1-Byte length with attribute 'W'**

For example, the MASK Register's Command Address is 0x02. This command enables the interrupts that signal the host through INT pin of AP33772S. To enable a specific interrupt, set the corresponding bit to one. For example, to enable OVP interrupt, set bit 4 of the MASK register to one by typing the following command under the command prompt:

***i2cset -y 1 0x52 0x02 0x10 b***

## 4.1.5 Write 2-Bytes Length Commands

To set the values of the 2-Bytes length command with the register attribute including 'W', type the following command format under the command prompt:

***i2cget -y 1 0x52 Command Data w***

Register	Command	Length	Attribute	Pwr-on
TR25	0x0C	2	RW	2710h
TR50	0x0D	2	RW	1041h
TR75	0x0E	2	RW	0788h
TR100	0x0F	2	RW	03CEh
PD_REQMSG	0x31	2	WO	0000h

**Table 5 – AP33772S command Summary of 2-Bytes length with attribute 'W'**

For example, the PD\_REQMSG Register's Command Address is 0x31. This command initiates a PDO request negotiation procedure. For example, if the source PDO3 is a Fixed\_15V PDO, then set the PD\_REQMSG register to 0x3800. This means to request the PDO index 3 with an operating current of 3A; type the following command under command prompt:

***i2cset -y 1 0x52 0x31 0x3800 w***

## Chapter 5 Practical Examples

### 5.1 Example 1: Bash I2C-Tools Example: ap33772s\_getpdo.sh

This example shows all valid PDOs and information on the PDO index, PDO type, voltage, and current capabilities.

#### 5.1.1 Code Details

```
#!/bin/bash
#
#This program get all source PDO and display the PDO information
#

RPI_I2CBUS=1
I2C_ADDR=0x52
PDO_CMD=0x21    #SRC_SPR_PDO1

printf "\nGet Source PDOs ... \n\n"

for i in {0..12}; do
    #Read PDO info 2-bytes long each
    #Command Address 0x21~0x2D
    PDO=$(i2cget -y $RPI_I2CBUS $I2C_ADDR $((PDO_CMD+i)) w)
    DETECT=$(( ($PDO & 0x8000) == 0x8000))

    if (($DETECT == 1)); then
        TYPE_APDO=$(( ($PDO & 0x4000) == 0x4000))
        #Get CURRENT_MAX and map it to a string
        CURR=$(( ($PDO & 0x3C00) >> 10))
        case "$CURR" in
            0)
                CURR_ST='1.00A'
                ;;
            1)
                CURR_ST='1.25A'
                ;;
            2)
                CURR_ST='1.50A'
                ;;
            3)
                CURR_ST='1.75A'
                ;;
            4)
                CURR_ST='2.00A'
                ;;
            5)
                CURR_ST='2.25A'
                ;;
            6)
                CURR_ST='2.50A'
                ;;
            7)
                CURR_ST='2.75A'
                ;;
            8)
                CURR_ST='3.00A'
                ;;
            9)
                CURR_ST='3.25A'
                ;;
            10)
                CURR_ST='3.50A'
                ;;
            11)
                CURR_ST='3.75A'
                ;;
            12)
                CURR_ST='4.00A'
                ;;
            13)
                CURR_ST='4.25A'
                ;;
        esac
    fi
done
```

```

14)          CURR_ST='4.50A'
           ;;

15)          CURR_ST='5.00A'
           ;;

esac
#Get VOLTAGE_MAX and change units to V(voltage)
#then get VOLTAGE_MIN and map it to a string
if (($TYPE_APDO == 1)); then
    if (($i >= 7)); then
        VOLT=$(( ($PDO & 0x00FF)*200/1000 ))
        VOLTMIN=$(( ($PDO & 0x0300) >>8 ))
        case "$VOLTMIN" in
            1)          VOLTMIN_ST='15'
                       ;;
            2)          VOLTMIN_ST='20'
                       ;;
            *)          VOLTMIN_ST='0'
                       ;;
        esac
        printf "PDO[%d] : AVS  %s~%dV %s \n" $((i+1)) $VOLTMIN_ST $VOLT $CURR_ST
    else
        VOLT=$(( ($PDO & 0x00FF)*100/1000 ))
        VOLTMIN=$(( ($PDO & 0x0300) >>8 ))
        case "$VOLTMIN" in
            1)          VOLTMIN_ST='3.3'
                       ;;
            2)          VOLTMIN_ST='5'
                       ;;
            *)          VOLTMIN_ST='0'
                       ;;
        esac
        printf "PDO[%d] : PPS  %s~%dV %s \n" $((i+1)) $VOLTMIN_ST $VOLT $CURR_ST
    fi
else
    if (($i >= 7)); then
        VOLT=$(( ($PDO & 0x00FF)*200/1000 ))
    else
        VOLT=$(( ($PDO & 0x00FF)*100/1000 ))
    fi
    printf "PDO[%d] : Fixed  %dV %s \n" $((i+1)) $VOLT $CURR_ST
fi

done
printf "\n"

exit 0

```



### 5.1.2 Code Execution and Outputs

Type the following command under the command prompt:

***bash ap33772s\_getpdo.sh***

The output display differs depending on the source PDO the AP33772S EVB is connected to. For example, an EVB connected to the APPLE 140W adapter output is as follows.

```
canyon@raspberrypi:~/bin $ bash ap33772s_getpdo.sh  
  
Get Source PDOs ...  
  
PDO[1] : Fixed 5V 3.00A  
PDO[2] : Fixed 9V 3.00A  
PDO[3] : Fixed 15V 3.00A  
PDO[4] : Fixed 20V 5.00A  
PDO[8] : Fixed 28V 5.00A  
PDO[9] : AVS 15~28V 5.00A
```

Figure 10 – Example ap33772s\_getpdo.sh Output Display

## 5.2 Example 2: Bash I2C-Tools Example: ap33772s\_req.sh

This example shows all valid PDOs and requests all PDOs one by one. Afterwards, a demo of PDO1 is requested through a special request message.

### 5.2.1 Code Details

```
#!/bin/bash
#
#This program get all source PDO and request all PDO one by one
#Finally, a demo of PDO1 is requested through a special request message
#

RPI_I2CBUS=1
I2C_ADDR=0x52
STATUS=0x01 #1-byte: STATUS
VOLTAGE=0x11 #2-byte: VOLTAGE
PDO_CMD=0x21 #2-byte: SRC_SPR_PDO1
REQ_CMD=0x31 #2-byte: PD_REQMSG
PDOS=()

#Demo get all source PDO
#Read SRC_SPR_PDOx(0x21~0x27) and SRC_EPR_PDOx(0x28~0x2D)
#

status=$(i2cget -y $RPI_I2CBUS $I2C_ADDR $STATUS b)
printf "\nCheck AP33772S Status : 0x%2X \n" $status

printf "Get Source PDOs ..."
for i in {0..12}; do
    sleep 0.1
    PDO=$(i2cget -y $RPI_I2CBUS $I2C_ADDR ${SPDO_CMD+i} w)
    PDOS=${PDOS[@]} $PDO
done
printf "... OK \n\n"
sleep 1

#Demo request source PDO
#Write PD_REQMSG(0x31)
#
printf "===== \n"
printf "Start to request all source PDO ... \n\n"
for i in ${!PDOS[@]}; do
    DETECT=$(( ${PDOS[i]} & 0x8000 ) == 0x8000 )
    if (( $DETECT == 1 )); then
        #Generate Request Message
        TYPE_APDO=$(( ${PDOS[i]} & 0x4000 ) == 0x4000 )
        CURR=$(( ${PDOS[i]} & 0x3C00 ) >> 10 )
        printf "Request PDO[%d] \n" ${i+1}

        if (( $i >= 7 )); then
            VOLT=$(( ( ${PDOS[i]} & 0x00FF ) * 200 / 1000 ) )
        else
            VOLT=$(( ( ${PDOS[i]} & 0x00FF ) * 100 / 1000 ) )
        fi

        if (( $TYPE_APDO == 1 )); then
            RDO=$(( ( ${i+1} << 12 | $CURR << 8 ) | $VOLT ) )
        else
            RDO=$(( ( ${i+1} << 12 ) | $CURR << 8 ) )
        fi
    fi
done
```

```
#Request PDO
i2cset -y $RPI_I2CBUS $I2C_ADDR $REQ_CMD $RDO w
printf "Write PD_REQMSG : 0x%4X \n" $RDO

#Wait STATUS.READY=1
status=0
while (( ($status & 0x02) != 0x02 )); do
    status=$(i2cget -y $RPI_I2CBUS $I2C_ADDR $STATUS b)
    sleep 0.1
done

#Read Voltage
voltage=$(i2cget -y $RPI_I2CBUS $I2C_ADDR $VOLTAGE w)
voltage=$(( $voltage * 80 ))
printf "Read VOLTAGE : %d mV \n\n" $voltage
sleep 5
fi
done

#Demo special request message
#Write PD_REQMSG(0x31)=0x1FFF
#
printf "Request PDO[1] Fixed 5V ... \n"
i2cset -y $RPI_I2CBUS $I2C_ADDR $REQ_CMD 0x1FFF w
printf "Write PD_REQMSG : 0x1FFF \n\n"

exit 0
```

### 5.2.2 Code Execution and Outputs

Type the following command under the command prompt:

***bash ap33772s\_req.sh***

The output display differs depending on the source PDO the AP33772S EVB is connected to. For example, an EVB connected to the APPLE 140W adapter output is as follows.

```
canyon@raspberrypi:~/bin $ bash ap33772s_req.sh

Check AP33772S Status : 0x 7
Get Source PDOs ..... OK

=====
Start to request all source PDO ...

Request PDO[1]
Write PD_REQMSG : 0x1800
Read VOLTAGE : 5280 mV

Request PDO[2]
Write PD_REQMSG : 0x2800
Read VOLTAGE : 8960 mV

Request PDO[3]
Write PD_REQMSG : 0x3800
Read VOLTAGE : 15040 mV

Request PDO[4]
Write PD_REQMSG : 0x4F00
Read VOLTAGE : 20480 mV

Request PDO[8]
Write PD_REQMSG : 0x8F00
Read VOLTAGE : 28000 mV

Request PDO[9]
Write PD_REQMSG : 0x9F1C
Read VOLTAGE : 28000 mV

Request PDO[1] Fixed 5V ...
Write PD_REQMSG : 0x1FFF
```

Figure 11 – Example ap33772s\_req.sh Output Display

The output waveform is as follows:



Figure 12 – Example ap33772s\_req.sh Output Waveform

## 5.3 Example Code Download

### 5.3.1 List of Example Codes

1. **ap33772s\_getpdo.sh**: retrieves all valid PDOs and lists PDO information
2. **ap33772\_req.sh**: retrieves all valid PDOs and requests all PDOs one by one



## Chapter 6 References

1. AP33772S Datasheet (I2C Interface USB PD3.1 EPR Sink Controller): <https://www.diodes.com/part/view/AP33772S/>
2. AP33772S I2C USB PD Sink Controller EVB User Guide: <https://www.diodes.com/assets/Evaluation-Boards/AP33772S-Sink-Controller-EVB-User-Guide.pdf/>
3. Raspberry Pi Zero 2 W: <https://www.raspberrypi.com/products/raspberry-pi-5/>
4. Raspberry Pi OS: <https://www.raspberrypi.com/software/>
5. I2C-Tools utility: <https://linuxhint.com/i2c-linux-utilities/>

## IMPORTANT NOTICE

1. DIODES INCORPORATED (Diodes) AND ITS SUBSIDIARIES MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARDS TO ANY INFORMATION CONTAINED IN THIS DOCUMENT, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION).
2. The Information contained herein is for informational purpose only and is provided only to illustrate the operation of Diodes' products described herein and application examples. Diodes does not assume any liability arising out of the application or use of this document or any product described herein. This document is intended for skilled and technically trained engineering customers and users who design with Diodes' products. Diodes' products may be used to facilitate safety-related applications; however, in all instances customers and users are responsible for (a) selecting the appropriate Diodes products for their applications, (b) evaluating the suitability of Diodes' products for their intended applications, (c) ensuring their applications, which incorporate Diodes' products, comply the applicable legal and regulatory requirements as well as safety and functional-safety related standards, and (d) ensuring they design with appropriate safeguards (including testing, validation, quality control techniques, redundancy, malfunction prevention, and appropriate treatment for aging degradation) to minimize the risks associated with their applications.
3. Diodes assumes no liability for any application-related information, support, assistance or feedback that may be provided by Diodes from time to time. Any customer or user of this document or products described herein will assume all risks and liabilities associated with such use, and will hold Diodes and all companies whose products are represented herein or on Diodes' websites, harmless against all damages and liabilities.
4. Products described herein may be covered by one or more United States, international or foreign patents and pending patent applications. Product names and markings noted herein may also be covered by one or more United States, international or foreign trademarks and trademark applications. Diodes does not convey any license under any of its intellectual property rights or the rights of any third parties (including third parties whose products and services may be described in this document or on Diodes' website) under this document.
5. Diodes' products are provided subject to Diodes' Standard Terms and Conditions of Sale (<https://www.diodes.com/about/company/terms-and-conditions/terms-and-conditions-of-sales/>) or other applicable terms. This document does not alter or expand the applicable warranties provided by Diodes. Diodes does not warrant or accept any liability whatsoever in respect of any products purchased through unauthorized sales channel.
6. Diodes' products and technology may not be used for or incorporated into any products or systems whose manufacture, use or sale is prohibited under any applicable laws and regulations. Should customers or users use Diodes' products in contravention of any applicable laws or regulations, or for any unintended or unauthorized application, customers and users will (a) be solely responsible for any damages, losses or penalties arising in connection therewith or as a result thereof, and (b) indemnify and hold Diodes and its representatives and agents harmless against any and all claims, damages, expenses, and attorney fees arising out of, directly or indirectly, any claim relating to any noncompliance with the applicable laws and regulations, as well as any unintended or unauthorized application.
7. While efforts have been made to ensure the information contained in this document is accurate, complete and current, it may contain technical inaccuracies, omissions and typographical errors. Diodes does not warrant that information contained in this document is error-free and Diodes is under no obligation to update or otherwise correct this information. Notwithstanding the foregoing, Diodes reserves the right to make modifications, enhancements, improvements, corrections or other changes without further notice to this document and any product described herein. This document is written in English but may be translated into multiple languages for reference. Only the English version of this document is the final and determinative format released by Diodes.
8. Any unauthorized copying, modification, distribution, transmission, display or other use of this document (or any portion hereof) is prohibited. Diodes assumes no responsibility for any losses incurred by the customers or users or any third parties arising from any such unauthorized use.
9. This Notice may be periodically updated with the most recent version available at <https://www.diodes.com/about/company/terms-and-conditions/important-notice>

The Diodes logo is a registered trademark of Diodes Incorporated in the United States and other countries.  
All other trademarks are the property of their respective owners.  
© 2025 Diodes Incorporated. All Rights Reserved.

[www.diodes.com](http://www.diodes.com)