Додаток А

(Обов'язковий)

ПРОГРАМНІ КОДИ

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <EEPROM.h>
LiquidCrystal_I2C _lcd1(0x27, 16, 2);
int _dispTempLength1=0;
boolean _isNeedClearDisp1;
const byte _menuParametrsArray[]  PROGMEM =  {1, 1, 2, 0, 0, 0, 2, 3, 1, 4, 2, 0, 2, 1, 1, 0, 0, 0, 2, 3, 1, 4, 1, 0};
unsigned long _menuValueArray_unsignedlong[2];
const unsigned long _menuConstantValuesArray_unsignedlong[]  PROGMEM =  {1, 1000000, 0};
const char _flprogMenuNameString1[] PROGMEM = "NEW PASSWORD";
const char _flprogMenuNameString2[] PROGMEM = "ENTER PASSWORD";
const char* const _flprogMenuStringsArray[] PROGMEM = { _flprogMenuNameString1, _flprogMenuNameString2};
struct _menuItemStricture
{
    int startInArrayIndex;
}
;
struct _menuMainStricture
{
   String tempString;
   int startIndex;
   int stopIndex;
   bool isSwitchMenuAroundRing;
   _menuItemStricture  currentItem;
}
;
_menuItemStricture _MenuItems[2];
_menuMainStricture _MainMenus[1];
bool _gtv1 = 0; //D
bool _gtv2 = 0; //C
bool _gtv3 = 0; //B
bool _gtv4 = 0; //A
bool _gtv5 = 0; //#
bool _gtv6 = 0; //9
bool _gtv7 = 0; //6
bool _gtv8 = 0; //3
bool _gtv9 = 0; //0
bool _gtv10 = 0; //8
bool _gtv11 = 0; //5
bool _gtv12 = 0; //2
bool _gtv13 = 0; //*
bool _gtv14 = 0; //7
bool _gtv15 = 0; //4
bool _gtv16 = 0; //1
bool _gtv17 = 0;
bool _gtv18 = 0;
bool _gtv19 = 0;
bool _gtv20 = 0;
bool _pzs4OES = 0;
bool FTrig_1_Out = 0;
bool FTrig_1_OldStat = 0;
```

```
bool _pzs11OES = 0;
int _disp2oldLength = 0;
bool _changeNumber1_Out = 0;
int _changeNumber1_OLV;
bool _tim2I = 0;
bool _tim2O = 0;
unsigned long _tim2P = 0UL;
bool _pzs5OES = 0;
bool FTrig_3_Out = 0;
bool FTrig_3_OldStat = 0;
bool _pzs12OES = 0;
bool _trgrt2 = 0;
bool _trgrt2I = 0;
int _convertStringToNamberOutput_3 = 0;
bool _pzs6OES = 0;
bool _gen1I = 0;
bool _gen1O = 0;
unsigned long _gen1P = 0UL;
bool _tim5I = 0;
bool _tim5O = 0;
unsigned long _tim5P = 0UL;
bool _tim3I = 0;
bool _tim3O = 0;
unsigned long _tim3P = 0UL;
bool _mkb1C1xP1 = 0;
bool _mkb1C1xP2 = 0;
bool _mkb1C1xP3 = 0;
bool _mkb1C1xP4 = 0;
bool _mkb1C2xP1 = 0;
bool _mkb1C2xP2 = 0;
bool _mkb1C2xP3 = 0;
bool _mkb1C2xP4 = 0;
bool _mkb1C3xP1 = 0;
bool _mkb1C3xP2 = 0;
bool _mkb1C3xP3 = 0;
bool _mkb1C3xP4 = 0;
bool _mkb1C4xP1 = 0;
bool _mkb1C4xP2 = 0;
bool _mkb1C4xP3 = 0;
bool _mkb1C4xP4 = 0;
bool _tim4I = 0;
bool _tim4O = 0;
unsigned long _tim4P = 0UL;
bool _pzs7OES = 0;
bool _pzs1OES = 0;
bool FTrig_2_Out = 0;
bool FTrig_2_OldStat = 0;
bool _pzs8OES = 0;
bool _pzs2OES = 0;
bool _trgrt1 = 0;
bool _trgrt1I = 0;
bool _pzs9OES = 0;
bool _tim1I = 0;
```

```
bool _tim1O = 0;
unsigned long _tim1P = 0UL;
bool _pzs3OES = 0;
bool _pzs10OES = 0;
int _disp1oldLength = 0;
bool _MenuBlock_67075352_AMO_234729918 = 0;
String _MenuBlock_67075352_MNO;
String _MenuBlock_67075352_VNO;
bool _MenuBlock_67075352_OEIS = 0;
bool _MenuBlock_67075352_ORIS = 0;
bool _MenuBlock_67075352_IDI_0 = 0;
bool _MenuBlock_67075352_IDI_1 = 0;
bool _MenuBlock_67075352_IDI_2 = 0;
bool _MenuBlock_67075352_IDI_3 = 0;
bool _MenuBlock_67075352_IDI_4 = 0;
bool _MenuBlock_67075352_IDI_5 = 0;
bool _MenuBlock_67075352_IDI_6 = 0;
bool _MenuBlock_67075352_IDI_7 = 0;
bool _MenuBlock_67075352_IDI_8 = 0;
bool _MenuBlock_67075352_IDI_9 = 0;
bool _MenuBlock_67075352_BackspaceIOS = 0;
bool _tempVariable_bool;
String _tempVariable_String;
int _tempVariable_int;
void setup()

{
   pinMode(13, OUTPUT);
   digitalWrite(13, 0);
   Wire.begin();
   delay(10);
   if(((readByteFromEEPROM(0, 0, 0x0))) != 117)
   {
      (updateByteToEEPROM(0, 0, 0x0, (117)));
      (updateUnsignedLongToEEPROM(1, 0, 0x0, (0)));

   }

   _lcd1.init();
   _lcd1.backlight();
   pinMode(8, INPUT_PULLUP);
   pinMode(9, INPUT_PULLUP);
   pinMode(10, INPUT_PULLUP);
   pinMode(11, INPUT_PULLUP);
   pinMode(2, OUTPUT);
   digitalWrite(2, HIGH);
   pinMode(3, OUTPUT);
   digitalWrite(3, HIGH);
   pinMode(4, OUTPUT);
   digitalWrite(4, HIGH);
   pinMode(5, OUTPUT);
   digitalWrite(5, HIGH);
   _MenuItems[0].startInArrayIndex = 0;
```

```
    _MenuItems[1].startInArrayIndex = 12;
    _MainMenus[0].tempString = "";
    _MainMenus[0].startIndex = 1;
    _MainMenus[0].isSwitchMenuAroundRing = 1;
    _MainMenus[0].stopIndex = 2;
    _MainMenus[0].currentItem = _MenuItems[0];
    _menuValueArray_unsignedlong[0] = (readUnsignedLongFromEEPROM(1, 0, 0x0));

}

void loop()

{
    if (_isNeedClearDisp1)
    {
        _lcd1.clear();
        _isNeedClearDisp1= 0;
    }

    //Плата:1
    if(_gtv19)
    {
        _tim5O = 1;
        _tim5I = 1;
    }
     else
    {
         if(_tim5I)
        {
            _tim5I = 0;
            _tim5P = millis();
        }
         else
        {
             if (_tim5O)
            {
                if (_isTimer(_tim5P, 3000)) _tim5O = 0;
            }

        }

    }

    FTrig_3_Out = 0;
    if ((!(_tim5O))&&(FTrig_3_OldStat))
    {
        FTrig_3_Out = 1;
    }

    FTrig_3_OldStat = _tim5O;
    if(((_gtv17) || (_gtv18) || (_gtv20)))
    {
        _tim2O = 1;
```

```
    _tim2I = 1;
}
 else
{
     if(_tim2I)
    {
        _tim2I = 0;
        _tim2P = millis();
    }
     else
    {
         if (_tim2O)
        {
            if (_isTimer(_tim2P, 1500)) _tim2O = 0;
        }

    }

}

if (_tim2O)
{
     if (! _gen1I)
    {
        _gen1I = 1;
        _gen1O = 1;
        _gen1P = millis();
    }

}
 else
{
    _gen1I = 0 ;
    _gen1O= 0;
}

if (_gen1I)
{
      if (_isTimer (_gen1P , 1))
    {
         _gen1P = millis();
        _gen1O = ! _gen1O;
    }

}

if (_gtv17)
{
     if (_trgrt1I)
    {
         _trgrt1 = 0;
    }
     else
```

```
    {
       _trgrt1 = 1;
       _trgrt1I = 1;
    }

}
 else
{
    _trgrt1 = 0;
    _trgrt1I = 0;
}
;
if(_trgrt1)
{
    _tim1O = 1;
    _tim1I = 1;
}
 else
{
     if(_tim1I)
    {
       _tim1I = 0;
       _tim1P = millis();
    }
     else
    {
        if (_tim1O)
       {
          if (_isTimer(_tim1P, 1500)) _tim1O = 0;
       }

    }

}

FTrig_1_Out = 0;
if ((!(_tim1O))&&(FTrig_1_OldStat))
{
    FTrig_1_Out = 1;
}

FTrig_1_OldStat = _tim1O;
if (!(0))
{
    _tempVariable_bool  =  1;
    if (! _MenuBlock_67075352_OEIS)
    {
       _MenuBlock_67075352_OEIS = 1;
       (_MainMenus[0]).tempString = "";

    }
```

```
      _tempVariable_int                                                    =
pgm_read_byte(&_menuParametrsArray[((_MainMenus[0].currentItem).startInArrayIndex)+10]);
      _MenuBlock_67075352_MNO          =          _readStringFromProgmem
((char*)pgm_read_word(&(_flprogMenuStringsArray[_tempVariable_int - 1])));
      _MenuBlock_67075352_VNO = _menuOutputValueString (0);
      _MenuBlock_67075352_AMO_234729918                                    =
pgm_read_byte(&_menuParametrsArray[(_MainMenus[0].currentItem).startInArrayIndex]) == 2;

   }
    else
   {
      _tempVariable_bool  =  0;
      if (_MenuBlock_67075352_OEIS)
      {
         _MenuBlock_67075352_OEIS = 0;
         _menuUpdateToEEpromItems();

      }

      _MenuBlock_67075352_AMO_234729918 = 0;
      _MenuBlock_67075352_MNO = "";
      _MenuBlock_67075352_VNO = "";

   }

   if(FTrig_3_Out)
   {
       if (!_MenuBlock_67075352_ORIS)
      {
         _MenuBlock_67075352_ORIS = 1;
         if(_tempVariable_bool)
         {
            _MainMenus[0].currentItem = _MenuItems[0];
            _menuUpdateToEEpromItems();
            (_MainMenus[0]).tempString = "";

         }

      }

   }
    else
   {
      _MenuBlock_67075352_ORIS = 0;
   }

   if(FTrig_1_Out)
   {
      if (_tempVariable_bool)
      {
         (_MainMenus[0]).currentItem = _MenuItems[1];
         _menuUpdateToEEpromItems();
         (_MainMenus[0]).tempString = "";
```

```
    }

}

if(_gtv9)
{
    if (! _MenuBlock_67075352_IDI_0)
    {
        _MenuBlock_67075352_IDI_0 = 1;
        if(_tempVariable_bool)
        {
            _menuDirectInputKeyPressEvents(0, '0');
        }

    }

}
 else
{
    _MenuBlock_67075352_IDI_0 = 0;
}

if(_gtv16)
{
    if (! _MenuBlock_67075352_IDI_1)
    {
        _MenuBlock_67075352_IDI_1 = 1;
        if(_tempVariable_bool)
        {
            _menuDirectInputKeyPressEvents(0, '1');
        }

    }

}
 else
{
    _MenuBlock_67075352_IDI_1 = 0;
}

if(_gtv12)
{
    if (! _MenuBlock_67075352_IDI_2)
    {
        _MenuBlock_67075352_IDI_2 = 1;
        if(_tempVariable_bool)
        {
            _menuDirectInputKeyPressEvents(0, '2');
        }

    }
```

```
}
 else
{
   _MenuBlock_67075352_IDI_2 = 0;
}

if(_gtv8)
{
    if (! _MenuBlock_67075352_IDI_3)
   {
      _MenuBlock_67075352_IDI_3 = 1;
      if(_tempVariable_bool)
      {
          _menuDirectInputKeyPressEvents(0, '3');
      }

   }

}
 else
{
   _MenuBlock_67075352_IDI_3 = 0;
}

if(_gtv15)
{
    if (! _MenuBlock_67075352_IDI_4)
   {
      _MenuBlock_67075352_IDI_4 = 1;
      if(_tempVariable_bool)
      {
          _menuDirectInputKeyPressEvents(0, '4');
      }

   }

}
 else
{
   _MenuBlock_67075352_IDI_4 = 0;
}

if(_gtv11)
{
    if (! _MenuBlock_67075352_IDI_5)
   {
      _MenuBlock_67075352_IDI_5 = 1;
      if(_tempVariable_bool)
      {
          _menuDirectInputKeyPressEvents(0, '5');
      }

   }
```

```
}
 else
{
   _MenuBlock_67075352_IDI_5 = 0;
}

if(_gtv7)
{
    if (! _MenuBlock_67075352_IDI_6)
   {
      _MenuBlock_67075352_IDI_6 = 1;
      if(_tempVariable_bool)
      {
          _menuDirectInputKeyPressEvents(0, '6');
      }

   }

}
 else
{
   _MenuBlock_67075352_IDI_6 = 0;
}

if(_gtv14)
{
    if (! _MenuBlock_67075352_IDI_7)
   {
      _MenuBlock_67075352_IDI_7 = 1;
      if(_tempVariable_bool)
      {
          _menuDirectInputKeyPressEvents(0, '7');
      }

   }

}
 else
{
   _MenuBlock_67075352_IDI_7 = 0;
}

if(_gtv10)
{
    if (! _MenuBlock_67075352_IDI_8)
   {
      _MenuBlock_67075352_IDI_8 = 1;
      if(_tempVariable_bool)
      {
          _menuDirectInputKeyPressEvents(0, '8');
      }
```

```
    }

}
 else
{
    _MenuBlock_67075352_IDI_8 = 0;
}

if(_gtv6)
{
     if (! _MenuBlock_67075352_IDI_9)
    {
        _MenuBlock_67075352_IDI_9 = 1;
        if(_tempVariable_bool)
        {
            _menuDirectInputKeyPressEvents(0, '9');
        }

    }

}
 else
{
    _MenuBlock_67075352_IDI_9 = 0;
}

if(_gen1O)
{
     if (! _MenuBlock_67075352_BackspaceIOS)
    {
        _MenuBlock_67075352_BackspaceIOS = 1;
        if(_tempVariable_bool)
        {
            _menuDirectInputKeyPressEvents(0, '<');
        }

    }

}
 else
{
    _MenuBlock_67075352_BackspaceIOS = 0;
}

_tempVariable_String = _MenuBlock_67075352_VNO;
_convertStringToNamberOutput_3 = strtol(_tempVariable_String.c_str(),NULL,10);
if (_changeNumber1_Out)
{
    _changeNumber1_Out = 0;
}
 else
{
    _tempVariable_int = _convertStringToNamberOutput_3;
```

```
    if (_tempVariable_int != _changeNumber1_OLV)
    {
       _changeNumber1_OLV = _tempVariable_int;
       _changeNumber1_Out = 1;
    }


}

_gtv19 = ((_changeNumber1_Out) && (_MenuBlock_67075352_AMO_234729918));
if(_changeNumber1_Out)
{
   _tim4O = 1;
   _tim4I = 1;
}
 else
{
    if(_tim4I)
   {
       _tim4I = 0;
       _tim4P = millis();
   }
    else
   {
        if (_tim4O)
       {
          if (_isTimer(_tim4P, 6000)) _tim4O = 0;
       }

   }

}

FTrig_2_Out = 0;
if ((!(_tim4O))&&(FTrig_2_OldStat))
{
   FTrig_2_Out = 1;
}

FTrig_2_OldStat = _tim4O;
_gtv20 = FTrig_2_Out;
if(((_MenuBlock_67075352_VNO).equals(String("256000"))))

{
   if((!_pzs2OES))
   {
       tone(6, 1330, 180);
       _pzs2OES =1;
   }

}
 else
{
```

```
    if(_pzs2OES)
    {
       noTone(6);
       _pzs2OES =0;
    }

}

_gtv17 = ((_MenuBlock_67075352_VNO).equals(String("256000")));
if (!(0))
{
    _dispTempLength1 = ((_MenuBlock_67075352_VNO)).length();
    if (_disp2oldLength > _dispTempLength1)
    {
       _isNeedClearDisp1 = 1;
    }

    _disp2oldLength = _dispTempLength1;
    _lcd1.setCursor(int((16 - _dispTempLength1)/2), 1);
    _lcd1.print((_MenuBlock_67075352_VNO));

}
 else
{
    if (_disp2oldLength > 0)
    {
       _isNeedClearDisp1 = 1;
       _disp2oldLength = 0;
    }

}

if (!(0))
{
    _dispTempLength1 = ((_MenuBlock_67075352_MNO)).length();
    if (_disp1oldLength > _dispTempLength1)
    {
       _isNeedClearDisp1 = 1;
    }

    _disp1oldLength = _dispTempLength1;
    _lcd1.setCursor(int((16 - _dispTempLength1)/2), 0);
    _lcd1.print((_MenuBlock_67075352_MNO));

}
 else
{
    if (_disp1oldLength > 0)
    {
       _isNeedClearDisp1 = 1;
       _disp1oldLength = 0;
    }
```

```
}

digitalWrite(2, 0);
_mkb1C1xP1 = ! (digitalRead(8));
_mkb1C1xP2 = ! (digitalRead(9));
_mkb1C1xP3 = ! (digitalRead(10));
_mkb1C1xP4 = ! (digitalRead(11));
digitalWrite(2, 1);
digitalWrite(3, 0);
_mkb1C2xP1 = ! (digitalRead(8));
_mkb1C2xP2 = ! (digitalRead(9));
_mkb1C2xP3 = ! (digitalRead(10));
_mkb1C2xP4 = ! (digitalRead(11));
digitalWrite(3, 1);
digitalWrite(4, 0);
_mkb1C3xP1 = ! (digitalRead(8));
_mkb1C3xP2 = ! (digitalRead(9));
_mkb1C3xP3 = ! (digitalRead(10));
_mkb1C3xP4 = ! (digitalRead(11));
digitalWrite(4, 1);
digitalWrite(5, 0);
_mkb1C4xP1 = ! (digitalRead(8));
_mkb1C4xP2 = ! (digitalRead(9));
_mkb1C4xP3 = ! (digitalRead(10));
_mkb1C4xP4 = ! (digitalRead(11));
digitalWrite(5, 1);
_gtv16 = _mkb1C4xP4;
if(_mkb1C4xP4)

{
   if((!_pzs3OES))
   {
       tone(6, 200, 50);
      _pzs3OES =1;
   }

}
 else
{
   if(_pzs3OES)
   {
      noTone(6);
      _pzs3OES =0;
   }

}

_gtv15 = _mkb1C4xP3;
if(_mkb1C4xP3)

{
   if((!_pzs4OES))
   {
```

```
        tone(6, 200, 50);
       _pzs4OES =1;
    }

}
 else
{
    if(_pzs4OES)
    {
       noTone(6);
       _pzs4OES =0;
    }

}

_gtv14 = _mkb1C4xP2;
if(_mkb1C4xP2)

{
    if((!_pzs5OES))
    {
        tone(6, 200, 50);
       _pzs5OES =1;
    }

}
 else
{
    if(_pzs5OES)
    {
       noTone(6);
       _pzs5OES =0;
    }

}

_gtv13 = _mkb1C4xP1;
_gtv12 = _mkb1C3xP4;
if(_mkb1C3xP4)

{
    if((!_pzs6OES))
    {
        tone(6, 200, 50);
       _pzs6OES =1;
    }

}
 else
{
    if(_pzs6OES)
    {
       noTone(6);
```

```
        _pzs6OES =0;
    }

}

_gtv11 = _mkb1C3xP3;
if(_mkb1C3xP3)

{
    if((!_pzs7OES))
    {
         tone(6, 200, 50);
        _pzs7OES =1;
    }

}
 else
{
    if(_pzs7OES)
    {
        noTone(6);
        _pzs7OES =0;
    }

}

_gtv10 = _mkb1C3xP2;
if(_mkb1C3xP2)

{
    if((!_pzs8OES))
    {
         tone(6, 200, 50);
        _pzs8OES =1;
    }

}
 else
{
    if(_pzs8OES)
    {
        noTone(6);
        _pzs8OES =0;
    }

}

_gtv9 = _mkb1C3xP1;
if(_mkb1C3xP1)

{
    if((!_pzs9OES))
    {
```

```
        tone(6, 200, 50);
       _pzs9OES =1;
    }

}
 else
{
    if(_pzs9OES)
    {
       noTone(6);
       _pzs9OES =0;
    }

}

_gtv8 = _mkb1C2xP4;
if(_mkb1C2xP4)

{
    if((!_pzs10OES))
    {
        tone(6, 200, 50);
       _pzs10OES =1;
    }

}
 else
{
    if(_pzs10OES)
    {
       noTone(6);
       _pzs10OES =0;
    }

}

_gtv7 = _mkb1C2xP3;
if(_mkb1C2xP3)

{
    if((!_pzs11OES))
    {
        tone(6, 200, 50);
       _pzs11OES =1;
    }

}
 else
{
    if(_pzs11OES)
    {
       noTone(6);
       _pzs11OES =0;
```

```
    }

  }

  _gtv6 = _mkb1C2xP2;
  if(_mkb1C2xP2)

  {
    if((!_pzs12OES))
    {
       tone(6, 200, 50);
      _pzs12OES =1;
    }

  }
   else
  {
    if(_pzs12OES)
    {
      noTone(6);
      _pzs12OES =0;
    }

  }

  _gtv5 = _mkb1C2xP1;
  _gtv4 = _mkb1C1xP4;
  _gtv3 = _mkb1C1xP3;
  _gtv2 = _mkb1C1xP2;
  _gtv1 = _mkb1C1xP1;
  if                                                    ((((String((_menuValueArray_unsignedlong[1]),
DEC))).equals((String((_menuValueArray_unsignedlong[0]), DEC)))))
  {
     if (_trgrt2I)
    {
       _trgrt2 = 0;
    }
     else
    {
      _trgrt2 = 1;
      _trgrt2I = 1;
    }

  }
   else
  {
    _trgrt2 = 0;
    _trgrt2I = 0;
  }
  ;
  _gtv18 = _trgrt2;
  if(_trgrt2)
  {
```

```
      _tim3O = 1;
      _tim3I = 1;
   }
    else
   {
       if(_tim3I)
      {
         _tim3I = 0;
         _tim3P = millis();
      }
       else
      {
          if (_tim3O)
         {
            if (_isTimer(_tim3P, 3000)) _tim3O = 0;
         }

      }

   }

   if(_tim3O)

   {
      if((!_pzs1OES))
      {
          tone(6, 130, 80);
         _pzs1OES =1;
      }

   }
    else
   {
      if(_pzs1OES)
      {
         noTone(6);
         _pzs1OES =0;
      }

   }

   digitalWrite(13, _tim3O);

}

unsigned long  _stringToUnsignedLong(String value)

{
   unsigned long result;
   if((value.length())<10)
   {
      result= value.toInt();
      return result;
```

```
    }

    result=(value.substring(0,9)).toInt();
    result=result*10;
    result=result+(value.substring(9,10)).toInt();
    return result;

}

bool _isTimer(unsigned long startTime, unsigned long period)

{
    unsigned long currentTime;
    currentTime = millis();
    if (currentTime>= startTime)
    {
        return (currentTime>=(startTime + period));
    }
     else
    {
        return (currentTime >=(4294967295-startTime+period));
    }



}

String _readStringFromProgmem (char *string)

{
    String result = String("");
    while (pgm_read_byte(string)!='\0')

    {
        result=result+ char(pgm_read_byte(string));
            string++;


    }

    return result;

}

String _menuOutputValueString (int menuIndex)

{
    int                itemType                =                pgm_read_byte(&_menuParametrsArray[
(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+1]);
```

```
   int valIndex = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+2]);
   int indexMin = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+7]);
   int indexMax = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+6]);
   if(valIndex == 0)
   {
      return "";
   }

   int convFormat = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+9]);
   if(itemType == 1)
   {
      return _convertNamber(itemType, convFormat, valIndex, indexMax, indexMin);
   }

   return "";

}

void _menuDirectInputKeyPressEvents(int menuIndex, char inputSymbol)

{
   int valIndex = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+2]);
   int temp;
   if(valIndex == 0)
   {
      return;
   }

   int itemType = pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIndex)+1]);
   if(itemType == 7)
   {
      return;
   }

   if(inputSymbol == '<')
   {
      temp = ((_MainMenus[menuIndex]).tempString).length();
      if(temp == 0)
      {
         return;
      }
```

```
        else

       {
          if(temp == 1)
          {
             (_MainMenus[menuIndex]).tempString ="";
          }
           else
          {
             (_MainMenus[menuIndex]).tempString                                    =
((_MainMenus[menuIndex]).tempString).substring(0, temp - 1);
          }

       }


    }
     else
    {
       (_MainMenus[menuIndex]).tempString  =  (_MainMenus[menuIndex]).tempString  +  in-
putSymbol;


    }

    if(itemType == 1)
    {
       _menuValueArray_unsignedlong[valIndex              -              1]              =
(_stringToUnsignedLong(((_MainMenus[menuIndex]).tempString)));
       temp                                                                          =
pgm_read_byte(&_menuParametrsArray[(((_MainMenus[menuIndex]).currentItem).startInArrayIn
dex)+6]);
       if (temp  != 0)
       {
          if          ((_menuValueArray_unsignedlong[valIndex          -          1])          >
(pgm_read_dword(&_menuConstantValuesArray_unsignedlong[temp - 1])))
          {
             _menuValueArray_unsignedlong[valIndex          -          1]          =
(pgm_read_dword(&_menuConstantValuesArray_unsignedlong[temp                          -
1]));(_MainMenus[menuIndex]).tempString = "";
          }

       }


    }


}

void _menuUpdateToEEpromItems()

{
```

```
   (updateUnsignedLongToEEPROM(1, 0, 0x0, ((_menuValueArray_unsignedlong[0]))));
}

String _convertNamber(int itemType, int convFormat, int valIndex, int indexMax, int indexMin)

{
   if (itemType== 1)
   {
      if (convFormat == 4)
      {
          return String((_menuValueArray_unsignedlong[valIndex - 1 ]),DEC);
      }

      if (convFormat == 5)
      {
          return String((_menuValueArray_unsignedlong[valIndex - 1]),HEX);
      }

      if (convFormat == 6)
      {
          return String((_menuValueArray_unsignedlong[valIndex -1]),BIN);
      }


   }

}

byte readByteFromEEPROM(int addres, byte bitAddres, byte chipAddres)

{
   return EEPROM.read(addres);
}

void updateByteToEEPROM(int addres, byte bitAddres, byte chipAddres, byte value)

{
   return EEPROM.update(addres, value);
}

unsigned long readUnsignedLongFromEEPROM(int addres, byte bitAddres, byte chipAddres)

{
   byte x[4];
   for(byte i = 0; i < 4; i++)
   {
       x[i] = readByteFromEEPROM((addres+i),  bitAddres,  chipAddres);
   }

   unsigned long *y = (unsigned long *)&x;
   return y[0];

}
```

```
void updateUnsignedLongToEEPROM(int addres, byte bitAddres, byte chipAddres, unsigned long value)

{
   byte *x = (byte *)&value;
   for(byte i = 0; i < 4; i++)
   {
      updateByteToEEPROM((addres+i),  bitAddres,  chipAddres, x[i]);
   }
```

Додаток Б

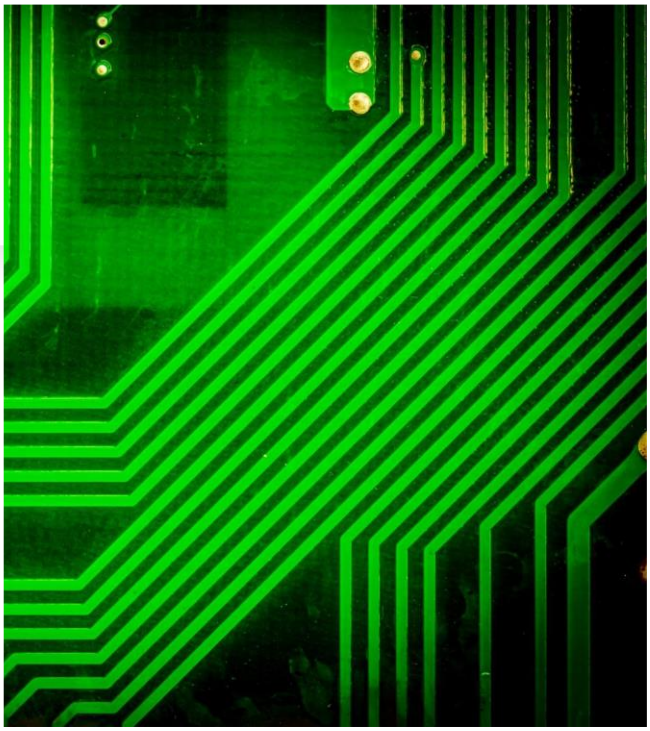
(Рекомендований)

КОПІЇ ПРЕЗЕНТАЦІЇ

Розробка кодового замку для допуску у приміщення закритих секцій

Вступ

- Необхідність розробки схеми електронного замка
  - Аналогічні пристрої використовують застарілу елементну базу
  - Обмеження на розширення або зміну функціоналу без апаратного доопрацювання
- Методики розробки на базі мікроконтролерів
  - Здешевлення процесу розробки
  - Обмеження на розробника у вигляді процесу програмування мікроконтролера

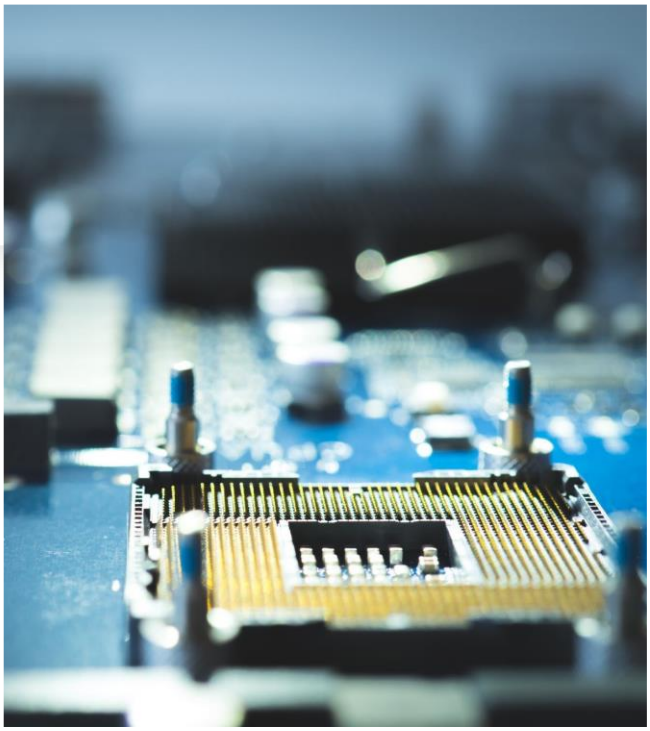


Формулювання основних технічних вимог до пристрою

- Електричні параметри
  - Напруга живлення 5В±10% через USB
  - Введення паролю з клавіатури
  - Зміна паролю з клавіатури
  - Індикація вводу паролю на дисплеї
  - Збереження паролю при відсутності живлення
  - Звуковий супровід набору паролю
- Умови роботи (кліматичні)
  - Температура навколишнього середовища від мінус 20 до +50°С
- Вимоги до конструкції
- Вимоги до надійності
- Інші вимоги

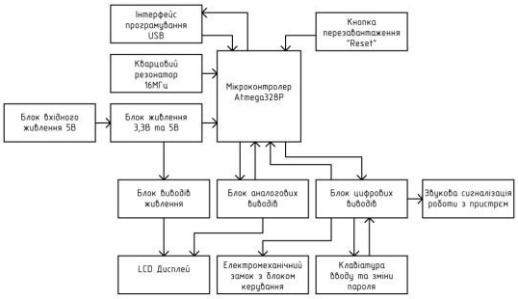
Аналіз технічного завдання

- Використання мікроконтролера для реалізації функціоналу
  - Обрано платформу Arduino
  - Використання плати Arduino nano
  - Можливість використання Arduino UNO
- Вибір програмного забезпечення
  - Обмеження на вибір програмного забезпечення для мікроконтролера
  - Використання програм конструкторів для пришвидшення розробки



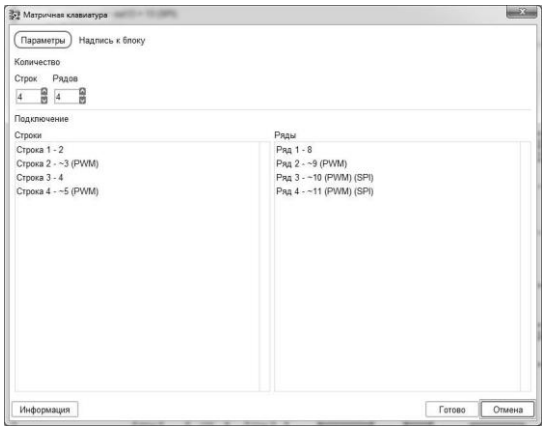
Опис структурної електричної схеми

- Програмна складова
  - Програмне забезпечення для роботи мікроконтролера
  - Використовується комп'ютер з програмним забезпеченням FLProg та Arduino IDE
- Апаратна складова
  - Інтерфейс програмування мікроконтролера
  - Мікроконтролер Atmega328P
  - Кварцовий резонатор на 16МГц
  - Кнопка перезавантаження

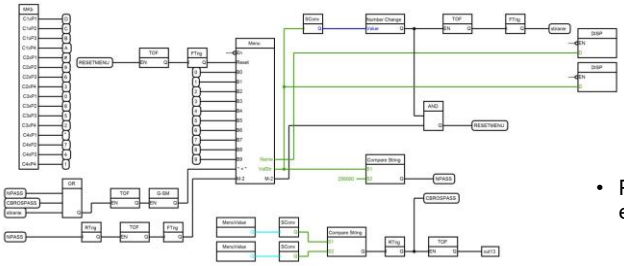


Аналіз принципу дії та призначення

- Основою пристрою є Arduino nano
  - Програмування мікроконтролера здійснюється за допомогою програми візуального програмування FLProg
- Програма для мікроконтролера складається шляхом з'єднання блоків з бібліотеки компонентів
  - Для вводу паролю буде використовуватись матрична клавіатура
- Основний блок меню містить два пункти меню
  - ENTER PASSWORD
  - NEW PASSWORD



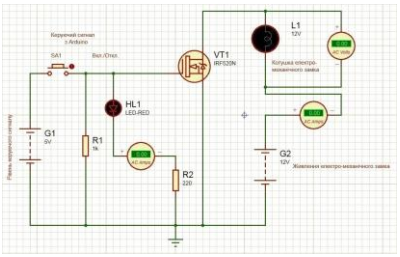
Аналіз принципу дії та призначення



- Реалізовано безпечне користування та зміну паролю
  - Схема програми зображена на рисунку 1.15
- Звукове супроводження функцій
  - Натискання цифр на клавіатурі - один сигнал
  - Вхід в режим адміністрування паролю – другий сигнал
  - Відкриття замка – третій сигнал
  - При відсутності паролю в пам'яті пристрою- сигнал аналогічний першому
- Реалізація звукового супроводження на елементах Buzzer
  - Можливість зміни тональності звучання кожної окремої кнопки чи пункту меню
  - Схема програми з елементами Buzzer зображена на рисунку 1.16

Моделювання роботи функціонального вузла

- Моделювання буде проводитись в програмі Proteus 8
  - Вузлом для моделювання буде виступати вихідний ключ керування навантаженням
  - На схему було додано вимірювальні прилади
  - Для задання керуючого сигналу на схемі встановлено елемент живлення G1
  - Кнопка SA1 дає змогу змоделювати два положення схеми
  - Резистор R1 задає зміщення на затвор транзистора VT1
  - Ланцюг з HL1 та R2 є індикатором наявності керуючого сигналу з Arduino



Обґрунтування вибору конструктивних матеріалів покриття

- Місце використання
  - Кодовий замок буде використовуватись в спеціально відведеному місці в нерухомій частині дверей.
- Вибір матеріалу для корпусу
  - Пластмаса як матеріал для корпусу.
  - Зменшення вартості виробництва.
  - Немає необхідності у формуванні отворів.
- Захист корпусу
  - Листовий метал як матеріал для захисту корпусу.
  - Метал ґрунтується та покривається спеціальною фарбою.
  - Монолітність конструкції.
  - Міцність та товщина металу залежить від місця використання.
- Захист для дерев'яних дверей

Опис роботи схеми

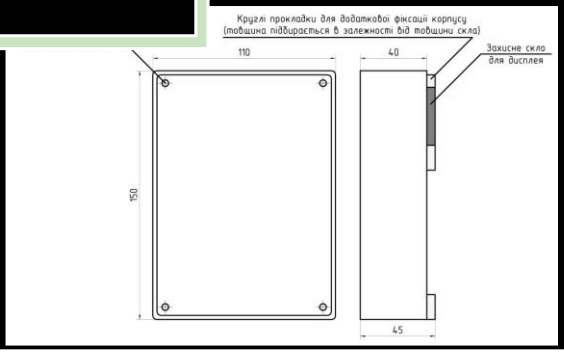
- При подачі напруги живлення на пристрій, відбувається відпрацювання програми
  - Програма була заздалегідь завантажена в пам'ять мікроконтролера
- В результаті виводиться меню на LCD дисплей
  - Можна вводити пароль доступу
- При вводі вірного паролю на вихідне навантаження буде подано керуючий імпульс
  - Імпульс на відкриття електромеханічного замка
- Для зміни паролю передбачено пароль адміністратора
  - При вводі адміністративного паролю на LCD дисплей виводиться меню зміни паролю доступу
- При зміні паролю доступу є особливості
  - Пароль заноситься в енергонезалежну пам'ять EEPROM
  - Це дає можливість використання без необхідності постійно ввімкненого джерела живлення

Концепція побудови пристрою

- Обмеження несанкціонованого доступу до внутрішніх вузлів та блоків пристрою
  - Використання клавіатури з антивандальними властивостями
  - Запобігання ушкодження LCD дисплею шляхом встановлення/приклеювання захисного скла
- Вбудовування корпусу пристрою в нерухому частину дверей

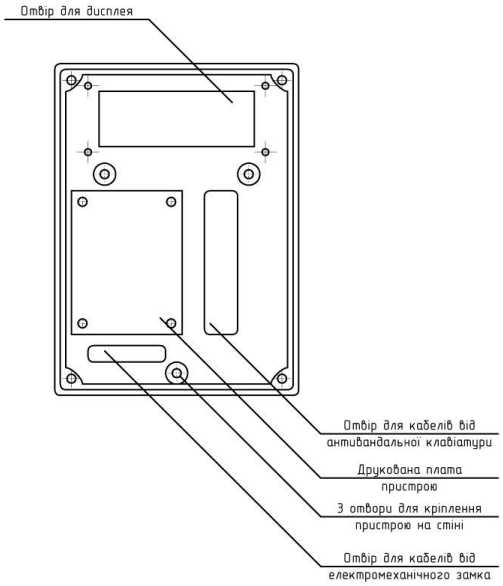
Зовнішня компоновка пристрою

- Користувачі мають доступ лише до зовнішніх органів керування
  - Матрична клавіатура (ATIS AM601W)
  - Електромеханічний замок (ATIS lock SS CK)
- Замок залишається закритим при відсутності живлення
  - Можливість отримати доступ за допомогою звичайного ключа
- Конструкція пристрою розроблена для вбудовування
  - Запобігання несанкціонованого доступу
- На задній кришці пристрою є лише 4 отвори для кріплення
- На бокових сторонах відсутні будь-які отвори та органи керування
- На передній кришці передбачені необхідні отвори для виведення кабелів
  - Антивандальна клавіатура
- Можливість встановлення/приклеювання захисного скла для LCD дисплея



Внутрішня компоновка пристрою

Розгляд внутрішнього компонування пристрою на двох рисунках



Висновок

Розробка електронного замка на базі мікроконтролера Arduino Nano дозволяє створити сучасний, надійний та функціональний пристрій. Завдяки використанню програмованої платформи забезпечується можливість легкої модифікації функціоналу без значних фінансових витрат.

Застосування візуальних середовищ програмування дозволяє знизити вимоги до кваліфікації розробників, що робить цей підхід економічно вигідним та доступним для широкого кола фахівців. Моделювання в середовищі Proteus 8 забезпечує високу точність і надійність при тестуванні розроблених схем.

Проект електронного замка відповідає сучасним вимогам до безпеки та надійності, що робить його перспективним для широкого застосування в системах доступу.

Додаток В

(Обов’язковий)

ВІДОМОСТІ АТЕСТАЦІЙНОГО ПРОЕКТУ

№ п/п	Позначення	Найменування	Додаткові відомості
		Текстові документи	
		Пояснювальна записка	49 с.

Зм	Арк.	№ докум.	Подп.	Дата	ГЮІК.467957.000Д4		
Розроб.		Неводничий			Розробка кодового замку для допуску у приміщення закритих секцій Відомість атестаційного проекту	Літера	Аркуш	Аркушів
Перевір.		Мерзлікін				У	1	1
Н. контр.		Бітченко				ХНУРЕ Кафедра РТІКС		
Затв.		Зарудний						