WOJSKOWA AKADEMIA TECHNICZNA

im. Jarosława Dąbrowskiego

WYDZIAŁ ELEKTRONIKI

PRACA DYPLOMOWA

PROJEKT KONCEPCYJNY MODUŁU KODERA SYGNAŁÓW RADARU WTÓRNEGO

(temat pracy dyplomowej)

sierż. pchor. Bartłomiej Jarosław ZĄBEK s. Jarosława

(stopień wojskowy, tytuł zawodowy, imiona i nazwisko, imię ojca dyplomanta)

ELEKTRONIKA I TELEKOMUNIKACJA

(kierunek studiów)

RADIONAWIGACJA

(specjalność)

STACJONARNE, STUDIA PIERWSZEGO STOPNIA INŻYNIERSKIE

(forma i rodzaj studiów))*

ppłk dr inż. Grzegorz CZOPIK

(stopień wojskowy, tytuł i stopień naukowy, imię i nazwisko promotora pracy dyplomowej)

WARSZAWA 2020

W O J S K O W A A K A D E M I A T E C H N I C Z N A

im. Jarosława Dąbrowskiego

WYDZIAŁ ELEKTRONIKI

INSTYTUT RADIOELEKTRONIKI

"AKCEPTUJĘ"
DZIEKAN
WYDZIAŁU ELEKTRONIKI

.....
Prof. dr hab. inż. Andrzej P. DOBROWOLSKI
Dnia ..25 WRZ. 2019..2019 r.

Z A D A N I E

do pracy dyplomowej

Wydane studentowi **plut. pchor. Bartłomiej Jarosław ZĄBEK**
(*stopień, imiona i nazwisko*)

ELEKTRONIKA I TELEKOMUNIKACJA
(*kierunek studiów*)

STACJONARNE STUDIA PIERWSZEGO STOPNIA - INŻYNIERSKIE
(*forma i rodzaj studiów*)

I. Temat pracy: **PROJEKT KONCEPCYJNY MODUŁU KODERA SYGNAŁÓW RADARU WTÓRNEGO**

II. Treść zadania:

1. Sformułowanie problemu pracy
2. Opracowanie założeń do realizacji projektu
3. Przegląd dostępnych rozwiązań technicznych
4. Wykonanie i przebadanie modelu modułu kodera
5. Wnioski z realizacji projektu

III. W rezultacie wykonania pracy należy przedstawić:

a/ notatkę objaśniającą z obliczeniami

b/ wykresy: niezbędne do realizacji pracy

c/ rysunki: niezbędne do realizacji pracy

IV. Opiekun pracy dyplomowej (§ 42, ust.4 RSW): .....

V. Konsultant: .....

VI. Opiekun merytoryczny: dr hab. inż. Jerzy PIETRASIŃSKI, prof. WAT

VII. Termin zdania przez studenta ukończonej pracy: 21.01.2020 r.

VIII. Data wydania zadania: 07.10.2019 r.

PROMOTOR PRACY DYPLOMOWEJ	DYREKTOR INSTYTUTU
.....	.....
ppłk dr inż. Grzegorz CZOPIK	dr hab. inż. Piotr KANIEWSKI, prof. WAT

Zadanie otrzymałem dnia 07.10 ..... 2019 r.

.....
(podpis studenta)

Spis treści

Wykaz akronimów:

SSR – (ang. *Secondary Surveillance Radar)* - radar wtórny

SPI – (*ang. Special Position Indication)* - impuls służący do szybkiej identyfikacji statku powietrznego

PCB – (ang. *Printed Circuit Board)* – płyta z połączeniami do montażu podzespołów elektronicznych

IFF – (ang. Identification Friend or Foe) - identyfikacja "swój–obcy"

1. Wstęp

Współcześnie wykorzystywanych jest wiele różnorodnych urządzeń radionawigacyjnych. Fakt ten pokazuje jak bardzo potrzebne stały się w lotnictwie systemy radioelektroniczne pomagające statkom powietrznym określić swoje położenie. Pomoce radionawigacyjne umożliwiają nie tylko określanie położenia obiektu w przestrzeni powietrznej, lecz także są w stanie precyzyjnie naprowadzić statek powietrzny na pas startowy podczas lądowania, nawet w warunkach braku widoczności (mgła, noc). Dodatkowo na bazie transpondera zainstalowanego na każdym samolocie powstał system antykolizyjny TCAS (ang. Traffic Alert and Collision Avoidance System). System TCAS ostrzega pilotów o niebezpieczeństwie zderzenia z innym statkiem powietrznym, co zwiększa bezpieczeństwo szczególnie w zatłoczonej przestrzeni powietrznej.

Systemów radionawigacyjnych oraz powstających systemów wykorzystujących nawigację satelitarną jest wiele. Temat pracy został zainspirowany działaniem radaru wtórnego. Działanie systemu opartego o radar SSR nie byłoby możliwe bez transpondera zainstalowanego na pokładzie statku powietrznego. Transponder odbiera sygnał zapytania i nadaje kodowany sygnał odpowiedzi. Urządzenie kodujące jeden z możliwych sygnałów odpowiedzi zostało skonstruowane w ramach tej pracy.

Rysunek 1 Widok na system antenowy. Górna prostokątna antena jest anteną radaru wtórnego [1]

Współcześnie produkowane są coraz doskonalsze i tańsze układy mikroprocesorowe. Mikrokontrolery dają bardzo szerokie możliwości budowania wszelkiego rodzaju urządzeń i układów o różnorodnych funkcjonalnościach. Projekt opisany w pracy oparty jest o ośmiobitowy mikrokontroler ATMEGA32A należący do rodziny AVR, produkowany przez firmę Atmel. Praca ta łączy wiedzę z zakresu specjalności radionawigacja związanej z tokiem studiów oraz z obszaru programowania wspomnianych mikrokontrolerów. Motywem przewodnim do zaprojektowania własnego układu jest pokazanie możliwości techniki wykorzystującej niedrogie mikroprocesory.

W następnym rozdziale przedstawiono założenia do realizacji projektu związanego z zadaniem dyplomowym. Trzeci rozdział stanowi wprowadzenie teoretyczne związane z urządzeniem. W następnych rozdziałach omówiono z kolei elementy jakie zostały wykorzystane do zbudowania układu, a później algorytm działania programu wczytanego do pamięci mikrokontrolera. W szóstym rozdziale przedstawione zostaną etapy praktycznej realizacji układu kodera sygnału Squawk. Ostatni rozdział poświęcony jest wnioskom i możliwościom ewentualnego rozwoju projektu.

2. Założenia do realizacji projektu

Celem pracy jest zbudowanie kodera sygnału kodującego i generującego sekwencje kodu Squawk, który nadawany jest przez transponder pracujący w modzie 3/A. Projekt zakłada zbudowanie urządzenia generującego kod Squawk o dowolnej wartości.

Poza obudowę układu wyprowadzono złącze BNC oraz dwa piny - jeden podłączony do masy, drugi do wyjścia sygnałowego. Złącze BNC pozwala połączyć układ z generatorem wielkiej częstotliwości i z oscyloskopem, natomiast złącza w postaci pinów umożliwiają zbadanie sygnału z wykorzystaniem innych przyłączy. Jako peryferia wejściowe zastosowano pięć przełączników astabilnych oraz przełącznik suwakowy, natomiast przy wykorzystaniu wyświetlacza LCD wyświetlana jest aktualnie generowana wartość kodu. Za pomocą czterech przycisków astabilnych, wprowadzane są zmiany kodu, impuls SPI generowany jest lub nie, w zależności od położenia przełącznika suwakowego. Po wciśnięciu piątego przycisku astabilnego generowana jest jedna sekwencja kodu Squawk. Układ zasilany jest przez złącze USB napięciem 5 V, co pozwala na wykorzystanie do tego celu powszechnie stosowanych ładowarek do telefonów, tak zwanych „powerbank'ów" czy też złącza USB występującego przy każdym komputerze. Ze względu na ograniczenia sprzętowe wynikające z niskiej częstotliwości pracy procesora, czas generacji kodu Squawk jest dłuższy o trzy rzędy wielkości.

W projekcie do napisania oprogramowania sterującego pracą mikrokontrolera wykorzystano zintegrowane środowisko programistyczne AtmelStudio 7. Środowisko to służy do pisania i testowania programów dla mikrokontrolerów rodziny AVR i SAM. Program napisany jest w języku C. Do zaprojektowania układu posłużył program Eagle firmy Autodesk. Program ten pozwala narysować schemat połączeń elektrycznych oraz na podstawie tego schematu zaprojektować układ połączeń elektrycznych, który może zostać wytrawiony na płytce PCB. Schematy blokowe przedstawiające algorytm działania programu powstały w programie DiagramDesigner.

3. Podstawy teoretyczne

1.2. Radar wtórny

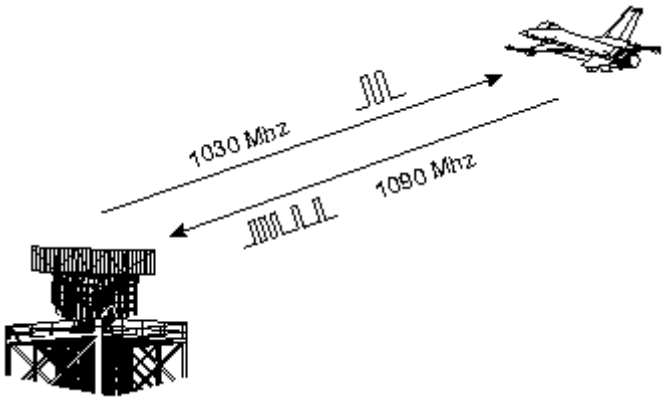
Radar wtórny powstał w wyniku potrzeby rozróżniania własnych statków powietrznych od wrogich, jaka pojawiła się w trakcie II wojny światowej. Koncepcja używanego współcześnie systemu IFF pochodzi ze wspomnianego konfliktu zbrojnego. Z czasem radary wtórne zyskiwały na funkcjonalności, powstało kilka modów pracy w jakich operują oraz zostały wykorzystane przez lotnictwo cywilne.

Radar wtórny (SSR) jest radarem z aktywnym zapytaniem i z aktywną odpowiedzią. Określenie aktywna odpowiedź oznacza, iż statek powietrzny po odebraniu przez antenę transpondera sygnału zapytania pochodzącego od radaru wtórnego, emituje własny sygnał odpowiedzi. Pasywną odpowiedzią jest echo sygnału wyemitowane w wyniku odbicia sygnału radarowego od powierzchni obiektu. Emitowana przez transponder odpowiedź ma większą moc niż echo sygnału w przypadku pasywnej odpowiedzi. Zwiększa to zasięg działania systemu oraz dzięki specjalnemu kodowaniu sygnału odpowiedzi umożliwia przesyłanie dodatkowych informacji przez statek powietrzny. Sygnał zapytania nanoszony jest na falę nośną o częstotliwości 1030 MHz, natomiast odpowiedzi nadanej przez transponder statku powietrznego na 1090 MHz.

1.3. Koder sygnału radaru wtórnego

Kodery sygnału radaru wtórnego są zintegrowane z transponderem. Mogą pracować w różnych modach pracy w każdym z nich przekazując informacje, o czym zostanie napisane szczegółowo w dalszym podrozdziale. Z perspektywy niniejszej pracy najistotniejszy jest mod 3/A, który stosowany jest do nadawania kodu Squawk.

.



Rysunek 2 Poglądowa zasada komunikacji transpondera z radarem SSR [2]

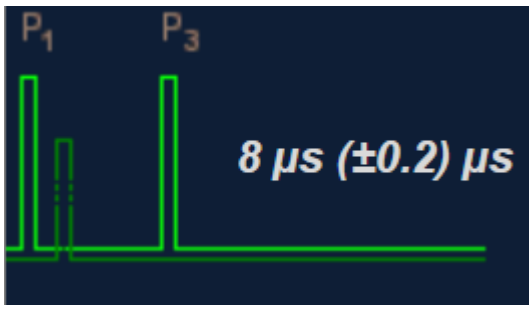
1.4. Mody pracy transpondera

W niniejszym podrozdziale opisany zostanie ściśle związany z realizacją projektu mod 3/A oraz dodatkowo mod C o identycznej strukturze sygnału odpowiedzi, jednak informacja jest kodowana w inny sposób. Istnieją także inne mody pracy stosowane przez wojsko i przez lotnictwo cywilne. Mod 3/A jest wykorzystywany zarówno przez siły powietrzne jak i lotnictwo cywilne. Mody 1 oraz 2 stosowane są wyłącznie przez wojsko. Do cywilnych modów pracy poza modem 3/A należą mody B, C ,D oraz S. Mody B oraz D nie są obecnie stosowane.

1.4.1. Mod 3/A

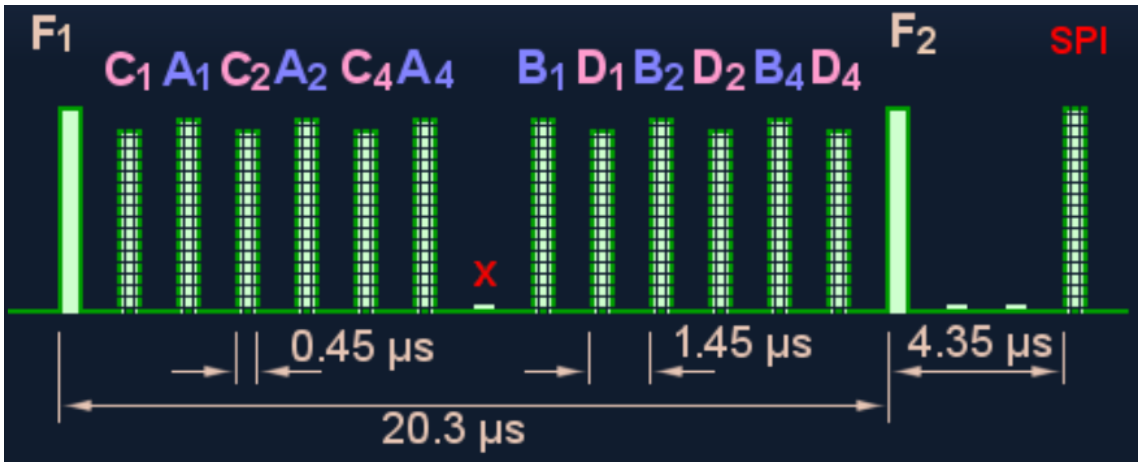
Sygnałem odpowiedzi w modzie 3/A jest kod Squawk. Wspomniany kod jest wykorzystywany do identyfikacji statku powietrznego przez kontrolę ruchu lotniczego, powiązany jest ze znakiem wywoławczym statku powietrznego oraz planem lotu. Kod Squawk przydzielany jest poszczególnym statkom powietrznym przez kontrolę ruchu lotniczego i zostaje zapisany w bazie danych. Dzięki temu rozwiązaniu kontrola ruchu lotniczego jest dokładnie zorientowana z jakim samolotem ma do czynienia (np. LOT4170, WAE321).

Kod Squawk składa się z czterech cyfr. Zapewnia 4096 możliwych kombinacji kodów. Kodowany jest przez 12 impulsów – 3 impulsy na każdą z czterech cyfr. Dodatkowo występują dwa impulsy: F1 - rozpoczynający ramkę oraz F2 - zamykający ramkę. W samym środku ramki pozostawione jest miejsce na impuls X, który nie jest emitowany, jeżeli dekoder współpracujący z radarem odbierze w tym miejscu impuls, uzna odpowiedź za zakłóconą przez odpowiedź z transpondera innego samolotu i automatycznie ją odrzuci. Każda z liczb kodu Squawk jest kodowana osobno, binarnie z wykorzystaniem 3 bitów reprezentowanych przez poszczególne impulsy. Impulsy A1, A2 i A4 odpowiedzialne są za kodowanie pierwszej cyfry kodu Squawk, impulsy B drugiej, a C oraz D trzeciej i czwartej.



Rysunek 3 Sygnał zapytania w modzie 3/A [3]

Przerwa czasowa pomiędzy impulsami P1 i P3 wynosi 8 µs. W sygnale zapytania środkowy impuls P2 (na zamieszczonej grafice oznaczony przerywaną linią w połowie swojej wysokości), umożliwia odróżnienie zapytań wyemitowanych w listkach bocznych charakterystyki anteny radaru wtórnego, od zapytań wyemitowanych w listku głównym. Poza głównym kierunkiem charakterystyki antenowej, impuls ten emitowany jest z większą mocą niż impulsy P1 i P3, co pozwala na rozróżnienie czy sygnał został wyemitowany przez listki boczne.



Rysunek 4 Ramka odpowiedzi transpondera [4]

Impuls SPI jest wykorzystywany przez kontrolerów ruchu lotniczego w celu potwierdzenia identyfikacji poszczególnych samolotów. Pilot może włączyć generację tego impulsu na polecenie kontrolera ruchu lotniczego. Wskaźnik radaru wyróżni wtedy samolot który nadał impuls SPI.

1.4.4 Mod C

Mod C służy do przekazania kontroli naziemnej informacji o wysokości barometrycznej na jakiej znajduje się samolot, pozwalając w ten sposób na określenie poziomu lotu (FL - Flight Level). Do kodowania sygnału odpowiedzi wykorzystywana jest identyczna ramka jak w przypadku modu 3/A jednak–sama informacja kodowana jest w oparciu o kod Grey'a. Poniżej przedstawiono przykładowy sposób kodowania dla wybranych wysokości lotu. Sygnał zapytania w modzie C różni od sygnału zapytania w modzie 3/A jedynie przerwą czasową pomiędzy impulsem P1 i P3, która wynosi 21 µs.

ALTITUDE	A1	A2	A4	B1	B2	B4	C1	C2	C4	D1	D2	D4	SQUAWK
0	0	0	0	0	1	1	0	1	0	0	0	0	0620
100	0	0	0	0	1	1	1	1	0	0	0	0	0630
200	0	0	0	0	1	1	1	0	0	0	0	0	0610
300	0	0	0	0	1	0	1	0	0	0	0	0	0210
400	0	0	0	0	1	0	1	1	0	0	0	0	0230
500	0	0	0	0	1	0	0	1	0	0	0	0	0220
600	0	0	0	0	1	0	0	1	1	0	0	0	0260
700	0	0	0	0	1	0	0	0	1	0	0	0	0240
800	0	0	0	1	1	0	0	0	1	0	0	0	0340

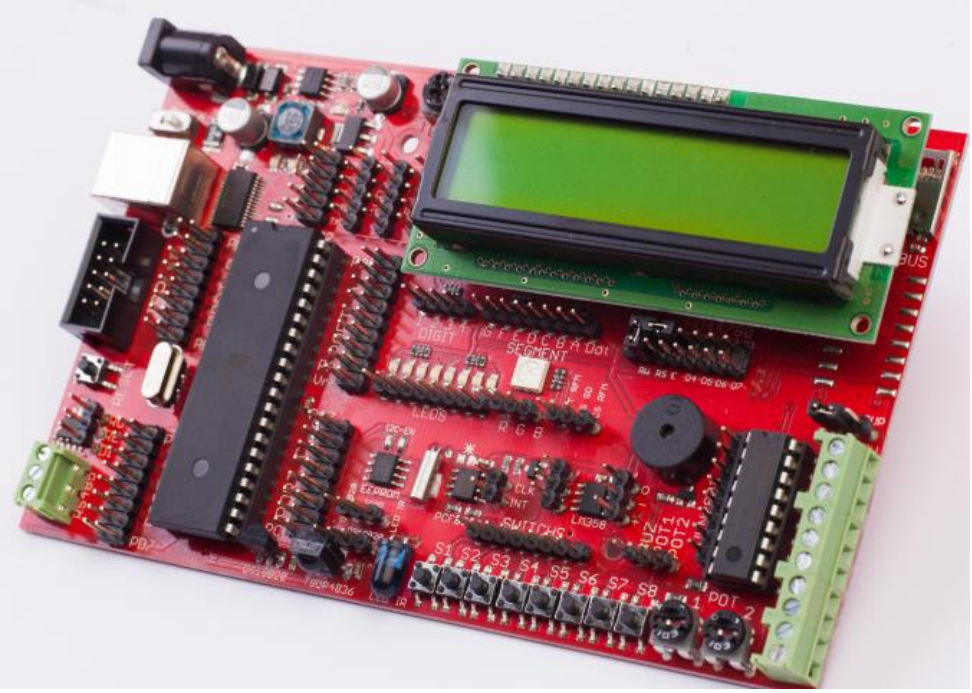
Tabela 1 Tabela przedstawiające sposób kodowania niektórych wysokości lotu w odpowiedzi modu C [5]

3. Dobór elementów do budowy układu

W niniejszym rozdziale zostaną przedstawione najważniejsze parametry wykorzystanego w projekcie mikrokontrolera ATMEGA32A, zestaw uruchomieniowy EvB 5.1, oraz inne elementy.

3.1. Zestaw uruchomieniowy EvB 5.1.

Do opracowania prototypu urządzenia oraz do przeprowadzania testów oprogramowania wykorzystany został zestaw uruchomieniowy EvB 5.1. Zestaw umożliwia sprawne testowanie własnych koncepcji i projektów, zdecydowanie szybciej niż z wykorzystaniem płytki stykowej.



Rysunek 5. Zestaw uruchomieniowy EvB 5.1 [6]

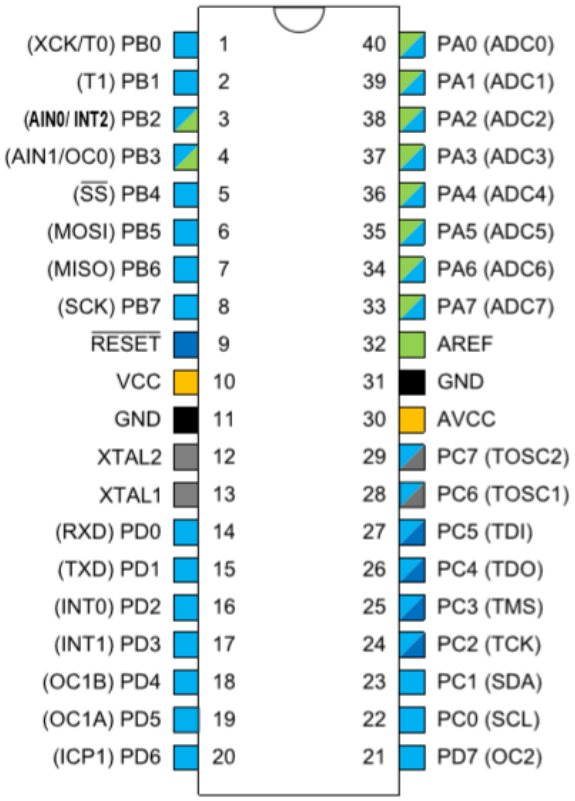
3.2. Mikrokontroler ATmega32A

Do realizacji zadania dyplomowego wykorzystano mikrokontroler ATMEGA32A. Jest to 8-bitowy mikroprocesor, co oznacza że jednostka arytmetyczno – logiczna ALU (ang. *Arithmetic Logic Unit)* stanowiąca część procesora, wykonuje operacje na ośmiobitowych liczbach, a wszystkie szyny danych są ośmiobitowe. Omawiany mikrokontroler jest urządzeniem o niskim poborze mocy, zbudowanym na bazie architektury AVR RISC. Posiada 32 rejestry ogólnego przeznaczenia.

ATmega32A posiada 32 linie I/O (input/output) ogólnego przeznaczenia, możliwość wykorzystania wewnętrznych i zewnętrznych przerwań, interfejs JTAG, USART, TWI, SPI, przetwornik analogowo-cyfrowy, programowalny „watchdog”. Główną rolę w napisanym programie pełnią liczniki. ATmega32A ma wbudowane dwa liczniki 8-bitowe oraz jeden 16-bitowy.

Parametr	Wartość
Ilość pinów	40
Pamięć flash (KB)	32
Pamięć SRAM (KB)	2
Pamięć EEPROM (KB)	1
Ilość pinów wejścia/wyjścia do ogólnego zastosowania	32
Napięcie zasilania	2.7 – 5.5 V
Maksymalna częstotliwość pracy	16 MHz

Tabela 3. Podstawowe parametry ATMEGA32A [7]



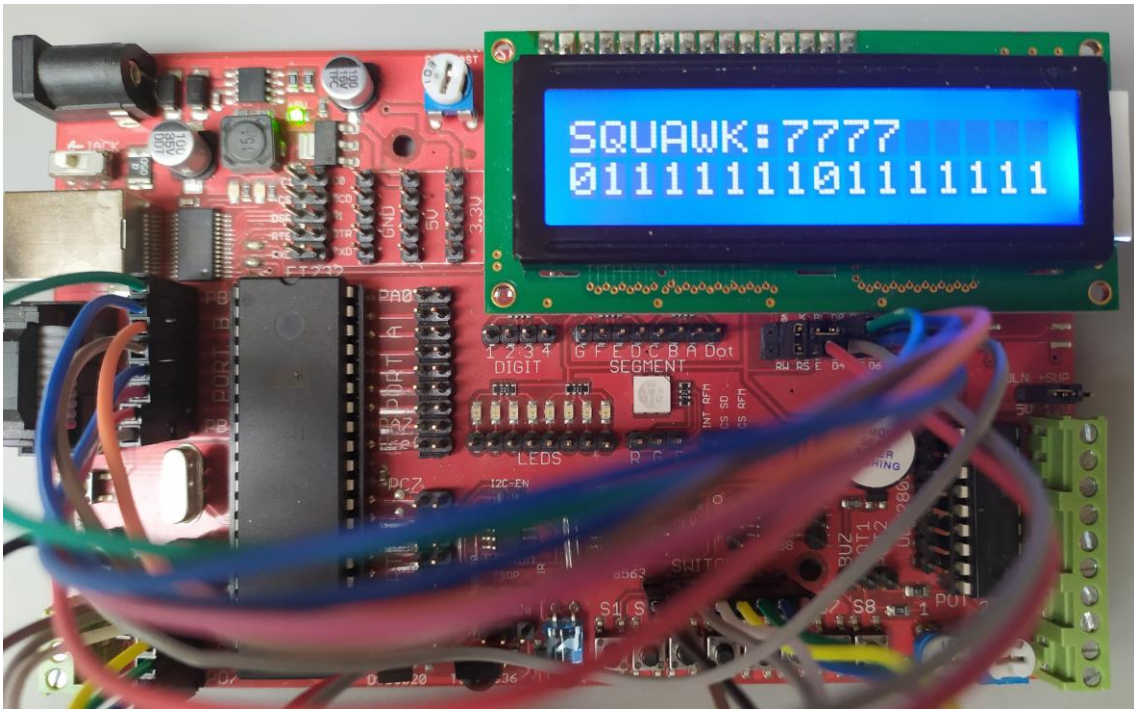
Rysunek 6. Mikrokontroler ATMEGA32A [8]

3.3 Inne elementy

W projekcie zastosowano także wyświetlacz LCD ze sterownikiem w standardzie HDD44780, przyciski astabilne, przełącznik suwakowy oraz gniazdo BNC aby umożliwić podłączenie urządzenia z oscyloskopem i gniazdo USB w celu podłączenia zasilania. Połączenia elektryczne wytrawione zostały na laminacie światłoczułym firmy OEM.

4. Wykonanie prototypu.

Prototyp oraz testy prawidłowego działania oprogramowania zostały wykonane na przedstawionym we wcześniejszym rozdziale zestawie uruchomieniowym EvB 5.1. Widoczny na zamieszczonym niżej zdjęciu wyświetlacz LCD pokazuje aktualnie kodowany numer Squawk, natomiast wiersz poniżej potrzebny był do poglądowego sprawdzania prawidłowego funkcjonowania programu, który pożądany numer Squawk zamienia na tablicę zer oraz jedynek, na podstawie której inne funkcje programu generują pożądany sygnał. W ostatecznej wersji układu linia ta nie jest wyświetlana. Wstępne badanie układu oraz otrzymanie właściwych struktur sygnału były podstawą do montażu urządzenia w wersji docelowej.



Rysunek 7. Prototyp wykonany na zestawie EvB 5.1

5. Oprogramowanie mikrokontrolera

W niniejszym rozdziale opisany został sposób działania programu zaimplementowanego w mikrokontrolerze. Opisane są zasady działania liczników wbudowanych w mikrokontroler.

5.1 Przerwania od liczników

Przerwanie polega na zatrzymaniu przez procesor pracy programu głównego w wyniku pojawienia się sygnału zewnętrznego (przerwanie zewnętrzne) lub od wbudowanego elementu (przerwanie wewnętrzne) jak np. licznik, moduł przetwornika analogowo-cyfrowego, moduł do transmisji danych jak UART/USART, TWI, SPI. Procesor po otrzymaniu sygnału wymuszającego przerwanie, zatrzyma działanie programu głównego i wznowi jego działanie dopiero po wykonaniu procedury obsługi przerwania. Zostaną omówione przerwania których źródłem jest sygnał od licznika sprzętowego. Liczniki generują przerwania po upływie określonego czasu.

W celu ustawienia pożądanego czasu po upływie którego nastąpi przerwanie należy sprawdzić częstotliwość taktowania procesora i ustawić odpowiednią wartość preskalera. Czas po jakim nastąpi przerwanie można obliczyć za pomocą poniższego wzoru:

$$t_{przerwania} = \frac{1}{f_{procesora}} * ilość\ taktów\ procesora * preskaler$$

$t_{przerwania}$ – czas po jakim wystąpi przerwanie od licznika

$f_{procesora}$ – częstotliwość taktowania procesora

ilość taktów procesora – ilość cykli pracy jakie wykona procesor w trakcie zliczania przez licznik

preskaler – ustawiona w rejestrze licznika wartość preskalera

Należy obliczyć czas trwania pojedynczego taktu procesora, pomnożyć razy ilość ustawionych taktów, a następnie pomnożyć przez wartość preskalera. Preskaler powoduje, że nie wszystkie takty procesora zmieniają wartość rejestru licznika, tj. jeżeli ustawimy preskaler na osiem to będzie zliczany tylko co ósmy takt procesora, co opóźni czas po jakim zostanie wygenerowane przerwanie ośmiokrotnie. Ustawienie wartości 0 preskalera spowoduje zatrzymanie odliczania. Ilość taktów jakie zliczy licznik zanim wygeneruje sygnał powodujący przerwanie ustawia się wykorzystując przeznaczone tego rejestry sterujące. Liczniki sprzętowe mogą generować przerwania pracując w dwóch trybach: zwykłego licznika oraz CTC (Clear Timer on Compare).

W trybie zwykłego licznika przerwanie następuje po przepełnieniu licznika, czyli po odliczeniu od zera lub od innej zadanej wartości początkowej do maksymalnej liczby, w której zakresie pracuje licznik. Liczniki 8-bitowe zliczają do 255, natomiast 16-bitowe do 65625 po czym następuje przepełnienie. Czas po jakim nastąpi przerwanie ustawia się poprzez rejestr TCNTx (w miejscu „x” jest zawsze numer licznika). Jeżeli ustawimy wartość na tym rejestrze na 100, to w przypadku licznika 8-bitowego sygnał

przerwania dotrze do procesora po odliczeniu przez licznik od 100 do 255 i przerwanie wystąpi po zliczeniu 155 taktów (oczywiście uwzględniając preskaler).

W trybie CTC licznik zlicza wartość od 0 do wartości rejestru OCRx. Przerwanie nie następuje po przepełnieniu, lecz w momencie gdy rejestr TCNTx przechowujący aktualnie zliczoną wartość będzie miał taką samą wartość jak rejestr OCRx. Jeżeli rejestr OCRx zostanie ustawiony np. na wartość 100, to licznik zliczy od 0 do 100 i przerwanie nastąpi po zliczeniu 100 taktów procesora. Przy pisaniu programu wykorzystany został tryb pracy CTC.

5.2 Algorytm działania programu

Dwudziestoelementowa tablica *sygnal[20]* jest podstawą działania programu. W zależności od wartości danego elementu tablicy, generowany będzie jeden z impulsów (trwający 0,45ms) lub nie. Pomiędzy każdym impulsem występuje jednomilisekundowa przerwa, która nie ma swojego odzwierciedlenia w elementach tablicy. Pierwszy element tablicy jest potrzebny ze względu na sposób działania programu. Dzięki dwóm elementom o wartości 0, pomiędzy F2 i SPI możliwe jest wygenerowanie 4,35 ms przerwy między wspomnianymi impulsami. Cała tablica jest zbiorem zer oraz jedynek. Gdy dany element ma wartość 1 impuls zostanie wygenerowany.

```
                    //   F1 C1 A1 C2 A2 C4 A4 X  B1 D1 B2 D2 B4 D4 F2 PRZERWA  SPI
uint8_t sygnal[20] = {0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,    0, 0, 0};
```

Rysunek 8. Tablica sygnal[20] - zrzut ekranu z środowiska AtmelStudio

Funkcja *wprowadzanieKoduSQ* odpowiada za wprowadzanie pożądanych wartości do tablicy *sygnal[20]*. W bibliotece *koder.h* zostały napisane procedury przerwań, funkcje włączające lub wyłączające liczniki sprzętowe oraz ustawienia rejestrów określających tryb pracy, wartość preskalera i rejestru OCRx. Timer'y pracują w trybie CTC. Działanie programu przedstawione zostało za pomocą trzech schematów, jeden schemat pokazuje działanie pętli głównej programu, natomiast dwa pozostałe przedstawiają działanie funkcji *wprowadzanieKoduSQ* oraz sposobu w jaki generowany jest kod Squawk.

Aktualna wartość kodu jest na bieżąco wyświetlana przez wyświetlacz LCD. Osobny wiersz wyświetlacza informuje użytkownika czy impuls SPI zostanie wygenerowany.

5.2.1 Działanie pętli głównej programu

Najistotniejszymi zmiennymi są zmienne A,B,C,D reprezentujące poszczególne cyfry kodu Swuawk oraz tablica *sygnal[20]*. Za wartość każdej z cyfr odpowiada jeden przycisk astabilny, jego wciśnięcie powoduje zwiększenie wartości cyfry lub jej wyzerowanie, jeżeli po zwiększeniu miałaby osiągnąć wartość większą od 7. Osobny przycisk powoduje generacje kodu. Impuls SPI generowany jest w zależności od położenia przełącznika suwakowego.

5.2.2 Działanie funkcji zmianaKoduSQ

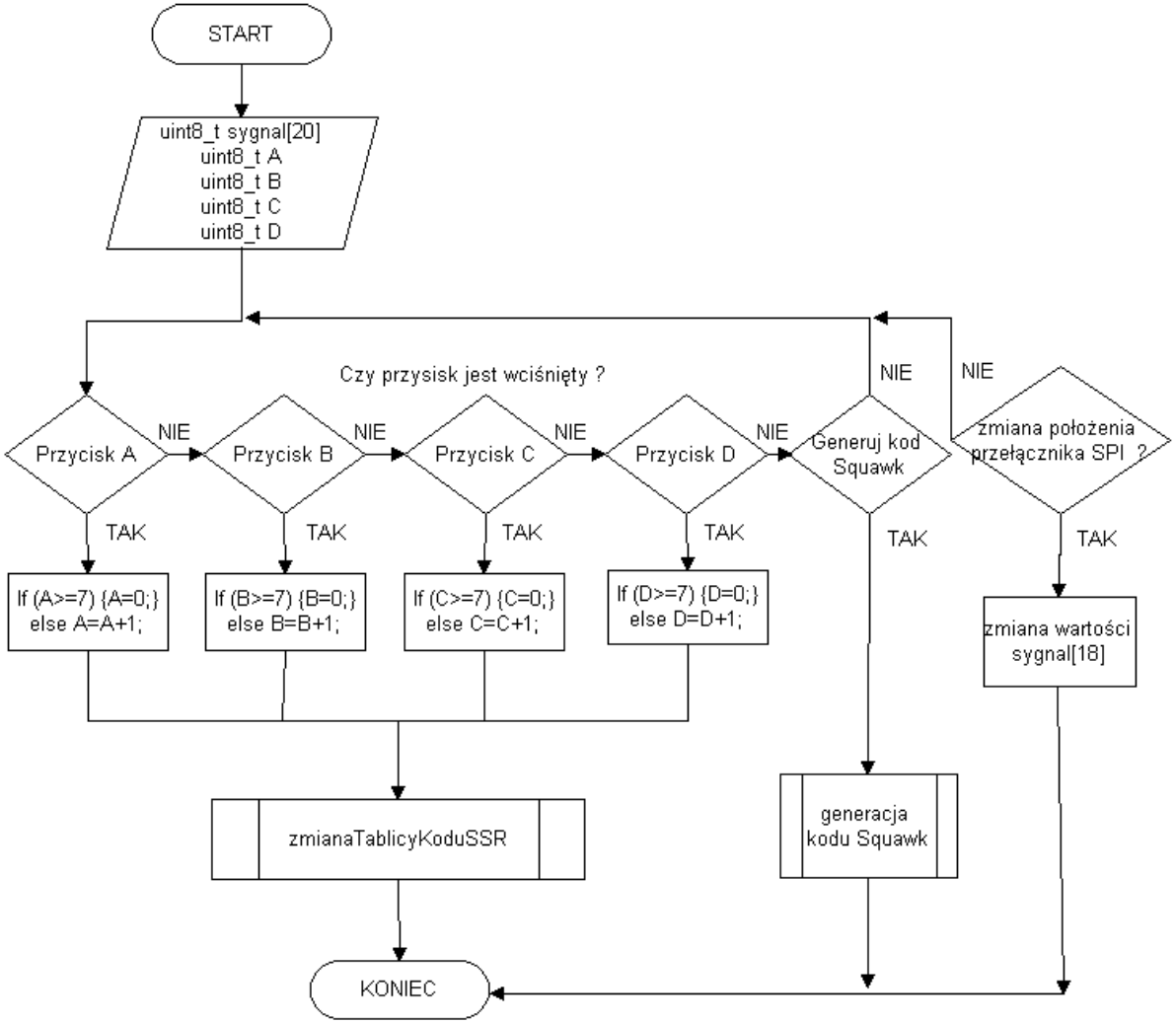
Funkcja ta odpowiada za możliwość zmiany kodu Squawk. Zmienia ona parametry tablicy *sygnal[20]* z programu głównego poprzez podstawienie pod odpowiednie adresy pamięci wartości z trzyelementowej tablicy *tabSQUAWK[3]*. Funkcja w pierwszym etapie zmienia wprowadzoną jako argument funkcji cyfrę, na wcześniej wspomnianą tablicę *tabSQUAWK[3]*. Następnie, w zależności czy dana cyfra odpowiada literze A,B,C lub D przypisuje zmiennej „*poczatek”* taką wartość, aby w dalszej części pracując na adresie tablicy *sygnal[20]* przypisać dane z trzyelementowej tablicy *tabSQUAWK[3]* do tablicy sygnal[20]. Wskaźnik *wskSygnal* wskazuje na adres tablicy sygnal[20] z pętli głównej. Adres wskaźnika jest trzykrotnie zwiększany o dwa, ponieważ impulsy kodujące tę samą literę kodu nie występują obok siebie.

5.3.3 Generacja kodu Squawk

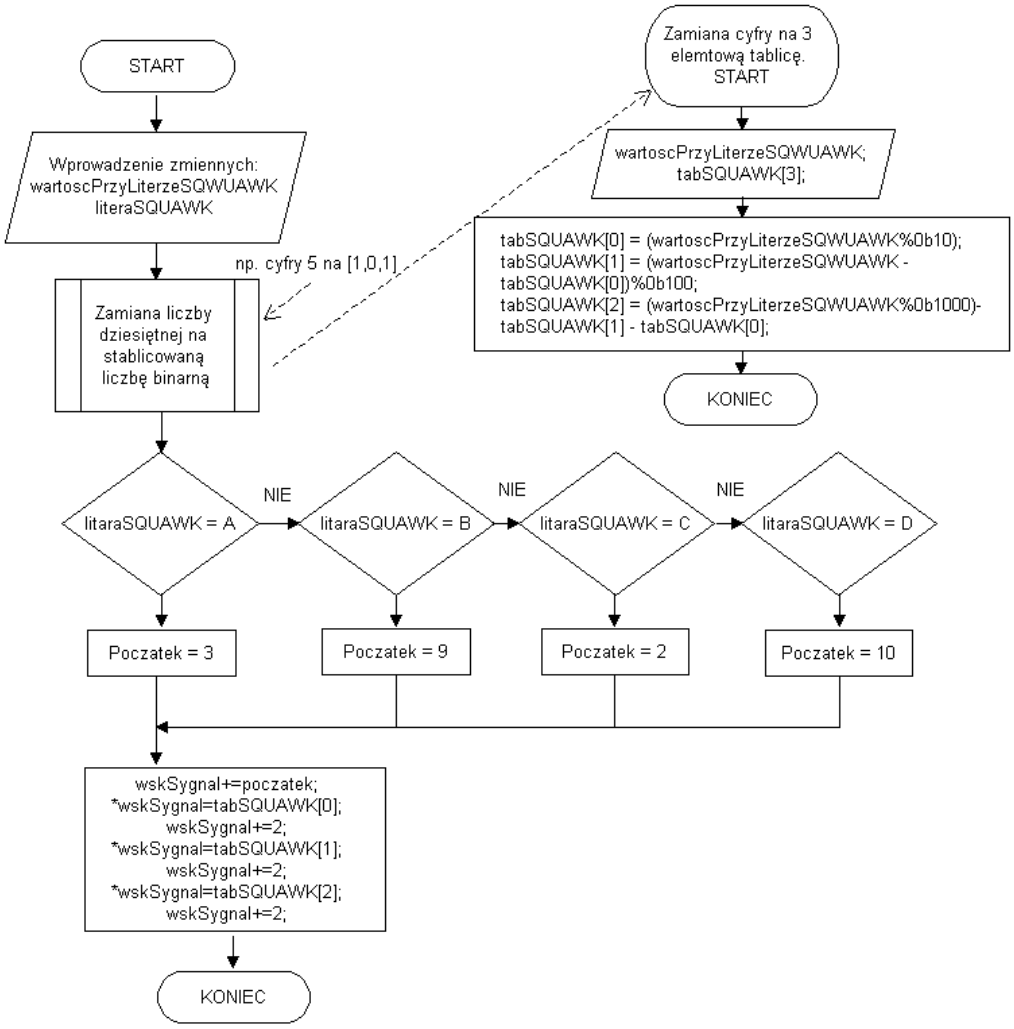
Liczniki, dalej zwane timer’ami, odmierzają czas generowania całości sekwencji kodu jak i pojedynczych impulsów. Szesnastobitowy TIMER1 ustawiony jest na 27 ms, po upłynięciu tego czasu wyłącza wszystkie liczniki także samego siebie, przerywa generację kodu Squawk oraz zeruje zmienna iteracyjną *„i”*, co pozwala wygenerować kod ponownie.

TIMER0 oraz TIMER2 są ośmiobitowymi licznikami. Działają one naprzemiennie. Pierwszy uruchamiany jest TIMER2 odmierzający czas 0,45 ms, czyli czas trwania jednego impulsu, a po odmierzeniu zadanego czasu licznik ten przerywa odliczanie i uruchamia TIMER0 ustawiony na czas 1 ms. TIMER0 odlicza czas przerwy między impulsami i zmienia stan na wyjściu mikrokontrolera na niski lub wysoki, w zależności od wartości elementu tablicy sygnal[i]. Po upływie zadanego czasu 1 ms, licznik ten tak samo jak TIMER2, przestaje odliczać takty procesora oraz włącza TIMER2, który ustawia stan niski na wyjściu mikrokontrolera. Zmienna iteracyjna „i” jest zwiększana w procedurze przerwania dla TIMER0.

.

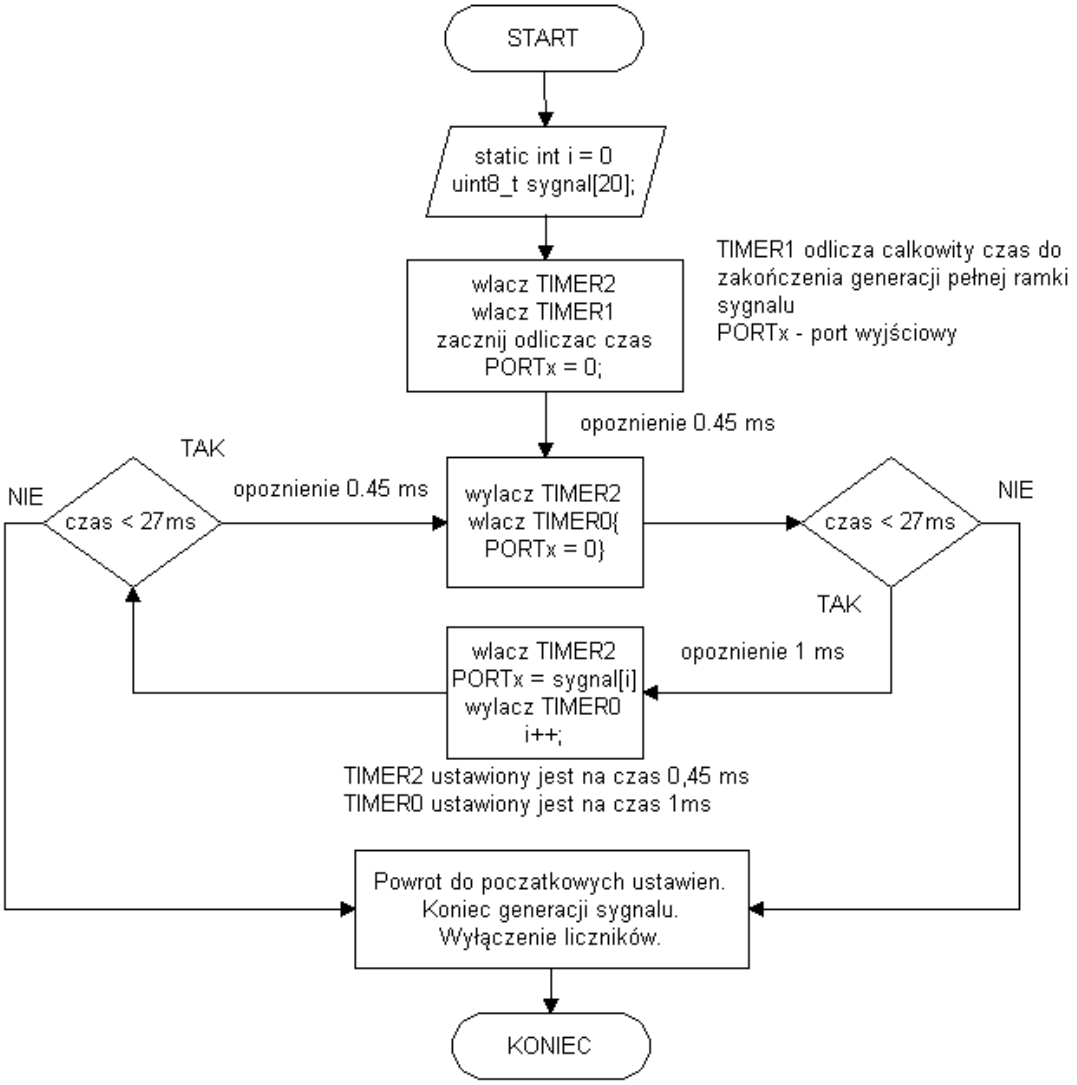


Rysunek 9. Schemat blokowy pętli głównej programu



Rysunek 10. Schemat blokowy funkcji zmianaKoduSQ

.

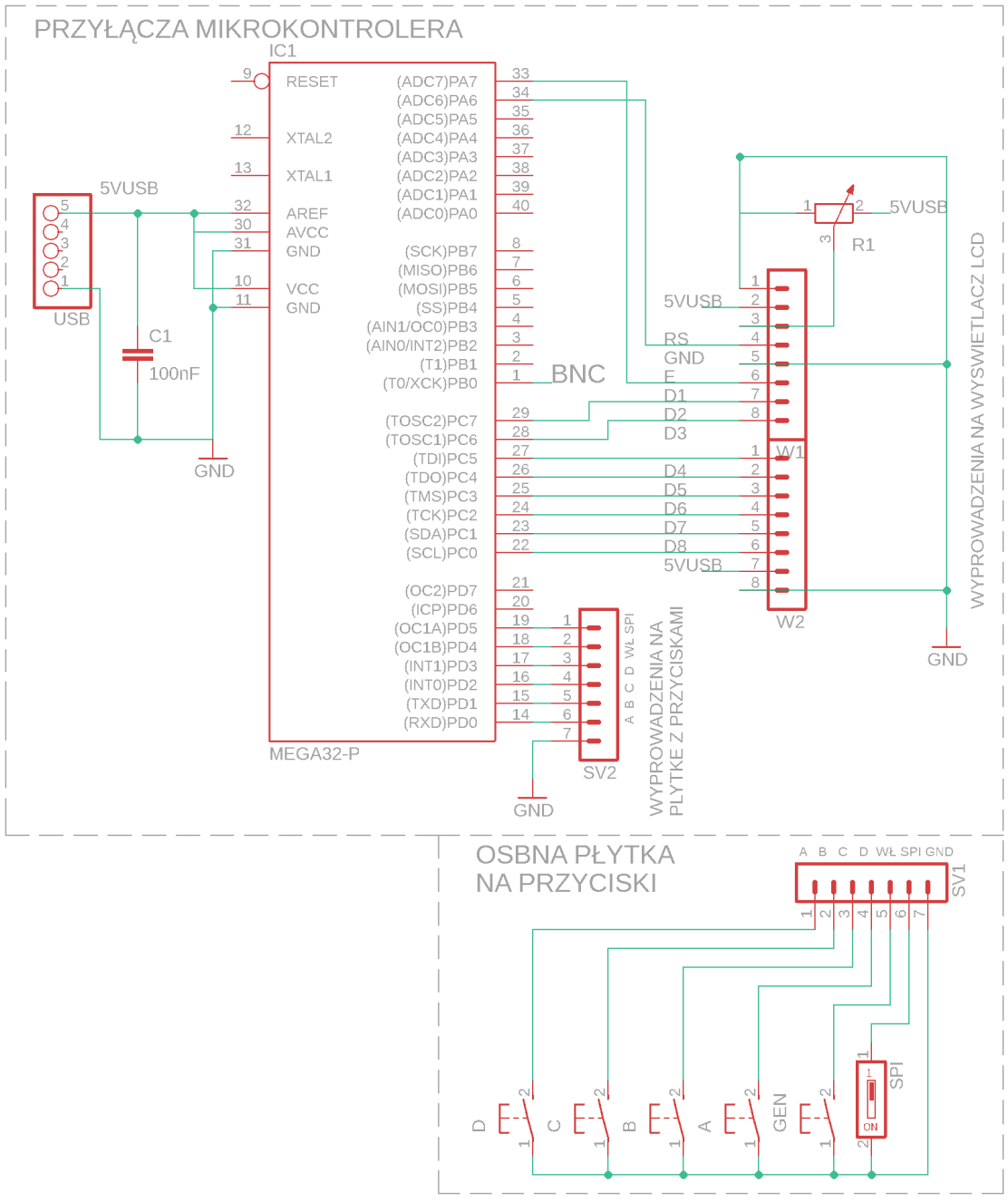


Rysunek 11. Algorytm generowania kodu Squawk

6. Realizacja układu docelowego

W niniejszym rozdziale przedstawione zostaną etapy realizacji docelowego układu elektronicznego. W pierwszym etapie po przetestowaniu zaprogramowanego prototypu narysowano schemat połączeń elektrycznych w programie Eagle. Następnie w tym samym programie i na podstawie wykonanego schematu połączeń powstał projekt płytki PCB. Następnym etapem było wytrawienie płytki metodą fotochemiczną, nawiercenie otworów, wykonanie połączeń lutowanych i przetestowanie prawidłowego działania urządzenia.

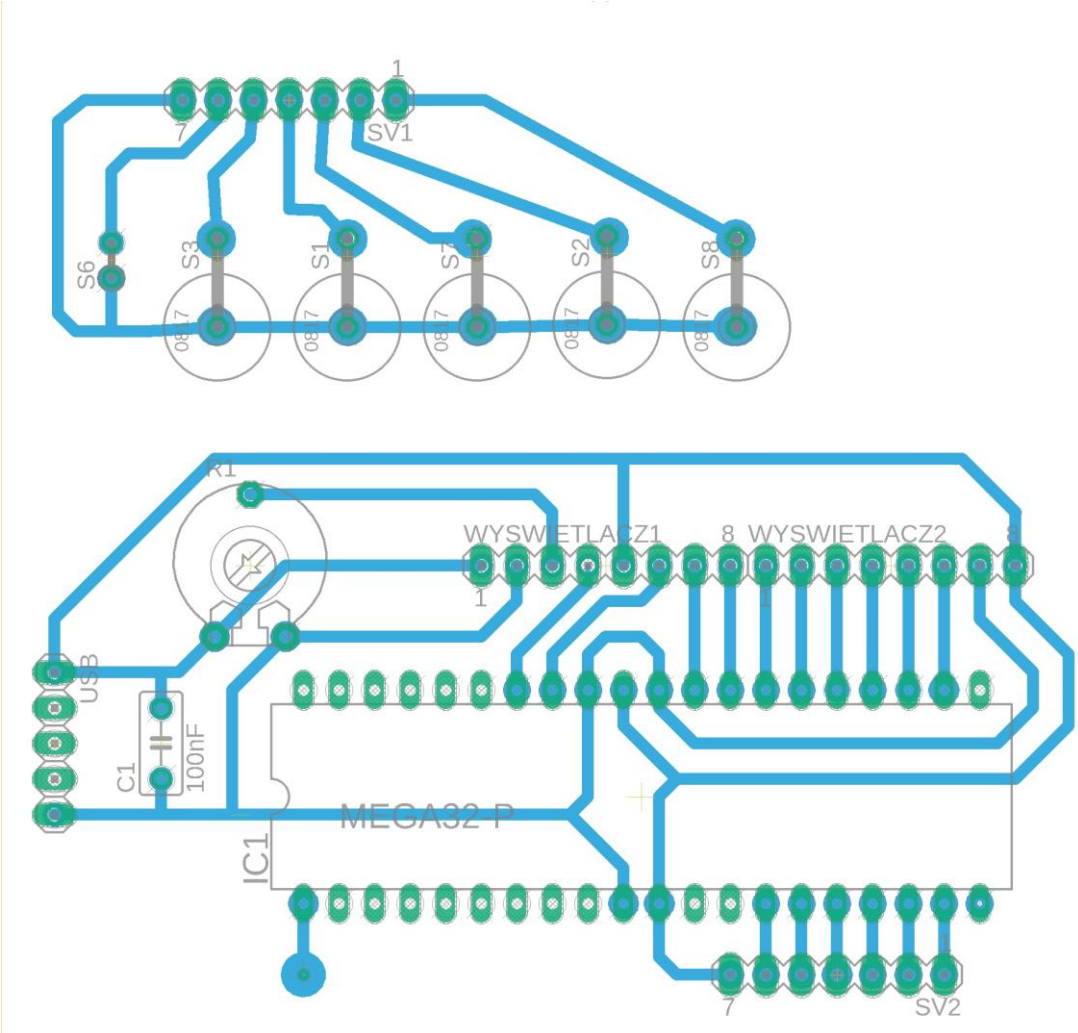
6.1 Schemat elektryczny



Rysunek 32 Schemat elektryczny układu

Układ składa się z trzech części – płytki z zamontowanymi przełącznikami, płytki głównej, na której znajduje się mikrokontroler i z wyświetlacza LCD - połączonych ze sobą przewodami. Główną częścią spajającą urządzenie jest płytka z zamontowanym mikrokontrolerem. Na płytce z mikrokontrolerem jest zamontowane złącze micro USB wykorzystywane do zasilania układu napięciem 5 V, za pomocą np. ładowarki stosowanej do ładowania telefonów komórkowych lub przenośnego „powerbanka'a". Równolegle do złącza USB podłączony jest kondensator o pojemności 100nF którego zadaniem jest ograniczenie zakłóceń. Na głównej płytce znajdują się dwa złącza typu „goldpin". Jedno składające się z szesnastu wyjść służy do komunikacji z wyświetlaczem LCD, natomiast drugie z siedmioma wyjściami przeznaczone jest do połączenia z płytką z przełącznikami. Złącze BNC będące wyjściem sygnału kodu Squawk połączone jest z wyjściem PB1 mikroprocesora.

6.2 Projekt płytki PCB



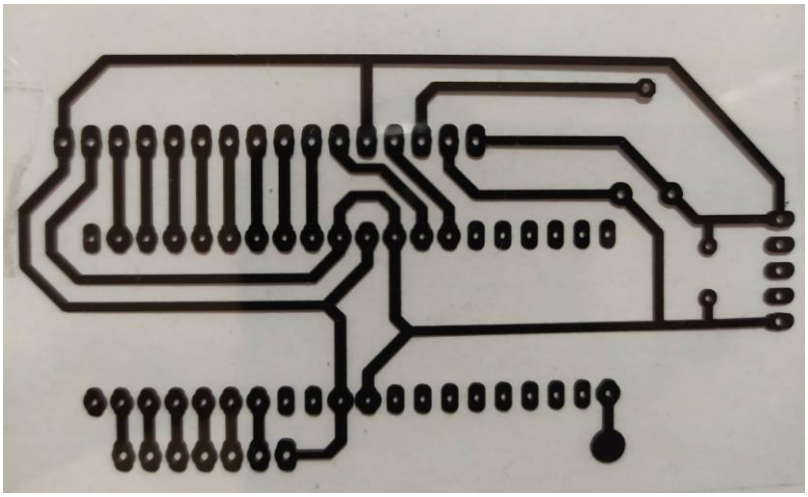
Rysunek 43 Projekt układu na płytce PCB

W górnej części projektu układu widać układ połączeń płytki z przełącznikami. Wyprowadzenia w postaci pinów w gotowym układzie są umieszczone pod spodem płytki, aby przewody nie były widoczne po zamontowaniu układów na obudowie. Na płytce znajduje się miejsce dla pięciu przełączników astabilnych, oraz jednego

przełącznika przesuwnego od którego położenia zależy generacja impulsu SPI. Dodatkowy pin został przewidziany w celu podłączenia masy. W projekcie połączeń są narysowane rezystory zamiast przełączników. W bibliotekach programu Eagle nie występują dokładnie takie same jak używane w pracy elementy, dlatego zostały zastąpione rezystorami o takim samym rozstawie nóżek.

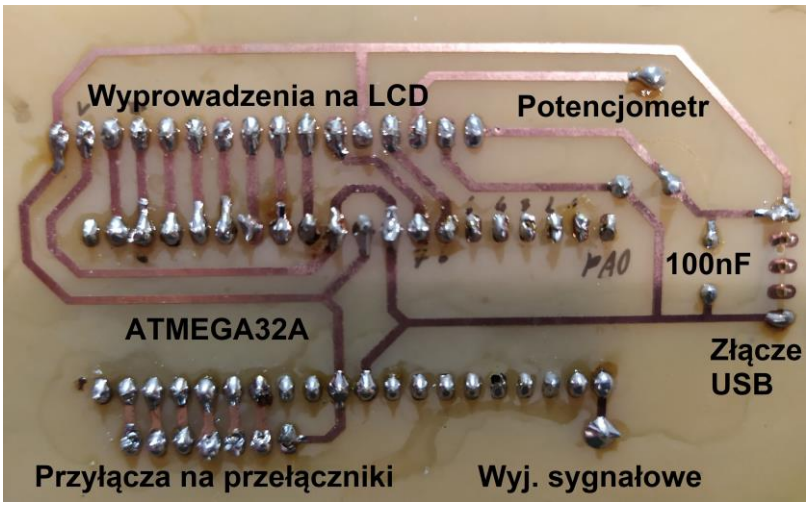
6.3 Wytrawienie i montaż

Następnym etapem praktycznego wykonania pracy było wytrawienie miedzianych połączeń na płytce PCB. W tym celu z programu Eagle został wyeksportowany monochromatyczny, czarno-biały obraz do programu Microsoft Word. Obraz został wydrukowany na folii wykorzystując drukarkę laserową. Czarnym kolorem zostały nadrukowane połączenia elektryczne jakie znajdą się na wytrawionej płytce PCB.

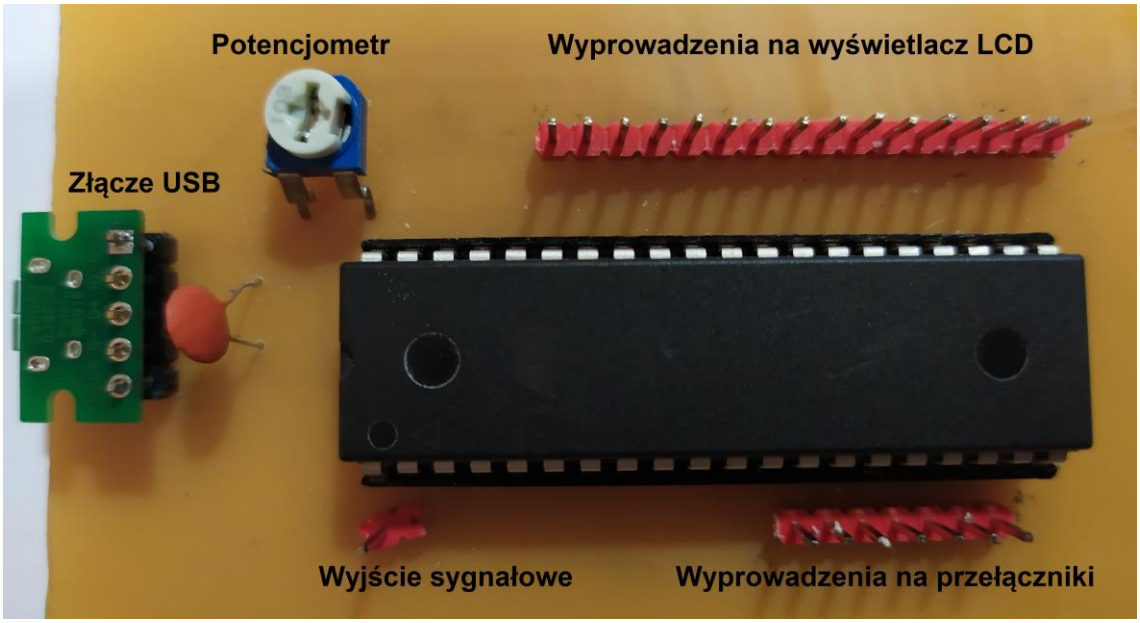


Rysunek 14 Fotomaska wydrukowana na przezroczystej folii.

Wydrukowana fotomaska została nałożona na światłoczułą stronę laminatu, po czym laminat został włożony do naświetlarki. Po wyjęciu z naświetlarki został wywołany w roztworze wywoływacza, do momentu wyraźnego pojawienia się pierwszych śladów ścieżek. Po wywołaniu płytka została umieszczona w roztworze wytrawiacza. Po wytrawieniu strona z połączeniami elektrycznymi została przemyta acetonem w celu usunięcia zanieczyszczeń. Ostatecznie po wytrawieniu zostały nawiercone otwory i przylutowano elementy elektroniczne.

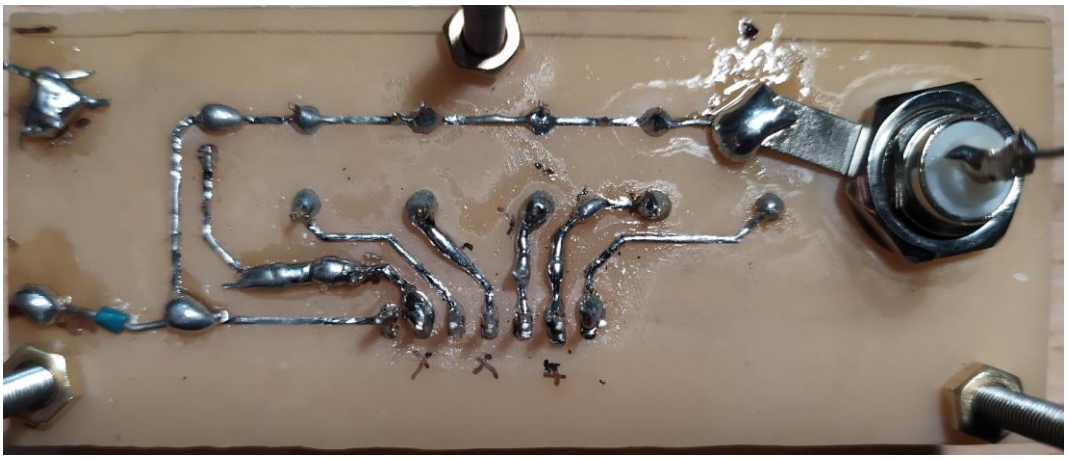


Rysunek 15 Płytka PCB po wytrawieniu i przylutowaniu połączeń - widok od spodu



Rysunek 16 Płytka z mikrokontrolerem - widok od góry

Dokładnie tą samą metodą wytrawiania fotochemicznego, została wykonana druga płytka przeznaczona na przyciski. Złącze BNC oraz piny wystające poza płytkę zostały zamontowane w trakcie praktycznej realizacji układu, dlatego nie są uwzględnione w projekcie płytki PCB wykonanym w programie Eagle. Proces wytrawiania płytki z przełącznikami nie przebiegł pomyślnie. Połączenia elektryczne były w wielu miejscach przerwane, dlatego ścieżki pokryte są cyną.



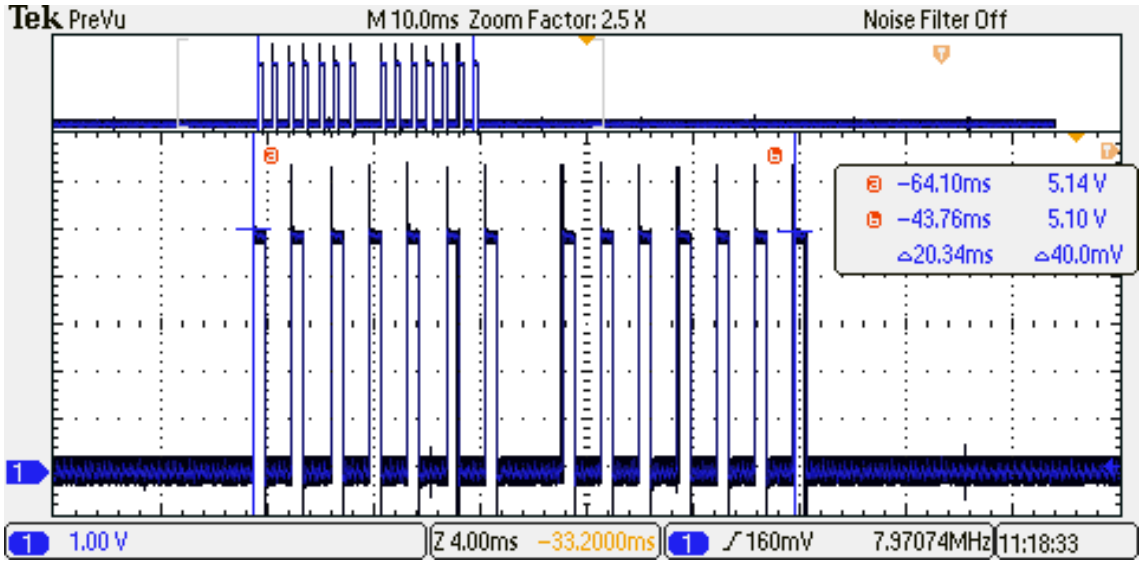
Rysunek 17 Płytka z zamontowanymi przełącznikami – widok od spodu

Rysunek 18 Widok na płytkę z zamontowanymi przełącznikami od góry

Rysunek 195 Widok na układ zamknięty w obudowie

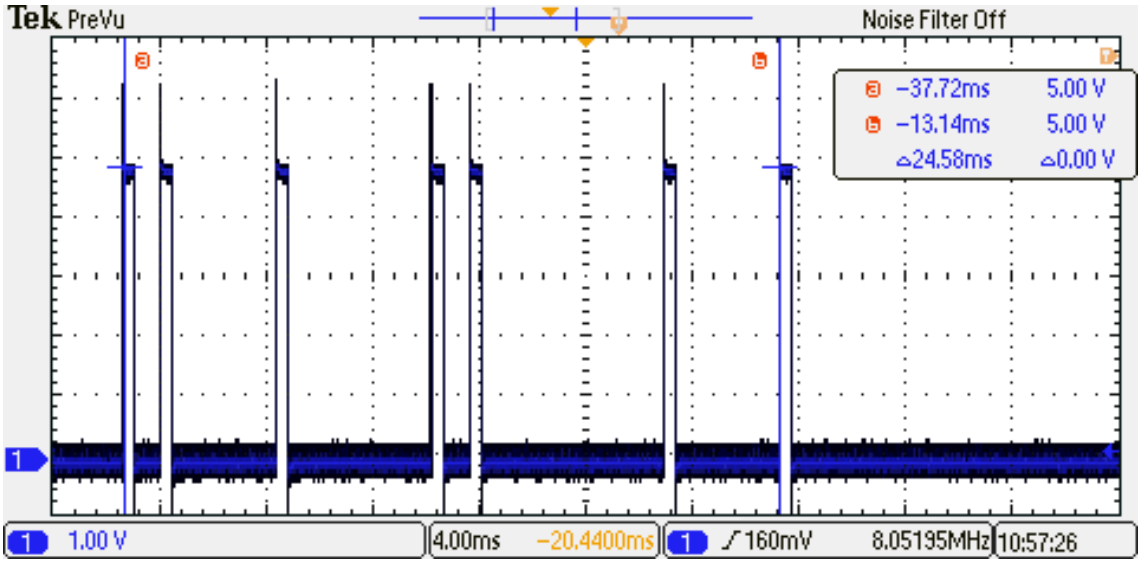
6.4 Badanie układu docelowego

Układ został zbadany z wykorzystaniem oscyloskopu Tektronix MSO 2024B. Poniższe obrazy są zrzutami z ekranu tego urządzenia. Generowany kod Squawk przebadano dla wartości 2111 (kolejne cyfry oznaczają po kolei litery A,B,C,D kodu Squawk), 0000, 7777. Sprawdzono czy prawidłowo generowany jest impulsu SPI. Zmierzone zostały odstępy czasowe między zboczami narastającymi sąsiednich impulsów (powinny wynosić 1,45 ms), między zboczem opadającym a narastającym (1ms), długość pojedynczych impulsów (0,45 ms), odstęp czasowy od impulsu F1 do F2 (20,3 ms) oraz odstęp między ostatnim impulsem F2, a impulsem SPI (4,35 ms). Kursory oznaczone są białymi literami „a” i „b” napisanymi w okręgu na pomarańczowym tle. Zmierzony czas można odczytać z prawej strony zrzutu ekranu, oznaczony jest znakiem Δ.



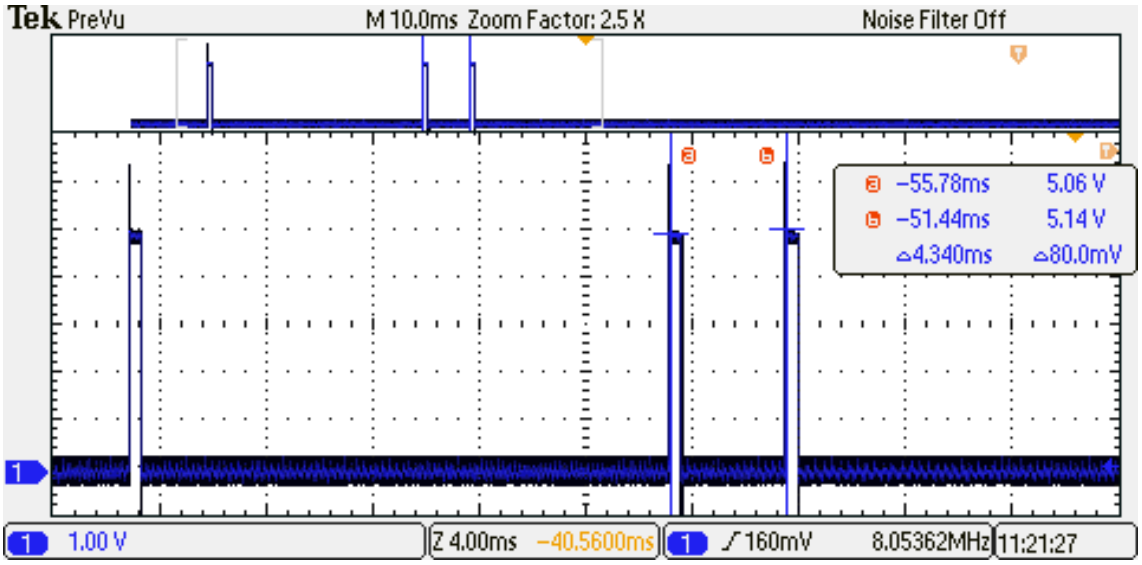
Rysunek 20 kod 7777, czas zmierzony między impulsem F1 i F2

Zmierzony czas pomiędzy impulsami F1 i F2 wynosi 20.34 ms.



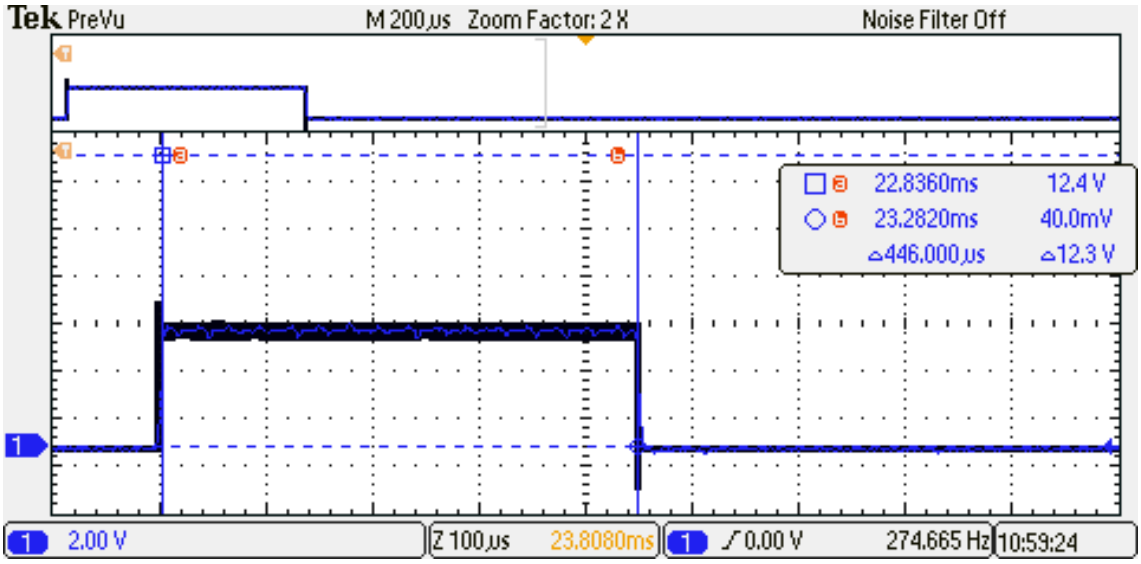
Rysunek 21 kod 2111, czas zmierzony między impulsem F1 i SPI

Zmierzony czas między impulsami F1 i F2 wynosi 24.58 ms.



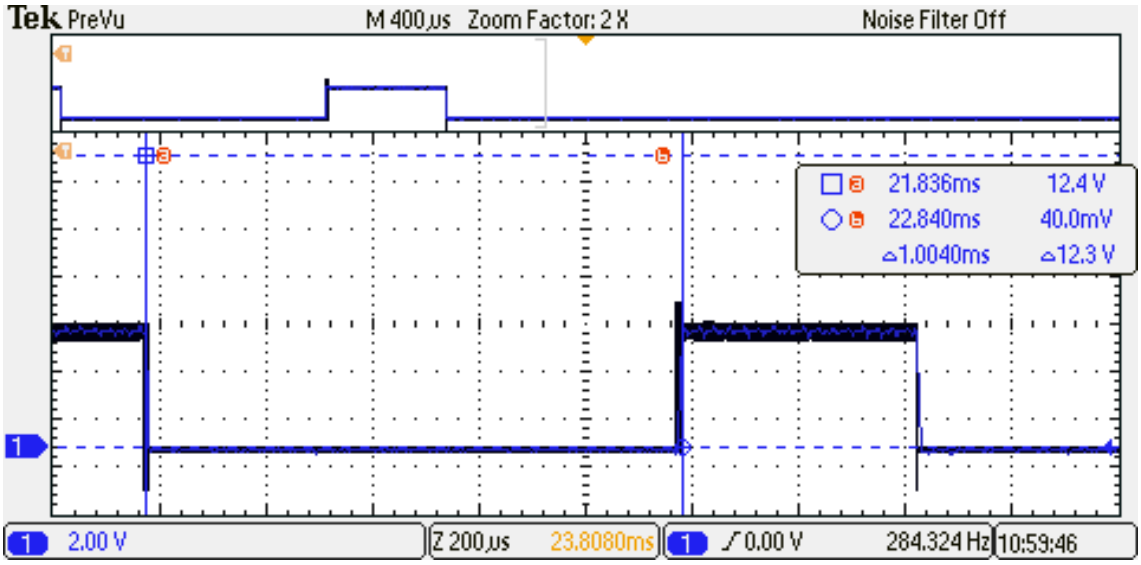
Rysunek 22 kod 0000, czas zmierzony między impulsem F2 i SPI

Zmierzony czas między impulsem F2 i SPI wynosi 4.340 ms.



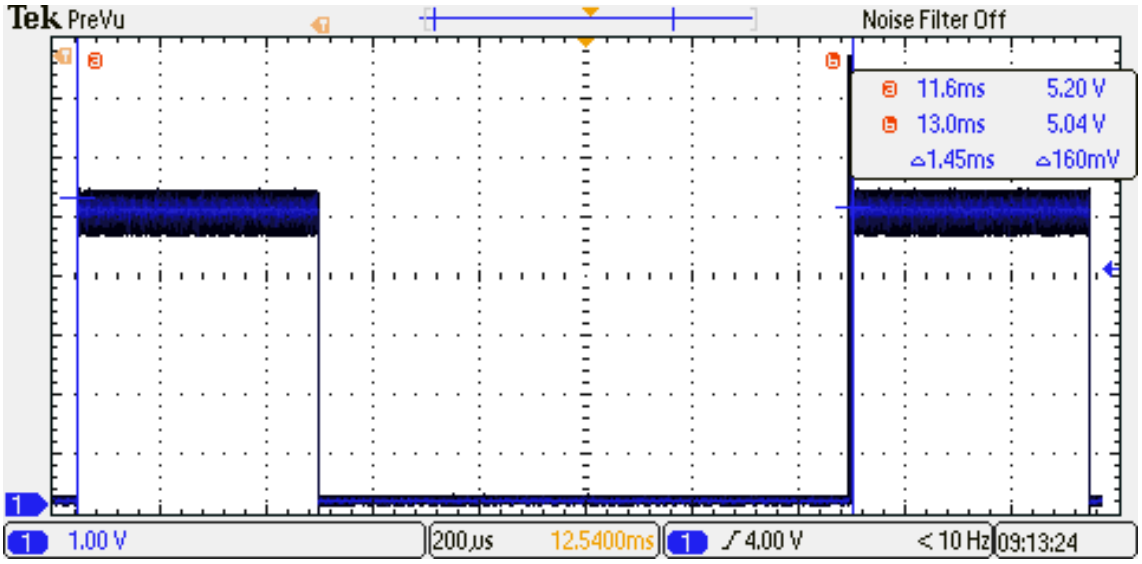
Rysunek 63 Czas trwania pojedynczego impulsu

Zmierzony czas trwanie impulsu 446.000 µs.



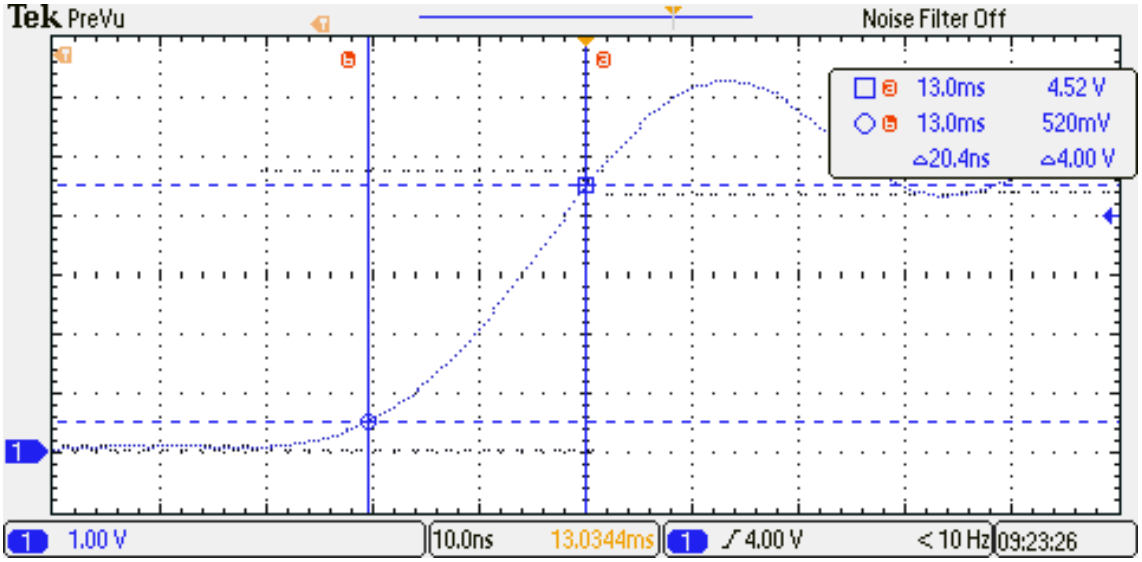
Rysunek 77 Czas zmierzony między zboczem opadającym i narastającym sąsiednich impulsów

Zmierzony czas między zboczem narastającym, a opadającym sąsiednich impulsów wynosi 1.0040 ms.



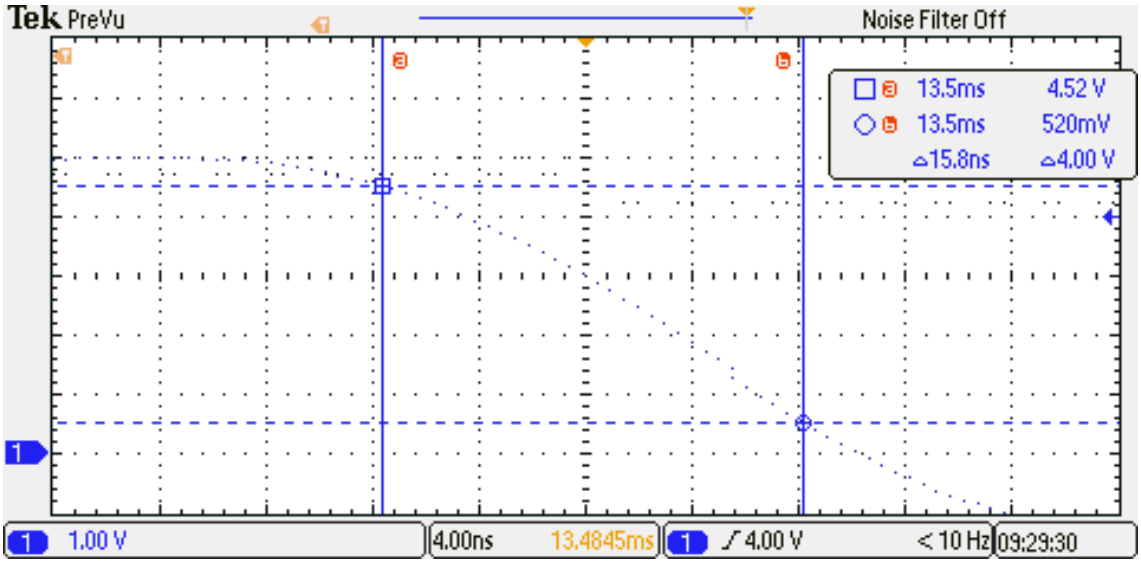
Rysunek 84 Czas zmierzony między zboczami narastającymi sąsiednich impulsów

Zmierzona odległość między sąsiednimi impulsami wynosi 1.45 ms.



Rysunek 95 Pomiar czasu trwania zbocza narastającego

Zmierzony czas narastania zbocza wynosi 20.4 ns.



Rysunek 26 Pomiar czasu trwania zbocza opadającego

Zmierzony czas opadania zbocza wynosi 15.8 ns.

7. Podsumowanie

W trakcie realizacji pracy dyplomowej najtrudniejszym zadaniem było opracowanie algorytmu, który generowałby kod Squawk na wyjściu układu. Niska częstotliwość taktowania procesora uniemożliwiła wygenerowanie sekwencji kodu w czasie 24.65 μs. Wynika to z faktu, że na wykonanie każdej instrukcji programu procesor potrzebuje określonej liczby taktów. Procesor nie był w stanie nadążyć zmieniać sygnału w tak krótkim czasie jak to było potrzebne. Próba rozwiązania tej kwestii polegała na wykorzystaniu wszystkich trzech liczników sprzętowych do generacji kodu Squawk, zamiast tylko jednego. Taka zmiana pozwoliła znacznie skrócić procedury przerwań, które zmieniały stan napięcia na wyjściu układu. Możliwe stało się skrócenie czasu impulsów, jednak praca na wartościach rzędu dziesiątych części mikrosekundy wciąż pozostawała daleko poza możliwościami urządzenia.

Podczas fizycznej budowy układu wystąpiły rozmaite trudności niemal na każdym etapie. Płytka z przełącznikami została nieprawidłowo wytrawiona, co zostało naprawione poprzez poprawienie połączeń lutownicą. Wielokrotnie należało badać układ w poszukiwaniu przerw w obwodzie elektrycznym. W przypadku płytki z mikroprocesorem należało poprawić jedno z połączeń, ponieważ wyświetlacz LCD wyświetlał inne niż zaprogramowane znaki.

Projekt ma szerokie możliwości rozszerzenia funkcjonalności i udoskonalenia. Pierwszym udoskonaleniem mogłoby byłoby skrócenie czasu generowania sygnału, do tego należałoby wykorzystać mikroprocesor o wyższej częstotliwości taktowania. Dobrym rozwiązaniem byłoby wykorzystanie mikrokomputera z rodziny Raspberry Pi których procesory pracują na częstotliwościach ponad 1 GHz. Tańszą alternatywą byłoby zastosowanie mikrokontrolera rodziny STM32 których częstotliwość taktowania procesora jest wielokrotnie wyższa od wykorzystanego w pracy mikroprocesora ATMEGA32A. Pewnym utrudnieniem w przypadku wspomnianych szybszych procesorów jest ich obudowa typu LQFP, a co za tym idzie także sposób montażu w warunkach amatorskich.

W ramach rozwoju projektu, można by było zaprogramować urządzenie do pracy w modzie C i kodowanie informacji o wysokości lotu, określanej na podstawie ciśnienia barometrycznego. Możliwe także jest zaprogramowanie urządzenia aby działało w innych modach pracy transpondera.

Ciekawym projektem byłoby także zaprojektowanie urządzenia które generowałoby sygnały zapytania takie same jak emitowane przez interrogator radaru wtórnego, na które koder wysyłałby odpowiedź.

Skonstruowane w niniejszej pracy dyplomowej urządzenie może znaleźć zastosowanie dydaktyczne przy nauce struktury sygnału odpowiedzi w modzie 3/A. Istnieją także szerokie możliwości rozwoju lub budowania nowych układów bazując na doświadczeniu zdobytym przy projektowaniu omawianego kodera.

Bibliografia:

[1] Mirosław Kardaś, *Mikrokontrolery AVR język C – podstawy programowania,* wydawnictwo Atmel, Szczecin 2013

[2] https://www.radartutorial.eu

[3] http://www.radary.az.pl/

[4] http://blog.katowice-airport.com/identyfikacja-samolotow-2014-07/

[5] http://ww1.microchip.com/downloads/en/DeviceDoc/doc2503.pdf -Nota katalogowa mikrokontrolera ATMEGA 32A

Wykaz obrazów – źródła:

[1] https://boonton.com/resource-library/application-briefs/artmid/1866/articleid/1202/secondary-surveillance-radar-ssr-rf-power-measurement-challenges

[2] Źródło http://www.radary.az.pl/

[3] [4] Źródło: https://www.radartutorial.eu

[5] *Źródło: http://www.aeroelectric.com/articles/Altitude_Encoding/modecascii.txt*

[6] Instrukcja obsługi zestawu uruchomieniowego EvB 5.1 v5 - *https://and-tech.pl/wp-content/uploads/downloads/2013/04/Instrukcja-EvB5.1-v1.pdf*

[7] [8] N*ota katalogowa mikrokontrolera ATMEGA32A*