A digitális PID szabályozó

(oktatási segédanyag)

Készítette: Szűcs Zoltán
Budapesti Műszaki és Gazdaságtudományi Egyetem, Elektronikus Eszközök Tanszéke
H-1521 Budapest, Goldmann György tér 3.
E-mail: szucs@eet.bme.hu

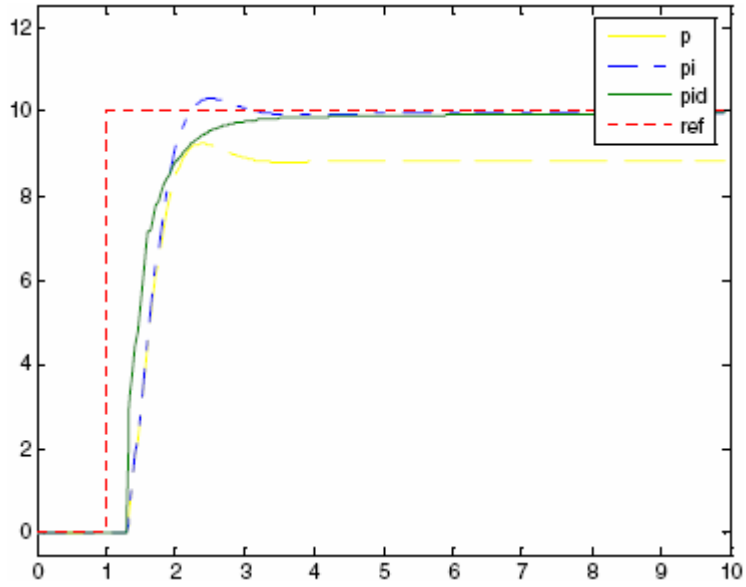
Előszó

Ez a dokumentum röviden bemutatja a digitális PID szabályozók méretezéséhez szükséges elméleti kérdéseket. Először röviden áttekintjük a PID szabályozó működésének alapjait és a digitális szabályozó működésének számítógépes szimulációját C-nyelven, egy egyszerű példán, majd megvizsgáljuk, hogy hogyan lehet egy szabályozást megvalósítani mikrokontrollerrel. Az elméleti áttekintés egyes részeit és bizonyos ábrákat a megadott referenciából vettem át. (Szűcs Zoltán, Bp., 2008. szeptember 20.)

1. Bevezetés

Amikor olyan alkalmazást fejlesztünk, melyben egy rendszer kimenetét a bemenetre adott referencia jel szerint szeretnénk szabályozni, szabályozóra és szabályozóalgoritmusra van szükségünk. Ilyen alkalmazás lehet pl.: motorszabályozás, hőmérséklet, nyomás, áramlási sebesség, sebesség, erő vagy egyéb változó szabályozása. PID szabályozót minden olyan mennyiség szabályozására készíthetünk, amely mérhető és melynek értékét a rendszer valamely bemeneti mennyiségével befolyásolni lehet.

Tudnunk kell, hogy sokféle szabályozási módszer létezik, de a PID az a típus, amely – egyszerűsége és jó teljesítőképessége miatt – mintegy „ipari szabványként” a legjobban elterjedt [1]. Az 1. ábra összefoglalóként mutatja a P, PI és PID szabályozók tipikus válaszait. Ezeket az alábbiakban kicsit részletesebben megvizsgáljuk.



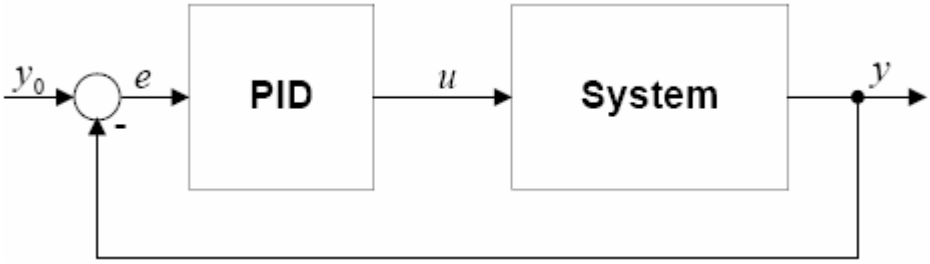
1. ábra – A P, PI és PID szabályozók egységugrás gerjesztésre adott tipikus válaszai

2. A PID szabályozó működése

2. 1. A PID szabályozó felépítése

A 2. ábrán látható egy PID szabályozóval ellátott zárt szabályozási rendszer hatásvázlata. A PID szabályozó a *szabályozott jellemző* mért ***y*** értékét összehasonlítja az y_0 *alapjel*lel. Az e két mennyiség különbségeként képezett *hibajel*ből (***e*** – *error*) ezután kiszámítható az új *beavatkozó jel* (***u***) értéke oly

módon, hogy annak hatására a szabályozott mennyiség értéke közelebb kerüljön az alapjel értékéhez. (A PID szabályozó dolga tulajdonképpen ***u*** előállítása ***e*** függvényében.) A feladat megfogalmazása tehát egyszerű: a szabályozott mennyiség minél gyorsabban és minél pontosabban kövesse az általunk adott alapjel értékét!

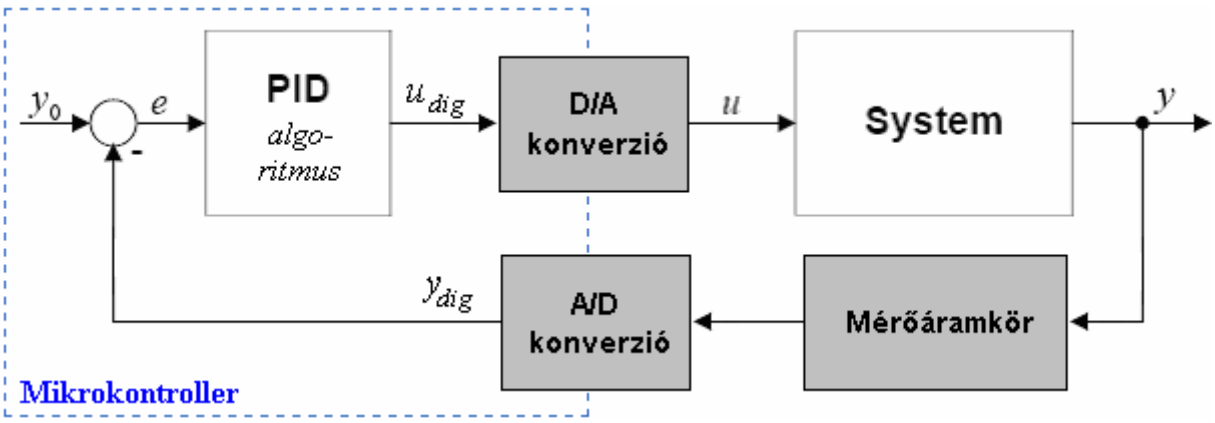


2. ábra – Zárt szabályozási séma PID szabályozóval

Zárt hurkú szabályozás helyett egy lehetséges másik alternatíva a nyílt hurkú vezérlés lenne, de a visszacsatolást nem tartalmazó nyílthurkú rendszerek működése a legtöbb esetben nem kielégítő és sokszor nem is megvalósítható a rendszer tulajdonságai miatt. A szabályozott jellemző értékének visszacsatolásával a rendszer tulajdonságai javíthatók.

Az egyszerű szabályozási algoritmusoktól eltérően a PID szabályozó képes figyelembe venni a kimeneti jel előző időbeli viselkedését és változási sebességét is. Ennek eredménye egy pontosabb és stabilabb szabályozott rendszer.

Bár az alapvető működés és a matematikai modell meglehetősen egyszerű, egy PID-vel szabályozott rendszer fizikai megvalósításához azonban sok feladatot kell elvégeznünk. Célszerű rögtön áramkörtervezői szemmel nézni a szabályozott rendszerünkre. A digitális szabályozó megvalósítását szem előtt tartva (3. ábra) könnyen rájöhetünk, hogy magát a szabályozót és a szabályozó algoritmust egy mikrokontroller, illetve az azon futó program fogja realizálni. Egyszerű esetben az y_0 alapjelet is képezhetjük magával a programmal, vagy pl. megadhatjuk a mikrokontrollernek valamelyik interfészén keresztül egy digitális számként. Tovább gondolkodva arra juthatunk, hogy ha már a szabályozandó rendszert (*System*) realizáltuk, akkor gondoskodni kell a szabályozott mennyiség (***y***) méréséről, illetve a mért érték analóg-digitális konverziójáról és a szabályozó algoritmus által kiszámolt u_{dig} beavatkozó jel digitális-analóg konverziójáról. Szerencsés esetben e két utóbbi művelet is megvalósítható a mikrokontroller saját A/D illetve PWM fokozatával, így „csak” egy mérőáramkört kell megépítenünk.



3. ábra – Digitális PID szabályozási séma

Vizsgáljuk meg az alábbiakban, hogy milyen algoritmus szerint határozzák meg a PID szabályozó egyes tagjai az *u* beavatkozó jelet a hibajel függvényében.

2.2. Az arányos tag

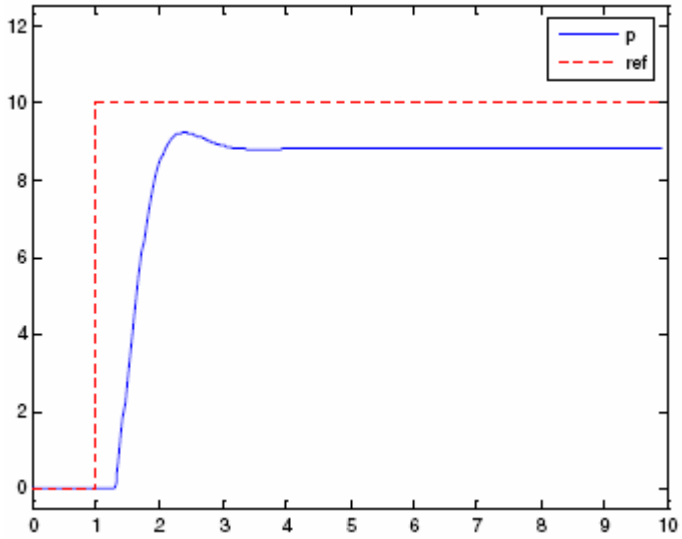
Az arányos, vagy P-tag (P = proportional = arányos) a mindenkori hibajellel (***e***) arányos vezérlőjelet (***u***) szolgáltat. Az arányos tag átviteli függvénye a következőképpen írható fel:

$$H(s) = \frac{u}{e}(s) = K_P$$

Ezt az összefüggést kicsit átrendezve kapjuk azt a formát, melyet mikrokontrollerbe kell majd programoznunk:

$$u(n) = K_P \cdot e(n)$$

Az egyenletben a K_P állandót tulajdonképpen *erősítés*nek is nevezhetjük, hiszen a vezérlő jel a mindenkori hibajel K_P-szerese lesz. Diszkrét időlépték lévén, az aktuális ütem számát *n*-el jelöltük. A 4. ábrán látható folytonos, kék görbe az arányos taggal szabályozott rendszer időbeli válaszának tipikus alakját mutatja. Az ábrát nézve azonnal nyilvánvalóvá válik, hogy kizárólag P-tag segítségével az igényes szabályozás követelményei nem elégíthetők ki. Ha csak P-taggal működtetjük a szabályozónkat, akkor azt tapasztaljuk, hogy a szabályozott jel bizonyos nagyságú hibával éri el az állandósult állapotot (kivéve akkor, amikor az alapjel (*ref*) nulla és a kimenő jel értéke ezzel egyenlő). Ez az állandósult hiba az arányos tag *erősítés*ének növelésével csökkenthető—elvileg tetszőleges mértékben, de a valóságban a túl nagy K_p-jű P-tag instabil rendszert eredményezhet. A célunk a következőkben az lesz, hogy az állandósult hibától megszabaduljunk.



4. ábra – Az arányos tag ugrásválasza

2.3. Az integráló tag

A 4. ábrát nézve, szemléletesen fogalmazva azt mondhatnánk, hogy a referencia jel és a kimenő jel állandósult állapotbeli különbsége (a hiba) már olyan kicsi, hogy ezt a kis hibajelet a P-tag nem képes kompenzálni, mert a kis hiba által okozott beavatkozó jel nagysága nem elég a hiba eltűntetéséhez (értsd: csak a rendszer veszteségeinek fedezéséhez elég). Olyan járulékos tag bevezetésére lenne szükség, amiben a nagyon kis hibajel az idő folyamán mintegy „felhalmozódik” és ily módon válik detektálhatóvá és persze kompenzálhatóvá. Ez matematikailag *integrálás*t jelent, vagyis be kell vezetnünk az I-tagot.

Az integráló tag tehát olyan járulékos vezérlőjelet szolgáltat, amely a hibák előző időbeli összegétől (pontosabban integráljától) függ. Megint kicsit hétköznapiasan fogalmazva mondhatjuk, hogy ez a tag „*veszi figyelembe a kimenő jel történetét*”.

Nagyon könnyű belátni, hogy az integráló tag eltűnteti az állandósult állapotbeli hibát: elég csak abba belegondolni, hogy a kis hibák integrálása mindaddig nullától különböző „összeget” ad eredményül, amíg az **állandósult** hiba értéke zérus nem lesz. Csak ebben az egy esetben nem változik már tovább a hibaösszeg.

Az integráló tag átviteli függvénye a következőképpen írható fel:

$$H(s) = \frac{u}{e}(s) = \frac{1}{T_i \cdot s}$$

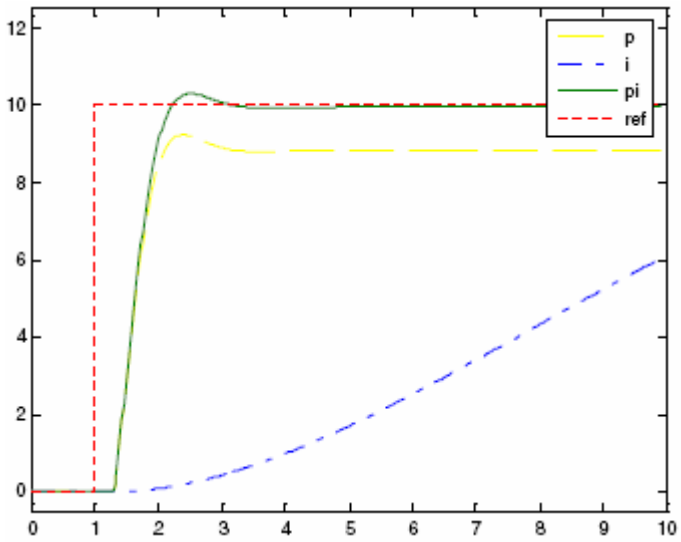
Komplex frekvenciatartomány helyett ezt időtartományban felírva a szokásos integrálást látjuk, de ezt nyilván nem tudjuk beprogramozni egy mikrokontrollerbe, ezért diszkrét időben az integrálás műveletét összegzéssel közelítjük a következő képlet szerint:

$$\int_0^t e(\tau)d\tau \approx T\sum_{k=0}^{n} e(k)$$

Az integráló tag hozzájárulása a P-tag vezérlő jeléhez diszkrét szabályozó esetében tehát a következő lesz:

$$u(n) = K_i \cdot \sum_{k=0}^{n} e(k)$$

Ahol K_i az integráló tag együtthatója—ez, mint konstans paraméter szerepel majd a mikrokontroller programjában.

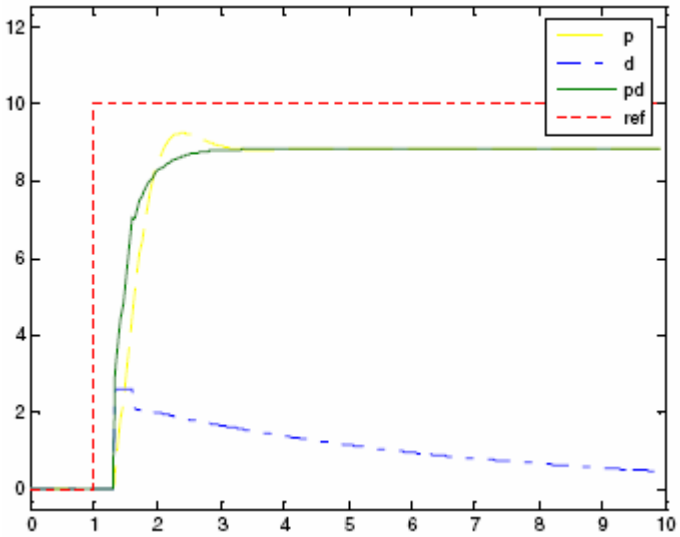


5. ábra – Az I és a PI-tag ugrásválasza

2.4. A deriváló tag

A deriváló tag járulékos vezérlőjelet szolgáltat a hibajel változási sebességétől függően. A hibajelben bekövetkező gyors változás nagymértékben megnöveli a vezérlőjelet így a D-tag az alapjelben, vagy a rendszer állapotában bekövetkező gyors változások esetén javítja a követési tulajdonságokat. A D-tagot tipikusan a P és a PI-tagokkal együtt használjuk PD vagy PID szabályozóként.

A túl nagy D-tag általában instabil rendszert eredményez. A 6. ábrán látható a D és a PD-tag ugrásválasza. Látható, hogy a PD szabályozó alkalmazása gyorsabb felfutást eredményez, mint a P. Mondhatnánk, hogy a D-tag alapvetően „úgy viselkedik, mint egy felüláteresztő szűrő” – a hibajel magasabb frekvenciájú komponenseire érzékeny, így aztán könnyebben instabilitást visz a rendszerbe, egyszersmind érzékenyebbé téve azt a zajokra is.



6. ábra – A D és PD-tag ugrásválasza

A D-tag viselkedését időtartományban és komplex frekvenciatartományban a következő két képlet írja le:

$$u(t) = T_d \cdot \frac{de(t)}{dt}$$

$$H(s) = \frac{u}{e}(s) = T_d \cdot s$$

Az első, időtartománybeli képletből kiindulva könnyen kitalálható, hogy mit kell majd a mikrokontrollerbe programoznunk. Mivel egzakt deriválást megvalósítani mikrokontrollerrel nem tudunk, ezért a következő közelítést vezetjük be:

$$\frac{de(t)}{dt} \approx \frac{e(n) - e(n-1)}{T}$$

Ahol *e(n)* az n-edik, *e(n-1)* pedig az n-1-edik ütembeli hiba, és *T* pedig az ütem hossza, vagyis a két mintavételezés között eltelt idő. A hibajel deriváltját tehát egyszerűen a hiba előző ütembeli megváltozásával helyettesítettük. Ez annál jobban közelíti a valódi deriváltat, minél kisebb a diszkrét időlépték, azaz minél nagyobb frekvenciájú a mintavételezés.

A mikrokontrollerbe beírandó képlet a D-tag esetében a fentiek alapján a következő lesz:

$$u(n) = K_d \cdot (e(n) - e(n-1))$$

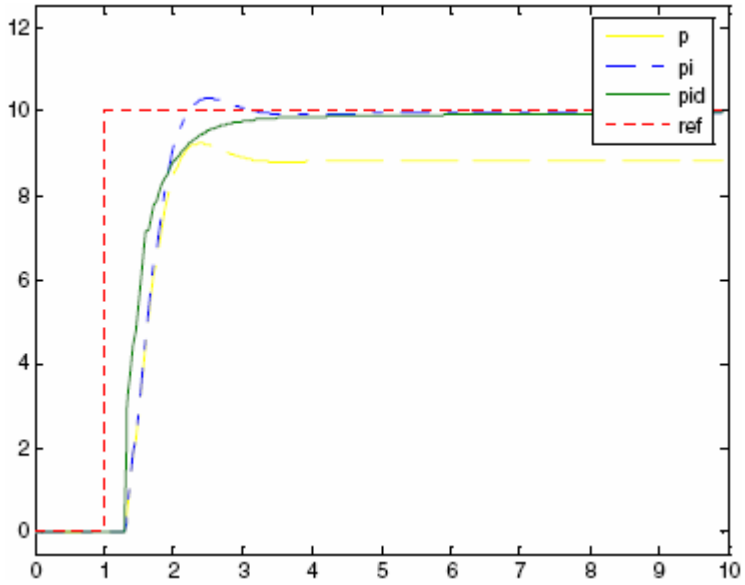
Az előző tagokhoz hasonlóan, itt is *K*-val jelöltük a tag együtthatóját, amely szintén konstansként szerepel majd a programunkban.

2.5. A PID szabályozó digitális megvalósítása

Tudjuk, hogy a folytonos idejű PID szabályozó a 7. ábrán látható ugrásválaszt adja, ha helyesen vannak megválasztva az egyes tagok időállandói, illetve az erősítés. Az előző három pont alapján nagyon könnyen fel lehet írni a PID szabályozó digitális megvalósításához szükséges képletet:

$$u(n) = K_p \cdot e(n) + K_i \cdot \sum_{k=0}^{n} e(k) + K_d \cdot (e(n) - e(n-1))$$

Ezt az összefüggést probléma nélkül be lehet programozni egy mikrokontrollerbe. Néhány dologra azonban vigyázni kell. Az I és D-tag csúlcsordulást okozhat a mikrokontroller regisztereiben, ezért szükséges figyelni ezek abszolút értékét és egy paraméterként előre megadott értéknél lekorlátozni.



7. ábra – A folytonos idejű P, PI és PID szabályozó ugrásválasza

3. Példa PID szabályozó alkalmazására

Ebben a fejezetben egy valódi példán próbáljuk meg érzékeltetni a PID szabályozó működését. Egy egyszerűsített matematikai modell alapján leírjuk a rendszerünket, majd ezt a leírást használva számítógép segítségével szimuláljuk a szabályozott rendszer működését.

3.1. A szabályozott rendszer modellezése

Tegyük fel, hogy egy zárt tartályban lévő *m* tömegű víz hőmérsékletét szeretnénk szabályozni. A tartály fala hőszigetelt és a szigetelés tökéletlensége miatt hőveszteség keletkezik. Az egyszerűség kedvéért csupán egy R_{th} termikus ellenállást vegyünk figyelembe és hanyagoljunk el minden más veszteséget okozó hatást.

Feltételezzük továbbá, hogy a vizet egy P_{max} teljesítményű fűtőszállal melegítjük és termoelemes mérőáramkörrel mérjük a hőmérsékletet. Ahhoz, hogy ne kelljen *holtidő*vel számolnunk, tegyük fel, hogy a vizet a tartályban folyamatosan keringetjük és ily módon a fűtés bekapcsolásakor a mérőáramkör azonnal észleli a víz melegedését.

A rendszer működését leíró egyenletek ekkor a következők lesznek:

1.)	$Q = c \cdot m \cdot \Delta T$	*c* fajhőjű és *m* tömegű anyag hőmérsékletének ΔT fokkal történő megváltoztatásához szükséges hőmennyiség (energia, J)
2.)	$P = \frac{T_H - T_C}{R_{th}}$	Adott R_{th} termikus ellenálláson *időegység alatt átáramló energia* (teljesítmény, J/s) a melegebb pont (T_H) felől a hidegebb pont (T_C) felé. Ez az Ohm-törvény termikus megfelelője.

Az első összefüggést Δt-vel elosztva (ez lesz majd a diszkrét időlépték) ott is teljesítményre jutunk:

$$\frac{Q}{\Delta t} = P = c \cdot m \cdot \frac{\Delta T}{\Delta t}$$

A rendszerbe a fűtőszál segítségével bevitt P_{futo} teljesítmény két dologra fordítódik: a víz hőmérsékletének növelésére és a tartály falán keresztül elszivárgó hő utánpótlására. Ezt a következő képlettel lehet matematikailag megfogalmazni:

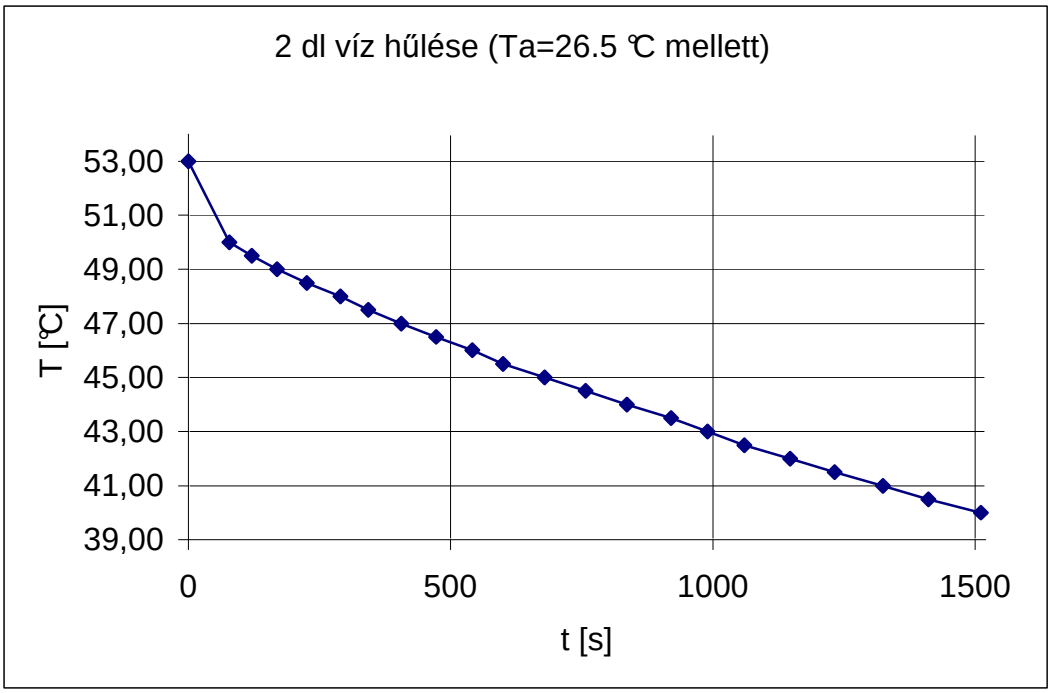
$$P_{futo} = c \cdot m \cdot \frac{\Delta T}{\Delta t} + \frac{T_{akt} - T_a}{R_{th}}$$

A képletben Δt lesz a két mintavételezés között eltelt idő, c a fajlagos hőkapacitás, m a tömeg, ΔT a Δt idő alatt bekövetkező hőmérsékletváltozás, T_{akt} az aktuális hőmérséklet, T_a a környezeti hőmérséklet, R_{th} pedig a már említett termikus ellenállás.

Ahhoz, hogy a rendszer működését számítógéppel szimulálni tudjuk, szükség lenne az R_{th} termikus ellenállás ismeretére. Ezt próbáljuk az alábbiakban hozzávetőlegesen megmérni egy egyszerű kísérlettel: egy bögrébe 2 dl vizet (m=0.2 kg) teszünk és mikrohullámú sütőben felmelegítjük legalább 55 °C-ra. Melegítés után alaposan megkeverjük és beleteszünk egy pontos hőmérőt (8. ábra).

8. ábra – Egyszerű kísérlet a termikus ellenállás mérésére

A magárahagyott rendszer időbeli viselkedését figyeljük és bizonyos időközönként feljegyezzük a mért hőmérsékletet (vagy épp fordítva). Minden extra jelenség hatását elhanyagolva, egy exponenciális hűlési görbére számítunk, melynek egyenlete – két pontjának ismeretében – könnyen felírható. Mérjük le, hogy kb. mennyi idő alatt hűl le a víz $T(t_1) = 50$°C-ról $T(t_2)=40$ °C-ra (nyilvánvaló, hogy olyan alsó hőmérsékletet kell választani ami magasabb, mint a környezeti hőmérséklet). Az általam elvégzett mérés eredményeit a 9. ábra foglalja össze.



9. ábra – Mért hűlési görbe (a görbe kezdeti szakaszán még a keverés hatása látható!)

A víz hőmérséklete $t_1 = 78$ s-nál érte el a $T(t_1) = 50$ °C-ot és $t_2 = 1511$ s-nál a $T(t_2) = 40$ °C-ot. Fontos volt azt is megmérni, hogy a környezeti hőmérséklet eközben végig $T_a = 26.5$ °C volt.

A kérdéses $\Delta t = t_2 - t_1$ idő ismeretében már ki lehet számolni R_{th} értékét a következő gondolatmenettel:

1.)	$T(t) = T_a + (T_0 - T_a) \cdot e^{-\frac{t}{\tau}}$	Ez írja le a hűlés görbéjét. A görbe két pontja ismeretében szeretnénk meghatározni a τ időállandót.
2.)	$T(t_1) = T_a + (T_0 - T_a) \cdot e^{-\frac{t_1}{\tau}}$ $T(t_2) = T_a + (T_0 - T_a) \cdot e^{-\frac{t_2}{\tau}}$	E két egyenlet írja le a görbe két pontját. Fejezzük ki mindkettőből az exponenciális tényezőt…
3.)	$\frac{T(t_1) - T_a}{T_0 - T_a} = e^{-\frac{t_1}{\tau}}$ $\frac{T(t_2) - T_a}{T_0 - T_a} = e^{-\frac{t_2}{\tau}}$	Osszuk el az első egyenletet a másodikkal!
4.)	$\frac{T(t_1) - T_a}{T(t_2) - T_a} = e^{-\frac{t_1}{\tau} + \frac{t_2}{\tau}}$	Láthatjuk, hogy az ismeretlen T_0 ezzel ki is esik. Most vegyük mindkét oldal természetes logaritmusát (és cseréljük meg az oldalakat)!
5.)	$\left(\frac{\Delta t}{\tau}\right) = \frac{t_2 - t_1}{\tau} = \ln \frac{T(t_1) - T_a}{T(t_2) - T_a}$	t_2-t_1-et Δt-vel helyettesítjük és kifejezzük a τ időállandót…
6.)	$\tau = -\frac{\Delta t}{\ln \frac{T(t_2) - T_a}{T(t_1) - T_a}}$	A jobb oldal kiértékelhető, hiszen lemértük, hogy: $T(t_1) = 50$ °C; $T(t_2) = 40$ °C;

		T_a = 26.5 °C és Δt = 1433 s. τ értékére ekkor 2585.2 s-ot kapunk.
7.)	$\tau = R_{th}C \approx 2585.2s$	A $C = c \cdot m$ hőkapacitás ismeretében R_{th} már számítható. A víz tömege m = 0.2 kg, és a függvénytáblázat alapján tudjuk, hogy a víz fajhője c = 4183 J/(kg · °C)
8.)	$R_{th} = \frac{\tau}{C} = \frac{\tau}{c_{víz} \cdot m} = \frac{2585.2}{4183 \cdot 0.2} \approx 3.09 \frac{°C}{W}$	A veszteséget tehát egy 3 °C/W értékű termikus ellenállással modellezzük majd a következő alfejezetben.

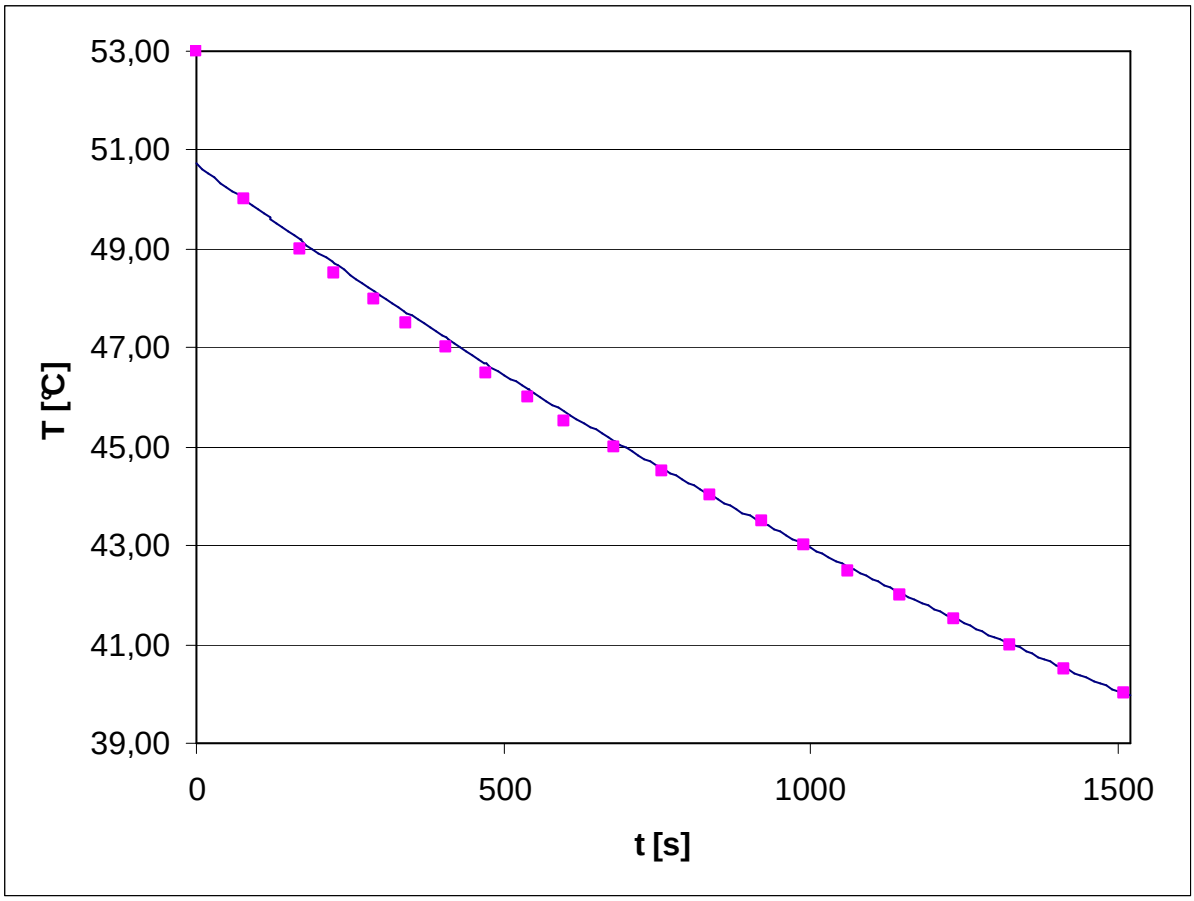
Természetesen kíváncsiak vagyunk, hogy vajon mi az eredeti függvény és hogy ennek görbéje milyen pontosan illeszkedik a hőmérővel mért értékeinkre (vagyis: hogy mennyire lett jó a közelítésünk?).

9.)	$T_0 = \frac{T(t_1) - T_a}{e^{-\frac{t_1}{\tau}}} + T_a$	Fejezzük ki T_0-t, majd τ, illetve az első mérési pont adatainak visszahelyettesítésével határozzuk meg értékét!
10.)	$T_0 = \frac{50 - 26.5}{e^{-\frac{78}{2585.2}}} + 26.5 = 50.72$ °C	t_1 = 78 s $T(t_1)$ = 50 °C; T_a = 26.5 °C és τ = 2585.2 s

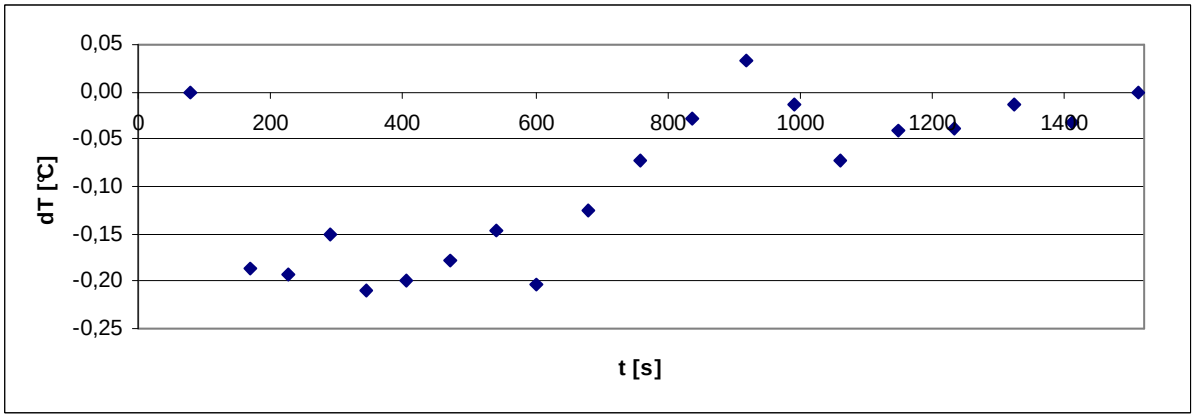
Tehát az exponenciális görbe az y tengelyt a T_0= 50.72 °C-nál metszi. T_0 és τ ismeretében immár fel tudjuk írni a görbe egyenletét, ami a következő:

$$T(t) = 26.5 + (50.72 - 26.5) \cdot e^{-\frac{t}{2585.2}}$$

Ábrázoljuk ezt a görbét a mért adatainkkal együtt, közös koordinátarendszerben (10. ábra)! Láthatjuk, hogy ebben a tartományban a közelítés elfogadható (a legnagyobb eltérés -0.21 °C volt (11. ábra)).



10. ábra – A számolt görbe összevetése a mért értékekkel



11. ábra – Mérési hibák az egyes mérési pontokban

Térjünk vissza most a szabályozástechnikai vizsgálathoz. Kiszámítottuk R_{th} értékét, így már le tudjuk írni a rendszerünk viselkedését matematikailag. A további teendőnk az, hogy kiegészítsük ezt a leírást a PID szabályozó matematikai leírásával és az egész működést egy C-programmal szimuláljuk.

3.2. A szabályozóval kiegészített rendszer számítógépes szimulációja

Az egyszerűség kedvéért a szimulációt egy C-nyelven megírt programmal végezzük. (Természetesen léteznek erre a célja más alkalmazások is, pl. MATLAB, de ezek az eszközök általában nem ingyenesek.) Az alábbiakban látható tehát egy néhány soros C-program, részletes megjegyzésekkel ellátva. A programot futtatva egy „pid_sim.csv” nevű fájlt kapunk eredményül, amely MS-Excellel megnyitható és ott néhány kattintással grafikon készíthető a nyert adatokból.

```
//PID_SIM - PID szabályozó szimulátor program
//Szűcs Zoltán - 2008. szeptember 20.

#include <stdio.h>
#include <stdlib.h>

int main(void) {
//A RENDSZER PARAMÉTEREI:
    double cviz=4183.2; //A víz fajlagos hőkapacitása: J/(kg*°C)
    double m=0.2;       //A víz tömege: kg
    double Ta=25;       //Környezeti hőmérséklet: °C
    double T0=25;       //A víz kezdeti hőmérséklete: °C
    double Rth=3.09;    //A víz és a környezet közötti hőellenállás: °C/W

//A BEAVATKOZÓ SZERV PARAMÉTERE(I):
    double Pmax=400;    //A maximális fűtőteljesítmény: W

//A SZABÁLYOZÓ PARAMÉTEREI:
    double Kp=100;      //Az arányos tag együtthatója
    double Ki=10;       //Az integráló tag együtthatója
    double Kd=60;       //A deriváló tag együtthatója

    double MaxP=1000;   //A P-tag maximális ABSZOLUTÉRTÉKE!!!
    double MaxI=1000;   //Az I-tag maximális ABSZOLUTÉRTÉKE!!!
    double MaxD=1000;   //A D-tag maximális ABSZOLUTÉRTÉKE!!!
    double MaxU=1000;   //A vezérlőjel maximális értéke

//SZIMULÁCIÓS PARAMÉTER(EK):
    double dt=1e-3;     //A diszkrét időlépték: s
       int StopTime=200;//A szimuláció futási ideje: s
    double Tref=40;     //A referencia hőmérséklet: °C

//EGYÉB VÁLTOZÓK:
       int k=0;         //A diszkrét ütemszám változója
    double t;           //Az aktuális idő másodpercben
    double P=0;         //Az arányos tag értéke
    double I=0;         //A hibaintegrál értéke
    double D=0;         //A jelváltozási sebesség értéke
    double Takt, Tdig;  //Az aktuális vízhőmérséklet
    double eelozo=0;    //Az előző hiba értéke
    double e;           //Az aktuális hiba értéke
    double Pfuto;       //Az aktuális fűtőteljesítmény
    double deltaT;      //A hőmérséklet növekménye
    double U;           //A vezérlőjel változója
//PARAMÉTERLISTA VÉGE
    FILE * pFile;
    pFile = fopen ("pid_sim.csv","w"); //Az eredményt tartalmazó szöveges file

//A SZIMULÁCIÓ:
    Takt=T0;            //A kezdeti vízhőmérséklet beállítása
    for (k=0; k<(StopTime/dt); k++)
    {
        t=k*dt;         //Az aktuális idő meghatározása
        Tdig=Takt;      //"Az A/D végez egy mérést" digitalizáljuk a hőmérsékletet

        eelozo=e;       //Az előző hiba elmentése
        //if (k==30000) Tref=40;
        e=Tref-Tdig;    //Az aktuális hiba kiszámítása

        //A P-tag értékének kiszámítása:
        if ((P=Kp*e)>MaxP) P=MaxP;
        else if (P<(-1*MaxP)) P=-1*MaxP;

        //A D-tag értékének kiszámítása:
        if ((D=Kd*(e-eelozo))>MaxD) D=MaxD;
        else if (D<(-1*MaxD)) D=-1*MaxD;

        //Az I-tag értékének kiszámítása:
```

```
            if ((I+=(Ki*e))>MaxI) I=MaxI;
            else if (I<(-1*MaxI)) I=-1*MaxI;

            //A beavatkozó jel (k) kiszámítása:
            U=P+I+D;            //A PID szabályozó képlete
            if (U>MaxU) U=MaxU;
            else if (U<0) U=0; //Itt figyelünk rá, hogy a fűtőtelj. negatív nem lehet:
                               //vagyis a rendszer csak fűteni tud, hűteni nem!

            //A fűtőteljesítmény kiszámítása: [Ez tk. a D/A rész!]
            //Itt a vezérlőjelet át kell transzformálni a 0..1 tartományba.
            Pfuto=(U/1000)*Pmax;

            //A hőmérsékletváltozás a k-adik ütemben:
            deltaT=(Pfuto-(Takt-Ta)/Rth)*dt/(cviz*m);
            Takt+=deltaT;

            if (!(k%40))        //Minden 40. ciklusban elmentjük az értékeket
            {
            fprintf(pFile,"%d;%.3lf;%.3lf;%.3lf;%.3lf;%.3lf\n",k,t,Takt,P/10,I/10,D/10);
            printf("A %d-dik ciklus: %f;P=%.3lf; I=%.3lf; D=%.3lf\n", k, Takt,P,I,D);
            }
        }
        fclose (pFile);
}
```

3.3. Szimulációs eredmények

A fenti programmal elvégzett szimuláció eredményei láthatóak a 12. ábrán. A legfontosabb paraméterek a következők voltak:

- $m_{víz}$ = 0.2 kg a melegített víz tömege
- P_{futo} = 400 W a fűtőteljesítmény maximális értéke.
- A szabályozó paraméterei:
  - K_p = 100 az arányos tag erősítése
  - K_i = 10 az integráló tag együtthatója
  - K_d = 60 a deriváló tag együtthatója
- A diszkrét időlépték: dt = 1 ms (azaz a mintavételi frekvencia: 1 kHz)
- A szimuláció futási ideje: 200 s
- Az elérni kívánt (referencia)hőmérséklet: T_{ref} = 40 °C

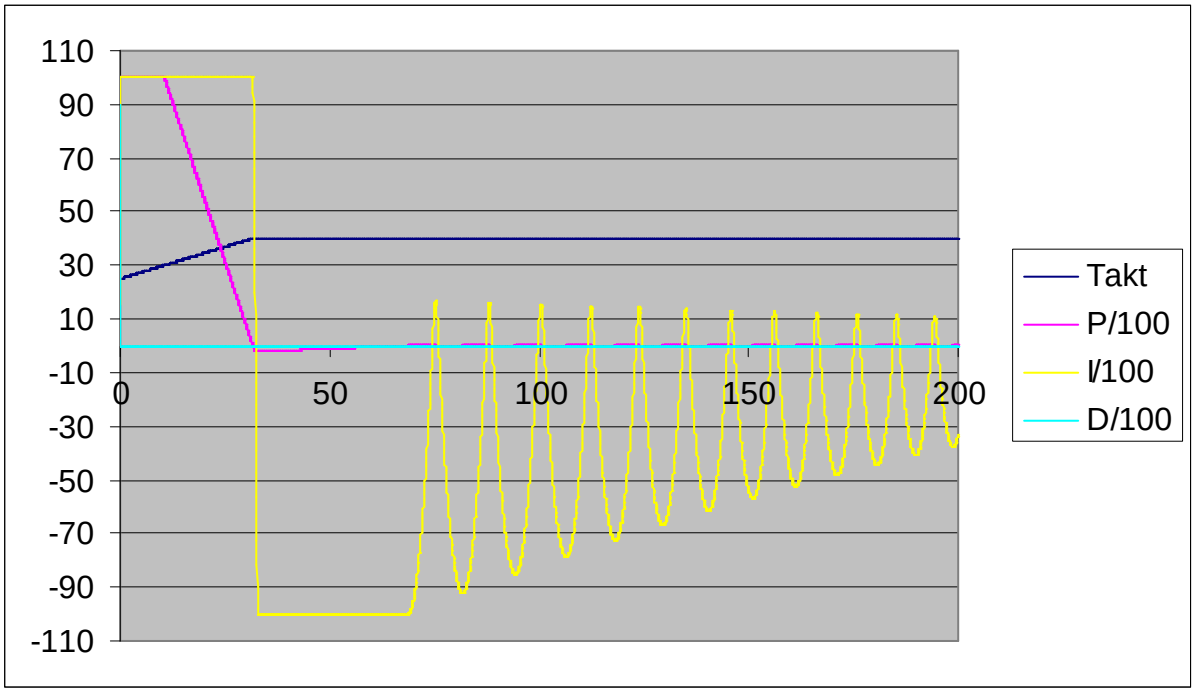
Foglaljuk össze szöveggel, hogy mit látunk a grafikonon! A sötétkék görbe reprezentálja a melegített víz mindenkori hőmérsékletét. Látható, hogy ez t = 0-ban 25 °C-ról indul, és közel lineárisan növekszik, amíg t = 33 s környékén a 40 °C-os referenciahőmérsékletet el nem éri, majd látszólag nem változik tovább.

A rózsaszín görbe az arányos (P) tag hozzájárulása a vezérlőjelhez. Azt látjuk, hogy kezdetben, amíg a hiba nagy, ez a jel a maximális értéknél korlátozva van, majd ahogy a víz melegszik, t = 11 s környékén a hiba csökkenésével arányosan elkezd csökkenni, s amikor a vízhőmérséklet eléri a referenciahőmérsékletet éppen nulla lesz és a továbbiakban nulla közelében tartózkodik.

A sárga görbét is vizsgáljuk meg! Ez ugyanis az integráló tag hozzájárulása a vezérlőjelhez. Nyilvánvaló, hogy nagy hibajel esetén (kezdetben) a hibaintegrál hamar eléri a maximális értéket, ahol a szoftveres korlátunk működésbe lép, és egészen addig nem csökken, amíg a hőmérséklet a referenciahőmérséklet fölé nem megy (t = 33 s környékén, hiszen csak ekkor válik majd negatívvá a hibajel). Ezután a túllendülés jelensége miatt, a görbe bizonyos ideig eléri a negatív korlátot is, mert a rendszer lassú hűlése miatt sok negatív hiba halmozódik fel, majd ahogy a lengések csillapodnak, a továbbiakban már a korlátokon belül tartózkodik.

A D-tag világoskék görbéjét nézve megállapíthatjuk, hogy ebben a beállításban ez a tag kevéssé járul hozzá a szabályozáshoz, így a szabályozónk inkább csak PI-jellegű. Más beállítást kellene keresnünk ahhoz, hogy a D-tag hatása is számottevő legyen.

A P, I és D tagok együtthatóit nem csak számítással, hanem kísérleti úton is meg lehet határozni. Ekkor tk. *behangoljuk* a szabályozót, melyre többféle eljárást is találhatunk a szakirodalomban (pl. Ziegler-Nichols módszer), de ezekkel itt most nem foglalkozunk.

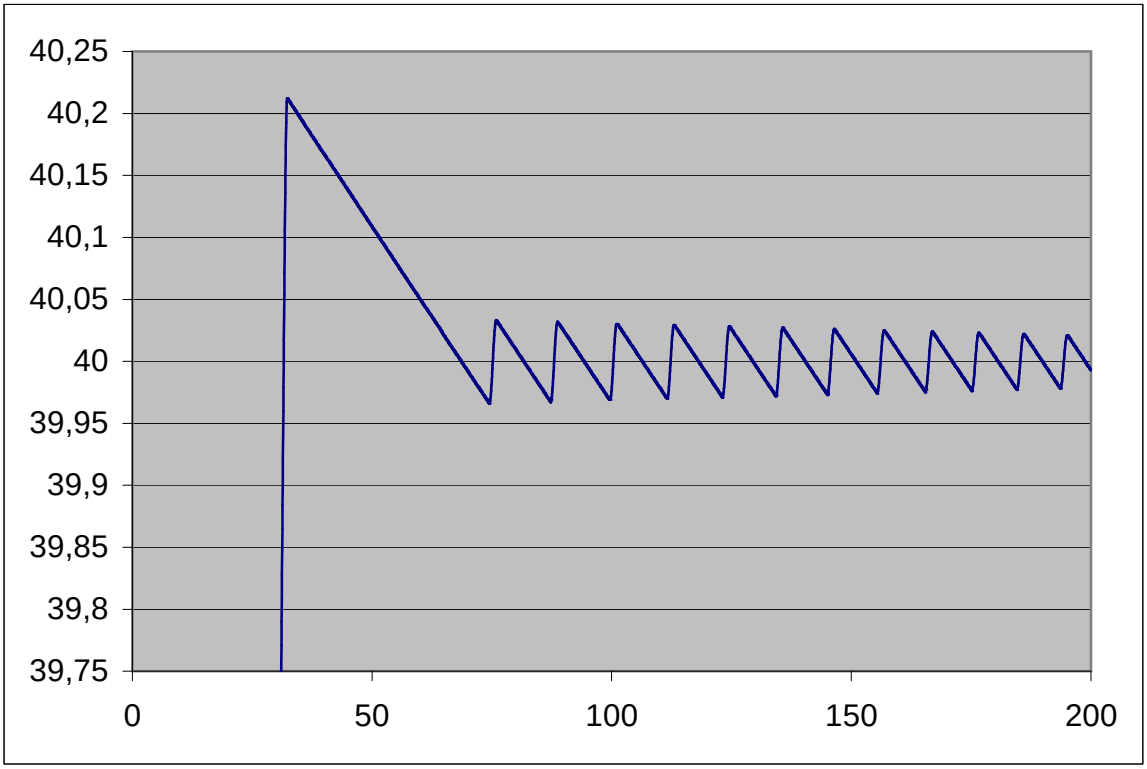


12. ábra – Szimulációs eredmények

Vizsgáljuk meg kicsit közelebbről is a 12. ábra által mutatott képet minőségileg! Arra vagyunk elsősorban kíváncsiak, hogy a víz hőmérséklete mennyire pontosan követi a referenciajelet.

A 13. ábrán kinagyítva látjuk az aktuális vízhőmérsékleti görbét. Láthatóvá válik, hogy a hőmérséklet a kezdeti tranziens után sem állandó, hanem kicsi (és csillapodó) amplitúdóval ingadozik a referenciahőmérséklet 40 °C-os értéke körül.

A túllövés, alig több mint 0.2 °C és t = 70 s után már beálltnak tekinthetjük a hőmérsékletet. Azonnal szembeötlő azonban szabályozásunk aszimmetrikus mivolta! A fűtési és a hűlési sebesség ugyanis nem egyforma, hiszen a fűtést egy P_{max} = 400 W teljesítményű fűtőszállal végezzük, a hűlést pedig csupán a környezet biztosítja a tartály falán keresztül így az jóval lassabb folyamat. Ezt mutatják a görbe negatív meredekségű részei.



11. ábra – Az aktuális hőmérsékleti görbe közelebbről

3.4. A szabályozó realizálása

A szimuláció után következő feladat a szabályozó áramkör realizálása, amely már nem csak szabályozástechnikai, hanem némi áramkörtervezési jártasságot is igényel. Erről lesz szó a későbbiekben…

Irodalomjegyzék

[1] AVR221: Discrete PID controller – ATMEL Application note

Tartalomjegyzék

1. Bevezetés 1
2. A PID szabályozó működése 1
2. 1. A PID szabályozó felépítése 1
2.2. Az arányos tag 2
2.3. Az integráló tag 3
2.4. A deriváló tag 4
2.5. A PID szabályozó digitális megvalósítása 5
3. Példa PID szabályozó alkalmazására 6
3.1. A szabályozott rendszer modellezése 6
3.2. A szabályozóval kiegészített rendszer számítógépes szimulációja 10
3.3. Szimulációs eredmények 12
3.4. A szabályozó realizálása 14
Tartalomjegyzék 15