New Equations and New Algorithms of Energy Injection to facilitate automation tasks

Juan Manuel Andrade Morales, MSc. Ingeniería de Sistemas, Ingeniero Electricista[1]
[1]Escuela Colombiana de Ingeniería Julio Garavito, Colombia, juan.andrade@escuelaing.edu.co

Abstract– ***The study of microprocessors and PWM interfaces, applied on automation of continuous variables, force to reconsider the energy injection equations at control systems due to the PWM interface dose the energy differently. These new equations must be known by students and engineers due to the possibility of doing fast, cheap and efficient automatic control applications. Current automatic control books show a compendium of mathematics modelling, however none of these show an integration of mathematics modelling, physical analysis, digital electronics, analog electronics, power electronics and algorithms design in machine language. This article intends to integrate the preceding knowledge and proposes the application of the previously deduced equations in order to implement the control strategies of Proportional Integral (PI) and Proportional Integral Derivative (PID) controllers in a microprocessor that has a PWM digital interface.***
(Keywords in English) Microprocessors, Digital Interface PWM, Real-time Automatic Control, Control of continuous variables, Automation, Real-time algorithms.

Digital Object Identifier (DOI):http://dx.doi.org/10.18687/LACCEI2018.1.1.310
ISBN: 978-0-9993443-1-6
ISSN: 2414-6390

Nuevas Ecuaciones y Nuevos Algoritmos de Inyección de Energía para Facilitar las Tareas de Automatización

Juan Manuel Andrade Morales, MSc. Ingeniería de Sistemas, Ingeniero Electricista[1]
[1]Escuela Colombiana de Ingeniería Julio Garavito, Colombia, juan.andrade@escuelaing.edu.co

Abstract*– El estudio de los microprocesadores y las interfaces PWM, cuando se aplican a la automatización de variables continuas, obliga a replantear las ecuaciones de inyección de energía en los sistemas de control, ya que la interfaz PWM dosifica de manera diferente la energía. Estas nuevas ecuaciones deben ser conocidas por estudiantes e ingenieros debido a que existe una gran posibilidad de hacer aplicaciones de control automático, rápidas, baratas y eficientes que deben estar a su alcance. Los libros actuales de control automático son un compendio de modelamiento matemático pero ninguno muestra una integración entre modelamiento matemático, análisis físico, electrónica digital, electrónica análoga, electrónica de potencia y diseño de algoritmos en lenguaje de máquina. Este artículo hace un esfuerzo por integrar los anteriores conocimientos y aporta la aplicación de las nuevas ecuaciones deducidas anteriormente para poder hacer realidad las estrategias de control proporcional integral (PI) y proporcional integral derivativo (PID) en un microprocesador que posee una interfaz digital PWM.

Keywords—Microprocesadores, Interfaz Digital PWM, Control Automático en tiempo real, Control de variables continuas, Automatización, Algoritmos en tiempo real.

I. Introducción

Siempre que un ingeniero de automatización industrial diseña un algoritmo para controlar variables de máquinas, tiene que enfrentarse a un reto permanente: "¿Cómo dosificar la energía para inyectársela a la planta que se está controlando?".

En la Fig. 1 se puede apreciar que al encender una máquina a plena energía, el comportamiento de la misma puede ser el de la curva 4. Por otro lado, si se desea que la curva 4 no sobrepase la recta 5(valor deseado), se deben aplicar trozos parciales de energía y es posible que la curva 4 se transforme "mágicamente" en la curva 1.

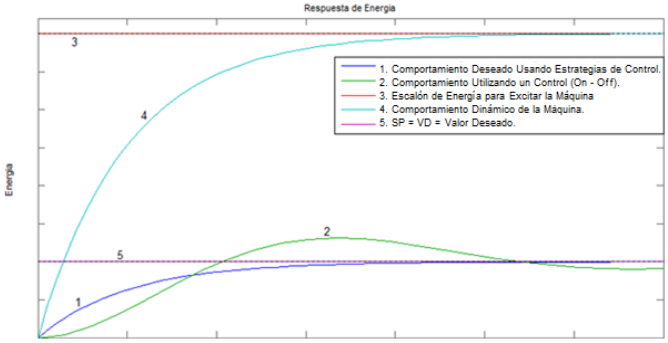

Fig. 1. Respuestas de los sistemas dinámicos a diferentes formas de inyección de energía.

Para hacer este "encantamiento", se requiere de amplio conocimiento, gran experiencia y sobre todo, mucha ayuda. Una de las ayudas, entre otras, son los sistemas de dosificación de energía.

Hasta hace unos años, el sistema "campeón" se construía con base en amplificadores operacionales y amplificadores de potencia, sin embargo, con la aparición de los computadores digitales este campeón está pasando al olvido, y hoy, los nuevos protagonistas son: El conversor Digital-Analógico y el Modulador de ancho de pulso, con sus respectivas adaptaciones de potencia.

Es importante aclarar que para poder entender la magnitud de esta investigación se requiere que el lector domine la Electrónica de la interfaz digital PWM y sus adaptaciones con electrónica de potencia para cargas de alta potencia (Ver Fig. 1A, 1B y 1C).

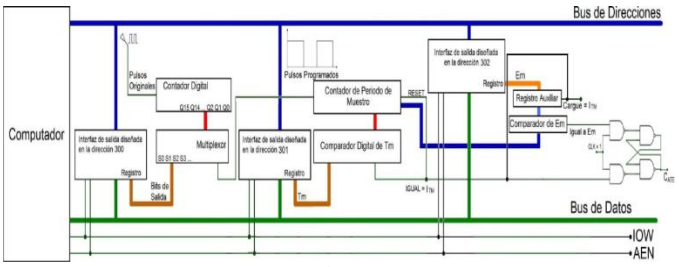

Fig. 1A. Interfaz Digital PWM

Digital Object Identifier (DOI): http://dx.doi.org/10.18687/LACCEI2018.1.1.310
ISBN: 978-0-9993443-1-6
ISSN: 2414-6390

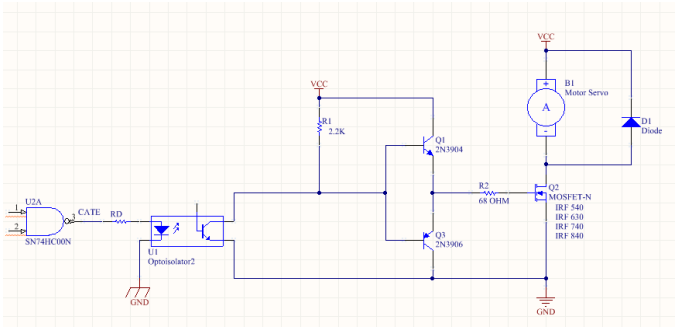

Fig. 1B. Uso de un MOSFET de potencia o un IGBT con cargas de alta potencia y de alta frecuencia desde una interfaz digital PWM.

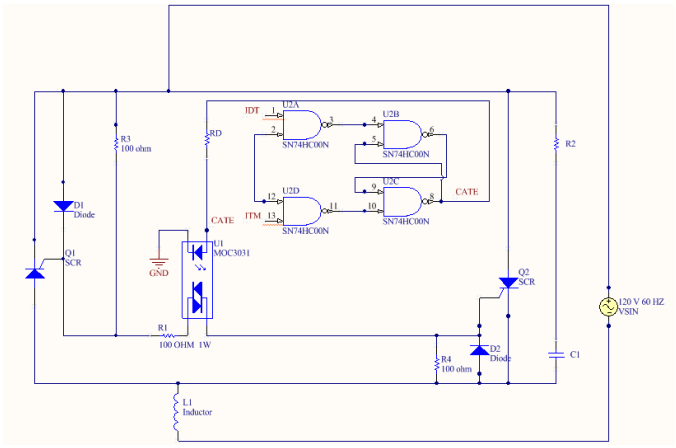

Fig. 1C. Utilización de 2 SCR's en anti paralelo para gobernar cargas de alta potencia desde una interfaz PWM Digital.

II. REPLANTEAMIENTO DE LAS ECUACIONES DE INYECCIÓN DE ENERGÍA CUANDO SE USA UNA INTERFAZ DIGITAL PWM

El objetivo de esta nueva estrategia es el de integrar conceptos físicos y matemáticos a los nuevos avances en ingeniería electrónica, específicamente a las nuevas herramientas para dosificar energía en los microprocesadores.

Si se va a emplear el sistema de dosificación de energía de la interfaz digital PWM, las ecuaciones deben armonizarse para ser coherentes con esta interfaz.

Si recordamos los conceptos explicados de la interfaz digital PWM, al registro de periodo de muestreo debe llegar un número binario lo más cercano a FF si este registro es de 8 Bits, o lo más cercano a FFFF si este registro es de 16 Bits, y así sucesivamente. En consecuencia, esto permitirá tener aproximadamente 256 anchos de pulsos diferentes, en registros de 8 Bits y aproximadamente 65,536 anchos de pulso diferentes, en registros de 16 Bits (recordar conceptos del diseño digital del sistema PWM agregado a un microprocesador o a un DSC (Digital Signal Control)).

La expresión matemática que representa la energía a inyectar al sistema dinámico tiene que acomodarse para ser tratada en sistemas digitales y mediante algoritmos que manejen números fraccionarios.

Una expresión de energía a inyectar a un sistema dinámico, en el tiempo continuo, con la estrategia de control PID y cuya construcción real se lleva a cabo mediante amplificadores operacionales, del tipo,

$$E = K_c\left(e + T_D\frac{de}{dt} + \frac{1}{T_I}\int edt\right) \quad (1)$$

No sirve para ser tratada mediante un microprocesador programado en lenguaje de máquina (que sería el algoritmo más rápido posible).

Donde:

E: Energía, (debe transformarse para poder ser aplicada desde un microprocesador digital en una expresión de tiempo discreto) tal como:

$$E = \left[e_i + T_D\left(\frac{e_i - e_{i-1}}{T_m}\right) + \frac{1}{T_I}\sum_{n=p}^{n=i} e_n T_m\right] K_c \quad (2)$$

Aquí se puede observar que se ha hecho un cambio sencillo: La derivada $\frac{de}{dt}$ se ha reemplazado por la definición original de derivada $\left(\frac{e_i - e_{i-1}}{T_m}\right)$ (derivada simplemente es la razón de cambio de una variable en un período de tiempo T_m).

Cuando se empiezan a trabajar estas ecuaciones con microprocesadores, las derivadas de las variables sólo se pueden calcular cuando se tiene acceso a dos valores consecutivos de la variable, en este caso $e_i - e_{i-1}$, y el intervalo de tiempo en que se toman estos dos valores no puede ser mayor al período de muestreo T_m (recordar el teorema del muestreo). En consecuencia, la derivada en tiempo continuo queda transformada en:

$$\left(\frac{e_i - e_{i-1}}{T_m}\right) \quad (3)$$

Ahora, se debe reemplazar la integral $\int edt$.

Recordemos que una integral es una sumatoria de áreas que tiene de largo e y de ancho dt, por lo tanto, puedo reemplazar la integral por:

$$\sum_{n=p}^{n=i} e_n T_m \quad (4)$$

Donde p puede ser igual a cero o desde otro valor que el diseñador considere pertinente.

T_c : Intervalo de tiempo que para el caso no puede ser mayor que el período de muestreo

T_m: Deducido del ancho de banda del comportamiento dinámico que exhibe la variable.

En consecuencia, la integral será reemplazada por $\sum_{n=p}^{n=i} e_n T_m$.

Ahora se debe pensar en que la expresión resultante:

$$E = \left[e_i + T_D\left(\frac{e_i - e_{i-1}}{T_m}\right) + \frac{1}{T_I}\sum_{n=p}^{n=i} e_n T_m\right] K_c \qquad (5)$$

Está en las mismas unidades del error (si estoy controlando presión, estará en libras sobre pulgada cuadrada, si estoy controlando temperatura, estará en grados centígrados o en grados Kelvin, si estoy controlando humedad, estará en porcentaje de agua por centímetro cúbico de aire, si estoy controlando velocidad de motores, estará en revoluciones por minuto, etc). Esta expresión en estas unidades, no sirve para ser usada con una interfaz digital PWM. Con esta interfaz, estamos obligados a llevar cualquier unidad, por más extraña que sea, a unidades de tiempo en segundos. Esto implica que a la expresión anterior se le debe encontrar un denominador formalmente deducido, que anule las unidades en que está escrita la expresión del numerador y además se necesita que la expresión resultante sea menor o igual a 1 (recordemos que en las interfaces digitales PWM el ancho del pulso sólo puede ser menor o igual al período de muestreo). Este es el reto.

Para solucionar este acertijo de manera formal para que los estudiantes sientan la belleza de la deducción y no sientan la frustración de que la solución salió de la manga de la camisa. Se seguirá el siguiente proceso:

A. Deducción

1. Si existe un denominador que vuelva la expresión menor o igual a 1, ¿qué significado a nivel físico tendría esto?
2. ¿Cómo se deduce?
3. Como el período de muestreo es el máximo ancho de pulso (Tiempo Útil= Duty Time=DT_i), ¿Cómo hago coherente la expresión deducida en los puntos 1 y 2 con éste concepto?

Todos los sistemas dinámicos controlados inexorablemente, con las primeras inyecciones de energía, siguen la misma trayectoria del sistema dinámico no controlado. Si se observa detenidamente la Fig. 2 ella sugiere que existe un instante en que el sistema controlado se separa del sistema no controlado. ¿Por qué se separa?

Pues porque se ha empezado a dosificar la inyección de energía. Entonces ¿En qué momento empiezo a dosificar la energía?

Esta respuesta ya no es fácil, se necesita recurrir a todo el modelamiento matemático de los sistemas físicos:

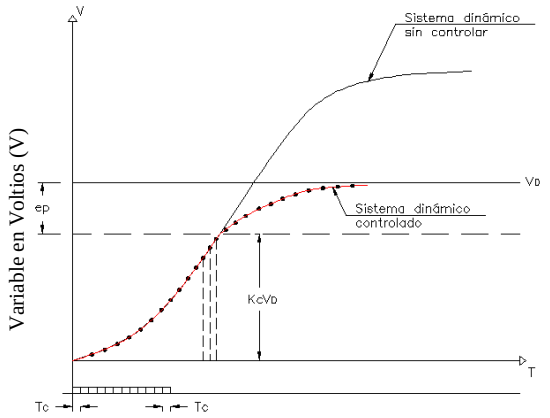

Fig. 2. Separación del comportamiento del sistema dinámico controlado del sistema dinámico no controlado. Tiempo en segundos (T).

Empecemos por recordar los amplificadores operacionales y sus aplicaciones al control automático de variables continuas. (Ver Fig. 3)

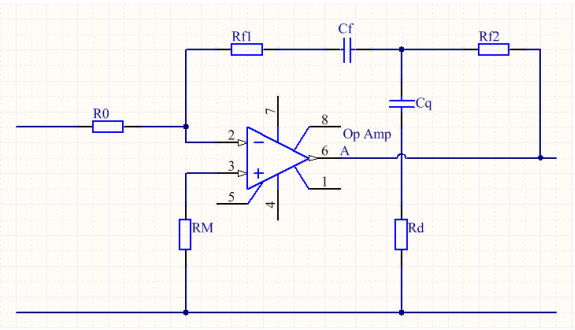

Fig. 3. Control PID Analógico.

En este tipo de control, la salida es un voltaje que debe ser adaptado a nivel de electrónica de potencia para actuar sobre la máquina a controlar. Las constantes K_c, T_I y T_D son introducidas al circuito jugando con los valores de las resistencias y los condensadores del lazo de realimentación y con la ganancia de voltaje del circuito.

Pensemos por un instante ¿cuál será la primera orden de inyección de energía que este sistema entrega a la planta?

Pues esta respuesta es fácil si pensamos que en este primer instante, el tiempo es igual a cero (t=0). Por lo tanto, la expresión que corre un circuito de control con base en amplificadores operacionales para t=0 será:

$$E = K_c\left(e + T_D\frac{de}{dt} + \frac{1}{T_I}\int e dt\right) \qquad (6)$$

El error (e) será el valor deseado (V_D), porque todavía no se ha inyectado energía, hasta ahora va a aparecer la primera orden.

La derivada $\frac{de}{dt}$ es igual a cero $\left(\frac{de}{dt} = 0\right)$ porque todavía no hay diferencia entre el error anterior y el error actual.

La integral $\frac{1}{T_I}\int edt$ es $\left(\frac{V_D * dt}{T_I}\right)$ que si se lleva al mundo digital sería $\left(\frac{V_D * T_m}{T_I}\right)$

Por consiguiente, la energía a inyectar en el instante inicial será:

$$\left[V_D + \left(\frac{V_D * T_m}{T_I}\right)\right] K_c \quad (7)$$

Esto significa que un sistema de control PID con base en amplificadores operacionales intentará llevar la planta hasta el valor (7), que representa la máxima inyección de energía a la planta. A partir de ahí el sistema de control entregará menores dosis de energía y el sistema de control empezará a dosificar la inyección de energía.

Si se llevan los anteriores conceptos al mundo digital creado por los microprocesadores y las interfaces digitales PWM, significa que a partir del anterior valor, los trozos de energía a inyectar son de un ancho de pulso menor que T_m y antes de llegar al anterior valor, los trozos de energía serán iguales al período de muestreo T_m.

Si se observa la Fig. 2 existe una zona entre el valor deseado V_D y el valor $\left[V_D + \left(\frac{V_D * T_m}{T_I}\right)\right] K_c$ en donde se hace la dosificación de energía. En esta zona el error es menor que:

$$V_D - \left[V_D + \left(\frac{V_D * T_m}{T_I}\right)\right] K_c \quad (8)$$

Por consiguiente, los algoritmos de control deberán preguntar permanentemente si el error es menor que la anterior expresión. Si la respuesta a la anterior pregunta es que el error es mayor que la anterior expresión, se deben entregar trozos de energía completos con ancho de pulso iguales a T_m. Si el error (valor deseado – valor medido) es menor que la anterior expresión, se debe calcular la expresión para control PID:

$$DT_i = \left[\frac{e_i + T_D\left(\frac{e_i - e_{i-1}}{T_m}\right) + \frac{1}{T_I}\sum_{n=p}^{n=i} e_n T_m}{V_D - \left[V_D + \left(\frac{V_D * T_m}{T_I}\right)\right] K_c}\right] T_m \quad (9)$$

Y para control PI la siguiente expresión:

$$DT_i = \left[\frac{e_i + \frac{1}{T_I}\sum_{n=1}^{n=i} e_n T_m}{V_D - \left[V_D + \left(\frac{V_D * T_m}{T_I}\right)\right] K_c}\right] T_m \quad (10)$$

Los microprocesadores que se deben escoger para hacer estas operaciones, implican algunas particularidades que se deben tener en cuenta con detenimiento:

1. Deben ser capaces de hacer multiplicaciones y acumular las respuestas parciales para poder hacer las sumatorias. Instrucción de multiplicación con acumulación.
2. El microprocesador debe ser capaz de hacer divisiones.
3. El microprocesador debe tener instrucciones con números fraccionarios o el ingeniero debe tener muy claro cómo se representan números fraccionarios en números binarios si va a usar lenguaje de máquina.

Para facilitar los cálculos el denominador de la anterior expresión se puede simplificar, factorizando V_D , el denominador quedaría:

$$V_D\left[1 - K_c\left(1 + \frac{T_m}{T_I}\right)\right] \quad (11)$$

Donde, para una estrategia de control Proporcional Integral (PI) sobre una planta de primer orden:

$$K_c = \frac{2\zeta\omega_n\tau - 1}{K} \quad (12)$$

$$T_I = \frac{2\zeta\omega_n\tau - 1}{\omega_n^2\tau} \quad (13)$$

Y además, para una estrategia de control Proporcional Integral Derivativa (PID) sobre una planta de segundo orden:

$$K_c = \frac{\omega^2(1+2\alpha\zeta)T_1T_2 - 1}{K_p} \quad (14)$$

$$T_I = \frac{\omega^2 T_1 T_2(1+2\alpha\zeta) - 1}{\alpha\omega^3 T_1 T_2} \quad (15)$$

$$T_D = \frac{T_1T_2\omega(2\zeta+\alpha) - (T_1+T_2)}{\omega^2 T_1 T_2(1+2\alpha\zeta) - 1} \quad (16)$$

Ahora, si el cociente $\frac{T_m}{T_I}$ es mucho menor que 1 (el ingeniero deberá pre-calcular este cociente) el denominador todavía se puede simplificar más, quedaría:

$$V_D[1 - K_c] \quad (17)$$

A este denominador, simplificado o no, de aquí en adelante lo llamaremos *"Coeficiente Andrade"* (CA) que reemplaza a la constante empírica e_p que indicaba colocar en el denominador el 10% del valor deseado sin dar ninguna explicación. Se sugiere a los profesores, salvo que sea estrictamente necesario, evitar el uso de constantes empíricas no deducidas. Cuando las constantes provienen de una deducción, a los estudiantes les queda una sensación de orgullo intelectual de saber "de dónde salen las cosas" y los prepara para enfrentar otros retos con la confianza de encontrar nuevas y originales soluciones.

Por lo tanto las expresiones para inyección de energía usando una interfaz digital PWM agregada a un microprocesador quedarán:

Para control PID

$$DT_i = \left[\frac{e_i + T_D\left(\frac{e_i - e_{i-1}}{T_m}\right) + \frac{1}{T_I}\sum_{n=p}^{n=i} e_n T_m}{CA}\right] T_m \quad (18)$$

Para control PI

$$DT_i = \left[\frac{e_i + \frac{1}{T_I}\sum_{n=1}^{n=i} e_n T_m}{CA}\right] T_m \quad (19)$$

III. DISEÑO DE LOS ALGORITMOS PARA VOLVER REALIDAD LAS ANTERIORES ECUACIONES

Todo el trabajo anterior debe inspirarnos para concretar una serie de instrucciones ejecutables por un microprocesador, un DSP, un microcontrolador o un PLC.

El algoritmo, considerándolo como una secuencia de instrucciones para alcanzar un objetivo, se puede desarrollar en decenas de lenguajes (Python, C, C++, Visual Basic, entre otros) sin embargo, el lenguaje de ejecución más rápida será siempre el lenguaje de máquina.

Cuando se usa una interfaz digital PWM agregada a un microprocesador es inevitable el manejo de números fraccionarios. Este manejo también se enfrentará en lenguaje de máquina para entender hasta los detalles más sutiles del manejo de multiplicaciones de números fraccionarios cuando se calcula la cantidad de energía a inyectar a una planta, la cual debe resultar en fracciones del período de muestreo (T_m).

Ejemplo Didáctico para concretar ideas:

Si se tiene un horno eléctrico trifásico cuya constante de tiempo τ es 200 segundos y la función de transferencia después de la identificación es:

$$G_p(s) = \frac{K}{1+s\tau} \quad (20)$$

Donde $K = 0.8$

Como ω_n máxima es $(1/\tau) = (1/200\ seg.) = 0.005$
Recordar que

$$\omega_n máx = 2\pi F\ máx \quad (21)$$

$$F\ máx = \frac{\omega_n máx}{2\pi} = \frac{0.005}{6.28} = 0.0007962\ Hz \quad (22)$$

El teorema del muestreo obliga a tomar muestras a una tasa de 2F máx que implica un período de muestreo menor o igual a $\left(\frac{1}{2F\ máx}\right)$.

Por lo tanto,

$$T_m \leq \left(\frac{1}{2F\ máx}\right) \quad (23)$$

$$T_m \leq \left(\frac{1}{2(0.0007962\ Hz)}\right) \quad (24)$$

$$T_m \leq (628\ seg.) \quad (25)$$

Por lo tanto, se puede tomar un período de muestreo de 10 segundos (seg) y se cumple de esta manera con el teorema del muestreo.

Ahora,

$$K_c = \frac{2\zeta\omega_n\tau - 1}{K} \quad (26)$$

Por lo tanto,

$$K_c = \frac{2(1)\left(\frac{0.8}{\tau}\right)\tau - 1}{0.8} \quad (27)$$

$$K_c = 0.75 \quad (28)$$

Si el valor deseado es de 77°C y el conversor análogo digital del micro-controlador es de 10 BITS, el voltaje de referencia (+) es 5 voltios (v), el voltaje de referencia (-) es cero (0) voltios y se usa un sensor de temperatura que entrega 10 milivoltios (mv) por cada grado Kelvin, por lo tanto, se espera un número decimal igual a 716 (recordar que para 5 v de entrada al conversor análogo digital, la salida sería del número hexadecimal 3FF que equivale al número decimal 1023).

Para un valor deseado de 77°C, que equivalen a 350°K, se espera que el sensor entregue al conversor análogo digital un voltaje de 3.5 v, que a su vez arrojará un número decimal correspondiente a 716.

Usando la fórmula exacta del coeficiente Andrade (CA):

$$V_D\left[1 - K_c\left(1 + \frac{T_m}{T_I}\right)\right] \quad (29)$$

$$T_m = 0.001\ \text{seg} \quad (30)$$

$$T_I = \frac{2\zeta\omega_n\tau - 1}{\omega_n^2\tau} = \frac{2\left(\frac{0.8}{\tau}\right)\tau - 1}{\left(\frac{0.8}{\tau}\right)^2\tau} = \frac{0.6}{\frac{0.64}{200\ seg.}} = 187.5\ seg. \quad (31)$$

$$\frac{T_m}{T_I} = 5.33x10^{-6} \quad (32)$$

$$K_c = \frac{2\zeta\omega_n\tau - 1}{K} = \frac{0.6}{0.8} = 0.75 \quad (33)$$

$$V_D = 77°C \quad (34)$$

$$CA = (77°C)[1 - (0.75)(1 + 5.33x10^{-6})] \quad (35)$$

$$CA = 19.25°C \quad (36)$$

Usando la fórmula aproximada del Coeficiente Andrade (CA) como:

$$V_D[1 - \mathrm{K_c}] \quad (37)$$

$$CA = (77°C) \ \ [1 - (0.75)] \quad (38)$$
$$CA = 19°C \quad (39)$$

Para el caso se va a utilizar la fórmula aproximada, ya que el conversor análogo digital de 10 BITS no genera una diferencia entre el número binario que representa a 19.25°C con respecto a 19°C. Ahora se tiene que analizar cómo este número en grados centígrados será representado por el microprocesador para hacer los cálculos coherentemente.

Con un conversor análogo digital de 10 BITS para representar voltajes de 0 a 5 v que representan temperaturas entre 0°K (-273°C) y 500°K (227°C), con un sensor que entregue 10 mv por cada grado kelvin, tengo una resolución aproximada de 0.5°C ó 0.5°K. Esto significa que por cada 2 códigos binarios consecutivos se avanza sólo un (1) grado K ó C.

El coeficiente Andrade (CA) fue deducido anteriormente como de 19°C, esto implica que el micro procesador necesita saber el número binario correspondiente que iría en el denominador de la expresión que calcula el ancho del trozo de energía. La anterior necesidad obliga a un pensamiento delicado para deducir el número binario que se va a colocar correspondiente al CA.

Asociemos ideas:

Si el coeficiente Andrade representa una distancia a partir de la cual se empieza a dosificar la energía, el error (valor deseado – valor medido) en este instante debe ser igual al CA (en este caso 19°C). Cuando se haga la resta (valor deseado – valor medido), ésta se hará en números binarios puros, donde el sustraendo es el número binario que entrega el conversor análogo digital de 10 bits. El A/D tiene resolución de 0.5°C, por lo tanto, una diferencia de 1°C será representada con el número binario 0000000010, esto implica que cuando haya una diferencia de 19°C, el número binario será 0000100110. Por lo tanto, el CA será el número decimal 38 o el número hexadecimal 26. Ahora viene algo más delicado, para calcular la expresión de:

$$DT_i = \left[\frac{\mathrm{e_i} + \frac{1}{\mathrm{T_I}}\sum_{n=1}^{n=i} \mathrm{e_n T_m}}{CA}\right] \mathrm{T_m} \quad (40)$$

Será muy conveniente, hasta donde se pueda, no hacer divisiones. Ojalá se pueda calcular la expresión DT_i con sólo multiplicaciones. Para ello, en vez de dividir por CA, se multiplicará por $\left[\frac{1}{CA}\right]$:

$$\frac{1}{CA} = \frac{1}{38} = 0.0263 \quad (41)$$

Este número fraccionario decimal se debe llevar al formato binario con el que va a trabajar el microprocesador. Aquí usted debe repasar los formatos de representación de números decimales en los microprocesadores.

En formato Q_{15} (usando un microprocesador de 16 BITS en el bus de datos), el número decimal 0.0263 es igual a:

$$0.000001101001000 = 0348 \ en \ hexadecimal$$

Ahora,

$$\frac{\mathrm{T_m}}{\mathrm{T_I}} = 5.33x10^{-6} \quad (42)$$

En formato Q_{15} :

$$5.33x10^{-6} = 0.000000000000001 \quad (43)$$

En hexadecimal es:

$$0x0001 \quad (44)$$

Con estos cálculos previos, que después se deben automatizar, se puede empezar a crear el siguiente algoritmo:

```
/* PI */

      mov tamb, W0
      sub tbeb,WREG
      BRA N,CERO
    NO_NE: mov W0,W3
      mov W3,W5;copia del error para W5
      mov W3,W8;copia del error para W8
      mov #0x0026,W1;CA=19 grados centigrados
      mov W1,coef.andrade
      sub coef.andrade,WREG
      BRA N, BYPASS
    NO_N:  mov #0x0348,W1;1/CA
      mov #0x0001,W4;TM/TI
      mac W4*W5,A
      sftac   A,#-1
      sac A,W7
      sftac   A,#1
      mul.UU W1,W7,W2
      mov  W2,W6;tomo solo la parte decimal
      mul.UU  W1,W5,W2;multiplico e subi por 1/CA

      add W2,W6,W1;Sumo las dos partes decimales en formato Q.15 y la dejo en W1
      mov #0x0fff,W7;gravo en W7 el numero binario diseñado para el periodo de muestreo
      mul.uu  W1,W7,W2
      rlc  W2,W2
      rlc  W3,W3;en W3 queda el valor entero para el Pdc1
      mov W3,error
```

```
      mov W3, PDC1
     GOTO TRANS
BYPASS:  mov    #0x0fff,W0
      clr A
      mov W0,error
      mov W0, PDC1
      GOTO    TRANS

CERO:   mov #0x0000,W0
       clr A
      mov W0,error
      mov W0, PDC1
      GOTO    TRANS
.end
```

A continuación se explicará la estrategia algorítmica para llevar a la realidad todos los pensamientos anteriores:

En el encabezado del algoritmo se encuentran los pasos necesarios para llevar el valor deseado a una posición de memoria que se denominará ***tamb*** y el valor medido proveniente del conversor análogo digital se llevará a una posición de memoria llamada ***tbeb*** .

```
mov tamb, W0
      sub tbeb,WREG
      BRA N,CERO
     NO_NE: mov W0,W3
```

Se mueve el valor deseado al registro ***W0*** mediante la instrucción ***mov tamb, W0***.

Se le resta el valor medido y la respuesta se deja en ***W0*** mediante la instrucción ***sub tbeb,WREG***. De esta manera se logra mover los códigos de condición, específicamente el código de condición N. Esto permite definir con claridad si el valor medido es mayor que el valor deseado. Si se presenta esta situación se debe inmediatamente apagar el calefactor. Mediante la instrucción ***BRA N,CERO*** se determina si el valor medido es mayor que el valor deseado. Esta es una ramificación condicional que si se cumple, el programa envía la secuencia de ejecución a la instrucción marcada con el nombre ***CERO***.

```
CERO:   mov #0x0000,W0
       clr A
      mov W0,error
      mov W0, PDC1
```

Esta instrucción graba el número ***0x0000*** en el registro ***W0*** para después sacarlo al registro ***PDC1*** quien es el registro donde se deposita el ancho del pulso en la interfaz digital PWM. Esto significa inyección de energía igual a cero para el calefactor.

Ahora, si el valor medido es menor que el valor deseado, se debe averiguar si el error (valor deseado-valor medido) es mayor que el coeficiente Andrade (CA), porque si es mayor, se deben entregar trozos de energía iguales al período de muestreo. Si el error es menor que CA, se debe entrar a dosificar la energía, y en este caso se debe calcular la ecuación deducida anteriormente para el DT_i. En el siguiente bloque de instrucciones se puede observar que se está en la situación en que el valor medido es menor que el valor deseado y se prepara al microprocesador para hacer cálculos matemáticos. Se sacan copias del error que está inicialmente en ***W0*** y se guardan en ***W3***, ***W5*** y ***W8***, se mueve el número hexadecimal ***0x0026*** a ***W1*** (recordar que este número hexadecimal es igual a CA).

```
 NO_NE: mov W0,W3
      mov W3,W5;copia del error para W5
      mov W3,W8;copia del error para W8
     ; mov W3,W0
      mov #0x0026,W1;CA=19 grados centigrados
      mov W1,coef.andrade
      sub coef.andrade,WREG
```

Ahora, se compara el error contra CA mediante la instrucción ***sub coef.andrade,WREG***.

```
BRA N, BYPASS
    NO_N:  mov #0x0348,W1;1/CA
      mov #0x0001,W4;TM/TI
      mac W4*W5,A
      sftac   A,#-1
      sac A,W7
      sftac   A,#1
```

Si el error es mayor que CA se debe inyectar energía durante un tiempo igual al período de muestreo y esto se hace en la instrucción bautizada como ***BYPASS***. Si el error es menor que CA, se inician los cálculos de la expresión:

$$DT_i = \left[\frac{e_i+\frac{1}{T_I}\sum_{n=1}^{n=i} e_n T_m}{V_D-\left[V_D+\left(\frac{V_D*T_m}{T_I}\right)\right]K_c}\right]T_m \qquad (45)$$

Recordar que el denominador de DT_i es el coeficiente Andrade (CA) y que el cálculo de $\frac{1}{CA}$ en formato decimal Q_{15} es $0348\ en\ hexadecimal$.

La instrucción ***NO_N: mov #0x0348,W1;1/CA*** coloca en $\boldsymbol{W1}$ el número hexadecimal 0348.

La instrucción ***mov #0x0001,W4;TM/TI*** deposita en ***W4*** el resultado de la división de $\frac{T_m}{T_I} = 5.33x10^{-6}$ que en formato Q_{15} es el número $0x0001\ en\ hexadecimal$. Recordar que se está tratando de hacer sólo multiplicaciones en el algoritmo.

La instrucción ***mac W4*W5,A*** multiplica el error (recordar que una copia del error fue guardada en ***W5*** mediante las instrucciones ***NO_NE: mov W0,W3-mov W3,W5;copia del error para W5***) por $\frac{T_m}{T_I}$ y lo acumula en el acumulador ***A*** del microprocesador. Esta operación es la clave para poder hacer integrales por medio de multiplicaciones acumuladas. En otras palabras, se está ejecutando la expresión:

$$\frac{1}{T_I}\int edt \tag{46}$$

Mediante la expresión:

$$\frac{1}{T_I}\sum_{n=0}^{n=i} e_n T_m \tag{47}$$

Que para facilidad del diseño del algoritmo se ha transformado en:

$$\frac{T_m}{T_I}\sum_{n=0}^{n=i} e_n \tag{48}$$

Ya que T_m es una constante.

El resultado de la multiplicación de ***W4*W5*** se guarda en un registro acumulador de 40 Bits. ***W4*** y ***W5*** son registros de 16 Bits que contienen un número decimal en formato Q_{15} y un número entero de 16 Bits respectivamente. La respuesta a esta multiplicación según el fabricante del microprocesador queda depositada en el acumulador ***A***, pero como se está multiplicando un entero por un decimal en formato Q_{15} , se deben despreciar 15 cifras decimales binarias en el resultado y acomodar en el acumulador sólo cifras enteras, ya que al registro PDC1 sólo se pueden llevar cifras enteras para modular el ancho de pulso. Es por lo anterior que se debe a continuación utilizar la instrucción:

sftac A,#-1

Que hace el siguiente juego:

Desplaza la cifra binaria contenida en el acumulador ***A*** un bit a la izquierda buscando que el bit de valencia sub 15 (este bit es el primer bit entero, pero está ubicado como bit de mayor valencia de los 16 bits de menor valencia del acumulador) quede como bit de menor valencia de los 16 bits intermedios del acumulador ***A*** (Ver figura 4) logrando así tener sólo 24 bits de números enteros dentro del acumulador ***A*** (recordar que la operación ***sftac A,#-1*** desplaza un bit a la izquierda a todos y cada uno de los bits del acumulador).

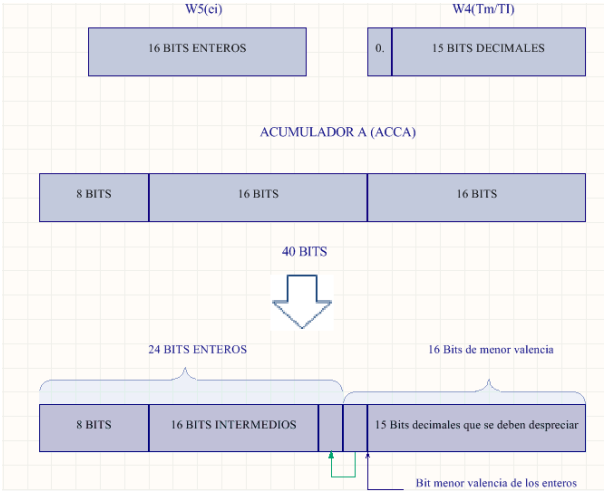

Figura 4. Acomodamiento de los bits decimales y enteros dentro del acumulador de 40 bits.

La siguiente instrucción ***sac A,W7*** saca los 16 bits intermedios del acumulador ***A*** y los guarda en el registro ***W7***. Esto significa que en el registro ***W7*** quedan guardados los 16 bits de menor valencia de 24 bits posibles de números enteros que estaría depositados en el acumulador ***A***. En términos prácticos sólo se van a tener en cuenta estos 16 Bits guardados en ***W7*** (el llenado de estos 16 bits será muy lento y entre más pequeña sea la relación de T_m/T_I, el aporte de la componente integral será más lento).

La instrucción ***sftac A,#1*** recupera el estado original del acumulador para que la sumatoria que genera la integral no se altere. ***sftac A,#1*** significa desplace el acumulador un bit a la derecha. Recordar que hace dos instrucciones se desplazó el acumulador hacia la izquierda 1 bit.

```
mul.UU W1,W7,W2
   mov  W2,W6;tomo solo la parte decimal
```

La instrucción ***mul.UU W1,W7,W2*** multiplica el contenido de los registros ***W1*** y ***W7*** tomados sin signo y deposita el resultado en los registros ***W2*** y ***W3*** concatenados. Recordar que en ***W1*** se tiene el valor numérico en binario puro de la constante $1/CA$ y en ***W7*** se tiene el resultado parcial de la integral que se irá acumulando. Con esta multiplicación se concreta el cálculo numérico de la expresión:

$$\frac{\frac{T_m}{T_I}\sum_{n=0}^{n=i} e_n}{CA} \tag{49}$$

Que todavía es un pedazo de la expresión total a calcular para DT_i.

La instrucción ***mov W2,W6*** toma sólo la parte decimal de la multiplicación anterior que está guardada en ***W2*** y la traslada a ***W6*** para su posterior procesamiento.

mul.UU W1,W5,W2;multiplico e subi por 1/CA
add W2,W6,W1;Sumo las dos partes decimales en formato Q.15 y la dejo en W1

La instrucción siguiente ***mul.UU W1,W5,W2*** multiplica el error sub i (e_i), contenido en ***W5***, por $1/CA$, contenido en ***W1***. Si ahora sumo los resultados de estas dos últimas multiplicaciones tendré un número decimal menor que 1, que para terminar, debe ser multiplicado por el período de muestreo. El resultado de esta multiplicación es por fin el número binario entero que representa a DT_i que debe ser cargado en el registro de ancho de pulso de la interfaz digital PWM. Para hacer esto se necesita de las siguientes 6 instrucciones:

```
mov #0x0fff,W7;gravo en W7 el numero binario diseñado para el periodo de muestreo
    mul.uu  W1,W7,W2
    rlc  W2,W2
    rlc  W3,W3;en W3 queda el valor entero para el Pdc1
    ;mov W3,error
    mov W3, PDC1
```

El número hexadecimal ***0x0fff*** representa el número diseñado que representa un ancho de pulso igual al período de muestreo. Este número se graba en el registro ***W7*** y después se multiplica por el número decimal en formato Q_{15} contenido en el registro ***W1***. Recordar que en ***W1*** quedó el resultado de la expresión:

$$\left[\frac{e_i+\frac{1}{T_I}\sum_{n=1}^{n=i}e_n T_m}{V_D-\left[V_D+\left(\frac{V_D*T_m}{T_I}\right)\right]K_c}\right] \quad (50)$$

Como la multiplicación ***mul.uu W1,W7,W2*** es de un número decimal (***W1*** en formato Q_{15}) por un número entero (***W7***), el resultado es acuñado en la concatenación de los registros ***W2*** y ***W3*** teniendo en ***W2*** la parte de menor valencia. Los 15 Bits de menor valencia de ***W2*** contienen la parte decimal. Como se tiene interés en la parte entera del resultado (ésta está concatenada en 16 Bits de ***W3*** y el Bit $\boldsymbol{W2_{15}}$) se debe entonces hacer una instrucción de ***rlc W2,W2***. Esta instrucción hace un desplazamiento a la izquierda de un (1) bit colocando el bit de mayor valencia ($\boldsymbol{W2_{15}}$) en el carry (C) y el contenido previo del carry pasa a ser el bit de menor valencia ($\boldsymbol{W2_0}$). (Ver Figura 5)

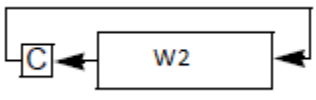

Figura 5. Instrucción rlc

Si ahora se ejecuta la misma instrucción ***rlc W3,W3*** (Ver Figura 6) se logrará colocar a $\boldsymbol{W2_{15}}$ como el bit de menor valencia del registro ***W3*** permitiendo por fin el tener en ***W3*** la parte entera a colocar en el registro de ancho de pulso de la interfaz digital PWM.

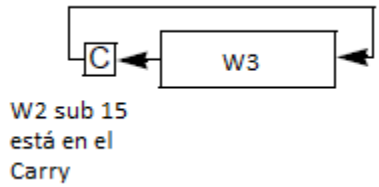

Figura 6. Colocando a $\boldsymbol{W2_{15}}$ como bit de menor valencia de W3

Con la instrucción ***mov W3, PDC1*** por fin se coloca el resultado de la expresión

$$DT_i = \left[\frac{e_i+\frac{1}{T_I}\sum_{n=1}^{n=i}e_n T_m}{V_D-\left[V_D+\left(\frac{V_D*T_m}{T_I}\right)\right]K_c}\right]T_m$$

en el registro de ancho de pulso de la interfaz PWM (PDC1).
Los algoritmos en lenguaje de máquina siempre serán los más rápidos y podrán responder con máxima precisión y velocidad a las exigencias de cualquier máquina, desde un juguete hasta un cohete. Disfrútenlos!!

AGRADECIMIENTOS

Agradezco a la Escuela Colombiana de Ingeniería Julio Garavito por el apoyo a esta investigación.

REFERENCES

[1] Fröhr , F. y Orttenburger, F. (1986). Introducción al control electrónico. Siemens y Marcombo. Pg 171.
[2] Grupo de Desarrollo ARIAN S.A. (). Control PID, una revelación tutorial de los enigmas. ARIAN. Nota técnica 10.
[3] Hart, D.W. (2001). Electrónica de Potencia. Pearson Educación S.A. Madrid. Pg 415
[4] Hayes, J.(1997). Computer Architecture and Organization. McGraw Hill.
[5] Hill and Peterson.(1987) Digital Systems: Hardware Organization and Design. Prentice Hall. Third Edition.
[6] Jacob, J.M. (1989). Industrial Control Electronics: Applications and Design.Prentice Hall. United States.
[7] Leigh, J.R. (1985) Applied digital control: Theory, Design and Implementation. Prentice Hall.
[8] MICROCHIP. (2013). DSPIC30F4012. Microchip Technology Inc
[9] Morris, M. (2003). Diseño Digital. Tercera Edición. Pearson Educación. México
[10] Morris, M. (2004). Digital Logic and Computer Design. Prentice Hall
[11] MOTOROLA. (1995). Thyristor. Device Data: TRIACs & SCRs.
[12] Tan, K.K., Q. Wang y C.C. Hang. (1999). Advances in PID Control. Springer-Verlag London.