X SICEL 2021

Transición energética en la 4ta revolución industrial

Universidad Tecnológica de Pereira

UNIVERSIDAD NACIONAL DE COLOMBIA

XSICEL 2021
Transición energética en la 4ta revolución industrial

Temperature Control Using The Simulink PLC Coder and The IEC 61131 Standard

Authors: Barrera Cuestas Andrés Felipe, Mantilla Castañeda Marlon Giovany, Giral Ramírez Diego Armando, Montoya Giraldo Oscar Danilo

Institutions: Universidad Distrital Francisco José de Caldas

Contents

I. Introduction

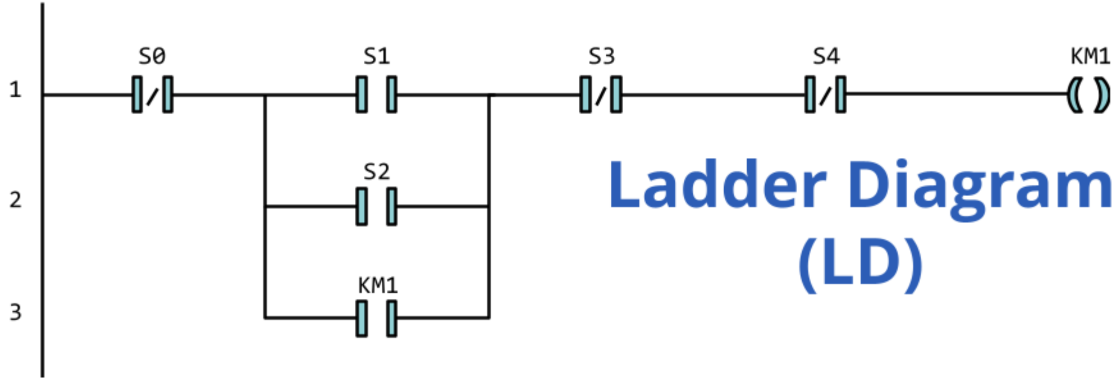
```
PROGRAM PLC_PRG
VAR
        S0 : BOOL;
        S1 : BOOL;
        S2 : BOOL;
        KM1 : BOOL;
        KM2 : BOOL;
        Timer01 : TON;
        Timer02 : TON;
END_VAR
```

II. Theoretical aspects

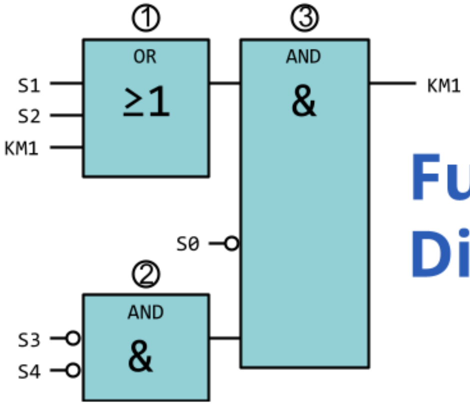
IEC

8 SECTIONS

II. Theoretical aspects



Ladder Diagram (LD)



Function Block Diagram (FBD)

LDN	S0
ANDN	S3
ANDN	S4
AND (	S1
OR	S2
OR	KM1
)	
ST	KM1

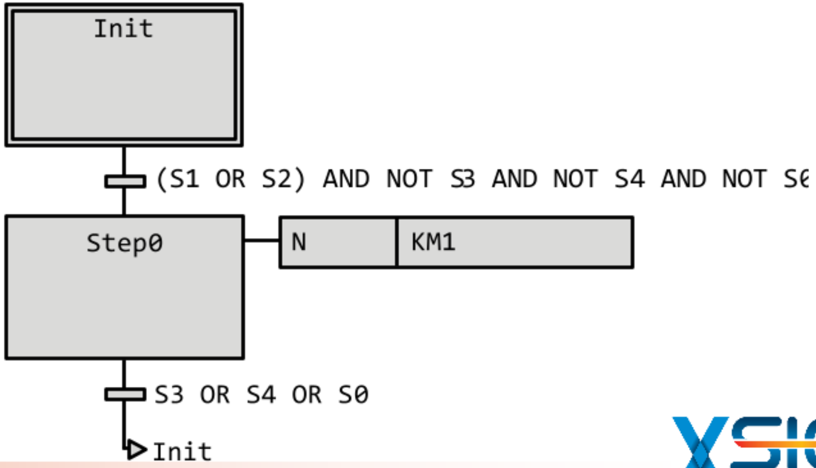
Instruction List (IL)

II. Theoretical aspects

```
IF S0 OR S3 OR S4 THEN

    KM1:=FALSE;

ELSIF (S1 OR S2) AND (NOT S0 OR NOT S3 OR NOT S4) THEN

    KM1:=TRUE;

END_IF
```

Structured Text (ST)

Sequential Function Chart (SFC)



III. Proposed methodology

Stage 1: Discrete time transfer function

Plant characterization

System identification toolbox

Transfer function model

correspondence < 90%

No

Yes

Check behavior

No

Efficient behavior?

Yes

End stage 1

Stage 2: PID controller parameters

PID Tuner

Load transfer function

Set control, speed and aggressiveness

Get Kp, Ki and Kd

End stage 2

Stage 3: Plant and control simulation

Simulink

Implement plant and control

Check plant and generate disturbances

Correct operation?

No

Stage 1: Discrete time transfer function

Yes

Get Kp, Ki and Kd

End stage 3

Stage 4: Simulink PLC Coder

Generate subsystem of the architecture

PLC Coder

Check plant and generate disturbances

Path to save the generated code

Set up automation software

Run PLC coder compiler

End stage 4

Stage 5: PLC coder environment Codesys

Run Codesys

Create SFC project

Import the code from PLC coder

1

Perform adjustment

End stage 5

Stage 6: PLC coder environment Codesys

Output parameters to compare Codesys and Simulink

No

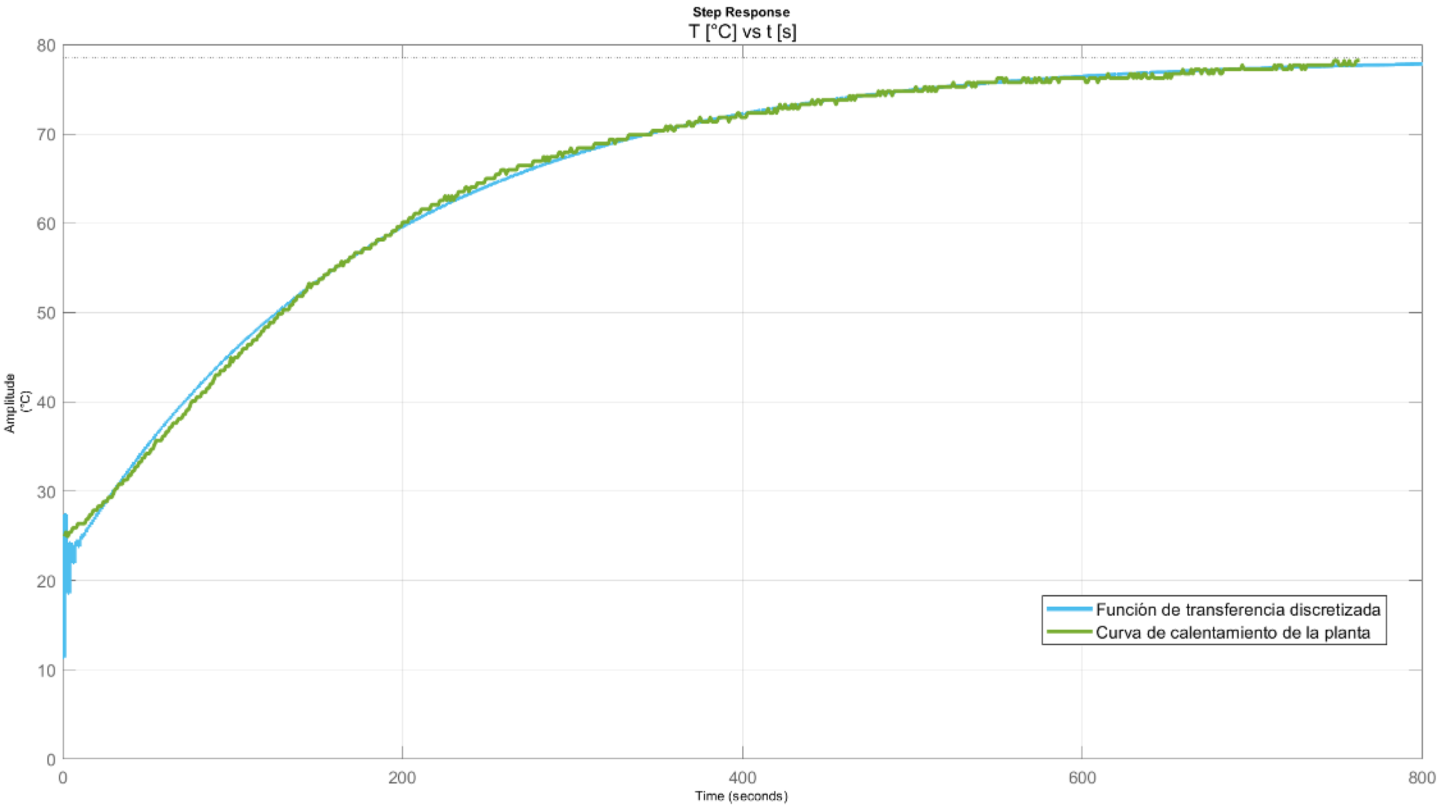
Correct operation?

1

Yes

End stage 6

IV. Results



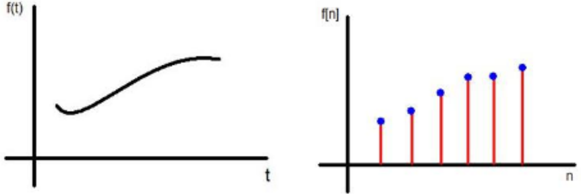
Temperature Range

78,2°C

24,93°C

IV. Results

Toolbox **System Identification**
Command **c2d**

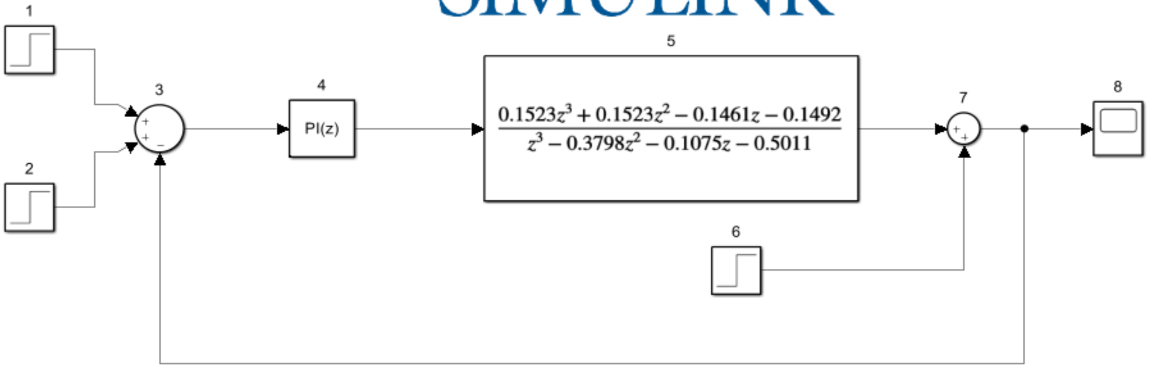


Toolbox **pidtool**

PI Controller

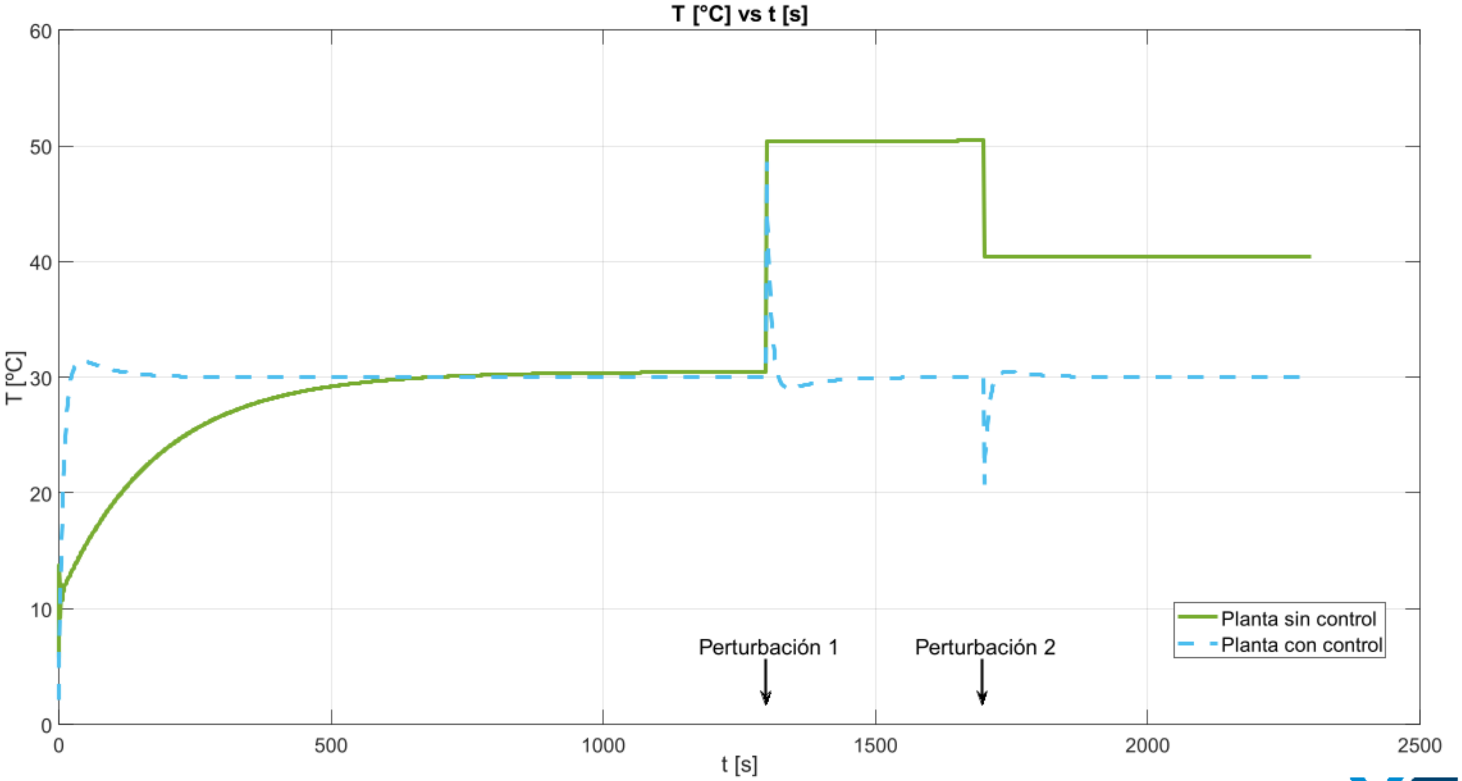
Kp	0,49989
Ki	0,45537

SIMULINK

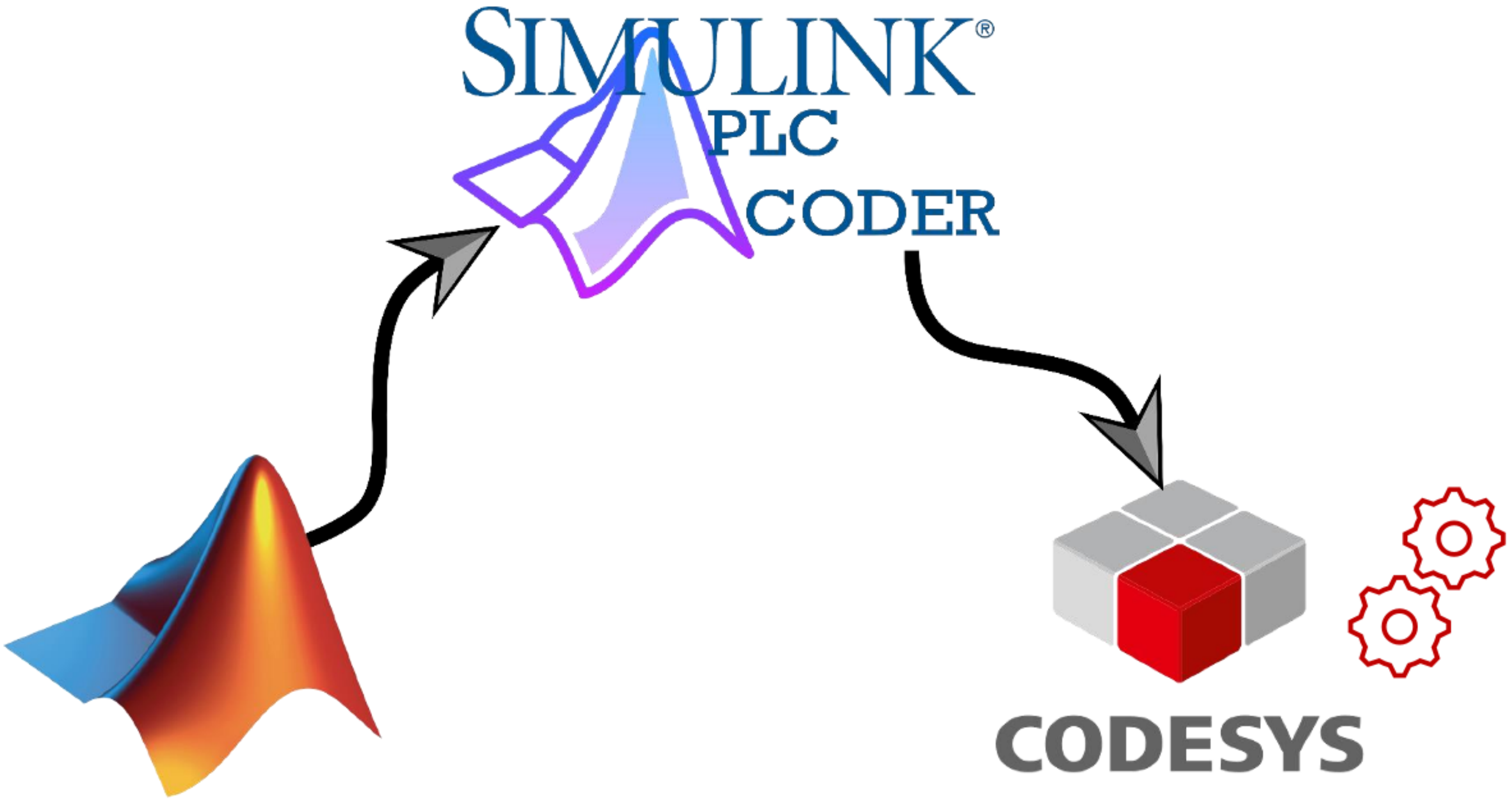


IV. Results

Plant without control and with control PI



IV. Results



IV. Results

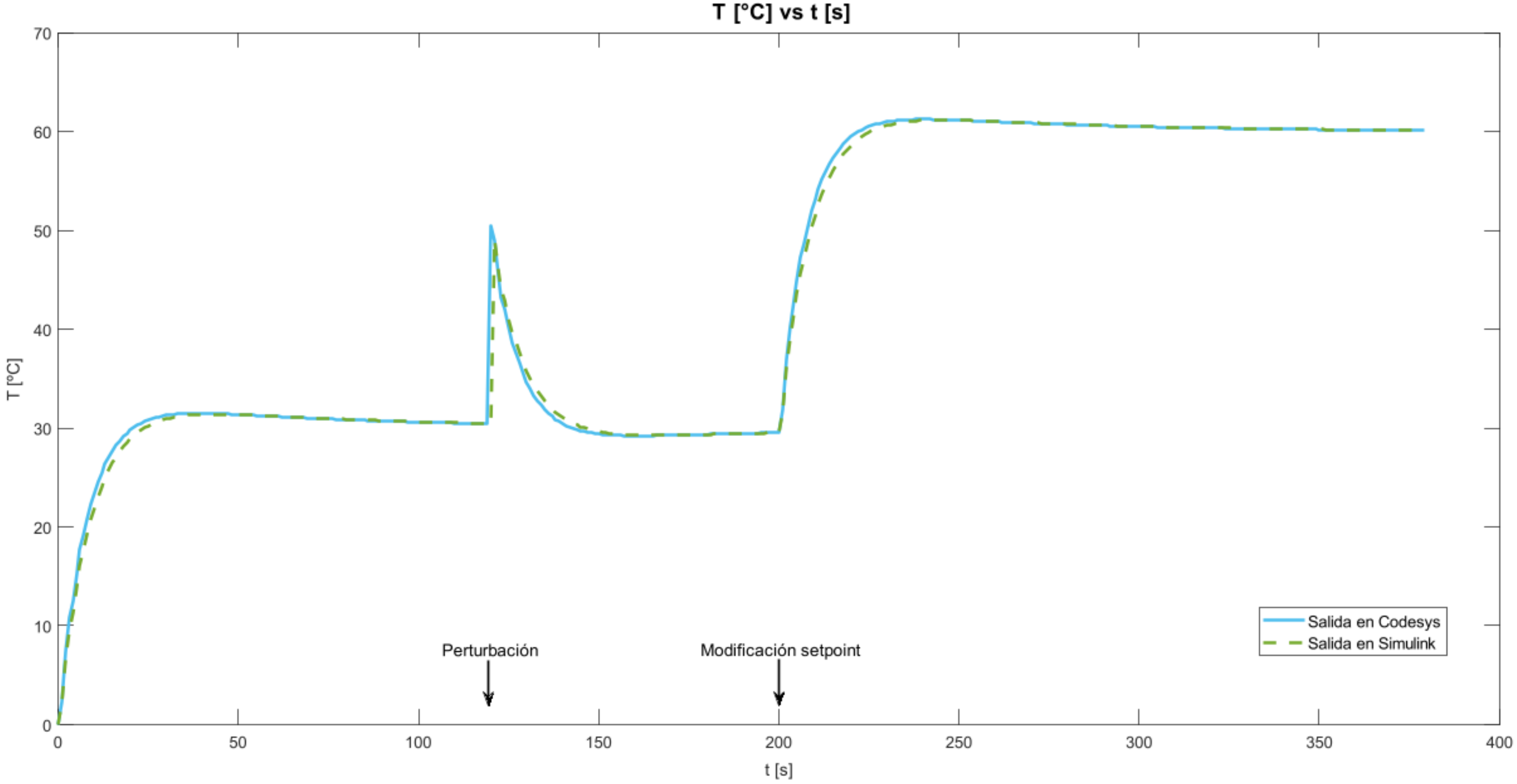
File: arq_planta.st

```
(*
 *
 * File: arq_planta.st
 *
 * IEC 61131-3 Structured Text (ST) code generated for subsystem "arq_planta/Subsystem"
 *
 * Model name                      : arq_planta
 * Model version                   : 1.8
 * Model creator                   : AXONGROUP
 * Model last modified by          : AXONGROUP
 * Model last modified on          : Sat Apr 10 23:45:21 2021
 * Model sample time               : 0s
 * Subsystem name                  : arq_planta/Subsystem
 * Subsystem sample time           : 1s
 * Simulink PLC Coder version      : 3.3 (R2020b) 29-Jul-2020
 * ST code generated on            : Sun May 16 22:02:00 2021
 *
 * Target IDE selection            : 3S CoDeSys 3.5
 * Test Bench included             : No
 *
 *)
FUNCTION_BLOCK Subsystem
VAR_INPUT
    ssMethodType: SINT;
    u: LREAL;
    In2: LREAL;
END_VAR
VAR_OUTPUT
    Out1: LREAL;
END_VAR
VAR
    Integrator_DSTATE: LREAL;
    DiscreteTransferFcn_states: ARRAY [0..2] OF LREAL;
END_VAR
VAR_TEMP
    rtb_DiscreteTransferFcn: LREAL;
    DiscreteTransferFcn_tmp: LREAL;
END_VAR
CASE ssMethodType OF
    SS_INITIALIZE:
        Integrator_DSTATE := 0.0;
        DiscreteTransferFcn_states[0] := 0.0;
        DiscreteTransferFcn_states[1] := 0.0;
        DiscreteTransferFcn_states[2] := 0.0;
    SS_STEP:
        DiscreteTransferFcn_tmp := ((((0.49989 * u) + Integrator_DSTATE) - (-0.3798 * DiscreteTransferFcn_states[0])) - (-0.1075 * DiscreteTransferFcn_states[1])) - (-0.5011 * DiscreteTran
        rtb_DiscreteTransferFcn := (((0.1523 * DiscreteTransferFcn_tmp) + (0.1523 * DiscreteTransferFcn_states[0])) + (-0.1461 * DiscreteTransferFcn_states[1])) + (-0.1492 * DiscreteTransf
        Integrator_DSTATE := (0.45537 * u) + Integrator_DSTATE;
        DiscreteTransferFcn_states[2] := DiscreteTransferFcn_states[1];
        DiscreteTransferFcn_states[1] := DiscreteTransferFcn_states[0];
        DiscreteTransferFcn_states[0] := DiscreteTransferFcn_tmp;
        Out1 := rtb_DiscreteTransferFcn + In2;
END_CASE;
END_FUNCTION_BLOCK
VAR_GLOBAL CONSTANT
    SS_INITIALIZE: SINT := 0;
    SS_STEP: SINT := 1;
END_VAR
```

Resulting code Simulink PLC Coder

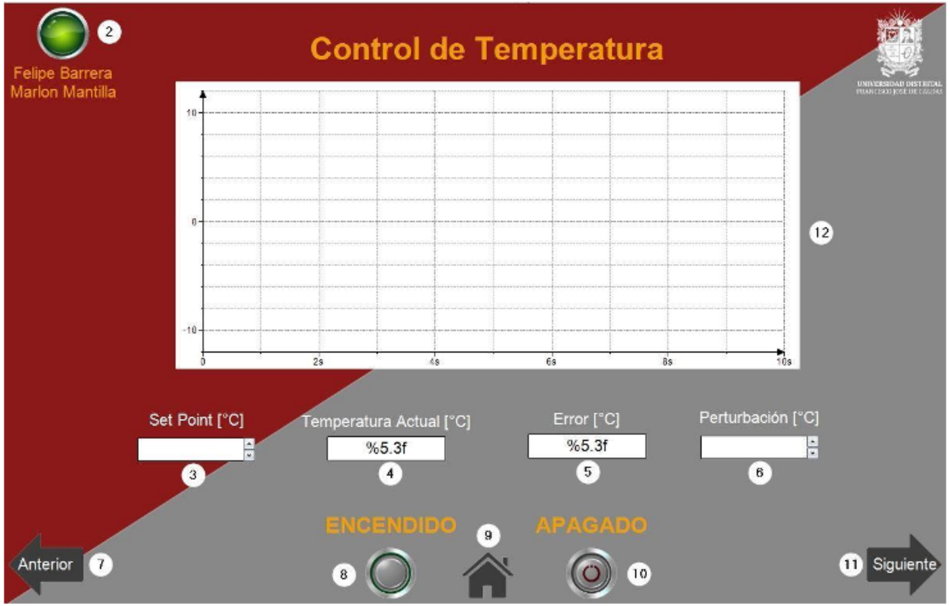
IV. Results

Comparison of Codesys and Matlab results



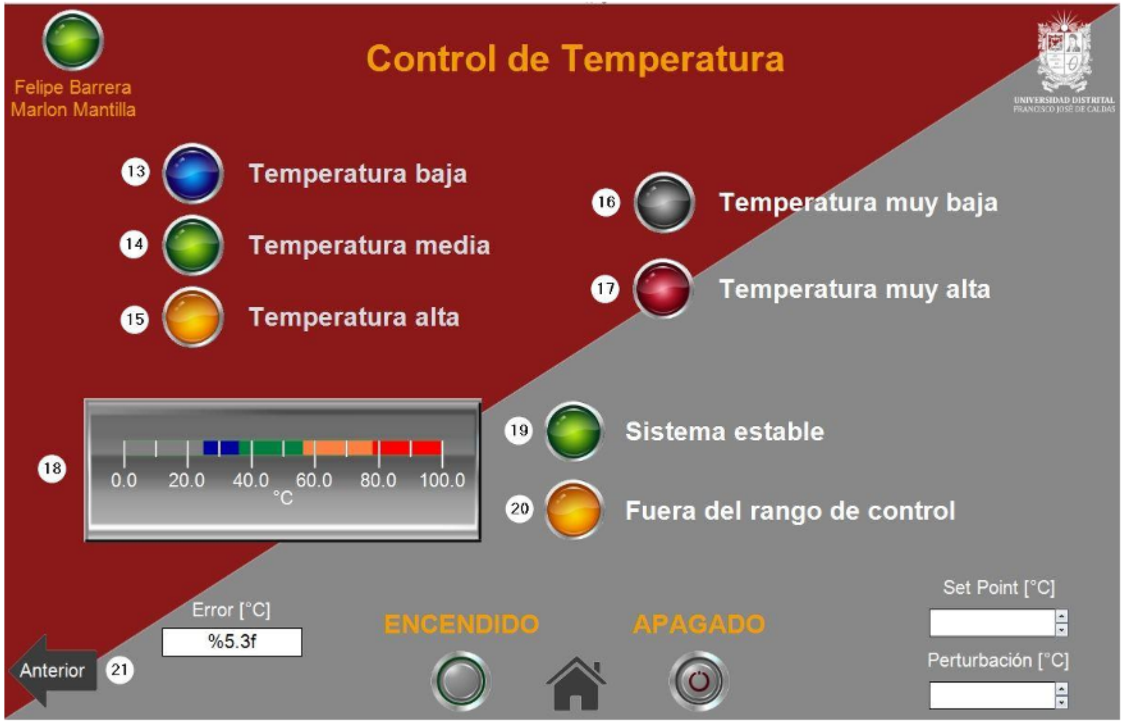
IV. Results

Human Machine Interface (HMI)



IV. Results

Human Machine Interface (HMI)



Very low temperature: T < 24,93°C, for more tan 20 seconds the temperatura control will be deactivated.

Low temperature: 24,93°C c T < 36°C.

Average temperature: 36°C ≤ T < 56°C.

High temperature: 56°C ≤ T ≤ 78,2°C

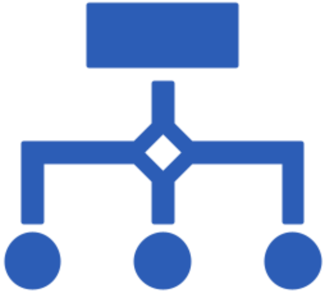
Very high temperatura: T > 78,2°C, for more than 20 seconds the temperatura control will be deactivated.

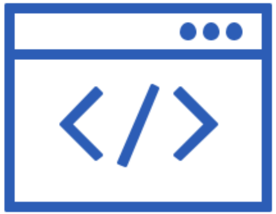
Stable system: Error ≤ 0,5°C for 15 segundos.

Out of control range: Set point is outside the control range (24,93°C ≤ T ≤ 78,2°C), temperature control will not be performed.

IV. Conclusions

With the study of the programming languages presented in the **IEC 61131-3** standard, it can be highlighted the facilities that brings to develop a project in high-level programming languages (**ST** and **SFC**). Regarding **ST**, the simplicity in the declaration of selection and iteration type sentences is highlighted, as well as the easy adaptability for a programmer who does not know this language. On the other hand, **SFC** highlights the ease of developing projects where sequential, combinational or parallel systems predominate, in addition to integrating the possibility of integrating programming languages and taking visual control in any of its transitions and stages.





IV. Conclusions

The comparison between languages allows establishing that **ST** turns out to be the best alternative when applying it in a real project due to advantages such as a compilation process and code execution in less cycles and time, a light output file size, reduced use of auxiliary variables and, above all, it prioritizes the consumption of resources by the process to be developed.

On the other hand, in spite of its easy implementation, the late compilation and execution of the programming code and the weight of the project file means that **SFC** is not seen as a viable alternative when implementing it in an industrial process where the appropriate use of PLC resources is a priority.

IV. Questions