

PASCO Florian

Hi there! 

 I'm Florian Pasco, a **Master's student in Engineering** specializing in **Electronics, Computer Science, and Mechatronics**.

 I am passionate about creating and sharing **open-source projects** that are reusable and easy to understand, helping others grow in the fields of **electronics, programming, and mechanics**.

 Recently, I developed a **full-stack project** using **React, Next.js, Django, Python, and PostgreSQL**, showcasing my skills in modern web and backend development.

 Beyond technology, I enjoy **self-training**, exploring **personal development**, and learning from resources like the **French Polar Institute**.

 I love traveling and discovering new places—whether it's **hiking, biking**, or going on **road trips in my car!**

 I've also applied my skills in **reverse engineering**, successfully diagnosing and repairing **20+ electronic units**. Additionally, I designed and worked on a **thermal resistance control board** with feedback from a thermal resistance sensor for precise regulation.

BatteryLevelIndicator



🔋 Battery status with LED display!

The **Battery Level Indicator** is an open-source project designed to visually display the charge level of a battery using **five green LEDs** and a **sixth red LED** for critical low levels. The board also features an **ON/OFF button** that activates the system, illuminating **two white LEDs** when turned on and **two blue LEDs** at other times.

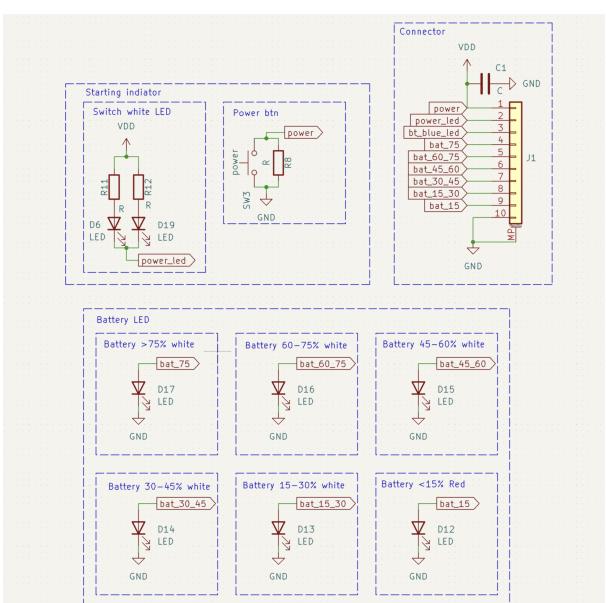
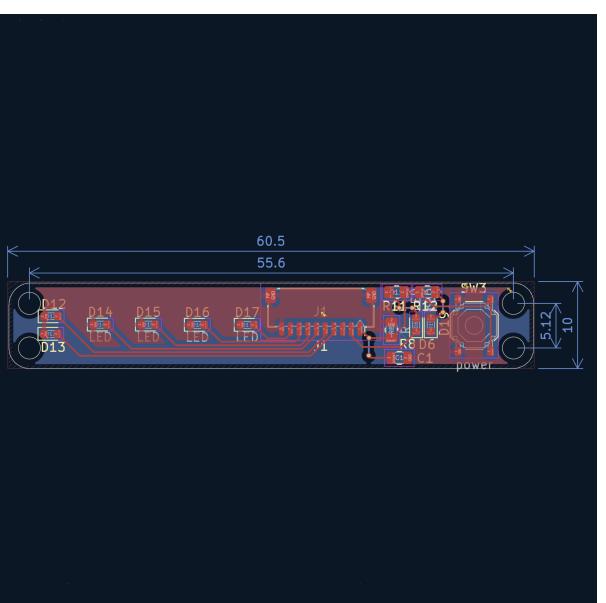
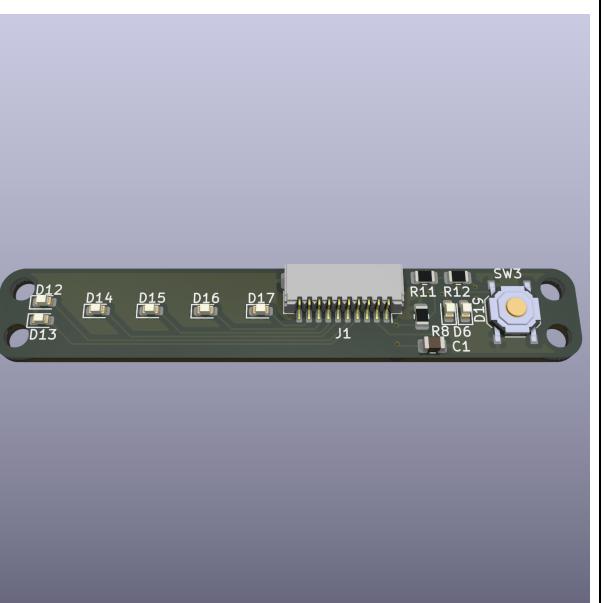
🎯 Purpose

- 🔋 **Battery Monitoring:** Provides an intuitive LED-based display of battery charge levels.
- 🔴 **Power Control:** Includes an ON/OFF switch for system activation.
- 🟡 **User-Friendly:** Simple and effective visual indicators for battery status.
- 🛠️ **Open-source & Customizable:** Modify and adapt for various battery-powered projects.

📝 Features

🏷️ Feature	🔍 Description
🔋 Battery Level Indication	5 green LEDs indicate battery charge level
⚠️ Low Battery Warning	1 red LED lights up when the battery is critically low
🔴 Power Switch	ON/OFF button for system control
💡 ON Indicator	2 white LEDs light up when the system is ON
🔵 Standby Indicator	2 blue LEDs light up in other situations
💻 PCB Design	Open-source & customizable
🌐 Use Cases	Battery-powered devices, embedded systems, and monitoring applications

📐 PCB Design Preview

📋 Schematic	💻 PCB Layout	🏗 3D
		

BluetoothSpeakerKeyboard



Minimalist PCB for speaker control

Bluetooth Speaker Keyboard is a minimalist open-source PCB designed to provide simple control functionalities for Bluetooth speakers. The board features essential buttons for **volume control**, **Bluetooth pairing**, and **play/pause functionality**. It follows the **Adafruit-compatible footprint**, making it easy to integrate into existing projects.

Purpose

- **Seamless audio control:** Easily manage volume, pairing, and playback.
- **Compact and efficient:** Designed for integration with Bluetooth speaker systems.
- **Open-source and customizable:** Modify the design to fit your specific needs.
- **Adafruit-compatible:** Fits within Adafruit's standard PCB footprint for easy use in DIY projects.

Features

💡 Feature	🔍 Description
🔊 Volume Control	Adjust the speaker's volume up and down
⏯ Play/Pause Button	Start or stop the music playback
🌐 Bluetooth Pairing Button	Activate pairing mode for easy device connection
🔧 Minimalist Design	Only essential components for simplicity
💻 PCB Design	Open-source & customizable
🌐 Use Cases	Embedded in DIY Bluetooth speakers, home automation systems
🔌 Adafruit-Compatible	Follows Adafruit's standard PCB footprint

PCB Design Preview

Schematic	PCB Layout	3D

Esp32InterfacePcb



🔌 Interface ESP32 modulaire

Esp32InterfacePCB is an open-source PCB designed to provide a reliable interface between an **Espressif ESP32-WROOM-32E** module and various external modules using **JST-SM & JST-SH connectors**. This PCB simplifies ESP32 integration into embedded, IoT, and home automation projects.

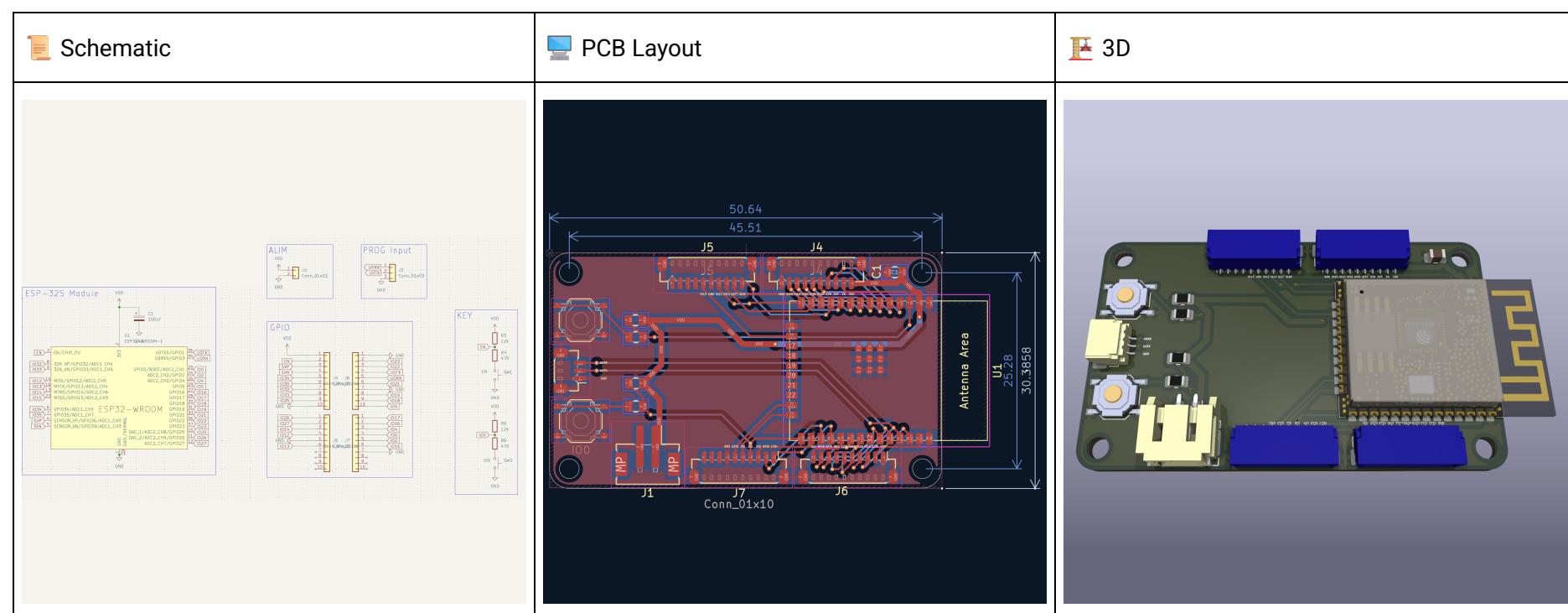
🎯 Purpose

- 🔌 **Modular Interface:** Facilitates the connection between an ESP32-WROOM-32E and other peripherals.
- 📏 **Compact & Optimized Design:** Suitable for embedded project constraints.
- 🛠️ **Open-source & Customizable:** Modify and adapt the design to fit your specific needs.

📝 Features

💡 Feature	🔍 Description
🔌 ESP32 Interface	Compatible with the ESP32-WROOM-32E module
📡 Connectors	JST-SM & JST-SH for easy connections
⚡ Power Supply	Supports 3.3V
кнопкі GPIO Access	Access to key GPIO pins for seamless integration
Capacitors	Decoupling capacitors for signal stability
💻 PCB Design	Open-source & customizable
🌐 Use Cases	IoT, embedded systems, automation, robotics, rapid prototyping

📐 PCB Design Preview



HV2LV-PowerJST



Regulator 4.8V-15V -> 3.3V

HV2LV-PowerJST is an open-source PCB designed to step down **4.8V - 15V** to a stable **3.3V** output using an **AMS1117 voltage regulator**. The board features **JST-PH connectors** for easy integration into embedded projects, IoT devices, and prototyping setups.

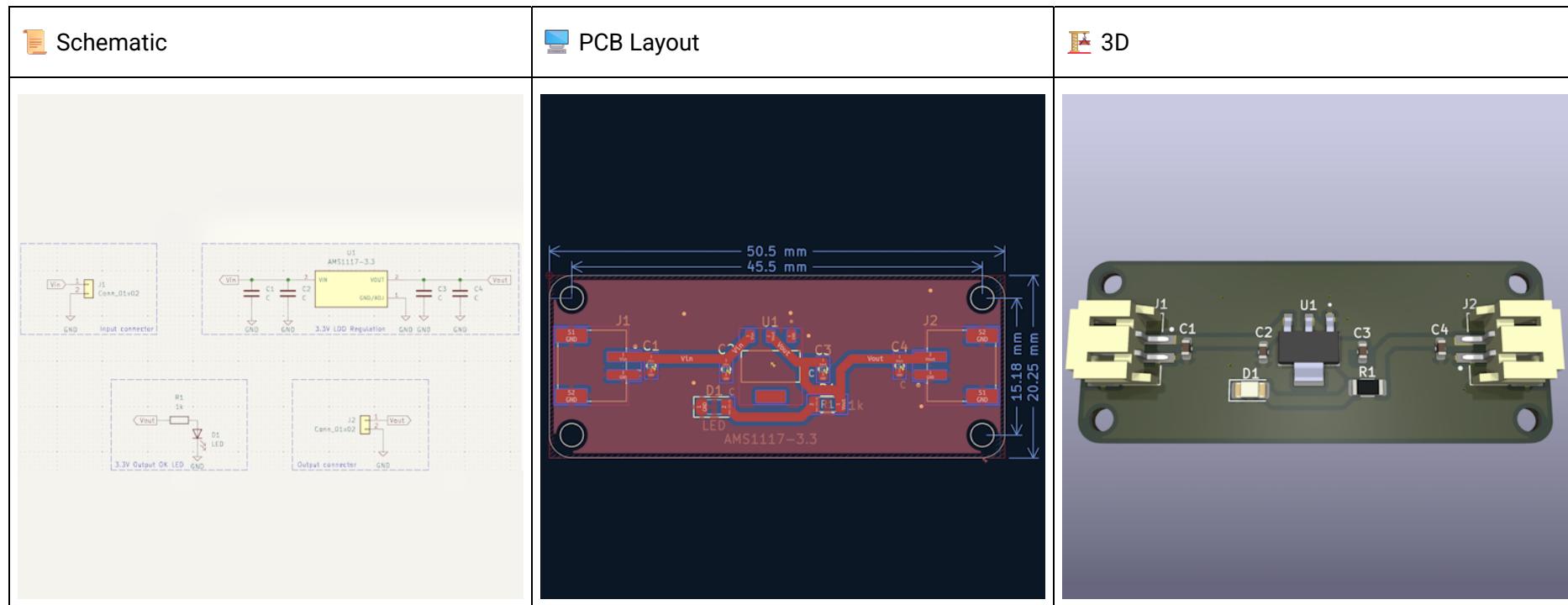
Purpose

- 🔌 **Efficient power conversion:** Converts input voltages from **4.8V to 15V** down to a fixed **3.3V**.
- 📏 **Compact and Adafruit-compatible:** Designed to fit within Adafruit's standard PCB footprint.
- 🛠️ **Open-source and customizable:** Modify and adapt the design to suit your specific needs.

Features

💡 Feature	🔍 Description
⚡ Input Voltage	4.8V - 15V
🔋 Output Voltage	3.3V (fixed)
🛠️ Regulator	AMS1117-3.3
🔌 Connector 1	JST-PH (Input Voltage)
🔌 Connector 2	JST-PH (3.3V Output)
Capacitors	4 decoupling capacitors for stability
💻 PCB Design	Open-source & customizable
🌐 Use Cases	Powering 3.3V embedded systems, IoT devices, and sensors

PCB Design Preview



HeaterControl-Shield



🔥 STM32 Shield for Heat Control

HeaterControl-Shield is an open-source electronic board designed to control a **heating resistor**, measure the **consumed current**, and detect the **temperature** near the heating resistor. This board is shaped as an **Arduino shield** and optimized for use with an **STM32 Nucleo**.

🎯 Main Features

- 🔥 Heating resistor control via an **N-channel MOSFET**.
- ⚡ Current consumption measurement using a **shunt resistor** and an **operational amplifier**.
- 🌡️ Temperature sensing using a **thermistor** integrated into a **Wheatstone bridge**, amplified by an **instrumentation amplifier**.
- 🛠️ Open-source and customizable design to adapt to specific project needs.

📝 Technical Specifications

🏷️ Feature	🔍 Description
🔌 Power Input	5V - 12V
🔥 Heating Control	N-channel MOSFET for power control
⚡ Current Measurement	Shunt resistor + Operational amplifier
🌡️ Temperature Sensing	Wheatstone bridge + Instrumentation amplifier
🌐 Interface	Compatible with Arduino Shield and STM32 Nucleo
💻 PCB Design	Open-source & customizable
🌐 Applications	Thermal control projects, embedded systems, temperature regulation

📐 PCB Preview

📋 Schematic	💻 PCB Layout	🏗️ 3D View

🔗 Main Connections

Pin	Function
VIN	Main power input
GND	Ground
HEAT_CTRL	PWM signal to activate heating
CURR_SENSE	Amplified output of current measurement
TEMP_SENSE	Amplified output of temperature measurement

MicroUSB2JST



🔌 MicroUSB -> JST-SH & JST-PH

MicroUSB2JST is an open-source PCB that acts as a bridge between a Micro USB port and JST connectors (JST-SH and JST-PH). This module is designed to simplify connections for embedded projects, prototyping, and power distribution.

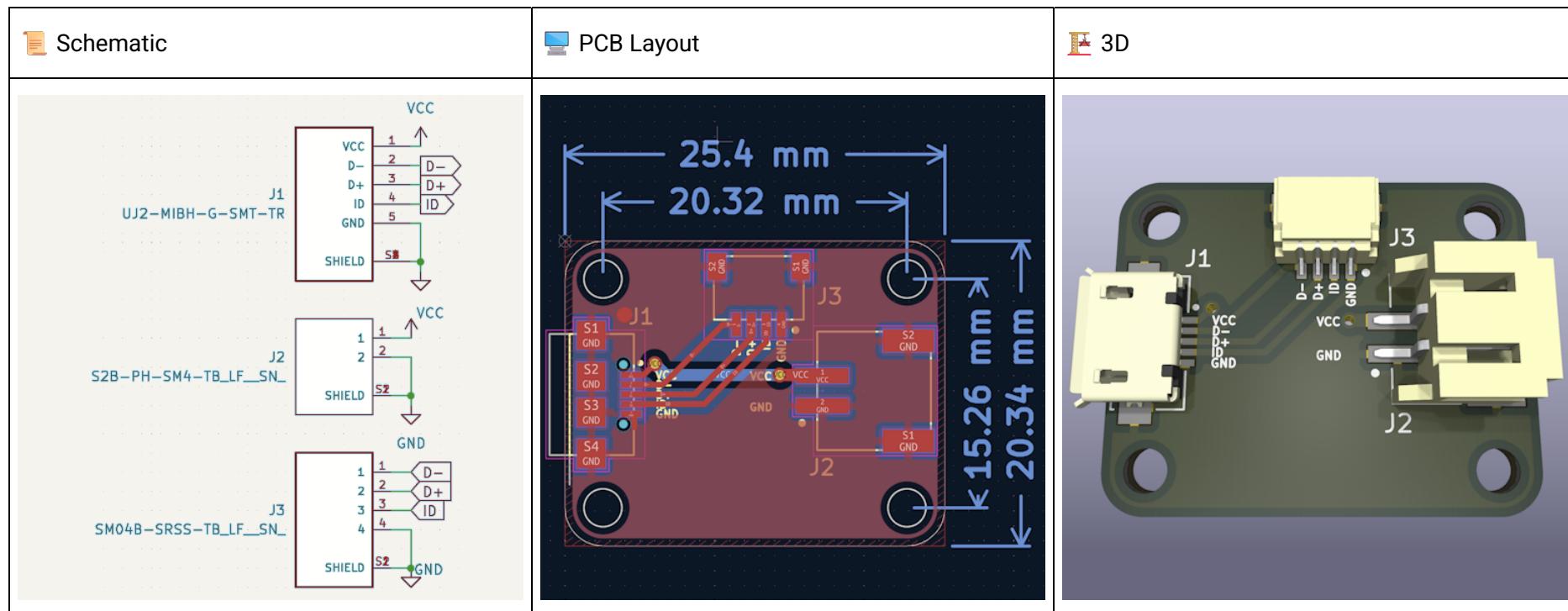
🎯 Purpose

- 🔌 **Convenient power and data interface:** Easily connect USB power or data lines to JST-equipped devices.
- 📏 **Compact and Adafruit-compatible:** Designed to fit within Adafruit's standard PCB footprint.
- 🛠️ **Open-source and customizable:** Modify and adapt the design to suit your specific needs.

📝 Features

💡 Feature	🔍 Description
🔌 Connector 1	Micro USB (⚡ power & ⚡ data)
🔌 Connector 2	JST-PH (⚡ higher current capacity)
🔌 Connector 3	JST-SH (📏 small form factor)
💻 PCB Design	🆓 Open-source & 🖌️ customizable
🎯 Use Cases	⚡ Power distribution, 📈 sensor connections, 🤖 embedded systems

📐 PCB Design Preview



UsbUartBridge



USB-to-UART bridge for embedded systems

UsbUartBridge is an open-source PCB designed to provide a **USB-to-UART bridge** for seamless serial communication between a computer and microcontrollers. The board features **JST-SH & JST-PH connectors** for easy integration into embedded projects, IoT devices, and prototyping setups.

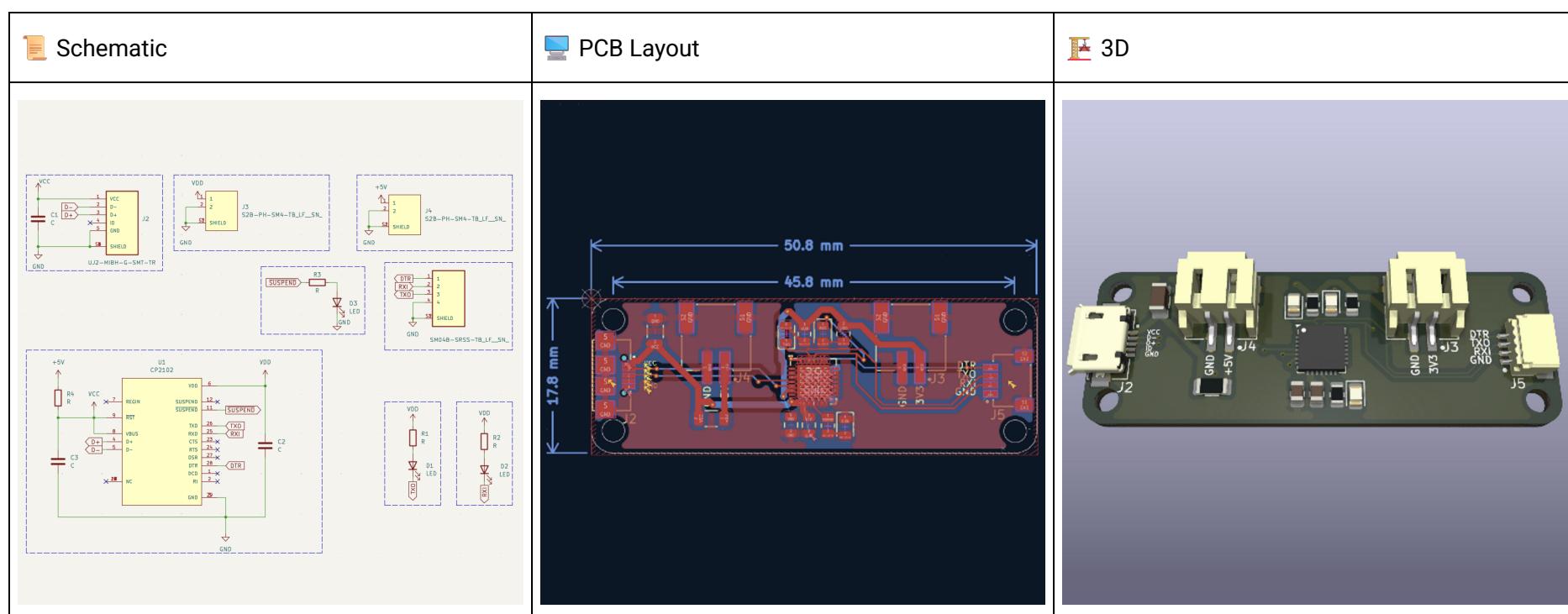
Purpose

- 🔌 **Reliable USB-to-Serial Conversion:** Enables communication between a PC and embedded systems via UART.
- 📏 **Compact and Adafruit-compatible:** Designed to fit within Adafruit's standard PCB footprint.
- 🔧 **Open-source and customizable:** Modify and adapt the design to suit your specific needs.

Features

💡 Feature	🔍 Description
🔌 USB Interface	Micro-USB (for PC connection)
📡 UART Interface	TX, RX, DTR
⚡ Voltage Options	3.3V & 5V selectable output
🔗 Connectivity	JST-SH & JST-PH connectors for easy integration
⚡ Capacitors	Decoupling capacitors for stability
💻 PCB Design	Open-source & customizable
🌐 Use Cases	Debugging & Programming microcontrollers, IoT device communication, serial data transfer

PCB Design Preview



StateMachineSafe



🔒 Safe control via state machine!

StateMachineSafe is a project aimed at designing and simulating a state machine to manage the opening and closing of a safe. This machine is built using D flip-flops and logic gates, ensuring precise logical control of the mechanism.

Purpose

- 📋 **State Machine Design:** Development of a state graph defining the safe's behavior.
- 📋 **Transition Table and Karnaugh Map:** Derivation of logical equations necessary for system operation.
- ⚡ **LTSimulation:** Verification and validation of the logical circuit through LTSpice simulation.
- 🔑 **Secure Access Management:** Implementation of a reliable mechanism ensuring the safe's opening and closing according to a defined sequence.

Features

🏷️ Feature	🔍 Description
🔄 State Machine	Modeling of the control system with a state graph
📊 Transition Table	Definition of transitions between states
📝 Karnaugh Map	Simplification of logical equations
🔧 Hardware Implementation	Use of D flip-flops and logic gates
💻 LTSimulation	Verification of behavior through simulation

System Architecture

📅 State Graph	📋 Transition Table	⚡ Logic Circuit																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																																										
	<table border="1"><thead><tr><th>Eid</th><th>bit 0</th><th>bit 1</th><th>bit 2</th><th>bit 3</th><th>bit 4</th><th>bit 5</th><th>bit 6</th><th>bit 7</th><th>bit 8</th><th>bit 9</th><th>bit 10</th><th>bit 11</th><th>bit 12</th><th>bit 13</th><th>bit 14</th><th>bit 15</th><th>bit 16</th><th>bit 17</th><th>bit 18</th><th>bit 19</th><th>bit 20</th><th>bit 21</th><th>bit 22</th><th>bit 23</th><th>bit 24</th><th>bit 25</th><th>bit 26</th><th>bit 27</th><th>bit 28</th><th>bit 29</th><th>bit 30</th><th>bit 31</th><th>bit 32</th><th>bit 33</th><th>bit 34</th><th>bit 35</th><th>bit 36</th><th>bit 37</th><th>bit 38</th><th>bit 39</th><th>bit 40</th><th>bit 41</th><th>bit 42</th><th>bit 43</th><th>bit 44</th><th>bit 45</th><th>bit 46</th><th>bit 47</th><th>bit 48</th><th>bit 49</th><th>bit 50</th><th>bit 51</th><th>bit 52</th><th>bit 53</th><th>bit 54</th><th>bit 55</th><th>bit 56</th><th>bit 57</th><th>bit 58</th><th>bit 59</th><th>bit 60</th><th>bit 61</th><th>bit 62</th><th>bit 63</th><th>bit 64</th><th>bit 65</th><th>bit 66</th><th>bit 67</th><th>bit 68</th><th>bit 69</th><th>bit 70</th><th>bit 71</th><th>bit 72</th><th>bit 73</th><th>bit 74</th><th>bit 75</th><th>bit 76</th><th>bit 77</th><th>bit 78</th><th>bit 79</th><th>bit 80</th><th>bit 81</th><th>bit 82</th><th>bit 83</th><th>bit 84</th><th>bit 85</th><th>bit 86</th><th>bit 87</th><th>bit 88</th><th>bit 89</th><th>bit 90</th><th>bit 91</th><th>bit 92</th><th>bit 93</th><th>bit 94</th><th>bit 95</th><th>bit 96</th><th>bit 97</th><th>bit 98</th><th>bit 99</th><th>bit 100</th><th>bit 101</th><th>bit 102</th><th>bit 103</th><th>bit 104</th><th>bit 105</th><th>bit 106</th><th>bit 107</th><th>bit 108</th><th>bit 109</th><th>bit 110</th><th>bit 111</th><th>bit 112</th><th>bit 113</th><th>bit 114</th><th>bit 115</th><th>bit 116</th><th>bit 117</th><th>bit 118</th><th>bit 119</th><th>bit 120</th><th>bit 121</th><th>bit 122</th><th>bit 123</th><th>bit 124</th><th>bit 125</th><th>bit 126</th><th>bit 127</th><th>bit 128</th><th>bit 129</th><th>bit 130</th><th>bit 131</th><th>bit 132</th><th>bit 133</th><th>bit 134</th><th>bit 135</th><th>bit 136</th><th>bit 137</th><th>bit 138</th><th>bit 139</th><th>bit 140</th><th>bit 141</th><th>bit 142</th><th>bit 143</th><th>bit 144</th><th>bit 145</th><th>bit 146</th><th>bit 147</th><th>bit 148</th><th>bit 149</th><th>bit 150</th><th>bit 151</th><th>bit 152</th><th>bit 153</th><th>bit 154</th><th>bit 155</th><th>bit 156</th><th>bit 157</th><th>bit 158</th><th>bit 159</th><th>bit 160</th><th>bit 161</th><th>bit 162</th><th>bit 163</th><th>bit 164</th><th>bit 165</th><th>bit 166</th><th>bit 167</th><th>bit 168</th><th>bit 169</th><th>bit 170</th><th>bit 171</th><th>bit 172</th><th>bit 173</th><th>bit 174</th><th>bit 175</th><th>bit 176</th><th>bit 177</th><th>bit 178</th><th>bit 179</th><th>bit 180</th><th>bit 181</th><th>bit 182</th><th>bit 183</th><th>bit 184</th><th>bit 185</th><th>bit 186</th><th>bit 187</th><th>bit 188</th><th>bit 189</th><th>bit 190</th><th>bit 191</th><th>bit 192</th><th>bit 193</th><th>bit 194</th><th>bit 195</th><th>bit 196</th><th>bit 197</th><th>bit 198</th><th>bit 199</th><th>bit 200</th><th>bit 201</th><th>bit 202</th><th>bit 203</th><th>bit 204</th><th>bit 205</th><th>bit 206</th><th>bit 207</th><th>bit 208</th><th>bit 209</th><th>bit 210</th><th>bit 211</th><th>bit 212</th><th>bit 213</th><th>bit 214</th><th>bit 215</th><th>bit 216</th><th>bit 217</th><th>bit 218</th><th>bit 219</th><th>bit 220</th><th>bit 221</th><th>bit 222</th><th>bit 223</th><th>bit 224</th><th>bit 225</th><th>bit 226</th><th>bit 227</th><th>bit 228</th><th>bit 229</th><th>bit 230</th><th>bit 231</th><th>bit 232</th><th>bit 233</th><th>bit 234</th><th>bit 235</th><th>bit 236</th><th>bit 237</th><th>bit 238</th><th>bit 239</th><th>bit 240</th><th>bit 241</th><th>bit 242</th><th>bit 243</th><th>bit 244</th><th>bit 245</th><th>bit 246</th><th>bit 247</th><th>bit 248</th><th>bit 249</th><th>bit 250</th><th>bit 251</th><th>bit 252</th><th>bit 253</th><th>bit 254</th><th>bit 255</th><th>bit 256</th><th>bit 257</th><th>bit 258</th><th>bit 259</th><th>bit 260</th><th>bit 261</th><th>bit 262</th><th>bit 263</th><th>bit 264</th><th>bit 265</th><th>bit 266</th><th>bit 267</th><th>bit 268</th><th>bit 269</th><th>bit 270</th><th>bit 271</th><th>bit 272</th><th>bit 273</th><th>bit 274</th><th>bit 275</th><th>bit 276</th><th>bit 277</th><th>bit 278</th><th>bit 279</th><th>bit 280</th><th>bit 281</th><th>bit 282</th><th>bit 283</th><th>bit 284</th><th>bit 285</th><th>bit 286</th><th>bit 287</th><th>bit 288</th><th>bit 289</th><th>bit 290</th><th>bit 291</th><th>bit 292</th><th>bit 293</th><th>bit 294</th><th>bit 295</th><th>bit 296</th><th>bit 297</th><th>bit 298</th><th>bit 299</th><th>bit 300</th><th>bit 301</th><th>bit 302</th><th>bit 303</th><th>bit 304</th><th>bit 305</th><th>bit 306</th><th>bit 307</th><th>bit 308</th><th>bit 309</th><th>bit 310</th><th>bit 311</th><th>bit 312</th><th>bit 313</th><th>bit 314</th><th>bit 315</th><th>bit 316</th><th>bit 317</th><th>bit 318</th><th>bit 319</th><th>bit 320</th><th>bit 321</th><th>bit 322</th><th>bit 323</th><th>bit 324</th><th>bit 325</th><th>bit 326</th><th>bit 327</th><th>bit 328</th><th>bit 329</th><th>bit 330</th><th>bit 331</th><th>bit 332</th><th>bit 333</th><th>bit 334</th><th>bit 335</th><th>bit 336</th><th>bit 337</th><th>bit 338</th><th>bit 339</th><th>bit 340</th><th>bit 341</th><th>bit 342</th><th>bit 343</th><th>bit 344</th><th>bit 345</th><th>bit 346</th><th>bit 347</th><th>bit 348</th><th>bit 349</th><th>bit 350</th><th>bit 351</th><th>bit 352</th><th>bit 353</th><th>bit 354</th><th>bit 355</th><th>bit 356</th><th>bit 357</th><th>bit 358</th><th>bit 359</th><th>bit 360</th><th>bit 361</th><th>bit 362</th><th>bit 363</th><th>bit 364</th><th>bit 365</th><th>bit 366</th><th>bit 367</th><th>bit 368</th><th>bit 369</th><th>bit 370</th><th>bit 371</th><th>bit 372</th><th>bit 373</th><th>bit 374</th><th>bit 375</th><th>bit 376</th><th>bit 377</th><th>bit 378</th><th>bit 379</th><th>bit 380</th><th>bit 381</th><th>bit 382</th><th>bit 383</th><th>bit 384</th><th>bit 385</th><th>bit 386</th><th>bit 387</th><th>bit 388</th><th>bit 389</th><th>bit 390</th><th>bit 391</th><th>bit 392</th><th>bit 393</th><th>bit 394</th><th>bit 395</th><th>bit 396</th><th>bit 397</th><th>bit 398</th><th>bit 399</th><th>bit 400</th><th>bit 401</th><th>bit 402</th><th>bit 403</th><th>bit 404</th><th>bit 405</th><th>bit 406</th><th>bit 407</th><th>bit 408</th><th>bit 409</th><th>bit 410</th><th>bit 411</th><th>bit 412</th><th>bit 413</th><th>bit 414</th><th>bit 415</th><th>bit 416</th><th>bit 417</th><th>bit 418</th><th>bit 419</th><th>bit 420</th><th>bit 421</th><th>bit 422</th><th>bit 423</th><th>bit 424</th><th>bit 425</th><th>bit 426</th><th>bit 427</th><th>bit 428</th><th>bit 429</th><th>bit 430</th><th>bit 431</th><th>bit 432</th><th>bit 433</th><th>bit 434</th><th>bit 435</th><th>bit 436</th><th>bit 437</th><th>bit 438</th><th>bit 439</th><th>bit 440</th><th>bit 441</th><th>bit 442</th><th>bit 443</th><th>bit 444</th><th>bit 445</th><th>bit 446</th><th>bit 447</th><th>bit 448</th><th>bit 449</th><th>bit 450</th><th>bit 451</th><th>bit 452</th><th>bit 453</th><th>bit 454</th><th>bit 455</th><th>bit 456</th><th>bit 457</th><th>bit 458</th><th>bit 459</th><th>bit 460</th><th>bit 461</th><th>bit 462</th><th>bit 463</th><th>bit 464</th><th>bit 465</th><th>bit 466</th><th>bit 467</th><th>bit 468</th><th>bit 469</th><th>bit 470</th><th>bit 471</th><th>bit 472</th><th>bit 473</th><th>bit 474</th><th>bit 475</th><th>bit 476</th><th>bit 477</th><th>bit 478</th><th>bit 479</th><th>bit 480</th><th>bit 481</th><th>bit 482</th><th>bit 483</th><th>bit 484</th><th>bit 485</th><th>bit 486</th><th>bit 487</th><th>bit 488</th><th>bit 489</th><th>bit 490</th><th>bit 491</th><th>bit 492</th><th>bit 493</th><th>bit 494</th><th>bit 495</th><th>bit 496</th><th>bit 497</th><th>bit 498</th><th>bit 499</th><th>bit 500</th><th>bit 501</th><th>bit 502</th><th>bit 503</th><th>bit 504</th><th>bit 505</th><th>bit 506</th><th>bit 507</th><th>bit 508</th><th>bit 509</th><th>bit 510</th><th>bit 511</th><th>bit 512</th><th>bit 513</th><th>bit 514</th><th>bit 515</th><th>bit 516</th><th>bit 517</th><th>bit 518</th><th>bit 519</th><th>bit 520</th><th>bit 521</th><th>bit 522</th><th>bit 523</th><th>bit 524</th><th>bit 525</th><th>bit 526</th><th>bit 527</th><th>bit 528</th><th>bit 529</th><th>bit 530</th><th>bit 531</th><th>bit 532</th><th>bit 533</th><th>bit 534</th><th>bit 535</th><th>bit 536</th><th>bit 537</th><th>bit 538</th><th>bit 539</th><th>bit 540</th><th>bit 541</th><th>bit 542</th><th>bit 543</th><th>bit 544</th><th>bit 545</th><th>bit 546</th><th>bit 547</th><th>bit 548</th><th>bit 549</th><th>bit 550</th><th>bit 551</th><th>bit 552</th><th>bit 553</th><th>bit 554</th><th>bit 555</th><th>bit 556</th><th>bit 557</th><th>bit 558</th><th>bit 559</th><th>bit 560</th><th>bit 561</th><th>bit 562</th><th>bit 563</th><th>bit 564</th><th>bit 565</th><th>bit 566</th><th>bit 567</th><th>bit 568</th><th>bit 569</th><th>bit 570</th><th>bit 571</th><th>bit 572</th><th>bit 573</th><th>bit 574</th><th>bit 575</th><th>bit 576</th><th>bit 577</th><th>bit 578</th><th>bit 579</th><th>bit 580</th><th>bit 581</th><th>bit 582</th><th>bit 583</th><th>bit 584</th><th>bit 585</th><th>bit 586</th><th>bit 587</th><th>bit 588</th><th>bit 589</th><th>bit 590</th><th>bit 591</th><th>bit 592</th><th>bit 593</th><th>bit 594</th><th>bit 595</th><th>bit 596</th><th>bit 597</th><th>bit 598</th><th>bit 599</th><th>bit 600</th><th>bit 601</th><th>bit 602</th><th>bit 603</th><th>bit 604</th><th>bit 605</th><th>bit 606</th><th>bit 607</th><th>bit 608</th><th>bit 609</th><th>bit 610</th><th>bit 611</th><th>bit 612</th><th>bit 613</th><th>bit 614</th><th>bit 615</th><th>bit 616</th><th>bit 617</th><th>bit 618</th><th>bit 619</th><th>bit 620</th><th>bit 621</th><th>bit 622</th><th>bit 623</th><th>bit 624</th><th>bit 625</th><th>bit 626</th><th>bit 627</th><th>bit 628</th><th>bit 629</th><th>bit 630</th><th>bit 631</th><th>bit 632</th><th>bit 633</th><th>bit 634</th><th>bit 635</th><th>bit 636</th><th>bit 637</th><th>bit 638</th><th>bit 639</th><th>bit 640</th><th>bit 641</th><th>bit 642</th><th>bit 643</th><th>bit 644</th><th>bit 645</th><th>bit 646</th><th>bit 647</th><th>bit 648</th><th>bit 649</th><th>bit 650</th><th>bit 651</th><th>bit 652</th><th>bit 653</th><th>bit 654</th><th>bit 655</th><th>bit 656</th><th>bit 657</th><th>bit 658</th><th>bit 659</th><th>bit 660</th><th>bit 661</th><th>bit 662</th><th>bit 663</th><th>bit 664</th><th>bit 665</th><th>bit 666</th><th>bit 667</th><th>bit 668</th><th>bit 669</th><th>bit 670</th><th>bit 671</th><th>bit 672</th><th>bit 673</th><th>bit 674</th><th>bit 675</th><th>bit 676</th><th>bit 677</th><th>bit 678</th><th>bit 679</th><th>bit 680</th><th>bit 681</th</th></tr></thead></table>	Eid	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	bit 11	bit 12	bit 13	bit 14	bit 15	bit 16	bit 17	bit 18	bit 19	bit 20	bit 21	bit 22	bit 23	bit 24	bit 25	bit 26	bit 27	bit 28	bit 29	bit 30	bit 31	bit 32	bit 33	bit 34	bit 35	bit 36	bit 37	bit 38	bit 39	bit 40	bit 41	bit 42	bit 43	bit 44	bit 45	bit 46	bit 47	bit 48	bit 49	bit 50	bit 51	bit 52	bit 53	bit 54	bit 55	bit 56	bit 57	bit 58	bit 59	bit 60	bit 61	bit 62	bit 63	bit 64	bit 65	bit 66	bit 67	bit 68	bit 69	bit 70	bit 71	bit 72	bit 73	bit 74	bit 75	bit 76	bit 77	bit 78	bit 79	bit 80	bit 81	bit 82	bit 83	bit 84	bit 85	bit 86	bit 87	bit 88	bit 89	bit 90	bit 91	bit 92	bit 93	bit 94	bit 95	bit 96	bit 97	bit 98	bit 99	bit 100	bit 101	bit 102	bit 103	bit 104	bit 105	bit 106	bit 107	bit 108	bit 109	bit 110	bit 111	bit 112	bit 113	bit 114	bit 115	bit 116	bit 117	bit 118	bit 119	bit 120	bit 121	bit 122	bit 123	bit 124	bit 125	bit 126	bit 127	bit 128	bit 129	bit 130	bit 131	bit 132	bit 133	bit 134	bit 135	bit 136	bit 137	bit 138	bit 139	bit 140	bit 141	bit 142	bit 143	bit 144	bit 145	bit 146	bit 147	bit 148	bit 149	bit 150	bit 151	bit 152	bit 153	bit 154	bit 155	bit 156	bit 157	bit 158	bit 159	bit 160	bit 161	bit 162	bit 163	bit 164	bit 165	bit 166	bit 167	bit 168	bit 169	bit 170	bit 171	bit 172	bit 173	bit 174	bit 175	bit 176	bit 177	bit 178	bit 179	bit 180	bit 181	bit 182	bit 183	bit 184	bit 185	bit 186	bit 187	bit 188	bit 189	bit 190	bit 191	bit 192	bit 193	bit 194	bit 195	bit 196	bit 197	bit 198	bit 199	bit 200	bit 201	bit 202	bit 203	bit 204	bit 205	bit 206	bit 207	bit 208	bit 209	bit 210	bit 211	bit 212	bit 213	bit 214	bit 215	bit 216	bit 217	bit 218	bit 219	bit 220	bit 221	bit 222	bit 223	bit 224	bit 225	bit 226	bit 227	bit 228	bit 229	bit 230	bit 231	bit 232	bit 233	bit 234	bit 235	bit 236	bit 237	bit 238	bit 239	bit 240	bit 241	bit 242	bit 243	bit 244	bit 245	bit 246	bit 247	bit 248	bit 249	bit 250	bit 251	bit 252	bit 253	bit 254	bit 255	bit 256	bit 257	bit 258	bit 259	bit 260	bit 261	bit 262	bit 263	bit 264	bit 265	bit 266	bit 267	bit 268	bit 269	bit 270	bit 271	bit 272	bit 273	bit 274	bit 275	bit 276	bit 277	bit 278	bit 279	bit 280	bit 281	bit 282	bit 283	bit 284	bit 285	bit 286	bit 287	bit 288	bit 289	bit 290	bit 291	bit 292	bit 293	bit 294	bit 295	bit 296	bit 297	bit 298	bit 299	bit 300	bit 301	bit 302	bit 303	bit 304	bit 305	bit 306	bit 307	bit 308	bit 309	bit 310	bit 311	bit 312	bit 313	bit 314	bit 315	bit 316	bit 317	bit 318	bit 319	bit 320	bit 321	bit 322	bit 323	bit 324	bit 325	bit 326	bit 327	bit 328	bit 329	bit 330	bit 331	bit 332	bit 333	bit 334	bit 335	bit 336	bit 337	bit 338	bit 339	bit 340	bit 341	bit 342	bit 343	bit 344	bit 345	bit 346	bit 347	bit 348	bit 349	bit 350	bit 351	bit 352	bit 353	bit 354	bit 355	bit 356	bit 357	bit 358	bit 359	bit 360	bit 361	bit 362	bit 363	bit 364	bit 365	bit 366	bit 367	bit 368	bit 369	bit 370	bit 371	bit 372	bit 373	bit 374	bit 375	bit 376	bit 377	bit 378	bit 379	bit 380	bit 381	bit 382	bit 383	bit 384	bit 385	bit 386	bit 387	bit 388	bit 389	bit 390	bit 391	bit 392	bit 393	bit 394	bit 395	bit 396	bit 397	bit 398	bit 399	bit 400	bit 401	bit 402	bit 403	bit 404	bit 405	bit 406	bit 407	bit 408	bit 409	bit 410	bit 411	bit 412	bit 413	bit 414	bit 415	bit 416	bit 417	bit 418	bit 419	bit 420	bit 421	bit 422	bit 423	bit 424	bit 425	bit 426	bit 427	bit 428	bit 429	bit 430	bit 431	bit 432	bit 433	bit 434	bit 435	bit 436	bit 437	bit 438	bit 439	bit 440	bit 441	bit 442	bit 443	bit 444	bit 445	bit 446	bit 447	bit 448	bit 449	bit 450	bit 451	bit 452	bit 453	bit 454	bit 455	bit 456	bit 457	bit 458	bit 459	bit 460	bit 461	bit 462	bit 463	bit 464	bit 465	bit 466	bit 467	bit 468	bit 469	bit 470	bit 471	bit 472	bit 473	bit 474	bit 475	bit 476	bit 477	bit 478	bit 479	bit 480	bit 481	bit 482	bit 483	bit 484	bit 485	bit 486	bit 487	bit 488	bit 489	bit 490	bit 491	bit 492	bit 493	bit 494	bit 495	bit 496	bit 497	bit 498	bit 499	bit 500	bit 501	bit 502	bit 503	bit 504	bit 505	bit 506	bit 507	bit 508	bit 509	bit 510	bit 511	bit 512	bit 513	bit 514	bit 515	bit 516	bit 517	bit 518	bit 519	bit 520	bit 521	bit 522	bit 523	bit 524	bit 525	bit 526	bit 527	bit 528	bit 529	bit 530	bit 531	bit 532	bit 533	bit 534	bit 535	bit 536	bit 537	bit 538	bit 539	bit 540	bit 541	bit 542	bit 543	bit 544	bit 545	bit 546	bit 547	bit 548	bit 549	bit 550	bit 551	bit 552	bit 553	bit 554	bit 555	bit 556	bit 557	bit 558	bit 559	bit 560	bit 561	bit 562	bit 563	bit 564	bit 565	bit 566	bit 567	bit 568	bit 569	bit 570	bit 571	bit 572	bit 573	bit 574	bit 575	bit 576	bit 577	bit 578	bit 579	bit 580	bit 581	bit 582	bit 583	bit 584	bit 585	bit 586	bit 587	bit 588	bit 589	bit 590	bit 591	bit 592	bit 593	bit 594	bit 595	bit 596	bit 597	bit 598	bit 599	bit 600	bit 601	bit 602	bit 603	bit 604	bit 605	bit 606	bit 607	bit 608	bit 609	bit 610	bit 611	bit 612	bit 613	bit 614	bit 615	bit 616	bit 617	bit 618	bit 619	bit 620	bit 621	bit 622	bit 623	bit 624	bit 625	bit 626	bit 627	bit 628	bit 629	bit 630	bit 631	bit 632	bit 633	bit 634	bit 635	bit 636	bit 637	bit 638	bit 639	bit 640	bit 641	bit 642	bit 643	bit 644	bit 645	bit 646	bit 647	bit 648	bit 649	bit 650	bit 651	bit 652	bit 653	bit 654	bit 655	bit 656	bit 657	bit 658	bit 659	bit 660	bit 661	bit 662	bit 663	bit 664	bit 665	bit 666	bit 667	bit 668	bit 669	bit 670	bit 671	bit 672	bit 673	bit 674	bit 675	bit 676	bit 677	bit 678	bit 679	bit 680	bit 681</th
Eid	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8	bit 9	bit 10	bit 11	bit 12	bit 13	bit 14	bit 15	bit 16	bit 17	bit 18	bit 19	bit 20	bit 21	bit 22	bit 23	bit 24	bit 25	bit 26	bit 27	bit 28	bit 29	bit 30	bit 31	bit 32	bit 33	bit 34	bit 35	bit 36	bit 37	bit 38	bit 39	bit 40	bit 41	bit 42	bit 43	bit 44	bit 45	bit 46	bit 47	bit 48	bit 49	bit 50	bit 51	bit 52	bit 53	bit 54	bit 55	bit 56	bit 57	bit 58	bit 59	bit 60	bit 61	bit 62	bit 63	bit 64	bit 65	bit 66	bit 67	bit 68	bit 69	bit 70	bit 71	bit 72	bit 73	bit 74	bit 75	bit 76	bit 77	bit 78	bit 79	bit 80	bit 81	bit 82	bit 83	bit 84	bit 85	bit 86	bit 87	bit 88	bit 89	bit 90	bit 91	bit 92	bit 93	bit 94	bit 95	bit 96	bit 97	bit 98	bit 99	bit 100	bit 101	bit 102	bit 103	bit 104	bit 105	bit 106	bit 107	bit 108	bit 109	bit 110	bit 111	bit 112	bit 113	bit 114	bit 115	bit 116	bit 117	bit 118	bit 119	bit 120	bit 121	bit 122	bit 123	bit 124	bit 125	bit 126	bit 127	bit 128	bit 129	bit 130	bit 131	bit 132	bit 133	bit 134	bit 135	bit 136	bit 137	bit 138	bit 139	bit 140	bit 141	bit 142	bit 143	bit 144	bit 145	bit 146	bit 147	bit 148	bit 149	bit 150	bit 151	bit 152	bit 153	bit 154	bit 155	bit 156	bit 157	bit 158	bit 159	bit 160	bit 161	bit 162	bit 163	bit 164	bit 165	bit 166	bit 167	bit 168	bit 169	bit 170	bit 171	bit 172	bit 173	bit 174	bit 175	bit 176	bit 177	bit 178	bit 179	bit 180	bit 181	bit 182	bit 183	bit 184	bit 185	bit 186	bit 187	bit 188	bit 189	bit 190	bit 191	bit 192	bit 193	bit 194	bit 195	bit 196	bit 197	bit 198	bit 199	bit 200	bit 201	bit 202	bit 203	bit 204	bit 205	bit 206	bit 207	bit 208	bit 209	bit 210	bit 211	bit 212	bit 213	bit 214	bit 215	bit 216	bit 217	bit 218	bit 219	bit 220	bit 221	bit 222	bit 223	bit 224	bit 225	bit 226	bit 227	bit 228	bit 229	bit 230	bit 231	bit 232	bit 233	bit 234	bit 235	bit 236	bit 237	bit 238	bit 239	bit 240	bit 241	bit 242	bit 243	bit 244	bit 245	bit 246	bit 247	bit 248	bit 249	bit 250	bit 251	bit 252	bit 253	bit 254	bit 255	bit 256	bit 257	bit 258	bit 259	bit 260	bit 261	bit 262	bit 263	bit 264	bit 265	bit 266	bit 267	bit 268	bit 269	bit 270	bit 271	bit 272	bit 273	bit 274	bit 275	bit 276	bit 277	bit 278	bit 279	bit 280	bit 281	bit 282	bit 283	bit 284	bit 285	bit 286	bit 287	bit 288	bit 289	bit 290	bit 291	bit 292	bit 293	bit 294	bit 295	bit 296	bit 297	bit 298	bit 299	bit 300	bit 301	bit 302	bit 303	bit 304	bit 305	bit 306	bit 307	bit 308	bit 309	bit 310	bit 311	bit 312	bit 313	bit 314	bit 315	bit 316	bit 317	bit 318	bit 319	bit 320	bit 321	bit 322	bit 323	bit 324	bit 325	bit 326	bit 327	bit 328	bit 329	bit 330	bit 331	bit 332	bit 333	bit 334	bit 335	bit 336	bit 337	bit 338	bit 339	bit 340	bit 341	bit 342	bit 343	bit 344	bit 345	bit 346	bit 347	bit 348	bit 349	bit 350	bit 351	bit 352	bit 353	bit 354	bit 355	bit 356	bit 357	bit 358	bit 359	bit 360	bit 361	bit 362	bit 363	bit 364	bit 365	bit 366	bit 367	bit 368	bit 369	bit 370	bit 371	bit 372	bit 373	bit 374	bit 375	bit 376	bit 377	bit 378	bit 379	bit 380	bit 381	bit 382	bit 383	bit 384	bit 385	bit 386	bit 387	bit 388	bit 389	bit 390	bit 391	bit 392	bit 393	bit 394	bit 395	bit 396	bit 397	bit 398	bit 399	bit 400	bit 401	bit 402	bit 403	bit 404	bit 405	bit 406	bit 407	bit 408	bit 409	bit 410	bit 411	bit 412	bit 413	bit 414	bit 415	bit 416	bit 417	bit 418	bit 419	bit 420	bit 421	bit 422	bit 423	bit 424	bit 425	bit 426	bit 427	bit 428	bit 429	bit 430	bit 431	bit 432	bit 433	bit 434	bit 435	bit 436	bit 437	bit 438	bit 439	bit 440	bit 441	bit 442	bit 443	bit 444	bit 445	bit 446	bit 447	bit 448	bit 449	bit 450	bit 451	bit 452	bit 453	bit 454	bit 455	bit 456	bit 457	bit 458	bit 459	bit 460	bit 461	bit 462	bit 463	bit 464	bit 465	bit 466	bit 467	bit 468	bit 469	bit 470	bit 471	bit 472	bit 473	bit 474	bit 475	bit 476	bit 477	bit 478	bit 479	bit 480	bit 481	bit 482	bit 483	bit 484	bit 485	bit 486	bit 487	bit 488	bit 489	bit 490	bit 491	bit 492	bit 493	bit 494	bit 495	bit 496	bit 497	bit 498	bit 499	bit 500	bit 501	bit 502	bit 503	bit 504	bit 505	bit 506	bit 507	bit 508	bit 509	bit 510	bit 511	bit 512	bit 513	bit 514	bit 515	bit 516	bit 517	bit 518	bit 519	bit 520	bit 521	bit 522	bit 523	bit 524	bit 525	bit 526	bit 527	bit 528	bit 529	bit 530	bit 531	bit 532	bit 533	bit 534	bit 535	bit 536	bit 537	bit 538	bit 539	bit 540	bit 541	bit 542	bit 543	bit 544	bit 545	bit 546	bit 547	bit 548	bit 549	bit 550	bit 551	bit 552	bit 553	bit 554	bit 555	bit 556	bit 557	bit 558	bit 559	bit 560	bit 561	bit 562	bit 563	bit 564	bit 565	bit 566	bit 567	bit 568	bit 569	bit 570	bit 571	bit 572	bit 573	bit 574	bit 575	bit 576	bit 577	bit 578	bit 579	bit 580	bit 581	bit 582	bit 583	bit 584	bit 585	bit 586	bit 587	bit 588	bit 589	bit 590	bit 591	bit 592	bit 593	bit 594	bit 595	bit 596	bit 597	bit 598	bit 599	bit 600	bit 601	bit 602	bit 603	bit 604	bit 605	bit 606	bit 607	bit 608	bit 609	bit 610	bit 611	bit 612	bit 613	bit 614	bit 615	bit 616	bit 617	bit 618	bit 619	bit 620	bit 621	bit 622	bit 623	bit 624	bit 625	bit 626	bit 627	bit 628	bit 629	bit 630	bit 631	bit 632	bit 633	bit 634	bit 635	bit 636	bit 637	bit 638	bit 639	bit 640	bit 641	bit 642	bit 643	bit 644	bit 645	bit 646	bit 647	bit 648	bit 649	bit 650	bit 651	bit 652	bit 653	bit 654	bit 655	bit 656	bit 657	bit 658	bit 659	bit 660	bit 661	bit 662	bit 663	bit 664	bit 665	bit 666	bit 667	bit 668	bit 669	bit 670	bit 671	bit 672	bit 673	bit 674	bit 675	bit 676	bit 677	bit 678	bit 679	bit 680	bit 681</th		

AMS1117DC3V3



⚡ AMS1117 3.3V Buck reverse-engineered

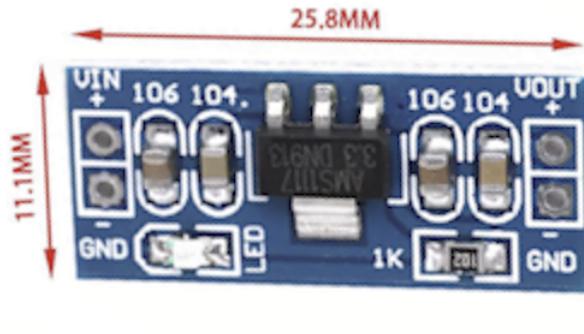
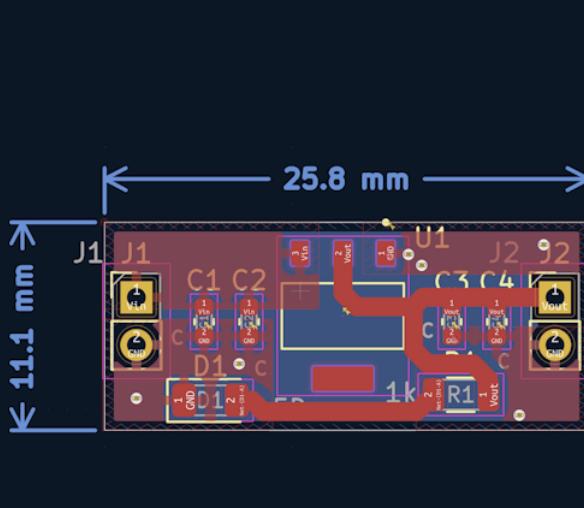
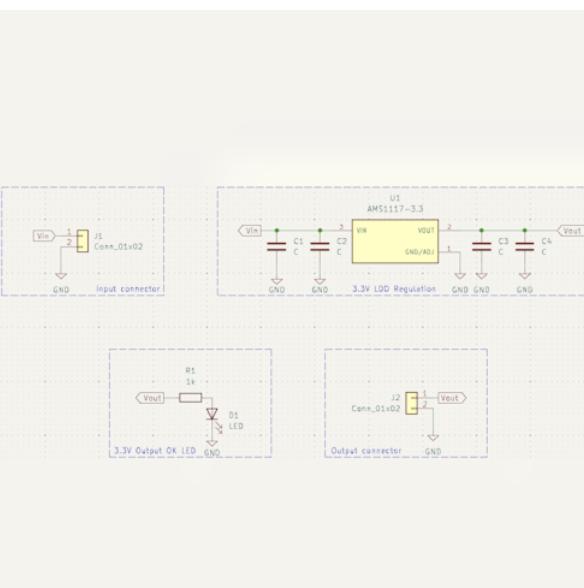
An open-source reverse-engineered version of the AMS1117 3.3V DC-DC buck converter module, based on the original component available [here](#).

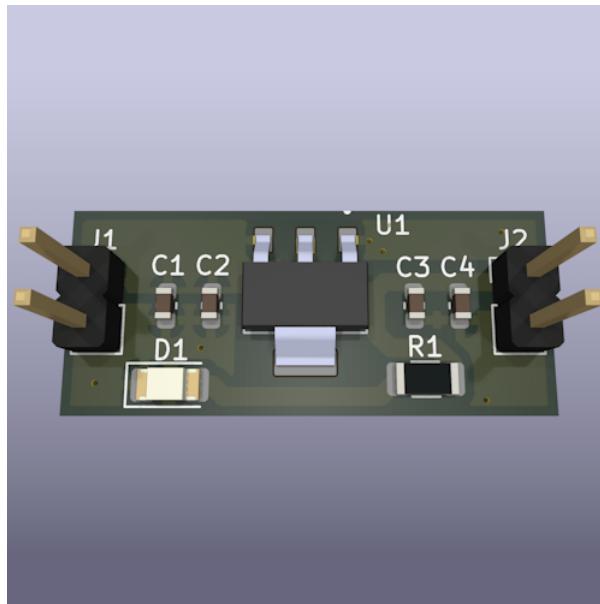
This project aims to provide insights into DC-DC voltage regulation and offer a customizable alternative for power management in embedded systems.

Purpose

- 🔍 **Reverse engineering:** Understanding the design and functionality of the AMS1117-based voltage regulator.
- 🛠️ **Skill development:** Enhancing expertise in PCB design and power electronics.
- 🔄 **Future adaptation:** Leveraging this knowledge to develop custom voltage regulation solutions for embedded applications.

📝 Features Comparison: Original vs. Reverse-Engineered

Feature	Original Module	Reverse-Engineered Version
💻 PCB Design	Proprietary	Open-source & customizable
🔌 Input Voltage	4.8V - 15V	4.8V - 15V
⚡ Output Voltage	3.3V (fixed)	3.3V (fixed)
📦 Max Current	1500 mA	1500 mA
(chip) Regulator Chip	AMS1117-3.3	AMS1117-3.3
👉 Mechanical Drawing		
➡️ Reverse-Engineered Schematic	N/A	

Feature	Original Module	Reverse-Engineered Version
Photo		

🛠️ How to Use 📌 Wiring Guide

Pin	Description
VIN	Input Voltage (4.8V - 15V)
GND	Ground
VOUT	Regulated 3.3V Output

CP2102USB2UART



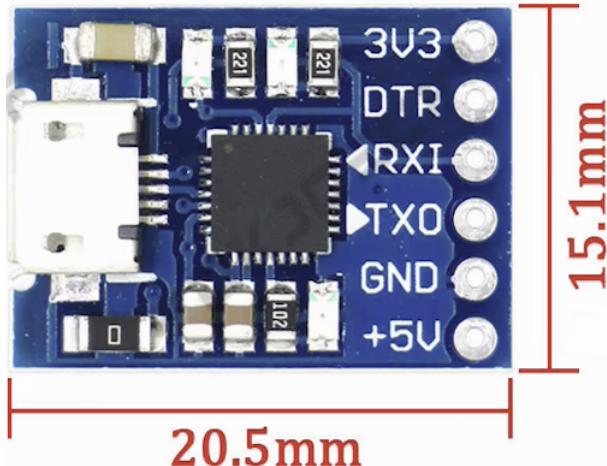
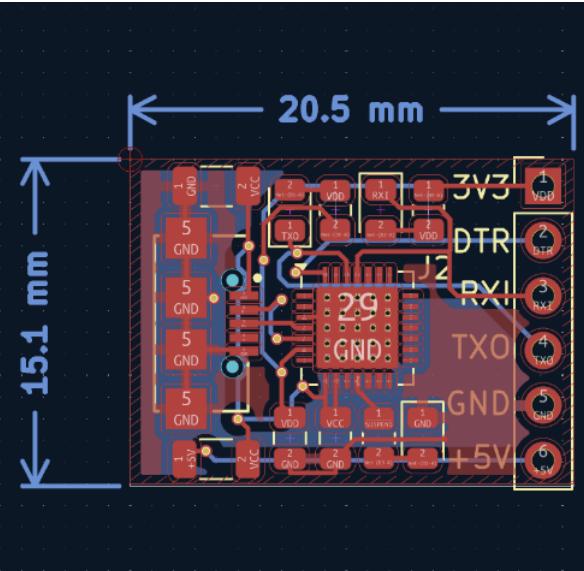
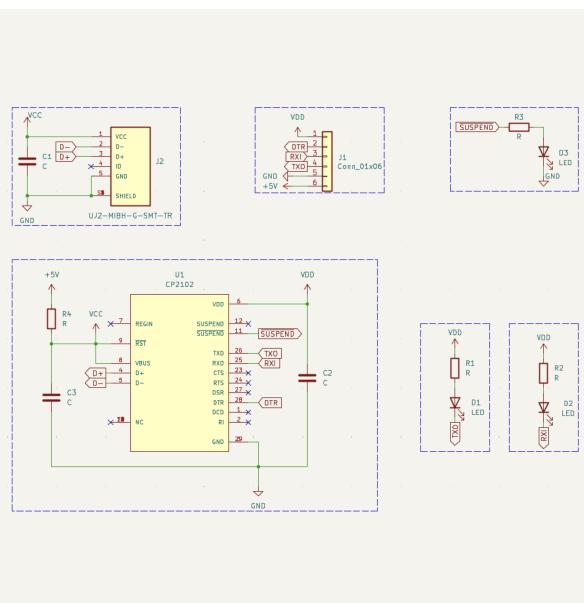
💡 CP2102 USB to UART reverse-engineered

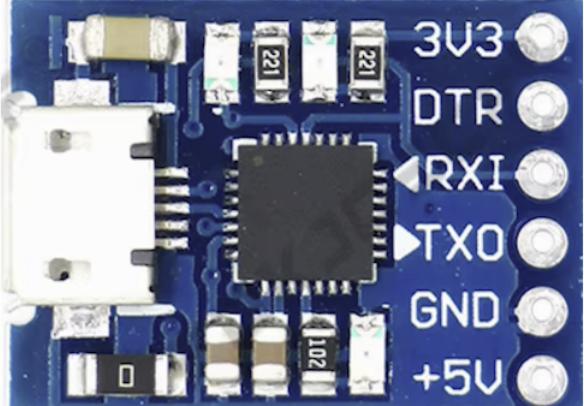
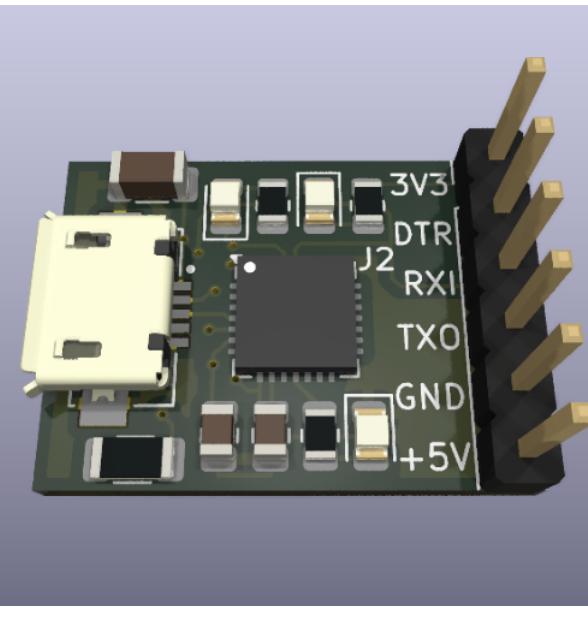
An open-source reverse-engineered version of the CJMCU CP2102 USB to UART TTL adapter, based on the original component available [here](#). This project aims to provide insights into USB-to-serial communication and offer a customizable alternative for embedded system developers.

🎯 Purpose

- 🔍 **Reverse engineering:** Understanding the design and functionality of the CP2102USB2UART.
- 🛠️ **Skill development:** Enhancing expertise in PCB design and USB-to-serial communication.
- 🌐 **Future adaptation:** Leveraging this knowledge to develop custom USB-to-UART solutions for embedded systems.

📝 Features Comparison: Original vs. Reverse-Engineered

Feature	Original Module	Reverse-Engineered Version
💻 PCB Design	Proprietary	Open-source & customizable
🔌 USB Connector	Micro USB	Micro USB
(chip) Chipset	CP2102	CP2102
📌 Pin Mapping	6-Pin UART TTL	6-Pin UART TTL
⚡ Supported Voltage	3.3V / 5V	3.3V / 5V
🟡 Mechanical Drawing		
📝 Reverse-Engineered Schematic	N/A	

Feature	Original Module	Reverse-Engineered Version
Photo		

🔧 How to Use 📌 Wiring Guide

CP2102 Pin	Description
TXD	Transmit Data
RXD	Receive Data
GND	Ground
3V3	3.3V Power Output
5V	5V Power Output
DTR	Data Terminal Ready

MicroUSB2DIP



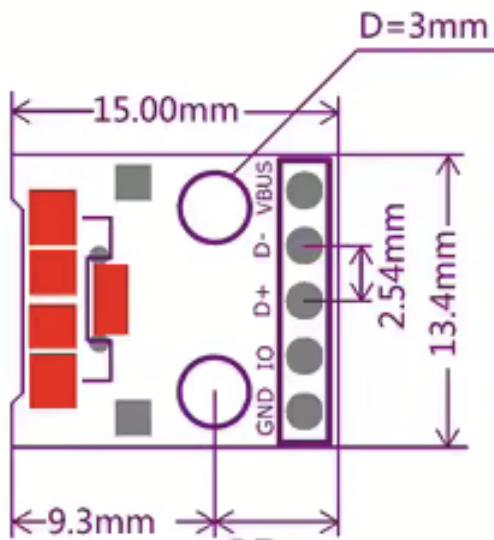
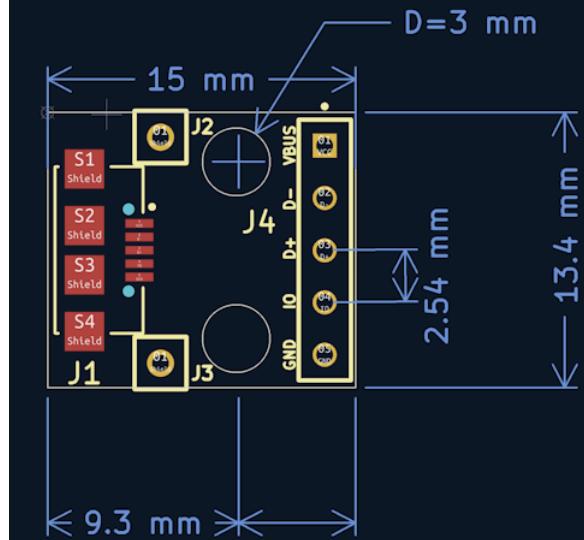
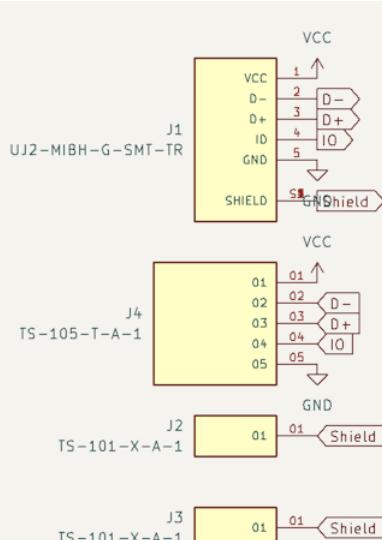
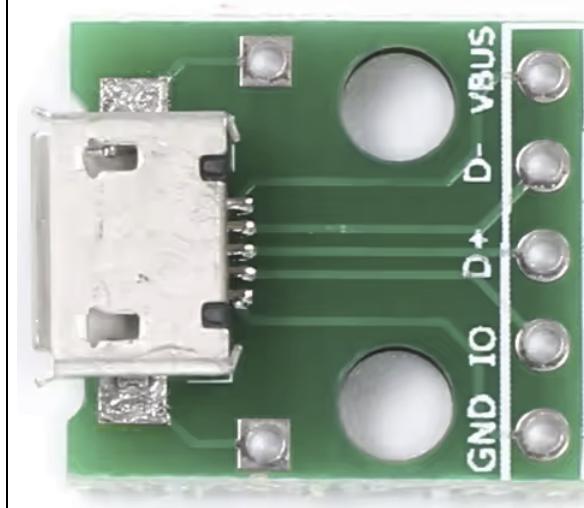
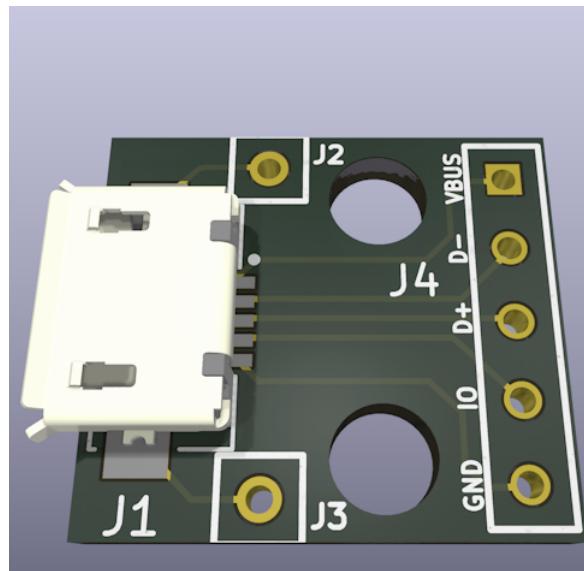
💡 Micro USB to DIP reverse-engineered

This is an open-source reverse-engineered version of a Micro USB to DIP adapter, based on the original component available [here](#). The goal of this project was to practice reverse engineering as a learning exercise and to prepare for a future adaptation in a larger project.

🎯 Purpose

- 🔍 **Reverse engineering:** Understanding the design and functionality of this micro USB to DIP connector.
- 🛠️ **Skill development:** Enhancing expertise in PCB design and hardware analysis skills.
- 🌐 **Future adaptation:** Leveraging this knowledge for embedded applications.

📝 Features Comparison: Original vs. Reverse-Engineered

Feature	Original Module	Reverse-Engineered Version
💻 PCB Design	Proprietary	Open-source & customizable
🔌 Connector Type	Micro USB	Micro USB
📌 Pin Mapping	Standard DIP	Standard DIP
🟡 Mechanical Drawing		
📝 Reverse-Engineered Schematic	N/A	
📷 Photo		



📌 Wiring Guide

Pin	Function
VBUS	+5V
D-	Data -
D+	Data +
ID	Mode detect (A: GND, B: Open)
GND	Ground

AntiVuvuzelaFilter



🎵 Noise filter for clear audio 🎤

Anti-Vuvuzela Filter is an open-source project dedicated to **second-order analog filters** and beyond. This project was initially developed to design an “**anti-vuvuzela**” **filter**, aiming to attenuate the distinctive and persistent sound of vuvuzelas while preserving the clarity of commentators’ voices during the **2010 FIFA World Cup**.

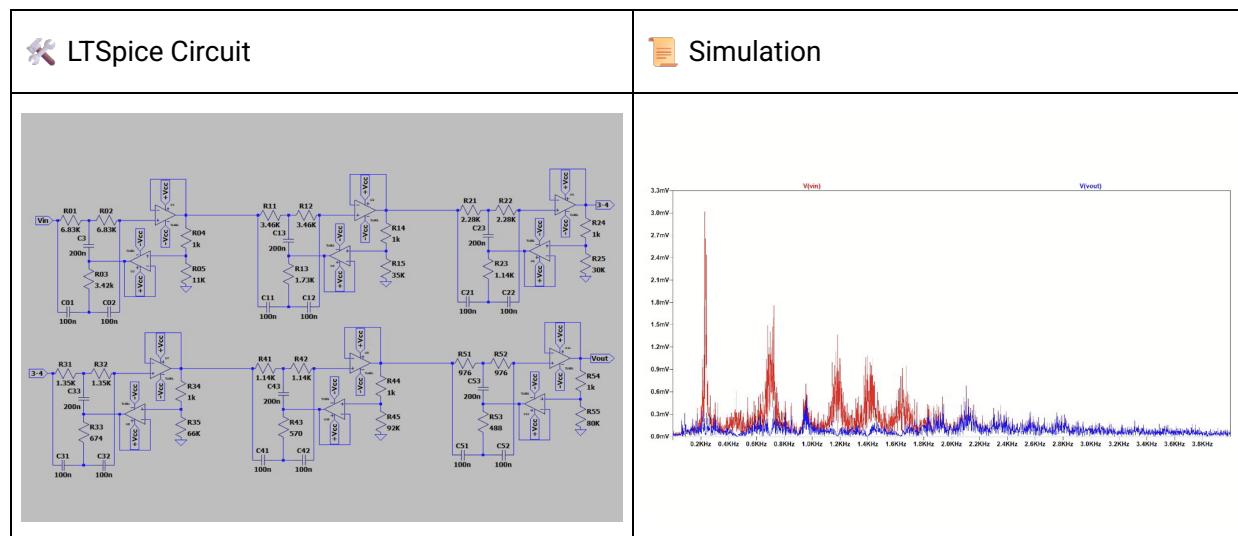
🎯 Purpose

- 📌 **Targeted Noise Reduction:** Specifically designed to **attenuate vuvuzela noise** while maintaining the intelligibility of speech.
- ⌚ **Second-Order Analog Filtering:** Utilizing advanced filtering techniques for efficient noise cancellation.
- 🛠️ **Open-source and Customizable:** Modify and adapt the design for other audio filtering applications.

📝 Features

💡 Feature	🔍 Description
🎵 Filter Type	Second-order analog filter
🎯 Target Frequency	233 Hz (typical vuvuzela frequency)
🎙 Voice Preservation	Maintains speech clarity
🔧 Components	Resistors, capacitors, and operational amplifiers
💻 Simulation Tools	Jupyter Notebook, LTSpice
🛠 Real-world Testing	Assembled and tested in real conditions
🔌 Input	Analog audio signal
🔊 Output	Cleaned audio signal with reduced vuvuzela noise
🌐 Use Cases	Audio signal processing, speech enhancement, noise reduction

📐 Simulation & Testing



DtmfCodeAnalyzer



Analyze and filter DTMF signals

DtmfCodeAnalyzer is an open-source project designed to analyze and filter DTMF (Dual-Tone Multi-Frequency) signals used in conventional telephony. The project provides functionalities to detect and identify keypresses from audio recordings, as well as remove DTMF tones to isolate voice signals.

Purpose

- **DTMF Signal Recognition:** Identify the key pressed based on the audio recording of its emitted frequencies.
- **DTMF Noise Removal:** Extract and suppress DTMF tones from an audio sample to reveal underlying speech.
- **Mathematical Approach:** Use vector projection in an Euclidean space to determine the closest frequency matches.

Features

Feature	Description
DTMF Frequencies	Combination of two distinct tones per key
Key Identification	Detects and determines the key pressed
Noise Filtering	Removes DTMF tones while preserving speech
Mathematical Model	Projects signals into a vector space for analysis
Audio Processing	Works with recorded audio samples
Open-Source	Fully customizable and modifiable

Signal Processing Approach

Frequency Vector Representation	Euclidean Projection	Filtering
<pre>def function_u(i,t): return np.sin(t*((i)%7)*2*np.pi+(i//7)*np.pi/2) def createOrthonormalBasis(sample_rate,signal,freq): u = [] t = np.linspace(0,0.1,freq) # intervalle d'échantillonage for i in range(14): u.append(np.array(function_u(i,t))) return u</pre>	<p>3.1.3 Finding the Linear Combination that Created It Then, we will find the linear combination that originated the emitted sound. To do this, we will perform an orthogonal projection of our sound onto the orthonormal basis created previously.</p> <pre>def scalarProduct(u,v): return 2*integral(sp.multiply(u,v)) def coordinates(u,v): c = [] for i in range(len(u)): c.append(scalarProduct(u[i],v)) return c def projectionOrtho(u,v): c = coordinates(u,v) p = np.zeros(len(u)) for i in range(len(u)): p[i] = c[i]*u[i] return (c,p)</pre> <p>To perform this orthogonal projection, we will compute the inner product of our sound with each vector in the basis.</p> <pre>def scalarProduct(u,v): return integral(u*v) def integral(u): dt = 1/(len(u)-1) sum_u = 0 for i in range(len(u)-1): sum_u += u[i]*dt return sum_u*dt</pre>	<p>3.1.4 Identifying the Two Main Frequencies To form a DTMF code, two sine waves of different frequencies are used. The goal here is to identify these two frequencies.</p> <pre>#def findingDtmfFrequencies(): # c,p = c_processing() # m1 = c[0].index(c_processing, key=abs) # m2 = c[1].index(c_processing, key=abs) # #remove the largest number using its index # if abs(m1) > abs(m2): # m1 = -m1 # else: # m2 = -m2 # return (m1,m2)</pre> <p>3.1.5 Identifying the Associated Digit Each combination of vectors is associated with a telephone digit. The question here is to determine which digit it is.</p> <pre>#def whatXisIt(x1,x2): # smallest = min((x1,x2)) # biggest = max((x1,x2)) # matrice = [[1,2,3],[2,4,5],[3,5,6],[4,6,7],[5,7,8],[6,8,9],[7,9,0],[8,0,1]] # if matrice[smallest][biggest] == 4: # return matrice[smallest][biggest-1] # else: # return "La combinaison linéaire ne permet pas de déterminer une touche"</pre>

AntUpRising



Global Game Jam 2023

AntUpRising is an engaging 2D platformer created during the Global Game Jam 2023. Step into the shoes of an ant and dive into a vibrant world where your mission is to bring food back to the queen ant. But beware! Strange roots have taken over the anthill, causing the ants to become aggressive and unpredictable.

Purpose

- 🍎 **Food Retrieval:** Collect resources to nourish the queen ant.
- ✂️ **Avoid Threats:** Dodge the crazed ants and other dangers along your journey.
- 🤰 **Reach the Queen:** Overcome obstacles to make your way to the queen's chamber.

Features

- 🌎 **Vibrant 2D Environment:** Immerse yourself in a colorful and detailed world.
- 🚶 **Dynamic Gameplay Mechanics:** Interact with enemies and tackle various challenges.
- 🎮 **Immersive Gaming Experience:** Enjoy an exciting adventure through the tunnels of the anthill.

Team Members

- Mael Madec
- Florian Pasco
- Romain Cloâtre
- Théo De Morais
- Romain Fauvel

BoardMapper



PCB placement map generator

BoardMapper is an open-source tool designed to automatically generate PCB layout. It labels component references (e.g. U1, R1, C1) directly on the circuit image, facilitating component identification for reverse engineering purposes.

Purpose

- **Automation:** Eliminates the need for manual placement annotation on PCB layouts.
- **Efficiency:** Saves time for engineers and makers working on PCB assembly and debugging.
- **Clarity:** Provides a clear visual reference for debugging, testing, and manufacturing.
- **Cross-Platform:** Works on Windows, Linux, and macOS systems.

Annotation

Position	Original	Annotated
Top		
Bottom		

Requirements

- **Python:** Version 3.6 or higher
- **Required Libraries:**
 - `opencv-python` (for image processing)
 - `lxml` (for XML parsing)

Installation Instructions

Setup

1. **Clone the repository** or **Download the project** to your local machine.

2. Labeling the PCB:

- **Step 1:** Take a photo of both the top and bottom layers of the chosen PCB.
- **Step 2:** Place the `top.png` and `bottom.png` images into the `input` folder.
- **Step 3:** Install the latest version of `LabelImg`.
- **Step 4:** Open `top.png` in LabelImg and draw bounding boxes around each component. Label each component according to its type:
 - **R:** Resistor
 - **C:** Capacitor
 - **L:** Inductor
 - **F:** Fuse
 - **POT:** Potentiometer
 - **D:** Diode
 - **LED:** LED
 - **Q:** Transistor (BJT, MOSFET)
 - **U:** Integrated Circuit (IC)
 - **J:** Connector
 - **K:** Relay
 - **SW:** Switch
 - **Y:** Quartz / Resonator
 - **SP:** Speaker
 - **ANT:** Antenna

LabelImg Shortcuts:

-  **W:** Draw a new rectangular bounding box (RectBox)
 -  **D:** Delete the last drawn bounding box
 -  **Ctrl + S:** Save the annotation as an XML file
 -  **Ctrl + Z:** Undo the last action
 -  **Ctrl + C:** Copy a bounding box
 -  **Ctrl + V:** Paste a copied bounding box
 -  **Ctrl + A:** Select all bounding boxes
 -  **Ctrl + R:** Rotate the image (for better labeling)
 -  **Esc:** Cancel the current operation or close a dialog box
- **Step 5:** After labeling the `top.png`, save the annotation as `top.xml`.
 - **Step 6:** Repeat the labeling process for the `bottom.png` and save it as `bottom.xml`.
 - **Step 7:** Place both `top.xml` and `bottom.xml` into the `input` folder.

3. Running the Tool:

- **Windows:** Double-click on `setup_and_run.bat` to automatically run the script. The tool will read the XML annotations, draw bounding boxes on the images, and save the annotated images.
- **Linux/macOS:** You can run the script from the terminal:

```
chmod +x script.sh  
./script.sh
```

4. Output:

- After the script has executed, navigate to the `output` folder to find the resulting annotated images:
 - `top_annotated.png`
 - `bottom_annotated.png`

Contributions

If you'd like to contribute to the project, please follow these steps:

1. Fork the repository
2. Create a feature branch
3. Commit your changes
4. Push to the branch
5. Open a pull request

We welcome any contributions to improve BoardMapper! 



CubeCourse2D

👤 Fast-paced random platformer!

Game Preview

CubeCourse2D is a fast-paced platformer where the player must survive as long as possible by jumping from platform to platform. But be careful—platforms are generated randomly, making each game unique and unpredictable!

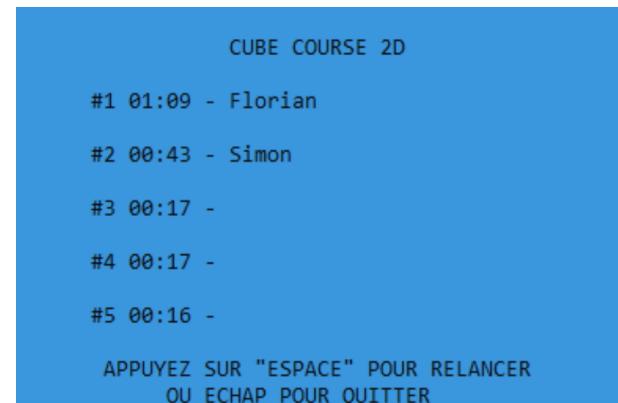
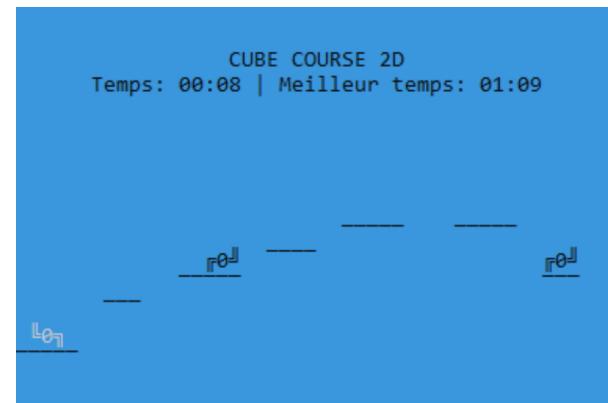
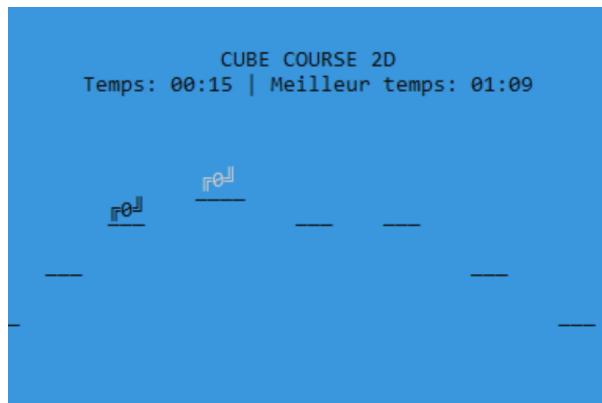
🎯 Objective

- 🏃‍♂️ **Survival:** Stay alive by jumping across platforms.
- ⚠️ **Random Challenges:** Platforms and enemies appear randomly, keeping you on your toes.
- 🏆 **Competition:** Test your reflexes and beat your high scores!

📝 Features

- 🎨 **Minimalist Design:** A simple yet effective aesthetic for full immersion.
- 🎲 **Random Generation:** No two levels are the same—every run is a new challenge!
- 🗡️ **Enemy Elimination:** Jump on enemies to defeat them, but be careful not to miss!
- 🎶 **Inspired by Geometry Dash:** A fast-paced and addictive gameplay experience.

📷 Screenshots



VATN-WaterGame



💧 A strategic game on water management!

VATN (from the Old Norse word for “water”) is a game designed to raise awareness about water management. Players must make critical daily decisions to address major water-related issues in their country, influencing key parameters that determine the nation’s survival.

🎯 Purpose

- 💧 **Water Management Awareness:** Educate players on the importance of sustainable water policies.
- 🎮 **Engaging Decision-Making:** Every day presents a new challenge that affects the nation’s status.
- 🏆 **Survival & Strategy:** Keep your country alive as long as possible by maintaining stability.
- 📊 **Progress Tracking:** Visualize the evolution of key parameters through in-game graphs.

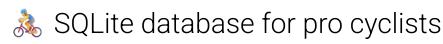
📝 Features

💡 Feature	🔍 Description
🌐 Game Type	Strategic Decision-Making Simulation
📅 Daily Choices	Players make decisions each day affecting country parameters
📈 Dynamic Statistics	Key indicators fluctuate based on player actions
💀 Game Over	The country collapses if the population reaches 0
📁 Save & Load	Resume previous games using saved files
🥇 Rankings	Compare results with previous local games
📊 Graphical Summary	Track country status evolution over time

📺 Gameplay Overview

🎮 Main Menu	📊 In-Game Statistics	🏆 Endgame Rankings
VATN ▶ Lancer Nom utilisateur ⬇️ Reprise de partie 📊 Classement	VATN: Jour X Eau disponible : 10L Eau nécessaire : 10L Population : 60 millions Satisfaction population 100% Electricité : 100% Habitat : 100% Nourriture : 100% ⚠️ Evenement XXX ⚠️ A: ----- B: ----- C: ----- D: -----	VATN 1 PLAYER 9564 2 PLAYER 8564 3 PLAYER 7564 4 PLAYER 6564 5 PLAYER 5564

WorldTourRidersDatabase



WorldTourRidersDatabase is an SQLite-based project designed to manage professional cyclists and their teams affiliated with the International Cycling Union (UCI). The database enables structured data storage, querying, and management of riders and their associations with World Tour teams.

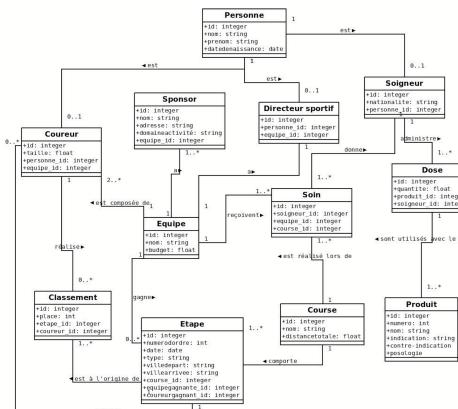
Purpose

-  **Data Storage:** Efficiently store and manage rider information within an SQLite database.
 -  **Advanced Queries:** Utilize SQL queries to retrieve, filter, and analyze cycling-related data.
 -  **Table Associations:** Implement relational connections between riders, teams, and relevant cycling entities.
 -  **UML Modeling:** Design and visualize the database schema using UML modeling techniques.

Features

Feature	Description
Rider Management	Store and organize details about professional cyclists
Team Associations	Link riders to their respective World Tour teams
SQL Queries	Perform complex queries to extract meaningful insights
Database Normalization	Ensure efficient and scalable data structure
SQLite Integration	Fully functional within an SQLite environment
UML Diagrams	Visualize table relationships for better understanding

Database Structure & Design



ZEthyDex



📱 Android app like a Pokédex

An open-source application designed in the style of a Pokédex, allowing users to explore and catalog various creatures and features. This project aims to provide a fun and interactive way to engage with the Pokémon universe.

🎯 Purpose

- 📍 **Cataloging:** Create an extensive database of creatures with detailed information.
- 🎮 **Interactive Experience:** Offer a user-friendly interface for exploration.
- 🛠️ **Open Source Development:** Encourage collaboration and contributions from developers.

📝 Features

- 🌟 **User-Friendly Interface:** Easy navigation and search functionalities.
- 🔍 **Search Functionality:** Quickly find information on specific creatures.
- 📸 **Image Gallery:** Visual representation of each creature.

📷 Screenshots

The screenshots show the app's main screen, a detailed creature card, and an add-new creature screen.

Main Screen: Shows a list of creature cards with small images and captions. Captions include "Dernier Test Je crois en avoir fini", "Ça fait bcp de test là, nan ? Oui mais il faut tester", "Plusieurs Test Je réalise de nombreux test", "Test salé Ceci est le sel de Gérande", and "Test3 Ceci est toujours un test".

Detailed Creature Card: Shows a larger card for a creature named "ZEthy". It includes fields for "Description" (Je réalise de nombreux test), "Appartement" (567), "Boisson favorite" (Boisson de luxe), and a trash bin icon. At the bottom, it shows "Avancement du ZEthyDex 100%" with a progress bar.

Add-New Creature Screen: A form to input creature details. It has fields for "Nom du ZEthy" (with placeholder "Nom du ZEthy"), "Description" (with placeholder "Description"), "Appartement" (with placeholder "Appartement"), "Boisson favorite" (with placeholder "Boisson favorite"), and a "CHARGER IMAGE" button with a placeholder image of a dragon-like creature. A large "CONFIRMER" button is at the bottom.

learning



🌐 Jekyll site to archive my knowledge

An open-source website created using Jekyll and the “al-folio” theme, designed to archive my learning journey and share knowledge with anyone interested. This project serves as a personal portfolio and a resource hub.

🎯 Purpose

- 📘 **Knowledge Sharing:** Documenting and sharing what I've learned.
- 🌐 **Personal Portfolio:** Showcasing projects and skills.
- 🛠️ **Open Source Development:** Encouraging contributions and feedback from the community.

🌟 Features

- 📄 **Responsive Design:** Optimized for various devices.
- 🔍 **Search Functionality:** Easily find content and resources.
- 🛠️ **Customization Options:** Adapt the site to your preferences.



📄 Simple LaTeX template for ENIB

A beautiful, simple, and clean LaTeX template designed for ENIB students who want to write project reports. If you like the template, feel free to give it a star!

🎯 Purpose

- 👉 **Streamlined Reporting:** Facilitate the creation of professional-looking project reports.
- 🎓 **Student Resource:** Tailored specifically for ENIB students' needs.
- 🛠️ **Open Source Development:** Encourage collaboration and customization from users.

📝 Getting Started

To use this template, I recommend following [this tutorial](#).

After that, simply write your LaTeX code in the `body.tex` file.

