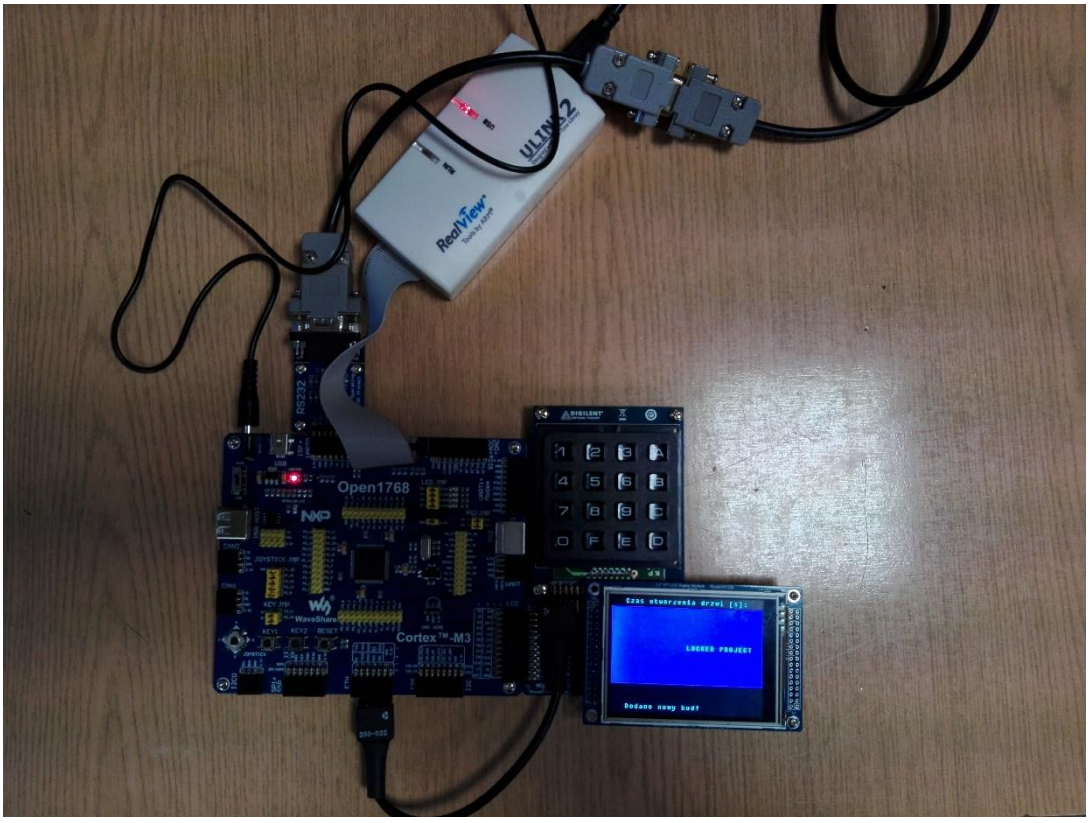
Dokumentacja projektu wykonanego w ramach zajęć Systemy Wbudowane

Autor: Hadam Paweł

1. OPIS PROJEKTU:

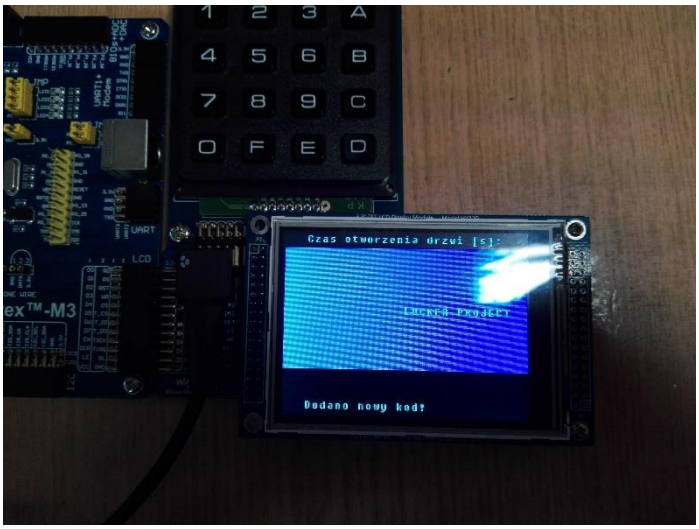
Projekt opisuje kontroler zamka na szyfr, którego można używać na przykład przy drzwiach wejściowych, czy też sejfach.



Rysunek 1 Locker Project

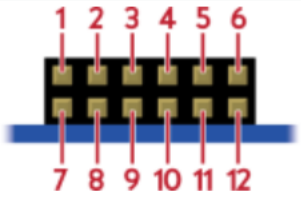
2. ZAŁOŻENIA PROJEKTU:

Projekt składa się z mikrokontrolera Cortex-M3, oraz klawiatury macierzowej PmodKYPD podłączonej do wejścia ETH mikrokontrolera w sposób przedstawiony na rysunku 1 (w projekcie dodatkowo nie są używane piny 5,6,11,12 klawiatury macierzowej), wyświetlacza LCD ILI9325 podłączonego do wejścia LCD oraz złącza RS232 podłączonego do wejścia UART0 służącego do komunikacji z PC.



Rysunek 2 LCD ILI9325

Rysunek 3 Klawiatura macierzowa PmodKYPD



Pin 1	COL4
Pin 2	COL3
Pin 3	COL2
Pin 4	COL1
Pin 5	GND
Pin 6	VCC
Pin 7	ROW4
Pin 8	ROW3
Pin 9	ROW2
Pin 10	ROW1
Pin 11	GND
Pin 12	VCC

Rysunek 4 Wyprowadzenia nóżek klawiatury matrycowej

Klawiatura macierzowa składa się z przycisków służących do wprowadzania danych: 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, F, E, D, B oraz przycisku akceptacji A (Accept), anulowania wyboru C (Cancel).

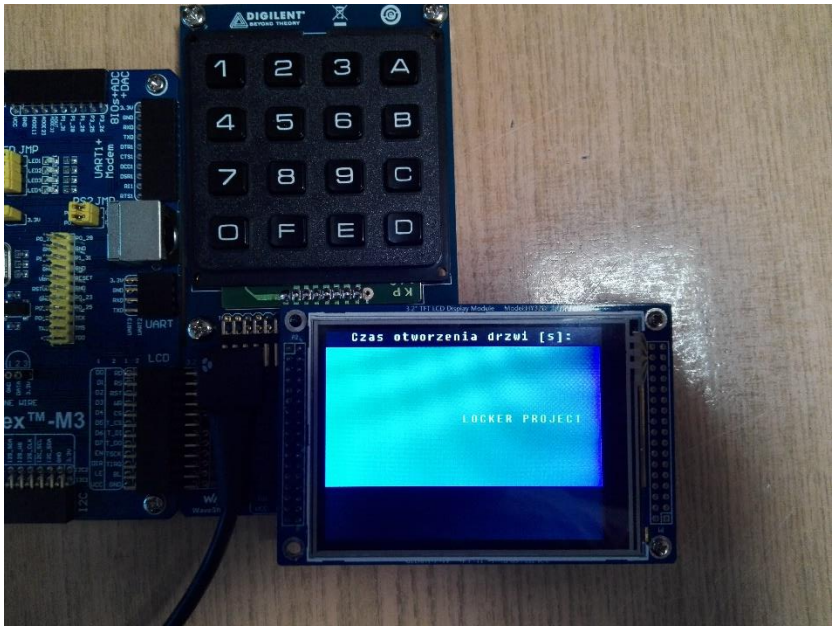
Program pozwala na:

- Odczyt szyfru z klawiatury macierzowej,
- Włączenie trybu wprowadzania nowych szyfrów poprzez wpisanie odpowiedniego kodu serwisowego („F997”) oraz akceptacji klawiszem „A”,
- Po włączeniu trybu wprowadzania nowych szyfrów, wpisaniu dowolnego kodu oraz akceptacji go klawiszem „A”, pozwala na dodanie szyfru do puli szyfrów kontrolera,
- Odblokowanie drzwi po wpisaniu poprawnego szyfru i akceptacji go klawiszem „A”,
- Cofnięcie obecnie wpisywanego szyfru, na przykład na skutek pomyłki poprzez wciśnięcie klawisza „C”,
- Cofnięcie obecnie wpisanego szyfru, który nie został akceptowany klawiszem „A” po upływie paru sekund,
- Zamknięcie drzwi po momencie od otwarcia ich,
- Wyświetlenie ostatnich 8 czasów otwarcia drzwi od uruchomienia kontrolera w sekundach, wyświetlanie odbywa się jako lista cykliczna, przy liczbie otwarć większej niż 8 najstarsze otwarcie znika, czasy otwarć są przesuwane, a w miejsce ostatniego czasu dodawane jest ostatnie otwarcie drzwi,
- Komunikację z użytkownikiem za pomocą komunikatów wysyłanych na LCD przy każdej czynności użytkownika,

- Pokazywanie obecnej ilości znaków szyfru za pomocą „*”,
- Wysyłanie komunikatów debugujących na UART wysyłanych przy każdej czynności użytkownika,
- Program posiada ograniczenie długości szyfru do maksymalnie 11 znaków,
- Dodatkowo dodano zabezpieczenie przez dodaniem kodu serwisowego jako kodu otwierającego drzwi.

3. INTERFEJS UŻYTKOWNIKA:

Po włączeniu kontrolera widzimy podstawowy widok:

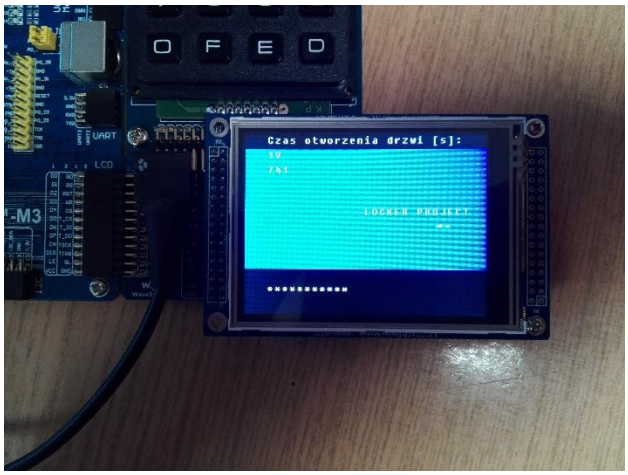


Rysunek 5 Widok początkowy

Przy wpisywaniu szyfru pojawiają się na ekranie gwiazdki określające długość wpisywanego szyfru:

Rysunek 6 Wpisywane hasło część 1

Rysunek 7 Wpisywane hasło część 2



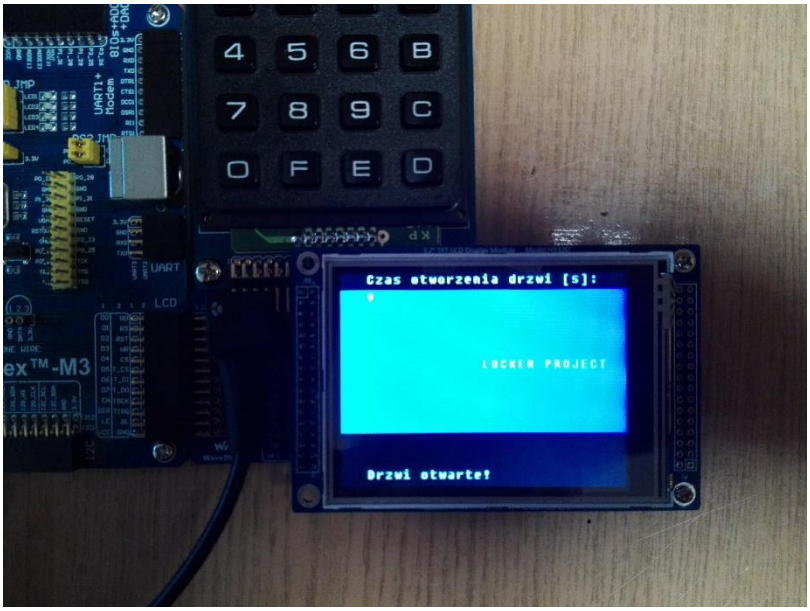
Rysunek 8 Maksymalna długość wpisywanego hasła

Wpisywany szyfr można w każdej chwili anulować przyciskiem „C” znajdującym się na klawiaturze:

Rysunek 9 Anulowanie wpisywanego kodu

W przypadku nie aktualizowania wpisywanego kodu lub nie podjęcia próby wpisania kodu przez określony czas kod jest resetowany i wyświetlane są komunikaty o minięciu czasu oraz anulowania wpisywania kodu.

W przypadku wpisania szyfru i akceptacji go przyciskiem „A” znajdującym się na klawiaturze uzyskamy następujący komunikat w sytuacji poprawnego kodu:



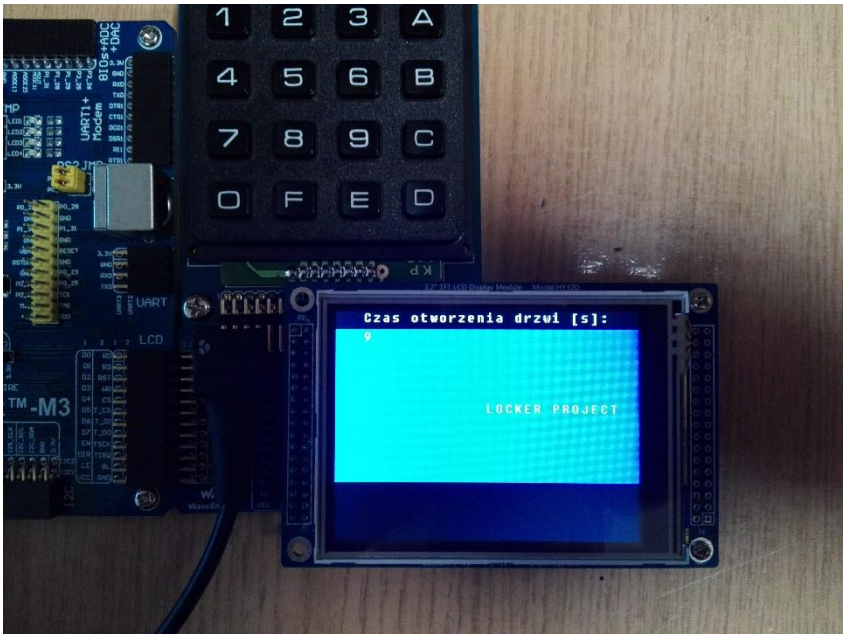
Rysunek 10 Komunikat o otwartych drzwiach – poprawny szyfr

Pozwala nam to na otworzenie drzwi w tym momencie, po chwili kontroler wraca do poprzedniego stanu, czyli zamkniętych drzwi oraz oczekiwania na wpisanie kodu.

W przypadku wpisania złego szyfru i akceptacji go przyciskiem „A” dostajemy poniższy komunikat:

Rysunek 11 Niepoprawny szyfr

W przypadku otwarcia drzwi do historii czasów otwarcia drzwi dodawany jest obecny czas.

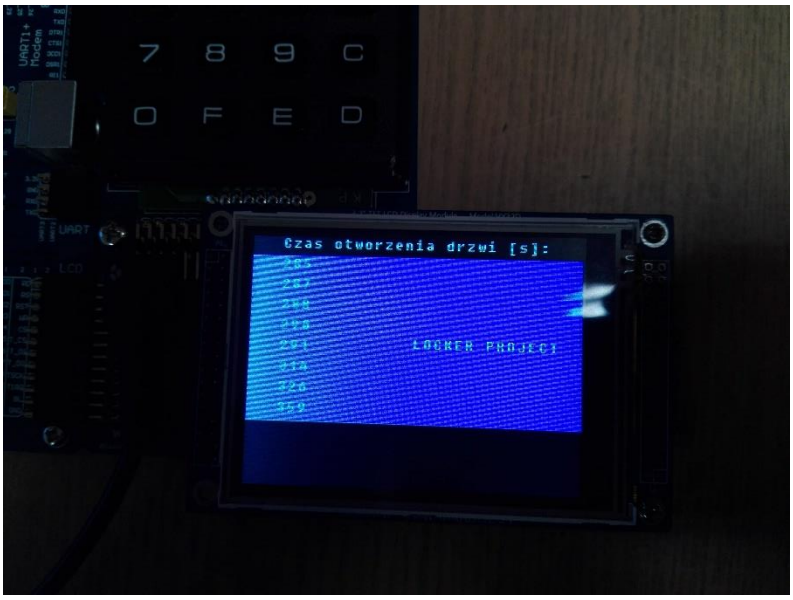


Rysunek 12 Pierwsze otwarcie w historii

W chwili gdy historia otwarć jest wypełniona (8 ostatnich otwarć) i otwieramy drzwi kolejny raz uruchamia się algorytm przesuwania listy, najstarsze otwarcie jest usuwane, reszta rekordów jest przesuwana w wstecz, a na ostatni element historii dodawany jest obecny czas otwarcia.

Rysunek 13 Pełna historia otwarć część 1

Rysunek 14 Pełna historia otwarć część 2



Rysunek 15 Pełna historia otwarć część 3

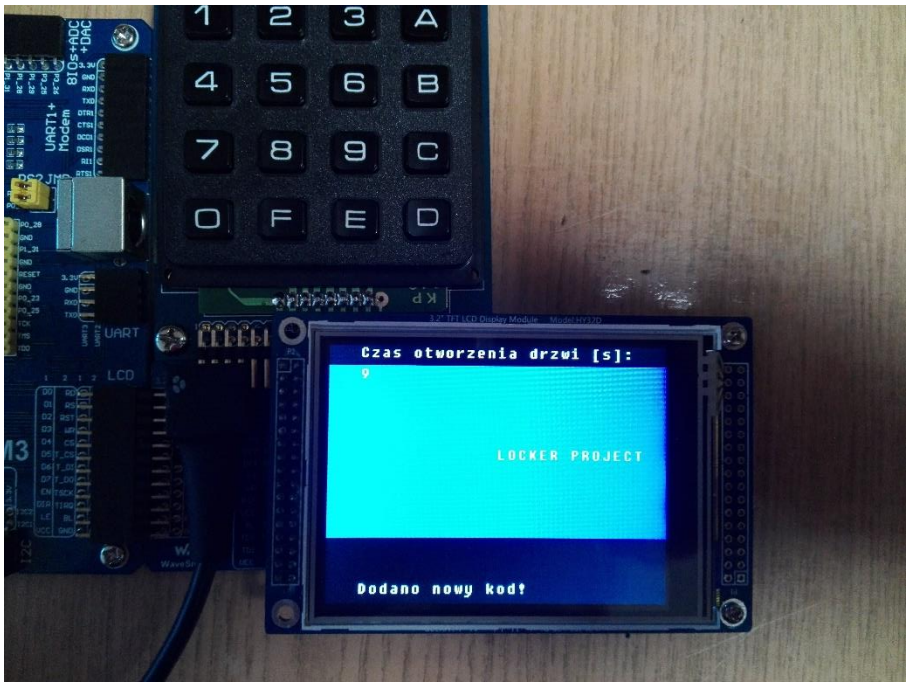
Gdy użytkownik wpisze kod serwisowy i kolejno go akceptując przyciskiem „A” na klawiaturze to wchodzimy w specjalny tryb serwisowy dodatkowo informując użytkownika o tym trybie:

Rysunek 16 Tryb serwisowy

Tryb ten pozwala nam na dodawanie kodów którymi możemy otworzyć drzwi.

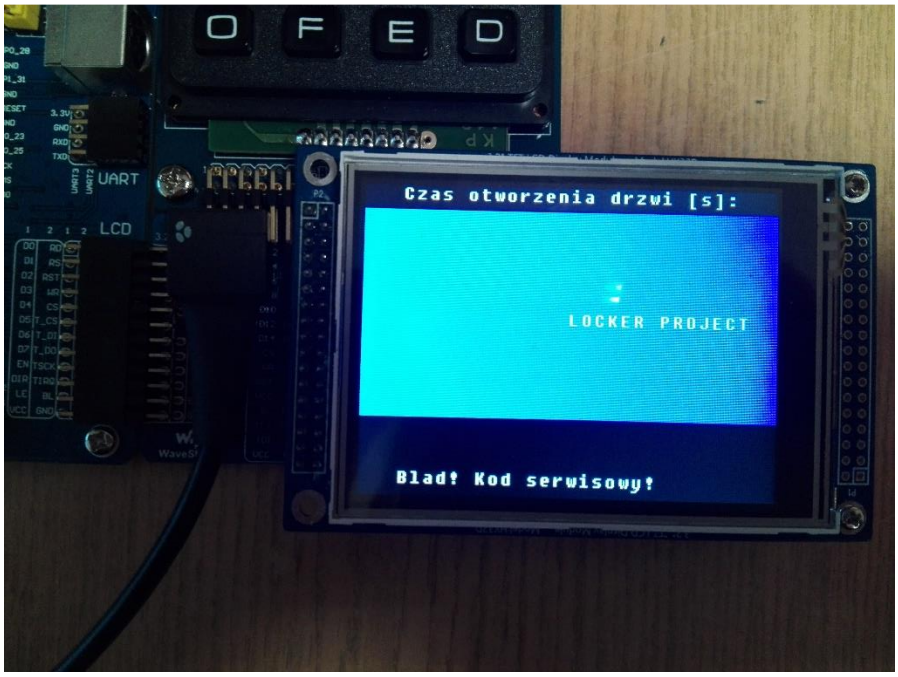
W przypadku nie aktualizowania wpisywanego kodu lub nie podjęcia próby wpisania kodu przez określony czas, tryb serwisowy automatycznie jest zamykany informując jednocześnie użytkownika o miniętym czasie i anulowania wpisywania kodu. Wyjście z trybu serwisowego odbywa się również poprzez wciśnięcie przycisku „C”.

W trybie serwisowym akceptacja kodu przyciskiem „A” pozwala na dodanie kodu do puli akceptowalnych kodów, dodanie kodu odbywa się wraz z pokazaniem odpowiedniego komunikatu:



Rysunek 17 Dodanie nowego kodu

W przypadku próby dodania kodu serwisowego do puli akceptowalnych kodów dostajemy komunikat o błędzie:



Rysunek 18 Próba dodania kodu serwisowego

4. ALGORYTMY I FRAGMENTY KODU:

W projekcie użyto:

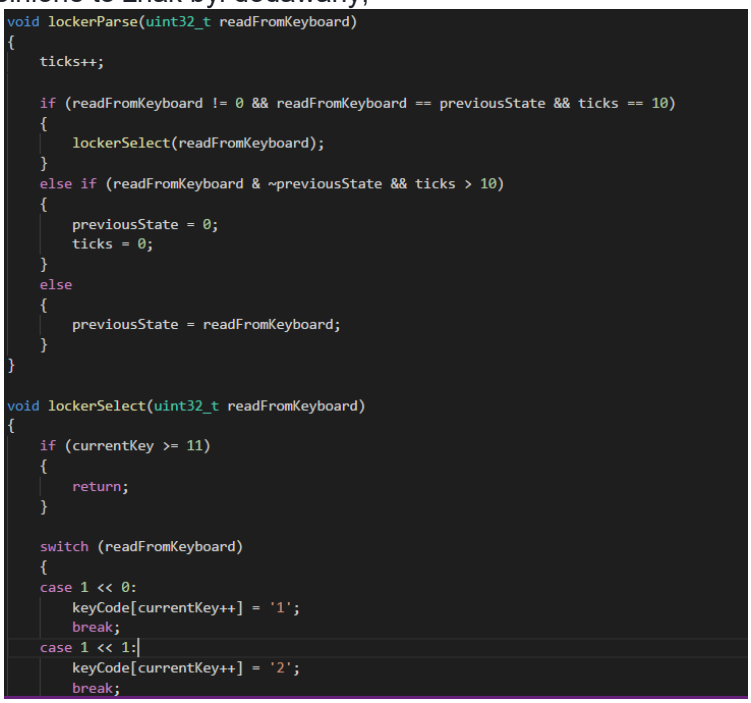
- SysTick:
  - Używany w opóźnieniu pomiędzy odczytem a parsowaniem odczytanego kodu.
  - Użyte opóźnienie 1 milisekunda,
- Timer0:
  - Użyty do wszelkich operacji w aplikacji, między innymi, otwieranie/zamykanie drzwi, wyświetlanie komunikatów, zliczanie czasu od uruchomienia mikrokontrolera.
  - Przerwanie Timera 0 generowane jest co sekundę,
- A także algorytmy takie jak:
  - Algorytm odczytu kodu z klawiatury
    - Podczas inicjalizacji konfigurowane są odpowiednie piny mikrokontrolera, wiersze ustawiane są na wyjście, a kolumny na wejście (jest to typowa konfiguracja typu pulldown).,
    - Wczytywanie polega na iteracji po wszystkich wierszach klawiatury macierzowej i ustawieniu obecnego wierszu na tryb zapisu i odczytu kolumny wciśniętego klawisza razem bitowym przesunięciem w lewo do zmiennej typu uint32_t,

```
uint16_t lockerRead()
{
    uint32_t readFromKeyboard = 0x0;

    for (int i = 3; i >= 0; --i)
    {
        setRowToRead(i);
        readFromKeyboard <<= 4;
        readFromKeyboard |= readColumn();
    }

    return readFromKeyboard;
}

void setRowToRead(int row)
{
    switch (row)
    {
    case 0:
        GPIO_PinWrite(1, 0, 1);
        GPIO_PinWrite(1, 10, 0);
        GPIO_PinWrite(1, 8, 0);
        GPIO_PinWrite(1, 16, 0);
        break;
    case 1:
        GPIO_PinWrite(1, 0, 0);
        GPIO_PinWrite(1, 10, 1);
        GPIO_PinWrite(1, 8, 0);
        GPIO_PinWrite(1, 16, 0);
        break;
    case 2:
        GPIO_PinWrite(1, 0, 0);
        GPIO_PinWrite(1, 10, 0);
```

Rysunek 19

Rysunek 20 Algorytm odczytu z klawiatury macierzowej, w uciętej części znajduje się reszta warunków case która wygląda analogicznie jak case 0 oraz 1

- Algorytm parsowania odczytanych danych z klawiatury:
  - Algorytm ten polega na sprawdzaniu ustawień bitów w zmiennej typu int32_t oraz dodawaniu odpowiadających im znakom do obecnie wpisywanego kodu,
  - Jednocześnie odbywa się sprawdzanie czy stan klawiatury nie zmieniał się przez 10 ticków zegara systemowego, a także czy nie zostało wciśniętych parę przycisków naraz, jeżeli warunki zostały spełnione to znak był dodawany,



```
void lockerParse(uint32_t readFromKeyboard)
{
    ticks++;

    if (readFromKeyboard != 0 && readFromKeyboard == previousState && ticks == 10)
    {
        lockerSelect(readFromKeyboard);
    }
    else if (readFromKeyboard & ~previousState && ticks > 10)
    {
        previousState = 0;
        ticks = 0;
    }
    else
    {
        previousState = readFromKeyboard;
    }
}

void lockerSelect(uint32_t readFromKeyboard)
{
    if (currentKey >= 11)
    {
        return;
    }

    switch (readFromKeyboard)
    {
    case 1 << 0:
        keyCode[currentKey++] = '1';
        break;
    case 1 << 1:
        keyCode[currentKey++] = '2';
        break;
```

Rysunek 21 Algorytm parsowania wraz z sprawdzaniem, w uciętej części znajduje się reszta switch'a która wygląda podobnie jak przypadek 1

- Algorytm wypisywania tekstu na ekran
  - Algorytm ten konstruowaliśmy podczas zajęć, w pętli wypisywane są pojedyncze litery, dla których musimy pobrać kod ASCII a następnie przy użyciu lcdWriteReq wypisane na ekran.
- Algorytm listy/tablicy cyklicznej przesuwnej
  - W przypadku większej niż 8 ilości otwarć następuje usunięcie najstarszego czasu, przesunięcia w wstecz kolejnych rekordów i dodanie najnowszego do tablicy historii.

```
void addUnlockToHistory()
{
    if (currentUnlocks >= 8)
    {
        shiftValuesInHistoryArray();
    }

    char unlockSeconds[255];
    sprintf(unlockSeconds, "%d", unlockSecondsMissed);
    strcpy(historyOfUnlocks[currentUnlocks], unlockSeconds);

    if (currentUnlocks < 8)
    {
        currentUnlocks++;
    }
}

void shiftValuesInHistoryArray()
{
    for (int i = 0; i < currentUnlocks; ++i)
    {
        strcpy(historyOfUnlocks[i], historyOfUnlocks[i + 1]);
    }
}
```

Rysunek 22 Algorytm tablicy cyklicznej przesuwnej

- Algorytmy walidacji sprawdzające między innymi poprawność wpisywanego kodu, czy wpisany kod jest kodem serwisowym i czy nie przekracza kod zadanej długości znaków.
- Algorytmy obsługi zdarzeń wywoływanych przez użytkownika posługującego się klawiaturą.

5. KONTROLA BŁĘDÓW:

W projekcie dodano obsługę błędów między innymi takich jak:

- Próba dodania kodu serwisowego do puli kodów pozwalających otworzenie drzwi. – W trybie serwisowym program podczas akceptacji sprawdza czy nie został wpisany kod serwisowy (na przykład w skutek nieumyślności serwisanta). Próba kończy się błędem dodania.
- Wciśnięcie paru klawiszów jednocześnie. Obsługa błędu zaprezentowana na rysunku 20.
- Zbyt szybkie wpisywanie kodów (na przypadek prób użycia ataku brute force na klawiaturze powoduje to dłuższy czas odnalezienia poprawnego szyfru). Obsługa błędu zaprezentowana na rysunku 20.