pocketRadio

Radioodbiornik kieszonkowy z RDS (1)

W praktyce elektronika nadchodzi taki moment, gdy z nostalgią wraca do korzeni wspominając początki swojej pasji. Nie inaczej było w moim wypadku. Początki mojej przygody z elektroniką sięgają już prawie ćwierć wieku wstecz. Z rozrzewnieniem wspominam swoje pierwsze konstrukcje, które mimo prostoty dawały wiele satysfakcji. Jak chyba każdy w tym czasie, swoją przygodę rozpoczynałem od skonstruowania nieskomplikowanego radyjka, złożonego zaledwie z kilku elementów. Mimo prostoty konstrukcji, układ tego rodzaju nie był wcale łatwy do uruchomienia, a to za sprawą wielu elementów indukcyjnych, które znacząco wpływały na efekt końcowy. Postanowiłem niejako zawrócić czas i ponownie skonstruować radioodbiornik amatorski, ale z uwzględnieniem nowych umiejętności i 25 lat rozwoju elektroniki.

Rekomendacje: *estetyczny, funkcjonalny radioodbiornik kieszonkowy, który może być kapitalnym prezentem, bazą dla własnej konstrukcji lub użytkowany jak pełnowartościowy radioodbiornik przenośny.*

Ponieważ ćwierć wieku w elektronice stanowi przepaść technologiczną, przyjąłem znacznie bardziej ambitne założenia. Celem moim było skonstruowanie nowoczesnego, przenośnego i energooszczędnego odbiornika na pasmo FM, wyposażonego w RDS oraz efektowny interfejs użytkownika. Pracę rozpocząłem od poszukiwania peryferiów, przy których użyciu mógłbym osiągnąć cel. Przygotowania nie trwały zbyt długo, gdyż przypadkowo natknąłem się na instrukcję serwisową telefonu komórkowego ze scalonym odbiornikiem radiowym FM. Jak się później okazało, układy tego typu stosowane są przez wielu producentów telefonów komórkowych czy odtwarzaczy MP3, z którymi zintegrowano odbiornik lub transmitter FM. Mowa o układzie scalonym firmy Silicon Labs typu Si4703. Jest on kompletnym odbiornikiem radiowym przeznaczonym do odbioru emisji w paśmie FM charakteryzującym się następującymi, wybranymi cechami użytkowymi:

- odbiór stacji radiowych w zakresie 76…108 MHz,
- cyfrowa synteza częstotliwości z wbudowanym oscylatorem VCO,
- AFC (Automatic Frequency Control) i AGC (Automatic Gain Control),
- obsługa konfigurowalnej funkcji przeszukiwania pasma,
- pomiar mocy sygnału antenowego,
- wbudowana funkcja regulacji głośności sygnału wyjściowego,
- wbudowany układ oscylatora dla rezonatora kwarcowego 32768 Hz,
- obsługa interfejsów I²C oraz SPI,
- brak konieczności strojenia obwodów LC,
- szeroki zakres napięcia zasilania (2,7…5,5 V),
- mały pobór mocy i wbudowany regulator napięcia typu LDO,
- obsługa systemu RDS/RDBS.

W ofercie AVT*
AVT-5401 A

Podstawowe informacje:
- Napięcie zasilania: 2,3…3,3 V DC (2 baterie AA).
- Maksymalny prąd obciążenia (wyświetlacz załączony/przyciemniony/wyłączony): 55 mA/30 mA/17 mA.
- Zakres częstotliwości radioodbiornika FM: 87,5÷108 MHz.
- Typ obsługiwanych wiadomości RDS: PS (Program Service), RT (Radio Text), CT (Clock & Time).
- Maksymalna moc wyjściowa audio: 150 mW.
- Impedancja obciążenia: 16 Ω.

Link do video prezentującego możliwości urządzenia: http://youtu.be/e5SZwS1ZJ1U.

Dodatkowe materiały na CD/FTP:
ftp://ep.com.pl, user: 63241, pass: 741obq51
- wzory płytek PCB
- karty katalogowe i noty aplikacyjne elementów oznaczonych w Wykazie elementów kolorem czerwonym

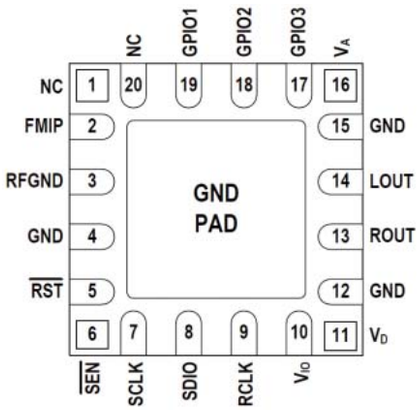
Projekty pokrewne na CD/FTP:
(wymienione artykuły są w całości dostępne na CD)

AVT-5317	Lampowo-tranzystorowy odbiornik UKF (EP 11/2011)
AVT-5242	Radioodbiornik internetowy (EP 7/2010)
AVT-5016	Amplituner FM z RDS (EP 6-7/2001)
AVT-2469	Odbiornik UKF FM (EdW 1/2001)
AVT-2330	Miniaturowy odbiornik FM stereo (EdW 2/1999)

* **Uwaga:**
Zestawy AVT mogą występować w następujących wersjach:

AVT xxxx UK	to zaprogramowany układ. Tylko i wyłącznie. Bez elementów dodatkowych.
AVT xxxx A	płytka drukowana PCB (lub płytki drukowane, jeśli w opisie wyraźnie zaznaczono), bez elementów dodatkowych.
AVT xxxx A+	płytka drukowana i zaprogramowany układ (czyli połączenie wersji A i wersji UK) bez elementów dodatkowych.
AVT xxxx B	płytka drukowana (lub płytki) oraz komplet elementów wymienionych w załączniku pdf
AVT xxxx C	to nic innego jak zmontowany zestaw B, czyli elementy wlutowane w PCB. Należy mieć na uwadze, że o ile nie zaznaczono wyraźnie w opisie, zestaw ten nie ma obudowy ani elementów dodatkowych, które nie zostały wymienione w załączniku pdf
AVT xxxx CD	oprogramowanie (niezęsto spotykana wersja, lecz jeśli występuje, to niezbędne oprogramowanie można ściągnąć, klikając w link umieszczony w opisie kitu)

Nie każdy zestaw AVT występuje we wszystkich wersjach! Każda wersja ma załączony ten sam plik pdf! Podczas składania zamówienia upewnij się, którą wersję zamawiasz! (UK, A, A+, B lub C). **http://sklep.avt.pl**

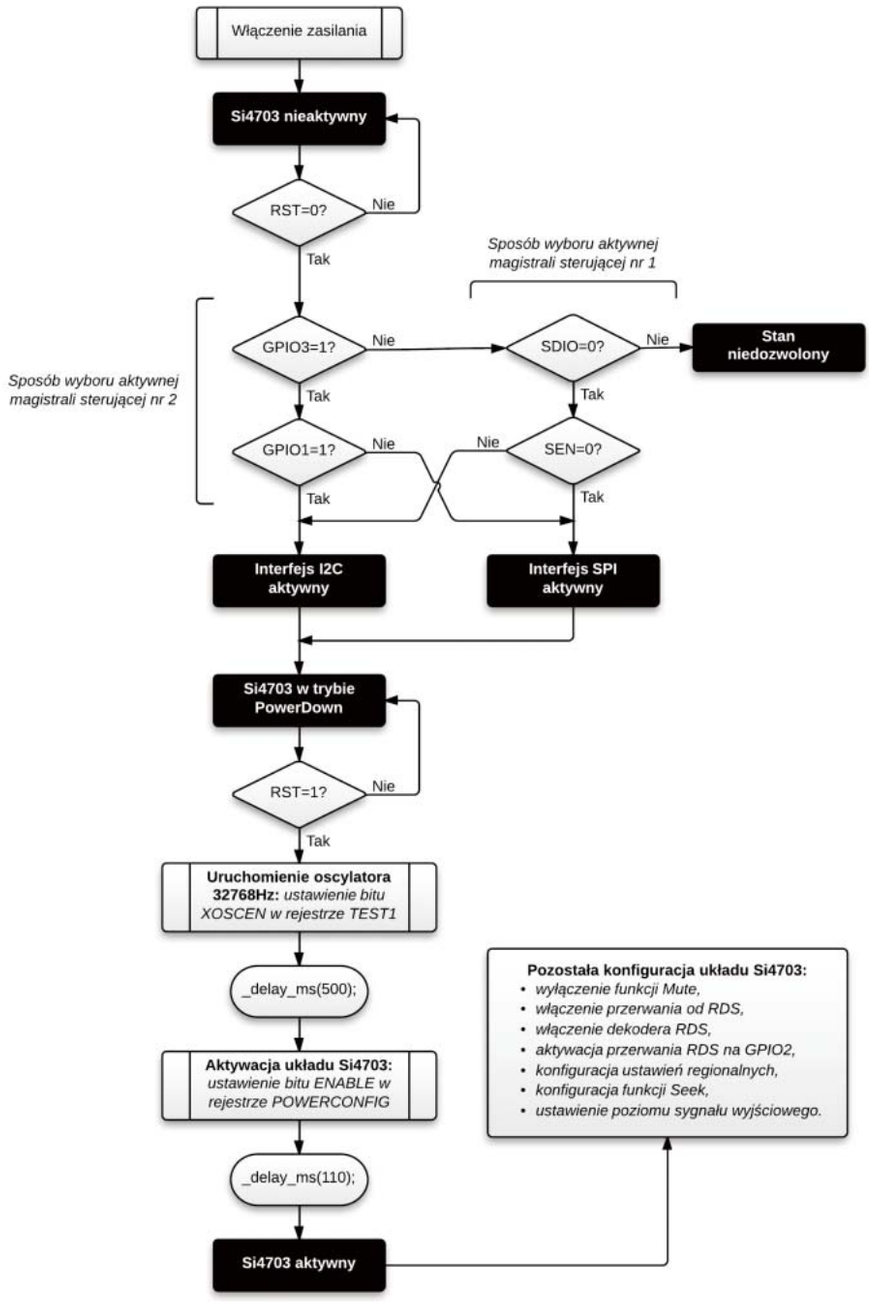


Numer pinu	Nazwa	Opis
1, 20	NC	Niepodłączone
2	FMIP	Wejście sygnału antenowego
3	RFGND	Masa części radiowej układu (należy połączyć z polem masy PCB)
4, 12, 15, PAD	GND	Masa (należy połączyć z polem masy PCB)
5	RST	Reset układu (aktywny stan niski)
6	SEN	Wejście aktywacji i wyboru rodzaju magistrali sterującej (aktywny stan niski)
7	SCLK	Wejście zegarowe magistrali sterującej
8	SDIO	Wejście/wyjście danych magistrali sterującej
9	RCLK	Wejście zewnętrznego sygnału zegarowego syntezera częstotliwości
10	VIO	Napięcie zasilania układów wejścia/wyjścia odbiornika
11	VD	Napięcie zasilania części cyfrowej układu
13	ROUT	Wyjście audio – kanał prawy
14	LOUT	Wyjście audio – kanał lewy
16	VA	Napięcie zasilania części analogowej układu
17	GPIO3	Uniwersalny, programowalny port IO (może pełnić rolę wskaźnika sygnału Stereo)
18	GPIO2	Uniwersalny, programowalny port IO (może pełnić rolę przerwania od gotowości danych RDS lub zakończenia strojenia/przeszukiwania pasma)
19	GPIO1	Uniwersalny, programowalny port IO

Rysunek 1. Wygląd obudowy układu Si4703

Jak widać, powyższe cechy użytkowe naszego elementu idealnie predestynują go do zastosowań w sprzęcie przenośnym zwłaszcza, że aplikacja układu ogranicza się do zaledwie kilku elementów zewnętrznych, ponieważ w budowie układu Si4703 wykorzystano zaawansowany, cyfrowy tor przetwarzania sygnału. Układ Si4703 jest przykładem jednego z wielu produktów z rodziny Si47xx, w której jest dostępnych ponad 25 układów. Są to odbiorniki i nadajniki na wszystkie dostępne pasma radiowe, zróżnicowane pod względem funkcjonalności. Na **rysunku 1** pokazano wygląd obudowy układu Si4703.

Obsługa i konfiguracja układu Si4703 jest możliwa za pomocą interfejsu I²C lub SPI. Aby jednak rozpocząć właściwą transmisję konieczne jest poprawne zainicjowanie układu, które ma na celu wybór aktywnego interfejsu I²C i SPI oraz uruchomienie wewnętrznego oscylatora niezbędnego z punktu widzenia części radiowej układu.



Rysunek 2. Graf kompletnej procedury inicjalizacji układu Si4703

Przewidziano dwa sposoby wyboru rodzaju aktywnej magistrali sterującej różniące się liczbą niezbędnych wyprowadzeń układu Si4703 zaangażowanych w ten proces. Pierwszy zakłada użycie wyprowadzeń GPIO3, SEN oraz SDIO układu, zaś drugi GPIO3 i GPIO1. Zalecanym sposobem wyboru aktywnej magistrali sterującej przy wykorzystaniu wewnętrznego oscylatora dla rezonatora 32768 Hz jest pierwszy, gdyż w takim wypadku wewnętrzne moduły peryferyjne układu Si4703 zapewniają niezbędne ściągnięcie wyprowadzenia GPIO3 pracującego w układzie oscylatora 32768 Hz do masy w czasie, gdy sygnał RST jest wyzerowany. Po wykonaniu procedury wyboru aktywnej magistrali sterującej (tu I²C) jest niezbędne uruchomienie oscylatora 32768 Hz oraz aktywacja układu Si4703 (dzięki bitom Enable/Disable rejestru POWERCONFIG). Graf kom-

Wykaz elementów

Rezystory: (obudowy SMD 0603)
R1: 158 kΩ 1%
R2: 18,2 kΩ 1%
R3: 47 kΩ
R4: 820 kΩ
R5…R9, R12…R13: 20 kΩ
R10, R11: 4,7 kΩ

Kondensatory:
C1: 10 µF/16V (ceramiczny X5, SMD 0805)
C2, C15…C16: 1 µF/16V (ceramiczny X5R, SMD 0603)
C3, C6…C10, C17…C18: 100 nF (ceramiczny X5R, SMD 0603)
C4: 3,3 nF (ceramiczny X5R, SMD 0603)
C5A, C5B: 10 µF/25V (ceramiczny X5R, SMD 0805)
C11…C14: 4,7 µF/20 V (tantalowy, typ B, EIA 3528-21R)
C19: 10 µF/10 V (tantalowy , typ B, EIA 3528-21R)
C20…C21: 47 µF/10 V (tantalowy, typ B, EIA 3528-21R)
C22: 22 nF (ceramiczny X5R, SMD 0603)
C23: 1nF (ceramiczny X5R, SMD 0603)
C24…C25: 22 pF (ceramiczny, SMD 0603)

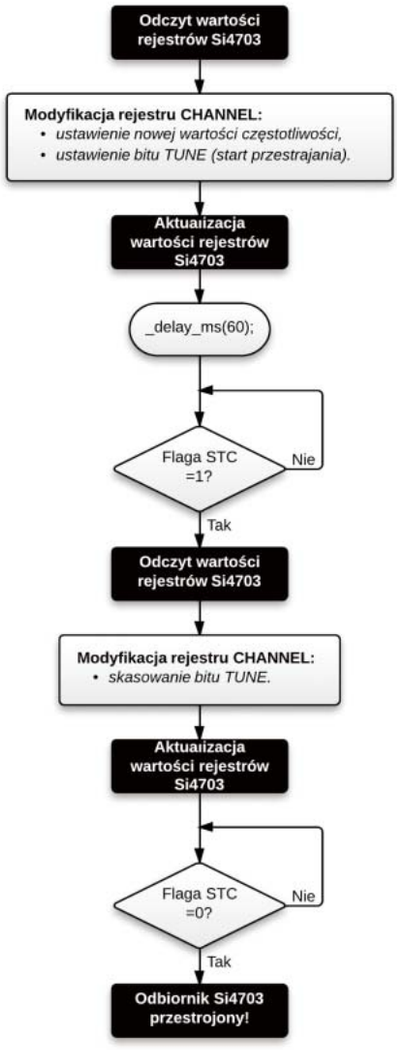
Półprzewodniki:
U1: TPS61085 (TSSOP8)
U2: ATmega164A (TQFP44)
U3: Si4703 (QFN20)
U4: TPA6111 (SO8)
D1: SS14 (SMA)

Inne:
OLED: wyświetlacz OLED Winstar WEX012864LLPP3N00000
L1: dławik mocy 3,3 µH typu DLG-0504-3R3
L2: dławik 10 µH (obudowa SMD 1206)
L3: dławik 270 nH (obudowa SMD 1206)
FERR1, FERR2: koralik ferrytowy BLM18BD252SN1D 2,5 kΩ@100MHz MURATA (obudowa SMD 1206)
Q1: kwarc 32768 Hz
BATT: gniazdo męskie kątowe 90 st. 2pin (NSL25-2W)
JACK: gniazdo Jack 3,5 mm do montażu przewlekanego
UP, DOWN: microswitch z ośką 10 mm do montażu przewlekanego
BAND1…BAND4, MODE: microswitch kątowy 90° do montażu przewlekanego
ZIF: złącze typu ZIF do montażu powierzchniowego (raster 0,5 mm, 30-pin, górny kontakt)

pletnej procedury inicjalizacji układu Si4703 z opcjonalnym wyborem rodzaju magistrali sterującej pokazano na **rysunku 2**.

Po wykonaniu inicjalizacji układu, możemy przystąpić do konfiguracji parametrów sprzętowych. Aby tego dokonać niezbędna jest znajomość struktury rejestrów konfiguracyjnych. Układ Si4703 wyposażono w szereg 16-bitowych rejestrów (adresy 0x00…0x0F), dzięki którym możemy skonfigurować go oraz na bieżąco monitorować aktualny stan pracy. Opis wszystkich, dostępnych opcji wykraczałoby poza ramy tego artykułu i dlatego skupimy się wyłącznie na omówieniu rejestrów, których konfiguracja jest konieczna z punktu widzenia projektu.

Już na tym etapie należy jednak zauważyć pewną nietypową właściwość układu



Rysunek 3. Graf procedury odpowiedzialnej za przestrojenie układu Si4703

Si4703 w zakresie operacji zapisu/odczytu wykonywanych za pomocą I²C. Typowym rozwiązaniem stosowanym przez producentów układów mających ten interfejs jest wyposażenie ich w możliwość adresacji rejestrów czy też komórek pamięci do/z których informacje chcemy pozyskać lub do których to informacje chcemy zapisać. Zwykle odbywa się to w ten sposób, iż po wysłaniu sekwencji *Start* oraz sprzętowego adresu danego układu (w trybie zapisu lub odczytu), następuje wysłanie adresu, spod którego dane chcemy odczytać lub pod który to dane chcemy zapisać, po czym następuje sekwencyjny zapis/odczyt interesujących nas danych przy automatycznej inkrementacji licznika adresów wykonywanej przez wspomniane peryferium. W przypadku układu Si4703 (jak i całej rodziny Si47xx) sytuacja wygląda zgoła inaczej, gdyż producent nie przewidział możliwości adresacji rejestrów roboczych, co w nieznaczny sposób komplikuje procedury odczytu jak i zapisu. W rozwiązaniu firmy Silicon Labs zakłada się, iż dla operacji odczytu pierwszym adresem, spod jakiego zostaną odczytane dane **jest zawsze** starszy bajt rejestru 0x0A (STATUSRSSI), po nim następuje odczyt młodszego bajta tegoż rejestru i sekwencja powtarza się dalej przez kolejne adresy rejestrów konfiguracyjnych aż do młodszego bajta rejestru 0x0F (RDSD), po czym następuje automatyczne wyzerowanie wewnętrznego licznika adresów i odczyt „rusza" od starszego bajta rejestru 0x00 (DEVICEID), pamiętamy wszak, iż rejestry tego układu są 16-bitowe. W przypadku operacji zapisu jest jeszcze inaczej. Zapis taki „rusza" **zawsze** od starszego bajta rejestru 0x02 (POWERCONFIG), następnie zapisywany jest młodszy bajt tegoż rejestru i sekwencja powtarza się dalej przez kolejne adresy rejestrów konfiguracyjnych aż do młodszego bajta rejestru 0x0F (RDSD) po czym następuje automatyczne wyzerowanie wewnętrznego licznika adresów i zapis „rusza" od starszego bajta rejestru 0x00 (DEVICEID). Na listingach **List.1** i **List.2** przedstawiono kod funkcji odpowiedzialnych za odczyt i zapis grupy rejestrów konfiguracyjnych układu Si4703 (wykorzystano globalna tablicę uint16_t Registers[16] przechowującą bieżące wartości poszczególnych rejestrów).

Opis wybranych rejestrów konfiguracyjnych

Przejdźmy, zatem do specyfikacji poszczególnych, wybranych rejestrów konfiguracyjnych, których modyfikacje są niezbędne z punktu widzenia funkcjonalności naszego urządzenia. Zaczniemy od rejestru *POWERCONFIG*, który umożliwia dokonanie podstawowej konfiguracji pracy układu Si4703 jak i jego aktywacji/dezaktywacji.

Znaczenie poszczególnych bitów rejestru ***POWERCONFIG*** przedstawia się następująco:

DSMUTE: wyzerowanie tego bitu (ustawienie domyślne) zezwala na programowe sterowanie funkcją wyciszenia dźwięku (sygnału wyjściowego),

DMUTE: wyzerowanie tego bitu (ustawienie domyślne) wycisza dźwięk,

MONO: wyzerowanie tego bitu (ustawienie domyślne) umożliwia stereofoniczny odbiór transmisji FM. Ustawienie tego bitu wymusza odbiór monofoniczny,

RDSM: wyzerowanie tego bitu (ustawienie domyślne) wymusza standardowy tryb pracy wbudowanego dekodera RDS. Ustawienie tego bitu wymusza tryb rozszerzony. W trybie podstawowym układ Si4703 ustawia (na czas minimum 40ms) flagę odebrania

POWERCONFIG (0x02)																
Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Symbol	DSMUTE	DMUTE	MONO	0	RDSM	SKMODE	SEEKUP	SEEK	0	DISABLE	0	0	0	0	0	ENABLE

wiadomości RDS (bit RDSR w rejestrze STATUSRSSI) wyłącznie wtedy, gdy wiadomość taka nie zawierała jakichkolwiek błędów lub zawierała błędy w takiej liczbie (maksymalnie 5), które można było z powodzeniem poddać algorytmom korekcji. W trybie rozszerzonym, flaga ta ustawiana jest zawsze, niezależnie od liczby błędów, jakie mogły pojawić się w takiej wiadomości, a których to ilość można każdorazowo odczytać z rejestrów BLERA…BLERD (odpowiednio dla rejestrów danych RDS: RDSA…RDSD) co pozwala na ewentualne, programowe procedowanie otrzymanych, błędnych wiadomości,

SKMODE: wartość tego bitu określa tryb pracy funkcji przeszukującej pasmo FM w poszukiwaniu sygnału stacji radiowej, tzw. Seek. Wyzerowanie tego bitu (ustawienie domyślne) powoduje, iż uruchomiona procedura Seek nie kończy swojego działania z chwilą napotkania na górną/dolną granicę pasma (jeśli wcześniej nie została znaleziona jakakolwiek stacja radiowa) tylko kontynuuje przeszukiwanie (od dolnej granicy w górę lub górnej granicy w dół w zależności od wybranego kierunku) aż do częstotliwości od jakiej rozpoczęło się przeszukiwanie. Ustawienie tego bitu wymusza zatrzymanie procedury Seek dla górnej/dolnej granicy przeszukiwanego pasma częstotliwości,

SEEKUP: wartość tego bitu określa kierunek przeszukiwania funkcji Seek (0: przeszukiwanie w dół, 1: przeszukiwanie w górę),

SEEK: ustawienie tego bitu inicjuje procedurę przeszukiwania pasma FM w celu znalezienia stacji radiowej. Procedura ta kończy swoje działanie z chwilą znalezienia stacji radiowej, przeszukania całego pasma zakończonego niepowodzeniem (SKMODE=0) lub napotkania dolnej/górnej granicy pasma (SKMODE=1). Efektem działania tejże procedury jest ustawienie odpowiednich flag w rejestrze STATUSRSSI (flagi STC i SF/BL). Flagi te muszą zostać wyzerowane przed ponownym uruchomieniem funkcji Seek lub Tune, co osiąga się poprzez wyzerowanie bitu SEEK,

DISABLE: ustawienie tego bitu wymusza dezaktywację układu Si4703 i przejście do trybu o obniżonym poborze mocy PowerDown. Niezbędne jest zachowanie odpowiedniej kolejności sekwencji sygnałów/rozkazów sterujących,

ENABLE: ustawienie tego bitu wymusza aktywację układu Si4703. Niezbędne jest zachowanie odpowiedniej kolejności sekwencji sygnałów/rozkazów sterujących.

Rejestr ***CHANNEL*** przeznaczony jest do przeprowadzania operacji przestrajania odbiornika FM.

Znaczenie poszczególnych bitów rejestru ***CHANNEL*** przedstawia się następująco:

TUNE: ustawienie tego bitu inicjuje procedurę przestrojenia odbiornika FM. Efektem działania tejże procedury jest przestrojenie układu Si4703 oraz ustawienie flagi STC w rejestrze STATUSRSSI. Po zakończeniu strojenia należy wyzerować bit STC, aby możliwe było kolejne uruchomienie funkcji Seek lub Tune,

CHANNEL[9:0]: wartość tych bitów określa żądaną częstotliwość strojenia radioodbiornika FM dla procedury Tune, którą to dla ustawień regionalnych naszego urządzenia zawartych w rejestrze SYSCONFIG2 oblicza się według następującego wzoru: f = 87.5+ 0.1* CHANNEL [MHz].

Rejestr ***SYSCONFIG1*** służy do konfiguracji podstawowych właściwości sprzętowych i ustawień regionalnych układu Si4703.

Znaczenie poszczególnych bitów rejestru ***SYSCONFIG1*** przedstawia się następująco:

RDSIEN: ustawienie tego bitu (domyślnie wykasowany) aktywuje przerwanie od gotowości danych RDS radioodbionika,

STCIEN: ustawienie tego bitu (domyślnie wykasowany) aktywuje przerwanie od zakończenia procedur Seek i Tune radioodbionika,

RDS: ustawienie tego bitu (domyślnie wykasowany) uruchamia dekoder RDS układu Si4703,

DE: ustawienie tego bitu (domyślnie wykasowany) determinuje parametry układu deemfazy radioodbiornika FM (0: 75μs, 1: 50μs),

AGCD: wyzerowanie tego bitu (ustawienie domyślne) załącza układ AGC radioodbiornika (Automatic Gain Control),

BLNDADJ[1:0]: wartość tych bitów określa poziom sygnału radiowego, dla którego załączony zostanie odbiór sygnału stereofonicznego (domyślnie 31…49 RSSI dBμV),

GPIO3[1:0]: wartość tych bitów określa funkcjonalność wyjścia GPIO3 układu Si4703 według następującej specyfikacji: [0:0] – stan wysokiej impedancji (ustawienie domyślne), [0:1] – wskaźnik odbioru sygnału stereofonicznego (stan wysoki na wspomnianym wyjściu w przypadku odbioru audycji stereofonicznej), [1:0] – stan niski na wyjściu, [1:1] – stan wysoki na wyjściu,

GPIO2[1:0]: wartość tych bitów określa funkcjonalność wyjścia GPIO2 układu Si4703 według następującej specyfikacji: [0:0] – stan wysokiej impedancji (ustawienie domyślne), [0:1] – wskaźnik przerwania RDS/STC (stan niski o czasie trwania ok.5ms na wspomnianym wyjściu w przypadku wystąpienia przerwania RDS/STC, w przeciwnym wypadku stan wysoki), [1:0] – stan niski na wyjściu, [1:1] – stan wysoki na wyjściu,

GPIO1[1:0]: wartość tych bitów określa funkcjonalność wyjścia GPIO1 układu Si4703 według następującej specyfikacji: [0:0] – stan wysokiej impedancji (ustawienie domyślne), [0:1] – Zarezerwowane, [1:0] – stan niski na wyjściu, [1:1] – stan wysoki na wyjściu,

Rejestr ***SYSCONFIG2*** służy do konfiguracji ustawień regionalnych układu Si4703 oraz niektórych właściwości użytkowych.

Znaczenie poszczególnych bitów rejestru ***SYSCONFIG2*** przedstawia się następująco:

SEEKTH[7:0]: wartość tych bitów określa minimalny poziom sygnału antenowego brany pod uwagę w trakcie działania funkcji Seek. Wartość domyślna: minimum RSSI,

BAND[1:0]: wartość tych bitów określa szerokość pasma FM radioodbiornika według następującej specyfikacji: [0:0] – 87.5…108MHz, [0:1] – 76…108MHz, [1:0] – 76…90MHz, [1:1] – Zarezerwowane,

SPACE[1:0]: wartość tych bitów określa odstęp kanałów radioodbiornika według następującej specyfikacji: [0:0] – 200kHz, [0:1] – 100kHz, [1:0] – 50kHz, [1:1] – Zarezerwowane,

VOLUME[3:0]: wartość tych bitów określa poziom głośności wyjściowego sygnału audio radioodbiornika. Wartość domyślna: 0x00 (Mute).

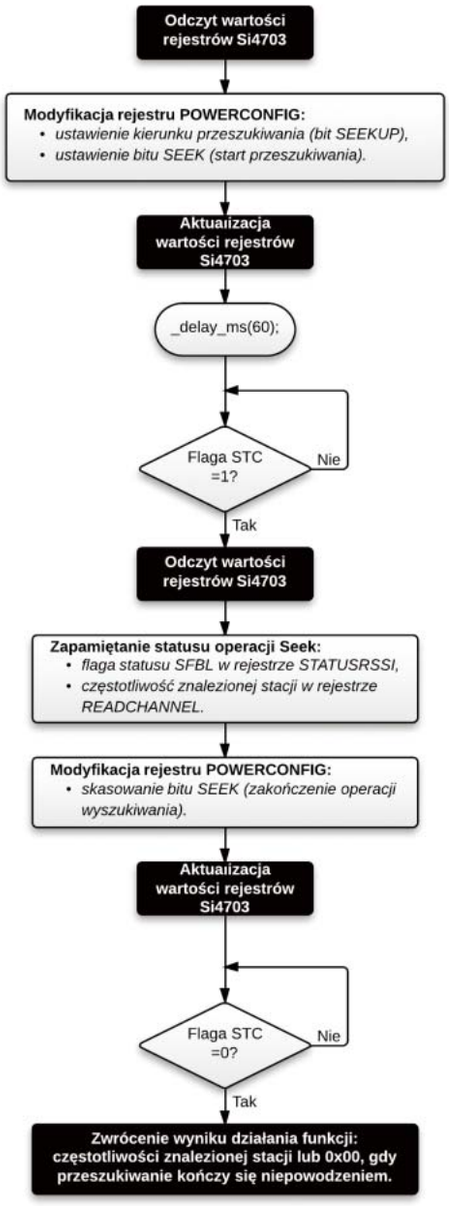
Rejestr ***SYSCONFIG3*** służy do konfiguracji właściwości audio oraz progów działania funkcji Seek.

Znaczenie poszczególnych bitów rejestru ***SYSCONFIG3*** przedstawia się następująco:

SMUTER[1:0]: wartość tych bitów określa szybkość wyciszania/pogłaśniania sygnału audio dla funkcji SoftMute według następującej specyfikacji: [0:0] – najszybsza, [0:1] – szybka, [1:0] – wolna, [1:1] – najwolniejsza,

CHANNEL (0x03)																
Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Symbol	TUNE	0	0	0	0	0	CHANNEL [9:0]									

SYSCONFIG1 (0x04)																
Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Symbol	RDSIEN	STCIEN	0	RDS	DE	AGCD	0	0	BLNDADJ [1:0]		GPIO3 [1:0]		GPIO2 [1:0]		GPIO1 [1:0]	



Rysunek 4. Graf procedury odpowiedzialnej za przeszukiwanie pasma FM układu Si4703

SMUTEA[1:0]: wartość tych bitów określa wzmocnienie sygnału audio dla funkcji SoftMute według następującej specyfikacji: [0:0] – 16dB, [0:1] – 14dB, [1:0] – 12dB, [1:1] – 10dB,

VOLEXT: ustawienie tego bitu włącza tryb rozszerzonej regulacji głośności wyjściowego sygnału audio,

SKSNR[3:0]: wartość tych bitów określa minimalny poziom parametru SNR sygnału antenowego (odstęp sygnału od szumu), dla którego funkcja Seek determinuje odbierany sygnał jako ważną transmisję radiową według następującej specyfikacji: 0 – funkcja wyłączona, 1 – poziom minimalny (największa liczba detekcji), 8 – poziom maksymalny (najmniejsza liczba detekcji),

SKCNT[3:0]: wartość tych bitów określa minimalną liczbę detektowanych impulsów transmisji FM, dla którego funkcja Seek determinuje odbierany sygnał jako ważną transmisję radiową według następującej specyfikacji: 0 – funkcja wyłączona, 1 – poziom minimalny (największa liczba detekcji), 15 – poziom maksymalny (najmniejsza liczba detekcji),

Rejestr ***TEST1*** jest kluczowym rejestrem konfiguracyjnym, jeśli chodzi o obwód oscylatora wbudowanego w strukturę układu Si4703 zapewniający sygnał zegarowy o częstotliwości 32768Hz.

Znaczenie poszczególnych bitów rejestru ***TEST1*** przedstawia się następująco:

XOSCEN: ustawienie tego bitu (domyślnie wykasowany) uruchamia wewnętrzny oscylator 32768Hz, przy czym w takim przypadku niezbędne jest dołącznie do układu Si4703 zewnętrznego rezonatora kwarcowego o takiej częstotliwości (do wyprowadzeń RCLK i GPIO3). Oscylator ten musi być uruchomiony przed aktywowaniem układu Si4703, na wstępie procedury inicjalizacyjnej. Czas stabilizowania się parametrów pracy tak zbudowanego oscylatora wynosi ok.500ms,

AHIZEN: ustawienie tego bitu (domyślnie wykasowany) powoduje odłączenie od wyjść audio wewnętrznych rezystorów zapewniających ich polaryzację potencjałem równym połowie napięcia zasilającego.

Rejestr ***STATUSRSSI*** jest kluczowym rejestrem z punktu widzenia monitorowania pracy układu Si4703, gdyż zawiera szereg flag sprzętowych (przeznaczony jest wyłącznie do odczytu).

Znaczenie poszczególnych bitów rejestru ***STATUSRSSI*** przedstawia się następująco:

RDSR: bit ten jest flagą nowej wiadomości RDS,

STC: bit ten jest flagą sygnalizującą zakończenie operacji typu Seek/Tune (przeszukiwania lub strojenia),

SF/BL: bit ten określa status ostatnio przeprowadzonej operacji typu Seek według następującej specyfikacji: 0 – operacja zakończona powodzeniem (znaleziono stację radiową), 1 – operacja zakończona niepowodzeniem (SKMODE=0) lub osiągnięto limit (dolny/górny) pasma FM (SKMODE=1),

AFCRL: wartość tego bitu uaktualniana jest po każdej operacji typu Seek lub Tune. Ustawienie tego bitu informuje o znalezieniu użytecznego sygnału radiowego (o odpowiednich wartościach RSSI i SNR) o offsecie częstotliwości spoza dozwolonego zakresu nośnej (typowy offset dla FM = 30kHz),

RDSS: bit ten odzwierciedla stan dekodera RDS w trybie rozszerzonym według następującej specyfikacji: 0 – dekoder RDS niezsynchronizowany, 1 - dekoder RDS zsynchronizowany (szczegóły w dokumentacji trybu rozszerzonego, który nie jest wspierany przez urządzenie pocketRadio),

BLERA[1:0]: wartość tych bitów określa liczbę błędów, jakie pojawiły się w pierwszym bloku bieżącej wiadomości RDS (rejestr RDSA układu Si4703) dla trybu rozszerzonego według następującej specyfikacji: [0:0] – brak błędów, [0:1] – 1 do 2 błędów wymagających korekcji, [1:0] – 3 do 5 błędów wymagających korekcji, [1:1] – 6 i więcej błędów w związku, z czym korekcja nie jest możliwa,

ST: bit ten jest flagą sygnału stereofonicznego (ustawiony, gdy odbierana audycja nadaje dźwięk stereofoniczny),

RSSI[7:0]: bity te określają poziom sygnału antenowego, przy czym nie są raportowane wartości większe niż 75 dBμV (75dec).

Rejestr ***READCHANNEL*** jest rejestrem informacyjnym przeznaczonym wyłącznie do odczytu a przechowującym informacje o stopie błędów wiadomości RDS jak i częstotliwości, na jakiej pracuje odbiornik FM.

Znaczenie poszczególnych bitów rejestru ***READCHANNEL*** przedstawia się następująco:

BLERB[1:0]: wartość tych bitów określa liczbę błędów, jakie pojawiły się w drugim bloku bieżącej wiadomości RDS (rejestr RDSB układu Si4703) dla trybu rozszerzonego według następującej specyfikacji: [0:0] – brak błędów, [0:1] – 1 do 2 błędów wymagających korekcji, [1:0] – 3 do 5 błędów

SYSCONFIG2 (0x05)

Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Symbol	SEEKTH[7:0]								BAND [1:0]		SPACE [1:0]		VOLUME [3:0]			

SYSCONFIG3 (0x06)

Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Symbol	SMUTER[1:0]		SMUTEA[1:0]		0	0	0	VO-LEXT	SKSNR[3:0]				SKCNT[3:0]			

TEST1 (0x07)

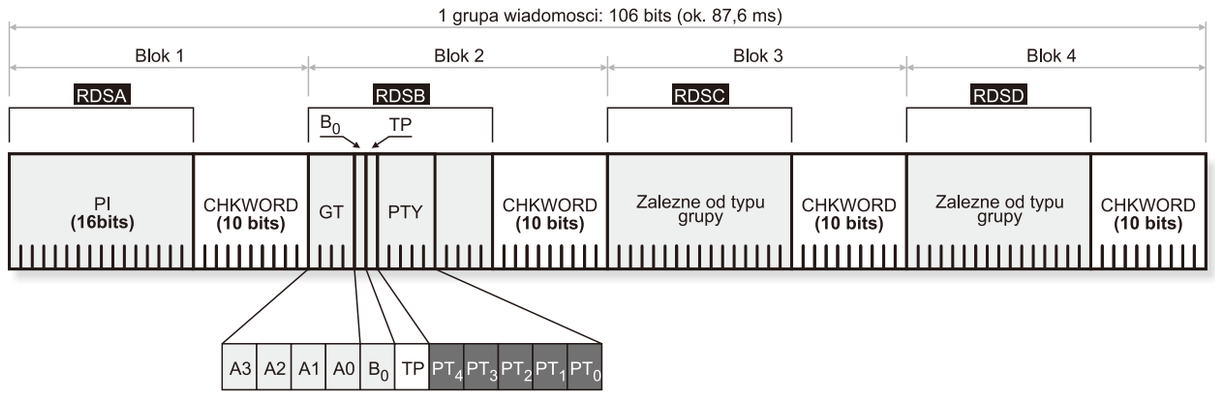
Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Symbol	XO-SCEN	AHI-ZEN	ZAREZERWOWANY													

STATUSRSSI (0x0A)

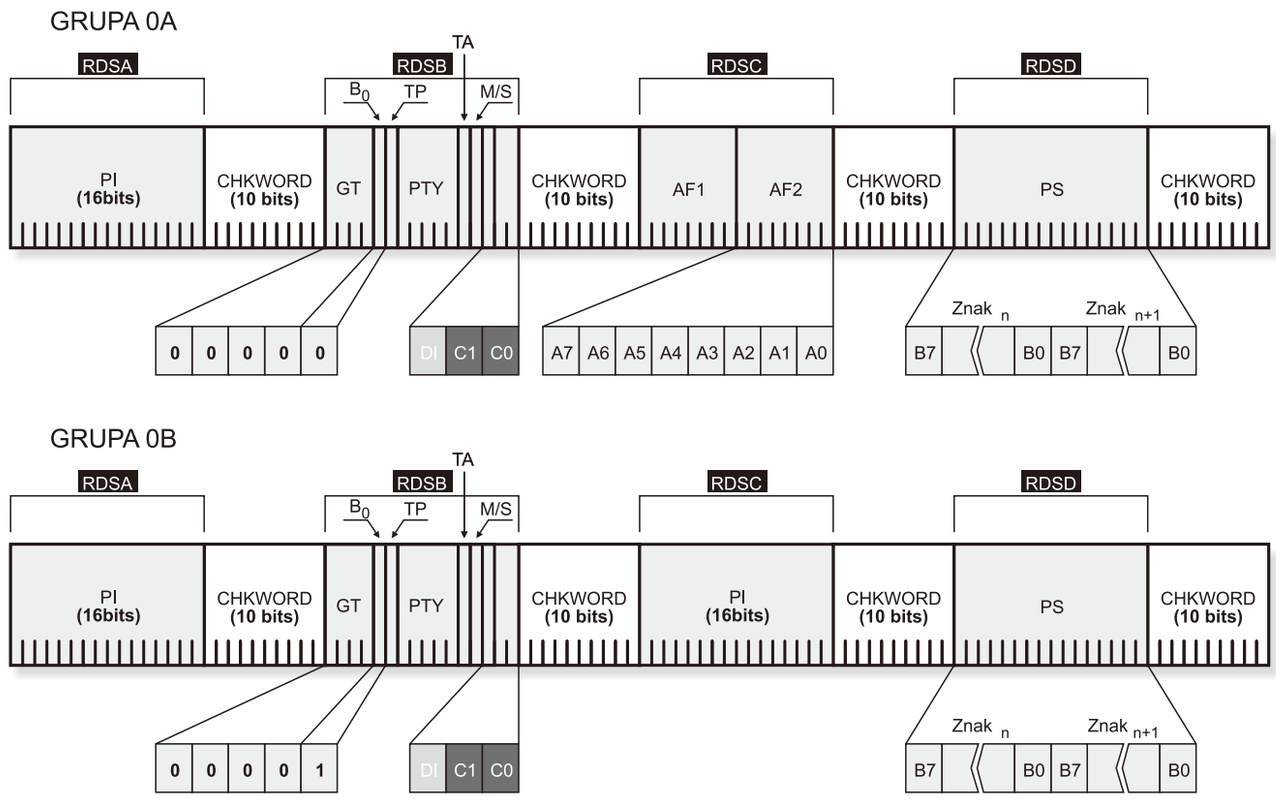
Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Symbol	RDSR	STC	SF/BL	AF-CRL	RDSS	BLERA[1:0]		ST	RSSI[7:0]							

READCHANNEL (0x0B)

Bit	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
Symbol	BLERB[1:0]		BLERC[1:0]		BLERD[1:0]		READCHAN[9:0]									



Rysunek 5. Wygląd kompletnej grupy danych RDS z zaznaczeniem charakterystycznych struktur



Rysunek 6. Struktura danych dla grup wiadomości typu 0A/0B

wymagających korekcji, [1:1] – 6 i więcej błędów w związku, z czym korekcja nie jest możliwa,

BLERC[1:0]: wartość tych bitów określa liczbę błędów, jakie pojawiły się w trzecim bloku bieżącej wiadomości RDS (rejestr RDSC układu Si4703) dla trybu rozszerzonego według specyfikacji jak powyżej,

BLERD[1:0]: wartość tych bitów określa liczbę błędów, jakie pojawiły się w czwartym bloku bieżącej wiadomości RDS (rejestr RDSD układu Si4703) dla trybu rozszerzonego według specyfikacji jak powyżej,

READCHAN[9:0]: wartość tych bitów określa bieżącą częstotliwość, na jaką nastrojony jest radioodbiornik FM, któ rą to dla ustawień regionalnych naszego urządzenia zawartych w rejestrze SYSCONFIG2 oblicza się według następującego wzoru: f = 87.5+ 0.1* CHANNEL [MHz].

Na tym zakończymy opis rejestrów konfiguracyjnych układu Si4703. Przykłady praktycznego ich zapisu i odczytu zawierają procedury inicjalizacyjne zamieszczone na **listingu 3**. Do „pełni szczęścia" brakuje nam procedur odpowiedzialnych za przestrojenie odbiornika oraz wyszukiwanie sygnału stacji radiowej. Graf funkcjonalny procedury odpowiedzialnej za przestrojenie układu Si4703 pokazano na **rysunku 3**, natomiast funkcję na **listingu 4.** Na **rysunku 4** przedstawiono graf funkcjonalny procedury odpowiedzialnej za przeszukiwanie pasma FM, a funkcję pokazano na **listingu 5**.

Podczas opisu rejestrów konfiguracyjnych kilkakrotnie wspominano o rejestrach RDSA…RDSD, które przechowują zawartość kolejnych bloków wiadomości RDS. Nie podano przy tym jakiejkolwiek specyfikacji formatu informacji, która jest tam przechowywana w wyniku odebrania danych RDS. Aby w pełni zrozumieć to zagadnienie należy choćby ogólnie zapoznać się ze standardem RDS.

Europejski system RDS (*Radio Data System*), odpowiednik amerykańskiego systemu RDBS, jest systemem do transmisji cyfrowych danych dodatkowych w radiofonii UKF na trzeciej podnośnej (57 kHz) pilota sygnału stereofonicznego (19 kHz). System ten w swoim pierwotnym założeniu przeznaczony był dla użytkowników odbiorników samochodowych, dla których ciągła zmiana położenia, a co za tym idzie siły sygnału, powodowała konieczność odbioru słuchanego programu z innego nadajnika tej samej stacji radiowej (a więc konieczność przestrajania odbiornika). Dla ich wygody system ten miał dostarczać informacje o nazwie i rodzaju nadawanego programu oraz usługach dodatkowych, takich jak informacje drogowe czy informacje o zagrożeniach. Równocześnie miała być zachowana kompatybilność sygnału dla starszych odbiorników radiowych, niemających dekodera RDS. Uzyskano ją dzięki wprowadzeniu dodatkowej podnośnej leżącej na częstotliwości 57 kHz (+/- 2,5kHz), na której transmitowany jest zmodulowany fazowo (BPSK), ciągły strumień danych RDS o przepływności 1187,5 bit/s (1/48 częstotliwości podnośnej).

Dane są zorganizowane w grupy składające się z 4 bloków, każdy po 26 bitów informacji. Pojedynczy blok zawiera 16-bitowe słowo przenoszące właściwą informację oraz 10 bitów danych korekcyjnych, przeznaczonych do sprzętowej korekcji błędów i synchronizacji bloków oraz grup danych. Wygląd kompletnej grupy danych RDS z zaznaczeniem charakterystycznych struktur pokazano na **rysunku 5**. Każda grupa danych zawiera 4 bloki użytecznej informacji (zaznaczono odpowiednie rejestry układu Si4703 tj. RDSA…RDSD). Dwa pierwsze bloki (Blok1…Blok2) zawierają podstawowe struktury danych charakterystyczne dla konkretnej stacji radiowej oraz bieżącej informacji, która jest przez nią transmitowana jak i identyfikator grupy danych (bity A3…A0 oraz B_0), który determinuje zawartość pozostałych bloków danych (Blok3…Blok4). Znaczenie poszczególnych struktur danych jest następujące:

PI (Program Identification) jest zapisanym szesnastkowo, unikatowym kodem identyfikacyjnym stacji radiowej. Pierwsza cyfra informuje o kraju, z którego nadawany jest program (3: Polska, 5: Słowacja, 2: Czechy, D: Niemcy, F: Francja), druga mówi o zasięgu (0: program lokalny, 1: międzynarodowy, 2: ogólnokrajowy, 3: ponadregionalny, 4…F: regionalny). Pozostałe dwie zawierają inne informacje (np. numer programu). Na podstawie kodu PI odbiornik radiowy identyfikuje stację radiową, co w powiązaniu z funkcją AF powoduje, iż odbiornik taki nie przełączy się na nową częstotliwość, jeżeli kody obu (starej i nowej) nie pokrywają się. W związku z powyższym kod PI należy do najczęściej nadawanych struktur w sygnale RDS - jest powtarzany co najmniej 11 razy na sekundę (występuje w każdym bloku Blok1).

GT (Group Type) określa typ danych bieżącej grupy danych determinując tym samym znaczenie informacji zawartej w blokach danych Blok3…Blok4 oraz częściowo w bloku Blok2. Najczęściej spotykane grupy danych (spośród 32 typów) to typy: 0A/0B (Basic Tuning and Switching Information only) niosące informacje o nazwie stacji radiowej (PS: Program Service) oraz o alternatywnych częstotliwościach stacji radiowej (AF: Alternative Frequencies), 2A/2B (RT: Radio Text only) przeznaczone do przenoszenia informacji tekstowych oraz 4A (CT: Clock

Listing 1. Funkcja realizująca odczyt rejestrów konfiguracyjnych układu Si4703

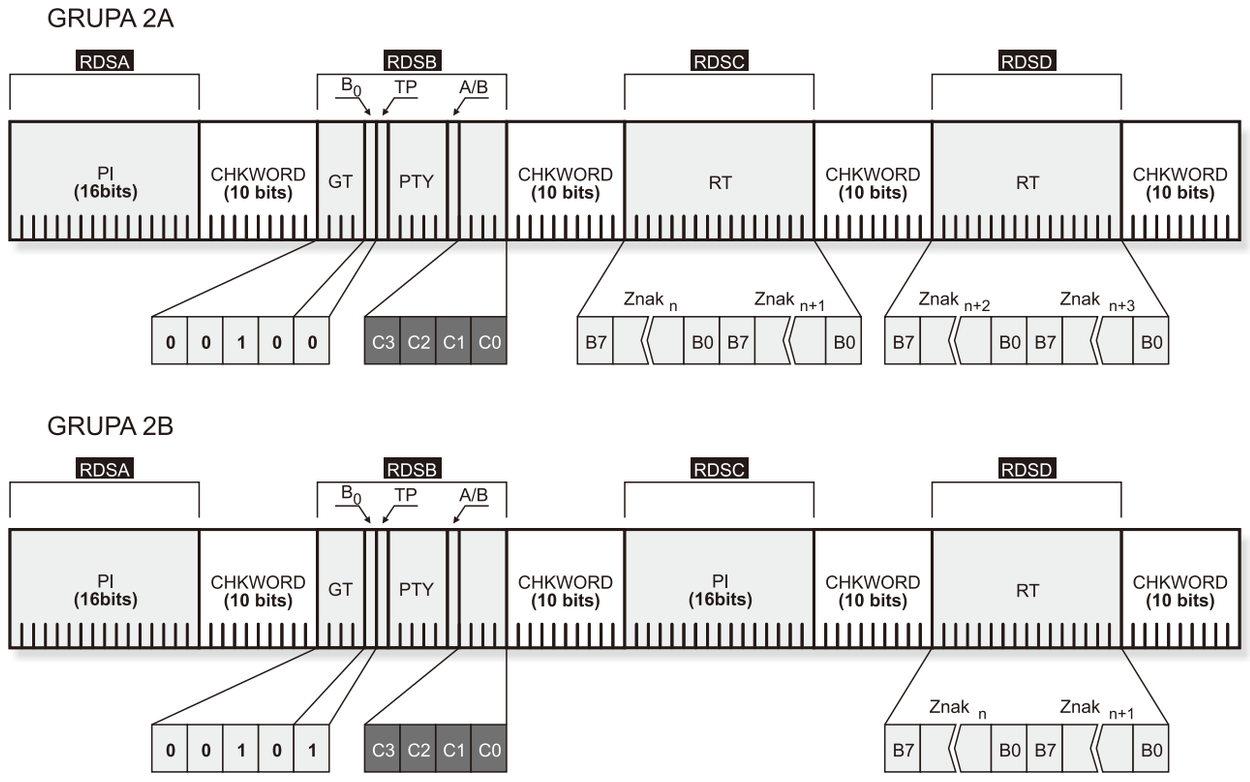
```
void readRadioRegisters(void)
{
  register uint8_t reg;
  TWI_Start();
  TWI_WriteByte(SI4703RD_ADDR);  //Adres układu w trybie odczytu
  for(reg = 0x0A ; ; reg++)
  {
    if(reg == 0x10) reg = 0x00; //Zaczynamy od rejestru 0x00
    Registers[reg] = TWI_ReadInteger(NACK);
    if(reg == 0x09) break; //Odczytano wszystkie rejestry Si4703
  }
  TWI_Stop();
}
```

Listing 2. Funkcja realizująca zapis do rejestrów konfiguracyjnych układu Si4703

```
/* Rejestry 0x00 i 0x01 są przeznaczone tylko do odczytu.
Nie interesują nas, gdyż zawierają jedynie ID układu. */
void updateRadioRegisters(void)
{
  register uint8_t reg;
  TWI_Start();
  TWI_WriteByte(SI4703WR_ADDR);  //Adres układu w trybie zapisu
  for(reg = 0x02 ; reg<0x10 ; reg++) TWI_WriteInteger(Registers[reg]);
  TWI_Stop();
}
```

Listing 3. Funkcja inicjalizacji i konfiguracji układu Si4703

```
void initRadio(uint8_t Volume)
{
  RADIO_CTRL_PORT |= (1<<SCLK_PIN);  //Ustawienie SCLK=1 (nota aplikacyjna,
str.20, Tab.9, nota (2))
  RADIO_CTRL_DDR |= ((1<<RES_PIN) | (1<<SDIO_PIN) | (1<<SCLK_PIN));  //RES=0,
SDIO=0, SCLK=1
  _delay_us(100);
  RADIO_CTRL_PORT |= (1<<RES_PIN);  //RES=1, SDIO=0, SEN=1 (rezystor)->wybór
magistrali I2C układu Si4703 (nota aplikacyjna, str.19)
  _delay_us(100);
  TWI_Init();  //Inicjalizacja magistrali I2C mikrokontrolera (wyłącznie
ustawienie prędkosci transmisji)
  readRadioRegisters();  //Odczyt aktualnych wartosci wszystkich rejestrów
konfiguracyjnych układu Si4703
  Registers[TEST1_REG] |= (1<<XOSCEN);  //Uruchomienie oscylatora 32768Hz,
AN230 str.5 i 12
  updateRadioRegisters();  //Aktualizacja rejestrów konfiguracyjnych układu
Si4703
  _delay_ms(500);  //Niezbędny czas na start i stabilizację oscylatora, AN230
str.5 i 12
  readRadioRegisters();  //Odczyt aktualnych wartosci wszystkich rejestrów
konfiguracyjnych układu Si4703
  //Wlasciwa konfiguracja wszystkich parametrów układu Si4703 poza wartosciami
domyslnymi, które odpowiadają wymaganiom aplikacji
  //Errata, solution 2, AN230 str.5 i 12
  Registers[RDSD_REG] = 0x00;
  //Wyłączenie funkcji Mute oraz uruchomienie układu Si4703 (sekwencja
PowerUp, AN230 str.6)
  Registers[POWERCFG_REG] |= ((1<<DMUTE) | (1<<ENABLE));
  //Włączenie przerwania od RDS, włączenie RDS, de-emphasis=50uS (Europa),
aktywacja przerwania RDS na GPIO2 (stan niski jako aktywny)
  Registers[SYSCONFIG1_REG] |= ((1<<RDSIEN) | (1<<RDS) | (1<<DE) |
(1<<GPIO20));
  //Channel spacing=100kHz (Europa), bity 3..0 to głosnosc czyli parametr
wywołania funkcji
  Registers[SYSCONFIG2_REG] &= 0xFFF0;  //Wyczyszczenie bitów głosnosci
  Registers[SYSCONFIG2_REG] |= (Volume & 0x0F);  //Ustawienie głosnosci
  Registers[SYSCONFIG2_REG] |= (1<<SPACE0);
  //Konfiguracja  funkcji Seek: ustawienie poziomu sygnału dla detekcji ważnej
stacji  (poziom RSSI) oraz liczby impulsów FM
  Registers[SYSCONFIG3_REG] &= 0xFF00;  //Wyczyszczenie bitów: Seek SNR
Threshold (7..4) i Seek FM Impulse Detection Threshold (3...0)
  Registers[SYSCONFIG3_REG] |= 0x0040;  //Ustawienie Seek SNR Threshold = 4
  Registers[SYSCONFIG3_REG] |= 0x0008;  //Ustawienie Seek FM Impulse Detection
Threshold = 8
  updateRadioRegisters();  //Aktualizacja rejestrów konfiguracyjnych układu
Si4703 (w tym uruchomienie układu Si4703)
  _delay_ms(110);  //Niezbędny czas na uruchomienie układu Si4703  (nota
aplikacyjna, str.13, Tab.8)
  RADIO_IRQ_PORT |= (1<<IRQ_PIN);  //Podciągnięcie wejscia IRQ mikrokontrolera
do plusa zasilania (wyjscie GPIO2 Si4703)
  EICRA |= (1<<ISC01);  //Konfiguracja przerwania INT0: opadające zbocze na
wejsciu INT0
  EIMSK |= (1<<INT0);  //Uruchomienie przerwania INT0
}
```



Rysunek 7. Struktura danych dla grup wiadomości typu 2A/2B

Listing 4. Kod funkcji odpowiedzialnej za przestrojenie układu Si4703

```
/* Funkcja ustawia zadaną argumentem wywołania częstotliwość: zakres 875...1080 odpowiada częstotliwości 87.5...108 MHz
(krok 100kHz). */

void setRadioFrequency(uint16_t Frequency)
{
  uint16_t channel;
/* Zablokowanie przerwania INT0 by w czasie manipulacji na rejestrach układu Si4703 (dzięki transmisji I2C) nie doszło
do wyzwolenia tego przerwania, które także korzysta ze wspomnianej magistrali i mogłoby powodować błędy w działaniu
funkcji obsługi radia. */
  EIMSK &= ~(1<<INT0);
  channel = Frequency - 875;  //Obliczamy żądany kanał według specyfikacji układu Si4703 i ustawień regionalnych
  readRadioRegisters();  //Odczyt aktualnych wartosci wszystkich rejestrów konfiguracyjnych układu Si4703
  Registers[CHANNEL_REG] &= 0xFC00;  //Wyczyszczenie bitów odpowiedzialnych za żądaną częstotliwosc
  Registers[CHANNEL_REG] |= channel;  //Ustawienie nowego kanału
  Registers[CHANNEL_REG] |= (1<<TUNE);  //Start przestrajania
  updateRadioRegisters();
  _delay_ms(60);  //nota aplikacyjna, str.13, Tab.8 (można pominąc)
//Czekamy na ustawienie flagi STC w rejestrze STATUSRSSI sygnalizującej zakończenie strojenia
  do readStatusRegister(); while((Registers[STATUSRSSI_REG] & (1<<STC)) == 0);
  readRadioRegisters();  //Odczyt aktualnych wartosci wszystkich rejestrów konfiguracyjnych układu Si4703
  Registers[CHANNEL_REG] &= ~(1<<TUNE);  //Skasowanie flagi TUNE po zakończeniu strojenia - wymóg dokumentacji, AN230
str.22
  updateRadioRegisters();
//Czekamy na wyzerowanie flagi STC w rejestrze STATUSRSSI wymuszone skasowaniem flagi TUNE (powyżej)
  do readStatusRegister(); while( (Registers[STATUSRSSI_REG] & (1<<STC)) == 1);
  EIMSK |= (1<<INT0);  //Odblokowanie przerwania INT0
}
```

Time and Date only) przeznaczony do transmisji znacznika czasu.

B0 to bit determinujący wersję typu grupy danych (A lub B, stąd 32 typy grup danych).

TP (Traffic Program Information Code) jest znacznikiem informującym, iż stacja radiowa emituje od czasu do czasu informacje drogowe. Pojawienie się dodatkowego znacznika TA (Traffic Announcement) powoduje automatyczne przełączenie się takiego odbiornika z trybu TAPE lub CD na emisję radiową na czas trwania wspomnianych komunikatów. Dodatkowo, niektóre odbiorniki mogą też samoczynnie zmieniać głośność, a nawet brzmienie dźwięku w taki sposób, aby komunikat taki był bardziej zrozumiały.

PTY (Program Type Code) określa predefiniowany w standardzie RDS rodzaj programu (np. NEWS, TRAFFIC, EDUCAT, CULTUR itd.).

Specyfikacja RDS przewiduje 32 typy grup wiadomości (a dokładnie 16 typów, każdy w wersji A oraz B), które to determinują znaczenie informacji zawartej w blokach danych Blok2 (ostatnie 5 bitów bloku) oraz Blok3…Blok4. Warto zauważyć, iż częstotliwość powtarzania informacji dla każdej z grup danych jest inna i zależy od przeznaczenia załączonych danych. Co więcej, w ramach ciągłego strumienia danych możliwe jest transmitowanie różnych grup danych jedna po drugiej nawet jeśli oczekiwany jest dla przykładu spójny ciąg danych reprezentujący nazwę stacji (Program Service). Aby ułatwić dekodowanie sygnału RDS dla tego typu transmisji, wprowadzono dodatkowe bity synchronizujące, które to określają numer bieżącego segmentu danych (np. dla grupy 0A/0B czy 2A/2B).

Dokładne omówienie konstrukcji bloków danych dla każdego z 32 typów danych wykraczałoby poza ramy tego artykułu. Tu skupimy się wyłącznie na konstrukcji bloków danych dla grup typu 0A/0B, 2A/2B oraz 4A, czyli grup, które są obsługiwane przez nasz odbiornik. Na **rysunku 6** pokazano strukturę danych dla grup wiadomości typu 0A/0B. Poza wymienionymi wcześniej strukturami danych, grupa ta zawiera następujące, dodatkowe informacje:

Listing 5. Kod funkcji odpowiedzialnej za przeszukiwane pasma FM układu Si4703

```
/* Funkcja przeszukująca pasmo radiowe w poszukiwaniu sygnału. Argument seekDirection określa kierunek przeszukiwania
(stałe w pliku h). Funkcja zwraca częstotliwość znalezionej stacji lub 0x00, jeśli żadna stacja nie została znaleziona
(przeszukiwane jest całe pasmo w górę lub w dół, po czym następuje przejście do początku/końca pasma, do punktu,
w którym rozpoczęło się wyszukiwanie - standardowe ust. */
uint16_t seekRadio(uint8_t seekDirection)
{
//Częstotliwosc znalezionej stacji Frequency = 875 +channel;
  uint16_t frequency;
  register uint8_t searchFlag, loops = 0;
//Zablokowanie przerwania INT0 by w czasie manipulacji na rejestrach układu Si4703 (dzięki transmisji I2C) nie doszło
do wyzwolenia
//tego przerwania, które także korzysta ze wspomnianej magistrali i mogłoby powodowac błędy w działaniu funkcji obsługi
radia
  EIMSK &= ~(1<<INT0);
//Odczyt aktualnych wartości wszystkich rejestrów konfiguracyjnych układu Si4703
  readRadioRegisters();
//Pozwolenie na przekraczanie granicy 108->87.5 itd (allow wrap)
  Registers[POWERCFG_REG] |= (1<<SKMODE);
//Ustawienie kierunku przeszukiwania - bit SEEKUP
  if(seekDirection == SEEK_DOWN_DIRECTION) Registers[POWERCFG_REG] &=~(1<<SEEKUP);
    else Registers[POWERCFG_REG] |= (1<<SEEKUP);
  Registers[POWERCFG_REG] |= (1<<SEEK);  //Start przeszukiwania
  updateRadioRegisters();
  _delay_ms(60);  //nota aplikacyjna, str.13, Tab.8 (można pominąc)
//Czekamy na ustawienie flagi STC w rejestrze STATUSRSSI sygnalizującej zakończenie bieżącego przeszukiwania
  do
  {
    readStatusRegister();
    loops++;
    _delay_ms(15);
  }while( ((Registers[STATUSRSSI_REG] & (1<<STC)) == 0) && loops != 255);
  readRadioRegisters();  //Odczyt aktualnych wartosci wszystkich rejestrów konfiguracyjnych układu Si4703
  searchFlag = Registers[STATUSRSSI_REG] & (1<<SFBL);  //Odczyt statusu operacji Seek (!=0 gdy nie znaleziono żadnej
stacji)
  if(searchFlag) frequency = 0; else frequency = 875 + (Registers[READCHAN_REG] & 0x03FF);  //Odczyt częstotliwosci
znalezionej stacji (rejestr READCHAN)
Registers[POWERCFG_REG] &=~(1<<SEEK);  //Skasowanie flagi SEEK po zakończeniu przeszukiwania - wymóg dokumentacji, AN230
str.20
  updateRadioRegisters();
  loops = 0;
//Czekamy na wyzerowanie flagi STC w rejestrze STATUSRSSI wymuszone skasowaniem flagi SEEK (powyżej)
  do
  {
    readStatusRegister();
    loops++;
    _delay_ms(10);
  } while( ((Registers[STATUSRSSI_REG] & (1<<STC)) == 1) && loops != 255);
  EIMSK |= (1<<INT0);  //Odblokowanie przerwania INT0
  return frequency;
}
```

TA (Traffic Announcement) jest znacznikiem umożliwiającym identyfikację stacji, która w danym momencie nadaje komunikat drogowy. Na podstawie tego znacznika, radioodbiornik ma możliwość automatycznego przełączenia się z trybu TAPE/CD na radio na czas jego trwania.

M/S (Music/Speech) jest znacznikiem, dzięki któremu radioodbiornik automatycznie dobiera głośności oraz barwę dźwięku w zależności od tego, czy jest nadawana audycja muzyczna, czy słowna.

DI (Decoder Information) jest informacją dla dekodera RDS służącą zmianie trybu jego pracy i niesie informację służącą dopasowaniu demodulatora odbiornika do odbieranego sygnału.

Bity C1…C0 określają index segmentu danych bieżącego bloku danych typu PS (nazwa stacji), dzięki czemu procedura dekodująca po stronie odbiornika jest w stanie w sposób prawidłowy zdekodować tego typu informację. Bity te w połączeniu z bitem DI określają również rodzaj trybu pracy dekodera RDS.

Bajty AF1…AF2 określają alternatywne częstotliwości stacji radiowej (z dostępnej, predefiniowanej listy częstotliwości).

Bajty PS zawierają kody ASCII kolejnych znaków nazwy stacji radiowej (Program Service), przy czym położenie tych znaków w nazwie stacji określają bity C1…C0. Wysyłane są zawsze 4 grupy typu 0A/0B nawet wtedy, gdy nazwa stacji jest w rzeczywistości krótsza niż 8 znaków (puste miejsca wypełniane są spacjami).

Z przytoczonych informacji widać również, że różnica pomiędzy typem 0A i 0B wynika wyłącznie z zawartości bloku danych Blok3, który dla typu 0B zawiera powtórzoną strukturę PI, a nie wartości częstotliwości alternatywnych jak to ma miejsce w przypadku typu 0A.

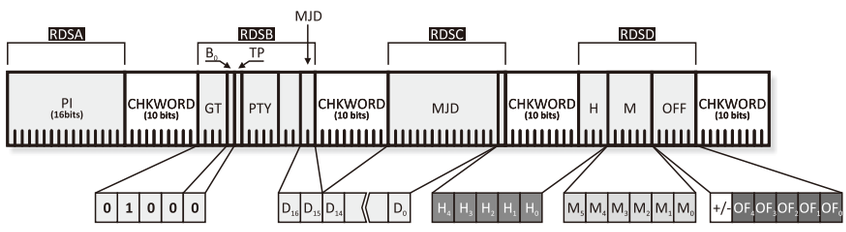
Strukturę typu 2A/2B pokazano na **rysunku 7**. Tym razem mamy do czynienia z dodatkowymi informacjami, które nie występowały w poprzednich typach grup danych, a mianowicie:

Bit A/B jest flagą, która informuje procedurę dekodującą o zmianie danych w wiadomości RT (Radio Text) rozumianej jako cały, przeznaczony do wyświetlenia tekst. Flaga ta pozwala radioodbiornikom niewyposażonym w dostateczną ilość wyświetlanych znaków na synchronizację procedur odpowiadających za wyświetlanie informacji (np. procedur przewijających tekst na ekranie). Flaga ta zmieniana jest za każdym razem, gdy zmianie ulega transmitowany tekst. Specyfikacja RDS zakłada nawet kilkukrotne powtarzanie tych samych grup danych, a co za tym idzie – kilkakrotne przesyłanie tej samej wiadomości RT.

Bity C3…C0 określają indeks segmentu danych bieżącego bloku danych typu RT (dowolny tekst), dzięki czemu procedura dekodująca po stronie odbiornika jest w stanie prawidłowo zdekodować informacje tego typu.

4 (dla typu 2A) lub 2 (dla typu 2B) bajty RT zawierające kody ASCII kolejnych znaków tekstu (Radio Text), przy czym położenie tych zna-

Ustawienia ważniejszych fusebitów:
CKSEL3…0: 0010
SUT1…0: 10
CKDIV8: 1
CKOUT: 1
JTAGEN: 1
OCDEN: 1
EESAVE: 0



Rysunek 8. Struktura danych dla grup wiadomości typu 4A

Listing 6. Procedura obsługi przerwania INT0 odpowiedzialna za odbiór i dekodowanie wiadomości RDS

```
/* Procedura ISR INT0 odpowiedzialna za obsługę wiadomosci RDS. Komplet danych RDS (4, 16-bit. rejestry układu Si4703)
sygnalizowany jest poprzez sciągnięcie wyprowadzenia GPIO2 układu Si4703 (wejscie INT0 mikrokontrolera) do „0” przez
czas 5ms. Poszczególne rejestry RDS zawierają następujące dane:
RDSA - pomijamy, gdyż zawiera nieinteresujące nas dane (Program Inentyfication Code tzw.PI)
RDSB - bity 15...11 okreslają typ wiadomosci (Group Type +Version) zas bity 1...0 dla typu 0A/0B lub 3...0 dla typu
2A/2B określają indeks przesłanego segmentu danych (segment danych to 2 kolejne znaki wiadomosci dla typu 0A/0B czy 2B
lub 4 znaki dla typu 2A). W przypadku wiadomości typu 4A (CT) najmłodsze 2 bity tego rejestru stanowią najstarsze bity
daty zapisanej w konwencji MJD.
RDSC - dla typu 0A/0B i 2B pomijamy, dla typu 2A zawiera kody ASCII 2 znaków bieżącego segmentu danych (wiadomosci typu
PS lub RT), zaś dla typu 4A zawiera datę w zapisie MJD oraz najstarszy bit godziny (bit0 tegoż rejestru).
RDSD - dla typu 0A/0B i 2A/2B zawiera kody ASCII 2 znaków bieżącego segmentu danych (wiadomosci typu PS lub RT) zas dla
typu 4A zawiera godzinę (bity 15...12), minuty (bity 11...6) oraz offset lokalnego czasu (bity 5...0). */

ISR(INT0_vect)
{
  static uint8_t PScharIndex;  //Indeks odebranego znaku wiadomosci PS (Program Service)
  static uint8_t RTcharIndex;  //Indeks odebranego znaku wiadomosci RT (Radio Text)
  static char PStext[9];  //Deklaracja lokalnej tablicy przechowującej nazwę stacji (Program Service): max. 8 znaków +
terminator
  static char RTtext[65];  //Deklaracja lokalnej tablicy przechowującej tekst (Radio Text): max. 64 znaki + terminator
  register uint8_t groupType;  //Typ odebranej wiadomosci RDS umieszczony w starszym bajcie rej. RDSB
  register uint8_t receivedCharIndex;  //Index bieżącego znaku obliczany na podstawie przesłanego indeksu segmentu
danych PS/RT
  register uint8_t A_B;  //Wskaźnik zmiany wiadomosci typu RT (niezbędny, gdyż ta sama wiadomosc może byc powtarzana
wielokrotnie)
  static uint8_t RTchanged;  //Znacznik zmiany tresci wiadomosci typu RT
  register uint8_t idx, messageReady=0;  //Zmienne pomocnicze
  //Odczyt bieżących wartosci wyłącznie rejestrów 0x0A...0x0F układu Si4703 (rejestry o adresach 0x0C...0x0F to
rejestry RDSA...RDSD)
  TWI_Start();
  TWI_WriteByte(SI4703RD_ADDR);
  for(idx = 0x0A ; idx<0x10; idx++) Registers[idx] = TWI_ReadInteger( (idx !=0x0F) ? ACK: NACK);  //Nie potwierdzamy
ostatniego bajta
  TWI_Stop();
  groupType = (Registers[RDSB_REG]>>11);  //Typ przechwyconej wiadomosci RDS
  switch(groupType)
  {
    case RDS_GROUP_TYPE_0A:  //Basic Tuning and Switching Information only - maks. 8 znaków
    case RDS_GROUP_TYPE_0B:  //Basic Tuning and Switching Information only - maks. 8 znaków
//Na podstawie przesłanego indeksu segmentu danych obliczamy bieżący indeks znaku (segment*2, gdyż odbieramy po 2
znaki)
      receivedCharIndex = ((Registers[RDSB_REG] & 0x03)<<1);
//Sprawdzamy czy odebrany segment znaków ma oczekiwaną wartość indeksu co oznacza, że nie pominięto żadnej ramki
      if(PScharIndex == receivedCharIndex)
      {
        idx = (Registers[RDSD_REG]>>8);  //Pierwszy, odebrany znak
        if( idx<’ , || idx>’}’ ) idx = , ,;  //Poprawienie ewentualnych nieodpowiednich kodów znaków
        PStext[PScharIndex] = idx;
        idx = Registers[RDSD_REG];  //Drugi, odebrany znak
        if( idx<’ , || idx>’}’ ) idx = , ,;  //Poprawienie ewentualnych nieodpowiednich kodów znaków
        PStext[PScharIndex+1] = idx;
        PScharIndex +=2;  //Zwiększamy bieżący index o 2 znaki
        if(PScharIndex == 8)  //Odebrano kompletną wiadomosc typu Program Service (nazwa stacji)
        {
          PStext[8] = ,\0’;  //Terminator na końcu stringa
          strcpy(RDS_PS, PStext);  //Kopiujemy otrzymaną wiadomosc do tablicy dostępnej dla innych modułów
          RDS_PSready = 1;  //Ustawiamy flagę otrzymania nowej wiadomosci RDS typu PS
          PScharIndex = 0;  //Zerujemy index znaku
        }
      } else PScharIndex = 0;  //Zerujemy index znaku, gdyż zgubiono częsc wiadomosci typu PS
      break;
    case RDS_GROUP_TYPE_2A:  //Radio Text only - maks. 64 znaki
//Na podstawie przesłanego indeksu segmentu danych obliczamy bieżący index znaku (segment*4, gdyż odbieramy po 4 znaki)
      receivedCharIndex = ((Registers[RDSB_REG] & 0x0F)<<2);
//Sprawdzamy czy odebrany segment znaków posiada oczekiwaną wartosc indeksu co oznacza, że nie pominięto żadnej ramki
      if(RTcharIndex == receivedCharIndex)
      {
        RTtext[RTcharIndex] = (Registers[RDSC_REG]>>8);  //Pierwszy, odebrany znak
        RTtext[RTcharIndex+1] = Registers[RDSC_REG];  //Drugi, odebrany znak
        RTtext[RTcharIndex+2] = (Registers[RDSD_REG]>>8);  //Trzeci, odebrany znak
        RTtext[RTcharIndex+3] = Registers[RDSD_REG];  //Czwarty, odebrany znak
//Korekta odebranych znaków: LF=’ ,, CR=’\0’ (znacznik końca stringa)
        for(idx=0; idx<4; idx++)
        {
          if(RTtext[RTcharIndex+idx] == 0x0A) RTtext[RTcharIndex+idx] = ‘ ‘;  //LF
          if(RTtext[RTcharIndex+idx] == 0x0D) { messageReady =1; break; }
        }
        if(!messageReady) { RTcharIndex +=4; idx=0; } //Zwiększamy bieżący index o 4 znaki
//Jesli nawet w przesłanej wiadomosci nie wystąpił znacznik końca stringa (CR) ale odebrano 64 znaki to kończymy
//odbieranie bieżącej wiadomosci tekstowej ustawiając odpowiednią flagę i kopiując wiadomosc do zmiennej globalnej
//pod warunkiem, iż tresc odebranej wiadomosci jest inna niż ostatnio przesłana - o tym „mówi” wskaźnik A-B
        if(messageReady || RTcharIndex == 64)  //Odebrano kompletną wiadomosc typu Radio Text (informacje tekstowe)
        {
//Sprawdzamy czy odebrana własnie wiadomosc typu RT nie jest kopią poprzednio przesłanej - bit ten zmienia się
//na wartosc przeciwną za każdym razem, gdy przesyłana jest nowa tresc. Dla kopii wiadomosci pozostaje bez zmian
          A_B = (Registers[RDSB_REG]>>4) & 0x01;
          if(RTchanged != A_B)
          {
            RTtext[RTcharIndex+idx] = ‘\0’;  //Terminator na końcu stringa
            strcpy(RDS_RT, RTtext);  //Kopiujemy otrzymany tekst do tablicy dostępnej dla innych modułów
            RDS_RTready = 1;  //Ustawiamy flagę otrzymania nowej wiadomosci RDS typu PS
            RTchanged = A_B;  //Aktualizujemy
          }
          RTcharIndex = 0;  //Zerujemy index znaku
        }
      } else RTcharIndex = 0;  //Zerujemy index znaku, gdyż zgubiono częsc wiadomosci typu RT
      break;
    case RDS_GROUP_TYPE_2B:  //Radio Text only - maks. 32 znaki
//Na podstawie przesłanego indeksu segmentu danych obliczamy bieżący index znaku (segment*2, gdyż odbieramy po 2 znaki)
      receivedCharIndex = ((Registers[RDSB_REG] & 0x0F)<<1);
//Sprawdzamy czy odebrany segment znaków posiada oczekiwaną wartosc indeksu co oznacza, że nie pominięto żadnej ramki
      if(RTcharIndex == receivedCharIndex)
```

Listing 6. c.d.

```
      {
        RTtext[RTcharIndex] = (Registers[RDSD_REG]>>8);  //Pierwszy, odebrany znak
        RTtext[RTcharIndex+1] = Registers[RDSD_REG];  //Drugi, odebrany znak
//Korekta odebranych znaków: LF=' ,, CR='\0' (znacznik końca stringa)
        for(idx=0; idx<2; idx++)
        {
          if(RTtext[RTcharIndex+idx] == 0x0A) RTtext[RTcharIndex+idx] = ' ';  //LF
          if(RTtext[RTcharIndex+idx] == 0x0D) { messageReady =1; break; }
        }
        if(!messageReady) { RTcharIndex +=2; idx=0; } //Zwiększamy bieżący index o 2 znaki
//Jesli nawet w przesłanej wiadomosci nie wystąpił znacznik końca stringa (CR) ale odebrano 32 znaki to kończymy
//odbieranie bieżącej wiadomosci tekstowej ustawiając odpowiednią flagę i kopiując wiadomosc do zmiennej globalnej
//pod warunkiem, iż tresc odebranej wiadomosci jest inna niż ostatnio przesłana - o tym „mówi" wskaźnik A-B
        if(messageReady || RTcharIndex == 32)  //Odebrano kompletną wiadomosc typu Radio Text (informacje tekstowe)
        {
//Sprawdzamy czy odebrana własnie wiadomosc typu RT nie jest kopią poprzednio przesłanej - bit ten zmienia się
//na wartosc przeciwną za każdym razem, gdy przesyłana jest nowa tresc. Dla kopii wiadomosci pozostaje bez zmian
          A_B = (Registers[RDSB_REG]>>4) & 0x01;
         if(RTchanged != A_B)
         {
           RTtext[RTcharIndex+idx] = '\0';  //Terminator na końcu stringa
           strcpy(RDS_RT, RTtext);  //Kopiujemy otrzymany tekst do tablicy dostępnej dla innych modułów
           RDS_RTready = 1;  //Ustawiamy flagę otrzymania nowej wiadomosci RDS typu PS
           RTchanged = A_B;  //Aktualizujemy
         }
         RTcharIndex = 0;  //Zerujemy index znaku
        }
     } else RTcharIndex = 0;  //Zerujemy index znaku, gdyż zgubiono częsc wiadomosci typu RT
     break;
     case RDS_GROUP_TYPE_4A:  //Clock Time and Date only
     Minuts = (Registers[RDSD_REG]>>6) & 0b111111;
     Hours = (Registers[RDSD_REG]>>12);
     Hours |= ((Registers[RDSC_REG] & 0x01) <<4);
//Korygujemy obliczoną wartosc godzin o offset czasu lokalnego ( RDSD.5=0 -> offset+,RDSD.5=1 -> offset- )
//Offset podany jest jako wielokrotnosc wartosci pół godziny
     if(Registers[RDSD_REG] & 0b100000) Hours -= ((Registers[RDSD_REG] & 0b11111)>>1);
       else Hours += ((Registers[RDSD_REG] & 0b11111)>>1);
//Ustawiamy flagę otrzymania nowej wiadomosci RDS typu CT tylko w przypadku poprawnych danych
     if(Hours<24 && Minuts<60) RDS_CTready = 1;
     break;
  }
}
```

ków w przesyłanym tekście określają bity C3…C0. Możliwe jest wysłanie maksymalnie 64 znaków dla typu 2A lub 32 znaków dla typu 2B. W przypadku, gdy przesyłany tekst jest krótszy, niż dostępny dla danego typu grupy danych, na końcu takiej wiadomości musi wystąpić znacznik końca tekstu, czyli bajt o wartości 0x0D (CR). Dla urządzeń o rozbudowanym interfejsie użytkownika, które są w stanie wyświetlać kilka linii znaków przewidziano dodatkowy znak sterujący o kodzie 0x0A (LF), który powinien być zidentyfikowany jako znacznik przejścia do nowej linii.

Ostatnim typem grupy danych, które obsługiwane są przez nasze urządzenie jest typ 4A, którego strukturę pokazano na **rysunku 8**. Przeznaczeniem tego typu grupy danych jest przesyłanie informacji o czasie, która to może być wykorzystana przez radioodbiornik do synchronizacji wbudowanego zegara czasu rzeczywistego. Struktura tego typu danych zawiera następujące elementy:

MJD jest 17-bitowym słowem reprezentującym datę zapisaną w konwencji MJD (Modified Julian Date).

H jest wartością godzin czasu UTC w zapisie 24-godzinnym.

M jest wartością minut czasu UTC.

OFF jest offsetem godzin czasu lokalnego w stosunku do wysłanego znacznika czasu podanym jako wielokrotność 30 minut. Znacznik ± określa znak offsetu (0 znacza offset dodatni a 1 oznacza offset ujemny).

Na koniec tego opisu przedstawię procedurę obsługi przerwania zewnętrznego INT0, która to odpowiedzialna jest za odbiór transmisji RDS oraz jej poprawne zdekodowanie tym bardziej, że przeglądając dziesiątki stron poświęconych programowaniu układów z rodziny Si47xx nie spotkałem się z gotowym jak i kompleksowym rozwiązaniem tego rodzaju. Kod wspomnianej procedury przedstawiono na **listingu 6**.

Robert Wołgajew, EP