IMPLEMENTACIÓN EN "HARDWARE IN THE LOOP" DEL SISTEMA CARRO-PÉNDULO INVERTIDO CON BASE EN EL MICROCONTROLADOR HERCULES RM57L843 DE TEXAS INSTRUMENTS

Marcelino Martínez Aragón

Universidad Tecnológica de la Mixteca

mtz.marcelino@gmail.com

Fermín Hugo Ramírez Leyva

Universidad Tecnológica de la Mixteca

hugo@mixteco.utm.mx

José Anibal Arias Aguilar

Universidad Tecnológica de la Mixteca

anibal@mixteco.utm.mx

Resumen

En este trabajo se presenta el desarrollo de un simulador "Hardware-In-the-Loop" (HIL) de un sistema carro-péndulo invertido con base en la tarjeta Hercules™ RM57Lx Launchpad de Texas Instruments. Para su implementación se obtiene el modelo dinámico del sistema con las ecuaciones de Euler-Lagrange, posteriormente se programa en el MCU, el cual lo ejecuta en tiempo real con un periodo de muestreo de 1 ms. Desde una interfaz gráfica en LabVIEW se capturan la respuesta del sistema además de que se pueden cambiar los parámetros del sistema. Para incrementar el realismo, en un entorno virtual Unity3D se hizo un modelo gráfico del sistema, en el cual se observan los movimientos del carro péndulo. Para verificar la correcta implementación del sistema se hace una comparación con una simulación desarrollada en Matlab/Simulink.

Palabras Claves: Carro-péndulo invertido, microcontrolador, simulador HIL, Unity3D.

Abstract

This paper presents the development of a "Hardware-In-the-Loop" (HIL) simulator of an inverted pendulum-cart system based on the Texas Instruments Hercules ™ RM57Lx Launchpad. For its implementation, the dynamic model of the system is obtained with the Euler-Lagrange equations, later it is programmed in the MCU, which executes it in real time with a sample period of 1 ms. From a graphical interface in LabVIEW, the system response is captured and the system parameters can be changed. To increase the realism, in a virtual environment Unity3D was made a graphic model of the system, in which the movements of the pendulum car are observed. To verify the correct implementation of the system, a comparison is made between a simulation made with Matlab / Simulink.

Keywords: *HIL simulator, microcontroller, pendulum-cart, Unity3D.*

1. Introducción

El trabajo en el laboratorio es indispensable en la formación de cualquier estudiante de ingeniería, sin esta experiencia es difícil que se comprenderían plenamente sus aplicaciones y limitaciones [Usenmez, 2014]. A pesar de sus virtudes, la formación de un laboratorio convencional tiene problemas tales como: espacio, costo, disponibilidad, límite de plantas, etc. Una solución a éstas es implementar controladores reales sobre hardware en interacción con modelos físicos simulados en tiempo real, que permitan dar una idea muy aproximada al comportamiento de un sistema [Martinez, 2013]. Este tipo de implementación es comúnmente llamada Hardware-In-The-Loop (HIL), un simulador HIL es una técnica de prueba que simula el comportamiento de entradas y salidas de un sistema físico que es conectada a un controlador en tiempo real [Washington, 2008]. Una de sus aplicaciones más importantes es simular sistemas que son difíciles de construir o bien su costo de fabricación es extremadamente alto [Kruckenberg, 2011], [Subramanian, 2012], [Brito, 2014]. Ésta técnica de simulación es ideal para sustituir a una planta real en un laboratorio.

Una simulación HIL requiere de hardware que se encargue de procesar las entradas, analógicas o digitales, ejecutar el algoritmo en tiempo real del modelo

que emula y generar las salidas. En el mercado existen diversas tecnologías que permiten realizar esta tarea como son la xPC target de mathworks [Romero, 2017] y dSPACE [Zhu, 2009], si bien son tarjetas especializadas para trabajar en tiempo real son altamente costosas. Actualmente se tiene en el mercado plataformas de desarrollo con microcontroladores de bajo costo, las que son muy rápidas y se pueden usar para la realización de sistemas HIL. Un dispositivo de este tipo es la tarjeta de evaluación Hercules Launchpad RM57L de la firma Texas Instruments.

El carro-péndulo es una de las plantas de laboratorio más utilizados para mostrar técnicas de control no lineal. Este sistema es la base para aplicaciones como el control de cohetes y control antisísmico de edificios [Fantoni, 2002]. Muchos sistemas robóticos se han diseñado con base en el péndulo invertido sobre ruedas tal como el Segway. Contar con una planta de este tipo es costoso y difícil de construir. Una alternativa es realizar este sistema en una plataforma HIL, el cual responde en tiempo real. Para incrementar su realismo se puede mostrar en un ambiente gráfico, los movimientos del mismo lo que mejora la experiencia del usuario. Algunos ambientes para el desarrollo ambientes gráficos virtuales es Unity3D y para el intercambio de información y configuración es el LabVIEW.

LabVIEW (Laboratory Virtual Instrument Engineering Workbench), es un entorno de programación desarrollado por National Instruments en el cual se crean programas usando una notación gráfica (Lenguaje G). Es un programa interactivo diseñado específicamente para científicos e ingenieros que desarrollan sistemas de medidas y control. Puede tomar mediciones, adquirir y analizar datos y presentar resultados al usuario. Como LabVIEW tiene una interfaz gráfica muy fácil de manejar, se puede crear exactamente el tipo de instrumento virtual que se necesite [Travis, 2006].

Unity 3D es un software para desarrollar videojuegos, permite la creación de juegos en múltiples plataformas como son PlayStation, Xbox, PC, iOS, Android, etc. Los resultados que se obtienen se da un gran realismo y sin demasiadas complicaciones en su programación. Aun teniendo una versión gratuita del software, la calidad de los juegos que se puede desarrollar es excelente. Estas características convierten a Unity en una excelente opción para los

desarrolladores independientes y los estudios de videojuegos [Arrioja, 2013]. Se pueden crear aplicaciones 3D en tiempo real y multimedia, algunos juegos famosos desarrollados en Unity son: Assassin's Creed Identity, Temple run y Angry Birds 2.

En este trabajo se muestra la implementación un simulador HIL del sistema carro-péndulo invertido de bajo costo, donde se prueba el comportamiento de la planta en lazo abierto. Para incrementar el realismo se desarrolló una interfaz gráfica en LabVIEW y Unity3D. En LabVIEW se mostrarán las gráficas de la posición y velocidad del péndulo y del carro, emulado en el Microcontrolador. En Unity se mostrará el movimiento animado de traslación y rotación del sistema carro-péndulo.

2. Métodos

En la figura 1 se muestra el diagrama a bloques utilizado para el simulador HIL del sistema carro-péndulo, el bloque de color azul representa la tarjeta de desarrollo Hercules Launchpad RM57Lx. Los bloques de color blanco son los módulos internos que se programaron en el microcontrolador, los bloques externos de color verde son los módulos con los que la tarjeta interactúa y compartirá datos a través del puerto serial. Los datos que se compartirán son la posición del péndulo y la posición del carro. El módulo *ADC* convierte el voltaje (de control) analógico a digital para su posterior uso en las ecuaciones. El bloque *Modelo dinámico* contiene el modelo en variables de estado donde la entrada de control es μ. El bloque siguiente soluciona estas ecuaciones diferenciales mediante el método numérico de Euler obteniendo como salida $x, \dot{x}, \theta, \dot{\theta}$ donde x y $\dot{x}$ son la posición y velocidad del carro, θ y $\dot{\theta}$ son la posición y velocidad angular del péndulo. En el módulo *DAC(PWM)* se simula las salidas analógicas de las posiciones y velocidades, usando una señal PWM, estas señales pasan por un bloque de *acondicionamiento de señales* para adaptar los voltajes de la tarjeta al controlador, de igual forma el controlador devuelve el voltaje de control y pasa por el bloque de acondicionamiento. Finalmente, las posiciones y velocidades se

envían por el puerto serial a una interfaz gráfica en donde se muestra el movimiento del sistema carro-péndulo.

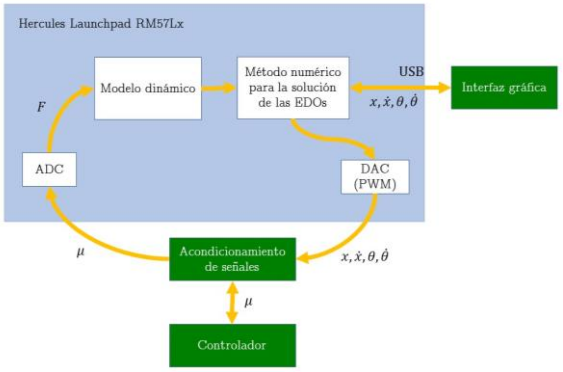

Figura 1 Diagrama a bloques general del simulador HIL.

Modelo Dinámico

Un sistema carro-péndulo invertido consiste en un péndulo colocado sobre un carro, el péndulo oscila libremente sobre un eje y el carro se mueve a través de una banda a lo largo del eje x, esta banda lo mueve un motor de CD tal como se muestra en la figura 2. El objetivo de control es mantener al péndulo en una posición vertical superior.

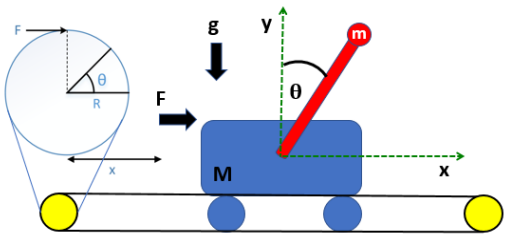

Figura 2 Sistema carro-péndulo.

Para poder simular al sistema o aplicar algunas técnicas de control, es necesario conocer el modelo dinámico del mismo. El modelo se obtiene partiendo del Lagrangiano ($L = K - P$), donde K es la energía cinética y P es la energía potencial del sistema carro-péndulo. Tomando la energía cinética del carro más la del péndulo y restando la energía potencial, se obtiene la ecuación de Lagrange, ver ecuación 1.

$$L=\frac{1}{2}(M+m)\dot{x}^2+m\dot{x}l\cos(\theta)\dot{\theta}+\frac{1}{2}(I+m)l^2\dot{\theta}^2-mgl\cos(\theta) \quad (1)$$

En la figura 2 se muestran las fuerzas que actúan en el carro-péndulo considerando los parámetros de la tabla 1, estos datos fueron obtenidos de la hoja de datos de la empresa Feedback Instruments [Feedback Instruments].

Tabla 1 Parámetros del carro-péndulo.

Parámetro	Descripción	Valor	Unidades
M	Masa del carro.	1.12	kg
m	Masa del péndulo.	0.11	kg
l	Distancia de la articulación al centro de gravedad del péndulo.	0.3434	m
I	Momento de inercia del péndulo en relación con su centro de gravedad.	0.0136	kgm^2
g	Aceleración debida a la gravedad.	9.81	m/s^2
x	Distancia del centro de masa del carro respecto a su posición inicial.		m
θ	Ángulo que forma el péndulo con respecto a la vertical.		rad
F	Fuerza aplicada sobre el carro		N
F_p	Fricción del péndulo	0.007	Nms/rad
F_c	Fricción del carro	0.05	Ns/m

La ecuación 2 y ecuación 3 son las ecuaciones de Euler-Lagrange. Donde x indica la posición del carro, θ indica la posición del péndulo, D es la función de disipación de Rayleigh y F es la fuerza que produce el motor de DC para mover al carro.

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right)-\frac{\partial L}{\partial x}+\frac{\partial D}{\partial \dot{x}}=F \quad (2)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right)-\frac{\partial L}{\partial \theta}+\frac{\partial D}{\partial \dot{\theta}}=0 \quad (3)$$

Se obtienen las derivadas parciales de la ecuación 1 y se sustituye en las ecuaciones 2 y 3, obteniendo las ecuaciones no lineales que rigen el comportamiento del sistema.

$$(M+m)\ddot{x}+ml\left[-\sin(\theta)\dot{\theta}^2+\cos(\theta)\ddot{\theta}\right]+F_c\dot{x}=F \quad (4)$$

$$(ml^2 + I)\ddot{\theta} + ml\left[\ddot{x}\cos(\theta)\right] - mgl\sin(\theta) + F_p\dot{\theta} = 0 \quad (5)$$

Un motor de DC, es quien produce la fuerza (F) necesaria para mover al carro, por lo tanto, se acopla el modelo dinámico del motor en el modelo del carro-péndulo. La ecuación 6 y 7 son del modelo dinámico de un motor de DC.

$$L_a \frac{di_a}{dt} = u - R_a i_a - k\omega \quad (6)$$

$$J\frac{dw}{dt} = ki_a - B\omega - \tau_L \quad (7)$$

Los parámetros y variables del motor se describen en la tabla 2.

Tabla 2 Parámetros del motor de DC del sistema carro-péndulo.

Parámetro	Descripción	Valor	Unidades
R_a	Resistencia de armadura: Es el valor de la resistencia al paso de corriente de la carcasa del motor.	3.9	Ω
L_a	Inductancia del circuito de armadura: Inductancia del embobinado que está dentro del rotor.	8.9E-3	mH
K_e	Constante eléctrica: Relaciona la velocidad angular de la flecha y el voltaje generado.	53.5E-3	v_s/rad
K_m	Constante mecánica: Relaciona la corriente eléctrica del circuito y el torque generado.	53.5E-3	Nm/A
K	Simboliza la igualdad entre $K_e = K_m$	53.5E-3	
B	Coeficiente de fricción viscosa rotacional: Representa la oposición al movimiento del motor.	50E-6	Nms/rad
J	Momento de inercia: Es la inercia rotacional asociada a la geometría del rotor.	0.11E-4	kgm^2
i_a	Corriente del circuito de armadura.		mA
ω	Velocidad angular: Velocidad angular a la que gira el rotor.		rad/s
τ_L	Par de carga constante desconocido.		Nm
μ	Entrada de control: Es el voltaje que se aplica en las terminales del motor.		V
R	Radio del disco montado en el eje del motor	0.02616	m

Igualando ambas dinámicas, se obtiene la ecuación acoplada del sistema carro-péndulo invertido, ecuación 8.

$$\left[\frac{J}{R^2}+(M+m)\right]\ddot{x}+\left[\frac{k^2}{R^2R_a}+\frac{B}{R^2}+F_c\right]\dot{x}+ml\left[\ddot{\theta}\cos(\theta)-\dot{\theta}^2\sin(\theta)\right]=\frac{k}{RR_a}u \tag{8}$$

Despejando $\ddot{\theta}$ y $\ddot{x}$ de la ecuación 5 y ecuación 8 se obtiene ecuaciones 9 y 10.

$$\ddot{\theta}=-\frac{F_p}{\beta}\dot{\theta}-\frac{a_3^2}{a_1\beta}\dot{\theta}^2\sin(\theta)\cos(\theta)+\frac{a_5}{\beta}\sin(\theta)+\frac{a_2a_3}{a_1\beta}\cos(\theta)\dot{x}-\frac{a_3b_1}{a_1\beta}\cos(\theta)*u \tag{9}$$

$$\ddot{x}=-\frac{a_2}{a_1}\dot{x}-\frac{a_3}{a_1}\left[\ddot{\theta}\cos(\theta)-\dot{\theta}^2\sin(\theta)\right]+\frac{b_1}{a_1}u \tag{10}$$

Donde

$$\beta=a_4-\frac{a_3^2}{a_1}\cos^2(\theta),\ a_1=\left[\frac{J}{R^2}+(M+m)\right],\ a_2=\left[\frac{k^2}{R^2R_a}+\frac{B}{R^2}+F_c\right],\ a_3=ml,\ a_4=ml^2+I,$$

$$a_5=mgl \text{ y } b_1=\frac{k}{RR_a}.$$

El modelo dinámico se representa en variables de estado, ver ecuación 11, tomando como referencia: $x_1=x$-posición del carro, $x_2=\dot{x}$-velocidad del carro, $x_3=\theta$-posición angular del péndulo,$x_4=\dot{\theta}$-velocidad angular del péndulo.

$$\begin{aligned}
&\dot{x}_1=x_2\\
&\dot{x}_2=-\frac{a_2}{a_1}x_2-\frac{a_3}{a_1}*\left[\left(-\frac{F_p}{\beta}x_4-\frac{a_3^2}{a_1\beta}x_4^2\sin(x_3)\cos(x_3)+\frac{a_5}{\beta}\sin(x_3)+\right.\right.\\
&\left.\left.\frac{a_2a_3}{a_1\beta}\cos(x_3)x_2-\frac{a_3b_1}{a_1\beta}\cos(x_3)*\mathrm{u}\right)\cos(x_3)-x_4^2\sin(x_3)\right]+\frac{b_1}{a_1}u\\
&\dot{x}_3=x_4\\
&\dot{x}_4=-\frac{F_p}{\beta}x_4-\frac{a_3^2}{a_1\beta}x_4^2\sin(x_3)\cos(x_3)+\frac{a_5}{\beta}\sin(x_3)+\frac{a_2a_3}{a_1\beta}\cos(x_3)x_2-\\
&\frac{a_3b_1}{a_1\beta}\cos(x_3)u
\end{aligned} \tag{11}$$

Programación de la Tarjeta Hercules RM57L

El kit de desarrollo Hercules™ RM57Lx Launchpad™ de Texas Instruments se basa en el microcontrolador RM57L843 de 32 bits, integra dos CPU ARM Cortex-

R5F de punto flotante, operando en tiempo real, el cual ofrece una velocidad de 1.66 DMIPS/MHz y una frecuencia de operación de hasta 330 MHz. El dispositivo tiene 4 MB de flash, 512 kB de RAM de datos, dos módulos convertidores analógico digital de 12 bits, siete módulos mejorados ePWM, Interfaces múltiples de comunicación (Ethernet, CAN, MibSPI, UART), entre otras [Texas Instruments, 2017]. Por lo cual es capaz de resolver el modelo dinámico del carro péndulo en 1 ms. Para que el simulador HIL funcione correctamente, la transmisión ejecución del modelo dinámico debe ser en tiempo real. La comunicación entre el Hércules y la interfaz gráfica o el Unity no tienen este requerimiento y pueden funcionar como sniffer.

Un sistema en tiempo real es capaz de realizar varias operaciones simultaneas al mismo tiempo. Usualmente en el uso de microcontroladores, la ejecución en paralelo se consigue como la sucesión rápida de actividades secuenciales. Se cuenta con varias técnicas para ejecutar tareas en paralelo, una de ellas es mediante interrupciones, con base a ésta se pueden comunicar los módulos de la tarjeta, mediante banderas (flags), variables, contadores (timers).

En la figura 3 se muestran las variables de entrada y salidas que emula la tarjeta Hercules RM57Lx. El microcontrolador recibe la señal de control, que en este caso es el voltaje del motor de CD y en función de ésta resuelve el modelo dinámico para obtener las posiciones y velocidades del carro y del péndulo, ambas posiciones son enviadas a una interfaz gráfica, donde muestra los movimientos del carro-péndulo como historial o los movimientos en la interfaz de Unity. Si la entrada de control es cero, el modelo se resuelve a partir de las condiciones iniciales.

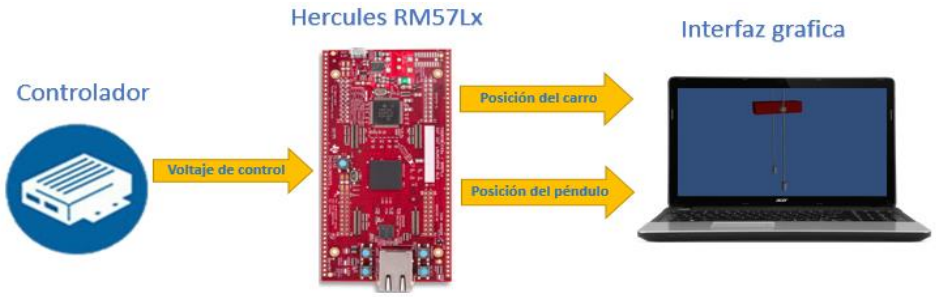

Figura 3 Entradas y salidas del simulador HIL.

Para aproximar la solución del sistema de ecuaciones diferenciales del modelo dinámico, se utiliza el método numérico de Euler. Para lo cual se discretiza el sistema de ecuaciones obtenidos en ecuación 11, obteniendo en la ecuación 12 el siguiente sistema de ecuaciones:

$$\begin{aligned}
&x1k = x1a + h*x2a \\
&x2k = x2a + h*(-(a2/a1)*x2a - (a3/a1)*x4k*cos(x3a) + (a3/a1)* \\
&\quad x4a^2*sin(x3a) + (b1/a1)*u) \\
&x3k = x3a + h*x4a \\
&x4k = x4a + h*(-(Fp/\beta)*x4a - (a3^2/(a1*\beta))*x4a^2*sin(x3a)* \\
&\quad cos(x3a) + ((a5*sin(x3a))/\beta) + ((a2*a3)/(a1*\beta))*cos(x3a)*x2a - \\
&\quad ((a3*b1)/(a1*\beta))*cos(x3a)*u);
\end{aligned} \tag{12}$$

Donde $a1,a2,a3,a4,a5,b1$ y β son constantes. $x1k,x2k,x3k,x4k$ son las variables actuales por calcular, son las variables anteriores calculadas, las condiciones iniciales de estas variables son 0. u es la entrada de control, $x1k$ es la posición del carro, $x2k$ velocidad del carro, $x3k$ es la posición angular del péndulo, $x4k$ es la velocidad angular del péndulo.

Estas ecuaciones se programaron en la tarjeta Hercules RM57Lx, se usó una interrupción del contador, de tal manera que cada 1 ms se produzca una interrupción. El MCU resuelve el sistema de ecuaciones con la que calcula la posición del carro y la posición del péndulo. Fuera de la interrupción por puerto serie envía estos datos a LabVIEW o a Unity3D. En el diagrama de bloques de la figura 4 se muestra este proceso de interrupción.

Como se observa en la figura 3, la tarjeta de desarrollo recibe una señal de control, esta señal es un voltaje analógico que proviene de un controlador externo. Es necesario convertir esta señal analógica a digital para poder procesarla junto al modelo dinámico del sistema carro-péndulo, entonces se utiliza el módulo mejorado ADC de la tarjeta para llevar a cabo este proceso.

Interfaz Gráfica en LabVIEW

Para visualizar el comportamiento de las posiciones del sistema, configurar los parámetros, verificar el funcionamiento de control, se desarrolló un panel virtual en

LabVIEW. El programa realizado activa el puerto de comunicación, envía un carácter a la tarjeta y hasta entonces recibe la posición del péndulo, la posición del carro y el voltaje de control, estos en una cadena de datos separados por una coma, los separa y grafica para finalmente cerrar el puerto de comunicación serial. Si se presiona el botón de 'enviar parámetros' LabVIEW envía una cadena de datos separados por punto y coma a la tarjeta de desarrollo.

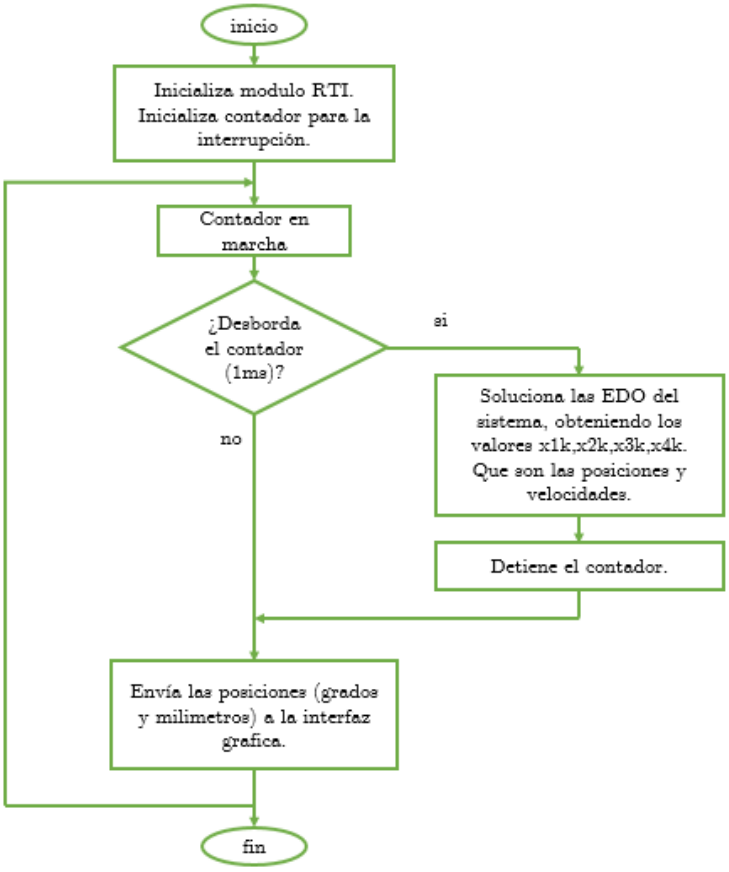

Figura 4 Diagrama de flujo del manejo de interrupción.

En la figura 5 se muestra la interfaz gráfica elaborada en LabVIEW, consta de tres paneles: Graficas: en este panel se muestra en indicadores numéricos el voltaje de control (V), la posición del péndulo (grados) y la posición del carro (cm). Tiene dos subpaneles para mostrar las gráficas, la de la izquierda muestra la posición del péndulo en tiempo real y la gráfica derecha muestra la misma posición desde el tiempo inicial hasta que se finalice la simulación.

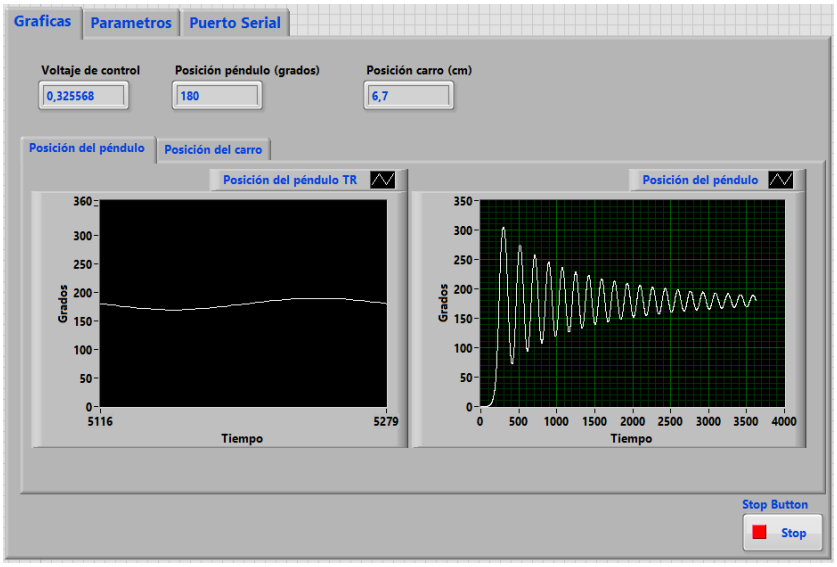

Figura 5 Interfaz gráfica desarrollada en LabVIEW.

Parámetros: en este panel se muestran los parámetros del sistema carro-péndulo de acuerdo con la tabla 1 y tabla 2. En este subpanel se puede modificar algún parámetro, dando clic en enviar, figura 6, se envían los parámetros a la tarjeta Hercules, estos datos se actualizan en tiempo de ejecución y el simulador HIL calcula las variables de salida en función de los nuevos parámetros.

Puerto serial: en este panel se configura el puerto de comunicación serial, la velocidad de transmisión de datos, bits de parada, etc.

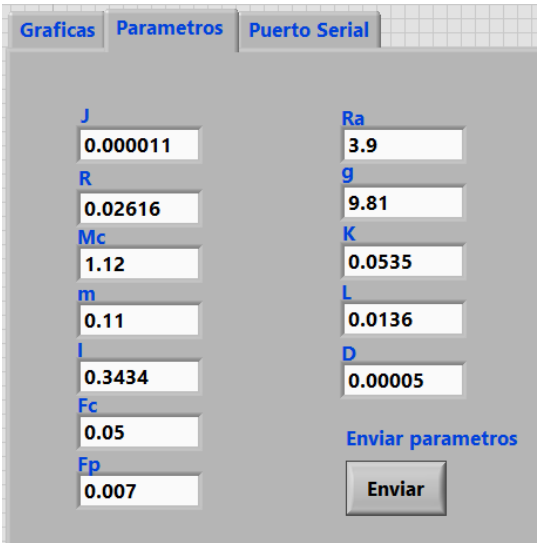

Figura 6 Parámetros modificables del carro-péndulo.

Interfaz Gráfica en Unity3D

Para darle un mayor realismo y sobre todo que el usuario se aliente a profundizar en el ámbito de control, se desarrolló un entorno grafico en el motor de juegos Unity3D. EL microcontrolador envía los valores de las posiciones, el entorno grafico en cada iteración posiciona el carro y el péndulo en función de estos datos, dando como resultado la animación de rotación y traslación, en la figura 7 se observa el diseño del sistema carro-péndulo.

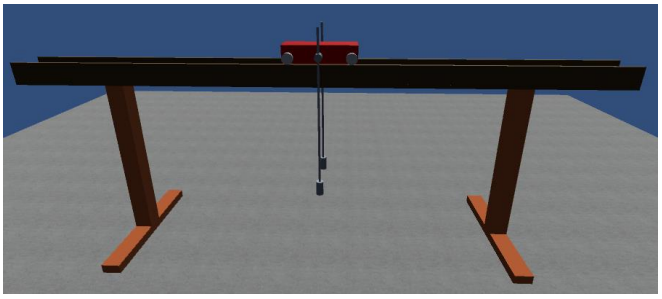

Figura 7 Carro-péndulo diseñado en Unity3D.

3. Resultados

Para verificar el desempeño del simulador, se diseña el modelo dinámico en MATLAB/Simulink como lo muestra la figura 8, con la finalidad de comparar la posición del péndulo que entrega Simulink y la que entrega el simulador HIL.

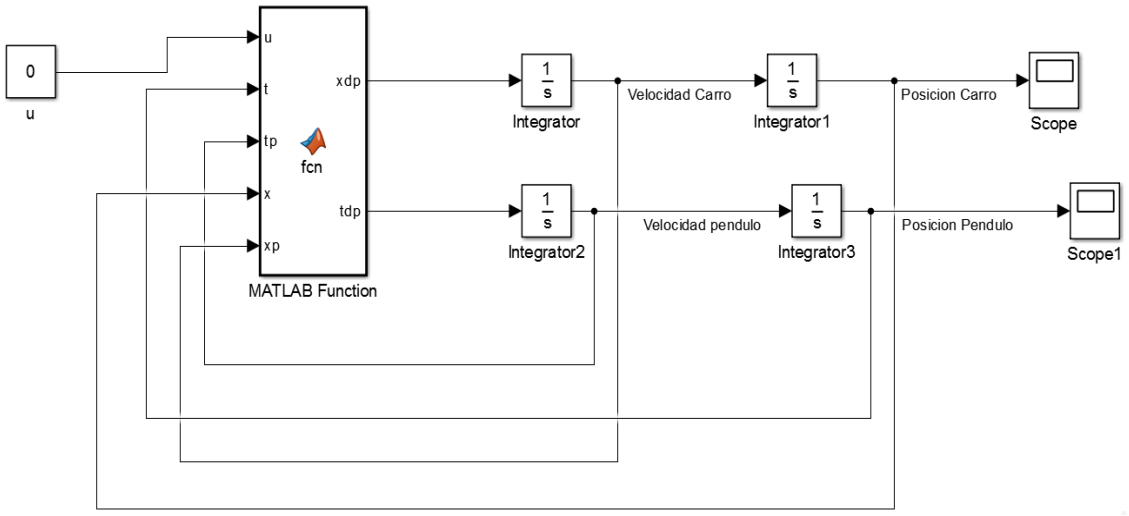

Figura 8 Modelo dinámico desarrollado en MATLAB/Simulink.

Se configuró Simulink para que solucione las ecuaciones diferenciales con el método de Euler con un tiempo de muestreo de 10 ms. De igual forma se aplicaron estas restricciones al programar la tarjeta de desarrollo.

Físicamente si un péndulo con fricción se deja caer desde la vertical superior, oscilará bajo la acción gravitatoria y se detendrá en algún momento, con base a esta afirmación, a la entrada de control del modelo se le asignó el valor de cero (u=0) como lo muestra el bloque de la figura 8. El resultado se muestra en la figura 9, se observa la señal de posición del péndulo, esta grafica indica que inicialmente se suelta el péndulo desde la posición vertical superior (0 grados) esta comienza a oscilar y en el transcurso del tiempo se va acercando (deteniendo) hasta alcanzar el valor 180 grados.

Se puede observar que las salidas son muy semejantes, a cómo avanza el tiempo, la señal del simulador HIL tiene un pequeño error en amplitud lo que atrasa la convergencia a 180 grados.

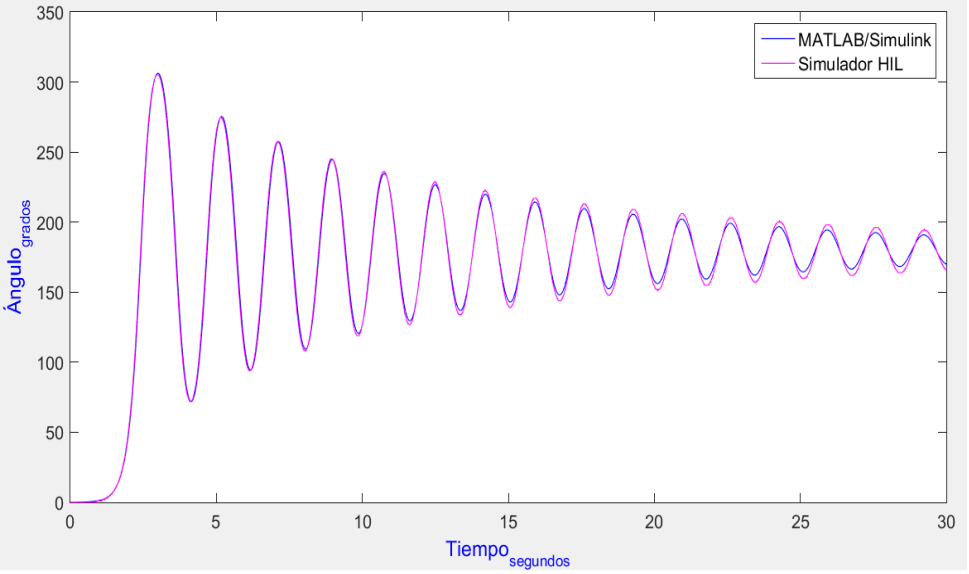

Figura 9 Grafica comparativa con un periodo de 10 ms.

Se realizó una segunda comparación, pero en esta vez con un tiempo de muestreo de 1 ms. En la figura 10 se muestra la comparación de señales, tienen más semejanza que la gráfica anterior, además el tiempo que tarda en converger a 180 grados, es menor.

En la figura 11, se muestra el simulador HIL embebido en la tarjeta de desarrollo Hercules Launchpad RM57Lx enviando datos a la interfaz gráfica diseñada en Unity.

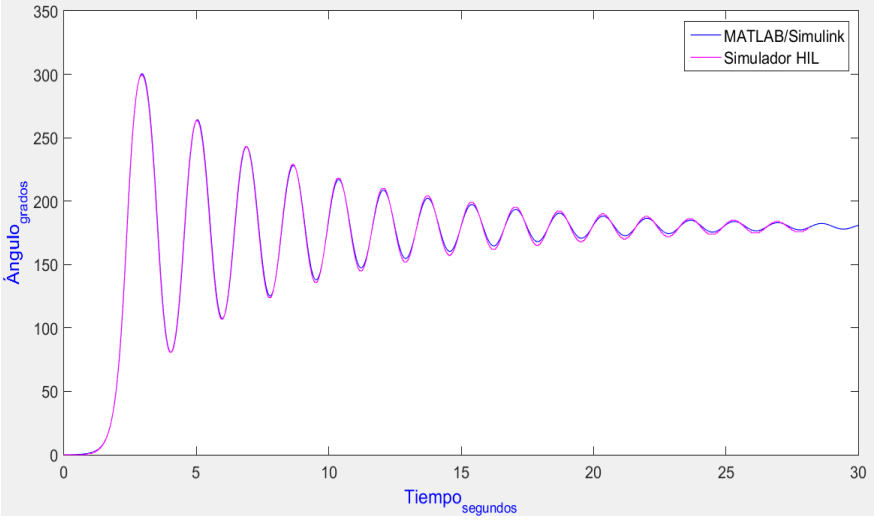

Figura 10 Grafica comparativa con un periodo de 1 ms.

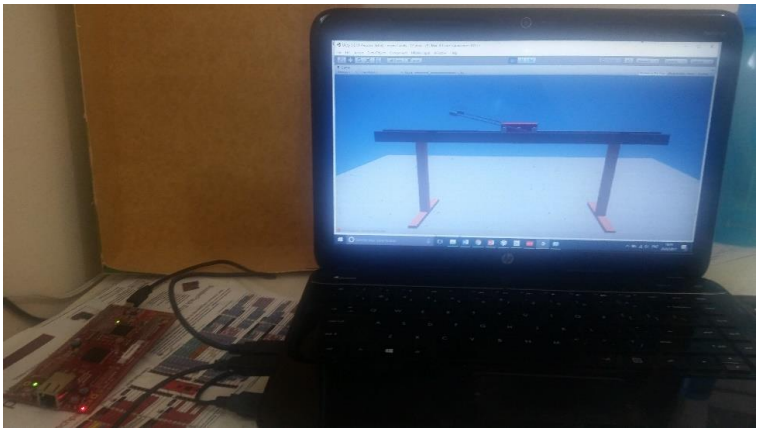

Figura 11 Simulador HIL con Unity en tiempo de ejecución.

4. Discusión

En las gráficas comparativas obtenidas con 10 ms y 1 ms son muy semejantes, significa que los resultados de los cálculos de senos, cosenos, sumas y divisiones realizadas por Matlab/Simulink y el microcontrolador son muy cercanas y en ocasiones iguales, en Matlab/Simulink estas operaciones no son problema puesto que el software corre en un procesador AMD A6, en el microcontrolador una

frecuencia de operación de 330 MHz y debido a la unidad doble de punto flotante, los cálculos tienen mayor precisión y velocidad.

5. Conclusiones

Se desarrolló un simulador HIL del sistema carro-péndulo con una interfaz gráfica desarrollada en LabVIEW y Unity3D, se probó de manera satisfactoria el funcionamiento en lazo abierto del sistema, dando mejores resultados al configurar el tiempo de muestreo en 1 ms.

Un aporte importante es el desarrollo de la simulación del sistema en tiempo real sobre la tarjeta Hercules Lacunhpad RM57Lx, se aprovecha la capacidad de este procesador que es de muy bajo costo a diferencia de hardware especializado pero costoso.

Al desarrollar el entorno virtual en Unity3D da un mejor realismo al simulador puesto que el diseño es en 3D, además permite explorar todo el escenario, se puede dar un giro de 360 grados visualizando cada detalle del carro-péndulo, con ello se logra que el usuario o alumno tenga mayor interés en el área de control.

Como trabajo a futuro, se propone realizar toda la interfaz en Unity3D o bien que LabVIEW transmita los datos a través del protocolo TCP/IP a Unity para que el simulador sea más funcional.

6. Bibliografía y Referencias

[1] Arrioja, N. Unity, Fox Andina. Buenos Aires, 2013.

[2] Brito, M. A. & Gonzalez, S. A. R., Simulación en tiempo real del campo magnético terrestre para una misión orbital. pp. 322–327, 2014.

[3] Fantoni, I., Lozano, R. & Sinha, S., Non-linear Control for Underactuated Mechanical Systems. vol. 55. no. 4. Londres: Springer, 2002.

[4] Feedback Instruments, Digital Pendulum System. vol. 44. no. 1160. FI Ltd, England. pp. 18.

[5] Martinez, J. C. & Andrade, J., Implementación de controladores en Sistemas Retroalimentados usando electrónica embebida y simulación Hardware In The Loop. Universidad Tecnológica de Pereira. Pereira, 2013.

[6] Kruckenberg, J., Fault Diagnosis and Hardware in the Loop Simulation for the EcoCAR Project.Ohio State University, 2011.

[7] Romero, J., Rodríguez, E. & Bernal, E., Desarrollo de una planta piloto basada en xpc target. Rev. Ingeniería, Matemáticas y Ciencias de la Información. vol. 4. pp. 35–46, 2017.

[8] Subramanian, S., George, T. & Thondiyath, A., Hardware-in-The-Loop verification for 3D obstacle avoidance algorithm of an underactuated flat-fish type AUV. 2012 IEEE Int. Conf. Robot. Biomimetics. ROBIO 2012 - Conf. Dig. pp. 545–550, 2012.

[9] Texas Instruments, Hercules RM57Lx LaunchPad Development Kit. 2016. [Online]. Available: http://www.ti.com/tool/LAUNCHXL2-RM57L, Accessed: 02-May-2017.

[10] Travis, J. & Kring, J., LabVIEW For Everyone: Graphical Programming Made Easy and Fun. 3rd ed. Prentice Hall, 2006.

[11] Usenmez, S., Yaman, U., Dolen, M. & Koku, A. B., A new hardware-in-the-loop simulator for control engineering education. IEEE Glob. Eng. Educ. Conf. EDUCON, pp. 1–8, 2014.

[12] Washington, C. & Delgado, S., Improve Design Effici iency and Test Capabilities with HIL Si imulation. IEEE Autotestcon. no. September, pp. 8–11, 2008.

[13] Zhu, Y., Hu, H., Xu, G. & Zhao, Z., Hardware-in-the-Loop Simulation of Pure Electric Vehicle Control System. Int. Asia Conf. Informatics Control. Autom. Robot. pp. 254–258, 2009.