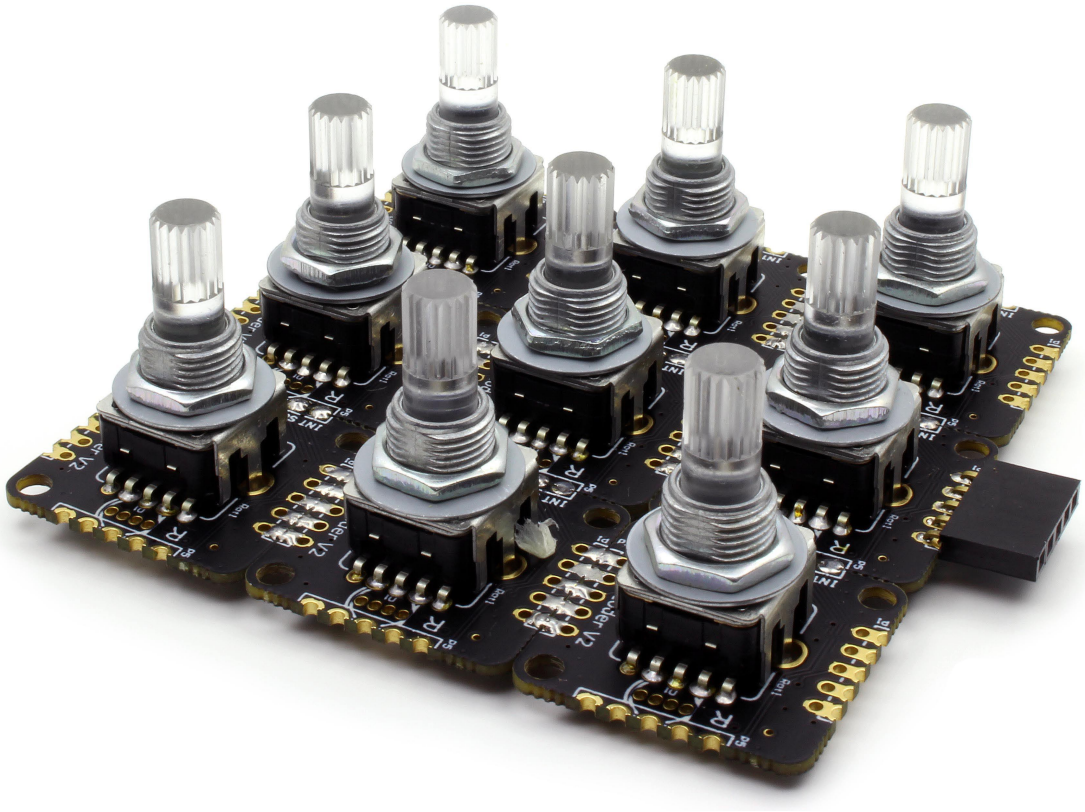


I2C Encoder V2

HW V2.0

Revision History

Revision	Date	Author(s)	Description
1.0	24.06.18	Simone	First draft version
1.1	16.08.18	Simone	Added fade and timing registers
1.2	20.08.18	Simone	Added more details
1.3	15.09.18	Simone	Inverted the register FSTATUS
1.4	04.11.18	Simone	Added the section Issues. Added Bug #1
1.5	19.12.18	Simone	Added some I^2C address setting examples and added Bug #2

Contents

1 Device Overview **3**
- 1.1 Electrical characteristics . . . 4
- 1.2 Connection . . . 4
- 1.3 I^2C interface . . . 5
- 1.4 Rotary encoder . . . 7
- 1.5 GP pins . . . 8
- 1.6 Fade function . . . 9
- 1.7 EEPROM . . . 10
- 1.8 Interrupt . . . 10

2 Registers **11**
- 2.1 Configuration . . . 12
  - 2.1.1 General Configuration . . . 12
  - 2.1.2 GP1 Configuration . . . 13
  - 2.1.3 GP2 Configuration . . . 13
  - 2.1.4 GP3 Configuration . . . 14
- 2.2 Interrupt output Configuration . . . 15
- 2.3 Status . . . 16
  - 2.3.1 Encoder Status . . . 16
  - 2.3.2 Secondary interrupt status . . . 17
  - 2.3.3 Fade process status . . . 18
- 2.4 Encoder registers . . . 19
  - 2.4.1 Counter Value . . . 19
  - 2.4.2 Counter Max . . . 19
  - 2.4.3 Counter Min . . . 20
  - 2.4.4 Increment step . . . 20
- 2.5 LEDs registers . . . 21
  - 2.5.1 LED Red intensity . . . 21
  - 2.5.2 LED Green intensity . . . 21
  - 2.5.3 LED Blue intensity . . . 21
- 2.6 GPs registers . . . 22
  - 2.6.1 GP1 register . . . 22
  - 2.6.2 GP2 register . . . 22
  - 2.6.3 GP3 register . . . 22
- 2.7 Timing registers . . . 23
  - 2.7.1 Anti-bouncing period . . . 23
  - 2.7.2 Double push time . . . 23
  - 2.7.3 Fade RGB encoder LED . . . 23
  - 2.7.4 Fade GP ports . . . 23

3 Reference **24**

4 Issues **25**
- 4.1 Bug #1 . . . 25
- 4.2 Bug #2 . . . 25

5 Schematic **26**

1. Device Overview

The I2C Encoder V2 is a small board where you can use a classical mechanical encoder, or an illuminated RGB encoder on I^{2}C bus. The device has also 3 configurable GPIOs with the same footprint of RGB LED. It's possible to connect up to 127 boards in cascade and read all of them with the same I^{2}C bus.
The I2C Encoder V2 has a series of 8 bit registers where it is possible to configure the parameters and four 32 bit of registers. These 32 bit registers store *counter value*, value of *increment steps*, *maximum* and *minimum thresholds*. Every time when encoder rotates at least one step, the *counter value* increases or decreases according to the rotation direction by the value of the *increment steps* register .
When the *counter value* is outside of the limit set by the *thresholds registers*, the counter value can be wrapped or can stuck on the threshold value reached.
The I2C Encoder V2 also has an open-drain interrupt pin. It is set to logic low every time an interrupt occurs, the source of interrupt can be customized.

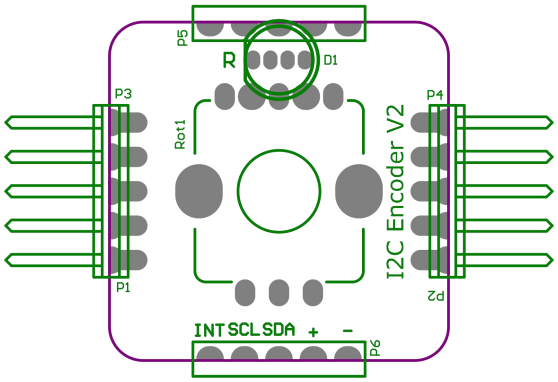


Figure 1.1: Top view of the board

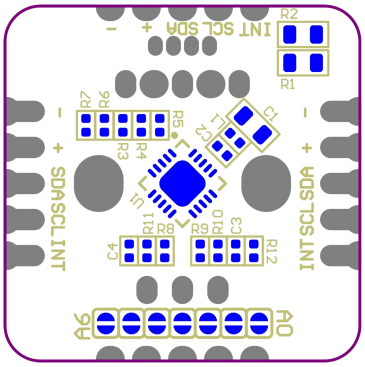


Figure 1.2: Bottom view of the board

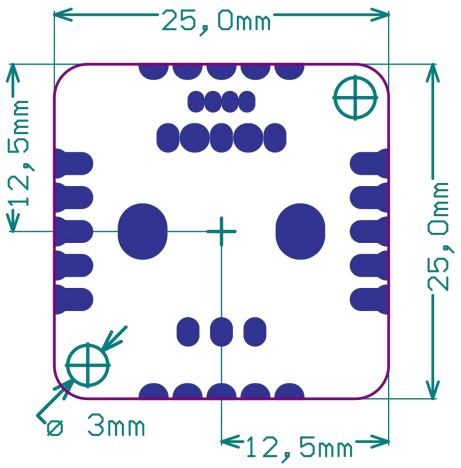


Figure 1.3: Dimensions of the board

1.1 Electrical characteristics

Parameter	Symbol	Min	Max
Supply voltage	V_{DD}	3V	5V
I^2C input-low level	V_{IL}	0	0.3 * V_{DD}
I^2C input-high level	V_{IH}	0.8 * V_{DD}	V_{DD}
I^2C clock input frequency	f_{SCL}		400kHz
Supply current (LEDs off)	I_{DD}		1.8mA
Interrupt pull-up resistor	R_{INT}	15kΩ	120kΩ

1.2 Connection

Figure 1.4 shows the pin-out of the I2C Encoder V2.

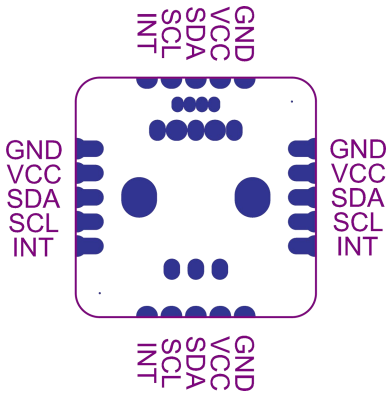


Figure 1.4: Pin-out of the board

Pin	I/O Type	Function
GND	Power	Ground reference for logic
Vcc	Power	Positive supply for logic
SDA	I/O	I^2C data
SCL	I	I^2C clock
INT	OD	Open-drain interrupt output

There are two 5 pin headers on the right and left sides of the I2C Encoder V2. There are also 5 castellated holes on each side.
The I2C Encoder V2 can be connected in cascade by soldering the castellated holes as showed in figure 1.5. In order to avoid I^2C address conflict, the address of each device must be different. In the section 1.3, it is described how to set the address.

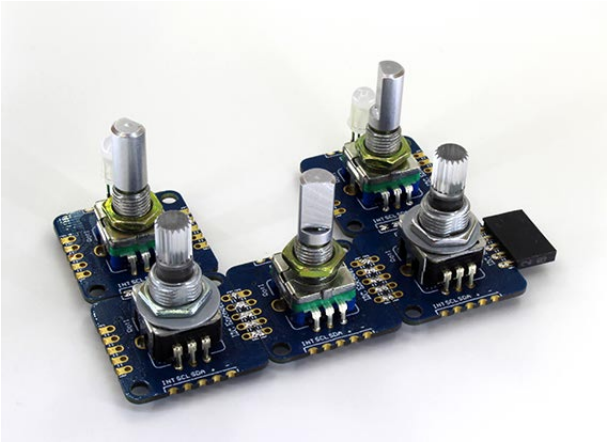


Figure 1.5: Example of 5 boards connected in a matrix with the castellated holes

1.3 I²C interface

The I2C Encoder V2 is a I²C slave, it's possible to the set 127 different addresses. The 7-bit I²C address is represented in a binary number with the jumpers A0 - A6 shown in the figure 1.6. It can be customized by soldering the jumpers. When the jumper is open, it means a logic 0. if jumper is shorted it means a logic 1. In figure 1.7 is shown some examples of I²C address setting.

Avoid to set the address 0, all the jumpers open. The address 0 is a reserved address in the I²C bus.

I²C address							
7	6	5	4	3	2	1	0
A6	A5	A4	A3	A2	A1	A0	R/W

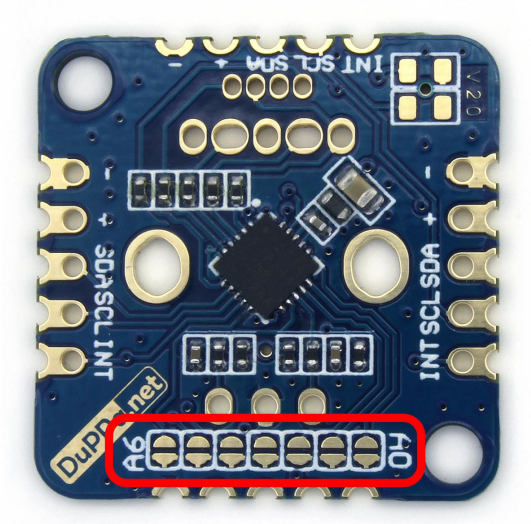


Figure 1.6: Jumpers location

The I2C Encoder V2 has I²C pull-up resistors, by default they are not soldered. It's possible to solder two resistors **R1** and **R2**. This must be done in case that the master doesn't have these resistors and must be enabled only one I2C Encoder V2 in a chain. A typical value of this resistors are 4.7kΩ.

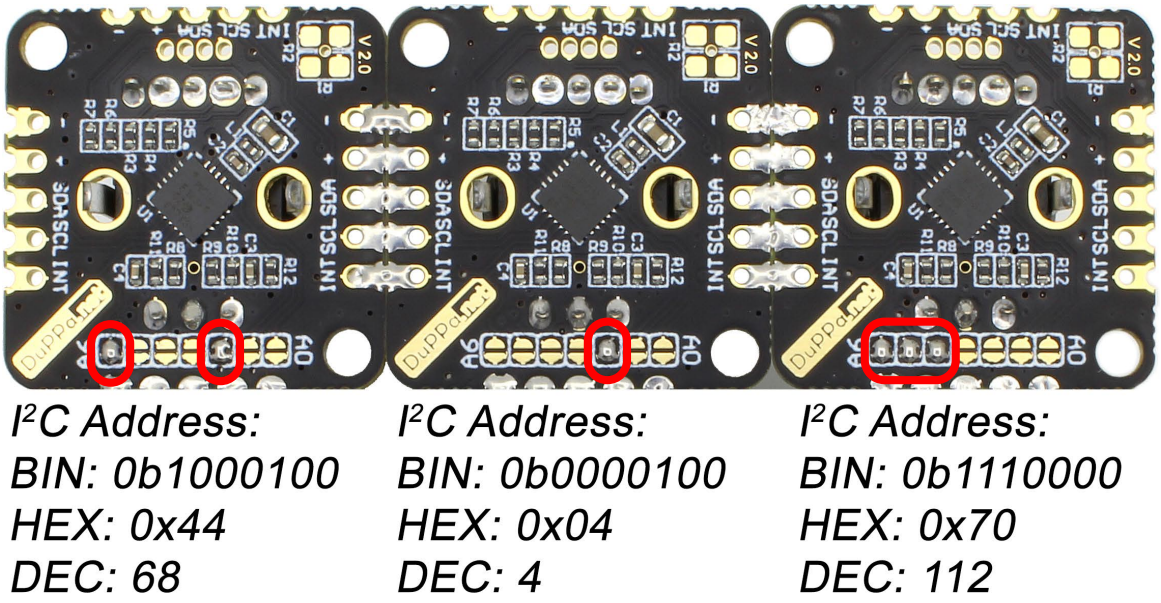


Figure 1.7: Example of the address setting

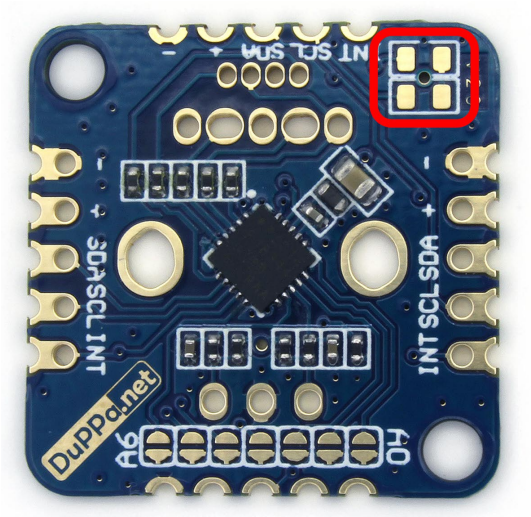


Figure 1.8: Pull-up resistors location

1.4 Rotary encoder

On the I2C Encoder V2, it is possible to solder a standard rotary encoder and an illuminated RGB encoder.

Figure 1.9: EC11 encoder with 20mm shaft

Figure 1.10: Illuminated RGB encoder

It's possible to configure several option by setting the registers with the I^2C bus. With the **GCONF** register, it is possible to configure several parameters.
In the configuration, it's possible to set the polarity of the encoder quadrature signals and also to set if the output of the encoder is X1 or X2.

For reading the rotary encoder movement, there are 4 32bit registers: **CVAL**, **CMAX**, **CMIN** and **ISTEP**. All of these 4 registers can be configured to work as 32bit int or as IEEE 754 floating number, this format can be set in the **GCONF** register.
The counter limits are the following:

- **32bit INT:** from $2.147.483.648$ to $+2.147.483.647$
- **IEEE 754 float:** from $126,0$ to $+127,0$

In case of 32bit int, it is not necessary to read all the 4 byte, you can read only the first 8 bit or the first 16 bit. For example, if you want to count between 0 and 10, you can read only the first byte of the **CVAL** register. In this way you can save I2C transactions.
Every time the encoder moves one step, the value of the **CVAL** register is increased or decreased of the value of **ISTEP**. The direction of the rotation decides if **ISTEP** is added or subtracted from **CVAL**.
CMAX and **CMIN** are used for setting a minimum and maximum thresholds of **CVAL**. In the **GCONF** register, there is **WRAPE** bit. This bit is used to enable or disable a wrap functionality of **CVAL** when it exceeds from the thresholds.
For example, if i configure the I2C Encoder V2 as following:

- **CVAL**= 0
- **CMAX** = 5
- **CMIN** = -5
- **ISTEP**= 1

I will have **CVAL** is incremented of 1 at each rotation step of the encoder. The maximum value that **CVAL** can reach will be 5 while the minimum is -5. In the figure 1.11 shows the value of **CVAL**.
As showed in the figure 1.12, when **WRAPE** is set to 1 when **CVAL** reaches the value of 5, at the next increment **CVAL** it will be wrapped to -5.
Every time when the encoder is rotated one step and when **CVAL** touch the thresholds, an interrupt is generated and is possible to read in the register **ESTATUS**.

The I2C Encoder V2 support also the rotary encoder with the push button. When the push button is pressed an interrupt is generated at the rising and falling edge. Inn this way, it is possible to check when the push button is pressed or released.
There is also possibility to read a fast double push by setting a window time in the register **DPPERIOD**. When a double push is made inside of the **DPPERIOD** window, an interrupt is generated.
If the **DPPERIOD** is 0, the double push function is disabled.
All the above interrupt are possible to read in the register **ESTATUS**, and can be also disabled with the register **INTCONFIG**.

The I2C Encoder V2 has a RC filer as anti-bouncing filter, but sometimes is not enough especially when the encoder is rotated slowly. The bouncing can happen and you can see it as fast rotation in the opposite direction. To avoid this problem, there is the register **ANTBOUNC**. With this register is possible to set a period where the inversion of rotation can be ignored.

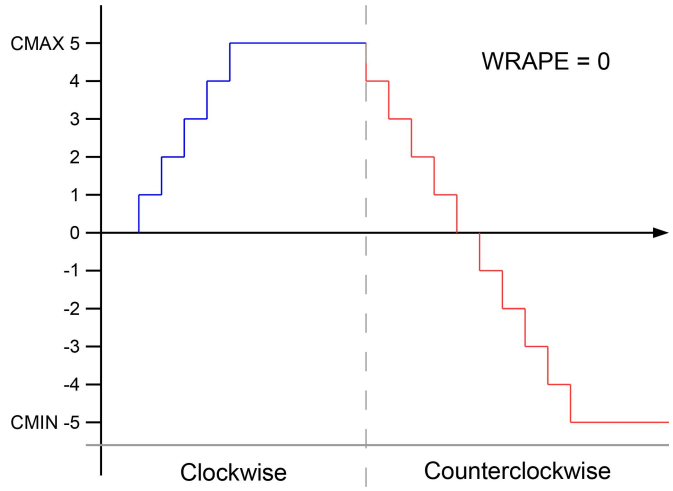


Figure 1.11: Blue and red line are the **CVAL** values when the encoder is rotate and the **WRAPE** is disabled

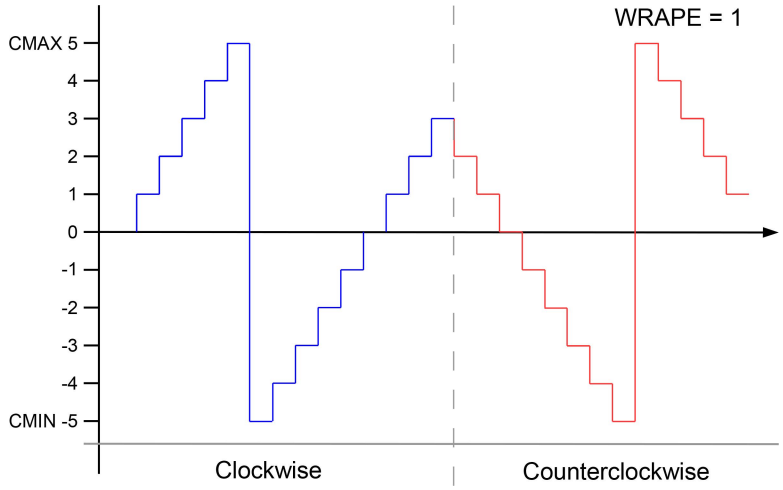


Figure 1.12: Blue and red line are the **CVAL** values when the encoder is rotate and the **WRAPE** is enabled

1.5 GP pins

There are 3 configurable GP pins, they are called GP1, GP2 and GP3. They have the same footprint of an 5mm RGB led. In figure 1.13, it is shown the position. Each pin can be configured to be:

➤ **PWM:** The pins is configured to be a PWM output, the resolution is 8 bit.

➤ **Analog input:** The pins is configured to be an input of the internal ADC, the resolution is 8 bit.

➤ **GPIO output:** The pins is configured to be a digital output.

➤ **GPIO Input:** The pin is configured to be a digital input. It's possible to set an interrupt at every signal edge.

In case of a RGB encoder is mounted, the GP3 will not be available. This because GP3 is internally connect to the red color of the RGB led and it is automatically configured to be a PWM.

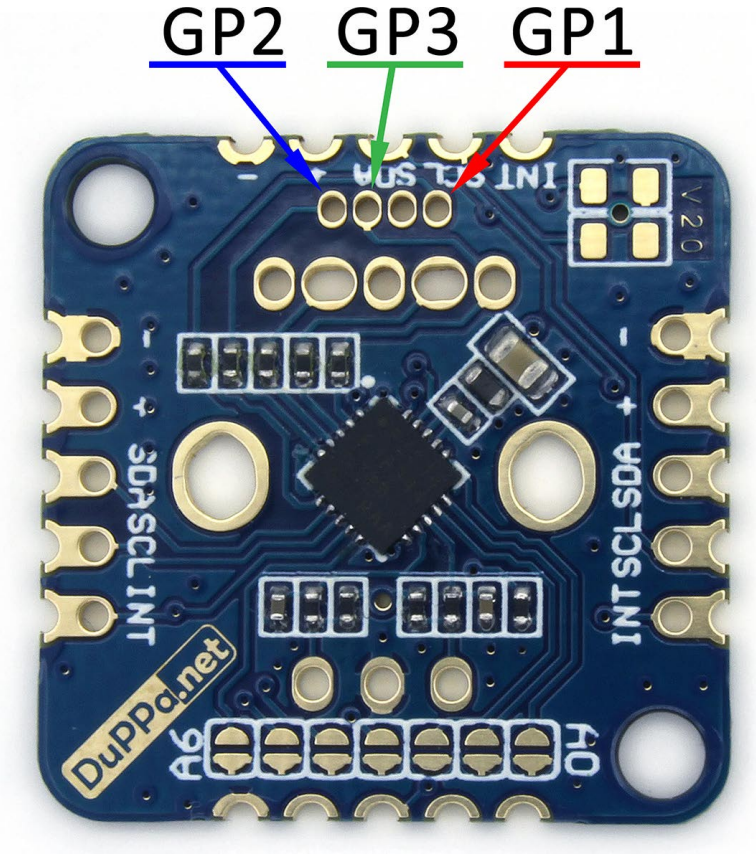


Figure 1.13: GP pins pinout

1.6 Fade function

The I2C Encoder V2 has an auto fading feature. This function can be enabled for the RGB LED integrated on the rotary encoder, or in the PWM output of the GP pins.
There are two registers:

- **FADERGB**: for setting the RGB LED of the rotary encoder
- **FADEGP**: for setting the GP pins

The value you write inside these registers is the step speed of the fading ramp, and it's in *ms*. If the value is 0, the fade feature is disabled. It means that when the new value of PWM is written, it is immediately updated to the output.
The fading process starts when a PWM value is written. Fading process is complete according to the PWM value you have set. When the internal fade PWM value is the same of the PWM value (it means that when fading is complete), an interrupt will be generated. Let's make an example:

At the startup, i write as:

- **RLED** = 0
- **GLED** = 0
- **BLED** = 0
- **FADERGB** = 1

In above case, the LEDs of the encoder are off and the fade has 1ms step size.
Now i write as: *RLED=0xFF*.
This moment, the ramp of the **RLED** starts and reaches the value 0xFF in 255ms. (the other LEDs remain OFF)
When the ramp reaches 0xFF, i get an interrupt from the **INT** pin.
After this, i write as: *RLED=0x00*. At this point, the fade ramp starts again and it will turn off the LED in 255ms, and an interrupt will be generated at the end.
The figure 1.14 is showing an example of the fade output when a PWM value is written in different moments, as well as when the interrupt of the fade is generated.

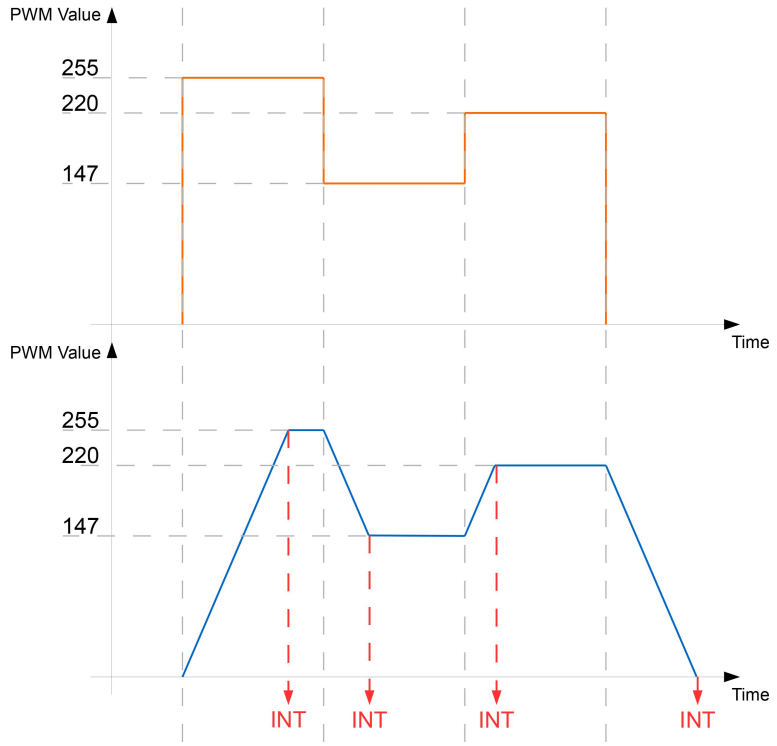


Figure 1.14: Time diagram of the fade output when writing on the PWM register

1.7 EEPROM

The I2C Encoder V2 has 256 bytes of EEPROM.
This memory is divided in two banks of 128 bytes. With the bit **MBANK** in the **GCONF** register, it is possible to choose the bank 1 or the bank 2.
The memory area is between 0x80 and 0xFF address. To use the EEPROM, user only needs to perform reading or a writing in these address areas.

1.8 Interrupt

The I2C Encoder V2 has multiple interrupt source previously described. When an interrupt is generated, the **INT** pin is tied low. By reading the register **ESTATUS**, the interrupts are cleared and the **INT** pin returns high.
The INT pin is open-drain output. Hence it requires an external pull-up resistor, or internal pull-up resistor can be enabled by setting the bit **IPUD** to 1.
In a chain of I2C Encoder V2 all the **INT** pins can be connected together, like the pin of the I^2C.When an interrupt occurs, user has to scan the boards in the chain to find who generates the interrupt.
With the register **INTCONF**, it is possible to enable or disable interrupt. When an interrupt is disabled, the corresponding bit is set, but the **INT** pin is not affected.

2. Registers

In this section, the internal registers of I2C Encoder V2 is described.

Address range	Name	Description	Dimension	Default value
0x00	GCONF	General Configuration	1 Byte	0
0x01	GP1CONF	GP 1 Configuration	1 Byte	0
0x02	GP2CONF	GP 2 Configuration	1 Byte	0
0x03	GP3CONF	GP 3 Configuration	1 Byte	0
0x04	INTCONF	INT pin Configuration	1 Byte	0
0x05	ESTATUS	Encoder Status	1 Byte	0
0x06	I2STATUS	Secondary interrupt status	1 Byte	0
0x07	FSTATUS	Fade process status	1 Byte	0
0x08 - 0x0B	CVAL	Counter Value	4 Byte	0
0x0C - 0x0F	CMAX	Counter Max value	4 Byte	0
0x10 - 0x13	CMIN	Counter Min value	4 Byte	0
0x14 - 0x17	ISTEP	Increment step value	4 Byte	1
0x18	RLED	LED red color intensity	1 Byte	0
0x19	GLED	LED green color intensity	1 Byte	0
0x1A	BLED	LED blue color intensity	1 Byte	0
0x1B	GP1REG	I/O GP1 register	1 Byte	0
0x1C	GP2REG	I/O GP2 register	1 Byte	0
0x1D	GP3REG	I/O GP3 register	1 Byte	0
0x1E	ANTBOUNC	Anti-bouncing period	1 Byte	25
0x1F	DPPERIOD	Double push period	1 Byte	0
0x20	FADERGB	Fade timer RGB Encoder	1 Byte	0
0x21	FADEGP	Fade timer GP ports	1 Byte	0
0x80 - 0xFF	EEPROM	EEPROM memory	128 Byte	0

2.1 Configuration

2.1.1 General Configuration

GCONF Address: **0x00**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RESET	MBANK	ETYPE	RMOD	IPUD	DIRE	WRAPE	DTYPE

❖ **DTYPE** Data type of the register: **CVAL**, **CMAX**, **CMIN** and **ISTEP**.

1: The registers are considered float numbers IEEE 754

0: The registers are considered int 32bit

❖ **WRAPE** Enable counter wrap.

1: Wrap enable. When the counter value reaches the **CMAX**+1, restart to the **CMIN** and vice versa

0: Wrap disable. When the counter value reaches the **CMAX** or **CMIN**, the counter stops to increasing or decreasing

❖ **DIRE** Direction of the encoder when increment.

1: Rotate left side to increase the value counter

0: Rotate right side to increase the value counter

❖ **IPUD** Interrupt Pull-UP disable.

1: Disable

0: Enable

❖ **RMOD** Reading Mode.

1: X2 mode

0: X1 mode

❖ **ETYPE** Set the encoder type

1: RGB illuminated encoder

0: Standard encoder

❖ **MBANK** Select the EEPROM memory bank. Each bank are 128 byte wide

1: Second memory bank

0: First memory bank

❖ **RST** Reset of the I2C Encoder V2

1: Reset

0: No reset

2.1.2 GP1 Configuration

GP1CONF Address: **0x01**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	-	GP1INT		GP1PUL	GP1MODE	

❖ **GP1MODE** Configuration of the pin GP1

00: PWM output
01: GPIO output Push-Pull
10: Analog input
11: GPIO Input

❖ **GP1PUL** Enable or disable the internal pull-up.

0: Pull-UP disable
1: Pull-UP enabled

❖ **GP1INT** Configuration of the interrupt, available only when the pin is configured as input

00: Interrupt disabled
01: Interrupt on positive edge
10: Interrupt on negative edge
11: Interrupt on both edges

2.1.3 GP2 Configuration

GP2CONF Address: **0x02**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	-	GP2INT		GP2PUL	GP2MODE	

❖ **GP2MODE** Configuration of the pin GP2

00: PWM output
01: GPIO output Push-Pull
10: Analog input
11: GPIO Input

❖ **GP2PUL** Enable or disable the internal pull-up.

0: Pull-UP disable
1: Pull-UP enabled

❖ **GP2INT** Configuration of the interrupt, available only when the pin is configured as input

00: Interrupt disabled
01: Interrupt on positive edge
10: Interrupt on negative edge
11: Interrupt on both edges

2.1.4 GP3 Configuration

GP3CONF Address: **0x03**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	-	GP3INT		GP3PUL	GP3MODE	

Note: When the **ETYPE** bit is set, this pin is in PWM mode automatically and can't be changed.

❖ **GP3MODE** Configuration of the pin GP3

00: PWM output
01: GPIO output Push-Pull
10: Analog input
11: GPIO Input

❖ **GP3PUL** Enable or disable the internal pull-up.

0: Pull-UP disable
1: Pull-UP enabled

❖ **GP3INT** Configuration of the interrupt, available only when the pin is configured as input

00: Interrupt disabled
01: Interrupt on positive edge
10: Interrupt on negative edge
11: Interrupt on both edges

2.2 Interrupt output Configuration

This register is used for enable or disable the interrupt source selectively. When an interrupt event occurs, the **INT** pin goes low and the event is stored in the status register.

INTCONF Address: **0x04**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
INT2	IRMIN	IRMAX	IRDEC	IRINC	IPUSHD	IPUSHP	IPUSHR

❖ **IPUSHR** Push button release bit

1: Interrupt enabled when the push button is released.

0: Interrupt disabled

❖ **IPUSHP** Push button press bit

1: Interrupt enabled when the push button is pressed.

0: Interrupt disabled

❖ **IPUSHD** Push button double press

1: Interrupt enabled when the push button is double pressed.

0: Interrupt disabled

❖ **IRINC** Rotary encoder direction of increase

1: Interrupt enabled when the encoder is rotated in the direction of increase

0: Interrupt disabled

❖ **IRDEC** Rotary encoder direction of decrease

1: Interrupt enabled when the encoder is rotated in the direction of decrease

0: Interrupt disabled

❖ **IRMAX** **CVAL** reaches **CMAX** bit

1: Interrupt enabled when **CVAL** reaches **CMAX**

0: Interrupt disabled

❖ **IRMIN** **CVAL** reaches **CMIN** bit

1: Interrupt enabled when **CVAL** reaches **CMIN**

0: Interrupt disabled

❖ **INT2** Enable the secondary interrupts

1: Secondary interrupt enabled

0: Secondary interrupt disabled

2.3 Status

2.3.1 Encoder Status

This register if only readable, when is read is automatically cleared.

ESTATUS Address: **0x05**							
7	6	5	4	3	2	1	0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
INT2	RMIN	RMAX	RDEC	RINC	PUSHD	PUSHP	PUSHR

❑ **PUSHR** Status of the push button of the encoder

1: Push button is released

0: Push button is not released

❑ **PUSHP** Status of the push button of the encoder

1: Push button is pressed

0: Push button is not pressed

❑ **PUSHD** Status of the push button of the encoder

1: Push button is double pressed

0: Push button is not double pressed

❑ **RINC** Rotary encoder is rotated in the increase direction

1: Encoder is rotated

0: Encoder is not rotated

❑ **RDEC** Rotary encoder is rotated in the decrease direction

1: Encoder is rotated

0: Encoder is not rotated

❑ **RMAX** Status of the counter value

1: **CVAL** reaches the **CMAX** value

0: **CVAL** is below the **CMAX** value

❑ **RMIN** Status of the counter value

1: **CVAL** reaches the **CMIN** value

0: **CVAL** is above the **CMIN** value

❑ **INT2** Secondary interrupt status

1: Secondary interrupt event occurs

0: No secondary event occurs

2.3.2 Secondary interrupt status

In this register is possible to check the event of the fade process as well as the GP pins when is configured as input. This register if only readable, when is read is automatically cleared.

I2STATUS Address: **0x06**							
7	6	5	4	3	2	1	0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
-	FADE	EGP3		EGP2		EGP1	

❑ **GP1** Configuration of the GP1

00: No event occurs

01: Positive edge event

10: Negative edge event

❑ **GP2** Configuration of the GP2.

00: No event occurs

01: Positive edge event

10: Negative edge event

❑ **GP3** Configuration of the GP3. Available only with the standard encoder

00: No event occurs

01: Positive edge event

10: Negative edge event

❑ **FADE** Fade event occurs

1: Fade process event

0: No event in the fade process

2.3.3 Fade process status

With this register is possible to check the status of the PWM channel during the fade process. This register is only readable and the bits are set or cleared automatically during the fade process.

FSTATUS Address: **0x07**							
7	6	5	4	3	2	1	0
R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
-	-	FGP3	FGP2	FGP1	FEB	FEG	FER

❑ **FER** Fade status process of the RGB Encoder color red

1: Fade process is running

0: Fade process terminated

❑ **FEG** Fade status process of the RGB Encoder color green

1: Fade process is running

0: Fade process terminated

❑ **FEB** Fade status process of the RGB Encoder color blue

1: Fade process is running

0: Fade process terminated

❑ **GP1** Fade status process of the GP pin 1

1: Fade process is running

0: Fade process terminated

❑ **GP2** Fade status process of the GP pin 2

1: Fade process is running

0: Fade process terminated

❑ **GP3** Fade status process of the GP pin 3

1: Fade process is running

0: Fade process terminated

2.4 Encoder registers

2.4.1 Counter Value

CVAL Address: **0x08**							
31	30	29	28	27	26	25	24
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVAL BYTE 4 <31 - 24>							
Address: **0x09**							
23	22	21	20	19	18	17	16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVAL BYTE 3 <23 - 16>							
Address: **0x0A**							
15	14	13	12	11	10	9	8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVAL BYTE 2 <15 - 8>							
Address: **0x0B**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CVAL BYTE 1 <7 - 0>							

2.4.2 Counter Max

CMAX Address: **0x0C**							
31	30	29	28	27	26	25	24
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMAX BYTE 4 <31 - 24>							
Address: **0x0D**							
23	22	21	20	19	18	17	16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMAX BYTE 3 <23 - 16>							
Address: **0x0E**							
15	14	13	12	11	10	9	8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMAX BYTE 2 <15 - 8>							
Address: **0x0F**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMAX BYTE 1 <7 - 0>							

2.4.3 Counter Min

CMIN Address: **0x10**							
31	30	29	28	27	26	25	24
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMIN BYTE 4 <15 - 8>							
Address: **0x11**							
23	22	21	20	19	18	17	16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMIN BYTE 3 <7 - 0>							
Address: **0x12**							
15	14	13	12	11	10	9	8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMIN BYTE 2 <15 - 8>							
Address: **0x13**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CMIN BYTE 1 <7 - 0>							

2.4.4 Increment step

ISTEP Address: **0x14**							
31	30	29	28	27	26	25	24
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ISTEP BYTE 4 <15 - 8>							
Address: **0x15**							
23	22	21	20	19	18	17	16
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ISTEP BYTE 3 <7 - 0>							
Address: **0x16**							
15	14	13	12	11	10	9	8
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ISTEP BYTE 2 <15 - 8>							
Address: **0x17**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ISTEP BYTE 1 <7 - 0>							

2.5 LEDs registers

2.5.1 LED Red intensity

RLED Address: **0x18**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
RED LED PWM Value <7 - 0>							

This register is used for setting the PWM of the red LED of the RGB encoder. A value of **0x00** means PWM at 0%, LED OFF. A value of **0xFF** means PWM at 100%, LED completely ON.

2.5.2 LED Green intensity

GLED Address: **0x19**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GREEN LED PWM Value <7 - 0>							

This register is used for setting the PWM of the green LED of the RGB encoder. A value of **0x00** means PWM at 0%, LED OFF. A value of **0xFF** means PWM at 100%, LED completely ON.

2.5.3 LED Blue intensity

BLED Address: **0x1A**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BLUE LED PWM Value <7 - 0>							

This register is used for setting the PWM of the blue LED of the RGB encoder. A value of **0x00** means PWM at 0%, LED OFF. A value of **0xFF** means PWM at 100%, LED completely ON.

2.6 GPs registers

The usage of these register depends of the configuration of the GP pins:

➤ **PWM** Is possible to write the PWM value.

➤ **GPIO output:** Is possible to write 1 or 0 for setting the output logic level.

➤ **Analog input:** Is possible to read the ADC value

➤ **GPIO Input:** Is possible to read the logic level of the output

2.6.1 GP1 register

GP1REG Address: **0x1B**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GP1 Value <7 - 0>							

2.6.2 GP2 register

GP2REG Address: **0x1C**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GP2 Value <7 - 0>							

2.6.3 GP3 register

GP3REG Address: **0x1D**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GP3 Value <7 - 0>							

2.7 Timing registers

This register are used for changing some timing parameter

2.7.1 Anti-bouncing period

This register is used for setting the anti-bounce period. The value is in ms × 10. For example, when you rotate the encoder especially at low speed, some times bouncing can happen and you can see it when rotation is in the opposite direction. With this register, it is possible to set the period where the opposite rotation is ignored. The default value is 25, that means that the de-bounce time is 250ms.

ANTBOUNC Address: **0x1E**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-1
PBOUN <7 - 0>							

2.7.2 Double push time

This register is used for setting the double push of the rotary encoder switch. The value is in ms × 10. When this register is 0 this function is disabled.

DPPERIOD Address: **0x1F**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PDPUSH <7 - 0>							

2.7.3 Fade RGB encoder LED

This register is used for setting the fade speed of the RGB LED of the rotary encoder. The value is in ms and indicate the time of one step. If a standard rotary encoder is used, this register is not used.

FADERGB Address: **0x20**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FADERGB <7 - 0>							

2.7.4 Fade GP ports

This register is used for setting the fade speed of the GP ports. The value is in ms and indicate the time of one step. This register is working only if the GP port are configured in PWM

FADEGP Address: **0x21**							
7	6	5	4	3	2	1	0
R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
FADEGP <7 - 0>							

3. Reference

This project is open source, the HW and the FW as well as some example can be found on GitHub:
`https://github.com/Fattoresaimon/I2CEncoderV2`

The HW project can be found on Circuit Maker site:
`https://workspace.circuitmaker.com/Projects/Details/Simone--Caron/I2C-Encoder-V2`

4. Issues

In this section, there are listed the known bugs of the I2C Encoder V2.

4.1 Bug #1

Cause: Every time the GCONF (add:0x00) register is wrote, the ISTEP (add:0x14 - 0x17) register is reset to the default value of 1.

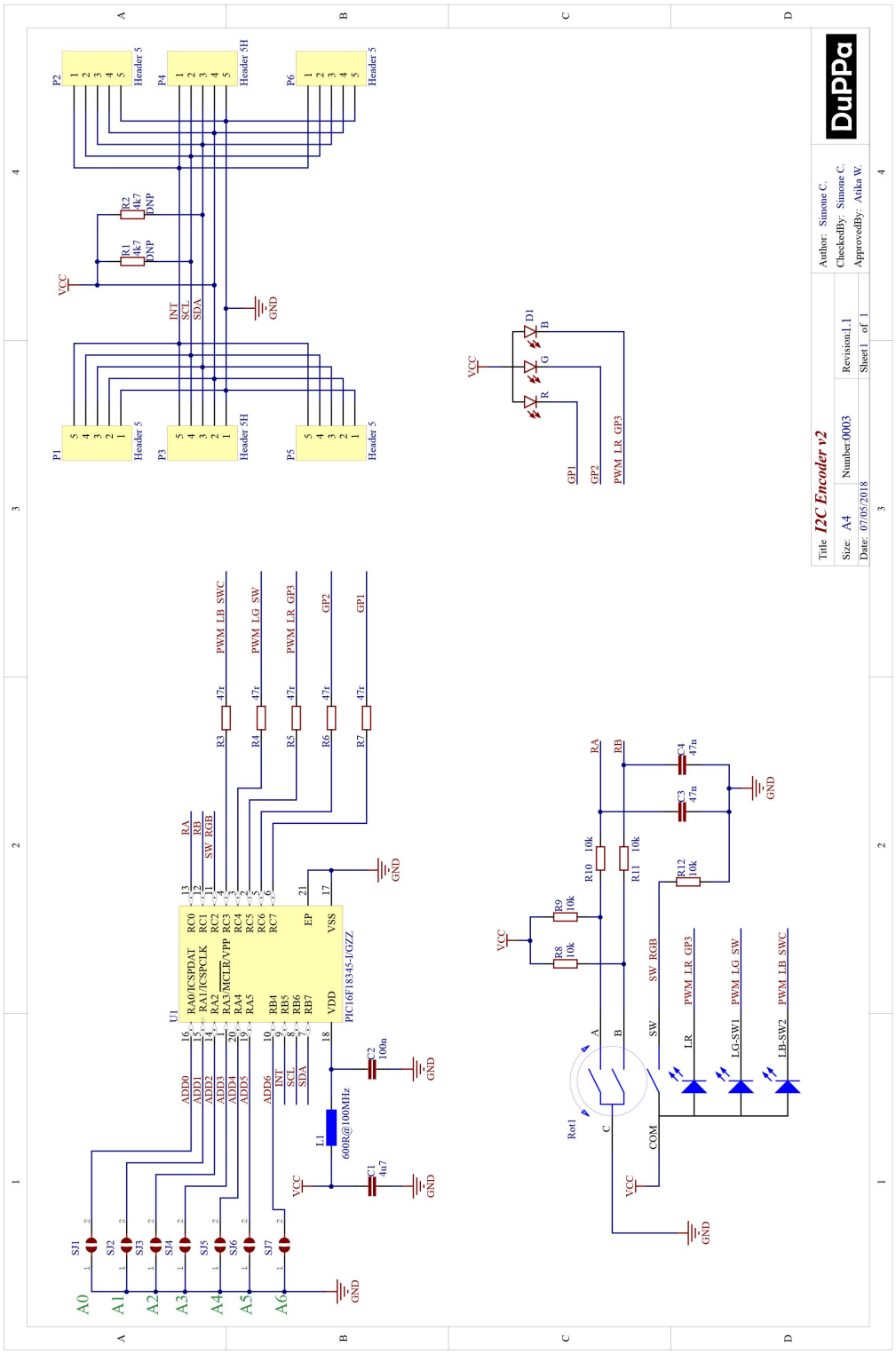
Workaround: After writing the GCONF register restore the correct value of the ISTEP.

4.2 Bug #2

Cause: Program for scanning the device on the I^2C bus, like the i2cdetect or I2C Scanner, doesn't work well with the I2C Encoder V2.

Workaround: Instead of finding the I^2C address in that way, check the address jumpers and communicate directly with the I2C Encoder V2.

5. Schematic


DuPPa
Title I2C Encoder v2
Size: A4
Number:0003
Revision:1.1
Date: 07/05/2018
Sheet1 of 1
Author: Simone C.
CheckedBy: Simone C.
ApprovedBy: Atika W.
P1
P2
P3
P4
P5
P6
Header 5
Header 5H
VCC
GND
INT
SCL
SDA
R1 4k7 DNP
R2 4k7 DNP
D1
R
G
B
GP1
GP2
PWM_LR_GP3
U1
PIC16F18345-I/GZZ
RA0/ICSPDAT
RA1/ICSPCLK
RA2
RA3/MCLR/VPP
RA4
RA5
RB4
RB5
RB6
RB7
RC0
RC1
RC2
RC3
RC4
RC5
RC6
RC7
VDD
VSS
EP
ADD0
ADD1
ADD2
ADD3
ADD4
ADD5
ADD6
SJ1
SJ2
SJ3
SJ4
SJ5
SJ6
SJ7
A0
A1
A2
A3
A4
A5
A6
R3 47r PWM_LB_SWC
R4 47r PWM_LG_SW
R5 47r PWM_LR_GP3
R6 47r GP2
R7 47r GP1
L1 600R@100MHz
C1 4u7
C2 100n
R8 10k
R9 10k
R10 10k
R11 10k
R12 10k
C3 47n
C4 47n
Rot1
A
B
C
SW
COM
SW_RGB
LR
LG-SW1
LB-SW2
PWM_LR_GP3
PWM_LG_SW
PWM_LB_SWC