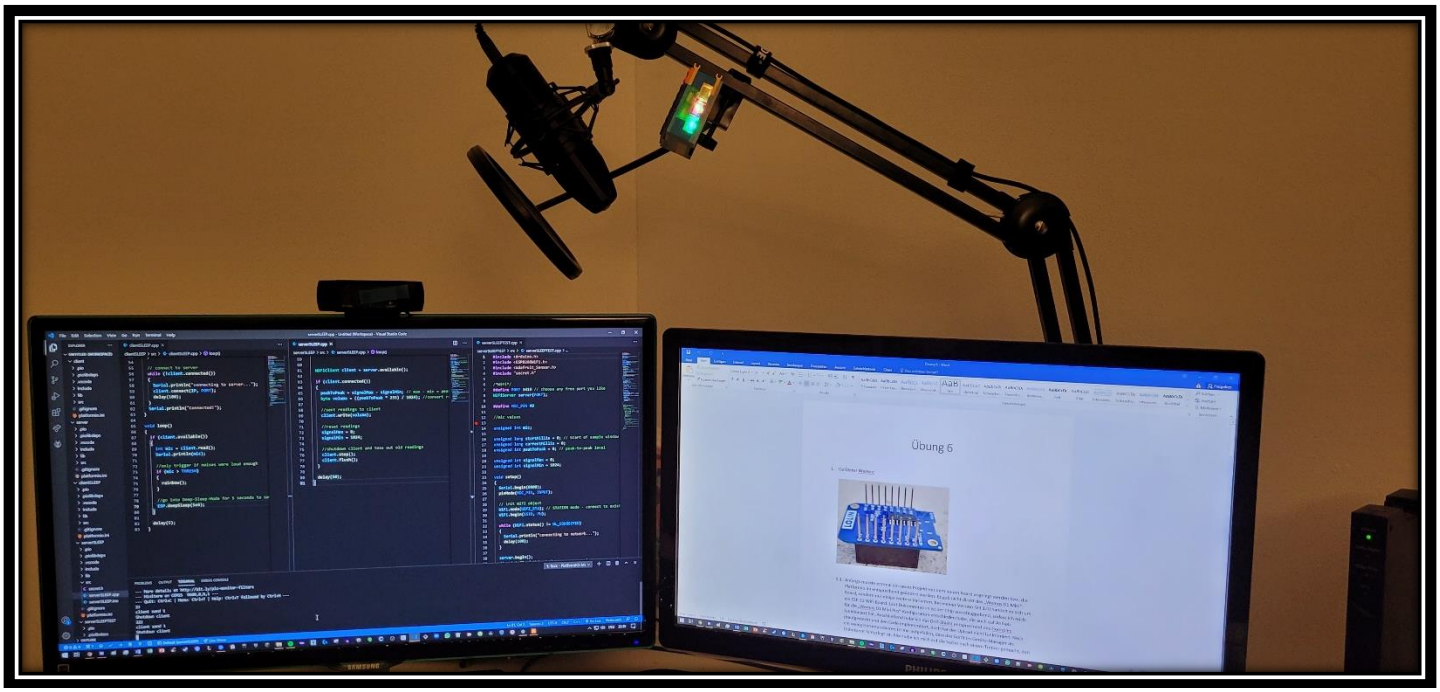
Projektbeschreibung

C²ube – die visuelle Klingel



Team
Maximilian Schlenczek

Konzept

Als Headset-Liebhaber hatte ich immer wieder dasselbe Problem: Egal ob beim vertieften Arbeiten, beim Musikhören oder beim Gamen, ich habe die Klingel nicht gehört. Ziel meines Projektes war es, eine Lösung zu finden, die sich spielerisch in die Schreibtischumgebung integriert. Mein Prototyp erkennt, wenn es klingelt und fängt an zu leuchten, sodass der Nutzer direkt reagieren kann. Auch wenn er die Klingel selber gar nicht gehört hat.

Implementierung

Der Prototyp besteht aus zwei Cubes: der Black-Box, einem stationären Modul, dass in der Nähe der Klingel angebracht wird. Und dem Note-Cube, der sowohl per Magnet befestigt werden kann, aber dank batteriebetrieb auch mobil überall mithin genommen werden kann. Bei der Black-Box erfasst ein Mikrophon Umgebungsgeräusche und sendet diese drahtlos per Wifi an den Note-Cube. Dieser hat einen LED-Streifen integriert, der, wenn es klingelt, anfängt in einem Regenbogenmuster zu leuchten. Ansonsten bleibt er ausgeschaltet.

Status, Erweiterungsmöglichkeiten

Mein Prototyp ist prinzipiell überall einsetzbar, wo Lautstärke eine Benachrichtigung auslösen soll. Allerdings muss er für jede Umgebung individuell eingestellt werden. Man braucht Richtwerte für die Lautstärke und der Umgebung, um eine zuverlässige Benachrichtigung zu ermöglichen und Fehler zu vermeiden. Außerdem haben beide Cubes nur eine begrenzte Reichweite, es muss also ein einigermaßen starkes WLAN-Signal verfügbar sein.
Erweiterungen bzw. Anpassungen sind viele möglich. Ich wollte ein Signal, dass mich bei vertieften Aufgaben am wenigsten stört. Für mich ist das ein visuelles Signal. Man kann aber auch ein akustisches Signal wiedergeben oder den Note-Cube vibrieren lassen. Sollte man eine digitale Lösung suchen, kann man sich z.B. auf dem Desktop direkt benachrichtigen lassen oder eine Mail schicken lassen, sodass man auch mobil Zugriff darauf hätte.

Bedienungsanleitung

C²ube – die visuelle Klingel

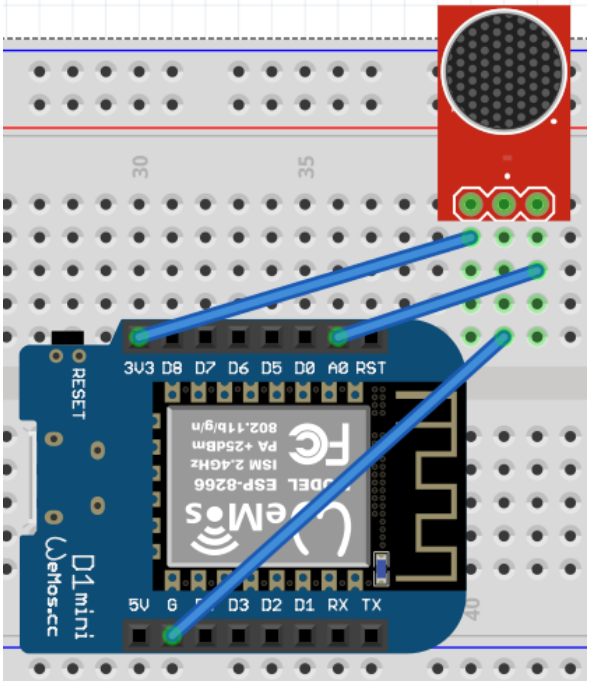
Setup
Beide Cubes brauchen eine Internetverbindung und müssen mit ausreichender WLAN-Signalstärke aufgestellt werden.
Die Black-Box ist Kabel gebunden. Sie kann durch den Travel Adapter entweder normal über die Steckdose oder über jeden USB-Anschluss betrieben werden. Die Note-Box hat eine integrierte Batterie und sollte sofort benutzt werden können. Falls notwendig kann die Batterie mit einem Mini-USB-Kabel wieder aufgeladen werden. Der Ladestand kann seitlich anhand der LED-Farbe abgelesen werden

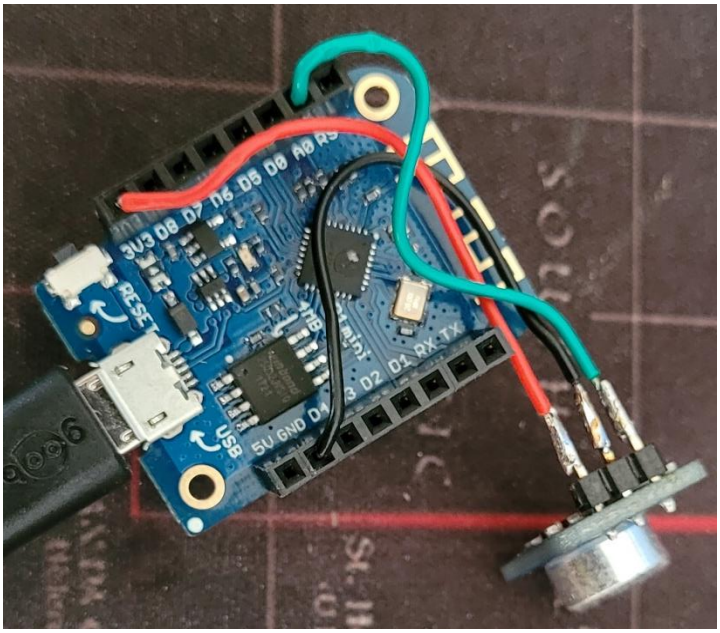
Bedienungsanleitung
Nach dem Setup ist der Prototyp ein „stiller Helfer“. Er kann problemlos mitgenommen werden, wenn man sich in der Wohnung umher bewegt. Ansonsten ist aber keine direkte Interaktion vorgesehen.

Konzept
Die Black-Box erfasst dauerhaft die Lautstärke aller Umgebungsgeräusche und speichert den höchsten (also lautesten) Wert ab, der nach einer bestimmten Zeit den Wert an den Note-Cube übertragen wird. Dieser vergleicht dann den übertragenen Wert mit einem vorher bestimmten Richtwert. Wird dieser überschritten, leuchtet der LED-Streifen. Nach der Abfrage werden alle alten Werte verworfen und die Erfassung beginnt von vorne. Der Note-Cube schaltet sich dann automatisch für eine bestimmte Zeit aus, um Energie zu sparen und die Akkulaufzeit zu erhöhen.

Implementierung
Der Prototyp besteht aus zwei Teilen: Der Black-Box, die als Server benutzt wird und dem Note-Cube, der als Client dient.

Black-Box:
Die Black-Box besteht aus besteht aus einem Wemos D1 Mini und einem Mikrofon:





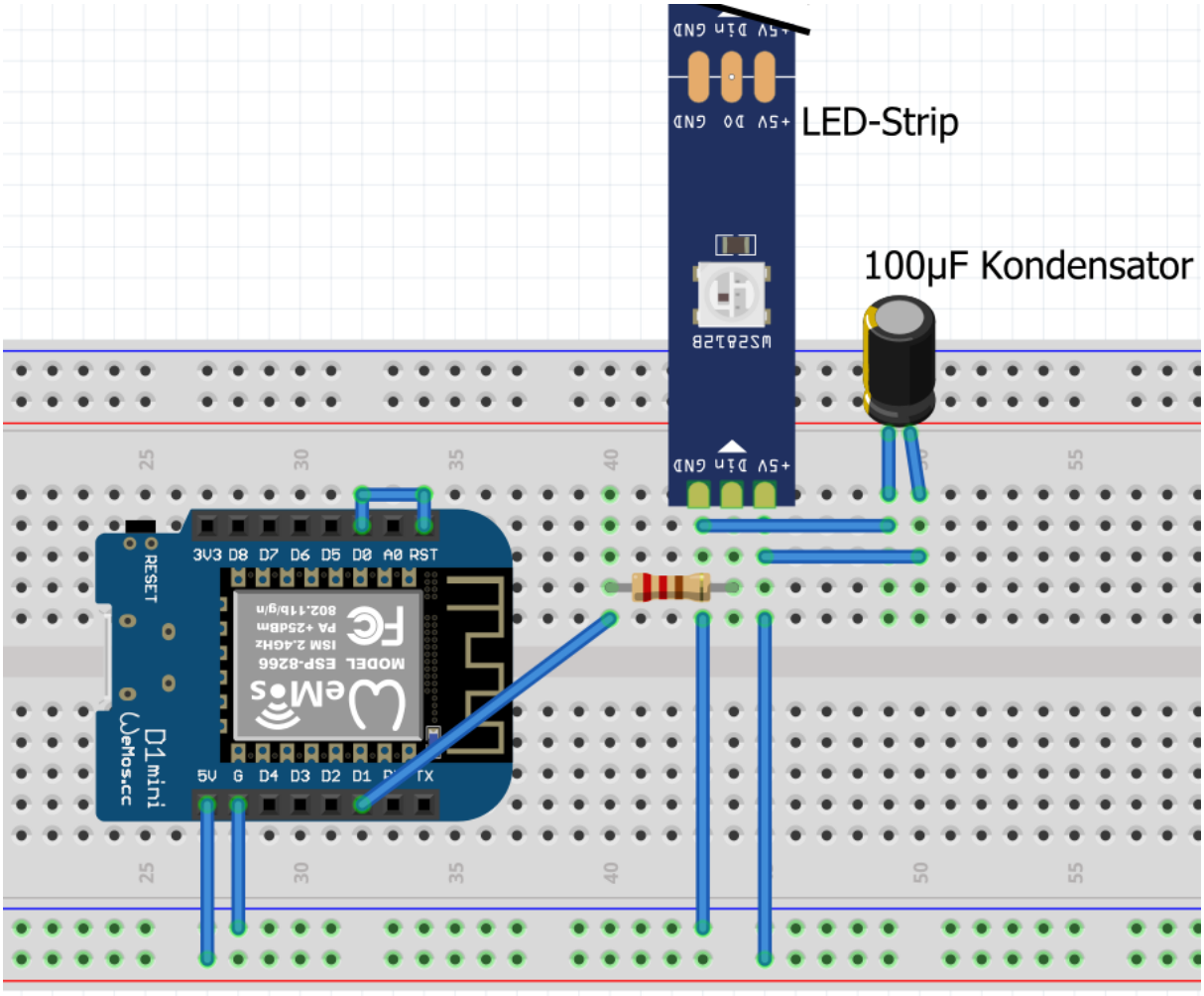
Wichtig dabei ist, dass der Output-Pin des Mikrophons an einen Analog-Pin angeschlossen wird und die „leiseste“ Stromversorgung gewählt wird, da am Out-Pin eine Gleichstromvorspannung von Vcc/2 anliegt. Ich habe ein MAX4466 Mikrophon verwendet, dass mit 2.4-5.5V betrieben werden kann. Beim Wemos wäre die leiseste Variante 3.3V. Strom bekommt der Wemos über den integrierten Mini-USB-Port.

Der Server übernimmt außerdem die Umrechnung der gesammelten Mikrofon-Werte. Da das MAX4466 Frequenz wahrnimmt und diesen mit einem entsprechenden Wert zwischen 0 und 1024 wiedergibt, müssen die Werte noch in Lautstärke umgerechnet werden. Dazu wird die Amplitudenhöhe der einzelnen Frequenzen genommen und gegeneinander gerechnet. Das wird solange gemacht, bis sich ein Client verbindet. Hat sich ein Client verbunden, wird der höchste Lautstärkewert dann auf einen Byte-Wert gemappt und anschließend an den Client gesendet:

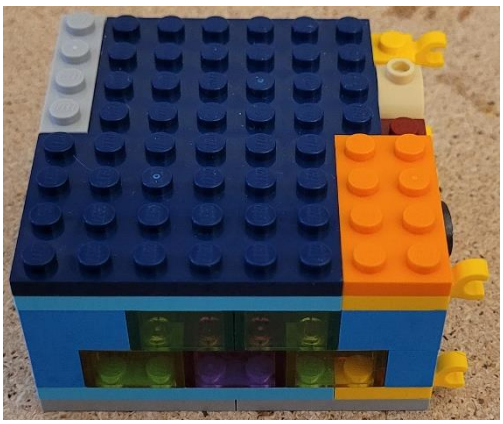
```
void loop()
{

  mic = analogRead(MIC_PIN);

  if (mic < 1024) // toss out spurious readings
  {
    if (mic > signalMax)
    {
      signalMax = mic; // save just the max levels
    }
    else if (mic < signalMin)
    {
      signalMin = mic; // save just the min levels
    }
  }


  WiFiClient client = server.available();

  if (client.connected())
  {
    peakToPeak = signalMax - signalMin; // max - min = peak-peak amplitude
    byte volume = ((peakToPeak * 255) / 1024); //convert readings between 0 and 1024 to byte rep

    //sent readings to client
    client.write(volume);
```

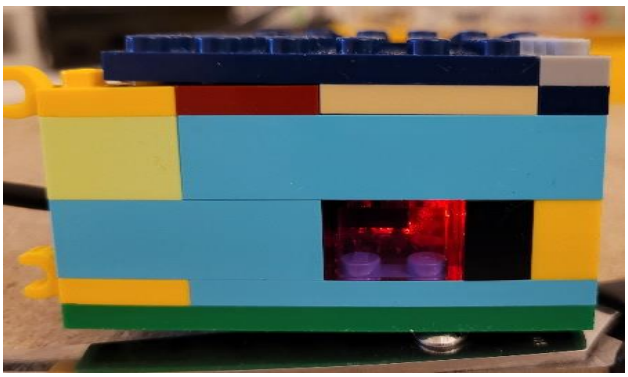
Anschließend wird alles wieder auf 0 gesetzt und alle vorhandenen Werte verworfen. Durch das Verwerfen können einige Messungen nicht berücksichtigt werden. Die Differenz zwischen verwerfen und erneut lesen liegt aber bei maximal 100ms. Meine Klingeldauer liegt 2s bzw. 2000ms. Somit wird ein Klingeln definitiv korrekt erkannt.
Die Server-Schaltung ist relativ klein und hat kaum Größenbeschränkungen. Sie kann problemlos in den meisten Gehäusen verbaut werden.

Note-Cube:
Der Note-Cube besteht ebenfalls aus einem Wemos D1 Mini. Hier wird allerdings zusätzlich ein Battery-Shield aufgesteckt. Des Weiteren wird eine LED-Strip-Schaltung und der Pin D0 muss direkt mit dem RST-Pin verbunden werden. An das Battery-Shield wird dann für den Betrieb auch eine Batterie angeschlossen. Ich persönlich habe eine 720mAh-Batterie benutzt. Damit die Batterielaufzeit verlängert werden kann, wird der Wemos nach einer erfolgten Abfrage in den Deep-Sleep-Mode versetzt. Der designierte Aufwach-Pin ist D0. Daher wird dieser auch direkt mit dem Reset-Pin verbunden. Bevor der Wemos in Deep-Sleep-Mode geht, wird der empfangene Wert noch mit dem festgelegten Schwellwert verglichen und nur falls die wahrgenommene Lautstärke über dem Schwellwert liegt, leuchtet der LED-Strip auf. Der Schwellwert variiert dabei sehr stark an verschiedenen Positionen. Um zuverlässige Ergebnisse zu erhalten, müssen vor dem aktiven Gebrauch des Prototypen Richtwerte erhoben werden. Da unterschiedliche Geräusche unterschiedlich stark wahrgenommen werden und Vibrationen sehr starken Einfluss auf die Messwerte haben, kann nur durch eigenes Testen ein passender Schwellwert herausgefunden werden. Dabei muss ganz besonders auf nicht regelmäßige Ereignisse geachtet werden. So haben mir die Waschmaschine als auch die Geräusche der Nachbarn im Treppenhaus dafür gesorgt, dass ich die Black-Box nicht direkt an der Klingel befestigen konnte, sondern sie ca. 90cm entfernt in der Garderobe untergebracht habe. Der einzig übrig gebliebene Störfaktor ist hier, wenn ich selber den Flur betrete und entsprechend laut bin. Allerdings bin ich dann auch im direkten Umfeld der Tür; ich werde ein Klingeln also definitiv selber mitbekommen. Auch der minimale Batterie-Mehrverbrauch ist hier zu vernachlässigen. Die Schaltung dazu sieht wie folgt aus:



Bei der Client-Schaltung spielt die Größe eine sehr wichtige Rolle. Das fertige Gerät soll transportabel sein, braucht eine Möglichkeit die Lichtimpulse nach außen zu tragen und muss aufladbar sein, ohne jedes Mal das gesamte Gerät auseinander zu bauen. Außerdem ist ein Indikator für den Batterie-Ladezustand hilfreich. Dementsprechend sollten die Schaltung und alle weiteren Teile möglichst klein verbaut werden und ein passendes Gehäuse ausgewählt werden. Ich habe das Gehäuse selber aus Lego gebaut. Überall dort, wo Licht wichtig ist, habe ich durchsichtige Steine verwendet und für den Lade-Port habe ich eine Tür benutzt, die ganz einfach geöffnet und geschlossen werden kann. Für meine Zwecke habe ich außerdem einen Magneten verbaut, mit dem ich den Cube an meinen Mikrofon-Arm (siehe erstes Bild) befestigen kann:



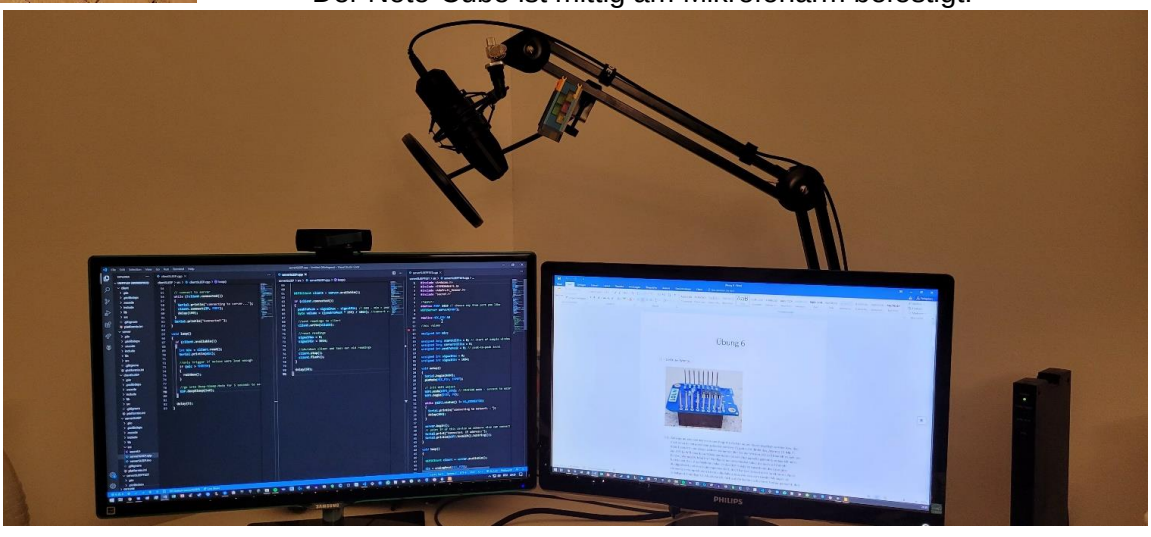


Die Befestigung besteht aus zwei kleinen Magneten. Einer davon ist in einem mittelgroßen Lego-Rad eingespannt und anschließend in einem Fenster verkeilt. Der zweite Magnet wird auf die andere Seite des Fensters vorgesetzt, um eine Erhöhung zu schaffen:

Abschließendes Setup
Hier noch einmal abschließend die Positionen der beiden Cubes. Außerdem ist die Position der Klingel mit eingezeichnet:

Links unten ist die Black-Box, rechts neben der Tür die Klingelanlage.

Der Note-Cube ist mittig am Mikrofonarm befestigt:



Dokumentation

C²ube – die visuelle Klingel

Code

Der vollständige Code für Client und Server kann in diesem Github-Repository gefunden werden. WICHTIG: Die Datei „secret.h“ fehlt absichtlich. Sie enthält sensible Netzwerk-Daten. Damit der Code vollständig funktioniert, müssen die SSID des Netzwerks, das Netzwerk-Passwort für Client und Server sowie eine IPAddress IP für den Server erstellt werden.

Bauteile

Nachfolgend sind alle großen Bauteile, die ich benutzt habe, mit den entsprechenden Datasheets aufgelistet:

- Adafruit Max4466: https://www.makershop.de/module/audio/adafruit-max4466/
- Wemos Battery Shield: https://www.wemos.cc/en/latest/d1_mini_shield/battery.html
- Wemos D1 Mini: https://www.makershop.de/plattformen/d1-mini/d1-mini-pro/
- Kompatible Batterie: https://www.exp-tech.de/zubehoer/batterien-akkus/lipo-akkus/9365/3.7v-750mah-lithium-polymer-akku-mit-jst-ph-anschluss

Referenzen

Neben den Datasheets und Standard-Beispiele sind außerdem folgende Quellen zu empfehlen:

- Lautstärke messen: https://learn.adafruit.com/adafruit-microphone-amplifier-breakout/measuring-sound-levels
- Energieverbrauch: https://diyi0t.com/how-to-reduce-the-esp8266-power-consumption/
- Wemos Sleep Modes: https://www.mischianti.org/2019/11/21/wemos-d1-mini-esp8266-the-three-type-of-sleep-mode-to-manage-energy-savings-part-4/