De oorzaak van **software-bugs** draadloos opsporen

circulaire buffer en webserver op de ESP32

Laurent Labbe (Hong Kong)

Het loggen van debug-gegevens van een microcontroller op een laptop is niet altijd even makkelijk. Of je hebt geen reserve-laptop bij de hand, of het ontwikkelboard bevindt zich op een moeilijk bereikbare plaats. Dit Elektor Labs-project lost dit probleem niet alleen op met een groot buffergeheugen voor debug-data en een WiFi-verbinding. Het heeft ook een 3D-printbare behuizing en is in hoge mate uitbreidbaar.

De ontwikkeling van software voor microcontrollers is vaak een uitdaging. Op enkele uitzonderingen na bieden microcontrollers een soort debug-oplossing waarmee de processor kan worden gestart en gestopt en waarmee de inhoud van variabelen en registers kan worden onderzocht. Dit helpt bij het vaststellen van de oorzaak van een probleem. Veel maker-boards, zoals Arduino Uno en BBC micro:bit, bieden die ondersteuning echter niet van huis uit, waardoor de ontwikkelaar herhaaldelijk code moet bewerken en downloaden tot de oorzaak van het probleem is gevonden. Maar debuggers zijn niet altijd het wondermiddel dat ze lijken te zijn. Bij het ontwikkelen van real-time code voor Ethernet, WiFi of USB, of in toepassingen zoals motorbesturing, kun je de microcontroller niet zomaar halverwege stoppen. Dit resulteert in een onderbreking van de communicatie en kan zelfs schade toebrengen aan de voedingsschakeling als een MOSFET in ingeschakelde toestand komt.

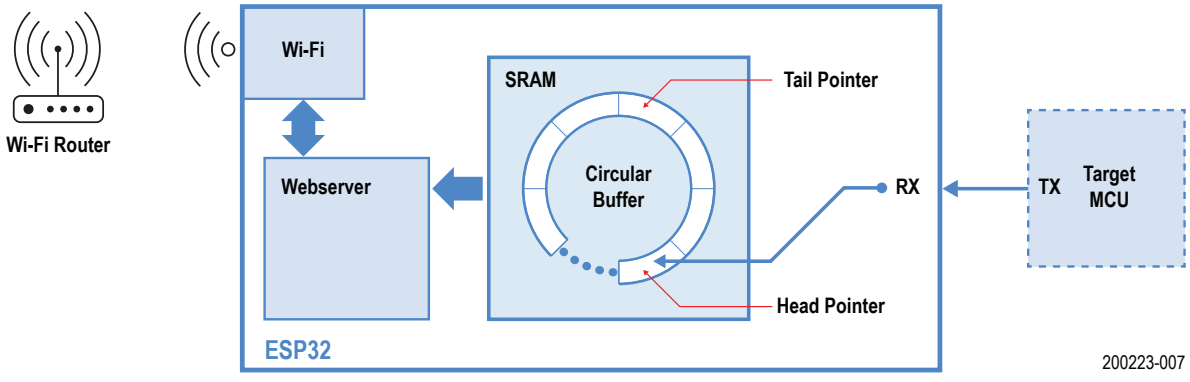
Dankzij het gemak waarmee een seriële uitgang kan worden ingericht op maker-boards, voegen de meeste ontwikkelaars een tekst-gebaseerde debug-output aan hun code toe waarmee berichten via USB naar een terminal op een PC worden gestuurd. Wanneer de uitvoer na een testrun wordt geanalyseerd, kan de ontwikkelaar het door de code gevolgde pad traceren, vandaar dat deze methode van debuggen van code bekend staat als 'software trace'. Dit maakt ook het debuggen mogelijk van toepassingen die tijdens de uitvoering niet kunnen worden gestopt.

Het project

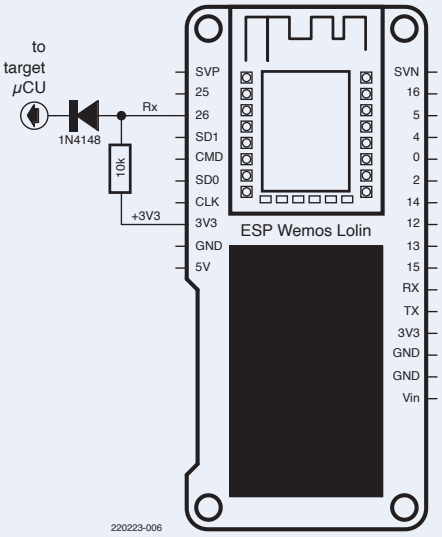
Laurent Labbe, geen onbekende op Elektor Labs, gebruikt vaak een seriële interface voor het traceren van de uitvoering van embedded code, maar heeft niet altijd een laptop bij de hand om de berichten te loggen. Daarom begon hij over een alternatief na te denken. Gewapend met een ESP32, een OLED-display en een 3D-printer bedacht hij het project "Wireless trace for debug" [1].

Hier wordt een ESP32 als buffer gebruikt voor debug-berichten, verbonden met de target-controller via een seriële interface; de buffer is toegankelijk via WiFi en een webserver (**figuur 1**).

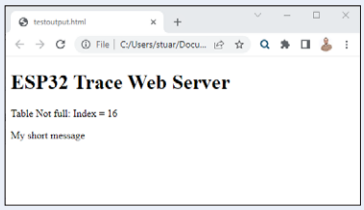
De enorme hoeveelheid SRAM op de ESP32 maakt het mogelijk om een grote circulaire buffer te implementeren. Laurent's projecten gebruiken gewoonlijk een lage snelheid van 9600 baud, maar dit kan worden aangepast. Telkens wanneer een karakter wordt ontvangen, wordt het in de kop van de circulaire buffer gepusht. De buffer is 65530 bytes (unsigned char) groot, zodat er voldoende ruimte is voor het verzamelen van trace-berichten. *Serial1* wordt geïnitialiseerd voor het verzamelen van gegevens via pinnen 25



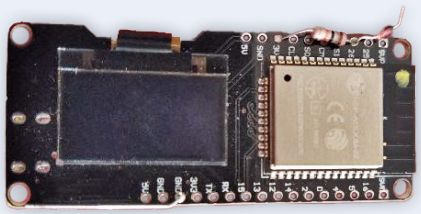
Figuur 1. Dit vereenvoudigde blokschema toont hoe inkomende seriële gegevens worden opgeslagen in de circulaire buffer. Op verzoek worden de gegevens via de WiFi-interface opgehaald in de vorm van een webpagina.



Figuur 2. Op deze manier kan een 5V-microcontroller op de 3,3V-ESP32 worden aangesloten.



Figuur 3. Een voorbeeld van de webpagina-uitvoer.



Figuur 4. De ESP32 compleet met 5V-interface.

Figuur 5. Een draadloze seriële debugger compleet met 3D-geprinte behuizing.

en 26 (respectievelijk TX en RX), hoewel alleen de RX-pin nodig is. Omdat de ESP32 op 3,3 V draait, heeft Laurent een weerstand/diode-netwerkje opgenomen om 5,0 V microcontrollers te kunnen aansluiten op de seriële interface (**figuur 2**).
Zoals gezegd maakt het project gebruik van de WiFi-functionaliteit van de ESP32, en dat is waar dit ontwerp uniek is. Na registratie bij het netwerk, zoals gedefinieerd in de code, kan elke lokale laptop of mobiel apparaat toegang krijgen tot een webpagina die door de ESP32 wordt aangeboden. De ESP32 levert vervolgens een eenvoudige pagina met de huidige inhoud van de circulaire buffer (**figuur 3**).

Tweede netwerk, seriële interface en display

Er zijn nog een paar andere coole functies ingebouwd in de code. Zo worden bijvoorbeeld een tweede SSID en wachtwoord voor een alternatieve WiFi-router ondersteund. Mocht de eerste verbinding om welke reden dan ook uitvallen, dan probeert de ESP32 automatisch verbinding te maken met het reserve-apparaat. Met mogelijk uren of dagen aan gegevens die op het apparaat zijn opgeslagen, biedt dit meer zekerheid bij het ophalen van de trace-berichten. De webserver gebruikt standaard poort 80, maar bij de initialisatie kan een alternatieve poort worden gedefinieerd.
De circulaire buffer implementeert ook 'wrap-around'. Mocht de buffer dus helemaal vol zijn, dan overschrijven nieuwe binnenkomende berichten de oudste gegevens. Voor de duidelijkheid levert de webpagina dan een toepasselijk bericht voordat de inhoud van de circulaire buffer van oud naar nieuw wordt weergegeven.
De ESP32 voert ook berichten uit via de seriële interface (via USB), zodat debugging van de code van dit project mogelijk is. Debug-berichten die via *Serial1* worden ontvangen, worden naar *Serial* uitgevoerd zodat ze direct zichtbaar zijn. Alle gegevens die via de *Serial* interface worden ontvangen, worden ook in de circulaire buffer geplaatst.
Informatie over het IP-adres en andere relevante details worden uitgevoerd naar een I²C-gebaseerd OLED-display aangesloten op pinnen 4 en 5. Het gebruikte display is de SSD1306 van Adafruit (samen met hun driver en GFX grafische bibliotheken).
Laurent gebruikte een WeMos Lolin32 OLED (**figuur 4**, het board is gebaseerd op de ESP32-WROOM-32 module met extra elektronica) om de USB-naar-UART-interface en ondersteuning voor het aansluiten en opladen van een LiIon/LiPo-batterij te implementeren. Samen met zijn 3D-geprinte behuizing, waarvoor een CAD-bestand wordt meegeleverd, kan deze draadloze debug trace-tool bijna overal worden ingezet om autonoom gegevens van een target te verzamelen (**figuur 5**).

Opties

Het fraaie van dit project is de combinatie van eenvoud en uitbreidbaarheid. De functionaliteit van de code is duidelijk, waardoor het voor ervaren ontwikkelaars eenvoudig is om dit project uit te breiden en aan te passen. Het is bijvoorbeeld niet moeilijk om het beschikbare SRAM uit te breiden met externe apparaten, en je zou zelfs de trace-berichten op een SD-kaart kunnen opslaan. De seriële interface-snelheden voor het loggen van gegevens kunnen ook worden aangepast, of in plaats daarvan kunnen gegevens van een I²C- of SPI-interface worden verzameld. En als je berichten een beetje opgeleukt moeten worden, dan kun je de juiste HTML of CSS invoegen voor het genereren van webpagina's.

200223-03

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de redactie van Elektor via redactie@elektor.com.

GERELATEERDE PRODUCTEN

- **WeMos Lolin ESP32 with OLED Display Module (SKU 18575)**
  www.elektor.nl/18575

WEBLINK

[1] Projectpagina op Elektor Labs:
www.elektormagazine.com/labs/wireless-trace-for-debug