Tutorial de trabajo con placa EDU CIAA

Suarez Facundo,Ezequiel Duperre,Jonathan Saez
Departamento de Física
Universidad Nacional de San Luis
San Luis, Argentina
Email: fmsuarez92@gmail.com

***Abstract*—En este trabajo se pretende desarrollar un tutorial que permita al usuario neófito poder iniciar sus primeros proyectos usando RTOS y comprender la arquitectura de los procesardores Cortex.**

***Index Terms*—Cortex M4, Firmware, Makefile, microprocesador,RTOS.**

I. Introduction

Para el desarrollo de sistemas embebidos es usual iniciarse utilizando placas de pruebas basadas en microcontroladores PIC, sin embargo, el avance del desarrollo de las arquitecturas de los procesadores Cortex, ha llevado a la necesidad del análisis de diversos sistemas embebidos basados en dicho procesador. La placa EDU CIAA nos brinda el ambiente adecuado para la comprensión y para la programación de códigos basados en la arquitectura ARM.

A. Descripción de la placa EDU CIAA

La CIAA es una plaqueta electrónica provista de un microcontrolador y puertos de entrada y salida, cuyo diseño se encuentra disponible en Internet, dicha placa fue concebida para ser utilizada para sistemas de control de procesos productivos, agroindustria, automatización, entre otras; es notable destacar que gracias a la posibilidad del acceso a la información de dicha plataforma, cualquier empresa que desee utilizarla para la elaboración de sus productos puede rediseñarla; de modo que esto fomenta el diseño y la fabricación nacional de sistemas electrónicos.

La placa EDU CIAA es la versión educativa de la placa, diseñada con el propósito de conseguir una plataforma base para el desarrollo de proyectos educativos, en este caso, se busca proporcionar las bases del desarrollo de códigos utilizando RTOS. Sobre la Figura 1 3 se proporciona el diagrama en bloques de la placa, puede observarse que la placa cuenta con 2 puertos micro-USB (uno para aplicaciones y debugging, otro para alimentación); 4 salidas digitales implementadas con leds RGB, 4 entradas digitales con pulsadores; 1 puerto de comunicaciones RS485 con bornera. La Figura 2 2 nos muestra una imagen frontal de la placa; nótese la presencia de dos puertos sobre los cuales se ubican los pines correspondientes a la placa, la Figura 3 **??** ilustra el distribución de dichos pines sobre cada puerto.

Sobre el puerto P1, se ubican los siguientes módulos:

1) 3 entradas analógicas ($ADC0_1, 2y3$)
2) 1 salida analógica (DAC0).

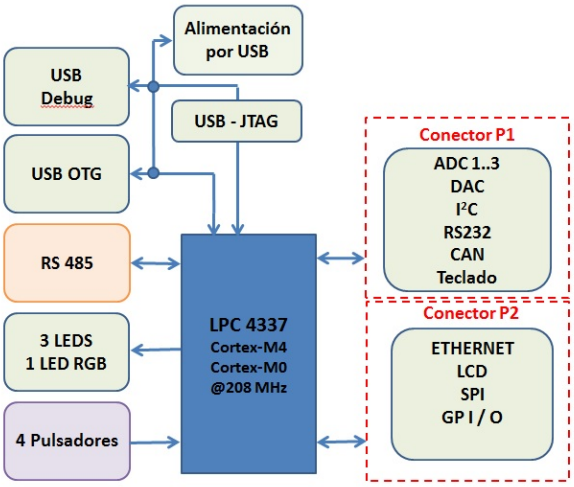

Fig. 1. Diagrama en bloques de EDU CIAA basado en LPC4337.

3) 1 puerto I2C.
4) 1 puerto asincrónico full duplex (para RS-232).
5) 1 puerto CAN.
6) 1 conexión para un teclado 3x4.

Sobre el puerto P1, se ubican los siguientes módulos:

1) 1 puerto Ethernet
2) 1 puerto SPI
3) 1 puerto para Display LCD con 4 bits de datos, Enable y RS.
4) pines genéricos de I/0.

II. Instalacion de Software

A. Conceptos previos

El desarrollo de codigos para sistemas embebidos tiene ciertas semejanzas con el desarrollo de aplicaciones en las PC, en nuestro caso particular se utiliza un compilador llamado *GCC* con soporte para la compilación de proyectos sobre los procesadores basados en la arquitectura ARM, en este caso particular, el compilador utilizado para el procesador de la EDU CIAA (el cual es el LPC4337) se lo denomina *arm-none-eabi-gcc*.

Para la ejecución de la depuración de algun programa previamente compilado, el hardware de la CIAA viene provisto con el chip *FT2232H*, que se encarga de hacer un puente entre la interfase JTAG del microcontrolador, y el USB que conecta

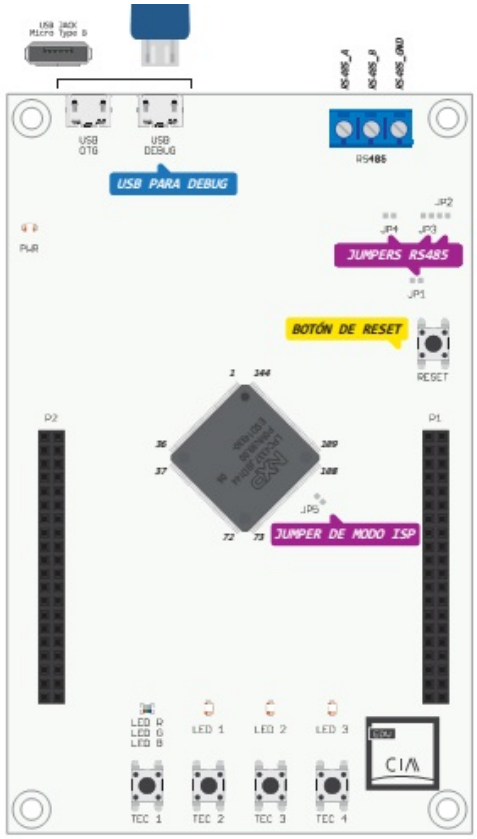

Fig. 2. Imagen frontal de placa

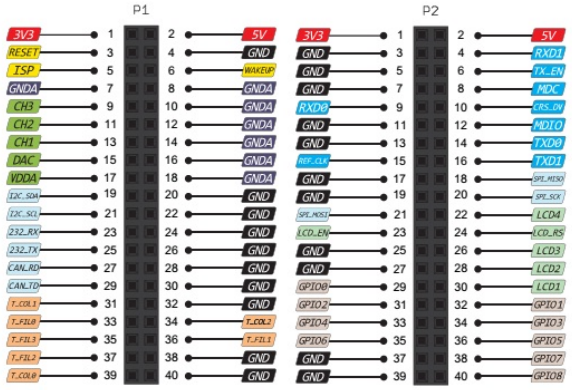

Fig. 3. Diagrama de distribucion de los pines sobre la EDU CIAA

a la PC en el puerto USB dedicado al debug. Mediante la herramienta de código abierto *OpenOCD (On Chip Debugger)* se controla el chip *FT2232H* por el USB y ademas todo lo referido al JTAG. Luego la herramienta de depuración *GDB* utilizado en el IDE-Eclipse que se instala, se comunica sobre el puerto 3333 (TCP) que el *Open OCD* tiene en escucha esperando la conexión.

Debe tenerse en cuenta que el chip *FT2232H* posee 2 canales de comunicación independientes (A y B), sin embargo, ambos

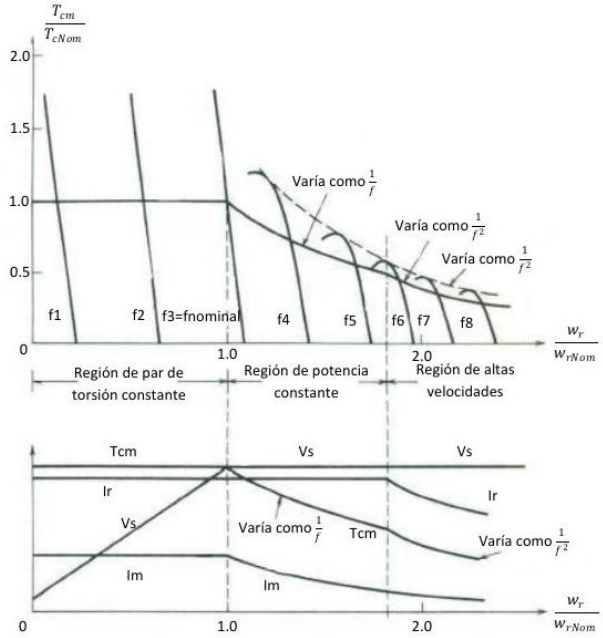

Fig. 4. Curvas características del motor de inducción [1].

salen por el mismo USB, de modo que la PC detecta 2 dispositivos distintos (en realidad es uno compuesto). Uno de ellos, se conecta al JTAG manejado por *OpenOCD* como fue mencionado, mientras que el otro se ve como un puerto virtual COM. Este último sirve principalmente para la depuración.

Dado que al fun

En este trabajo, el sistema de control de velocidad que se ha diseñado trabaja en la zona lineal (región de par de torción constante), es decir, se mantiene constante la relación entre tensión de alimentación y frecuencia de la señal trifásica de alimentación sin sobrepasar los valores nominales. Esto permite mantener el flujo magnético constante sin saturar el núcleo magnético. Una justificación matemática se puede deducir de la ley de Faraday, mostrada en la ecuación (1).

$$v\left(t\right) = -N\frac{d\phi}{dt} \tag{1}$$

Si se aplica un voltaje $v\left(t\right) = V_m \sin(\omega t)$ al núcleo, el flujo ϕ resultante es:

$$\phi(t) = \frac{1}{N_p}\int v(t)dt = \frac{1}{N_p}\int V_M \sin\left(\omega t\right)dt \tag{2}$$

$$\phi(t) = -\frac{V_M}{\omega N_p} cos(\omega t) \tag{3}$$

La frecuencia eléctrica aparece en el denominador de la ecuación 3. Entonces, si se mantiene constante la relación entre la tensión y la frecuencia eléctrica aplicada al estator, el flujo en el núcleo del motor se mantiene constante, al igual que la corriente de magnetización, manteniendo el par motor constante. Esto permite mantener el torque aproximadamente constante entre $0.2 * V_{nominal}$ y $V_{nominal}$. Si se mantiene

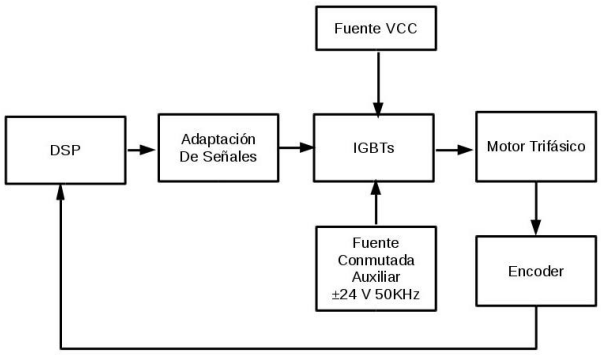

Fig. 5. Diagrama esquemático del sistema.

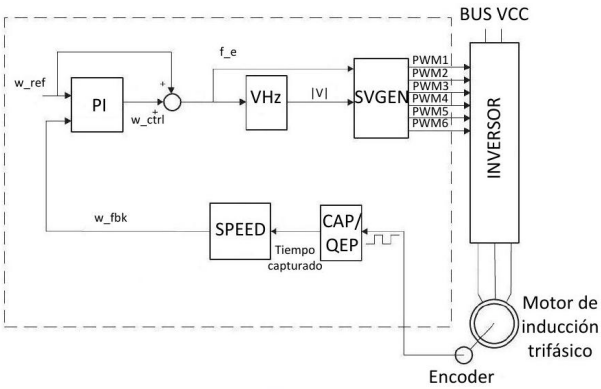

Fig. 6. Diagrama de bloques de la implementación del driver para control V/f de un motor de inducción trifásico.

constante la tensión y se aumenta la frecuencia de alimentación por encima del valor nominal, el torque comienza a disminuir como se observa en la Fig. 4.

III. Configuración del proyecto

El proyecto está formado por los distintos módulos que se pueden observar en la Fig. 5, los cuales son: DSP, Inversor, Motor Trifásico, Encoder, Bus VCC, Fuente Conmutada Auxiliar y Adaptación de Señales.

A. Implementaión del Control V/f en el DSP

Para la implementación del control V/f se han utilizado las librerías dadas por Texas Instrument a través de su Control Suite [4]. Estas librerías permiten un rápido prototipado y puesta a punto del software que se va a cargar en el DSP. En este trabajo en particular se ha utilizado como guía la nota de aplicación [5] que indica los pasos a seguir para implementar este tipo de control. En la Fig. 6 se puede observar un diagrama de bloques con las librerías usadas perteneciente a esta guía. Para la compilación del código se ha usado Code Compose Studio V6 [7].

Los bloques que están encerrados por la linea punteada de la Fig. 6, corresponden a los bloques software con los cuales se ha programado el DSP. Lo que está fuera de la línea punteada corresponde al inversor, al motor trifásico y al encoder utilizados.

- **Bloque PI:** este bloque define un controlador PI y es el encargado de generar una señal que permita ajustar la velocidad de salida del motor con la velocidad seteada. Esta velocidad se calcula internamente en por unidad (pu).
- **Bloque V/Hz:** Este bloque calcula el módulo de la tensión a aplicar al motor en función del valor de velocidad que recibe.
- **Bloque SVGEN:** Este bloque calcula los coeficientes que son utilizados para generar las señales PWM que generarán las tensiones del estator utilizando la técnica de SVPWM (Space Vector Pulse Width Modulation[1][2]).
- **Bloque SPEED:** Este bloque calcula la velocidad usando como dato el tiempo capturado por el bloque **CAP/QEP**.
- **CAP/QEP:** Este bloque está conectado directamente con un módulo hardware del DSP que permite capturar las señales del encoder en forma independiente del procesador.

Este sistema trabaja con interrupción de un timer que permite ajustar la frecuencia del PWM, la cual es de 20 KHz. El flujo de ejecución del software es como sigue:

Dentro del main

1) Se inicializan los módulos software.
2) Se inicializan las bases de tiempo.
3) Se habilita la base de tiempo CNT_zero de interrupción del EPWM1 y las interrupciones globales.
4) Se inicializan otros sistemas y parámetros de otros módulos en el caso de que sea necesario.
5) Se ingresa en un bucle infinito para esperar las interrupciones del EPWM1.

Dentro de la interrupción EPWM1

1) Se guarda el contexto y se limpian las banderas de interrupción.
2) Se ejecuta el módulo PI.
3) Se ejecuta el módulo V/Hz.
4) Se ejecuta el módulo SVGEN.
5) Se ejecuta el módulo CAP/QEP.
6) Se ejecuta el módulo SPEED.
7) Se cargan los valores calculados al módulo PWM.
8) Se restaura el contexto.
9) Se retorna al bucle infinito.

Los bloques mostrados en la Fig. 6 se describe a continuación.

B. Inversor

Un inversor trifásico es un circuito que convierte corriente contínua (CC) en corriente alterna (CA) de tres fases. Esto se hace mediante la conmutación de dispositivos semiconductores que lo componen, los cuales actúan como llaves. Actualmente son muchas las topologías que se utilizan. En este trabajo se ha utilizado la topología VSI (*Voltage Source Inverter*)[2], mostrada en la Fig. 7.

La naturaleza de este circuito impone dos reglas que garantizan su correcto funcionamiento, a saber:

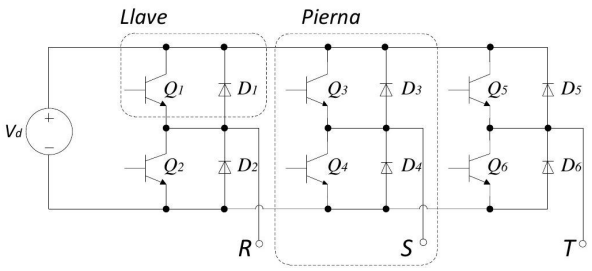

Fig. 7. Diagrama esquemático de un inversor trifásico.

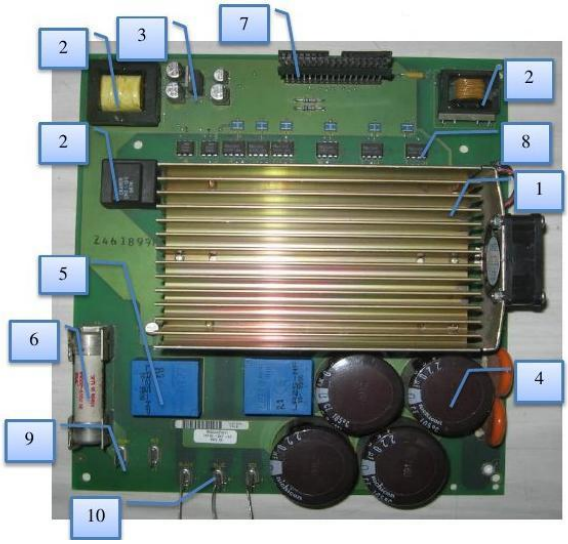

Fig. 8. Placa inversora.

- Las llaves de una misma pierna no pueden estar cerradas al mismo tiempo, ya que esto produciría un cortocircuito en el bus de CC.
- Las llaves de una misma pierna no pueden estar abiertas al mismo tiempo, ya que esto dejaría que la tensión en el punto medio de la pierna dependa del sentido de circulación de la corriente por la carga.

Las reglas propias de estos sistemas ya están contempladas en las librerías que propone Texas Instrument. En este caso es el bloque **PWM** de la librería ***"f2833xpwm.h"*** el que se encarga de que en nigún momento estén cerradas o abiertas al mismo tiempo dos llaves de una misma pierna.

En este trabajo el inversor que se usó se muestra en la Fig. 8, el cual posee un módulo PM25RSK120 marca Mitsubishi, Fig. 9.

En la Fig. 8 se puede ver el disipador (1), debajo del cual se encuentra el módulo PM25RSK120, sus respectivas fuentes de alimentación (2), una fuente de 5 V (3), filtros del bus de CC (4), circuitos de monitoreo (5), protección (6), conectores para el control de llaves (7), optoacopladores (8), conectores para el bus de CC (9) y para las líneas de salida de la señal trifásica (10). El módulo PM25RSK120, es un módulo inteligente de potencia compuesto por siete llaves IGBT con sus respectivos circuitos de activación y protecciones.

Fig. 9. Módulo: PM25RSK120, Marca: Mitsubishi.

Fig. 10. Motor trifásico y encoder acoplado a su eje.

C. Bus de VCC

Para el bus VCC se ha utilizado una fuente doble puesta en serie, la cual entrega una tensión de 60 V y 3 A máximos. Esto permite realizar las pruebas del proyecto sin correr riesgos.

Como el motor utilizado funciona con una tensión trifásica de 380 V y 50 Hz, se tuvo que calcular la nueva frecuencia máxima de trabajo, suponiendo que el motor va a trabajar sin carga.

En el caso de este trabajo y debido a que el motor trabaja en vacío la frecuencia máxima de alimentación se fijó a 20 Hz.

D. Fuente Auxiliar

La placa inversora necesita para su funcionamiento una fuente que entregue una tensión de onda cuadrada de $\pm 24V$ a 50 KHz.

E. Motor trifásico

Se utilizó un motor trifásico conectado en estrella modelo TE1BFOXO marca WEG tipo W22 [8], como se observa en la Fig. 10.

F. Encoder

Para medir las RPM desarrolladas por el motor trifásico se ha utilizado un encoder óptico incremental ENB-200-3-1. Este encoder tiene como salida las señales A, B y Z en totem pole, con una resolución de 200 pulsos por revolución.

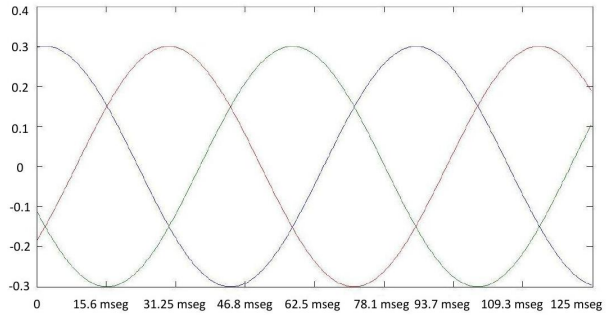

Fig. 11. Señales Ta-Tb, Tb-Tc y Tc-Ta al 30% de la tensión nominal

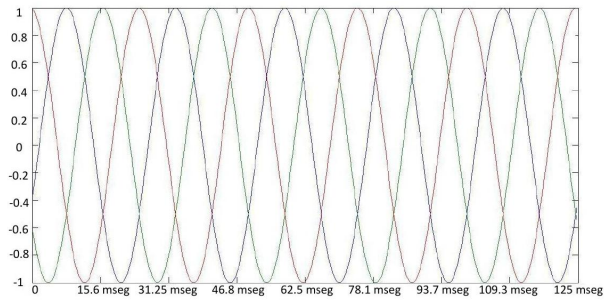

Fig. 12. Señales Ta-Tb, Tb-Tc y Tc-Ta a la tensión nominal

IV. Pruebas

Para las pruebas del sistema se usaron las indicaciones de la nota de aplicaciones [5]. La primera de ella fue probar el código a lazo abierto para ver si las señales PWM correspondían con las esperadas. Se graficó la diferencia entre las señales Ta, Tb y Tc que entran al bloque pwm. El resultado que se obtuvo se muestra en la Fig. 11

En la Fig. 11 se puede observar que para una tensión $0.3 * V_{nom}$ la frecuencia es un 30% de la frecuencia nominal. En el caso de la Fig. 12 la tensión aplicada es la nominal, por lo tanto la frecuencia es la nominal. Con esto queda demostrado que el sistema cumple con los requerimientos.

A. Ajuste del controlador PI

El módulo PI responde a la ecuación 4

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau \tag{4}$$

en donde

- u(t) es la salida del PID
- e(t) es el error entre la referencia y la retroalimentación
- K_p es la ganancia proporcional del PID
- K_i es la ganancia integral del PID

El módulo PI se ha configurado siguiendo los siguientes pasos:

Paso 1.
Establecer la ganancia integral $K_i = 0$ y la ganancia proporcional $K_p = 1$.

Paso 2.

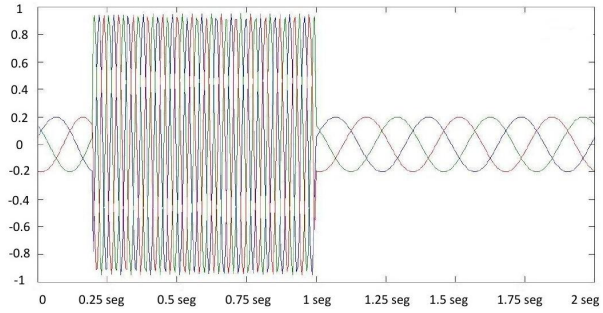

Fig. 13. Señales Ta-Tb, Tb-Tc y Tc-Ta dada una entrada escalón

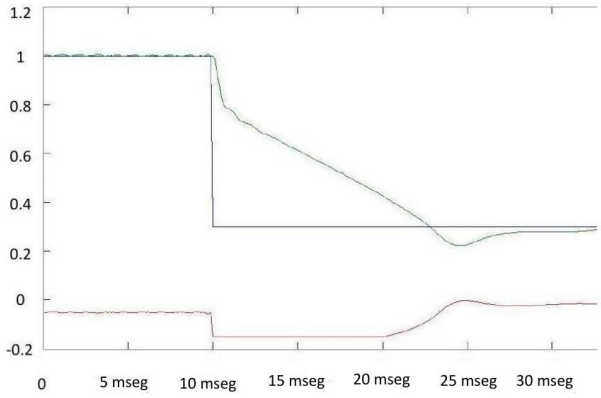

Fig. 14. Velocidad, salida del controlador y señal escalón cuando el escalón es descendente

Mientras se le inyecta una señal escalón a la entrada, ajustar poco a poco la variable de ganancia proporcional K_p, hasta alcanzar alcanzar el tiempo de subida y overshoot óptimos.

Paso 3.
Si es necesario, aumentar gradualmente la ganancia integral K_i para optimizar el retorno de la salida de estado estacionario a nominal. El controlador será muy sensible a este término y se puede volver inestable así que se debe asegurar de comenzar con un número muy pequeño. La ganancia integral se traducirá en un aumento de sobreimpulso y de oscilación, por lo que puede ser necesario disminuir ligeramente el término K_p de nuevo para encontrar el mejor equilibrio.

B. Pruebas finales

Para la prueba del sistema se desarrolló una señal tipo escalón que variaba entre 0.3 y 1. Esta señal tiene un período de 5 segundos. Los resultados se muestran en las Fig. 13, 14 y 15.

V. Conclusiones

En este trabajo se ha logrado realizar un contro V/f usando las librerías propias de Texas Instrument. Se logró modificar las librerías y acondicionarlas para el **DSP TMS320F28335**. Este DSP puede trabajar directamente con datos tipo float, lo cual permitió modificar las librerías y pasarlas a float. En las pruebas realizadas también se aumentó la carga aplicada al eje del motor y se comprobó que el sistema buscaba ajustar la

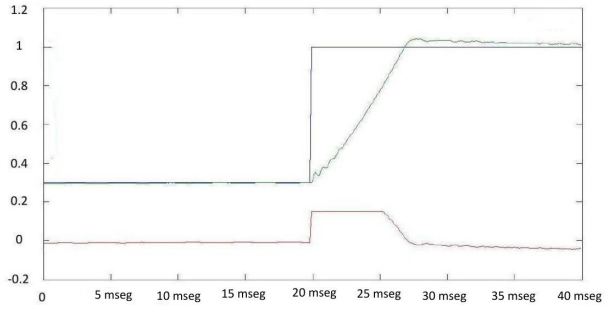

Fig. 15. Velocidad, salida del controlador PI y señal escalón cuando el escalón es ascendente

velocidad de acuerdo con la velocidad de referencia. También se frenó el motor hasta pararlo, y este volvía a su estado de régimen a la velocidad de referencia.

REFERENCES

[1] Ned Mohan, Tore M. Undeland, William P Robbins. *Electrónica de Potencia. Convertidores, aplicaciones y diseño.* McGraw Hill. Tercera Edición

[2] Muhammad Rashid. *Electróncia de Potencia. Circuitos, dispositivos y aplicaciones.* Pearson Prentice Hall. Tercera Edición.

[3] Alfonso Álzate, Duberney Murillo Yarce, Marcela González Valencia. ”Control de velocidad mediante relación voltaje-frecuencia”. ISSN 0122-1701

[4] http://www.ti.com/tool/controlsuite

[5] Scalar (V/f) Control of 3-Phase Induction Motors. Application Report. http://www.ti.com/lit/an/sprabq8/sprabq8.pdf

[6] http://www.ti.com/product/TMS320F28335

[7] Code Composer Studio http://www.ti.com/tool/ccstudio

[8] http://www.weg.net/w22/index-es.php?market=am-latina