ESZTERHÁZY KÁROLY EGYETEM
TERMÉSZETTUDOMÁNYI KAR

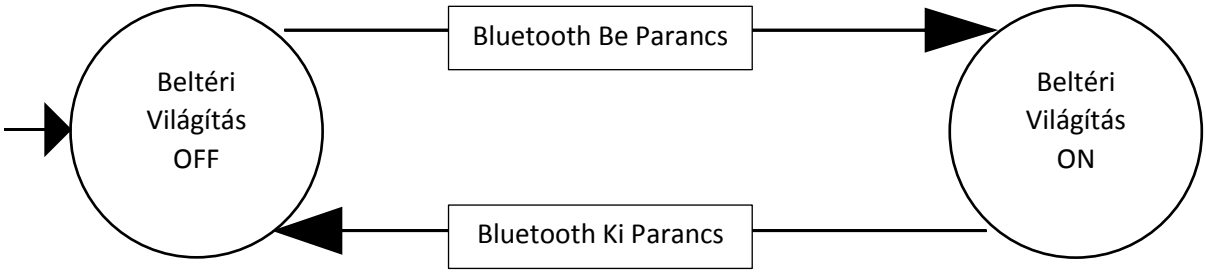
Robotika alapjai c. tárgy beadandó feladatának dokumentációja (Okosház vezérlés)

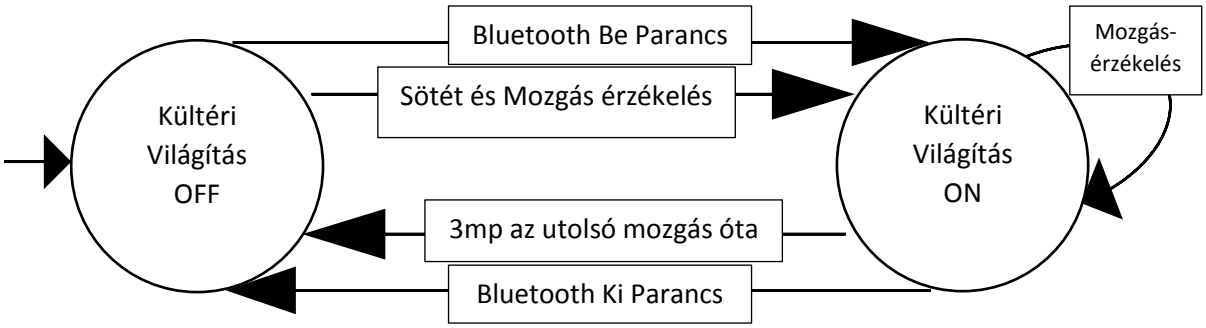
Készítette:

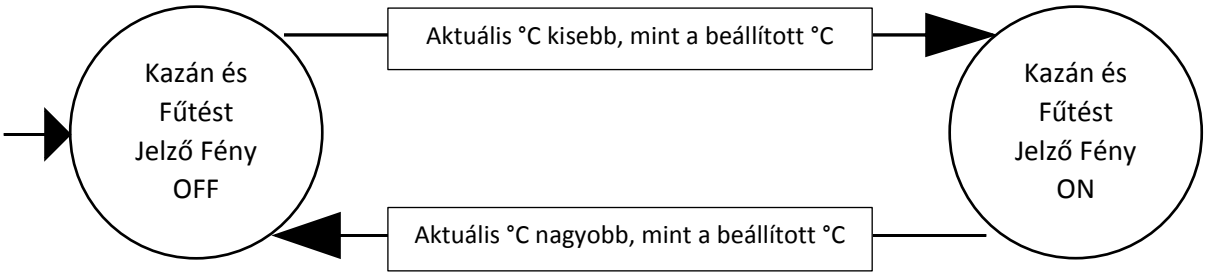
Dobozi András
Eged Zsolt
Balázs Gergő
Bíró Patrik

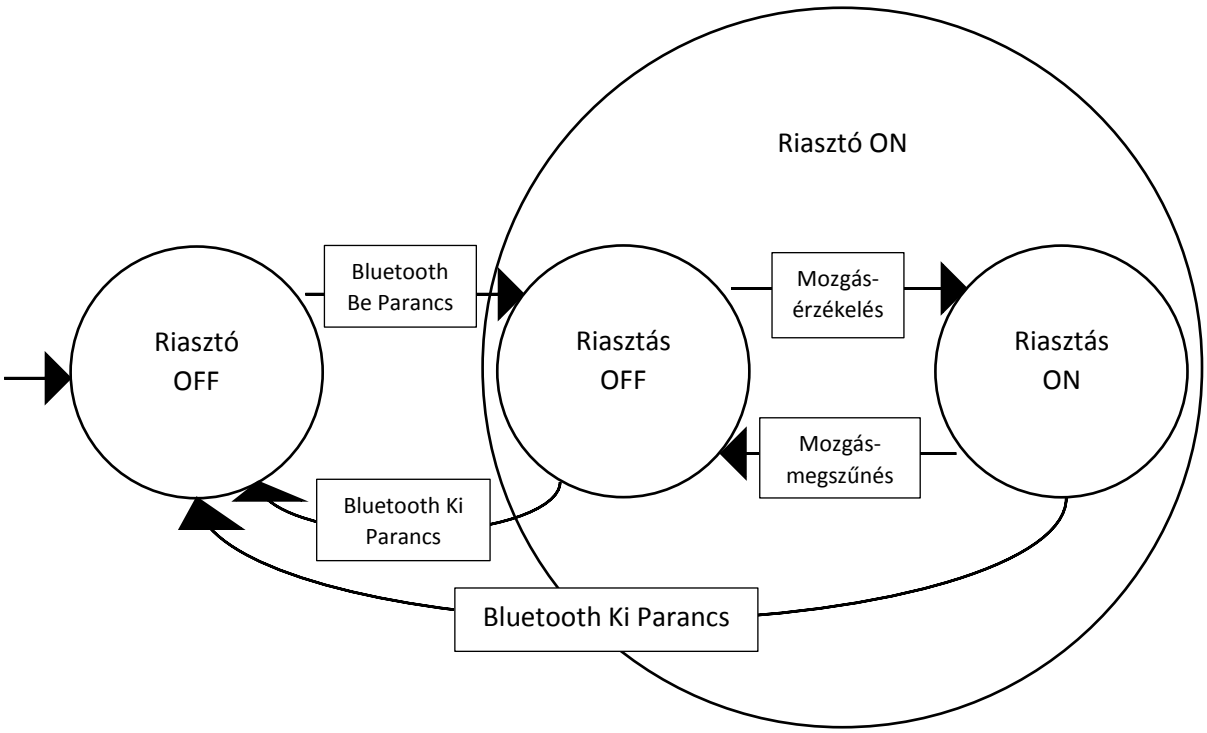
Eger, 2017

Állapotgép:

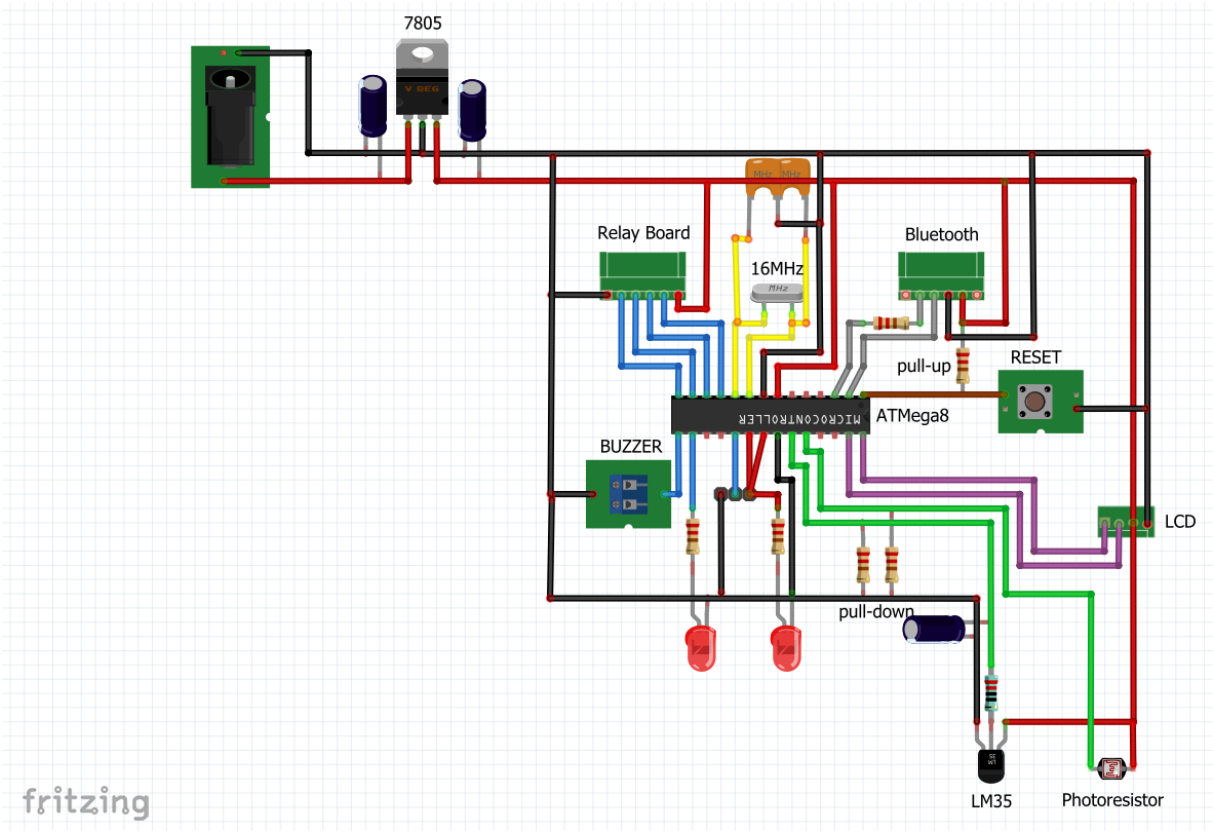

Beltéri
Világítás
OFF
Bluetooth Be Parancs
Beltéri
Világítás
ON
Bluetooth Ki Parancs

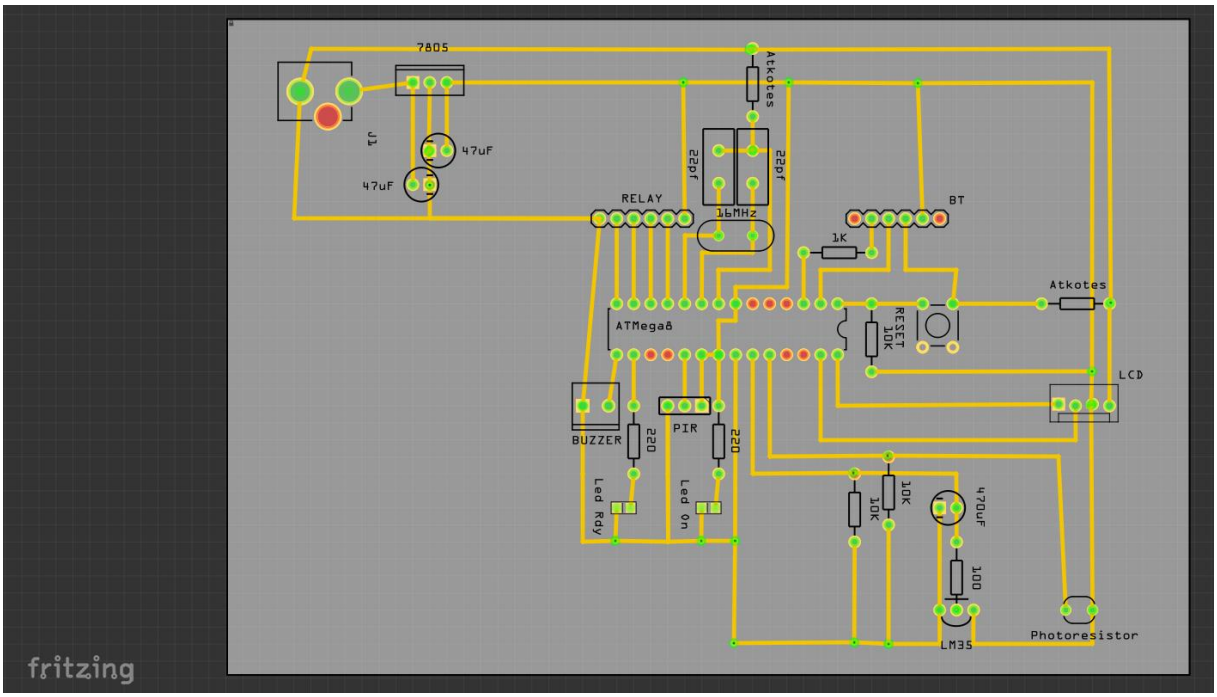

Kültéri
Világítás
OFF
Bluetooth Be Parancs
Sötét és Mozgás érzékelés
Kültéri
Világítás
ON
Mozgás-
érzékelés
3mp az utolsó mozgás óta
Bluetooth Ki Parancs


Kazán és
Fűtést
Jelző Fény
OFF
Aktuális °C kisebb, mint a beállított °C
Kazán és
Fűtést
Jelző Fény
ON
Aktuális °C nagyobb, mint a beállított °C


Riasztó ON
Riasztó
OFF
Bluetooth
Be Parancs
Riasztás
OFF
Mozgás-
érzékelés
Riasztás
ON
Mozgás-
megszűnés
Bluetooth Ki
Parancs
Bluetooth Ki Parancs

Fritzing ábrák:





Működés leírás:

A projekt egy okosotthon vezérlését valósítja meg, mely két főbb részből áll. Egy mikrokonktrollerből és egy okostelefon alkalmazásból (továbbiakban telefon).

Az eszköz egy *ATMega328P* által vezérelt elektronika, mely egy *bluetooth modul*on keresztül kapcsolódik az androidos okostelefonhoz megírt vezérlő alkalmazáshoz.

Főbb funkciók:

- **Világításvezérlés:**

  Kinti és benti világítás vezérlésére van lehetőség. A *kinti világítás* egy időzítővel van ellátva, mely letelte után automatikusan lekapcsol. Felkapcsolására telefonon keresztül van lehetőség, valamint ha a fényerősség érzékelő *fotoellenállás* sötétet érzékel, valamint egyidejűleg a *pir* mozgásérzékelő szenzor is aktiválódik akkor is felkapcsol. Amint a mozgás megszűnik elindul a számláló és a letelte után lekapcsolódik, valamint a telefonon is van egy erre való gomb (a számlálót a végállásba állítja).

  A *benti világítás* csak a telefonos alkalmazás által kapcsolható fel és le.

- **Kazánvezérlés:**

  Az eszköz fel van szerelve egy *LM35*-ös hőmérséklet szenzorral, mely figyeli a szobai hőmérsékletet. A telefonos alkalmazásban be lehet állítani a szobában kívánt hőfokot és amennyiben ez kisebb a mértnél a termosztát *relé*je behúz ami elindítja a kazánt. Ez egészen addig így marad, amíg a mért hő meg nem haladj a beállított értéket.

- **Riasztás funkció:**

  A riasztás funkciót telefonról lehet be és kikapcsolni. Bekapcsolt állapotban amennyiben a *pir* mozgást érzékel a *buzzer* csipogó hangot ad ki egészen addig amíg meg nem szűnik a mozgás, vagy ki nem kapcsoljuk a riasztás funkciót.

- **LCD kijelző:**

  2 x 16 karakteres *LCD kijelző* van az eszközhöz kapcsolva. A felső sorban az aktuális és a kívánt hőfokot jelzi ki, az alsó sorban pedig a riasztó állapotát, hogy be van-e kapcsolva, valamint hogy a *pir* szenzor érzékel-e éppen mozgást.

Felhasznált eszközök:

- ATMega328P mikrokontroller
- HC-05 bluetooth modul
- 4 csatornás relé modul
- HC-SR501 Infravörös mozgásérzékelő szenzor (pir)
- LM35 hőérzékelő szenzor
- Fotoellenállás
- 2 x 16 karakteres LCD kijelző
- Piezo hangszóró
- 7805-ös 5V-os stabilizátor IC
- 1x 470 uF kondenzátor
- 2x 47 uF kondenzátor
- 2x 22 pF kondenzátor
- 16 MHz kvarckristály
- 3x 10K ellenállás
- 1x 1K ellenállás
- 2x 220 ohm ellenállás
- 1x 100 ohm ellenállás

Forráskód:

Arduino_app

```
#include <LiquidCrystal_I2C.h>                                  //lcd-hez kell
LiquidCrystal_I2C lcd(0x27, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);    //lcd-hez kell

//PIN-ek
const int rxPin = 0;
const int txPin = 1;
const int kintiVilagitas = 8;
const int bentiVilagitas = 7;
const int termosztatFutson = 6;
const int pir = 13;
const int hofokPin = A0;
const int fenyPin = A1;
const int piezoPin = 9;
const int readyLedPin = 10;

//hőfokméréshez
const int numReadings = 10;         //hány mérést átlagoljon
int temp;                           //aktuális mérés
int readings[numReadings];          //mérések eredményei
int readIndex = 0;                  //a tömb indexe
int total = 0;                      //mérések összege
int average = 0;                    //mérések átlaga, legpontosabb!!!

//változók
char btFogadottAdat = ' ';          //BT app által küldött karakter
int kivantHofok = 20;               //a kívánt hőfok a szobában
int mozgasVan = 0;                  //érzékel-e mozgást a pir
int riasztoBekapcsolva = 0;         //be van-e kapcsolva a roasztó
int ertekGyujtemeny[4] = {0,0,0,0}; //értékek gyűjteménye ami elküldésre kerül
int ertekGyujtemeny_[4] = {0,0,0,0};//átmeneti tároló a változás vizsgálatához
int fenyero = 0;                    //a fotoellenállás értéke
int kintiVilagitasSzamlalo = 0;     //ez az érték kövekszik ciklusonként
int kintiVilagitasIdotartam = 25;   //meddig világítson a kinti világítás (nem mp.)
int kintiVilagitasFenyErzek = 300;  //hol legyen a sötét / világos fordulópont
```

```
void setup()
{
  Serial.begin(9600);
  delay(200);
  lcd.begin(16,2);
  lcd.backlight();

  //kimenetek
  pinMode(kintiVilagitas, OUTPUT);
  pinMode(bentiVilagitas, OUTPUT);
  pinMode(termosztatFutson, OUTPUT);
  pinMode(piezoPin, OUTPUT);
  pinMode(readyLedPin, OUTPUT);

  //bemenetek
  pinMode(pir, INPUT);
  pinMode(hofokPin, INPUT);
  pinMode(fenyPin, INPUT);

  //mérések átlagait beállítja 0-ra
  for (int thisReading = 0; thisReading < numReadings; thisReading++)
    {
      readings[thisReading] = 0;
    }

  //benti világíáts kikapcsolva induljon
  digitalWrite(bentiVilagitas, HIGH);
  digitalWrite(readyLedPin, HIGH);
  //kis szünet indulás előtt
  delay(500);
}

void loop()
{
  hofokMeres();                       //eredménye az: average
  lcdKijelzes();                      //LCD kijelző frissítése
  bluetoothEsemenyek();               //bejövő parancsok figyelése
  kazanVezerles();                    //hőfokok függvényében ki és bekapcsolja a kazánt
  mozgasErzekeles();                  //figyeli, hogy van-e mozgás
  kintiVilagitasVezerles();           //ez egy számláló, hogy egy idő után kapcsoljon le
  riasztoVezerles();                  //figyeli, hogy be van-e kapcs. a riasztó és van-e mozgás
  ertekekOsszegyujtese();             //változók értékeit összegyűjti egy tömbbe
  ertekekValtozasanakEllenorzese();   //a tömb tartalmát serialon elküldi ha változott valami
  delay(200);                         //kis szünet
}

void hofokMeres()
{
  temp = ((5.0 * analogRead(hofokPin) * 100.0) / 1024) -2 ;
  total = total - readings[readIndex];
  readings[readIndex] = temp;
  total = total + readings[readIndex];
  readIndex = readIndex + 1;
  if (readIndex >= numReadings)
    {
      readIndex = 0;
    }
  average = total / numReadings;
  delay(1);
}
```

```
void lcdKijelzes()
{
  lcd.setCursor(0,0);
  lcd.print("Temp:");
  lcd.setCursor(9,0);
  lcd.print("Set:");
  lcd.setCursor(0,1);
  lcd.print("Alarm:");
  lcd.setCursor(6,0);
  lcd.print(average);
  lcd.setCursor(14,0);
  lcd.print(kivantHofok);
  //Riasztó állapot
  lcd.setCursor(7,1);
  if (riasztoBekapcsolva == 0)
  {
    lcd.print("OFF");
  }
  else if (riasztoBekapcsolva == 1)
  {
    lcd.print("ON ");
  }
  //Mozgás állapot
  lcd.setCursor(11,1);
  if (mozgasVan == 0)
  {
    lcd.print("     ");
  }
  else if (mozgasVan == 1)
  {
    lcd.print("AKTIV");
  }
}

void bluetoothEsemenyek()
{
  //bejövő parancsok figyelése
  if (Serial.available() > 0)
    {
      btFogadottAdat = Serial.read();

        switch(btFogadottAdat)
        {
          case 'a':   digitalWrite(bentiVilagitas, LOW);   //a relék LOW-on vannak bekapcsolva
                      break;

          case 'b':   digitalWrite(bentiVilagitas, HIGH);  //HIGH-on pedig ki
                      break;

          case 'c':   kintiVilagitasOn();
                      break;

          case 'd':   kintiVilagitasOff();
                      break;

          case 'e':   kivantHofokFel();
                      break;

          case 'f':   kivantHofokLe();
                      break;

          case 'g':   riasztoBekapcsolva = 1;
                      break;

          case 'h':   riasztoBekapcsolva = 0;
                      break;
```

```
          case 'i':   ertekekKuldese(); //amikor az android onResume-ja meghívódik,
                      break;            //akkor friss adatokat kér a háztól

          default:    break;
        }
    }
}

void kazanVezerles()
{
  if (kivantHofok > average)
  {
    digitalWrite(termosztatFutson, LOW);
  }
  else if (kivantHofok < average)
  {
    digitalWrite(termosztatFutson, HIGH);
  }
}


void mozgasErzekeles()
{
  if (digitalRead(pir) == HIGH)
  {
    mozgasVan = 1;

    //ha mozgás és sötét van, akkor nullázza a számlálót, azaz indítja a kinti világítást
    if (fenyero < kintiVilagitasFenyErzek)
    {
      kintiVilagitasSzamlalo = 0;
    }
  }
  else if (digitalRead(pir) == LOW)
  {
    mozgasVan = 0;
  }
}

void kintiVilagitasVezerles()
{
  fenyero = analogRead(fenyPin);

  if (kintiVilagitasSzamlalo < kintiVilagitasIdotartam)
  {
    digitalWrite(kintiVilagitas, LOW);
    kintiVilagitasSzamlalo++;
  }
  else
  {
    digitalWrite(kintiVilagitas, HIGH);
  }
}
```

```
void kintiVilagitasOn()
{
  kintiVilagitasSzamlalo = 0;
}

void kintiVilagitasOff()
{
  kintiVilagitasSzamlalo = kintiVilagitasIdotartam;
}

void riasztoVezerles()
{
  if ( (riasztoBekapcsolva == 1) && (mozgasVan == 1) )
  {
    tone(piezoPin, 1000, 70);
  }
  else
  {
    noTone(piezoPin);
  }
}

void kivantHofokFel()
{
  if (kivantHofok < 35)
    kivantHofok++;
}

void kivantHofokLe()
{
  if (kivantHofok > 15)
    kivantHofok--;
}


void ertekekOsszegyujtese()
{
  ertekGyujtemeny[0] = mozgasVan;
  ertekGyujtemeny[1] = riasztoBekapcsolva;
  ertekGyujtemeny[2] = kivantHofok;
  ertekGyujtemeny[3] = average;
}

void ertekekValtozasanakEllenorzese()
{
  if (  (ertekGyujtemeny_[0] != ertekGyujtemeny[0]) ||
        (ertekGyujtemeny_[1] != ertekGyujtemeny[1]) ||
        (ertekGyujtemeny_[2] != ertekGyujtemeny[2]) ||
        (ertekGyujtemeny_[3] != ertekGyujtemeny[3]) )
    {
      ertekekKuldese();
    }
}

void ertekekKuldese()
{
  Serial.print('#');                    //a csomag elejére: #
  for(int k=0; k<4; k++)
  {
    Serial.print(ertekGyujtemeny[k]);   //értékek
    Serial.print('+');                  //az értékek közé: +
  }
  Serial.print('@');                    //a csomag végére: @
  Serial.println();                     //új sor

  //az "előző" értékeknek megadja az aktuálisat
  (ertekGyujtemeny_[0] = ertekGyujtemeny[0]);
  (ertekGyujtemeny_[1] = ertekGyujtemeny[1]);
  (ertekGyujtemeny_[2] = ertekGyujtemeny[2]);
  (ertekGyujtemeny_[3] = ertekGyujtemeny[3]);
}
```

Mentés kész.