Sisteme de reglare automata

• Obiective

- Prezentarea unui sistem de reglare automata
- Prezentarea unui sistem SCADA care contine sisteme de reglare automata
- Prezentarea unui sistem SCADA pentru controlul debitului
- Prezentarea unui sistem SCADA pentru controlul temperaturii

• Organizarea sarcinilor de lucru

• Parcurgeti cele patru capitole ale cursului.

• In cadrul fiecarui capitol urmariti exemplele ilustrative si incercati sa le realizati in medul de dezvoltare "Citect".

- Fixati principalele idei ale cursului, prezentate în rezumat.
- Completati testul de autoevaluare.
- Timpul de lucru pentru parcurgerea testului de autoevaluare este de 15 minute.

1. Prezentarea unui sistem de reglare automata

SRA - Sistemele de Reglre Automata sunt sisteme cu bucla de reactie (loop control) care functioneaza pe baza analizei in permanenta a valorii de iesire (marimii reglate), preluata prin intermediul reactiei negative. Diferenta dintre valoarea de iesire si valoarea de referinta, numita "eroare", este folosita de SRA pentru eliminarea acesteia sau mentinerea ei in anumite intervale prestabilite.



Un SRA se compune din urmatoarele elemente:

- RA Regulator automat;
- PF Partea fixata(Procesul condus);

Marimile definite:

- r Referinta;
- e Eroarea;
- u Comanda;

- v Perturbatia;
- y Variabila de proces(Marimea reglata, iesirea);

Daca definim $H_r(s)$ functia de transfer a regulatorului si $H_f(s)$ functia de transfer a partii fixate, Schema unui SRA devine:



Partea fixata(PF) contine procesul tehnologic(PT) asupra caruia actioneaza regulatorul automat(RA).

Procesul tehnologic(PT) este comandat de catre regulatorul automat(RA) prin intermediul elementelor de executie(EE). Citirea marimii reglate(y - variabila de proces) se face prin intermediul traductorilor(T) care ofera marimea x proportionla cu marimea y adica x=k*y.

Putem deci reprezenta un SRA sub forma:



Unde:

- RA Regulator automat;
- PT Procesul tehnologic;
- EE Element de Executie;
- T Traductor;

Marimile definite:

- r Referinta;
- e Eroarea;
- u Comanda;
- m Executia;
- v Perturbatia;
- y Variabila de proces(Marimea reglata, iesirea);

• x - Reactia;

Mult mai fireasca ar fi reprezentarea in care procesul tehnologic(PT) ar fi reprezentat separat, la care se adauga in mod firesc dispozitivul de automatizare. Aceasta reprezentare ar corespunde dezvoltarii firesti a sistemelor tehnologice in care initial se realizeaza un proces tehnologic dupa care se automatizeaza.



2. Sisteme de reglare automata in SCADA TIA-Portl

Un SRA de ti PID mentine valoarea iesirii y in jurul valorii de referinta r prin intermediul comenzii u. Comanda u este generata de SRA prin intermedul unei functii de transfer caracterizata de trei constante: kp,ki,kd si Ti.

- kp este constanta Proportionala (cu valori intre 0-4)
- ki este constanta Integrativa (cu valori intre 0-5)
- kd este constanta Derivativa (cu valori intre 0.2-2)
- Te este timpul de esantionare (250 ms adica timpul de Refresh al HMI)

In imaginea de jos este reprezentata comanda u (cu rosu) si iesirea y(cu verde).

Sistem de Regla	are Automata PID
Your browser does not support	ort the HTML5 canvas tag.
r:	kp:
ki:	kd:

Pentru implementarea regulatorului PID se va folosi metoda "Velocity". Conform acesti metode, comanda u din momentul k depinde de referinta r, de iesirea y si de e_v (e, din momentul k-1). Intervalul de timp dintre momentul k si momentul k-1 este Te(Timpul de esantionare).

Comanda u se obtine prin insumarea elementului proportional, integrativ (inte) si derivativ(deriv) astfel:

```
e = r - y;
inte = inte + e * Te;
deriv = (e - e v) / Te;
```

```
u = kp * e + ki * inte + kd * deriv;
e_v = e;
```

In care:

- y este iesirea din pasul curent
- r este valoareareferinta
- e este eroarea calculata (r-y) din pasul curent
- e_v este valoarea calculata pentru e in pasul anterior
- u este valoarea calculata pentru comanda din pasul curent
- kp este constanta Proportionala (cu valori intre 0-4)
- ki este constanta Integrativa (cu valori intre 0-5)
- kd este constanta Derivativa (cu valori intre 0.2-2)
- Te este timpul de esantionare (250 ms adica timpul de Refresh al HMI)

Variabila de proces adica iesirea y reprezentand totodata marimea reglata ar trebui sa provina din procesul tehnologic(PT) prin plasarea unui traductor. Folosind de exemplu functia de transfer a unui sistem de ordinul 5 si anume: $H_f(s)=1/(5s+1)$. Dupa discretizare obtinem relatia:

$$y = (u^{*}Te + 5 * y_v) / (5 + Te);$$

In care:

- y este valoarea calculata pentru iesire din pasul curent
- y_v este valoarea calculata pentru y in pasul anterior
- u este valoarea calculata pentru comanda din pasul curent
- Te este timpul de esantionare (250 ms adica timpul de Refresh al HMI)

Dupa cum se observa ea depinde de de $y_v(y \text{ din pasul anterior})$, comanda u din pasul curent si de timpul de esantionare Te.

Sistem de reglare automata PID realizat in SCADA TIA-Portal

Pe baza ecuatiilor descrise mai sus, vom realiza proiectul SCADA: <u>Sist_regl_000</u> proiect ce contine un SRA (sistem de reglare automata) de tip PID (proportional-integrativ-derivativ) discret.

In cadrul acestui proiect, vom crea pentru inceput, Screen-ul **Sra_pid_init** pentru initializarea parametrior.

SIEMENS SIMATIC HMI Seta	ire para	metri				▽ 5/12/2022 :37:47 PM
CPU 1215C DC/DC/DC	TP 700 Conf	or1	SR	A PID	Setare	parametri
	Ref	Кр	Ki	Kd	Te	
	500 (000) 450 (000) 400 (000) 350 (000) 350 (000) 300 (000) 300 (000) 300 (000) 300 (000) 300 (000)	000 политически 800 политически 600 политически 400 политически 200 политически 0	500 ници 400 ници 300 ници 200 ници 100 ници 0	200	500 400 300 200 100 0	
	400	75	120	1	5	
	400	0.75	1.20	0.01	0.05	Sist_regl_000

Pentru a crea acest Screen, avem nevoie de tag-urile: Kp, Ki, Kd, Te de tip "real" si r (Refr:set point)bde tip "int".

Obiectele de tip "Slider" utilizate pentru setarea parametrilor, nu admit decat atribuirea de tag-uri de tip "int".

Vom defini tag-urile: Kp_int, Ki_int, Kd_int, Te_int de tip "int" dupa care vom calcula valoarea tag-urilor: Kp, Ki, Kd, Te prin impartirea acestora la 100 daca dorim doua zecimale de exemplu.

Pe evenimentul "Change al fiecarui obiect "Slider" vom atribui functia "sra_init_p()"

```
Sub init p()
       If r=0 Then
               SmartTags("Kp int")=75
               SmartTags("Ki int")=120
               SmartTags("Kd int")=1
               SmartTags("Te int")=5
               SmartTags("r")=400
       End If
       SmartTags("y")=0
       SmartTags("y_v")=0
       SmartTags("inte")=0
       SmartTags("deriv")=0
       SmartTags("e")=0
       SmartTags("e v")=0
       SmartTags("u")=0
       SmartTags("Kp") = SmartTags("Kp int") /100
       SmartTags("Ki") = SmartTags("Ki_int")/100
       SmartTags("Kd") = SmartTags("Kd int")/100
       SmartTags("Te")=SmartTags("Te int")/100
       SmartTags("r scal")=SmartTags("r")/10
End Sub
```

Implementarea algoritmului de reglre automata in HMI

Pentru inceput vom implementa algoritmul pentru SRA in HMI.

Vom crea in continuare, in cadrul acestui proiect, Screen-ul **Sra_pid_hmi** unde vom afisa grafic evolutia in timp a marimilor u,y,r, folosind obiectul "Trend View". Obiectul "Trend View" admite adaugarea de trend-uri dar tag-urile corespunzatoare trebuie sa fie scalate in domeniul 0-100. Vom introduce pe linga Tagurile u,y,r tag-urile u_scal, y_scal, r_scal scalate in domeniul 0-100

HMI	tags			
1	lame 🔺	Tag table	Data type	Connection
-0	Contor	Default tag table	UInt	HMI_Connect
	deriv	Default tag table	Real	HMI_Connect
	e	Default tag table	Real	HMI_Connect
-	e_v	Default tag table	Real	HMI_Connect
-01	inte	Default tag table	Real	HMI_Connect
	Kd	Default tag table	Real	HMI_Connect
-	Kd_int	Default tag table	Int	<internal tag<="" td=""></internal>
-0	Кі	Default tag table	Real	HMI_Connect
-	Ki_int	Default tag table	Int	<internal tag<="" td=""></internal>
	Кр	Default tag table	Real	HMI_Connect
-	Kp_int	Default tag table	Int	⊲nternal tag
-00	mod	Default tag table	UInt	HMI_Connect
-0	r	Default tag table	Int	HMI_Connect
-	r_scal	Default tag table	Int	HMI_Connect
-	Tag_ScreenNumber	Default tag table	UInt	⊲nternal tag
	Те	Default tag table	Real	HMI_Connect
-	Te_int	Default tag table	Int	<internal tag<="" td=""></internal>
	u	Default tag table	Real	HMI_Connect
-	u_scal	Default tag table	Int	HMI_Connect
	У	Default tag table	Real	HMI_Connect
	y_scal	Default tag table	Int	HMI_Connect
-00	y_v	Default tag table	Real	HM_Connect

Algoritmul pentru implementarea SRA presupune reglarea in mai multe iteratii.

Pe evenimentul "Load" al screen-ului "Sra_pid_hmi" se vol lansa functiile "init_p()" si "pid_hmi()".



Functia "pid_hmi()" implementeaza algoritmul PID.

```
Sub pid hmi()
    Dim inte, deriv,tm
       tm=Timer()
       Do While SmartTags("Tag ScreenNumber")=3
               If Timer > tm+0.01 Then
                       e = r - y
                       inte = inte + e * Te
                       deriv = (e - e v) / Te
                       u = Kp * e + Ki * inte + Kd * deriv
                       e_v = e
                       y = (u*Te + 5 * y_v) / (5 + Te)
                       y_v=y
                       u_scal=u/10
                       y scal=y/10
                       r scal=r/10
                       tm=Timer()
               End If
       Loop
End Sub
```

Implementarea algoritmului de reglre automata in PLC

Implementarea algoritmul de reglre automata in HMI, nu este recomandat deoarece functionarea procesului reglat depinde de HMI care este mult mai lent. Pe de alta parte HMIul trebuie sa fie tot timpul pornit, functionarea procesului depinzand de HMI. Solutia mai eficienta consta in implementarea algoritmului de reglre automata in PLC.

	PLC ta	gs					
	N	lame	Tag table	Data type	Address	Retain	Acces
1	-0	Contor	Default tag table	UInt	%MW0		
2	-	e	Default tag table	Real	%MD4		
3		e_v	Default tag table	Real	%MD8		
4	-	Kd	Default tag table	Real	%MD12		
5	-	Ki	Default tag table	Real	%MD16		
6	-	Кр	Default tag table	Real	%MD20		
7		Те	Default tag table	Real	%MD24		
8		u	Default tag table	Real	%MD28		
9		у	Default tag table	Real	%MD32		
10		y_v	Default tag table	Real	%MD36		
11	-	r	Default tag table	Int	%MW40		
12	-	r_scal	Default tag table	Int	%MW42		
13		u_scal	Default tag table	Int	%MW44		
14		y_scal	Default tag table	Int	%MW46		
15		inte	Default tag table	Real	%MD48		
16		deriv	Default tag table	Real	%MD52		
17		mod	Default tag table	UInt	%MW56		

Vom crea in continuare, in cadrul acestui proiect, Screen-ul **Sra_pid_plc** si vom implementa functia de reglare in PLC. Avem nevoie de urmatoarele tag-uri PLC:

Vom defini in Program Block -- Cyclic Interrupt --urmatoarea rutina:

```
"Contor" := "Contor" + 1;
IF "Contor" > 999 THEN
    "Contor" := 0;
END IF;
IF \overline{mod} = 2 THEN
    "e" := INT TO REAL("r") - "y";
    "inte" := "inte" + "e" * "Te";
    "deriv" := ("e" - "e v") / "Te";
    "u" := "Kp" * "e" + "Ki" * "inte" + "Kd" * "deriv";
    "e v" := "e";
    "y" := ("u" * "Te" + 5 * "y v") / (5 + "Te");
    "y v" := "y";
    "u scal" := REAL TO INT("u" / 10);
    "y scal" := REAL TO INT("y" / 10);
    "r scal" := REAL TO INT("r" / 10);
END_IF;
```

Toate variabilele utilizate in rutina de sus trebuiesc initializate, vom defini in Program Block -- Startup -- urmatoarea rutina:

"mod" := 0;
"Contor" := 0;
"Kp" := 0.75;
"Ki":= 1.2;
"Kd":= 0.01;
"r":= 400;
"r_scal" := 0;
"e" := 0;
"e v" := 0;
"y" := 0;
"y v" := 0;
"r" := 400;
"y_scal" := 0;
"inte":= 0;
"deriv":= 0;
"u":= 0;
"u scal":= 0;

Putem crea acum creen-ul Sra_pid_plc.



Pe evenimentul "Load" al screen-ului "Sra_pid_hmi" se va lansa functia "init_p()" si vom seta Tag-ul Mod=2, lansandu-se astfel functia de reglare din PLC.

Sistem de reglare automata a nivelului unui lichid

Urmatorul screen **Sra_pid_nivel** foloseste functia de reglare din PLC pentru a regla nivelul de lichid dintr-un tank.

Pe evenimentul "Load" al screen-ului "Sra_pid_nivel" se va lansa functia "init_p()" si va seta Tag-ul Mod=2



3. Utilizarea unui bloc PID Compact Capitol realizat cu contributia stud: Iulia Coros

Vom folosi in continuare blocul PID Compact pentru a realiza proiectul: <u>Sist_regl_001</u> proiect ce contine un SRA de tip PID Compact.

Configurare bloc PID Compact

Pasul 1 – Adăugăm în proiectul nostru un nou bloc de tipul Cyclic interrupt în limbaj Leader și setăm Cyclic time pe 100 de milisecunde.



lame:			
yclic interrupt_2			
Organization block	 Program cycle Startup Time delay interrupt Cyclic interrupt Hardware interrupt Time error interrupt Diagnostic error interrupt Pull or plug of modules 	Language: Number: Cyclic time (ms): Description:	LAD LAD Automatic
Function block	 Rack or station failure Time of day Status Update Profile MC-Interpolator MC-Servo MC-PreServo MC-PostServo 	A "Cyclic interrupt" (programs at period independently of cy The intervals can be in the properties of	value between 1 t i and 60000. clic program execution. e defined in this dialog or the OB.
Data block		more	
Additional inform	nation		

Pasul 2 – În acest bloc creat adăugăm blocul PID Compact pe care îl găsim la categoria Technology.

In	structions		🗖 🗉 🕨
0	ptions		
	ini tin	5 [°] 5⊡ 1	
>	Favorites		
>	Basic instructions		
>	Extended instructions		
~	Technology		
Na	me	Description	Version
•	Counting TID Control		V1.1
	 Compact PID 		<u>V6.0</u>
	PID_Compact	Universal PID controller	V2.3
	PID_3Step	PID controller with inte	V2.3
	= PID_Temp	PID controller for temp	V1.1
	Auxiliary functions		V1.2
	Motion Control		<u>V7.0</u>
	SINAMICS		V10

Pasul 3 – Ne creăm în tabelul de tag-uri următoarele variabile.

	1	Name	Data type	Address	Retain	Acces	Writa	Visibl
1		SetPoint	Int	%MW0				
2	-00	Output_val	Real	%MD2				
3		State	Int	%MW4				
4	-00	Input	Int	%MW6				

Atenție! Variabila pe care dorim să o atribuim ieșirii (Output) trebuie să fie de tip REAL.

Atribuim aceste tag-uri blocului PID în felul următor:



Putem vedea că regulatorul a fost adăugat în folderul System blocks și în Technology Objects.



Pasul 4 – Pentru meniul de configurare a regulatorului apăsăm pe această iconiță:



Din aceste setări de configurare putem selecta tipul de regulator în funcție de parametrul pe care dorim să îl reglăm.



Foarte important! Regulatorul funcționează în modul automat doar după ce pornim simularea și dăm un restart (Stop CPU – Start CPU) unității centrale de procesare de pe butoanele:

😤 🖬 🖽							
 Basic settings 	0	and the second second					
Controller type	0	Controller type					
Input / output parameters	0						
 Process value settings 	0	Volume		1	-		
Process value limits	0	Invert control	logic				
Process value scaling	0		logic				
 Advanced settings 	0	Activate Mode	after CPU restart				
Process value monitoring	0		Set Mode to:	Automatic n	node	-	
PWM limits	0						
Output value limits	0						
PID Parameters	0						

Alegem tipul intrării și ieșirii pe care dorim să le folosim.

🕶 Basic settings 🛛 🥪		
Controller type 🤣	Input / output parameters	
Input / output parameters 🥪	-	
🛛 Process value settings 🛛 🥑	Setpoint:	
Process value limits 🛛 🥪		
Process value scaling 📀		
🗸 Advanced settings	Input:	Output:
Process value monitoring 🥪	Input 🚽 📙	/ Output 💌
PWM limits 🥪	Input	
Output value limits 🤣	Input_PER (analog)	
PID Parameters 🤣		



Setăm limitele superioare și inferioare ale semnalelor de intrare și ieșire.

2 6 8	
Basic settings Controller type	rocess value limits
Process value settings Process value limits Process value scaling Advanced settings Process value monitoring PWM limits Output value limits	Process value high limit: 10010 1
PID Parameters	Process value low limit: 0.0 I
PID → PLC_1 [CPU 1215C D	C/DC/DC] Technology objects PID_Compact_1 [DB1]
PID → PLC_1 [CPU 1215C C	C/DC/DC] → Technology objects → PID_Compact_1 [DB1]
PID → PLC_1 [CPU 1215C D Basic settings Controller type Input fourtout parameters	C/DC/DC] > Technology objects > PID_Compact_1 [DB1]
PID → PLC_1 [CPU 1215C D Basic settings Controller type Input / output parameters Process value settings Process value limits Process value settings	C/DC/DC] > Technology objects > PID_Compact_1 [DB1] Output value limits Output value limits
PID ➤ PLC_1 [CPU 1215C D Basic settings Controller type Input / output parameters Process value settings Process value settings Process value settings Process value monitoring Process value monitoring PWM limits Output value limits	C/DC/DC] > Technology objects > PID_Compact_1 [DB1] Output value limits Output value limits Output value high limit: 100.0 %

Tot din acest meniu putem seta manual parametri regulatorului sau schimba tipul acestuia (PID sau PI).

Basic settings Controller type	00	PID Parameters	_	
 Process value settings Process value limits 	00	Enable manual entry		
Process value scaling	õ	Proportional gain:	1.0	
Advanced settings	0	Integral action time:	20.0	s
Process value monitoring		Derivative action time:	0.0	5
PWM limits	2	Derivative delay coefficient:	0.0	
PID Parameters	2	Proportional action weighting:	0.0	
ino i oranneters	×	Derivative action weighting	0.0	
		Sampling time of PID algorithm:	1.0	5
		Tuning rule		
		Controller structure:	PID	
	10		PID	

Exemplu de utilizare – Sistem de reglare automată a turației unei turbine (SRAV)



Am adăugat blocul PID Compact și i-am setat ca și intrare nivelul apei din baraj, referința este turația turbinei pe care dorim să o menținem mereu la o valoare constantă, și la ieșire controlăm deschiderea valvei de admisie.



Așadar, noi vrem să menținem o turație constantă a turbinei și pentru acest lucru trebuie să reglăm deschiderea valvei de admisie în funcție de nivelul apei din baraj. Cu cât nivelul apei din baraj este mai mare, cu atât presiunea este mai mare și asta înseamnă ca forța cu care apa împinge lamelele turbinei este mai mare. Deci modelând deschiderea valvei, reglăm forța cu care apa împinge lamelele turbinei și aceasta se va învârti cu o turație constată.

Dacă creștem nivelul apei din baraj, deschiderea valvei trebuie să scadă.

Dacă scădem nivelul apei din baraj, deschiderea valvei trebuie să crească.