Hardware Design

Version: 0x00
Applies to: HW Rev. 0x00

This guide describes design considerations for the BCD-0o27. Note that not all of these were used in the actual design.

Purpose of the document

This document describes the hardware design options and design decisions for the BalcCon Cyberdeck BCD-0o27.

General HW Design Considerations and Decisions

According to the concept document the cyberdeck needs to support the following functionality:

1. Some means of input to solve riddles / control programms stored on the badge
2. A display to display programm / riddle output
3. Storage to store programs and riddles on the badge
4. Some means to blink patterns on leds and some means to record blinking patterns from other badges
5. Run on battery power and alternatively be powered by USB. Optionally support charging of the battery over USB
6. Support wireless communication to connect to mutli user dungeon (mud)
7. Support a means to communicate with a computer using the badge to access the mud

Core Module

The core modules ideally supports wireless communication WiFi and bluetooth out of the box, offers a multitude of serial interfaces and has either built in flash storage or can easily be connected to external storage. Further, the core module should be able to run code for passing through serial to WiFi for MUD access as well as offer the mini games and riddle that run badge local.
Considered options are:

1. Using an ESP32 as the core module.
    - ***Pro***: We know it well and it supports WiFi, Bluetooth and many serial communication protocols. Also the ESP32 offers many GPIO pins. The price for modules is low, sourcing is easy and there is a big community for support.

    - ***Con***: ?
2. Using an STM32 as the core module.
    - ***Pro***: Low power CPU used in many projects.
    - ***Con***: Knowledge about the STM32 is small with the authors. Prices currently raise for this chip due to global shortage. Does not have WiFi or bluetooth built in.

(***DECISION***) We will use the ESP32 as core module for the badge.

Controls

The badge shall support a mode of direct input that allows navigating a menu and playing little games / enter text or codes for applications running localy on the badge. The options for input are:

1. Only push buttons. Due to space constraints, a full push button keyboard does not seem feasible. Therefore a layout of four push buttons to naviagte left / right / up / down makes sense. Further at least two push buttons, one for selecting an input and one for canceling / go back is needed. This layout would also allow for simple games such as pac man, snake or simple jump and runs.
    - ***Pro***: Requires only digital inputs, no analog to digital converter. The buttons can be arranged in a matrix which requires 5 digital input pins. Left / right / up / down can be all pressed at same time, which supports diagonal movement in games.
    - ***Con***: Limits controls in little games as movement if full or none. Pressing buttons for left / right / up / down can be a little fiddly in games. eft / right / up / down can be all pressed at same time, which needs to be checked if it makes sense (eg. up and down simultuously often does not make sense.)
2. Similar to the configuration above, but instead of push buttons for left / right / up / down use a digital mini joystick.
    - ***Pro***: More natural steering in games. Avoids the checks for simultanious button presses as only one direction can be active at a time. Still requires only digital inputs and can be arranged in a matrix (TODO - maybe even more optimized as simultanuous button presses can not happen).
    - ***Con***: Limits controls in games due to digital logic (full or none). Combinations that make sense might not be available as the joystick might not support eg. top + left combinations (TODO - check if there are digital versions that support this. Note: Also NES and old consoles did not support this, so it is possible to design around that.)

3. Have an analog mini joystick (potentially also including a button press if pushed) for movement in menus and games and add two (or potentially more) buttons for select and back.
    - ***Pro***: Natural movement in games using analog signals to capture intensity of the move. Only sensible inputs for navigation (no up and down simultaniously).
    - ***Con***: Requires at least four analog pins hooked up to an analog to digital converter (ADC). ADC2 cannot be used on the ESP32 as we use WiFi. ADC1 can be used but might also be used for other functionality such as LDO and LED. Might require having a separate ADC circuit and using a serial protocol for input to save GPIOs on the ESP32.

(***TODO***) Choose option and derive requirements in the chapter below. Proposal is to go with the simplistic 8 button variant first and only maybe later consider going to a more complex version. Simplicity allows us to use directly 6 digital pins and reduces complexity of the circuit, cost and development time.

Display output

TODO - describe different display options (size / technology)

Balcon Badge Blinky Logic Circuit (B3LC)

The B3LC consists of two parts: A set of six photosensitive elements to measure light and a set six light emitting diodes (LED) to emit light. We start with some considerations for the photosensitive elements.

The photosensitive elements need to capture the light of the LEDs of other badges to decode the blinking patterns those have. There are some options on the photosensitive elements to use:

1. Light dependent resistors (LDR) are resistors whose resistance drops with the amount of light that falls on them. This allows to determine the brightness of the light source.
    - ***Pro***: Simplicity of the circuit, availability and low cost of parts.
    - ***Con***: Each LDR needs to connect to an anaolg pin which means either the LDRs exhaust all available PINs on the ESP32 (ADC1, as AD" canot be used when using WiFi) or the LDRs need to be hooked up using an ADC multiplexer (ADC-MUX) which increases design complexity (but on the other hand allows us to use a more exact ADC than the one of the ESP).

       Stray light is an issue with the LDRs as it influences the read values. This must be designed around by either developing a robust decoding protocol with coarse granularity and fault tolerance and / or design

the BCD-0o27 with some casing elements to snug fit LED on LDR when two badges are combined.

2. Using a digital red, green and blue color light sensor (like for example the ISL29125) allows to directly detect light and its colors and decode it in a digital format.
    - ***Pro*** Allows to use a bus to communicate with the sensor and directly use digital pins. There are sensors with infrared blocking filter which reduces the amount of error. Also, detecting rgb components of light allows for using more complex blinking patterns and introducing better error correction.
    - ***Con*** Cost of parts. Sourcing of parts. Also depending on what bus the parts use to communicate with the ESP32 more or less pins are used. If the part uses I2C it needs to be ensured that 6 different addresses are available and no adress collisions with other parts (eg. maybe display) occure. Parts using I2S buses do not suffer from these conditions but use more pins for chip select.

The LEDs need to emit light in a pattern such that it can be captured by the photosensitive elements and decoded. For this the following options exist:

1. Use a single colour (potnetially different ones for each) LED that either is off or on.
    - ***Pro***: Easy to source (also as SMD), low cost, low complexity. Can be directly connected to 6 digital output pins of the ESP32. Keeps the circuit complexity low and does not need additional parts.
    - ***Con***: Does only allow simplistic light patterns. Also limits the complexity of patterns that can be recognised as codes can only use on / off state and complexity of patterns must be introduced by using longer codes (not necesserily a bad thing). Maybe some poor peoples PWM to have different brightness levels could be implemented in software to allow for interesting patterns (using interrupts for exact timing)?
2. Use single colour LEDs but steer them with pulse width modulation (PWM) to allow for different brightness.
    - ***Pro***: Allows for more complex protocols having multiple states depending on LED brightness (limited by accuracy of decoding). More important, allows for more interesting light patterns (pulsating lights and so on). The ESP32 supports 16 pwm channels grouped in two 8 channel groups.
    - ***Con***: None pwm specific. (?)
3. Use rgb tri-color LED steered with pwm.
    - ***Pro***: Allows for very interesting LED effects and also allows to come up with a sophisticated light coding protocol.

    - ***Con***: Litle more expensive than single color leds, little increased power consumption. Needs either a bus to steer leds or 18 channels. This exhausts the 16 available pwm channels on the ESP32 and also its not sure if enough pins are available (TODO - revisit after deciding on other peripherals) to have eg. 16 pwm channels and 2 digital.

(***TODO***) Add more options (maybe) and decide on what option to implement first. While RGB LED with RGB light sensor is most interesting, it is also the most complex and could be reserved for future badges. Maybe go with simple LDR and pwm controlled leds (single to start, then maybe RGB as for LDR only intensity, not colour matters).

Variants

On the one hand, device cost will be crucial for the BCD-0o27. On the other the plan is that the BCD-0o27 can be used for several years to come (even if there might be new version, we strive for backwards compatability). This means some more robust and thus more expensive design choices might be justified (for example the USB port could be protected against ESD or USB Power Delivery 2.0 could be implemented to ensure the USB port where the BCD-0o27 is connected can source enough power instead of just not functioning porperly when connected).

Depending on the time available, we might offer multiple variants of the BCD-0o27 at different price points:

- IFL variant: The "I feel lucky" variant will be designed with minimum safety and cheapest possible parts.
- TRD variant: "The real deal" variant will implement bells and whistles and use quality, but more expensive chips for some functionality.

Core Module Design

Requirements

- (REQ-COR-001) Run programms and mini games
- (REQ-COR-002) Establish and maintain connection to mud
- (REQ-COR-003) Redirect input from serial line to mud
- (REQ-COR-004) Redirect output from mud to serial line
- ...

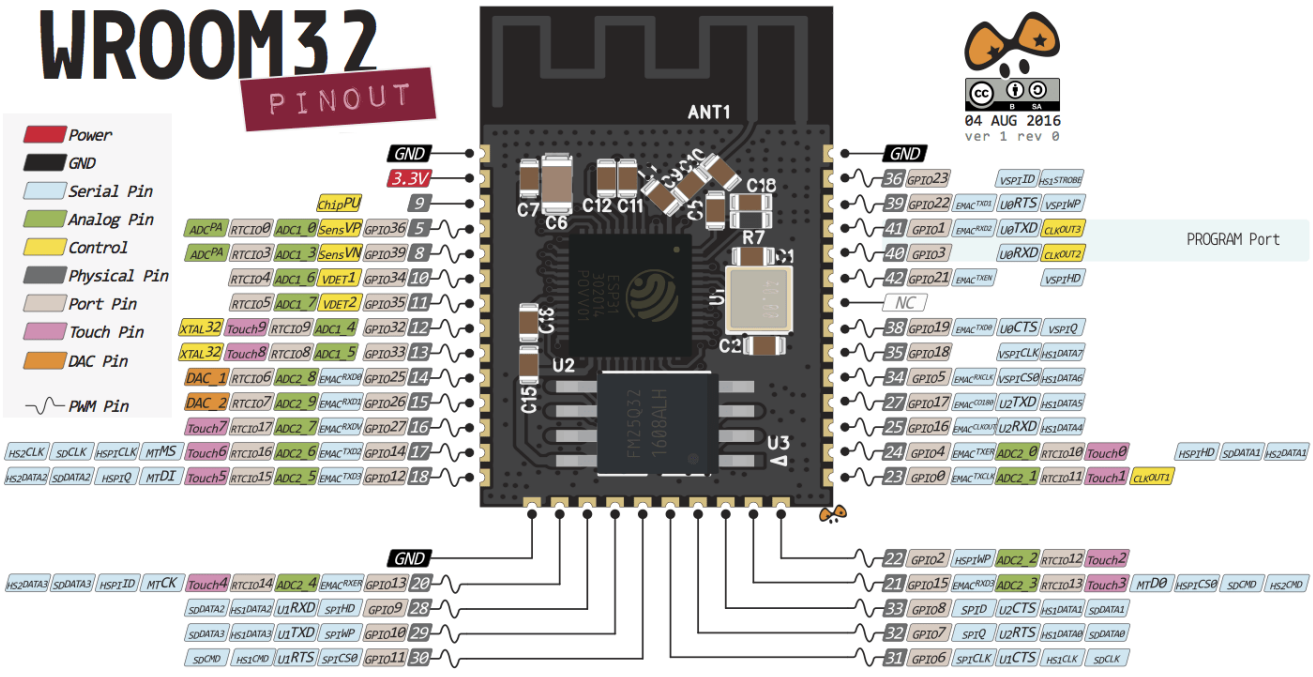
MMU

At the heart of the BCD-0o27 an ESP32 will be used. The ESP32 is a highly integrated MCU that offers WiFi and Bluetooth functionality. At its core is a Xtensa 32 bit microprocessor (single and dual core available). It has rom and

sram memory and supports various clocks and timers. The ESP32 offers advanced interfaces to serials through 34 programmable general purpose input and output pins (GPIOs). It supports SPI, I2S, I2C and UART interfaces. It also offers low power sleep modes which makes it a good choice to run on battery power. Further it has some security mechanisms such as secure boot, flash encryption and hardware acceleration for some common algortihms. Last but not least the ESP32 features a mature development environment, comes with a large community that supports developers and can be acquired cheaply. A complete overview about the ESP32 can be found in the ESP32 Series Datasheet

(***NOTE***) The final decision on what ESP32 module to use exactly still needs to be taken. Experiments will be conducted currently with a ESP32-WROOM-32E module and / or ESP32 DEVKIT (which includes USB power and serial interface). All the Pinouts given in this document refer to the ESP32-WROOM module. Refere to the ESP32-WROOM-32E datasheet
for further information.

The WROOM32 has the following pinout (***WARNING*** physical pin numbers in the image do not match the datasheet. We use the pin numbers from the datasheet. Those start with pin 1 in in the top left corner and then increment by one counterclockwise).



Power Management

The ESP32-WROOM needs the following pins to be connected for power:

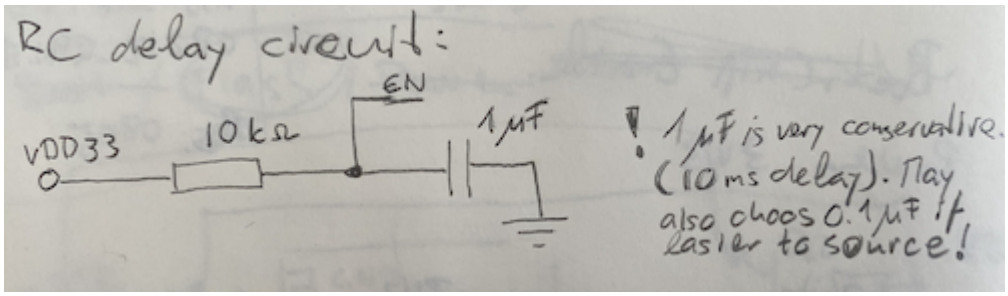
Pin	Name	Function
1	GND	Ground
2	3V3	3.3V power supply (2.3V ~ 3.6V, min 500mA)

Pin	Name	Function
3	EN	Module enable signal (active high)

The power provided to the module must be within 2.3V and 3.6V. The EN pin can be used to deactivate the module completely. However, the design of the BCD-0o27 does not foresee this. To preserve power, the ESP32 will be put into deep sleep, but always be active. Therefore the EN pin will also be connected to the 3.3V rail of the power supply. To ensure the power supply to the ESP32 chip during power-up, it is advised to add an RC delay circuit at the EN pin. The recommended setting for the RC delay circuit is usually R = 10kΩ and C = 1µF.

(***DECISION***) The EN pin will be connected to a switch that is connected to ground. This switch then operates as a reset switch to reset the ESP32.

(***DECISION***) An RC delay circuit with R = 10kΩ and C = 1µF will be added at the EN pin.



(***DECISION***) A 0.1µF and a 22µF capacitor in parallel will be added close to the 3V3 pin to filter ripples in the power supply rail.

GPIO

The ESP32 features 34 GPIOs. Each GPIO has a mximum current draw rating of 40mV. Due to the usage of the internal flash memory and the wireless some GPIOs will not be available. The following GPIOs are reserved and canot be used:

PIN	GPIO	Reserved for
17	9	Connected to integrated SPI flash
18	10	Connected to integrated SPI flash
19	11	Connected to integrated SPI flash
20	6	Connected to integrated SPI flash
21	7	Connected to integrated SPI flash
22	8	Connected to integrated SPI flash

The remaining GPIOs listed in the table below are available for use in the BCD-0o27. However, the use of some requires to take special care as they fulfil important functions:

PIN	GPIO	Type	Limitation	Used for
13	I / O	-	unused	
23	15	I / O	Strapping pin, pwm at boot	SPI Bus Chip Select (Display)
24	2	I / O	Strapping pin	SPI Bus Data Command (Display)
25	0	I / O	Strapping pin	unused
26	4	I / O	-	SPI Bus Reset (Display)
27	16	I / O	-	unused
28	17	I / O	-	unused
29	5	I / O	Strapping pin, pwm at boot	unused
30	18	I / O	-	SPI Bus Clock (Display)
31	19	I / O	-	SPI Bus MISO (None)
33	21	I / O	-	Add on connector (I2C) Interface (SDA)
34	3	I / O	RX, high at boot	unused
35	1	I / O	TX, debug output at boot	unused
36	22	I / O	-	Add on connector (I2C) Interface (SCL)
37	23	I / O	-	SPI Bus MOSI (Display)

Strapping pins are sampled during the sample the voltage level during the chip's system reset relase (power-on-reset, brown out reset, watchdog reset) as strapping bits of "0" and "1". Those strapping bits determine the devices boot mode, the operating volatage of VDD_SDIO and other system settings. See chapter Boot Up.

Boot Up

In order to boot the ESP32 up correctly the strapping pins must be set as follows:

Strapping PIN	Set	Effect	Mapping WR32E
MTDI	0	Set voltage to 3.3V	IO12 / pin 14

Strapping PIN	Set	Effect	Mapping WR32E
GPIO0	1/0	1 on normal boot, 0 to flash new code	IO0 / pin 25
GPIO2	?/0	Must be 0 when flashing, don't care otherwise	IO2 / pin 24
MTD0	0	Debug output for boot (set 1 for development)	IO15 / pin 23
GPIO5	1	Rising edge sampling and output	IO5 / pin 29

Note that the strapping pins must be set during the system reset release (boot phase) of the chip. The system reset release can be triggered by three conditions:

1. Startup of the chip, power on reset (EN pin goes high and system is powered)
2. RTC watchdog reset (TODO - excat conditions? Every wakeup?)
3. Brownout reset (voltage was too low or unstable to continue operation)

After system reset release (when the ESP32 enters normal operation) the pins are no longer relevant can be used as regular gpios.

It follows an explanation on the choice made above:

- MTDI: MUST always be set to 0, as the module only operates on 3.3V.
- GPIO0: In normal operation must be set to 1 to boot normally. When flashing new code on the chip must be set to 0.

  OPEN DECISION: Shall we allow flashing new code after development?

  ```
    Pro: Could release new updates to the batch easily. Also we could
  either
    design it in a way where it is easily trigerable (using a switch)
    encouraging tinkering with the badge or we could make it more
  difficult by
    bringing out a trace that can then be hooked up to a flasher that
  pulls the
    pin low making this more difficult, disencouraging fundamentally
  changing
    the badge.

    Con: Reflashing something completely new could render the badge to
  be
    used in ways not foreseen and make devleoping a interesting game
    experience more difficult (more fault cases). In this case the HW
    design must be such that it is really difficult to set the pin to
  low
    at bootup.
  ```

```
  Decision: TODO
```

- GPIO2: When flashing (GPIO0 set to 0) must be set to 0. Otherwise does not matter.

  OPEN DECISION: Set to 0 during boot to be on the safe side when reflashing or don't care?

  ```
    Pro: Setting to 0 at boot rules out one source of failure for
  flashing
    new code.

    Con: Depending on what is connected to the GPIO2 it could make the
    circuit and software design more difficult to ensure that after boot
    the value needed by the connected chip or device is there.

    Decision: TODO
  ```

- MTD0: Sets whether debug output is given (on TXD0 / GPIO 1, which is pin 35 on the ESP32-WROOM32E module). Can be set to 1 for development but is disencouraged for use in release. Therefore must be set to 0 in release.
- GPIO5: If MTD0 is set to 0 does not matter. However, if MTD1 is set to 1 this pin defines the timing of the SDIO slave together with MTD0. TODO - decide what to use. Currently chosen 1 randomly.

Entering Flash Mode

If we decide to allow for easy flashing, we will design two ways to flahs the ESP32: Auto (if flasher supports it) and manual by button press. When using a flasher (like esptool.py for example),
some flashers will use DTR and RTS to signal that they want to flash. Thus those can drive transistors to switch the boot mode. Also, as some tools do not support this a button to be manually pressed will be added. This button needs to be debounced using a capacitor.

Blinky Interface

TODO

Display

TODO

Controls

The BCD-0o27 will feature a set of inputs:

For the joystick we will use a simple analogue

Circuit Design

We design the input of the buttons using a shift register. This allows to record multiple buttons presses while only needing two pins on the ESP32 (serial in and clock).

D-Pad Design

TODO

Button Design

TODO

Serial Interface

TODO

(OPIONAL) Debug Interface

TODO

Power Supply

(***WARNING***) Currently this chapter is very rough as components are not yet decided. It must be adapted during the process. It is important to do this as changing power requirements can rule out some options. We likely aim to high at the moment, but this is preferable to too low.

The BCD-0o27 must be mainly powered by batteries to allow it to operate on its own. When hooked up to an alternate power source this source shall be used to power the badge. Charging the battery is optional.

Requirements

The power supply design for the BCD-0o27 must fulfil the following requirements:

- (REQ-POW-001) Provide enough output current to power the BCD-0o27. For the exact amount check calculation in

- (REQ-POW-002) Provide a 3.3V rail (min. 2.3V / max. 3.3V - may change depending on additional parts used as they might have different power requirements)
- (REQ-POW-003) Switch from battery power to external power source if attached
- (REQ-POW-004) Batteries shall be easy to source and replace
- (REQ-POW-005) Less hazardous battery technology shall be prefered
- (REQ-POW-006) Protect from reverse polarity

Output current

The minimal amount of current the power supply must deliver is estimated by the sum of the current need of all components at typical full operation (ie. for ESP32 this means wireless enabled and computing) and then add 10% as safety margin.

NOTE: This current estimate is subject to change due to component selection for the not yet selected components. Update accordingly.

```
ESP32            500 mA (1)
6xLED            360 mA (2)
LCD               60 mA (3)
3xShift reg.       3 mA (4)
-----------------------
Total            923 mA


(1) Typical operating current. Cummulative IO output current is 1200 mA.
(2) If using WS
(3) Measured ILI operated at 3.3V
(4) To be confirmed
```

Battery Options

The following options are considered for the BCD-0o27:

- NiMH AA batteries
- Alkaline AA batteries
- LiFePo AA batteries
- LiFePo 18650 batteries
- Li-Ion 18650 batteries
- Li-Ion accu 3.7V (single cell)

The following table summarises properties of the different battery types. Note that depending on the manufacturer these values might differ slightly, however

this will not affect the design.

Battery	Voltage nom/max/min	Capacity mAh	Max. discharge
NiMH AA	1.2 / 1.5 / 1.2	600–2750	3C
Alkaline	1.5 / 1.5 / 1.2	1800–2850	0.5A - 0.75A
LiFePo AA	3.2 / 3.65 / 2.0	600	3C (1800mA)
LiFePo 18650	3.2 / 3.65 / 2.0	1400	3C (4500mA)
Li-Ion 18650	3.6 / 4.2 / 2.5	2500	8C (20A)
LiPo pack 1S	3.7 / 4.2 / 2.75	2000	1C (2000mA)

NiMH and Alkaline AA Batteries

NiMH and alkaline AA batteries have the big advantage that they can be found anywhere and their price is low. Alkaline batteries however only provide a continuous discharge current of 0.5A - 0.75A. While this should be enough for regular operation, it does not provide many reserves and alkaline batteries in general operate better at low discharge currents.

Both, NiMH and alkaline AA batteries can directly supply enough voltage for the ESP32 without any voltage conversion. This reduces the complexity of the power supply design. The same should go for the OLED display, however, this must be verified (TODO). The charging circuit for rechargeable NiMH AA batteries is a little more complex, as two cells should be balanced to allow them both to fully charge. Also, if charging starts automatically when plugged in to an alternate power supply, it might be that non rechargeable batteries are in use and the cahrging circuit canot detect this. Therefore the easiest design choice would be to not implement a chargin circuit and let the user change batteries when running low on power.

Pro	Con
Battery safety	Charging circuit
Easy sourcing	Voltage below 3.3V (3V - 2.4V)
Low cost	Alkaline: low current < 750mA
No need for buck / boost converter	

TODO - charging circuit design (OPTIONAL)

LiPo Pack 1S

A single cell LiPo battery pack has the advantage that a charging circuit is not too complex as no cell balancing is needed. The single cell LiPo battery

also is rather small and will only slightly increase badge volume and weight. It offers decent capacity and enough continuous discharge current.

Safety is a concern with LiPo batteries as over charging them may lead to fire and over discharging them will permanently damage the battery. A wide range of ready made ICs for battery management exist and usually only require a couple capacitors and resistors asides the IC itself. Those battery management ICs are needed for charging and discharging. They will take care of over charge and over discharge protection.

As a fully charged LiPo has a voltage of 4.2 V, this must be stepped down to not damage the 3.3V logic components. The LiPo battery can be discharged to 2.5V although not discharging below 3V is recommended for battery longlifety. Also, the residual usable charge between 2.5V and 3.3V is around 3% of the total usable charge. In order to provide power to the ESP32 and the other components a 3.3V low dropout regulator (LDO) could be used. However, this would not allow us to use the full capacity range. Another option would be to use a LDO that outputs 3V or even 2.5V, which would be enough for the ESP32. However, as with NiMH AA batteries this would need to be verified for the display and the other components. Alternatively a buck-boost converter could be used to use the full range while providing a stable 3.3V power rail. The buck-boost converter on the other hand generates an electromagnetic field that could influence wireless quality if placed to closs to the ESP32.

Pro	**Con**
Can provide stable 3.3V power rail	Battery safety
Low cost	Needs battery management
High current (1C)	Sourcing need lead time
Reasonably simple charging circuit	
Low weight and volume	

(***DECISION***) Even though many LiPo batteries come with a built in protection circuit, we will still design the protection circuit in the BCD-0o27 to prevent bad things from happening if someone uses a battery without a protection circuit.

Li-Ion 18650 battery

For the LiPo 18650 battery the same applies as for the Li-Ion pack 1S with the only difference that the battery uses more volume but provides roughly 25% more charge and supports a higher average discharge current.

Pro	**Con**

Pro	Con
Can provide stable 3.3V power rail	Battery safety
Low cost	Needs battery management
High current (8C)	Sourcing need lead time
Reasonably simple charging circuit	Larger volume
Low weight	

For charging a single cell Li-Ion battery the
TP4056
standalone linear Li-Ion battery charger with termal regulation in SOP-8 package is frequently used. It charges the battery with a maximum of 1A directly from the USB port (5V supply) but can also be used with 6V solar panels. The chip features over voltage protection. It charges the battery to 4.2V.

For protecting the the battery from over discharge, a
DW01 in
combination with a 8205 dual mosfet can be used.

(***DECISION***) If we decide to use multiple Li-Ion 18650 batteries in parallel to increase total capacity we will add fuses between the cells to prevent fire hazard due to over discharge induced by a defective Li-Ion cell (short cell). The fuse disconnects the faulty cell.

LiFePo 18650

The main advantages of LiFePo over LiPo are that they are safer to use as they will not burst in fire if overcharged and they also have a maximum voltage of 3.65V which is just at the top limit of the ESP32. Therefore no LDO or buck-boost converter is needed.

TODO - check charge circuit and low power cut off necessity.

LiFePo AA

TODO

USB Power

The BCD-0o27 will support powering from the USB-C port (see
AN1953 Introduction to USB Type-C)
for a good overview). Therefore the port on the BCD-0o27 is configured as an upstream facing port (UFP) and using pull down resistors on CC1 and CC2 to announce supported power requirements. We will only support 5V power modes. To

indicate voltage and max current, a Rp pull-down resistor must be used on both CC1 and CC1. As the USB Type-C shall support charging at 1.5A@5V (at minimum) a 5.1kΩ±10% must be used. (Note: This results in a voltage divider when a cable is attached. Depending on the resistor Ra used in the downstream facing port The supported power mode is indicated. O

Note: The downstream facing port decides on the power capability it has to offer. The minimum is 500mA / 900mA (USB2 standard). If the BCD-0o27 needs more than 500mA we might run in problems when attached to a host that only supplies 500mA. To avoid this we should implement the power delivery protocol to indicate a minimal current requirement. However, we might also just ignore this for simplicity and tell users. To be decided once total power requirements are known.

When plugged in the USB power shall be used to power the badge to safe battery power. If we decide to use LiPo or LiFePo4 batteries, a charging circuit must also be powered from the USB port. If we decide upon NiMH batteries charging will not be supported as we want to avoid users using non-rechargeable batteries that are then charged inside the device. As chargers are cheap and everyone has them this is an acceptable design choice that also reduces complexity and cost.

Last but not least, we might add ESD protection. ESD protection vill prevent spikes due to ESD induced by humans plugging in the USB cable. Basically we have two options:

1. USBLC2-S2C6
   is designed for USB 2 applications. As we use a USB Type C plug with USB 2 protocoll, this is a feasible solution.
2. TPD4E02B04
   is a 4-Channel ESD Protection Diode for USB Type-C and HDMI 2.0 and is compatible to USB 3.1 GEN2, should we want to use those features. (unclear, but might even be needed if we want to use power delivery (TBC))

Decision: For the moment we will go with the USBLC2-S2C6 as we only use USB 2.0 features.

Power Rails

The operation of the ESP32 requires a 3.3V power rail. Further, the BCD-0o27 will be designed around 3.3V as much as possible to avoid needing another power rail and therefore limit the use of DC-DC converters (drives down cost and reduces wasted power).
TODO - 5V?

3.3V Power Rail

Multiple options, depending on the used power source exist. In order to output 3.3V either a low dropout regulator (ldo), a switching buck converter / buck-boost converter can be used.

Low dropout regulators are easy to use, but they have worse efficiency than switching dc-dc converters. Also they limit battery choices as the voltage of the battery that can be used must be higher than 3.3V+Vdo (droput voltage). On the upside, ldos do not generate electromagnetic waves and thus do not interfere with WiFi, bluetooth and other sensitive components.

Switching dc-dc converters on the other side

Reverse Polarity Proteciton

The battery could be inserted in the wrong way (not the USB) and therefore the power path from the battery must be protected agains reverse polarity. This can be done with a P-channel mosfet with the gate clamped to ground. Using a mosfet has the advantage over using a diode that it wastes less power. A diode wastes forward voltage times the current (Vf * I) power. A possible mosfet is AO3401A
Note that the AO3401A requires a minimum of 2.5V for its operation. But this is fine as we can live with cutting of at that value independent of the battery and boost converter we use. The maximum gate threshold voltage for the AO3401A is -1.3V. The minimal battery voltage as stated above will be 2.5V. The parasitic body diode of the AO3401A has a forward voltage Vf_max of 1.0V. Thus the gate-source voltage Vgs is at least 0V - (2.5V - 1V) = -1.5V which is larger than the required -1.3V threshold voltage and thus the mosfet will turn on. Once the mosfet has turned on, the resistance between drain and source will drop to max 85mOhm.