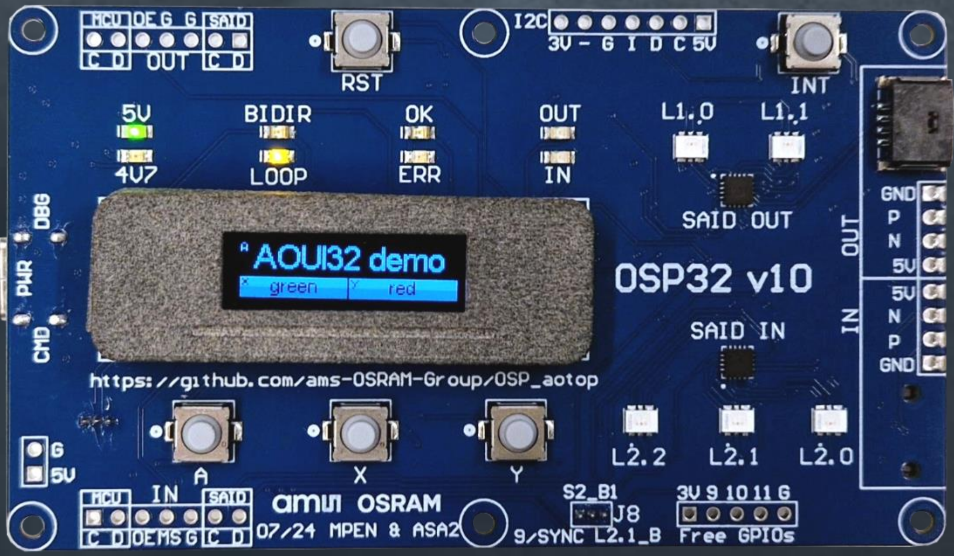
OSP on Arduino – Training – Appendix 3

Using the *Arduino OSP evaluation kit* – OTP burning of SAID



2025 February 28

Part A3 – OTP burning of SAID

OTP introduction

OTP in SAID

- The SAID has an OTP to store (configuration) data specific for itself
- OTP abbreviates One-Time Programmable memory
  - “One-time”: Think of it as a bunch of fuses
  - “Memory”: Intact a fuse represents a 0, burned it represents a 1

- The SAID OTP is split in two areas: foundry and customer
- foundry area
  - stores 13 bytes configuration data
  - set by ams-OSRAM during manufacturing
  - Examples are trim values for the internal oscillator, voltage, …
  - this area is locked for writing
- customer area
  - stores 19 bytes configuration data
  - set by the end-customer
  - 2 bytes (13 bits) are SAID hardware features the customer can configure
  - remaining 17 bytes are free, e.g. for color calibration values
  - this area has a lock, by default 0/unlocked, but the customer can write a 1/lock

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
00								
01								
02								
03								
04								
05								
06								
07								
08								
09								
0A								
0B								
0C								
0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		
0F								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
1A								
1B								
1C								
1D								
1E	cust_lock							
1F	crc or other usage							

Foundry area

Customer area

A3 OTP burn

ch_clustering

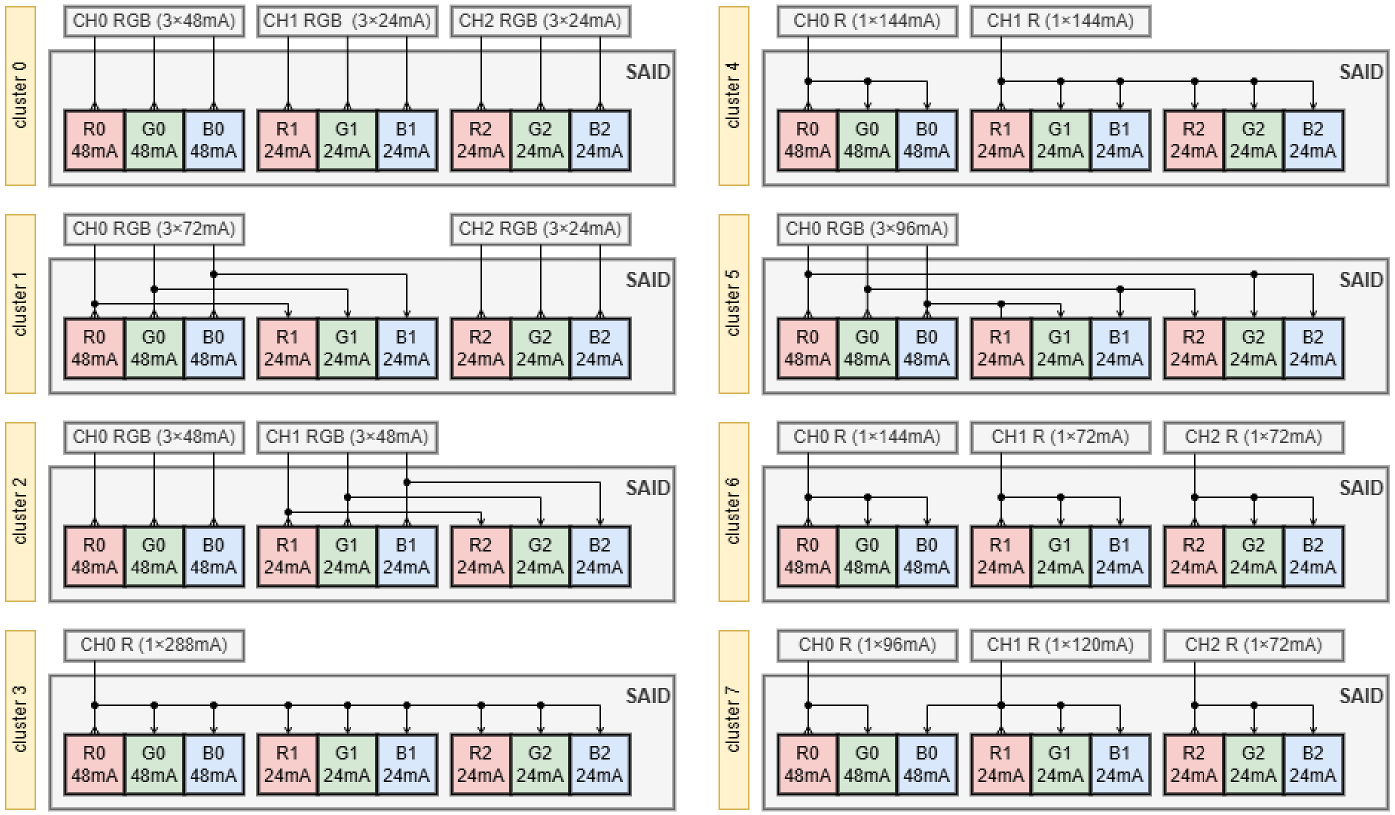
For higher drive currents

0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		
1E	cust_lock							
1F	crc or other usage							

- The SAID has 9 drivers (in 3 channels)
- Each capable of driving 24 mA or even 48 mA

- It is possible to *cluster* drivers
  - One driver becomes the main (triangle arrow tail)
  - the others the aggregates (arrow-head)
  - 8 clustering option available
- The PWM of main is enforced on its aggregates
- The current of main is enforced on its aggregates

- On the PCB:
  the main and aggregates need to be wired together
- The OTP:
  needs to be burned for the associated cluster mode in bits **ch_clustering**



A3 OTP burn

0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		
1E	cust_lock							
1F	crc or other usage							

haptic_driver

Using LED drivers for a haptic actuator

– When the **haptic_driver** bit is set in the OTP, the SAID acts as a driver for a haptic actuator

– This implicitly selects clustering mode 3 (all driver aggregated to one output)

– Gives a power of $\underbrace{48+48+48}_{\text{ch0}} + \underbrace{24+24+24}_{\text{ch1}} + \underbrace{24+24+24}_{\text{ch2}} = 288\text{mA}$

– Also, the driver frequency is implicitly divided by 4, resulting in 150 Hz mode or 300 Hz mode (see FAST_PWM)

PWM speed	FAST_PWM	PWM label	PWM actual	Haptic actual	Haptic label
Slow	0	500 Hz	586 Hz	146 Hz	150 Hz
Fast	1	1 000 Hz	1 172 Hz	293 Hz	300 Hz

7.13.8 READSETUP (0x4C)

Read SETUP register. No payload. Answer is 8-bit.

Table 28: READSETUP register

Addr: 0x4C **READSETUP**

Bit	Bit name	Default	Access	Bit description
7	FAST_PWM	0x0	RW	This bit allows to control the PWM speed and by this the dynamic range of the color: "0": 500Hz PWM frame rate with 15 bits dynamic range "1": 1000Hz PWM frame rate with 14 bits dynamic range

A3 OTP burn

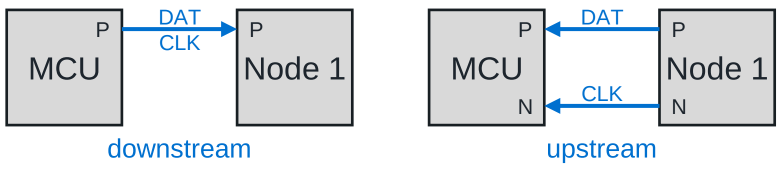
spi_mode

Manchester or 2-wire SPI

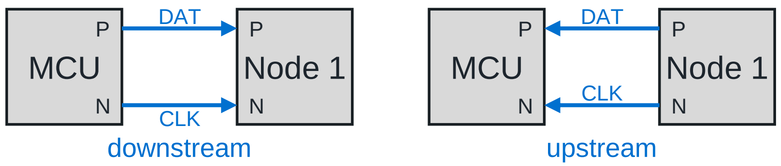
0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		
1E	cust_lock							
1F	crc or other usage							

- The first SAID in the chain is typically configured for MCU mode (with a pull-up on P and a pull-down on N)
- By default, the MCU mode is of type A
  - Downstream telegrams (“commands”) use Manchester encoding on P line
  - Upstream telegrams (“responses”) use P line for DATA and N line for CLK
- By setting OTP bit **spi_mode**, the mode becomes type B aka 2-wire SPI
  - Downstream telegrams (“commands”) and upstream telegrams (“responses”) both use P line for DATA and N line for CLK
- “RAM-hackers”: SAID would need to have this bit set before receiving **reset** and **init** therefore, setting this bit in P2RAM is ineffective

MCU mode type A



MCU mode type B



A3 OTP burn

sync_pin_en

Sync via telegram or pin

0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		

1E	cust_lock							
1F	crc or other usage							

- Sending **setpwm** telegrams to multiple nodes causes variation in their color/brightness-switch moment

- The SAID has a sync feature
- 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 signal can be either software, given form the **sync** telegram, or hardware, from a toggling of the sync pin

- The OTP bit **sync_pin_en** configures the method (default 0 is telegram, 1 is sync pin)

- Pin B1 is used as the sync pin
- When **sync_pin_en** is set, B1 can no longer be used as LED driver

Table 34: READCURCHN register

Addr: 0x50		READCURCHN		
Bit	**Bit name**	**Default**	**Access**	**Bit description**
15	reserved	0x0	RW	-
14	SYNC_EN	0x0	RW	Enable synchronization of PWM register for the chain: the values written in PWM register are transferred to the PWM engine only after that a SYNC command is received or the SYNC pin is triggered.
13	bvb	0x0	RW	Must be set to 0

Table 2: Pin description of AS1163

Pin number	Pin name	Pin type[1]	Description[2]
QFN 3x3 16			
12	R2	ANA	Driver 0 of channel 2 (Red), also SDA
13	B1	ANA	Driver 2 of channel 1 (Blue), also SYNC
14	G1	ANA	Driver 1 of channel 1 (Green), also for parallel address configuration

A3 OTP burn

0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		

1E	cust_lock							
1F	crc or other usage							

i2c_bridge_en

Configure channel 2 for I2C

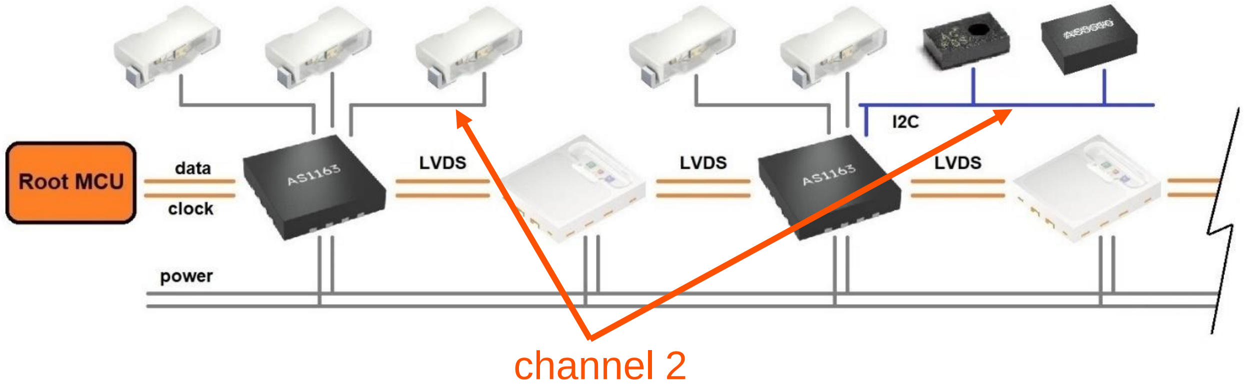
– The SAID has a built-in I2C bridge

– This is an alternative function of the drivers in channel 2
– Enabled by setting **i2c_bridge_en** to 1

– When **i2c_bridge_en** is set,
R2, G2, and B2 can no longer be used as LED driver

Table 2: Pin description of AS1163

Pin number	Pin name	Pin type[(1)]	Description[(2)]
QFN 3x3 16			
10	B2	ANA	Driver 2 of channel 2 (Blue), also INT
11	G2	ANA	Driver 1 of channel 2 (Green), also SCL
12	R2	ANA	Driver 0 of channel 2 (Red), also SDA
13	B1	ANA	Driver 2 of channel 1 (Blue), also SYNC



A3 OTP burn

Star network

Enabling and setting a branch mask

0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		
1E	cust_lock							
1F	crc or other usage							

– The SAID supports star networks

– This is implemented as an address filter
– The filtering must be enabled and the filter mask configured
– The SAID is said to be a “branch starter”

– To enable filtering set **star_net_en** in OTP to 1

– To set the filter in OTP, set **otp_addr_en** to 1
and **star_net_otp_addr** to the wanted filter mask
– To set the filter via pins, set **otp_addr_en** to 0
and use R1 and G1 (**star_net_otp_addr** remains unused)

– Pins R1 and G1 define the mask through a resistor configuration
– With **star_net_en**=1 and **otp_addr_en**=0,
R1 and G1 can no longer be used as LED driver

– “RAM-hackers”: SAID would need to have this bit set before receiving **reset** and **init** therefore, setting this bit in P2RAM is ineffective

Table 2: Pin description of AS1163

Pin number	Pin name	Pin type[1]	Description[2]
QFN 3x3 16			
14	G1	ANA	Driver 1 of channel 1 (Green), also for parallel address configuration
15	R1	ANA	Driver 0 of channel 1 (Red), also for parallel address configuration
16	B0	ANA	Driver 2 of channel 0 (Blue)

Table 10: Address mask configuration

G1	R1	Internal bits	Corresponding parallel address
VDD/2	VDD/2	00 00	0b000
VDD/2	GND	00 01	0b001
VDD/2	VDD	00 10	0b010
GND	VDD/2	01 00	0b011
GND	GND	01 01	0b100
GND	VDD	01 10	0b101
VDD	VDD/2	10 00	0b110
VDD	GND	10 01	0b111

A3 OTP burn

cust_lock

Blocking OTP modifications

0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		
1E	cust_lock							
1F	crc or other usage							

- Once **cust_lock** has been set to 1, no more OTP modifications (customer area) are possible
- Option: use 1F as a CRC checksum area
- One could use the same CRC as for the telegrams; see aoosp_crc module

```
/*!
    @brief  Computes the OSP CRC for a sequence of bytes.
    @param  buf
            A pointer to a sequence of bytes.
    @param  size
            Number of bytes of buf to compute the CRC for.
    @return The OSP CRC.
    @note   Implemented via table lookup.
*/
uint8_t aoosp_crc(const uint8_t * buf, int size);
```

Part A3 – OTP burning of SAID

The customer area

The burning process

Burning in practice

OTP mirror

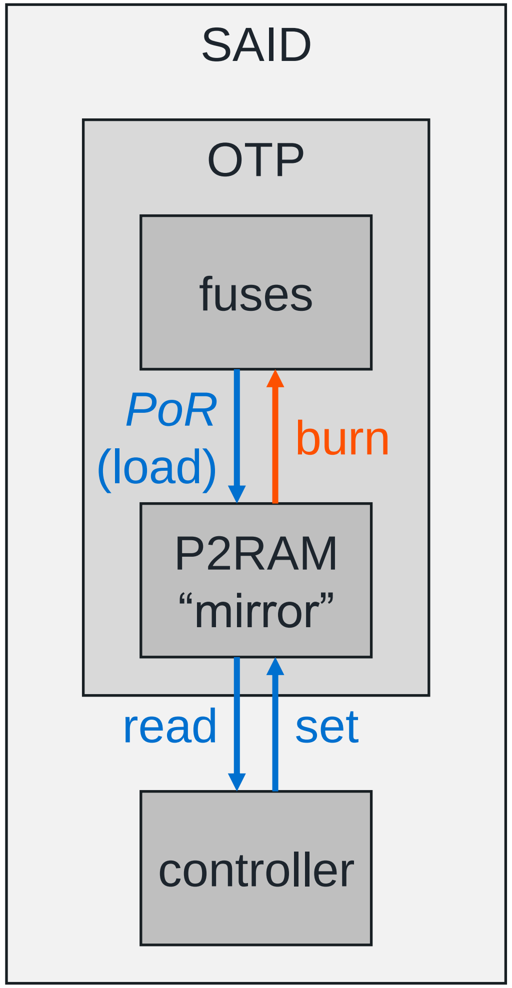
Also known as P2RAM

The OTP contents is mirrored in a RAM

- Known as the OTP mirror or as P2RAM
- Sometimes OTP means just fuses (narrow sense: “the persistent tech”)
- Sometimes OTP means fuses+P2RAM (broad sense: “configuration store”) [these slides]

Behavior

- At power-on reset (“PoR”) the configuration bits are “copied” from the fuses to the mirror
- The controller in the SAID always reads config bits from the mirror, never from the fuses
- Telegrams can be sent to the controller to modify the mirror (**readopt**, **setotp**)
  The controller can then use the “new” bits
- These bits are in P2RAM, so not persistent, they are lost after a power cycle
  (also lost after a **load** telegram; a **reset** telegram is not enough)
- The **burn** telegram burns the P2RAM into the fuses
  (making the configuration persistent)



A3
OTP burn

ams OSRAM

Intermezzo on authenticating for OTP changes

Setting the SAID password

- Telegrams related to changing the OTP require elevated privileges
- Those telegrams include (for details, see next slides)
  - setotp
  - cust, burn, idle
  - gload, load

- The MCU must first authenticate by sending the correct password (using the **settestpw** telegram)
- After sending this password, the SAID is in “test mode”
- In test mode, the SAID does not fully function (e.g. does not forward telegrams)
- Leave test mode by sending the incorrect password (e.g. 0)

- The aolibs have a password store: function **aoosp_said_testpw_get()** returns the password
- This function is used by e.g. macro **aoosp_exec_setotp()** to set password, read/modify/write otp and clr password

- By default, the password store is empty, see **aoosp_said_testpw_get()** for options on how to fill it
- Ask ams-OSRAM representative for the actual SAID password

OTP burn process

First phase: preparing the OTP image

1 IMAGE

A3 OTP burn

- First step is to prepare a configuration image in the P2RAM
- After a PoR all fuses are loaded into the P2RAM

- Make the SAID addressable (send a **reset** and an **init** telegram)
- Read the “old” (current) contents of the OTP (mirror) via telegrams **readotp**.

- Authenticate for OTP writes by sending the correct password (via telegram **settestpw**)
- Create “new” OTP image in P2RAM by sending **setotp** telegrams
- De-authenticate to make the SAID behave normal again (via telegram **settestpw(0)**)

The “macro” function aoosp_exec_setotp() takes care of set pw, set OTP, clr pw

- The P2RAM mirror should now contain the desired image to be burned in OTP fuses
- Recall: every 1 in the fuses must be a 1 in the mirror – a 1 can never be changed back to a 0
- Unless the lock is set, a next burn will be possible setting even more bits to 1 (e.g. lock bit)

- Next step is the actual burning

OTP burn process

Second phase: burning

2 BURN

A3 OTP burn

- Second step is to burn the P2RAM image into the fuses
- Assumption is that we continue from previous step, so SAID is addressable and not authenticated

- Authenticate for burn by sending the correct password (via telegram **settestpw**)
- Make sure the customer area is activated (send telegram **cust**)
- Lower the supply voltage (see upcoming slide)
- Start the burning (copy P2RAM to fuses) by send telegram **burn**
- Wait 5ms for burning to be complete
- Deactivate the customer area (send telegram **idle**)
- Raise the supply voltage to normal
- De-authenticate to make the SAID behave normal again (via telegram **settestpw(0)**)

- Next step is the verification

OTP burn process

Third phase: verification

3 VERIFY

A3 OTP burn

- Third step is to verify the fuses are burned correctly
- Assumption is that we continue from previous step, so SAID is addressable and not authenticated

- Suggested is to first clear the P2RAM (e.g. writing 0xAA everywhere) with **setotp**

- Authenticate for load by sending the correct password (via telegram **settestpw**)
- Make sure the customer area is activated (send telegram **cust**)
- Suggested: configure different guard-band for more thorough check (send telegram **gload**)
- Load the fuses into the P2RAM mirror (send telegram **load**)
- Deactivate the customer area (send telegram **idle**)
- De-authenticate to make the SAID behave normal again (via telegram **settestpw(0)**)
- Check the P2RAM (and thus the fuses) have correct values (use **readotp**)

Burning needs a low supply voltage

Using a lab power supply

The actual burn process needs to occur at 2V7

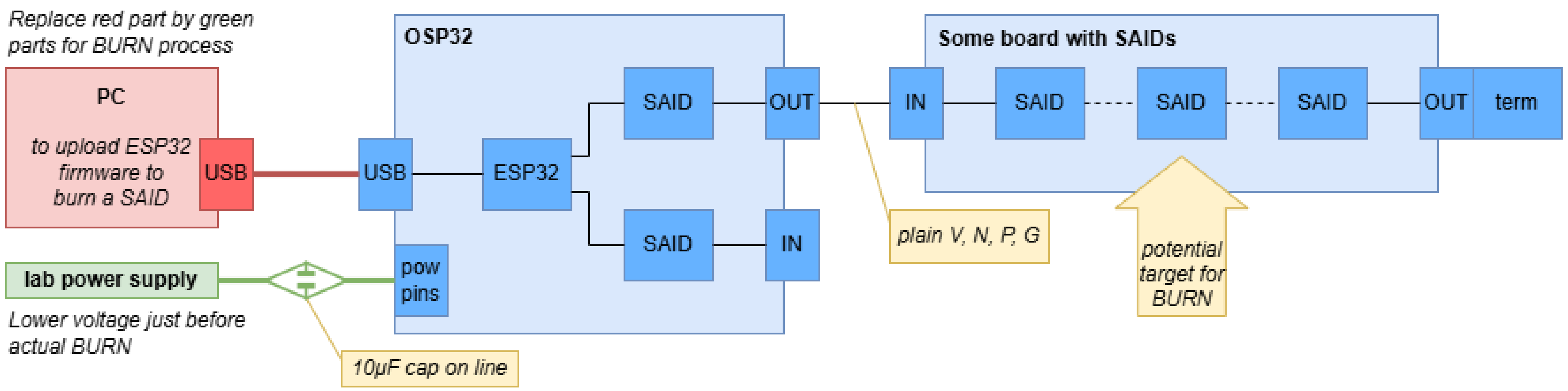
We can relatively easily try it on the OSP32 board – stretching the SAID specs

Flash the firmware sketch (that burns the OTP)

- Over USB flash the sketch that burns the firmware
- Add a prompt for the operator to lower the voltage, and sketch waits (e.g. for press on A button) to start burning

Execute the firmware sketch to burn the OTP

- Replace USB cable with lab power supply; start at 5V, lower (without interruption) to 4V when prompted
- 4V is out-of-spec for SAID burning, but is needed for the 3V3 LDO on the OSP32 board (with headroom)



Part A3 – OTP burning of SAID

The customer area

The burning process

Burning in practice

Complete example

aoosp_otpburn

The library aoosp comes with a complete example

- It is called **aoosp_otpburn**
- By default, it writes a 0x01 to OTP location 0x1D for the SAID with address 0x003
- Gives prompt on OLED to lower and raise voltage, and waits for A-button

Notes

- Need to get the SAID password and get that in the aolib’s password store
- Need to flash firmware over USB, then replace USB by lab supply as shown before
- Can only try this sketch once, other runs need different OTP address or value
- Sketch operates the SAID out of spec
- This leads to higher failures rates then wanted in mass production

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
00								
01								
02								
03								
04								
05								
06								
07								
08								
09								
0A								
0B								
0C								
0D	ch_clustering			haptic_driver	spi_mode	sync_pin_en	star_net_en	i2c_bridge_en
0E					otp_addr_en	star_net_otp_addr		
0F								
10								
11								
12								
13								
14								
15								
16								
17								
18								
19								
1A								
1B								
1C								
1D	1							
1E	cust_lock							
1F	crc or other usage							

Mass manufacturing

Lowering ppm failures

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 ams OSRAM