Technical design

By Manon Sijben, Marlon Reijlink, Emiel Visser and Joris Bol

HAN_

DOCUMENT ORGANIZATIONAL DETAILS:

Document Type:	
Skills Consultant:	Paul van Wegen
Project Client:	Johan Korten
Education:	Embedded Systems Engineering
Institute:	School of Engineering and Automotive, Hogeschool Arnhem-Nijmegen
Report Date:	16-1-2024
Report Version:	v1.0
Project Duration:	
Education Term:	
Students:	2108100 (Joris Bol), 2111740(Emiel Visser), 2106573 (Manon Sijben), 2107003 (Marlon Reijlink)

Version management

Version:	Date:	Change:	Name:
V1.0	5-9-2023	Creation document.	J. Bol
V1.0	17-9-2023	Added paragraphs.	E. Visser
V1.0	16-10-2023	Added info.	E. Visser
V1.0	27-11-2023	Added info.	E. Visser
V1.0	1-12-2023	Added info.	E. Visser
V1.0	9-12-2023	Added info.	E. Visser
V1.0	10-12-2023	Added info.	E. Visser
V1.0	18-12-2023	Added info.	J. Bol
V1.0	16-1-2023	Spellcheck.	E. Visser

Table of contents

1 – Technical design

About the exact reason why we choose these sensors and actuators we would like to point you to our preliminary research.

4.1- Architecture

In figure 1 a diagram is shown showing how our system will be controlled. The boxes contain one or more items which the board will interface with. The arrows between the boxes and board indicate the direction of the data/energy flow. The protocol/voltage is placed on the arrow.

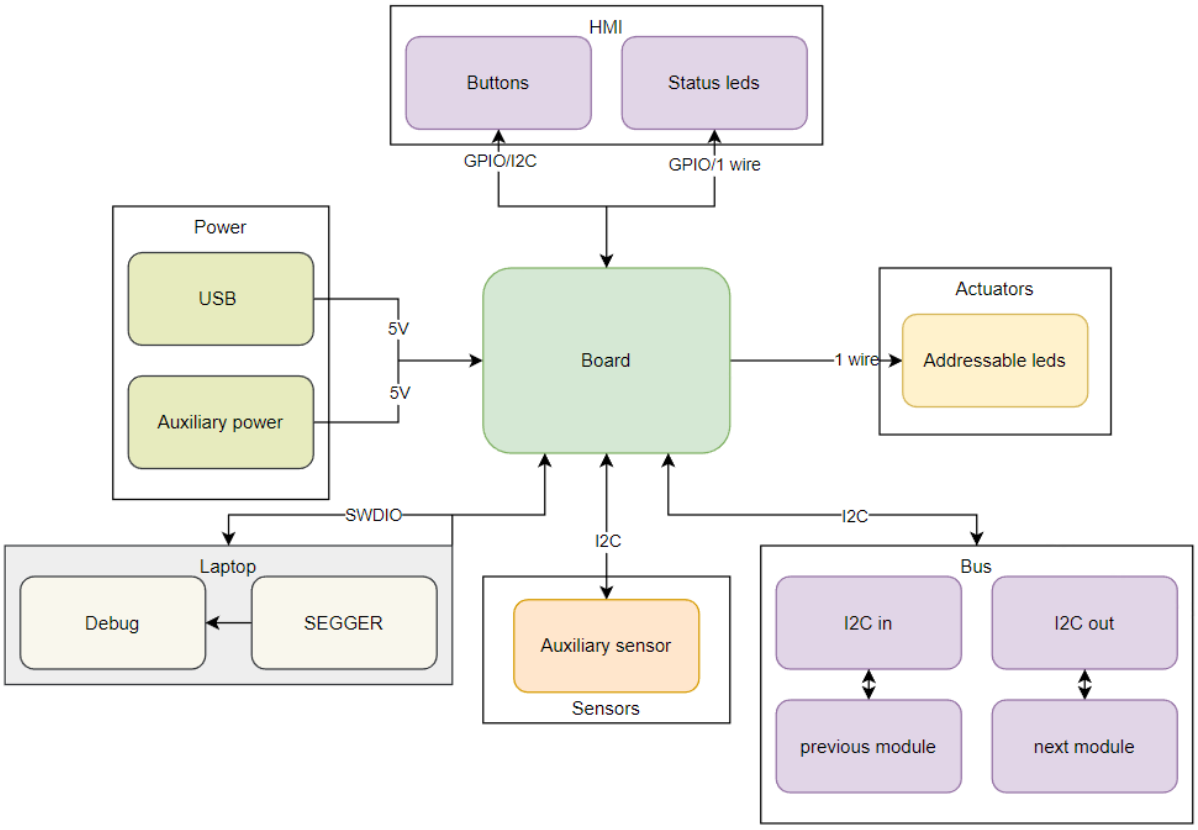


Figure 1: system interconnections

4.2 – Interfaces

In this paragraph all of the interfaces between the board and the items are described. An overview of the interfaces and items can be seen in figure 1.

4.2.1 – Power

In this paragraph the power section of our system will be described.

4.2.1.1 – USB-C

To deliver power and upload code to our board we have added a UCB type C port to our PCB. We have added two test points so that the USB protocol can be debugged if necessary. The USB termination resistors are placed in the main sheet and are there to reduce ringing on the USB data lines. An ESD protection diode protects the host (e.g. a laptop) from ESD. The USB C port can deliver a maximum of 3A in this configuration.

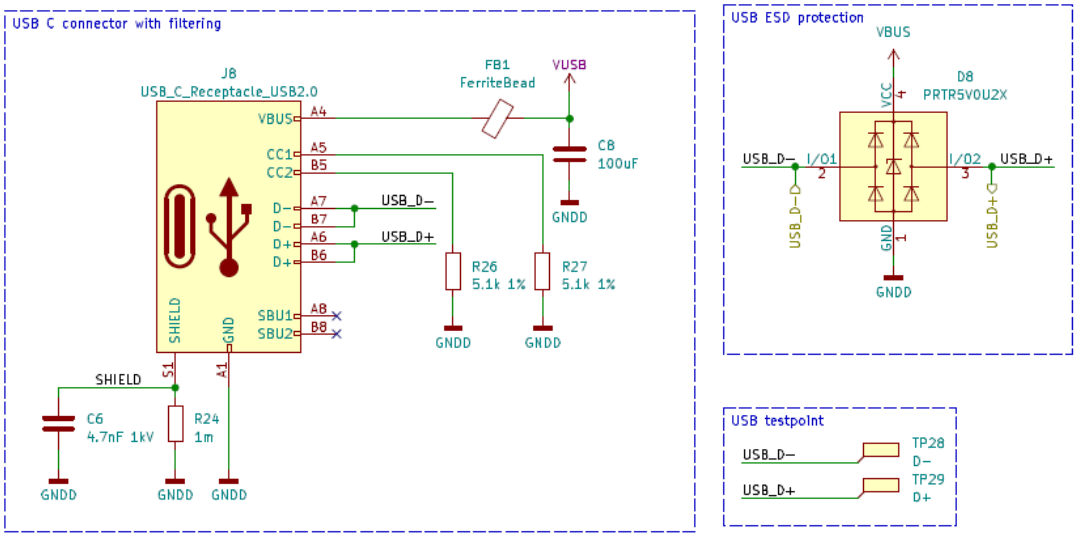
Note: Don't come close to this maximum though because the traces coming out of the port might not be able to handle 3A!

Requirement **supports T7:**

Specification 1.1:

No software scheme for mounting holes and fiducials.

Specification 1.2:



4.1.1.2 – Auxiliary power

A USB C plug can be quite bulky, so we decided to add another way of powering the system. That's way we added a JST-XH port where 5V power can be fed into the system. We also added a fuse holder. We recommend using a fast blow fuse to protect the system in an overcurrent event.

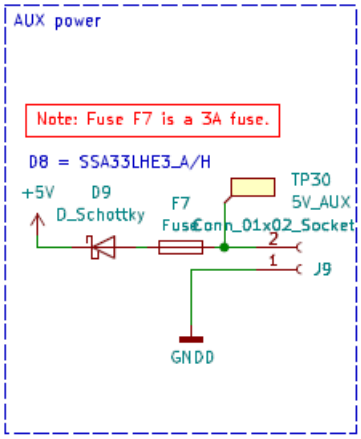
Note: The maximum current for prolonged periods is about 2.5A. For short periods 3A is possible.

Requirement **supports T7:**

Specification 2.1:

No software scheme for aforementioned element.

Specification 2.2:



4.1.1.3 – 3.3V regulator

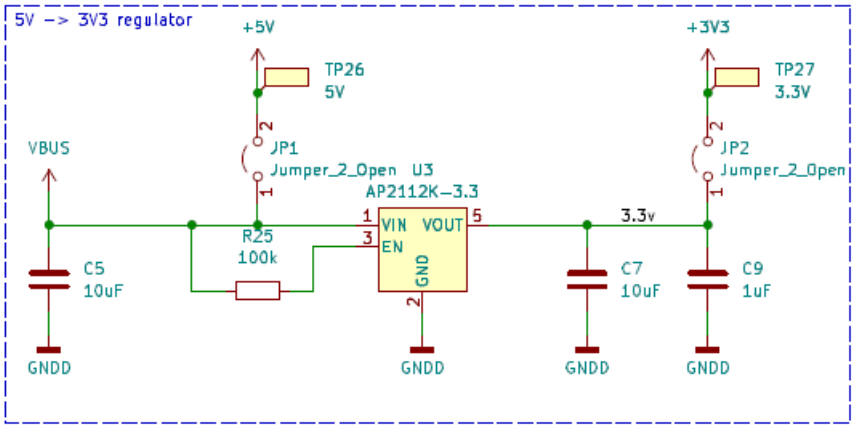
We have decided top go with an AP2112K type regulator. This regulator drops the auxiliary and/or USB voltage down to 3.3V which is used by the microcontroller and other components. The regulator is decoupled using three decoupling capacitors and will deliver 3.3V when 5V is applied to its input.

Requirement **supports T7:**

Specification 3.1:

No software scheme for aforementioned element.

Specification 3.2:



4.2.2 – Status leds

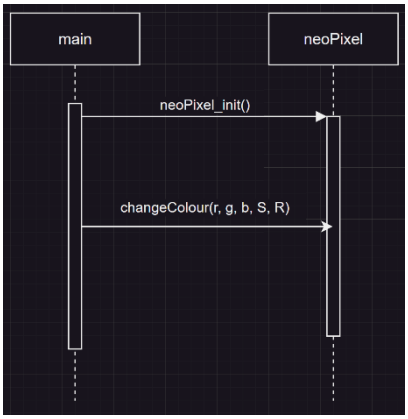
We have integrated multiple leds onto the board to provide users with clear visual indicators for various processes taking place on the controller board. These leds serve as a user-friendly interface, offering real-time feedback on the ongoing activities and status of the system.

4.2.2.1 – Status led (neopixel)

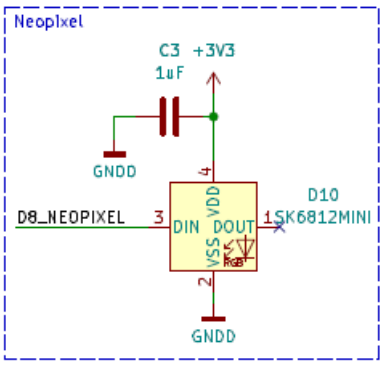
We have opted for the SK6812MINI Neopixel led due to its compact package size. Additionally, these leds offer significant IO savings, a crucial factor because we decided to go with the ATSAMD21G18 instead of the original ATSAMD21J18 microcontroller. Another compelling reason for choosing this specific led is its compatibility with the existing protocol we are using for other leds. This not only streamlines our development process but also minimizes the need for additional code, promoting efficiency and consistency across our lighting system.

Requirement: **supports T10**

Specification 4.1:



Specification 4.2:

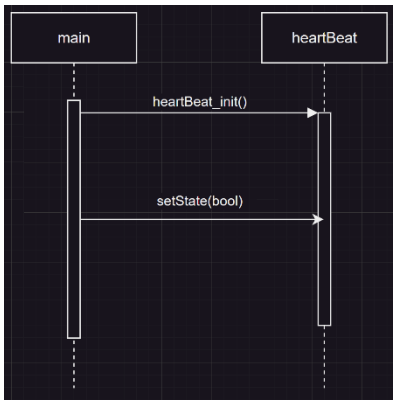


4.2.2.2 – HB led (heartbeat)

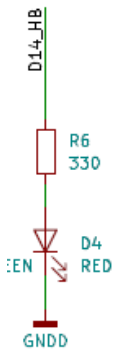
The Heartbeat led blinks at a fixed frequency. In the event that this led stops blinking, it indicates a system crash or that the system is stuck in a loop. Typically, a system reset is the recommended solution to address such issues.

Requirement **supports T10**

Specification 5.1:



Specification 5.2:

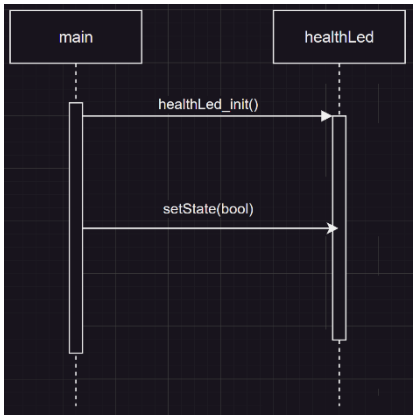


4.2.2.3 – Health led

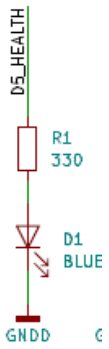
This led will be on when the board is in good health. And will turn of when the board encounters errors that it can't handle properly.

Requirement **supports T10**

Specification 6.1:



Specification 6.2:



4.2.2.4 – Voltage indicator leds

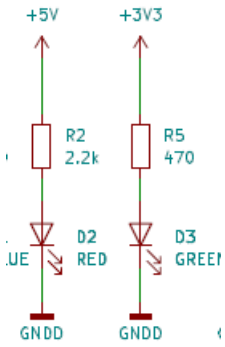
We've added voltage indicator leds on the board for a quick check on the availability of the two supply voltages.

Requirement **supports T10**

Specification 7.1:

No software scheme for aforementioned element.

Specification 7.2:



4.2.2.5 – Receive and transmit leds

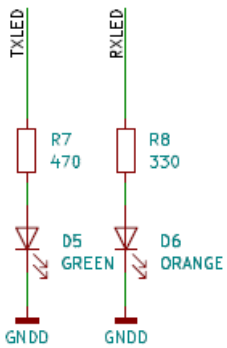
To give the user an indication if the serial bus is working, we have added a TX and RX led.

Requirement **T10**

Specification 8.1:

No software scheme for aforementioned element.

Specification 8.2

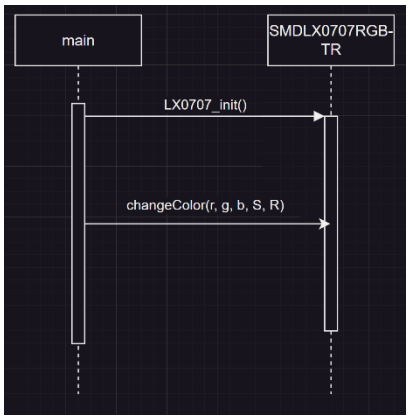


4.2.3 – Main leds

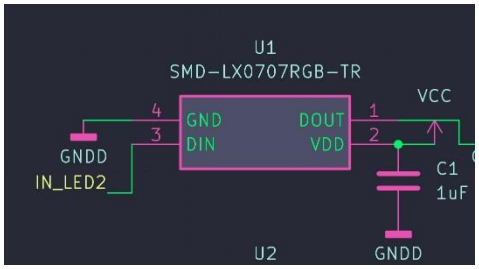
We have found that the SMD-LX0707RGB-TR leds fit our requirements.

Requirement **T5.3**

Specification 9.1:



Specification 9.2



4.2.3.1- Neopixel output drivers

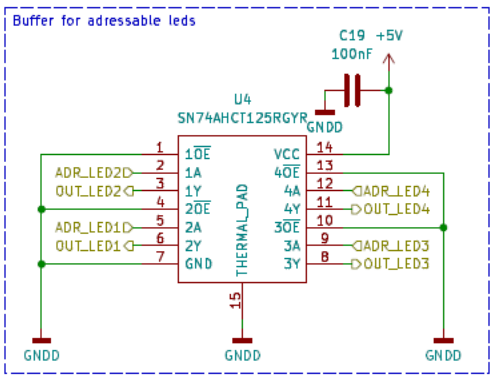
To convert the 3.3V microcontroller output signal to a 5V output signal which results in a more stable signal. We have chosen the SN74AHCT125RGYR buffer IC. In our tests we have used the SN74AHCT125DR. the RGYR version has a smaller package which is why we went with this version for our latest revision. Every in- and output has a test point.

Requirement **supports T5.3.**

Specification 10.1:

No software scheme for aforementioned element.

Specification 10.2:



4.2.3.2 – FFC/FPC flat cable connectors

To connect the flex PCBs to our control PCB we are using 16pin FFC/FPC connectors. We are using 1A resettable fuses on the 5V and 3.3V power rails. The 62-ohm resistors are there to prevent ringing on the data lines. We have also opted to breakout an I2C bus to pin 11 and 12 on the connector which enables a next group to add a I2C enabled device to be used on the flex PCBs. We also decided to breakout two IO's to pin 8 and 9 on the connector. The user can use the for general purpose applications.

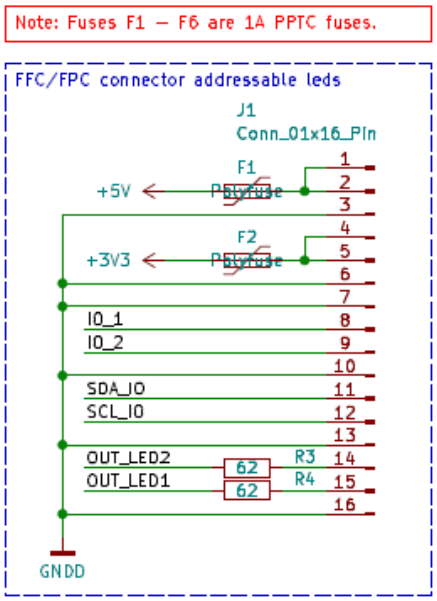
Note: The connector only makes contact on the under side so if you want your parts to be on top of the flexible PCB you need to place the connector on the underside!

Requirement **T5**

Specification 11.1:

No software scheme for aforementioned element.

Specification 11.2:



4.2.4 – Microcontroller

We have chosen the ATSAMD21G18A microcontroller. This microcontroller is used by the Adafruit feather M0 express board which makes integration with Arduino or some other platform a lot easier.

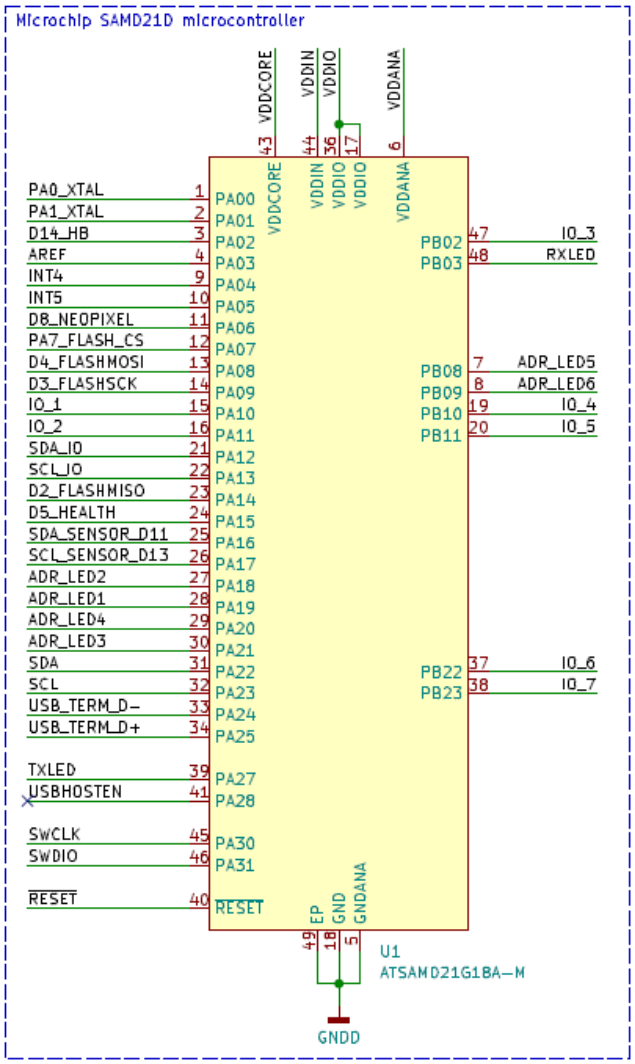
Requirement **T1**

Specification 12.1:

The software for the microcontroller is the combination of the other specified software schematics.

Specification 12.2:

Note: The decoupling capacitors and filtering for the MCU can be found in the *Power* sheet of the schematic.



4.2.4.1 – Clock

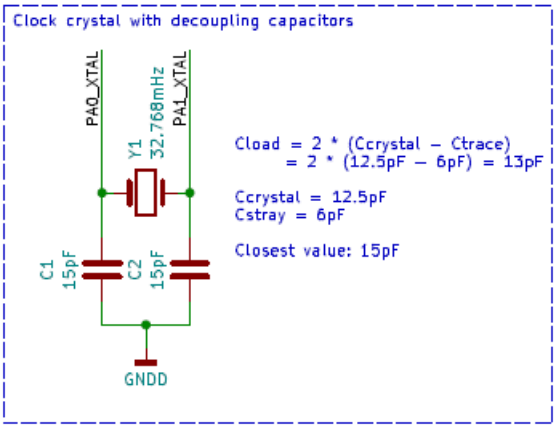
The SAMD21G18A runs on a 32.768mHz clock crystal. C1 and C2 are load capacitors. These are crucial for getting the correct clock speed. If you end up going with a different clock crystal, you will need te recalculate the load capacitors. The formula is in the schematic. But this formula is incorrect. You can verify the clock speed at the test points we have placed on the clock pins.

Requirement **supports T1**

Specification 13.1:

No software scheme for aforementioned element.

Specification 13.2:

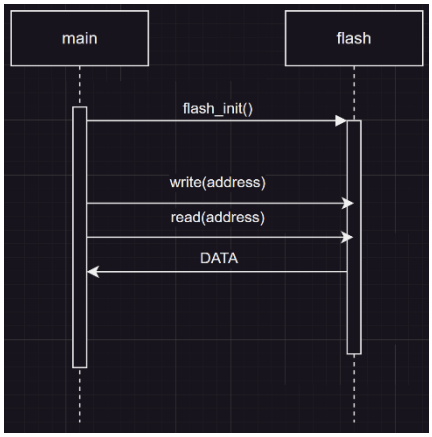


4.2.4.2 – Flash memory

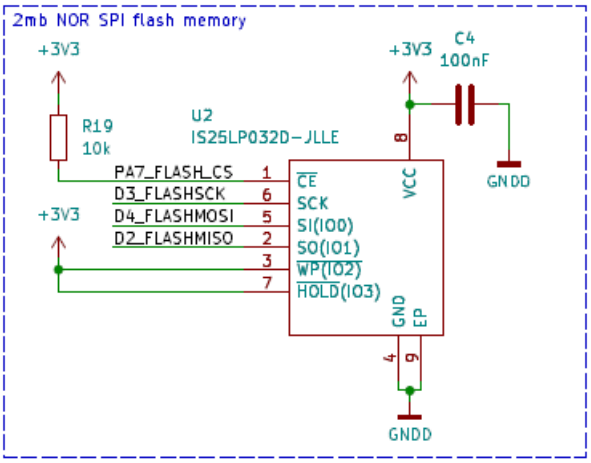
We are using a 32kb flash chip. The chip communicates via SPI. Every communication pin has been connected to a test point making debugging easier.

Requirement **general functionality**

Specification 14.1:



Specification 14.2:



4.2.4.3 – Debug header

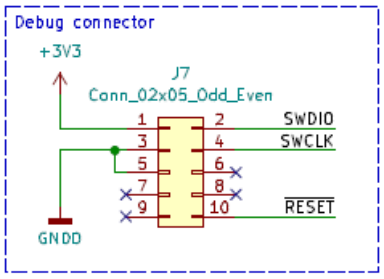
If you want to write software to the chip (e.g. flashing a bootloader) you can use this port with a programmer like the J-LINK mini edu. This port can also be used to debug the processor and peripherals. We have added test points on the reset and data lines just in case.

Requirement **general functionality**

Specification 15.1:

No software functions for debug header.

Specification 15.2:



4.2.4.3 – Reset button

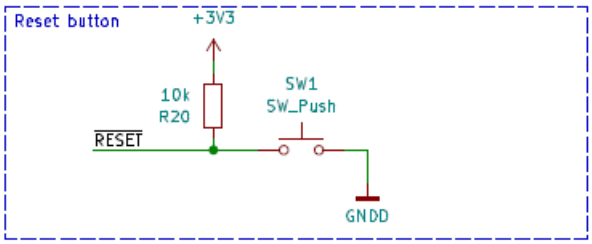
To reset the system, we can use the physical reset button.

Requirement **general functionality**

Specification 16.1:

No software schemes for reset button.

Specification 16.2:

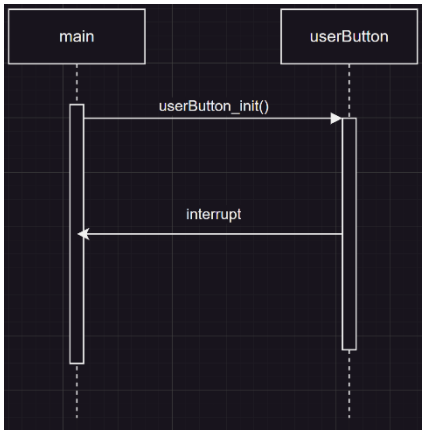


4.2.5 – User button

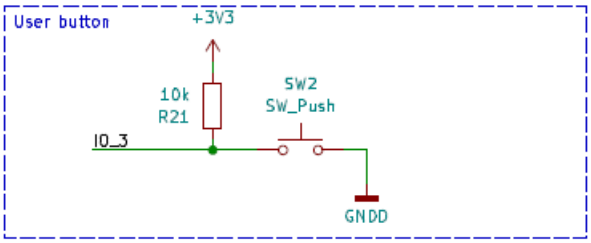
We had one IO left so we routed that to a button. This button also has a test point.

Requirement **general functionality**

Specification 17.1:



Specification 17.2:



4.2.6 – I2C bus components

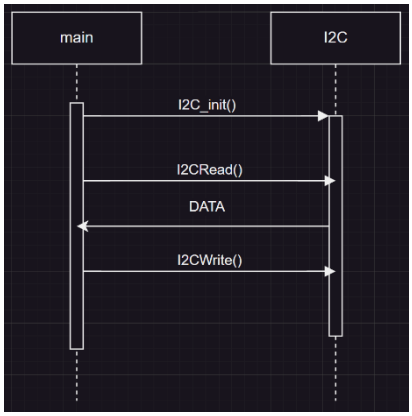
There are three I2C busses available. All I2C busses have test points.

4.2.6.1 – Main bus

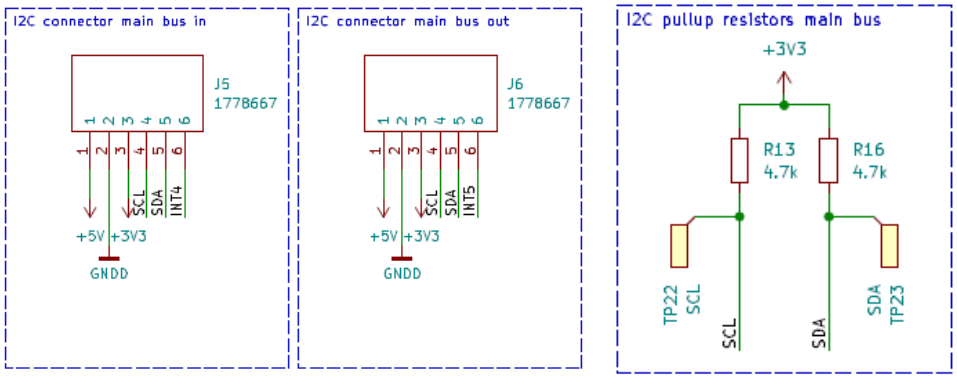
The main bus is the communication bus that communicates with other modules in the baby doll. The main bus has two ports. Main bus in and main bus out. You should connect the output cable of the previous module to the main bus in port on the PCB INT4 is connected to main bus in. The main bus out port connects to the next module. INT5 is connected to this port.

Requirement **T7**

Specification 18.1:



Specification 18.2:



4.2.6.2 – IO bus

We wanted to maintain the possibility to add other I2C enabled devices to the flex PCBs. This is why we have broken out one I2C bus to every FFC/FPC connector. This bus also has test points.

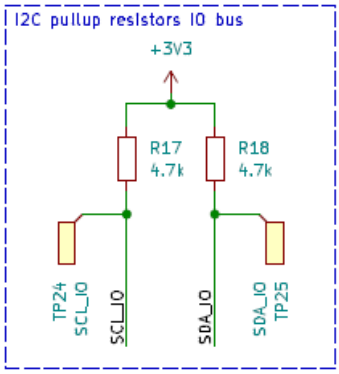
Requirement **supports T6**

Specification 19.1:

See specification 18.1.

Specification 19.2:

Note: I2C output pins can be found in paragraph 4.2.3.2 – FFC/FPC flat cable connectors



4.2.6.3 – Auxiliary bus

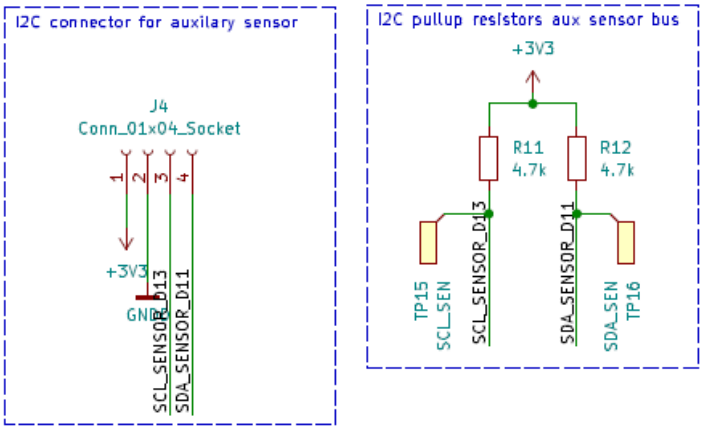
This bus is meant for auxiliary sensors. This bus does not have an interrupt pin connected to it and the output voltage is 3.3V only. We have used a JST-XH connector. We also added more test points tot the data pins.

Requirement **supports T6**

Specification 20.1:

See specification 18.1.

Specification 20.2:



4.2.7 – Mounting holes and fiducials

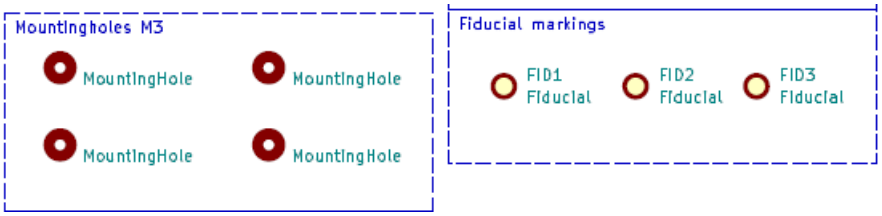
We have added mounting holes to the PCB the hole size is 3.2mm which means a M3 screw can be used to secure the PCB into place. The exposed copper pads near the screw hole are **not** connected to ground. The fiducials are positioning marks for the PCB manufacturer and pick-and-place machine.

Requirement **general functionality**

Specification 21.1:

No software scheme for mounting holes and fiducials.

Specification 21.2:



4.3 – Software

In this paragraph the software structure is described.

4.3.1 – Main structure