

How to hack a brainwave EEG toy - Force Trainer II

What is a NeuroSky Brainwave?

NeuroSky Brainwave is a technology that uses EEG (Electroencephalography) sensors to detect electrical activity in the brain. The NeuroSky chip processes these brain signals and translates them into digital data, which can be used in applications like gaming, meditation tracking, and brain-computer interfaces (BCIs).

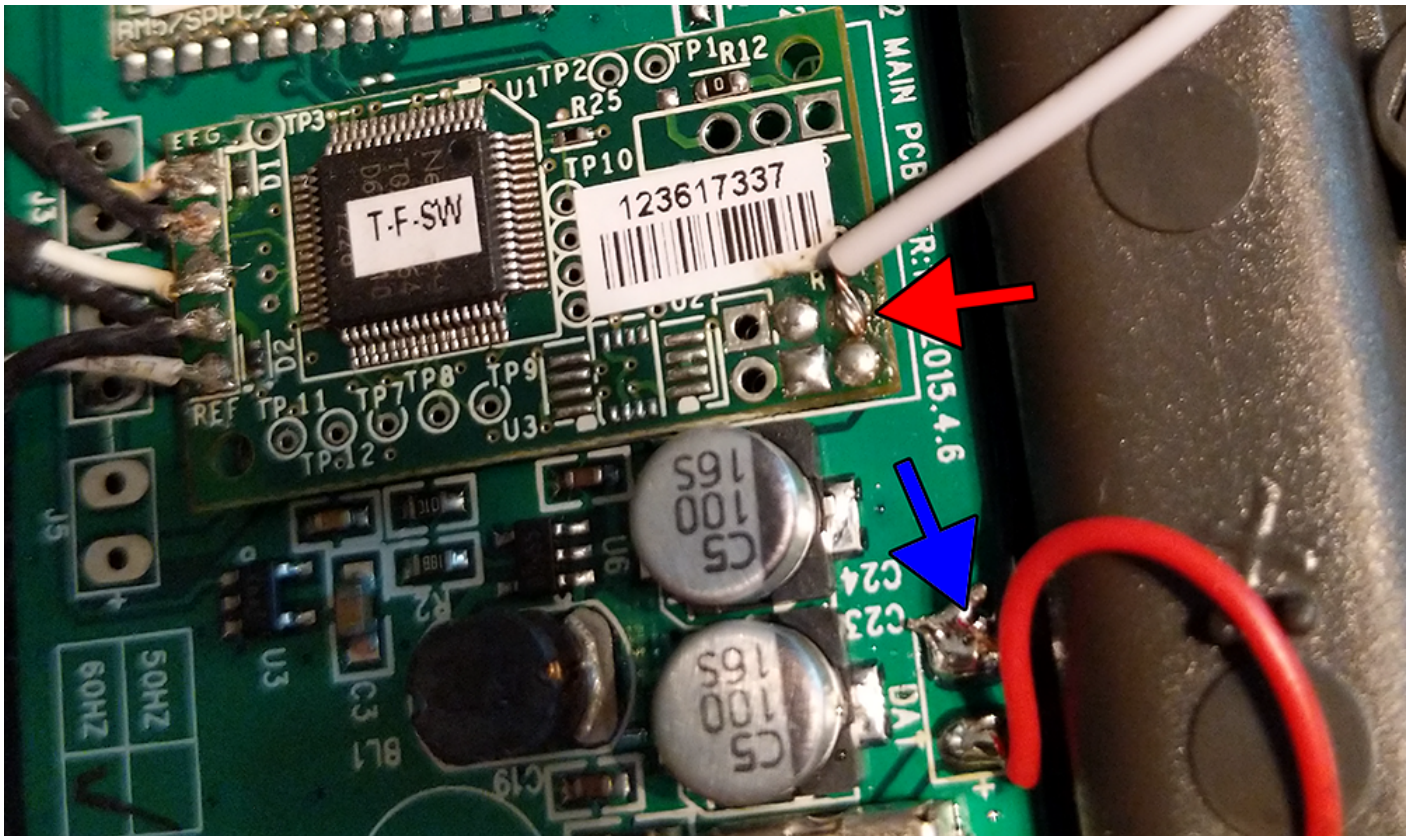
Although the actual headset is quite pricey, the NeuroSky chip can be found on brainwave-controlled toys such as the Star Wars Force Trainer. The toy can be hacked with a arduino. This tutorial is based on JimRosKind's [tutorial](#).



Wiring

In this tutorial, we are using an UNO, hardware Serial (pins 0 & 1) is shared with the USB connection, therefore we need to use software serial to communicate with the NeuroSky chip.

1. Solder a wire from the **T pin (red arrow)** (red circle & white wire) on the chip and connect that wire to **pin 2** on Arduino
2. Solder a wire from the **- (blue arrow)** on the chip next to the battery and connect that wire to **GND** on Arduino.



Getting started

The following is a sketch that will print out the signal quality, attention and meditation data.

```
#include <SoftwareSerial.h>

// Define the RX/TX pins for SoftwareSerial
SoftwareSerial mindwaveSerial(2, 3); // RX, TX

void setup() {
  Serial.begin(57600);    // Serial monitor baud rate
  mindwaveSerial.begin(57600); // NeuroSky baud rate
}

void loop() {
  readMindwave();
}

void readMindwave() {
  static int state = 0;
  static byte payload[256];
  static int payloadLength = 0, payloadIndex = 0;
  static byte checksum = 0;
```

```
while (mindwaveSerial.available()) {  
    byte byteRead = mindwaveSerial.read();  
  
    switch (state) {  
        case 0: // Waiting for first sync byte (0xAA)  
            if (byteRead == 0xAA) state = 1;  
            break;  
  
        case 1: // Waiting for second sync byte (0xAA)  
            if (byteRead == 0xAA) state = 2;  
            else state = 0;  
            break;  
  
        case 2: // Read payload length  
            if (byteRead > 169) { // Invalid length, reset  
                state = 0;  
            } else {  
                payloadLength = byteRead;  
                payloadIndex = 0;  
                checksum = 0;  
                state = 3;  
            }  
            break;  
  
        case 3: // Read payload  
            payload[payloadIndex++] = byteRead;  
            checksum += byteRead;  
            if (payloadIndex == payloadLength) state = 4;  
            break;  
  
        case 4: // Verify checksum  
            checksum = ~checksum & 0xFF; // Compute checksum  
            if (checksum == byteRead) {  
                processPayload(payload, payloadLength);  
            }  
            state = 0;  
            break;  
    }  
}
```

```
void processPayload(byte *payload, int length) {  
  for (int i = 0; i < length; i++) {  
    byte code = payload[i];  
  
    if (code == 0x02) { // Signal Quality  
      int signalQuality = payload[++i];  
      Serial.print("Signal Quality: ");  
      Serial.println(signalQuality);  
    }  
    else if (code == 0x04) { // Attention  
      int attention = payload[++i];  
      Serial.print("Attention: ");  
      Serial.println(attention);  
    }  
    else if (code == 0x05) { // Meditation  
      int meditation = payload[++i];  
      Serial.print("Meditation: ");  
      Serial.println(meditation);  
    }  
  }  
}
```

Revision #4

Created 4 March 2025 17:17:55 by Joanne Leung

Updated 6 March 2025 15:40:42 by Joanne Leung