

# ESP8266 delovanje pri nizki porabi in ESP-NOW

*Če gradite nekakšno napravo za nadzor z WiFi povezljivostjo, je bil ESP8266 v preteklosti priljubljena izbira. Medtem ko je novejši ESP32 (v več različicah) veliko bolj zmogljiva naprava, je v mnogih primerih cenejši ESP8266 še vedno povsem ustrezen.*

Če pa uporabljate takšno napravo na baterijsko napajanje, je treba upoštevati veliko stvari, da bi pripravili ustrezen dizajn in vedeli, kako velika baterija bo potrebna.

Upoštevati je treba tri glavne teme in vse so do neke mere medsebojno povezane:

- 1) V kakšnem intervalu mora biti enota popolnoma operativna za izvedbo naloge? (ESP8266 je lahko v stanju globokega spanja ali pa je ves drugi čas popolnoma izklopljen).
- 2) Katere ukrepe je mogoče sprejeti, da čim bolj zmanjšate mirovni tok, ki ga naprava porabi med časom »spanja« med cikli?
- 3) Kateri WiFi protokol bo uporabljen za komunikacijo z gostiteljem? Ali je za vsak merilni/kontrolni cikel potrebna povezava z gostiteljem, ali se to lahko zgodi manj pogosto?

Očitno je, da je točka 1 v celoti odvisna od aplikacije, zato je ni mogoče obravnavati neposredno v tem članku, razen navajanja nekaj primerov v izračunih, ki jih ponujam. Oglejmo si točki 2 in 3.

## Poraba toka ESP8266 v režimih spanja

Način dela	Poraba
Oddajanje Wi-Fi signala	170-400 mA
Sprejem Wi-Fi signala ali aktivno stanje	55-80 mA
Modem način spanja	15 mA
Lahek način spanja	0.5 mA
Globok način spanja	10 -20 $\mu$ A

**Tabela 1:** Poraba ESP8266 modula v različnih načinih delovanja

Tabela 1 prikazuje porabo energije ESP8266 v različnih načinih delovanja. Vidite lahko, da med prenosom WiFi signala uporablja do 400 mA toka. Upati je, da se to ne dogaja prepogosto in o tem bom govoril kasneje. Ko prejme WiFi signal ali samo izvaja kodo, bo ESP8266 še vedno črpal okoli 80 mA. Tudi ta nižja vrednost bi bila za LiPo baterijo precejšen zalogaj, če bi se to dogajalo ves čas.

Za dolgotrajno delovanje LiPo baterije je resnično uporaben le način globokega spanja. V bistvu se v tem načinu napajajo samo ULP (procesor z ultra nizko porabo energije), ura realnega časa (RTC) in nekaj RTC RAM-a. S temi funkcijami lahko napravo zbudite s spremembo stanja na GPIO ali timerjem spanja ESP8266. Timer spanja ESP8266 ima najdaljšo periodo 232 mikrosekund (71 minut). Uporabite lahko tudi RTC RAM ESP8266 za shranjevanje nekaterih spremenljivk iz ene periode globokega spanja v drugo.

V večini časa, ko je ESP8266 v globokem režimu spanja, bo porabil približno 20  $\mu$ A. Vendar pa morate tej številki dodati mirovni tok katerega koli regulatorja, ki ga uporabljate, da zmanjšate napetost baterije na 3,3 V, ki jih zahteva ESP8266.

Če bi na primer uporabili običajni LM1117MPX-3.3, je njegov mirovni tok 5 mA, kar bi preseгло 20  $\mu$ A tok globokega spanja ESP8266. Vendar pa lahko izberete 3,3 V LDO, ki ima veliko nižji mirovni tok, kot je MCP1702, ki črpa le 2,0  $\mu$ A mirovnega toka. Kasneje bom opisal svojo uporabo regulatorja MIC5323 LDO, ki ima mirovni tok le približno 1  $\mu$ A in omogoča tudi popoln odklop napajanja ESP8266 - s čimer se odpravi njegov 20  $\mu$ A tok globokega spanja.

Drugi porabniki toka so lahko kateri koli senzorji, ki jih uporabljate. V nekaterih primerih ti senzorji ne trošijo več kot 5 mA, ki jih lahko zagotovi GPIO priključek na ESP8266. Torej, če za napajanje tega senzorja namenite GPIO priključek, bo trošil energijo samo v času, ko ga ESP8266 napaja za izvedbo meritve.

Drug vir porabe toka bo odvisen od tega, kako boste ESP8266 vključili v svoj dizajn. Če uporabljate zgolj ESP8266 (kot ga najdete na Espressif WROOM-02 modulu), potem ni drugih porabnikov toka. Če pa uporabljate veliko trenutno razpoložljivih "razvojnih" ESP8266 modulov, bodo običajno vgrajene tudi druge naprave, ki trošijo tok.

Na primer, številne plošče vsebujejo USB-UART čip (CP2102 ali CH341) za programiranje/razhroščevanje napak. Te naprave porabijo veliko več energije kot ESP8266 v načinu globokega spanja in jih ne morete zlahka izolirati od črpanja energije. Nekatere plošče imajo tudi LED-ice za napajanje in celo majhna, slabo osvetljena LED-ica

lahko porabi 1 mA, kar spet odpravlja nekatere prednosti načina globokega spanja ESP8266.

Prav tako večina razvojnih plošč ne uporablja LDO z najnižjim mirovnim tokom, kot so tisti, ki sem jih omenil zgoraj. V tem članku bom uporabil modul WROOM-02, prikazan na sliki 1.



**Slika 1: WROOM-02 ESP8266 modul, ki sem ga uporabil za ta članek.**

Z uporabo modula WROOM-02, načina globokega spanja in regulatorja z nizkim mirovnim tokom lahko dosežete tok globokega spanja v območju 20-25  $\mu$ A. Če vam ESP8266 ni treba prepogosto zbuditi in vam tudi ni treba pogosto komunicirati z WiFi gostiteljem, potem je vredno raziskati načine, kako lahko zmanjšate porabo 20-25  $\mu$ A celo na manj. Z drugimi besedami, višji kot je odstotek časa, ko je vaš projekt lahko v načinu »spanja«, pomembnejše je zmanjšati ta tok »spanja«.

Slika 2 prikazuje pričakovano življenjsko dobo baterije, ki napaja ESP8266 z uporabljenim načinom globokega spanja in LiPo baterijo 1000 mAh. Predpostavlja, da se bo ESP8266 povezal z WiFi dostopno točko v časovnem intervalu, prikazanem na osi X. Povezava z WiFi dostopno točko bo trajala 3-5 sekund - podrobnosti o tej trenutni porabi bom opisal kasneje v članku. Upoštevajte, da se timer globokega spanja ESP8266 lahko podaljša le do 71 minut, zato je zgornji del grafa teoretičen – ni ga mogoče doseči samo z načinom globokega spanja ESP8266.

Da bi še dodatno zmanjšal porabo energije v stanju mirovanja, sem se odločil za uporabo MIC5323-3.3 LDO regulatorja z omogočeno linijo, glejte shemo na sliki 3. To omogočeno linijo nadzorujem z izhodnim priključkom alarma čipa ure realnega časa MCP79401 (napaja se neposredno iz LiPo celice). Če to storite na ta način, se ESP8266 popolnoma izklopi, kar sem prej omenil kot model »spanja«.

Zdaj je skupni mirovni tok le približno 4  $\mu$ A: 1  $\mu$ A troši regulator MIC5323-3.3, 1  $\mu$ A troši MCP79401 in 2  $\mu$ A, ki teče skozi 2,2 megohmski vlečni upor linije Enable, ko MCP79401 RTC ohranja LDO v onemogočenem načinu. Interval »spanja« zdaj nadzoruje funkcija alarma v MCP79401 RTC in ni omejen na dolžino časa mirovanja, ki ga je mogoče programirati.

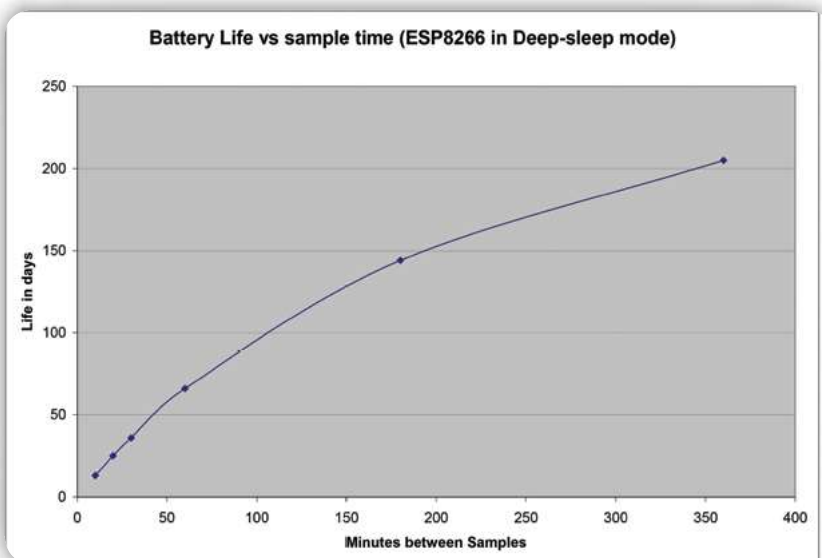
Moja izbira Microchip MCP79401 Real-Time Clock čipa ni bila poljubna. Na voljo je veliko RTC čipov in v preteklosti sem uporabljal RTC Dallas in

NXP (Philips). Vendar ima MCP79401 kombinacijo funkcij, ki niso prisotne v teh drugih čipih in so potrebne za to vezje:

- MCP79401 bo deloval pri Vcc do 5,5 voltov. To je potrebno, če projekt napajate neposredno iz LiPo celice, ki pri popolnoma napolnjeni bateriji znaša približno 4,1 volta (ali 3 alkalne baterije, ki imajo približno 4,8 voltov).
- MCP79401 ima zelo vsestransko funkcijo alarma. Najpomembnejša lastnost je, da ga je mogoče programirati za alarmni izhod aktivno-visoko ali aktivno-nizko. MIC5323 LDO regulator potrebuje aktivni visoki signal, da ga omogoči. Večina RTC ne bo delovala tukaj, saj imajo samo izhod z odprtim kolektorjem, ki spusti priključek alarma nizko, ko se alarm sproži.
- Vsebuje 64-bajtov SRAM-a. To trajno shranjevanje lahko uporablja ugnježena programska oprema za spremljanje števila ciklov ponovnega zagona, ki so se zgodili, in drugih vrednosti, ki morajo biti trajno shranjene.
- Njegova poraba toka je le 1,2  $\mu$ A pri 3,3 volta - nekoliko višji je pri polni LiPo bateriji.

Poglejmo, kako deluje MCP79401 v tem vezju. Dokler MCP79401 nima konfiguriranega alarma, bo njegov MFP priključek z odprtim ponorom (večfunkcijski priključek) deloval kot nepriključeno vezje. Zato bo Enable na MIC5323 v visokem stanju, zaradi česar bo regulator deloval.

To zagotavlja 3,3-V za ESP8266 in katero koli drugo vezje, ki ga napaja MIC5323 regulator. ESP8266 se bo zagnal in naredil, kar je potrebno za določeno ponovitev vklopa. Po tem bo nastavljen trenutni čas na neko fiksno vrednost, nato pa nastavljen čas alarma za toliko časa, kolikor ga želite med ponovnimi zagoni ESP8266. Obseg tega intervala ponovnega zagona je v bistvu neomejen – za razliko od časovnika mirovanja ESP8266, ki ima največje obdobje 232 mikrosekund (71 minut).



**Slika 2: Graf življenjske dobe baterije glede na časovni interval vzorca, ko je ESP8266 med vzorci nameščen v način globokega spanja.**

Ko je alarm omogočen, se MFP priključek MCP79401 spusti na nizko za čas tega intervala ponovnega zagona, kar izklopi napajanje ESP8266 in vseh drugih vezij, ki jih napaja regulator. Po intervalu ponovnega zagona, ko se sproži alarm, bo MFP priključek postavil na visoko in cikel se bo začel znova. V svoji ugnezdjeni programski opremi vzdržujem spremenljivko, imenovano `rebootCount`, ki je shranjena v SRAM-u v MCP79401. Ob vsakem ponovnem zagonu se število `rebootCount` poveča in ugnezdena programska oprema se lahko odloči, katere funkcije naj se izvajajo glede na to številko cikla ponovnega zagona. Rutine za dostop do MCP79401 so vsebovane v moji Arduino skici in se začnejo s predpono `RTC_`. Obstajajo rutine za nastavitvev trenutnega časa, nastavitvev časa alarma in omogočanje alarma. Obstajajo tudi rutine, ki berejo in pišejo v SRAM pomnilnik MCP79401, prav tako z uporabo predpone `RTC_`.

Ker ESP8266 med oddajanjem potegne 350-400 mA tokovne konice, sem na izhod regulatorja postavil kondenzator 470  $\mu$ F za obvladovanje teh konic. To nekoliko počasni dvig 3,3-voltne napajalne napetosti in pričakujem, da moram zato počakati 100 milisekund, preden preberem RAM MCP79401. Če ne bi uporabljali RTC RAM-a, bi lahko to zamudo odpravili (vrstica 84 v skici) in skrajšal čas izvajanja programa ESP8266 za 100 milisekund, kar ni nepomembno.

31

na splošno imeti možnost spremljati napetost baterije, da jo lahko zamenjate, preden vaš projekt preneha delovati. Morda je bolj pomembno v zvezi z LiPo celicami to, da jim ne smete dovoliti, da se izprazni pod približno 3 volte, sicer so lahko uničene. Pri majhnih LiPo baterijah v obliki vrečke je ta težava običajno označena z dejstvom, da se napihnejo. To sem večkrat videl v prejšnjih projektih, ki niso ščitili pred prekomerno izpraznjeno LiPo baterijo.

Ker je med LiPo in ESP8266 LDO regulator, ne morete samo izmeriti 3,3-voltnega izhoda LDO in iz tega natančno predvideti stanje baterije. Namesto tega morate meriti napetosti baterije pred LDO regulatorjem. ADC ESP8266 ima celoten obseg 1,0 volt, zato potrebujete uporovni napetostni delilnik, da zmanjšate 4,1-voltno napetost polno napolnjene LiPo baterije. Na sliki 3 lahko vidite, da imata upora R13 (2,2 megohm) in R14 (470 k) razmerje 5,7, kar zmanjša napetost polno napolnjene LiPo baterije na 0,72 V, kar je v območju 1,0 V celotnega obsega ADC.

Odločil sem se za uporabo visoko vrednih 2,2 megaohm in 470k uporov za napetostni delilnik, ker bi te visoke vrednosti uporovnega delilnika trošile manj kot 2  $\mu$ A iz LiPo celice. To je pomembno, ker je uporovni delilnik ves čas povezan z LiPo.

Vendar pa bo v mnogih MCU-jih ADC med merjenjem trošil visok tok. Zaradi tega bodo njihovi podatkovni listi pogosto navajali, da izvirna impedanca ne glede na to, kar se meri, ne sme biti večja od 10 k, da se doseže razumna natančnost. ESP8266 je drugačen: predstavlja visoko impedanco na svoji vhodni liniji ADC in rezultati, ki sem jih opazil, so bili v bistvu enaki, kot če napetostnega delilnika R13/R14 sploh ne bi imel ESP8266 za obremenitev, čez R14 uporovnega delilnika.

Vzporedno z R14 sem vezal 0,1  $\mu$ F kondenzator, da bi filtriral kakršen koli šum, saj je to vezje z visoko impedanco. Čudno je, da je ADC vhodni priključek v podatkovnem listu ESP8266 označen kot Tout - morda ga je mogoče konfigurirati tudi kot izhodni priključek timerja, vendar tega nisem preučil. Mimogrede, veriga delilnika za spremljanje napetosti baterije doda < 2  $\mu$ A porabo toka k prej omenjeni številki porabe 4  $\mu$ A za to vezje.

Z MCP79401 RTC smo dosegli zelo nizek tok v režimu spanja, pa tudi v bistvu neomejen čas "spanja". Vendar boste opazili, da sem dodal priključni blok X3, ki se povezuje na linije Vin in EN MIC5323. Če bi želeli to vezje uporabiti za spremljanje takojšnjih varnostnih alarmnih stikal (brez časovne zakasnitve mirovanja), lahko priključite normalno odprto stikalo na priključni blok X3.

Potem bi se ESP8266 vklopil takoj, ko bi bila takšna stikala zaprta. Če bi program ESP8266 preveril in ugotovil, da čas alarma NI dosežen, bi vedel, da je bilo varnostno stikalo sproženo.

Zagotovil sem Arduino skico z naslovom "ESP8266\_NOW\_MCP79401". To je zelo osnovna skica, ki zažene MCP79401 RTC, nastavi trenutni čas na fiksno vrednost in alarm na čas 1 minuto pozneje. Nato izmeri napetost baterije s pomočjo ESP8266 ADC-ja in njeno vrednost pošlje na serijska vrata.

Ta skica vključuje 100 milisekundno zakasnitev, potrebno za pravilno branje RAM-a v MCP79401, kar sem že omenil. Za shranjevanje spremenljivke z imenom RebootCount uporabljam enega od bajtov MCP79401 RAM-a. Ob vsakem vklopu se vrednost poveča in prikaže. Ko doseže 12, se ponastavi na nič.

To je samo za primer – omogoča vam, da izberete, koliko obdobj vklopa preteče, preden naredite nekaj, kar je treba izvajati manj pogosto (na primer prenos podatkov gostitelju preko WiFi).

Med 100 milisekundno zakasnitvijo vgrajeni zvočnik piska samo za namene razhroščevanja. Zadnja stvar, ki jo skica naredi, je aktiviranje RTC alarma. To postavi MFP priključek na MCP79401 na nizek nivo in to odklopi napajanje ESP8266 (in vse ostalo, kar napaja 3,3-voltni regulator).

Upoštevajte, da ko naložite to skico in jo začnete izvajati, se bo ESP8266 vklopil le za nekaj sto milisekund, ko se sproži alarm MCP79401 (s čimer se dvigne MFP priključek in omogoči LDO regulator). Kako lahko ponovno programirate ESP8266 v tem stanju?

Najlažji način je, da izklopite sponki 1 in 2 na X3, zaradi česar bo regulator MIC5323 ves čas omogočen - s čimer bo napajanje ESP8266 neprekinjeno. Nato ga lahko preklopite v programski način na običajen način, tako da držite pritisnjeno stikalo PROG, medtem ko preklapljate RESET.

Ena stvar, ki sem jo opazil, ko sem eksperimentalno s to ploščo je, da ESP8266 izvaja kodo približno 70 milisekund, preden dejansko zažene vašo skico. Glejte sliko 4 za obseg zajema trenutne porabe (napetost, izmerjena na upor 0,22  $\Omega$  v seriji z negativnim priključkom LiPo celice). V začetnem intervalu 70 ms ESP8266 črpa približno 36 mA. Na točki 100 ms je kratek 270 mA tokovni skok za prenos WiFi in dve drugi zelo kratki 400 mA konici pozneje - čeprav v skici ne zahtevam nobene WiFi dejavnosti.

V zgoraj omenjeni skici uporabljam funkcijo millis(), da določim, koliko milisekund je preteklo od zagona. Vedel sem, da bo ta čas odvisen od tega, kdaj je bil milisekundni časovnik ESP8266 inicializiran na nič in zagnan. Vendar nisem pričakoval, da bo preteklo 70 milisekund, preden se je ta časovnik zagnal in ko je bila vaša skica izvedena.

Tega sem se zavedal, ko sem izmeril dejanski čas, ko je bila moč uporabljena na ESP8266. Na 'scope capture', prikazanem na sliki 4, skupni čas vklopa energije znaša 338 ms. Vendar pa je funkcija millis(), ki je bila poklicana tik





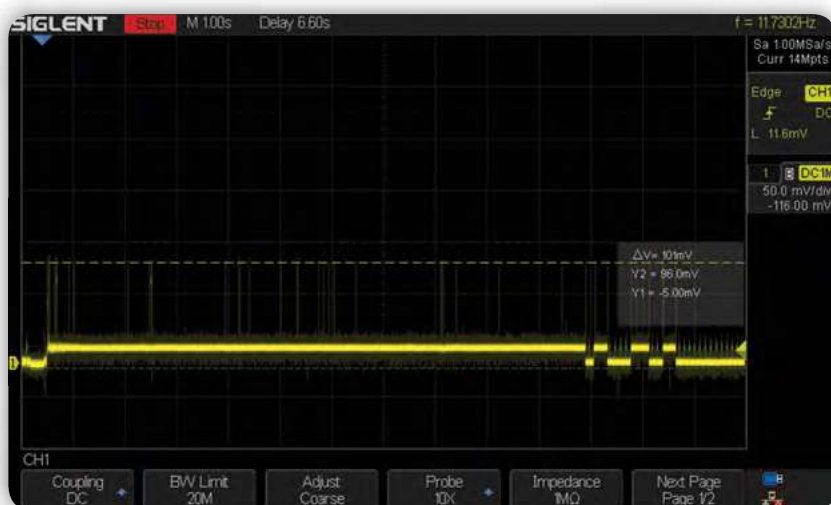
**Slika 4:** Obseg zajema trenutne porabe ESP8266 v 80 ms po vklopu, vendar preden skica pridobi nadzor nad MCU ESP8266.

pred aktiviranjem MCP79401 alarma (ki izklopi ESP8266), vrnila 268 ms, kar je razlika 70 ms.

Zato morate domnevati, da bo ESP8266 deloval 70 ms in v povprečju porabil približno 36 mA, ne glede na to, kako hitro vaša skica opravi svojo zahtevano funkcijo in nato preklopi ESP8266 v način globokega spanja ali ga popolnoma izklopi, z uporabo MCP79401 RTC.

Tudi z nameščenim MCP79401 RTC vezjem in pravkar omenjenimi podrobnostmi o porabi energije je še vedno problem visoke porabe energije ESP8266 za relativno dolgo časovno obdobje, ki je potrebno za povezavo z vašo WiFi dostopno točko. Med prejemanjem ali drugim izvajanjem kode sem izmeril, da ESP8266 porabi približno 80 mA pri 3,3-V.

V tem načinu je njegovo VF vezje (WiFi/Bluetooth) napajano in lahko sprejema ne glede na vrsto VF signala, za



**Slika 5:** Prikaz porabe energije ESP8266 za 3-5 sekund, potrebnih za povezavo z dostopno točko WiFi mojega doma.

katerega je bila enota konfigurirana: WiFi ali BLE, vendar NE oboje. Ko ESP8266 oddaja signal, se bo ta poraba toka dramatično povečala s številnimi kratkimi "oddajnimi" močnimi konicami približno 350-400 mA.

**Povprečna** poraba toka ni blizu največjega toka teh konic. Vendar pa se pri vzpostavljanju povezave z domačim WAP-om ti skoki pogosto pojavljajo v 3-5 sekundah, potrebnih za dokončanje povezave. Ta dolg čas povezave je posledica zapletenega WiFi protokola, ki se izvaja, in dodatnih rutin zaradi varnostnih razlogov.

Slika 5 prikazuje trenutno porabo med tipično povezavo z mojo WiFi dostopno točko. Vidite lahko konice med oddajo, vendar so tako ozke, da je težko natančno izmeriti njihov prispevek k skupni trenutni porabi. Medtem ko lahko moj digitalni multimeter meri povprečni enosmerni tok, opravi le eno ali dve meritvi na sekundo, njegovi odčitki med trajanjem povezave močno nihajo.

Vendar pa bi iz slike na osciloskopu ocenil, da bi oddajne konice lahko dodale približno 15 % k povprečnemu toku 80 mA, ki ga črpa ESP8266, ko ne oddaja WiFi signala. Ta tok sem izmeril preko padca napetosti na upor 0,25 Ω v negativnem napajalnem kablu. Oddajne konice dosežejo približno 400 mA, poraba energije med sprejemom pa je približno 80 mA.

Iz serijskega izhoda programa, ki se izvaja na ESP8266, lahko ugotovim, da je WiFi povezava trajala približno 3-5 sekund. Vendar pa na sliki 5 lahko vidite, da prenosi vztrajajo približno 8 sekund – čeprav je program v zanki in ne počne nič drugega, kot da pošilja sporočilo na serijska vrata vsako sekundo po vzpostavitvi WiFi povezave. Ne vem, zakaj je temu tako, vendar ima to za posledico veliko porabo energije, tudi če program uporabnika ne zahteva nobene WiFi dejavnosti.

Iz zgornjega je očitno, da bo povprečni tok približno 92 mA (80 + 15%) črpal 3-5 sekund samo za vzpostavitev začetne povezave z WAP. Predpostavimo, da bi lahko ESP8266 popolnoma izklopili za daljša časovna obdobja med meritvami/povezavo z WiFi in se vprašajmo, kako dolgo bi zdržala majhna LiPo celica?

Slika 6 prikazuje rezultate spletnega kalkulatorja življenjske dobe baterije za scenarij, v katerem je bil ESP8266 popolnoma izklopljen med tako imenovanim intervalom »spanja« in deluje le enkrat na vsakih 15 minut, da odčita in pošlje podatke do gostitelja prek WiFi.

Izračun pokaže 109 dni, kar ni zelo dolgo časovno obdobje in upoštevajte, da predvidevam, da