SOFTWARE AEROESPACIAL PARA EL PROCESO DE PRUEBA ELÉCTRICA EN TIEMPO REAL

Verónica Quintero Rosas

Tecnológico Nacional de México/ITMexicali

veronicaquintero@itmexicali.edu.mx

Francisco Ibañez Salas

Tecnológico Nacional de México/ITMexicali

ibanezsalasf@itmexicali.edu.mx

Gilberto García Gómez

Tecnológico Nacional de México/ITMexicali

GilGG@itmexicali.edu.mx

Mario Camarillo Ramos

Tecnológico Nacional de México/ITMexicali

camarillomario@itmexicali.edu.mx

Heber Samuel Hernández Tabares

Tecnológico Nacional de México/ITMexicali

hsamuel@itmexicali.edu.mx

Resumen

El objetivo de este trabajo es analizar las condiciones de combustible y oxígeno en un sistema aeronáutico simulado, utilizando software aeroespacial en relación a prueba eléctrica que se desarrolla en una *test-bed*, con funciones de modelado, diseño y análisis. Esta investigación se basa en la certificación DO-178B para evidenciar el cumplimiento de requisitos de seguridad para los sistemas que resguardan todo lo referente a combustible y oxígeno en una aeronave. Esta investigación simula mediante la tarjeta HC11 los planes de seguridad de software respecto a prueba eléctrica, definiendo tareas secuenciales analizando requisitos,

análisis de nivel de código, análisis de pruebas y análisis de cambios de procesos respecto a acciones en la aeronave. Cualquier software que ordene, controle y monitoree las funciones críticas de seguridad en un módulo aviónico debe estar certificado bajo la norma DO-178B por lo cual, se utilizaron parámetros definidos en el manual de la Administración Federal de Aviación de los Estados Unidos. La prueba eléctrica está basada en el proceso de recertificación de información, respecto a sensores dedicados, instrumentos de control y análisis de base de datos. Para ello se diseñó código en MISRA C, para la interfaz de control correlacionando las acciones con los programas de simulación hechos en la tarjeta HC11. Esto es para definir la integración de la prueba eléctrica de cada instrucción lo cual hace una redundancia de operación del software, lo que permite mayor seguridad de acción en un caso de emergencia en la aeronave.

Palabra(s) Clave: Análisis de código, Prueba eléctrica, Software aeroespacial.

Abstract

The objective of this work is to analyze the fuel and oxygen conditions in a simulated aeronautical system, using aerospace software in relation to electrical test that is developed in a test-bed, with modeling, design and analysis functions. This research is based on the DO-178B certification to demonstrate the fulfillment of safety requirements for systems that protect everything related to fuel and oxygen in an aircraft. This research simulates through the HC11 the software security plans regarding electrical test, defining sequential tasks analyzing requirements, code level analysis, analysis of tests and analysis of changes of processes with respect to actions in the aircraft. Any software that orders, monitors and monitors critical safety functions in an avionic module must be certified under DO-178B, using parameters defined in the US Federal Aviation Administration manual. The electrical test is based on the process of recertification of information, regarding dedicated sensors, control instruments and database analysis.For this, the code was designed in MISRA C, for the control interface correlating the actions with the simulation programs made in the HC11 card. This define the integration of the electrical test of each instruction which makes a

redundancy of operation of the software, which allows greater security of action in an emergency case in the aircraft.

Keywords: *Aerospace software, Code analysis, Electrical test.*

1. Introducción

El intercambio de información entre aplicaciones en un sistema aeronáutico [Kritzinge, 2009], utiliza una interfaz de software que sincroniza, enlaza y da seguridad a una aplicación para interactuar con otras aplicaciones incluyendo hardware y diferentes tipos de sistemas operativos; en esta investigación se utilizan por separados las aplicaciones para emergencias en combustible y otra para oxígeno. Sin embargo, en el momento de ocurrir dicha emergencia estas dos aplicaciones deberán sincronizarse para jerarquizar los recursos y minimizar el fallo en la aeronave utilizando un sistema de verificación de prueba eléctrica para asegurar las tramas de envió de señales (oxígeno - combustible) a cada uno de los sensores, mecanismos que se desee comunicar y/o activar.

El intercambio de información entre aplicaciones ayuda en la complejidad de la comunicación en sistemas distribuidos y heterogéneos. Aún se encuentra en desarrollo, ya que los proveedores en la línea de software espacial han visualizado que el desarrollo de código abierto es muy eficaz en la portabilidad y la lógica del intercambio de información entre las aplicaciones. Las investigaciones financiadas por el Departamento de Defensa de Estados Unidos [FAA, 2012], han detectado la importancia de la lógica de intercambio de información con los sistemas en tiempo real para simplificar el desarrollo de aplicaciones; ya que a medida que las redes de sistemas aeroespaciales crecen, la diversidad de la funcionalidad integrada también aumenta.

Este intercambio de información fue analizado en una *test-bed* donde se tiene un ambiente de desarrollo controlado para verificar si las señales del software eran correlacionadas respecto a las señales eléctricas. Los módulos de combustible y oxígeno trabajan de manera independiente pero relacionados, jerarquizadas y estructuradas en un mismo sistema. En caso de una emergencia el sistema debe ser capaz de reorganizar funciones, priorizar y administrar caída de máscaras de

oxígeno, cantidad de flujo de aire por presión, eliminar flujo de aire en asientos donde no existan personas, anular zonas de electrificación innecesarias (focos de lecturas en los asientos etc.), verificar el correcto funcionamiento de los instrumentos en cabina, turbinas, etc (figura 1).

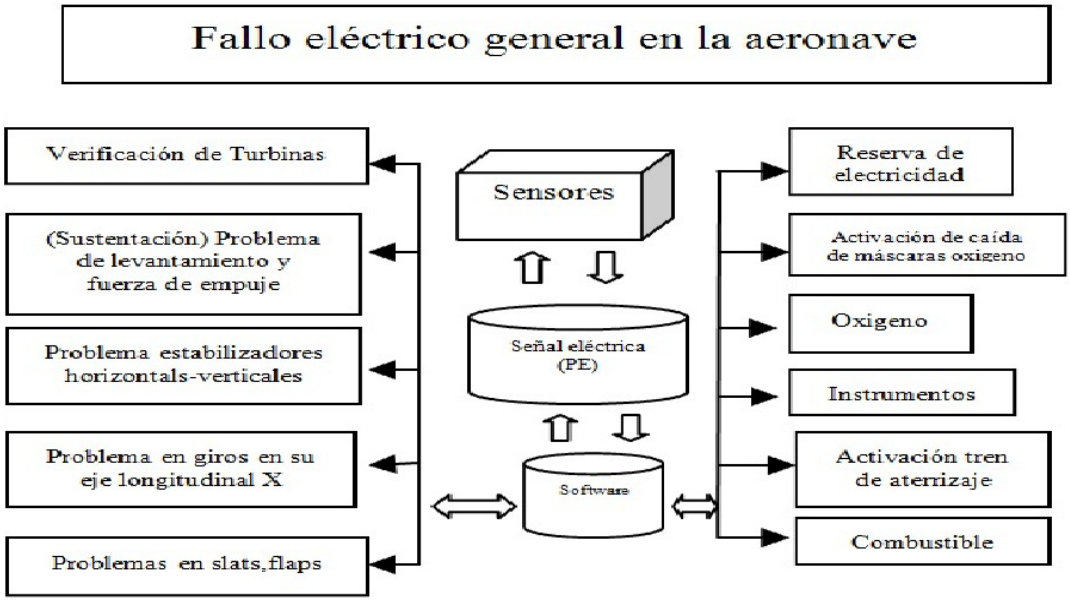

Figura 1 Correlación de la señal eléctrica respecto a software en una emergencia.

Todo lo referente a prueba eléctrica [Lindeburg, 2012] requiere de seguridad y fiabilidad ya que correlacionar software implica señales de entrada y salida que deben ser atendidas en un tiempo determinado y de manera correcta o puntual.

La certificación DO-178B define en sus diferentes postulados la forma correcta de enviar y recibir parámetros de comunicación respecto a software haciendo una redundancia o recertificando cada parámetro antes, durante y después de la acción que se definió. Esta recertificación se hizo mediante credenciales de autentificación de variables, funciones, punteros, parámetros generales y métodos específicos de código.

En el área aeroespacial, donde se desarrolló esta investigación se tienen una serie de niveles de pruebas experimentales en un desarrollo de software distinto, y cada nivel de desarrollo tiene un nivel de prueba correspondiente; donde el código y la prueba están en un estado de ciclo repetido. Pero para los problemas de

rendimiento u optimización de software respecto a prueba eléctrica, estos niveles se unifican para hacer una prueba maestra donde se involucran señales como la utilización de piloto automático, señales de navegación, comunicación, respecto a niveles de turbosina (C) y oxígeno (O) para racionar y prevenir fallos que puedan afectar (O, C) debido a otros fallos en la aeronave. Es muy importante recordar que la seguridad en software está definida por la arquitectura de redundancia en hardware y software. Cuando existe una emergencia y una aplicación derivada de esta, trata de tomar la señal para ser ejecutada, esta se jerarquiza en un intercambio de aplicaciones para dar prioridad a los protocolos establecidos en aviación dependiendo del tipo de fallo de la aeronave. Este intercambio entre aplicaciones ya está regulado mediante la Comisión Radiotécnica para la Aeronáutica [RTCA, 2015]; la cual realiza recomendaciones y son base en la FAA (Administración Federal de Aviación de los Estados Unidos de América). Dichas recomendaciones certifican el software para aeronaves. A pesar de la utilización digital, actualmente se conserva en los sistemas aeronáuticos una copia de seguridad análoga en los sistemas críticos, por lo que la redundancia tanto en hardware como en software sigue permaneciendo.

Cada línea o ruta redundante se denomina canal. Estos canales incluyen sensores, cableado y convertidores tanto análogo a digital, como digital a analógico. Se utilizan algoritmos de gestión para determinar si una entrada o respuesta es redundante, o si ya está siendo procesada. De tal forma que existen múltiples canales procesando información simultáneamente, y estos deben tener una sincronización para dar respuesta. No se debe olvidar que toda reconfiguración de software debe estar presente en casos de fallo ya que esto preserva, aísla y da seguridad. Los sistemas aeronáuticos integran miles de componentes y conexiones, lo cual aumenta la complejidad de configuración y esto a su vez aumenta la posibilidad de fallos. Otra técnica para resguardar el intercambio de información entre aplicaciones o sistema en general es mediante el guardado periódico del estado de la aplicación; haciendo recuperaciones de la información constantemente.

2. Metodología

Pruebas de división de software vs prueba eléctrica

Está basado en pruebas de integración, donde se hacen pruebas aisladas primeramente de software; al cual se analiza y verifica: estructura, proceso, gramática, revisión de código, métricas de calidad de código etc. En esta investigación se utilizó un análisis de software mediante MISRA C (The Motor Industry Software Reliability Association) [Ashford *et a*l, 2007] el cual denomina un conjunto de recomendaciones para el desarrollo de software; dando seguridad y portabilidad al código. La lista de verificación que MISRA C da a la industria aeroespacial, es para estandarizar las pruebas de software; respecto a las pruebas eléctricas que puedan ser examinadas. Actualmente MISRA C es utilizado en la industria automotriz y médica, ya que es una herramienta de análisis de códigos complejos y muy grandes. MISRA C puede analizar desde punteros no válidos, variables no inicializadas hasta problemas en temporizadores y almacenamiento en memoria. Se utilizan pruebas unitarias, las cuales son métodos que prueban una unidad de código o requerimiento. Básicamente las pruebas unitarias aseguran el correcto funcionamiento de las interfaces, o flujo de datos entre componentes, además de ser repetibles y predecibles. Las pruebas unitarias [Marcano, 2008] en el sector aeroespacial generalmente se utilizan para casos de prueba de diseño o desarrollar scripts de prueba de unidades automatizadas. También se generan pruebas de integración las cuales comprueban unidades o componentes de software, módulos, configuraciones etc. Estas pruebas, verifican si el software y el hardware controlado por el software pueden interactuar correctamente; y es en esta fase donde se mide el rendimiento funcional. Estas pruebas se desarrollan de la siguiente manera:

- Físicamente: Prueba de simulación física, se establece una conexión de todo el sistema; es decir software y hardware junto con los dispositivos físicos como sensores, transductores especiales etc. Se hace mediante un control cerrado de retroalimentación, donde se puede garantizar la autencidad de que las pruebas en tiempo real son más reales.

- Semifísico: Integra algunos módulos de hardware y el resto son simulados; es económico y seguro, además de responder con un factor tolerante a los resultados de pruebas.
- Simulación digital: Sistema de software que simula el hardware del sistema, su entorno externo; a través de la combinación de control, interrupciones, reloj, emulador, etc. Es débil respecto a la interfaz ya que la autenticidad en tiempo real es limitada.

Definir la prueba eléctrica para cada módulo o tarea de software es muy importante, debido a que será el diagnóstico del sistema o síntomas que puedan detectarse de algún instrumento, software, sensor, sistema operativo, aplicación etc. Ya que, a pesar de tener autonomía al hacer pruebas aisladas, cada módulo de software y prueba eléctrica se valida así mismo, al interconectarse y encontrar un error se auto validará para posteriormente volver a analizar el error y verificar que el error no lo genera el propio módulo de prueba eléctrica. En una aplicación embebida ubicua [Molloy, 2011] en el cual se comparte hardware, software, sensores, electrónica para comunicar, diagnosticar, almacenar e interconectar multifactorialmente todo el proceso de manera sistemática se trabaja de manera modular e independiente todo el sistema aéreo; pero definido en una arquitectura que analiza todo el sistema como un ecosistema complejo. También es importante no perder de vista las normas para pruebas como lo son las normas Federales de Procesamiento de Información [NIST, 2015], normas desarrolladas por el gobierno federal de los Estados Unidos, para el uso de sistemas informáticos de agencias gubernamentales no militares y contratistas. Estas normas establecen requisitos para garantizar la seguridad informática.

Las consideraciones de seguridad de los sistemas de software aeroespacial [Anderson, 2010] radican en la compatibilidad de: hardware-software, software-software, aeronave-software. Dando lugar a técnicas para detección de daños, efectos de carga, errores de software etc. Actualmente existen métodos de encriptado [Rierson, 2013] que permiten aumentar la seguridad en la comunicación del software y de hardware. Además de la norma O-178B [Klamert,

2014] que certifica software en sistemas y equipos aeroespaciales. Básicamente es una guía de la seguridad del software y es un estándar para el desarrollo de sistemas de software de aviónica. De esta se deriva la DO-178B [Hilderman, 2013] que certifica y evidencia el cumplimiento de requisitos de seguridad. Asigna planes de seguridad de software y define tareas de análisis de seguridad de software en pasos secuenciales (análisis de requisitos, análisis de diseño de nivel superior, análisis de diseño detallado, análisis de nivel de código, análisis de pruebas y análisis de cambios). Cualquier software que ordene, controle y monitoree las funciones críticas de seguridad debe estar certificado bajo esta norma.

También se debe tomar en consideración la norma IDAL que determina el nivel de garantía del software, es decir determina a partir de procesos de evaluación y análisis de peligros en caso de fallo en el sistema. Todas estas normas están en la biblia de normas de la FAA aplica DO-178B como guía para determinar si el software funcionará confiablemente en un entorno aeroespacial. En los Estados Unidos, el DO-178B se establece en el título 14: Aeronáutica y Espacio del Código de Reglamentos Federales.

Desarrollo

El modelo propuesto para la verificación de prueba eléctrica respecto a software de los módulos de oxígeno y combustible en una aeronave, está dado por placas con configuración ya especificados por la empresa Honeywell aeroespace, donde se hicieron los experimentos; en este caso de estudio después de auto validar cada segmento de software y hardware por sí mismo el siguiente paso es detectar pistas abiertas, cortos, componentes faltantes, verificar valores de componentes como resistencias, capacitores, bobinas, detectar pines sin soldar etc.

Este proceso de prueba eléctrica se hace mediante "fixtures" [Lindeburg, 2012] el cual es un mecanismo de interfaz entre éste y el equipo de prueba. Contiene pines de interfaz que hacen contacto con los puntos de prueba de la placa, para su medición (sistema de control de oxígeno, sistema de control de combustible). Este a su vez será energizado con una fuente programable con voltaje de corriente directa, durante las pruebas y mediante las puntas de prueba se verifica

continuidad entre los pines de la interfaz del fixture y los pines de las tarjetas de pines; incluyendo amplificadores y relevadores, además de multiplexear la señal generando un estímulo con una impedancia específica. La respuesta es tomada por un receptor y medida lógicamente, como lo muestra la figura 2.

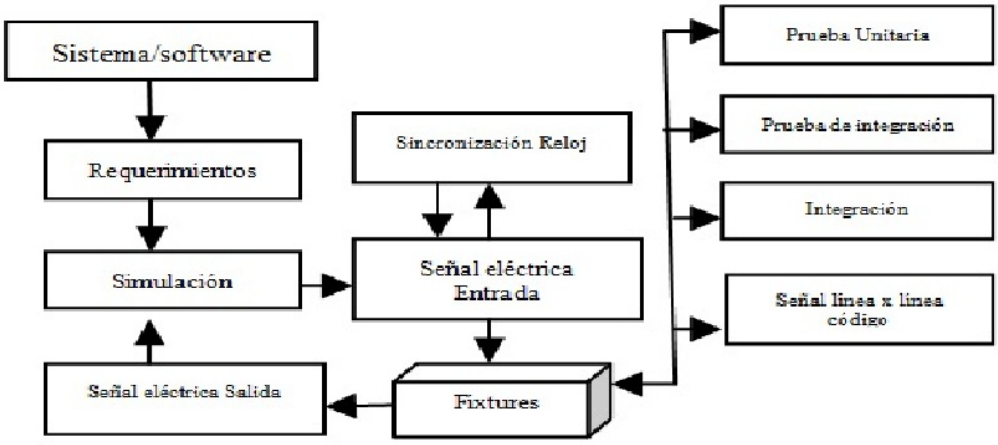

Figura 2 Modelo de verificación de prueba.

La test bed utilizada fue mediante la tarjeta Motorola HC11 [Fox, 2012] la cual administra y aísla módulos desde un bajo nivel de código, el cual permite configurar la activación de la señal de alerta generalizada de la aeronave (W), sincronizar las aplicaciones (WPORTD) mediante estampilla de tiempo de la test bed, señal de caída de mascarillas de oxígeno (SPE), verificación de sensor en asientos ocupados (RST), niveles de combustible (MSTR), apagado de luces, excepto línea de pasillo central (DS). Sin olvidar el recalculo de la trayectoria para aterrizaje más cercano respecto al combustible (Tr).

$$\begin{bmatrix} RST\ 1 \\ RST\ 2 \\ RST\ x \end{bmatrix} = \sum WPORTD\ (SPE)^{W}\ \ DS_{RST} \Big/ \sum Tr\ (MSTR) \leq W\ \geq vie \qquad (1)$$

Aplicando el algoritmo (1) experimental, los resultados de sobrevivir a un fallo eléctrico generalizado en la aeronave (vie) es de alrededor del 25% ya que los resultados que arroja la prueba eléctrica es muy bajo; como lo muestra la figura 5, debido a que el sistema colapsa a la falta de señal mediante falsos verdaderos y si este error ocurre cuando la aeronave se encuentra en los 10,000 metros a una

velocidad de 800 kilómetros/hora el algoritmo de alerta empezará el mecanismo de emergencia pero a pesar que todas las mascarillas de oxígeno caen y se respeta los asientos vacíos y se reduce el gasto eléctrico, se verifica el nivel de combustible este no está preparado para los cortos que se puedan producir en la aeronave por el propio fallo generalizado (figura 3).

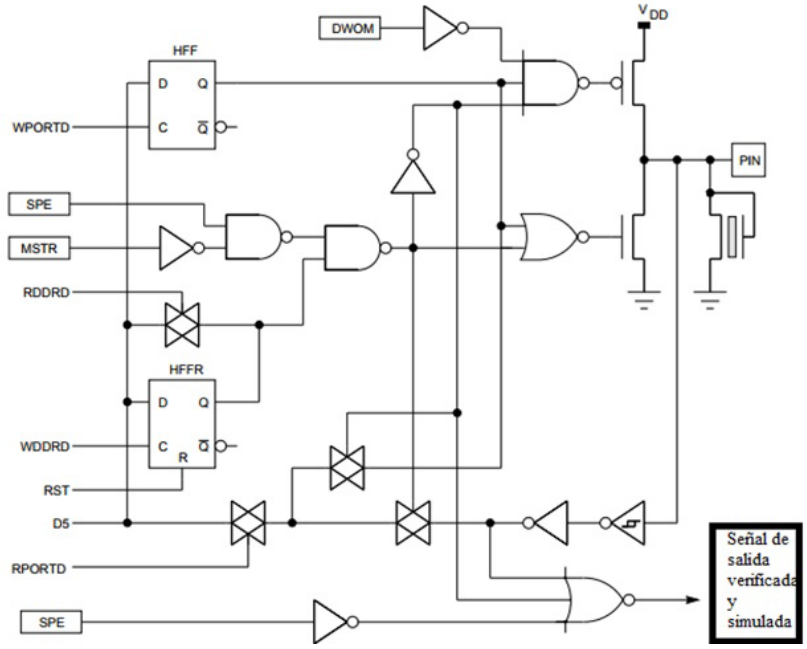

Figura 3 Circuito de prueba para señales de activación de alerta en oxígeno y combustible en *test-bed* HC11 con fallo eléctrico generalizado.

La codificación como se muestra en la figura 4 se realizó simulando en la tarjeta HC11 los módulos de formas aisladas, (se aísla para asegurar si un módulo falla este no afecte el sistema) se utilizó tarjetas Galileo Arduino, los cuales contenían las aplicaciones de control de oxígeno y combustible pero comunicados y verificados con MISRA C.

3. Resultados

Esta investigación está basada en el análisis experimental de los módulos de oxígeno y combustible en una aeronave mediante pruebas eléctricas durante la comunicación en una señal de emergencia. La simulación arroja resultados moderados de casi el 80% de buena comunicación, caída de mascarillas de

oxígeno del 100%, análisis de combustible correcto. Pero, al aplicar un fallo eléctrico generalizado a la aeronave estas pruebas caen hasta en un 25% (figura 5) de eficiencia lo que da como resultado catastrófico.

```
CLR IC1DUN ; Señal de inicio de prueba
BCLR TFLG1;  Señal de oxigeno "O"
BCLR TFLG2;  Señal de combustible  "C"
FFDA-DB Entrada de reloj (sincronización)
BSET TMSK1;  Activación O
BSET TMSK2; Activación  C
FFDC-DD Sobre flujo (estampilla de tiempo)
PSHY PSHA
FFDE-DF Reloj final de sobre flujo
BSR REFORM;  Re-formateo de prueba última
LDX #SPI1; Envió de inicio de trama
BCLR PORTD;  Puerto de O
STAA SPDR; Escritura del registro
WAIT1 LDAA; Espera respuesta eléctrica
BPL WAIT1
CPY #DA1-1; No caída de máscaras en asientos vacíos
LSRA  #SPI2; Envió de señal test bed
```

```
BCLR PORTD
NXTBIT BSET PORTD; Bit de recolección
BITA 0; Prueba del bit enviado
BSET PORTD
ZBIT BCLR PORTD
BRA ENDBIT
ENDBIT BCLR PORTD verificación de señal mascaras de Oxigeno
BNE NXTBIT; Respuesta
STAA $00EE //carga dirección en test bed
LDD #IRQ
STD $00EF
STAB $1009
IRQ: LDAA #$20
```

Figura 4 Pseudocódigo del proceso de oxígeno para señales de software respecto a la prueba eléctrica en un fallo generalizado de la aeronave.

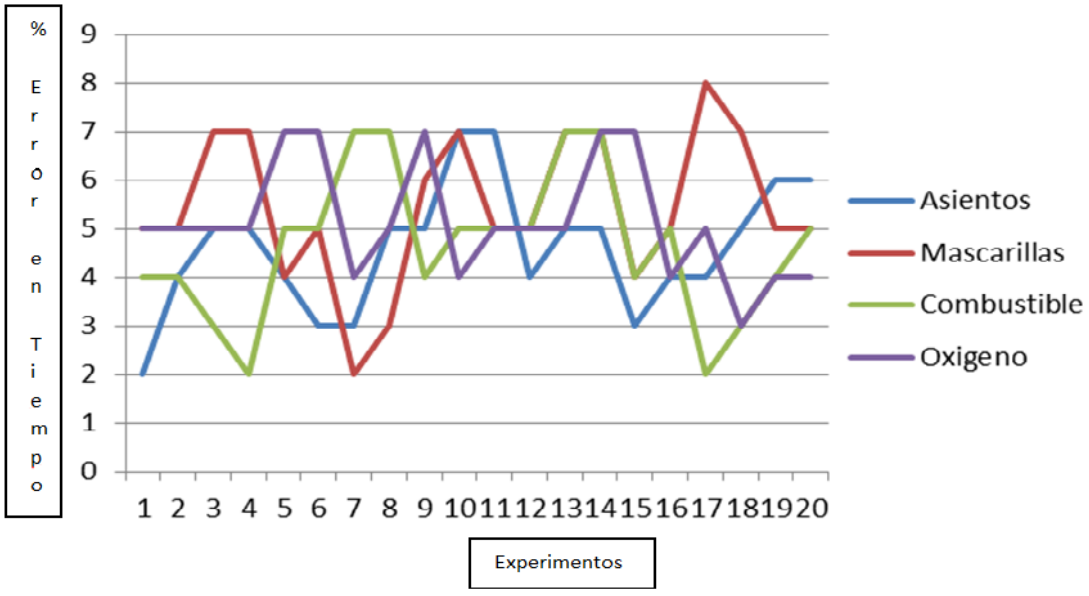

Figura 5 Resultados del experimento por orden de acción.

Debido a que un fallo eléctrico generalizado arroja falsos positivos lo cual interfiere en el correcto funcionamiento digital de nuestro sistema. Esto implica que a pesar del correcto funcionamiento de nuestro sistema la aeronave y sus pasajeros morirían. Es necesario hacer más análisis y pruebas por ejemplo adherir una fuente eléctrica de emergencia, externa a la fuente principal de la aeronave; de acuerdo con la *International Civil Aviation Organization* (ICAO), los fallos principales de accidentes aéreos son por fallos humanos, fallas mecánicas y fallos electrónicos.

Esta investigación necesita resolver dichos fallos eléctricos generalizados, actualmente se analiza utilizando osciloscopios por frecuencias de comportamientos en las diferentes señales, al producirse un fallo de esta índole. Actualmente el algoritmo está en una fase de entrenamiento simulando y resolviendo fallos en un ambiente controlado respecto al oxígeno y combustible, pero sabemos que no es suficiente ya que en la vida real una aeronave no viaja en un ambiente controlado teniendo posibles movimientos por bolsas de aire, turbulencias, radiación solar, problemas climáticos, fallos humanos etc.

4. Discusión

La utilización de sistemas basados en tiempo real con módulos aislados en su diseño de aplicación es una forma de garantizar el buen funcionamiento en un sistema aeroespacial, sin embargo en la experimentación de esta investigación se pudo constatar que un fallo eléctrico generalizado en una aeronave no garantiza el buen funcionamiento de las aplicaciones en la aeronave ya que al forzar mediante prueba eléctrica una señal se verifica su envió y llegada pero al haber un fallo eléctrico general esto no garantiza el correcto envío y recepción de señales,. En programación se corrompe el código debido a los falsos positivos eléctricos. Investigaciones anteriores aseguran el buen funcionamiento de aplicaciones, pero no se garantiza en un fallo eléctrico generalizado. Se pudo constatar que según estadísticas la principal causa de accidentes fatales en aeronaves son errores humanos, fallas mecánicas y un fallo eléctrico generalizado en la aeronave.

La forma como este experimento se apegó a las normas aeronáuticas fue mediante el respeto a los lineamientos de comunicación y protocolos de codificación de redundancia en la programación. Se tomó en consideración el cálculo multivariable en el módulo aviónico de interpretación de programas, según la normativa de la FAA. Además de cumplir con las metodologías de revisión de la certificación DO-178B. Es importante destacar que cada función, método y módulos de programación se estipulo según la revisión de revocación de licencias o permisos para los diseños aeronáuticos expuestos en la IO-A638 concernientes a el postulado 86 de la DO-178B. También se estableció una bitácora de vuelo en la simulación de acuerdo con la dirección general de aeronáutica civil de la Secretaria de Comunicaciones y Transportes de México (SCT), respetando los lineamientos de operación de cada acción en la aeronave.

Esta investigación funciona en un ambiente controlado donde no existirán errores humanos, fallas mecánicas y mucho menos un fallo eléctrico generalizado. Por lo cual estamos en una etapa de experimentación analógica, ya que mediante software los registros se corrompen y es necesaria la redundancia analógica de todas las aplicaciones y programas puestos en una aeronave. Falta hacer mucha más experimentación en esta área.

5. Bibliografía y Referencias

[1] Ashford, L. & Sanjit, A. S. (2007). Introduction to Embedded Systems - A Cyber-Physical Systems Approach. ISBN-13: 978-0557708574

[2] Anderson D. (2010). Aerospace Software Engineering: A Collection of Concepts. ISBN-13: 978-1563470059

[3] FAA. (2012). Administración Federal de Aviación: https://www.faa.gov/.

[4] Fox T. (2012). Programming and Customizing the HC11 Microcontroller. ISBN-13: 978-0071344067

[5] Hilderman V. (2013). Avionics Certification: A Complete Guide to DO-178 (Software), DO-254 (Hardware). ISBN-13: 978-1885544254 .

[6] Kritzinge Duan, (2009). Aircraft System Safety: Military and Civil Aeronautical Applications". ASIN: B00M3Z1I9C

[7] Klamert M. (2014). Services Liberalization in the EU and the WTO: Concepts, Standards and Regulatory Approaches. ISBN-13: 978-1107034594

[8] Lindeburg, M. (2012). FE Electrical and Computer Review Manual. ISBN-13: 978-1591264491.

[9] Marcano, A. (2008). Testing: Pruebas Automatizadas con Visual Estudio.Net. ISBN-13: 978-1530299690

[10] Molloy, D. (2011). Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux. ISBN-13: 978-1118935125

[11] National Institute of Standards and Technology [NIST] (2015). Federal Information Processing Standards Publications. ISBN-13: 978-1547148240

[12] Rierson L. (2013). Developing Safety-Critical Software: A Practical Guide for Aviation Software and DO-178C Compliance. ISBN-13: 978-1439813683

[13] RTCA. (2015). Comisión Radiotécnica para la Aeronáutica: https://www.rtca.org/.