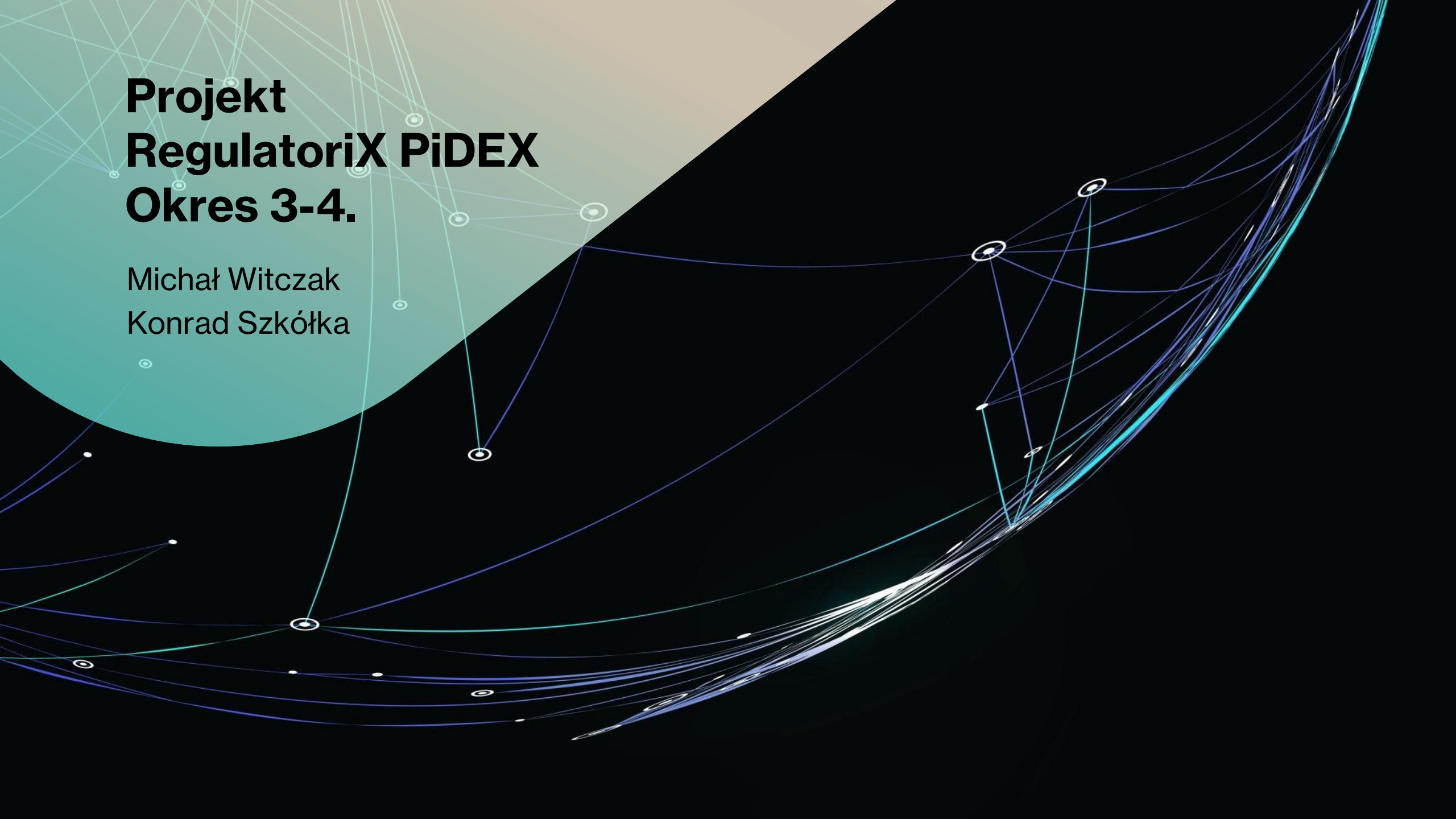
Projekt RegulatoriX PiDEX Okres 3-4.

Michał Witczak
Konrad Szkółka



Największe zadowolenie sprawiła klasa ZapisOdczytUAR

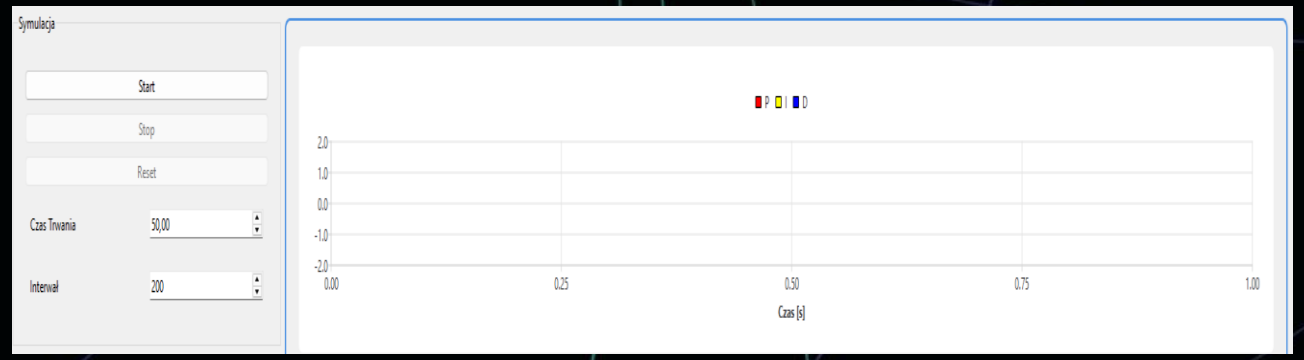
```cpp
class ZapisOdczytUAR
{
public:
    ZapisOdczytUAR() = default;
    ~ZapisOdczytUAR() = default;

    // Zapis konfiguracji (ARX, PID, Generator) do pliku JSON.
    bool zapiszDoPliku(const QString& sciezka,
                       const SymulatorUAR& symulator) const;

    // Odczyt konfiguracji z pliku JSON i ustawienie jej w SymulatorUAR.
    bool odczytajZPliku(const QString& sciezka,
                        SymulatorUAR& symulator) const;
};
```

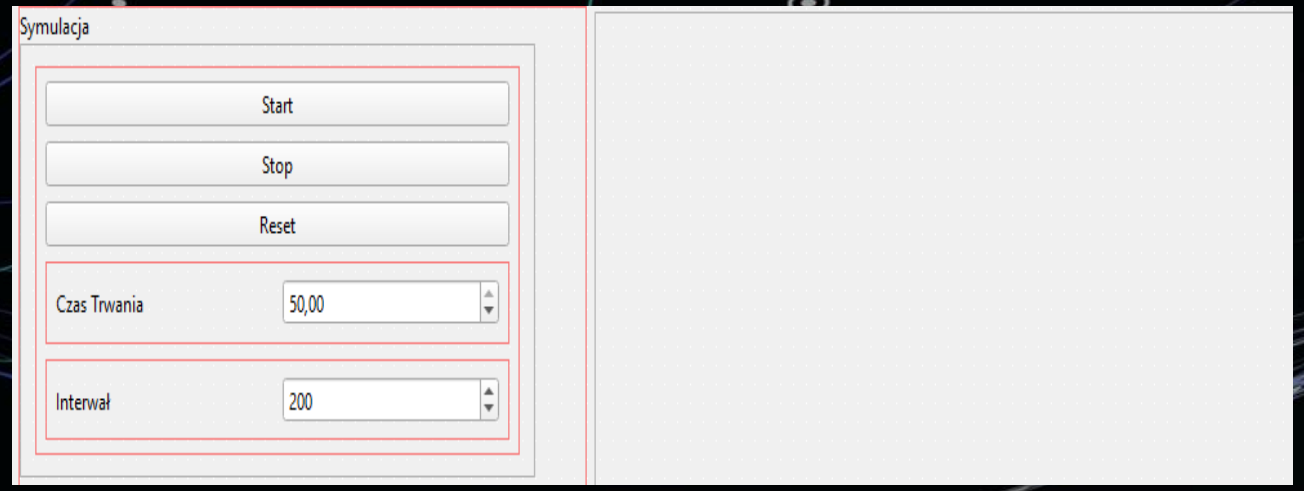
test.json

Plik Edytuj Wyświetl


```json
{
    "generator": {
        "amplituda": 1,
        "okres": 50,
        "okresTRZ": 10,
        "skladowaStala": 0,
        "typ": 1,
        "wypelnienie": 0.5,
        "wypelnienieProc": 50
    },
    "model": {
        "A": "1, -0.4, -0.2",
        "B": "0.6, 0.2, 0",
        "maxVal": 10,
        "minVal": -10,
        "ogr": true,
        "opoznienie": 1,
        "szum": 0
    },
    "regulator": {
        "kp": 1,
        "ogrMax": 10,
        "ogrMin": -10,
        "td": 0,
        "ti": 1,
        "trybCalk": 1
    },
    "symulacja": {
        "czasTrwaniaS": 50,
        "interwalMs": 200
    }
}
```

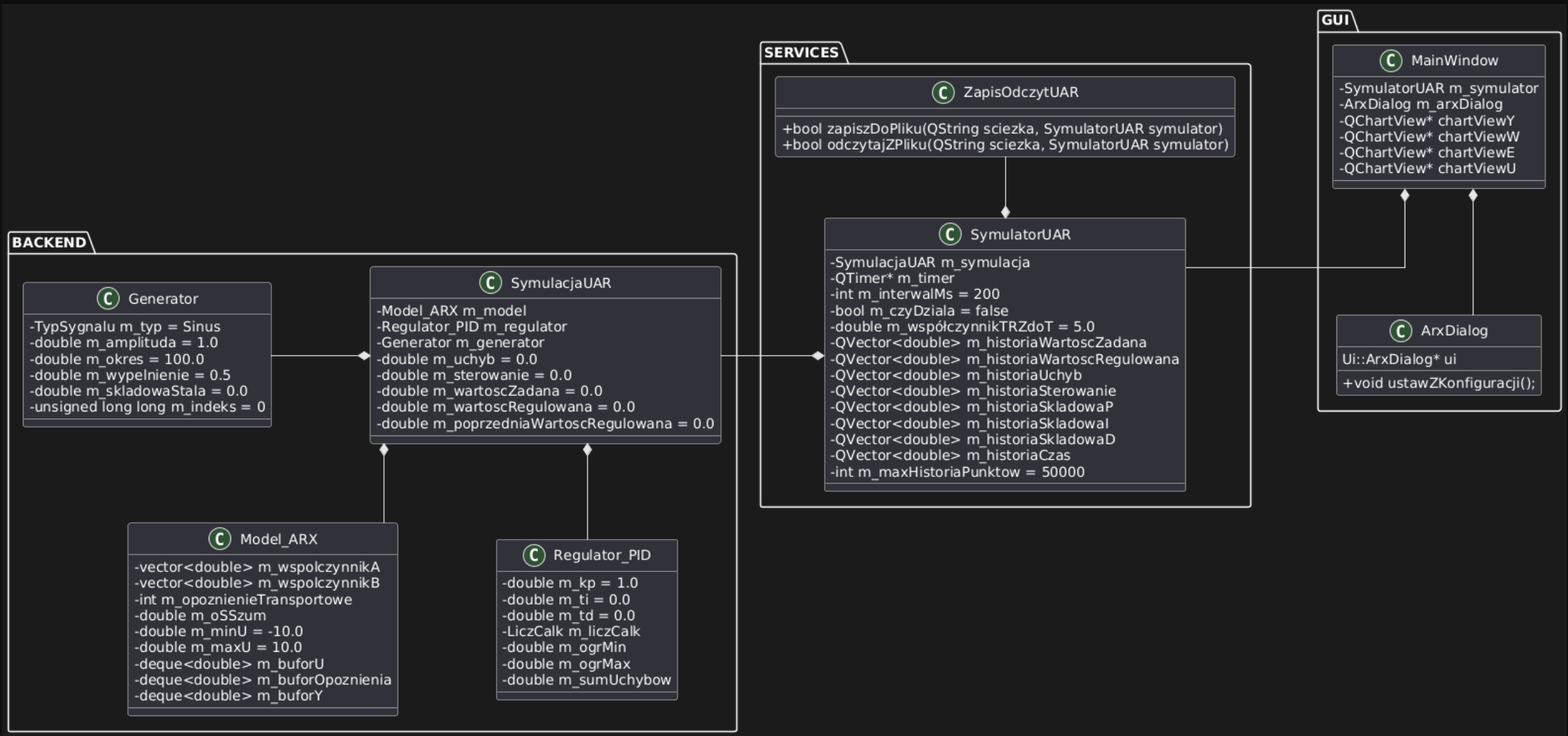
Największe kłopoty sprawiły wykresy






```
62      QWidget *chartsContainer = new QWidget(this);
63      QVBoxLayout *chartsLayout = new QVBoxLayout(chartsContainer);
64      chartsLayout->setSpacing(2);
65      chartsLayout->setContentsMargins(0, 0, 0, 0);
66
67      // ------------------------------------------------------------------
68      // CH1: PID (P czerwony, I żółty, D niebieski)
69      // ------------------------------------------------------------------
70      chartY = new QChart();
71      //chartY->setTitle("P (czerw), I (żół), D (nieb)");
72
73      seriesP = new QLineSeries();
74      seriesP->setName("P");
75      QPen penP(QColor(255, 0, 0), 2);
76      seriesP->setPen(penP);
77      chartY->addSeries(seriesP);
78
79      seriesI = new QLineSeries();
80      seriesI->setName("I");
81      QPen penI(QColor(255, 255, 0), 2);
82      seriesI->setPen(penI);
83      chartY->addSeries(seriesI);
84
85      seriesD = new QLineSeries();
86      seriesD->setName("D");
87      QPen penD(QColor(0, 0, 255), 2);
88      seriesD->setPen(penD);
89      chartY->addSeries(seriesD);
90
91      chartY->createDefaultAxes();
92      chartY->axisX()->setTitleText("Czas [s]");
93      chartY->legend()->setVisible(true);
94      chartY->axisX()->setMinorGridLineVisible(true);
95
96      if (auto *ay = qobject_cast<QValueAxis *>(chartY->axisY()))
97          ay->setRange(-2.0, 2.0);
98
99      chartViewY = new QChartView(chartY, this);
100     chartViewY->setRenderHint(QPainter::Antialiasing);
101     chartViewY->setMinimumHeight(150);
102     chartsLayout->addWidget(chartViewY);
```

Shemat UML


GUI
MainWindow
-SymulatorUAR m_symulator
-ArxDialog m_arxDialog
-QChartView* chartViewY
-QChartView* chartViewW
-QChartView* chartViewE
-QChartView* chartViewU
ArxDialog
Ui::ArxDialog* ui
+void ustawZKonfiguracji();
SERVICES
ZapisOdczytUAR
+bool zapiszDoPliku(QString sciezka, SymulatorUAR symulator)
+bool odczytajZPliku(QString sciezka, SymulatorUAR symulator)
SymulatorUAR
-SymulacjaUAR m_symulacja
-QTimer* m_timer
-int m_interwalMs = 200
-bool m_czyDziala = false
-double m_współczynnikTRZdoT = 5.0
-QVector<double> m_historiaWartoscZadana
-QVector<double> m_historiaWartoscRegulowana
-QVector<double> m_historiaUchyb
-QVector<double> m_historiaSterowanie
-QVector<double> m_historiaSkladowaP
-QVector<double> m_historiaSkladowaI
-QVector<double> m_historiaSkladowaD
-QVector<double> m_historiaCzas
-int m_maxHistoriaPunktow = 50000
BACKEND
Generator
-TypSygnalu m_typ = Sinus
-double m_amplituda = 1.0
-double m_okres = 100.0
-double m_wypelnienie = 0.5
-double m_skladowaStala = 0.0
-unsigned long long m_indeks = 0
SymulacjaUAR
-Model_ARX m_model
-Regulator_PID m_regulator
-Generator m_generator
-double m_uchyb = 0.0
-double m_sterowanie = 0.0
-double m_wartoscZadana = 0.0
-double m_wartoscRegulowana = 0.0
-double m_poprzedniaWartoscRegulowana = 0.0
Model_ARX
-vector<double> m_wspolczynnikA
-vector<double> m_wspolczynnikB
-int m_opoznienieTransportowe
-double m_oSSzum
-double m_minU = -10.0
-double m_maxU = 10.0
-deque<double> m_buforU
-deque<double> m_buforOpoznienia
-deque<double> m_buforY
Regulator_PID
-double m_kp = 1.0
-double m_ti = 0.0
-double m_td = 0.0
-LiczCalk m_liczCalk
-double m_ogrMin
-double m_ogrMax
-double m_sumUchybow

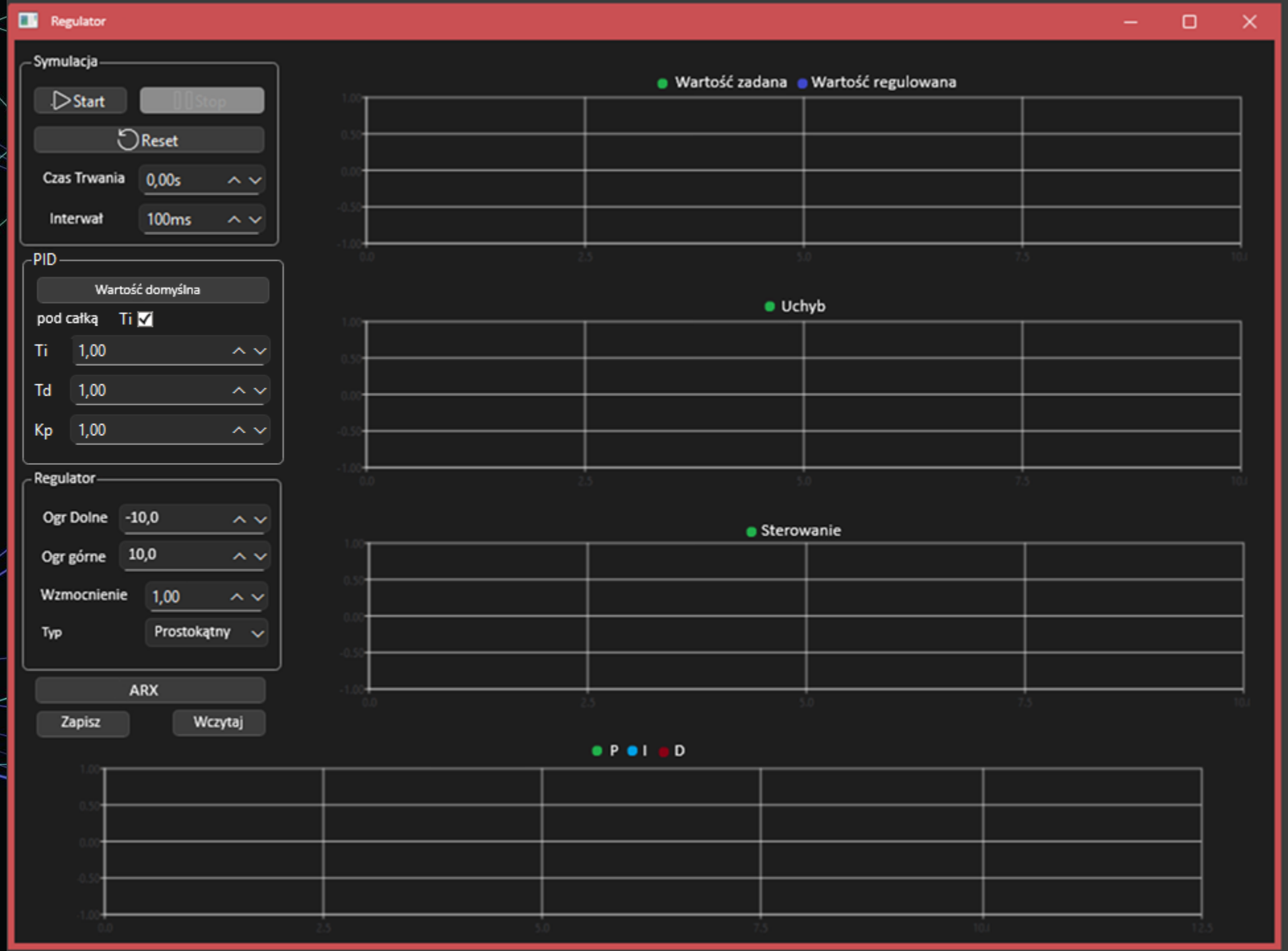
Komunikacja warstwa prezentacji a warstwa usług.

```h
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include "arxdialog.h"
#include "../../BACKEND/BACKEND/SymulatorUAR.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

QT_FORWARD_DECLARE_CLASS(QChartView)
QT_FORWARD_DECLARE_CLASS(QLineSeries)
QT_FORWARD_DECLARE_CLASS(QChart)

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    explicit MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

private slots:
    void on_StartPB_clicked();
    void on_StopPB_clicked();
    void on_ResetPB_clicked();
    void updateChart(double czas, double dummy);

    void on_ARXpushButton_clicked();

    bool eventFilter(QObject *obj, QEvent *event);

    void on_zapiszPushButton_clicked();

    void on_wczytajPushButton_clicked();

private:
    Ui::MainWindow *ui;
    SymulatorUAR m_symulator;
    // dialog ARX
    ArxDialog m_arxDialog;
```

```cpp
MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
    , m_symulator()
    , m_arxDialog()
```

```cpp
void MainWindow::on_StartPB_clicked()
{
    int interwal = ui->interwalSpinBox->value();
    m_symulator.uruchom(interwal);

    ui->StartPB->setEnabled(false);
    ui->StopPB->setEnabled(true);
    ui->ResetPB->setEnabled(true);
}
```

```cpp
void MainWindow::updateChart(double czas, double)
{
    double w = m_symulator.getWartoscZadana();
    double y = m_symulator.getWartoscRegulowana();
    double e = m_symulator.getUchyb();
    double u = m_symulator.getSterowanie();
    double p = m_symulator.getSkladowaP();
    double i = m_symulator.getSkladowaI();
    double d = m_symulator.getSkladowaD();
```


Regulator
Symulacja
Start
Stop
Reset
Czas Trwania
0,00s
Interwał
100ms
PID
Wartość domyślna
pod całką
Ti
Ti
1,00
Td
1,00
Kp
1,00
Regulator
Ogr Dolne
-10,0
Ogr górne
10,0
Wzmocnienie
1,00
Typ
Prostokątny
ARX
Zapisz
Wczytaj
Wartość zadana
Wartość regulowana
Uchyb
Sterowanie
P
I
D


ARX
ARX
A
-0.4, -0.4, 0.6
B
-0.4, -0.4, 0.6
Opóźnienie
2
Szum
0,01
Min
-10,0
Max
10,0
Ograniczenia
OK
Anuluj

MainWindow

Symulacja

Start

Stop

Reset

Czas Trwania 50,00

Interwał 200

Regulator

Wzmocnienie 10,00

Okres 10,00

Wypełnienie 50

Składowa stała 0,00

Typ wykresu Sinusoidalny

PID

Ti 1,00

Td 0,00

Kp 1,00

Ogr Dolne -10,00

Ogr Górne 10,00

Całkowanie

ARX

Zapisz Wczytaj