SightLine APPLICATIONS

# EAN-GPIO and I²C

**PN:** EAN-GPIO-and-I²C

1/24/2019

**Contact:**

Web: sightlineapplications.com

Sales: sales@sightlineapplications.com

Support: support@sightlineapplications.com

Phone: +1 (541) 716-5137

### Export Controls

Exports of SightLine products are governed by the US Department of Commerce, Export Administration Regulations (EAR); classification is ECCN 4A994. The export summary sheet located on the support/documentation page of our website outlines customers responsibilities and applicable rules. SightLine Applications takes export controls seriously and works to stay compliant with all export rules.

### Copyright and Use Agreement



### Alerts

The following notifications are used throughout the document to help identify important safety and setup information to the user:

⚠ **CAUTION:** Alerts to a potential hazard that may result in personal injury, or an unsafe practice that causes damage to the equipment if not avoided.

ⓘ **IMPORTANT:** Identifies crucial information that is important to setup and configuration procedures.

*Used to emphasize points or reminds the user of something. Supplementary information that aids in the use or understanding of the equipment or subject that is not critical to system use.*

## Contents

**List of Figures**



**List of Tables**

# 1 Overview

This document covers how to configure GPIO lines to work with a custom ARM application that triggers the 1500 or 3000-OEM to take a snapshot, or to be able to start a video recording using a button. It also covers how to send commands through the video processing boards for communicating with I²C devices. The ability to perform I2C read and write commands through SLA Command Protocol (SLAI2CCommand) has additionally been added in release 2.24.xx.

## 1.1 Associated Documents

EAN-Startup Guide 1500-OEM: Describes steps for connecting, configuring, and testing the 1500-OEM video processing board on the 1500-AB accessory board.

EAN-Startup Guide 3000-OEM: Describes steps for connecting, configuring, and testing the 3000-OEM video processing board on the 3000-IO interface board.

EAN-ARM Application Development: Describes how to setup a PC to develop applications that can be run on the ARM processor of the 1500-OEM or the 3000-OEM video processing boards.

Interface Command and Control (IDD): Describes the native communications protocol used by the SightLine Applications product line. The IDD is also available as a local download on the Software Download page.

Panel Plus User Guide: A complete overview of settings and dialog windows in Panel Plus. Located in The Help menu of the Panel Plus application.

## 1.2 SightLine Software Requirements

ⓘ **IMPORTANT:** The Panel Plus software version should match the firmware version running on the board.

### 1.2.1 Third Party Software

Tera Term or PuTTY: Terminal emulator programs used for debug output, or to issue commands on SLA hardware.

## 2 GPIO Framework

| | |
|---|---|
| **GPIO** | General Purpose IO lines found on the J4 connector of the 1500-OEM. |
| **1500-OEM and 3000-OEM** | Main processing boards that have an ARM processor for supporting customer developed applications. |
| **VideoTrack 1500** | A custom application for the ARM developed by SightLine that handles primary command and control for the 1500-OEM. |
| **SightLine Command and Control** | Primary API for communicating with the 1500-OEM either through serial port 0 or Ethernet. |
| **1500-AB** | A bench-top interface board for working with the 1500-OEM that exposes the GPIO through headers. |
| **3000-USB** | Interface board for working with the 3000-OEM that exposes the GPIO through J3 and J5. |
| **3000-IO** | IO board for working with the 3000-OEM board. |

The GPIO pins are available on the J4 (50 pin) connector on the 1500-OEM. This connector is also used to attach to a variety of digital camera input boards (Tau, Sony, Hitachi). If an external camera interface board is attached, then the GPIO pins will not be available.

The examples below refer to connector J10 of the 1500-AB board. This is functionally equivalent to the J4 connector of the 1500-OEM.

The GPIO pins are available on the J2 (60 pin) connector on the 3000-OEM. The examples below will often refer to the J3 and J5 connectors of the 3000-USB board.

# 3 VIOSEL - Powering the 1500-OEM IO

The voltage level on all GPIO pins is set using VIOSEL. A voltage of 3.3V must be supplied to J10:46 (VIOSEL) to enable the GPIO. All GPIO output levels will follow VIOSEL. This must match the voltage output level (CMOS) of camera digital data, which is typically 3.3V.

3.3V and ground can be supplied from an external source or use the 3.3V power and GND available through the following points on the 1500-AB board:

- J9 Pin 10 (3.3V) and J9 Pin 12 (GND) [REV A and above]
- The test points labeled 3.3V and GND [Rev D].

**Table 1: 1500-OEM IO Power and Ground**

| J10 Pin # | Signal Name | Values | Purpose |
|---|---|---|---|
| 46 | VIOSEL | 1.8V, 2.5V, 3.3V | Sets voltage level for some GPIO |
| 18 | TAUDET | GND, NC, 1.8V | Enables or disables some GPIO |
| 5 | GND | Ground | |
| 17 | GND | Ground | |
| 27 | GND | Ground | |
| 37 | GND | Ground | |
| 41 | GND | Ground | |
| 45 | GND | Ground | |
| 47 | GND | Ground | |
| 49 | GND | Ground | |

# 4 GPIO Connections

## 4.1 1500-AB

Five GPIO ports are available through headers on the 1500-AB boards.

**Table 2: 1500-AB Rev E J10 Pin**

| J10 Pin # | Signal Name | Voltage Level | Default Direction | TAUDET | |
|---|---|---|---|---|---|
| | | | | Enable | Disable |
| 9 | GPIO175 | VIOSEL | Input | NC (PULLUP) | GND |
| 13 | GPIO174 | VIOSEL | Input | | |
| 14 | GPIO173 | VIOSEL | Input | | |
| 40 | GPIO172* | VIOSEL | Input | | |
| 19 | GPIO178 | 3.3V | Input | NA | NA |

**GPIO172 is used as a receiver enable signal for some camera adapter boards. These include the 1500-Sony board and the 1500-CL board.*

**Table 3: 1500-AB Rev J J11 Pin**

| J11 Pin # | Signal Name | Voltage Level | Default Direction | TAUDET | |
|---|---|---|---|---|---|
| | | | | Enable | Disable |
| 13 | GPIO172 | VIOSEL | Input | NC (PULLUP) | GND |
| 14 | GPIO173 | VIOSEL | Input | | |
| 15 | GPIO174 | VIOSEL | Input | | |
| 16 | GPIO175 | VIOSEL | Input | | |
| 17 | GPIO178 | 3.3V | Input | NA | NA |

*GPIO 144 and 145 are also available at 3.3V as test points on 1500-AB REV J board (see Figure 2).*

### 4.1.1 1500-OEM (Rev E)

The following GPIOs are available on J5.

**Table 4: 1500-OEM (Rev E) J5 Pin**

| J5 Pin # | Signal Name | Voltage Level | Default Direction | TAUDET | |
|---|---|---|---|---|---|
| | | | | Enable | Disable |
| 9 | GPIO145 | 1.8V | Input | NA | NA |
| 12 | GPIO144 | 1.8V | Input | NA | NA |

## 4.2 3000-OEM

Six GPIO are available on the 3000-USB board through the 3-pin header pins (J3 and J5).

The GPIO default as inputs. These can be changed in the software application (see Linux Hardware Configuration).

**Table 5: 3000-OEM J3 Pin**

| J3 Pin # | Signal Name | Voltage Level | Default Direction | TAUDET | |
|---|---|---|---|---|---|
| | | | | Enable | Disable |
| 1 | GPIO16 | 3.3V | Input | NC (PULLUP) | GND |
| 2 | GPIO22 | 3.3V | Input | | |
| 3 | GPIO24 | 3.3V | Input | | |

**Table 6: 3000-OEM J5 Pin**

| J5 Pin # | Signal Name | Voltage Level | Default Direction | TAUDET | |
|---|---|---|---|---|---|
| | | | | Enable | Disable |
| 1 | GPIO17 | 3.3V | Input | NC (PULLUP) | GND |
| 2 | GPIO23 | 3.3V | Input | | |
| 3 | GPIO28 | 3.3V | Input | | |

# 5 1500-AB Board Setup

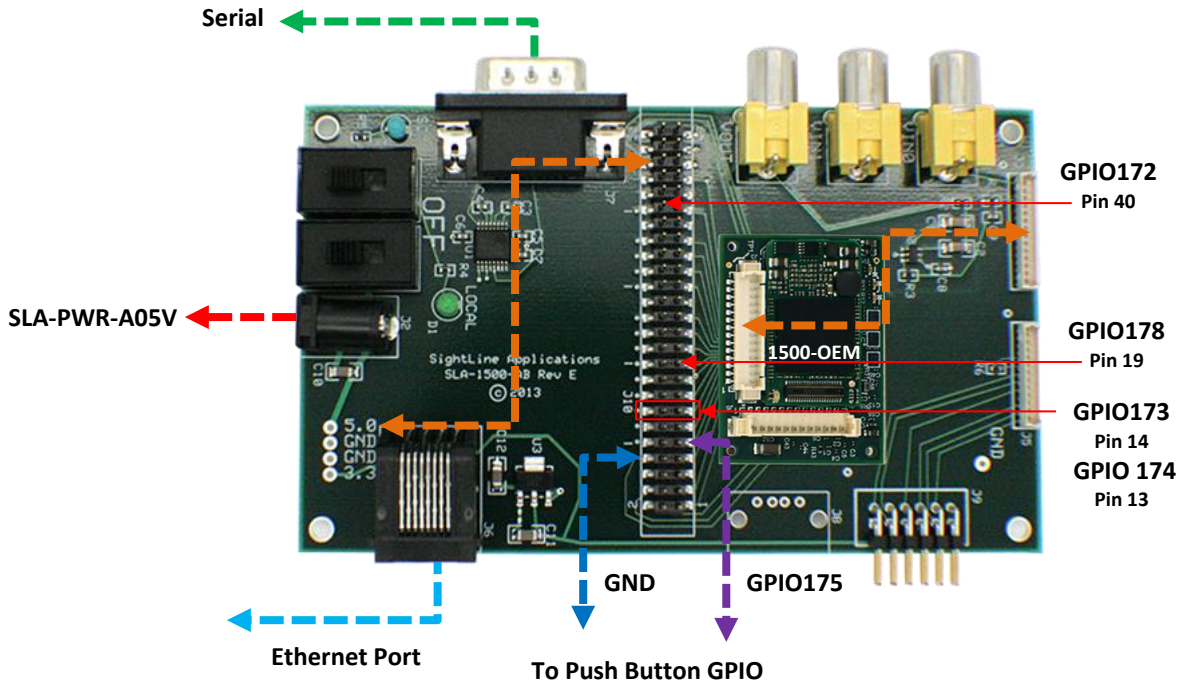



Figure 1: 1500-AB (Rev E) + 1500-OEM (Rev C) GPIO Setup

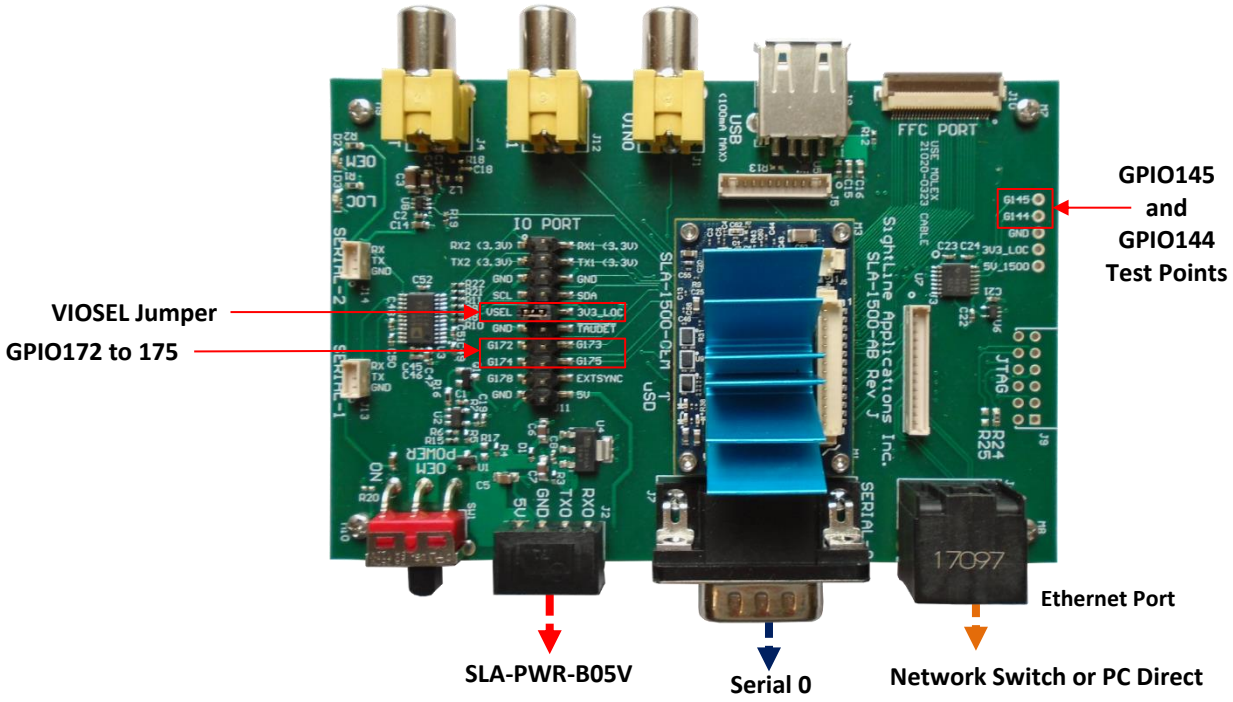



Figure 2: 1500-AB (Rev J) + 1500-OEM (Rev E) GPIO Setup

# 6 3000-USB Board Setup

Connect the 3000-USB board to the 3000-OEM board.

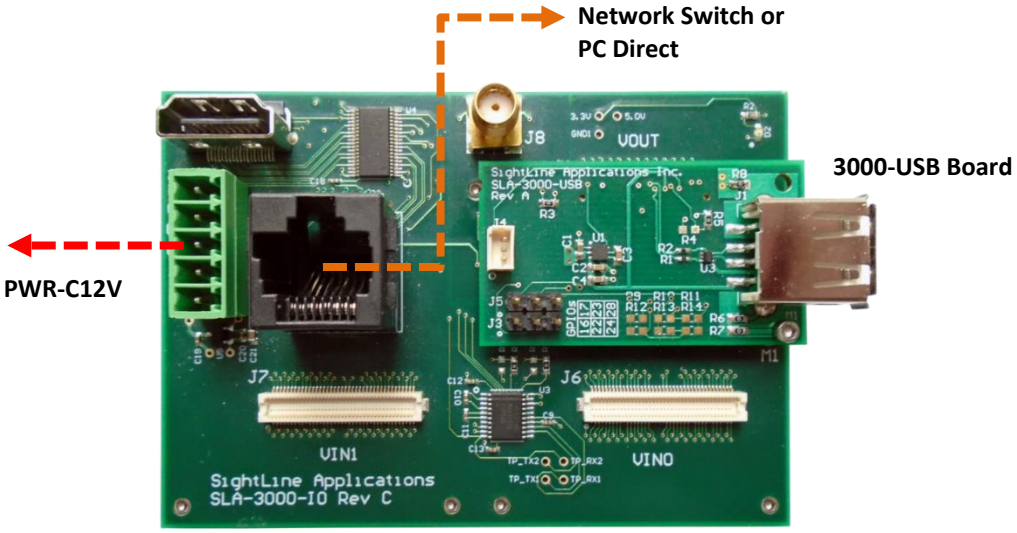



**Figure 3: 3000-OEM + 3000-IO + 3000-USB**

# 7 GPIO Circuit and Requirements

## 7.1 GPIO Level Translators and Output Impedance

GPIO signal levels are voltage translated to match VIOSEL. To allow GPIOs to be used as both inputs and outputs bi-directional voltage translators are used. SLA hardware currently uses Texas Instruments TXB0104 voltage translators with automatic direction sensing.

Texas Instruments provides a comprehensive guide to working with the TXB output.

The TXB chip has an output impedance of 4K ohm. This chip is intended to have a very weak output drive (~2ma), with the intention that an external device with a stronger drive can overcome this weak drive and reverse the direction of the translator. This allows GPIOs to work as inputs and outputs without separate I/O direction select lines.

External hardware attached to this output must have an input impedance of 50K ohm or higher. This is due to the weak drive and resultant bidirectional nature. Input capacitance of external hardware must be less than 70Pf. The capacitance requirement can mean short wire lengths. Issues have been reported using scope probes in 1X mode vs 10X mode (10X has higher impedance than 1X).

For external circuitry that has low input impedance a MOSFET should be part of the design. This provides high input impedance to the TXB and proper drive levels for external circuitry.

Chose a MOSFET that meets the voltage and current needs of the external circuitry with a Vgs(th) max that is less than the VIOSEL voltage (typically VIOSEL= 3.3V). Vgs(th) is the voltage at which the MOSFET will start to turn on. We recommend choosing a Vgs (th) less than 3.3V, e.g., 2V.

The max and min specs show possible variances among individual devices. In this case choosing the max value is the best option.

### 7.2 GPIO Input

Button closure

Requires an external pullup resistor

VCC = VIOSEL (or 3.3V for GPIO178)

R1 ~= 1 K ohm

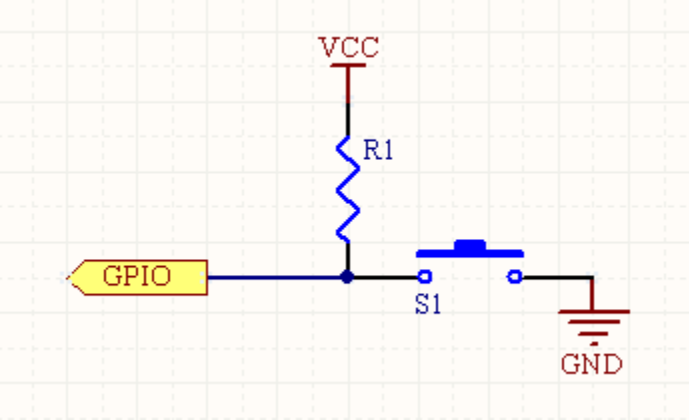



**Figure 4: Push Button GPIO**

### 7.3 GPIO Output

Drives LED

Requires an external circuit

VCC = VIOSEL (or 3.3V for GPIO178)

R1 ~= 500 ohm (depends on LED)

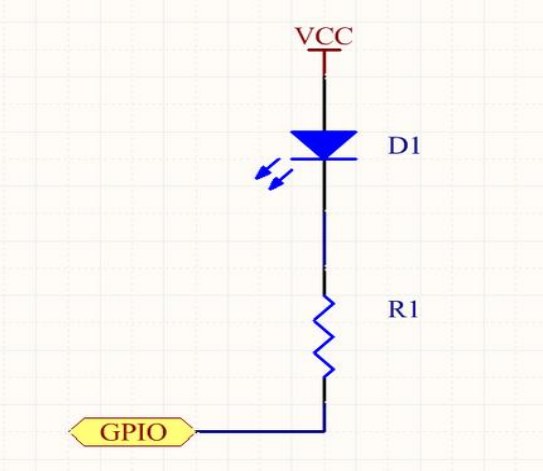



**Figure 5: GPIO Drives LED**

# 8 Testing GPIO

GPIO can be exercised from the Linux command line. This is useful when testing the GPIO circuit.

## 8.1 Configure GPIO for Output

```
echo 175 > /sys/class/gpio/export
cat /sys/class/gpio/gpio175/direction
echo out > /sys/class/gpio/gpio175/direction
echo “0” > /sys/class/gpio/gpio175/active_low
echo “0” > /sys/class/gpio/gpio175/value
echo “1” > /sys/class/gpio/gpio175/value
```

## 8.2 Configure GPIO for Input

```
echo 175 > /sys/class/gpio/export
cat /sys/class/gpio/gpio175/direction
echo “in” > /sys/class/gpio/gpio175/direction
echo “0” > /sys/class/gpio/gpio175/active_low
echo “both” > /sys/class/gpio/gpio175/edge
for (( ; ; ))
do
  cat /sys/class/gpio/gpio175/value
done
```

# 9 GPIO Example Application

An example Linux application is available in the SLAArmExamplesInstaller project. The installer creates a compilable example in C:\SightLine Applications directory of the host PC.

## 9.1 SLAGPIO\gpioMain2013

A Visual Studio 2013 project which generates a PC console application (gpioMain) that does the following:

- Communicates with 1500-OEM or 3000-OEM hardware using the FIP protocol.
- Reads and reports current SD Card video record status.
- Toggles SD Card video record on/off with the F1 function key.
- Intended as an easy way of understanding and debugging interface with FIP and connection to 1500-OEM and 3000-OEM. Uses the same main code/threads/objects as the Linux application.
- Application requires changing the IP address *#define IPADDR_VIDEOTRACK 192.168.1.107* to match the IP address of the SLA hardware. This can be read by connecting Panel Plus to the SLA hardware and then reading the IP address from the lower left corner of the application.

### 9.2 GPIO Linux Application

A Code Composer Studio 5 project. Details of installation and compiling are in the EAN-ARM Application Development document.

GPIO requires Ubuntu Linux image *UbuntuSLA_2_21.7z* or higher to enable GPIO pins. The 1500-OEM board requires installation of VideoTrack1500 application 2.21 or higher. The 3000-OEM board requires the VT3000_Release application 2.22 or higher.

Code is common with the PC application, except for the implementation of the *GPIOCtrl* class, which uses Linux GPIO pins as opposed to the F1 key.

The application uses the following GPIO pins by default:

**1500-OEM**

- GPIO178 (input) - SD Card video record toggle.
- GPIO174 (input) - Initiate an SD card snapshot. Snapshot settings must be setup and persisted using the Panel Plus application.
- GPIO175 (output) - Display SD Card video record status on an LED.

**3000-OEM**

- GPIO16 (input) - SD Card video record toggle.
- GPIO22 (input) - Initiate an SD card snapshot. Snapshot settings must be setup and persisted using the Panel Plus application.
- GPIO24 (output) - Display SD Card video record status on an LED.

## 10 Linux Hardware Configuration

Once the GPIO Linux application has been customized, it must be setup to run in parallel with the VideoTrack1500 application on the 1500-OEM hardware. This requires modifying the */etc/rc.d/rc.local* file on the 1500-OEM.

Section of code that needs modified:

```
if [ -f slStabTrackOMAP.out ]; then
    #./VideoTrack1500 -S0 0 -Q -CAPTURE sony_720p &
    #sleep 10
    #./SLAGimbal &
    ./VideoTrack1500 -Q &
```

Modify code to:

```
if [ -f slStabTrackOMAP.out ]; then
    ./VideoTrack1500 -Q &
    sleep 20
    ./SLAGPIO1500 &
```

*The -Q option runs VideoTrack1500 in quiet mode. The sleep command is to let the FIP command processing begin.*

*A similar change should be made to the /home/root/sla3000_init.sh file on the 3000-OEM.*

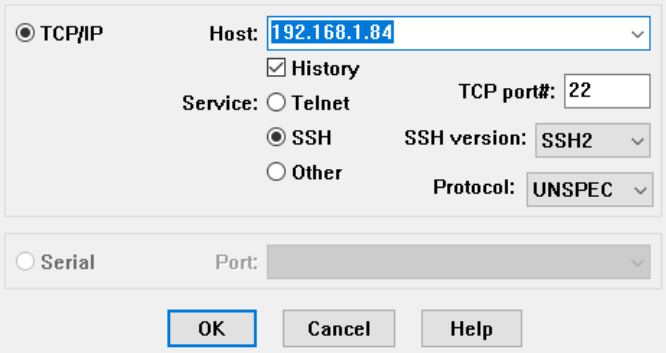
## 11 Configuring I²C

This section describes how to login into the 1500-OEM and 3000-OEM boards using Tera Term and read/write data to support I²C devices using Linux commands. It also provides simple example C code that reads/writes I²C data.

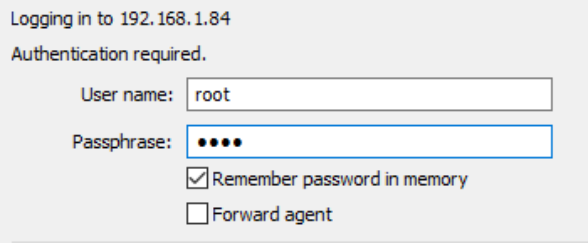
For testing and system configuration, read and write commands I²C commands can be performed through the SLA Command Protocol (SLA I²C Command, release 2.24). Details on command structure and examples of I²C read/write operations are included in the IDD.

Login to the 1500-OEM and 3000-OEM:

1. Open the Tera Term application.
2. From the main menu go to *File » New Connection*.
3. Enter the IP address of the OEM board and click *OK*.




4. Enter the User name (*root*) and password (*root*), and then click *OK*. If prompted to key cache, click *Yes* (or *OK*).




5. At the command prompt, use the following commands when working with the I²C bus:
   - *I2cdetect* - Lists devices on the bus
   - *I2cget* - Read registers from the I²C device
   - *I2cset* - Write registers on the I²C device

## 11.1 I²C Bus Tera Term Commands

Listing I²C buses:

```
root@sla3000:~# i2cdetect -l
i2c-1   i2c             OMAP I2C adapter                        I2C adapter
i2c-4   i2c             OMAP I2C adapter                        I2C adapter
root@sla3000:~#
```

Listing I²C devices on the bus:

```
root@sla3000:~# i2cdetect -a -r -y 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00: -- -- -- -- -- -- -- -- -- -- -- -- 0c -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- UU UU -- --
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
root@sla3000:~#
```

Reading values:

```
root@sla3000:~# i2cget -f -y 1 0x68 0x75
0x92
root@sla3000:~#
```

Writing values:

```
root@sla3000:~# i2cget -f -y 1 0x68 0x6c
0x00
root@sla3000:~# i2cset -f -y 1 0x68 0x6c 0x05
root@sla3000:~# i2cget -f -y 1 0x68 0x6c
0x05
root@sla3000:~#
```

## 11.2 3000-OEM I²C Connectors

**Table 7: 3000-OEM I²C Connectors**

| 3000-OEM Connector | Common Name | I²C Bus | Bus Speed |
|---|---|---|---|
| J2 | Vout | I2c-1 | 400 kHz |
| J3 | Vin0 (Camera 0) | I2c-1 | |
| J4 | Vin1 (Camera 2) | I2c-4 | 400 kHz |

### 11.3 I²C Slave Addressing

In device specifications I²C addresses are sometimes specified as 7-bit or 8-bit. In the examples shown in the I²C section, the I²C address is 0x68, which is 7-bits. The 8-bit address would be *0x68<<1 == 0xD0*. This is because the bottom bit in the slave address is a read/write indicator.

If you are having trouble reading or writing to the I²C device, try >>1 (divide by 2 in hex) in the slave address.

All references to slave address in this document (including sample code) refer to the 7-bit slave address.


**Figure 6: Slave Address**

- ✓ Verify the I²C device is powered up.
- ✓ Check that the SDA and SCL are connected correctly between the SightLine hardware and the target device.
- ✓ Check the bus speed of the target device. 3000-OEM I²C bus runs at 400 kHz.
- ✓ SightLine hardware I²C bus runs at 3.3V levels.
- ✓ There are multiple I²C buses exposed by the SightLine Hardware. Verify the I²C device is connected to the correct bus.

### 11.4 Example Linux Code

This section illustrates a simple example of using a Linux driver to read and write bytes from an I²C device.

Example of calling the code:

```
u8 buf[2] = {0x28, 0x04};
int ret = SLI2cSet(buf, 2, 2, 0x5D); // returns -1 on
failure, 0 on success
```

Example code:

```
#include <fcntl.h>
#include <unistd.h>
#include <sys/ioctl.h>
#include <stdio.h>
#include <linux/i2c-dev.h>
#include <linux/i2c.h>
```

```
int I2cGet(unsigned char *buf, long nbytes, unsigned
char bus, unsigned char addr)
{
 char fname[20];
  sprintf(fname, "/dev/i2c-%d", bus);
  int file;
  if((file = open(fname, O_RDWR)) < 0) {
    printf("failed to open %s\n", fname);
    return -1 ;
  }
```

```
  if(ioctl(file, I2C_SLAVE, addr) < 0) {
    printf("Failed to access device 0x%x\n", addr);
    close(file);
    return -1 ;
  }
```

```
  if(read(file, buf, nbytes) != nbytes) {
    printf("Failed to read from %s (%d)\n", fname, bus);
    close(file);
    return -1;
  }
  close(file);
  return 0;
}
```

```
int I2cSet(unsigned char *buf, long nbytes, unsigned
char bus, unsigned char addr)
{
  char fname[20];
  sprintf(fname, "/dev/i2c-%d", bus);
  int file;
  if((file = open(fname, O_RDWR)) < 0) {
    printf("failed to open %s\n", fname);
    return -1;
  }
```

```
  if(ioctl(file, I2C_SLAVE_FORCE, addr) < 0) {
    printf("Failed to access device 0x%x\n", addr);
    close(file);
    return -1;
  }
```

```
  if(write(file, buf, nbytes) != nbytes) {
    printf("Failed to write to %s (%d)\n", fname, bus);
    close(file);
    return -1;
  }
  close(file);
  return 0;
}
```

## 12 Questions and Additional Support

For questions and additional support, please contact Technical Support. Additional support documentation and Engineering Application Notes (EANs) can be found on the Support pages of the SightLine Applications website.