BIBLIOTEKA
POLSKIEGO KRÓTKOFALOWCA

58

Krzysztof Dąbrowski
OE1KDA

Mini- i mikrokomputery w krótkofalarstwie

Tom 1

Wiedeń 2021

Mini- i mikrokomputery w krótkofalarstwie

Tom 1

Krzysztof Dąbrowski OE1KDA

Wydanie 1
Wiedeń, kwiecień 2021

Spis treści

Sommaire
Mini- et microordinateurs pour radio amateurs

Wstęp

Komputery różnych klas i rodzajów znalazły już od dłuższego czasu zastosowanie w krótkofalarstwie. Umożliwiają one prowadzenie łączności emisjami cyfrowymi RTTY, PSK31, Olivia, SSTV, FT8, JT65 i wieloma innymi w dowolnych pasmach amatorskich. Drugą szeroką dziedziną zastosowań jest korzystanie z odbiorników programowalnych (ang. SDR). Odbiorniki takie mogą być połączone lokalnie z komputerem użytkownika albo być dostępne internetowo. Po zainstalowaniu dodatkowych programów mogą także służyć jako bramki radiowo-internetowe APRS albo do odbioru komunikatów lotniczych. Trzecim polem jest prowadzenie łączności echolinkowych lub w systemach cyfrowego głosu D-Star, DMR, C4FM itd. W tym przypadku komputer służy jako urządzenie wejściowo-wyjściowe dla systemu komunikacji. Część trasy transmisji przebiega przez Internet (czasami także przez Hamnet), a reszta, począwszy od przemiennika, reflektora, kółeczka lub grupy rozmówców (ang. *room*) – drogą radiową. Łączności takie można rozumieć jako korzystające ze zdalnej obsługi przemiennika lub przemienników na drugim końcu łącza internetowego. Rozwiązania takie mogą być bardzo wygodne, ponieważ nie wymagają zabierania w podróż dodatkowego sprzętu jeśli jest to z innych względów utrudnione albo niepożądane. Możliwe jest także prowadzenie łączności z krajów, w których uzyskanie licencji jest trudne, kosztowne, wymaga długiego oczekiwania, jest praktycznie nie do zrealizowania albo grozi pobytem za kratkami.

Oprócz tego istnieje duża grupa programów pozwalających na śledzenie aktywności stacji DMR-owych, odbiór komunikatów APRS, podgląd na mapach internetowych odbieranych stacji APRS, programów diagnostycznych i innych pomocnych przy pracy krótkofalarskiej.

Wszystkie te zadania wymagają oczywiście zainstalowania odpowiedniego oprogramowania, którego omówieniu poświęcona jest znaczna część skryptu. Autor zakłada u czytelnika pewną znajomość omawianych tu emisji i systemów i dlatego w przypadku wątpliwości co do działania danego systemu transmisji, znaczenia parametrów, stosowanych podzakresów, przebiegu łączności, nadawanych raportów itp. warto zapoznać się z odpowiednimi tomami obecnej serii lub z inną literaturą. Spis opublikowanych dotąd tomów znajduje się na końcu skryptu.

Przy pracy emisjami cyfrowymi komputer może być połączony z radiostacją za pomocą odpowiedniego modemu albo w najprostszym razie sprzężony akustycznie przez zbliżanie mikrofonów do głośników. Stąd też w dalszym ciągu przedstawiono również niektóre rozwiązania modemów i innych układów pomocniczych. Dodatkowe atrakcyjne możliwości uzyskuje się również dzięki połączeniu komputerów androidowych z radiostacjami D-Starowymi itp. Umożliwia to przykładowo transmisję obrazów, pisma i innych danych.

W dalszym ciągu skryptu androidowe komputery tabliczkowe (ang. *tablet*) i telefony jak również te same urządzenia wyposażone w system iOS występują dla uproszczenia pod wspólną nazwą minikoputerów – również z racji niewielkich rozmiarów i znacznych mocy obliczeniowych. Pod nazwą mikrokomputerów rozumiane są natomiast *Maliny* i podobne rozwiązania, Arduino i układy z mikroprocesorami ESP8266, PIC, AVR i podobnymi. Duża ilość programów dla systemu Windows wymagałaby osobnego potraktowania i dlatego w niniejszym tomie są one wspominane jedynie na marginesie, przykładowo gdy istnieją windowsowe odpowiedniki omawianych tu programów albo jeśli jest to związane ze współpracą z mikro- i minikomputerami.

Dla ograniczenia objętości i tematyki skryptu w pierwszym rzędzie omawiany jest dokonany przez autora wybór programów związanych z prowadzeniem łączności fonią i innymi emisjami cyfrowymi, a także niektórych programów pomocniczych. Autor zrezygnował z prezentowania programów obliczeniowych służących do projektowania i symulacji układów elektronicznych albo anten, programów przeznaczonych do zdalnego sterowania stacji, sterujących obrotnicami, urządzeniami dodatkowymi do radiostacji, pomiarowymi, programów przeznaczonymych do nauki telegrafii itp.

Sprawy programowania mikrokomputerów Arduino, *Maliny* i podobnych zostały przedstawione w stopniu minimalnym mogącym ułatwić czytelnikom znającym inne języki programowania zrozumienie kodu programów albo w celu przypomnienienia niektórych spraw, które mogły ulecieć z pamięci, albo tych które w wydawnictwach poświęconych ogólnym sprawom programowania są rzadziej omawiane j.np. problematyka przerwań. Wyczerpujące kursy programowania Arduino, Pythona itd. znajdą czytelnicy w wydawnictwach innych autorów.

Jako uzupełnienie lektury proponujemy tomy 20, 21 poświęcone rozwiązaniom dla Arduino, 24 poświęcony *Malinie*, 17 i 33 omawiające sprawy radiolatarniom na mikrokontrolerach i telemetrii oraz 57, w którym przedstawiono autonomiczne nadajniki WSPR małej mocy.
W tomach 1, 26, 326, 34 i 19 znajdą natomiast czytelnicy podstawowe informacje dotyczące systemów cyfrowego dźwięku (D-STAR, DMR, C4FM) i Echolinku, a w tomie 8 – transmisji danych pozycyjnych i komunikatów APRS.

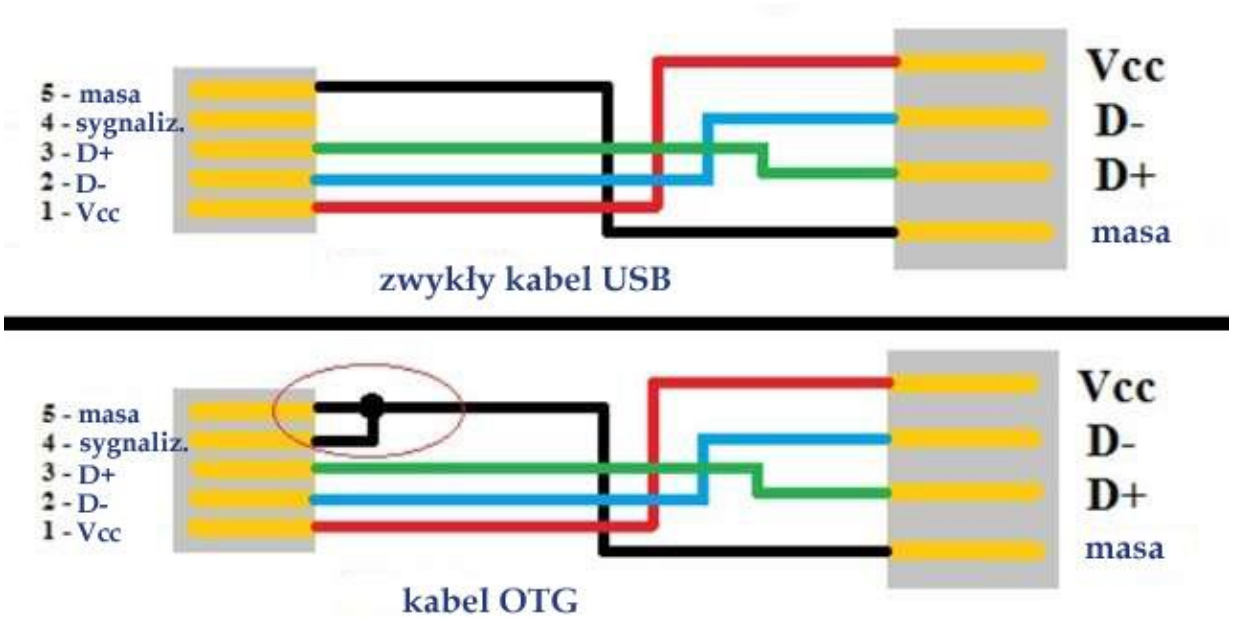
Krzysztof Dąbrowski
Wiedeń, 24 kwietnia 2021

1. Komputery androidowe

1.1 Android

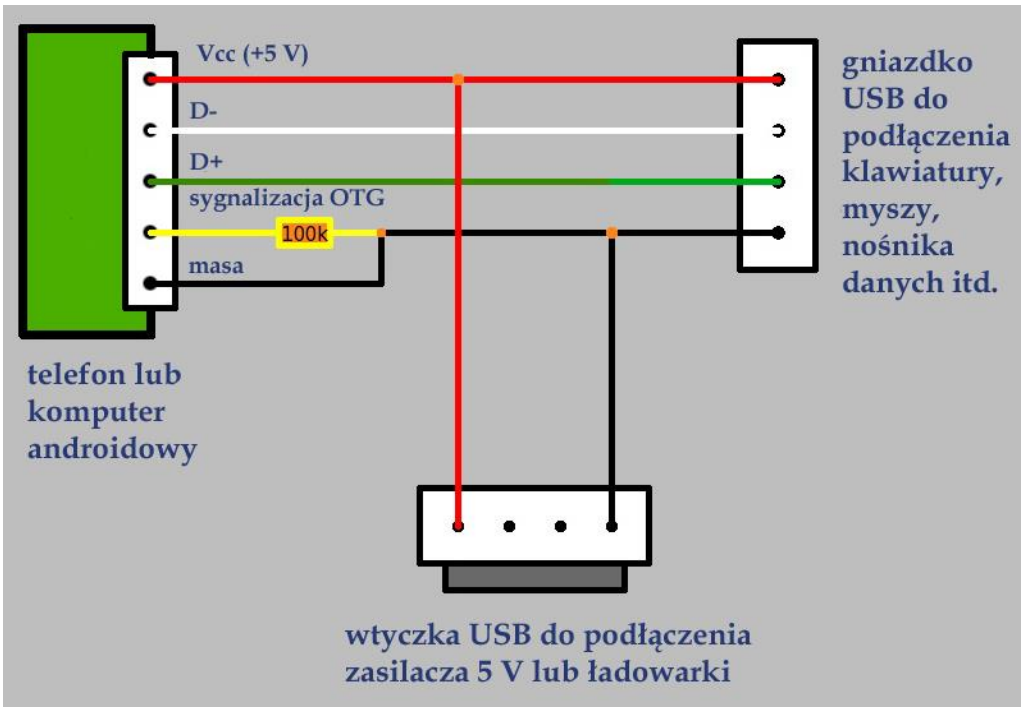
System operacyjny Android jest rozpowszechniony i znalazł zastosowanie na komputerach tabliczkowych (ang. *tablet*) i inteligentnych telefonach (ang. *smartphone*) wszystkich marek poza produktami firmy Apple. Korzystanie z niego jest stosunkowo proste i intuicyjne dzięki graficznej powierzchni obsługi i ekranom dotykowym. Możliwe jest także podłączenie przez złącze USB (OTG) lub Bluetooth klawiatury i myszy. W odróżnieniu od komputerów windowsowych użytkownik nie ma pełnych praw administratora (*root*) i nie zawsze może usunąć programy zainstalowane fabrycznie. Oprogramowanie dotatkowe (pliki z rozszerzeniem .apk) do standardowego najwygodniej jest pobrać ze sklepu internetowego *Google Play*, ale możliwe jest także instalowanie programów pochodzących z innych źródeł. Uzyskanie praw administratora przez użytkowników (ang. *rooting*) jest wprawdzie oficjalnie niepożądane ale również możliwe (wymaga to zainstalowania dodatkowego uzupełnienia systemu w rodzaju *Magiska.*.
Począwszy od wersji 6 Android jest wyposażony standardowo w prosty administrator (*menażer*) plików pozwalajacy na kopiowanie, przesuwanie i kasowanie plików i zmiany ich nazw.
Wymiana plików z innymi komputerami jest możliwa przy użyciu zewnętrznych nośników USB albo przez lokalną sieć WiFi. Niektóre z nośników są nawet wyposażone w dwie wtyczki USB – małą pasującą do urządzenia androidowego i standardową typu A do połączenia z komputerami windowsowymi. W przypadku ogólnym podłączenie zewnętrznego nośnika danych lub innych urządzeń dodatkowych wymaga użycia kabla USB-OTG (ang. *USB On-The-Go*).

1.1.1. Kable OTG

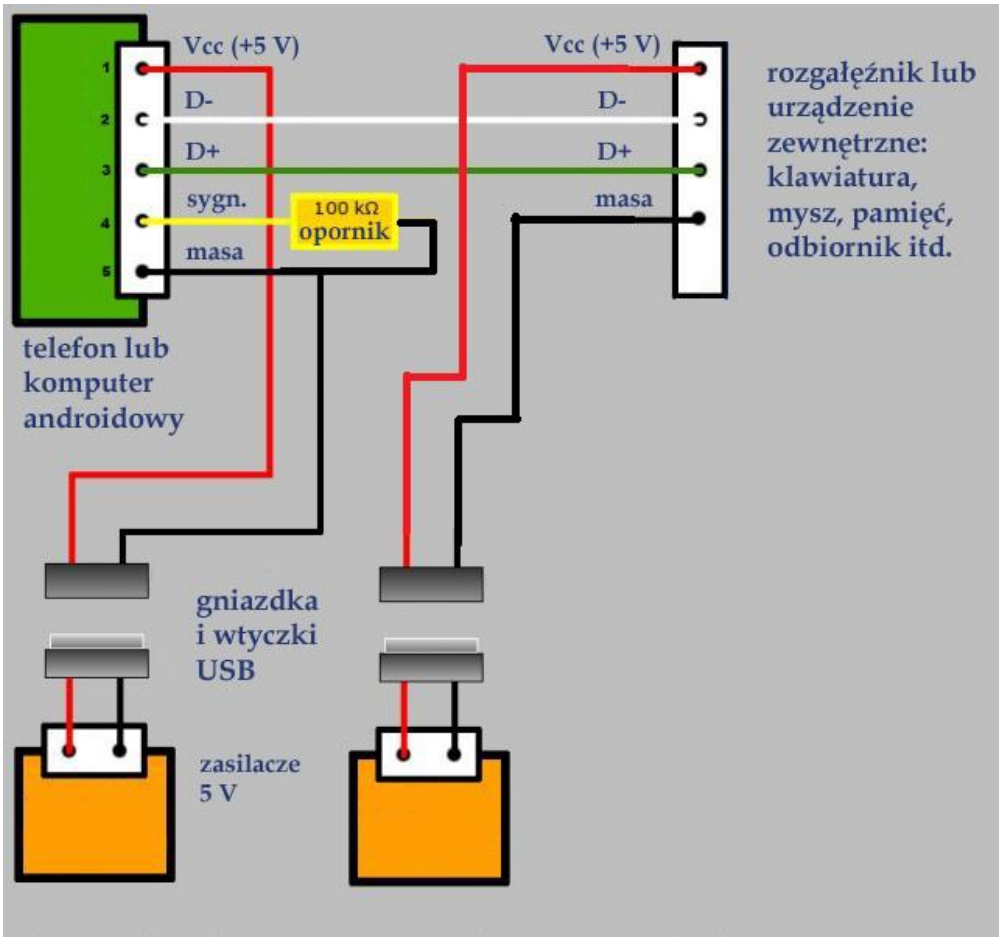


Rys. 1.1.1.1. Różnice w połączeniach wewnątrz zwykłego kabla USB i kabla OTG polega na połaczeniu przewodu sygnalizacji (4) do masy w kablu OTG

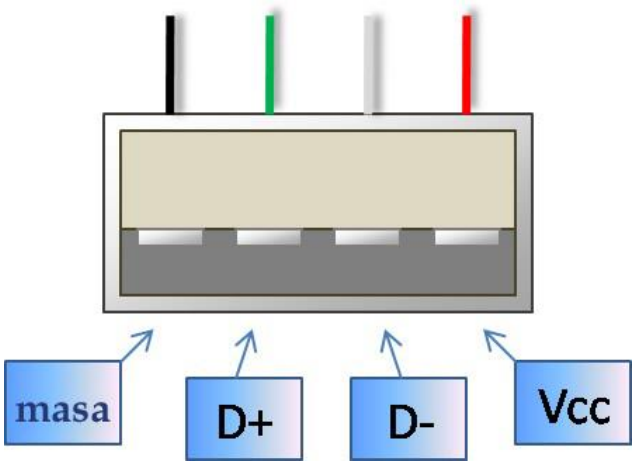
Dzięki połączeniu kontaktu sygnalizacyjnego nr 4 z masą włożenie kabla do gniazdka powoduje przełączenie telefonu lub komputera w tryb komputera nadrzędnego (ang. *host*). Pozwala to na podłączenie do złącza USB dowolnych urządzeń zewnętrznych takich jak klawiatura, mysz, pamięci USB, odbiorniki programowalne (SDR) i podobne. Jedynie niektóre starsze urządzenia nie dysponują taką możliwością. Oprócz zwykłych kabli o połączeniach przedstawionych na schemacie 1.1.1.1 dostępne są także kable rozgałęzione pozwalające na zasilanie urządzenia peryferyjnego z dodatkowego zasilacza i zmniejszenia dzięki temu obciążenia komputera.



Rys. 1.1.1.2. Rozgałęziony kabel OTG pozwalający na zasilanie urządzenia dodatkowego z zasilacza zamiast z akumulatora. Ładowanie akumulatora w komputerze jest możliwe dzięki połączeniu przewodów 4 i 5 przez opornik 100 kΩ



Rys. 1.1.1.3. Połączenie do ładowania akumulatora z ładowarki od kompletu i niezależne zasilanie urządzenia dodatkowego z dowolnego innego zasilacza 5 V. Ładowanie wymaga połączenia przewodów 4 i 5 przez opornik 100 kΩ



Rys. 1.1.1.4. Kolejność kontaktów w zwykłej wtyczce USB typu A patrząc od przodu. W gniazdku jest to oczywiście kolejność odwrotna

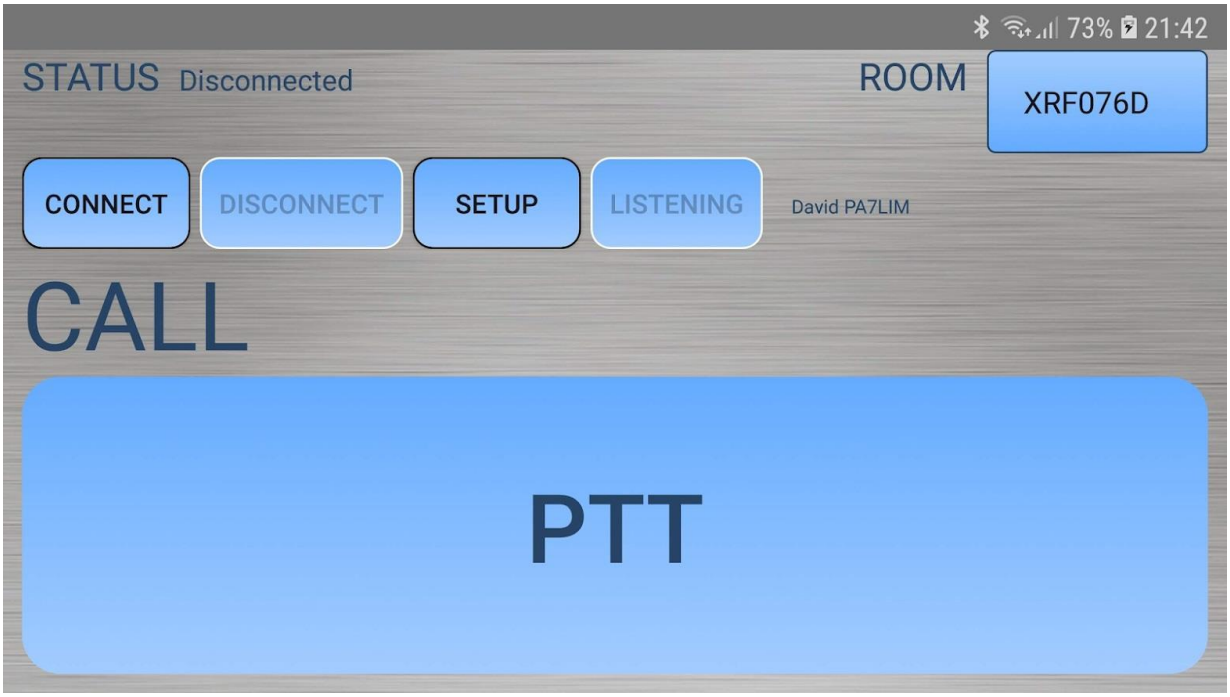
1.1.2. „FK File Commander”

Dostępny bezpłatnie w sklepie internetowym *Google Play File Commander* jest przykładem praktycznego administratora plików umożliwiającego dostęp do każdego pliku na komputerze androidowym w jego ogólnej pamięci, w ewentualnej dodatkowo zainstalowanej pamięci SD, na nośnikach USB-OTG lub w chmurze internetowej. Program informuje również użytkownika o zajętości poszczególnych „dysków” pamięci i pozostającym na nich wolnym miejscu. Oczywiście posiada on wszytkie typowe możliwości jak poszukiwanie plików, kopiowanie, kasowanie, zmiany nazw itp. Co jest szczególnie wygodne umożliwia on także wymianę plików z komputerami PC w domowej sieci WiFi przy użyciu zwykłej przeglądarki internetowej, czyli bez konieczności instalowania na PC jakichkolwiek dodatkowych programów.
Po zainstalowaniu dodatkowego rozszerzenia *File Commander* pozwala także na konwersję plików do innych formatów. W wersji profesjonalnej możliwe jest także szyfrowanie danych.
Użytkownicy pragnący przekazać niewielką liczbę plików na dowolny inny własny komputer mogą przesłać je sami do siebie pocztą elektroniczną jako załączniki i odebrać je na komputerze docelowym. Rozwiązanie to nie wymaga użycia żadnych kabli (a więc jest przydatne gdy niema ich pod ręką), nie wymaga też instalowania *File Commandera* ani podobnych programów i nie ogranicza się do lokalnej sieci. Przesłanie większej liczby plików może być jednak żmudne.
W sklepie internetowym można znaleźć również inne programy tego rodzaju.

1.2. Łączności D-Starowe przez „Peanuta”

Komputerowy dostęp do sieci D-Starowej pozwala na zapoznanie się z systemem bez korzystania z radiostacji i może ułatwić podjęcie decyzji o ewentualnym zakupie sprzętu albo okazać się wygodnym rozwiązaniem w czasie pobytu poza domem. Korzystanie z niego pozwala na prowadzenie łączności amatorskich także w czasie pobytu za granicą, w kraju, w którym byłoby trudne lub nawet niemożliwe uzyskanie licencji krótkofalarskiej.
Rozwiązania takie jak „Peanut” i inne przedstawione dalej programy mogą budzić wprawdzie pewne wątpliwości, do jakiego stopnia jest to jeszcze krótkofalarstwo, ale można traktować je również jako sposób zdalnej obsługi odległych stacji przemiennikowych. Jeżeli więc dalsza część połączenia od zdalnej stacji przemiennikowej do korespondenta przebiega radiowo to można je zaliczyć do krótkofalarstwa.
Opracowany przez PA7LIM program „Penaut” funkcjonuje zasadniczo podobnie jak znany już od dawna program echolinkowy korzystając z wbudowanego mikrofonu i głośniczka. Inne rozwiązania wymagają zastosowania lokalnego wokodera. Program dostępny w wersjach dla systemów Windows i Androida obsługuje obecnie jedynie łączności w systeme D-STAR. Prowadzenie łączności w sieci D-

Starowej wymaga uprzedniego zarejestrowania się w niej. Rejestracja jest bezpłatna i dokonuje się jej tylko jeden raz. Pozwala ona na korzystanie z dostępu w dowolny sposób – radiowo lub komputerowo. Dla korzystania z reflektorów skrośnych D-STAR/DMR konieczna jest również rejestracja w sieci DMR. Szczegóły podane są w tomach 1, 26 i 326 „Biblioteki" (patrz spis na końcu skryptu).
Rozwiązanie nie wymaga wyposażenia komputera we własny wokoder ani inne urządzenia dodatkowe. Przekodowywanie dźwięku z PCM na standard D-Starowy i z powrotem odbywa się na serwerach sieciowych wyposażonych w wokoder AMBE3000. Wokoder ten nadaje się do użytku również w systemach DMR oraz YAESU C4FM i znalazł już zastosowanie w innych konstrukcjach amatorskich takich jak „Portable AMBE Server", „DVMEGA DVstick 30", „NWDR ThumbDV" itd.



Fot. 1.2.1. Okno robocze programu. Po jego uruchomieniu należy wybrać pożądany reflektor i nacisnąć przycisk połączenia („Connect"). Przycisk „Setup" służy do wywołania okna konfiguracji, „Disconnect" do rozłączenia się z reflektorem, a „PTT" – do nadawania

Sam program jest dostępny bezpłatnie w sklepie internetowym *Google Play*, ale jego wykorzystanie wymaga (również bezpłatnej) rejestracji pod adresem [1.2.2]. Użytkownik po wprowadzeniu tam imienia, adresu poczty elektronicznej i znaku wywoławczego otrzymuje kod, który musi wpisać w oknie programu dodatkowo do znaku wywoławczego, po czym należy nacisnąć przycisk „Submit" (fot. 1.2.2). Rejestracja ma na celu niedopuszczenie do pracy w eterze osób nielicencjonowanych, analogicznie jak rejestracja w Echolinku. Użytkownik musi być również zarejestrowany w sieci D-STAR, a w przypadku korzystania z reflektorów połączonych z grupami DMR konieczna jest również rejestracja i w tej sieci. Korzystanie z połączeń skrośnych z siecią C4FM nie wymada dodatkowej rejestracji w tej sieci.
Okno programu (fot. 1.2.1) zawiera przyciski służące do połączenia się z wybranym reflektorem lub kółeczkiem (grupą rozmówców), do rozłączenia się, oraz przycisk nadawania. Czas nadawania jest ograniczony, a pozostała do końca reszta jest wyświetlana na ekranie na przycisku nadawania. Przycisk nadawania przyjmuje kolor czerwony w trakcie transmisji, zielony przy odbiorze sygnału i niebieski w pozostałym czasie.

Fot. 1.2.2. Wprowadzanie kodu rejestracyjnego. Po jego wprowadzeniu należy potwierdzić dane za pomocą przycisku „Submit”

W widocznej u góry po prawej stronie głównego okna rozwijanej liście wybierany jest pożądany reflektor albo internetowa grupa rozmówców (tab. 1.2.1). O ile łączności prowadzone przez reflektory są transmitowane radiowo przez połączone z nimi przemienniki, o tyle grupy internetowe są dostępne jedynie przez komputer. W trakcie rejestracji możliwe jest podanie ulubionego, najczęściej używanego reflektora, co może przyczynić się do jego udostępnienia w programie. Niektóre dostępnych reflektorów są połączone z grupami rozmówców sieci DMR albo reflektorami C4FM, a więc właściwie można korzystać i z nich, ale tylko w ograniczonym stopniu.
Oprócz telefonów i komputerów androidowych (z wersją 4.0 lub nowszą) program pracuje także na internetowej „radiostacji” TM-7 firmy Inrico i na zbliżonych modelach. Dla wersji windowsowej „Peanuta” wymagany jest system „Windows” 10.
Pulpit programu (ang. *dashboard*) – *http://peanut.pa7lim.nl* – zawiera wiele przydatnych informacji. Wśród nich jest lista prowadzonych akurat łączności, co oszczędza długiego poszukiwania aktywności na reflektorach (fot. 1.2.3), spis odbieranych stacji (fot. 1.2.4) oraz spisy dostępnych reflektorów i grup (tab. 1.2.1). Pojawianie się w spisach oznaczeń typu N0CALL może wynikać z wejścia stacji przez inną sieć np. przez sieć DMR do D-Starowej. Oczywiście użytkownik programu może wybrać dowolny reflektor i nadawać na nim wywołanie.

Fot. 1.2.3. Spis bieżących łączności na pulpicie „Peanuta”

Fot. 1.2.4. Spis odbieranych stacji

Początkowo wśród grup internetowych istniała równiez grupa przeznaczona dla polskich krótkofalowców, ale najprawdopodobniej nie cieszyła się ona dostatecznym powodzeniem i dlatego ustąpiła miejsca innym bardziej uczęszczanym. Krótkofalowcy polscy mogliby więc spotykać się w nie przypisanej do żadnego kraju grupie THE-CAFE lub w innej akurat nie używanej (albo na mało używanym reflektorze) jeśli nie poszukują właśnie łączności z którymś krajem.

Tabela 1.2.1
Grupy – kółeczka, ang. *room* – i reflektory (stan z kwietnia 2021). Grupy internetowe są dostepne wyłącznie komputerowo i nie mają wyjścia w eter

Oznaczenie „Penaut”	Moduły lub grupy D-Star, DMR, C4FM	Kraj
DCS001G	DCS001G	Bawaria (Niemcy)
DSC002H	DCS002H	Haiti
DCS015A	DCS015A	Niemcy, połączenia z grupami IPSC2 (DMR+), reflektorami C4FM i NXDN
DCS018D	DCS018D	Hiszpania
DCS905A	DCS905A	Austria, reflektor skrośny, XLX232, XLX022, DMR4001
DCS905Y	DCS905Y	Austria, reflektor skrośny, XLX232, XLX022, DMR 4025, YSF
DCS945A	DCS945A	Niemcy, reflektor skrośny
DCS945Y	DCS945Y	Niemcy, reflektor skrośny
DMR10002	DMR	Kanada, DMR IPSC2 (DMR+)
DMR4382	DMR	Meksyk, reflektor skrośny
REF018B	REF018B	Brazylia
REF030C		USA, nie połączony z reflektorem REF030C
REF037C	REF037C	USA, Orlando

REF069C	REF069C	USA, Connecticut
REF075C	REF075C	Hiszpania, kraje latynoskie
REF084C	REF084C	Francja
REF520T	REF520T	Tajlandia, reflektor skrośny D-STAR/DMR
XLX037A	XLX037A	USA, DX-LINK2
XLX039B	XLX039B	Włochy, skrośny, TG22292 (BM)
XLX039R	XLX039R	Włochy, Toskania, reflektor skrośny
XLX043A	XLX043A	USA, ARES
XLX051H	XLX051H	Hiszpania, DMR TG214
XLX099U	XLX099U	Włochy, Apulia
XLX117D	XLX117D	Urugwaj, reflektor skrośny D-STAR-DMR-C4FM
XLX190A	XLX190A	Argentyna, reflektor skrośny D-STAR-DMR-YSF
XLX201B	XLX201B	Wielka Brytania
XLX213A	XLX213A	Trynidad i Tobago
XLX270B	XLX270B	Luksemburg
XLX287A	XLX287A	Nowa Zelandia, reflektor skrośny DMR TG530287
XLX301C	XLX301C	Nowa Zelandia
XLX302D	XLX302D	Kanada, Ontario
XLX339B	XLX339B	Niemcy
XLX391S	XLX391S	Włochy, DMR TG2242
XLX469B	XLX469B	USA, reflektor skrośny
XLX508J	XLX508J	Niemcy, skrośny, XLX508J/BM-26447/YSF26447
XLX602M	XLX602M	Lotaryngia
XLX606B	XLX606B	USA, Lomond
XLX606C	XLX606C	USA, Lomond
XLX750S	XLX750S	Nowa Zelandia, skrośny, TG530530, AllaStar 46803, Echolink ZL1OZ-L
XLX774A	XLX774A	Kanada, Montreal
XLX888C	XLX888C	Włochy, skrośny, TG222994/HBL 11881/YSF IT ZONA9
XLX888D	XLX888D	Włochy, skrośny, TG222111 BM
XLX899A	XLX899A	W. Brytania
XRF021B	XRF021B	Niemcy, skrośny TG262810
XRF059A	XLX059A	USA
XRF115A	XRF115A	Szwajcaria, skrośny, franc.-jęz.
XRF125H	XLX125H	Węgry
XRF227B	XRF227B	Rumunia
XRF229D	XRF229D	Szwajcaria
XRF231H	XRF231H	Włochy, Sardynia, reflektor skrośny, TG2231 BM, YSF92365, WIRES-X 47439
XRF241B	XRF241B	Hiszpania
XRF266Y	XRF266Y	Hiszpania
XRF268E	XLX268E	Portugalia
XRF268F	XLX268F	Portugalia
XRF295A	XRF295A	USA
XRF299B	XRF299B	N. Zelandia, reflektor skrośny
XRF299K	XRF299K	N. Zelandia, reflektor skrośny
XRF299R	XRF299R	N. Zelandia, skrośny YSF75161, BM TG530080
XRF334B	XLX334B	USA
XRF359B	XRF359B	Bułgaria
XRF421D	XRF421D	Niemcy, reflektor skrośny BM TG26429
XRF466A	XRF466A	USA
XRF502B	XRF502B	Honduras
XRF502C	XLX502C	Honduras

XRF503C	XRF503C	San Salwador
XRF522D	XRF522D	Malezja, reflektor skrośny BMTG50210
XRF600E	XRF600E	W. Brytania
XRF625P	XRF625P	USA
XRF706C	XRF706C	Włochy, reflektor skrośny, TG222773
XRF706G	XLX706G	Włochy, połączenie z grupą TG2230 DMR+/BM
XRF750D	XRF750D	N. Zelandia, reflektor skrośny BM TG53099
XRF766D	XLX766D	Brazylia
XRF930U	XLX930U	Włochy
XRF989P	XLX989P	USA
XRF991D	XLX991D	Włochy, reflektor reflektor skrośny, BM TG22400, Wires-X 41984, YSF 18255
XRF991U	XLX991U	Włochy, skrośny, BM TG2244
XRF997C	XLX997C	Włochy
XRF997M	XLX997	Włochy, reflektor skrośny, BM TG2236
XRFCATV		Rumunia
YSF-5158		Filipiny
YSF-ALME		Hiszpania
YSF-ANDA		Hiszpania, Andaluzja
YSF-AR		Argentyna
YSF-ARMI		Filipiny, DX1ARM
YSF-CQUK		Połączenie W. Brytania – Australia
YSF-EA8		Hiszpania, W. Kanaryjskie
YSF-EURO		Hiszpania
YSF-IRE		Irlandia, reflektor skrośny, DMR TG 2724
YSF-KPHN		Filipiny
YSF-NWUK		W. Brytania, reflektor skrośny YSF13344
YSF-OMGA		Filipiny
YSF-P555		Filipiny
YSF-PARA		Filipiny
YSF-PERU		Peru, reflektor skrośny
YSF5153		Filipiny, reflektor skrośny, TG5153
YSF800		Bułgaria, XLX800B
YSFAUACT		Australia, Canberra
YSFBSPOM		USA, OMG Los Angeles
YSFCHARC		Kanada, reflektor skrośny, Wires-x 28104
YSFCNARC		Filipiny, DX1ARC
YSFCOQUI		USA, reflektor skrośny, BM TG31631, YSF 88084
YSFCVARN		Filipiny, DX2ACV
YSFEUREG		Belgia
YSFFDCCU		USA
YSFFILMR		USA
YSFINDIA		Indie
YSFPAHUB		Panama
YSFPINOY		Filipiny
YSFSPAIN		Hiszpania, YSF49810
YSFXLX52		Filipiny, XLX552 – BM TG5152
112SPAIN		Hiszpania, łączności ratunkowe, grupa internetowa
3CHARLIE		Chile, grupa internetowa, DX3CHARLIE
ARABIC		Arabska grupa internetowa
ARAGON-R		Grupa internetowa krótkofalowców aragońskich
ARGENTIN		Argentyna, grupa internetowa
AUSSIE		Australia, grupa internetoa

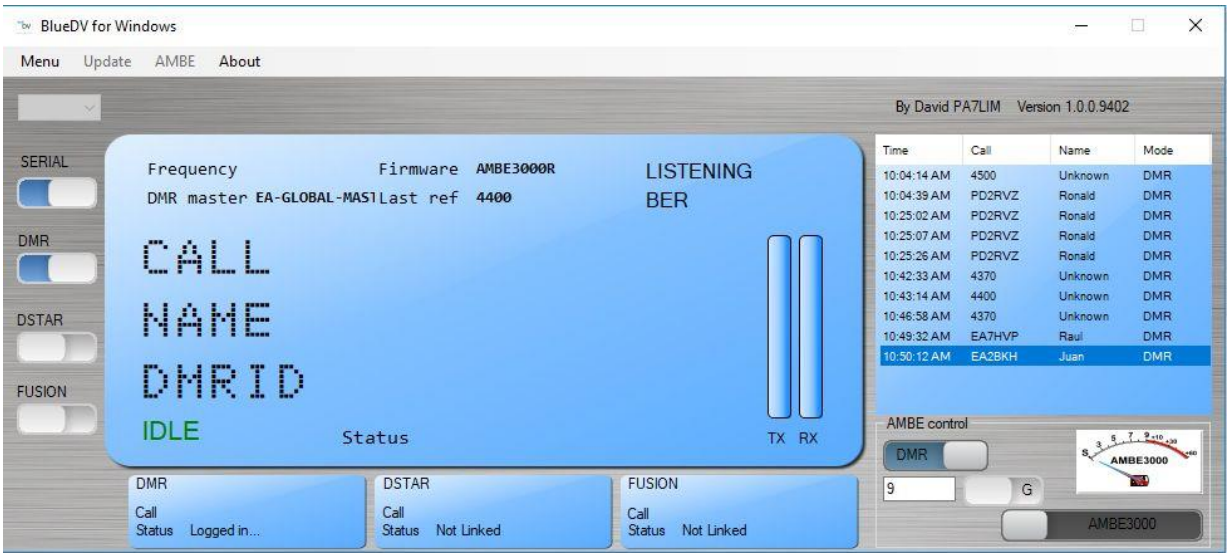
AUSTRIA		Austria, grupa internetowa
BRASIL		Brazylia, grupa internetowa
CARIBE-3		Grupa internetowa Kuby, Republiki Dominikańskiej i Puerto Rico
CATALAN		Katalonia, grupa internetowa
CHILE		Chile, grupa internetowa
CHINESE		Chińska grupa internetowa
CLAMANCHA		Kastylia la Mancha, grupa internetowa
COLOMBIA		Kolumbia, grupa internetowa
CQCANADA		Kanada, TGIF/YSF, grupa internetowa
CWONLY		Grupa internetowa wyłącznie CW, nie dla mowy
CW-SPAIN		Hiszpańska grupa internetowa dla ćwiczących telegrafię
DANISH		Dania, grupa internetowa
DUTCH		Holandia, grupa internetowa
EACW		Hispania, grupa internetowa do nauki telegrafii
ECHOECHO		Serwer echa, grupa internetowa
EMERG		Holenderska grupa internetowa do łączności ratunkowych
ENGLISH, ENGLISH1		Anglia, USA itd., grupy internetowe
EQUADOR		Ekwador, grupa internetowa
FOTOCHAT		Grupa internetowa do dyskusji o fotografii
FRENCH		Francuska grupa internetowa
GALICIA		Grupa internetowa hiszpańskiej Galicji
GERMAN, GERMAN1		Austria, Niemcy, Szwajcaria, grupy internetowe
GREECE		Grecja, grupa internetowa
HAMCAMIN		Grupa internetowa Ham Cam International (HCI)
HBL1, HBL48455	DMR	Honduras, DMR, HBLink
HBL21430	DMR	Hiszpania, DMR, HBLink
HBL27246	DMR	W. Brytania, DMR, HBLink
HBL8	DMR	Niemcy, DMR, HBLink
HEBREW		Hebrajska grupa internetowa
ILOCANO		USA, Ilocano, grupa internetowa
IMAGEQSO		Transmisje obrazów, FT8 i innych emisji cyfrowych, także rozmowy, grupa internetowa
INDIA		Indyjska grupa internetowa
IRLAND		Irlandzka grupa internetowa
ITALIAN		Włochy, grupa internetowa
JAPANESE		Japońska grupa internetowa
KOREA		Korea, grupa internetowe
KREYOLHT		Kreolska grupa inteernetowa
LATINOS		USA, latynoska grupa internetowa
MADEIRA		Grupa internetowa krótkofalowców z Madery
MALTESE		Grupa internetowa krótkofalowców z Malty
MANITALK, MATULIS		Filipińskie grupy internetowe
MEXICO		Meksyk, grupa internetowa
NEPAL		Nepal, grupa internetowa
NW-UK		Brytyjska grupa internetowa
PORTUGAL		Portugalia, grupa internetowa
PSK-SSTV		Grupa internetowa do łączności PSK, SSTV
QO-100		Grupa internetowa poświęcona łącznościom na QO-100

RC-UTIEL		Hiszpania, grupa internetowa operatorów z klubu UTIEL
RE-DMR		Hiszpańska grupa intenetowa poświęcona DMR
RE-SPAIN		Hiszpańska grupa internetowa *RADIOENLACES.NET*
RF-TEST		Amerykańska grupa internetowa do prób
RUSSIAN		Rosja, grupa internetowa
SOTA		SOTA, grupa internetowa
SOUTHAFR		Południowa Afryka, grupa internetowa
SPANISH		Hispania, grupa internetowa
TAGALOG		Grupa internetowa Tagalog
TECHTALK		technika, grupa internetowa
TEXTQSO		Grupa internetowa TTY, PSK, JS8CALL i do rozmów
TGF1335	DMR	Kanada, DMR
TGF1972	DMR	Trynidad i Tobago, YSF 81520, XLX172A, XRF172C, TGIF1972
TGF204	DMR	USA, grupa poświęcona technice
TGF248	DMR	Wielka Brytania, TG248
TGF36250	DMR	Wielka Brtania, Northampton Digital Radio Group
TGF3720	DMR	Haiti TG3720
TGF4412	DMR	USA TG4412
THAI		Tajlandzka grupa internetowa
THE-CAFE		Ogólna grupa internetowa
TORCW		Hiszpańska grupa internetowa do ćwiczeń telegrafii
TURKISH		Turcja, grupa internetowa
UK-CHAT		Brytyjska grupa internetowa
USA		Amerykańska grupa internetowa
VE2VPS-C	VE2VPS A	Kanadyjski reflektor, Montreal

1.3. Łączności D-Starowe przez „BlueDV”

BlueDV zapewnia połączenia z reflektorami D-Starowymi, DMR-owymi i C4FM korzystając z podłączonego lokalnie wokodera. W obu przypadkach nie potrzeba korzystać z radiostacji – tak jak w komputerowych łącznościach przez Echolink.
O ile przedstawiony w poprzednim punkcie *Peanut* korzysta z wokoderów podłączonych do serwera sieciowego o tyle *BlueDV* tego samego autora wykorzystuje lokalne (indywidualne) wokodery i w ten sposób może łączyć się z dowolnymi reflektorami wszystkich systemów (DCS, REF, XRF, XLX, BM, DMR+, YSF, FCS). Oprócz wersji dla Androida i Windows istnieje także ograniczona pod względem funkcjonalności wersja dla systemu iOS 10, 11 lub nowszych – nie obsługująca jednak reflektorów D-Starowych – oraz wersja dla Linuksa działająca także na *Malinie*. Oprócz wersji przeznaczonej jedynie do prowadzenia indywidualnych łączności istnieje również oprogramowanie serwera AMBE udostępniające usługę przekodowywania również innym urządzeniom tego samego użytkownika albo innym.
Najnowszym rozwiązaniem wokodera jest *DVMEGA DVstick33* podłączany do złącza USB i pracujący pod oprogramowaniem *BlueDV AMBE3003 Server*. Jest on oparty o wokoder AMBE3003 dysponujący trzema dupleksowymi kanałami cyfrowego dźwięku – odpowiada to trzem wokoderom starszego typu AMBE3000 – a więc możliwe jest równoległe korzystanie z niego na dwóch lub trzech komputerach albo telefonach komórkowych. Oprogramowanie dla Windows znajduje się w witrynie [1.3.1]. Wokoder AMBE3003 jest przystosowany do potrzeb systemów telefonii komórkowej i pracuje w trybie pakietowym, w którym dane dźwiękowe i skomprymowane dane kanałowe są transmitowane przez to samo złącze. Rozróżnia on automatycznie mowę i przerwy, może w nich dodawać sygnał szumów symulujący szumy analogowe, zapobiegając wrażeniu wystąpienia przerwy w łączności. Wokoder posiada także funkcję tłumienia przeszkadzających odgłosów otoczenia i dekoduje tony DTMF. W sieci D-Starowej są one używanie do łączenia się z reflektorami jako alternatywna możliwość w stosunku do poleceń zapisanych tekstowo w pamięci radiostacji, przykładowo poleceniu DCS002GL odpowiada

sekwencja tonów D207, a REF032AL – *32A. Wokoder AMBE3003 jest kompatybilny z systemami D-STAR, DMR, dPMR, C4FM i APCO25.



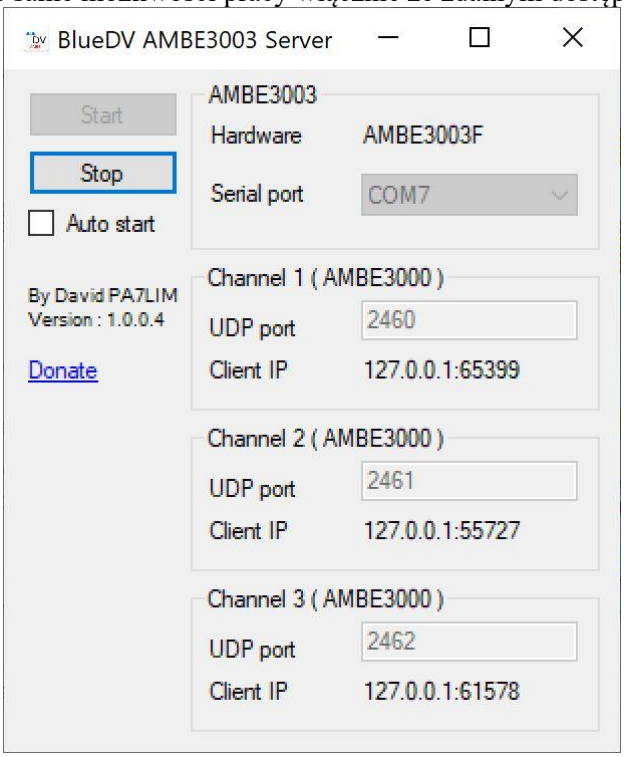
Rys. 1.3.1. Okno główne *BlueDV* dla Windows w trakcie połączenia C4FM

Rys. 1.3.2. Okno w wersji anroidowej

Oprócz niego w sprzedaży są wokodery USB *DVMEGA DVstick 30* [1.3.3], *NWDR ThumbDV* [1.3.4] i *ZUM AMBE Board* [1.3.5] wyposażone w obwód DVSINC AMBE3000. Ten ostatni model może być połączony z komputerem także przez WiFi albo Ethernet [1.3.3]. Możliwe jest także podłączenie do niego wyświetlacza OLED.

Praktycznym rozwiązaniem jest *Portable AMBE Server* [1.3.6, 1.3.7]. Ten pracujący autonomicznie serwer-wokoder D-STAR/DMR jest połączony z domową siecią WiFi i może być wykorzystywany lokalnie przez windowsowe i androidowe wersje *BlueDV*, a także zdalnie przez Internet (rys. 1.3.5). W tym przypadku konieczne jest posiadanie statycznego adresu IP albo skorzystanie z takich usług sieciowych jak *noip* czy *DynDNS* dla zapewnienia stałej dostępności z zewnątrz. Dostęp wokodera do Internetu można uzyskać także korzystając z funkcji punktu dostępowego (ang. *hotspot*) telefonu komórkowego. Warto jednak zawczasu upewnić się czy nie pociąga to za sobą dodatkowych kosztów.

Z serwerem może być połączone tylko jedno urządzenie naraz (komputer lub telefon komórkowy). Wokoder mieszczący się w obudowie 9 x 4,5 x 2,5 cm jest zasilany napięciem 5 V ze standardowego zasilacza USB i pobiera prąd 300 mA. Również *DVMEGA DVstick 30* w połączeniu z *Maliną* tworzy serwer AMBE i daje takie same możliwości pracy włącznie ze zdalnym dostępem przez Internet.

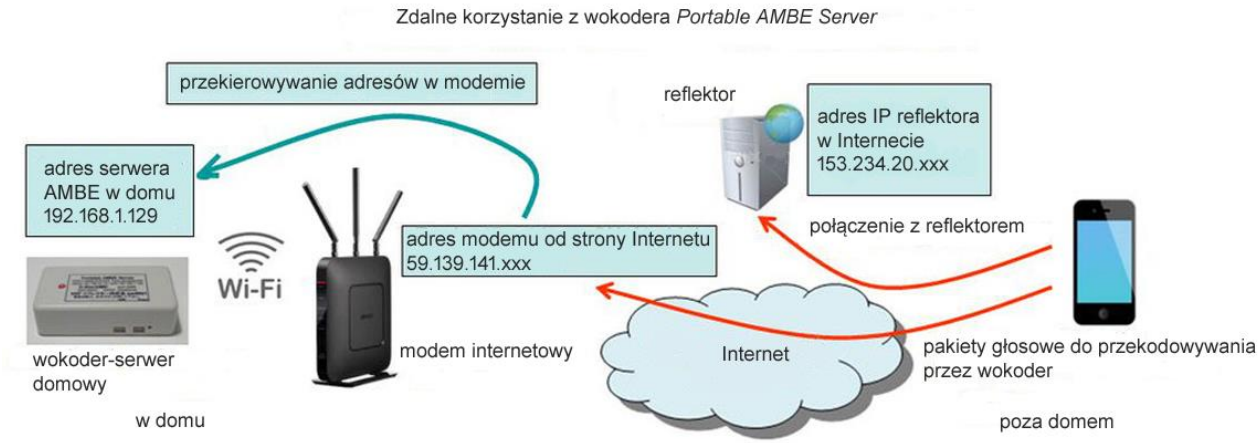


Rys. 1.3.3. Konfiguracja oprogramowania serwera AMBE3003

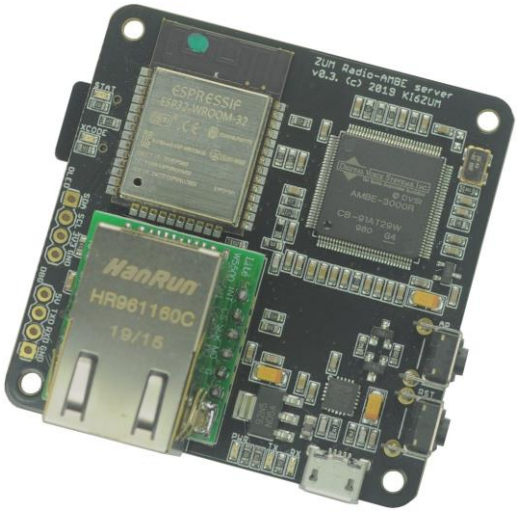
Rys. 1.3.4. Schemat blokowy stacji z *BlueDV*

Na tym nie kończą się możliwości programu *BlueDV*. Połączenie androidowego komputera lub telefonu z radiostacją DMR wyposażoną w oprogramowanie wewnętrzne OpenGD77 (Radioddity GD-77, GD-77S, Tytera MD-760, MD-730, Baofeng DM-1801, DM-860, RD-5R itd.) tworzy punkt dostępowy do sieci DMR. Współpraca *BlueDV* z radiostacją wymaga uruchomienia programu pośredniczącego TCPUART. Do połączenia radiostacji z komputerem używany jest kabel służący do jej programowania

wraz z przejściówką OTG z miniaturowego gniazdka USB na standardowe pasujące do wtyczki USB typu A na kablu. Od tego rozwiązania są jednak wygodniejsze warianty mikroprzemienników korzystające z OpenSpota czy MMDVM. Są one też bardziej uniwersalne gdyż nie ograniczają się tylko do sieci DMR-owej. Do pierwszych kontaktów z cyfrowym dźwiękiem lepszy jest jednak *Peanut* ponieważ nie wymaga dodatkowego wyposażenia i związanych z tym wydatków.



Rys. 1.3.5. *Portable AMBE Server* przydaje się w trakcie pracy poza domem



Fot. 1.3.6 (po lewej). Wokoder ZUM-AMBE3000
Fot. 1.3.7 (po prawej). Wokoder USB DVMEGA oparty o AMBE3003

Fot. 1.3.8. *Portable AMBE Server* składa się z *Maliny* z dodaną płytką wokodera

1.4. Łączności DMR i C4FM w programie „Droidstar”

Droidstar autorstwa AD8DP pozwala na dostęp do pozostałych amatorskich sieci cyfrowego głosu: DMR (BM, IPSC2), C4FM (reflektorów YSF, FCS), NXDN, P25, do węzłów *AllStar* i do opracowanego przez polskich krótkofalowców systemu M17. W odróżnieniu od *Peanuta* i *Blue-DV* korzystających ze sprzętowego wokodera AMBE (przy czym pierwszy z nich używa wokoderów połączonych z serwerem PA7LIM, a drugi lokalnych u użytkownika) *Droidstar* wykorzystuje wokoder programowy. Sytuacja prawna tego rozwiązania jest dość niepewna i najprawdopodobniej nie jest ono licencjonowane przez DVSI. Jednocześnie ze strony operatorów sieci DMR zgłaszane są zastrzeżenia dotyczące niewystarczającej zgodności protokółu stosowanego przez program z przyjętym standardem. Miałoby to stanowić nawet zagrożenie dla bezpieczeństwa w sieci Brandmeistra i ułatwiać podszywanie się użytkowników pod innych. W związku z tym wprowadzono od 1 marca 2021 r. obowiązek korzystania z indywidualnych haseł dostępu przez prywatne mikroprzemienniki (ang. *hotspot*). Nie dotyczy to sieci IPSC2 (DMR+). Hasła dostępu do sieci BM definiuje się w witrynie *brandmeister.network* w punkcie *SelfCare*.
Pomimo tych zastrzeżeń interesujące jest wypróbowanie *Droidstara*. Pracujący pod Androidem program pozwala wprawdzie także na korzystanie z sieci D-Starowej jednak jakość dźwięku jest na tyle zła, że nie należy tego próbować nadawczo, a jedynie do nasłuchów. Jest to związane z faktem, że D-STAR korzysta ze starszej wersji wokodera AMBE.

Rys. 1.4.1. Okno główne programu Rys. 1.4.2. Okno konfiguracyjne programu

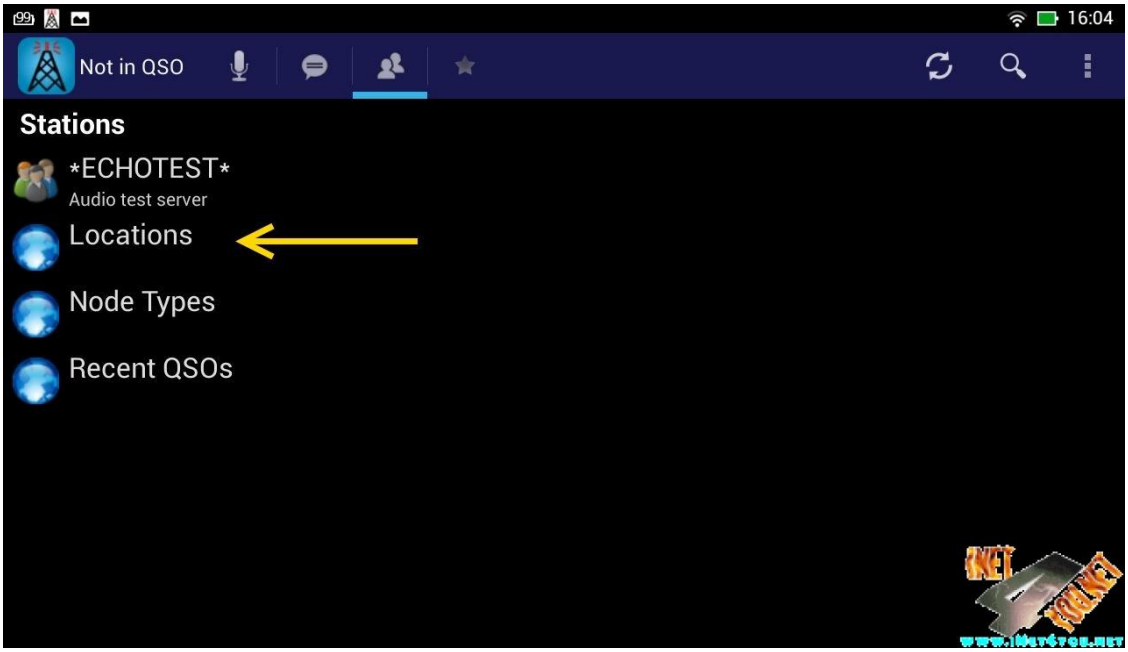
Program jest stosunkowo prosty w obsłudze. W oknie konfiguracji należy wprowadzić własny znak wywoławczy w polu *callsign* i w przypadku korzystania z sieci DMR-owych także identyfikator DMR w polu DMRID. Rejestracja w sieci DMR jest opisana w tomach 26 i 326 „Biblioteki”. Dla sieci Brandmeistra konieczne jest też podanie indywidualnego hasła. Do pracy w sieci C4FM wystarczy

podanie znaku. W przypadku równoległego korzystania z własnego mikroprzemiennika identyfikator DMR należy uzupełnić o rozszerzenie podawane w polu ESSID. Jest ono liczbą dwucyfrową z zakresu 00 – 99 i służy do jednoznacznej identyfikacji urządzeń korzystających z tego samego identyfilatora DMR. Rozszerzenie musi być jednoznaczne dla każdego z nich, przy czym jedno z czynnych równolegle urządzeń może korzystać z identyfikatora bez rozszerzenia.
Po wpisaniu danych konfiguracyjnych w oknie łączności wybierany jest system transmisji (dla nasłuchu w sieci D-Starowej rodzaj reflektora: REF, DCS, XRF, numer albo oznaczenie reflektora lub numer grupy rozmówców w sieciach DMR. Połączenie uzyskuje się po naciśnięciu przycisku *Connect*, a rozłączenie za pomocą przycisku *Disconnect*. Po uzyskaniu połączenia z wybranym celem u dołu ekranu wyświetla się niebieski przycisk nadawania. Przy naciśnięciu zmienia on barwę na czerwoną i na ekranie wyświetlane są dodatkowe informacje, takie jak znaki albo identyfikatory nadawcy i adresata, a także kolejne numery pakietów danych.
Droidstar jest dostępny bezpłatnie w sklepie internetowym *Google-Play*. Wymaga on *Androida* w wersji 5 lub nowszej.

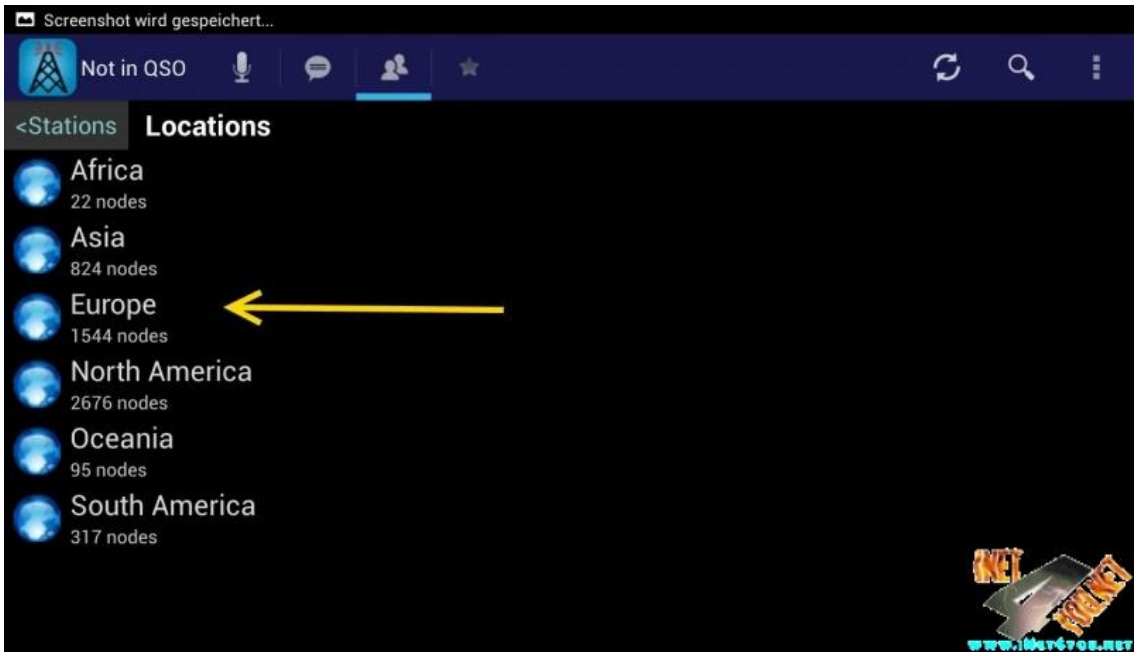
1.5. Łączności echolinkowe

Pomimo rozpowszechnienia się systemów cyfrowego głosu *Echolink* jest w dalszym ciągu atrakcyjnym rozwiązniem dla łączności komputerowo-radiowych. Pozwala na nawiązywanie połączeń z okolicami, w których nie ma jeszcze przemienników cyfrowego głosu i gdzie krótkofalowcy nie są wyposażeni w taki sprzęt. Mogą to być przykładowo takie egzotyczne kraje jak wyspy karaibskie, kraje Ameryki Środkowej, kraje azjatyckie, a nawet stosunkowo bliskie kraje jak Gruzja i Armenia.
Korzystanie z *Echolinku* przez komputer wymaga zarejestrowania się i uwierzytelnienia licencji w sposób opisany w tomie 19 obecnej serii. Po jednorazowym i bezpłatnym zarejestrowaniu się w sieci instalacja programów echolinkowych na dalszych dowolnych komputerach lub telefonach wymaga jedynie podania znaku i ustalonego w trakcie rejestracji hasła, które warto sobie gdzieś zapisać. Oprócz wersji windowsowej istnieją wersje oprogramowania dla Andoida i iOS.
Użytkownicy *Echolinku* mają do wyboru połączenia z dowolnie wybranymi stacjami przemiennikowymi (rozpoznawanymi przez rozszerzenie -R w znaku), simpleksowymi stacjami pomocniczymi (z rozszerzeniem -L), z użytkownikami wchodzącymi do sieci przez Internet (znaki bez rozszerzenia) i z konferencjami – grupami lub kółeczkami dyskusyjnymi. Stacje z rozszerzeniami -R i -L pozwalają na prowadzenie łączności przebiegających dalej drogą radiową. Kółeczka konferencyjne (ang. *conference*) noszą przeważnie skrócone nazwy i oznaczenia informujace o głównej tematyce rozmów. Wśród nich jest też serwer *ECHOTEST* o adresie 9999 przeznaczony do prób i oceny jakości własnego dźwięku.
Echolinkowe wycieczki w dalsze okolice globu są przy dostępie komputerowym o tyle wygodniejsze, że użytkownicy wybierają na ekranie części świata, następnie położone w nich kraje i na koniec znajdujące się na ich terytorium stacje. Nie muszą pamiętać wielocyfrowych adresów stacji i wprowadzać ich za pomocą klawiatury DTMF na radiostacji. W odróżnieniu od wersji dla Windows użytkownik może być połączony tylko z jedną stacją w danej chwili. Spośród kilku możliwych trybów w pracy przy korzystaniu z dostępu do sieci przez telefon komórkowy albo przez WiFi najkorzystniejszym jest tryb „Relay” korzystający z serwerów pośredniczących w dostępie do stacji echolinkowych. Nie wymagane są zmiany w konfiguracji modemu dostępowego do Internetu, odblokowywanie kanałów logicznych (ang. *port*) itp. co poza domem jest przeważnie niemożliwe. Połączenia między użytkownikami komputerowymi nie są zawsze możliwe, jest to zależne od serwerów „Relay”, z którymi są połączeni. Czasami połączenie udaje się uzyskać po kilku próbach. Można także korzystać z publicznego serwera „Proxy” albo obaj korespondenci mogą połączyć się z którąś z akurat nie używanych stacji przemiennikowych aby nikomu nie przeszkadzać.
Oprócz tego dostępny jest tryb bezpośredni („Direct”) wymagający udostępnienia kanałów logicznych 5198 i 5199 dla UDP oraz tryby korzystające z publicznych lub prywatnych serwerów pośredniczących „Proxy”.
Echolink dla Androida jest oferowany bezpłatnie w sklepie internetowym *Google Play*. Może on być zainstalowany na dowolnej liczbie komputerów, ale nie wolno korzystać z niego równolegle na dwóch

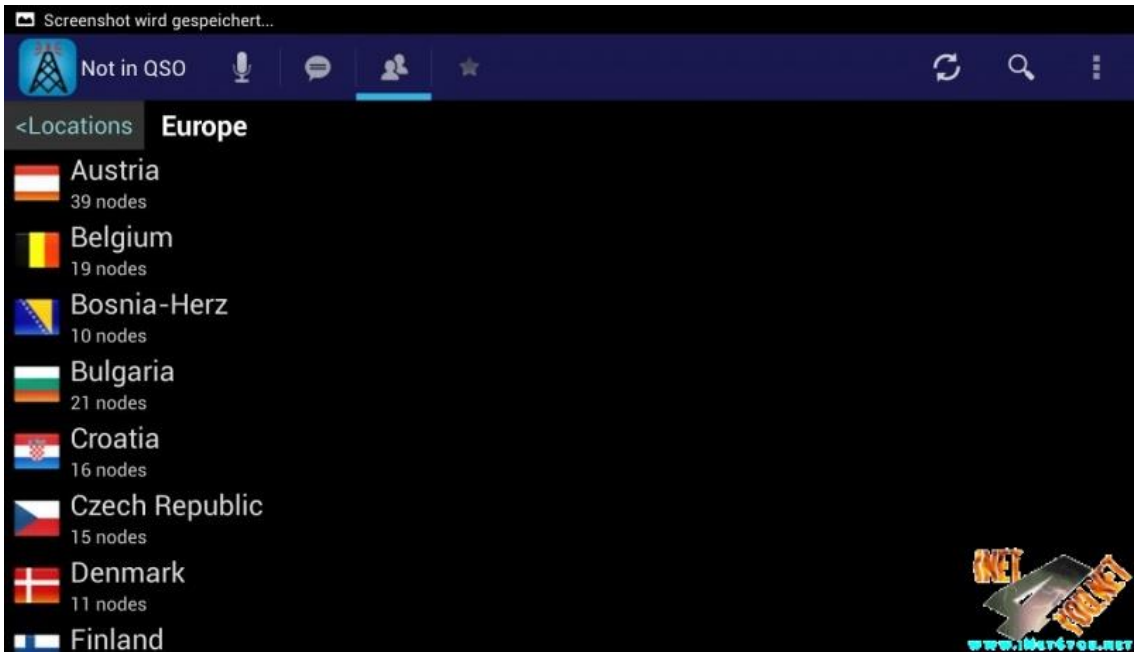
lub wiecej urządzeniach. Nie można także korzystać równolegle z Echolinku na dwóch lub więcej komputerach nawet przy różniących się znakach wywoławczych jeżeli korzystają one z tego samego publicznego adresu IP. W przypadku korzystania z Echolinku na tym samym komputerze naprzemian przez dwie osoby lub więcej można założyć oddzielne profile dla każdego znaku wywoławczego.
Spis aktywnych stacji jest odświeżany w pewnych odstępach czasu, dlatego też możliwa jest sytuacja, w której jakaś stacja jest jeszcze widoczna w spisie na ekranie ale jest już nieosiagalna w rzeczywistości. Użytkownik otrzymuje wówczas meldunek o przekroczeniu czasu podejmowania prób połączenia („timed out"). Każda ze stacji echolinkowych może też odmówić przyjęcia połączenia i jest to zależne od ustawień dokonanych przez jej operatora lub jej zajęcia przez inne połączenia. Liczba uczestników kółeczek konferencyjnych jest także ograniczona. Liczba stacji połączonych ze stacjami występującymi w spisie jest podawana w nawiasie po znaku wywolawczym.
W przypadku zapomnienia hasła należy zażądać usunięcia go na stronie *www.echolink.org* i następnie wprowadzić nowe. W tej samej witrynie można także skorygować dane użytkownika takie jak adres elektroniczny, znak wywoławczy itp.
Grupy dyskusyjne (kółeczka) o tematyce krótkofalarskiej można znaleźć także na serwerach sieci *Teamspeak 3* i *Zello*, ale nie są to rozwiązania typowo krótkofalarskie, w odróżnieniu od *Echolinku*. Odpowiednie programy komunikacyjne są dostępne w sklepie internetowym *Google Play*.



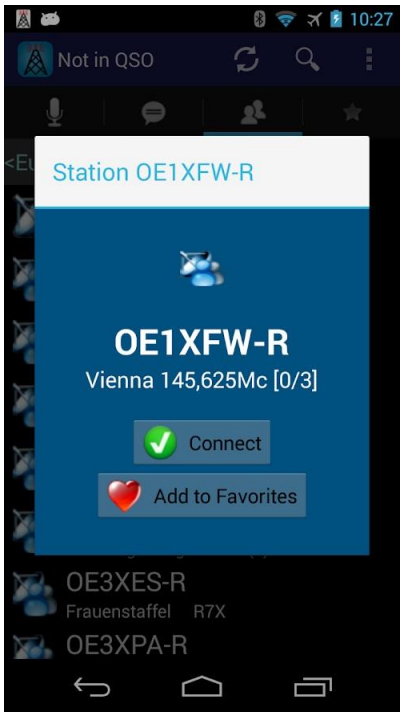
Rys. 1.5.1. Po wywołaniu programu użytkownik ma do wyboru spis stacji uporządkowany według lokalizacji, według rodzajów stacji (-R, -L, użytkownicy komputerowi, konferencje) albo spis znaków ostatnich korespondentów. Na górze znajduje się wywołanie serwera próbnego *ECHOTEST*. Zaokrąglone strzałki w górnym pasku służą do ponownego wczytania (aktualizacji) spisu stacji, lupa – do poszukiwania znaków, a pod wielokropkiem ukrywają się dalsze punkty. Wśród nich jest okno konfiguracyjne

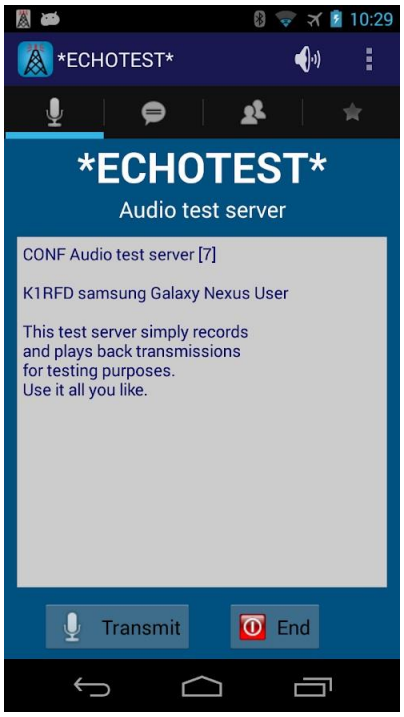


Rys. 1.5.2. Kolejnym oknem w spisie według lokalizacji jest okno wyboru kontynentów



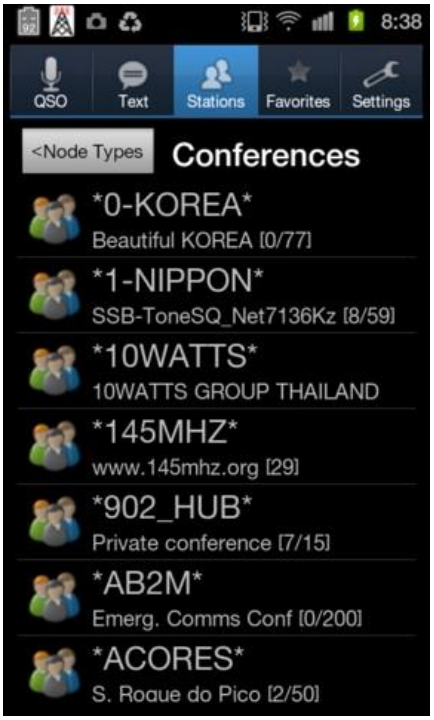
Rys. 1.5.3. Dla wybranego kontynentu wyświetla się spis krajów z podannymi pod ich nazwami liczbami czynnych stacji





Rys. 1.5.4 (po lewej). Po wybraniu kraju wyświetlany jest spis czynnych stacji, a po naciśnięciu pożądanej stacji na ekranie pojawia się okienko pozwalające na połączenie się z nią (przycisk „Connect") albo na dodanie jej do ulubionych (przycisk „Add to Favorites"). Tym razem jest to ujęcie z ekranu telefonu, a nie komputera jak poprzednie

Rys. 1.5.5 (po prawej). Okno dla połączenia posiada dwa przyciski: nadawczy („Transmit") i przerywający połączenie („End"). Na szarym polu tekstowym wyświetlana jest informacja o stacji i połączonych z nią korespondentach (w przykładzie jest to serwer echa). Po przerwaniu połączenia można wrócić do spisu stacji lub lokalizacji. W oknie wyświetlanym poziomo przyciski znajdują się po lewej stronie



Rys. 1.5.6 (po lewej). Ekran w trakcie nadawania. U dołu znajduje się paskowy wskaźnik poziomu modulacji. Przejście na odbiór wymaga naciśnięcia mikrofonu na ekranie

Rys. 1.5.7 (po prawej). Spis konferencji

Bezpłatny i nie zaśmiecony reklamami program *EcholinkFinder* znajduje w oparciu o położenie geograficzne stacji najbliższe przemiennniki echolinkowe i wyświetla ich lokalizacje na mapie. Nie jest to wprawdzie funkcja niezbędna do pracy w *Echolinku* za pomocą komputera, ale przydaje się w przypadku korzystania z przemienników drogą radiową. Po naciśnięciu na symbol stacji na mapie wyświetlane są dodatkowe informacje o niej: znak wywoławczy, częstotliwość pracy, ton CTCSS, adres węzła w sieci Echolinku, dokładne współrzędne geograficzne, moc nadajnika, rodzaj anteny itd.
Użytkownik może także wprowadzić sam lokalizację stacji lub oprzeć się na odczytanej przez komputer ze źródeł wybranych w jego ustawieniach.
Prowadzenie łączności za pośrednictwem *Echolinku* drogą radiową nie wymaga żadnej rejestracji, a jedynie posiadania ważnej licencji amatorskiej – jak w każdym przypadku wyjścia w eter. Użytkownik znajdujący się w zasięgu przemiennika połączonego z *Echolinkiem* może za pośrednictwem klawiatury DTMF nadać do przemiennika rozkaz połączenia z dowolnie wybranym węzłem sieci (przemiennikiem, konferencją itp.) i po nawiązaniu połączenia prowadzi łączność tak, jak przez zwykły przemiennik FM, tyle tylko, że na większe odległości. Zaleca się zachowywanie między relacjami odstępów 3-sekundowych, aby dać innym użytkownikom interenetowym szansę włączenia się do rozmowy. W radiostacjach nie posiadających własnej klawiatury DTMF można przeważnie w zapisać w pamięci pewną liczbę kodów DTMF (w tym przypadku adresów echolinkowych) i wywoływać wybrany z nich do połączenia ze stacją echolinkową. Można także użyć dodatkowej klawiatury DTMF albo programu komputerowego generującego te sygnały. W trakcie nadawania kodów komputer lub klawiatura są trzymane w pobliżu mikrofonu radiostacji.
Program *DVSwitch Mobile* umożliwia nawiązywanie techniką echolinkową (VoIP) połączeń z siecią *AllStarLink*.

1.6. D-Starowa transmisja obrazów

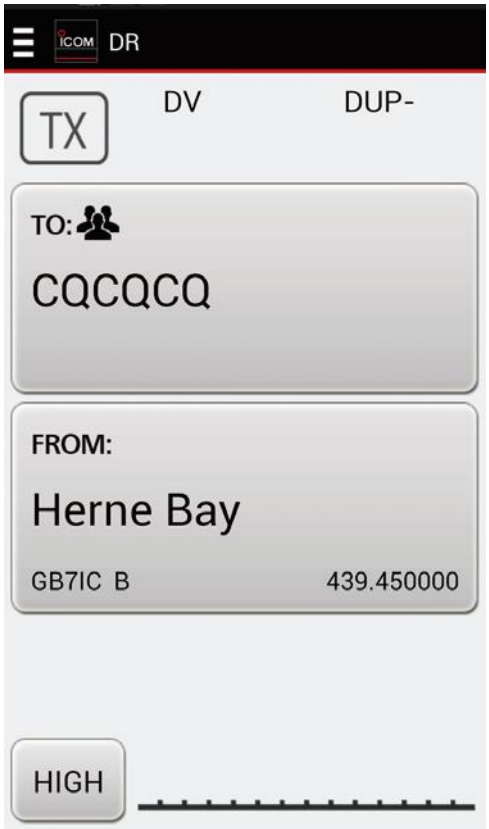
Icomowski program RS-MS1A dla Androida w wersji 5.0 lub nowszej umożliwia wymianę obrazów i tekstów między stacjami D-Starowymi, a także konfigurację najważniejszych ustawień radiostacji. Komputer musi dysponować funkcją OTG na złączu USB. Do połączenia z radiostacją dla większości modeli konieczny jest kabel OPC-2350LU, w przypadku modeli wyposażonych w łącze Bluetooth takich jak IC-4100E i IC-5100E można korzystać z niego zamiast z połączenia kablowego. Główne okno programu przedstawia rys. 1.6.1.

Rys. 1.6.1. Okno główne RS-MS1A

Ponumerowane na nim punkty oznaczają kolejno:
1 [DR] – pozwala na ustawianie zawartości pól [FROM] (przemiennika wejściowego) i [TO] (adresu docelowego) dla funkcji DR.
2 [Share Pictures] – umożliwia wymianę obrazów ze standardową lub zwiększona szybkością transmisji, w zależności od modelu radiostacji, i wyświetlanie ich na ekranie komputera androidowego.
3 [Text Messaging] – umożliwia transmisję tekstów i wyświetlanie ich na ekranie komputera.

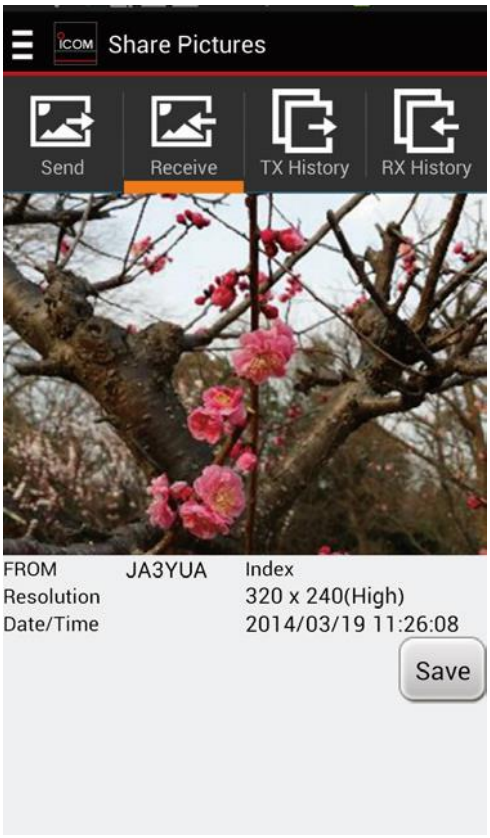
4 [Mapping] – wyświetla na mapie pozycje korespondenta lub stacji przemiennikowej. Możliwe jest także ustawianie zawartości pól [FROM] i [TO].
5 [RX History] – wyświetla informacje o odbieranej stacji D-Starowej i stacjach odbieranych poprzednio.
6 [Your Call sign] – służy do wprowadzania i modyfikacji znaków korespondentów dla łączności D-Starowych.
7 [Transceiver Setting] – pozwala na zmianę niektórych, ale nie wszystkich, parametrów konfiguracyjnych radiostacji.
8 [Application Setting] – pozwala na zmianę ustaień programu, takich jak np. wyświetlanych jednostek.
9 [Import] – umożliwia wczytanie spisu przemienników („Repeater List") i spisu znaków docelowych (Your Call").
10 [Export] – wysłanie spisu przemienników, znaków docelowych i spisu odebranych stacji („RX History").
11 [Bluetooth Connection] – połączenie zradiostacją przez łącze Bluetooth.
12 [Exit Application] – zakończenie pracy programu.

1. W trybie DR użytkownik ma możliwość wprowadzania danych do pól adresu docelowego [TO] i przemiennika wejściowego [FROM] (rys. 1.6.2). Do rozpoczęcia nadawania konieczne jest naciśnięcie przycisku na radiostacji
2. Funkcja wymiany obrazów pozwala w punkcie nadawania [Send] na nadanie zdjęć zrobionych aparatem fotograficznym komputera lub plików graficznych zapisanych na nim uprzednio. Po naciśnięciu czarnego pola w oknie nadawczym (rys. 1.6.3) przez sekundę użytkownik może wybrać dowolny plik jpg z komputera albo uruchomić aparat fotograficzny. W polu tym może znajdować się symbol zdjęcia gór i słońca jak na ilustracji, ale może też go nie być. W przypadku gdy w oknie nadawczym widoczny jest poprzednio nadawany obraz należy nacisnąć przez sekundę na jego powierzchnię. W polu odbiorcy („Receiver") można podać znaki adresatów. Obraz jest widoczny u wszystkich odbierających sygnał D-Starowy nadawczy.

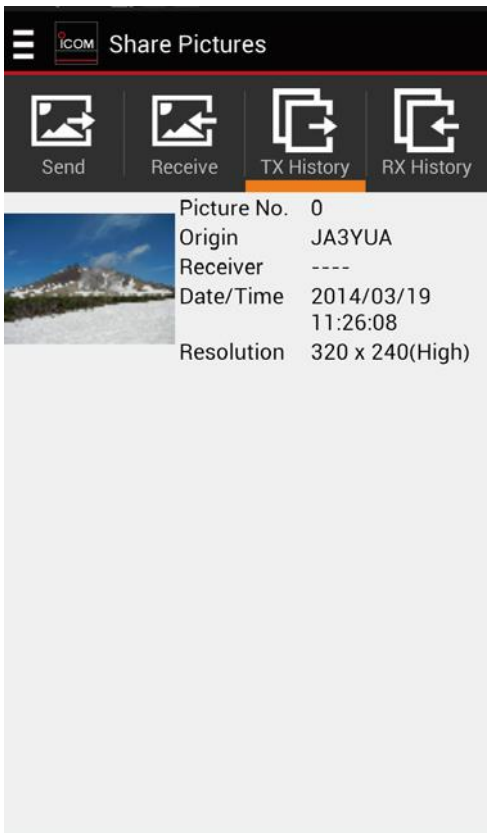


Rys. 1.6.2. Okno DR

Rys. 1.6.3. Okno nadawcze obrazów



Rys. 1.6.4. Okno odbioru obrazów



Rys. 1.6.5. Okno historii transmisji

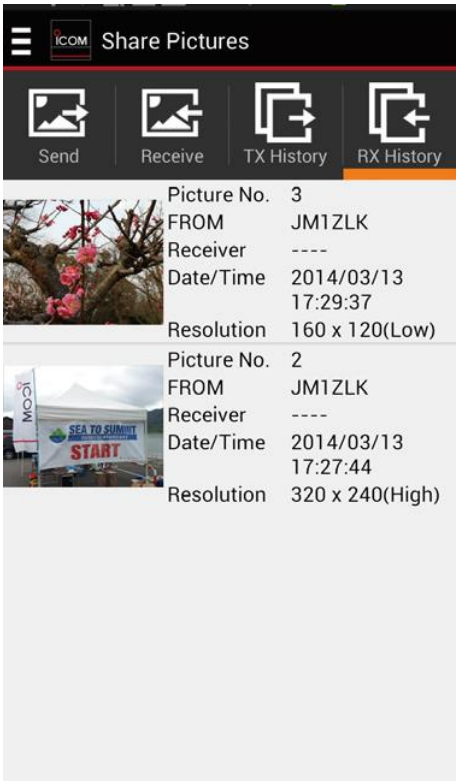
Obrazy są transmitowane w postaci niezależnych bloków cyfrowych, odpowiadających ich kwadratowym fragmentom, kolejno wyświetlanych w trakcie odbioru. Transmisja bloków nie jest kwitowana przez stację odbiorczą. W przypadku przekłamań w transmisji zakłócone bloki są wyświetlane w postaci czarnych pól. Każdy z bloków posiada wprawdzie własne dane korekcyjne, ale w przypadku błędnego odbioru nie są one powtarzane. Program nadawczy pozwala także na transmisję części obrazu i dzięki temu na inteligentne uzupełnianie brakujących bloków. Do wyboru są rozdzielczości 160 x 120, 320 x 240 i 640 x 480.

Najnowsze modele: IC-9700, IC-705 i ID-52 umożliwiają nawet transmisję i wyświetlanie obrazów bez korzystania z komputera albo telefonu.

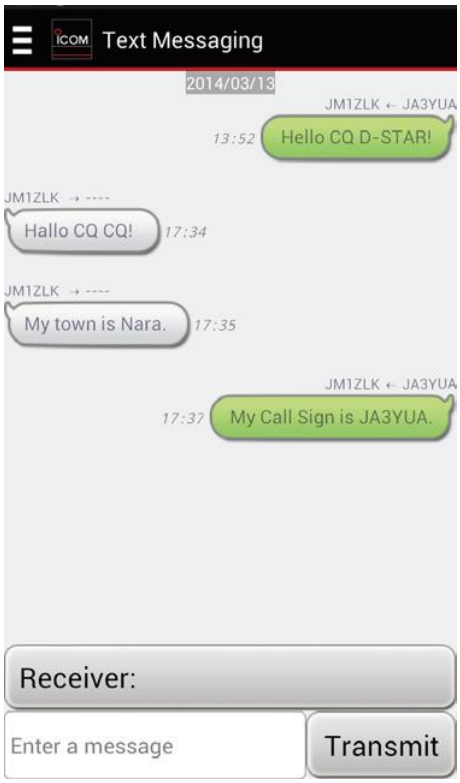
Najbardziej rozpowszechnionym formatem jest format 320 x 240 punktów, zapewniający wystarczająco dobrą jakość obrazu przy około dwuminutowym czasie jego transmisji ze standardową szybkością (pod względem rozdzielczości i czasu transmisji jest on zbliżony do analogowych norm SSTV Martin 1 i Scottie 1). Szybkość standardowa pozwala na korzystanie z dowolnych modeli mikroprzemienników, a czas transmisji nie przekracza najczęściej stosowanych na przemiennikach sieci ograniczeń czasu nadawania. Przy rozdzielczości 640 x 480 punktów i wysokiej jakości obrazów ich transmisja trwa w przybliżeniu 6 minut przy standardowej szybkości przekazu. Przy dłuższym czasie transmisji rośnie jednak nie tylko prawdopodobieństwo przerwania wskutek ograniczeń czasu nadawania przemienników, ale i prawdopodobieństwo wystąpienia zakłóceń odbioru. Nowsze modele radiostacji pozwalają na skrócenie czasu transmisji dzięki szybkiej transmisji danych. W tym trybie cała przepustowość kanału 3600 bit/s jest przeznaczona dla danych obrazowych (w trybie standardowym jest ona podzielona między dane głosowe – 2400 bit/s – i obrazowe – 1200 bit/s). Stosowanie trybu szybkiej transmisji niesie jednak ze sobą dalsze wyzwania techniczne.

Stacja nadawcza musi być przełączona przez operatora na większą szybkość przed rozpoczęciem transmisji, natomiast stacja odbiorcza dostosowuje się automatycznie. Dotyczy to jedynie nowszych modeli takich jak ID-51 PLUS(2), ID-5100, ID-52, IC-9700 czy IC-705. Starsze pięćdziesiątki jedynki, IC-7100 itp. dysponują tylko standardową szybkością. O ile transmisja przez lokalny przemiennik zasadniczo przebiega bezproblemowo, o tyle w sieci sprawa nie jest już taka pewna i wymaga eksperymentowania. Przy dostatecznej sile sygnału bezpośrednie łączności simpleksowe nie powinny również przysparzać problemów.

Zdjęcia przeznaczone do nadania można wykonać aparatem wbudowanym do telefonu lub komputera i w miarę potrzeby można je poddać obróbce w programie graficznym, a następnie zapisać w galerii obrazów. Pozwala to na przygotowanie materiałów zawczasu. Obrazy odebrane można również zapisać w galerii i ewentualnie później przenieść na PC. Obrazy przeznaczone do nadania muszą mieć kształt poziomego prostokąta o stosunku boków 4:3, jedną z trzech wymienionych rozdzielczości i być zapisane w formacie jpg. Do obróbki obrazów ICOM opracował program ST-4001 w wersjach dla Androida i iOS. Pozwala on na dopasowanie wymarów obrazu do wymaganego formatu, dopisanie znaku wywoławczego itd. Niezależnie od rozdzielczości do transmisji można wybrać jakość niską, standardową lub wysoką, a przez to także czas transmisji. Obrazy odebrane są automatycznie zapisywane w galerii, przy czym jej pojemność jest ograniczona do 500 plików. Po przekroczeniu tej granicy nowe pliki zastępują najstarsze.
Przyjęty w transmisjach D-Starowych system raportów podano w tabeli 1.6.1. Jest to pierwsza propozycja oceny jakości obrazu przy transmisji cyfrowej i różni sie od innych odwrotnym kierunkiem skali: P1 oznacza najlepszą jakość, a nie najgorszą jak w telewizji amatorskiej. Raporty można podawać głosem albo dopisać do następnej wysyłanej ilustracji. Oprogramowanie IC-9700 i IC-705 nie pozwala wprawdzie na dopisywanie tekstów, ale wystarczy ich połączenie kablem OPC-2350LU z komputerem aby korzystać z pełni możliwości.
Spośród wielu modeli prywatnych mikroprzemienników (ang. *hotspot*) świetnie spełniających swoją rolę w transmisji głosu i danych ze standardową szybkością, tylko część pozwala na korzystanie z większej szybkości w obu kierunkach (patrz tabela 1.6.2). W większości przypadków krótkofalowcy korzystają jednak z szybkości standardowej, aby umożliwić udział jak najszerszemu gronu operatorów. Pozwala to też na równoległe przekazywanie głosem dodatkowych informacji o nadawanym obrazie.
W oknie odbioru wyświetlane są informacje o odebranym obrazie, a znajdujący się w nim przycisk „Save" służy do jego zapisu w katalogu „Pictures".
Okno historii transmisji wyświetla katalog uprzednio nadawanych obrazów. Ich maksymalna liczba jest ograniczona do 500. Naciśnięcie w nim obrazu przez czas przekraczający 1 sekundę powoduje skasowanie go. W oknie historii odbioru wyświeltany jest katalog obrazów odebranych również w maksymalnej liczbie 500. Po odebraniu 501 obrazu najstarszy zostaje skasowany. Przyciśnięcie wybranego obrazu przez ponad sekundę powoduje jego skasowanie.



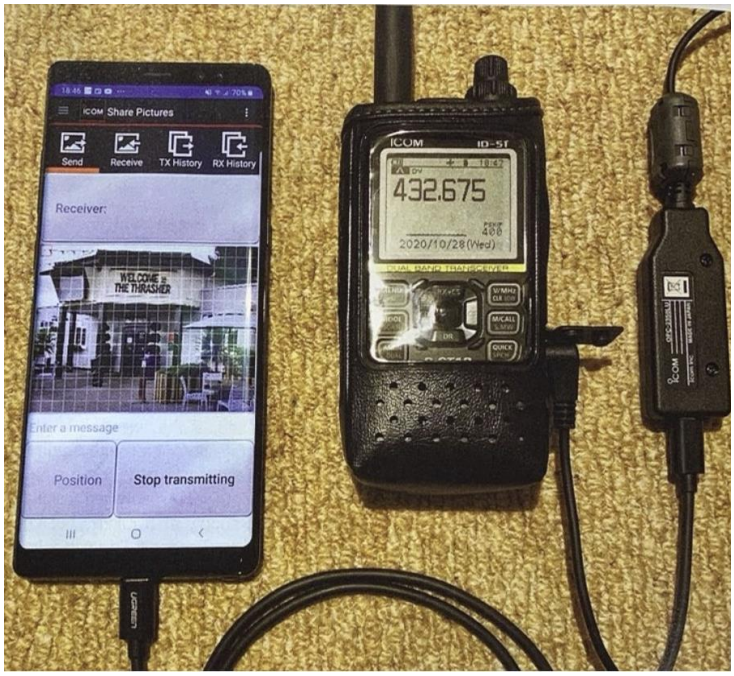
Rys. 1.6.6. Okno historii odbioru



Rys. 1.6.7. Okno transmisji wiadomości tekstowych

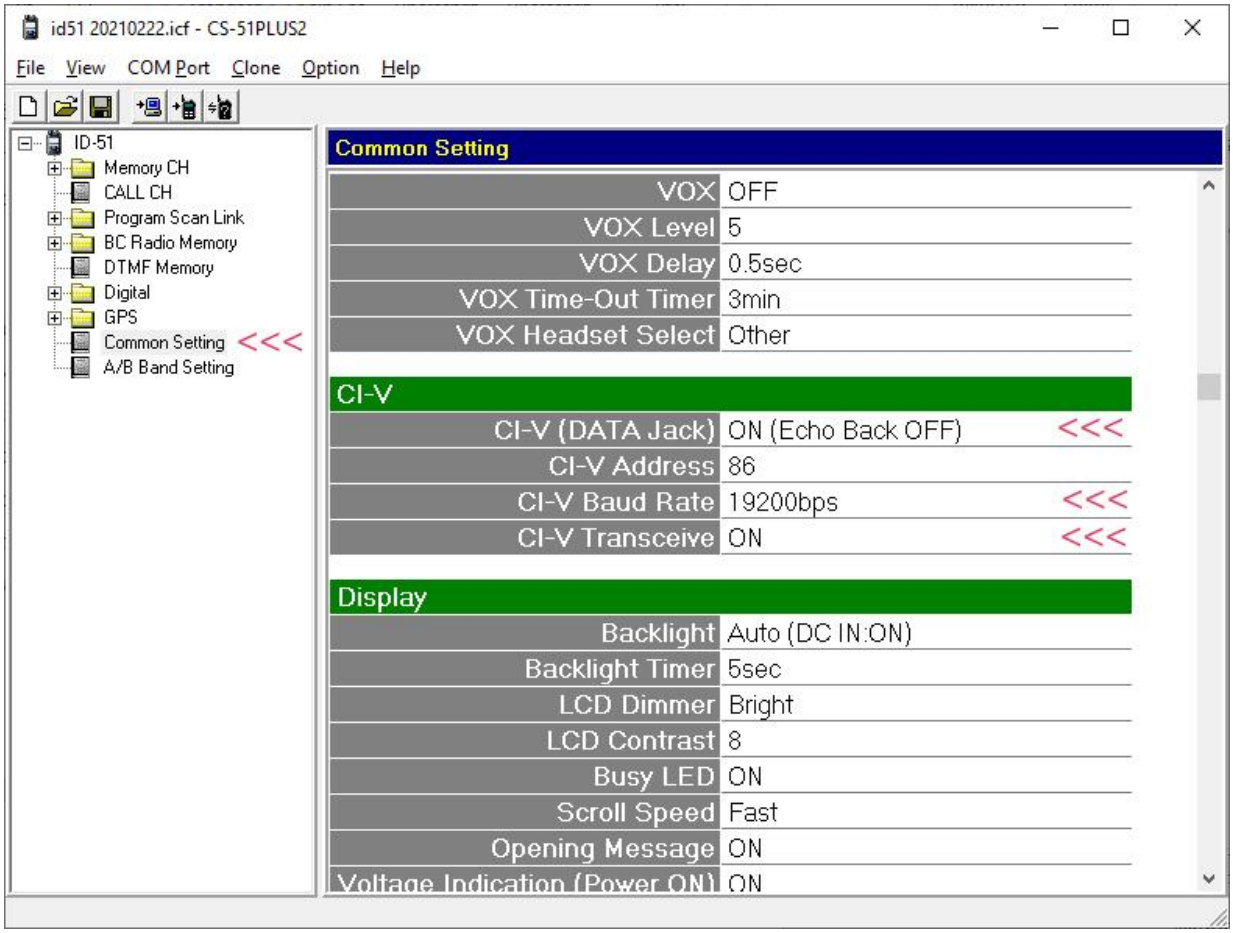
3. U dołu okna transmisji tekstów RS-MS1A znajduje się pole dla wprowadzania wiadomości przeznaczonych do nadania. Dla ich nadania należy nacisnąć przycisk „Transmit”. Nadany tekst jest wyświetlany w górnej części okna po prawej stronie, a teksty odebrane – po lewej (podobnie jak w „Skypie”). W polu „Receiver” można podać znak adresata.
4. W oknie map (rys. 1.6.8) wyświetlane jest położenie stacji przemiennikowych D-Starowych albo FM, albo położenie stacji korespondenta, o ile stacje te nadają dane pozycyjne (współrzędne). Po naciśnięciu symbolu stacji na mapie wyświetlane są dodatkowe informacje o niej. Naciśnięcie z kolei tej informacji powoduje przejęcie znaku stacji do pól [FROM] lub [TO] dla funkcji DR. Po wczytaniu nowego spisu przemienników należy w menu nacisnąć pozycję „Repeater station OFF” a następnie „Repeater station ON” w celu zaktualizowania położenia stacji na mapie.
5. W oknie historii odbioru wyświetlany jest spis odebranych wiadomości o maksymalnej pojemności 10000 wpisów. Po przekroczeniu tej granicy kasowany jest plik najstarszy.
6. W polu znaków korespondentów [Your call signal] można podać do 500 znaków z informacjami dodatkowymi. Do wprowadzenia nowego znaku służy ekranowy przycisk z plusem.
7. Pozycja *Import* pozwala na wczytanie do programu spisu przemienników lub adresów docelowych w formacie *csv*. Spis przemienników można wczytać z internetu, a spis korespondentów z modułu pamięciowego SD radiostacji.
8. Pozycja *Export* pozwala na zapisanie w katalogu *Export* komputera spisu przemienników, korespondentów i historii odbioru w plikach csv.

Kabla OPC-2350LU można użyć do połączenia z komputerem androidowym następujących modeli radiostacji: ID-31E PLUS, ID-51E PLUS, PLUS 2, IC-9700, IC-7100, ID-51E, a ID-4100E, ID-5100E i IC-705 dysponują łączami Bluetooth (konieczne jest zainstalowanie odpowiednio modułów UT-137 lub UT-133). IC-705 wymaga kabla typu OPC-2417 albo OPC-2418 jeżeli nie jest wykorzystywane złącze Bluetooth. W odróżnieniu od pozostałych modeli ID-31E, ID-51E i IC-7100 jako starsze nie dysponują zwiększoną szybkością transmisji danych („DV Fast data”), funkcją DR ani możliwością konfiguracji przez komputer. Radiostacja TH-D74 Kenwooda współpracuje z programem przez złącze Bluetooth, ale konieczne jest sparowanie jej najpierw z komputerem, a dopiero potem można wywołać program. Do przygotowania zdjęć do transmisji można posłużyć się programem ST-4001A. Pozwala on także na przesyłanie obrazów przez WiFi bezpośrednio do radiostacji typów ID-52, IC-705 (także przez złącze *Bluetooth*) i IC-9700 (wymaga oprogramowania wewnętrznego w wersji 1.2 lub nowszej) w celu ich nadania. ST-4001A pracuje pod Androidem 5.0 i nowszymi wersjami.



Rys. 1.6.8. Okno mapy Fot. 1.6.9. Przenośny zestaw do transmisji obrazów

W trybie szybkiej transmisji danych pełna przepustowość kanału 3600 bit/s jest przeznaczona dla danych, natomiast w trybie standardowym jest ona podzielona w stosunku 2400 bit/s do 1200 bit/s między cyfrowy sygnał głosu i dane.
W zależności od wersji oprogramowania wewnętrznego niektórych modeli radiostacji konieczne może być używanie nowszych lub nieco starszych wersji RS-MS1A. W niektórych wersjach dodatkowo do wymienionych powyżej punktów dodano możliwość wyświetlania pozycji stacji na mapach zapamiętanych lokalnie bez konieczności łączenia się w tym celu z Internetem („Offline map") i możliwość wyświetlania odczytanego z radiostacji spisu przemienników („Repeater list").
Korzystanie z programu RS-MS1A i połączenia komputera z radiostacją wymaga dostosowania niektórych ustawień radiostacji. Dla ID-51E PLUS2 są to:
- w punkcie „CI-V (Data Jack)" ustawienie „ON (Echo Back OFF)",
– w punkcie „C-IV Transceive" ustawienie „ON",
– w punkcie „C-IV Baud Rate" szybkość 19200 bit/s.
W przypadku niewłaściwych ustawień w radiostacji program melduje brak połączenia albo brak kabla USB. Oznaczenia parametrów dla innych modeli radiostacji Icoma są albo identyczne albo bardzo zbliżone. Szczegóły są podane w ich instrukcjach obsługi (ewentualnie w instrukcjach rozszerzonych – „Advanced" – a nie w podstawowych). Zmiany ustawień można dokonać albo w menu radiostacji albo w wygodniejszy sposób w jej programie konfiguracyjnym CS (rys. 1.6.10).
Wersja programu dla systemu iOS pozwala jedynie na komunikację przez złącze *Bluetooth* z radiostacją ID-4100E.



Rys. 1.6.10. Ustawienia niezbędne dla korzystania z RS-MS1A na przykładzie radiostacji ID-51E PLUS2 w programie konfiguracyjnym CS-51PLUS2

Tabela 1.6.1.
System raportów dla transmisji obrazów w systemie D-STAR

Raport	Znaczenie
P1	Obraz idealny
P1 minus	Brakuje 5% bloków lub mniej
P2	Brakuje do 25% treści obrazu
P3	Brakuje do 50% treści obrazu
P4	Brakuje do 75% treści obrazu
P5	Obraz nieużyteczny lub ponad 75% straconej treści

Tabela 1.6.2.
Transmisja obrazów przez niektóre mikroprzemienniki

Typ	Szybka transmisja	Uwagi
DV4mini na Pi-3B z oprogramowaniem DARC A23 MMDVM	Niemożliwa	Nie przewidziano szybkiej transmisji obrazów, najwidoczniej jest to słaba strona MMDVM; może odbić się też na innych modelach
DVMega + BlueStack, połączenie przez złącze USB z klientem BlueDV	Tak, w wersjach beta dla Androida i Windows	Odbiór funkcjonuje bardzo dobrze. Od niedawna istnieje nowa wersja BlueDV dla obu systemów z szybką transmisją; w wersji androidowej konieczne połączenie przez USB; DVMega bez BlueDV korzysta z serwera MMDVM, co oznacza ograniczenia j.w.
SharkRF openSPOT 1	Niemożliwa	Szybka transmisja nie przewidziana, poza tym praca stabilna
SharkRF openSPOT2	Tak	Nie zauważono problemów
SharkRF openSPOT 3	Tak	Możliwy odbiór i nadawanie, lepsze wyniki dla oprogramowania w wersji v37 i przy silnym sygnale WiFi
MMDVM własnej konstrukcji + „Star Hotspot”	Tylko odbiór	Zależnie od wykonania MMDVM, jego oprogramowania i modelu Pi. Zalecane korzystanie z najnowszych wersji oprogramowania „Pi-Star”
ZUMspot-Rpi MMDVM	Tylko odbiór	Korzysta z oprogramowania Pi-Star jak powyżej
Bezpośrednio przez przemienniki	Możliwa, ale nie gwarantowana	Zależy od wyposażenia przemienników i bramek na trasie transmisji, przemienniki Icoma w wersji G3 na pewno, dla innych możliwa

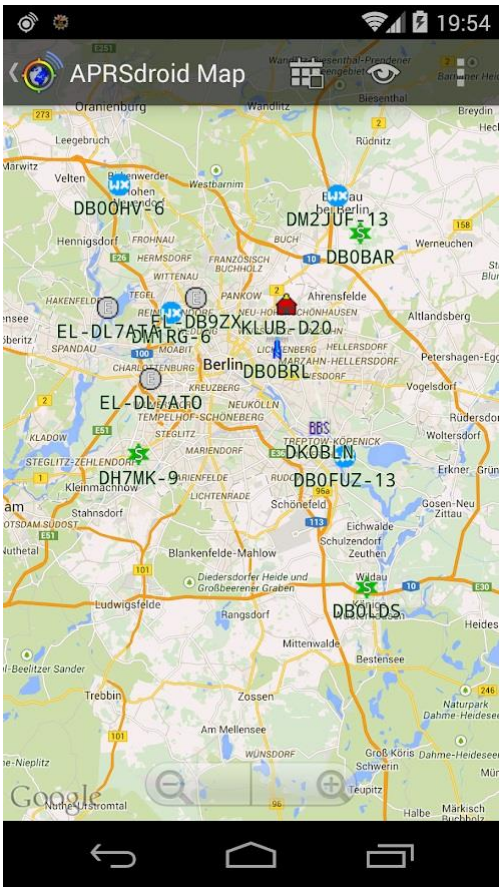
1.7. APRS

1.7.1. „APRSdroid”

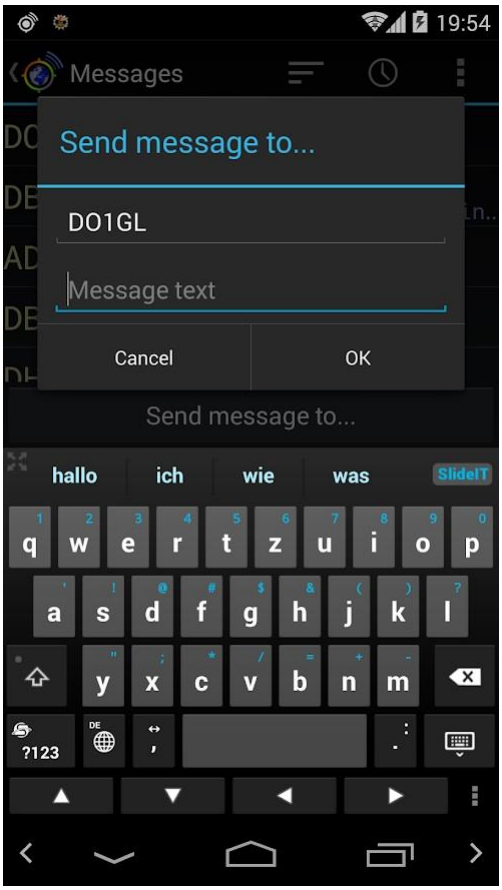
Dostępny w *Google Play* APRSdroid pozwala na nawiązanie połączenia z siecią APRS internetowo z APRS-IS lub radiowo. W tym drugim przypadku program generuje i odbiera sygnały dźwiękowe kluczowane AFSK. Połączenie z radiostacją odbywa się dźwiękowo przez głośniki i mikrofony obu urządzeń. Oprócz tego może on komunikować się z modemem TNC przez złącze *Bluetooth*. APRSdroid wymaga wersji 4.0 Androida lub nowszej.

Program nadaje komunikaty APRS z własną pozycją do sieci albo radiowo, odbiera komunikaty innych stacji (indywidualnych i przemiennikowych) i przedstawia ich położenie na mapie. Możliwe jest także nadawanie komunikatów tekstowych do wybranego korespondenta. Telefon lub komputer muszą być wyposażone w moduł odbiornika GPS. Podobne możliwości daje także *APRS Messenger*.

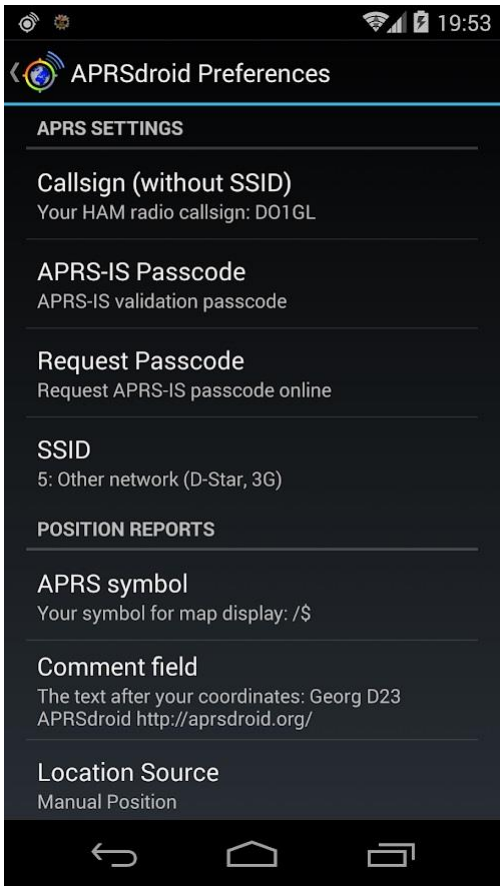
Rys. 1.7.1.1. Odbiór komunikatów APRS



Rys. 1.7.1.2. Pozycje statcji na mapie

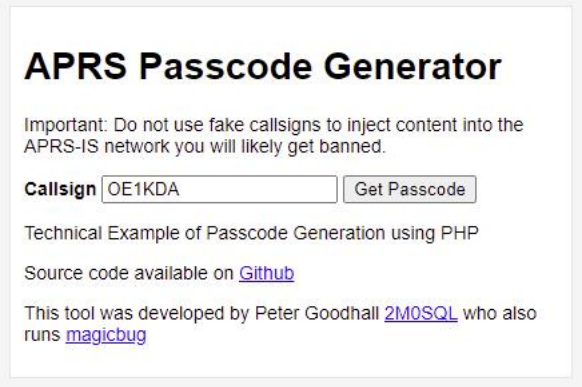


Rys. 1.7.1.3. Nadawanie komunikatów tekstowych



Rys. 1.7.1.4. Ustawienia

Połączenie z serwerem sieci APRS-IS (np. *rotate.aprs.net:1450*) wymaga uzyskania hasła dostępu. Strony, na których po podaniu własnego znaku wywoławczego generowane jest hasło można znaleźć w Internecie m.in. pod adresami *https://apps.magicbug.co.uk/passcode/, http://n5dux.com/ham/aprs-passcode/* i *https://www.george-smart.co.uk/aprs/aprs_callpass/.*



Rys. 1.7.1.5. Strona generatora haseł dostępu do APRS-IS z pierwszego adresu

1.7.2. Bramka internetowa „Igate2 Pro”

Program *Igate2 Pro* pozwala na uruchomienie odbiorczej bramki internetowej APRS. Do odbioru sygnałów APRS stacji amatorskich wymaga on podłączenia odbiornika programowalnego (SDR) RTL-SDR. Są to odbiorniki USB przeznaczone pierwotnie do odbioru telewizji, ale już od dawna dzięki niskiej cenie znalazły szerokie zastosowanie w krótkofalarstwie. Ze względu na znaczny pobór prądu przez odbiornik zaleca się korzystanie z zasilacza sieciowego. Do podłączenia odbiornika konieczny jest kabel OTG. Program wymaga dostępu do współrzędnych geograficznych (lokalizacji) komputera.

Informacje zawarte w odebranych pakietach APRS (są to nienumerowane pakiety packet-radio typu UI) są po ich zdemodulowaniu i zdekodowaniu przekazywane do sieci APRS-IS, dzięki czemu pozycje stacji są wyświetlane na mapach internetowych m.in. pod adresem *aprs.fi*. Wymaga to otrzymania hasła dostępu w sposób opisany w poprzednim punkcie.

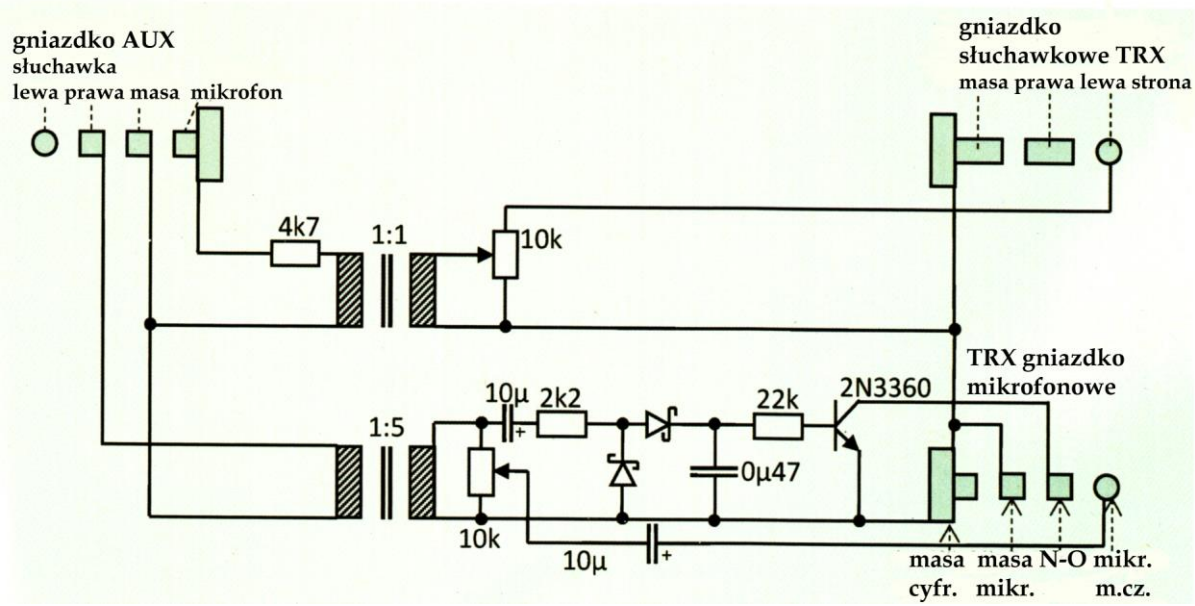
Program pozwala na podsłuch odbieranych sygnałów, co ułatwia dostrojenie odbiornika. Może on także pracować w tle równolegle do innych programów. W Europie podstawową częstotliwością pracy APRS jest 144,800 MHz, ale w okolicach o większym natężeniu ruchu stosowana jest dodatkowo częstotliwość 432,500 MHz, a czasami także inne.

Rys. 1.7.2.1. Odbierane komunikaty

1.8. Emisje cyfrowe

1.8.1. Modemy dla emisji cyfrowych

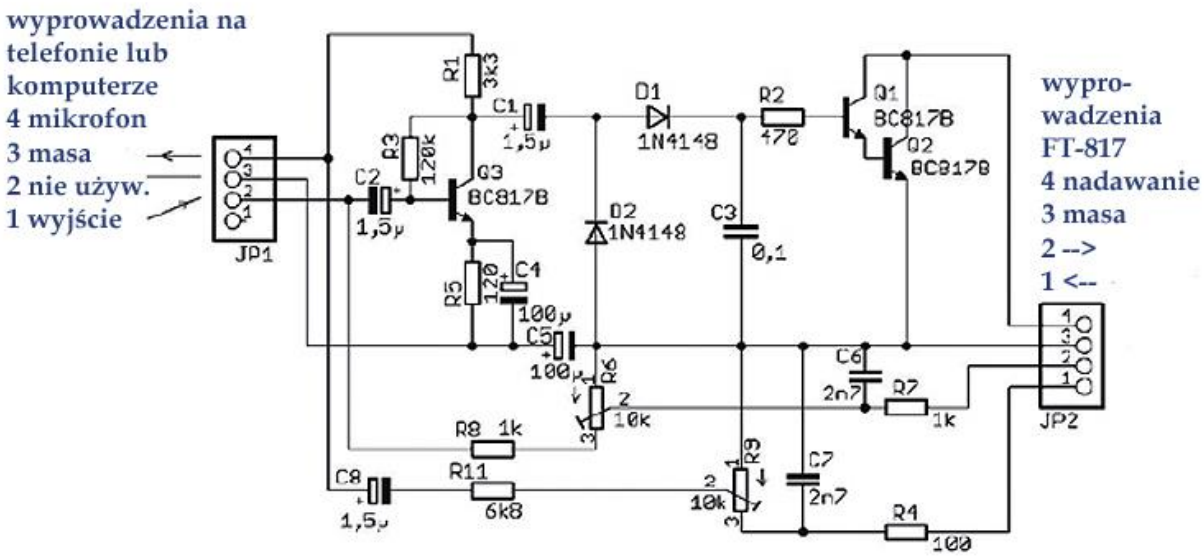
Również do pracy emisjami cyfrowymi poza domem wygodnie jest korzystać z przenośnego komputera androidowego lub telefonu. W wielu wypadkach wystarczy sprzężenie akustyczne między radiostacją i komputerem, ale wygodniej jest połączyć oba urządzenia za pomocą modemu (rys. 1.8.1.1). W odróżnieniu od łączności fonicznych praca emisjami cyfrowymi z wykorzystaniem modemu jest cicha i nie zakłóca spokoju sąsiadom w hotelu lub na kempingu ani nikomu z rodziny.
Układ jest stosunkowo prosty i łatwy do samodzielnego zbudowania. W układzie VOX-u zastosowano diody Schottkiego np. typu BAT65 ze względu na ich niskie napięcie progowe. Tranzystor NPN może być dowolnego typu, o współczynniku wzmocnienia 300–400 – przykładowo BC547B, BC337–40. Dla zapewnienia odpowiedniego napięcia sterującego tranzystor kluczujący przekładnia transformatora w torze nadawczym powinna wynosić co najmniej 1:2. Transformatory m.cz. są dowolnego typu. Konstrukcja układu nie jest krytyczna i można go zbudować na uniwersalnej płytce dziurkowanej.
W nowszych modelach telefonów i komputerów wyprowadzenia w gniazdku AUX są zgodne z normą CTIA – jak to przedstawiono na schemacie. W urządzeniach starszych typów należy odpowiednio skorygować połączenia. Do połączenia układu z komputerem najwygodniej jest zopatrzyć się w gotowy kabel (ekranowany) z wtyczką 4-kontaktową i obciąć jeden z jego końców ponieważ przylutowanie przewodów do wtyczki może być trudne ze względu na jej miniaturowe wymiary.



Rys. 1.8.1.1. Schemat ideowy modemu konstrukcji DH5AK. Do prowadzenia łączności za pomocą programów DroidRTTY i DroidPSK DH5AK korzystał z komputera przenośnego z ekranem 7-calowym i klawiaturą Blutooth. (źródło CQDL 3/2016)

Modem z rys. 1.8.1.2 jest przeznaczony do współpracy telefonów i komputerów z *Androidem* z ulubionymi przez wiele osób radiostacjami FT-817 i FT-818. Umożliwia on pracę emisjami PSK31, RTTY, SSTV, CW i innymi w terenie lub też oczywiście i w domu. Rozwiązanie jest udoskonaloną wersją urządzenia opisanego pod adresem *www.wolphi.com/interface* i zostało opublikowane w nrze 3/2016 miesięcznika *QST*.
Przy odbiorze emisji cyfrowych sygnał m.cz z radiostacji jest podawany przez kontakt JP2/1 i potencjometr montażowy R9 na kontakt JP1/4 – wejście mikrofonowe telefonu lub komputera tabliczkowego. Programy przeznaczone do pracy emisjami cyfrowymi można znaleźć pod adresem *www.wolphi.com*.
W trakcie nadawania sygnał m.cz. z telefonu lub komputera jest podawany przez kontakt JP1/2, potencjometr R6 i kontakt JP2/2 na wejście mikrofonowe lub wejście danych radiostacji. Sygnał ten po

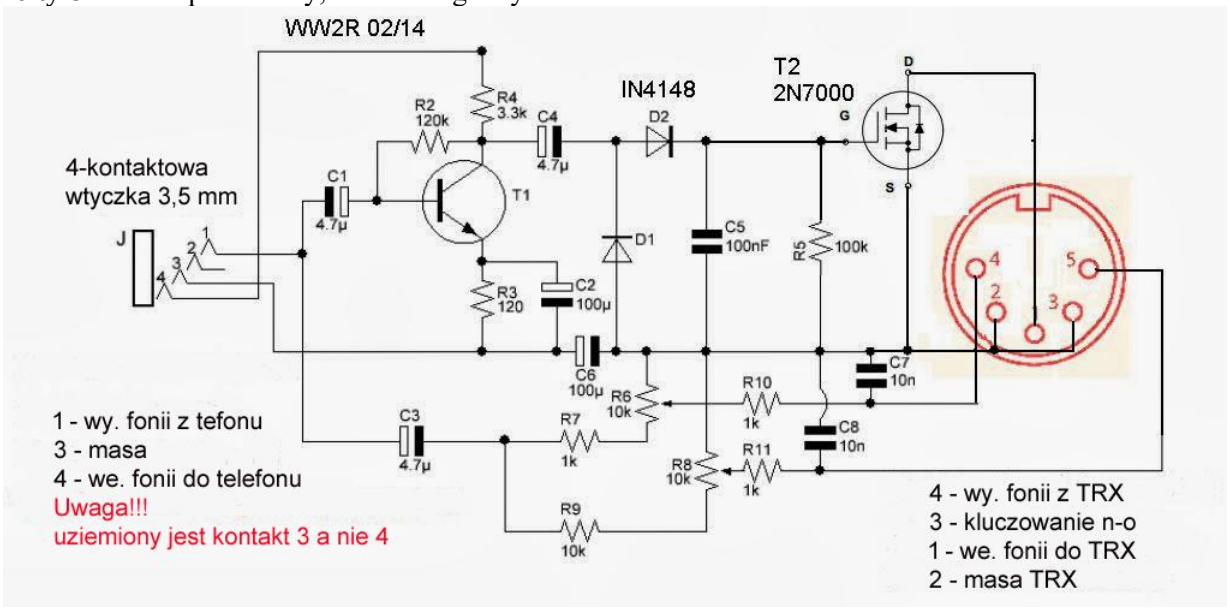
wzmocnieniu w tranzystorze Q3 i wyprostowaniu za pomocą diod D1 i D2 kluczuje nadajnik za pośrednictwem tranzystorów Q1 i Q2.



Rys. 1.8.1.2. Schemat ideowy modemu z QST

Modem *Wolphi-Link* z ilustracji 1.8.1.3 został opracowany przez firmę *Wolphi LLC* (*www.wolphi.com*) autora programów terminalowych *DroidPSK, DroidRTTY, DroidSSTV* i innych dla amatorskich emisji cyfrowych.

Modem nadawczo-odbiorczy jest wyposażony w potencjometry służące do regulacji poziomu sygnałów oraz układ automatycznego kluczowania nadajnika – VOX. Do połączenia z komputerem lub telefonem służy czterokontaktowa wtyczka koncentryczna o średnicy 3,5 mm. *Wolphi-Link* jest zasilany przez telefon lub komputer napięciem 1,8 V (lub wyższym). Instrukcja obsługi informuje również, że minimalne dopuszczalne napięcie zasilania wynosi 1,4 V. Do współpracy z radiostacją zastosowano natomiast 6-kontaktowe gniazdko mini-DIN. Oprócz modemu producent oferował kable z wtyczką pasującą do gniazd danych radiostacji FT-817, FT-818, FT-857, FT-897, IC-703, IC-706 i niektórych innych. Dla innych typów radiostacji można kupić jeden z gotowych kabli i po obcięciu wtyczki założyć pasującą do posiadanego sprzętu, albo wykonać kabel samodzielnie. Schematy wyprowadzeń zawierają instrukcje obsługi radiostacji. Obecnie modem nie jest już wprawdzie produkowany, ale jego układ jest na tyle nieskomplokowany, że można go wykonać samodzielnie.



Rys. 1.8.1.3. Schemat ideowy modemu *Wolphi-Link*

1.8.2. Programy nadawcze i nadawczo-odbiorcze

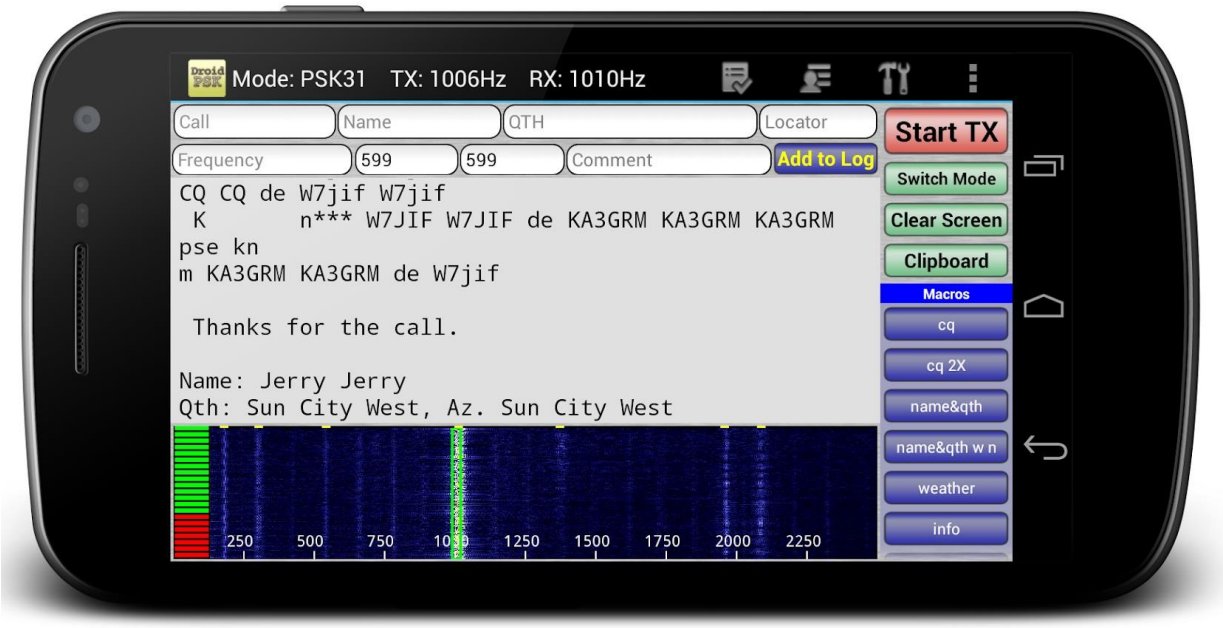
Przedstawione w dalszym ciągu programy są dostępne w sklepie internetowym częściowo odpłatnie, ale po przystępnych cenach, a w części bezpłatnie. Ich obsługa jest w większości przypadków na tyle prosta i intuicyjna, że nie wymaga szczegółowego omówienia. Dotyczy to zwłaszcza czytelników znających przedstawiane tu rodzaje emisji i ewentualnie mających także pewną praktykę w korzystaniu z podobnych programów na innych płaszczyznach sprzętowych.

Program *DroidRTTY* firmy *Wolphi LLC* jest przeznaczony do prowadzenia łączności dalekopisowych z szybkością transmisji 45,45 bodów i odstępem częstotliwości 170 Hz przy wykorzystaniu jednego z opisanych modemów albo przez sprzężenie akustyczne komputera (telefonu) z radiostacją. W jego oknie głównym oprócz nadawanych i odbieranych tekstów jest wyświetlany wskaźnik wodospadowy pokrywający zakres 100 – 2500 Hz. Nadawane teksty można pisać bezpośrednio na klawiaturze ekranowej albo korzystać z 20 przygotowanych uprzednio. Teksty te noszą gwarowo nazwę „makro" – od makrorozkazu – i zawierają parametry o wartościach przypisywanych na bieżąco, j.np. znak korespondenta, raport itp. Program prowadzi dziennik łączności, który można następnie przenieść na PC w formacie ADIF lub przesłać na serwery internetowe. DroidRTTY może także dekodować transmisje stacji profesjonalnych z szybkościami 50 bodów i odstępem częstotliwości 450 Hz.

Rys. 1.8.2.1. DroidRTTY

DroidPSK tej samej firmy służy do prowadzenia łączności emisjami PSK31 i PSK63 z wykorzystaniem modemu lub sprzężenia akustycznego z radiostacją. Ta ostatnia emisja jest spotykana również w zawodach. Charakteryzuje się wprawdzie niższą czułością niż PSK31, ale za to szybkość transmisji jest dwukrotnie wyższa.
Analogicznie jak poprzedni prowadzi on dziennik łączności (ang. *log*) i korzysta z 20 uprzednio przygotowanych tekstów, ale użytkownik może także pisać dowolne teksty na klawiaturze dotykowej. Archiwum instalacyjne programu zawiera zestaw przykładowych tekstów podręcznych, które można zmodyfikować dostosowując do własnych potrzeb. Każdemu z tekstów można nadać nazwę o długości do 15 znaków alfanumerycznych. Teksty mogą zawierać dowolną treść wpisaną na klawiaturze oraz informacje pochodzące z okna danych stacji „Station info", takie jak znak wywoławczy, imię operatora, lokalizację, opis wyposażenia itp. Do dyspozycji są także informacje wyświetlane na ekranie w trakcie łączności, takie jak raporty i dane korespondenta. Przyciskom służącym do wywołania podręcznych tekstów można nadać jeden z czterech kolorów, co pozwala na ich pogrupowanie pod względem od znaczenia.

Przed rozpoczęciem pracy w eterze należy upewnić się, że wszystkie sygnały dźwiękowe i alarmy systemu operacyjnego zostały wyłączone, gdyż w przeciwnym przypadku będą one niepotrzebnie nadawane.
Zakończone łączności można wpisać do dziennika stacji. Program pobiera datę i czas sytemowy z komputera (czas UTC lub lokalny w zależności od ustawień) i dodaje pozostałe potrzebne dane: częstotliwość w MHz, znak, raporty, QTH z pól znajdujących się w głównym oknie. Do zapisania w dzienniku służy przycisk ekranowy „Add to log". Zawartość dziennika daje się wyeksportować w formacie ADIF przydatnym do wpisania ich do internetowych dzienników takich jak LoTW albo do innych progamów przeznaczonych do prowadzenia dzienników. Plik ADIF jest zapisywany w module pamięci SD.
Obsługa progarmu jest wprawdzie prosta, ale w przypadku wątpliwości można skorzystać z instrukcji dostępnej w witrynie *Wolphi LLC* (*www.wolphi.com*).

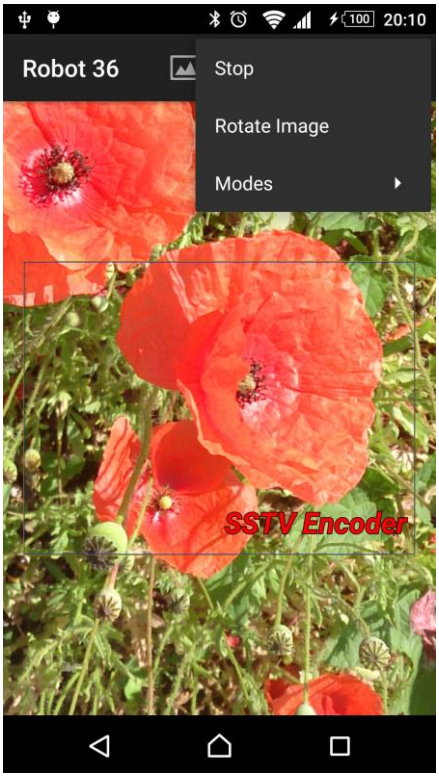


Rys. 1.8.2.2. DroidPSK

DroidSSTV firmy *Wolphi LLC* służy do prowadzenia łączności przez telewizję amatorską z wolną analizą obrazu, SSTV, w sposób identyczny jak poprzednie dwa dla łączności tekstowych. W obecnej wersji programu pozwala on korzystać z norm Martin 1, Martin 2 (używanych przez stacje europejskie) oraz Scottie 1, Scottie 2 i Scottie DX (stosowanych głównie przez stacje amerykańskie). Dostrojenie się do korespondenta ułatwia analizator widma, a automatyczna korekta pochylenia obrazu eliminuje różnice w częstotliwościach zegarowych komputerów obu korespondentów. Program automatycznie rozpoznaje normę, m.in. w oparciu o nagłówek VIS. Odebrane obrazy można automatycznie zapisywać w module pamięciowym SD. Tam też mogą być zapisane obrazy przygotowane do nadania w mksymalnej liczbie dziewięciu. Użytkownicy mogą dodawać teksty do obrazów, przykładowo znak wywoławczy lub wywołanie CQ a także teksty uprzednio przygotowane (standardowe).

Rys. 1.8.2.3. DroidSSTV

Bezpłatny program *SSTV-Kodierer* dla Androida także w wersji 10 dostępny bezpłatnie pod adresem *https://apkpure.com/de/sstv-encoder/om.sstvencoder* pozwala na pracę nie tylko najważniejszymi wariantami norm Martin i Scottie ale także takimi znacznie mniej popularnymi jak *Robot 36 Color*, *Robot 72 Color*, *Wraase SC2 180*, PD50, PD90, PD120, PD160, PD180, PD240 i PD 290.
Operator może korzystać ze zdjęć zrobionych na bieżąco albo ze zdjęć zapisanych już w pamięci. Po naciśnięciu na powierzchnię obrazu wywoływany jest edytor tekstów pozwalający na pisemne uzupełnienie obrazu przed jego nadaniem. Program dekoduje i wyświetla również odebrane obrazy.



Rys. 1.8.2.4. SSTV-Kodierer

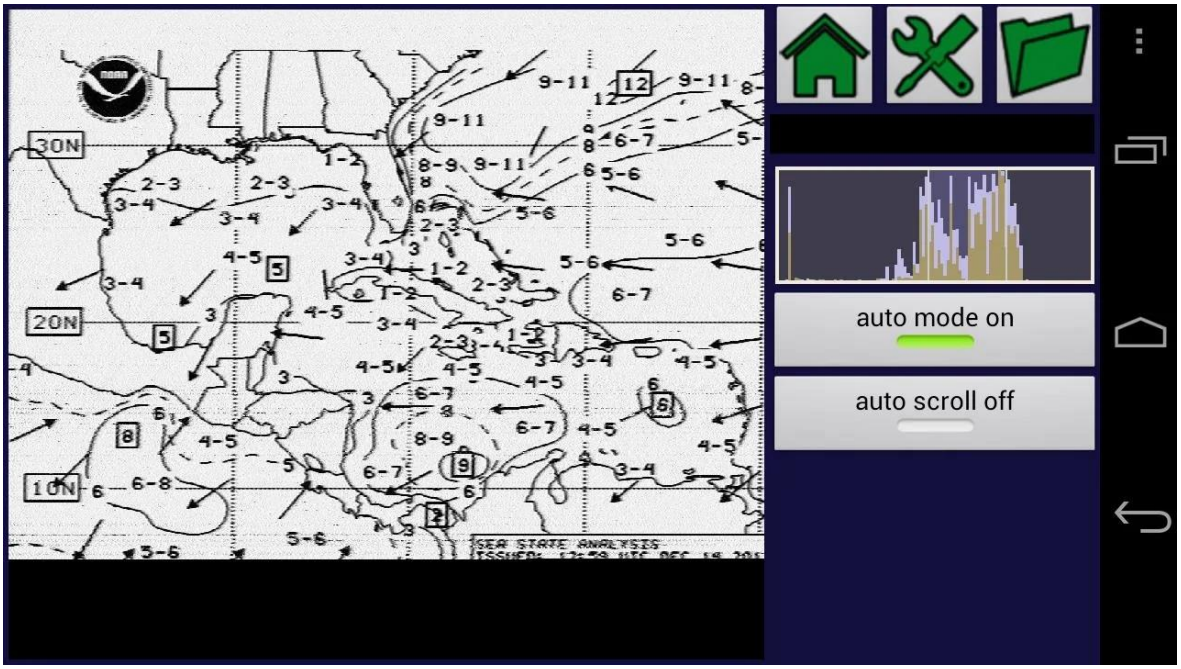
Niestety jak dotąd nie ma wersji WSJT-X dla Androida, ale bardziej zaawansowani w sprawach komputerowych i znający Linuks czytelnicy mogą zainstalować Linuks i pod nim wersję Linuksową programu. W sklepie internetowym *Google Play* dostępny jest wprawdzie program *FT Monitor*, ale pozwala on jedynie na zdalny podgląd przez lokalną sieć WiFi stacji odbieranych na komputerze Windowsowym i nie umożliwia zdalnego nadawania. Jest to więc rozwiązanie mało atrakcyjne.
Instalacja linuksowych wersji WSJT-X i *Fldigi* na komputerze androidowym wymaga od operatora uprawnień administratora (*root*) i może grozić w skrajnym przypadku zawieszeniem się sytemu operacyjnego. Czytelnicy o mniejszym doświadczeniu z systemami Android i Linuks powinni więc zrezygnować z tej instalacji. Do pracy programów konieczny jest dodatkowy podsystem dźwiękowy USB, rozgałęźnik USB OTG, kable oraz modem do połączenia z radiostacją.
Konieczne jest pobranie i zainstalowanie kolejno programów *BusyBox*, *Linux Deploy* (umożliwiającego instalację Linuksa) i *VNC Viewer*. *VNC Viewer* (*RealVNC*) umożliwia dostęp do Linuksa z poziomu Androida z wykorzystaniem pętli wewnętrznej (*localhost*) 127.0.0.1::5900.
Dopiero potem możliwe jest zainstalowanie wersji *WSJT-X for ARM* i *Fldigi for Arm* pod Linuksem. *Fldigi* jest uniwersalnym programem terminalowym umożliwiającym łączności emisjami PSK31, PSK63, RTTY, dalekopisowe w systemie Hella, Olivią, Contestią, MT63, MFSK i wieloma innymi, a WSJT-X emisją FT8 i innymi z rodziny WSJT. Szczegółowy opis instalacji podał VA3OSO w filmie dostępnym pod adresem:
https://www.youtube.com/watch?v=_AK4JnFeIY0.

W witrynie *apkpure.com* dostępny jest program *Hellschreiber RX/TX* – nadawczo odbiorczy program dla dalekopisów systemu Hella. Komputer może być sprzężony z radiostacją akustycznie lub przez modem.

Program *QRSS Beacon* dostępny odpłatnie w *Google Play* służy do nadawania wprowadzonego przez użytkownika tekstu radiolatarni wolną telegrafią QRSS z kluczowaniem QRSS (amplitudy) FSKCW i DFCW (częstotliwości). W drugim przypadku sygnał telegraficzny jest kluczowany z dewiacją kilku Hz, a w trzecim wszystkie elementy mają jednakową długość, z tym, że kresce odpowiada wyższa częstotliwość. Długość kropki jest wybierana w zakresie 1 60 sekund. Częstotliwość kluczowanej podnośnej akustycznej leży między 450 i 550 Hz. Do połączenia radiostacji z komputerem najlepiej jest użyć modemu i włączyć w niej automatyczne kluczowanie (VOX).

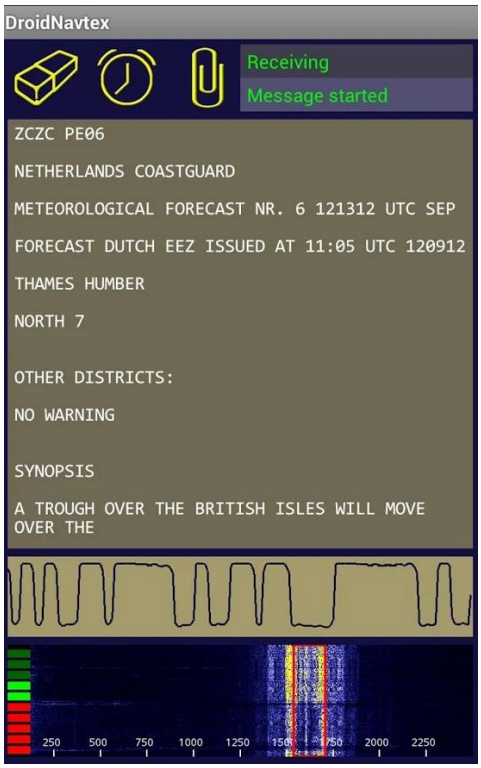
1.8.3. Programy odbiorcze

HF Weather Fax firmy *Wolphi LLC* służy do odbioru map pogody nadawanych faksymile w normie 120/576 (120 linii/min., współczynnik – indeks – kształtu 576) na falach krótkich i map nadawanych przez satelity meteorologiczne w paśmie 137 MHz. Ich odbiór może być szczególnie interesujący dla żeglugi (zwłaszcza amatorskiej bo w innych przypadkach jednostki pływające muszą być wyposażone w sprzęt profesjonalny). Program jest wyposażony w analizator widma ułatwiający dostrojenie odbiornika do sygnału stacji nadawczej. Rozpoznaje on też automatycznie tony APT i jest wyposażony w korekcję pochylenia obrazu. Odebrane mapy można automatycznie rejestrować w pamięci SD.
W najprostszym przypadku do pracy wystarczy akustyczne sprzężenie komputera z odbiornikiem.



Rys. 1.8.3.1. Mapa pogody odebrana na falach krótkich

DroidNavtex firmy *Wolphi LLC* dekoduje i wyświetla na ekranie komunikaty meteorologiczne i ostrzeżenia dla żeglugi nadawane w systemie Navtex. Do odbioru radiowego komunikatów nadawanych emisją SITOR-B (spokrewniona z nią jest stosowana dawniej przez krótkofalowców emisja AMTOR-B) służy odbiornik SSB dostrojony do częstotliwości 518 lub 490 kHz. Program rejestruje automatycznie odebrane komunikaty, co pozwala na ich późniejsze przeglądanie, przekazywanie dalej i kasowanie dopiero kiedy nie będą już potrzebne. Możliwy jest też odbiór komunikatów o zadanych godzinach. Spośród wielu typów komunikatów najbardziej interesujące dla żeglarzy-amatorów mogą być komunikaty typu A (ostrzeżenia nawigacyjne), B (ostrzeżenia meteorologiczne), D (komunikaty ratunkowe) i E (prognozy pogody).

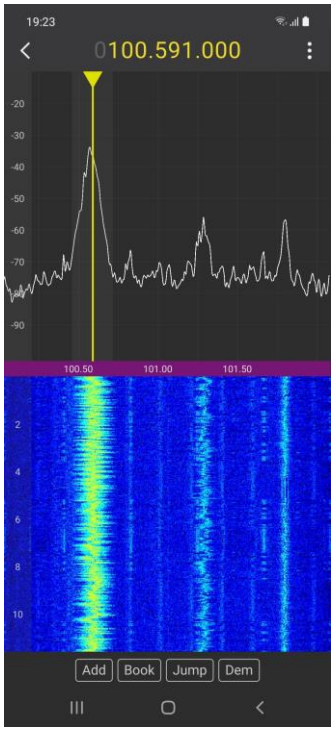


Rys. 1.8.3.2. DroidNavtex

1.9 Odbiorniki programowalne

1.9.1. Odbiorniki lokalne

Program *MagicSDR* dostępny w witrynie *apkpure.com* współpracuje z odbiornikami programowalnymi RTL-SDR i podobnymi, które mogą być udostępniane w sieci domowej przez serwer *rtl_tcp*. Program wyświetla szerokie widmo odbieranych sygnałów i detekuje emisje AM, SSB, CW, NFM i WFM. Odbiornik może być także podłączony do złącza USB komputera androidowego za pomocą kabla OTG.
Zakresy odbioru zależą od typu używanego odbiornika i jego wyposażenia dodatkowego takiego jak konwertery częstotliwości.

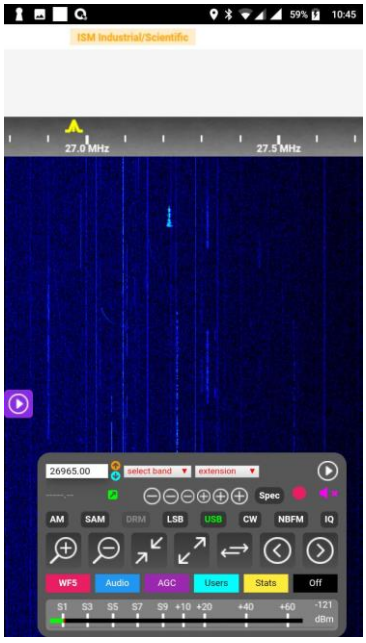


Rys. 1.9.1.1. Wskaźniki wodospadowy i widma

Podobne możliwości dają także programy *QuestaSDR* i *SDR Touch*. W wersji płatnej ten ostatni może dodatkowo nagrywać programy radiowe, wyświetlać dane RDS i posiada analizator widma o szerokości pasma 1 MHz. SDR Touch obsługuje też odbiorniki z serii DX-Patrol wyposażone w konwerter pokrywający fale krótkie.
Program *RF-Analyzer* współpracuje z odbiornikami RTL-SDR i SDR-Play tworząc w ten sposób przenośny analizator widma. Informacje o tych i innych programach można znaleźć m.in. w witrynie *RTL-SDR.com*.

1.9.2. Odbiorniki internetowe

KiwiSDR z witryny *apkpure* umożliwia korzystanie z dostępnych w internecie odbiorników KiwiSDR.. Są to szerokopasmowe odbiorniki programowalne pokrywające zakres fal długich, średnich i krótkich do 30 MHz. Obecnie w Internecie dostępna jest znaczna liczba takich odbiorników zlokalizowanych w wielu krajach i częściach świata. Pozwalają one przykładowo na odbiór stacji radiofonicznych z dalekich regionów świata, sprawdzenie siły i jakości sygnału własnej stacji amatorskiej, porównawcze badania anten, obserwacje radiometeorologiczne, warunków propagacji fal itd.
Z odbiorników *KiwiSDR* można korzystać także za pomocą zwykłej przeglądarki internetowej bez pomocy programu *KiwiSDR*, ale w programie można wybierać wygodnie odbiorniki dzięki przedstawieniu ich lokazacji na mapie.
Możliwe jest także zainstalowanie własnego odbiornika *KiwiSDR* w miejscu o dogodnych warunkach do odbioru i korzystanie z niego przez sieć lokalną w innych pomieszczeniach (patrz rozdział 6). Podobne możliwości daje także program *Android SDR*.



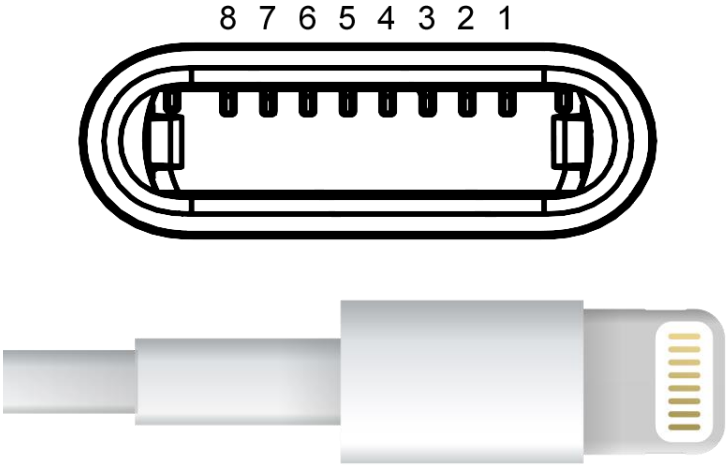
Rys. 1.9.2.1. Typowe okno odbiorcze

2. Komputery z iOS

2.1. System operacyjny iOS

IOS jest systemem operacyjnym komputerów tabliczkowych i telefonów firmy Apple. Podobnie jak w przypadku Androida korzystanie z niego za pomocą ekranu dotykowego i znajdujących się na nim symboli jest stosounkowo łatwe i intuicyjne. Komputery są fabrycznie wyposażone w szereg programów przydatnych dla szerszych rzesz użytkowników. Dodatkowe programy, w tym programy interesujące krótkofalowców są dostępne w internetowym sklepie *App Store*. Polityka sklepu jest bardziej restryksyjna, również od strony finansowej, niż w przypadku sklepu androidowego, dlatego też wybór programów krótkofalarskich jest mniejszy. Podobnie jak w sklepie androidowym znaczna ich część jest bezpłatna. Oprócz *App Store* istnieje również drugi sklep *iTunes Store* oferujący nagrania muzyczne i filmowe, oraz trzeci *Apple Book Store*, ale nie ma to zasadniczo związku z krótkofalarstwem.

System iOS począwszy od wersji 11 jest standardowo wyposażony w administrator plików mający na ekranie symbol folderu (niebieskiej teczki) i dlatego nie potrzeba instalować żadnych uzupełnień tego rodzaju. Administrator posiada wszystkie najważniejsze funkcje związane z zarządzanem plikami i dostępem do nich nie tylko lokalnie na komputerze ale i w chmurze internetowej. IOS korzysta z systemu plików APFS. Wymiany niewielkich ilości plików między dowolnymi komputerami tego sameg uzytkownika można dokonać przesyłając je jako załączniki na własny adres elektroniczny o odbierając na komputerze docelowym. Przeglądarka internetowa iOS nosi nazwę *Safari*. Do rozmów wideotelefonicznych służy zainstalowany standardowo program *Facetime*.



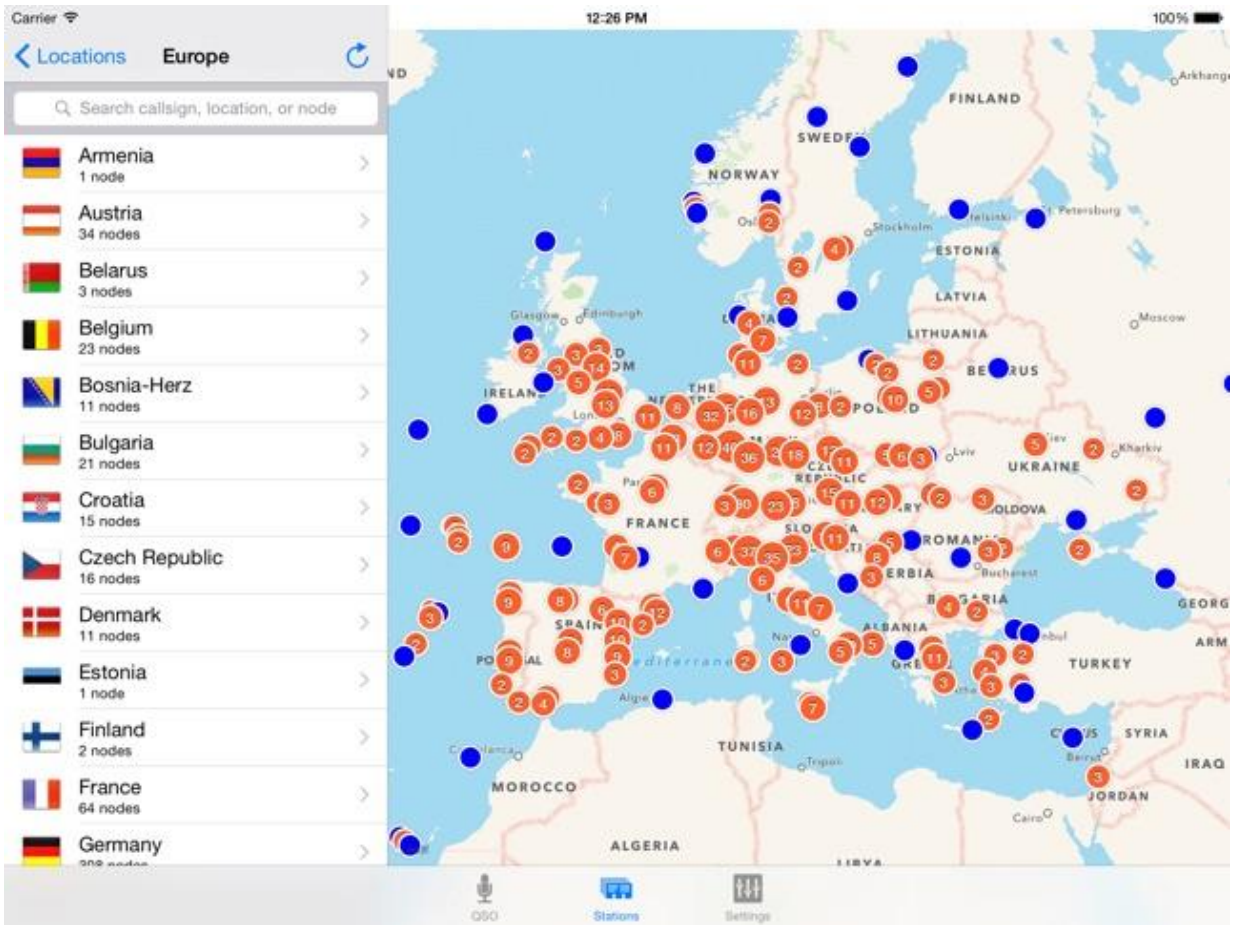
Rys. 2.1.1. Gniazdko i wtyczka „Lighting”

Tabela 2.1.1
Wyprowadzenia w gniazdku normy „Lighting”

1	masa
2	D0+
3	D0-
4	Identyfikator ID0
5	Zasilanie
6	D1-
7	D1+
8	Identyfikator ID1

2.2. Łączności echolinkowe

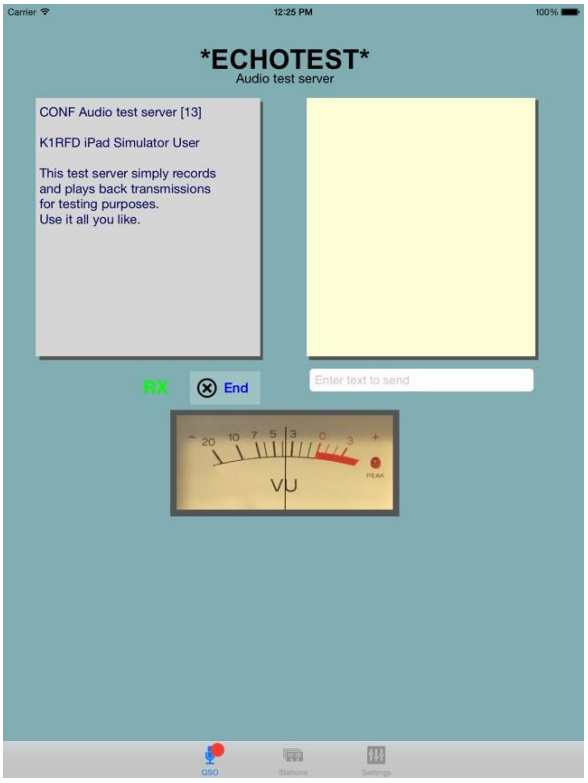
Echolink dla iOS pracuje na telefonach *iPhone* i komputerach tabliczkowych *iPad* pod iOS w wersjach od 10.0 wzwyż. Program jest bezpłatnie dostępny w sklepie internetowym *Apple App Store*. Tak jak zawsze po uprzednim zameldowaniu się i uwierzytelnieniu licencji instalacja wymaga jedynie podania własnego znaku i hasła ustalonego w trakcie pierwszego zameldowania. Identycznie jak w wersji androidowej użytkownik może być połączony w danym czasie tylko z jedną stacją echolinkową. Możliwości i sposób korzystania z programu oraz niektóre problemy omówiono w punkcie poświęconym wersji androidowej. Podobnie jak w niej użytkownik ma do wyboru tryb korzystający z serwerów pośredniczących „Relay" zalecany w przypadku dostępu do sieci przez telefon komórkowy lub przez WiFi – nie wymaga on żadnego odblokowywania kanałów logicznych (ang. *port*), tryb bezpośredni („Direct") i możliwości korzystania z serwerów pośredniczących „Proxy" publicznych lub prywatnych. Analogicznie jak w wersji dla Windows w trybie bezpośrednim konieczne jest otwarcie w modemie dostępowym do Internetu kanałów logicznych 5198 i 5199 dla UDP. W trybie „Relay" uzyskanie połączenia między użytkownikami internetowymi nie zawsze udaje się za pierwszym razem, można w takim przypadku spróbować ponownie albo połączyć się z którąć z akurat nie używanych stacji radiowych (-R lub -L) i poinformować o tym przyszłego korespondenta. W trybie automatycznym program sam wybiera wariant najlepszy w danej sytuacji.
Wygląd okien Echolinku jest zasadniczo bardzo podobny jak w wersji androidowej. Również przyciski są łatwe do rozpoznania. Wysterowanie przy nadawaniu wskazuje symulowany miernik wychyłowy zamiast paskowego. Dużym plusem jest natomiast przedstawienie na iPadzie stacji i ich lokalizacji na mapie, co ułatwia nie tylko orientację ale i wybór celu. Mapy nie są wyświetlane na iPhonie.



Rys. 2.2.1. Okno lokalizacji stacji na iPadzie

Program może współpracować z mikrofono-słuchawkami i mikrofono-głośnikami przez łącze Bluetooth. W przypadku korzystania z dostępu do Internetu prze telefon komórkowy warto najpierw spraw-

dzić, czy nie spowoduje to dodatkowych nieprzewidzianych kosztów. Przypadek taki może się zdarzyć zwłaszcza w czasie pobytów zagranicznych.



Rys. 2.2.2. Okno łączności ze wskaźnikiem siły sygnału

2.3. DMR

Bezpłatny program *DMR Monitor* odczytuje z sieci DMR na bieżąco znaki wywoławcze czynnych stacji i wyświetla je na ekranie informując w ten sposób użytkownika o bieżącej aktywności. Oprócz tego dostępne są listy użytkowników DMR z podaniem ich znaku wywoławczego, identyfikatora i imienia, spisy reflektorów i przemienników i okno przedstawiające lokalizacje przemienników na mapie.

2.4. APRS

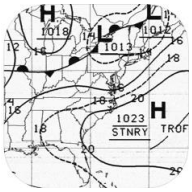
Bezpłatny program *MyAPRS* autorstwa Fabrice Aneche odbiera z APRS-IS docierające tam komunikaty pozycyjne stacji i wyświetla je na ekranie. Może on także dekodować komunikaty meteorologiczne. W drugim oknie lokalizacje stacji są wyświetlane na tle mapy. Do wyboru są także komunikaty i lokalizacje przemienników. Użytkownik podaje w ustawieniach swoją lokalizację dla ograniczenia komunikatów do ineresującej go okolicy. Program pracuje na komputerach i telefonach z iOS w wersji 8 lub nowszej. Dla przyszłych wersji zapowiadana jest możliwość połączenia programu z radiostacją APRS. Fabrice Aneche jest także autorem programuu *SatSat* służącego do śledzenia satelitów.
Podobne możliwości oferuje także płatny program *aprs.fi*. Dodatkowo może być on też stosowany w odbiorczych bramkach radiowo-internetowych. *APRS Mobile* nadaje natomiast własną pozycję do serwerów APRS-IS, i wyświetla na mapie położenie stacji na podstawie otrzymanych stamtąd komunikatów. Nie pozwala on na nadawanie innych komunikatów tekstowych.

2.5. Emisje cyfrowe

SSTV Slow Scan TV firmy *Black Cat Systems* (*www.blackcatsystems.com*) dla komputerów iPad i telefonów iPhone jest dostępny w sklepie internetowym *AppleStore*. Służy on do nadawania i odbioru analogowych obrazów SSTV. Do połączenia komputera z radiostacją można użyć modemu albo złącza Bluetooth, ale wystarczy także sprzężenie akustyczne.

Program obsługuje najpopularniejsze normy takie jak Martin M1 – M4, Scottie S1 – S4, DX, Robot 12C – 72C, czarno-białą 8 sekundową, 12, 24 i 36 sek., MP, MR, PD i inne, z tym, że część mniej rozpowszechnionych tylko odbiorczo.
Dekodowanie obrazów rozpoczyna się automatycznie w momencie odebrania sygnału. Program rozpoznaje automatycznie format obrazu, ale w trudnych warunkach odbioru operator może go nastawić ręcznie. Program zapisuje odebrane obrazy, a obrazy nadawane mogą pochodzić z zapisanych uprzednio plików albo z aparatu fotograficznego komputera.

Bezpłatny program *PSK31* firmy *Black Cat Systems* jest jedynie programem odbiorczym. Na ekranie oprócz zdekodowanych tekstów znajduje się wskaźnik wodospadowy ułatwiający dostrojenie do stacji. Do odbioru wystarczy akustyczne sprzężenie komputera z odbiornikiem.
Programem nadawczo-odbiorczym PSK31 jest *PSKer*.



Odpłatny program *HF Weather Fax* tej samej firmy umożliwia odbiór i wyświetlanie na ekranie map pogody odebranych na falach krótkich lub przez satelity NOAA w paśmie 137 MHz. Program dekoduje tony APT i jest dostępny również w wersjach dla systemów macOS i dla Windows.

Navtex Pad firmy *Black Cat Systems* służy nie tylko do odbioru komunikatów dla żeglugi nadawanych w systemie Navtex na częstotliwości 518 kHz i innych, ale także do odbioru transmisji dalekopisowych w kodzie Baudota i ASCII. Użykownik ma do wyboru różne szybkości transmisji i odstępy częstotliwości W najprostszym przypadku wystarczy akustyczne sprzężenie telefonu albo komputera z odbiornikiem. Odebrane teksty mogą być rejestrowane do późniejszego przejrzenia.
Ta sama firma oferuje również program dekodujący lotnicze transmisje w systemie ACARS dla iOS i Androida.

Program *Hellschreiber* firmy *Black Cat Systems* umożliwia odbiór i nadawanie w dalekopisowym systemie Hella w (podstawowej) normie *FeldHell*. Telefon lub komputer mogą być akustycznie sprzężone z radiostacją lub połączone z nią kablowo.

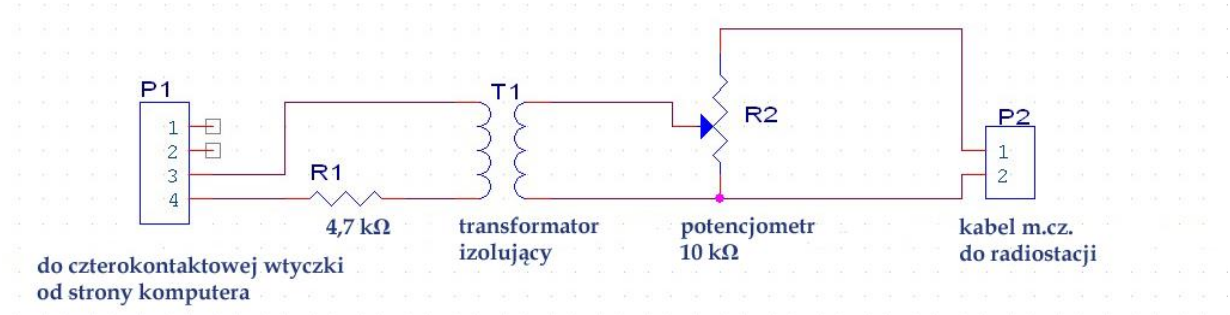
2.6. Analiza lokalnej sieci

IP Network Scanner wyświetla spis urządzeń połączonych z lokalną siecią wraz z ich adresami IP. Dane te mogą być przydatne do analizy sytuacji w sieci, wykrywania niepożądanych użytkowników lub ułatwienia połączenia się wybranym urządzeniem – przykładowo z mikroprzemiennikiem D-STAR/DMR (ang. *hotspot*) – dla zmiany ustawień, przełączenia rodzaju pracy, sieci, emisji, przemienników itp. W wersji bezpłatnej wyświetlany jest spis najwyżej dziesięciu wykrytych urządzeń. Program może także nadawać kontrolne datagramy *ping*.

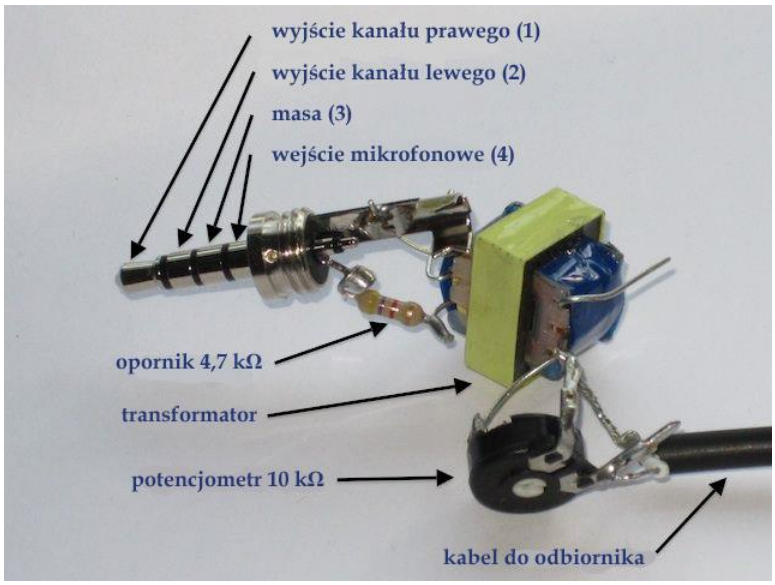
Dla systemu Windows istnieje również podobne, dobrze działające narzędzie o nazwie *Advanced IP Scanner* natomiast dla Androida dostępne są niestety wersje zaśmiecone reklamami i oknami sprzedaży różnych guciów. Guciów można wprawdzie nie kupować, ale utrudnia to sensowne korzystanie z programu.

2.7. Modem odbiorczy

Schemat modemu odbiorczego łączącego wyjście głośnikowe lub słuchawkowe odbiornika albo radiostacji z wejściem iPod albo iPhona przedstawiono na rysunku 2.7.1. W układzie użyty jest miniaturowy transtormatorek m.cz. 600 Ω i przekładni 1:1. W oryginalnym transformatorze firmy Xicon typu 42TL016-RC podanym na schemacie oporność uzwojenia pierwotnego dla prądu stałego wynosi 65 Ω, a pierwotnego 55 Ω, dopuszczalne napięcie między uzwojeniami 100 V, a pasmo przrnoszenia 300 – 3400 Hz. Zasadniczo nie powinno więc sprawić trudności znalezienie innego pasującego typu transformatora.



Rys. 2.7.1. Schemat ideowy modemu. Komputer musi na wejściu widzieć oporność około 5 kΩ dlatego konieczne jest włączenie opornika w szereg z uzwojeniem wtórnym transformatora. Potencjometr R2 służy do regulacji wysterowania wejścia komputera



Fot. 2.7.2. Konstrukcja i wyprowadzenia na wtyczce 3,5 mm

2.8. Odbiorniki programowalne

2.8.1. Odbiorniki lokalne

Odpłatny program *SDR Receiver* obsługuje odbiorniki RTL-SDR, Airspy HF+ i SDRplay. Muszą one być połączone z oddzielnym komputerem (Mac, PC lub Maliną) i dostępne w sieci lokalnej. *SDR Receiver* odbiera strumienie danych przez serwery tcp_sdr, hfp_tcp lub rsp_tcp. Dla odbiorników Airspy HF+ do wyboru są częstotliwości próbkowania 192 lub 384 kb/s. W odbiornikach RTL-SDR możliwa jest regulacja wzmocnienia przedwzmacniacza w.cz. Program demoduluje sygnały AM, wąskopasmową modulację FM i szerokopasmową stosowaną w radiofonii. Możliwy jest import i eksport spisów stacji w formatach csv lub tsv.

Współpracujący z odbiornikami RTL-SDR program *rtl_tcp SDR* demoduluje sygnały AM, SSB, CW i FM. Na ekranie wyświetlane jest widmo odbieranych sygnałów i wskaźnik wodospadowy. Dostęp do odbiornika odbywa się przez serwer *rtl_tcp* gdyż bezpośrednie podłączenie odbiornika do urządzeń z iOS nie jest możliwe. Poprzez serwery *hfp_tcp* i *rsp_tcp* możliwy jest też dostęp odpowiednio do odbiorników Airspy HF+ i SDRPlay.

2.8.2. Odbiorniki internetowe

Z odbiorników internetowych najlepiej korzystać przez przeglądarkę internetową. Nie wymaga to instalacji dodatkowego oprogramowania i nie pociąga dodatkowych kosztów. Temat ten, na przykładzie odbiornika „Kiwi" omówiono dokładniej w rozdziale 6.

3. Arduino

Arduino jest jednopłytkowym mikrokomputerem opracowanym dla celów dydaktycznych, ale szybko zyskał on popularność wśród krótkofalowców i innych hobbystów. Liczba opublikowanych w Internecie programów z różnych dziedzin i bibliotek programów ułatwiających życie programistom jest ogromna i ciągle rośnie. Wiele z nich można znaleźć w witrynach *https://arduino.cc*, *https://github.com* i w witrynach o tematyce krótkofalarskiej. Wyposażenie i możliwości Arduino nie mogą się wprawdzie równać z możliwościami *Maliny*, ale też do wielu prostych zadań szkoda byłoby zatrudniać *Malinę*.
Do większości modeli Arduino oferowane są też płytki rozszerzeń (ang. *shield*) nakładane bezpośrednio na płytkę procesora i dodające wiele przydatnych funkcji takich jak ethernetowe połączenie sieciowe, połączenie z siecią WiFi, komunikację w protokole ZigBee, transmisję packet-radio (AX.25), dodatkowe złącza USB, Bluetooth, dodatkowe wejścia i wyjścia logiczne, dodatkowe przetworniki analogowo-cyfrowe, przetworniki cyfrowo-analogowe, sterowanie silnikami, przekaźnikami, wyświetlanie danych na wyświetlaczach ciekłokrystalicznych, moduły zegarowe, odtwarzanie dźwięku, odbiór radiowy albo pasma amatorskich, odbiór GPS, kodery i dekodery DTMF i wiele innych. Są wśród nich płytki prototypowe.
W konstrukcjach amatorskich często stosowane są wyświetlacze ciekłokrystaliczne o formacie 2 linii po 16 znaków albo wyświetlacze Nokii 5110 oparte na sterowniku Philippsa PCD8544. Jako syntezery częstotliwości popularne są moduły z AD9850 i moduły Si5351A. Dla Arduino i *Maliny* dostępna jest też szeroka gama czujników rozmaitych wielkości fizycznych nadających się do zasosowań telemetrycznych itp.

Tabela 3.1. Najpopularniejsze modele Arduino

Parametr	Arduino UNO	Arduino MEGA 2560	Arduino Nano	Arduino MKRZero
Mikroprocesor (szerokość słowa)	ATmega328P (8 bitów)	ATmega2560 (8 bitów)	ATmega328 (8 bitów)	SAMD21 Cortex-MO + (32 bitowa ARM MCU)
Napięcie pracy [V]	5	5	5	3,3
Napięcie zasilania [V]	7 – 12	7 – 12	7 – 12	5
Pobór prądu [mA]			19	
We./wy. logiczne	14 (6 z mod. szer. impulsów)	54 (15 z mod. szer. impulsów)	22 (6 z mod. szer. impulsów)	22 (12 z mod. szer. impulsów)
Wy. z mod. szer. imp.	6	15	6	12
We. analogowe (A-C)	6 (10 bitów)	16 (10 bitów)	8 (10 bitów)	7 (8/10/12 bitów)
Wy. analogowe (C-A)				1 (10 bitów)
Złącze UART	1	4	1	1
Złącze I2C				1
Złącze SPI				1
Zewnętrzne przerwania				10
Obciąż. wyjścia [mA]	20	20	40	7
Obciąż. wy. 3,3 V [mA]	50	50		600
Pamięć programu [kB]	32 (0,5 program ładujący)	256 (8 program łądujący)	32 (2 program ładujący)	256 (+ 8 program ładujący)
Pamięć robocza [kB]	2	8	2	32
Pamięć EEPROM [kB]	1	4	1	brak
Częstotliwość zegarowa [MHz]	16	16	16	48 (lub 32,768 kHz gen. RC)
Długość [mm]	68,6	101,52	45	61,5
Szerokość [mm]	53,4	53,3	18	25
Masa [g]	25	37	7	

Uwagi:

– wyprowadzenia Arduino Mega 2560 są w podstawowej części kompatybilne z Arduino UNO (poza dodatkowymi na tylnej części płytki), można więc korzystać z tych samych płytek rozszerzeń.
– parametry innych modeli, płytek rozszerzeń i prototypowych znajdują się m.in. w witrynie https*://www.arduino.cc/en/Main/Products.*
– Arduino MKZero jest przykładowym reprezentantem serii modeli Zero.

3.0.1. Złącza

Procesory stosowane w Arduino posiadają pewną liczbę programowalnych wejść i wyjść logicznych wydających lub odbierających poziomy logiczne 0 (0 V) i jedynkę (5 lub 3,3 V) o kierunku ustalanym w programie. Są one wewnętrznie wyposażone w oporniki 20 – 50 kΩ podciągające je do napięcia zasilania, które można odłączyć programowo. Kilka z nich może także służyć jako wyjścia z modulacją szerokości impulsów z ośmiobitową rozdzielczością. Po włączeniu filtrów dolnoprzepustowych służą one jako wyjścia analogowe. Impulsy wyjściowe rozpoczynają się zawsze na początku wewnętrzrnego cyklu i kończą w jego trakcie zależnie od przetwarzanej wartości analogowej. Wnosi to dodatkowe błędy przetwarzania, których możnaby uniknąć gdyby impulsy były symetryczne względem środka cyklu. Lepszym rozwiązaniem w niektórych sytuacjach może być więc stosowanie zewnętrznych przetworników C-A, chociażby oporowych typu R-2R, a jeszcze lepiej specjalnych przetworników scalonych.
Oprócz tego procesory posiadają kilka wejść analogowych połączonych z przetwornikami analogowo-cyfrowymi o rozdzielczości zależnej od typu procesora, ale najczęściej jest to 10 bitów. Przetworniki cyfrowo-analogowe posiadają tylko niektóre typy procesorów (np. stosowane w Arduino Due, serii Zero). Do połączenia z komputerem PC i ładowania programów, a przy pracy autonomicznej także do innych celów służy dwuprzewodowe złącze szeregowe UART. Jest ono obsługiwane przez standardową bibliotekę *Serial*. Niektóre procesory są wyposażone w kilka sprzętowych (fizycznych) złączy szeregowych, w innych można korzystać z ich programowej emulacji (odpowiednie biblioteki można bez trudu znaleźć w Internecie). Oprócz tego przy wykorzystaniu dodatkowych bibliotek (lub wyposażenia fizycznego) do użytku są magistrale I2C, SPI, *1Wire* itd.

Magistrala *1Wire* jest stosowana do komunikacji z pamięciami zewnętrznymi i różnego rodzaju czujnikami z przepustowością do 16,3 kb/sek na odległości do 100 metrów. Do adresowania urządzeń podłączonych do wspólnej linii służy zapisany w nich fabrycznie 64-bitowy adres (numer). Niektóre z tych urządzeń mogą być także zasilane przez przewód magistrali.

Magistrala SPI (*Serial Peripherial Interface*) służy do komunikacji między jednym urządzeniem nadrzędnym, sterującym (ang. *master*) i większą liczbą podporządkowanych (ang. *slave*). Magistrala składa się z czterech przewodów: zegarowego (oznaczanego jako SCLK, SCK lub CLK), adresowego służącego do wyboru urządzenia (oznaczanego jako SS lub CS – *Slave Select* lub *Chip Select*), komunikacyjnego wyjściowego z urządzenia sterującego (oznaczanego jako MOSI – *Master Out Slave In*) i komunikacyjnego wejściowego (oznaczanego jako MISO – *Master In Slave Out*). Szybkości transmisji mogą przekraczać 100 Mb/sek zaleznie od realizacji i trybu pracy (0, 1, 2 lub 3). W Arduino UNO przewody SCLK, MISO i MOSI są połączone odpowiednio z wyprowadzeniami (ang. *pin*) 13, 12 i 11. Wyprowadzenie 10 może być użyte do wyboru pierwszego urządzenia na magistrali, ale w zasadzie do adresowania urządzenia może służyć dowolne wyprowadzenie.
W Arduino Mega2560 sygnały SCLK, MOSI i MISO odpowiadają kolejno wyprowadzeniom 52, 51 i 50, a do adresowania pierwszego urządzenia często stosuje się wyprowadzenie 53. Sygnały złącza SPI są w Arduino UNO R3, Mega2560, DUE i niektórych innych modelach doprowadzone do osobnych gniazd kontaktowych (ICSP). Istnieją także biblioteki służące do programowej symulacji złącza SPI. Biblioteki są czymś w rodzaju programowych „układów scalonych" zawierających zbiór podprogramów (funkcji) przeznaczonych do wykonywania określonych zadań. Programista nie musi znać ich rozwiązań wewnętrznych, a jedynie nazwy funkcji i sposoby ich wywołania (parametry) oraz ewentualne udzielane przez nie odpowiedzi, podobnie jak elektronikowi wystarczy wiedzieć jakie elementy elektroniczne musi podłączyć do wyprowadzeń układu scalonego i jak ich wartości wpływają na jego funkcję. Nie musi on natomiast znać dokładnego schematu ideowego wnętrza układu.

Magistrala I2C (*Inter-Integrateg Circiut bus*) służy do połączenia układów scalonych wchodzących w skład urządzenia z mikrokomputerem. W Arduino magistrala ta nosi również oznaczenie TWI (*Two-Wire Interface*). Jest to ośmiobitowa szeregowa magistrala przeznaczona do łączności dwukierunkowej. W wielu rozwiązaniach służy ona również do komunikacji z zewnętrznymi urządzeniami peryferyjnymi. Magistrala zawiera oprócz masy jedynie dwa przewody: szeregowych danych (SDA) i sygnału zegarowego (SCL). W Arduino UNO do tego celu użyto odpowiednio wyprowadzeń A4 i A5, a w Arduino Mega2560 i DUE odpowiednio wyprowadzeń 20 i 21. Arduino DUE i niektóre inne modele posiadają również drugie złącze z przewodami oznaczonymi jako SDA1 i SCL1.
W zależności od trybu pracy szybkości transmisji w złączu I2C wynoszą 10, 100, 400 kb/sek, 1 lub 3,4 Mb/sek. Standardowo w Arduino stosowana jest szybkość 100 kb/s, ale można ją zmienić modyfikując parametry w bibliotece *Wire*. Każde z przyłączonych do magistrali urządzeń dysponuje jednoznacznym 7- lub 10-bitowym adresem. W niektórych możliwy jest wybór adresu za pomocą zworek. W Arduino zakresy 0 – 7 i 120 – 127 są zarezerwowane, a więc dla urządzeń peryferyjnych pozostaje pula 112 adresów. Urządzenia peryferyjne są włączone do magistrali za pomocą stopni tranzystorowych z otwartym drenem, konieczne jest więc połączenie przewodów magistrali z napięciem zasilania przez oporniki podciągające o oporności około 4,7 kΩ.
Podobnie jak w przypadku pozostałych złączy istnieją również gotowe biblioteki I2C pozwalające na wykorzystanie innych dowolnych wyprowadzeń procesora. Sposobem szybszym i skuteczniejszym jest korzystanie ze złączy sprzętowych, a z symulowanych programowo jedynie wtedy kiedy nie ma innej możliwości.

3.0.2. Programy dla Arduino

Programy dla Arduino (ang. *sketch*) zawierają dwie podstawowe funkcje (części): *void setup()* i *void loop()* oraz mniejszą lub większą liczbę funkcji napisanych dla konkretnego celu.
Pierwsza z nich jest wykonywana tylko raz po starcie programu i służy do nadania wartości początkowych zmiennym, wprowadzenia parametrów do urządzeń peryferyjnych, wybrania sposobu ich pracy itp. Typ *void* oznacza, że funkcja nie zwraca żadnej odpowiedzi, która mogłaby być wykorzystana w jakimś miejscu progamu do wykrycia błędu, który pojawił się w trakcie pracy lub w innym celu. Funkcje innych typów int, float itd. dają odpowiedź w postaci zmiennej podanego typu. Mogą to być przykładowo wyniki obliczeń albo meldunki o wykonaniu zadania, informacje o zaistniałych błędach itd. W języku C i spokrewnionych poszczególne wydzielone części programów są nazywane funkcjami, podczas gdy w innych językach programowania stosowane są zamiast tego, równolegle lub w pewnym zakresie znaczeń takie nazwy jak podprogram, procedura itp.
Funkcja *loop()* jest wywoływana jako następna i jest ona wykonywana w nieskończonej petli aż do zatrzymania programu (wyłączenia mikrokomputera). W odróżnieniu od programów dla innych mikrokomputerów nie potrzeba więc zapisywać explicite nieskończonej pętli w programie. Funkcja *loop()* zawiera wszelkie polecenia wykonywane przez program, wywołania innych potrzebnych funkcji, obliczenia, modyfikacje wartości zmiennych, komunikację ze światem zewnętrznym, wyświetlanie danych itd. W razie potrzeby w nawiasach nastepujących po nazwie funkcji podawane są wartości jej parametrów, np. *obwod_okregu(5)* oznacza wywołanie funkcji obliczającej obwód okręgu o promieniu 5.
Czasmi spotykane są konstrukcje programów, w których w funkcji *setup()* po wstępnych krokach występuje nieskończona pętla *while()*, a funkcja *loop()* pozostaje pusta bo program i tak nie dochodzi do jej wywołania.
Na początku programu przed funkcją *setup()* zapisane są definicje zmiennych z podaniem ich typów (*int, char, byte, float, long*, itd. z ewentualnym modyfikatorem *unsigned* oznaczającym występowanie w niej tylko wartości dodatnich) przed nazwą. Są to zmienne globalne widoczne we wszystkich dalszych częściach (funkcjach) programu. W definicji można nadać im także wartości początkowe. W przypadku „zmiennych", których wartość powinna pozostać stała w programie na początku definicji podawany jest modyfikator *const*. Zmienne typów *int, char, byte, long* są zmiennymi stałoprzecinkowymi przyjmującymi tylko wartości całkowite, natomiast *float* i *double* są typami zmiennoprzecinkowymi przeznaczonymi dla liczb dziesiętnych. Możliwa jest zmiana typów zmiennych (explicite lub automatyczna), ale nie w każdej kombinacji jest to sensowne.
Zmienne zdefiniowane w ramach funkcji są zmiennymi lokalnymi, dostępnymi tylko w niej. Ich zawartość jest tracona po wyjściu z funkcji. Oznacza to, że nazwy zmiennych lokalnych mogą się powtarzać

dowolnie w różnych funkcjach (przykładem może być indeks lub licznik i stosowany w pętlach *for()* itp.). W przypadku ogólnym mogą być to zmienne pojedyńcze albo złożone obiekty zawierające w sobie właściwości (parametry, zmienne) i metody (funkcje, podprogramy). Nazwy elementów tych zmiennych złożonych składają się z nazwy obiektu na początku i następującej po niej nazwy elementu oddzielonych kropką. Przykładem może być *Serial.print()* – funkcja nadawania informacji przez złącze szeregowe w celu ich wyświetlenia. Funkcja *print()* jest elementem złożonego obiektu o nazwie *Serial*. Modyfikator *static* w definicji zmiennej lokalnej powoduje, że wprawdzie zmienna ta nie jest dostępna po wyjściu z funkcji, w której została zdefiniowana, ale w przeciwieństwie do zwykłej zmiennej lokalnej jej zawartość nie jest tracona bezpowrotnie i może być wykorzystana po ponownym wywołaniu tej samej funkcji.
Oprócz tego w programach, w znacznej części przypadków głównie na początku, występują polecenia poprzedzone krzyżykiem:
#define – służące do zdefiniowania pewnego oznaczenia i przypisania mu potrzebnej wartości, np. #define PI 3.14 pozwala na używanie w programie oznaczenia PI wszędzie tam, gdzie potrzebna jest jej wartość. Oznaczenia te zostają zastąpione przez podaną wartość w tekście programu tuż przed jego kompilacją. W przypadku gdyby okazało się, że wartość 3.14 nie jest wystarczająco dokładna wystarczy zastąpić ją w tym jednym miejscu przez potrzebną np. #define PI 3.14159 zamiast modyfikować wiele miejsc w programie. Polecenie #define ABC bez podania wartości oznacza, że symbolowi ABC jest przypisana wartość logicznej jedynki. Można wykorzystywać to w poleceniach względnych w rodzaju *#ifdef*. Zwyczajowo przyjęło się, że nazwy definiowanych w ten sposób symboli zapisuje się dla odróżnienia od innych dużymi literami, ale nie jest to wymóg języka.
#include<nazwa.h> lub #include ”nazwa.h” powoduje włączenie do tekstu programu zawartości plików o podanych nazwach. Pliki o rozszerzeniu .h są tak zwanymi plikami nagłówkowymi i mogą zawierać dowolne deinicje podane za pomocą polecenia #define, definicje zmiennych lub nawet kod źródłowy części programu. W nawiasach spiczastych podawana jest nazwa pliku znajdującego się w standardowym katalogu środowiska programistycznego, a w cudzysłowie – w katalogu programu. Pliki nagłówkowe wywołuje się w programach dla Arduino m.in. w celu włączenia do nich potrzebnych bibliotek jeśli są one zainstalowane w katalogu „libraries” środowiska programistycznego (IDE) Arduino. Standardowo biblioteka składa się z trzech plików: nagłówkowego z rozszerzeniem *.h*, tekstu z rozszerzeniem *.cpp* i *keywords.txt*. Wszystkie one (oraz ewentualne przykłady użycia biblioteki) są kopiowane do katalogu „libraries”.
Warunkowa konstrukcja

```
#ifdef ABC
.....
#else
.....
#endif
```

pozwala na kompilację pewnych części programu lub ich opuszczenie w zależności od wartości logicznej symbolu, w tym przykładzie symbolu ABC. Pozwala to na dodawanie do programu części potrzebnych w trakcie uruchamiania albo później do celów diagnostycznych i opuszczanie ich w wyjściowej wersji programu jedynie przez zmianę definicji (wartości logicznej) jednego symbolu w jednym miejscu bez żmudnego dodawania lub usuwania (albo zamiany na komentarz) wielu poleceń. W ten sam sposób można w jednym pliku źródłowym zapisać dwie wersje programu (lub więcej), przykładowo różniące się zastosowanym wyświetlaczem, ale identyczne w pozostałej części. Gałąź *#else* można opuścić jeśli nie jest potrzebna. Oprócz polecenia *#ifdef* istnieje również polecenie odwrotne *#ifndef*.

Język programowania Arduino składa się z elementów języków C i C++ i jest łatwy do opanowania po krótkim czasie, a tym bardziej dla programistów znających już inne języki. Trudno jednak zamieszczać w tym miejscu wyczerpujący kurs programowania Arduino ze względu na jego objętość. Dlatego też autor odsyła zainteresowanych czytelników do literatury poświęconej temu tematowi. Zamieszczone tutaj informacje mają za zadanie w pierwszym rzędzie pomóc w zrozumieniu omawianych dalej programów także czytelnikom znającym inne języki programowania, przypomnieć lub zwrócić uwagę na niektóre szczegóły, które mogły ulecieć z pamięci, i nie mogą w żadnym wypadku zastąpić systematycznego kursu progamowania dla osób nie mających dostatecznej wiedzy w dziedzinie programowania

Arduino. Użytkownicy, którzy chcą na początek korzystać z gotowych programów zmieniając w nich jedynie własne dane, mogą się ograniczyć do tego uproszczonego wyjaśnienia.
Dane takie jak znak wywoławczy, kwadrat lokatora, moc nadawania itp. są prawie zawsze podane w programach w postaci łańcuchów tekstowych wpisanych do pewnej zmiennej (tabeli) lub zdefiniowanych jako symboje, np.
char znak[] = ”OE1KDA” albo #define ZNAK “OE1KDA“ albo int moc=5. Wystarczy więc poszukać takich miejsc w programie (najczęściej można je znaleźć w pobliżu jego początku) i zamiast liczby albo treści w cudzysłowie wpisać własną uważając aby nie skasować niechcący cudzysłowu, średnika ani niczego innego.

Sprawą trochę rzadziej poruszaną w literaturze dotyczącej programowania dla zastosowań nie pracujących w czasie rzeczywistym nych są przerwania i sposób korzystania z nich. Przerwanie polega na tym, że pod wpływem sygnału ze źródła wewnętrznego takiego jak licznik impulsów albo czasomierz (ang. *timer*) albo sygnału pochodzącego z zewnętrznego urządzenia peryferyjnego wykonywanie głównego programu zostaje przerwane przez specjalny podprogram przerwania, którego zadaniem jest zareagowanie na występującą sytuację. W momencie wywołania podprogramu przerwania zostaje zapisany adres następnego polecenia z programu głównego, tak, że po zakończeniu przerwania kontynuowane jest wykonywanie programu głównego. Przerwania pozwalają na znacznie szybsze zareagowanie przez mikroprocesor na występującą sytuację (np. nadejście nowych danych na którymś ze złączy komunikacyjnych, impulsów na wejściu licznika) aniżeli gdyby odpytywał on cyklicznie wszystkie wchodzące w grę urządzenia. Przerwania związane z licznikami pozwalają także na wykonywanie pewnych zadań w ściśle określonych momentach, czyli w ściśle określonym rytmie. Bez skorzystania z przerwań nie możnaby zagwarantować ani dostatecznie szybkiej reakcji ani zachowania dokładnego rytmu, ponieważ zależałoby to od długości cyklu wykonywania głównego programu. Nie może ona być stała ponieważ zależy to m.in. od liczby wykonywanych poleceń i ich długości w zależności od warunków występujących w programie (rozgałęzień *if*), oczekiwania na dane itp. Przerwania i sposób korzystania z nich są więc sprawami ważnymi zwłaszcza w zastosowaniach technicznych. Należy także zauważyć, że najprostsze mikroprocesory mają możliwość zareagowania na jedno przerwanie z jednego wybranego źródła lub najwyżej na dwa różne, natomiast procesory bardziej rozbudowane dysponują większą liczbą przerwań o różnym priorytecie. Wykonywanie podprogramu przerwania niższego w hierarchii może być więc przerwane przez podprogram przerwania mającego wyższy priorytet. W przypadku ogólnym podprogram przerwania może również zablokować wszystkie pozostałe na początku pracy i włączyć na końcu. Również program główny może wyłączać przerwania w pewnych krytycznych miejscach i włączać po przejściu przez nie, ale należy korzystać z tego bardzo ostrożnie. Konkretne możliwości w tych sprawach zależą od typu procesora. Ponieważ podprogramy przerwań są wywoływane niezależnie od przebiegu głównego programu i praktycznie w jego dowolnym lub prawie dowolnym miejscu niemożliwe jest przekazywanie im parametrów w zwykły sposób. Jedenym sposobem przekazania danych może być skorzystanie ze zmiennych globalnych. To samo dotyczy również przekazania danych w drugą stronę – z programu przerwania do głównego. Podprogramy przerwań powinny być jak najkrótsze i ich wykonanie powinno zajmować możliwie jak najmniej czasu.
Środowisko programistyczne Arduino IDE jest obecnie dostępne w wersjach dla systemów operacyjnych Windows, Linuks i macOS. Pozwala ono na pisanie programów, ich kompilację oraz ładowanie do Arduino, a także na zarządzanie bibliotekami programów. Oprócz różnych procesorów stosowanych w Arduino środowisko umożliwia przygotowywanie progarmów dla innych procesorów np. ESP8266.
W skrypcie zamieszczono jedynie niektóre programy lub ich fragmenty mające ułatwić zrozumienie sposobu rozwiązania danych problemów. Wiekszość programów należy pobrać z podanych adresów z Internetu ponieważ ich zamieszczanie zajęłoby wiele dziesiątków lub nawet więcej stron i wymagalo od czytelników kopiowania tekstu do środowiska Arduino. Wygodniej jest więc pobrać gotowe pliki i otworzyć je w celu zmodyfikowania, kompilacji i wpisania do pamięci procesora.

3.0.3. Przerwania w Arduino

Stosowany w Arduino UNO, Mini i Nano procesor ATmega328 dysponuje możliwością skorzystania z przerwania wywoływanego przez wejścia logiczne 2 lub 3 czyli przez podłączone do nich urządzenia,

stosowany w Arduino Micro procesor 32u4 pozwala na korzystanie z wejść 0, 1, 2, 3, 7, natomiast procesor ATmega2560 – przerwaniami wywoływanymi przez wejścia logiczne 2, 3, 18, 19, 20 i 21.
Przypisanie podprogramu przerwania do wybranego wejścia zapewnia funkcja *attachInterrupt()*. Jej pierwszym parametrem jest numer przerwania, a do powiązania z wejściem przeznaczona jest funkcja *digitalPinToInterrupt()*, jeżeli przykładowo przerwanie ma być powiązane z wejściem 3 to należy podać wywołanie *digitalPinToInterrupt(3)* jako pierwszy argument. Zalecane jest korzystanie z tej funkcji zamiast bezpośredniego podawania numeru przerwania ze względu na ewentualne różnice między poszczególnymi modelami. Drugim argumentem jest adres wywoływanej funkcji przerwania (ang. ISR) czyli jej nazwa, a trzecim sposób pracy: LOW – reakcja na niski poziom na wejściu, CHANGE – reakcja na zmianę poziomu, RISING – reakcja na rosnące zbocze, FALLING – reakcja na opadające zbocze, w nowszych typach procesorów występuje także wariant HIGH – oznaczający wywołanie przerwania przy wysokim stanie logicznym na wejściu
(`attachInterrupt(digitalPinToInterrupt(wejście), adres, tryb)`).
W języku C i C++ nazwa funkcji lub tabeli jest jednocześnie wskazaniem do niej (*pointerem*) czyli adresem początkowym. Do uzyskania adresu innych obiektów programu stosuje się operator gwiazdki, np. *wfs oznacza adres zmiennej wfs.
Z działanem przerwań związany jest modyfikator *volatile* występujący w definicjach zmiennych. Oznacza to, że jej wartość może ulec zmianie w sposób niezauważalny dla progamu (np. w podprogramie przerwania), i że kompilator nie może stosować w stosunku do niej żadnych optymalizacji i musi za każdym razem pobierać jej wartość z pamięci
Przykładowy program wykorzystujący przerwanie:

```
const byte ledPin = 13;
const byte interruptPin = 2;
volatile byte state = LOW;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(interruptPin, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(interruptPin), blink, CHANGE);
}

void loop() {
  digitalWrite(ledPin, state);
}

void blink() {
  state = !state;
}
```

Odłączenie podprogramu przerwania następuje po wywołaniu funkcji *detachInterrupt():*
`detachInterrupt(digitalPinToInterrupt(pin)), np. z argumentem 3.`

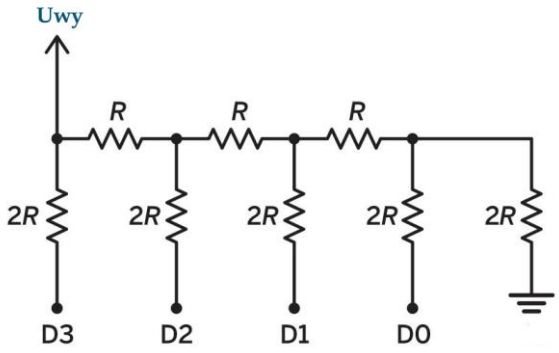
Do włączenia lub wyłączenia przerwań w programie służą odpowiednio funkcje *interrupts()* i *noInterrupts()*. Przykład użycia:

```
void loop()
{
 noInterrupts();
 // krytyczna część programu
 interrepts();
 // pozostała część programu
}
```

Biblioteka *Timer* (włączana do programu za pomocą polecenia *#include* <*Timer.h*>) pozwala na korzystanie z przerwań wywoływanych przez liczniki *Timer1* i *Timer2* po osiągnięciu przez nie zadanego stanu lub stanu maksymalnego. Umożliwia to wywoływanie przerwań i wykonywanie różnych czynności przez Arduino w stałym rytmie czasowym. Ich przeprogramowanie wpływa jednak na czasy trwania funkcji opóźnienia (delay(), millis() itp.), a więc należy mieć to na uwadze przy korzystaniu

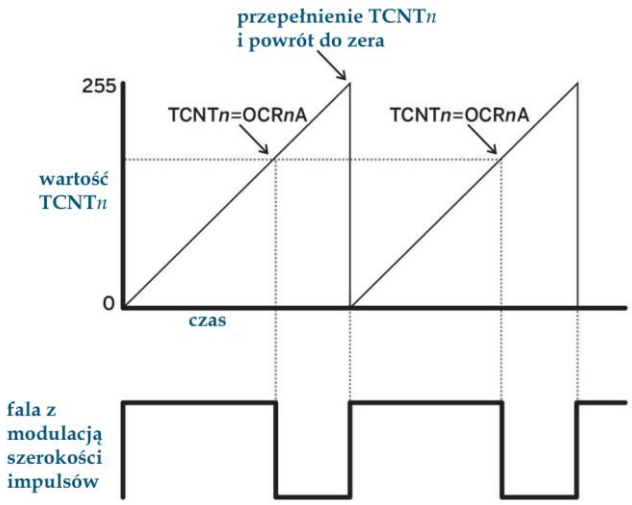
z przerwań programowych. W Arduino Mega2650 istnieją trzy dodatkowe liczniki *Timer3, Timer4* i *Timer5,* które nie są do niczego używane w standardowych biliotekach i poleceniach. Są one więc dostępne dla wszystkich innych dowolnych zastosowań, a więc również do wywoływania przerwań.

Najprostszym sposobem generacji sygnałów dźwiękowych (sygnałów niskiej częstotliwości; m.cz.) jest generowanie fali prostokątnej na jednym z wyjść logicznych przy użyciu funkcji *tone().* Wyjscia logiczne mogą sterować nawet głośniczki ośmioomowe o średniczch kulku centymetrów. Stosunek czasów trwania jedynki i zera daje współczynnik wypełnienia fali. Wywiera on wpływ na jakość i barwę dźwięku. Fala prostokątna o współczynniku wypełnienia 50% (symetryczna) zawiera jedynie nieparzyste harmoniczne, natomiast przy jego zmianie pojawiają się również parzyste harmoniczne i stosunki ich amplitud również ulegają zmianom. Sposób ten nadaje sie jedynie do prostych zastosowań w rodzaju brzęczyka sygnalizacyjnego. Funkcja *analogWrite(wyprowadzenie, wypełnienie)* generuje na wybranym wyjściu z modulacją szerokości impulsów (ang. PWM; pol. MSI) falę prostokątną o stałej częstotliwości i podanym jako argument współczynniku wypełnienia. Dla Arduino UNO, Nano i Mini na wyjściach 3, 9, 10 i 11 jest to częstotliwość 490 Hz, a na wyjściach 5 6 – 980 Hz, w Arduino Mega jest to ton 490 Hz na wyjściach 2, 3, 5 – 12 i 44 – 46, a na wyjściach 4 i 13 – ton 980 Hz. Częstotliwości dla pozostałych modeli Arduino są różne i należy upewnić się w dokumentacji (*www.arduino.cc*). Współczynnik wypełnienia leży w zakresie 0 – 255. Dla odfiltrowania harmonicznych należy na wyjściu dodać filtr dolnoprzepustowy. Przy generacji sygnałów o częstotliwościach stałych (PSK31, Hell itp.) albo zmieniających się w wąskim zakresie 9WSPR, FT8 itd.) dobór częstotliwości granicznej filtru nie przedstawia większej trudności.
Lepszą jakość dźwięku uzyskuje się generując sygnały analogowe za pomocą przetwornika cyfrowo-analogowego (C–A). Ponieważ Arduino nie jest wyposażony w przetwornik C–A konieczne jest dołączenie urządzenia zewnętrznego. Może być to przetwornik w postaci obwodu scalonego sterowany przez wyprowadzenia wyjściowe procesora, ale stosunkowo prostym i dającym dobre rezultaty (nie tylko w zastosowaniach amatorskich) jest przetwornik typu R-2R. Zawiera on oporniki o wartościach R (przykładowo 10 kΩ) i o wartościach 2R (w tym przykładzie 20 kΩ) w liczbie odpowiadającej rozdzielczości bitowej. Dla przetwornika 4-bitowego są to więc po cztery oporniki każdej wartości. Zmniejszenie wpływu obciążenia na pracę przetwornika zapewnia obciążenie go wtórnikiem napięciowym na wzmacniaczu operacyjnym itp. Na wyjściu przetwornika otrzymywany jest przebieg schodkowy. Jego wygładzenie wymaga włączenia filtru dolnoprzepustowego.



Rys. 3.0.3.1. Przetwornik R-2R

Dla uzyskania dobrych rezultatów przetwornik wymaga doboru oporników z dużą dokładnością.
Mikroprocesor ATmega328(P) posiada trzy czasomierze. Każdy z nich składa się z licznika zliczającego impulsy o częstotliwości zegarowej, które w momencie osiągnięcia maksymalnego stanu (w momencie przepełnienia, wystąpienia przeniesiena) wracają do stanu zerowego i kontynuują zliczanie. Są one w dokumentacji oznaczone jako TCNT*n*, gdzie *n* jest numerem licznika (0 – 2). Liczniki *Timer0* (TCNT0) i *Timer2* (TCNT2) są licznikami ośmiobitowymi o zakresie 0 – 255, a licznik TCNT1 jest licznikiem 16-bitowym o zakresie 0 – 16535. Może on także pracować jako licznik 8-bitowy. Każdy z liczników może pracować w kilku trybach. Licznik TCNT1 jako jedyny dysponuje trybem „fast PWM". W trybe tym w momencie przepełnienia wyjście przyjmuje stan wysoki (logicznej jedynki) sygnalizując rozpoczęcie nowego cyklu. TCNT1 posiada rejestr porównawczy OCR1A Przy osiągnięciu stanu równego wartości wpisanej do rejestru sygnał na wyjściu zmienia się na zero logiczne. Sytuacja ta powtarza się cyklicznie dzięki czemu generowany jest ciąg ompulsów o modulowanej szerokości.
Zmiana wartości rejestru OCR1A w regularnych odstępach czasu na wartości pobrane z tabeli chwilowych wartości (próbek) przebiegu umożliwia generowanie przebiegów o dowolnych kształtach. Tabela taka może być uprzednio przygotowana w trakcie programowania i zapisana w pamięci procesora.



Rys. 3.0.3.2. Generacja fali z modulacją szerokości impulsów

Poniżej podany jest przykładowy kod źródłowy programu generującego falę prostokątną.

```
#include <avr/interrupt.h> // użycie biblioteki przerwań licznika
/******** Parametry fali sinusoidalnej ********/
#define PI2 6.283185 // Definicja 2*PI
#define AMP 127 // Amplituda fali
#define OFFSET 128 // składowa stała aby wartości >0
/******** Tabela próbek ********/
#define LENGTH 256 // Długość tabeli
byte wave[LENGTH]; // Definicja tabeli

void setup() {
/* Wypełnienie tabeli wartościami próbek sinusoidy*/
for (int i=0; i<LENGTH; i++) { // Kolejne indeksy tabeli
  float v = (AMP*sin((PI2/LENGTH)*i)); // Obliczanie wartości
  wave[i] = int(v+OFFSET); // Zapis jako integer
 }
/****Przełączenie licznika 1 na tryb ośmiobitowy „fast PWM" ****/
 pinMode(9, OUTPUT); // Przełączenie wyprowadzenia 9 na wyjście PWM
 TCCR1B = (1 << CS10); // Włączenie wstępnego dzielnika 16MHz
 TCCR1A |= (1 << COM1A1); // Zero na wyjściu gdy TCNT1=OCR1A
 TCCR1A |= (1 << WGM10); // Tryb 8-bitowy „fast PWM"
 TCCR1B |= (1 << WGM12);
/******** Ustawienie licznika 2 do wywołania przerwania (ISR) ********/
 TCCR2A = 0; // W rejestrze A nie ma dodatkowych ustawień
 TCCR2B = (1 << CS21); // Nastawienie dzielnika na podział przez 8
 TIMSK2 = (1 << OCIE2A); // Wywołanie ISR gdy TCNT2 = OCRA2
 OCR2A = 32; // Ustawienie częstotliwości generowanej fali
 sei(); // Włączenie przerwań
}

void loop() { // Pętla pusta!
}

/******** Przerwanie za każdym razem gdy TCNT2 = OCR2A ********/
ISR(TIMER2_COMPA_vect) { // Wywołanie gdy TCNT2 == OCR2A
 static byte index=0; // Wskazywanie każej pozycji w tabeli
 OCR1AL = wave[index++]; // Korekta dla wyjścia PWMt
 asm("NOP;NOP"); // Precycyjne dobranie długości podprogramu
```

```
 TCNT2 = 6; // Kompensacja czasu wykonywania ISR
}
```

Program oblicza na początku wartości próbek i zapisuje je w tabeli (lepszym rozwiązaniem byłoby przygotowanie ich w czasie programowania i zapisanie w pamięci programu dla zaoszczędzenia pamięci roboczej). Wartości pobrane z pamieci są ładowane do rejestru porównawczego OCR1A we właściwych momentach czasu. Następnie licznik TCNT1 rozpoczyna generowanie ciągu impulsów o odpowiedniej szerokości. Licznik ten został przestawiony z trybu 16-bitowego na tryb 8-bitowy. Czasomierz TCNT2 generuje wywołania przerwania (podprogramu ISR) w momentach gdy jego stan zrówna się z zawartością OCR2A. Podprogram ISR nie różni się niczym od pozostałych funkcji poza tym, że nie ma zadeklarowanego typu. Przy zliczaniu częstotliwości zegarowej 16 MHz przerwania zystępowałyby częściej niż potrzeba. Dlatego też konieczne jest właczenie dzielnika wstepnego. Przy podziale przez 8 TCNT2 zlicza sygnał 2 MHz.
Częstotliwość generowanej fali zależy od wartości podanej w rejestrze OCR2A. Dla jej obliczenia należy zliczaną częstotliwość (2 MHz) podzielić przez zawartość OCR2 i następnie wynik podzielić przez długość tabeli. Przy długości tebeli 128 próbek otrzymuje się:

f_{TCNT2} / (zawartość_OCR2A) x długość_tabeli) = 2 MHz / (128 x 256) = 61,04 Hz.

Dla uwzględnienia czasu wykonywania podprogramu przerwania (ISR) stan początkowy licznika TCNT2 zamiast zera wynosi 6. Dokladniejsze dopasowanie czasu wykonywania podprogramu uzyskano przez dodanie polecenia *asm(„NOP;NOP”)*. Każda z operani NOP trwa przez jeden cykl zegarowy.
Dla wypełnienia tabeli można użyć dowolnych funkcji matematycznych generujących wartości próbek dla dowolnych kształtów fali np. trójkątnej, piłokształtnej itp. Korzystając z funkcji *random()* można generować przebieg pseudoszumowy (w tym i w następnym przykładzie do wypełnienia tabeli służy funkcja *waveform()* zamiast pętli zapisanej na czerowo w programie powyżej, a reszta programu nie ulega zmianie):

```
void waveform(byte w) {
 switch(w) {
  case SINE:          // sinusoida
   for (int i=0; i<LENGTH; i++)
    {float v = OFFSET+(AMP*sin((PI2/LENGTH)*i));
    wave[i]=int(v);
   }
   break;

 case RAMP:       // fala piłokształtna
   for (int i=0; i<LENGTH; i++) {
    wave[i]=i;
   }
   break;

 case TRIANGLE:  // fala trójkątna
   for (int i=0; i<LENGTH; i++) {
    if (i<(LENGTH/2)) {
      wave[i]=i*2;
    } else {
      wave[i]=(LENGTH-1)-(i*2);
    }
   }
   break;

 case SQUARE:     // fala prostokątna
   for (int i=0; i<(LENGTH/2); i++) {
    wave[i]=255;
```

```
   }
   break;

  case RANDOM:     // prabieg pseudoszumowy
   randomSeed(2);
   for (int i=0; i<LENGTH; i++) {
    wave[i]=random(256);
   }
    break;
  }
}
```

Do wypełnienie tabeli probkami brdziej złożonych przebiegów można także skorzystać z syntezy opartej o zasadę Fouriera. W poniższym przykładzie do przebiegu sinusoidalnego o częstotliwości podstawowej dodawana jest trzecia harmoniczna o amplitudzie 1/4 amplitudy częstotliwości podstawowej. Dla trzeciej harmonicznej wartość licznika w pętli jest pomnożona przez 3 a amplituda podzielona w tym przykładzie przez 4:

```
for (int i=0; i<LENGTH; i++) { // dlas każdego miejsca w tabeli
  float v = (AMP*sin((PI2/LENGTH)*i)); // Podstawowa
  v += (AMP/4*sin((PI2/LENGTH)*(i*3))); // Krok dla 3 harmonicznej
  wave[i]=int(v+OFFSET); // Zapis jako integer
  }
```

Na podobnej zasadzie można dodawać więcej dowolnych harmonicznych dla uzyskania potrzebnego przebiegu (w przykładzie poniżej 8 harmonicznych o amplitudach dzielonych przez zawartośći tabeli amplitude[]):

```
void additive(void) {
  #define PARTIALS 8
  float harmonic[PARTIALS] =  {1,3,5,7,9,11,13,15};
  float amplitude[PARTIALS] = {1,3,5,7,9,11,13,15};
  float v;
  int i,j;

  for (int i=0; i<LENGTH; i++) {
    v = 0;
    for (int j=0; j<PARTIALS; j++) {
      if (amplitude[j] > 0) {
        v += (AMP/(amplitude[j]+1)*sin((PI2/LENGTH)*(i*harmonic[j])));
      }
    }
    v = constrain(v+OFFSET,0,255);
    wave[i]=byte(v);
  }
}
```

W ostatnim przykładzie tabela próbek została wygenerowana za pomocą oddzielnego programu na PC i wposana do kodu żródłowego do pamięci programu dla zaoszczędzenia miejsca w pamięci programu.

```
/******** Załadowanie bibliotek przerwań czasowych AVR ********/
#include <avr/interrupt.h>
#include <avr/pgmspace.h>

/******** Lookup table ********/
#define LENGTH  256  // The length of the waveform lookup table
```

```
prog_char ref[256] PROGMEM = {
128,131,134,137,140,143,146,149,152,155,158,161,164,167,170,173,
176,179,182,185,187,190,193,195,198,201,203,206,208,210,213,215,
217,219,222,224,226,228,230,231,233,235,236,238,240,241,242,244,
245,246,247,248,249,250,251,251,252,253,253,254,254,254,254,254,
255,254,254,254,254,254,253,253,252,251,251,250,249,248,247,246,
245,244,242,241,240,238,236,235,233,231,230,228,226,224,222,219,
217,215,213,210,208,206,203,201,198,195,193,190,187,185,182,179,
176,173,170,167,164,161,158,155,152,149,146,143,140,137,134,131,
128,124,121,118,115,112,109,106,103,100, 97, 94, 91, 88, 85, 82,
79,  76, 73, 70, 68, 65, 62, 60, 57, 54, 52, 49, 47, 45, 42, 40,
38,  36, 33, 31, 29, 27, 25, 24, 22, 20, 19, 17, 15, 14, 13, 11,
10,  9,  8,  7,  6,  5,  4,  4,  3,  2,  2,  1,  1,  1,  1,  1,
1,   1,  1,  1,  1,  1,  2,  2,  3,  4,  4,  5,  6,  7,  8,  9,
10,  11, 13, 14, 15, 17, 19, 20, 22, 24, 25, 27, 29, 31, 33, 36,
38,  40, 42, 45, 47, 49, 52, 54, 57, 60, 62, 65, 68, 70, 73, 76,
79,  82, 85, 88, 91, 94, 97,100,103,106,109,112,115,118,121,124};

byte wave[LENGTH];

void setup() {
  for (int i=0; i<LENGTH; i++) {
    wave[i] = pgm_read_byte(&ref[i]);
  }

  /******** Ustawienia licznina 1 na 8-bitowy tryb „fast PWM” ********/
  pinMode(9, OUTPUT);
  TCCR1B  = (1 << CS10);
  TCCR1A |= (1 << COM1A1);
  TCCR1A |= (1 << WGM10);
  TCCR1B |= (1 << WGM12);

  /******** Ustawienia dla licznika 2 do wywoływania ISR ********/
  TCCR2A = 0;
  TCCR2B = (1 << CS21);
  TIMSK2 = (1 << OCIE2A);
  OCR2A = 32;
  sei();
}

void loop() {  // Pętla pusta!
}

ISR(TIMER2_COMPA_vect) {
  static byte index=0;
  OCR1AL = wave[index++];
  asm("NOP;NOP");
  TCNT2=6; //korekcja na czas wykonywania ISR
}
```

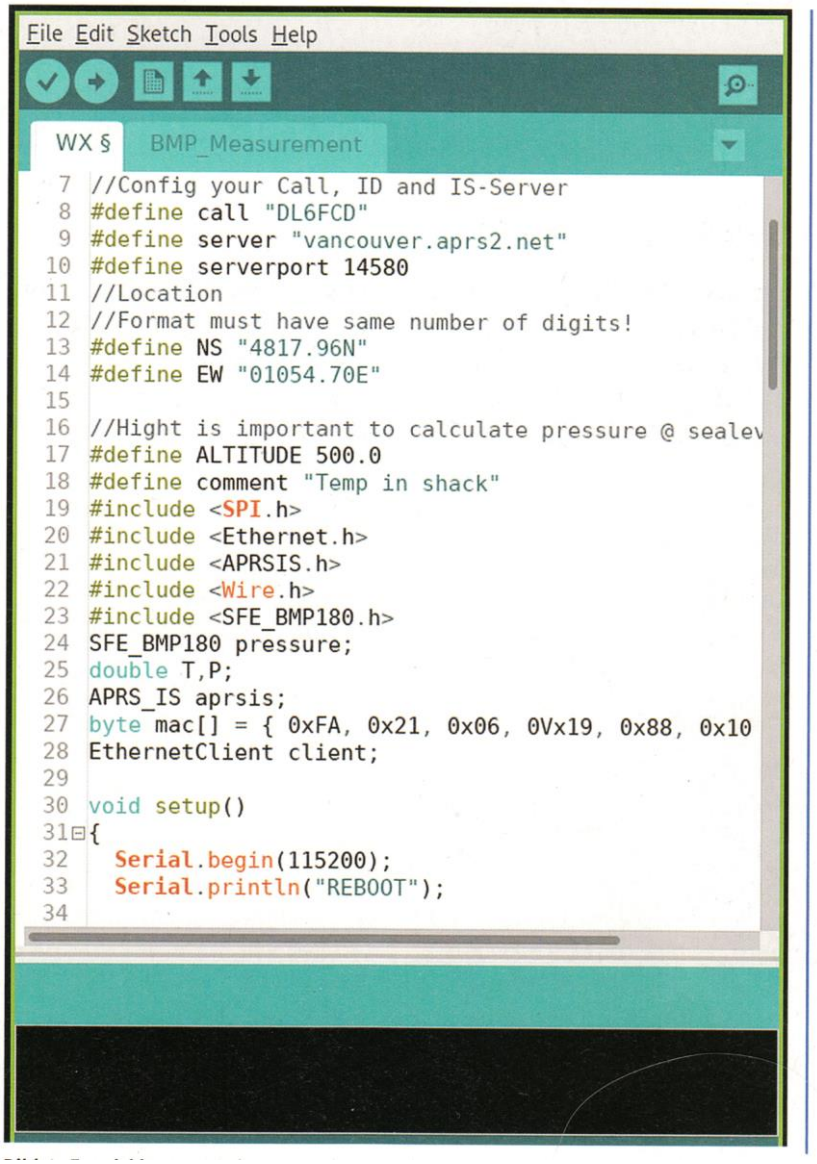
Powyższe przykłady powinny ułatwić zaprogramowania generatorów sygnałów różnych emisji amatorskich w rodzaju PSK31 itp. do wykorzystania w radiolatarniach albo innych stacjach specjalnych. Zamiast wydawać próbki w postaci impulsów o zmiennej szerokości można je wydawać równolegle na wyjściach logicznych na przetwornik R-2R.

3.1. APRS

3.1.1. Internetowy klient APRS

W systemie transmisji danych APRS, możliwe jest przekazywanie nie tylko znaków i współrzędnych geograficznych stacji, ale również i krótkich wiadomości o dowolnej treści. W Europie radiowa sieć APRS pracuje na częstotliwości 144,800 MHz, chociaż w rejonach o większej liczbie użytkowników stosowane są również częstotliwości w paśmie 70 cm (432,500 MHz).
Pakiety danych nadawane są emisją FM z przepływnością 1200 bit/s przy użyciu dwóch tonów akustycznych 1200 i 2200 Hz. W wielu krajach istnieje gęsta sieć stacji APRS retransmitujących odebrane pakiety przez radio (przekaźniki cyfrowe, ang. *digipeater*) lub przekazujących je do internetowych serwerów APRS-IS (bramki *iGate*). Dane przekazane do serwerów internetowych są wyświetlane na tle mapy pod adresem *aprs.fi*. Wyświetlane tam komunikaty mogą oprócz znaku i pozycji stacji zawierać informacje meteorologiczne, dane przemienników i lokalne aktualności.
Opracowane przez DL6FCD rozwiązanie umożliwia, za przystepną cenę nie tylko przekazywanie do sieci wybranych przez operatora wiadomości, ale także zdalną obsługę wybranych urządzeń, czyli reagowanie na ustalone pakiety danych o dowolnej treści. Możliwa jest przykładowo reakcja na zbliżanie się określonej stacji ruchomej.
Kod programu jest na tyle nieskomplikowany, że nawet osoby o mniejszym doświadczeniu w programowaniu mogą korzystać z niego i dokonywać potrzebnych modyfikacji. Praca klienta nie pociąga za sobą transmisji radiowych (wymiana danych następuje jedynie przez Internet), a więc nie wymagana jest stała obecność operatora przy stacji.
Koszty Arduino wraz pomocniczym oprogramowaniem są niewielkie, a dodatkowo w Internecie znajduje się wiele darmowych bibliotek ułatwiających programowanie i korzystanie przez Arduino z przewodowego dostępu do sieci. Płytkę Arduino i dodatkową płytkę ethernetową można kupić korzystnie, a środowisko programistyczne IDE i dokumentacja są dostępne bezpłatnie pod adresem [3.1.1.2]. Zasadniczo wystarczy wykorzystanie Arduino UNO i dodatkowej plytki ethernetowej zapewniającej kablowy dostęp do sieci. Jedynie bardziej rozbudowane stacje mogą wymagać zastosowania dodatkowych elementów. Kabel USB zapewnia połączenie z komputerem dla zaprogramowania Arduino i jego zasilanie. Do zasilania w trakcie stałej pracy wystarczy sieciowy zasilacz USB.
Przekazywanie danych do sieci APRS wymaga nawiązania połączenia z jednym z licznych serwerów APRS-IS, których aktualny spis znajduje się m.in pod adresem [3.1.1.3]. Wejście na stronę informacyjną serwera możliwe jest przez przeglądarkę internetową w kanale logicznym 14501. Strona udostępnia podstawowe informacje o serwerze i pozwala na sprawdzenie zameldowania się na nim własnego Arduino (po jego uruchomieniu). Arduino musi w celu wymiany danych nawiązać połączenie z serwerem APRS-IS w kanale logicznym 14580.
Zameldowanie na serwerze wymaga podania własnego znaku amatorskiego i obliczonego na jego podstawie hasła dostępu. Hasło to jest obliczane przy użyciu 16-bitowego algorytmu rozpraszającego (ang. *hash*). W celu obliczenia hasła dla własnego znaku wywoławczego można skorzystać z usługi internetowej dostępnej m.in. pod adresem [3.1.1.9]. Dodatkowo do hasła podawana jest nazwa i wersja używanego programu.
Przykładowy pakiet zawierający wszystkie potrzebne do zameldowania dane może wyglądać następująco: „user OE1KDA pass 12345 vers ArduPRS 1.0”. Serwer potwierdza prawidłowe zameldowanie za pomocą komunikatu *#login OE1KDA verified*. Od tego momentu rozpoczyna się wymiana pakietów danych w formacie TNC2. Dla odbioru pakietów należy wpisać kryterium filtrowania. Spis możliwych kryteriów znajduje się m.in. pod adresem [3.1.1.4]. Wbrane kryterium jest przesyłane do serwera w postaci specjalnego pakietu TCP/IP, przykładowo dla odbioru pakietów rozpoczynających się od znaku SP5AHT byłby to pakiet *filter b/SP5AHT**.
Pakiet w formacie TNC2 posiada następującą strukturę: *Znak nadawcy>znak adresata, znaki stacji pośredniach „via”, dane użytkowe*. Przykładowy meldunek pozycyjny wygląda jak następuje:
OE1KDA>APRX28, TCPIP, qAC, T2MSSOURI: !4821.76N/01621.98EriGate in Wien*. Przekazane prawidłowo dane są wyświetlane na mapach pod adresem *aprs.fi* i na innych stronach poświęconych APRS. Szczegółowe informacje na temat formatu pakietów znajdują się pod adresem [3.1.1.6] .
Dla odciążenia programistów od zajmowania się szczegółami tworzenia pakietów powstała biblioteka programów o nazwie APRSIS. Jest ona dostępna w Internecie pod adresem [3.1.1.5]. Pobrany plik ZIP

należy rozpakować do katalogu „library” znajdującego się w głównym katalogu środowiska „Arduino”. Można też dokonać tego w IDE korzystając z punktów menu „Sketch” -> „Include Library” -> „Add ZIP Library”. Po włączeniu biblioteki do środowiska Arduino można już modyfikować przykładowe programy (rys. 3.1.1.1).

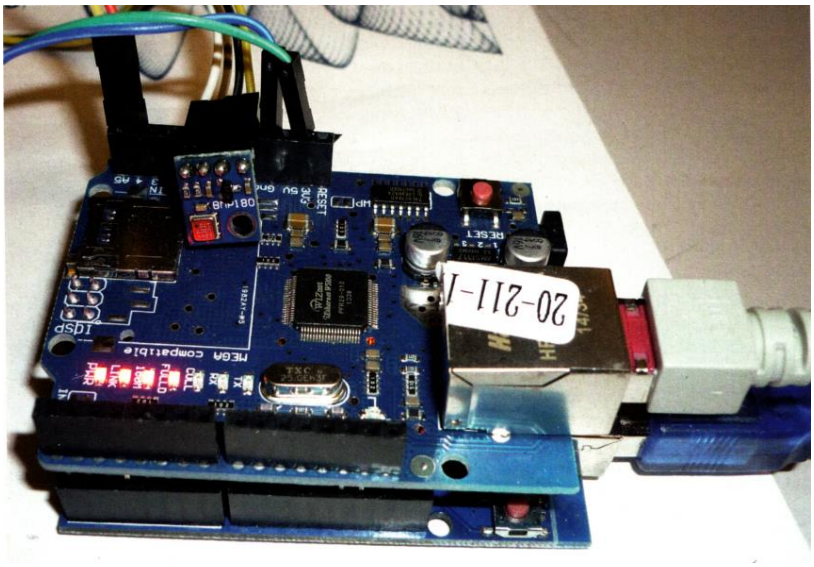


Rys. 3.1.1. Fragment przykładowego programu w oknie środowiska programistycznego Arduino

Rozpowszechnianie zawiadomień o wieczorach klubowych lub innych spotkaniach ułatwiają tzw. elementy (ang. *item*) o długości do 9 znaków alfanumerycznych. Na mapach APRS są one wyświetlane identycznie jak znaki wywoławcze z dodatkiem wybranego symbolu.
Do nadania pakietów z tego typu treścią służy biblioteczna funkcja aprsis.sendItem(), przykładowo:
aprsis.sendItem(„Spotkanie”, „4817.30N”, „01654.70E”,”K”, „/”, „co wtorek od 18:00”).
Pierwszym argumentem funkcji jest tytuł (opis) o długości do 9 liter, następnie podawane są współrzędne GPS we właściwym dla APRS formacie, kolejne dwa znaki – tutaj „K” i „/” – służą do wyboru symbolu z tabeli symboli APRS, i ostanim argumentem jest dowolny tekst informacyjny. Dla rozpowszechniania informacji o stacji przemiennikowej wywołanie funkcji mogłoby wyglądać następująco:
aprsis.sendItem(„SR5XXX”, „4817.30N”, „01654.70E”,”r”, „/”, „439,100 MHz, -7,6, FM”).
Dla komunikatów stacji amatorskich można użyć również funkcji aprsis.sendCall():
aprsis.sendCall(„SR5XXX”, „4817.30N”, „01654.70E”,”r”, „/”, „439,100 MHz, -7,6, FM”).
Na ilustracji 3.1.1.2 przedstawiony jest wycinek dostępnej publicznie mapy z „OpenStreetMap” z lokalizacją stacji DL6FCD.

Rys. 3.1.1.2. Przykładowy komunikat stacji klubowej zawiera oprócz podstawowych informacji o niej meldunek o temperaturze i ciśnieniu atmosferycznym. Położenie stacji jest widoczne na tle mapy „OpenStreetMap”

Dla rozsyłania komunikatów meteorologicznych można w najprostszym rozwiązaniu zamiast kosztownej stacji meteorologicznej podłączyć do Arduino czujnik temperatury i ciśnienia atmosferycznego BMP180 albo pasujące do tego celu czujniki innych typów (patrz poz. [3.1.1.10]). Procesor komunikuje się z czujnikiem za pomocą magistrali I2C, ale jest to znowu sprawa prosta dzięki istnieniu odpowiedniej biblioteki. Standardowe pakiety meteorologiczne muszą mieć specjalny format, szczegółowo opisany w dokumencie [3.1.1.8]. W witrynie pod adresem [3.1.1.7] DL6FCD udostępnił gotową wersję programu wykorzystującego czujnik BMP180. Program wymaga jedynie wpisania własnego znaku wywoławczego i współrzędnych geograficznych. Rozwiązanie to pracuje już od wielu miesięcy rozpowszechniając komunikaty o ciśnieniu atmosferycznym i temperaturze panującej w pomieszczeniu stacji co 15 minut. Zdaniem tłumacza bardziej interesujące byłoby podawanie w komunikatach temperatury zewnętrznej, ale wymagałoby to umieszczenia na zewnątrz Arduino wraz czujnikiem – w miejscu zabezpieczonym przed opadami i innymi wpływami atmosferycznymi. Oczywiście można też zrezygnować z podawania temperatury i pozostawić czujnik w zamkniętym pomieszczeniu. Proste komunikaty zawierające jedynie jedną lub dwie wielkości meteorologiczne nie muszą też obowiązkowo mieć standardowego formatu meteorologicznego, mogą to być zwykłe komunikaty tekstowe. W ten sam sposób można podawać zmierzone wartości innych wielkości fizycznych, jeśli tylko mogą one zainteresować odbiorców, przykładowo poziomu zanieczyszczeń unoszących się w atmosferze albo lokalnych zakłóceń atmosferycznych (aktywności elektromagnetycznej atmosfery). Wymaga to oczywiście zainstalowania odpowiedniego czujnika i dokonania niezbędnych zmian w programie dla jego odczytu. Programowanie Arduino nie jest jednak trudne i znalezienie pomocnych przykładów też nie...
Klient APRS może również odbierać pakiety danych i analizować ich (dowolną) treść. Pozwala to na zdalne sterowanie różnymi urządzeniami – w najprostszym przypadku na ich włączanie lub wyłączanie – z dowolnego miejsca na świecie, albo na automatyczną reakcję na zbliżanie się określonej stacji np. do lokalu klubu. Przykładowy program do tego celu znajduje się również pod adresem [3.1.1.7].
Skorzystanie z podanych powyżej przykładów nie wymaga kosztownego i skomplikowanego wyposażenia ani zaawansowanych umiejętności w dziedzinie programowania. Opublikowane w Internecie przykładowe kody źródłowe programów dają się łatwo modyfikować i dostosowywać do własnych potrzeb.



Fot. 3.1.1.3. Arduino UNO z ethernetową płytką rozszerzeń

3.1.2. Modem APRS

W nadawczo-odbiorczym modemie APRS (opartym na konstrukcji „MicroAPRS" duńskiego krótkofalowca Marka Qvista) zastosowano prosty i tani model „Arduino Nano". OE7MBT opracował wprawdzie płytkę drukowaną, na której dostępne są także wyprowadzenia Arduino przeznaczone do podłączenia odbiornika GPS, modułu Bluetooth i wyświetlacza [3.1.2.3], ale układ jest stosunkowo prosty i można umieścić go również na płytce uniwersalnej. Zamiast elementów powierzchniowych najlepiej wówczas użyć klasycznych podzespołów przewlekanych.

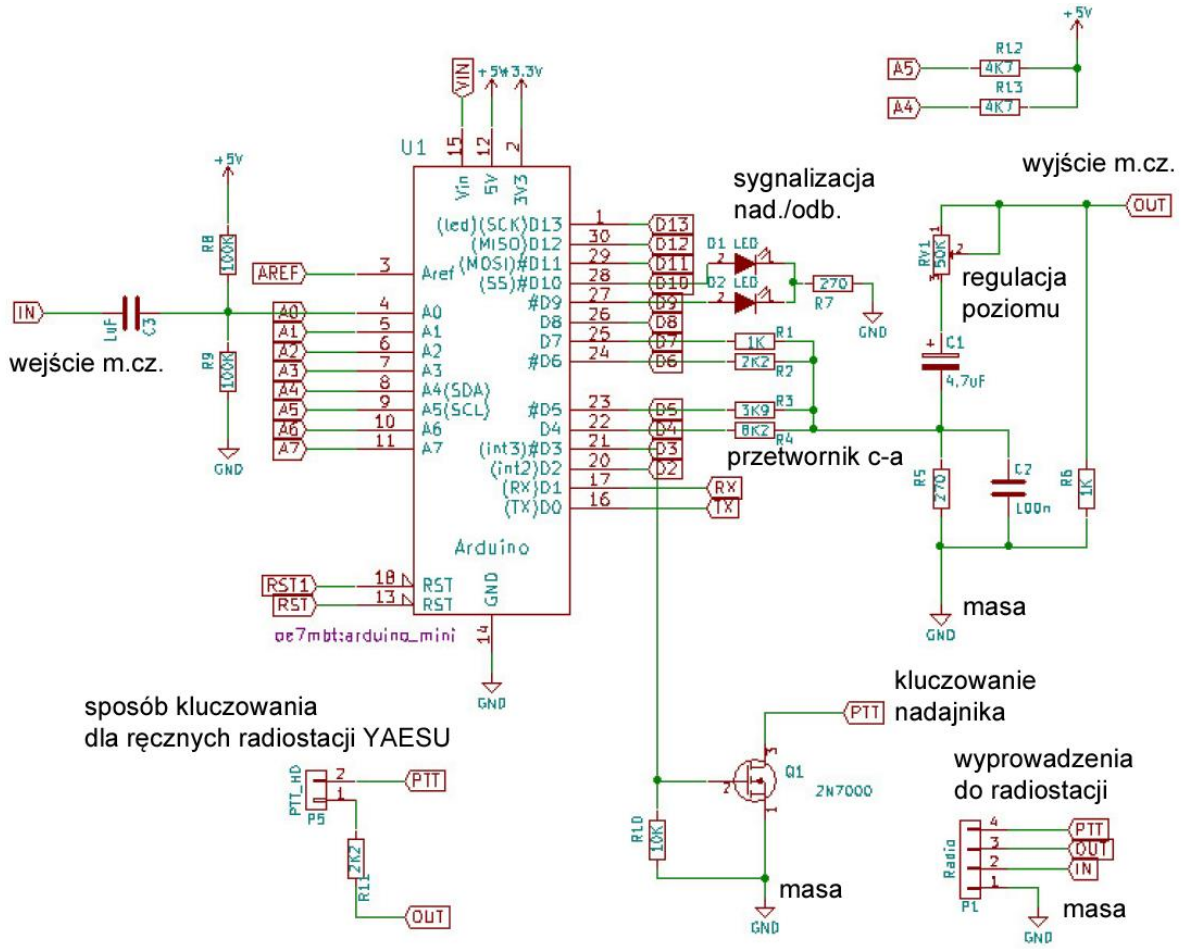
Oporniki R1 – R4 tworzą czterobitowy przetwornik cyfrowo-analogowy podłączony do wyjść D1 – D8 Arduino. Powstający na jego wyjściu sygnał schodkowy jest następnie filtrowany za pomocą filtru dolnoprzepustowego złożonego z oporności wyjściowej przetwornika i elementów R5C2, a do regulacji poziomu na wyjściu służy dzielnik napięcia złożony z potencjometru RV1 i opornika R6. Polowy tranzystor Q1 typu 2N7000 kluczuje nadajnik, a diody elektroluminescencyjne D1 i D2 sygnalizują stan pracy modemu – odpowiednio odbiór lub nadawanie. Wejście Arduino należy połączyć z gniazdkiem słuchawkowym radiostacji, a wyjście m.cz. z jej wejściem mikrofonowym. Sposób podłączenia tranzystora kluczującego nadajnik należy sprawdzić w instrukcji obługi sprzętu. W większości modeli musi on być połączony z osobnym kontaktem w gniazdku radiostacji, ale w niektórych modelach radiostaji Yaesu konieczne jest zwarcie przewodu mikrofonowego do masy przez opornik o podanej przez producenta wartości.

Oprogramowanie modemu jest dostępne w dwóch wersjach:dla trybu KISS lub dla trybu ze złączem szeregowym, a jego tor odbiorczy jest przystosowany do pracy przy otwartej blokadzie szumów. Maksymalna dopuszczalna długość pakietu KISS wynosi 564 bajty. Oprogramowanie wewnętrzne modemu i zestawienie poleceń dla niego (w postaci dokumentu PDF) są dostępne w witrynie [3.1.2.2].

W torze radiowym modem pracuje z typową przepływnością 1200 bitów/s (AFSK), natomiast na złączu szeregowym dla komputera stosowana jest szybkość 9600 bit/s z ustawieniem 8N1.

Spośród wymienionych w zestawieniu poleceń do najważniejszych należą wprowadzenie własnego z naku z rozszerzeniem, znaku docelowego z ewentualnym rozszerzeniem i polecenie nadania pakietu o podanej treści.

Mark Qvist opracował również bibliotekę APRS dla Arduino ułatwiającą pisanie własnych programów APRS [3.1.2.4] oraz oprogramowanie przekaźnika cyfrowego (ang. *digipeater*) APRS dla „MicroAPRS".

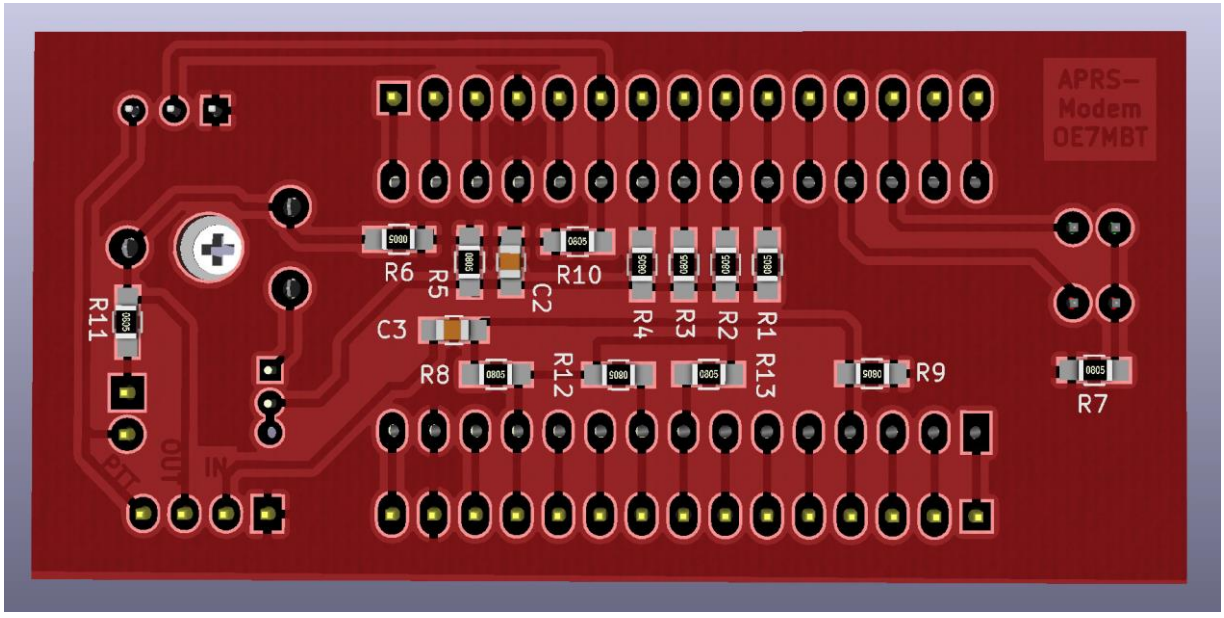


Rys. 3.1.2.1. Schemat ideowy modemu

Fot. 3.1.2.2. Modem APRS podłączony do ręcznej radiostacji

Tabela 3.1.2.1
Najważniejsze polecenia dla modemu w wersji ze złączem szeregowym

Polecenie	Znaczenie	Przykłady
c<własny znak>	Wprowadzenie własnego znaku wywoławczego	cOE1KDA sc11 S
sc<rozszerzenie>	Wprowadzenie rozszerzenia znaku	
S	Zapis konfiguracji	
mc<cel>	Wprowadzenie znaku docelowego	mcAPRS ms1 #Temperatura 20 stopni
ms<rozszerzenie>	Wprowadzenie rozszerzenia znaku docelowego	
#tekst	Nadanie podanego tekstu	
lla<szerokość geograficzna>	Wprowadzenie własnej szerokości geograficznej w formacie NMEA	lla48.21N llo16.21E @Arduino MicroAPRS
llo<długość geograficzna>	Wprowadzenie własnej długości geograficznej w formacie NMEA	
@<dodatkowy komentarz>	Dodanie komentarza do komunikatu pozycyjnego	
1<znak>	Wprowadzenie znaku pierwszego przemiennika na trasie	1WIDE1 s1-1 2WIDE2 s2-1 otrzymywanie: WIDE1-1,WIDE2-1
2<znak>	Wprowadzenie znaku drugiego przemiennika na trasie	
s1<rozszerzenie>	Wprowadzenia rozszerzenia dla pierwszego przemiennika	
s2<rozszerzenie>	Wprowadzenie rozszerzenia dla drugiego przemiennika	
!<komunikat>	Nadanie komunikatu pozycyjnego, surowego, nie przekodowanego, do wprowadzenia mocy, wysokości i zysku anteny służą odpowiednio polecenia lp, lh, lg z parametrem w zakresie 0 – 9	!=4821.10N/01621.70E-PHG2410Arduino MicroAPRS
Przykład powyżej: komunikat pozycyjny z dodatkiem mocy, wysokości anteny i jej zysku		

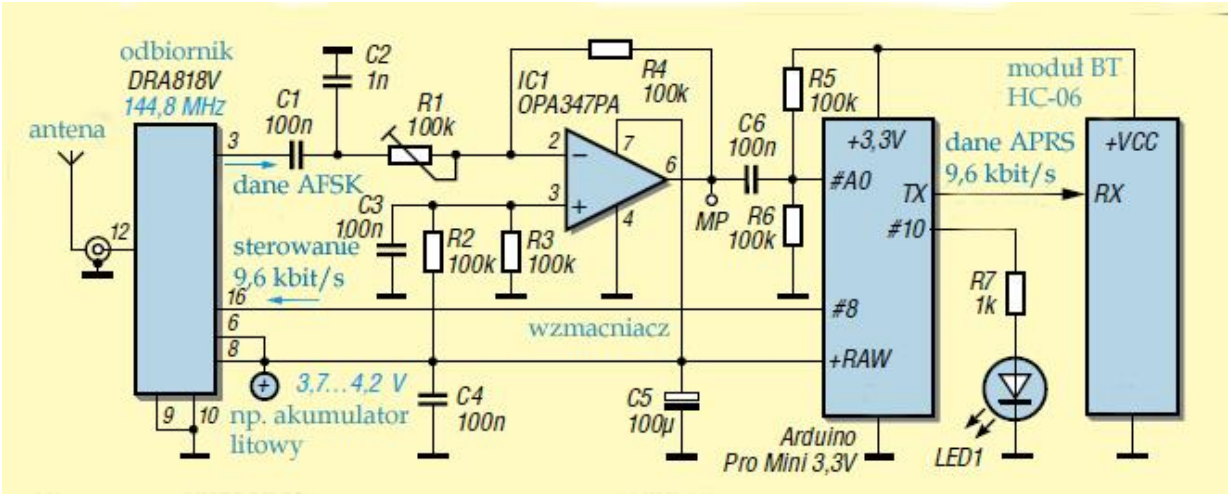


Rys. 3.1.2.3. Płytka drukowana modemu

3.1.3. Dekoder APRS

Opracowany przez DJ7OO (*Funkamateur* 8/2019) składa się z odbiornika na DRA818V (albo DRA818U dla odbioru alternatywnych częstotliwości w paśmie 70 cm), mikrokomputera Arduino Pro Mini w wersji 3,3 V i modułu Bluetooth HC-06. Nadawczo-orbiorczy moduł DRA818 pracujący z cyf-

rową obróbką sygnałów (COS, ang. DSP) jest używany wyłącznie jako odbiornik dostrojony do stałej częstotliwości 144,800 MHz, 432,500 MHz lub innej używanej lokalnie. Arduino służy jako modem i dekoder dla pakietów APRS, które są następnie nadawane przez złącze Bluetooth do telefonu albo komputera androidowego z programam APRSDroid. Zamiast niego można także korzystać z programu dekodującego opracowanego przez DJ7OO. Schemat ideowy układu przedstawiono na rysunku 3.1.3.1. Odebrany przez odbiornik DRA818V sygnał AFSK kluczowany z szybkością 1200 bodów jest doprowadzony przez wzmacniacz operacyjny OPA347PA do wejścia analogowego A0 (nóżka 3) mikroprocesora. Potencjometr montażowy R1 służy do regulacji wzmocnienia. Arduino korzysta z biblioteki opracowanej przez Marka Quista OZ7TMD i dostępnej pod adresem *http://unsigned.io/project/libaprs*. Użycie biblioteki upraszcza znacznie opracowanie programu. Po nieznacznej modyfikacji mikrokomputer dostarcza na wyjściu pakietów danych na złączu szeregowym również w formacie MIC-E z przepływnością 9600 bit/s. Są one transmitowane do komputera przez złącze Bluetooth pracujące na niedrogim module HC-06. Po każdym uruchomieniu odbiornika mikroprocesor przesyła dane sterujące do modułu odbiorczego.
Dekodowanie pakietów APRS wymaga włączenia w ustawieniach komputera androidowego złącza BT i jednorazowego sparowania go z modułem HC-06 odbiornika. Maksymalna dopuszczalna odległość między odbiornikiem i komputerem wynosi około 20 m. W APRSDroidzie należy wybrać jako lokalne źródło danych złącze BT i protokół TNC („Plain TNC"). *APRSDroid* pozwala na przestawianie pozycji i tras odbieranych stacji na mapach. Zamiast niego można w charakterze dekodera użyć programu W2APRS autorstwa JA7UDE.
Program dla Arduino znajduje się w Internecie pod adresem
www.kh-gps.de/APRS_DECODER_519.zip,
a program dekodujący DJ7OO dla Androida – pod adresem
www.kh_gps.de/APRS_APK_519.zip.

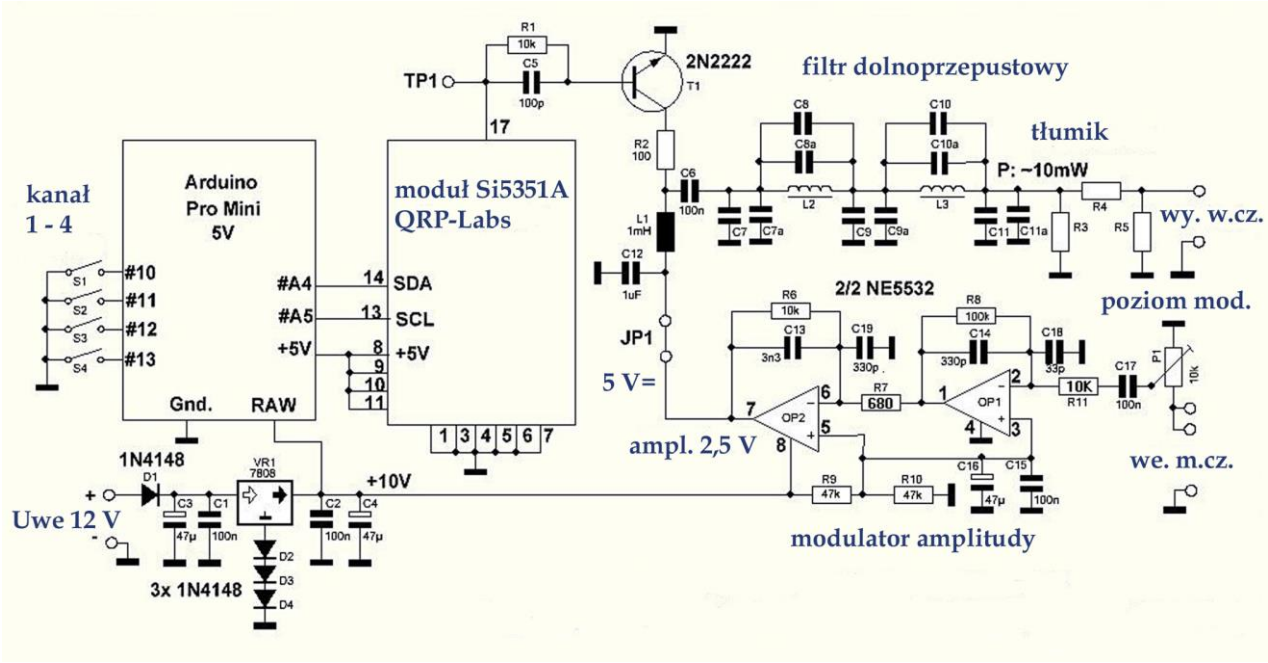


Rys. 3.1.3.1. Schemat ideowy

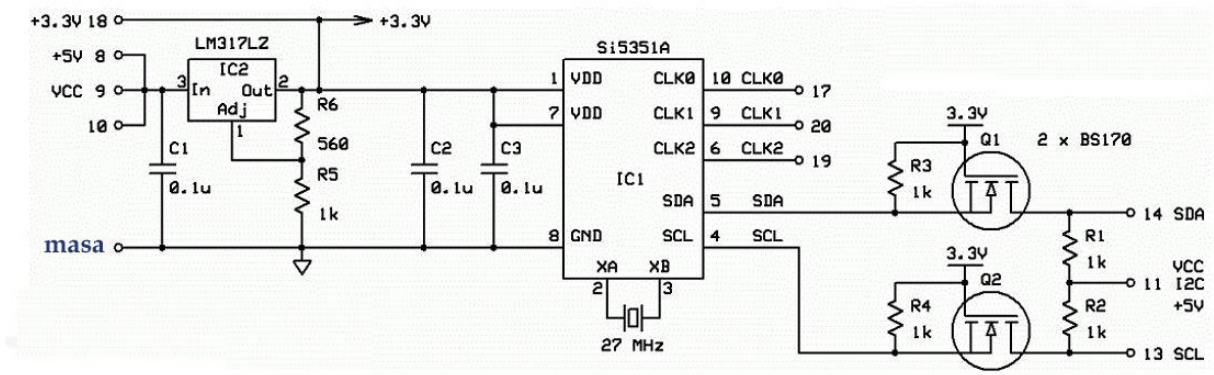
3.2. Generator sygnałowy AM

Układ generatora składa się z syntezera na układzie scalonym Si5351A, sterującego go przez złącze I2C mikrokomputera Arduino Mini Pro w wersji 5 V, modulatora amplitudy na wzmacniaczu operacyjnym NE5532, separującego wzmacniacza w.cz. na tranzystorze 2N2222 (lub dowolnym zbliżonym europejskiego typu) i filtru dolnoprzepustowego (FDP) typu Pi. W zależności od potrzeb generator może pracować na falach długich, średnich lub krótkich 0,15 – 30 MHz (sam syntezer pokrywa zakres 10 kHz – 200 MHz, a więc w razie potrzeby możliwe jest rozszerzenie zakresu pracy do pasma 2 m. W układzie przedstawionym na ilustracji 3.2.1 wyboru jednej z czterech z góry wybranych częstotliwości pracy dokonuje się za pomocą przełączników S1 – S4 połączonych z wejściami 10 – 13 płytki Arduino. Zamiast przełącznika można użyć obrotowego kodera po uzupełnieniu programu o funkcję jego odczytu. Program korzystający z kodera jest dostępny w Internecie pod adresem: *http://www.kh-gps.de/si5351_vfo1.zip*. W układzie wykorzystano gotową płytkę syntezera dostępną w różnych sklepach internetowych i nie tylko. Rozwiązanie stanowi dobry przykład sposobu sterowania częstotliwoś-

cią pracy syntezera i dlatego jest warte zaprezentowania. Układ bez modulatora może być użyty jako generator sterujący w układach nadajników. Po usunięciu układu modulatora amplitudy i dodaniu wzmacnicza mocy w dowolnym układzie na tranzystorze polowym albo w innym rozwiązaniu (np. jednym z opisanych w tym skrypcie albo w tomach 56, 57) Arduino z syntezerem mogą stanowić prosty nadajnik telegraficzny na dowolne pasmo amatorskie.
Do stabilizacji napięcia 10 V zasilającego Arduino i modulator zastosowano scalony stabilizator 8 V, którego wyprowadzenie masy jest połączone z masą układu przez trzy diody D2 – D4. Dioda D1 zabezpiecza przed uszkodzeniem w przypadku odwrotnego podłączenia napięcia zasilania.



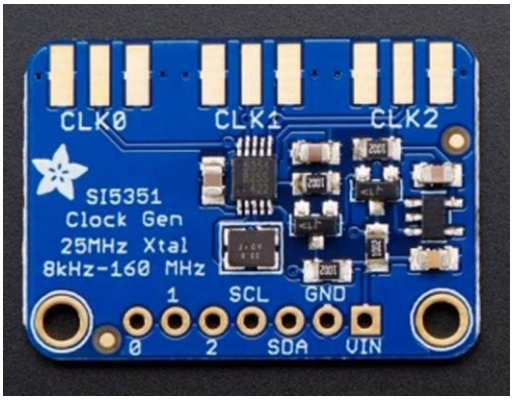
Rys. 3.2.1. Schemat ideowy generatora sygnałowego

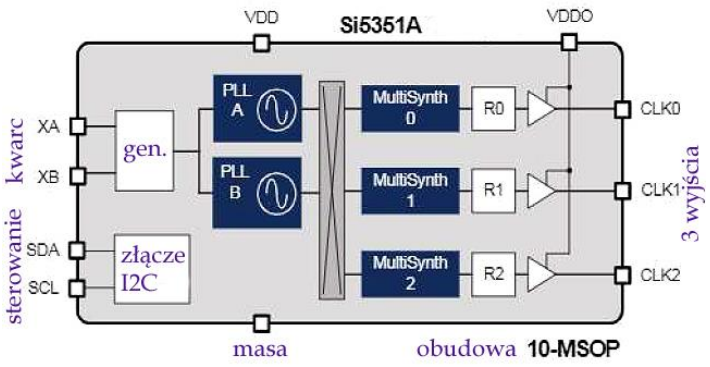


Rys. 3.2.2. Dopasowanie poziomów sygnałów SDA i SCL

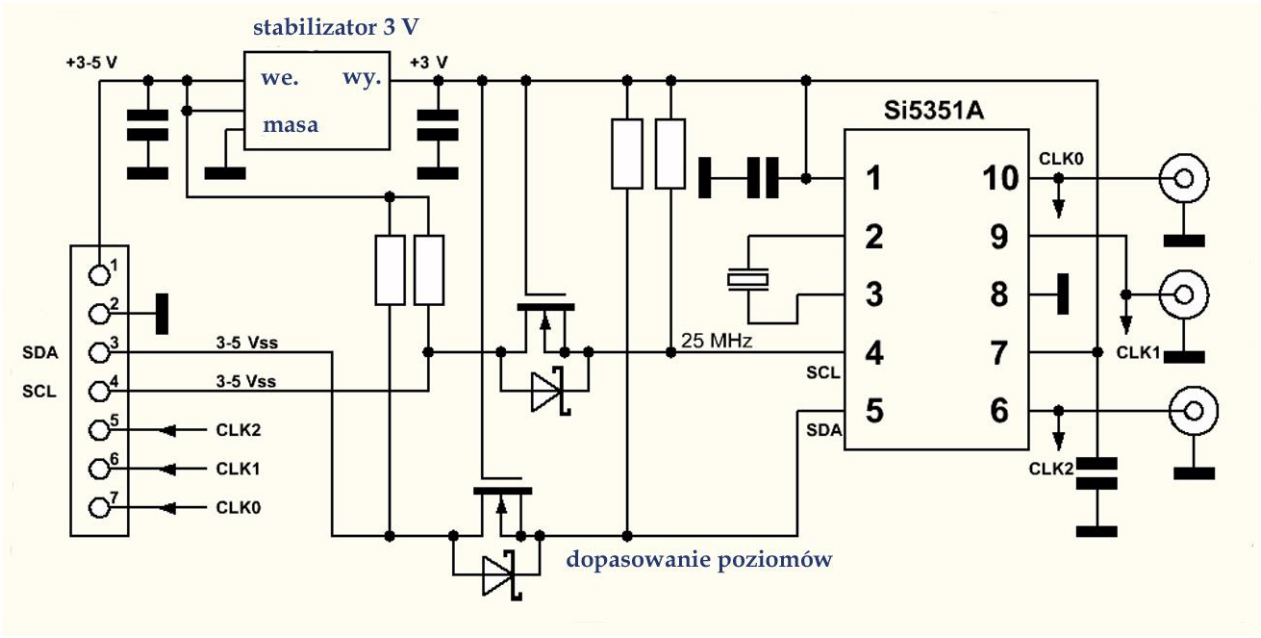
W przypadku zastosowania wersji Arduino na napięcie 3,3, V możliwe jest bezpośrednie połączenie sygnałów z SDA i SCL z Arduino do modułu syntezera, natomiast dla wersji 5 V konieczne jest dopasowanie poziomów jak na schemacie 3.2.2. Standardowym adresem Si5351 na magistrali I2C jest 0x60, ale w razie konfliktów można korzystać z innych adresów alternatywnych.
Zaletą przytoczonego dalej przykładowego programu opracowanego przez OE1CGS (*http://qrp-labs.com/synth/oe1cgs.html*) jest to, że nie korzysta on z żadnych dodatkowych bibliotek. Jako częstotliwości wyjściowe wybrano w nim kolejno dla kanałów S1 – S4: 675 kHz, 972 kHz, 1476 kHz i 1512 kHz. Ich zmiana w programie jest jednak nieskomplikowana. Moc wyjściowa generatora wynosi w przybliżeniu 10 mW, co przy bezpośrednim połączeniu z odbiornikiem daje na jego wejściu napięcie

117 dBµV. Dla jego obniżenia można zastosować tłumik na wyjściu. Dla otrzymania napięcia 84 dBµV (S9 + 50 dB) oporniki R3 i R5 w tłumiku muszą mieć oporności 52,3 Ω, a R4 1100 Ω.





Fot. 3.2.3a. Moduł Si5351A firmy *Adafruit* Rys.3.2.3b. Schemat blokowy Si5351A.
Zawiera on dwa synchronizowane fazowo generatory VCO (PLLA i PLLB) pracujące w zakresie 600 – 900 MHz i programowalne dzielniki pozwalające na uzyskanie ułamkowych stosunków podziału i w ten sposób pożądanych częstotliwości wyjściowych



Rys. 3.2.4. Moduł Si5351A firmy *Adafruit* posiada układ dopasowujący poziomy sygnałów SDA i SCL z 3,3 V do 5 V

Widoczny na fotografii 3.2.3 moduł syntezera Si5351A o wymiarach 22 x 31 mm generuje z częstotliwości zegarowej 25 lub 27 MHz trzy prostokątne sygnały w.cz. o niezależnych częstotliwościach w zakresie do 160 MHz i o mocy około 12 dBm – czyli przekraczającej 10 mW. W sytuacji gdy nie jest on narażony na znaczne różnice temparatur stabilność częstotliwości wystarcza do pracy emisją WSPR i innymi wąskopasmowymi emisjami amatorskimi. Syntezer może być sterowany przez mikrokomputery Arduino albo ESP8266 przy użyciu magistrali I2C. W niektórych zastosowaniach krótkofalarskich przydatna jest też możliwość modulacji z rozpraszaniem widma sygnału. Układy scalone Si5351B i -C generują 8 niezależnych sygnałów w.cz. Są one zasadniczo przeznaczone do generacji sygnałów zegarowych dla systemów mikroprocesorowych, komunikacyjnych i logicznych. Dla poprawienia stabilności częstotliwości można zastosować opisany dalej układ stabilizacji temperatury kwarcu sterującego syntezer.

3.2.1. Kod źródłowy programu OE1CGS

```
// SI5351_Code autorstwa OE1CGS
// zmodyfikowany przez DJ7OO do użycia na falach średnich przy   Fx = 27 MHz
// częstotliwość początkowa : 972 KHz  (możliwy zakres: 7,81 KHz - 200 MHz )
//
// http://www.qrp-labs.com/synth/oe1cgs.html
// http://www.qrp-labs.com/synth.html
#include <Wire.h>
float frq = 972000;
 float frq_old;
//////////////////////////////////////////////////////////////////////
void setup() {
 Serial.begin(9600);
 Serial.println("Start");
 delay(100);
 pinMode(10,INPUT_PULLUP);
 pinMode(11,INPUT_PULLUP);
 pinMode(12,INPUT_PULLUP);
 pinMode(13,INPUT_PULLUP);

 Wire.begin();                    // Inicjalizacja I2C-jako urządzenie nadrzędne
                                  // SDA na nóżce ADC04
                                  // SCL na nóżce ADC05

  if(digitalRead(10)==LOW) { frq = 675000 ; }   // S1: 675 kHz
  else
  if(digitalRead(11)==LOW) { frq = 972000 ; }   // S2: 972 kHz
  else
  if(digitalRead(12)==LOW) { frq = 1476000 ; }  // S3: 1476 kHz
  else
  if(digitalRead(13)==LOW) { frq = 1512000 ; }  // S4: 1512 kHz

  SetFrequency(frq);                    // Podanie częstotliwości TX
  Si5351a_Write_Reg (17, 128);          // Wyłączenie wyjścia CLK1
  Si5351a_Write_Reg (16, 79); // Włączenie wyjścia CLK0, dla PLLA kwarc jako sterujący i tryb całkowity („Integer Mode”)
}

//////////////////////////////////////////////////////////////////////
void loop()
{
  if(frq!=frq_old)
  {
  SetFrequency(frq);                    // Częstotliwość TX
  Si5351a_Write_Reg (17, 128);          // Wyłączenie wyjścia CLK1
  Si5351a_Write_Reg (16, 79); // Włączenie CLK0, dla PLLA kwarc i tryb całkowity („Integer Mode”)
                                        // CLK0 obciążalność = 8mA; poziom mocy := 0dB
  Serial.print("Frequency: ");
  Serial.print(frq/1000,0);
  Serial.println( " kHz");
  frq = frq_old;
  }
}
```

```
//////////////////////////////////////////////////////////////////////
void SetFrequency (unsigned long frequency) { // Częstotliwość w Hz; w zakresie [7810 Hz do 200 MHz]
  #define F_XTAL 27005701;            // Częstotliwość kwarcu sterującego
  #define c 1048574;                  // Część "c" stosunku powielania XTAL -> PLL
  unsigned long fvco;                 // Częstotliwość VCO (600-900 MHz) dla PLL
  unsigned long outdivider;           // Dzielnik wyjściowy w zakresie [4,6,8-900], najlepiej parzysty
  byte R = 1;                         // Dodatkowy dzielnik wyjściowy w zakresie [1,2,4,...128]
  byte a;                            // Część "a" stosunku powielania XTAL -> PLL w zakresie [15,90]
  unsigned long b;                    // Część "b" stosunku powielania XTAL -> PLL
  float f;                            // liczba zmiennoprzecinkowa potrzebna w obliczeniach
  unsigned long MS0_P1;    // Si5351a rejestry dzielnika wyjściowego MS0_P1, P2 i P3 podane poniżej
  unsigned long MSNA_P1;              // Si5351a rejestr MSNA_P1
  unsigned long MSNA_P2;              // Si5351a rejestr MSNA_P2
  unsigned long MSNA_P3;              // Si5351a rejestr MSNA_P3
  outdivider = 900000000 / frequency;  // Dla 900 MHz jako maks. częstotliwości PLL
  while (outdivider > 900){          // Przy przekroczeniu zakresu dzielnika (>900) konieczny dodatkowy
    R = R * 2;
    outdivider = outdivider / 2;
  }
  if (outdivider % 2) outdivider--;    // Obliczanie parzystego stosunku dla pożądanej częstotliwości
  fvco = outdivider * R * frequency;   // Obliczanie częstotliwości PLL przy podziale parzystym
  switch (R){                          // Przeliczanie dzielnika na postać bitowwą dla rejestru 44
    case 1: R = 0; break;              // Bits [6:4] = 000
    case 2: R = 16; break;             // Bits [6:4] = 001
    case 4: R = 32; break;             // Bits [6:4] = 010
    case 8: R = 48; break;             // Bits [6:4] = 011
    case 16: R = 64; break;            // Bits [6:4] = 100
    case 32: R = 80; break;            // Bits [6:4] = 101
    case 64: R = 96; break;            // Bits [6:4] = 110
    case 128: R = 112; break;          // Bits [6:4] = 111
  }
  a = fvco / F_XTAL;                   // Mnożnik z częstotliwości kwarcu na PLL.
  f = fvco - a * F_XTAL;               // Mnożnik = a+b/c
  f = f * c;                          // Obliczenia "int" i "float"
  f = f / F_XTAL;
  b = f;
  MS0_P1 = 128 * outdivider - 512;     // Obliczenia rejestrów MS0_P1 – MS0_P3 dzielnika
                                      // MS0_P2 = 0 i MS0_P3 = 1; wartości niezmienne, patrz niżej
  f = 128 * b / c;                    // Obliczenia rejestrów MSNA_P1 – MSNA_P3
  MSNA_P1 = 128 * a + f - 512;
  MSNA_P2 = f;
  MSNA_P2 = 128 * b - MSNA_P2 * c;
  MSNA_P3 = c;
  Si5351a_Write_Reg (16, 128);                  // Wyłączenie wyjścia w trakcie modyfikacji rejestrów
  Si5351a_Write_Reg (26, (MSNA_P3 & 65280) >> 8);   // Bity [15:8] MSNA_P3 w rejestrze 26
  Si5351a_Write_Reg (27, MSNA_P3 & 255);               // Bity [7:0] MSNA_P3 w rejestrze 27
  Si5351a_Write_Reg (28, (MSNA_P1 & 196608) >> 16); // Bity [17:16] MSNA_P1 w bitach [1:0] rejestru 28
  Si5351a_Write_Reg (29, (MSNA_P1 & 65280) >> 8);   // Bity [15:8] MSNA_P1 w rejestrze 29
  Si5351a_Write_Reg (30, MSNA_P1 & 255);               // Bity [7:0]  MSNA_P1 w rejestrze 30
  Si5351a_Write_Reg (31, ((MSNA_P3 & 983040) >> 12) | ((MSNA_P2 & 983040) >> 16)); // Części MSNA_P3 MSNA_P1
  Si5351a_Write_Reg (32, (MSNA_P2 & 65280) >> 8);   // Bity [15:8] MSNA_P2 w rejestrze 32
  Si5351a_Write_Reg (33, MSNA_P2 & 255);               // Bity [7:0] MSNA_P2 w rejestrze 33
```

```
  Si5351a_Write_Reg (42, 0);                              // Bity [15:8] MS0_P3 (zawsze 0) w rejestrze 42
  Si5351a_Write_Reg (43, 1);                              // Bity [7:0] MS0_P3 (zawsze 1) w rejestrze 43
  Si5351a_Write_Reg (44, ((MS0_P1 & 196608) >> 16) | R);  // Bity [17:16] MS0_P1 w bitach [1:0] i R
w [7:4]
  Si5351a_Write_Reg (45, (MS0_P1 & 65280) >> 8);    // Bity [15:8] MS0_P1 w rejestrze 45
  Si5351a_Write_Reg (46, MS0_P1 & 255);             // Bity [7:0]  MS0_P1 w rejestrze 46
  Si5351a_Write_Reg (47, 0);                    // Bity [19:16] MS0_P2 i MS0_P3 zawsze 0
  Si5351a_Write_Reg (48, 0);                    // Bity [15:8] MS0_P2 zawsze 0
  Si5351a_Write_Reg (49, 0);                    // Bity [7:0] MS0_P2 zawsze 0
  if (outdivider == 4){
    Si5351a_Write_Reg (44, 12 | R);               // Specjalne ustawienie dla R = 4 (patrz katalog)
    Si5351a_Write_Reg (45, 0);                    // Bity [15:8] MS0_P1 zawsze 0
    Si5351a_Write_Reg (46, 0);                    // Bity [7:0] MS0_P1 zawsze 0
  }
  Si5351a_Write_Reg (177, 32);                  // Zerowanie PLL A
}

//////////////////////////////////////////////////////////////////////
void Si5351a_Write_Reg (byte regist, byte value){   // Wpisanie "byte" do "regist" Si5351a przez I2C
  Wire.beginTransmission(96);                    // Rozpoczyna transmisję do podległego 96, jest to
                                                 // adres I2C Si5351a (patrz katalog Si5351a)
  Wire.write(regist);                            // Wpisanie bajtu z liczba do rejestru
  Wire.write(value);                          // Wpisanie bajtu zawierającego wartość do wpisania do rejestru
  Wire.endTransmission();                        // Nadanie danych i koniec transmisji
}
```

3.2.2. Płynne strojenie syntezera

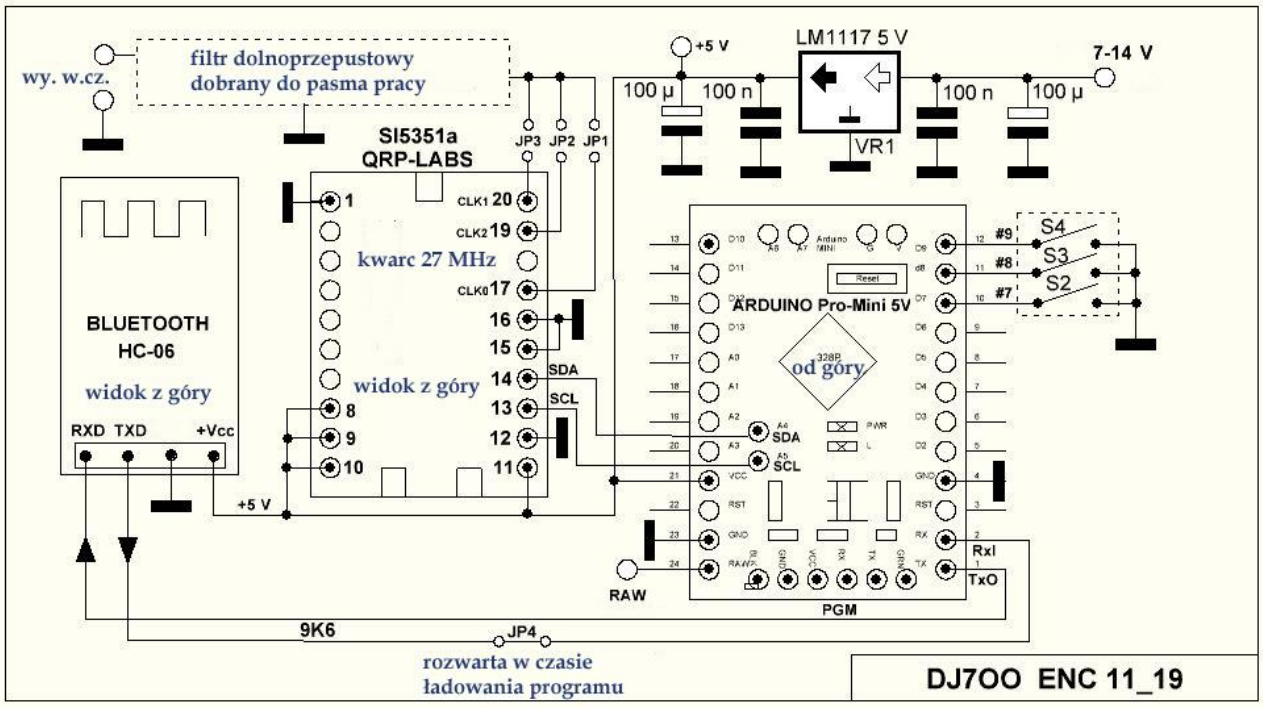
Poniższy przykładowy program ilustruje możliwość płynnego przestrajania generatora opartego na syntezerze Si5351A. Jako element strojeniowy służy potencjometr podłączony do analogowego wejścia A0. Odczytanym wartościom napięcia w zakresie 0 – 1023 przypisane są przez funkcję map() należącą do standardowej biblioteki matematycznej częstotliwości z zakresu 7000000 – 7040000 Hz. Program korzysta z biblioteki 5351.

```
// https://github.com/etherkit/Si5351Arduino
// http://nt7s.com/2014/06/Si5351-libraries-and-breakout-board/
#include „si5351.h“
#define PIN_POTI A0
Si5351 si5351;
void setup()
{ si5351.init(SI5351_CRYSTAL_LOAD_10PF);
  si5351.set_pll(SI5351_PLL_FIXED, SI5351_PLLA);    // 900 MHz
  si5351.clock_enable(SI5351_CLK0, 1);                    // włączenie CLK0
}
void loop ()
{                                                          // odczyt częstotliwości z potencjometru
  long frequenz = map (analogRead(PIN_POTI), 0, 1023, 7000000, 7040000);
  // nastawienie na syntezerze
  si5351.set_freq(frequenz, SI%#%!_PLL_FIXED, SI5351_CLK0);
}
```

3.3. Emisje cyfrowe

3.3.1. Dalekopis Hella

W rozwiązaniu ze schematu 3.3.1.1 Arduino mini steruje scalonym syntezerem Si5351A generującym sygnał w.cz. w wybranym w programie paśmie amatorskim (*http://www.kh-gps.de/feld-hell.htm*). W przykładowym programie jest to pasmo 20 m. Moduł Bluetooth HC-06 pozwala na przekazywanie z PC dowolnych tekstów przeznaczonych do nadania, a oprócz tego możliwy jest wybór jednego z dwóch stałych tekstów.
W przypadku rozwarcia przełączników S3 i S4 nadawana jest niemodulowana nośna, zwarcie S3 do masy powoduje nadawanie tekstu nr 1, zwarcie S4 do masy – tekstu nr 2, a zwarcie do masy obu – tekstu otrzymywanego przez złącze Bluetooth i doprowadzonego do wejścia szeregowego Arduino. Tekst pochodzący z komputera musi zawierać tylko duże litery i cyfry, ale nie może zawierać małych.
W przykładowym programie przełącznik S2 powoduje przełączanie między pasmami 20 m i 2 m.
W czasie ładowania progamu do Arduino należy odłączyć od niego moduł Bluetooth usuwając zworkę JP4. Użyta w programie biblioteka Si5351 pozwala na korzystanie z modułów sterowanych nie tylko standardowym kwarcem 25 MHz, ale także kwarcem 27 MHz.



Rys. 3.3.1.1. Schemat ideowy (źródło *www.kh-gps.de*)

Rezygnacja z doprowadzania tekstów przez złącze Bluetooth lub doprowadzanie ich w inny sposób pozwala na zrezygnowanie z modułu HC-06.
Ze względu na inny sposób obsługi syntezera przytaczamy poniżej przykładowy program. Biblioteka Si5351 jest dostępna w internecie. Syntezer Si5351 dostarcza sygnału prostokątnego i dlatego na jego wyjściu konieczny jest filtr dolnoprzepustowy.

3.3.1.1. Kod źródłowy programu

```
// wersja 2m/20m
// http://appnotes.etherkit.com/2015/08/arduino-feld-hell-beacon-on-the-si5351a-breakout-board/?print=print
//
// Prosta radiolatarnia w normie Feld Hell dla Arduino
// ********************************************************************
```

```
// * używa modułu Si5351a QRP-Labs (kwarc 27 MHz)                    *
// * korzysta z nowej wersji biblioteki Jasona Milldruma NT7S        *
// * Częstotliwości wyjściowe do pasm UKF                            *
// * 4 tryby pracy do wyboru                                         *
// * 1: nośna          2: tekst 1: np.: "TEST DE DJ7OO"              *
// * 3: tekst 2: np. "CQ DE DJ7OO" 4: dowolny z wejścia szeregowego  *
// * rozszerzony i zmodyfikowany przez DJ7OO 10/2019                  *
// ********************************************************************
//            Ważne! W tekstach dozwolone tylko duże litery
//
// Kod źródłowy Feld Hella dla Arduino autorstwa Marka Vandewetteringa K6HX,
// przystosowany do Si5351A przez Roberta Liesenfelda AK6L <ak6l@ak6l.org>.
// Ustawienie czasomierza aurorstwa Thomasa Knutsena LA3PNA.
//
// Copyright (c) 2015 Robert Liesenfeld AK6L <ak6l@ak6l.org>
//
// Dozwolone bezpłatne rozpowszechnianie i modyfikowanie
// pod warunkiem nie usuwania zastrzeżeń autiorskich i tekstu
// informującego o prawach do rozpowszechniania
//
// THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
// EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF
// MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT.
// IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR
// ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF
// CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION
// WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
//

#include <si5351.h>
#include "Wire.h"

// Definicje zmiennych.
Si5351 si5351;

uint64_t freq = 144495000;       // częstotliwość pracy 144,495 MHz
uint64_t freq2 = 14074000;       // częstotliwość pracy 14,074 MHz
uint64_t cal = 241000;           // poprawka
char * packet_buffer  = "                                                \n ";
//char floatbuf[32];
//char gradbuf[4];
String inData = "";
int ledPin = 13;

// Konieczny modyfikator 'volatile' aby uniknąć optymalizacji
// mogących kolidować z przerwaniami.
volatile bool proceed = false;

// Definicja struktury glyph
typedef struct glyph {
   char ch ;
   word col[7] ;
} Glyph ;
```

```
// Zestaw czcionek Feld Hella.  Tabela umieszczona w PROGMEM
// dla oszczędności pamięci roboczej
const Glyph glyphtab[] PROGMEM = {
  {' ', {0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000}},
  {'A', {0x07fc, 0x0e60, 0x0c60, 0x0e60, 0x07fc, 0x0000, 0x0000}},
  {'B', {0x0c0c, 0x0ffc, 0x0ccc, 0x0ccc, 0x0738, 0x0000, 0x0000}},
  {'C', {0x0ffc, 0x0c0c, 0x0c0c, 0x0c0c, 0x0c0c, 0x0000, 0x0000}},
  {'D', {0x0c0c, 0x0ffc, 0x0c0c, 0x0c0c, 0x07f8, 0x0000, 0x0000}},
  {'E', {0x0ffc, 0x0ccc, 0x0ccc, 0x0c0c, 0x0c0c, 0x0000, 0x0000}},
  {'F', {0x0ffc, 0x0cc0, 0x0cc0, 0x0c00, 0x0c00, 0x0000, 0x0000}},
  {'G', {0x0ffc, 0x0c0c, 0x0c0c, 0x0ccc, 0x0cfc, 0x0000, 0x0000}},
  {'H', {0x0ffc, 0x00c0, 0x00c0, 0x00c0, 0x0ffc, 0x0000, 0x0000}},
  {'I', {0x0ffc, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000}},
  {'J', {0x003c, 0x000c, 0x000c, 0x000c, 0x0ffc, 0x0000, 0x0000}},
  {'K', {0x0ffc, 0x00c0, 0x00e0, 0x0330, 0x0e1c, 0x0000, 0x0000}},
  {'L', {0x0ffc, 0x000c, 0x000c, 0x000c, 0x000c, 0x0000, 0x0000}},
  {'M', {0x0ffc, 0x0600, 0x0300, 0x0600, 0x0ffc, 0x0000, 0x0000}},
  {'N', {0x0ffc, 0x0700, 0x01c0, 0x0070, 0x0ffc, 0x0000, 0x0000}},
  {'O', {0x0ffc, 0x0c0c, 0x0c0c, 0x0c0c, 0x0ffc, 0x0000, 0x0000}},
  {'P', {0x0c0c, 0x0ffc, 0x0ccc, 0x0cc0, 0x0780, 0x0000, 0x0000}},
  {'Q', {0x0ffc, 0x0c0c, 0x0c3c, 0x0ffc, 0x000f, 0x0000, 0x0000}},
  {'R', {0x0ffc, 0x0cc0, 0x0cc0, 0x0cf0, 0x079c, 0x0000, 0x0000}},
  {'S', {0x078c, 0x0ccc, 0x0ccc, 0x0ccc, 0x0c78, 0x0000, 0x0000}},
  {'T', {0x0c00, 0x0c00, 0x0ffc, 0x0c00, 0x0c00, 0x0000, 0x0000}},
  {'U', {0x0ff8, 0x000c, 0x000c, 0x000c, 0x0ff8, 0x0000, 0x0000}},
  {'V', {0x0ffc, 0x0038, 0x00e0, 0x0380, 0x0e00, 0x0000, 0x0000}},
  {'W', {0x0ff8, 0x000c, 0x00f8, 0x000c, 0x0ff8, 0x0000, 0x0000}},
  {'X', {0x0e1c, 0x0330, 0x01e0, 0x0330, 0x0e1c, 0x0000, 0x0000}},
  {'Y', {0x0e00, 0x0380, 0x00fc, 0x0380, 0x0e00, 0x0000, 0x0000}},
  {'Z', {0x0c1c, 0x0c7c, 0x0ccc, 0x0f8c, 0x0e0c, 0x0000, 0x0000}},
  {'0', {0x07f8, 0x0c0c, 0x0c0c, 0x0c0c, 0x07f8, 0x0000, 0x0000}},
  {'1', {0x0300, 0x0600, 0x0ffc, 0x0000, 0x0000, 0x0000, 0x0000}},
  {'2', {0x061c, 0x0c3c, 0x0ccc, 0x078c, 0x000c, 0x0000, 0x0000}},
  {'3', {0x0006, 0x1806, 0x198c, 0x1f98, 0x00f0, 0x0000, 0x0000}},
  {'4', {0x1fe0, 0x0060, 0x0060, 0x0ffc, 0x0060, 0x0000, 0x0000}},
  {'5', {0x000c, 0x000c, 0x1f8c, 0x1998, 0x18f0, 0x0000, 0x0000}},
  {'6', {0x07fc, 0x0c66, 0x18c6, 0x00c6, 0x007c, 0x0000, 0x0000}},
  {'7', {0x181c, 0x1870, 0x19c0, 0x1f00, 0x1c00, 0x0000, 0x0000}},
  {'8', {0x0f3c, 0x19e6, 0x18c6, 0x19e6, 0x0f3c, 0x0000, 0x0000}},
  {'9', {0x0f80, 0x18c6, 0x18cc, 0x1818, 0x0ff0, 0x0000, 0x0000}},
  {'*', {0x018c, 0x0198, 0x0ff0, 0x0198, 0x018c, 0x0000, 0x0000}},
  {'.', {0x001c, 0x001c, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000}},
  {'?', {0x1800, 0x1800, 0x19ce, 0x1f00, 0x0000, 0x0000, 0x0000}},
  {'!', {0x1f9c, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000}},
  {'(', {0x01e0, 0x0738, 0x1c0e, 0x0000, 0x0000, 0x0000, 0x0000}},
  {')', {0x1c0e, 0x0738, 0x01e0, 0x0000, 0x0000, 0x0000, 0x0000}},
  {'#', {0x0330, 0x0ffc, 0x0330, 0x0ffc, 0x0330, 0x0000, 0x0000}},
  {'$', {0x078c, 0x0ccc, 0x1ffe, 0x0ccc, 0x0c78, 0x0000, 0x0000}},
  {'/', {0x001c, 0x0070, 0x01c0, 0x0700, 0x1c00, 0x0000, 0x0000}},
} ;

// Definicja wielkości tabeli glyphs.  Pozwala na dodawanie znaków
// bez modyfikacji je długości w programie
#define NGLYPHS        (sizeof(glyphtab)/sizeof(glyphtab[0]))
```

```
// Wektor przerwania czasowego. Przełącza zmieną kolumn dla zapewnienia ich dokladnego początku.
// Przerwanie wywołane po osiągnięciu przez „Timer 1” zadanego w setup() stanu
ISR(TIMER1_COMPA_vect) {
   proceed = true;
}

////////////////////////////////////////////////////////////
// Główna część programu radiolatarni. Dla danego znaku znajduje kod
// z tabeli glyph i kluczuje wyjście Si5351A w takt sygnału Feld Hella
void encodechar(int ch) {
   int i, x, y, fch;
   word fbits;

   for (i=0; i<NGLYPHS; i++) {
      // Poszukiwanie znaku wśród elementów tabeli glyphtab.
      fch = pgm_read_byte(&glyphtab[i].ch);
      if (fch == ch) {
         // Po znalezieniu znaku pobieranie kolumny po kolumnie
         for (x=0; x<7; x++) {
            fbits = pgm_read_word(&(glyphtab[i].col[x]));
            // kluczowanie w takt elementów znaku; 7 kolumn x 14 linii.
            for (y=0; y<14; y++) {
               if (fbits & (1<<y)) {
                 si5351.output_enable(SI5351_CLK0, 1);
                 digitalWrite(ledPin, HIGH);
               } else {
                 si5351.output_enable(SI5351_CLK0, 0);
                 digitalWrite(ledPin, LOW);
               }

               while(!proceed)         // Oczekiwanie na przerwanie czasowe przed dalszym krokiem.
               noInterrupts();         // Wyłączenie przerwań, jest zalecane...
               proceed = false;        // zerowanie sygnalizatora dla następnego znaku...
               interrupts();           // ponowne włączenie przerwań.
            }
         }
         break; // Opuszczenie pętli po zakończeniu znaku
      }
   }
}

////////////////////////////////////////////////////////////////
// Pętla pobierająca znaki tekstu.
void encode(char *str) {
   while (*str != '\0')
      encodechar(*str++) ;
}

////////////////////////////////////////////////////////////////
void setup() {
   Serial.begin(9600);
   Serial.println("Start");
   pinMode(7, INPUT_PULLUP);
   pinMode(8, INPUT_PULLUP);
   pinMode(9, INPUT_PULLUP);
```

```
// Dioda na płytce Arduino miga w takt kluczowania.
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);
// Initialize the Si5351
  si5351.init(SI5351_CRYSTAL_LOAD_8PF,27000000,cal);
// si5351.init(SI5351_CRYSTAL_LOAD_8PF,0,cal); // wersja dla kwarców 25 MHz

  if(digitalRead(7)==HIGH)
 {
 // Ustawienie CLK0 na 144,495 MHz z poprawką;
 // Właściwe dobranie poprawki zmniejsza nachylenie liter
   si5351.set_freq(freq * 100, SI5351_CLK0);
 }
 else if(digitalRead(7)==LOW)
 {
 // Ustawienie CLK0 na 14,074 MHz z poprawką;
 // Właściwe dobranie poprawki zmniejsza nachylenie liter
   si5351.set_freq(freq2 * 100, SI5351_CLK0);
 }

 // si5351.drive_strength(SI5351_CLK0, SI5351_DRIVE_8MA);//Nastaw. maks. mocy gdy pożądane
   si5351.output_enable(SI5351_CLK0,0); // Początkowe wyłączenie wyjścia

 // Nastawienie licznika Timer1 na częstotliwość przerwań 245 Hz.
   noInterrupts();             // Wyłączenie przerwań.
   TCCR1A = 0;                 // Ustawienie całego rejestru TCCR1A na 0; wyłącza
                       //   wyjścia przerwań, nastawienie normalnego trybu fali
                       //   Czasomierz Timer1 ustawiony jako licznik.
   TCNT1  = 0;                   // Nadanie początkowej wartości 0.
   TCCR1B = (1<<CS10) |     // Ustawienie bitu CS10; bez dzielnika wstępnego
      (1 << WGM12);            //   dla zegara systemowego.
   TIMSK1 = (1 << OCIE1A);  // Włączenie przerwania porównującego stan licznika.
   OCR1A = 0xFF1A;           // Włączenie wyzwalania przerwania;
                       //   zegar 16MHz / stan 0xFF1A == 245,00045938 Hz
   interrupts();           // Ponowne włączenie przerwań.
}

//////////////////////////////////////////////////////////////////////////
void loop() {
  // ************ Tryb 1   transmisja dowolnego tekstu **************
  if(digitalRead(8)==LOW&digitalRead(9)==LOW)
  {
   while (Serial.available() > 0)
   {
   char recieved = Serial.read();
   inData += recieved;
   if (recieved == '\n')
   {

 // Konwersja łańcucha inData na char_array
    int str_len = inData.length() + 1;
    char char_array[str_len];
    inData.toCharArray(char_array, str_len);
    Serial.print("RCVD: ");
```

```
    Serial.print(char_array);

 // send hell data
    packet_buffer = char_array;
    encode(packet_buffer);
    inData = "";
    packet_buffer = "";
   }
  }
}

   // ************ Tryb 2   tekst CQ **************
   else if(digitalRead(8)==LOW&digitalRead(9)==HIGH)
   {
   encode("CQ DE DJ7OO JN49CX");
   Serial.println("CQ DE DJ7OO JN49CX");
 // Odstęp 5 sekund między transmisjami.
   delay(5000);
   }

   // ********** Tryb 3   drugi tekst **************
   else if(digitalRead(8)==HIGH&digitalRead(9)==LOW)
   {
   encode("TEST DE DJ7OO JN49CX");
   Serial.println("TEST DE DJ7OO JN49CX");
 // Odstęp 5 sekund między transmisjami.
   delay(5000);
   }

   // ********** Tryb 4   niemodulowana nośna  ***************
   else if(digitalRead(8)==HIGH&digitalRead(9)==HIGH)
   {
   si5351.output_enable(SI5351_CLK0, 1);
   digitalWrite(ledPin, HIGH);
   }
}
```

3.3.2. WSPR

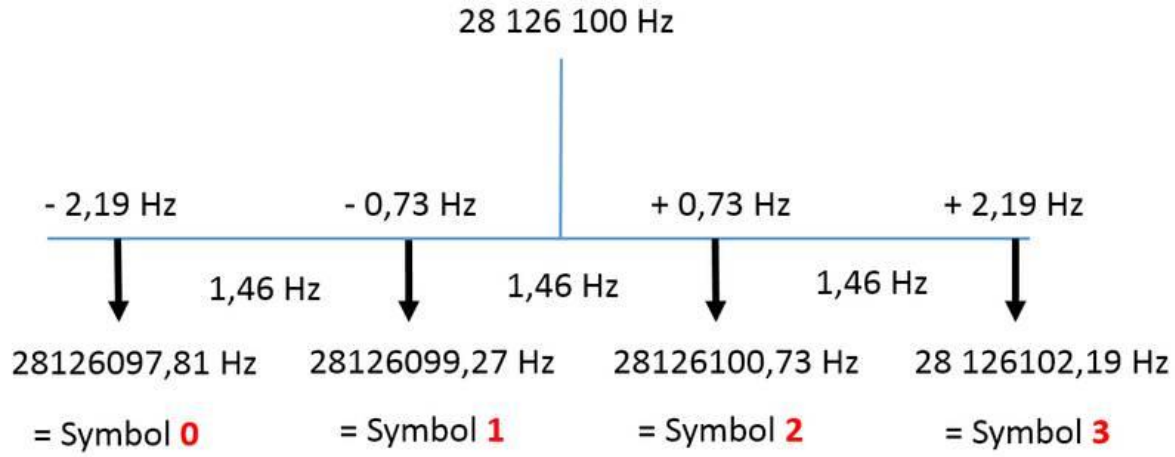
Standardowy komunikat WSPR zawiera znak wywoławczy stacji, czteropozycyjny kwadrat lokatora i moc nadajnika w dBm. Może on więc wyglądać następująco:
OE1KDA JN88 37
albo
DL0RI JO40 23.
Informacja jest nadawana w postacji 162 symboli z czteroelementowego zbioru o indeksach 0, 1, 2, 3 jako czteroelementowy sygnał kluczowany częstotliwościowo 4FSK. Szerokość pasma sygnału wynosi w przybliżeniu 6 Hz, a odstępy między poszczególnymi częstotliwościami 1,46 Hz. Czas nadawania każdego z symboli wynosi 682 ms, co daje całkowity czas transmisji równy 162 * 0,683 = 110,6 s.
Algorytm kodowania wymagałby osobnego omówienia, ale na szczęście w Internecie można znaleźć programy kodujące zadane teksty odpowiadające formatem opisanemu wzorcowi (np. *WSPR encoder.exe* w witrynie *https://github.com/f4goh/WSPR*). W wyniku ich działania otrzymuje się tabelę indeksów częstotliwości, jak w przykładzie z rys. 3.3.2.1.

```
Channel symbols:
      1 3 2 2 2 2 0 2 1 0 0 2 3 3 1 2 2 2 3 0 2 3 2 1 1 1 3 0 0 2
      2 2 2 2 1 0 2 3 0 3 2 0 2 2 0 2 1 0 1 1 0 2 1 3 2 1 0 2 0 3
      3 0 1 2 2 0 2 3 1 0 1 2 1 0 3 0 3 2 0 3 2 2 1 2 1 3 2 2 2 1
      1 0 1 2 3 0 2 2 3 0 2 2 0 2 3 2 0 1 0 0 1 3 3 0 3 3 0 2 1 1
      2 3 0 0 0 3 1 3 0 0 2 0 0 3 2 1 2 0 1 1 2 0 2 0 0 2 0 3 3 0
      3 2 3 1 2 2 2 1 1 0 0 2

Decoded message: DL0RI JO40 23
```

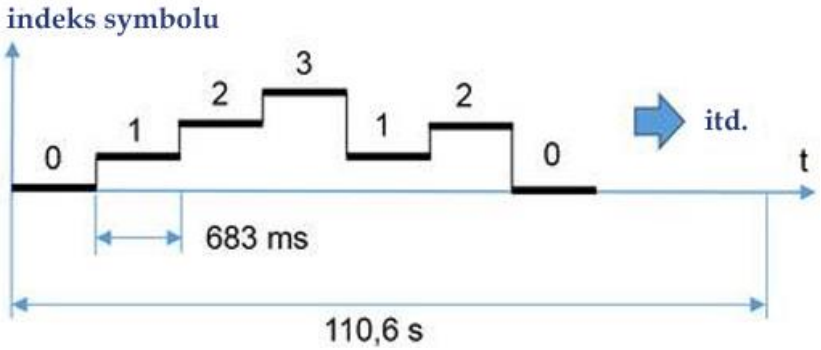
Rys. 3.3.2.1. Przykładowy ciąg symboli zakodowanego tekstu

Na następnej ilustracji przedstawione jest przykładowe widmo transmisji w paśmie 10 m, wokół częstotliwości 28,126100 MHz.

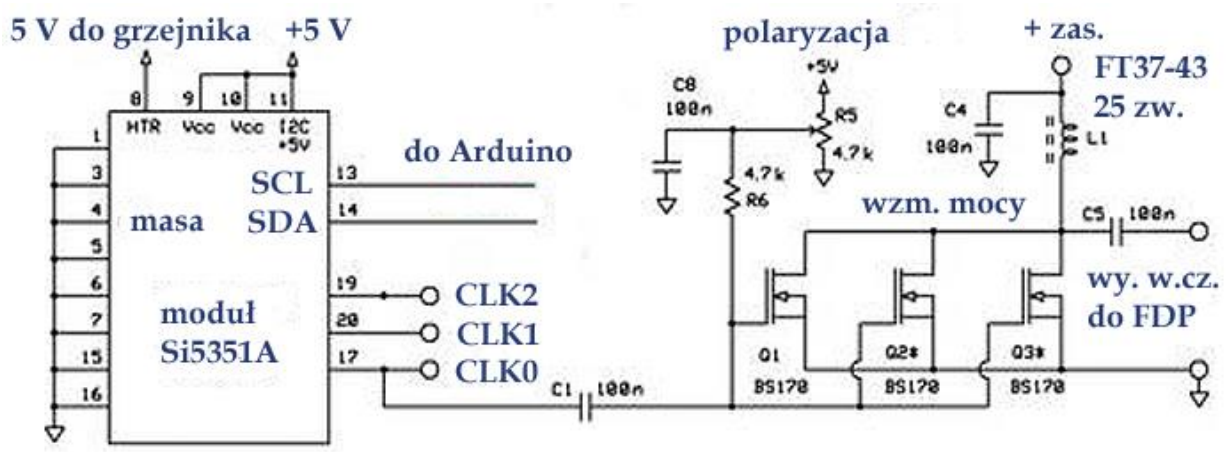


Rys. 3.3.2.2. Przykładowe widmo symboli

Odpowiadający ciągowi symboli ciąg częstotliwości w paśmie nadawania można uzyskać przez odpowiednie sterowanie częstotliwości (kluczowanie) syntezera Si5351A lub dowolnego innego. Możliwe jest też generowanie sygnału m.cz. podawanego następnie na wejście modulatora nadajnika SSB. Syntezer Si5351 jest sterowany przez złącze I2C. W warunkach amatorskich najprościej zapewnić to korzystając z mikrokomputerów Arduino. Dla jego odciążenia do programu ładowana jest gotowa tabela indeksów odpowiadająca nadawanemu tekstowi.



Rys. 3.3.2.3. Przykładowy ciąg symboli w funkcji czasu



Rys. 3.3.2.4. Schemat toru w.cz. z modułem Si5351A w TCXO

Najważniejsze fragmenty programu sterującego:
1) Tabela symboli zawarta w programie:
byte symbols[162] = {1,3,2,2,2,2,0,2,..... };
2) Tabela częstotliwości dla pasma 20 MHz:
unsigned long long frequency[4] = {f0,f1,f2,f3};
gdzie:
f0 = 28126097,81 Hz
f1 = 28126099,27 Hz
f2 = 28126100,73 Hz
f3 = 28126102,19 Hz.
Do sterowania syntezerem wykorzystano w tym przykładzie bibliotekę *Etherkit Si5351 Library*, którą można pobrać z witryny https://github.com/NT7S/Si5351.
3) Funkcja zmiany częstotliwości wyjściowej
si5351.set_freq(frequency[val]);
4) Pętla transmisji komunikatu WSPR:

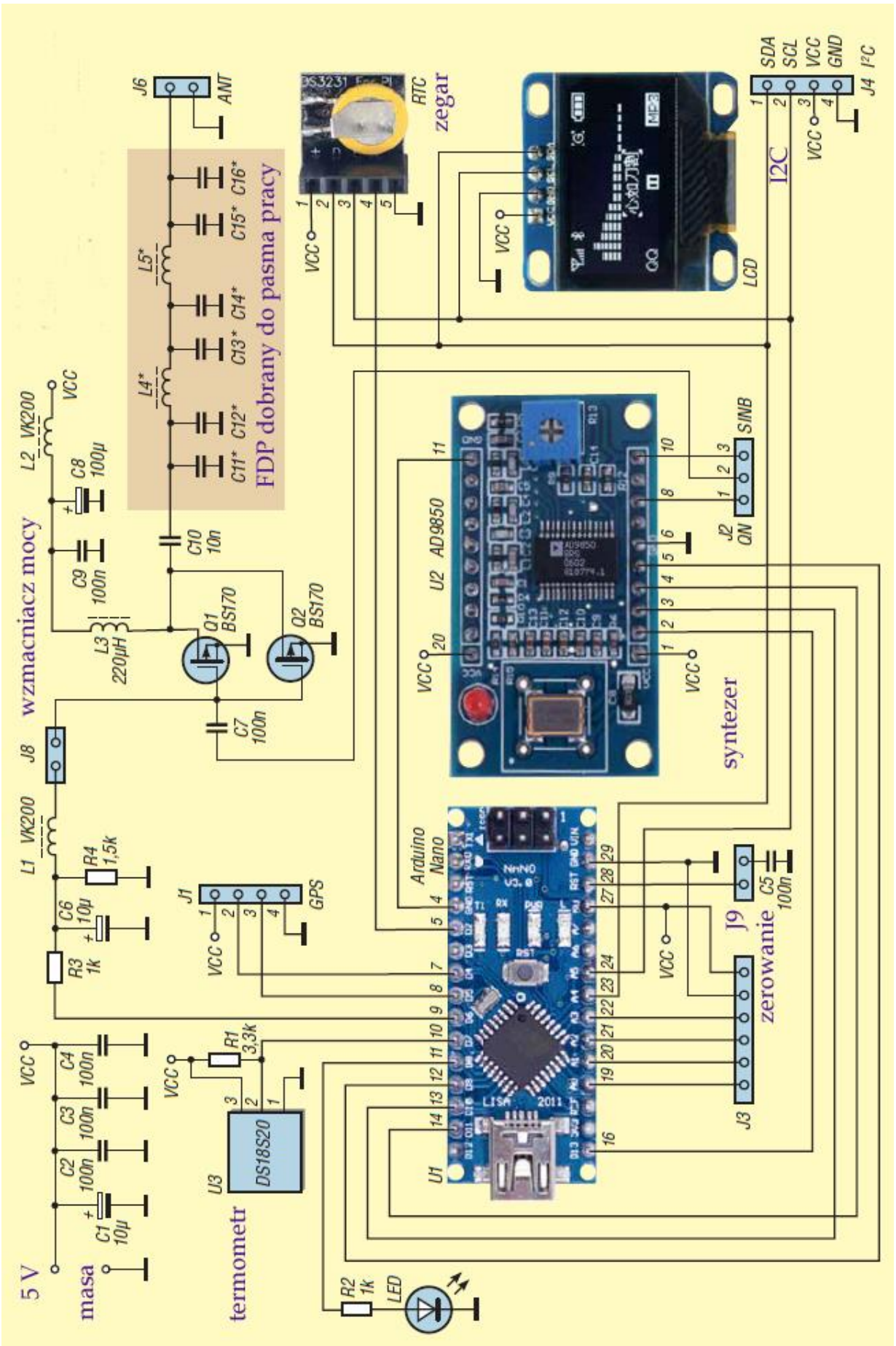
```
 for (byte i=0; i< 162; i++){
                si5351.set_freq(frequency[symbols[i]]);
                delay(683);
        }
```

Całość programu jest dostępna w Internecie pod adresem:
http://www.avsk.net/fileadmin/redakteure/Dateien/arduino_wspr_10m_1.0.ino. Do praktycznego wykorzystania najlepiej jest pobrać kod z witryny, ale dla ułatwienia analizy i nauki własnej kod przytoczono w dodatku B.
Transmisje WSPR wymagają dokładnej synchronizacji czasu. Zaczynają się one z początkiem parzystej minuty. W najprostszym przypadku można je rozpoczynać naciskając przycisk startowy we właściwym momencie opierając się na dokładnie nastawionym zegarku. Bardziej komfortowe są rozwiązania korzystające z czasu GPS albo z internetowych serwerów czasu.
Nadajnik WSPR konstrukcji F4GOH (KB1GOH) przedstawiony w numerze 8/2020 *Funkamateura* składa się z mikrokomputera Arduino Nano, cyfrowego syntezera częstotliwości AD9850, zegara DS3231 połączonego z magistralą I2C i wzmacniacza mocy w.cz. na dwóch tranzystorach BS170. Konstruktor przewidział także możliwość dodania czujnika temperatury DS18S20 i podłączenia odbiornika GPS. W miejscach o niedostatecznej sile sygnału GPS nadajnik korzysta z wbudowanego zegara. Termometr elektroniczny umożliwia używanie nadajnika jako prostej stacji meteorologicznej w sieci APRS. Wyświetlacz nie jest konieczny gdyż służy tylko do wyświetlania bieżącego czasu.
Syntezer częstotliwości dostarcza sygnałów sinusoidalnych na wyjściach SINA i SINB (J2, kontakty 9, 10) i prostokątnych na wyjściach QP i QN (J2, kontakty 7, 8). W przypadku użycia dwóch połączonych równolegle tranzystorów BS170 napięcie sterujące jest pobierane z wyjścia QN. Elementy R3, C6, R4, L1 są zbędne (zwora J8 jest otwarta). Kondensator C7 należy zastąpić przez zwarcie. Moc wyjściowa wynosi wówczas 100 mW (20 dBm).

Przy zastosowaniu we wzmacniaczu mocy tranzystorów złączowych (bipolarnych) sygnał sterujący jest pobierany z wyjścia SINB i podawany przez filtr dolnoprzepustowy R3, C6, R4, L1 i zworę J8 na bazy tranzystorów. Konieczny jest wówczas także kondensator C7. Na wyjściu wzmacniacza mocy musi być włączony filtr dolnoprzepustowy tłumiący harmoniczne. Jego elementy są dobierane w zależności od pasma pracy.
W czasie ładowania programu do Arduino neleży rozewrzeć zworę J2. Kod źródłowy programu *wsprSimple.ino* wraz z niezbędnymi bibliotekami i sterownikami jest dostępny pod adresem *www.github.com/f4goh/WSPR*.
Do zakodowania komunikatu WSPR można skorzystać z programu *WSPRcode.exe* dla PC. Przed jego wywołaniem należy w pliku wywoławczym *wspr.bat* podać swoje dane. W odpowiedzi otrzymuje się plik *symbols.txt* zawierający 162 symbole komunikatu. Dane te należy wprowadzić do tabeli *wsprSymb[]* w programie *wsprSimple.ino*, ale najpierw konieczne jest zastąpienie znaków odstępu przez przecinki aby otrzymać zawartość tabeli zgodną ze składnią języka.
Po pierwszym uruchomieniu programu konieczne jesr nastawienia zegara DS3231. Wymaga to otwarcia okna terminalowego w środowisku programistycznym Arduino, nastawienia szybkości transmisji 115200 bit/s i wpisania polecenia *h*. Następnie należy podać po kolei dane czasowe oddzielone przecinkami.
Przed włożeniem zworki J2 należy potencjometr na płytce syntezera ustawić tak, aby na wyjściu QN pojawił się sygnał prostokątny. Po zakończeniu transmisji panuje na nim napięcie 0 V.
Dla sprawdzenia prawidłowości transmisji można odebrać sygnał za pomocą odbiornika kontrolnego i obejrzeć w oknie WSJT-X. Do kalibracji częstotliwości służy parametr *factor* w *wsprSimple.ino*. Domyślną wartością jest 1500, a w celu dostrojenia częstotliwości nadawania do podzakresu WSPR można jego wartość zmieniać co 200 w górę lub w dół zależnie od sytuacji.

Tabela 3.3.2.1
Elementy filtru dolnoprzepustowego dla pasma 7 MHz

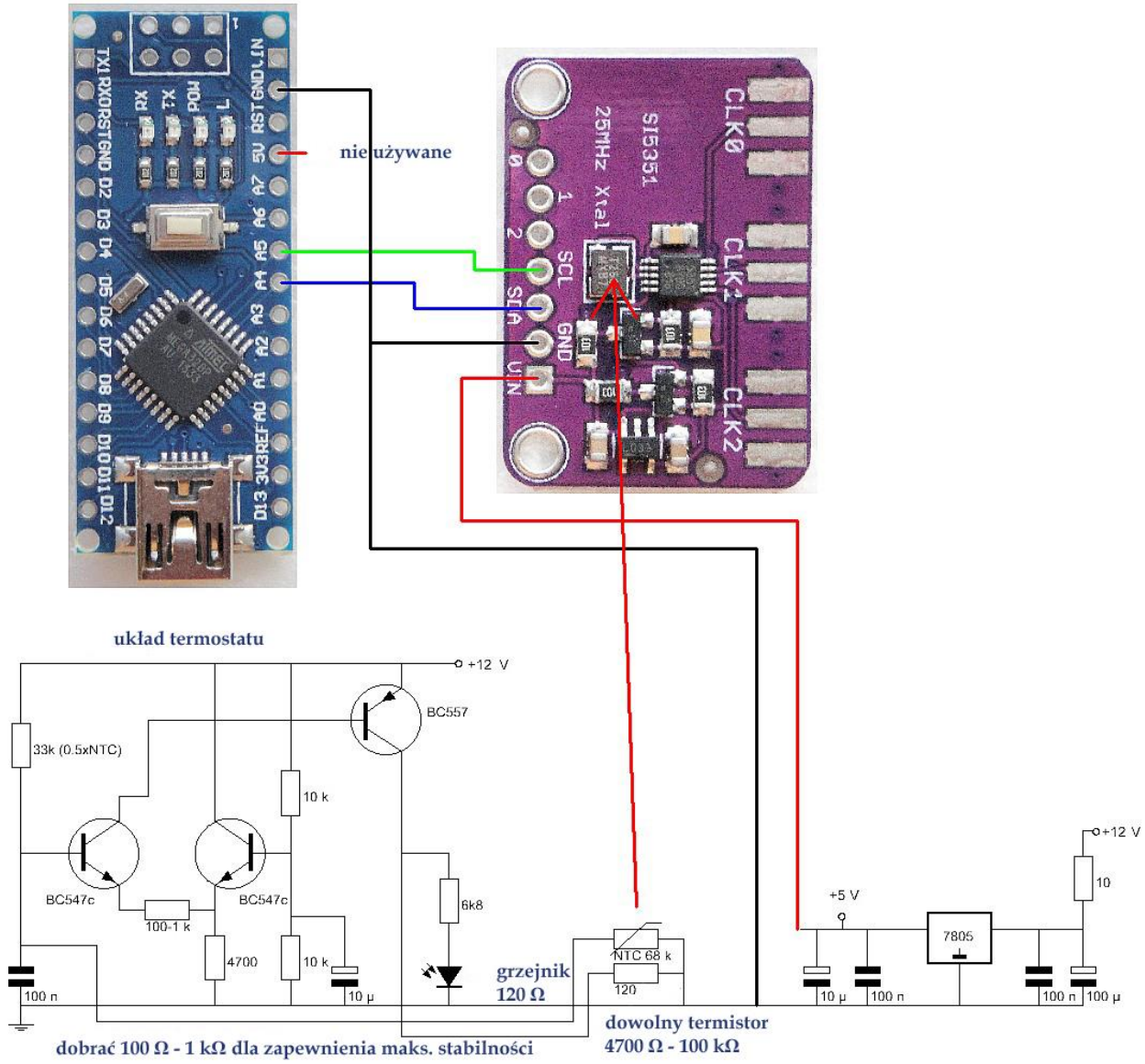
Element	Indukcyjność	Uzwojenie	Rdzeń
L3	136 µH	18 zw. CuEm 0,5 mm	FT37-43
L4, L5	1,5 µH	19 zw. CuEm 0,5 mm	T30-2



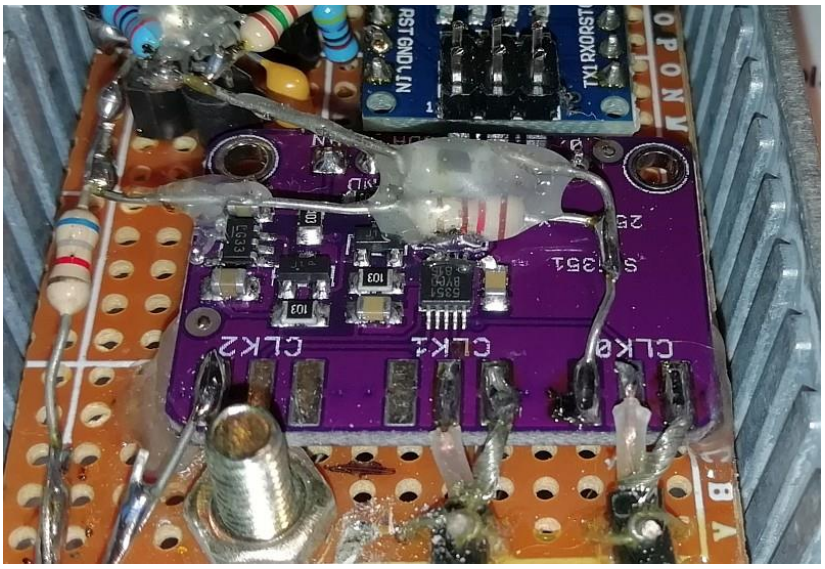
Rys. 3.3.2.5. Schemat ideowy nadajnika F4GOH

3.3.3. Stabilizacja temperatury dla Si5351

W układzie termostatu jako czujnik temperatury kwarcu służy termistor (NTC), a jako grzejnik – opornik 120 Ω. Zmiana oporności termistora powoduje zmiane napięcia wyjściowego z dzielnika oporowego. Napięcie to powzmocnieniu steruje graejnikiem poprzez trazystor pnp BC557. Dioda elektroluminescencyjna (DEL) w jego kolektorze wskazuje włączenie grzejnika.
Termostat można oczywiście wykorzystać dla stabilizacji temperatury dowolnych kwarców w innych układach generatorów i syntezerów.



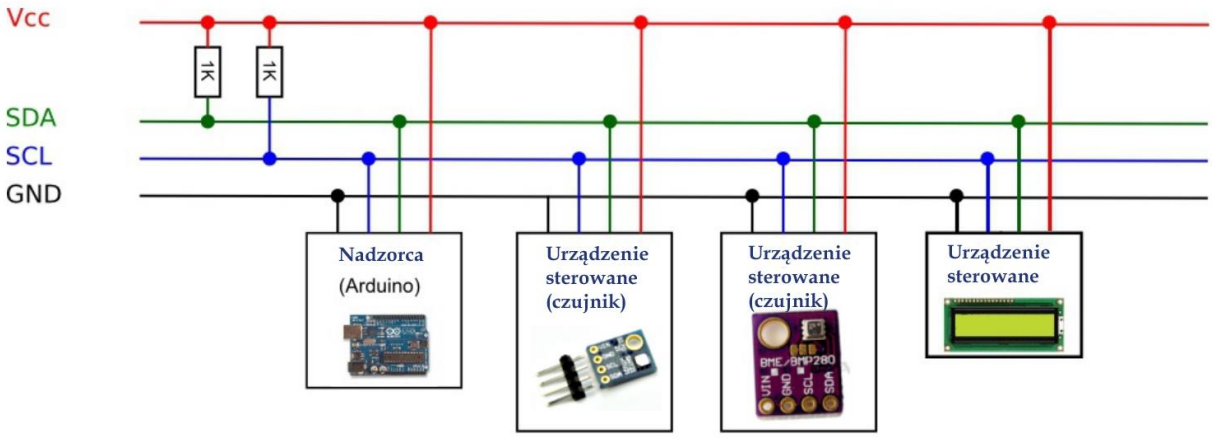
Rys. 3.3.3.1. Schemat termostatu (źródło: *https://www.qsl.net/pa2ohh/20vfobfo.htm*)



Fot. 3.3.3.2. Sposób wykonania termostatu. Opornik grzewczy i termistor są przyklejone do kwarcu na płytce syntezera (źródło: *https://www.qsl.net/pa2ohh/20vfobfo.htm*)

3.3.4. Magistrala I2C

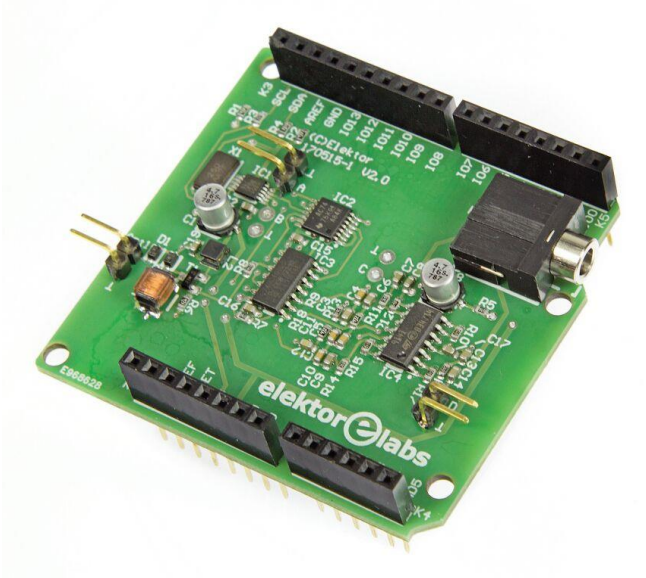
Magistrala I2C została opracowana z myślą o połączeniach wewnątrz urządzeń, a nawet ograniczonych tylko do elementów znajdujących się na wspólnej płytce drukowanej (ang. PCB). Urządzenie sterujące (najczęściej mikrokomputer) może komunikować się maksymalnie ze 127 urządzeniami podporządkowanymi takimi jak różnego rodzaju czujniki, wyświetlacze itp.
Magistrala składa się z dwóch przewodów: danych (SDA) i zegarowego (SCL), wymagających włączenia oporników podciągających do napięcia zasilania. Trzecim przewodem jest masa, a czwartym przewód zasilający. Transmisja z szybkościami 100 – 400 kb/s odbywa się synchronicznie. Dla wielu typów mikrokomputerów, w tym także dla Arduino istnieją gotowe biblioteki ułatwiające korzystanie z magistrali.



Rys. 3.3.4.1. Zasada pracy magistrali I2C

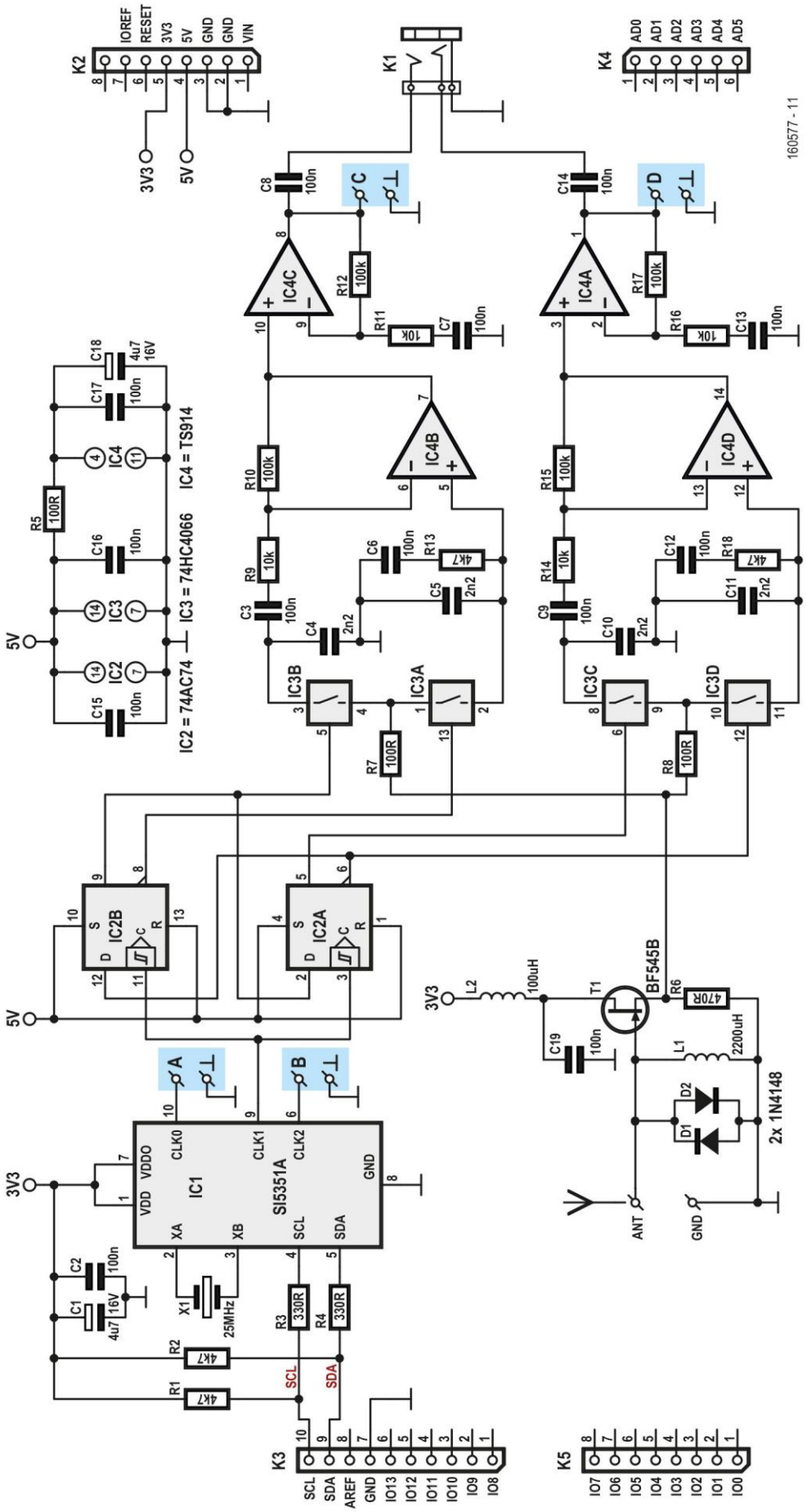
3.4. Odbiorniki programowalne z Arduino

Ograniczone zasoby i częstotliwość zegarowa Arduino nie pozwalają wprawdzie na bezpośrednią cyfrową obróbkę sygnałów radiowych, ale można dodać do niego płytkę rozszerzeń z gotowym odbiornikiem programowalnym.
Przykładem takiego rozwiązania jest odbiornik oferowany przez wydawnictwo Elektor w jego sklepie internetowym (*https://www.elektor.de/elektor-sdr-shield-2-0-module-170515-91*). Płytka odbiornika jest wyposażona we wtyki na krawędziach, pasujące bezpośrednio do gniazd na płytce Arduino UNO i podobnych modeli. Arduino służy do sterowania funkcjami odbiornika i do jego przestrajania.
Odbiornik zasilany napięciem 3,3, lub 5 V pokrywa zakres 150 kHz do 30 MHz i ma czułość 1 μV.



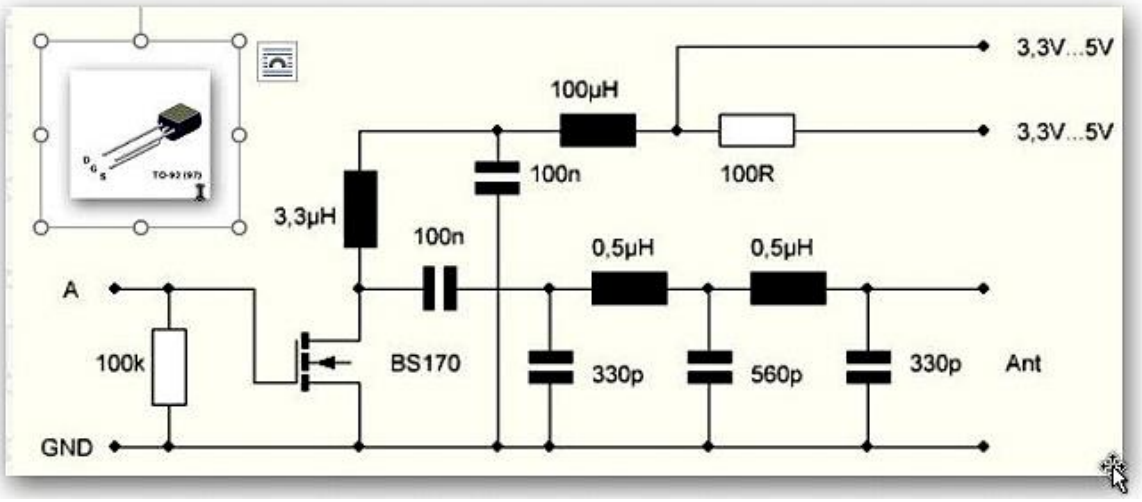
Fot. 3.4.1. Płytka odbiornika *Elektora*

Jako programu odbiorczego można użyć G8JCFSDR albo SDR#. Arduino jest podłączony do złącza COM symulowanego na wyjściu USB komputera PC, a wyjście m.cz. odbiornika do wejścia jego systemu dźwiękowego. Jako źródło sygnału należy więc w programach wybrać system dźwiękowy. Pasmo wyświetlane przez program na ekranie (pasmo odbierane) zależy od ustawienia częstotliwości próbkującej w programie odbiorczym. Może ona przykładowo wynosić 48, 96 kub 192 kHz.
Oprogramowanie do pracy odbiornika jest dostępne w witrynie wydawnictwa pod adresem:
https://www.elektor.de/amfile/file/download/file/2018/product/9354/
W komplecie z odbiornikiem są także rdzeń pierścieniowy T80-2 i ferrytowy dwuotworowy do nawinięcia ewentualnych transformatorów wejciowych albo symetryzatorów.
Po zainstalowaniu dodatkowych programów – DREAM, WSJT-X, WSPR, Fldigi i wirtualnego „kabla” VAC albo VB-CABLE – możliwy jest odbiór radiofonii cyfrowej DRM i krótkofalarskich emisji cyfrowych takich jak PSK31, WSPR, FT8 itd.
Schemat ideowy odbiornika przedstawiono na ilustracji 3.4.2. Jest to rozwiązanie często spotykane obecnie w prostych odbiornikach programowalnych (dawniej było to rozwiązanie standardowe) składające się z mieszacza kwadraturowego dostarczającego na wyjściach sygnału synfazowego (I) i kwadraturowego (Q). Mieszacz pracuje na czterech przełącznikach CMOS wchodzących w skład układu scalonego 74HC4066. Wyjściowe sygnały m.cz. z mieszaczy są wzmacniane za pomocą wzmacniaczy operacyjnych TS914 i doprowadzane odpowiednio do lewego i prawego kanału wejściowego systemu dźwiękowego komputera. Jako heterodyna pracuje scalony syntezer Si5351A. Heterodyna oscyluje na czterokrotnej częstotliwości odbioru i po podziale częstotliwości przez cztery za pomocą przerzutników D 74AC74, które dostarczają przesuniętych o 90° sygnałów sterujących przełączniki mieszcza. Tranzystor polowy T1 BF245B w układzie wtórnika źródłowego zapewnia dopasowanie anteny do wejścia mieszczy. Diody D1 i D2 zabezpieczają jego wejście przed przepięciami. Wejście odbiornika jest aperiodyczne i nie posiada żadnego obwodu selektywnego.


K2
IOREF
RESET
3V3
5V
GND
GND
VIN
K1
K4
AD0
AD1
AD2
AD3
AD4
AD5
160577 - 11
IC4C
IC4A
IC4B
IC4D
IC4 = TS914
IC3 = 74HC4066
IC2 = 74AC74
IC3B
IC3A
IC3C
IC3D
IC2B
IC2A
IC1
SI5351A
BF545B
2x 1N4148
SCL
SDA
K3
K5

Rys. 3.4.2

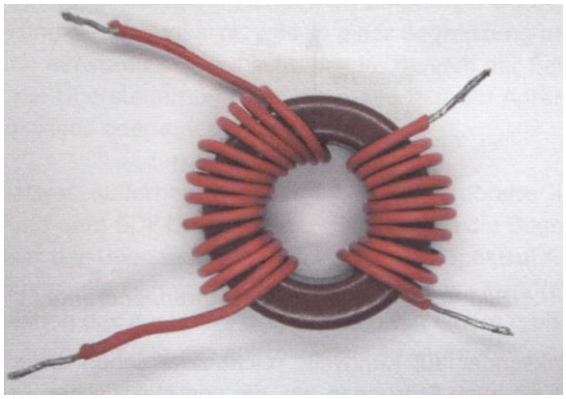
Dla uniknięcia zakłóceń spowodowanych m.in. przez odbiór stacji leżących w okolicach trzeciej i ewentualnie piątej harmonicznej heterodyny dobrze jest dodać na wejściu filtr dolnoprzepustowy lub przełączane filtry (prostokątny sygnał heterodyny jest bogaty w nieparzyste harmoniczne, teoretycznie jest ich nieskończenie wiele, w pratktyce poziomy wyższych niż 3 i ew. 5 są pomijalnie niskie). W celu zmniejszenia niebezpieczeństwa przesterowania odbiornika przy odbiorze przez długie anteny można na jego wejściu dodać tłumik przełączany lub regulowany płynnie (potencjometr 10 kΩ) albo włączyć równolegle do wejścia opornik 51 Ω.
W przypadku korzystania z anten znajdujących się poza domem i połączonych z odbiornikiem za pomocą kabla koncentrycznego można zastosować transformator w.cz. nawinięty na rdzeniu pierścieniowym. Do zacisków uzwojenia pierwotnego jest podłączona antena, a do zacisków wtórnego kabel antenowy prowadzący do wejścia odbiornika. Rozmiary rdzenia i przekładnia transformatora są mało krytyczne i można z nimi eksperymentować. W opisie odbiornika podana jest propozycja 14 zwojów po stronie pierwotnej i 10 po stronie wtórnej. Oba uzwojenia są nawinięte dowolnym przewodem w izolacji plastikowej na rdzeniu pierścieniowym T80-2 (rys. 3.4.4 i 3.4.5).
Syntezer jest sterowany przez Arduino poprzez magistralę I2C. Polecenia dla Arduino pochodzą z programu odbiorczego na PC i są podawane przez złącze USB. Sam mikrokomputer nie ma zasadniczo wiele do zrobienia. Syntezer i wtórnik źródłowy są zasilane napięciem 3,3 V, a reszta układu napięciem 5 V.
W odbiornikach programowalnych wyższej klasy stosowana jest obecnie bezpośrednia przemiana analogowo-cyfrowa. Przetwarzanie tak otrzymanego strumienia danych wymaga szybszego procesora.
Oprócz zwykłego zastosowania odbiornika do odbioru sygnałów radiowych docierających przez antenę odbiornik można wykozystać także do celów pomiarowych. Po skalibrowaniu wskazań mierników siły sygnału i wskaźników widma można dokładniej mierzyć poziomy sygnałów doprowadzonych do wejścia. Możliwe są także pomiary charakterystyk przenoszenia czwórników, a po dołączeniu mostka do pomiaru WFS także współczynnika fali stojącej. W przypadku sygnałów o wyższych poziomach można doprowadzać je przez tłumik o znanym tłumieniu.
Odbiornik jest przeznaczony do różniego rodzaju eksperymentów amatorskich i zapoznawania się z tą techniką i można go także uzupełnić o tor nadawczy. Program *JTEncoder* pozwala nawet na nadawanie emisji cyfrowych. Schemat wzmacniacza o mocy wyjściowej 200 mW (przy zasilaniu 5 V, dla 3,3 V moc wyjściowa wynosi około 100 mW) przedstawia rys. 3.4.3.



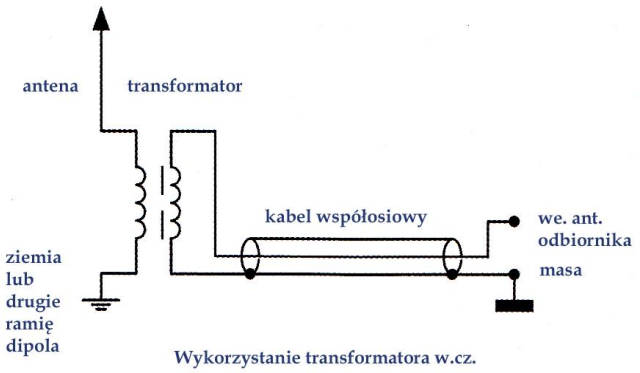
Rys. 3.4.3. Schemat ideowy wzmacniacza mocy 200 mW. Na wyjściu wyprowadzonym przez opornik 100 Ω jest ona o 10 dB niższa

Szczególnie interesujące przy pracy niskimi mocami jest nadawanie komunikatów WSPR. Ze względu na wąskie pasma WSPR przed rozpoczęciem nadawania należy dokładnie sprawdzić i wykalibrować częstotliwość transmisji.
Indukcyjności filtru dolnoprzepustowego nawinięto na rdzeniach pierścieniowych T50-1. Uzwojenia dla pasm 30 i 20 m, o indukcyjnościach 600 nH mają po 8 zwoi. Wejście wzmacniacza mocy jest połączone z wyjściem generatora CLK0 syntezera Si5351A oznaczonym na chemacie 3.4.2 literą A.

Literą B jest oznaczone (nieużywane tutaj) wyjście CLK2 syntezera. Zmodyfikowane oprogramowanie dla Arduino znajduje się pod adresem:
https://www.elektronik-labor.de/HF/SDRshield21.html
(https://www.elektronik-labor.de/HF/WSPR_SDR2.zip).

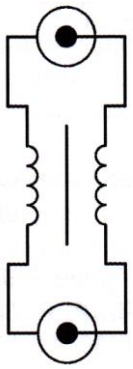


Fot. 3.4.4. Transformator wejściowy



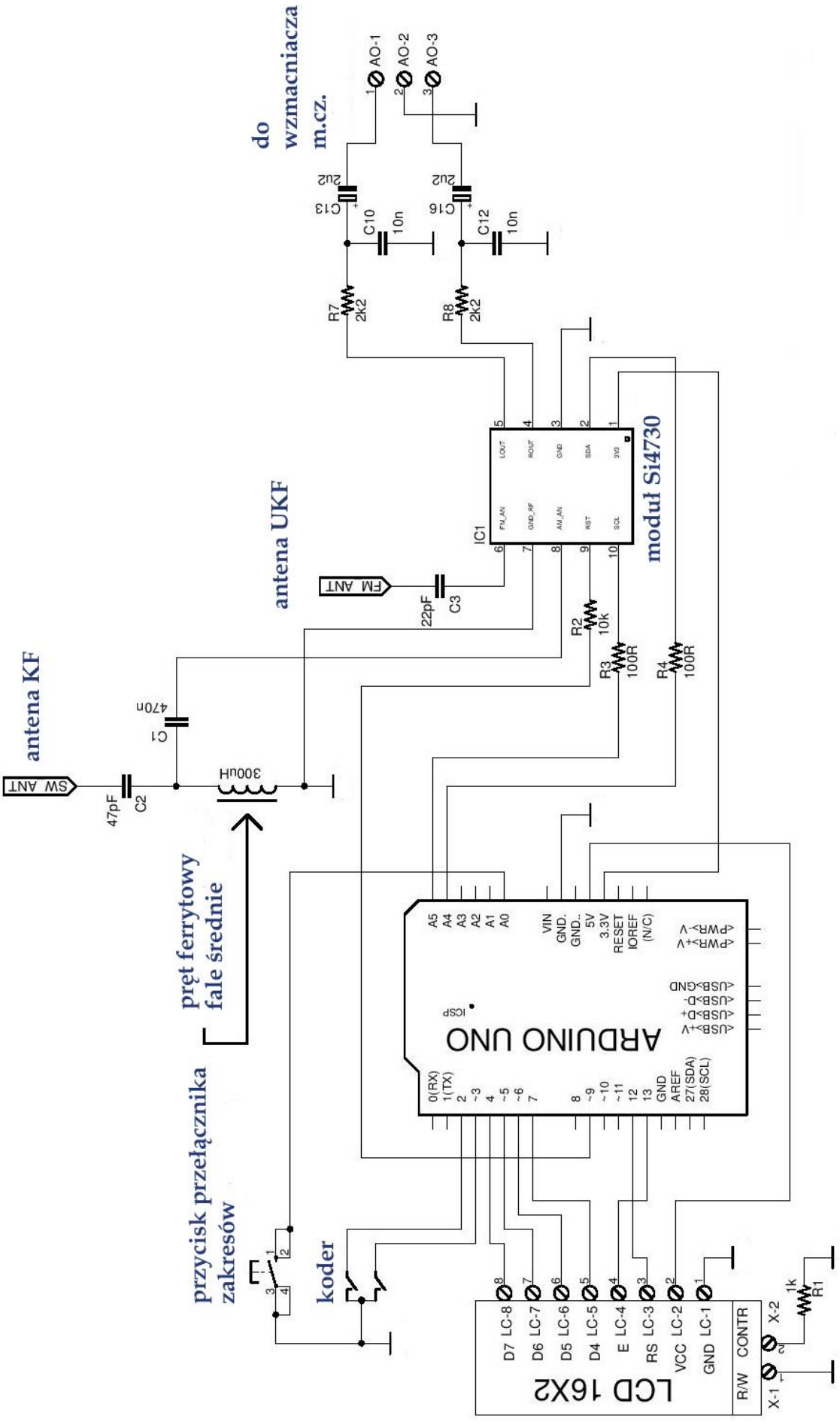
Rys. 3.4.5. Schemat podłączenia

Fot. 3.4.6. Wykonanie dławika fali powierzchniowej z kabla RG174



Rys. 3.4.7. Sposób włączenia

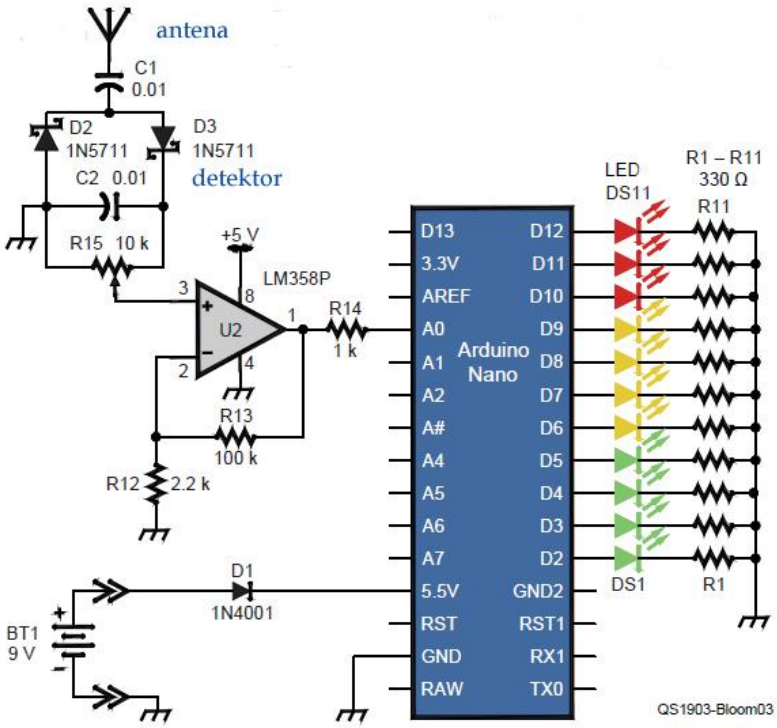
Scalone układy Si4730-D60 są kompletnymi odbiornikami pracującymi na zasadzie cyfrowej obróbki sygnałów i pokrywającymi radiofoniczne zakresy fal długich, średnich, krótkich i UKF pomiędzy 200 kHz i 108 MHz. Układy Si4735-D60 i Si4732-A10 pozwalają nawet na odbiór emisji SSB. W odróżnieniu od układów odbiorczych z serii Si483x przystosowanych do regulacji i strojenia za pomocą potencjometrów i do przełączania zakresów przy użyciu przełącznika wybierającego odpowiedni odczep oporowego dzielnika napięcia układy z serii Si473x są przystosowane do sterowania za pomocą mikrokomputerów przez magistralę I2C. W warunkach amatorskich może być to jeden z modeli Arduino, np. Arduino Nano. Do tego celu zostały też opracowane odpowiednie biblioteki programów. Scalone odbiorniki Si47xx są dostępne po niskich cenach w takich sklepach internetowych jak Aliexpress, Amazon itd. Do wyboru są także moduły z innymi typami odbiorników np. tylko UKF Si4703 oraz nadajników UKF z Si4713. Oprogramowanie dla Arduino do obsługi odbiornika Si4735 jest dostępne pod adresem:
https://github.com/pu2clr/SI4735.
Przełączanie zakresów i regulacja siły głosu wymagają w tej wersji oporogramowania odpowiednio pojedyńczego lud dwukrotnego naciśnięcia gałki kodera. Oprócz Arduino i układu odbiorczego w skład całości wchodzi jeszcze dwuliniowy wyświetlacz graficzny na ciekłych kryształach. Schemat ideowy odbiornika przedstawiono na ilustracji 3.4.8 na poprzedniej stronie (źródło:
https://create.arduino.cc/projecthub/mircemk/homemade-arduino-si4730-all-band-receiver-lw-mw-sw-fm-cbfdd6).


do wzmacniacza m.cz.
AO-1
AO-2
AO-3
C13 2u2
C16 2u2
C10 10n
C12 10n
R7 2k2
R8 2k2
antena UKF
moduł Si4730
IC1
FM ANT
22pF C3
R2 10k
R3 100R
R4 100R
antena KF
SW ANT
47pF C2
300uH
C1 470n
pręt ferrytowy fale średnie
przycisk przełącznika zakresów
koder
ARDUINO UNO
LCD 16X2
D7 LC-8
D6 LC-7
D5 LC-6
D4 LC-5
E LC-4
RS LC-3
VCC LC-2
GND LC-1
R/W CONTR
X-1
X-2
R1 1k

Fot. 3.4.9. Przykład wykonania odbiornika

3.5. Proste układy pomiarowe

Opisane tutaj proste układy pomiarowe mogą się przydać w trakcie uruchamiania nadajników małej mocy i innych przedstawionych w skrypcie układów elektronicznych. Wskaźnik natężenia pola na Arduino konstrukcji AC5YL został opublikowany w numerze 3/2019 amerykańskiego miesięcznika QST. Wskaźnik składa się z diodowego detekttora w.cz. na diodach Schottkiego, mikrokomputera Arduino Nano i rzędu kolorowych diod świecących DS1 – DS11. Diody można umieścić w jednej linii prostej albo wzdłuż linii o dowolnym kształcie. Oporniki R1 – R11 ograniczają prąd płynący przez diody i należe je dobrać w razie potrzeby zależnie od typu diod. W detektorze zamiast diod Schottkiego można użyć diod germanowych mających o połowę niższe napięcie progowe.



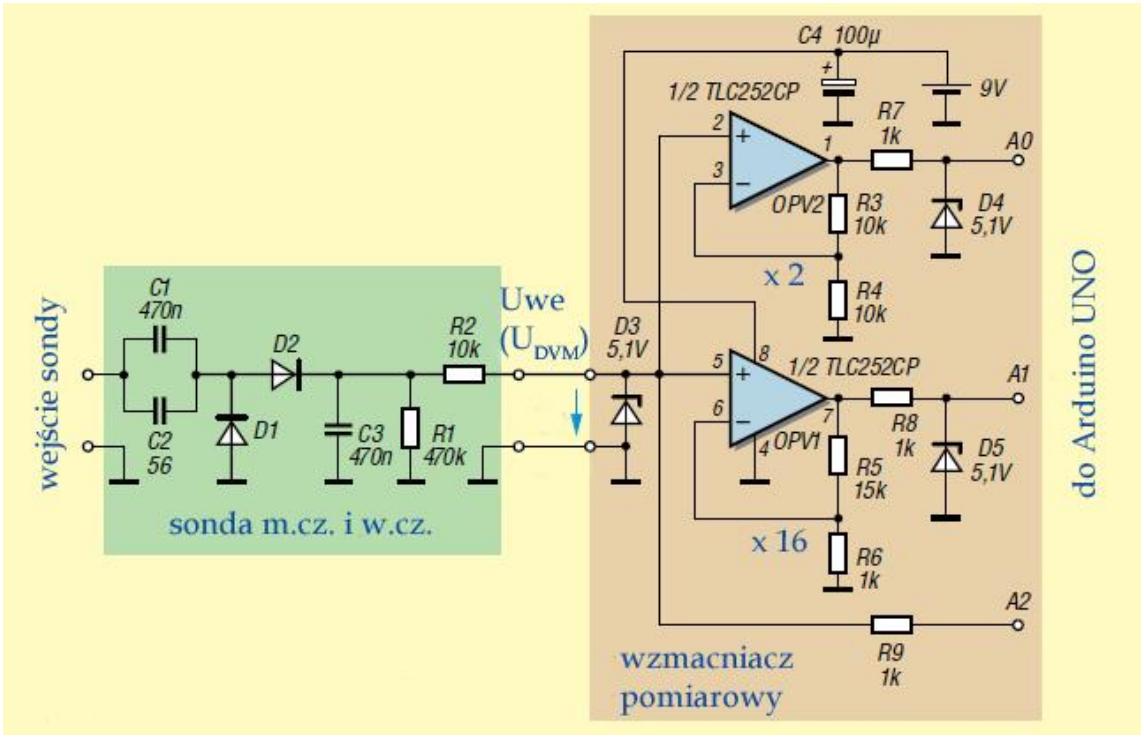
Rys. 3.5.1. Schemat ideowy wskaźnika. Wartości ułamkowe pojemności oznaczają mikrofarady, a wartości całkowite – pikofarady

Napięcie z detektora w.cz. jest po wzmocnieniu we wzmacniaczu operacyjnym podawane na wejście analogowe A0 Arduino. Najniższym natężeniom pola odpowiada świecenie diody D1, a w miarę jego wzrostu zaświecają się następne diody. Program dla Arduino dostępny jest wprawdzie tylko dla prenumeratorów QST pod adresem *www.arrl.org/qst-in-depth,* ale zasadniczo zadanie jest na tyle proste, że napisanie własnego programu nie powinno przysporzyć trudności nawet mniej zaawansowanym programistom Arduino. AC5YL wzorował się na programie z rozdziału „RF Probe with LED Bar Graph" z książki Glena Popiela KW5GP „Arduino for Ham Radio" wydanej przez ARRL.

Tabela 3.5.1
Spis elementów wskaźnika natężenia pola

Element	Wartość	Element	Wartość
BT1	Bateria 9 V	R1 – R11	330 Ω
C1, C2	0,01 µF	R12	2,2 kΩ
D1	Dioda prostownicza 1N4001	R13	100 kΩ
D1, D2	Diody Schottkiego 1N5711	R14	1 kΩ
DS1 – DS4	Zielone diody świecące	R15	Potencjometr 10 kΩ
DS5 – DS8	Żółte diody świecące	U1	Arduino Nano
DS9 – DS11	Czerwone diody świecące	U2	Wzm. op. LM358P

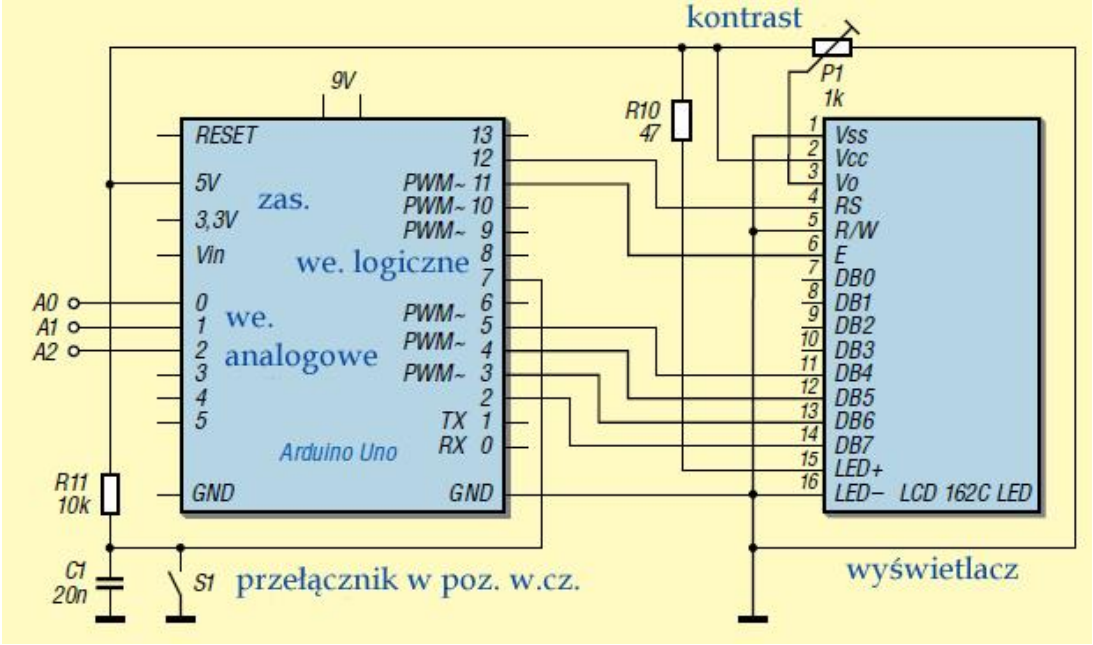
Woltomierz w.cz. i m.cz. na Arduino opracowany przez DF1RN został opisany w numerze 9/2017 *Funkamateura*. Woltomierz pozwala w zakresie m.cz. do 20 kHz na pomiar napięcia 0,11 – 2,23 V z dokładnością 2% (dla 1 kHz) i w zakresie 1 – 30 MHz na pomiar napięcia 0,06 – 2,23 V (–20 – 11 dBm) z dokładnością 0,5% (dla 8 MHz). Przy zasilaniu z baterii 9 V pobór prądu toru pomiarowego wynosi 5,5 mA.



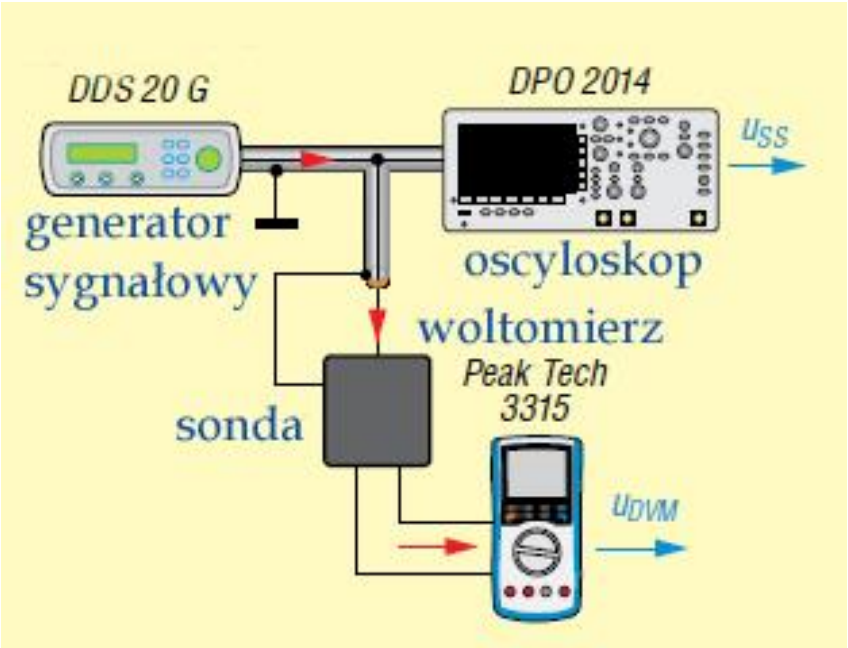
Rys. 3.5.2. Schemat ideowy sondy ze wzmacniaczem

Konstrukcja składa się z sondy detekcyjnej, wzmacniacza, mikrokomputera Arduino UNO i wyświetlacza ciekłokrystalicznego o formacie 2 x 16 znaków. Układ sondy detekcyjnej m.cz. i w.cz. z podwajaczem napięcia jest widoczny na rys. 3.5.2 na tle zielonym, a układ wzmacniacza na tle pomarańczowym. Zawiera on trzy równolegle tory bezpośredni i dwa o wzmocnieniu 16 i 2 razy, których wyjścia są doprowadzone odpowiednio do wejść analogowych mikroprocesora A2, A1 i A0. Wyjścia wzmac-

niaczy są zabezpieczone za pomocą diod Zenera 5,1 V aby nie uszkodzić mikroprocesora. Wybór zakresu (wejścia) jest dokonywany przez program przy napięciu wejściowym A2 poniżej 0,3 V mierzone jest napięcie na wejściu A1. Przy napięciu A2 przekraczającym 0,3 V mierzone jest napięcie wzmocnione dwukrotnie i podawane na wejście A0. Oznacza to, że napięcie na wejściu wzmacniaczy nie może przekraczać 2,5 V. Przy uwzględnieniu marginesu bezpieczeństwa przyjęto górną granicę 2,23 V. Wzmacniacz w torze A1 znajduje się w tym przypadku w nasyceniu, ale pobór prądu nie przekracza 5,5 mA. Przy zerowym napięciu wejściowym wynosi on około 0,3 mA. Na rysunku 3.5.3 pokazano schemat części cyfrowej miernika. Program oblicza mierzoną wartość w oparciu o zapisaną w pamięci krzywą kalibracji. Wybór zakresu częstotliwości jest dokonywany za pomocą przełącznika S1 (zwarcie do masy – pomiar m.cz., rozwarcie – w.cz.). Dla napięć w.cz. wyświetlana jest dodatkowo moc sygnału w dBm na oporności 50 Ω.
Kod źródłowy programu znajduje się w witrynie *www.funkamateur.de* na stronie „Downloads/Archiv | Downloads zum Heft" – *https://www.funkamateur.de/downloads-zum-heft.html* pod rocznikiem 2017.

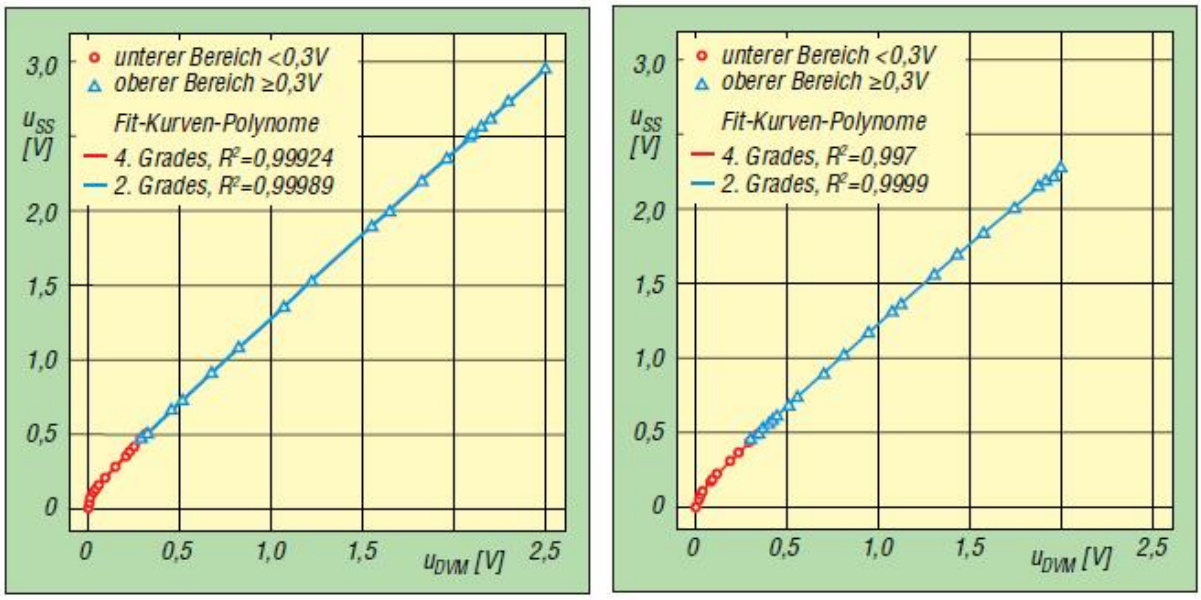


Rys. 3.5.3. Schemat ideowy części mikroprocesorowej



Rys. 3.5.4. Układ pomiarowy do kalibracji

W układzie z rysunku 3.5.4 przeprowadzono pomiary napięcia wyjściowego sondy dla częstotliwości 1 kHz i 8 MHz w zakresie wskazań woltomierza 0 – 2,5 V. Następnie zostały sporządzone wykresy zmierzonej oscyloskopem wartości międzyszczytowej napięcia (na rysunku Uss) w funkcji wskazań woltomierza cyfrowego U_{DVM} Uss = f(U_{DVM}). W oparciu o matematyczne przybliżenia otrzymanych krzywych pomiarowych otrzymano wartość spadku napięcia na diodach (uzyskaną z przekształcenia wzoru $u_{DVM} = 2\ u_{SS}/2 - 2\ u_d$), współczynnik 2 wynika z podwajania napięcia w detektorze:
$u_d = 0{,}5\ (u_{SS} - u_{DVM})$. Dla zakresu w.cz. spadek napięcia na diodach leży pomiędzy 10 – 210 mV.



Rys. 3.5.5. Otrzymane krzywe pomiarowe, po lewej dla 8 MHz, po prawej dla 1 kHz. Na czerwono przedstawiony jest dolny zakres napięć, a na niebiesko górny. W zakresie dolnym przybliżenie za pomocą wielomianu 4 rzędu, w górnym 2 rzędu

$$u_{SS} = f_{uM-HF}(u_{DVM}) = 0{,}02173 + 2{,}9317 \cdot u_{DVM} - 14{,}87012 \cdot u_{DVM}^2 + 54{,}55325 \cdot u_{DVM}^3 - 69{,}9154 \cdot u_{DVM}^4,$$

$$u_{SS} = f_{oM-HF}(u_{DVM}) = 0{,}12732 + 1{,}6729 \cdot u_{DVM} - 0{,}01438 \cdot u_{DVM}^2. \quad (2)$$

$$u_{SS} = f_{uM-HF}(u_{DVM}) = 0{,}00844 + 2{,}80221 \cdot u_{DVM} - 12{,}60763 \cdot u_{DVM}^2 + 41{,}55911 \cdot u_{DVM}^3 - 48{,}36415 \cdot u_{DVM}^4,$$

$$u_{SS} = f_{oM-HF}(u_{DVM}) = 0{,}09192 + 1{,}18577 \cdot u_{DVM} - 0{,}04652 \cdot u_{DVM}^2. \quad (5)$$

Rys. 3.5.6. Stosowane wielomiany przybliżające, po lewej dla pomiru w.cz., po prawej m.cz., u góry dla dolnych zakresów napięciowych, u góry dla wyższych

3.6. Różne pomysły

W Internecie można znaleźć wiele interesujących opisów rozwiązań, przykładowych dających się łatwo zmodyfikować programów i bibliotek stanowiących dużą pomoc i w innych podobnych projektach o tematyce krótkofalarskiej i nie tylko. Poniżej przedstawiamy niektóre adresy mogące zainteresować czytelników:

1. *http://www.amateurfunkbasteln.de/downloads/arduino_ad9850.zip* – program sterujący scalony syntezer cyfrowy AD9850. Strona DL1DMW;
2. *http://www.amateurfunkbasteln.de/downloads/arduino_bme280.zip* – program odczytujący czujnik barometryczny, termometr i wilgościomierz BME280. Strona DL1DMW;

3. *https://github.com/ninjablocks/433Utils* – programy do obsługi modułów radiowych na pasmo 433 MHz;
4. *www.kh-gps.de* – rozmaite konstrukcje na Arduino i nie tylko. Strona DJ7OO;
5. *www.qrp-labs.com* – konstrukcje radiolatarni WSPR i innych emisji;
6. *https://www.elektronik-labor.de/HF/WSPRrxtx.html* – radiostacja WSPR z Arduino i Si5351;
7. *https://github.com/etherkit/JTEncode* – biblioteka koderów emisji JT65, JT9, JT4, FT8, WSPR i FSQ;
8. *https://www.arduinolibraries.info/authors/etherkit* – bilioteki koderów, obsługi Si5351 i telegrafii;
9. *https://www.george-smart.co.uk/arduino/arduino_rtty/* – nadajnik RTTY na Arduino Uno i AD9851;
10. *http://radiohamtech.com/page19.html* – programy radiolatarni RTTY na Arduino z syntezerami AD9850 i Si5351;
11. *https://jontio.zapto.org/download/psk-dds-src.zip* – program do generacji sygnałów PSK31 przy użyciu Arduino i symtezera AD9850;
12. *https://jontio.zapto.org/hda1/BPSK_test_on_DDS.html* – krótki opis realizacji p. 11;
13. *http://groups.io/g/SoftwareControlledHamRadio* – programy do projektów opisanych w książce „Mictocontroller Projects for Amateur Radio” (patrz spis literatury);

Literatura i adresy internetowe

Immler, Christian, Bernauer,Hannah, „Raspberry Pi Serverbuch“, Franzisverlag, Monachium 2014, ISBN 978-3-645—60330-0

Kainka, Burkhard, DK7JD, „Software Defined Radio nutzen. Das SDR-Praxisbuch“, wydawnictwo *Elektor*, Akwizgran [Aachen] 2019, ISBN 978-3-89576-338-0, *www.elektor.de*

Klotz, Leigh L. jr., WA5ZNU, „Ham Radio for Arduino and PICAXE“, wydawnictwo ARRL 2013, ISBN 978-0-87259-324-4

Kofler, Kühnast, Scherbeck, „Raspberry Pi. Das umfassende Handbuch“, wydawnictwo Rheinwerk, wydanie 3, Bonn 2016, ISBN 978-3-8362-4220-2

Popiel, Glen, KW5GP, „Arduino Projects for Ham Radio”, wydawnictwo ARRL 2017, ISBN 978-1-62595-070-3

Purdum,Jack, W8TEE, Peter, Albert, AC8GY, „Microcontroller Projects for Amateur Radio“, ARRL 2020, ISBN 978-1-62595-128-1

Richards, Mike, G4WNC, „Raspberry Pi Explained for Radio Amateurs“, wydawnictwo RSGB 2020, ISBN 9781 9101 9384 6

Roczniki czasopism wymienionych w tekście.

Literatura i adresy internetowe do poszczególnych podrozdziałówww

[1.2.1] www.pa7lim.nl/penaut – witryna autora PA7LIM
[1.2.2] www.pa7lim.nl/penaut-request – strona rejestracji użytkowników
[1.2.3] http://penaut.pa7lim.nl:5678 – pulpit, wykaz aktywności
[1.2.4] https://play.google.com/apps/testing/peanut.peanut – dostęp do programu w witrynie Google Play
[1.3.1] www.pa7lim.nl – witryna PA7LIM, programy *Peanut* i *BlueDV*
[1.3.2] „D-STAR komputerowo”, Krzysztof Dąbrowski, OE1KDA, Świat Radio 5/2019, str. 59; 6/2019 str. 58
[1.3.3] www.combitronics.nl
[1.3.4] nwdigitalradio.com
[1.3.5] http://ambeboard.zumradio.com/
[1.3.6] https://reflectorloversclub.jimdofree.com/shop/
[1.3.7] https://reflectorloversclub.jimdofree.com/
[1.3.8] www.dvsinc.com/products/a300x.shtml – informacje o wokoderach firmy DVSI

[3.1.1.1] „APRS-Internet-Service-Client”, Franz G. Aletsee, DL6FCD, CQDL 7/2019, str. 26
[3.1.1.2] www.arduino.cc – informacje o Arduino, dokumentacja, przykładowe programy
[3.1.1.3] www.aprs2.net/serverstats.php – spis serwerów APRS-IS
[3.1.1.4] www.aprs-is.net/javAPRSFilter.aspx –
[3.1.1.5] www.github.com/fgaletsee/ArduPRS
[3.1.1.6] ham.zmailer.org/oh2mqk/aprx/PROTOCOLS
[3.1.1.7] www.github.com/fgaletsee/APRSEthernet
[3.1.1.8] www.aprs.org/doc/APRS101.pdf
[3.1.1.9] https://apps.magicbug.co.uk/passcode
[3.1.1.10] „Biblioteka polskiego krótkofalowca“, tom 33 „Telemetria”, www.swiatradio.com.pl, strona ‘Biblioteka Radioamatora”

[3.1.2.1] http://wiki.oevsv.at/index.php?title=APRS_Arduino-Modem&oldid=14810 – opis w witrynie OeVSV
[3.1.2.2] http://unsigned.io/projects/microaprs – witryna duńskiego konstruktora
[3.1.2.3] https://github.com/oe7mbt/APRS-Micromodem – projekt płyki drukowanej w formacie KiCAD
[3.1.2.4] https://github.com/markqvist/LibAPRS – biblioteka APRS dla Arduino

[6.1] „Build Your own D-STAR Hotspot”, Bob Wilton, KF5TPQ, QST 2/2020, str. 30
[6.2] www.pistar.uk – oprogramowanie i informacje praktyczne
[6.3] https://amateurradionotes.com/images/1-Playing_with_Pi-Star.pdf – instrukcja do programu „Pi-Star” po angielsku
[6.4] https://www.balena.io/etcher/ – program kopiujący na moduły SD
[6.5] https://sourceforge.net/projects/win32diskimager/ – program kopiujący „Disk Imager”
[6.6] https://www.chip.de/downloads/Win32-Disk-Imager_46121030.html
[6.7] www.advanced-ip-scanner.com – program wyświetlający adresy IP w sieci domowej

W serii „Biblioteka polskiego krótkofalowca” dotychczas ukazały się:

Nr 1 – „Poradnik D-STAR”, wydanie 1 (2011), 2 (2015) i 3 (2019)
Nr 2 – „Instrukcja do programu D-RATS”
Nr 3 – „Technika słabych sygnałów” Tom 1
Nr 4 – „Technika słabych sygnałów” Tom 2
Nr 5 – „Łączności cyfrowe na falach krótkich” Tom 1
Nr 6 – „Łączności cyfrowe na falach krótkich” Tom 2
Nr 7 – „Packet radio”
Nr 8 – „APRS i D-PRS”
Nr 9 – „Poczta elektroniczna na falach krótkich” Tom 1
Nr 10 – „Poczta elektroniczna na falach krótkich” Tom 2
Nr 11 – „Słownik niemiecko-polski i angielsko-polski” Tom 1
Nr 12 – „Radiostacje i odbiorniki z cyfrową obróbką sygnałów” Tom 1
Nr 13 – „Radiostacje i odbiorniki z cyfrową obróbką sygnałów” Tom 2
Nr 14 – „Amatorska radioastronomia”
Nr 15 – „Transmisja danych w systemie D-STAR”
Nr 16 – „Amatorska radiometeorologia”, wydanie 1 (2013) i 2 (2017)
Nr 17 – „Radiolatarnie małej mocy”
Nr 18 – „Łączności na falach długich”
Nr 19 – „Poradnik Echolinku”
Nr 20 – „Arduino w krótkofalarstwie” Tom 1
Nr 21 – „Arduino w krótkofalarstwie” Tom 2
Nr 22 – „Protokół BGP w Hamnecie”
Nr 23 – „Technika słabych sygnałów” Tom 3, wydanie 1 (2014), 2 (2016) i 3 (2017)
Nr 24 – „Raspberry Pi w krótkofalarstwie”
Nr 25 – „Najpopularniejsze pasma mikrofalowe”, wydanie 1 (2015) i 2 (2019)
Nr 26 – „Poradnik DMR” wydanie 1 (2015), 2 (2016) i 3 (2019), nr 326 – wydanie skrócone (2016)
Nr 27 – „Poradnik Hamnetu”
Nr 28 – „Budujemy Ilera” Tom 1
Nr 29 – „Budujemy Ilera” Tom 2
Nr 30 – „Konstrukcje D-Starowe”
Nr 31 – „Radiostacje i odbiorniki z cyfrową obróbką sygnałów” Tom 3
Nr 32 – „Anteny łatwe do ukrycia”
Nr 33 – „Amatorska telemetria”
Nr 34 – „Poradnik systemu C4FM”, wydanie 1 (2017) i 2 (2019)
Nr 35 – „Licencja i co dalej” Tom 1
Nr 36 – „Cyfrowa Obróbka Sygnałów”
Nr 37 – „Telewizja amatorska”
Nr 38 – „Technika słabych sygnałów” Tom 4, wydanie 1 (2018) i 2 (2020)
Nr 39 – „Łączności świetlne”
Nr 40 – „Radiostacje i odbiorniki z cyfrową obróbką sygnałów” Tom 4
Nr 41 – „Licencja i co dalej” Tom 2
Nr 42 – „Miernictwo” Tom 1
Nr 43 – „Miernictwo” Tom 2
Nr 44 – „Miernictwo” Tom 3
Nr 45 – „Testy sprzętu” Tom 1
Nr 46 – „Testy sprzętu” Tom 2
Nr 47 – „Licencja i co dalej” Tom 3
Nr 48 – „Jonosfera i propagacja fal”
Nr 49 – „Anteny krótkofalowe” Tom 1
Nr 50 – „Anteny ultrakrótkofalowe” Tom 1
Nr 51 – „Anteny krótkofalowe” Tom 2
Nr 52 – „Anteny ultrakrótkofalowe” Tom 2
Nr 53 – „Anteny mikrofalowe”

Nr 54 – „Proste odbiorniki amatorskie” Tom 1
Nr 55 – „Proste odbiorniki amatorskie” Tom 2
Nr 56 – „Proste nadajniki amatorskie” Tom 1
Nr 57 – „Proste nadajniki amatorskie” Tom 2
Nr 58 – „Mini- i mikrokomputery w krótkofalarstwie” Tom 1
Nr 59 – „Mini- i mikrokomputery w krótkofalarstwie” Tom 2