WOJSKOWA AKADEMIA TECHNICZNA
im. Jarosława Dąbrowskiego
WYDZIAŁ ELEKTRONIKI

PRACA DYPLOMOWA

UKŁAD STEROWANIA WYBRANYMI FUNKCJAMI INSTALACJI W DOMU JEDNORODZINNYM

(temat pracy dyplomowej)

Filip Mikołaj BAGIŃSKI s. Jacka

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

ELEKTRONIKA I TELEKOMUNIKACJA
(kierunek studiów)

INŻYNIERIA SYSTEMÓW BEZPIECZEŃSTWA
(specjalność)

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

dr inż. Marek SUPRONIUK

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

WARSZAWA 2019

WOJSKOWA AKADEMIA TECHNICZNA

im. Jarosława Dąbrowskiego

WYDZIAŁ ELEKTRONIKI

INSTYTUT SYSTEMÓW ELEKTRONICZNYCH

"AKCEPTUJĘ"
DZIEKAN
WYDZIAŁU ELEKTRONIKI
prof. dr hab. inż. Andrzej DOBROWOLSKI

Dnia 24 WRZ 2018 r.

ZADANIE

do pracy dyplomowej

Wydane studentowi Filipowi Mikołajowi BAGIŃSKIEMU
(stopień, imiona i nazwisko)

ELEKTRONIKA I TELEKOMUNIKACJA
(kierunek studiów)

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

I. Temat pracy: UKŁAD STEROWANIA WYBRANYMI FUNKCJAMI INSTALACJI W DOMU JEDNORODZINNYM

II. Treść zadania:

1. Przegląd zagadnień związanych z automatyką budynkową.
2. Projekt systemu sterowania.
3. Realizacja poszczególnych układów systemu.
4. Badania układów.
5. Podsumowanie i wnioski.

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: mgr inż. Piotr Paziewski

VI. Opiekun merytoryczny: dr hab. inż. Tadeusz Dąbrowski

VII. Termin zdania przez studenta ukończonej pracy: **23.01.2019**

VIII. Data wydania zadania: **04.10.2018**

PROMOTOR PRACY DYPLOMOWEJ

dr inż. Marek SUPRONIUK

(tytuł./stopień naukowy, imię, nazwisko, podpis)

DYREKTOR INSTYTUTU

dr hab. inż. Zbigniew WATRAL, prof. WAT

(tytuł./stopień naukowy, imię, nazwisko, podpis)

Zadanie otrzymałem dnia 04.10. 20 18 r.

Filip Bagański

(podpis studenta)

SPIS TREŚCI

Wstęp

Pierwsze systemy automatyki domowej zostały stworzone już w ubiegłym wieku. Zapewniają wygodę oraz bezpieczeństwo osób użytkujących, a także umożliwiają obniżenie kosztów korzystania z energii elektrycznej. Pojęcie automatyki domowej obejmuje między innymi systemy sterujące oświetleniem, obsługujące rolety, regulujące temperaturę w pomieszczeniu, a także alarmowe, czy informujące o zagrożeniu życia lub zdrowia [1]. Bywają też bardzo pomocne dla osób o ograniczonych zdolnościach ruchowych.

W dzisiejszych czasach coraz częściej spotyka się domy wyposażone w systemy zarządzające poszczególnymi funkcjami instalacji domowej. Pomimo spadku cen elementów elektronicznych w ostatnich latach ceny systemów automatyki domowej są wciąż wysokie. Niniejsza praca ma na celu stworzenie taniego systemu, który będzie sterować głównym oświetleniem w pomieszczeniu, gniazdem elektrycznym oraz roletą. Komunikacja pomiędzy układami realizowana będzie bezprzewodowo. W ramach pracy zostaną zaprojektowane i zmontowane na płytkach uniwersalnych układ nadawczy w formie pilota posiadającego pięć przycisków wraz z nadajnikiem podczerwieni oraz układy odbiorcze realizujące poszczególne funkcje.

1. Przegląd wybranych systemów automatyki domowej

Na rynku jest wiele systemów automatyki domowej różnych producentów, które odróżnia wykorzystana technologia, cena, sposób komunikacji pomiędzy układami oraz realizowane funkcje.

1.1 Systemy bezprzewodowe

Systemy bezprzewodowe podczas montażu nie wymagają ingerencji w już istniejącą instalację elektryczna. Z tego powodu zazwyczaj jego montaż jest łatwy. Wykorzystując transmisję radiową w komunikacji pomiędzy układami mogą wystąpić zakłócenia. Poważną wadą takich systemów jest konieczność użycia zasilania bateryjnego dla niektórych urządzeń, co wymaga wymiany baterii po pewnym czasie.

a. Fibaro

Jeden z polskich systemów bezprzewodowych automatyki domowej, który umożliwia na przykład sterowanie oświetleniem, roletami, zdalne monitorowanie budynku, a także może być zintegrowany z systemami grzewczymi czy zabezpieczającymi. Głównym elementem systemu jest centrala, która kontroluje całą jego prace. Urządzenia systemu Fibaro zawierają specjalny układ obsługującego protokół Z-wave w celu komunikacji pomiędzy urządzeniami. Protokół ten używa pasma o wartości 868,42 megaherców [2]. Sterowanie poszczególnymi modułami spełniającymi określone funkcje odbywa się przy pomocy zwykłych włączników lub korzystając z urządzeń mobilnych, czy poprzez stronę internetową.

b. xComfort

Kolejnym przykładem systemu bezprzewodowego może być xComfort, który ma możliwość sterowania oświetleniem, temperaturą pomieszczenia, roletami oraz innymi urządzeniami. Dodatkowo producent dostarcza czujniki pozwalające na monitorowanie stanu temperatury w pomieszczeniu, detekcję zalania, dymu, czy ruchu. Posiada główną jednostkę zarządzającą całym systemem, a sterowanie odbywa się przy użyciu sygnałów wysyłanych do odbiorników sterujących poszczególnymi układami, które realizują odpowiednią funkcję. Użytkownik obsługuje go przy pomocy aplikacji mobilnej, panelu zamontowanego na ścianie, czy programowalnego pilota sterującego [4].

1.2 Systemy kablowe

Charakteryzują się przewodową komunikacją pomiędzy układami dzięki czemu transmisja nie jest podatna na zakłócenia radiowe. Przeważnie są droższe od bezprzewodowych ,a podczas montażu wymagają specjalistycznej wiedzy na jego temat.

a. KNX

Komunikacja pomiędzy układami tego systemu może być realizowana przy użyciu wielu mediów transmisyjnych. Poprzez instalację elektryczną w budynku, fale radiowe, czy sieć Ethernet. Jednak najczęściej stosowanym medium transmisji jest skrętka dwuparowa. Nie posiada centrali, która zarządzałaby całą instalacją. Jest systemem rozproszonym. Korzysta z Europejskiej Magistrali Komunikacyjnej opracowanej wspólnie przez europejskich producentów systemów automatyki budynkowej. Stworzony standard komunikacyjny umożliwia dołączenie do systemu urządzeń pochodzących od innych producentów. Instalacja systemu KNX wymaga posiadania sporej wiedzy na jego temat, a także surowego stanu budynku. Dzięki niemu możliwe jest między innymi sterowanie oświetleniem, temperaturą pomieszczenia, roletami, kontrolą dostępu czy też zarządzanie systemem alarmowym [6].

b. Nexo

Polski system firmy Nexwell może sterować oświetleniem, roletami, regulować temperaturę pomieszczenia oraz zarządzać systemem alarmowym. Współpracuje wyłącznie z urządzeniami producenta. Zarządzanie poszczególnymi modułami i podejmowanie decyzji odbywa się poprzez centralę. W razie uszkodzenia centrali nie będą dostępne zaawansowane funkcje, lecz będzie można sterować poszczególnymi funkcjami przy użyciu włączników. System może być obsługiwany za pomocą aplikacji mobilnej NexoVision, strony internetowej oraz wiadomości SMS. Centrala przeważnie połączona jest z modułami przewodowo. Jednak jest również możliwość komunikacji bezprzewodowej z modułami, które zostały do tego zaprojektowane[2].

2. Przedstawienie projektu

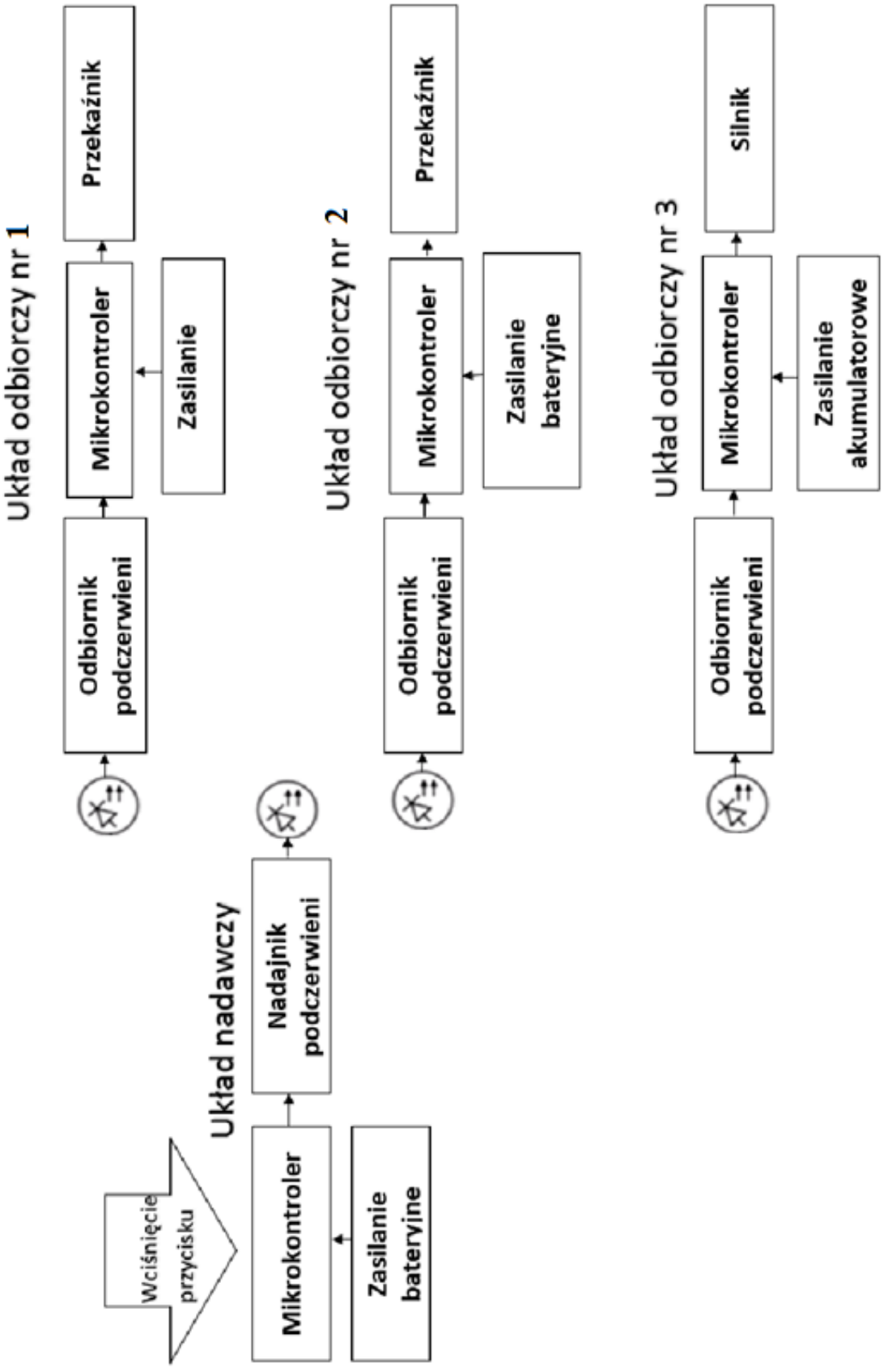
Celem projektu jest wykonanie czterech urządzeń. Jednego układu nadawczego oraz trzech układów odbiorczych, które mają pełnić funkcję prostego systemu realizującego poszczególne funkcje automatyki domowej.

2.1. Założenia projektu

Zadaniem układu nadawczego jest możliwość przesyłania sygnału w podczerwieni przy użyciu standardu RC5, który jest odbierany i dekodowany przez układy odbiorcze. W zależności od użytego przycisku układu nadawczego przypisany do niego układ odbiorczy reaguje i wykonuje odpowiednią funkcję.

Projekt ma pokazać, że przy niewielkim nakładzie finansowym i łatwo dostępnych elementach elektronicznych jest możliwość stworzenia prostego systemu sterującego wybranymi funkcjami instalacji w domu.

Na rys. 2.1. przedstawiony jest schemat blokowy opisywanego systemu składający się z poszczególnych modułów układu nadawczego oraz modułów układów odbiorczych. Każdy z układów posiada moduł odpowiedzialny za zasilanie. Układ odbiorczy nr 1 jest zasilany z sieci, układ odbiorczy nr 2 bateryjnie, a układ odbiorczy nr 3 za pomocą akumulatora. Układ nadawczy zawiera mikrokontroler, który odpowiedzialny jest za reakcję na odpowiedni przycisk i wysłanie zakodowanej ramki danych poprzez nadajnik podczerwieni. Natomiast zadaniem mikrokontrolerów znajdujących się w układach odbiorczych jest odebranie zakodowanej ramki danych korzystając z odbiornika podczerwieni i wykonaniu odpowiedniej funkcji jeżeli wysłana ramka danych skierowana jest do tego układu, a jeśli nie to mikrokontroler porzuca odebraną ramkę danych i oczekuje na odebranie następnej. Układ odbiorczy nr 1 włącza i wyłącza urządzenie podłączone do sieci. Zadaniem układu odbiorczego nr 2 jest sterowanie oświetleniem, a układ odbiorczy nr 3 obsługuje silnik sterujący roletą.

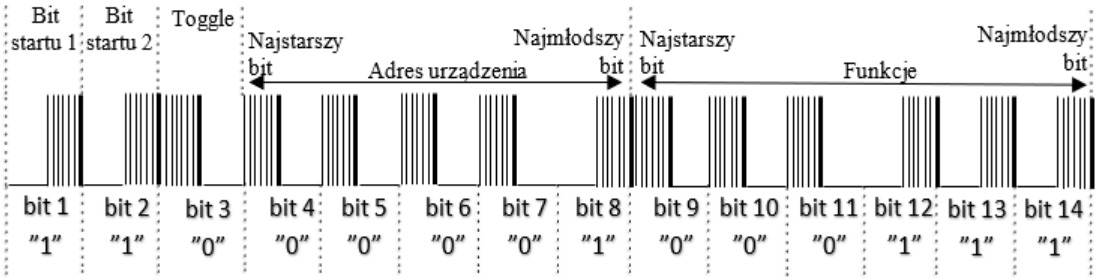


Rys. 2.1 Schemat blokowy systemu sterującego wybranymi funkcjami instalacji (opracowanie własne)

2.2. Opis standardu RC5

Wykorzystywany w projekcie standard RC5 jest jednym z wielu standardów kodowania sygnałów w podczerwieni. Został stworzony przez firmę Philips ponad 30 lat temu w celu przesyłania sygnałów z pilotów podczerwieni używanych w sprzęcie RTV.

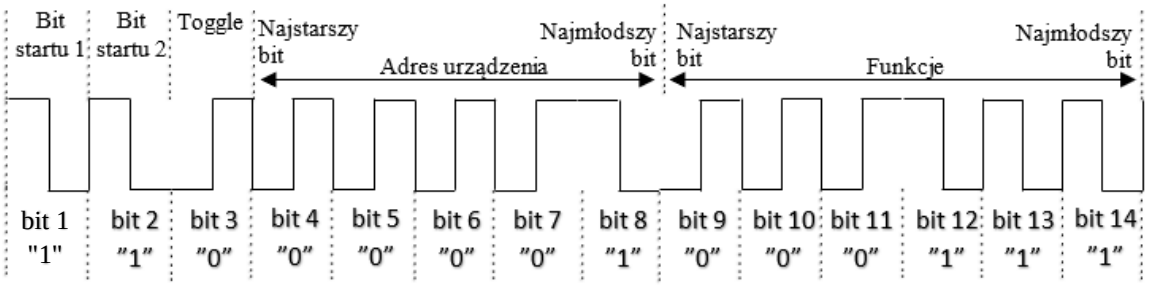
Zgodnie z tym standardem ramka danych wysyłana z układu nadawczego zawiera 14 bitów. Trzy początkowe bity to dwa bity startu informujące układ odbiorczy o nadchodzącej ramce danych oraz bit toggle, który zmienia swój stan na przeciwny po każdym wciśnięciu przycisku. Później występuje pięć bitów adresu przechowujących adres urządzenia do którego przesyłana jest ramka. Dzięki nim układ odbiorczy ma możliwość wykrycia czy dana ramka wysyłana przez układ nadawczy jest skierowana dla niego. Następnie 6 bitów określających funkcję do wykonania przez układ odbiorczy. Opisywany standard bazuje na kodowaniu bifazowym. Oznacza to, że każdy przesyłany bit w ramce danych określony jest przez dwa impulsy o przeciwnych stanach logicznych. Każdy impuls trwa przez dokładnie określony fragment czasu, w tym wypadku jest to około 889 mikrosekund. Dlatego też czas trwania całego bitu wynosi około 1778 mikrosekund. Podczas transmisji wykorzystywana jest częstotliwość nośna równa 36 kiloherców, która jest odpowiednio modulowana w celu uzyskania bitu o pożądanym stanie logicznym [7]. Przykładowa ramka danych wysyłana przez układ nadawczy znajduje się na rys. 2.2.



Rys. 2.2. Przykładowa konstrukcja ramki danych w standardzie RC5 wysyłanej przez układ nadawczy (opracowanie własne)

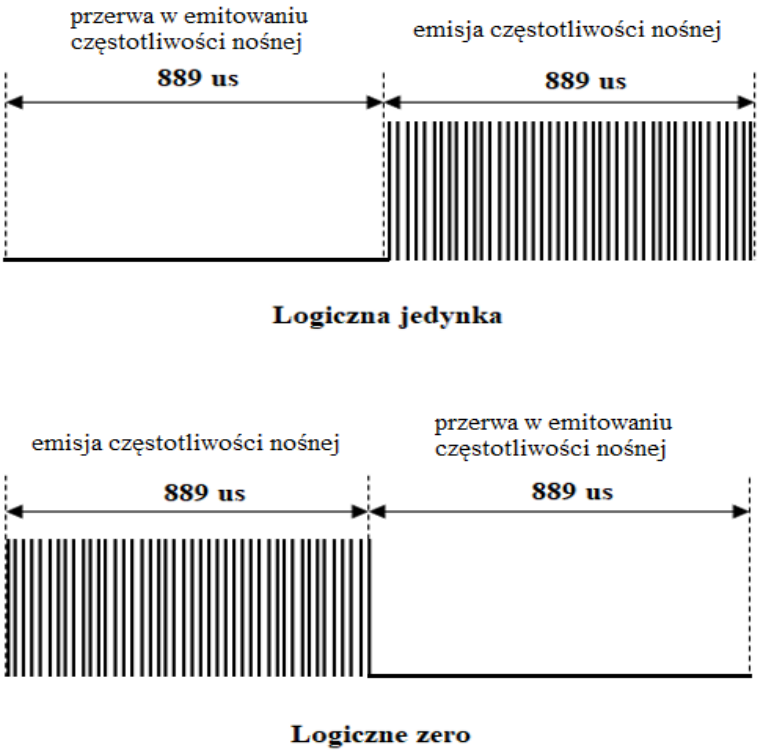
Od strony części odbiorczej po odfiltrowaniu częstotliwości nośnej przez odbiornik podczerwieni na jego pinie sygnałowym generowana jest zanegowana ramka danych. Negacja otrzymanej ramki spowodowana jest występującym tranzystorem bipolarnym na jego wyjściu.

Na rys. 2.3 pokazano ramkę widzianą od strony mikrokontrolera w układzie odbiorczym wysłanej przez układ nadawczy w takiej konfiguracji, jak przedstawia rys. 2.2.



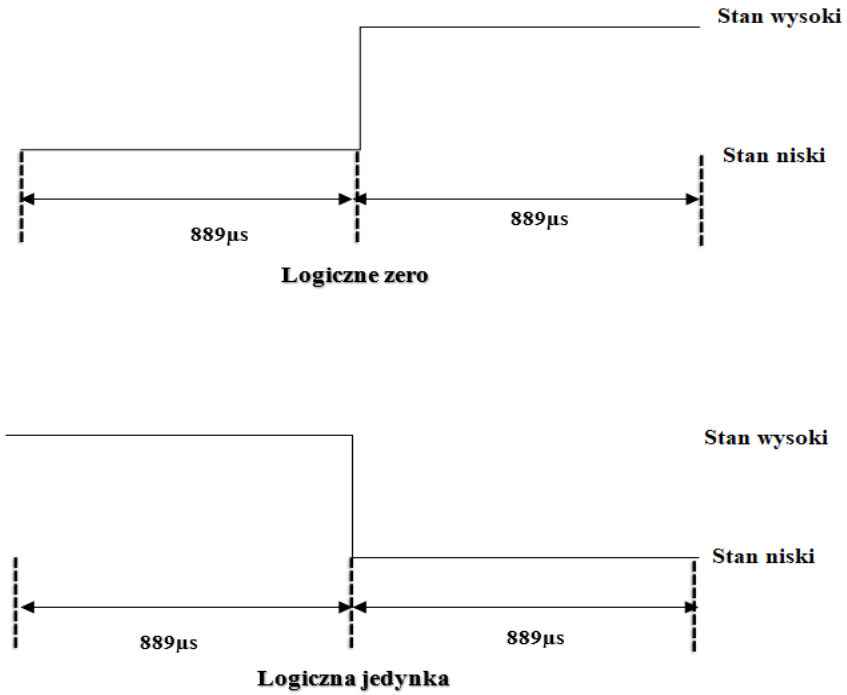
Rys. 2.3 Konstrukcja ramki danych odebranej przez mikrokontroler w układzie odbiorczym (opracowanie własne)

Jeżeli układ nadawczy ma wysłać bit o wartości logicznej jeden potrzebna jest przerwa w nadawaniu częstotliwości nośnej przez 889 mikrosekund, a następnie jej emitowanie przez 889 mikrosekund. Podczas wysyłania bitu o wartości logicznej zero należy pierw generować częstotliwość nośną po czym następuje przerwa w jej emisji także w ciągu 889 mikrosekund. Rys. 2.4 przedstawia, jak wysłać bit o pożądanych stanach logicznych dla standardu RC5. Można zauważyć, że od strony układu nadawczego, każdy bit niosący informację posiada zbocze w połowie okresu czasu trwania bitu [7].



Rys. 2.4 Reprezentacja stanów logicznych dla układu nadawczego z częstotliwością nośną 36 kiloherców (opracowanie własne)

Dla wartości logicznej jeden jest to zbocze narastające, a bit o wartości logicznej zero posiada zbocze opadające. W przypadku układu odbiorczego po odfiltrowaniu częstotliwości nośnej sygnału przez odbiornik podczerwieni otrzymujemy poszczególne bity ramki danych w postaci dwóch impulsów o różnych stanach. Mikrokontroler układu odbiorczego wykrywa stan niski na pinie sygnałowym odbiornika podczerwieni podczas emitowania częstotliwości nośnej przez układ nadawczy, a wykrywa stan wysoki podczas braku nadawania częstotliwości nośnej, co pokazuje rys. 2.5. Występujący na wyjściu odbiornika podczerwieni tranzystor bipolarny wchodzi w stan nasycenia po wykryciu fali nośnej, dlatego też odebranie sygnału w postaci fali nośnej jest sygnalizowane stanem niskim. Podczas braku emitowania fali nośnej przez układ nadawczy na wyjściu odbiornika podczerwieni panuje stan wysoki, co zapewnia rezystor podciągający zawarty w jego budowie. Oznacza to, że stany logiczne poszczególnych bitów wysłanych przez układ nadawczy są negowane przez odbiornik podczerwieni, co należy uwzględnić podczas pisania kodu źródłowego dla układu odbiorczego.



Rys. 2.5 Reprezentacja stanów logiczny dla układu odbiorczego po odfiltrowaniu częstotliwości nośnej przez odbiornik podczerwieni (opracowanie własne)

Przedstawione wyżej informacje są wystarczające, aby napisać kod źródłowy dla układu nadawczego w celu nadawania ramki danych w standardzie RC5 oraz dla układów odbiorczych, które będą miały możliwość dekodowania nadsyłanych ramek danych.

2.3. Dobór platformy

Ważną decyzją jest dobór platformy na której będzie opierał się cały system. Obecnie na rynku jest wiele dostępnych urządzeń takich, jak mikrokontrolery, płytki rozwojowe, czy też komputery jednopłytkowe.

a. Raspberry Pi

Platforma ta jest jednopłytkowym komputerem osobistym. Posiada własny system operacyjny Raspbian, który jest częścią rodziny Linux. Raspberry Pi wyposażony jest w procesor BCM2837, który jest wyposażony w cztery 64-bitowe rdzenie Cortex-A53 [10]. Na płytce znajdują się moduły wi-fi oraz Bluetooth. Jest też złącze HDMI służące do podłączenia monitora oraz złącza USB, dzięki którym możemy podłączyć mysz oraz klawiaturę. Zasilanie doprowadzone jest przez złącze USB. W projekcie nie zostanie wykorzystana opisywana platforma z powodu nadmiernej ilości zbędnych modułów oraz trudności napisania kodu programu, który miałby na celu realizować transmisję przy wykorzystaniu standardu RC5.

b. Arduino

Gotowy zestaw uruchomieniowym, który został stworzony, dla osób chcących nauczyć się pisać programy dla mikrokontrolerów, ale nie miały z tym za dużo wspólnego. Platforma wykorzystuje mikrokontrolery z rodziny AVR. Zasilanie odbywa się przez złącze USB znajdujące się na płytce ewaluacyjnej. Do wgrania kodu źródłowego nie jest potrzebny zewnętrzny kompilator, ponieważ płytka wyposażona jest już w zaprogramowany mikrokontroler służący do komunikacji z komputerem. Programowanie Arduino wspierane jest przez oprogramowanie komputerowe, które zawiera biblioteki ułatwiające pisanie kodu źródłowego mikrokontrolera, dzięki czemu nie trzeba znać poszczególnych nazw rejestrów mikrokontrolera AVR. Arduino nie zostanie wykorzystana w projekcie z powodu miejsca w obudowie jakie urządzenia mogłyby zajmować, gdyby została użyta ta platforma [11].

c. STM32

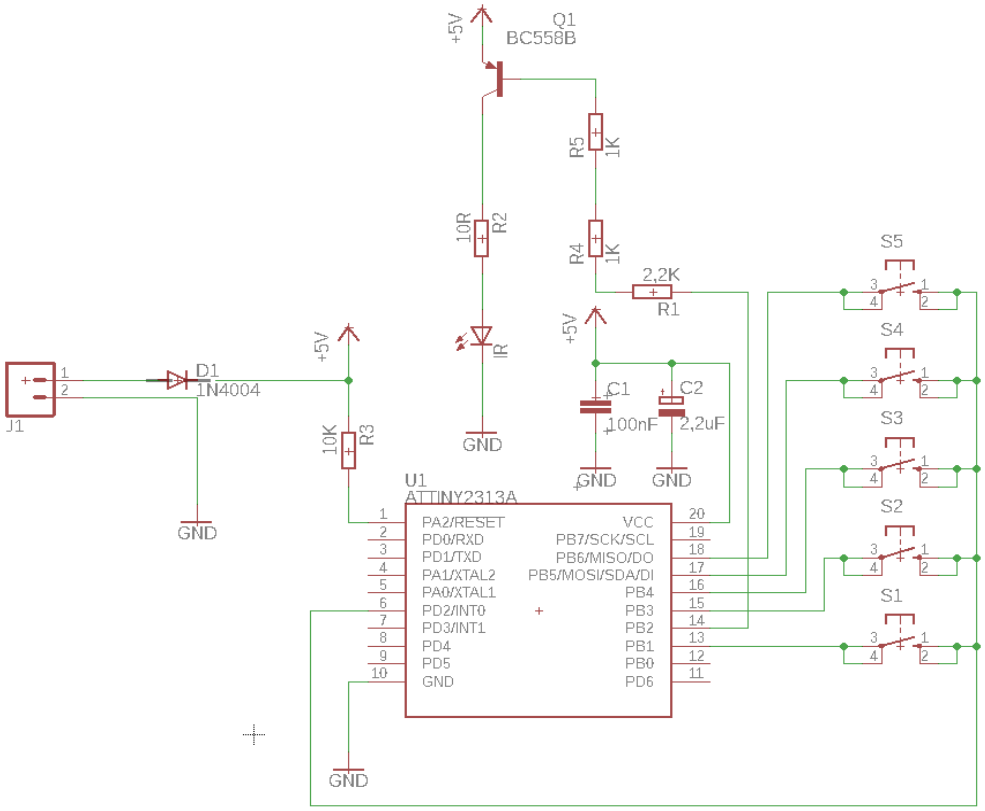
Są to mikrokontrolery 32-bitowe. W swojej strukturze zawiera wiele portów wejścia/wyjścia, zegarów, konwerterów analogowo-cyfrowych oraz wielu innych modułów. Producent udostępnia pokaźną ilość bibliotek pomagających napisać kodu źródłowy, ponieważ napisanie programu dla mikrokontrolera 32-bitowego nie jest łatwym zadaniem i zazwyczaj wymaga dobrej znajomości architektury mikrokontrolera oraz wygenerowania dużej ilości kodu w celu wykonania prostych funkcji dlatego nie będzie on wykorzystany w niniejszym projekcie.

d. AVR

Rodzina tych 8-bitowych mikrokontrolerów dzieli się na dwa typy. Jednym z nich jest ATtiny, którego zasoby sprzętowe są stosunkowe małe, niewielka ilość pamięci operacyjnej oraz przechowującej kod źródłowy. Posiadają przynajmniej jeden zegar sprzętowy. ATtiny umieszczone jest w małej obudowie, która zawiera 6 pinów w najmniejszej wersji. Kolejnym typem tej rodziny mikrokontrolerów jest ATmega, który ma większe zasoby sprzętowe, niż ATtiny. Posiada więcej pamięci operacyjnej oraz pamięci FLASH. Ma przynajmniej dwa zegary sprzętowe. W swojej strukturze zawiera przetworniki analogowo-cyfrowe, a jego budowa zawiera do 100 pinów. Producent tych mikrokontrolerów udostępnia biblioteki przez co dostęp do poszczególnych rejestrów mikrokontrolera zarządzających jego pracą jest bardzo prosty. Z uwagi na małą zajętość miejsca oraz prostotę programowania tych mikrokontrolerów oba typy zostały użyte w niniejszym projekcie.

3. Układ nadawczy

3.1. Budowa układu nadawczego

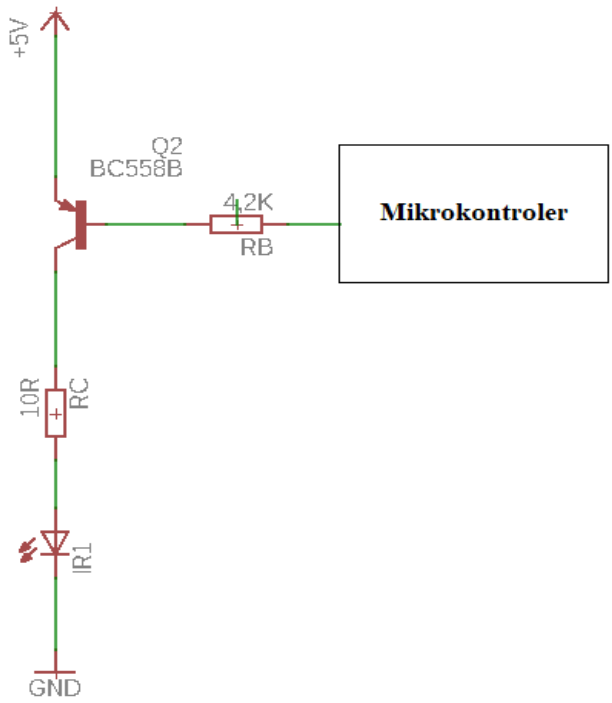


Rys. 3.1 Schemat połączeń dla układu nadawczego (opracowanie własne)

Na rysunku 3.1 znajduje się schemat połączeń pomiędzy elementami układu nadawczego. Układem zarządza niewielki mikrokontroler ATtiny2313 do którego doprowadzonych jest pięć przycisków. Każdy z klawiszy podłączony jest wspólnym połączeniem do pinu PD2, który wykorzystano do wykrywania przerwań zewnętrznych w tym przypadku wywołanych naciśnięciem przycisku. Z drugiej strony każdy przycisk podłączony jest do innego pinu portu B, aby mikrokontroler był w stanie zidentyfikować naciskany przycisk. Rezystory R1, R4 i R5 zostały użyte w celu ograniczenia natężenia prądu płynącego przez bazę tranzystora. Pin PA2 należy doprowadzić do dodatniej szyny zasilania, ponieważ jedną z ról tego pinu jest resetowanie mikrokontrolera, jeżeli wystąpi na nim stan niski. Zastosowano rezystor R1 w celu ograniczenia prądu pobieranego przez pin PA2. Pin V_{CC} podłączono do dodatniej szyny zasilania, a pin o nazwie GND doprowadzono do ujemnego bieguna baterii. Są to piny odpowiedzialne za zasilanie mikrokontrolera. Różnica potencjałów pomiędzy tymi pinami powinna zawierać się w przedziale od 4,5 woltów do 5,5 woltów jak wynika to z noty

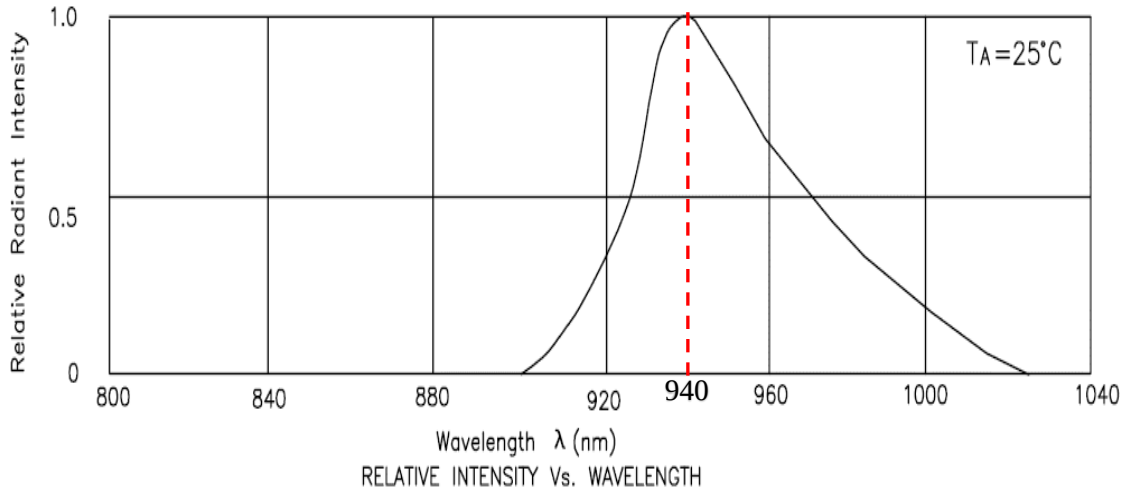
katalogowej. Z powodu braku dostępności baterii o takiej wartości napięcia, a zarazem małych rozmiarach w projekcie użyto baterii o wartości napięcia 6 woltów. Dioda prostownicza 1N4004 chroni układ przed niepoprawnym podłączeniem biegunów baterii do układu. Dodatkowo podczas normalnej pracy na diodzie odkłada się napięcie o wartość 0,7 woltów przez co różnica potencjałów pomiędzy pinami V_{CC} i GND jest zgodna z zaleceniami producenta. Złącze J1 służy do podłączenia baterii. Pomiędzy pinem V_{CC} do którego doprowadzone jest zasilanie, a masą zostały włączone równolegle kondensator elektrolityczny oraz ceramiczny, których zadaniem jest odpowiednia filtracja zasilania. Pomimo zasilania stałym napięciem do zasilania przedostają się zakłócenia spowodowane impulsową pracą nadajnika podczerwieni, a także charakterem pracy samego mikrokontrolera, którego wartość taktowania zegara została ustawiona programowo na wartość 8 megaherców. Poprzez pin PB2 sterowany prądem bazy tranzystor będzie włączał i wyłączał diodę podczerwieni zgodnie ze standardem RC5 w celu wysłania ramki danych skierowanej do układu odbiorczego przypisanego do konkretnego klawisza. Rezystor R2 ogranicza prąd płynący przez diodę podczerwieni.

3.2. Nadajnik podczerwieni



Rys. 3.2 Schemat nadajnika podczerwieni (opracowanie własne)

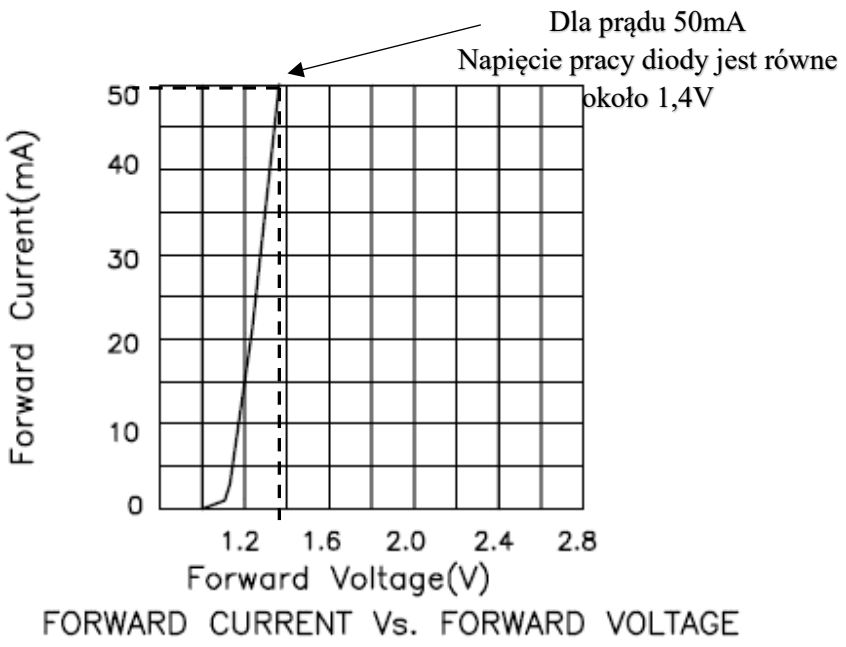
Rysunek 3.2 przedstawia jeden z elementów układu nadawczego jakim jest nadajnik podczerwieni składający się z diody L-53F3BT emitującej falę o długości 940 nanometrów, która zawiera się w zakresie bliskiej podczerwieni. Wartość ta została określona na podstawie umieszczonej na rys. 3.3 charakterystyki przedstawiającej zależność intensywności promieniowania diody od długości emitowanej fali. Kolejnym elementem jest tranzystor PNP BC558B pracujący jako wzmacniacz w układzie z wymuszonym prądem bazy. W celu wymuszenia stałego natężenia prądu bazy I_B łączy się bazę tranzystora poprzez rezystor R_B z masą wtedy też następuje przepływ prądu kolektora. Obciążeniem tranzystora jest wyżej wymieniona dioda wraz z szeregowo połączonym rezystorem R_C. Zadaniem rezystora R_C oraz R_B jest nie tylko odpowiednie zasilenie tranzystora, ale także ustalenie jego stałoprądowego punktu pracy, czyli stałego napięcia kolektor-emiter U_{CE} i stałego prądu kolektora I_C.



Rys. 3.3 Zależność intensywności promieniowania diody L-53F3BT od długości emitowanej fali [12]

3.3. Dobór elementów nadajnika podczerwieni

Na podstawie noty katalogowej diody podczerwieni emitującej światło podczerwone można wywnioskować, że maksymalny prąd stały, jaki możemy przez nią przepuścić jest równy 50 miliamperów. Nadajnik podczerwieni projektowany jest tak, aby na diodzie wydzielała się największa możliwa moc, ponieważ ma to wpływ na zasięg emitowania fali. Na rys. 3.4 znajduje się charakterystyka diody opisująca zależność pomiędzy prądem diody, a jej napięciem pracy.



Rys. 3.4 Zależność pomiędzy natężeniem prądu diody, a napięciem pracy diody L-53F3BT [12]

Nadajnik podczerwieni będzie emitował falę z częstotliwością 36 kiloherców, gdzie przez połowę okresu tranzystor będzie w stanie nasycenia i dioda będzie emitowała falę nośną, a przez drugą połowę okresu tranzystor będzie zatkany i prąd przez diodę nie będzie płynął. Dzięki takiej pracy nadajnika podczerwieni można zwiększyć moc wydzielaną przez diodę. Z noty katalogowej diody wynika, że maksymalna moc ciągła wydzielana na diodzie może być równa 100 miliwatów. Jednak diody podczerwieni przystosowane są do pracy impulsowej. I w tym wypadku wypełnienie impulsów wynosi 50% jak wynika z omawianego wcześniej standardu RC5. Korzystając z poniższego wzoru można w przybliżeniu obliczyć maksymalny prąd w impulsie.

$$I_{PMAX} = \frac{P_{MAX}}{(U_D \cdot k_D)} = \frac{100mW}{(1{,}4V \cdot 50\%)} = 143\ mA \qquad (1)$$

Gdzie:

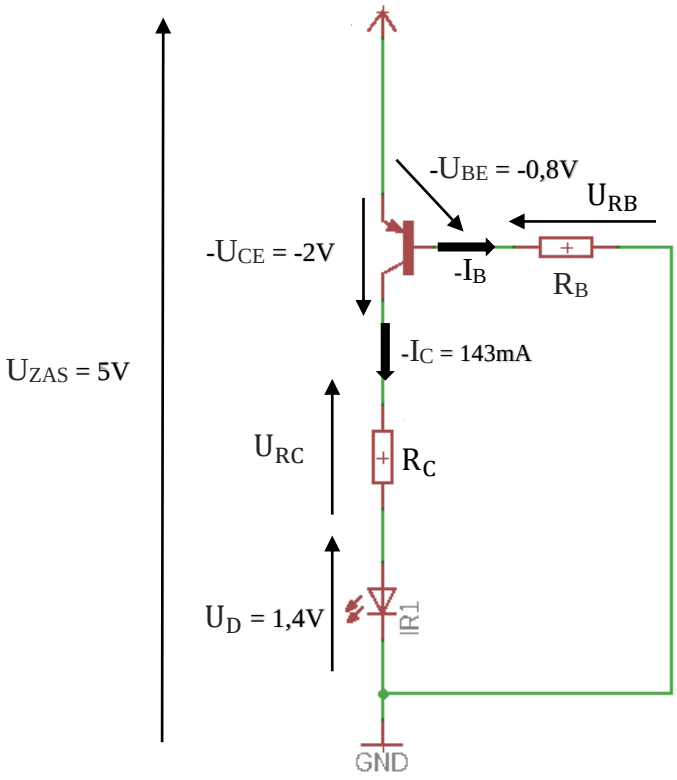
I_{PMAX} - maksymalny prąd w impulsie

P_{MAX} - maksymalna moc ciągła

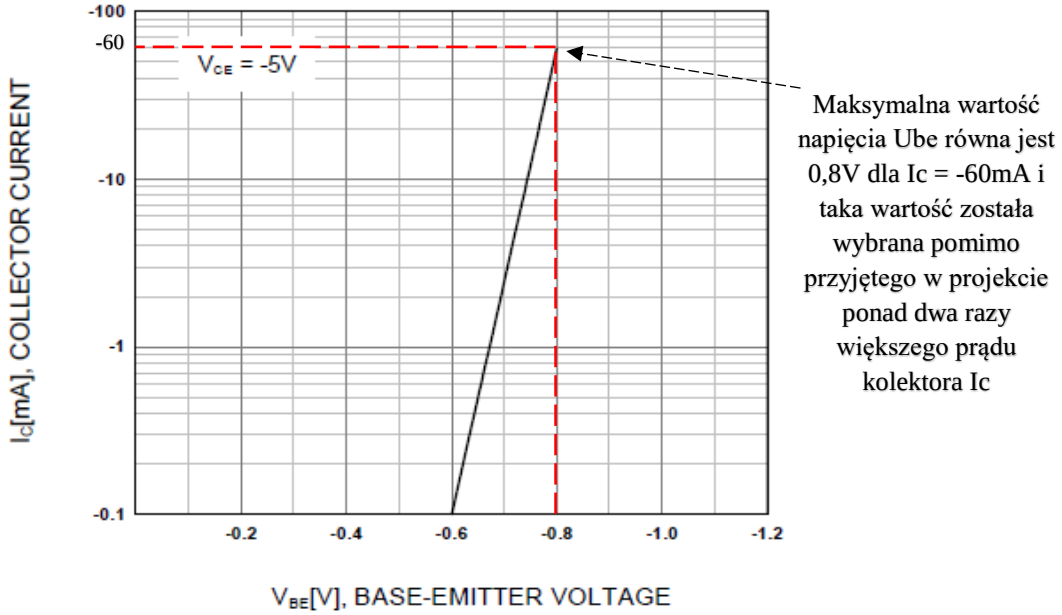
U_D - napięcie pracy diody

k_D - wypełnienie impulsów

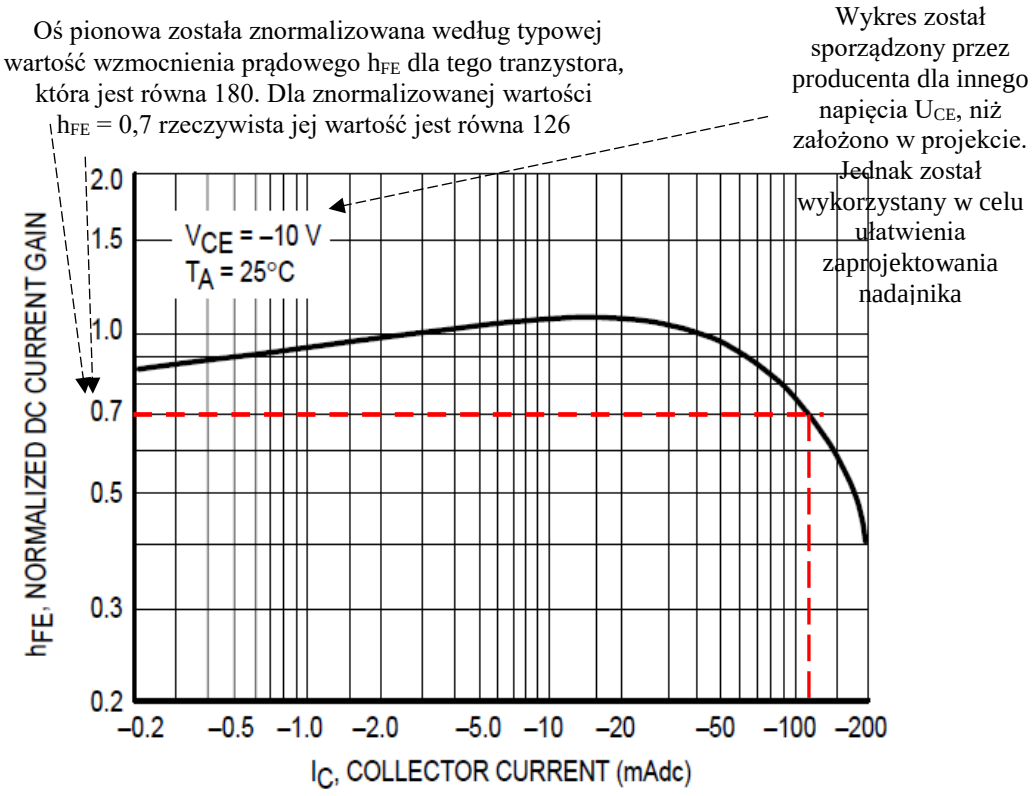
Mając wartość maksymalnego prądu w impulsie I_{PMAX} oraz napięcie pracy diody U_D można przystąpić do obliczenia wartości rezystancji R_C oraz R_B. Rysunek 3.5 obrazuje rozkład prądów i napięć dla nadajnika podczerwieni. Dla napięcia zasilającego U_{ZAS} równego 5 woltów założony punkt pracy tranzystora ma wartości I_C wynoszącą 143 miliamperów z powodu obliczonego maksymalnego prądu w impulsie dla diody podczerwieni oraz napięcia kolektor-emiter tranzystora U_{CE} wynoszącego -2 woltów. Pokazana na rys. 3.6 charakterystyka pozwala na wyznaczenie napięcia między bazą, a emiterem U_{BE} w zależności od prądu kolektora I_C. W projekcie przyjęto wartość napięcia U_{BE} równą -0,8 woltów. W celu wyznaczenia prądu bazy I_B potrzebna jest wartość współczynnika wzmocnienia tranzystora h_{FE}, którą można wyznaczyć korzystając z charakterystyki udostępnionej w nocie katalogowej, a przedstawionej na rys. 3.7. Posiadając wyżej wymienione wartości można przystąpić do obliczeń.



Rys. 3.5 Rozkład napięć i prądów nadajnika podczerwieni (opracowanie własne)



Rys. 3.6 Zależność między napięciem baza-emiter U_{BE}, a prądem kolektora I_C dla tranzystora BC558B [13]



Rys. 3.7 Zależność wzmocnienia prądowego h_{FE} od prądu kolektora I_C dla tranzystora BC558B [13]

Na podstawie II prawa Kirchhoffa oraz prawa Ohma przy pomocy zaznaczonych na rys. 3.5 rozkładem napięć i prądów można wyprowadzić wzory na szukane rezystancje R_C oraz R_B.

Obliczanie rezystancji R_C:

Wzór opisujący drugie równanie Kirchhoffa dla oczka z kolektorem i emiterem:

$$U_{ZAS} = U_D + U_{RC} - (-U_{CE}) = U_D + I_C \cdot R_C - (-U_{CE}) \qquad (2)$$

$$U_{ZAS} - U_D + (-U_{CE}) = I_C \cdot R_C$$

$$R_C = \frac{[U_{ZAS} - U_D + (-U_{CE})]}{I_C}$$

$$R_C = \frac{(5V - 1{,}4V - 2V)}{0{,}143A} = 11{,}188\Omega \approx 11\Omega$$

Obliczanie rezystancji R_B:

Uproszczony wzór na zależność prądu bazy I_B od prądu kolektora I_C :

$$I_B = \frac{I_C}{h_{FE}} \qquad (3)$$

$$I_B = \frac{0{,}143A}{126} = 1mA$$

Wzór opisujący drugie równanie Kirchhoffa dla oczka z bazą i emiterem:

$$U_{ZAS} = U_{RB} + (-U_{BE}) = I_B \cdot R_B + (-U_{BE}) \qquad (4)$$

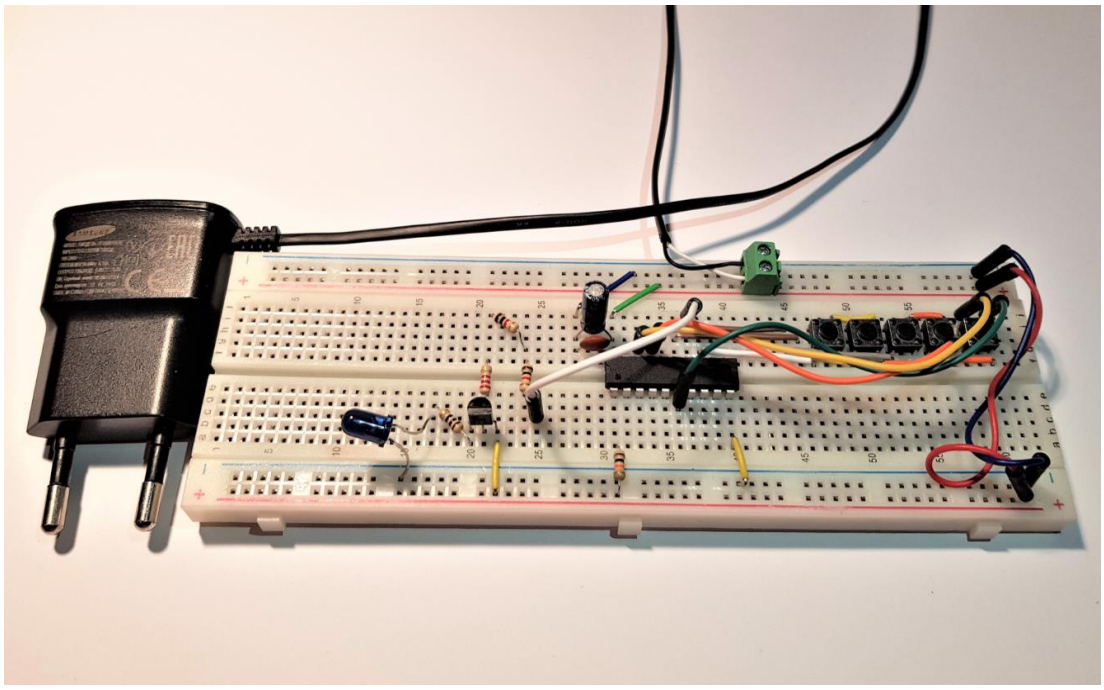
$$U_{ZAS} - (-U_{BE}) = I_B \cdot R_B$$

$$R_B = \frac{[U_{ZAS} - (-U_{BE})]}{I_B}$$

$$R_B = \frac{(5V - 0{,}8V)}{0{,}001\ A} = 4{,}2k\Omega$$

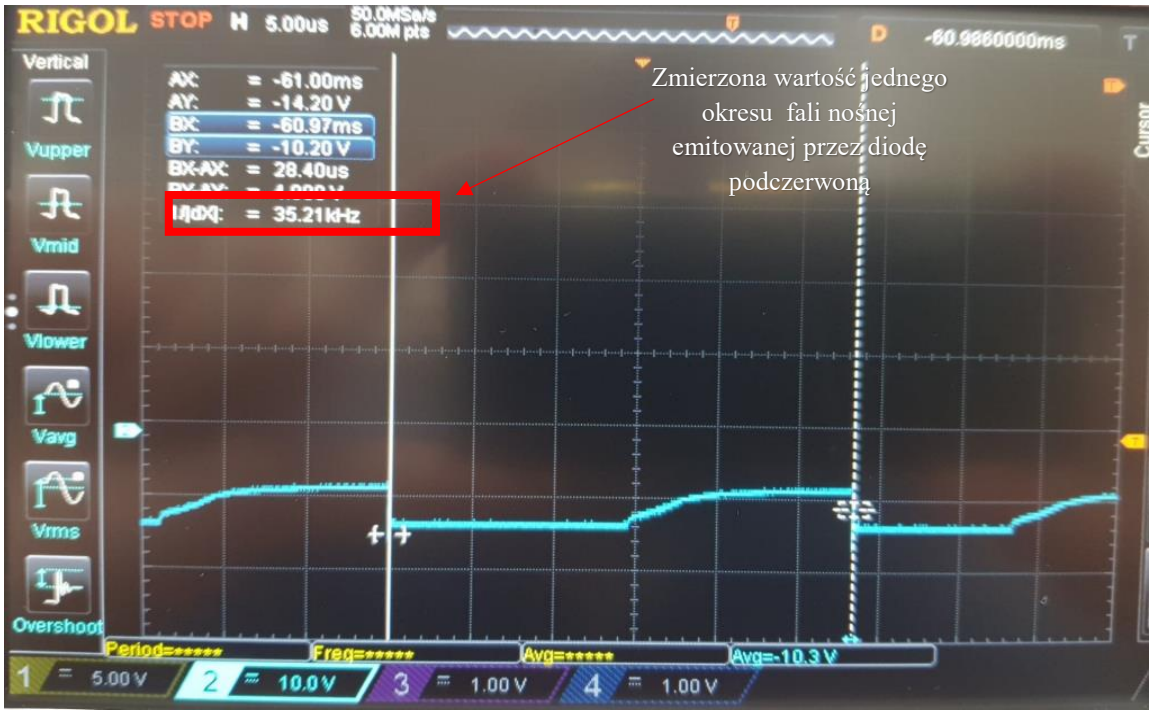
3.4. Prototyp

Załączony na rys. 3.8 prototyp układu nadawczego został zbudowany na płytce stykowej. Zmontowany układ ma na celu zbadanie poprawności wysyłanych bitów, co jest warunkiem poprawnej komunikacji pomiędzy układami. Docelowo układ będzie zasilany baterią 6 woltową jednak prototyp układu zasilany jest z zasilacza firmy Samsung, którego napięcie wyjściowe wynosi 5 woltów, co niweluje potrzebę obniżenia napięcia zasilania do wymaganej wartości.



Rys. 3.8 Prototyp układu nadawczego (opracowanie własne)

Przy pomocy multimetru zmierzono wartość 1,41 woltów napięcia pracy diody podczerwieni, która nieco odbiega od założonych wartości. Również założone natężenie prądu kolektora I_C odbiega od rzeczywistego. Multimetr zmierzył wartość natężenia prądu kolektora równą 86 miliamperów przez co założony punkt pracy tranzystora różni się od rzeczywistego. Podczas pomiaru napięcia pracy diody oraz prądu kolektora nóżka bazy tranzystora została doprowadzona poprzez rezystory o obliczonej wartości 4,2 kiloomów do masy układu. Pomiar tych wartości przy założonej w projekcie pracy impulsowej nadajnika podczerwieni byłby mało wiarygodny dla wykorzystywanego multimetru i dlatego pomiary zostały wykonane bardzo szybko z powodu możliwości uszkodzenia elementów nieprzystosowanych do pracy ciągłej dla tak dużego natężenia prądu. Różnice między wartościami założonymi i pomierzonymi wynikają z użycia kilku uproszczeń, które miały na celu ułatwić zaprojektowanie nadajnika podczerwieni. Otrzymane wartości działającego układu są akceptowalne w ramach projektu. Jednak będzie to miało wpływ na długość zasięgu emitowanej fali.



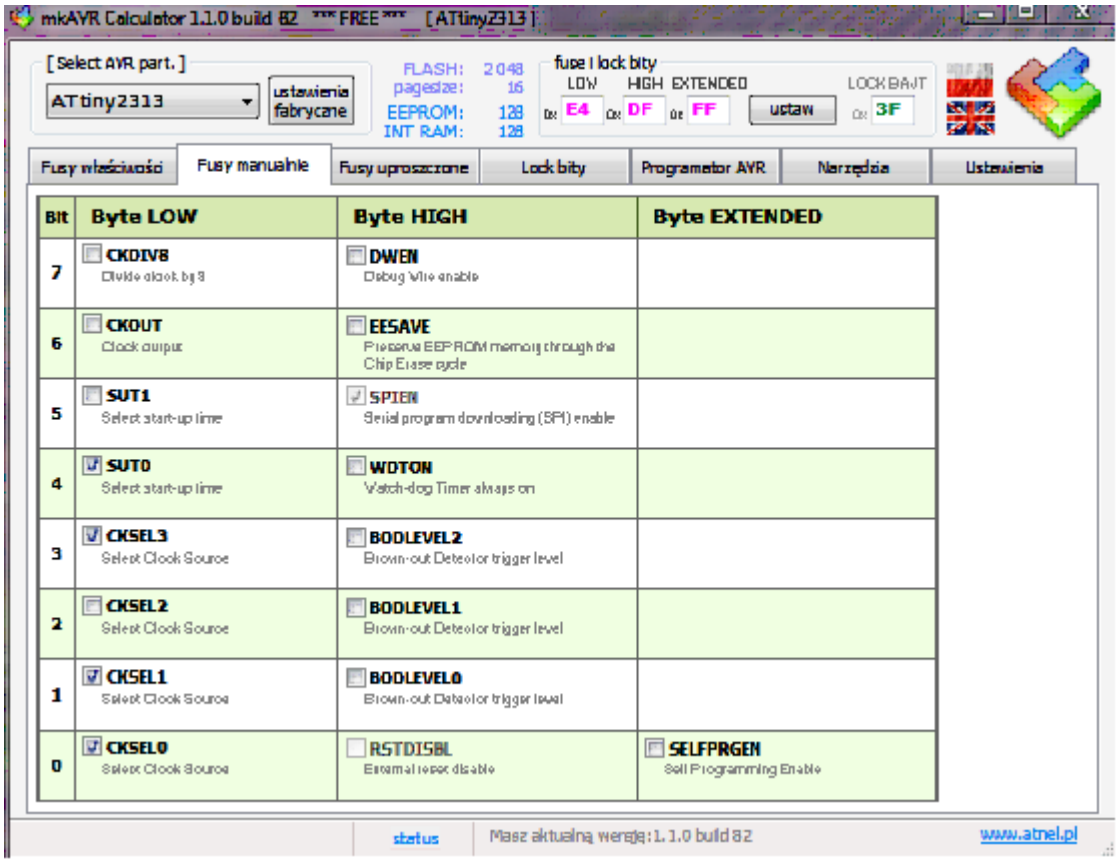
Rys. 3.9 Przebieg napięcia odkładanego na diodzie podczerwieni podczas emitowania fali nośnej (opracowanie własne)

W celu sprawdzenia częstotliwości nośnej sygnału emitowanego przez układ nadawczy podłączono sondy oscyloskopu do anody oraz katody diody podczerwonej. Rysunek 3.9 przedstawia zaznaczony wskaźnikami pionowymi jeden okres tego sygnału. Zmierzona oscyloskopem wartość częstotliwości nośnej wyniosła 35.21 kiloherców. Częstotliwość nośna emitowanego sygnału jest bardzo zbliżona do wartości założonej w projekcie. Różnice w wartościach wynikają z błędów pomiarowych oraz samej pracy mikrokontrolera. Pomimo tej różnicy układy odbiorcze przy pomocy odbiornika podczerwieni będą odbierały nadchodzący sygnał, ponieważ zawiera w swojej budowie filtr środkowoprzepustowy o częstotliwości środkowej równej 36 kiloherców, który będzie w bardzo małym stopniu tłumił częstotliwość nośną nadsyłanego sygnału. Jak można zauważyć zbocza narastające nie są strome z powodu charakteru pracy tranzystora.

3.5. Oprogramowanie

Dla mikrokontrolera układu nadawczego okd źródłowy zawarty w załączniku został napisany w języku C na podstawie informacji zawartych w książce [7] przy użyciu platformy programistycznej Eclipse. Produkcyjnie mikrokontrolery ATmega mają ustawioną częstotliwość taktowania 1 megaherca. Rysunek 3.10 ukazuje zrzut ekranu jednej z zakładek programu mkAVRCalculator w której można zmienić częstotliwość taktowania używanego

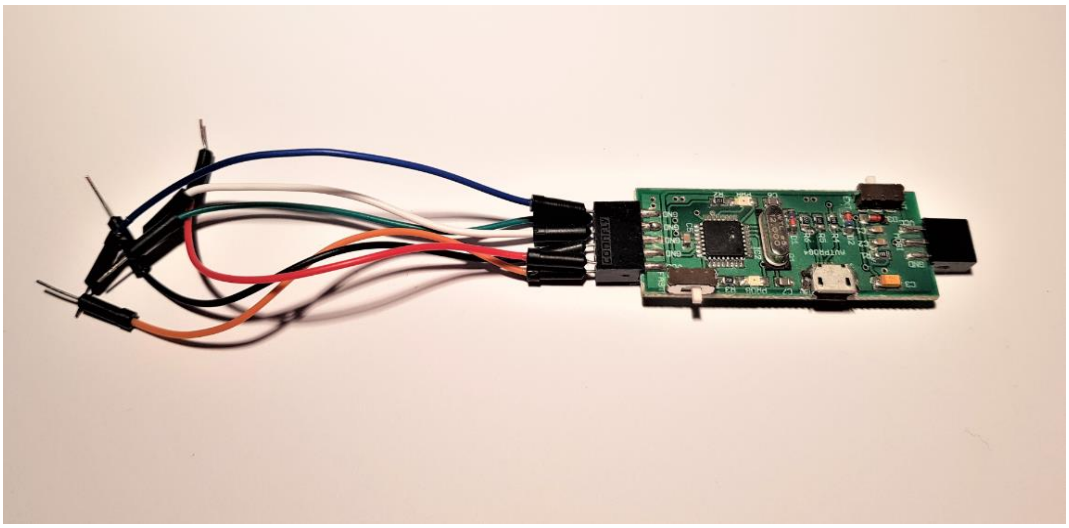
mikrokontrolera. W celu poprawnego działania programu ustawiono częstotliwość taktowania 8 megaherców.



Rys. 3.10 Zrzut ekranu jednej z zakładek programu mkAVRCalculator (opracowanie własne)

Kod źródłowy został wgrany na mikrokontroler ATtiny2313 za pomocą programatora USBasp pokazanego na rys. 3.11, który przeznaczony jest do pracy z mikrokontrolerami z rodziny AVR. Programator pośredniczy w komunikacji pomiędzy mikrokontrolerem, a komputerem. Napisany program wykorzystując rejestr określający kierunek pinów DDRB ustawia pin PB2 jako wyjście, aby móc sterować pracą nadajnika podczerwieni. Za pomocą rejestru TCCR0A oraz TCCR0B użyto zegar sprzętowy, który pozwala generować na PB2 sygnał o częstotliwości 36 kiloherców. Za pomocą rejestrów określających kierunki pinów podłączone do portu B przyciski zostały ustawione jako wyjścia, a pin PD2 jako wejście. W pętli głównej programu znajduje się jedynie funkcja wprowadzająca mikrokontroler w stan niskiego poboru energii, a wyłączany jest po wykryciu przerwania wywołanego stanem wysokim na PD2. Część kodu źródłowego odpowiedzialna za wysłanie ramki o odpowiednich wartościach zawarta jest w tym przerwaniu aktywujące się przy wciśnięciu jednego z przycisków. Mikrokontroler sprawdza jaki przycisk został użyty wykorzystując do tego rejestr PINB i zaczyna nadawać przypisaną do niego ramkę danych o odpowiednich wartościach.

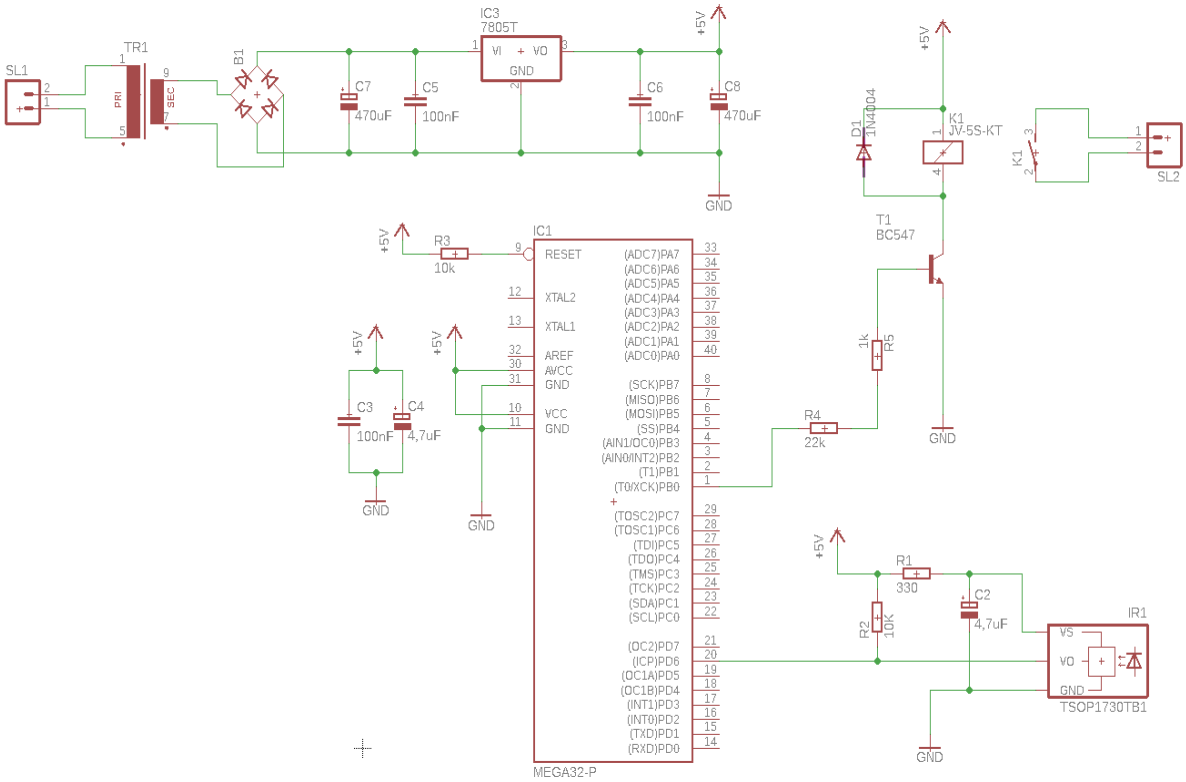
Emitowanie i przerwa w emitowania częstotliwości nośnej przez okres 889 mikrosekund odbywa się przy użyciu funkcji wykorzystującej pracę drugiego zegara sprzętowego, który jest ustawiony za pomocą jego rejestrów TCCR1B, TIFR oraz OCR1A w taki sposób, aby móc określać czas trwania emitowania częstotliwości nośnej lub jej braku. Zmienna funkcji określana jest w mikrosekundach. Następnie wykorzystano dwie funkcje wyślij_1 oraz wyślij_ wykorzystując powyżej opisaną funkcję wysyłają bity o odpowiadającej im wartości logicznej. Działanie tych funkcji polega na odpowiednim włączeniu zegara pierwszego wykorzystując jego rejestr TCCROA, który odpowiedzialny jest za włączenie generowania sygnału o danej częstotliwości na pinie PB2.



Rys. 3.11 Fotografia przedstawiająca programator USBasp (opracowanie własne)

4. Układy odbiorcze

4.1. Budowa układu odbiorczego nr 1

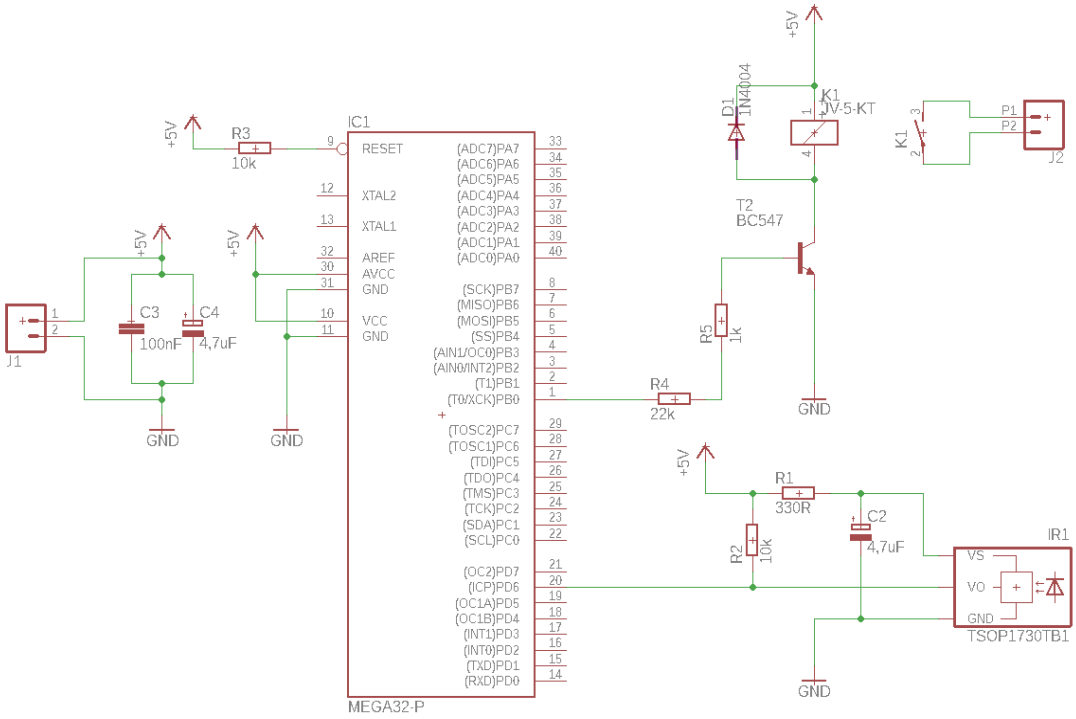


Rys. 4.1 Schemat układu odbiorczego nr 1 (opracowanie własne)

Na rys. 4.1 pokazany został schemat układu odbiorczego nr 1 różniącego się od układ odbiorczego nr 2 wyłącznie fragmentem układu odpowiedzialnym za doprowadzenie zasilania. Z powodu funkcji jaką będzie spełniał jest możliwość doprowadzenia z sieci zasilania prądu przemiennego o napięciu skutecznym 230 woltów. W celu obniżenia napięcia zasilania do 12 woltów prądu przemiennego zastosowano transformator obniżający napięcie TSZZ 2/007. Podczas jego wyboru kierowano się napięciem wtórnym, którego wartość powinna być większa niż wartości napięcia na wyjściu stabilizatora napięcia. Poprawna praca stabilizatora wymaga, aby jego napięcie wyjściowe było o 2,5 wolta niższe niż napięcie wejściowe. Mając na uwadze spadki napięć na diodach prostowniczych mostka Graetza przyjęto minimalną różnicę napięć pomiędzy wejściem stabilizatora, a transformatorem równą 4 wolty. Zwrócono również uwagę na jego wymiary oraz maksymalne natężenie prądu jakie można uzyskać na wyjściu transformatora, które powinno być większe od natężenia prądu pobieranego przez układ odbiorczy. W celu przekształcenia prądu przemiennego na jednokierunkowy prąd tętniący użyto pełno okresowego prostownika jakim jest mostek Graetza. W każdym półokresie odkłada się napięcie o wartości 0,7 woltów na każdej z dwóch diod prostowniczych. W Układzie

zasilania znajduje się stabilizator napięcia LM7805, który utrzymuje stabilne napięcie na jego wyjściu o stałej wartości nominalnej wynoszącej 5 woltów w celu zapewnienia wymaganego napięcia dla mikrokontrolera, odbiornika podczerwieni oraz przekaźnika. Na wejściu oraz wyjściu stabilizatora dołączono równolegle kondensatory elektrolityczne i ceramiczne w celu wygładzenia napięcia.

4.2. Budowa układu odbiorczego nr 2

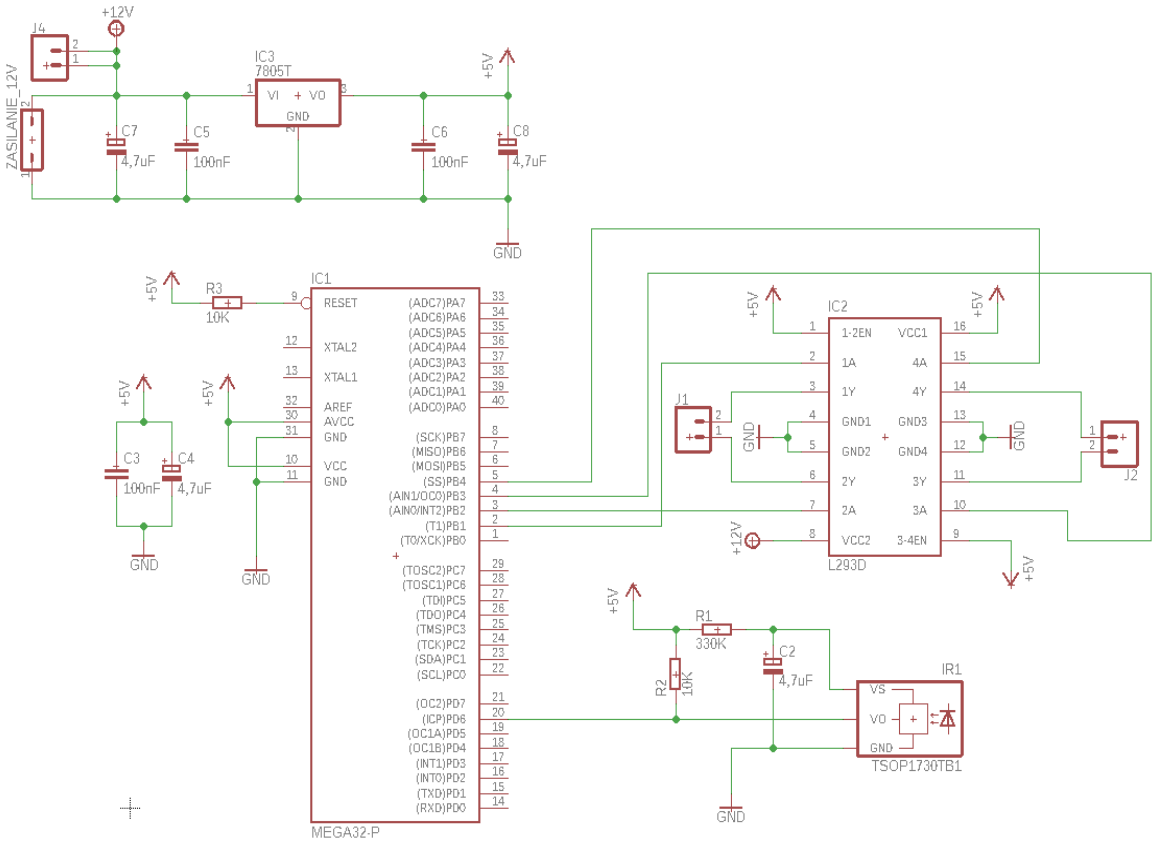


Rys. 4.2 Schemat układu odbiorczego nr 2 (opracowanie własne)

Na rys. 4.2 przedstawiony został schemat układu odbiorczego nr 2. Mikrokontroler ATmega32 odpowiada za zarządzanie całym układem. Odbiera sygnał nadchodzący z odbiornika podczerwieni dekodując ramkę danych określoną standardem RC5 i wykonuje określoną funkcję. Wyjście sygnałowe odbiornika podczerwieni zostało doprowadzone do pinu PD6 mikrokontrolera, który odpowiedzialny jest za generowanie przerwań wywołanych zboczami opadającymi i narastającymi występującymi na tym pinie. Rezystor R1 wraz z kondensatorem elektrolitycznym C2 odpowiedzialny jest za filtrację zasilania poprzez eliminację zakłóceń pochodzących od odbiornika podczerwieni ,a rezystor R2 odpowiedzialny jest za lepsze podciągnięcie wyjściowej linii danych odbiornika podczerwieni do szyny zasilającej jak wynika z noty katalogowej producenta. Również kondensatory C3 oraz C4 służą do filtracji zasilania. Układ jest zasilany z baterii podłączonej do złącza J1. Mikrokontroler sterując prądem bazy popularnego tranzystora BC547B steruje pracę przekaźnika. Wybierając przekaźnik JV-5S-KT sugerowano się jego napięciem pracy, które zostało dobrane tak, aby

było równe napięciu zasilania mikrokontrolera oraz nadajnika podczerwieni. Dioda prostownicza 1N4004 została umieszczona pomiędzy stykami cewki przekaźnika katodą skierowaną do napięcia zasilania w celu rozładowania się przez nią napięcia zaindukowanego w cewce przekaźnika występującego podczas zanikania pola magnetycznego tej cewki.

4.3. Budowa układu odbiorczego nr 3



Rys. 4.3 Schemat układu odbiorczego nr 3 (opracowanie własne)

Układ odbiorczy nr 3, którego schemat przedstawiono na rys. 4.3 będzie pełnił funkcję zdalnego sterownika rolet. Do tego celu został wykorzystany silnik krokowy unipolarny oraz sterownik L293D. Sterownik w swojej budowie zawiera dwa mostki H i przeznaczony jest do sterowania dwoma silnikami prądu stałego jednak przy odpowiednim połączeniu wyjść sterownika z silnikiem krokowym unipolarnym jest możliwa realizacja jego sterowania podając na sterownik w odpowiedniej kolejności impulsy sterujące pochodzące z mikrokontrolera. Układ tak jak pozostałe układy odbiorcze także zawiera mikrokontroler ATmega32 oraz odbiornik podczerwieni. Silnik zasilany będzie z akumulatora żelowego o wartości napięcia nominalnego równego 12 woltów. Stabilizator LM7805 utrzymuje stabilne napięcie na jego wyjściu o stałej wartości nominalnej wynoszącej 5 woltów, ponieważ takim napięciem należy

zasilić mikrokontroler oraz odbiornik podczerwieni. Do złącz J1, J2 oraz J4 został odpowiednio podłączony silnik krokowy unipolarny.

4.4. Obliczenia

Na rysunku 4.4 przedstawiona jest część układu odbiorczego nr 1 oraz układu odbiorczego nr 2 odpowiedzialna za wykonywanie określonej funkcji poprzez włączanie oraz wyłączanie przekaźnika. Z noty katalogowej wykorzystanego w projekcie przekaźnika JV-5S-KT można znaleźć wartość rezystancji cewki przekaźnika R_P równą 125 omów , nominalnego napięcia U_P wynoszącego 5 woltów, nominalnej mocy jego cewki P_P równej 200 miliwatów oraz minimalnej moc P_{MIN} wynoszącej 130 miliwatów dla której przekaźnik pracuje normalnie. Mając te wartości obliczono nominalną wartość natężenia prądu cewki za pomocą wzoru na moc wydzielaną na elemencie podczas pracy ciągłej (5).

$$P_P = U_P \cdot I_P \qquad (5)$$

$$I_P = \frac{P_P}{U_P}$$

$$I_P = \frac{0{,}2\ W}{5V} = 40\ mA$$

Gdzie:

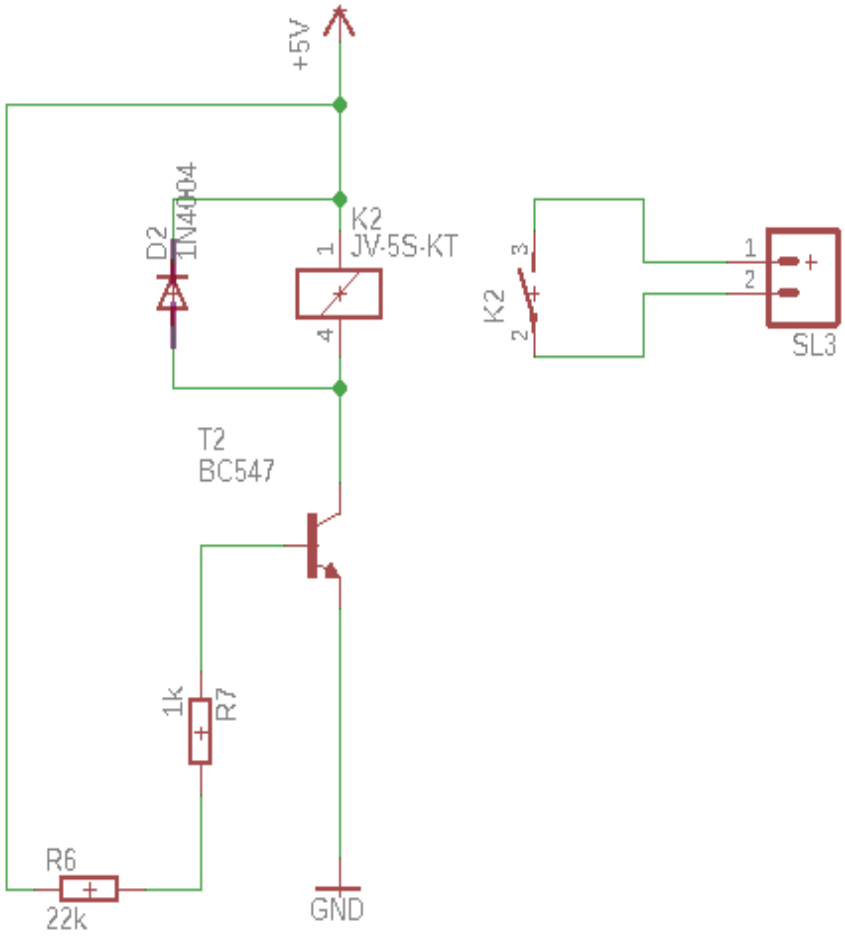
P_P - nominalna moc cewki przekaźnika

U_P - nominalne napięcie cewki przekaźnika

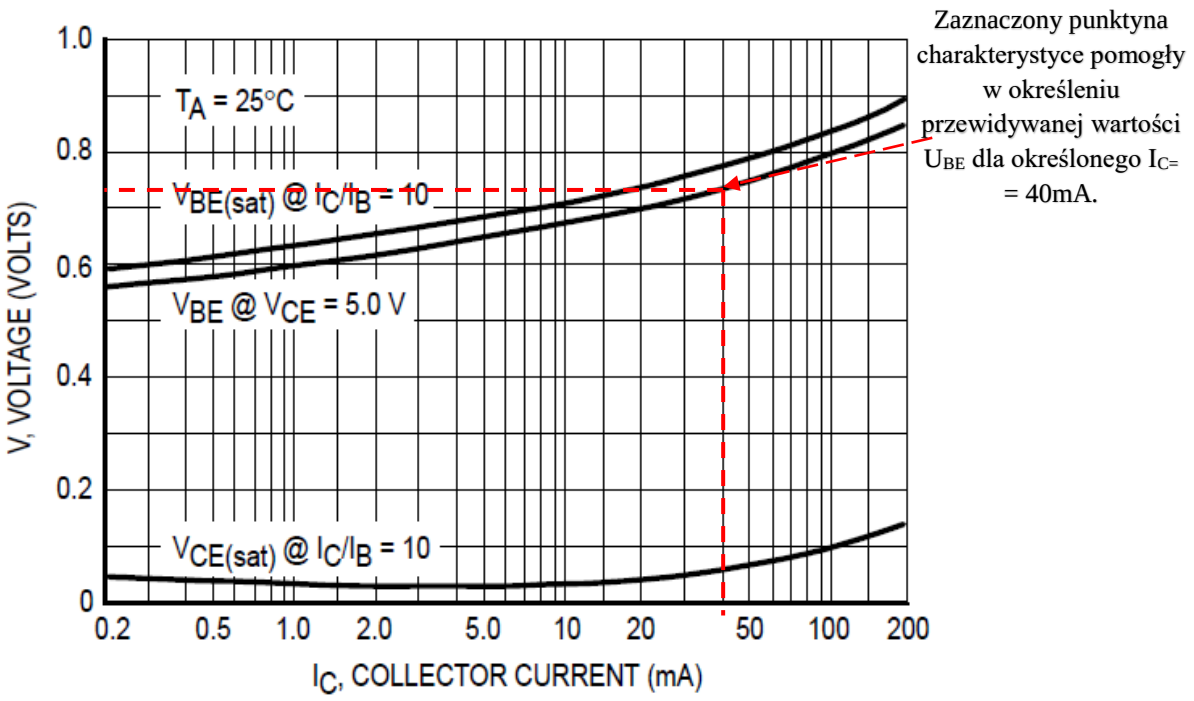
I_P - nominalne natężenie cewki przekaźnika

Wartość natężenia prądu kolektora I_C tranzystora BC547B powinna być równa obliczonej wartości nominalnego natężenia prądu cewki przekaźnika. Napięcie zasilania U_{ZAS} jest równe 5 wolt. Korzystając z noty katalogowej wykorzystywanego tranzystora wynika, że napięcie kolektor-emiter U_{CE} wynosi 0,8 woltów dla prądu kolektora I_C równego 40 miliamperów. Użyto jednej z charakterystyk pokazanej na rys. 4.5 w celu oszacowania napięcia baza-emiter U_{BE}. Przy założonym natężeniu prądu kolektora i napięciu U_{CE} wynika, że U_{BE} wynosi 0,72 woltów. Dla wybranego natężenia prądu kolektora współczynnik wzmocnienia prądowego h_{FE} wynosi 180. Posiadając wyżej wymienione wartości można przystąpić do obliczenia prądu bazy I_B w celu wyznaczenia brakującej wartości rezystancji R_B ograniczającej natężenie prądu bazy. Natężenie prądu kolektora I_C będzie nieco niższe niż założono, ponieważ przy ustalonym napięciu zasilania 5 woltów występujący spadek napięcia U_{CE} ograniczy go do wartości nieco

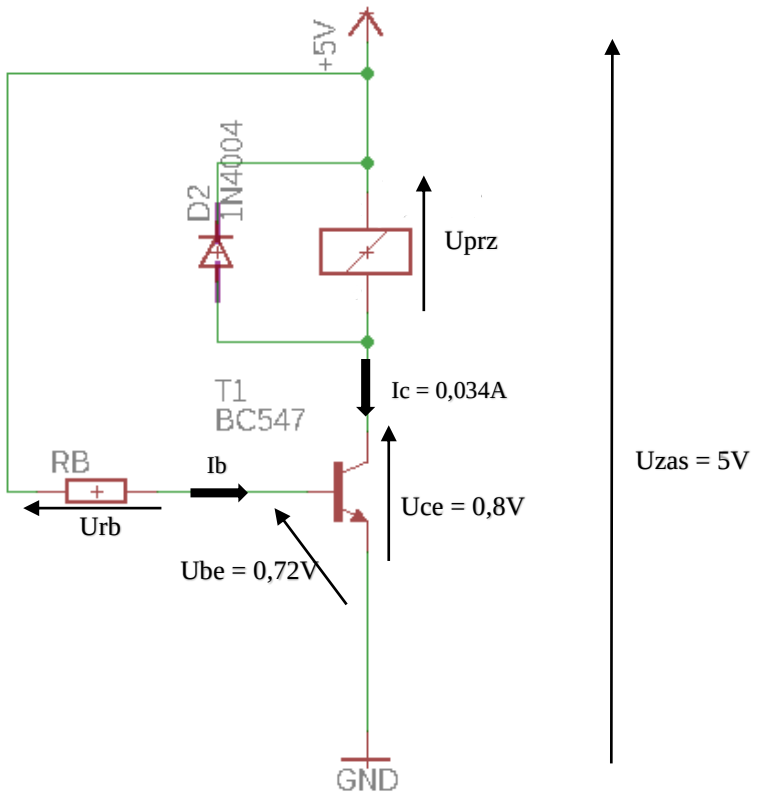
ponad 34 miliamperów. Taka wartość jest wystarczająca, aby sterować wybranym w projekcie przekaźnikiem. Z tego samego powodu napięcie odkładane na cewce przekaźnika także będzie nieco niższe od wartości nominalnej napięcia cewki, co nie będzie wpływało na prawidłową pracę przekaźnika.



Rys. 4.4 Część układu odbiorczego nr 1 oraz nr 2 sterująca przekaźnikiem (opracowanie własne)



Rys. 4.5 Charakterystyka przedstawiająca zależność napięcia kolektor-emiter i napięcia baza-emiter od natężenia prądu kolektora dla tranzystora BC547B [14]



Rys. 4.6 Rozkład napięć i prądów fragmentu układu odbiorczego nr 1 oraz nr 2 (opracowanie własne)

Na podstawie II prawa Kirchhoffa oraz prawa Ohma, a także zaznaczonym na rys. 4.6 rozkładem napięć i prądów w stanie nasycenia tranzystora można wyprowadzić wzory, aby wyznaczyć przewidywane napięcie na cewce przekaźnika różniące się od nominalnego napięcia oraz szukaną rezystancję R_B. Do obliczeń założono wartość natężenia prądu kolektora I_C równą 34 miliamperów oraz U_{CE} wynoszącą 0,8 woltów. Najpierw obliczono przewidywane napięcie na cewce przekaźnika U_{PRZ} w celu sprawdzenia wydzielanej mocy, która powinna mieć większą wartość, niż minimalna moc pracy cewki przekaźnika P_{MIN}.

Wzór opisujący drugie równanie Kirchhoffa dla oczka z kolektorem i emiterem:

$$U_{ZAS} = U_{PRZ} + U_{CE} \qquad (6)$$

$$U_{PRZ} = U_{ZAS} - U_{CE}$$

$$U_{PRZ} = 5V - 0{,}8V = 4{,}2V$$

Mając tą wartość obliczono moc wydzielaną na cewce przekaźnika w celu sprawdzenia poprawnej pracy przekaźnika.

$$P_{PRZ} = U_{PRZ} \cdot I_C \qquad (7)$$

$$P_{PRZ} = 4{,}2V \cdot 0{,}034A = 0{,}143W$$

Obliczona wartość mocy wydzielanej na cewce przekaźnika P_{PRZ} jest wystarczająca, aby przekaźnik działał poprawnie. Następnie obliczono wartość natężenia prądu bazy I_B przy pomocy uproszczonego wzoru na zależność natężenia prądu bazy od prądu kolektora. Mając wartość natężenia tego prądu można obliczyć wartość szukanej rezystancji R_B.

$$I_B = \frac{I_C}{h_{FE}} \qquad (8)$$

$$I_B = \frac{0{,}034A}{180} = 188\ \mu A$$

Wzór opisujący drugie równanie Kirchhoffa dla oczka z bazą i emiterem:

$$U_{ZAS} = U_{RB} + U_{BE} = I_B \cdot R_B - U_{BE} \qquad (9)$$

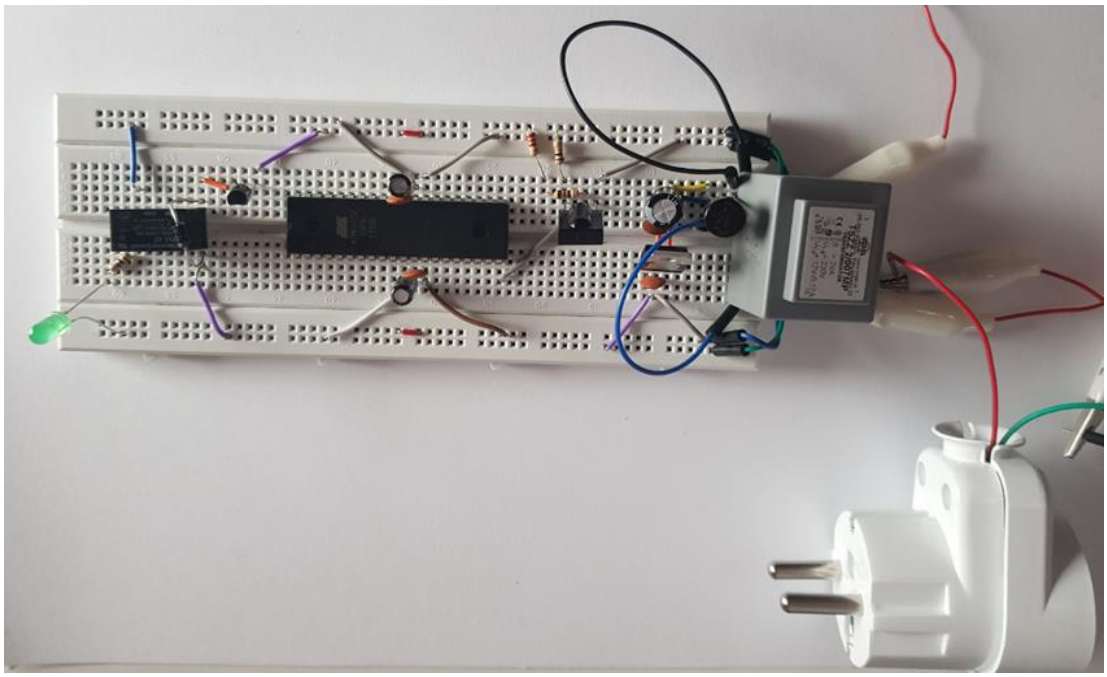
$$U_{ZAS} - U_{BE} = I_B \cdot R_B$$

$$R_B = \frac{(U_{ZAS} - U_{BE})}{I_B}$$

$$R_B = \frac{(5\ V - 0{,}72\ V)}{0{,}000188\ A} = \approx 23k\Omega$$

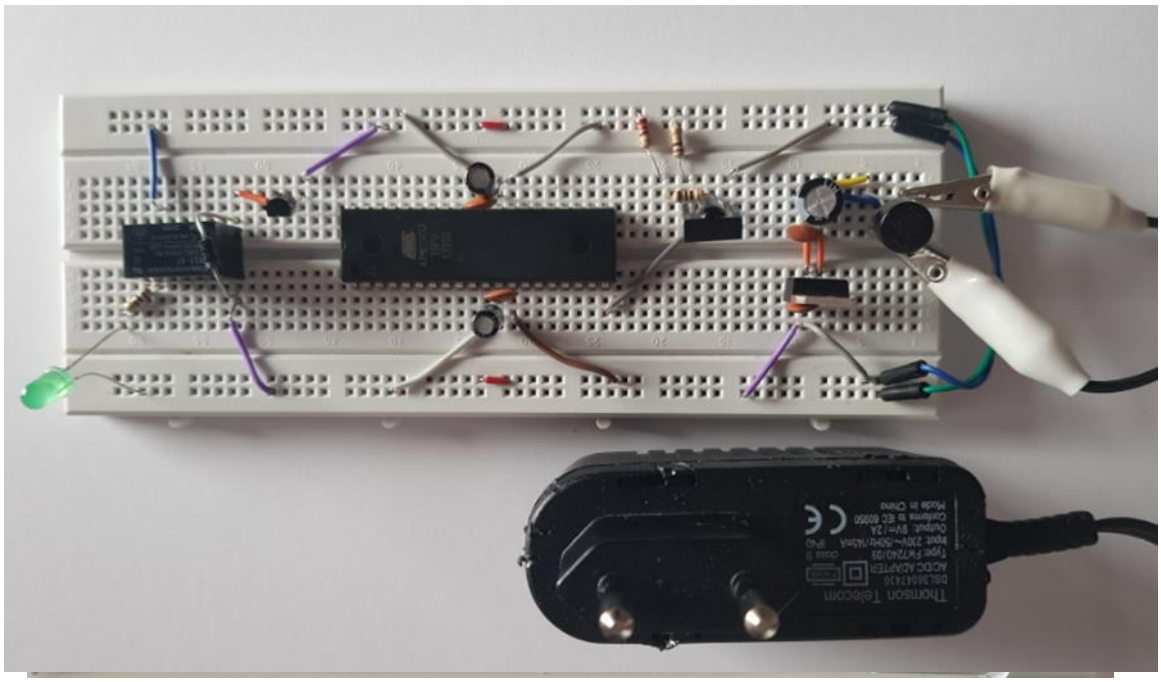
4.5. Prototypy

Układy prototypowe zostały zmontowane na płytkach stykowych. Ich celem jest sprawdzenie poprawności działania oraz zmierzenia poszczególnych wartości układów. Na rys. 4.7 przedstawiony został prototyp układu odbiorczego nr 1. Dla którego pomierzono wartość napięcia na wyjściu stabilizatora napięcia LM7805 podczas gdy układ zasilany jest z sieci. Poprawna praca układu zależy od wartości doprowadzonego napięcia zasilania. Mikrokontroler ATmega32 zasilany jest napięciem o stałej wartości 5 woltów i taką wartość zmierzył multimetr na wyjściu stabilizatora napięcia. Dla układu odbiorczego nr 1 oraz nr 2 na wyjściu przekaźnika dołączono diodę elektroluminescencyjną z szeregowo połączonym rezystorem w celu sprawdzenia poprawności działania układów.



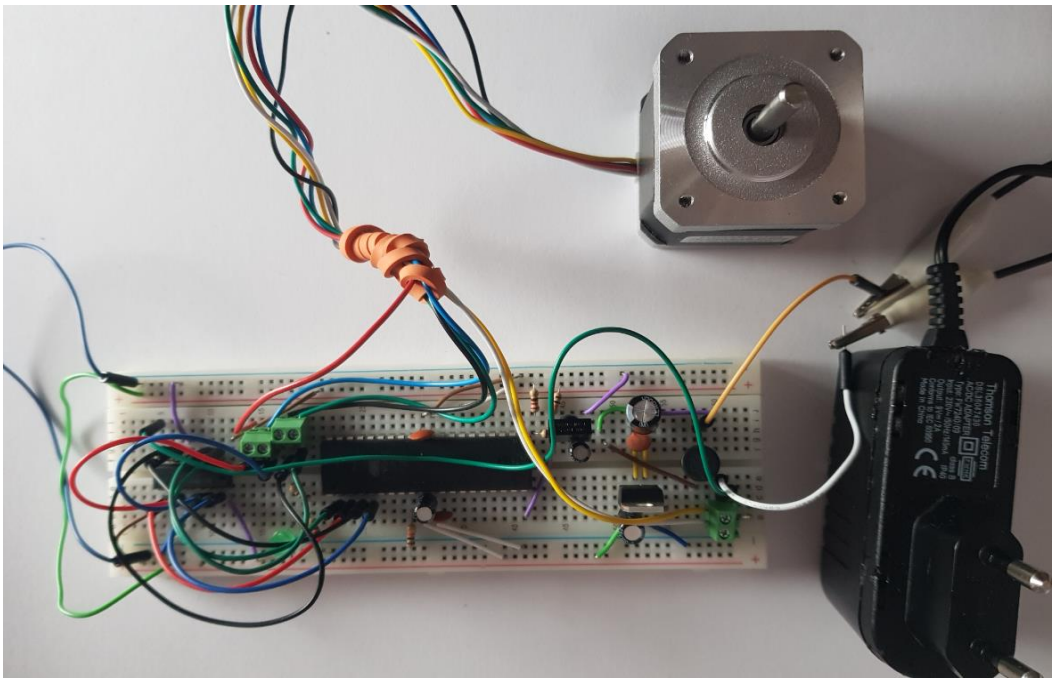
Rys. 4.7 Prototypowy układ odbiorczy nr 1 (opracowanie własne)

Za pomocą multimetru zmierzono wartości napięcia baza-emiter U_{BE}, napięcia kolektor-emiter U_{CE} oraz natężenia prądu kolektora Ic dla układu odbiorczego nr 1 oraz nr 2. Napięcie pomiędzy bazą, a emiterem jest zgodne z obliczoną wartością. Natężenie prądu kolektora wynosi 31 miliamperów, a napięcie U_{CE} jest równe 0,6 woltów. Pomierzone wartości minimalnie odbiegają od założonych jednak są akceptowalne w celu poprawnej pracy układów.



Rys. 4.8 Prototyp układu odbiorczego nr 2 (opracowanie własne)

Zaprezentowany na rys. 4.8 prototyp układu odbiorczego nr 2 zasilony został przy pomocy zasilacza firmy Thomson telecom na którego wyjściu występuje napięcie o stałej wartości 9 woltów. Dzięki umieszczonemu mostkowi Graetza nie trzeba się martwić o poprawne podłączenie zasilania. Korzystając z tego prototypu wykonano próby komunikacji układu nadawczego z odbiorczym w pokoju o przestrzeni siedmiu metrów na pięć metrów znajdujących się w punktach najdalej położonych od siebie. Pomiędzy układami nie wystąpiły żadne błędy w komunikacji. Jedynym warunkiem było skierowanie diody podczerwieni w stronę układu odbiorczego.



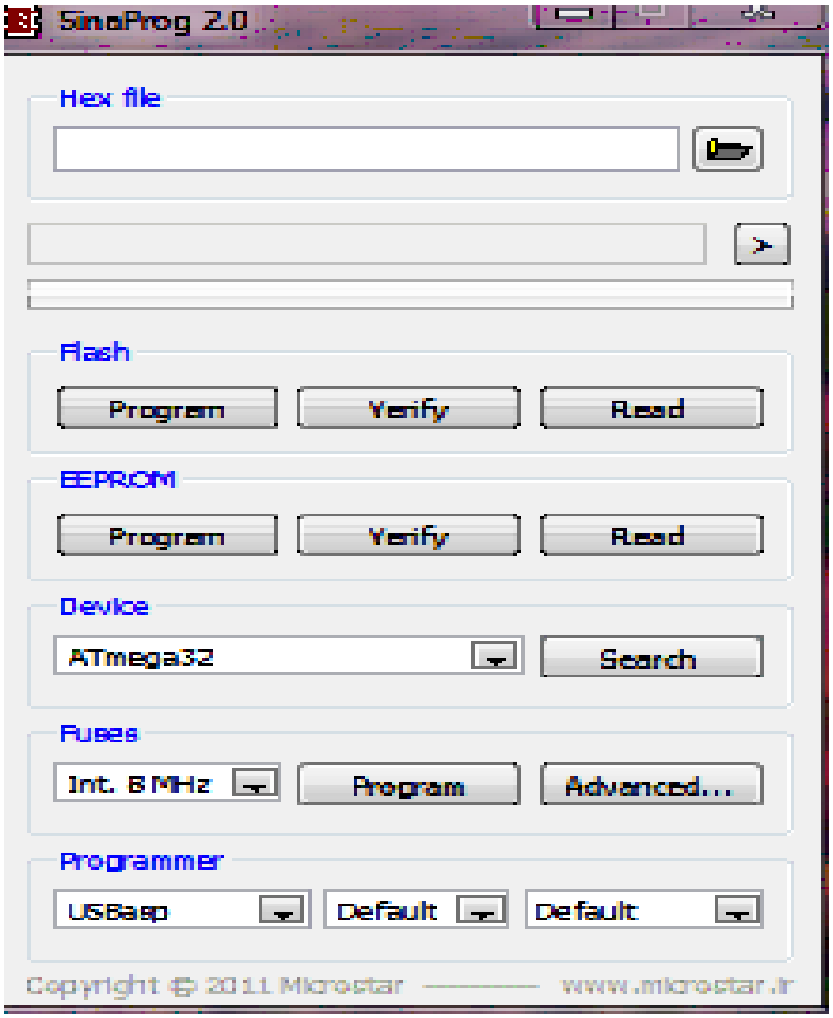
Rys. 4.9 Prototyp układu odbiorczego nr 3 (opracowanie własne)

Poprawność działania prototypu układu odbiorczego nr 3 pokazanego na rys. 4.9 sprawdzono korzystając z wcześniej zmontowanego prototypu układu nadawczego. Wysyłając odpowiednią ramkę danych zaobserwowano obracanie się silnika. Z obserwacji wynika, że układ działa poprawnie.

4.6. Oprogramowanie

Kod źródłowy układów odbiorczych zawiera bibliotekę złożoną z pliku źródłowego oraz pliku nagłówkowego niezbędnego w celu udostępnienia biblioteki innym plikom źródłowym. Biblioteka zawiera kod, który odpowiedzialny jest za dekodowanie sygnałów pochodzących z odbiornika podczerwieni wykorzystując standard RC5. Kod źródłowy biblioteki inicjalizuje pracę mikrokontrolera poprzez ustawienie portu PD6 jako wejście przy użyciu rejestru DDRD w celu pobierania wartości napięcia z odbiornika podczerwieni i przypisania im odpowiedniego stanu logicznego. Jeden z zegarów sprzętowych za pomocą rejestrów TCCR1B oraz TIMSK ustawiony zostaje tak, aby mierzyć czasy trwania poszczególnych impulsów występujących na porcie PD6 w celu stwierdzenia czy nadsyłana ramka danych jest prawidłowa i czy czas trwania poszczególnych impulsów jest zgodny ze standardem RC5. Omawiany zegar sprzętowy zaczyna zliczać czas trwania impulsów od momentu kiedy na PD6 pojawi się zbocze opadające lub narastające występujące w środku każdego z nadsyłanych bitów. Mikrokontroler rozpoznaje początek ramki poprzez wykrycie dwóch zboczy opadających bitów startu występujących na jej początku. Po zakończeniu dekodowania ramki danych poszczególne

wartości bitów zostają przypisane do zmiennych nazwanych adres, komenda oraz toggle, które są wykorzystane w kodach źródłowych układów odbiorczych. W celu zmiany częstotliwości taktowania mikrokontrolera ATmega32 z ustawionej produkcyjnie wartości 1 megaherca na wartość 8 megaherców użyto programu Sinaprog przedstawionego na rysunku 4.10, ponieważ darmowa wersja wcześniej użytego program mkAVRCalculator do ustawienia częstotliwości taktowania mikrokontrolera ATtiny2313 nie obejmuje mikrokontrolera ATmega32.



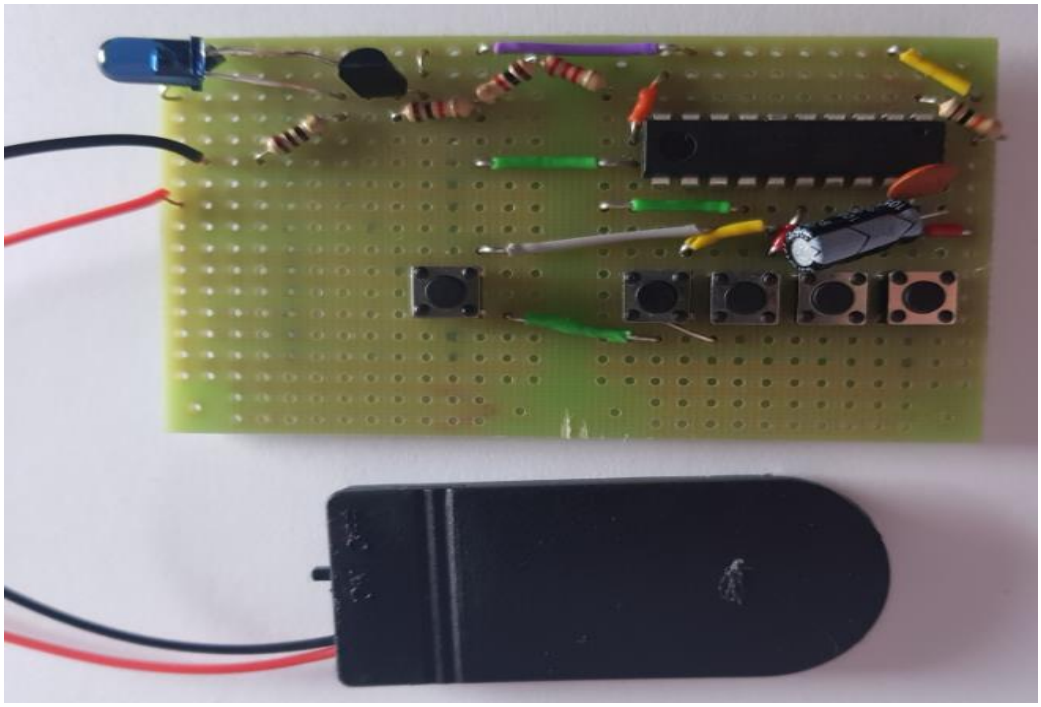
Rys. 4.10 Zrzut ekranu programu Sinaprog (opracowanie własne)

Kod źródłowy dla układu odbiorczego nr 1 powoduje pierw włączenie przekaźnika na krótką chwilę podczas podłączania zasilania w celu sprawdzenia poprawności jego działania. Następnie zostaje wykonana funkcja konfigurująca poszczególne rejestry mikrokontrolera w celu poprawnej pracy z odbiornikiem podczerwieni w celu umożliwienia dekodowania nadchodzących ramek od układu nadawczego. Funkcja została zdefiniowana w pliku biblioteki opisanej powyżej. W pętli głównej programu sprawdzana jest pierw możliwość skorzystania ze zmiennych do których przypisano wartości określające adres, komendę oraz wartość bitu toggle

po poprawnym zdekodowaniu ramki danych, a następnie porównywane są wartości adresu i komendy odebranej ramki danych z tymi, które założono w kodzie. Każdy z układów odbiorczych zawiera inny adres oraz komendę na którą reaguje realizując odpowiednią funkcję. Jeśli wartości te są równe, wtedy następuje zmiana stanu logicznego pinu PB0 na przeciwny, co powoduje włączenie lub wyłączenie przekaźnika. Na końcu programu odebrane wartości zdekodowanej ramki zostają wyzerowane, aby móc odebrać następną. Różnica w kodzie pomiędzy układem odbiorczym nr 1, a układem odbiorczym numer 2 polega na tym, że układ te reagują na odebraną ramkę danych o innych wartościach adresu oraz komendy. Program realizujący sterowanie silnikiem krokowym unipolarnym przez układ odbiorczy nr 3 pierw deklaruje, które piny zostaną wykorzystane do sterowania silnikiem. Następnie tworzy makrodefinicje sterujące tymi pinami. Korzystając z rejestru DDRB oraz rejestru PORTB ustawiono piny sterujące silnikiem jako wyjścia ze stanem niskim. Po odebraniu ramki skierowanej do tego układu mikrokontroler porównuje odebrane wartości adresu oraz komendy zawarte niej jeśli są równe z założonymi w kodzie tego programu to silnik zaczyna obracać się w odpowiednią stronę podając za pomocą pętli odpowiedniej kolejności impulsów sterujących silnikiem. Na końcu programu zerowane są wartości odebranej ramki danych, aby była możliwość odbioru następnej.

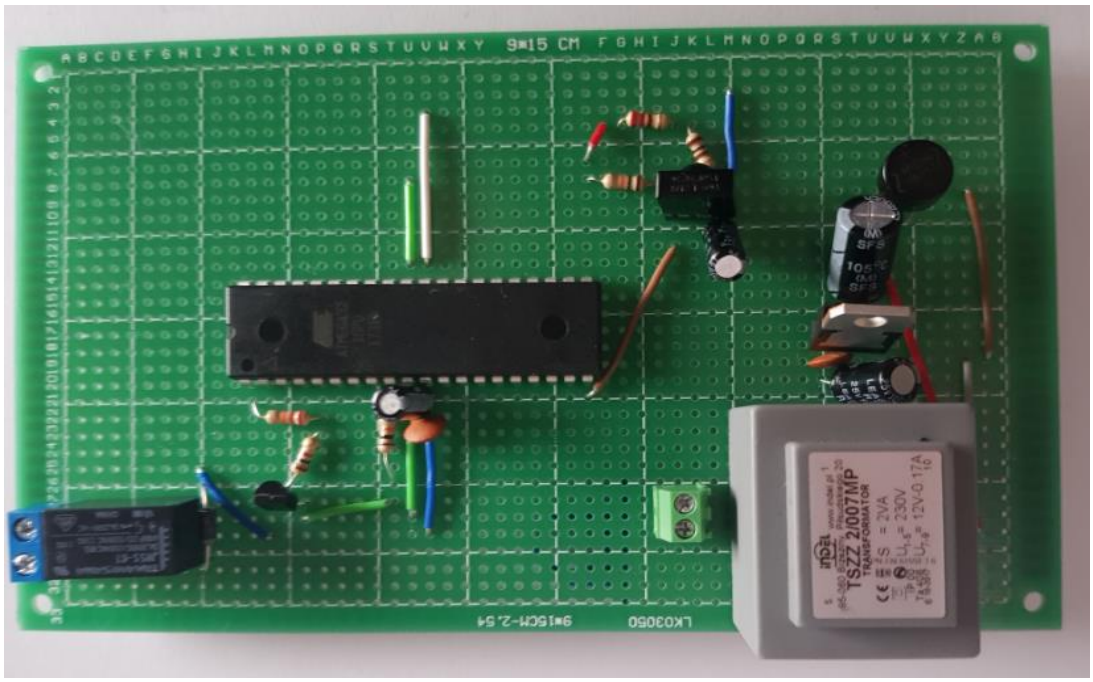
5. Wykonanie układów

Układy zostały wykonane na płytkach uniwersalnych. Wykorzystano elementy przystosowane do montażu przewlekanego, które zostały przylutowane do dolnej warstwy płytek. Na rysunku 5.1 przedstawiono zmontowany układ nadawczy zbudowany w formie pilota. Funkcję przycisków pełnią przełączniki typu microswitch.



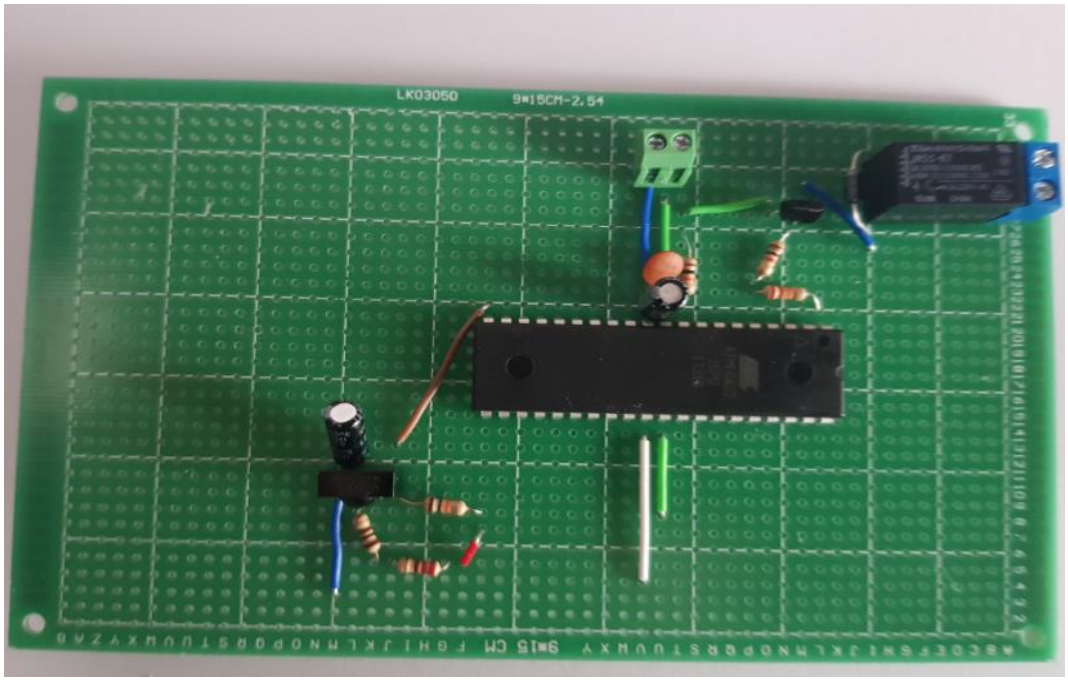
Rys. 5.1 Układ nadawczy (opracowanie własne)

Przedstawiony na rysunku 5.2 układ odbiorczy nr 1, który realizuje funkcję sterowania gniazdkiem sieciowym. Złącze znajdujące się blisko przekaźnika zostało podłączone do adaptera gniazdka sieciowego w sposób umożliwiający sterowanie przepływem prądu przez urządzenia podłączone do adaptera. Blisko transformatora również znajduje się złącze. Jego celem jest możliwość doprowadzenia energii elektrycznej do uzwojenia pierwotnego transformatora w celu zasilenia układu.

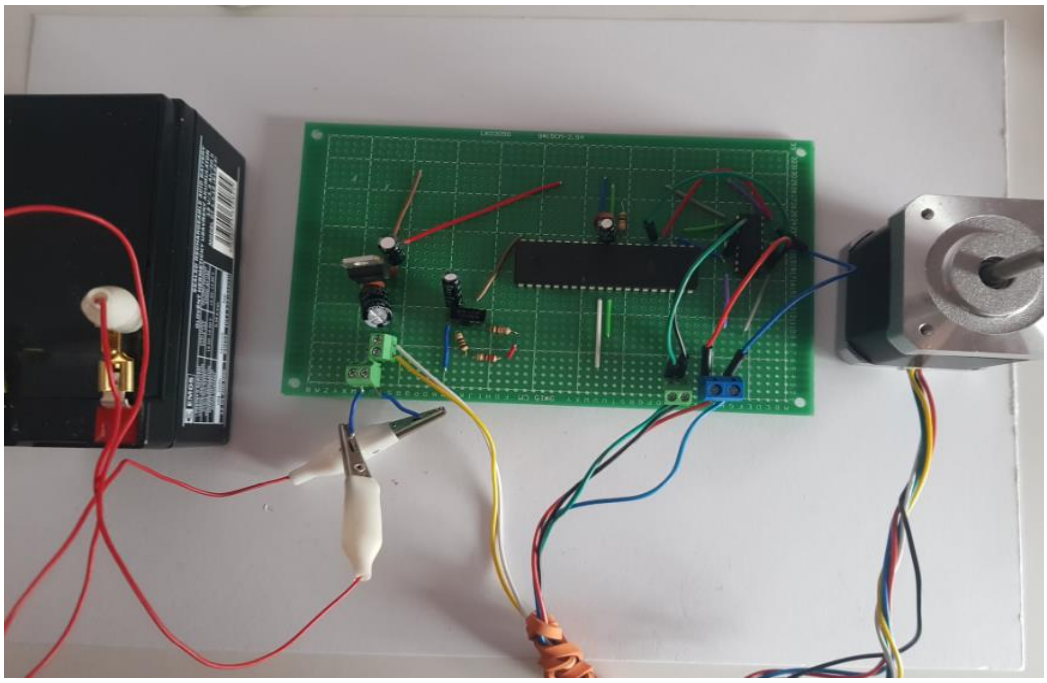


Rys. 5.2 Układ odbiorczy nr 1 (opracowanie własne)

Układ sterujący oświetleniem w pomieszczeniu został zaprezentowany na rys. 5.3. W celu umożliwienia podłączenia styków przekaźnika do instalacji elektrycznej wykorzystano złącze, do którego zostaną doprowadzone przewody. Układ ten zasilany jest z baterii.



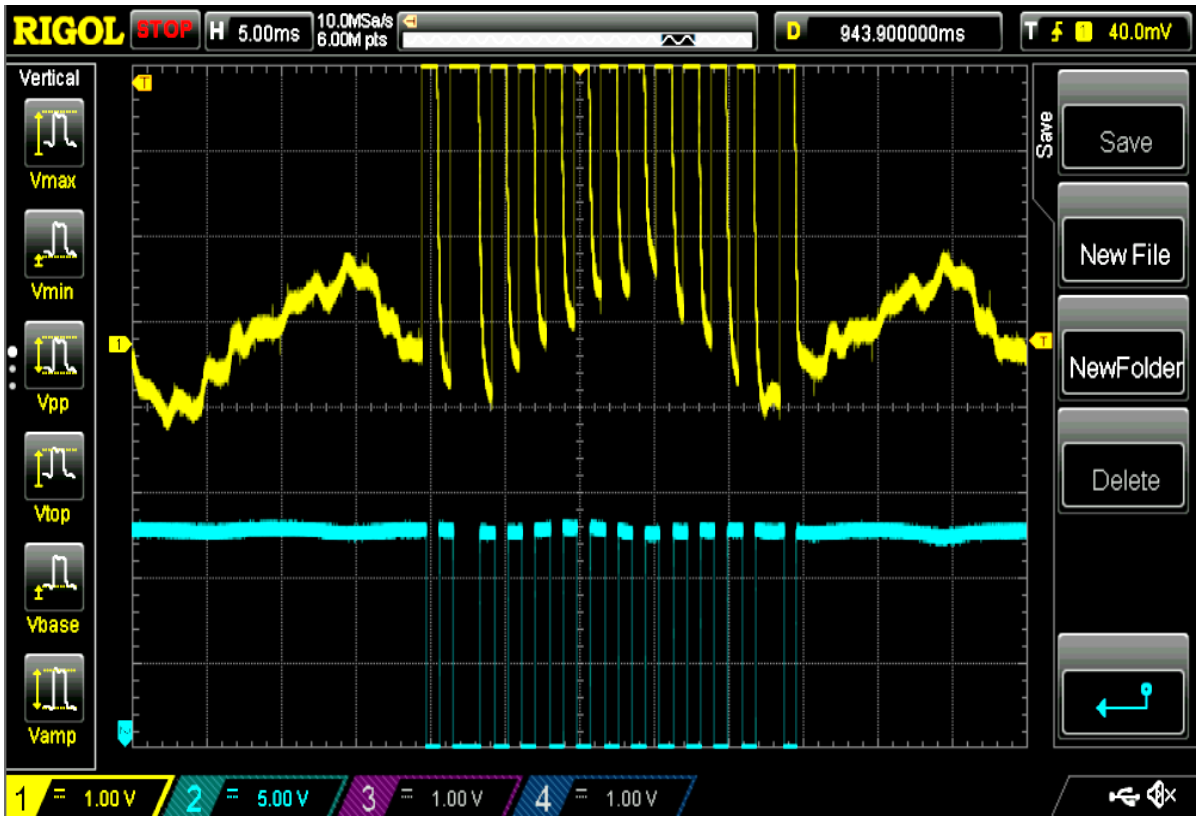
Rys. 5.3 Układ odbiorczy nr 2 (opracowanie własne)



Rys. 5.4 Układ odbiorczy nr 3 (opracowanie własne)

Zaprezentowany na rys. 5.4 układ odbiorczy nr 3 zasilany jest akumulatorem ołowiowym. Podczas jego wyboru kierowano się jego nominalnym napięciem wynoszącym 12 woltów, wagą oraz pojemnością, która jest równa 1,3 amperogodziny. Zastosowane złącza mają na celu umożliwić proste podłączanie akumulatora oraz silnika. Gdyby sterowana roleta okazała się zbyt ciężka, a moment obrotowy silnika zbyt mały to wykorzystane złącza umożliwiają podłączenie innego silnika.

Przy użyciu oscyloskopu dokonano sprawdzenia poprawności transmisji ramki danych z układu nadawczego do układów odbiorczych. Do pierwszego kanału oscyloskopu podłączono katodę oraz anodę diody podczerwieni, a sondę kanału drugiego podłączono do masy układu i pinu PD6, który służy do dekodowania ramki danych. Rysunek 5.5 przedstawia przebiegi dla układu nadawczego oraz układu odbiorczego nr 1. Na przedstawionych przebiegach można zauważyć występujące w układzie nadawczym zakłócenia o częstotliwości 50 herców. Powodowane są przez zasilacz impulsowy wykorzystany wcześniej dla układu prototypowego nadajnika, a który użyto podczas badań w celu sprawdzenia poprawności transmisji przy braku stałego napięcia zasilania dla jednego z układów. Układ odbiorczy nr 1 zasilono poprzez zaprojektowany układ zasilania ciągłego, który jak widać na przebiegu spełnia swoje zadanie utrzymania stabilnego napięcia stałego o wartości 5 woltów.

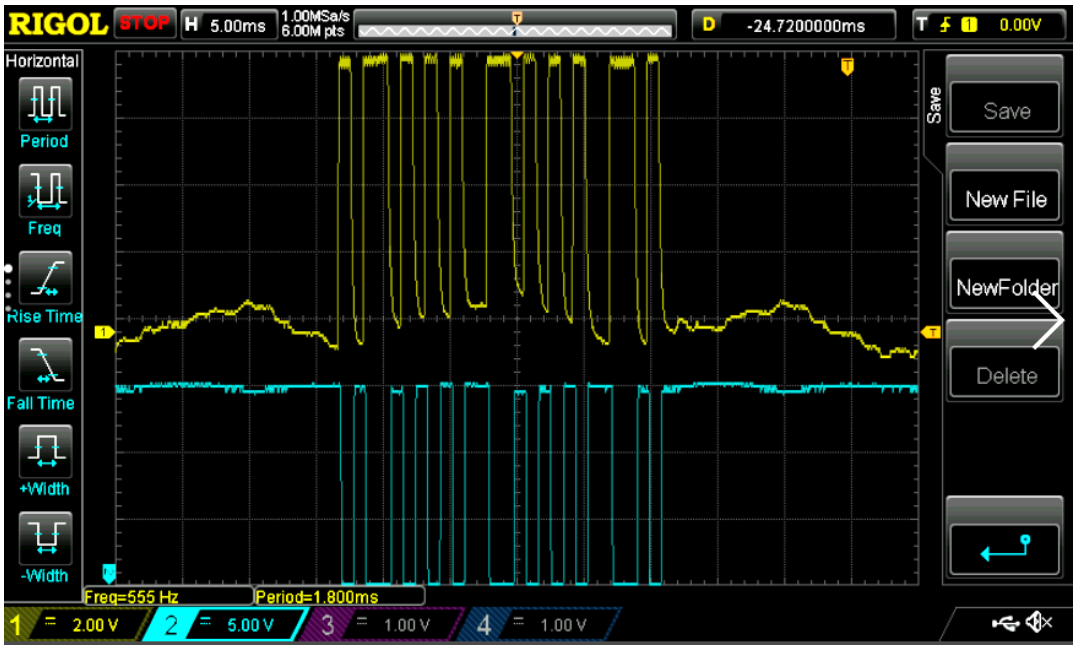


Rys. 5.5 Zrzut ekranu oscyloskopu obrazujący przebiegi ramek danych dla nadajnika oraz układu odbiorczego nr 1 (opracowanie własne)

Zaobserwowano poprawny odbiór ramki danych przez układ odbiorczy nr 1 pomimo zakłóceń występujących w układzie nadawczym. Ramka danych wysłana z nadajnika składa się z 14 bitów, a także odebrana w układzie odbiorczym składa się z tej samej ilości bitów. Odebrana wukładzie odbiorczym ramka danych została zanegowana przez odbiornik podczerwieni. Na rys. 5.6 można zauważyć podobne przebiegi, jednak różnią się poszczególnymi bitami odpowiedzialnymi za nadanie odpowiedniego adresu urządzenia oraz komendy. Wybrane przyciski wysyłają odpowiednio zakodowaną ramkę danych umożliwiającą wykonanie realizowanej funkcji przez układ odbiorczy przypisany do danego przycisku.

Rys. 5.6 Zrzut ekranu oscyloskopu obrazujący przebiegi ramek danych dla nadajnika oraz układu odbiorczego nr 2 (opracowanie własne)

Układ odbiorczy nr 3 został zasilony akumulatorem 12 woltowym. Dzięki użytemu stabilizatorowi LM7805 utrzymywane jest stabilne napięcie zasilania układu o wartości 5 woltów, jak wynika z rys. 5.7. Podczas badania układów nie zauważono błędnego odbioru ramek danych pomimo zakłóceń występujących w układzie nadawczym spowodowane użytym zasilaczem impulsowym.



Rys. 5.7 Zrzut ekranu oscyloskopu obrazujący przebiegi ramek danych dla nadajnika oraz układu odbiorczego nr 3 (opracowanie własne)

6. Podsumowanie

Realizacja projektu pokazała w jaki sposób można skonstruować prosty system zarządzający poszczególnymi funkcjami instalacji domowej. Małym kosztem finansowym udało się zbudować jeden układ nadawczy w postaci pilota oraz trzy układy odbiorcze realizujące sterowanie oświetleniem, roletą i gniazdem elektrycznym. Do komunikacji pomiędzy układem nadawczym, a układami odbiorczymi wykorzystano standard RC5, który pozwala na zaadresowanie ramki danych dla konkretnego układu odbiorczego w celu ich odróżniania przez układ nadawczy oraz wysłania danych informujących jaką funkcję układy mają wykonać.

Pomimo niższego natężenia prądu płynącego przez diodę podczerwieni w układzie nadawczym w trakcie przesyłania ramki danych niż założono w projekcie komunikacja pomiędzy układami odbywa się poprawnie. Podczas projektowania nadajnika podczerwieni oraz fragmentu układów odbiorczych odpowiedzialnego za realizację sterowania przekaźnikiem skorzystano z uproszczeń mających na celu wyznaczenia poszczególnych wartości elementów. Wykonując pomiary zauważono, że wartości rzeczywiste poszczególnych napięć tranzystora, współczynnika wzmocnienia oraz natężenia płynących prądów w tranzystorze nieznacznie odbiega od wartości przedstawionych w notach katalogowych. Wynika to z rozrzutu produkcyjnego parametrów tranzystora oraz ich zmiana spowodowana zmianą temperatury jego pracy. Pierwotnie układy miałby być przygotowane na płytkach miedzianych. Jednak po kilku nieudanych próbach przeniesienia układu połączeń na płytkę miedzianą techniką termotransferu postanowiono wykorzystać płytki uniwersalne.

W przyszłość planowana jest rozbudowa systemu o dodatkowe układy odbiorcze realizujące funkcje sterowania grzejnikiem, płynnej regulacji oświetleniem oraz monitorowania pomieszczenia za pomocą kamery. Korzystając z Rasbberry Pi zostanie stworzona komunikacja pomiędzy systemem a telefonem komórkowym.

BIBLIOGRAFIA

1. http://www.eae-elektronik.pl/automatyka-domowa/ (otwarto dnia 22.12.18)

2. yadda.icm.edu.pl/yadda/element/.../c/kurz_dariusz_porownanie_92_2017.pdf (pobrano dnia 02.01.19)

3.https://instalaton.pl/aktualnosci/28_systemy-inteligentnego-domu-charakterystyka-p.html (otwarto dnia 23.12.18)

4. http://www.xcomfort.pl/system (otwarto dnia 28.12.18)

5. http://www.eib.pl/media-transmisji-sygnalow-knx (17.01.19)

6. http://www.eib.pl/opis-ogolny-knx (otwarto dnia 4.01.19)

7. Kardaś M.: *Mikrokontrolery AVR. Język C - podstawy programowania.* Wydawnictwo: Atnel, Szczecin 2013, str. 329 - 347, 403 - 411.

8. https://forbot.pl/blog/kurs-elektroniki-ii-scalony-odbiornik-podczerwieni-tsop-id9641 (otwarto dnia 25.11.18)

9. https://forbot.pl/blog/kurs-budowy-robotow-zdalne-sterowanie-ir-rc5-id19364 (otwarto dnia 27.11.18)

10. https://forbot.pl/blog/kurs-raspberry-pi-od-podstaw-wstep-spis-tresci-id23139 (otwarto dnia 28.11.18)

11. https://forbot.pl/blog/kurs-arduino-srodowisko-jak-zaczac-programowac-id936 (otwarto dnia 28.11.18)

12. http://www.alldatasheet.com/datasheet-pdf/pdf/92890/ETC/L53F3BT.html (otwarto dnia 01.12.18) rys3.3

13. http://www.alldatasheet.com/datasheet-pdf/pdf/2898/MOTOROLA/BC558B.html (otwarto dnia 01.12.18)

14. http://www.alldatasheet.com/datasheet-pdf/pdf/2894/MOTOROLA/BC547B.html (otwarto dnia 02.12.18)

15. http://mirekk36.blogspot.com/2012/12/filtrowanie-zasilania-dlaczego-tak-wazne.html (otwarto dnia 5.12.18)

16. https://ea.elportal.pl/bipolarne.html (otwarto dnia 01.12.18)

17. http://www.mbmaster.pl/elektronika-tranzystor-bipolarny-npn-1.html (otwarto dnia 03.12.18)

18. http://www.alldatasheet.com/datasheet-pdf/pdf/77378/ATMEL/ATMEGA32.html (otwarto dnia 26.11.18)

19. http://www.alldatasheet.com/datasheet-pdf/pdf/80317/ATMEL/ATTINY2313.html (otwarto dnia 26.11.18)

20. http://www.alldatasheet.com/datasheet-pdf/pdf/22432/STMICROELECTRONICS/L293D.html (otwarto dnia 05.12.18)

21. http://www.alldatasheet.com/datasheet-pdf/pdf/26102/TEMIC/TFMS5360.html (otwarto dnia 03.12.18)

22. https://pl.mouser.com/datasheet/2/164/jv-18250.pdf (otwarto dnia 02.12.18)

SPIS RYSUNKÓW

7. Załącznik

```
/*
 * nadawanie.c
 *
 *  Created on: 15 wrz 2018
 *      Author: Pi
 */
#include <avr/io.h>                    //Dołączenie niezbędnych przez producenta
#include <avr/interrupt.h>             //bibliotek umożliwi napisanie kodu wykorzystując
#include <avr/sleep.h>                // mikrokontrolera w celu wykorzystania
#include <inttypes.h>
#include <util/delay.h>
#define F_CPU 8000000UL  // Zawarcie informacji dla kompilatora o ustawionej częstotliwości taktowania mikrokontrolera
/*Makra adresów nadawanych przez pilot dla 3 układów odbiorczych*/
#define ADRES_1 0                   //adresy są interpretowane przez układy odbiorcze
#define ADRES_2 1                   //jako ciąg sześciu bitów. Kompilator podczas kompilacji
#define ADRES_3 2                   //zamieni zdefiniowane makro na wartość bitową
/* Makra ułatwiające przydzielenie kodów komendy dla poszczególnych przycisków */
#define KOMENDA_1 1
#define KOMENDA_2 2
#define KOMENDA_3 3
#define KOMENDA_4 4
#define KOMENDA_5 5
/* Makrodefinicja portu obsługująca diodę podczerwieni IR */
#define IR_PORT B
#define IR_PIN  B
#define IR_DDR  B
#define NR_PINU  2                 // Baza tranzystora została podłączona poprzez rezystory do pinu 2 portu B
#define IR (1<<NR_PINU)            // Makrodefinicja ułatwiająca dostęp do pinu podłączonego
                                   // pod bazę tranzystora, który steruje diodą IR bedąca obciążeniem tego tranzystora
#define IR_ON  PORTB &= ~(IR)      // Stan niski na PORTB.1 załącza tranzystor
#define IR_OFF PORTB |= IR         // Stan wysoki zatyka tranzystor
#define CHWILA 200
/* Definicja portu obslugującego przyciski */
#define P_PORT B                    // Zarówno przyciski jak i baza tranzystora sterujacego diodą podczerwieni
#define P_PIN  B                    // są podłączone do portu B ,jednak inne makra zostały użyte, aby program
#define P_DDR  B                    // był bardziej czytelny

#define P1 (1 << PB1)                // Makra uproszczające dostęp do rejestru portu B
#define P2 (1 << PB3)                // odpowiedzialnego za stan pinów podłączonych
#define P3 (1 << PB4)                // do przycisków
#define P4 (1 << PB5)
#define P5 (1 << PB6)
#define P_MASKA (P1 | P2 | P3 | P4 | P5) // Maska dla pinów dotyczących obslugi przycisków
                                         // wykorzystana w programie w celu programowego wykrycia wciskanego przycisku
/*Makra użyte w celu przejrzystości kodu podczas dostępu do portów */
#define PORT(x) XPORT(x)             // sklejanie nazw oraz użycie operatora # do uzyskania
#define XPORT(x) (PORT##x)           // stringa z literą odpowiedniego portu
#define PIN(x) XPIN(x)
#define XPIN(x) (PIN##x)
#define DDR(x) XDDR(x)
#define XDDR(x) (DDR##x)

uint8_t adres;                 // zmienna przechowująca adres urządzenia
uint8_t komenda;               // zmienna przechowująca przypisaną funkcję dla układu wykonawczego

/* MAIN */
int main(void)
{
    /********* INICJALIZACJA DIODY IR i LED********/

    DDR(IR_DDR)  |= (IR);      // Ustawienie PORTB: pin2 (0C0A) jako wyjście
    PORT(IR_PORT) |= (IR);     // Stan wysoki na pinie podłączonym do bazy tranzysotra PNP
                               // zamyka przepływ prądu przez tranzystor

    // INICJALIZACJA PRZYCISKÓW
    DDR(P_DDR) |= (P1 | P2 | P3 | P4 | P5);    // piny podłączone do przycisków ustawione jako wyjścia
    PORT(P_PORT) &= ~(P1 | P2 | P3 | P4 | P5);   // oraz ustawienie na nich stanu niskiego

    // Załączenie diody przy podłączeniu zasilania w celach diagnostycznych
    IR_ON;          // Ustawienie stanu niskiego na pinie sterującym prądem bazy
    _delay_ms(CHWILA); // Przez 200 ms dioda przy włączeniu zasilania będsię świeciła
    IR_OFF;

    /********TIMER0 - USTAWIENIA*********************/
                                      // Timer0 Będzie służył do generowania
                                      //fali nośnej według standardu RC(~36kHz)
   OCR0A = 110;                        // OCR0A= (F_CPU/(2*preskaler)*Fout)-1
   TCCR0A |= (1<<WGM01);               // Tryb CTC dla timera0
   TCCR0B |= (1<<CS00);                // Ustawienie preskalera timera0 na wartośc = 1
   /*************************************************/

   /*********TIMER1 - USTAWIENIA*********************/
   //Timer1 przeznaczony jest do odmierzania opóźnień w funkcji czekaj_us() (z dokładnością do wielekrotności 1us)
   //Preskaler ustawiany na wartośc = 8 (podczs jego załączania)
   // Używana jest wtedy flaga 0CF0A
   // nie używamy przerwań
   TCCR1B |= (1<<WGM12); //Tryb CTC dla Timera1
   /**************USTAWIENIE OBNIŻONEGO POBORU PRĄDU*****************************/
  ACSR |= (1<<ACD);                   // Ustawienia komparatora analogowego w celu obniżenia poboru prądu
  WDTCSR = 0x00;                      // Wyłączenie tryby watch-dog
  MCUCR |= ((1<<SM0 | (1<<SM1)));   // Ustawienie tryby POWER_DOWN
  /*****************************************************************************/

 // Ustawienie Pinu PD2 (INT0) jako wejście  (połączone są z nim jedną końcówką przyciski)
 DDR(D) &= ~(1<<PD2);
 PORT(D) |= (1<<PD2);             // Podciagniecie pod VCC
 GIMSK |= (1<<INT0);              // zezwolenie na przerwania zewnętrzne od pinu PD2(INTO)
 EIFR |= (1<<INTF0);              // Skasowanie flagi wystąpienia przerwania

 /*** GLOBALNE ZEZWOLENIE NA PRZERWANIA ***/
                  sei();                    // funkcja udostępniona przez producenta w jednej
/*************************************/       // z bibliotek w celu ułatwienia dostępu do rejestru
/****** PĘTLA GŁÓWNA ********/               // zarządzacającego przerwaniami mikrokontrolera
  while(1)
  {
      sleep_mode();             // wprowadzneie procesora w tryb uśpienia w celu oszczędności energii
  }                             // oczekując na przerwanie od wciśniętego przycisku
}
//--------------------FUNKCJE UŁATWIAJACE KODOWANIE PRZY UŻYCIU STANDARDU RC5-----------------------//
```

Kod źródłowy programu dla układu odbiorczego (opracowanie własne)

```
void czekaj_us(uint16_t usekundy)           // Dokładna pętla opóźniająca wielokrotności 1us
{                                           // przy pomocy TIMERA1 , taktowanie ustawione na 8Mhz, preskaler 8.
OCR1A = usekundy;                           // Dzięki tej funkcji można łatwo włączyć diodę podczerwieni
TIFR |= (1<<OCF1A);                         // w celu emitowania częstotliwości nośnej przez ściśle określony czas
TCCR1B |= (1<<CS11);                        // wyrażony w mikrosekundach
while(!(TIFR & (1<<OCF1A))) {};
TCCR1B &= ~(1<<CS11);
}

void wyslij_1()
{
czekaj_us(889);
TCCR0A |= (1<<COM0A0); //Toggle OC0A on Compare Match (zmień stan pinu PB2(OC0A) przy porównaniu licznika)
czekaj_us(889);
TCCR0A &= ~(1<<COM0A0);
}

void wyslij_0()
{
    TCCR0A |= (1<<COM0A0); //Toggle OC0A on Compare Match (zmień stan pinu PB2(OC0A) przy porównaniu licznika)
    czekaj_us(889);
    TCCR0A &= ~(1<<COM0A0);
    czekaj_us(889);
}
/* Przesłanie całej ramki w standardzie RC5 */
 void wyslij_ramke ( uint8_t adr, uint8_t cmd, uint8_t tog)
 {
   uint16_t data = 0;
   uint8_t i = 15;
   // Formowanie ramki w standardzie RC5
   // ,aby to zrealizowac przesuwamy bity do wysłania w lewą stronę
   // przy pomocy dekrementacji zmiennej i

   data |= ((1<<15) | (1<<14) | (tog<<13) | (adr<<8) | (cmd<<2));

    // wysyłamy kolejno:
   //         2 bity startu ,  1 bit toggle , 5 bitów adresu , 6 bitów komendy
   do
   {
     if (!(data & (1 << i )))  wyslij_0();
     else wyslij_1();

   }while(--i >1);

 }
 /******KONIEC DEFINICJI FUNKCJI POMOCNYCH PRZY NADAWANIU SYGNAłU W STANDARDZIE RC5********/

/*****************************PRZERWANIE OD PINU INT0***********************************/
                  /*Wywołane wciśnięciem, któregoś z przycisków*/
 ISR(INT0_vect)
 {
     uint8_t przyciski = PIN(P_PIN);    // zmienna do której przypisany zostanie rejest PINB
                                        // zmienna pomocnicza przy wykrywaniu wciśniętego przycisku
     static uint8_t toggle_bit;
     // Przez czas trwania przerwania INT0
     // Przestawiony zostaje kierunek portów klawiszy .
     //PORTB ustawiony jako wejścia  ,a pin PORTD2(INTO) jako wyjście ze stanem niskim
     DDR(P_DDR) &= ~(P1 | P2 | P3 | P4 | P5);    // piny podłączone do przycisków ustawione jako wejścia
     PORT(P_PORT) |= (P1 | P2 | P3 | P4 | P5);   // podciągnięcie wejśś do Uzas

     DDR(D) |= (1<<PD2); //PORTD2 (INT0) jako wyjście
     PORT(D) &= ~(1<<PD2); // ustawienie stanu niskiego (obsługa klawiszy)
     if ( (przyciski & P_MASKA) != P_MASKA )  //sprawdzenie, czy wciśnięty jakikolwiek klawisz
     {
         _delay_ms(45);                     // pozbycie się drgań styków
         przyciski = PIN(P_PIN);            // Sprawdzamy dwa razy wciśnięty przycisk
         if( (przyciski & P_MASKA) != P_MASKA )
         {
           toggle_bit ^= (1<<0); // przed wysłaniem ramki danych zmieniamy wartośc bitu toggle na przeciwną

           /* przy wykorzystaniu pętli DO_WHILE przypisujemy dynamicznie do zmiennej komenda oraz adres odpowiednie
            wartości przypisane do poszczególnych przycisków
            */
           do {
               if(!(przyciski & P1))
                               {
                               komenda = KOMENDA_1;
                               adres = ADRES_1;
                               }

               else if(!(przyciski & P2))
                               {
                               komenda = KOMENDA_2;
                               adres = ADRES_2;
                               }
               else if(!(przyciski & P3))
                               {
                               komenda = KOMENDA_3;
                               adres = ADRES_2;
                               }
               else if(!(przyciski & P4))
                               {
                                komenda = KOMENDA_4;
                                adres = ADRES_3;
                                }
               else if(!(przyciski & P5))
                               {
                                komenda = KOMENDA_5;
                                adres = ADRES_3;
                               }

               wyslij_ramke(adres, komenda, toggle_bit);
               _delay_ms(120); //przerwa pomiedzy przesyłanymi ramkami
               przyciski =  PIN(P_PIN) ; // ponowne wczytanie stanów klawiszy

           } while ( (przyciski & P_MASKA) != P_MASKA ); // warunek powtarzający przesłanie ramki ,jeśli przycisk wciąż wciśnięty
         }
     }
   /********** Przywrócenie ustawień pinów po przerwaniu wywołanym wciśnięciem przycisku**********/

    DDR(D) &= ~(1<<PD2); // pin PORTD.2 (INTO) ustawiamy jako wejscie
    PORT(D) |= (1<<PD2);  // z podciągnięciem do stanu wysokiego w celu oczekiwania na impuls
                          // pochodzący od
    DDR(P_DDR) |= (P1 | P2 | P3 | P4 | P5);    // piny podłączone do przycisków ustawione jako wyjścia
    PORT(P_PORT) &= ~(P1 | P2 | P3 | P4 | P5); // ustawnienie na pinach stanu niskiego
    EIFR |=(1<<INTF0);                         // kasowanie flagi przerwania
 }
```

c.d. Kod źródłowy programu dla układu odbiorczego (opracowanie własne)

```c
/*
 * odbieranie.c
 *
 *  Created on: 11 wrz 2018
 *      Author: Pi
 */
#include <avr/io.h>
#include <avr/interrupt.h>
#include "odbieranie.h"
 //  Zmienne przechowujące poszczególne bity odbieranej ramki

volatile uint8_t adres;     // zawiera 5 bitów adresu urządzenia
volatile uint8_t komenda;  // zawiera 6 bitów definiujących rozkaz
volatile uint8_t toggle;  // 1 bit pomocniczy
volatile uint8_t flaga_dostepu;   // jeśli == 1 informuje o odebranym nowym kodzie
                                  // po odczytaniu ramki należy wyzerowac

volatile uint8_t licznik_rc5;   // Licznik występujacych zboczy

/* Funkcja inicjalizująca odpowiedni PORT oraz Timer */
void inicjalizacja()
{
    DDR(IR_PORT) &= ~(IR_IN);   // Ustawienie pinu jako wejście
    PORT(IR_PORT) |= IR_IN;     // Podciagnięcie pinu do Vcc
    /*Procedura ustawiajaca preskaler Timera */
#if PRESKALER == 1
    TCCR1B |= (1<<CS10);
#endif
#if PRESKALER == 8
    TCCR1B |= (1<<CS11);
#endif
#if PRESKALER == 64
    TCCR1B |= (1<<CS10) | (1<<CS11);;
#endif
#if PRESKALER == 256
    TCCR1B |= (1<<CS12);
#endif
  /* KONIEC PROCEDURY */
TCCR1B &= ~(1<<ICES1); // Reakcja na zbocze opadajace od pinu ICP1 (IR_IN)
licznik_rc5 = 0;       // Wyzerowanie licznika zboczy
TIMSK |= (1<<TICIE1); // Załączenie przerwania od
flaga_dostepu = 0; // Zerowanie flagi otrzymania kodu
}
 /* PROCEDURA OBSŁUGUJĄCE PRZERWANIE POCHODZĄCE OD PINU IPC1 */  // Przerwanie reaguje na zbocza wystepujące na pinie ICP1
ISR(TIMER1_CAPT_vect)
{
#define RESTART 0
#define DALEJ   1
#define KONIEC  2
#define BLAD    3
/* Deklaracja zmiennych statycznych używanych w przerwaniu */
      uint16_t Szerokosc_Impulsu; //Obliczanie czasu każdej połówki
static uint16_t Poprzedni_Impuls;  // pomocnicza zmienna przy obliczaniu Szerokości_impulsu
static uint16_t IR_dane;           //  16 bitowa zmienna przechwująca adres
static uint8_t IR_impulsy;
static uint8_t Status_Ramki;

Szerokosc_Impulsu = ICR1 - Poprzedni_Impuls; // Pobranie czasu trwania impulsu
Poprzedni_Impuls = ICR1;

TCCR1B ^= (1<<ICES1); // Zmiana zbocza wyzwalającego na przeciwne ,aby móc wywołac kolejne przerwanie zboczem przeciwnym do poprzedzającego

if (Szerokosc_Impulsu > MAX_BIT) licznik_rc5=0;

if (licznik_rc5 > 0) Status_Ramki = DALEJ;

if (licznik_rc5 == 0) //Początek ramki
 {
    IR_dane = 0;
    IR_impulsy = 0;
    TCCR1B |= (1<<ICES1);
    licznik_rc5++;
    Status_Ramki = KONIEC;
 }
if (Status_Ramki == DALEJ)
   {
    if (Szerokosc_Impulsu < MIN_POL_BIT ) Status_Ramki = RESTART; // W przypadku zakłóceń bądź błędu odbioru ramki

    if (Szerokosc_Impulsu > MAX_BIT ) Status_Ramki = RESTART;

    if(Status_Ramki == DALEJ)
     {
        if(Szerokosc_Impulsu > MAX_POL_BIT) licznik_rc5++;
        if(licznik_rc5 > 1)
        if( (licznik_rc5 % 2) == 0)
             {
               IR_dane = IR_dane << 1 ;
               if((TCCR1B & (1<<ICES1))) IR_dane |= 0x0001;
               IR_impulsy++;


               if(IR_impulsy > 12)
                {
                 if (flaga_dostepu == 0)
                  {
                   komenda = IR_dane & 0b0000000000111111;
                   adres = (IR_dane & 0b0000011111000000) >> 6;
                   toggle = (IR_dane & 0b0000100000000000) >> 11;
                 }
                     Status_Ramki = RESTART;
                     flaga_dostepu = 1;
             }
             }
          licznik_rc5++;
       }
     }
          if (Status_Ramki == RESTART)
          {
              licznik_rc5 = 0;
              TCCR1B &= ~(1<<ICES1);
          }
   }
```

Kod źródłowy odpowiedzialny za dekodowanie sygnału (opracowanie własne)

```
/*
 * odbieranie.h
 *
 *  Created on: 12 wrz 2018
 *      Author: Pi
 */

#ifndef ODBIERANIE_H_
#define ODBIERANIE_H_

 /* DEFINICJA PINU podłaczonego do Odbiornika */
#define IR_PORT D
#define IR_PIN 6
#define IR_IN (1<<IR_PIN)
/* DEFINICJA PINU podłaczonego do Odbiornika */


/*Przeliczenie na mikrosekundy liczby taktów zegara dla ustawionego preskalera */
#define PRESKALER 8
#define mikro_sec(us) ((us)*(8000000/1000000)/PRESKALER) // Gdy podzielimy przez 10^3 to pewna liczba taktów zegara
                                                         // w zależności od taktowania i preskalera
                                                         // będzie określała 1 mikrosekundę

/*stałe czasowe i zmienne potrzebne do obslugi standardu RC5*/
#define TOLERANCJA 200
#define MAX_POL_BIT mikro_sec(889 + TOLERANCJA)
#define MIN_POL_BIT mikro_sec(889 - TOLERANCJA)
#define MAX_BIT mikro_sec((889+889) + TOLERANCJA)


/*Makra ułatwiające dostęp do portów */
#define PORT(x) XPORT(x)       // sklejanie nazw oraz użycie operatora # do uzyskania
#define XPORT(x) (PORT##x)     // stringa z literą odpowiedniego portu

#define PIN(x) XPIN(x)
#define XPIN(x) (PIN##x)

#define DDR(x) XDDR(x)
#define XDDR(x) (DDR##x)

/* DEKLARACJE ZMIENNYCH DLA KAŻDEGO PLIKU PROGRAMU */

extern volatile uint8_t adres;
extern volatile uint8_t komenda;
extern volatile uint8_t toggle;
extern volatile uint8_t flaga_dostepu;

/* DEKLARACJE FUNKCJI */
void inicjalizacja();


#endif /* ODBIERANIE_H_ */

```

Plik nagłówkowy biblioteki odpowiedzialnej za dekodowanie sygnału (opracowanie własne)

```c
/*
 * main.c
 *
 *  Created on: 3 sty 2019
 *      Author: Pi
 */
#define F_CPU 8000000UL  // Przypisane wartości częstotliwości taktowania mikrokontrolera dla kompilatora
#include <avr/io.h>      // Dołączenie niezbędnych bibliotek od prodecenta w celu napisania oprogramowania
#include <avr/interrupt.h>
#include <util/delay.h>

#include "odbieranie.h"  // dołączenie pliku nagłówkowego biblioteki, która pozwala
                         // na dekodowanie sygnałów w standardzie RC5

#define ADRES_1 0        // Adres układu odbiorczego nr 2
#define KOMENDA_1 1      // Komendy przypisane do określonych przycisków
                      // KOMENDA_2 przydzielona jest do przycisku 2,
                         // KOMENDA_3 wysyłana jest przy wciśniętym przycisku 3

int main(void) {

    DDRB |= (1<<PB0);  // Ustawienie pinu PB0 portu B jako wyjście
    PORTB |= (1<<PB0); // Podanie stanu wysokiego na pin PB0 przez okres 1,5 sekundy
    _delay_ms(1500);   // w celu sprawdzenia działania kodu programu
    PORTB &= ~(1<<PB0);

    inicjalizacja();    /* inicjalizacja dekodowania IR */

    sei();  /* włączamy globalne przerwania */



    /* pętla nieskończonapozwalająca na ciągłą pracę programu */
    while(1) {

        if(flaga_dostepu) { /* jeśli odebrano prawidłowe kody z pilota */

            if( (adres == ADRES_1) && (komenda == KOMENDA_1) ) {             // Jeśli adres jest równy ADRES_2
                PORTB ^= (1<<PB0);                                           // to zmień stan na pinie PORTB.0

            }


            /* wyzerowanie flagi odbioru ramki oraz wartości odebranych kodów */
            flaga_dostepu=0;            //Wyzerowanie wartości pozwoli na odbiór kolejnej ramki danych
            komenda=0xff;               // z nowym zestawem bitów ramiki danych
            adres=0xff;
        }

    }
}
```

Kod źródłowy dla układu odbiorczego nr 1 (opracowanie własne)

```c
/*
 * main.c
 *
 *  Created on: 3 sty 2019
 *      Author: Pi
 */
#define F_CPU 8000000UL  // Przypisane wartości częstotliwości taktowania mikrokontrolera dla kompilatora
#include <avr/io.h>      // Dołączenie niezbędnych bibliotek od prodecenta w celu napisania oprogramowania
#include <avr/interrupt.h>
#include <util/delay.h>

#include "odbieranie.h"  // dołączenie pliku nagłówkowego biblioteki, która pozwala
                         // na dekodowanie sygnałów w standardzie RC5

#define ADRES_2 1        // Adres układu odbiorczego nr 2
#define KOMENDA_2 2      // Komendy przypisane do określonych przycisków
#define KOMENDA_3 3      // KOMENDA_2 przydzielona jest do przycisku 2,
                         // KOMENDA_3 wysyłana jest przy wciśniętym przycisku 3

int main(void) {

    DDRB |= (1<<PB0);  // Ustawienie pinu PB0 portu B jako wyjście
    PORTB |= (1<<PB0); // Podanie stanu wysokiego na pin PB0 przez okres 1,5 sekundy
    _delay_ms(1500);   // w celu sprawdzenia działania kodu programu
    PORTB &= ~(1<<PB0);

    inicjalizacja();    /* inicjalizacja dekodowania IR */

    sei();  /* włączamy globalne przerwania */



    /* pętla nieskończonapozwalająca na ciągłą pracę programu */
    while(1) {

        if(flaga_dostepu) { /* jeśli odebrano prawidłowe kody z pilota */

            if( (adres == ADRES_2) && (komenda == KOMENDA_2) ) {             // Jeśli adres jest równy ADRES_2
                PORTB ^= (1<<PB0);                                           // to zmień stan na pinie PORTB.0

            }
            else if( (adres == ADRES_2) && (komenda == KOMENDA_3) ) {        // Jeśli adres jest równy ADRES_2
                                                                             // i komenda jest równa KOMENDA_3
                PORTB ^= (1<<PB0);                                           // to zmień stan na pinie PORTB.0

            }

            /* wyzerowanie flagi odbioru ramki oraz wartości odebranych kodów */
            flaga_dostepu=0;            //Wyzerowanie wartości pozwoli na odbiór kolejnej ramki danych
            komenda=0xff;               // z nowym zestawem bitów ramiki danych
            adres=0xff;
        }

    }
}
```

Kod źródłowy dla układu odbiorczego nr 2 (opracowanie własne)

```c
/*
 * main.c
 *
 *  Created on: 3 sty 2019
 *      Author: Pi
 */
#define F_CPU 8000000UL  // Przypisane wartości częstotliwości taktowania mikrokontrolera dla kompilatora
#include <avr/io.h>      // Dołączenie niezbędnych bibliotek od prodecenta w celu napisania oprogramowania
#include <avr/interrupt.h>
#include <util/delay.h>

#include "odbieranie.h"  // dołączenie pliku nagłówkowego biblioteki, która pozwala
                         // na dekodowanie sygnałów w standardzie RC5
#define ADRES_3 2        // Adres układu odbiorczego nr 3
#define KOMENDA_4 4      // Komendy przypisane do określonych przycisków
#define KOMENDA_5 5      // KOMENDA_4 przydzielona jest do przycisku 2,
                         // KOMENDA_5 wysyłana jest przy wciśniętym przycisku 3
#define T1 (1<<PB1)
#define T2 (1<<PB2)    // Ustawienie podłaczonych do sterownika wyjśc mikrokontrolera
#define T3 (1<<PB3)
#define T4 (1<<PB4)

#define KROK_1 PORTB |= T1; PORTB &= ~(T2|T3|T4)  // Makrodefinicje wykorzystywane podczas
#define KROK_2 PORTB |= T2; PORTB &= ~(T1|T3|T4)  // podawania stanów logicznych w celu
#define KROK_3 PORTB |= T3; PORTB &= ~(T1|T2|T4)  // w celu sterowania silnikiem
#define KROK_4 PORTB |= T4; PORTB &= ~(T1|T2|T3)

int main(void) {
    uint16_t czas = 0;
    DDRB |= (T1|T2|T3|T4);  // Ustawienie pinu PB0 portu B jako wyjście dla sterowania silnikiem
    PORTB &= ~(T1|T2|T3|T4);// wyłączenie wszystkich pinów sterujących

    inicjalizacja();    /* inicjalizacja dekodowania IR */

    sei();  /* włączamy globalne przerwania */

    /* pętla nieskończonapozwalająca na ciągłą pracę programu */
    while(1) {

        if(flaga_dostepu) { /* jeśli odebrano prawidłowe kody z pilota */

            if( (adres == ADRES_3) && (komenda == KOMENDA_4) ) {
                czas = 0;                                       // Jeśli adres jest równy ADRES_3
                            while(czas < 1000){                 // Za pomocą pętli whileustawienie czasu trwania działania silnika
                             KROK_1;                            // A komenda równa KOMENDA_4
                             _delay_ms(5);                      // to steruj silnikiem w prawo
                             KROK_2;
                             _delay_ms(5);
                             KROK_3;
                             _delay_ms(5);
                             KROK_4;
                             _delay_ms(5);
                             czas++;
                            }

            }
            else if( (adres == ADRES_3) && (komenda == KOMENDA_5) ) {
                czas = 0;                                                   // Jeśli adres jest równy ADRES_3
                            while(czas < 1000){
                             KROK_4;                                        // A komenda równa KOMENDA_5
                             _delay_ms(5);                                  // to steruj silnikiem w lewo
                             KROK_3;
                             _delay_ms(5);
                             KROK_2;
                             _delay_ms(5);
                             KROK_1;
                             _delay_ms(5);
                             czas++;
                            }
                    }
            }

            /* wyzerowanie flagi odbioru ramki oraz wartości odebranych kodów */
            flaga_dostepu=0;            //Wyzerowanie wartości pozwoli na odbiór kolejnej ramki danych
            komenda=0xff;               // z nowym zestawem bitów ramiki danych
            adres=0xff;
        }

    }
```

Kod źródłowy dla układu odbiorczego nr 3 (opracowanie własne)

O Ś W I A D C Z E N I E

Ja niżej podpisany wyrażam zgodę na udostępnienie mojej pracy w czytelni Archiwum WAT oraz w ramach wypożyczeń międzybibliotecznych.

Filip Mikołaj BAGIŃSKI

Warszawa, dnia …….

Wykonano w jednym egzemplarzu.

Warszawa, dnia ……...