Podstawowe parametry:

- pomiar napięcia stałego o dowolnej polaryzacji w zakresie ±22 V lub 40 V przy ustawieniu unipolarnym,
- wykonywanie pomiarów z 2 wejść,
- pomiar oporności w zakresie od 10 Ω do 500 kΩ (maksymalnie do 1,5 MΩ przy rosnącym uchybie),
- pomiar pojemności w zakresie od 1 nF do 330 µF,
- pomiar indukcyjności w zakresie od około 100 µH do 100 mH,
- pomiar temperatury za pomocą czujnika DS18B20,
- test diod półprzewodnikowych,
- sterowanie i wyświetlanie danych na ekranie zdalnym (smartfona, laptopa) za pomocą dynamicznie generowanej strony dla przeglądarek obsługujących standard HTML5,
- zasilanie +5 V z zewnętrznego zasilacza, komputera, powerbanku, podawane przez gniazdo mini USB.

Dodatkowe materiały do pobrania ze strony www.ulubionykiosk.pl/media

AVT5507	Miernik UIPTR (EP 7/2015)
AVT5399	Dwukanałowy multimetr panelowy (EP 6/2013)
AVT5386	Podwójny woltomierz i amperomierz (EP 3/2013)
AVT5383	Miernik tablicowy UIPt (EP 2/2013)
AVT5333	Multimetr panelowy (EP 3/2012)
AVT2857	Moduł woltomierza/amperomierza (EdW 3/2008)
AVT5086	Programowany 4-kanałowy komparator/woltomierz (EP 11/2002)
AVT2004	Woltomierz do modułowego zestawu pomiarowego (EdW 1/1996)

W ofercie AVT*

AVT5954

* Uwaga! Elektroniczne zestawy do samodzielnego montażu. Wymagana umiejętność lutowania! Podstawową wersją zestawu jest wersja [B] nazywana potocznie KIT-em (z ang. zestaw). Zestaw w wersji [B] zawiera elementy elektroniczne (w tym [UK] – jeśli występuje w projekcie), które należy samodzielnie wlutować w dołączoną płytkę drukowaną (PCB). Wykaz elementów znajduje się w dokumentacji, która jest podlinkowana w opisie kitu. Mając na uwadze różne potrzeby naszych klientów, oferujemy dodatkowe wersje:
- wersja [C] – zmontowany, uruchomiony i przetestowany zestaw [B] (elementy wlutowane w płytkę PCB)
- wersja [A] – płytka drukowana bez elementów i dokumentacji

Kity, w których występuje układ scalony wymagający zaprogramowania, mają następujące dodatkowe wersje:
- wersja [A+] – płytka drukowana [A] + zaprogramowany układ [UK] i dokumentacja
- wersja [UK] – zaprogramowany układ

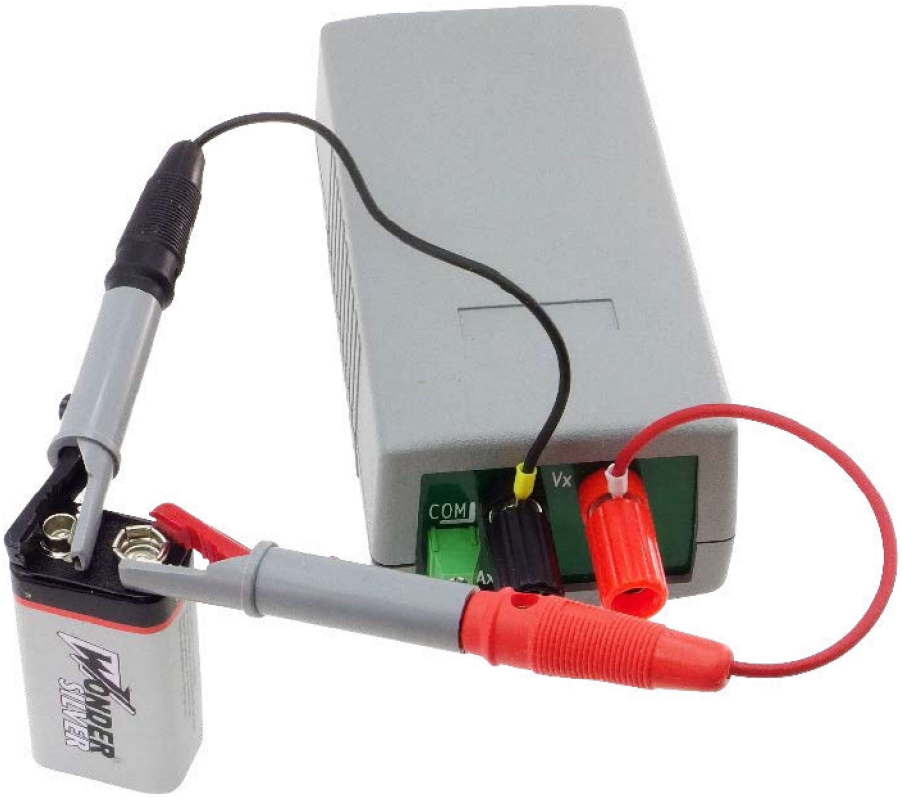
Nie każdy zestaw AVT występuje we wszystkich wersjach! Każda wersja ma załączony ten sam plik pdf! Podczas składania zamówienia upewnij się, którą wersję zamawiasz! – http://sklep.avt.pl.

W przypadku braku dostępności na stronie sklepu osoby zainteresowane zakupem płytek drukowanych (PCB) prosimy o kontakt Via e-mail: kity@avt.pl.

Warsztatowy multitester

Na bazie modułu ESP32, z możliwością pomiarów zdalnych

Miejsce pracy elektronika zwykle kojarzy się z płytkami elektronicznymi, narzędziami i przyrządem pomiarowym, najczęściej multimetrem. Pomiarów dokonuje się już na etapie montażu, sprawdzając niektóre z elementów, a potem, kiedy uruchamiane urządzenie złośliwie nie chce działać, miernik jest używany coraz częściej. I tak jak trzeciej ręki, często brakuje kolejnego przyrządu do kontroli parametrów w istotnych punktach układu. Jeżeli czytelnikowi zdarzają się takie przypadki, powinien się zainteresować prostym testerem, którym można zbadać poziom napięcia czy zmierzyć elementy dyskretne i który prześle wyniki na ekran wybranego smartfona.



Bezpośrednią inspiracją do opracowania przyrządu był projekt cyfrowego multimetru [1]. Taki multimetr dokonuje pomiarów za pomocą płytki Arduino. Następnie wyniki poprzez moduł Bluetooth przesyłane są do smartfona, gdzie specjalna aplikacja odpowiada za ich prezentację. W projekcie multitestera całą pracę, czyli pomiary, prezentację wyników i komunikację Wi-Fi ze smartfonem, wykonuje płytka z ESP32. Dodatkowe elementy tworzą kilka interfejsów, które zamieniają mierzone wielkości na równoważne sygnały napięciowe dla przetwornika lub liczników układu ESP32.

Ostatecznie zdecydowano, że projekt zamiast nazwy multimetr zadowoli się skromnym określeniem multitester.

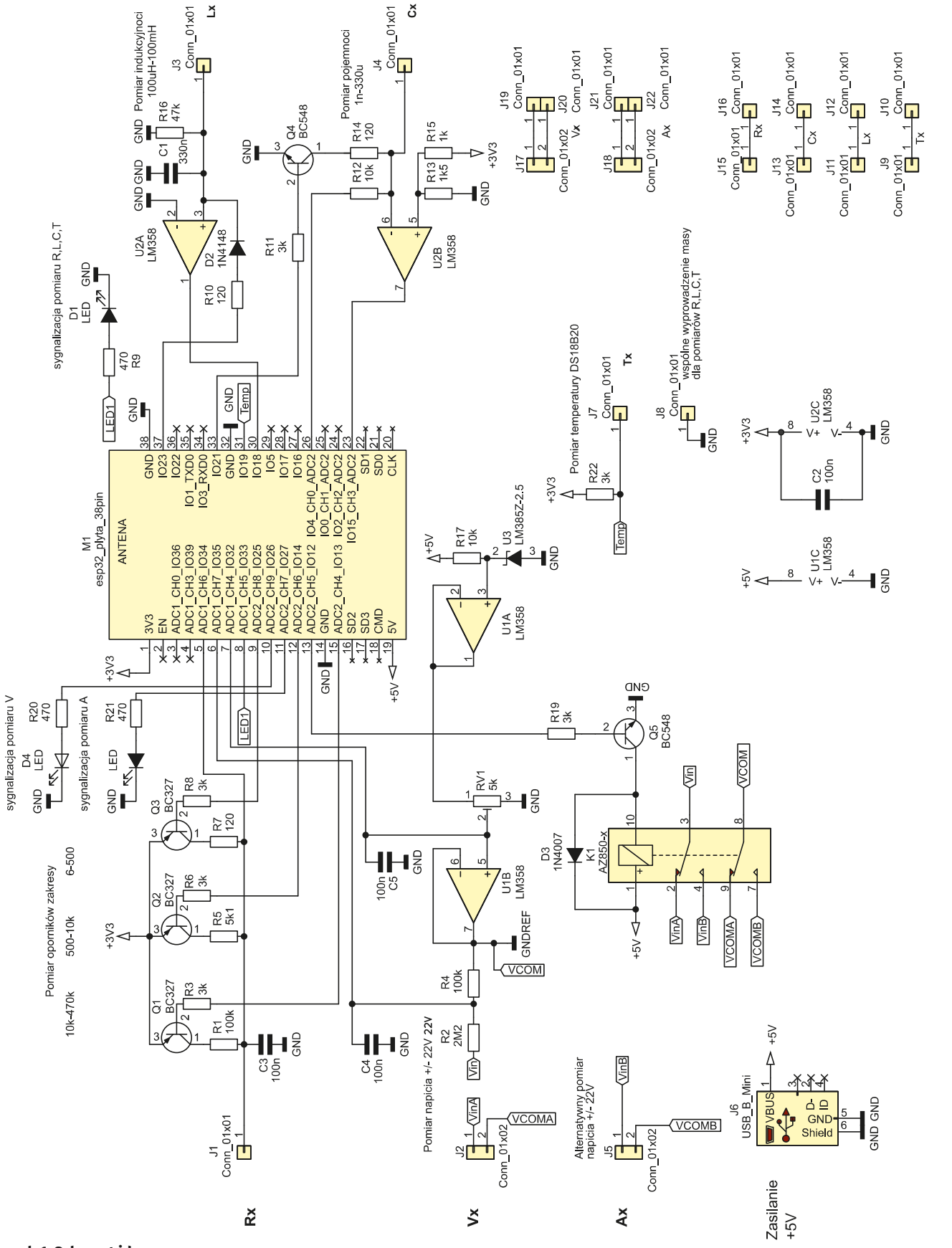
Budowa i działanie

Na **rysunku 1** został pokazany kompletny schemat ideowy multitestera. Za pomiary i komunikację z użytkownikiem odpowiada płytka rozwojowa z modułem ESP-WROOM-32. Zintegrowana z modułem antena służy do komunikacji Wi-Fi na odległość kilku, kilkunastu metrów. Komunikacja z urządzeniem sterującym, takim jak smartfon, może odbywać się w jednym z dwu trybów: jako stacja w ramach sieci tworzonej przez zewnętrzny router lub jako samodzielny AP (*access point*) z własną siecią Wi-Fi, do której loguje się smartfon.

Do wyprowadzonych na złącza płytki portów ESP32 podłączone są dodatkowe elementy tworzące interfejsy pomiarowe.

Interfejs do pomiaru oporności

W procedurze pomiaru oporności zastosowany jest dzielnik oporowy. Znana jest wartość oporu jednego opornika dzielnika oraz poziomy napięć wejściowego i wyjściowego, co umożliwia wyliczenie oporu drugiego opornika. Badany opornik podłączany jest do wejścia J1 i masy J8. Oporniki R7, R5, R1 stanowią człon dzielnika o znanej oporności. Są one przyłączane przez klucze tranzystorowe Q1...Q3 zależnie od wybranego podzakresu pomiarowego. Podział na podzakresy wynika z ograniczeń przetwornika ADC. Do wyboru są 3 przedziały pomiarowe: 6 Ω...500 Ω, 500 Ω...10 kΩ, 10 kΩ...470 kΩ. Napięcie z wyjścia dzielnika podawane jest na port IO34, który jest wejściem przetwornika ADC1/CH6.



Rysunek 1. Schemat ideowy

Interfejs do pomiaru napięcia stałego

Przetwornik ESP32 potrafi zmierzyć jedynie napięcie dodatnie o ograniczonym poziomie. Aby uniezależnić się od polaryzacji badanego napięcia i poszerzyć zakres, trzeba zastosować układ dopasowujący. Pomysł takiego układu został zaczerpnięty z projektu potrójnego woltomierza [2]. Źródło U3 wraz z wtórnikiem U1A tworzą stabilne napięcie referencyjne o poziomie 2,5 V dla wieloobrotowego potencjometru RV1. Napięcie z suwaka podawane jest na wejście przetwornika ADC1/CH4 oraz poprzez wtórnik U1B na wejście pomiarowe COM. Wejście to tworzy sztuczną masę, do której podłącza się jeden biegun mierzonego źródła napięcia. Poziom napięcia wejścia COM względem rzeczywistej masy testera ustawiany jest potencjometrem i powinien mieć wartość 1,25 V. Drugi biegun mierzonego źródła podawany jest na wejście Vin i z dzielnika R2, R4 napięcie trafia do wejścia ADC1/CH7. Zależnie od polaryzacji napięcie Vin będzie albo sumować się z potencjałem sztucznej masy COM, albo od niego odejmować, jednak dla przetwornika ADC będą to zawsze napięcia dodatnie, czyli możliwe do zmierzenia. Znając napięcia z obydwu wejść przetwornika oraz rezystancję oporników R2, R4, można obliczyć poziom mierzonego napięcia Vin.

Tester obsługuje dwa gniazda pomiarowe Vx i Ax. Styki przekaźnika K1 podłączają wybierane gniazdo J2 lub J5 do wejść Vin i COM.

Interfejs do pomiaru indukcyjności

Pomiar indukcyjności bazuje na zjawisku rezonansu układu LC – cewka, kondensator. Podstawę do obliczeń stanowi wzór na częstotliwość rezonansową:

$$F_R = \frac{1}{2\pi\sqrt{LC}}$$

Znając pojemność kondensatora i zmierzoną częstotliwość rezonansu, można obliczyć indukcyjność cewki w obwodzie LC.

Badaną cewkę podłącza się do wejścia J3 i masy J8. Pojemność obwodu rezonansowego stanowi kondensator C1. Wzmacniacz U2A pracuje jako komparator, przekształcając niewielkie amplitudowo oscylacje obwodu LC w impulsy prostokątne. Impulsy podawane są na port IO23 pracujący jako wejście przerwania wyzwalanego przez obydwa zbocza. Wystąpienie przerwania umożliwia zmierzenie czasu trwania impulsów i określenie częstotliwości używanej w dalszych obliczeniach. Opornik R10 i dioda D2 służą do podania do obwodu LC impulsu, który pompuje go energią. Gdy impuls zanika, obwód oddaje energię w gasnących oscylacjach o częstotliwości rezonansowej. Opornik R16 służy do wytłumiania przypadkowych wzbudzeń układu i może nie być montowany.

Interfejs do pomiaru pojemności

Pomiar pojemności bazuje na pomiarze stałej czasowej układu RC. Należy dokonać pomiaru czasu ładowania w pełni rozładowanego kondensatora do poziomu 63,2% wartości maksymalnej. Wzór na stałą czasową układu RC:

$$\tau = RC$$

Mierząc TC i znając wartość oporności R, można obliczyć C.

Badaną pojemność podłącza się do wejścia J4 i masy J8. Tranzystor Q4 i opornik R14 służą do pełnego rozładowywania kondensatora. Opornik R12 tworzy część układu RC i ładuje kondensator. Wzmacniacz U2B pracuje jako komparator. Jego wejście nieodwracające jest spolaryzowane napięciem z dzielnika R13, R15 do poziomu około 63% maksymalnego poziomu zasilania. Gdy w czasie testu napięcie na kondensatorze podczas ładowania przekroczy ten poziom, na wyjściu komparatora pojawi się zbocze opadające. Jest ono podawane na port IO15 będący wejściem przerwania, które współpracuje z wewnętrznym licznikiem służącym do zliczenia czasu ładowania kondensatora, a więc do określenia stałej RC.

Interfejs do pomiaru temperatury

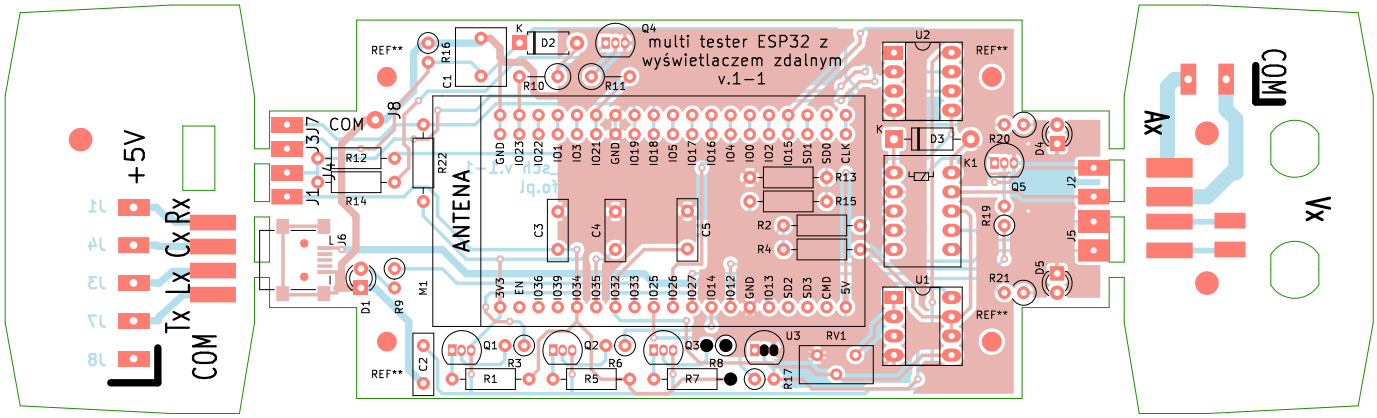
Wyjście J7 służy do podłączenia czujnika DS18B20 do pomiaru temperatury. Jest to magistrala 1-Wire podciągana do napięcia zasilania opornikiem R22. Magistralą czujnik przesyła dane cyfrowe zmierzonej temperatury.

Zasilanie i sygnalizacja

Gniazdem mini USB J6 podawane jest tylko napięcie zasilające. Diody LED sygnalizują stan urządzenia: świecenie D1 sygnalizuje aktywność któregoś z interfejsów R, L, C, T, świecenie D4 oznacza aktywny pomiar napięcia wejściem Vx, a świecenie D5 aktywny pomiar wejściem Ax.

Montaż

Dwustronna płytka drukowana została zaprojektowana do montażu przewlekanego, z elementami po stronie opisowej. Jej schemat został pokazany na **rysunku 2**. Jedynie gniazdo zasilania mini USB jest lutowane



Rysunek 2. Schemat płytki PCB

WYKAZ ELEMENTÓW, które możesz zamówić w sklepie AVT na stronie sklep.avt.pl lub bezpośrednio (ul. Leszczynowa 11, 03-197 Warszawa, tel. 48222578451, e-mail: handlowy@avt.pl):

Rezystory:
R1, R4: 100 kΩ
R2: 2,2 MΩ
R3, R6, R8, R19: 3 kΩ
R5: 5,1 kΩ
R7, R14: 120 Ω
R9: 470Ω
R10: 120Ω
R11: 3 kΩ
R12: 10 kΩ
R13: 1,5 kΩ
R15: 1 kΩ
R16: 47 kΩ
R17: 10 kΩ
R20, R21: 470 Ω
R22: 3 kΩ
RV1: 5 kΩ

Kondensatory:
C1: 330 nF
C2 C3 C4 C5: 100 nF

Półprzewodniki:
D1, D4, D5: LED 3 mm
D2: 1N4148
D3: 1N4007
Q3, Q2 Q1: BC327 (TO-92)
Q4, Q5: BC548 (TO-92)
U1, U2: LM358 (DIP-8)
U3: LM385Z-2.5 (TO-92)

Pozostałe:
K1: P-5 przekaźnik subminiaturowy
M1: ESP32 płyta 38 styków
J6: USB B mini

powierzchniowo. Zmontowana płytka z zamontowanym modułem z ESP32 została pokazana na **fotografii 1**.

Uwaga! Parametry siedmiu elementów należy przed wlutowaniem zmierzyć, a wyniki pomiarów zachować. Zapamiętane wartości są używane do obliczeń podczas pracy multitestera i mają duży wpływ na precyzję otrzymywanych wyników pomiarów.

Lista elementów, które trzeba zmierzyć przed montażem:

- R7, R5, R1 – wartości w Ω, gdzie k=1000 Ω, 1 M=1000000 Ω
- R2, R4 – wartości w Ω
- C1 – wartość w faradach, gdzie 1 F=1000000 µF=1000000000 nF
- R12 – wartość w Ω

Wyniki pomiarów elementów zostaną użyte na etapie uruchamiania do utworzenia pliku konfiguracyjnego.

Użyta płyta ESP32 powinna być zamontowana w gnieździe składającym się z 2 listew dla goldpinów po 19 styków każda i rastrze 2,54 mm. Płytka drukowana testera ma rozstaw otworów przystosowany do płyt o szerokości 22,86 mm lub 25,4 mm. Ważne jest, aby rozkład sygnałów na stykach zastosowanej płyty odpowiadał rozkładowi na schemacie ideowym.

Również układy scalone U1, U2 i miniaturowy przekaźnik K1 warto zamontować w gniazdach wykonanych z listew styków precyzyjnych o rastrze 2,54, ale o mniejszych otworach niż goldpiny. Wykonane z takich listew 2-stykowe gniazdka bardzo ułatwiają prawidłowe mocowanie diod sygnalizacyjnych LED.

Płytka drukowana została tak zaprojektowana, by można ją było zamontować w obudowie G410 o wymiarach zewnętrznych 120×60×40. Do płytek drukowanych, pełniących funkcję paneli bocznych, należy wlutować gniazda śrubowe kątowe do mocowania przewodów w pomiarach elementów R, L, C, T (**fotografia 2**) – zaciski laboratoryjne do pomiaru napięcia Vx oraz gniazdo śrubowe proste do pomiaru napięcia Ax.

Płytki boczne można na sztywno przylutować do głównej płytki, jednak przynajmniej na etapie uruchamiania i kalibracji lepiej zastosować połączenia krótkimi odcinkami kabli. Zapewni to możliwość odchylania bocznych płytek, co ułatwi włożenie wtyku USB do gniazda na płycie ESP32.

Uruchomienie

W przypadku multitestera większość pracy wykonuje oprogramowanie płyty ESP32. Napisany do obsługi testera kod korzysta z ESP-IDF (*Espressif IoT Development Framework*) w wersji 4.4. Jest to zbiór oprogramowania systemowego i narzędziowego stworzonego i rozwijanego przez producenta systemu ESP32. Nie jest tak popularny jak opracowany na jego bazie arduinowy klon dla ESP32, ale zapewnia nieco większe możliwości. Jednak wszyscy, którzy chcieliby wypróbować oprogramowanie dla multitestera nie muszą tylko z tego powodu instalować na swoim komputerze Eclipse i reszty środowiska programistycznego. Wystarczy za pomocą jednego z narzędzi, dostarczanych przez firmę Espressif, przenieść do pamięci flash płyty ESP32, 3 pliki wynikowe oprogramowania multitestera [3]. Pracujący z systemem Windows znajdą firmowe narzędzie do programowania [4], a opis programowania za jego pomocą pamięci flash w [5]. Dla użytkownika systemu linuksowego naturalnym wyborem będzie konsolowy *esptool.py* napisany w Pytonie. Trzy pliki powinny zostać zapisane pod następującymi adresami:

```
bootloader.bin – 0x1000,
partition-table.bin – 0x8000,
esp32_http_server_baza3.bin
– 0x10000.
```

Korzystając z esptool, sekwencja przesyłania nowego oprogramowania do pamięci flash może wyglądać następująco. Jeżeli nie chcemy podawać długich ścieżek dostępu, przenosimy się do katalogu, gdzie są zapisane 3 pliki binarne. Następnie wpisujemy:

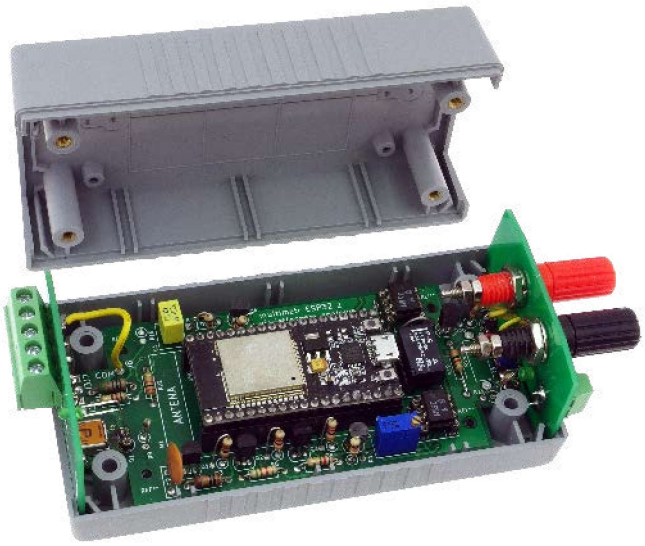
```
esptool.py erase_flash
```

To polecenie wyczyści pamięć flash. Nie ma potrzeby podawać numeru portu USB, do którego podłączona została płyta ESP poprzez zamontowane na jej pokładzie gniazdo, program sam go odszuka. Kolejne trzy linie inicjują zapis następnego pliku:

```
esptool.py write_flash 0x1000
bootloader.bin
esptool.py write_flash 0x8000
partition-table.bin
esptool.py write_flash 0x10000
esp32_http_server_baza3.bin
```

Jeżeli po zapisie ostatniego pliku i ewentualnie po resecie płyty ESP zacznie wolno migać dioda led D1, będzie to oznaczało, że prawdopodobnie wszystko poszło dobrze. Czy to oznacza, że można już korzystać z multitestera? Nie.

Teraz nadszedł czas na zapis do SPIFFS, wewnętrznego systemu plików utworzonego przez właśnie wgrane oprogramowanie, pliku konfiguracyjnego o dokładnie takiej nazwie: *plik_cfg.cfg*, którego przykład jest do pobrania [3]. Jest to zwykły plik tekstowy JSON. Podany przykład należy zmodyfikować, podając dane swojej sieci Wi-Fi



Fotografia 1. Wygląd zmontowanych płytek multitestera

Fotografia 2. Wygląd gniazda do pomiarów R, L, C, T

i zmierzone podczas montażu parametry elementów – **listing 1.** Znaczenie kolejnych pozycji pliku jest następujące:

- element 1. ("start_html") – nazwa pliku,który ma zostać automatycznie uruchomiony po resecie (dokładne wyjaśnienie znajduje się w dalszej części artykułu);
- elementy 2. i 3. ("ssid", "pass") – nazwa i hasło logowania do lokalnej sieci Wi-Fi;
- element 4. ("softAP_on") – jeśli „true", multitester zawsze będzie tworzył i pracował w obrębie własnej sieci Wi-Fi, jeśli „false" tester automatycznie zaloguje się w lokalnej sieci Wi-Fi, a gdyby była niedostępna, to po resecie stworzy własną;
- element 5. ("vref") – dokładna wartość napięcia zmierzona na wyprowadzeniu numer 1 (3V3) płyty ESP32, jak wszystkie parametry z zapisem z kropką, są to wartości zmiennoprzecinkowe;
- kolejne pozycje odnoszą się do rzeczywistych parametrów elementów zmierzonych przed zamontowaniem. Wszystkie wartości oporników są wyrażone w Ω, wartość kondensatora w faradach:
  – "Rref_zak_om" – R7,
  – "Rref_zak_kom" – R5,

Listing 1. Przykładowa zawartość pliku plik_cfg.cfg,

```
{
    "start_html":    "index.html",
    "ssid":          "nazwa_sieci",
    "pass":          "hasło_sieci",
    "softAP_on":     false,
    "vref":          3.31,
    "Rref_zak_om":   118.5,
    "Rref_zak_kom":  4970.0,
    "Rref_zak_100k":100600.0,
    "res_cap":       9529,
    "cap_induc":     0.000000322,
    "r_high_volt":   2267000,
    "r_low_volt":    99500
}
```

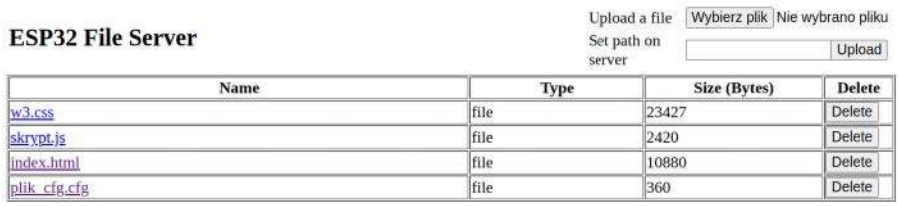
– "Rref_zak_100k" – R1,
– "res_cap" – R12,
– "cap_induc" – C1,
– "r_high_volt" – R2,
– "r_low_volt" – R4.

Wszystkie pliki przesyłane są do SPIFFS drogą radiową. **Przy pierwszym uruchomieniu tester zawsze zgłasza się jako softAP, tworząc własną sieć Wi-Fi**. Żeby się z nim połączyć, trzeba z listy dostępnych sieci wybrać sieć o nazwie „ESP32 Serwer" i haśle dostępu „mypassword". W pasku przeglądarki należy wpisać adres 192.168.4.1. Powinna wyświetlić się strona podobna do tej z **rysunku 3**. Jest to prosty menedżer plików obszaru SPIFFS. Pozwala na usuwanie i zapis nowych plików oraz na ich podgląd. Należy wskazać lokalizację utworzonego pliku *plik_cfg.cfg* i zapisać go do SPIFFS.

To dobry moment, żeby przesłać także pliki, które tworzą dla użytkownika graficzny interfejs i pozwalają komunikować się z multitesterem za pośrednictwem dowolnej przeglądarki obsługującej standard HTML5. Są to kolejne 3 pliki:

- *w3.css* – kaskadowy arkusz stylów wykorzystujący popularny szablon, który zapewnia jednolity wygląd elementów strony, a także jej responsywność;
- *skrypt.js* – plik z częścią skryptów JavaScript obsługujących przesył i prezentację danych na wyświetlaczu smartfona lub komputera;
- *index.html* – źródło generowanej strony dla przeglądarki i część skryptów JavaScript. To właśnie ten plik powinien zostać zadeklarowany w pliku konfiguracyjnym jako przeznaczony do automatycznego uruchomienia po resecie.

Pliki zapisywane są do SPIFFS w identyczny sposób jak plik *plik_cfg.cfg* i powinny być do pobrania z [3]. Teraz po resecie zależnie od ustawień zapisanych w pliku *plik_cfg.cfg*, multitester może ponownie zgłosić się we własnej sieci lub zalogować się do sieci podanej w pliku konfiguracyjnym. Ustawienie stanu sieci wpłynie na sposób świecenia diody LED 1. Gdy tester zacznie pracować jako Stacja i zaloguje się we wskazanej sieci, dioda zacznie migotać szybciej. Gdy aktywny jest tryb softAP i utworzona zostaje własna sieć, dioda świeci dłużej i miga wolniej.



Rysunek 3. Wygląd wygenerowanej strony z menedżerem plików obszaru SPIFFS

Można także podłączyć się do gniazda USB na płycie ESP32 i uruchomić program monitora danych z portu, pracujący z szybkością 115200. Po resecie wysyłanych jest portem szeregowym płyty ESP mnóstwo informacji o konfiguracji, w tym tryb pracy, nazwa sieci i adres IP testera w ramach sieci. Zaś w czasie pracy wysyłane są informacje o aktualnym statusie urządzenia.

Kiedy aktywna jest opcja automatycznego uruchamiania po resecie pliku *index.html*, dostęp do menedżera plików SPIFFS jest zablokowany. Jeżeli dostęp do niego jest potrzebny, np. żeby zmodyfikować plik konfiguracyjny, można go uruchomić poprzez zwieranie do masy w czasie resetu portu IO16 płyty ESP32.

Praca z multitesterem

Po resecie powinien pojawić się ekran z elementami sterującymi **rysunek 4**. Zależnie od szerokości dostępnego ekranu elementy mogą być ułożone w poziomie lub w pionie. Początkowo pomiary są wstrzymane a większość klawiszy nieaktywna i wybielona. Po naciśnięciu START smartfon lub komputer powinien nawiązać komunikację z multitesterem w trybie websocket – gniazda internetowego. Można wtedy wybrać rodzaj pomiaru:

- Przycisk U(V) otwiera dostęp do listy wyboru wejścia pomiarowego:
  – Vx to wejście poprzez zaciski laboratoryjne,
  – Ax jest wejściem alternatywnym z zamontowanego po tej samej stronie gniazda śrubowego.
- O wyborze informuje świecenie albo diody LED D4, albo LED D5;
- Przycisk R(Ω) daje możliwość wyboru podzakresu pomiarowego oporników lub testu złącza półprzewodnikowego. Przewody pomiarowe należy podłączyć do wejść COM, Rx wyprowadzonych na 5-stykowe gniazdo śrubowe;
- Przycisk C(F) uaktywnia pomiary pojemności. Naciśnięcie REL włącza lub wyłącza kompensację pasożytniczej pojemności zniekształcającej odczyty pojemności. Wejścia COM i Cx;
- Przycisk L(H) inicjuje pomiary cewek. Wejścia COM i Lx;
- Przycisk T(°C) rozpoczyna komunikację z zewnętrznym czujnikiem DS18B20 i pomiary temperatury w stopniach. Jeżeli czujnik wymaga podłączenia stałego zasilania, oprócz podłączeń COM i Tx trzecie wyprowadzenie czujnika łączy się z wyprowadzeniem Rx;
- Naciśnięcie przycisku STOP kończy pomiary.

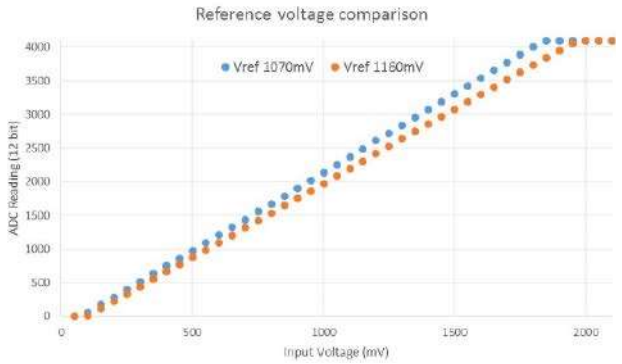
Rysunek 4. Wygląd wygenerowanej strony z interfejsem użytkownika

Gdy urządzenie przebywa w stanie nieaktywnym przez ponad 30 s, gniazdo websocked zamyka się.

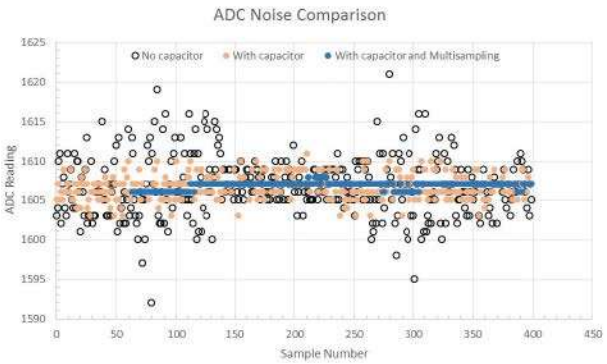
- Na koniec kilka wyników pomiarów porównawczych multitestera z typowym multimetrem:
- Pomiary porównawcze napięcia stałego w zakresie 0...26 V. Maksymalna różnica wskazań ±1,5%;
- Pomiary porównawcze oporności. Zakres 10...500 Ω, maksymalna różnica wskazań ±1,6%, zakres 500 Ω...10 kΩ, maksymalna różnica wskazań ±1,4%, zakres 10...500 kΩ, maksymalna różnica wskazań ±1,49%;
- Pomiary porównawcze pojemności. Zakres 3...10 nF, maksymalna różnica wskazań ±4,74%, zakres 10 nF...1 µF, maksymalna różnica wskazań ±0,4%, zakres 1...100 µF, maksymalna różnica wskazań ±2%.

Przetwornik analogowo-cyfrowy układu ESP32

Test napięć z wejść Vx, Ax jak i test oporności Rx opierają się na pomiarach napięcia przez wewnętrzny przetwornik ADC układu ESP32. Jego możliwości i ograniczenia wpływają na ostateczną precyzję pomiarów. W pomiarach użyty został jeden z dwu przetworników ADC oznaczony jako ADC1. Obydwa przetworniki są 12-bitowe i pracują z wewnętrznym napięciem odniesienia o nominalnej wartości 1100 mV, która może się zmieniać w poszczególnych egzemplarzach ESP32 od 1000 mV do 1200 mV. Dodatkowym ograniczeniem są wypłaszczenia krzywej pomiarów na obu krańcach. Przykładowy wygląd krzywych dla dwóch układów ESP32 różniących się wartością napięć odniesienia pokazano na **rysunku 5**.



Rysunek 5. Krzywa pomiarowa przetwornika ADC



Rysunek 6. Szumy odczytu przetwornika ADC

Listing 2. Procedury kalibracyjne w ramach inicjacji przetwornika ADC1

```
static esp_adc_cal_characteristics_t *adc_chars;

static const adc_channel_t channel  = ADC_CHANNEL_6;      //GPIO34 dla ADC1 wejście 6
static const adc_bits_width_t width = ADC_WIDTH_BIT_12;   //rozdzielczość 12 bit
static const adc_atten_t atten      = ADC_ATTEN_DB_11;    //tłumienie 11 dB
static const adc_unit_t unit           = ADC_UNIT_1;      //ADC1
/*
 * sprawdzenie ustawień efuse
 */
static void check_efuse(void){
    //Check if TP is burned into eFuse
    if (esp_adc_cal_check_efuse(ESP_ADC_CAL_VAL_EFUSE_TP) == ESP_OK) {
        printf("eFuse Two Point: Supported\n");
    } else {
        printf("eFuse Two Point: NOT supported\n");
    }
    //Check Vref is burned into eFuse
    if (esp_adc_cal_check_efuse(ESP_ADC_CAL_VAL_EFUSE_VREF) == ESP_OK) {
        printf("eFuse Vref: Supported\n");
    } else {
        printf("eFuse Vref: NOT supported\n");
    }
}
/*
 * ADC1 inicjacja
 */
void adc_Ini(void){
    check_efuse();
    adc1_config_width(width);
    adc1_config_channel_atten(channel, atten);
    //Characterize ADC
    adc_chars = calloc(1, sizeof(esp_adc_cal_characteristics_t));
    esp_adc_cal_value_t val_type = esp_adc_cal_characterize(
        unit, atten, width, DEFAULT_VREF,    adc_chars);
}
```

Istnieje możliwość poszerzenia zakresu pomiarowego przetwornika przez wykorzystanie wewnętrznych tłumików napięcia wejściowego. Przy zastosowaniu dostępnych tłumień i ze względu na kształt charakterystyk przetworników realne zakresy pomiarowe są następujące:

- tłumienie 0 dB 100...950 mV,
- tłumienie 2,5 db 100...1250 mV,
- tłumienie 6 dB 150...1750 mV,
- tłumienie 11 dB 150...2450 mV.

Podanie na wejście przetwornika napięcia spoza zakresu da wynik konwersji obarczony dużym błędem, np. dla różnych napięć otrzymany wynik będzie taki sam.

W skład systemowego oprogramowania IDF wchodzą procedury kalibracyjne, które pozwalają częściowo usuwać ograniczenia przetwornika. Możliwe jest odczytywanie z tzw. fuse bits fabrycznie zapisywanej w pamięci nieulotnej rzeczywistej wartości napięcia odniesienia. Na podstawie odczytanych wartości tworzone są formuły korekcyjne dostosowane do ustawionego tłumienia i kształtu charakterystyki. Procedury oprogramowania IDF udostępniają odczyty nie tylko surowych danych konwersji ADC, ale także obliczone i skorygowane wartości mierzonego napięcia w miliwoltach.

Na **listingu 2** pokazano procedury kalibracyjne w ramach inicjacji przetwornika ADC1. Na początku zadeklarowane są parametry ustawienia dla wejścia 6 przetwornika ADC1: rozdzielczość w bitach konwersji i tłumienie. Procedura *check_efuse()* podaje informacje o tym, jakie ustawienia fuse bits są dostępne w egzemplarzu płyty ESP32, na której uruchomiono oprogramowanie multitestera. Jako kolejne podczas inicjacji zostają ustawione parametry rozdzielczości i tłumienia dla wejścia 6 przetwornika. Po utworzeniu bufora na parametry wywołana zostaje sama procedura korekcyjna. Podana domyślna wartość napięcia odniesienia zostanie automatycznie zastąpiona rzeczywistą wartością zapisaną w fabrycznych ustawieniach fuse bits.

Przetwornik ADC jest podatny na wszelkiego rodzaju zakłócenia, co może zniekształcić odczyty – **rysunek 6**. W dokumentacji technicznej układu ESP32 proponowane są środki zaradcze na zaszumienie odczytów przez zastosowanie pojemności filtrujących 0,1 µF na używanych wejściach pomiarowych przetwornika oraz uśrednianie wyników. Procedura pomiarowa pokazana na **listingu 3** służy do pokazania, jak zastosować uśrednianie wielu pomiarów, a następnie jak za pomocą procedury korekcyjnej otrzymać przekształcony wynik pomiarów w miliwoltach.

Rozwój oprogramowania nowe wersje

Więcej informacji na temat testera, w tym kody sterujące przy pracy z zewnętrznym oprogramowaniem i pliki źródłowe, pojawi się w [6]. W planach są nowe wersje oprogramowania z dodatkowymi funkcjami: testerem zwarć, generatorem akustycznym, być może dołączanie dodatkowych zewnętrznych modułów rozszerzających, takich jak amperomierz.

Ryszard Szymaniak
biuro@ars.info.pl

Odnośniki:
[1] https://bit.ly/3rlkVkk
[2] https://bit.ly/3Cm3SVp
[3] https://bit.ly/3rtH8g3
[4] https://bit.ly/3dPN2oF
[5] https://bit.ly/3CmMekq
[6] https://ars.info.pl

Listing 3. Procedura pomiarowa z uśrednianiem wielu pomiarów

```
/*
 * pomiar mV dla 1 kanału ADC1
 */
uint32_t adc_mV_1kanal(adc1_channel_t channel)
{
    uint32_t adc_reading = 0;
    //Multisampling
    for (int i = 0; i < NO_OF_SAMPLES; i++) {
        if (unit == ADC_UNIT_1) {
            adc_reading += adc1_get_raw((adc1_channel_t)channel);
        } else {
            int raw;
            adc2_get_raw((adc2_channel_t)channel, width, &raw);
            adc_reading += raw;
        }
    }
    adc_reading /= NO_OF_SAMPLES;
    //Convert adc_reading to voltage in mV
    uint32_t voltage = esp_adc_cal_raw_to_voltage(adc_reading, adc_chars);
    return voltage;
}
```