Artikel für cqDL,
Autor: DJ0ABR, Kurt Moraw

LoRa APRS, iGate und Tracker bei DB0SL

DB0SL steht am Rusel-Absatz, dort wo Isar- und Donautal an die bayrischen Mittelgebirge grenzen. Der Einzugsbereich ist nach Süd, West und Nord mit 100 bis 150km Sichtweite sehr günstig. Mitglieder des Ortsverbands Landau, U07, Bayerwald, U02, des Vereins Relaisfunkstelle-Greising sowie weitere freischaffende Funkamateure haben seit einiger Zeit mit privaten LoRa APRS iGates experimentiert, welche an normalen 70cm Vertikalantennen zuhause betrieben werden. Dabei zeigte sich, dass speziell im städtischen Bereich nur geringe Reichweiten (1 bis 5 km) zu erzielen sind, da die Bebauung sehr stark dämpft. Zur Abhilfe sollte die Eignung von DB0SL als iGate untersucht werden. Zunächst wurde mit provisorischen Geräten die Funktion geprüft. Bereits wenige Stunden nach der Aktivierung des iGates DB0SL-10 konnte eine LoRa-APRS Station aus dem Münchener Stadtgebiet (DC1MBB, Distanz ca. 128km) gehört werden. Damit war die Entscheidung getroffen, das iGate fest zu installieren und auch interessierte Funkamateure mit entsprechenden LoRa-Trackern zu versorgen.


Bild 1: LoRa iGate bei DB0SL *(Dateiname: db0sl.jpg)*

Auswahl der Technik:

Wie bereits bei HamNET ist auch bei LoRa-APRS die Aktivität unserer Nachbarn in Österreich enorm. In den letzten Jahren wurde Hardware und Software entwickelt, die uns einen schnellen und einfachen Aufbau der Geräte ermöglicht. Bei der Suche nach passender Hardware findet man viele verschiedene Lösungen.

Unsere Anforderungen sind:
- Hardware einfach zu löten und aufzubauen
- Softwareinstallation ohne Programmierkenntnisse
- günstiger Preis

Diese Anforderungen erfüllt ein Projekt von OE5BPA, welches auf Github [1] quelloffen verfügbar ist und sowohl das iGate als auch den Tracker beinhaltet.

LoRa APRS iGate:

Als Hardware haben wir uns für das Board: Heltec WiFi LoRa 32 V2 in der 433MHz Version entschieden (Bild 2). Diese Platine wird fertig aufgebaut geliefert und hat standardmäßig ein kleines OLED-Display. Sie benötigt eine Stromversorgung von 5 Volt am Micro-USB Stecker. Der HF Anschluss ist als U.fl Stecker ausgeführt, der mit einem U.fl auf SMA Adapterkabel und weiter über einen SMA-N Adapter an die 70cm Antenne angeschlossen wird. Spezielle Filter waren bisher nicht erforderlich. Falls am gleichen Standort jedoch ein 70cm Relais betrieben wird, könnte ein passendes Notchfilter nützlich sein. Außer der Anschlusskabel und einer Antenne benötigt man nichts weiter um ein iGate aufzubauen. Die Kosten sind vergleichsweise gering und auch für kleine OVs kein Problem.

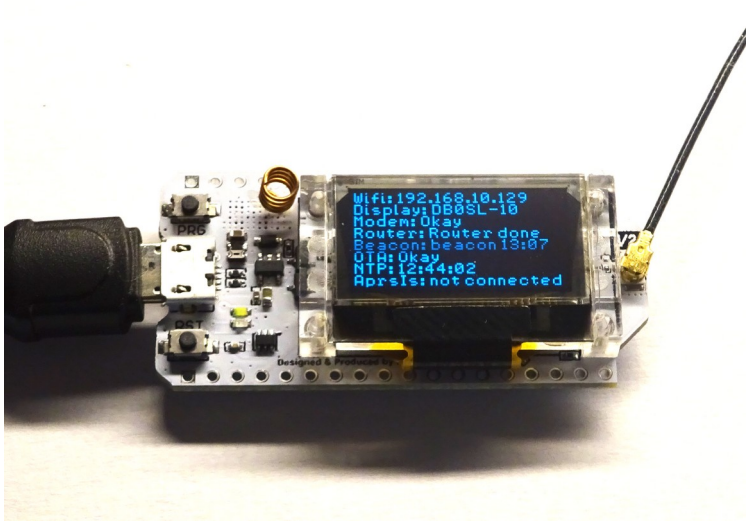

Bild 2: iGate DB0SL-10 *(Dateiname: igate.png)*

iGate - Firmware:

Die Firmware kann von [1] heruntergeladen werden. Sie läuft im Entwicklungssystem PlatformIO, welches als Erweiterung von Microsoft-Code installiert wird. Ich habe das System unter Ubuntu installiert. Das funktioniert problemlos mit den folgenden Schritten, weitere Informationen findet man unter [3]. Natürlich klappt das auch unter Windows, wobei man zuerst einen Sicherungspunkt erstellen sollte bevor man ein eher umfangreiches System mit Python installiert. Wer ganz sicher gehen will sein Windows nicht zu beschädigen macht das in einer virtuellen Maschine (z.B.: Virtual Box), damit ist man immer auf der sicheren Seite.
Zunächst wird Microsoft CODE installiert, es befindet sich auf [4] zum herunterladen. Nach der Installation startet man das Programm durch Eingabe von: code. Nach dem ersten Start wird man aufgefordert das deutsche Sprachpaket zu installieren. Sobald „code“ läuft, startet man den Erweiterungsmanager durch Klick auf das Symbol  *(Dateiname: ewm.png)*

Jetzt gibt man in die Suchzeile „platformio“ ein und sofort erscheint als erste Auswahl: PlatformIO IDE, welche man jetzt installiert. (Sollte eine Fehlermeldung kommen, so muss noch die passende Python Version durch Eingabe von: *sudo apt-get install python3-venv* installiert werden. Danach wiederholt man die Installation von PlatformIO).

Um das zuvor heruntergeladene Projekt zu starten, wählt man das Symbol  *(Dateiname: pio.png)* und dann PIO Home – Open. Das Startfenster wird angezeigt. Hier klickt man „Open Project“ und geht in das Verzeichnis mit den von github heruntergeladenen und entpackten iGate Dateien. Das Projekt wird jetzt geladen. Vor dem Flashen in die Lora-Platine muss es noch konfiguriert werden. Dazu öffnet man die Dateiauswahl mit dem Symbol  (*Dateiname: daw.png)* und öffnet die Datei data/is-cfg.json welche alle Einstellungen enthält. Man trägt das Rufzeichen des iGates ein, wobei üblicherweise die Erweiterung -10 für LoRa iGates benutzt wird. Diese Eingabe erfolgt einmal bei „callsign“ und nochmal bei „hostname“. Das iGate überträgt seine Daten via WiFi ins Internet. Im Abschnitt „wifi“ wird also die SSID und das Passwort des lokalen Wlans eingetragen. Im Abschnitt „beacon“ gibt man einen beliebigen Bakentext sowie Längen- und Breitegrad des Standorts ein (als volle Gradangabe, nicht in Grad und Minuten wie bei APRS sonst üblich). Als letzten Eintrag muss man in Rubrik „aprs-is“ das APRS-Passwort für dieses iGate angeben. Dieses Passwort bezieht sich auf das Rufzeichen des iGates. Wer ein iGate ernsthaft aufbauen möchte, wird wissen wie er zu diesem Passwort kommt.

Das Programm ist damit fertig zum flashen in die LoRa Platine, welche jetzt an USB angeschlossen wird. Unter Linux ist es eventuell notwendig Zugriffsrechte auf die Schnittstelle zu vergeben, entweder durch Setzen der Gruppe „dialout“, oder auf die schnelle durch Eingabe von: sudo chmod 777 /dev/ttyUSB0 (oder /dev/ttyACM0, je nach benutzter Hardware).

Jetzt klickt man das Pfeilsymbol, welches sich links unten in der Statuszeile befindet:  (*Dateiname: fl1.png)*. Das Flashen dauert nur ein paar Sekunden und sollte schnell erledigt sein. Jedoch hat es sich gezeigt, dass eine weitere Aktion erforderlich ist damit die Software zuverlässig läuft. Dazu klickt man wieder links auf  *(Dateiname: pio.png)* und wählt lora_board und Upload Filesystem Image. Ist dieser Vorgang beendet, so ist das iGate einsatzbereit.

Zunächst prüft man im Display ob eine IP-Adresse erscheint, ob die via NTP geladene Uhrzeit stimmt und ob eine Verbindung zum APRS-Server besteht (was einige Zeit dauern kann).

LoRa APRS Monitor:

für die iGate Platine gibt es eine alternative Firmware [6] zum Aufbau eines LoRa Monitors. Ähnlich wie das iGate empfängt auch diese Firmware die APRS Aussendungen und zeigt im Display die letzten 8 Rufzeichen zusammen mit der RSSI (Signalstärke in dBm) an.
Das ist ein kleiner und nützlicher Helfer bei der Prüfung von Trackern und zur Bewertung der Signalstärke, beispielsweise zur Antennenauswahl. Die Installation erfolgt mit der Arduino IDE und ist in der Projektbeschreibung dokumentiert.

LoRa APRS Tracker:

Auch hier haben wir das Projekt von OE5BPA übernommen, da es mit fertigen Komponenten auskommt und einfach aufzubauen ist.
Unterstützt wird unter anderem die Platine TTGO T-Beam Version 1.0 433 MHz, welche bei den üblichen Lieferanten für ca. 40 bis 60.- € erhältlich ist. Der Vorteil dieser Platine ist, dass sämtliche Komponenten für einen Tracker bereits enthalten sind, also uC, GPS und LoRa-Sender. Zusätzlich enthalten sind ein 18650 Akkuhalter, ein LiIon Ladegrät und ein OLED Display. Für einen Tracker benötigt man nur diese Platine, sonst nichts. Allerdings haben wir trotzdem ein paar Erweiterungen vorgenommen und eine neue GPS Antenne sowie eine optionale 1W PA mit PTT hinzugefügt.

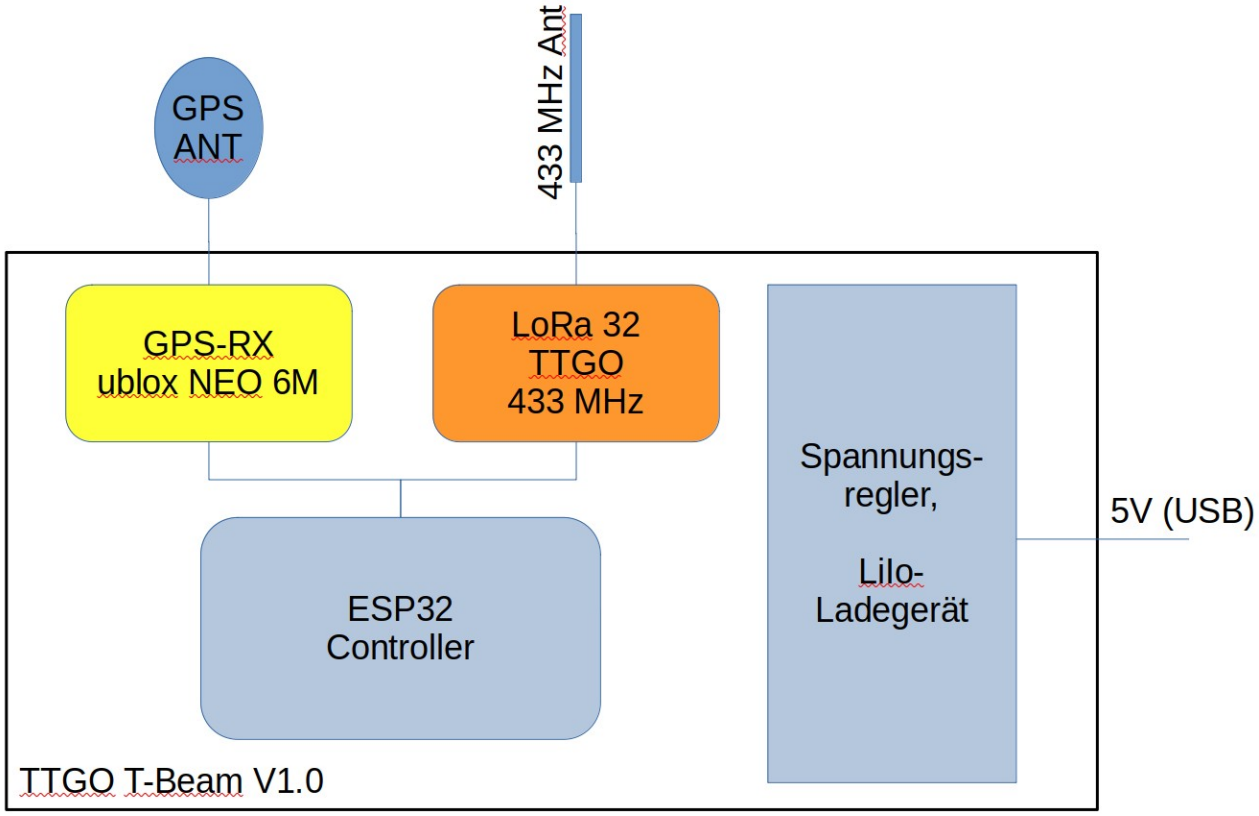

Bild 3: Blockschaltbild TTGO T-Beam *(Dateiname: ttgo.png)*

Der Tracker (Bild 3) besteht aus einem GPS Empfänger, einem LoRa Transceiver sowie einem ESP32 Controller (und noch einigen weiteren, unbenutzten Komponenten, wie z.B. Wifi). Die Ausgangsleistung auf 433,775 MHz beträgt 100mW. Damit lassen sich ähnliche Reichweiten erzielen wie bei herkömmlichem (Packet-Radio basiertem) APRS mit 10 Watt. Die TTGO T-Beam Platine zeigt Bild 4. Bei einer Ausgangsleistung von 100mW kann diese Platine ohne Änderungen betrieben werden.

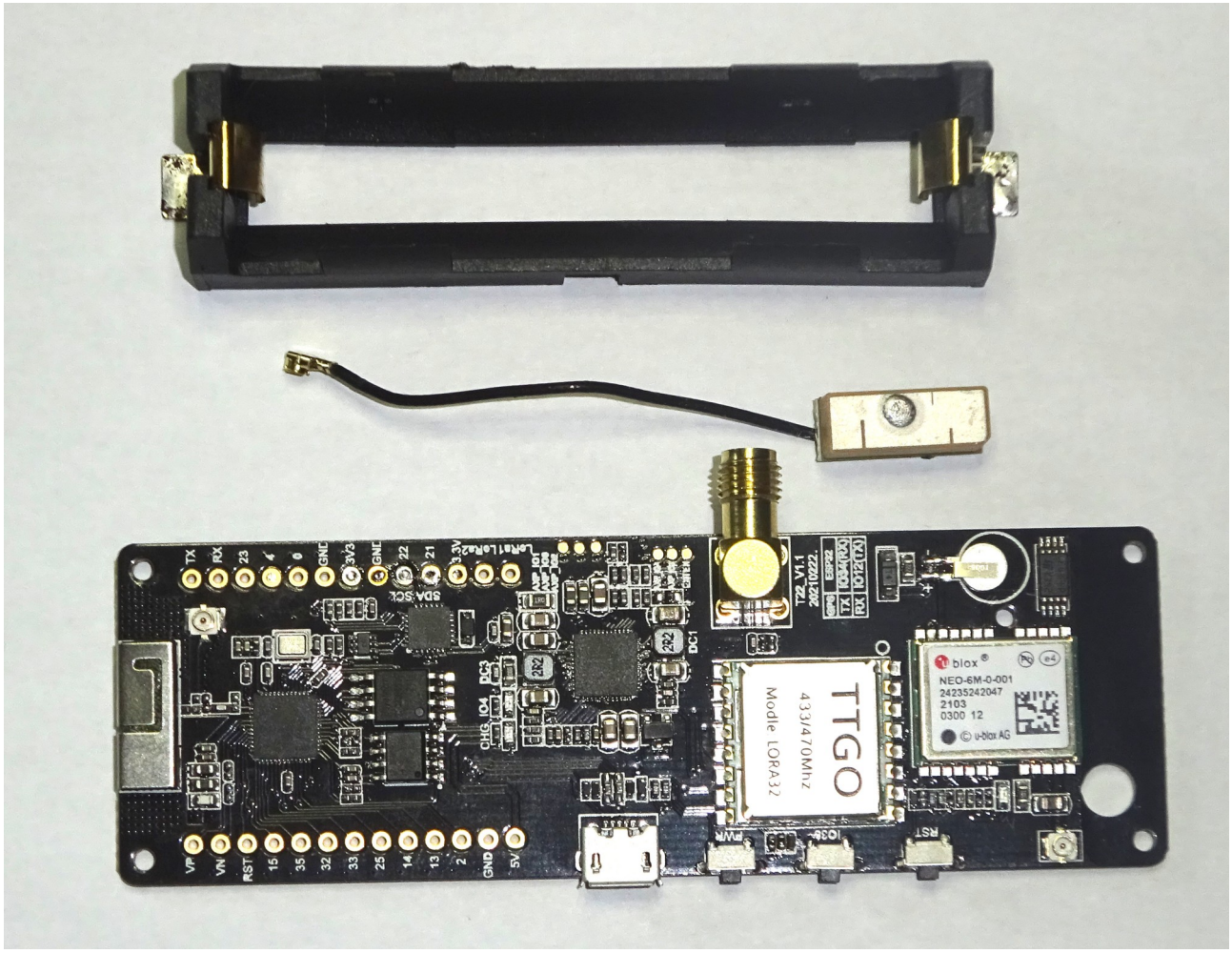

Bild 4: TTGO T-Beam V.10 *(Dateiname: ttgotbeam_1.png)*

Im Originalzustand ist auf der Rückseite der Platine ein Akkuhalter (LiIo 18650) aufgelötet, sowie eine kleine GPS Antenne befestigt. Die GPS Antenne muss man auf jeden Fall durch eine richtige Antenne ersetzen, da das kleine Teil kaum Empfang hat. Den Akku kann man für Portabelbetrieb belassen, für Betrieb im KFZ wird er nicht benötigt, ich habe ihn daher entfernt. Der Tracker kann über ein USB Kabel an einem 5V Adapter im 12V Stecker des KFZs betrieben werden. Die Stromaufnahme liegt bei ca. 100mA. Bei montiertem Akku wird dieser geladen, da die Platine über ein LiIo Ladegerät verfügt, wobei der Ladestrom ca. 400 bis 800mA beträgt.

Wenn man mit einer zusätzlichen 1 Watt PA an einer Gummiantenne arbeiten will, so muss der TTGO T-Beam in ein Metallgehäuse eingebaut werden, da die Platine nicht einstrahlfest ist und beim Senden abstürzt.

Wir haben uns entschieden aus Leiterplattenstücken eine Schirmung zu bauen. Das sieht zwar nicht schön aus, ist aber HF-mäßig ideal, da man viele Punkte direkt mit Masse verlöten kann. Danach kommt das ganze sowieso in ein 3D gedrucktes Gehäuse. +5V und PTT sind über 1nF Durchführungskondensatoren herausgeführt. Die Schirmung hat einen Boden, aber keinen Deckel. Bei der Sendeleistung mit 1 Watt waren nun keine Störungen mehr vorhanden.

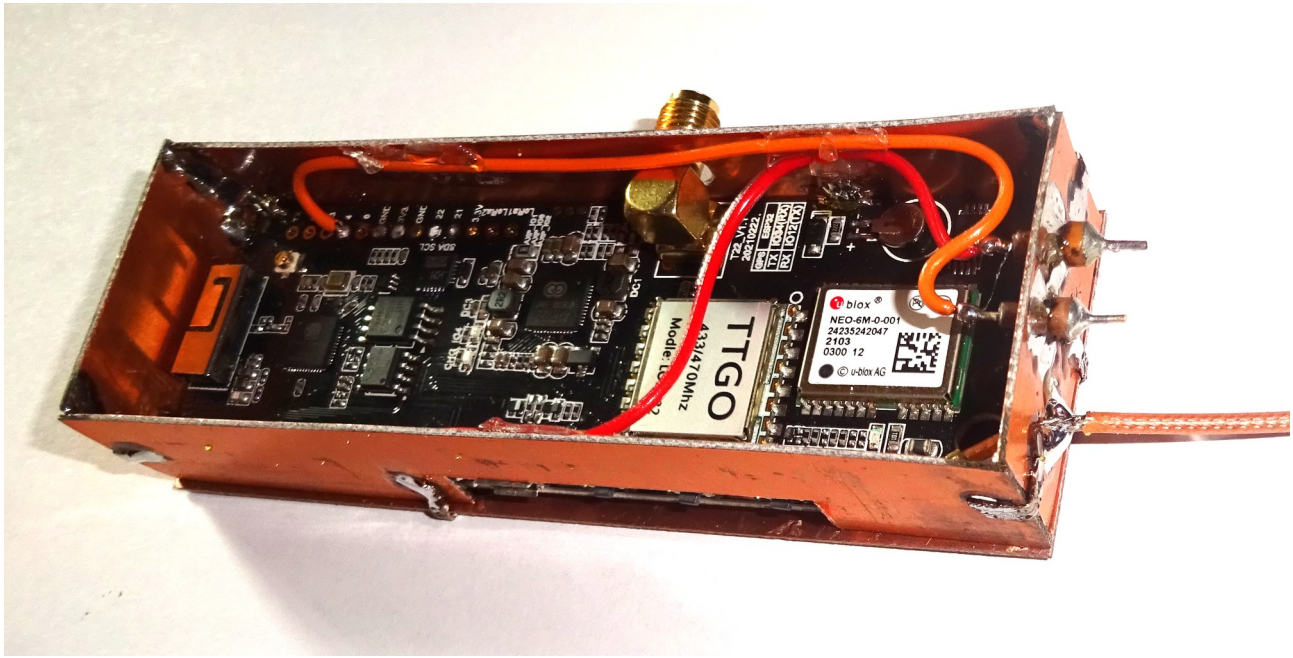

Bild 5: Schirmung aus Leiterplatten *(Dateiname: ttgotbeam_2.png)*

70cm PA, 1 Watt:

auch mit 100mW funktioniert der Tracker bereits sehr gut. Durch eine Leistungserhöhung auf 1 Watt hat man eine fast lückenlose Abdeckung auch unter schwierigen Bedingungen oder bei Benutzung einer winzigen Gummiantenne.
Für kleine Leistungen im 70cm Band, benutzt man heute fertige HF-Verstärker ICs im SMD Gehäuse. Diese brauchen nur mehr eine einfache Anpassung und eine Speisedrossel und liefern bereits bei 5 Volt die angegebene Ausgangsleistung, welche wie üblich, bei ca. 25% der von den fernöstlichen Herstellern angegebenen Leistung liegt. Im Internet findet man günstige fertige Platinen mit Kühlkörper (Bild 6) [5].

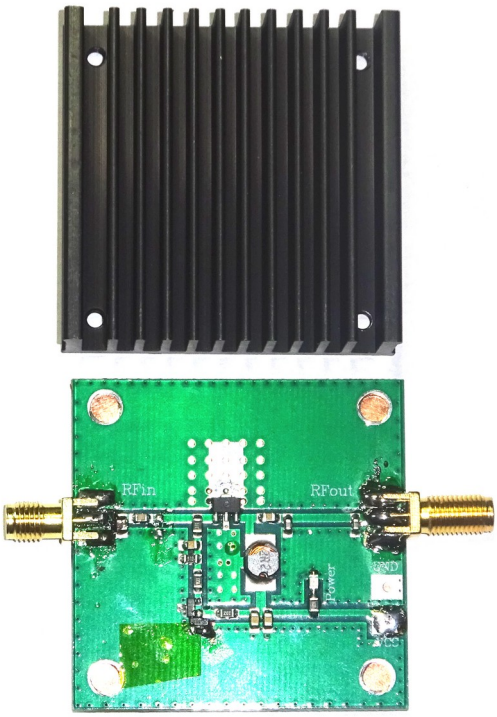

Bild 6: 70cm PA *(Dateiname: pa_1.png)*

Diese PA hat einen Ruhestrom von 200mA und benötigt daher normalerweise einen Kühlkörper. Im APRS Betrieb mit nur kurzen Aussendungen ist die Erwärmung vernachlässigbar, der Kühlkörper kann also entfernt werden. Stattdessen könnte man ein dünnes Kupferblech auf die Unterseite löten. Damit der Ruhestrom nur im Sendebetrieb fließt, muss die PA mit einer PTT Schaltung ergänzt werden.

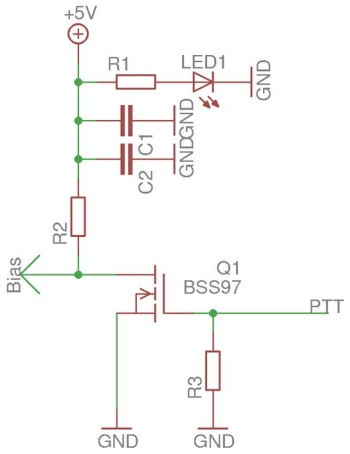

Bild 7: PTT (Dateiname: bias.png)

Bild 7 zeigt die Prinzipschaltung der Vorspannungserzeugung (Bias) dieser Endstufen. Man ergänzt diese Schaltung mit dem Mosfet Q1 (beliebiger kleiner N-Kanal Mosfet) sowie einem Gate-Widerstand von 10kOhm. Solange nicht gesendet wird, muss die PTT Leitung auf 3V oder mehr liegen. Dadurch schließt Q1 die Vorspannung kurz und es fließt kein Ruhestrom. Die TTGO T-Beam Platine gibt das PTT Signal an Pin 4 aus, die Polarität kann in der OE5BPA Software eingestellt werden, siehe Datei tracker.json:

```
„ptt_output“: {
        „active“: true,
        „io_pin“: 4,
        „start_delay“: 250,
        „end_delay“: 100,
        „reverse“: true
}
```

Tracker – Firmware:

die Tracker-Firmware wird genau so geflasht wie die iGate Firmware, obige Beschreibung ist daher auch hier gültig. Sobald man die Tracker Software von [2] herunterladen und in „code“ geladen hat, müssen auch hier ein paar Einstellungen vorgenommen werden. Diese befinden sich in der Datei: data/tracker.json

Im Abschnitt „beacons“ werden eingestellt „callsign“ und „message“. LoRa Rufzeichen bekommen üblicherweise die Erweiterung -7. Die Nachricht (message) ist ein beliebiger kurzer Text. Evt. möchte man noch das Symbol ändern (das Standardsymbol ist ein Fußgänger) und die PTT Steuerung (siehe 1W PA) aktivieren. Danach kann die Software in den TTGO T-Beam geflasht werden.
Sobald der GPS Empfänger einige Satelliten empfängt beginnt der APRS Tracker Positionsmeldungen zu senden. Man kann die Aussendung mit einem FM Empfänger auf 433,775 MHz prüfen oder man baut sich den beschriebenen LoRa Monitor [6].

Bild 8: der fertige 1-Watt Tracker *(Dateiname: tracker.png)*

Fazit:

Bereits wenige Stunden nach Installation des iGates wurden Stationen aus 100km und weiter gehört und ins APRS Netzwerk weitergeleitet. Dank des guten Link-Budgets der LoRa Modulation können auch kleine, akkubetriebene, Stationen mit nur 100mW einwandfreien APRS Betrieb machen, was das System für Wanderungen im bayrischen Wald ideal macht. Eine Akkuladung erlaubt den durchgehenden APRS Betrieb über 24 Stunden.

[1] … LoRa-APRS iGate, OE5BPA: https://github.com/lora-aprs/LoRa_APRS_iGate
[2] … LoRa-APRS Tracker, OE5BPA: https://github.com/lora-aprs/LoRa_APRS_Tracker
[3] … PlatformIO: https://platformio.org/
[4] … Microsoft Code: https://code.visualstudio.com/
[5] … 70cm PA Suchbegriff: „433 RF Leistungsverstärker 5 Watt SMA Stecker für 380-450“
[6] … LoRa-APRS Monitor: https://github.com/dj0abr/LoRa-APRS-Monitor