ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

CARRERA DE INGENIERÍA EN ELECTRÓNICA E INSTRUMENTACIÓN

"TECNOLOGÍAS DE BAJO COSTO PARA LA IMPLEMENTACIÓN DE CONTROL AVANZADO EN UN PROCESO INDUSTRIAL"

Autores:
Jorge Luis Buele León
John Javier Espinoza Mejía

Director:
Ing. Marco Pilatásig

CONTENIDO

- Planteamiento del problema
- Objetivos
- Caso de Estudio
- Propuesta
- Análisis de Resultados
- Conclusiones y Recomendaciones

PLANTEAMIENTO DEL PROBLEMA

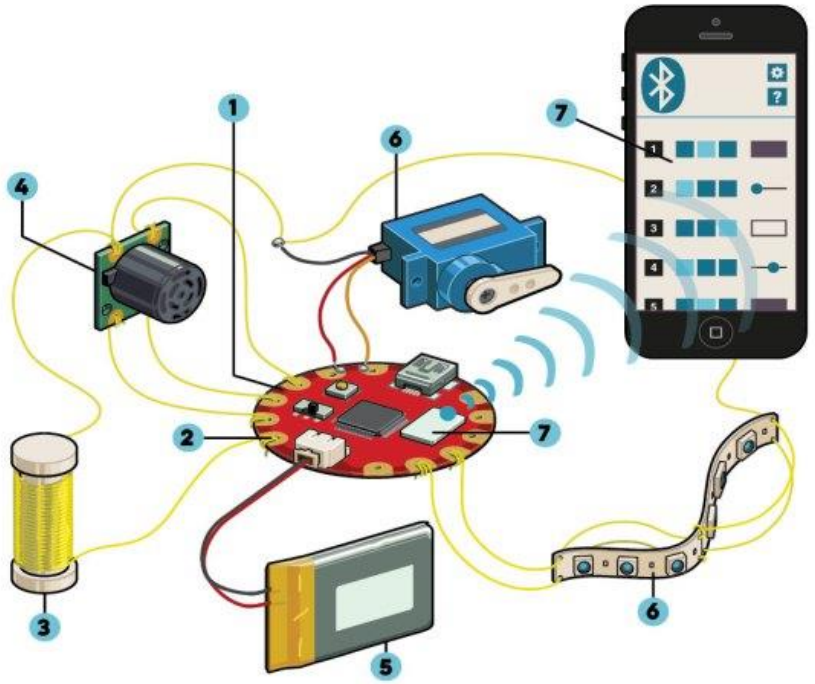
- Dispositivos industriales a altos costos.
- Necesidad del uso de software propietario.
- Limitaciones en el campo académico.
- Poca información sobre las tecnologías de bajo costo.

CONTENIDO

- Planteamiento del problema
- Objetivos
- Caso de Estudio
- Propuesta
- Análisis de Resultados
- Conclusiones y Recomendaciones

OBJETIVO GENERAL

Analizar las tecnologías de bajo costo para su uso en la implementación de control avanzado en un proceso industrial.



OBJETIVOS ESPECÍFICOS

- Investigar sobre las tecnologías de bajo costo y controles avanzados que se usan en el control automático de procesos industriales.
- Implementar un algoritmo de control avanzado usando tecnologías de bajo costo.
- Analizar las curvas de respuesta obtenidas de las diferentes tecnologías implementadas, así como las principales características técnicas y prestaciones que cada una ofrece.
- Determinar cuál de las tecnologías usadas presenta un mejor desempeño con el control avanzado implementado.

CONTENIDO

- Planteamiento del problema
- Objetivos
- Caso de Estudio
- Propuesta
- Análisis de Resultados
- Conclusiones y Recomendaciones

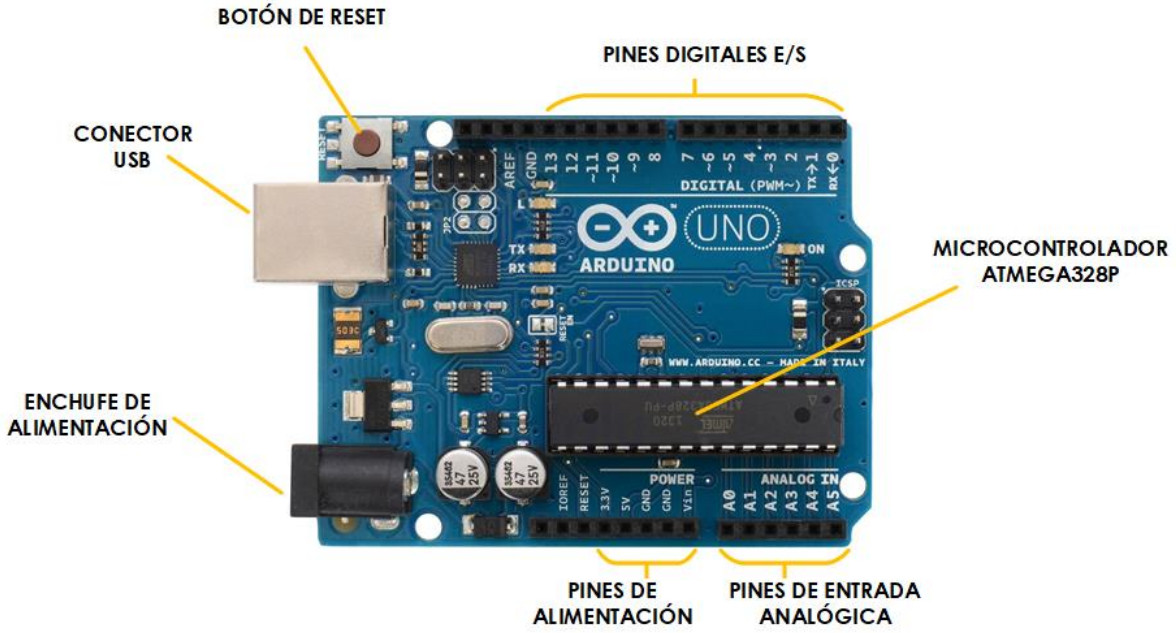
CASO DE ESTUDIO

ESTACIÓN DE FLUJO DE CAUDAL

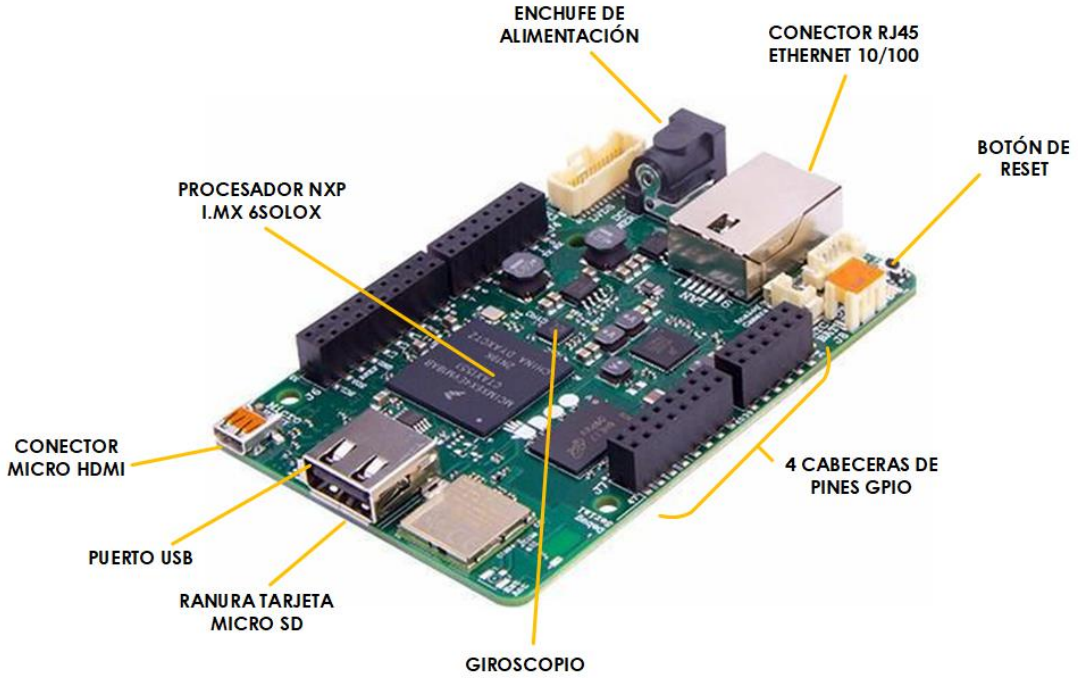
CASO DE ESTUDIO

TECNOLOGÍAS DE BAJO COSTO

ARDUINO UNO R3



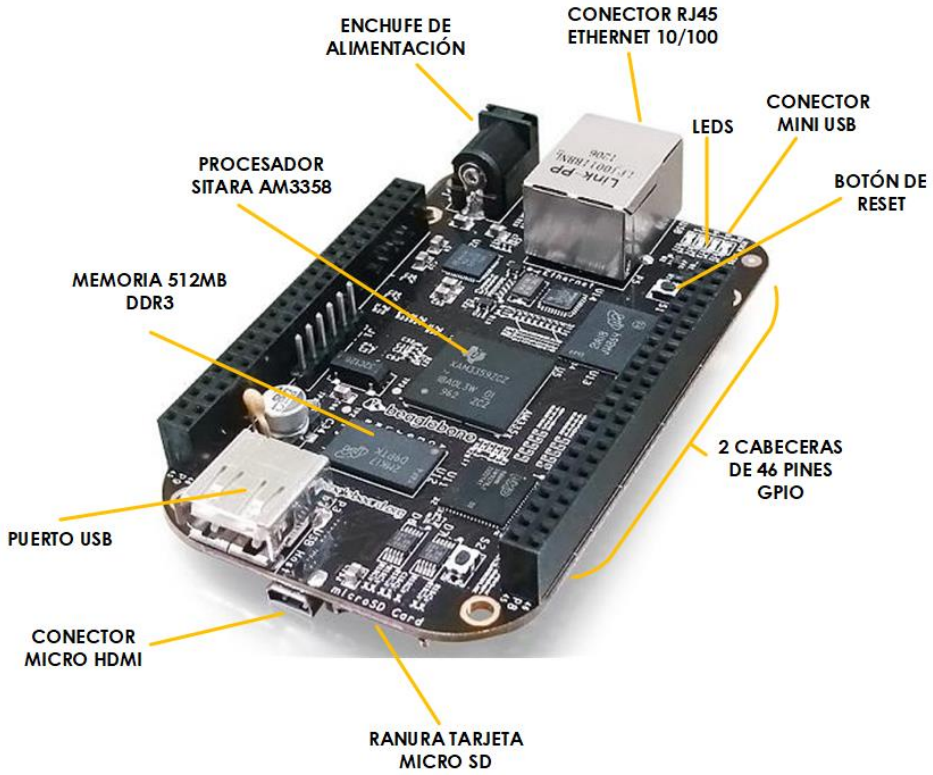
UDOO NEO FULL



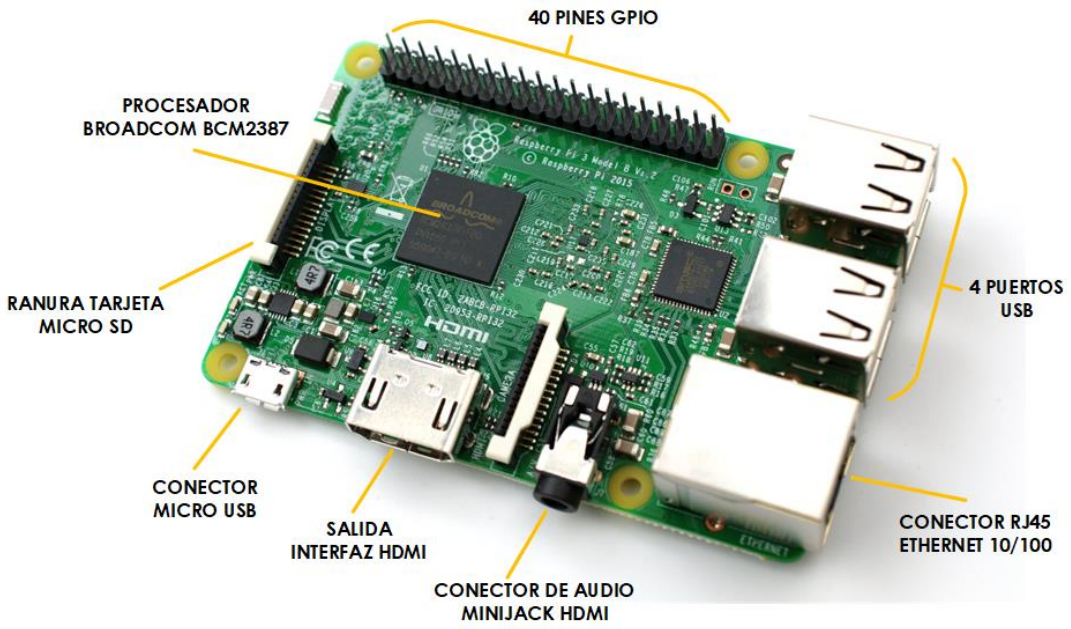
CASO DE ESTUDIO

TECNOLOGÍAS DE BAJO COSTO

BEAGLEBONE BLACK



RASPBERRY PI 3



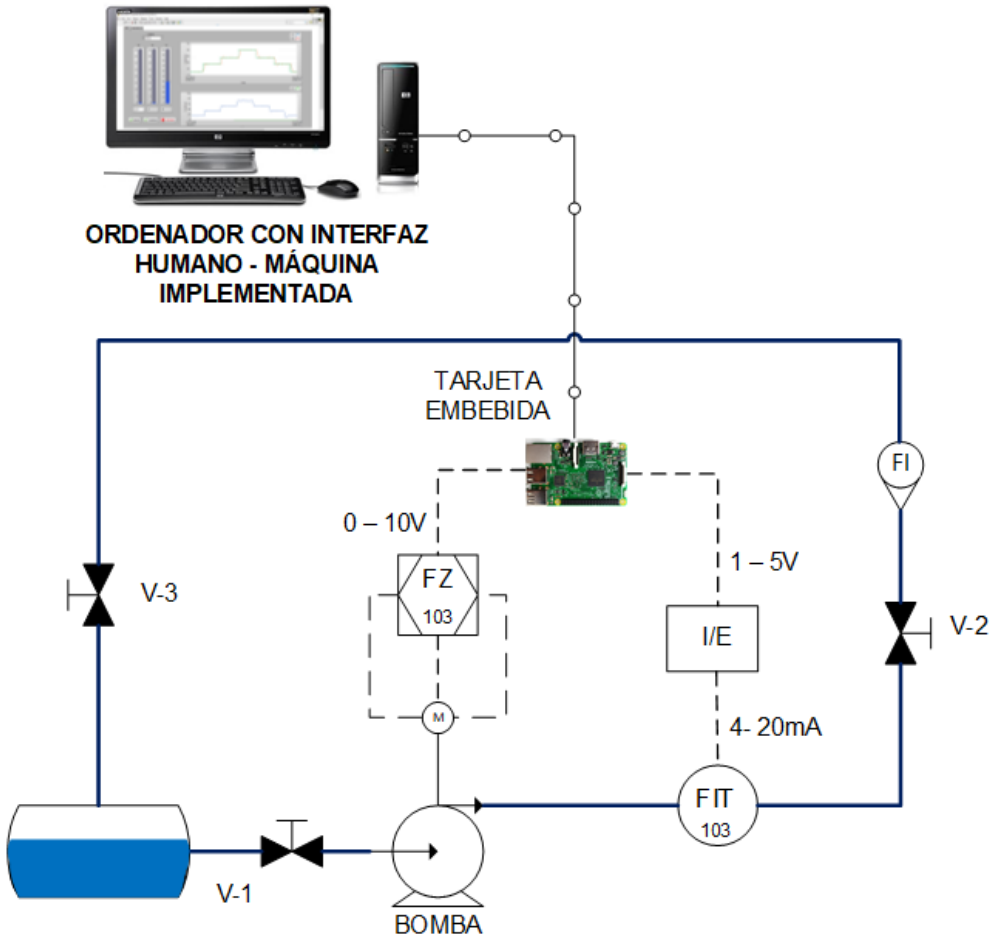
CASO DE ESTUDIO

COMPONENTE	ARDUINO UNO R3	RASPBERRY PI 3	BEAGLEBONE BLACK	UDOO NEO FULL
Procesador	Microcontrolador ATmega328P	Broadcom BCM2387 64-bit ARMv8 - Quad-Core Cortex-A53 a 1.2 GHz	SITARA AM335x a 1 GHz ARM Cortex-A8	NXP i.MX 6SoloX
Motor de gráficos	-	Co-procesador Multimedia Dual Core VideoCore IV®.	SGX530 3D, 20 M Polygons/S	Vivante GC420, acelerador gráfico integrado 2D/3D.
Memoria	Flash: 32 KB, SRAM: 2 KB, EEPROM: 1KB	1GB LPDDR2	512 MB DDR3L 800 MHz	1 GB
Almacenamiento masivo	-	Ranura para tarjeta MicroSD. Interfaz SDIO de de 8-bit	Ranura para tarjeta MicroSD. Interfaz de 8-bit eMMC	Ranura para tarjeta MicroSD. Interfaz SDIO de 8-bit
USB	1 conector USB 2.0. Puerto tipo B	4 puertos USB 2.0. Puertos tipo A	1 puerto USB 2.0. 1 enchufe tipo A	1 puerto USB 2.0. Puertos tipo A 1 puerto USB OTG (conector micro-AB)
Redes	-	1 conector RJ45 Fast ethernet 10/100 Mbps, Wi-Fi 802.11 b/g/n, Bluetooth 4.1 (clásico y de bajo consumo)	1 conector RJ45 Fast ethernet 10/100 Mbps	1 conector RJ45 Fast ethernet 10/100 Mbps, Wi-Fi 802.11 b/g/n, Bluetooth 4.0 Baja energía
Puertos seriales	1 puerto UART	2 puertos UART	5 puertos UART	3 puertos UART
Otras interfaces	1 interfaz I2C, 1 interfaz SPI	1 interfaz I2C, 2 interfaces SPI	1 interfaz I2C, 1 interfaz SPI	3 interfaces I2C, 1 interfaz SPI
Otros	ADC interno: 5 V a 10 bits	-	ADC interno: 1.8 V a 12 bits	Sensores: 1 acelerómetro de 3 ejes, 1 magnetómetro de 3 ejes, 1 giroscopio digita de 3 ejes. ADC interno: 3.3 V a 12 bits

CONTENIDO

- Planteamiento del problema
- Objetivos
- Caso de Estudio
- Propuesta
- Análisis de Resultados
- Conclusiones y Recomendaciones

PROPUESTA



ESTACIÓN DE CONTROL DE FLUJO DE CAUDAL

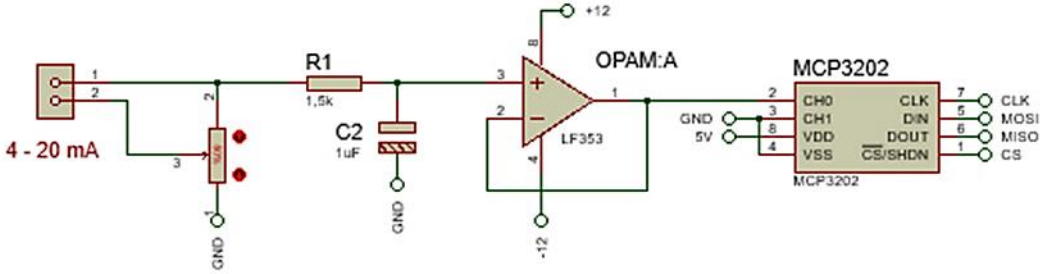
CONTROLADORES AVANZADOS

CONTROL BORROSO

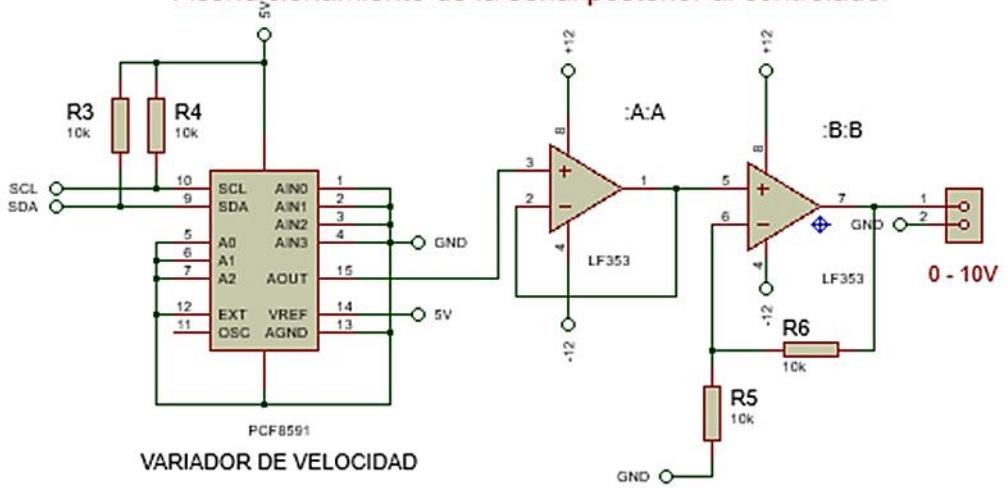
CONTROL PREDICTIVO POR MODELO

HARDWARE

Acondicionamiento de la señal previo al controlador

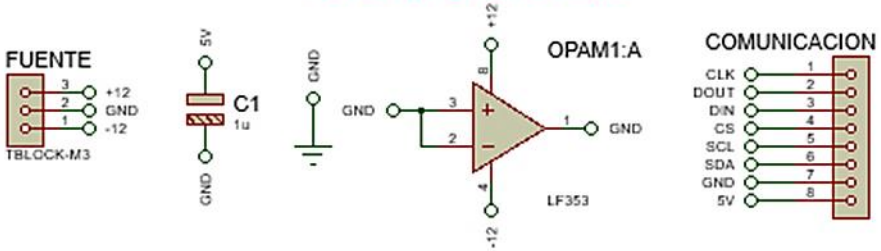


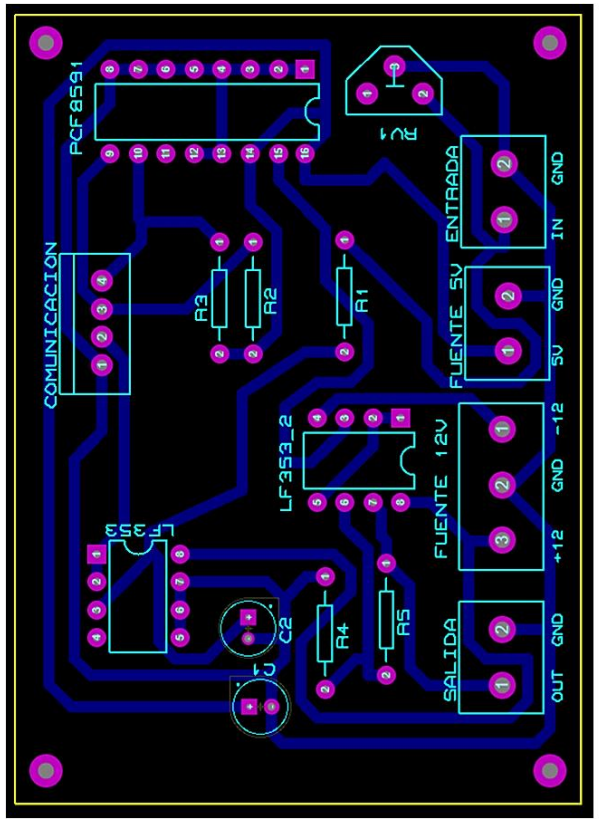
Acondicionamiento de la señal posterior al controlador

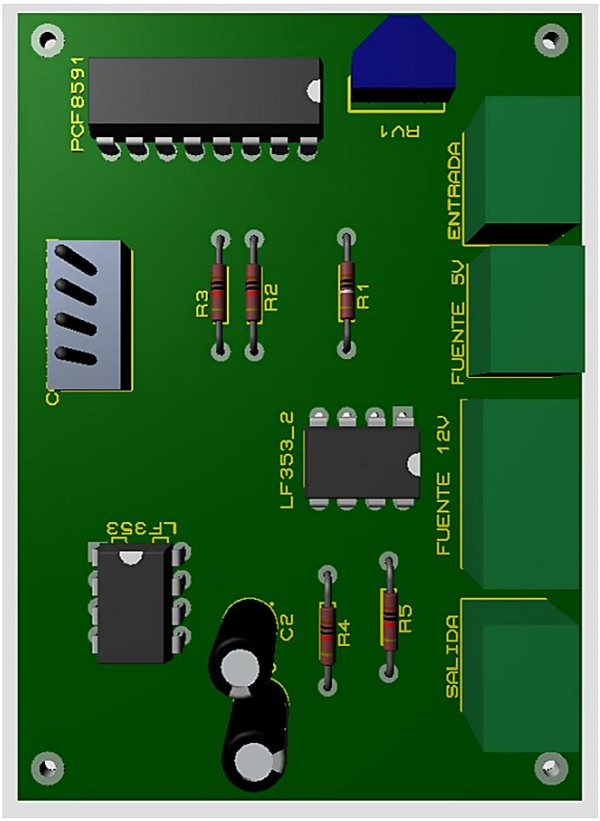


VARIADOR DE VELOCIDAD

Conexiones adicionales

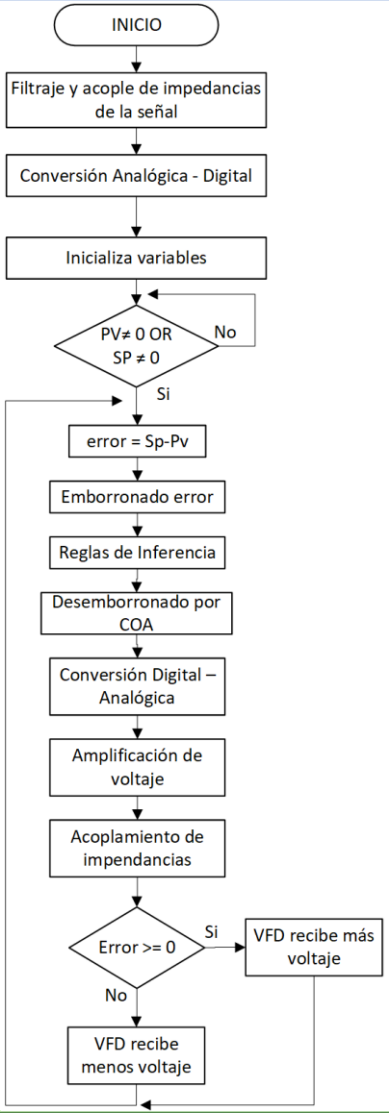




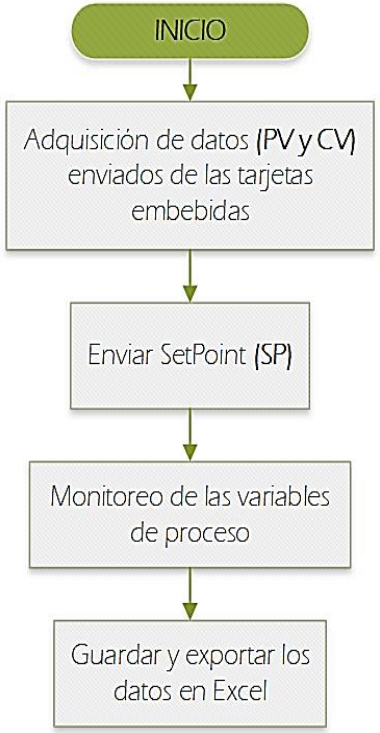


SOFTWARE

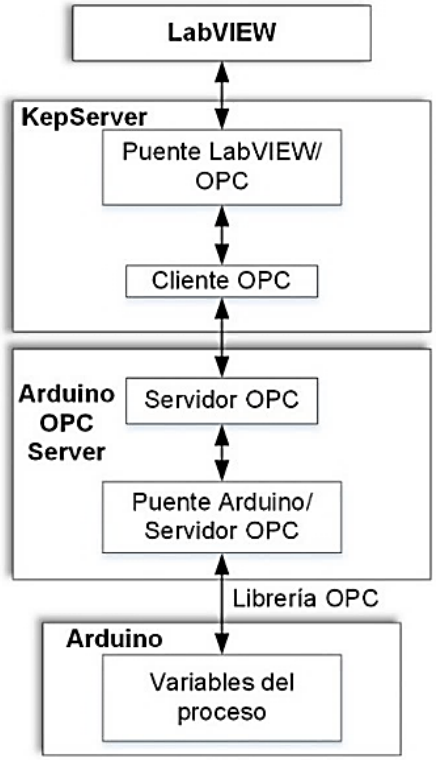
PROCESO DE CONTROL



HMI



COMUNICACIÓN OPC



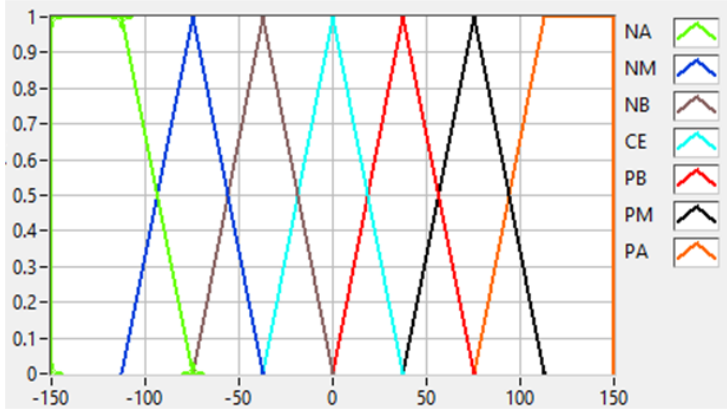
CONTROL BORROSO

DEFINICIÓN

Su objetivo es brindar un sistema frontal computacionalmente dotado de técnicas para trabajar con un razonamiento aproximado, en lugar de uno exacto. Todo está en términos del grado de pertenencia a un conjunto.

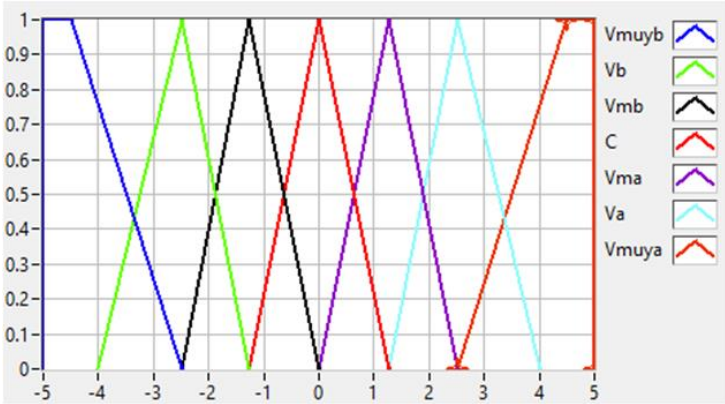
CONTROL BORROSO

CONJUNTOS DE ENTRADA




```
// Agrupa toddos los conjuntos borrosos  de entrada que pertenecen al mismo dominio.
  FuzzyInput* err_caudal = new FuzzyInput(1);
// Conjuntos borrosos de entrada
  FuzzySet* NA = new FuzzySet(-150,-150, -112.5, -75); // Error Negativo Alto
  err_caudal->addFuzzySet(NA);
  FuzzySet* NM = new FuzzySet(-112.5,-75, -75, -37.5); // Error Negativo Medio
  err_caudal->addFuzzySet(NM);
  FuzzySet* NB = new FuzzySet(-75,-37.5, -37.5, 0); // Error Negativo Bajo
  err_caudal->addFuzzySet(NB);
  FuzzySet* CE = new FuzzySet(-37.5,-0,0,37.5); // Error Neutro
  err_caudal->addFuzzySet(CE);
  FuzzySet* PB = new FuzzySet(0,37.5,37.5,75); // Error Positivo Bajo
  err_caudal->addFuzzySet(PB);
  FuzzySet* PM = new FuzzySet(37.5,75,75,112.5); // Error Positivo Medio
  err_caudal->addFuzzySet(PM);
  FuzzySet* PA = new FuzzySet(75,112.5,150,150); // // Error Positivo Alto
  err_caudal->addFuzzySet(PA);
  fuzzy->addFuzzyInput(err_caudal); //Adiciona los conjuntos de entrada al objeto Borroso
```

CONJUNTOS DE SALIDA




```
// Agrupa toddos los conjuntos borrosos  de entrada que pertenecen al mismo dominio.
  FuzzyOutput* Voltaje = new FuzzyOutput(1);
// Adicionando conjuntos borrosos de salida
  FuzzySet* Vmuyb = new FuzzySet(-5,-5,-4.5,-2.5); // Voltaje Muy Bajo
  Voltaje->addFuzzySet(Vmuyb);
  FuzzySet* Vb = new FuzzySet(-4, -2.5, -2.5,-1.275); // Voltaje Bajo
  Voltaje->addFuzzySet(Vb);
  FuzzySet* Vmb = new FuzzySet(-2.5,-1.275,-1.275,0); // Voltaje Medio Bajo
  Voltaje->addFuzzySet(Vmb);
  FuzzySet* C = new FuzzySet(-1.275, 0, 0, 1.275); // Voltaje Cero
  Voltaje->addFuzzySet(C);
  FuzzySet* Vma = new FuzzySet(0,1.275,1.275,2.5); // Voltaje Medio Alto
  Voltaje->addFuzzySet(Vma);
  FuzzySet* Va = new FuzzySet(1.275,2.5,2.5,4); // Voltaje Alto
  Voltaje->addFuzzySet(Va);
  FuzzySet* Vmuya = new FuzzySet(2.5,4.5,5,5); // Voltaje Muy Alto
  Voltaje->addFuzzySet(Vmuya); // Adicionando o FuzzySet T
  fuzzy->addFuzzyOutput(Voltaje); // Adicionando o FuzzyOutput no objeto Fuzzy
```

CONTROL BORROSO

BASE DE REGLAS

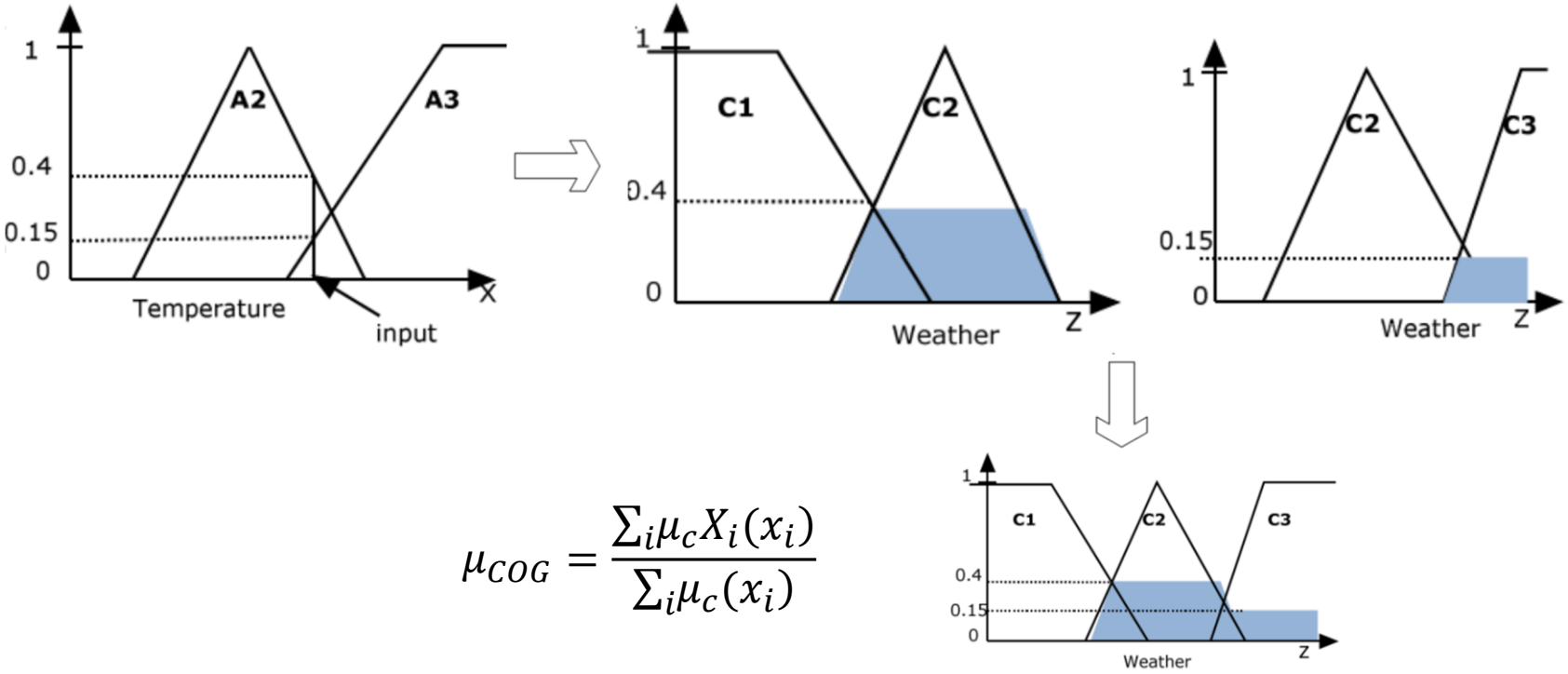
1. ***SI*** err_{caudal} *es* NA, ***ENTONCES*** *el voltaje del variador es* $Vmuyb$
2. ***SI*** err_{caudal} *es* NM, ***ENTONCES*** *el voltaje del variador es* Vb
3. ***SI*** err_{caudal} *es* NB, ***ENTONCES*** *el voltaje del variador es* Vmb
4. ***SI*** err_{caudal} *es* CE, ***ENTONCES*** *el voltaje del variador es* C
5. ***SI*** err_{caudal} *es* PB, ***ENTONCES*** *el voltaje del variador es* Vma
6. ***SI*** err_{caudal} *es* PM, ***ENTONCES*** *el voltaje del variador es* Va
7. ***SI*** err_{caudal} *es* PA, ***ENTONCES*** *el voltaje del variador es* T

```
// Base de Reglas: Inferencia de Mamdani
  FuzzyRuleAntecedent* SI_NA = new FuzzyRuleAntecedent(); //Instanciando un Antecedente
  SI_NA->joinSingle(NA); // Adicionando un Conjunto de Entrada al Antecedente
  FuzzyRuleConsequent* ENTONCES_Vmuyb = new FuzzyRuleConsequent(); // Instanciando un Consecuente
  ENTONCES_Vmuyb->addOutput(Vmuyb); // Adicionando un Conjunto de Salida al Consecuente

  FuzzyRuleAntecedent* SI_NM = new FuzzyRuleAntecedent();
  SI_NM->joinSingle(NM);
  FuzzyRuleConsequent* ENTONCES_Vb= new FuzzyRuleConsequent();
  ENTONCES_Vb->addOutput(Vb);

  FuzzyRuleAntecedent* SI_NB = new FuzzyRuleAntecedent();
  SI_NB->joinSingle(NB);
  FuzzyRuleConsequent* ENTONCES_Vmb = new FuzzyRuleConsequent();
  ENTONCES_Vmb->addOutput(Vmb);

  FuzzyRuleAntecedent* SI_CE = new FuzzyRuleAntecedent();
  SI_CE->joinSingle(CE);
  FuzzyRuleConsequent* ENTONCES_C = new FuzzyRuleConsequent();
  ENTONCES_C->addOutput(C);

  FuzzyRuleAntecedent* SI_PB = new FuzzyRuleAntecedent();
  SI_PB->joinSingle(PB);
  FuzzyRuleConsequent* ENTONCES_Vma = new FuzzyRuleConsequent();
  ENTONCES_Vma->addOutput(Vma);
```

CONTROL BORROSO

DESEMBORRONADO

```
fuzzy -> setInput(1, Error);
fuzzy -> fuzzify();
Salida = fuzzy -> defuzzify(1);
```

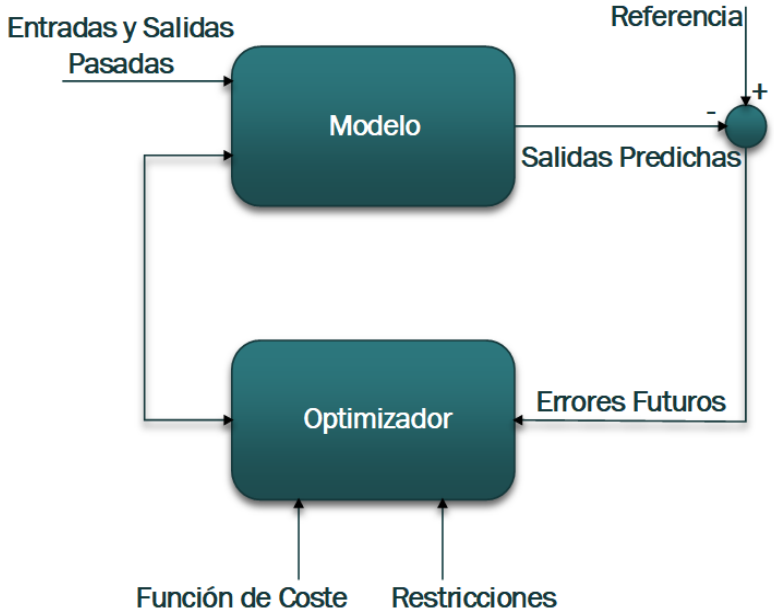


$$\mu_{COG} = \frac{\sum_i \mu_c X_i(x_i)}{\sum_i \mu_c(x_i)}$$

CONTROL PREDICTIVO POR MODELO

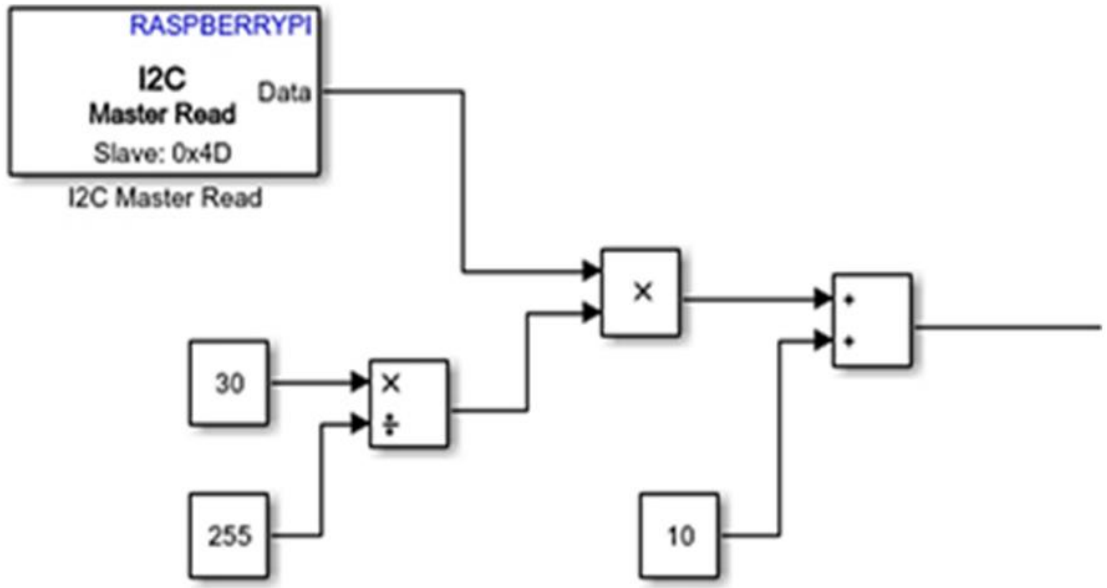
DEFINICIÓN

Permite optimizar el proceso de la planta, al predecir el desempeño futuro y calcular las acciones de las variables manipuladas y controladas del proceso para alcanzar los objetivos esperados



CONTROL PREDICTIVO POR MODELO

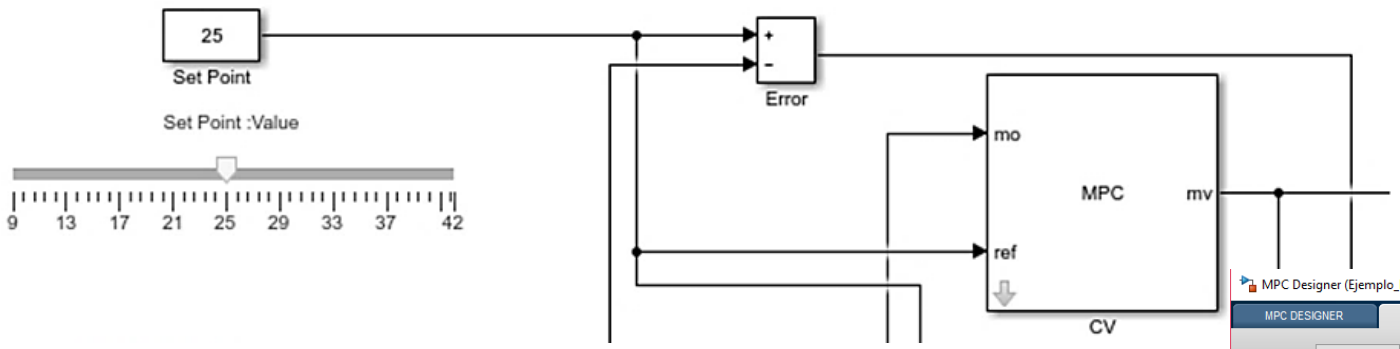
ADQUISICIÓN DE DATOS

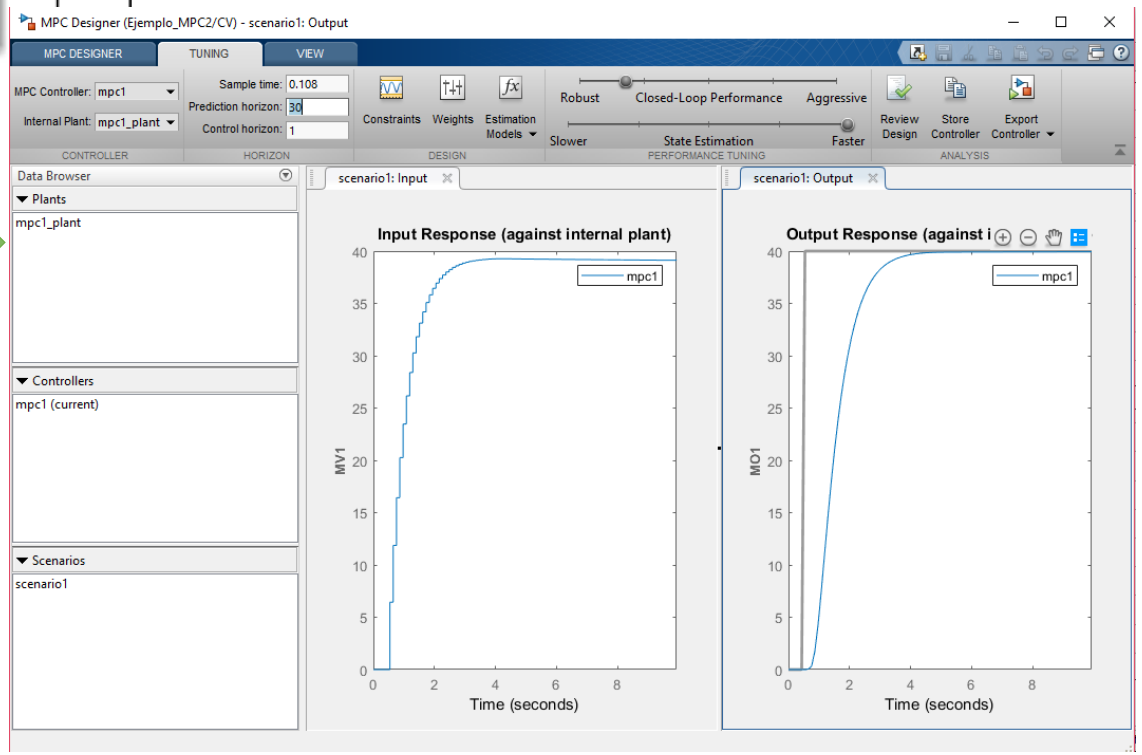


- **Board:** Elegir el modelo de la tarjeta **(Pi 3 Model B)**
- **Slave address:** Elegir la dirección del esclavo conectado **(4D = 77)**
- **Data type:** Se elige el tipo de dato que se obtendrá a la salida del bloque **(uint8).**
- **Sample time:** Elegir el tiempo de muestreo **(0.108).**

CONTROL PREDICTIVO POR MODELO

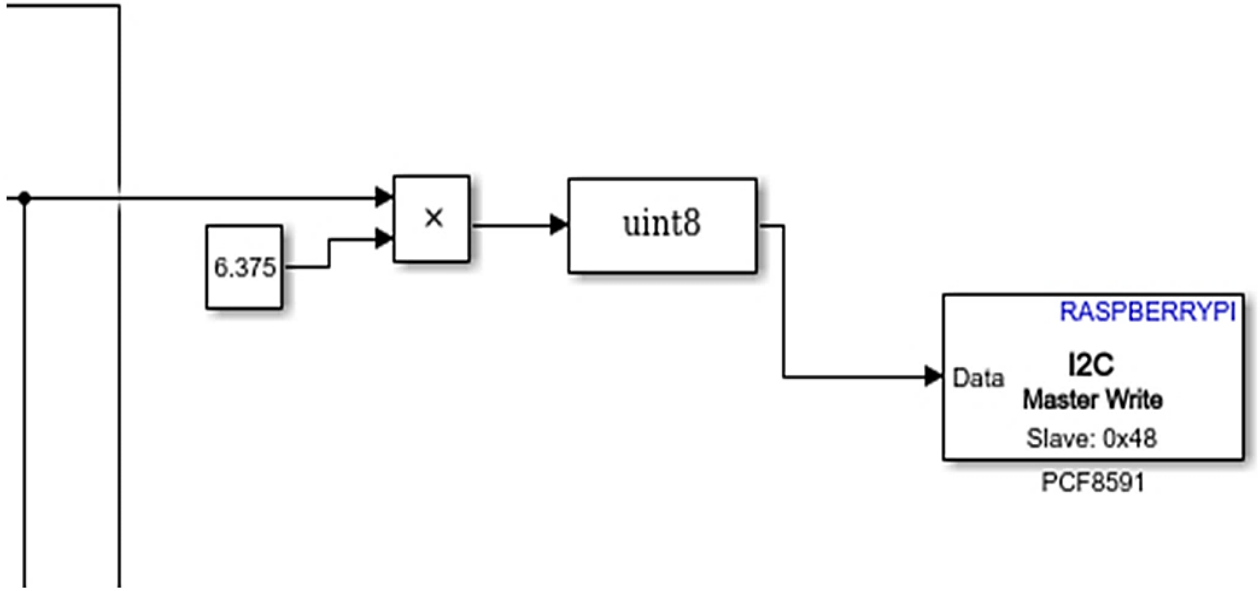
SETPOINT Y BLOQUE CONTROL





CONTROL PREDICTIVO POR MODELO

SALIDA DEL CONTROLADOR



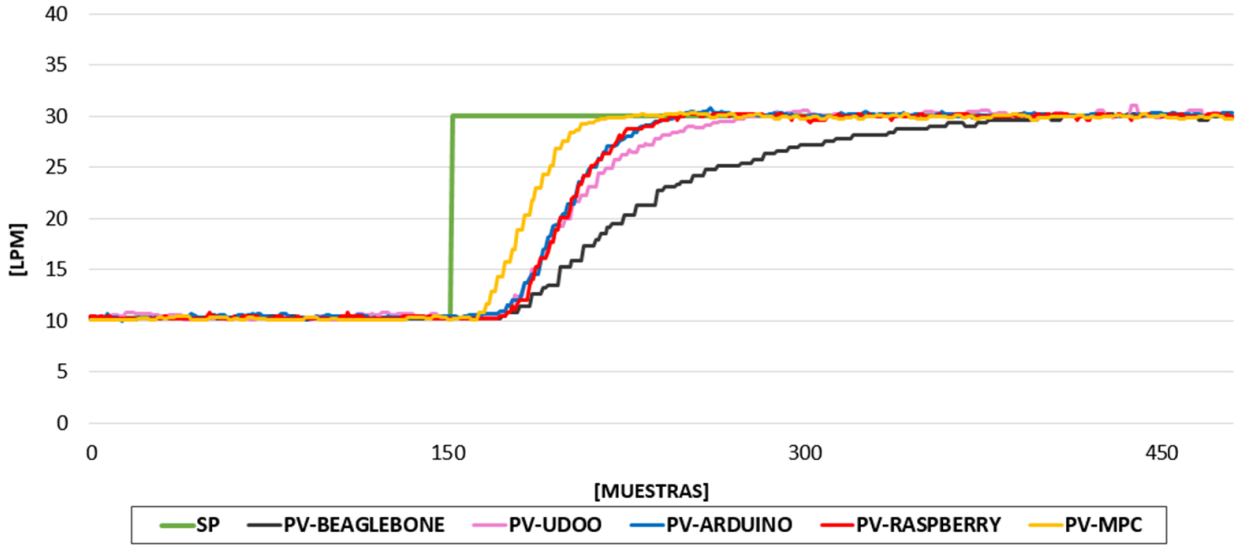
- **Board:** Elegir el modelo de la tarjeta **(Pi 3 Model B)**
- **Slave address:** Seleccionar la dirección del esclavo por donde se va a escribir los valores de 0 – 255 **(48 = 72).**
- **Slave register address:** Se elige la dirección del registro del esclavo, esto se lo puede encontrar en las hojas de datos del integrado. **(120)**

CONTENIDO

- Planteamiento del problema
- Objetivos
- Caso de Estudio
- Propuesta
- Análisis de Resultados
- Conclusiones y Recomendaciones

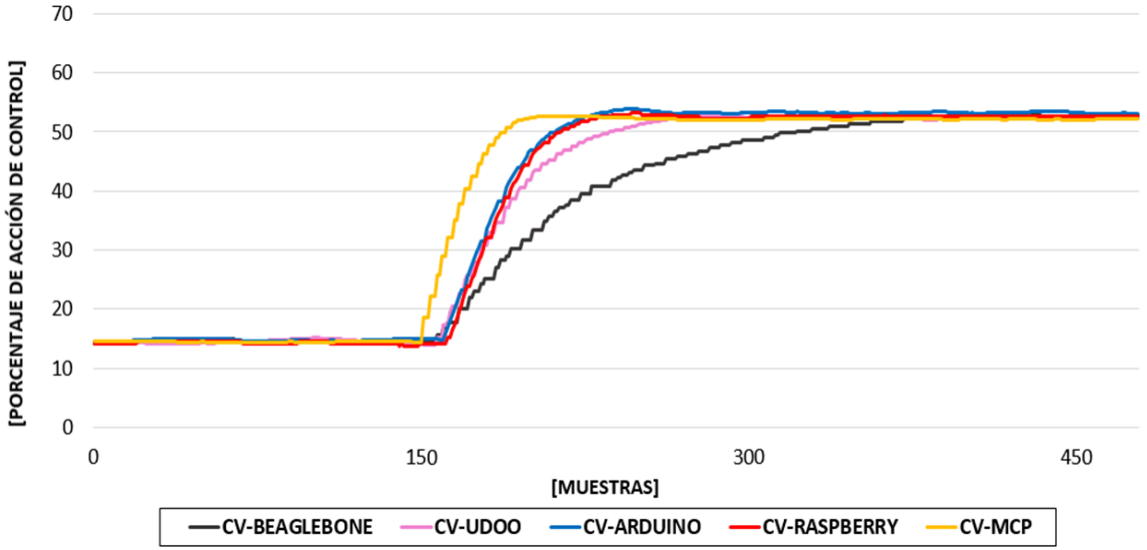
ANÁLISIS DE RESULTADOS

RESPUESTA ANTE UN ESCALÓN UNITARIO DE AMBOS CONTROLADORES



SP VS. PV

CV



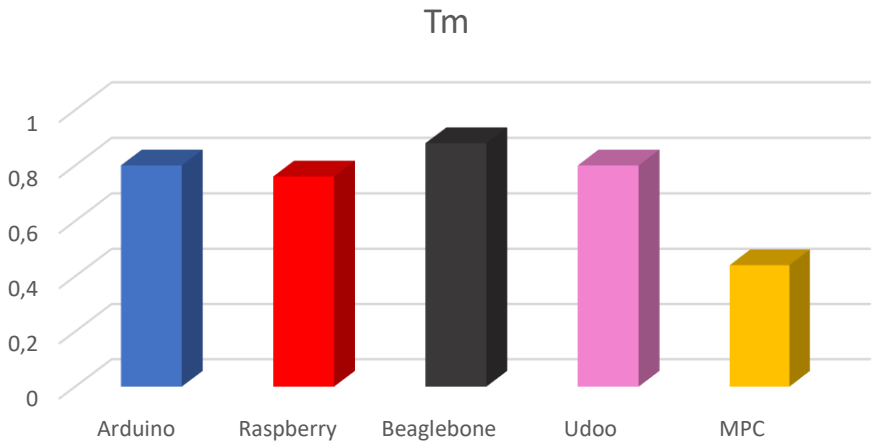
ANÁLISIS DE RESULTADOS

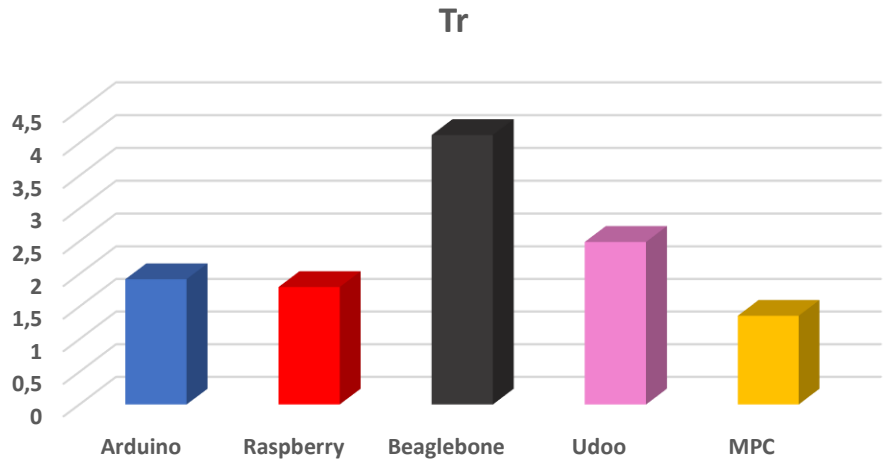
VALORES CARACTERÍSTICOS TEMPORALES Y DE SOBREOSCILACIÓN

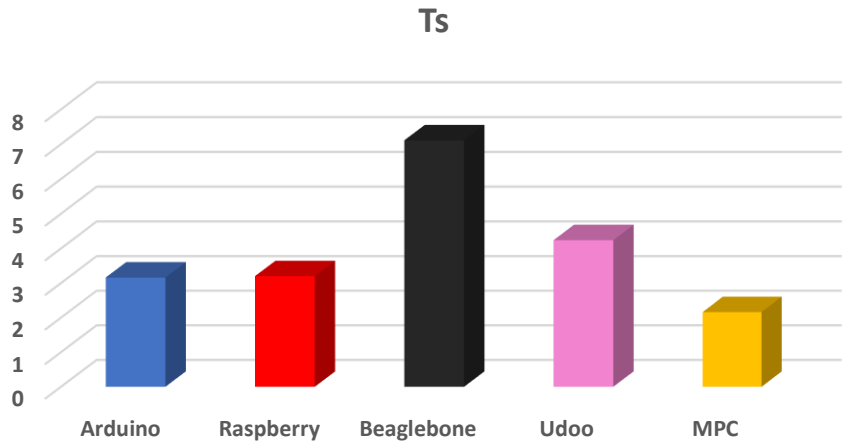
		Tarjeta				
		Arduino	Raspberry	Beaglebone	Udoo	MPC
Valores Característicos y de Sobreimpulso	Tm	0.8s	0.76s	0.88s	0.8s	0.44s
	Tr	1.92s	1.8s	4.12s	2.49s	1.36s
	Ts	3.16s	3.2s	7.11s	4.24s	2.16s
	%M	3%	0.25%	0.25%	2%	0.75%

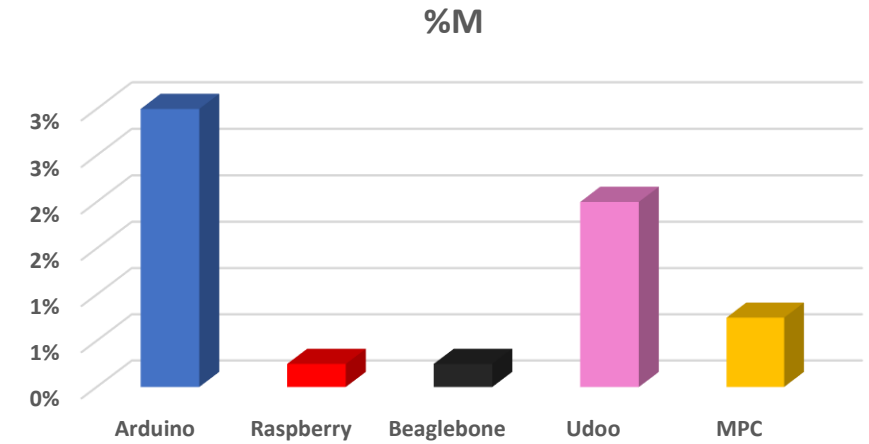
ANÁLISIS DE RESULTADOS

VALORES CARACTERÍSTICOS TEMPORALES Y DE SOBREOSCILACIÓN





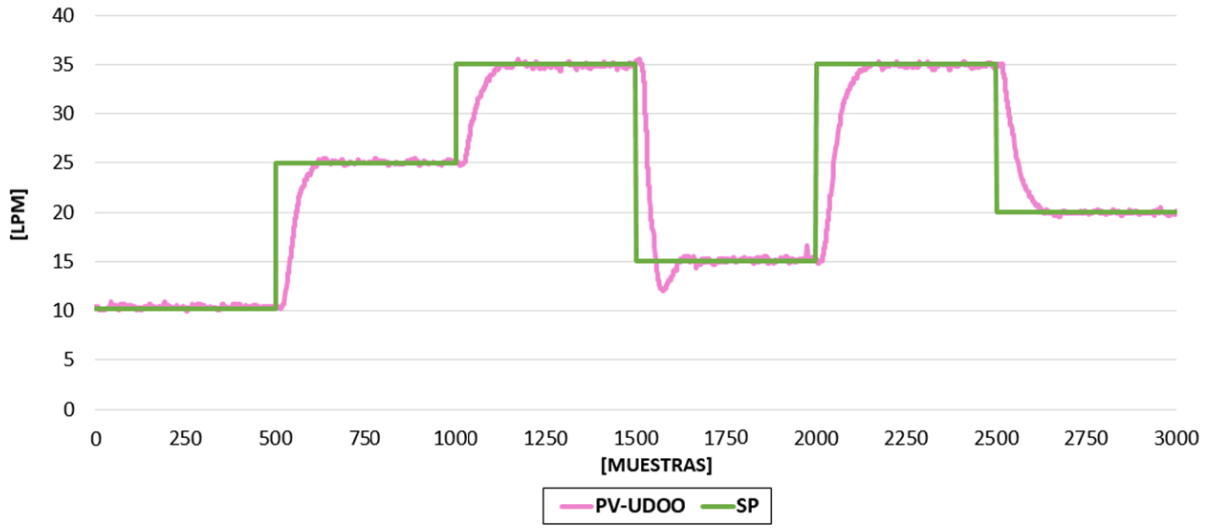




UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

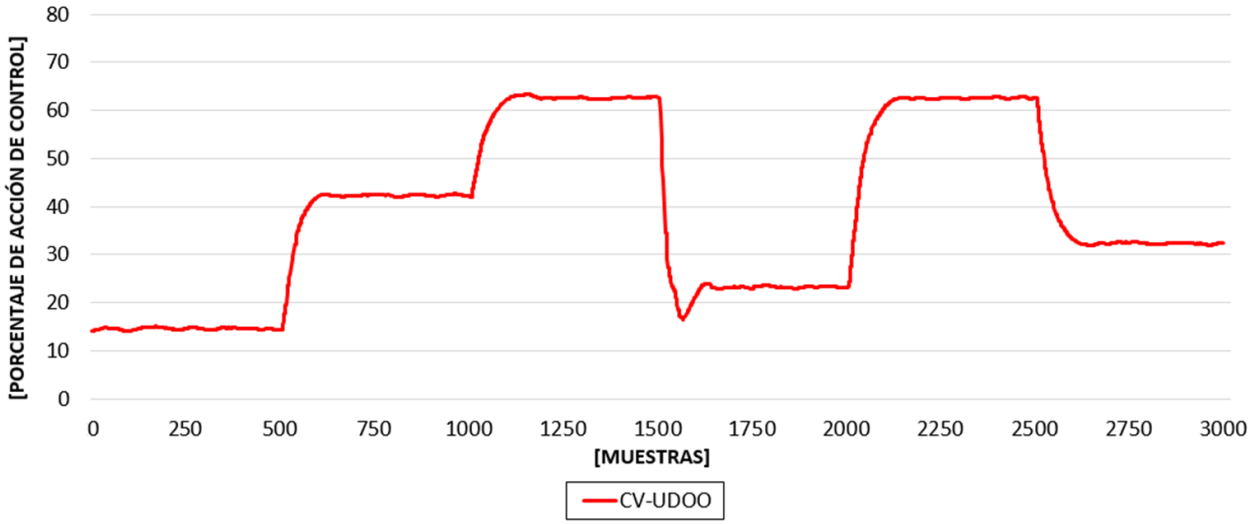
ANÁLISIS DE RESULTADOS

CONTROLADOR BORROSO EN UDOO NEO FULL



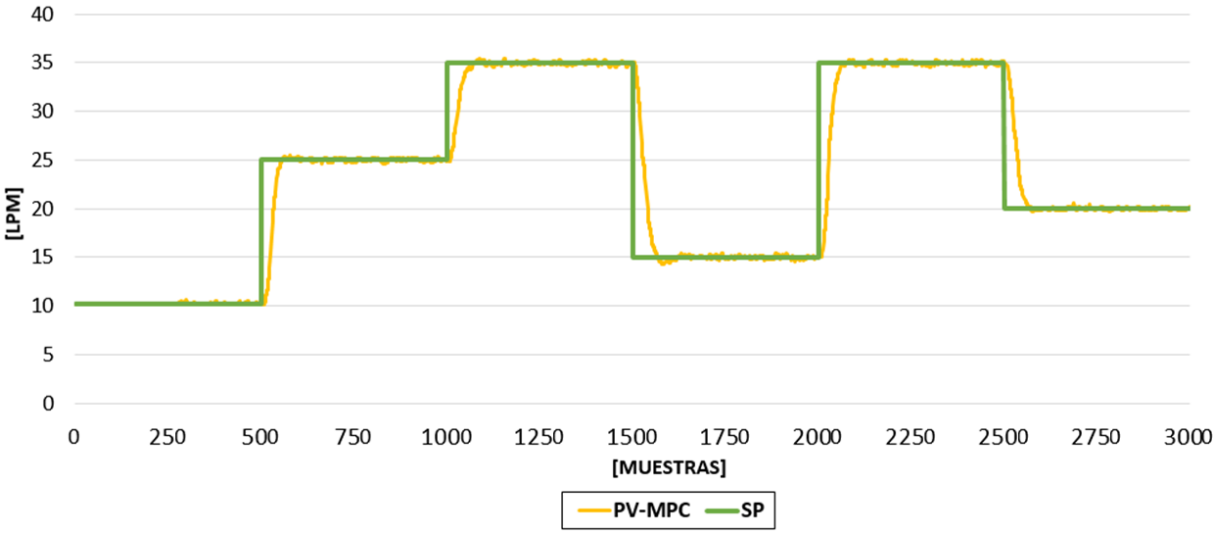
SP VS. PV

CV



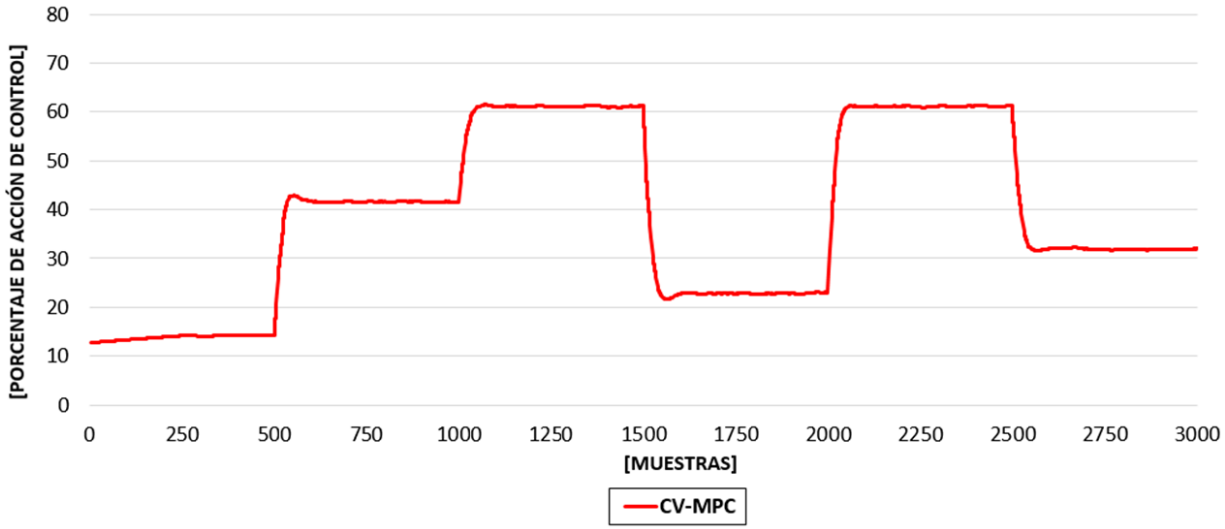
ANÁLISIS DE RESULTADOS

CONTROLADOR MPC EN RASPBERRY PI 3



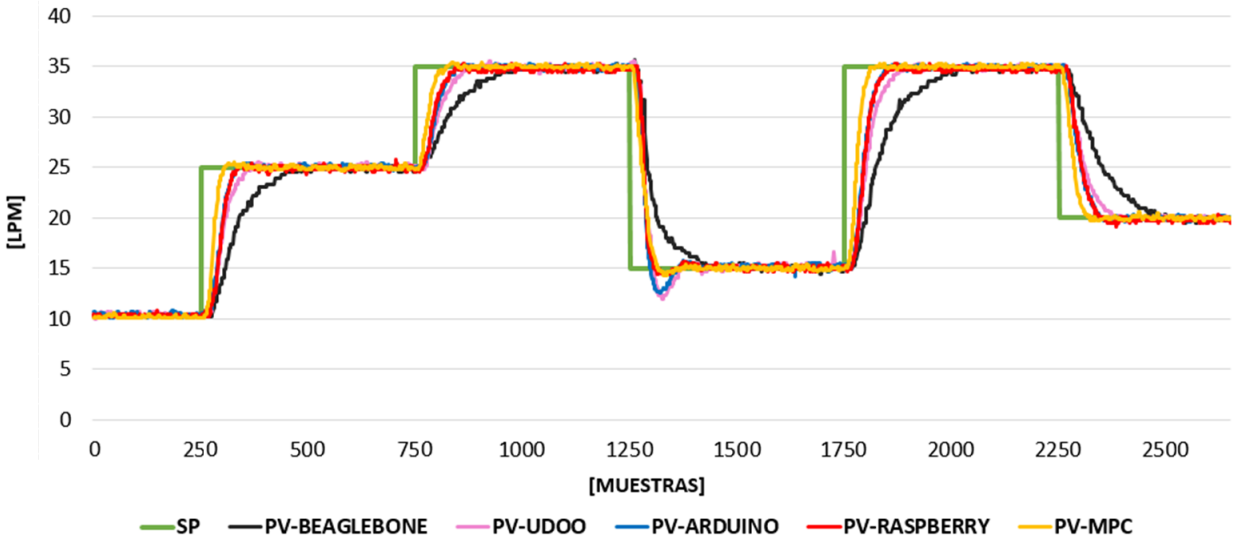
SP VS. PV

CV



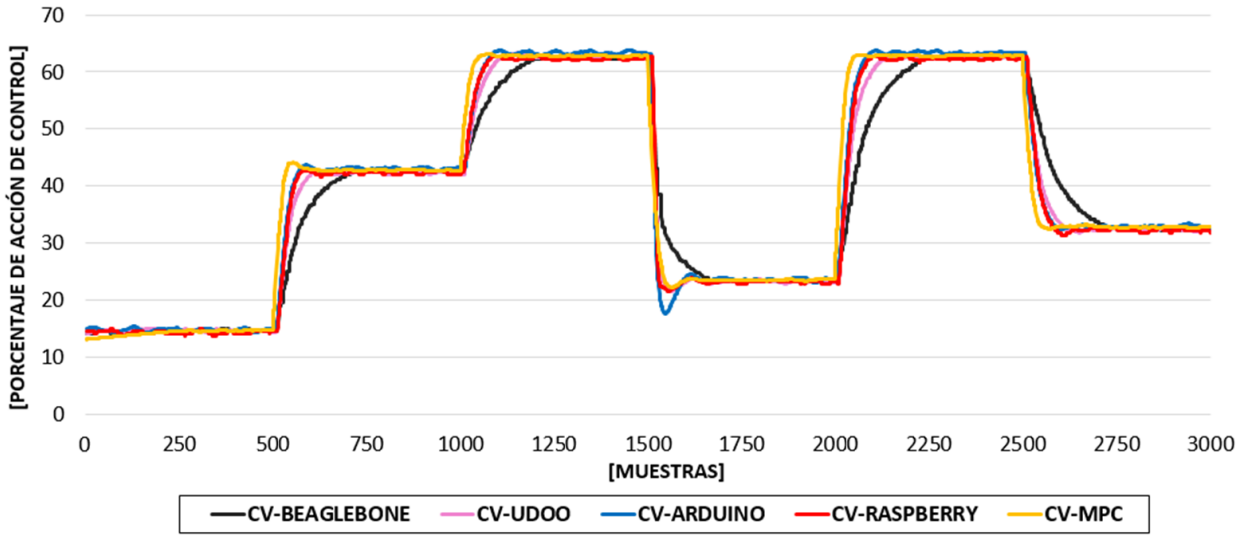
ANÁLISIS DE RESULTADOS

RESPUESTA CON MÚLTIPLES CAMBIOS DE CONSIGNA DE AMBOS CONTROLADORES



SP VS. PV

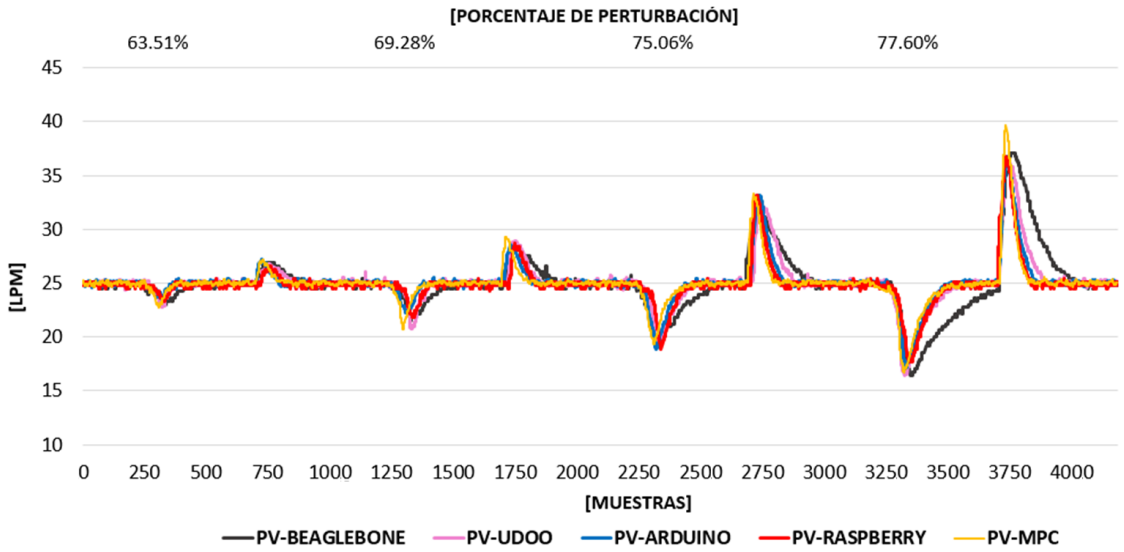
CV



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

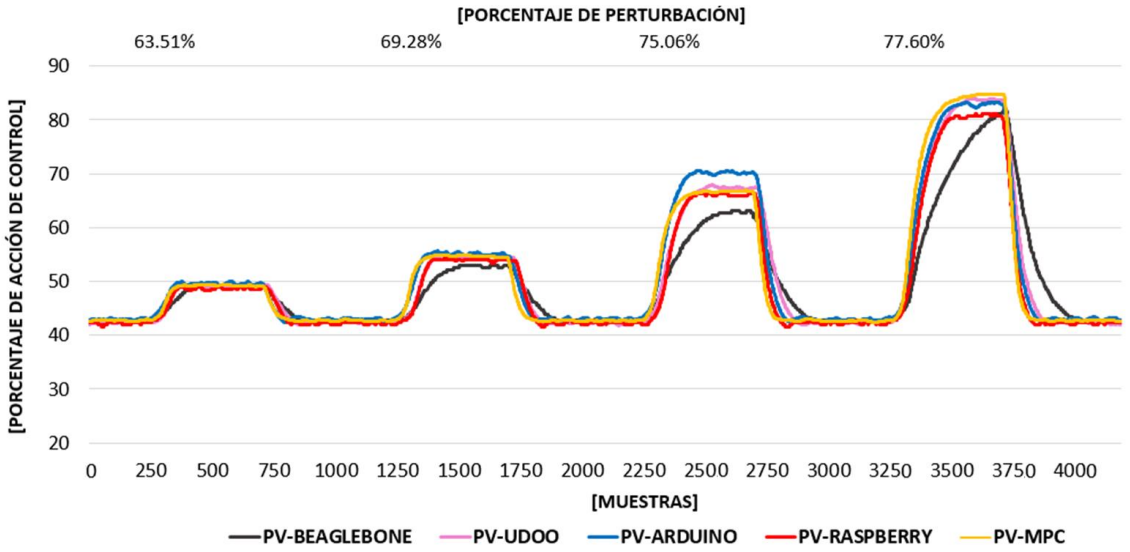
ANÁLISIS DE RESULTADOS

RESPUESTA ANTE UN PORCENTAJE DE PERTURBACIÓN DE AMBOS CONTROLADORES



SP VS. PV

CV



ANÁLISIS DE RESULTADOS

ANÁLISIS DE COSTOS

TARJETA	ARDUINO UNO R3	RASPBERRY PI 3	BEAGLEBONE BLACK	UDOO NEO FULL
COSTO (Incluye accesorios necesarios para trabajar)	$13.50 (Fuente Mercado Libre)	$78.00 (Fuente Mercado Libre)	$80.00 (Fuente Amazon)	$100.00 (Fuente Udoo Shop)
Amplificadores de Alta Impedancia (LF353)	$1.00	$1.00	$1.00	$1.00
Conversor Analógico – Digital (ADC MCP3202)	$4.50	$4.50	$4.50	
Conversor Digital - Analógico (DAC PCF)	$4.25	$4.25	$4.25	$4.25
Diseño Placas	$7.00	$7.00	$7.00	$7.00
Otros (Cajas, resistencias, capacitores, zócalos)	$10.25	$10.25	$10.25	$10.25
TOTAL	**$40.50**	**$105.00**	**$107.00**	**$122.50**

CONTENIDO

- Planteamiento del problema
- Objetivos
- Caso de Estudio
- Propuesta
- Análisis de Resultados
- Conclusiones y Recomendaciones

CONCLUSIONES

- La variedad de tarjetas embebidas que se encuentra en el mercado crece exponencialmente, no solo en marcas, sino en las versiones de las ya existentes, dependiendo de las necesidades y economía del usuario; para este trabajo se eligieron versiones potentes de microcomputadoras como lo son: Raspberry Pi 3, Beaglebone Black, Udoo Neo Full y la versión más básica e icónica de Arduino (UNO), las cuales ofrecieron buenas prestaciones en relación a costos reducidos.

- Se ha corroborado que las denominadas tecnologías de bajo costo a pesar de sus limitaciones técnicas permiten la implementación de control avanzado, mediante la utilización de programación de alto nivel, librerías especializadas y toolboxes de softwares matemáticos desarrollados específicamente para este tipo de dispositivos.

- El diseño del algoritmo de control borroso con inferencia tipo Mamdani se realiza con relativa facilidad, ya que no requiere el modelo matemático del proceso que se va a controlar; a pesar de ello ha presentado resultados bastante eficientes y dado que los requerimientos computaciones no son tan altos, ha permitido que se lo implemente en todas las tarjetas antes mencionadas, teniendo como base un código muy similar en todos los casos.

CONCLUSIONES

- La implementación del controlador MPC en tarjetas embebidas de bajo costo se realiza con mayor facilidad mediante la herramienta Simulink, siempre y cuando disponga de los paquetes de soporte adecuados para trabajar en ellas; el paquete para Raspberry Pi 3 incluye bloques para leer y escribir datos mediante comunicación I^2C, para Beaglebone Black se carece de ellos.

- El control MPC solo se pudo implementar en la tarjeta Raspberry Pi 3, por las prestaciones de la herramienta Simulink; en Arduino UNO R3 no se logró realizarlo debido a que el controlador genera un coste computacional alto y la capacidad de cálculo de esta tarjeta no es suficiente; en Beaglebone Black y Udoo Neo Full a pesar de tener capacidades de cálculo suficientes, no se dispone de los complementos necesarios y por ello tampoco se lo realizó.

- La tecnología que presentó mejores resultados ante la implementación de un controlador borroso y la única que permitió el desarrollo de un controlador MPC fue Raspberry Pi 3, a pesar de solo tener disponibilidad para señales de entrada y salida digitales, se demostró que utilizando elementos externos y acondicionando dichas señales se pueden desarrollar aplicaciones de control que ofrezcan resultados competitivos a los que brindan dispositivos de control industrial.

RECOMENDACIONES

- El código desarrollado para el controlador borroso a pesar de la similitud en todas las tarjetas, presenta variaciones en la forma en cómo se realiza la comunicación serial, por lo cual es adecuado verificar que buses de comunicación están activos y cuales están disponibles para establecer dicha comunicación.

- Cuando se envíe tramas por comunicación serial es recomendable usar programas externos (por ejemplo, el programa X-CTU), para verificar que la información que llega al puerto sea la adecuada, esto puede ayudar a encontrar y corregir rápidamente errores que se comenten al enviar datos por comunicación serial.

- Udoo en todas sus versiones presenta en su entorno una interfaz similar a la de Arduino en su escritorio remoto, en el cual se pueden cargar programas desarrollados en Arduino IDE, con la única diferencia que al momento de escribir en el puerto serial se debe utilizar el comando Serial0.

- Para implementar el controlador MPC en la tarjeta Raspberry Pi 3, es recomendable instalar la versión 2017a de MATLAB, esto se debe a que solo en esta versión Simulink posee los bloques de comunicación I2C y SPI.

MUCHAS GRACIAS