AS1163 – OTP burning

Application Note

Published by ams-OSRAM AG
Tobelbader Strasse 30,
8141 Premstaetten Austria
Phone +43 3136 500-0
ams-osram.com

AS1163 – OTP burning

Application Note No. AN001081

Valid for:
AS1163

Abstract

This application note outlines the procedures for programming the One-Time Programmable (OTP) customer area of the SAID (AS1163) device, which is used for storing device-specific configurations and calibration data that persist through power cycles. The document provides an overview of the OTP structure, detailing the designated bytes for customer settings, including features such as I²C bridge functionality, SPI mode communication, and synchronization settings.

The programming procedure is meticulously described, highlighting the necessary conditions, including voltage requirements and the need for a bypass capacitor. It details the step-by-step commands required to unlock and program the OTP area, including data preparation, sending relevant commands, and verification methods.

Key considerations regarding optimal burning conditions and error-checking mechanisms are discussed to ensure reliability and functionality. Additionally, once the ***Cust_lock*** bit is set, it prevents any further modifications to the OTP. This note serves as a comprehensive guide for engineers and developers to successfully configure and optimize SAID devices for various applications, enhancing their usability in dynamic lighting and other advanced applications.

Table of contents

1 Introduction

The OTP or one-time programmable memory stores configuration data persistently. The OTP is split into two areas: One for foundry configuration data and one for the customer configuration data. The foundry area stores all kinds of trim values; these differ from SAID to SAID; are determined during manufacturing, burned in the OTP, and finally the foundry area is locked in further modifications. The customer area is set up the same way. It contains chip specific configuration data, but on customer level. The customer area is not locked but does have its own lock. It stores 19 bytes, but 13 bits are mapped to hardware features the customer can configure. The remaining bits are free for any purpose, like color calibration parameters for attached LEDs.

At startup of the SAID, the OTP is copied to a RAM mirror, and the SAID reads the RAM, not the OTP. To change the OTP first the RAM needs to be updated (with the SETOTP telegram), then the RAM needs to be burned to the actual OTP memory (with the BURN telegram). It should be noted that it is possible to change the RAM mirror without a burn, and those settings are active until a power cycle (a RESET telegram is not enough).

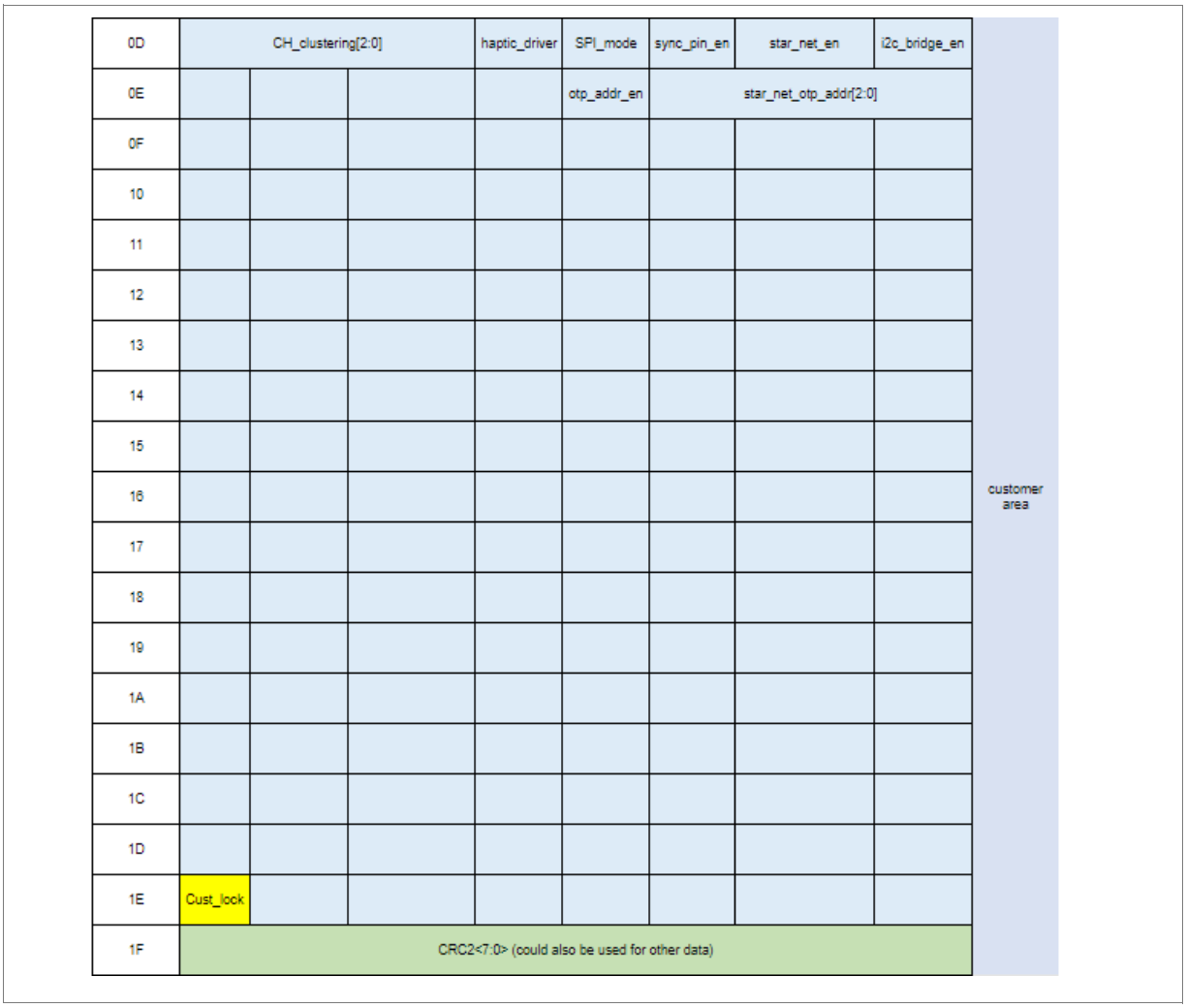
The One-Time programmable memory can be written (burned) several times. However, burning can only change a 0-bit to a 1-bit. It is not possible to burn a 1-bit to a 0.

2 OTP overview

The OTP contains trimming data burned during manufacturing as well as configuration bits. Bytes from 0x0D to 0x1F are designated for customer settings and data (bytes 0x00 to 0x0C are the foundry area instead). These bytes can enable features such as the I²C bridge, various star configuration aspects, synchronization via pin, and SPI mode communication in MCU mode.

The customer area of the OTP is structured as follows (see Figure 1 for the layout):

Figure 1: OTP structure

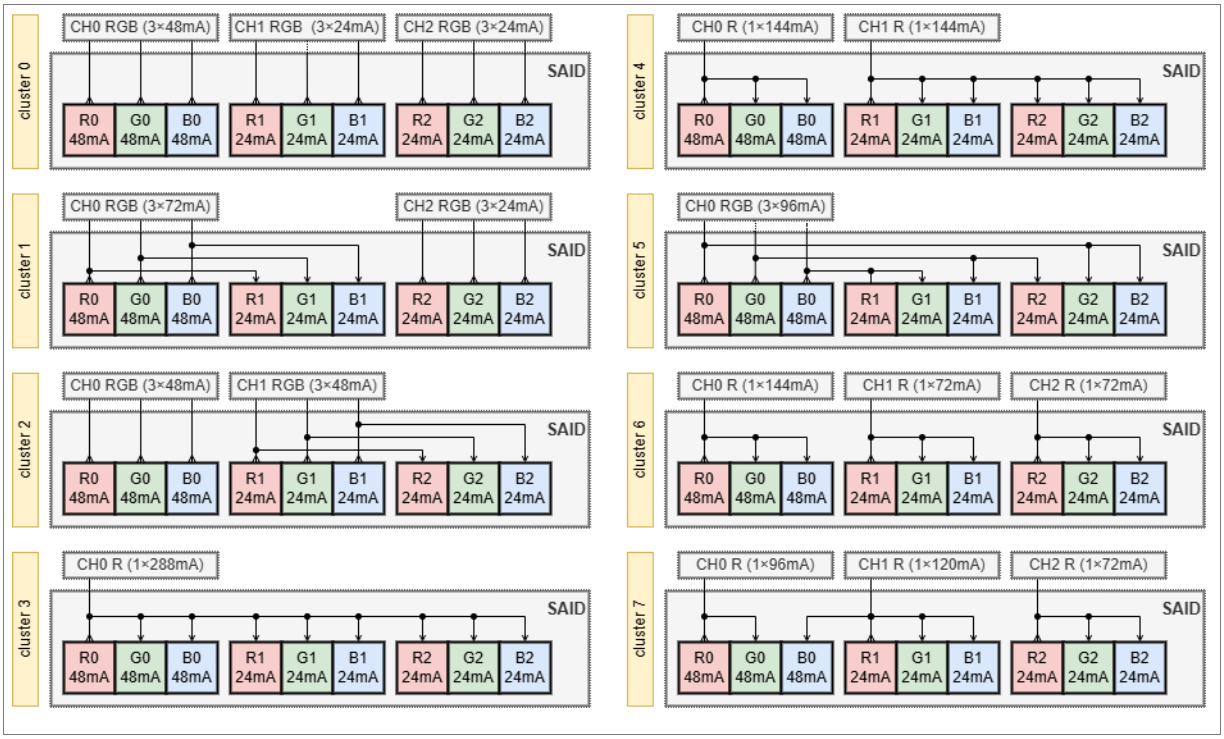


2.1 CH_clustering[2:0]

This 3-bit field configures the clustering of output channels. It is possible to connect the output drivers together to connect a single LED with a higher current. When the channels are combined to obtain a higher current, the PWM outputs are synchronized: The phase shift of the channels that are combined is eliminated and the PWM value of the channel that is aggregated is also assigned to the PWM value of the main channel. Also setting a current on the main channel results in the overall setting.

With the ***CH_clustering*** bits in the OTP it is possible to select the configurations that are shown in the following Figure 2.

Figure 2: Clustering configurations



2.2 haptic_driver

This bit enables the haptic driver functionality, allowing the device to control haptic feedback mechanisms. LED drivers will then be used to drive a haptic actuator.

When this bit is set in the OTP, all the drivers of all channels are combined with the same phase, so that it is possible to connect all the output pads together to drive the haptic actuator with 288mA current (if the maximum current is set in the current register of channel 0).

Also, the PWM code that will be executed will be the same for all output drivers and will correspond to the RED value of channel 0 (this is equivalent to ***CH_clustering*** = 3). When the haptic actuator driving mode is selected, the PWM frequency is divided by 4 with respect to the

normal frequency. So, it will be 146 Hz (usually it is referred to as 150 Hz mode) or 293 Hz (usually referred to as 300 Hz mode), depending on the 1 kHz mode bit setting. When haptic_driver bit is set, the CH_clustering bits are ignored.

2.3 SPI_mode

This bit enables SPI communication mode. When set, it allows the device to communicate using a MCU mode type-B interface, which is like SPI, as shown in Figure 4. This is an alternative option to MCU mode type-A, shown in Figure 3. In the type-B case, instead of sending a Manchester encoded data over SIO1_P, the MCU will send clock and data through SIO_N and SIO_P respectively. In either case (type-A or type-B) clock and data signals are received split respectively on SIO_N and SIO_P.

In other words, when the ***SPI_mode*** is set, both downstream and upstream communication use a regular SPI interface with clock and data signals with NRZ coding (no Manchester encoding):

Figure 3: MCU mode type-A

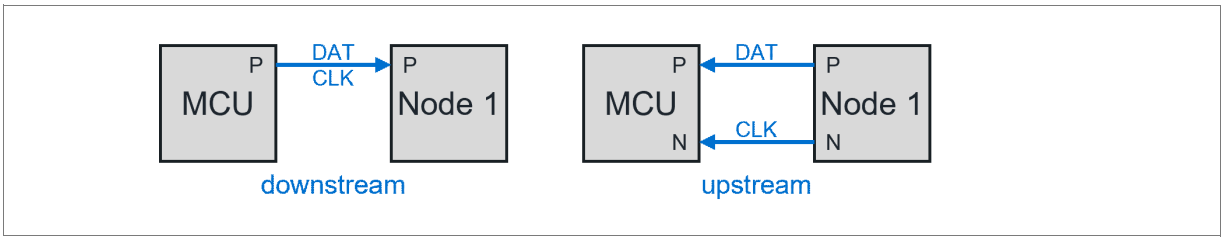
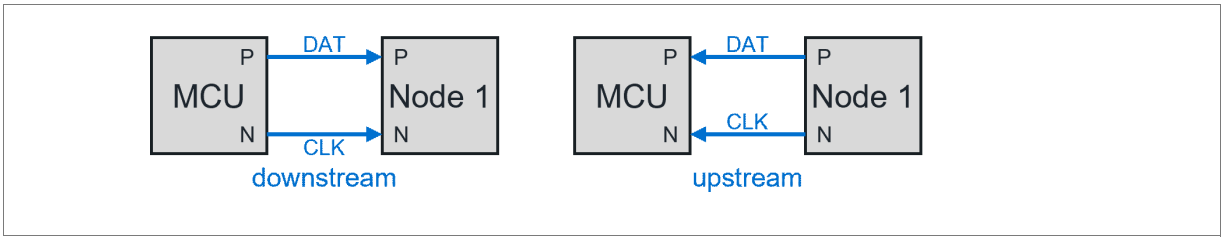


Figure 4: MCU mode type-B



2.4 Sync_pin_en

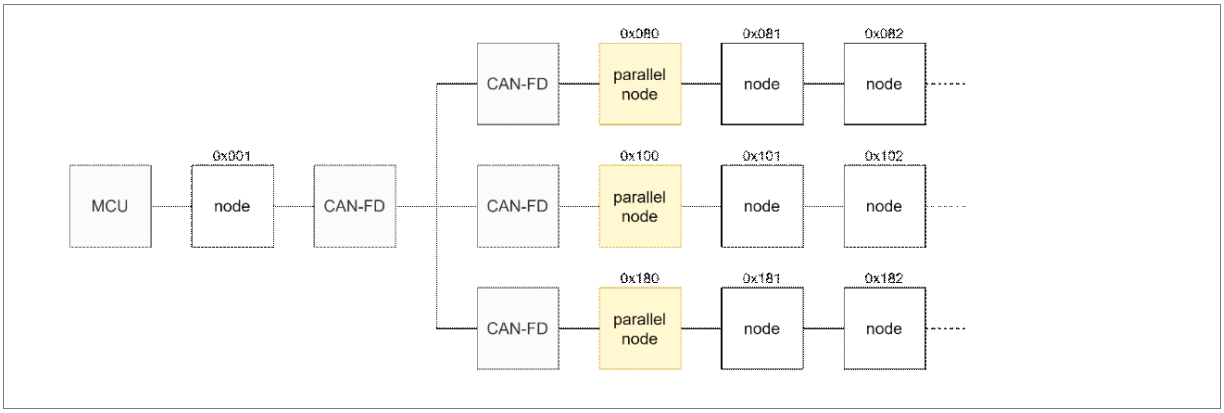
This bit enables synchronization via an external pin. This is important for applications that require synchronized lighting effects across multiple devices. When the SYNC_EN bit is set in the CURRENT register, new PWM values are not applied until a sync signal is received. The sync can be either software, by sending a SYNC telegram, or hardware, by toggling of the sync pin. In the latter case, sync pin must be high for at least 3 clock cycles and the ***sync_pin_enable*** bit must be set in the OTP.

The sync pin functionality is assigned to B1 driver so setting ***sync_pin_en*** disables B1 from driving an LED.

2.5 star_net_en

This bit enables the star network configuration, allowing multiple SAID devices to operate in a parallel configuration, like the topology shown in Figure 5. The parallel address (the 3-bit branch mask) is read during SAID's startup sequence, during the communication mode setting. There are two methods to configure the branch mask: either via the OTP or via external pins. The first method is selected by setting bit ***otp_addr_en***, the second method by clearing it. When the bit is set, use ***star_net_otp_addr*** to set the branch mask in the OTP. When the bit is clear, we use the open and short flags of G1 and R1 to generate the mask. Both drivers have 3 options (open, ok, short) so there are 9 combinations setting the 3-bit mask (see datasheet).

Figure 5: Parallel network topology



2.6 I2C_bridge_en

This bit enables the I²C bridge functionality, allowing the device to interface with I²C devices such as sensors and EEPROMs. Specifically, the third channel CH2 is configured to work as I²C bridge, R2 for SDA, G2 for SCL, and B2 for INT.

2.7 otp_addr_en

This bit enables to use the OTP to set the branch mask. Only relevant when ***star_net_en*** is set. When set, the device uses ***star_net_otp_addr*** to set the branch mask. When not set, use G1 and R1 to set the mask (see ***star_net_en***).

2.8 star_net_otp_addr[2:0]

This 3-bit field specifies the branch mask: the address for the star network configuration. Only relevant when ***star_net_en*** and ***otp_addr_en*** are set.

2.9 Customer lock

The Cust_lock bit in the OTP memory is used to prevent further modifications to the customer area. Once set, it ensures that the custom configuration is permanent and cannot be altered.

2.10 CRC2

The CRC2 field is located at the specific address 0x1F within the OTP memory. It is an 8-bit value that can be used for data integrity verification.

3 Programming the OTP customer area

To successfully program the One-Time Programmable (OTP) area of the SAID, follow these high-level steps:

- Hardware preparations
  - Wire up a supply and external components to the SAID that need burning
- Prepare the image to burn
  - Set up the data to be burned into the OTP memory.
- Actual burn process
  - Execute the commands to burn the data into the OTP.
- Verification
  - Verify the programmed values in the OTP.
- Locking the OTP
  - Finalize the programming by locking the OTP to prevent further modifications.

Burning conditions: Ensure that the burning conditions are within the optimal range to avoid yield loss. A typical yield loss is around 3 ppm/bit.

Error checking: If burning conditions are not met or if the Cust_lock is set incorrectly, the device may not function as intended.

This chapter 3 describes the process steps at a high level. Chapter 4 describes concrete steps using the EVK. Chapter 5 looks at mass manufacturing.

3.1 Hardware preparation

Connect the target node (SAID) to the programming board and an external programmable DC power supply using the appropriate connections.

Intercept the VDD and GND of the target node and use a lab power supply to ensure stable power delivery.

Add a 10 µF bypass capacitor to the VDD pin to stabilize voltage during programming.

Ensure that the Cust_lock bit is not set. This bit would prevent any further programming of the OTP area.

3.2 Prepare the image to burn

Power on the chain at 5V, and broadcast reset and initbidir commands to initialize the system.

Send the SETTESTPW command to the target node to allow OTP writing.

Read the OTP RAM mirror by sending READOTP telegrams to verify the current values.

Mask in new 1-bit as necessary to prepare the data for writing to the OTP using SETOTP telegrams.

Optionally, set the ***Cust_lock*** bit.

See section 4.1.2 for more details.

3.3 Actual burn process

Using the lab power supply, reduce the voltage of the target to 2.7V[1].

Send the CUST telegram to select the OTP section to be programmed (the customer area).

Send the BURN telegram and wait approximately 5 ms for the programming to complete.

Send the IDLE telegram to return the device to its idle state.

[1] Ensure the voltage both on the power supply and on the SIO1_x lines is 2.7V± 100mV. It might be necessary to increase the voltage depending on the MCU and the setup used.

Switch lab supply back to 5V.

Leave test mode by sending an invalid SETTESTPW (e.g. use password 0); otherwise, the node will remain in a non-operational state.

3.4 Verification of programmed values

Write different data into the “OTP mirror” (using SETOTP, use SETTESTPW first) to prepare for verification.

Send the CUST telegram.

Optionally, send the GLOAD telegram to enable additional verification checks[2].

Send the LOAD telegram to initiate loading of the OTP data in OTP mirror.

Send the IDLE telegram and invalidate the password afterward.

Read back all OTP data using the corresponding read commands (READOTP) and compare it with the written values to ensure correct programming.

See section 4.1.2 for more details.

3.5 Locking the OTP

Once the programming is verified, repeat the entire procedure to set the Cust_lock bit (0x1F.7) to 1 to prevent any further modifications to the OTP memory.

[2] When a bit is fused to 1, the fuse resistance changes from low to high value. A comparator is used to compare the voltage across the fuse resistor being loaded. When GLOAD is used, a higher comparator threshold is selected. The difference with the normal threshold serves to detect marginally fused bits, hence guard-banding load.

4 OSP32 for OTP programming

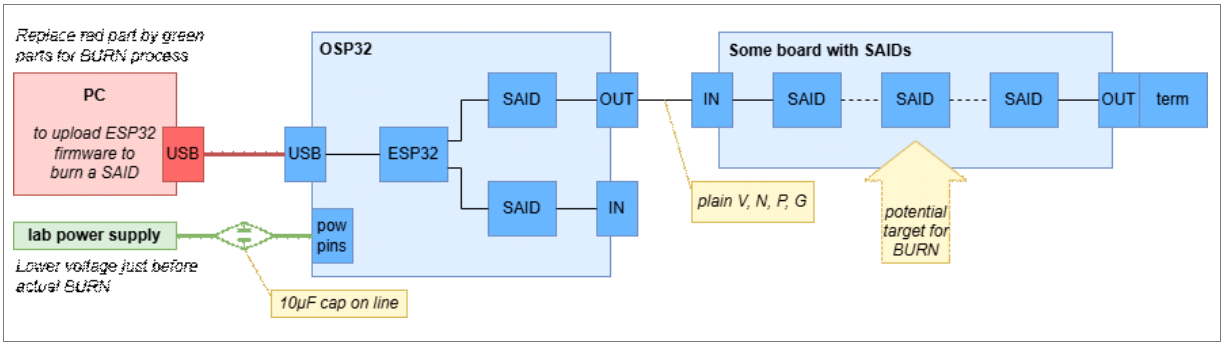
This chapter describes how to successfully program through the Software available the AS1163 (SAID) device using the Hardware Arduino OSP ecosystem Evaluation Kit (EVK). It is explained how to set 1-bit in the address 1D of the OTP of the first SAID of a SAIDlooker board.

Required components:

- OSP32 from the EVK
- AS1163 (SAID) device connected to the OSP32.OUT (e.g. SAID on the SAIDlooker demo board)
- USB cable to initially flash the ESP32 with some firmware that burns the OTP, like aoosp_otpburn
- External power supply to force the voltage during the burning

This process requires two steps. The first step is to flash the firmware inside the OSP32 MCU on the OSP32 board. This firmware shall implement the burn procedure described in the previous chapters. Once the SW is flashed, remove the USB cable and power the board through the external power supply. The second step, running from the lab supply, is to execute the ESP32 firmware so that an OTP is actually burned.

Figure 6: Hardware setup for OTP burning



With the previous setup, shown in Figure 6 it is possible to burn the OTP in any node the EVK. It is sufficient to drop the voltage on the OSP32 supply pins.

It is possible to burn any device on the chain in this way.

4.1 Software stack for OTP programming

The software stack for programming the OTP on the AS1163 involves several components. First you need to set up the environment to program the OSP32 board through the Arduino IDE.

4.1.1 Preliminary setup

Install the Arduino IDE, the ESP32 board package and OSP libraries as described in ams-OSRAM/OSP_aotop/blob/main/gettingstarted.md#installation

Once the libraries are installed, it is possible to access the examples. These are specific code examples which can be used to start the development. There are several ones, each with a specific function and feature to exploit.

OTP burning has also its specific example.

4.1.2 The OTP burn example

Open File>Examples>OSP telegrams aoosp>aoosp_otpburn.

For burning, only aoresult, aospi and aoosp are needed, but the aoosp_otpburn demo uses the OLED to show progress and requires the A button to confirm the step. Also, aoui32 library is needed for this example.

Figure 7: Arduino IDE flow

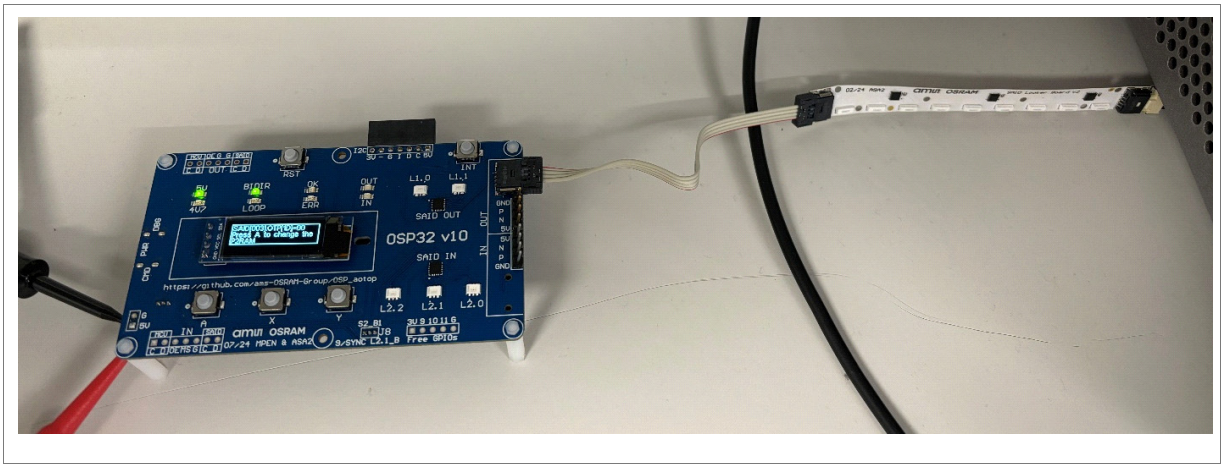
Once opened, as shown in Figure 7 there is always a commented section which describes the hardware needed, the behavior of the program and its steps. This example has the following steps described:

- Make sure you have the password (see aoosp.cpp line ~30)
- Update the constants ADDR, OTPADDR, OTPDATA.
  - These configure which SAID gets its OTP updated (ADDR), which address in the OTP will be updated (OTPADDR), and which data will be written (OR'ed) there (OTPDATA).
- Compile and upload this script as usual (it will not do OTP flash on its own).
- Now unplug all USB cables, and supply them with a lab supply 5V via the OSP32 J3C pinheader.
- Reset the ESP32, the OLED should say

"OTP[XX]=YY Press A to change P2RAM"

Where XX is OTPADDR and YY is the old value (not yet OTPDATA) (shown in the next Figure 8).

Figure 8: First OLED message



- Without interrupting power, reduce the voltage of the power supply from 5V to 4V (in Figure 9 the used setup is shown). Then press the A button on the OSP32 board.
- The OLED should say

"Power cycle (not reset) and check"

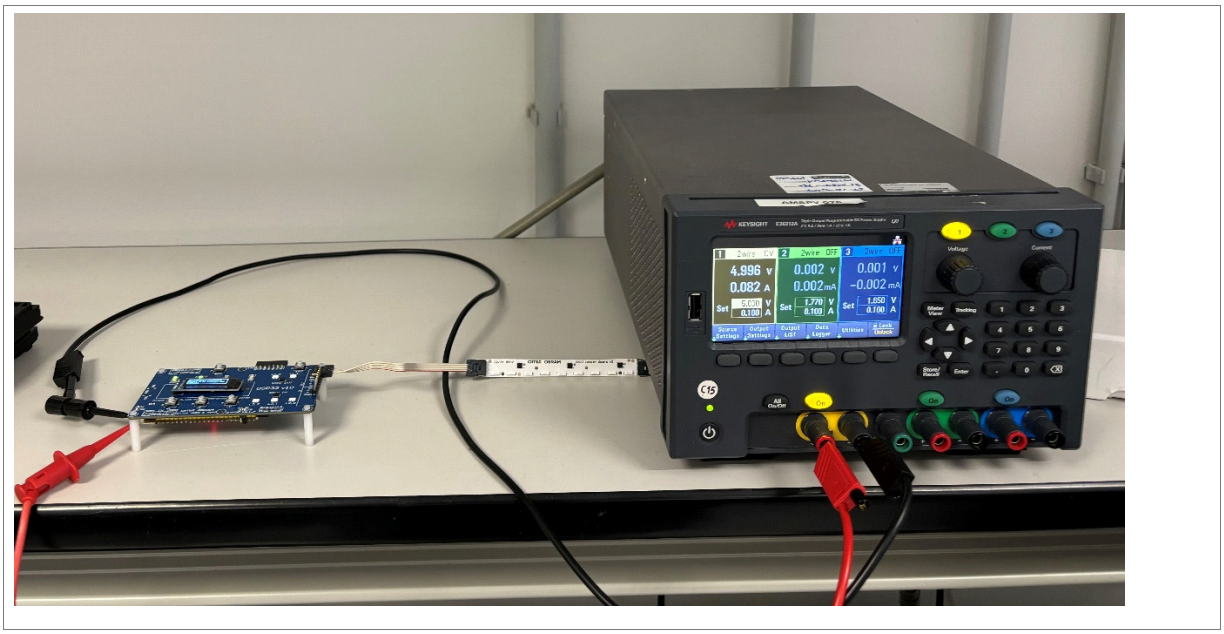
  - Disable the power supply, turn it up to 5V and enable it again.
  - Alternatively unplug the power supply and re-insert a USB cable.

- The OLED should say

“OTP[XX]=YY Press A to change P2RAM”

Where XX is OTPADDR and YY is the new value OTPDATA

Figure 9: OSP32 burning setup



5 Mass manufacturing

The OSP32 example will lead to higher failure rates (too high ppm due to wrong voltage).

The following sequence is proposed for mass production:

- reset/init
- settestpw
- setotp
- cust
- burn
- (within 15µs) lower voltage, wait (5ms), raise voltage
- idle
- Optionally verify: setotp(0), cust, gload, load, idle, readopt, compare
- settestpw(0)

This does require quick synchronization between sending the burn and lowering the voltage to 2.9 V ± 100 mV.

6 Summary / results

Programming the OTP customer area of the SAID (AS1163) allows for essential configurations and calibration that enhance the device's functionality. Following the outlined procedure ensures a successful programming operation, leading to optimal performance in applications utilizing the SAID device.

For questions or technical support, please contact ams OSRAM directly.

7 Revision information

Changes to current revision v1-00	Page
Initial production version	

- Page and figure numbers for the previous version may differ from page and figure numbers in the current revision.
- Correction of typographical errors is not explicitly mentioned.

ABOUT ams OSRAM Group (SIX: AMS)

The ams OSRAM Group (SIX: AMS) is a global leader in intelligent sensors and emitters. By adding intelligence to light and passion to innovation, we enrich people's lives. With over 110 years of combined history, our core is defined by imagination, deep engineering expertise and the ability to provide global industrial capacity in sensor and light technologies. Our around 20,000 employees worldwide focus on innovation across sensing, illumination and visualization to make journeys safer, medical diagnosis more accurate and daily moments in communication a richer experience. Headquartered in Premstaetten/Graz (Austria) with a co-headquarters in Munich (Germany), the group achieved EUR 3.6 billion revenues in 2023. Find out more about us on https://ams-osram.com

For further information on our products please see the Product Selector and scan this QR Code.

Published by ams-OSRAM AG
Tobelbader Strasse 30, 8141 Premstaetten, Austria
ams-osram.com © All Rights Reserved.

Published by ams-OSRAM AG
Tobelbader Strasse 30,
8141 Premstaetten Austria
Phone +43 3136 500-0
ams-osram.com