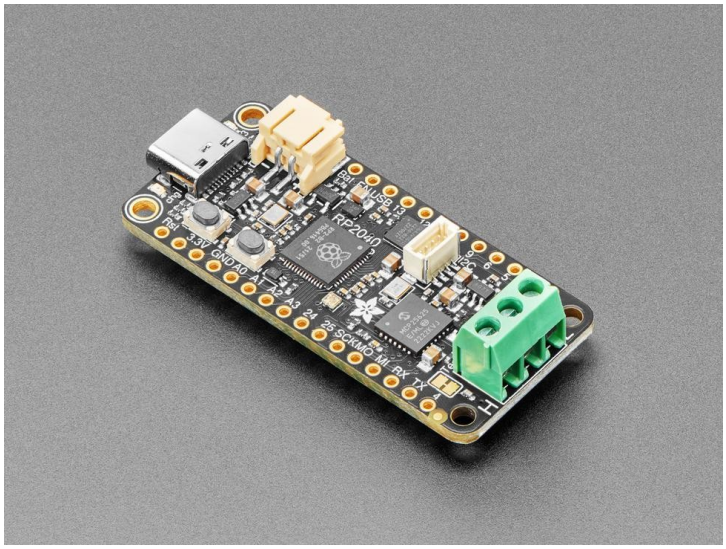
Datenblatt

Adafruit RP2040 CAN Bus Feather mit MCP2515 CAN Controller



Artikel-Nr.:	ADA5724
Hersteller:	Adafruit
Herkunftsland:	USA
Zolltarifnummer:	85359000
Gewicht:	0.004 kg

Wenn du schnell und ohne Löten mit dem CAN-Bus-Interfacing beginnen möchtest, ist unser RP2040 CAN Bus Feather von Adafruit mit einem Mikrocontroller, einem CAN-Chipsatz und Klemmenleisten sofort einsatzbereit. Der verwendete Controller ist der MCP25625 (auch bekannt als MCP2515 mit eingebautem Transceiver), ein äußerst beliebter und gut unterstützter Chipsatz der über Treiber in Arduino und CircuitPython verfügt und nur einen SPI-Port und zwei Pins für Chipselect und IRQ benötigt. Mit ihm kannst du Nachrichten im Standard- oder erweiterten Format mit bis zu 1 Mbit/s senden und empfangen.
Feather ist die Entwicklungsboard-Spezifikation von Adafruit, und wie sein Namensvetter ist es dünn, leicht und lässt dich fliegen! Wir haben Feather als neuen Standard für tragbare Mikrocontrollerkerne entwickelt.

CAN-Bus ist ein kleiner Netzwerkstandard, der ursprünglich für Autos und, ja, Busse entwickelt wurde, jetzt aber für viele Robotik- oder Sensornetzwerke verwendet wird, die eine bessere Reichweite und Adressierung als I2C benötigen und nicht über die Pins oder die Rechenleistung verfügen, um über Ethernet zu sprechen. CAN ist eine 2-Draht-Differenzialverbindung, die für große Entfernungen und laute Umgebungen geeignet ist.
Nachrichten werden mit einer Geschwindigkeit von etwa 1 Mbit/s gesendet - du legst die Frequenz für den Bus fest, und dann müssen alle "Joiner" damit übereinstimmen und eine Adresse vor dem Paket haben, damit jeder Knoten Nachrichten nur für ihn abhören kann. Neue Knoten können leicht angeschlossen werden, da sie nur an die beiden Datenleitungen irgendwo im gemeinsamen Netz angeschlossen werden müssen. Jedes CAN-Gerät sendet Nachrichten, wann immer es will, und kann dank einer cleveren Datenkodierung erkennen, wenn es eine Nachrichtenkollision gibt, und diese später erneut senden.
Wir haben dem Feather ein paar nette Extras hinzugefügt, die ihn in vielen gängigen CAN-Szenarien nützlich machen:

- **5V-Ladungspumpen-Spannungsgenerator**, d.h. auch wenn du 3,3V auf der Feather-Platine verwendest, erzeugt sie saubere 5V, wie sie der Interal-Transceiver benötigt.
- **3,5 mm verlötete Klemmenleiste** schneller Zugriff auf die High und Low Datenleitungen sowie einen Masse-Pin, ganz ohne Löten.
- **120-Ohm-Abschlusswiderstand auf der Platine**, du kannst die Terminierung einfach entfernen, indem du den mit TERM gekennzeichneten Jumper auf der Oberseite der Platine abschneidest.
- **CAN control CS, Reset, Int, standby pins** intern verbunden, damit du jeden FeatherWing ohne Pin-Konflikte verwenden kannst.

Das Herzstück des Feather ist ein RP2040-Chip, der mit 133 MHz getaktet ist und mit 3,3 V Logik arbeitet, derselbe, der auch im Raspberry Pi Pico verwendet wird. Dieser Chip hat satte 8 MB onboard QSPI FLASH und 264K RAM! Es ist sogar noch Platz für einen STEMMA QT Anschluss für Plug-and-Play von I2C Geräten.
Um den Einsatz für tragbare Projekte zu erleichtern, haben wir einen Anschluss für einen unserer 3,7-V-Lithium-Polymer-Akkus und eine integrierte Ladefunktion eingebaut. Du brauchst keinen Akku, er funktioniert auch direkt über den USB Typ C Anschluss. Wenn du aber einen Akku hast, kannst du ihn mitnehmen und dann den USB-Anschluss zum Aufladen anschließen. Der Feather schaltet automatisch auf USB-Strom um, sobald dieser verfügbar ist.

Hier sind ein paar praktische Daten! Du bekommst:

- Misst 2,0" x 0,9" x 0,28" (50,8mm x 22,8mm x 7mm) ohne eingelötete Anschlüsse
- ~~Leicht wie eine (große?) Feder - 6,3 Gramm~~
- RP2040 32-Bit Cortex M0+ Dual Core mit ~133 MHz bei 3,3 V Logik und Leistung
- 264 KB RAM
- **8 MB SPI FLASH** Chip zum Speichern von Dateien, Bildern und CircuitPython/MicroPython-Code. Kein EEPROM
- **Tonnenweise GPIO! 21 x GPIO-Pins mit folgenden Möglichkeiten:**
    - **Vier** 12-Bit-ADCs (einer mehr als beim Pico)
    - Zwei I2C-, zwei SPI- und zwei UART-Peripheriegeräte, von denen wir eines für die "Haupt"-Schnittstelle in Standard-Feather-Positionen beschriften
    - 16 x PWM Ausgänge - für Servos, LEDs, etc
- **Eingebautes 200mA+ Lipoly-Ladegerät** mit Ladestatusanzeige-LED
- **Pin #13 rote LED** für allgemeine Blinkzwecke
- **RGB NeoPixel** für die Vollfarbanzeige
- On-Board **STEMMA QT Anschluss** mit dem du schnell und ohne Löten Qwiic, STEMMA QT oder Grove I2C Geräte anschließen kannst!
- **Beide, Reset-Taste und Bootloader-Auswahltaste für schnelle Neustarts** (kein Ausstecken-Einstecken, um den Code neu zu starten)
- **USB Typ C Anschluss** ermöglicht den Zugriff auf den eingebauten ROM-USB-Bootloader und das Debugging der seriellen Schnittstelle
- 3,3V-Regler mit 500mA Spitzenstromausgang und Power Enable Pin
- 4 Befestigungslöcher
- 12 MHz-Quarz für perfektes Timing.
- **Unterstützungsschaltung für CAN-Bus mit SPI-Schnittstelle**

Wird zusammengebaut und getestet geliefert, mit einigen Headern. Du brauchst einen Lötkolben, um die Header für den Einbau in deine Feather zu befestigen. Wenn du die Header stapelst, kannst du einen weiteren FeatherWing darauf setzen.

Weitere Bilder: