Patrycja Paradowska
numer albumu: 244952

30 maja 2020r.
prowadzący: dr. inż. Przemysław Błaśkiewicz

Systemy wbudowane - projekt

System blokady drzwi i kontroli dostępu

Algorytmy realizowane przez system

1 Wstęp

W poprzednim zadaniu sporządziłam opis stanów, w jakich mój system może się znajdować. Zgodnie z tymi stanami i funkcjonalnościami, jakie mogą być w nich wykonywane, w tym pliku przeanalizuję realizowane przez system podstawowe, najważniejsze algorytmy, umożliwiające działanie systemu. Kiedy system jest w stanie "Wyłączonym", to wiadomo, że nie może wykonywać żadnych akcji, więc nie są wtedy realizowane żadne z wymienionych niżej algorytmów.

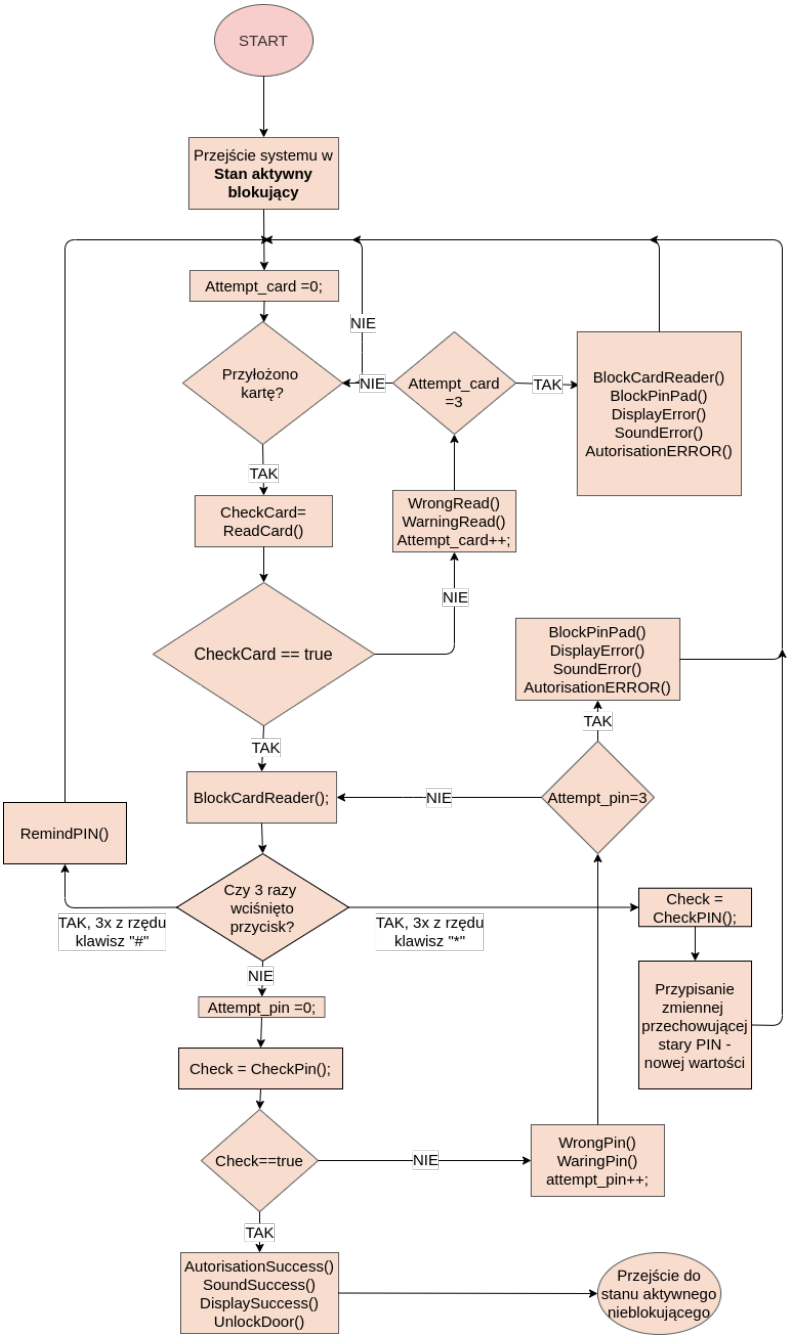
2 Wprowadzanie danych

Na początku przedstawię algorytm realizujący bardzo ważne zadanie w systemie - wprowadzanie danych - skutkujące wykonaniem pierwszego z przypadków użycia, czyli odblokowania drzwi. Po przejściu w stan "Aktywny blokujący" system ustawia attempt-card, czyli ilość błędnych odczytów karty na wartość równą zero i oczekuje na przyłożenie karty.
Jeżeli karta zostanie przyłożona do czytnika, to następuje przypisanie do zmiennej CheckCard typu bool wartość funkcji ReadCard(). Pozytywny odczyt przypisuje wartość 1, a negatywny wartość 0. Rozważmy dwie sytuacje:

- **Odczyt negatywny** - następuje wzrost licznika błędnych odczytów o 1, a funkcja WarningRead wyświetla komunikat o błędzie. Funkcja WrongRead() przekazuje do kontrolera dane o błędnym odczycie karty. Jeśli nastąpi sytuacja, że ilość błędnych odczytów osiągnie wartość równą 3, to funkcja BlockCardReader() blokuje mozliwość odczytu karty, nadaje komunikat o niepowodzeniu autoryzacji - AutorisationERROR(), blokuje klawiaturę (BlockPinPad()), informuje dźwiękowym sygnałem o błędzie odczytu - SoundError() oraz wyświetla powiadodmienie - DisplayError().
- **Odczyt pozytywny** - czytnik kart zostaje zablokowany (BlockCardReader()). Sprawdzamy, czy użytkownik nie chciał wykonać przypadku użycia zmiany hasła lub przypomnienia hasła. Jeśli naciśnięto 3 razy z rzędu klawisz "#", to funkcja RemindPIN() przekazuje informacje o wysłaniu do użytkownika powiadomienia SMS z obecnym hasłem. Jeśli wciśnięto trzy razy z rzędu klawisz "*", to sprawdzamy, czy numer PIN został wprowadzony i po pozytywnej weryfikacji hasła następuje przypisanie do zmiennej przechowującej stary PIN - nowej wartości. Jesli jednak nie wciśnięto trzy razy ani przycisku "#" ani "*", to następuje ustawienie ilości błędnych odczytów pinu na 0 (attempt-pin = 0) oraz sprawdzanie, czy numer PIN został wprowadzony (CheckPin()) i podajemy wartość z funkcji do zmiennej typu bool Check.

a) **Jeżeli zmienna Check ma wartość False**, to zwiększamy wartość licznika błędnych wprowadzeń numeru PIN o 1. Informujemy o niepoprawnym haśle (WrongPin()) oraz dźwiękowo o błędzie WarningPin(). Jeżeli wartość zmiennej attempt będzie równa 3, to wykona się lista poleceń analogiczna do sytuacji odczytu karty, czyli błąd autoryzacji, komunikaty oraz blokada klawiatury.

b) **Jeżeli zmienna Check ma wartość True**, to nadajemy komunikat o udanej autoryzacji - AutorisationSuccess(). Wyświetlamy komunikat - DisplaySuccess() oraz informujemy dźwiękiem o poprawnej autoryzacji - SoundSuccess(), a funkcja UnlockDoor() przekazuje informacje o możliwości odryglowania drzwi po przejściu poprawnej autoryzacji.

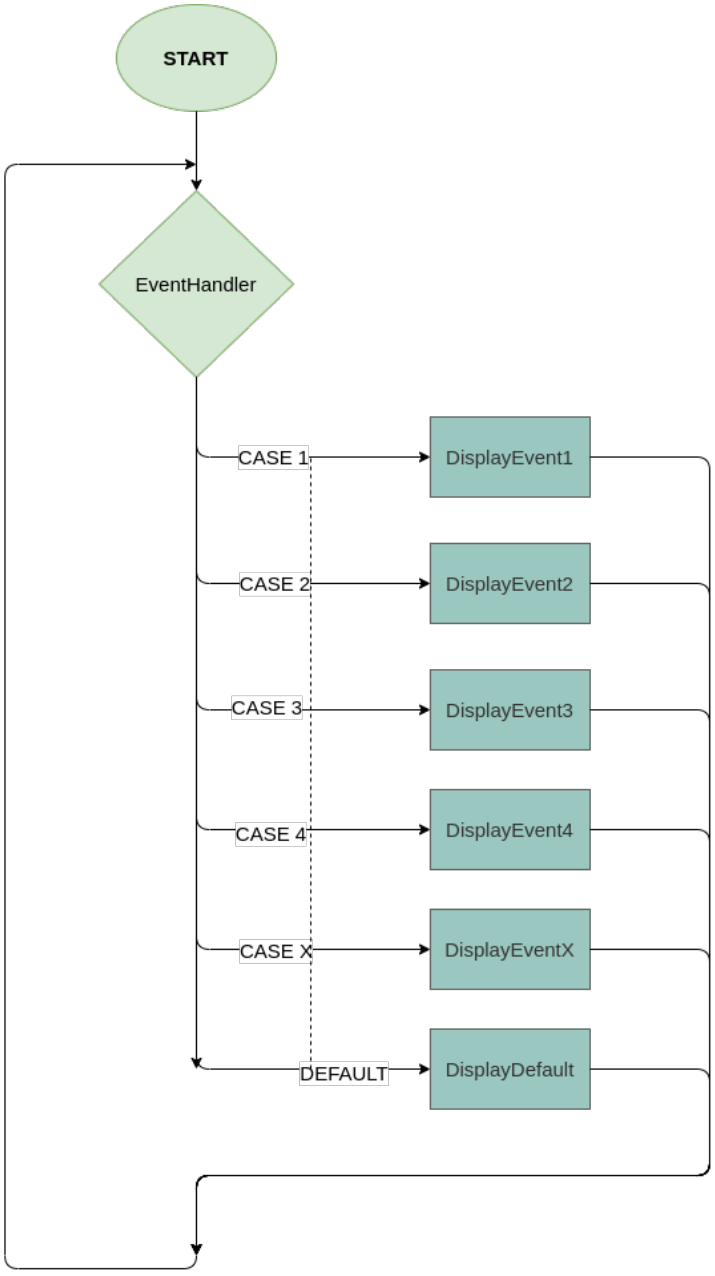


Rysunek 1: Schemat przedstawiający algorytm wprowadzania danych

Po udanym odblokowaniu drzwi system przechodzi do stanu aktywnego nieblokującego.

3 Wyświetlanie informacji

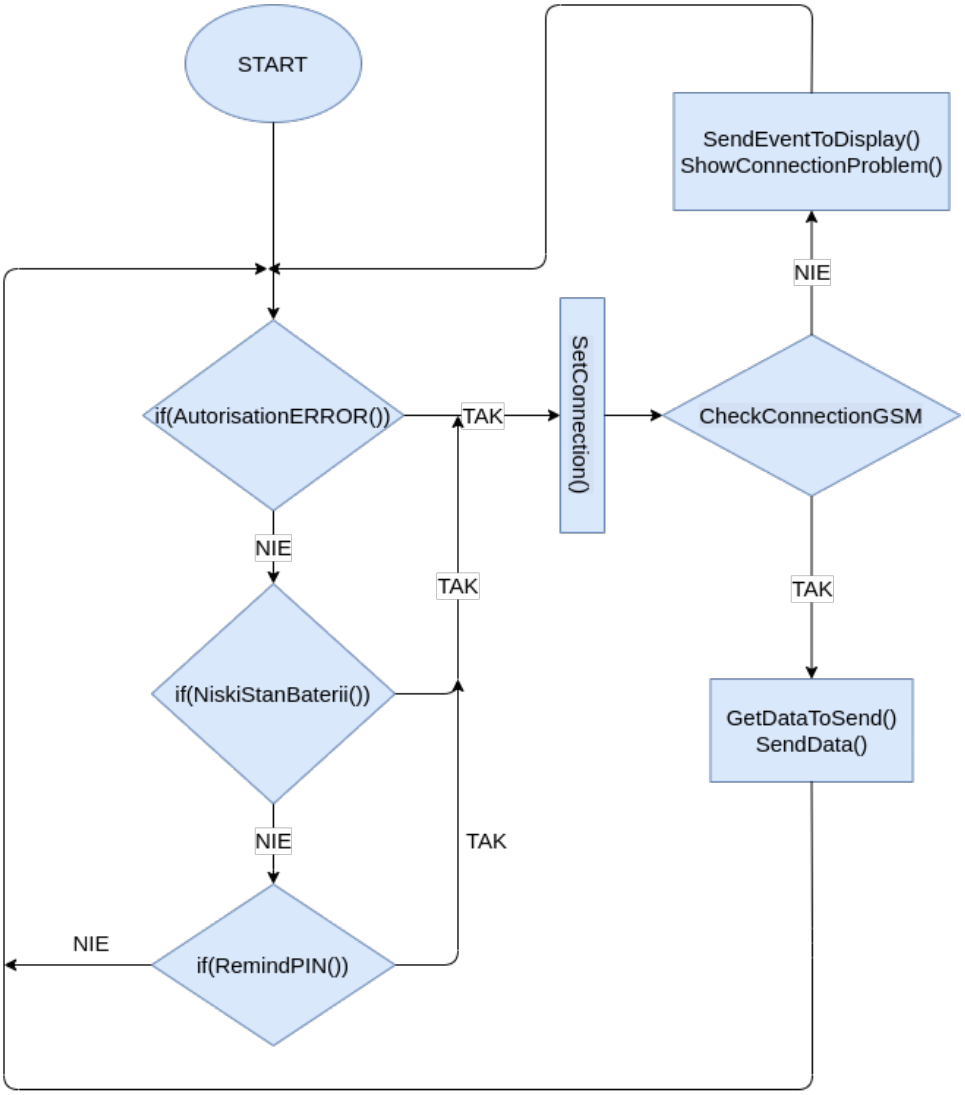
Kolejnym algorytmem, który chciałabym przedstawić, jest obsługa zdarzeń przez wyświetlacz. Działa on na zasadzie przełącznika (w kodzie – switch). Na podstawie zmiennej EventHandler przechwytuje zadaną wartość wydarzenia, które ma obsłużyć. Na podstawie tej zmiennej wyświetlacz posiada przełącznik mogący obsłużyć X rożnych wydarzeń do wyświetlenia, jak np. błędny numer PIN, informacje o PIN-ie, który właśnie jest wprowadzany itd. Posiada również domyślny stan (Default), który będzie wyświetlany w czasie oczekiwania systemu na rozpoczęcie akcji przez użytkownika.



Rysunek 2: Schemat przedstawiający obsługę zdarzeń przez wyświetlacz

4 GSM

Algorytm sprawdza, czy funkcja AutorisationERROR() zwróciła wartość true, jeżeli nie to sprawdza, czy została zwrócona wartość przez funkcję NiskiStanBaterii() lub RemindPIN(). Jeżeli żaden z warunków nie zostanie spełniony, to następuje powtarzanie sprawdzania. Gdy ktoryś z warunków zwrocił wartość TRUE, to następuje ustawienie połączenia GSM z siecią właściciela. Jeżeli połączenie zostanie utworzone, program pobierze dane o błędzie GetDataToSend(), a następnie prześle poprzez GSM SendData() wiadomość SMS do użytkownika.
Natomiast w sytuacji, kiedy nie udało się utworzyć połączenia, program przekaże do wyświetlacza zdarzenie o błędzie połączenia GSM, które na podstawie odpowiedniego numeru Event wyda polecenie do wyświetlenia odpowiednich informacji. Dodatkowo GSM zacznie sygnalizować diodą/dźwiękiem komunikat o błędzie ShowConnectionProblem(), po czym ponownie będzie próbować uzyskać połączenie.

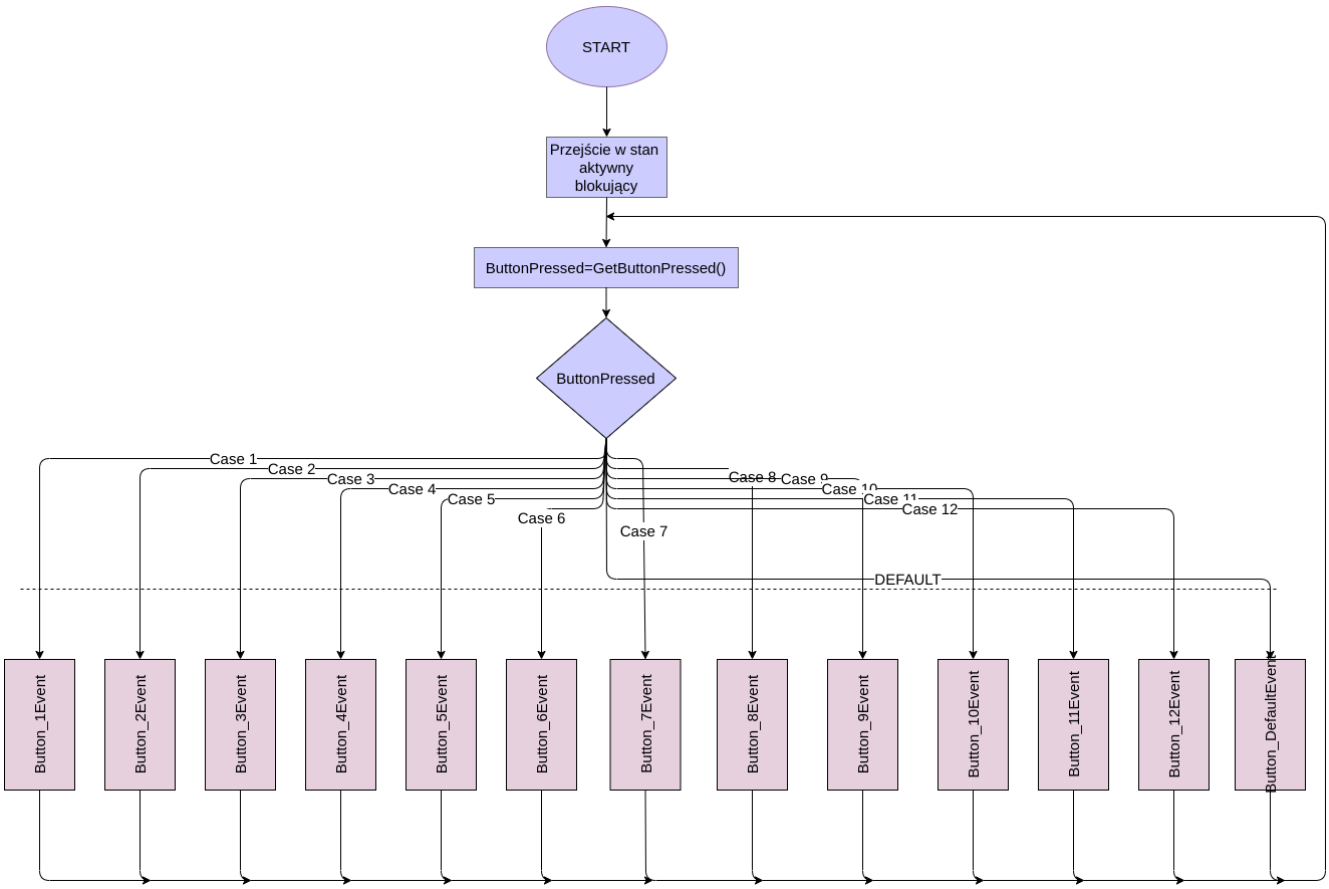


Rysunek 3: GSM

5 Klawiatura

Działanie klawiatury jest oparte na bardzo prostym algorytmie, który składa się z dwóch głównych części.

1. Funkcji GetButtonPressed() pobierającej kliknięty przycisk oraz przypisującej wartość przycisku do zmiennej ButtonPressed.
2. Drugą częścią tego algorytmu jest switch obsługujący naciśnięcie przycisku. Dla każdego z przycisków jest przypisana inna funkcja regulująca działanie całego programu. Przełącznik posiada również opcję domyślną DEFAULT, która posiada również swoją funkcję w przypadku nieklikniecia żadnego przycisku z klawiatury, na przykład w stanie aktywnym nieblokującym.



Rysunek 4: Schemat przedstawiający algorytm działania klawiatury

Podsumowując, wszystkie przedstawione algorytmy umożliwiają sprawne wykonywanie najważniejszych zadań opisywanego przeze mnie systemu.