Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática industrial 14 (2017) 205–216

www.elsevier.es/RIAI

Plataforma Experimental para el Estudio de la Vulnerabilidad Hardware en los Robots Móviles: el Bus I2C como Caso de Estudio.

F. Gomez-Bravo *, J. Medina García, R. Jiménez Naharro, J. A. Gómez Galán, M. Sánchez Raya

Departamento de Ingeniería Electrónica de Sistemas Informáticos y Automática, Grupo de Sistemas Electrónicos y Mecatrónica. Universidad de Huelva. Escuela Técnica Superior de Ingeniería, Carretera de Palos-La Rábida s/n. 21071 Huelva.

Resumen

Este artículo presenta una plataforma experimental para estudiar los efectos de las vulnerabilidades hardware de los robots móviles. La plataforma se ha diseñado de forma que los elementos hardware que intervienen en el proceso de navegación pueden ser monitorizados durante el funcionamiento del robot, y, si es el caso, su comportamiento puede ser alterado, simulando de esta forma una situación de fallo. El artículo muestra como caso particular de estudio la vulnerabilidad del Bus I2C cuando se producen anomalías en la señal de reloj. Se incluyen un conjunto de resultados experimentales que confirman el interés de las vulnerabilidades estudiadas y la aplicabilidad de la plataforma desarrollada.

Palabras Clave:
Robots móviles, vulnerabilidad hardware, FPGA, hardware configurable.

1. Introducción

En la última década, el número de aplicaciones robóticas utilizadas para solucionar problemas cotidianos o vinculados a tareas de seguridad ha crecido considerablemente. Así, está fuera de toda duda la utilidad de los robots en multitud de campos de aplicación: desde los relacionados con la vigilancia y el rescate de personas (Marques *et al*., 2007; García-Cerezo *et al*., 2007; Chen *et al*., 2012) hasta su utilización en el ámbito del hogar para tareas asistenciales o de limpieza (Jardón *et al*., 2008; Park *et al*., 2013; Kachouie *et al*., 2014). En este sentido, no hay que olvidar la ya contrastada eficiencia de soluciones robóticas en el mundo de la industria, donde realizan un amplio rango de actividades, muchas de ellas de naturaleza crítica (Moreno *et al*., 2012; Gómez *et al*., 2015).

Es por esto que, paralelamente a dicho desarrollo, se han publicado numerosos trabajos que estudian y proponen arquitecturas para aumentar la fiabilidad y la tolerancia a fallos de los sistemas robóticos (Ladd *et al*., 2004; Basu, 2004; Ferruz *et al*., 2011; Cañas *et al*., 2014). No obstante, este hecho contrasta con el relativamente bajo número de publicaciones que abordan el estudio de la vulnerabilidad de los robots.

En contraste con la calificación de fiable, un sistema se considera vulnerable si el fallo de alguno de sus componentes pone en riesgo su integridad. La vulnerabilidad engloba tanto los fallos generados por causas estocásticas, como aquellos que pueden ser provocados de forma intencionada. Por tanto, el análisis de la misma abarca un espacio de problemas más amplio que el de aquellos que tradicionalmente son considerados como fallos más probables.

Aunque la vulnerabilidad ha sido estudiada en lo que se refiere al diseño del software (Arora *et al*., 2008; Heelan, 2011), la investigación sobre la vulnerabilidad robótica no ha sido abordada con mucha asiduidad; de hecho, apenas se ha considerado el efecto de fallos o anomalías hardware que supongan un riesgo para el propio robot. En este sentido, cabe destacar algunos trabajos pioneros que estudian la vulnerabilidad del hardware de los robots frente a ciertos tipos de ataques externos (Nobile, 2012; Sheppard y Thompson, 2014).

Debido a la estructura de los sistemas robóticos modernos, las vulnerabilidades más críticas afectan al hardware de control y actuación. En la actualidad, prácticamente todas las plataformas robóticas se implementan en base al uso de microprocesadores. Aunque las vulnerabilidades de los sistemas basados en microprocesadores se han analizado en diversos trabajos (Huang, 2003; Bruschi *et al*., 2005; Tehranipoor y Koushanfaar, 2010; Karaklajic y Verbauwhede, 2013), el estudio específico de la vulnerabilidad de los sistemas de control y actuación de bajo nivel no ha sido suficientemente abordado. Recientemente se han publicado trabajos que consideran las consecuencias de ciertos ataques hardware en sistemas robóticos (Gomez-Bravo *et al*.,

* Autor en correspondencia.
Correos electrónicos: fernando.gomez@diesia.uhu.es (Fernando Gómez-Bravo),

http://dx.doi.org/10.1016/j.riai.2017.02.002
Disponible en www.sciencedirect.com

2015; Gomez-Bravo *et al*., 2016). Estos trabajos abordan el estudio de los ataques a la señal de reloj en el proceso de comunicación con los controladores de bajo nivel. Sin embargo, en ellos se contempla exclusivamente la viabilidad y los efectos que puede tener la inserción intencionada fallos sin describir el procedimiento necesario para hacer realidad la inserción, ni los detalles de la plataforma utilizada para ello. En contraste, el objetivo de este trabajo se centra en la descripción de una plataforma de 'hardware configurable' utilizada en el estudio de la vulnerabilidad y el correspondiente comportamiento de los módulos que permiten emular la inserción de fallos.

Por 'hardware configurable' se entiende que todos los elementos hardware que intervienen en el proceso pueden ser monitorizados, y, si es el caso, alterado su comportamiento durante el funcionamiento del robot, simulando de esta forma una situación de fallo. Dentro del concepto de controladores de bajo nivel se engloban todos aquellos elementos responsables de asegurar que los sistemas de actuación del robot se comporten de acuerdo con lo especificado por los elementos de jerarquía superior. Con este fin, se ha diseñado un sistema instrumental y una plataforma que emula la cinemática de un robot diferencial. La plataforma está configurada alrededor de un dispositivo FPGA (Field Programmable Gate Array) que implementa los módulos más críticos que deben ser convenientemente monitorizados. Por su parte, el sistema instrumental posibilita la realización de múltiples medidas digitales y analógicas que permitan caracterizar los comportamientos anómalos de los elementos vulnerados. Más concretamente, el artículo presenta un caso de estudio en el que se analiza la vulnerabilidad del bus I2C en situaciones en las que la señal de reloj se ve alterada. Hasta ahora, han sido pocos los trabajos que han abordado el estudio de la tolerancia a fallos en el bus I2C (Fukuhara *et al*., 2004; Alkalai *et al*., 2006). En particular, la vulnerabilidad propuesta no ha sido analizada con profundidad, y mucho menos, el efecto de la misma en la ejecución de tareas de navegación en robot móviles. Sin embargo, como quedará demostrado, la repercusión de ésta puede ser dramática, lo que justifica ampliamente que su estudio merezca la atención de la comunidad científica.

En la siguiente sección del artículo, se detalla la estructura de la plataforma experimental. Posteriormente se exponen las anomalías consideradas, y se muestran los resultados experimentales obtenidos para el análisis de esta vulnerabilidad. El artículo finaliza con las conclusiones y futuros trabajos.

2. Descripción de la Plataforma

La plataforma desarrollada tiene como objetivo emular lo más fielmente posible el comportamiento de un robot móvil, permitiendo la interacción y/o alteración de los elementos de control de bajo nivel durante el proceso de navegación. En esta sección se describen los distintos componentes de la plataforma detallando: las características de la arquitectura de control empleada; el controlador de alto nivel; la implementación de los módulos de bajo nivel y el sistema de instrumentación y medida.

2.1. Arquitectura

La arquitectura de control que ha inspirado el diseño de la plataforma (ver Figura 1) sigue las líneas tradicionales que han sido propuestas y ampliamente utilizadas en la bibliografía sobre robótica móvil (Ollero et al. 1999; Ollero, 2001; Mínguez et al., 2004; Carbone y Gomez-Bravo, 2015, Nakhaeinia et al, 2015).

En ella, un controlador de alto nivel es el encargado de tomar decisiones con cierto grado de abstracción, mientras que otro u otros controladores, que funcionan de forma concurrente con el primero (controladores de bajo nivel), son los responsables de asegurar que los motores sigan el perfil cinemático correspondiente.

El bucle de control se cierra en ambos niveles alto y bajo: los controladores de bajo nivel consideran individualmente los valores de las velocidades de cada una de las ruedas, mientras que un módulo de localización tiene en cuenta distintos tipos de información, con el fin de determinar la posición y orientación del robot y asegurar el cierre del bucle en el alto nivel.

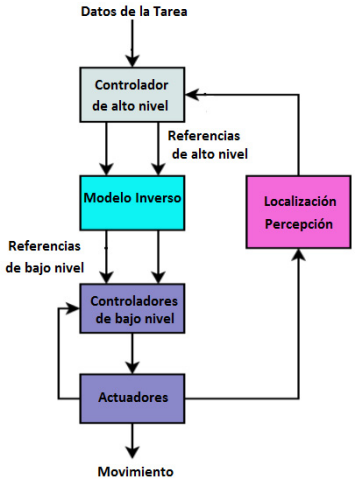


Figura 1: Arquitectura de control implementada en la plataforma.

El esquema funcional de la plataforma experimental desarrollada se muestra en la Figura 2 a). La plataforma se divide en tres zonas diferentes: un Ordenador Personal (PC) que ejecuta un programa, responsable entre otras cosas de actuar como controlador de alto nivel; un dispositivo FPGA para implementar los módulos controladores de bajo nivel; y un dispositivo I2C esclavo estándar, para tareas de verificación, responsable de controlar el movimiento de los motores, y por tanto, de las ruedas del robot.

En la Figura 2 b) se muestran las diferentes partes de la plataforma real sobre la que se han realizado los experimentos. En ella se distinguen las tres partes básicas (PC, FPGA, y controlador de motores, al que se le ha unido el sistema de potencia), mostrando, en negro, el flujo básico de información entre ellos. También se han incluido los elementos utilizados para realizar las mediciones (analizador lógico y medidor de corriente, al que se le ha unido el PC para la visualización y procesado de las medidas), mostrando, en azul, el flujo básico de información entre los bloques básicos y los de instrumentación.

La monitorización de las transiciones lógicas ha sido fundamental para establecer el procedimiento de inserción de fallos y validar el correcto funcionamiento de los módulos diseñados. Asimismo, ha sido importante el estudio del consumo del sistema durante la inserción de fallos, con el fin de confirmar si dicho procedimiento afectaba o no, de forma nociva, al funcionamiento interno del robot. Por todo ello, los elementos de

medida y verificación utilizados han sido: un analizador lógico para monitorizar el bus I2C y las principales señales generadas en el dispositivo FPGA y un sistema medidor de corriente que permite estimar la potencia consumida por los motores.

Dada la cantidad y el volumen de la instrumentación incluida en el sistema, cabe destacar que no se consideró conveniente la implementación de ésta sobre una plataforma móvil. De hecho, como podrá comprobarse en la sección de experimentación, la realización de los experimentos en movimiento real habría puesto en peligro la integridad del sistema. Por este motivo, las ruedas de la plataforma no han reposado sobre el suelo y la plataforma ha permanecido estática. No obstante, el movimiento se ha estimado aplicando técnicas de odometría, a partir de las medidas de los encoders. Esta situación no invalida las conclusiones alcanzadas, pues lo que se demuestra en este trabajo son las situaciones de vulnerabilidad y los puntos débiles del hardware de control utilizado, por lo que no ha sido necesaria la ejecución real del movimiento de la plataforma.

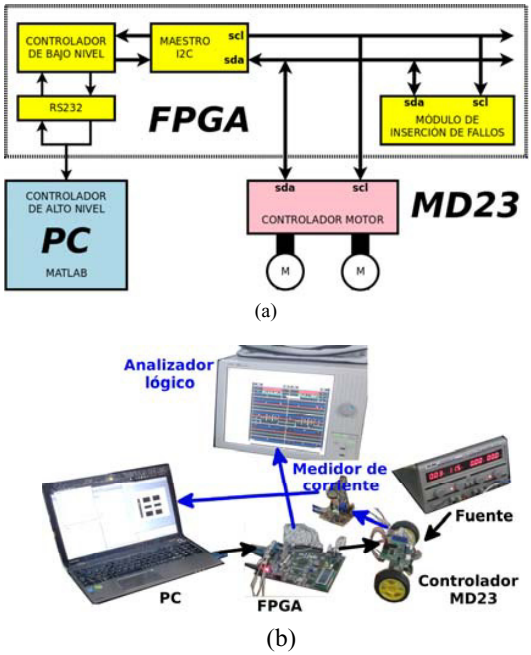


Figura 2: Plataforma experimental: (a) esquema funcional; (b) fotografía de la plataforma.

2.2. Controlador de alto nivel

Dado que el objeto fundamental de estudio ha sido la vulnerabilidad del hardware de bajo nivel, se ha elegido una aplicación que, utilizando la información procedente de dicho hardware, efectuara una tarea de navegación y a la vez permitiera identificar claramente los efectos de las vulnerabilidades detectadas. Así, se han descartado aplicaciones elaboradas donde la mayor complejidad de procesado podría haber dificultado la interpretación de los resultados. Concretamente, para el control de alto nivel se ha utilizado el algoritmo de persecución pura ("*pure pursuit*") (Ollero y Heredia, 1995; Ollero 2001, Morales et al., 2009). Se trata de un algoritmo de seguimiento de caminos ("*path following*") cuyo objetivo es asegurar que el robot sigue un camino predeterminado con la precisión adecuada. La elección de este algoritmo no restringe los resultados obtenidos en este trabajo. Conclusiones similares podrían haberse alcanzado con la elección de otras aplicaciones basadas en navegación reactiva, topológica, SLAM, etc si bien habría sido necesario utilizar un entramado sensorial más complejo que no habría aportado nada a la problemática vinculada al control de bajo nivel.

El algoritmo "pure pursuit" ha sido programado en un PC sobre una aplicación que se ejecuta sobre Matlab. En ella se integran tanto el cálculo de las referencias como la gestión de la interacción con la FPGA. En la Figura 3 se muestra el diagrama de flujo que ilustra el funcionamiento de dicha aplicación, la cual se ejecuta en paralelo con las rutinas propias de la FPGA.

Particularmente, el programa comienza con la configuración del puerto RS-232. A continuación, (considerando que son conocidos el camino a seguir y la posición y orientación del robot), se inicializa el proceso de seguimiento a través de un bucle que se ejecuta de forma iterativa. En el bucle se efectúan de forma secuencial los siguientes pasos. En primer lugar, el algoritmo de control de seguimiento, conociendo la configuración actual del robot, determina las velocidades que las ruedas han de adoptar para que el robot converja al camino. Las velocidades calculadas son enviadas a la FPGA y el programa queda a la espera de recibir los valores reales (medidos por los encoders) de las velocidades de las ruedas. A partir de ellos, y utilizando técnicas de odometría, un Filtro de Kalman Extendido (EKF) proporciona una estimación de la configuración alcanzada por el robot tras la ejecución del bucle. Este proceso es repetido sucesivamente, consiguiendo que el robot siga el camino establecido, hasta que éste alcance el final del mismo. En ese momento, se envía a los motores la orden de parada y finaliza el proceso.

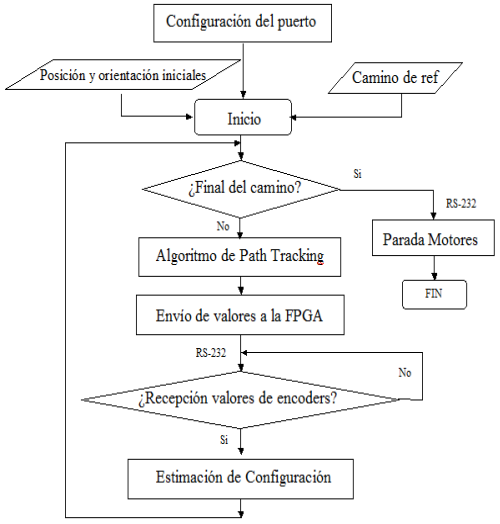


Figura 3: Diagrama de flujo que describe el funcionamiento del programa que se ejecuta en el PC.

Aunque en esta plataforma el posicionamiento del robot se ha obtenido únicamente mediante técnicas de odometría, la inclusión de otros tipos de técnicas de posicionamiento (como la utilización de GPS) no alteraría las conclusiones alcanzadas, porque las anomalías estudiadas no afectan a este procedimiento.

2.3. Implementación en FPGA.

El concepto de "hardware configurable", expresado al inicio de este artículo, puede materializarse de muchas maneras. La más

arcaica de todas sería la utilización de componentes discretos, lo que facilitaría el acceso a la monitorización y manipulación de todos los elementos que conforman la plataforma. Sin embargo, esta opción adolece de falta de flexibilidad, muy necesaria a la hora de solventar problemas o rediseñar la arquitectura del sistema. Por este motivo, se optó por la utilización de una FPGA. Gracias a ella, es posible tener un mayor control sobre las señales más críticas de la plataforma aprovechando la flexibilidad que dan los sistemas de hardware programable (Prieto et al., 2007). La placa de desarrollo utilizada es la placa Spartan 3AN Starter Kit Board (Chapman, 2006), que entraría en la categoría de dispositivos de bajo coste.

Los módulos desarrollados en dicho dispositivo son:

- Una UART basada en el protocolo RS232 para comunicar la FPGA con el PC.
- Un controlador de bajo nivel. Este módulo está compuesto por: un controlador de tipo PID para cada rueda; el interfaz que transforma los datos recibidos del controlador de alto nivel en datos interpretables por el maestro I2C y viceversa; el estimador de velocidades a partir de los valores leídos de los encoders.
- Un maestro I2C, para controlar la comunicación a través del protocolo I2C.
- Un módulo de inserción de fallos, para implementar las anomalías consideradas en las comunicaciones I2C

La utilización de un dispositivo FPGA como plataforma de implementación ha motivado el uso de un lenguaje de descripción de hardware de alto nivel, concretamente VHDL (Ashenden, 1990), para el diseño de los diferentes bloques y su conexionado. La metodología de diseño utilizada para los diferentes bloques ha sido su implementación en una máquina de estado algorítmica (Brown et al., 1981). En esta metodología, la implementación se divide en dos grandes bloques: un procesador, que realiza las operaciones del sistema; y un controlador, que realiza la secuencia de las operaciones. De esta forma, la implementación dispone de señales cuyo único significado son datos, y otras cuyo único significado son fases de operación.

A continuación se describen las características de los elementos más significativos dentro de la FPGA: controlador de bajo nivel; módulo de inserción de fallos y el sistema de monitorización de señales. La UART, y el módulo maestro han sido implementados siguiendo los estándares tradicionales definidos en la norma RS-232 y en la del bus I2C.

Controlador de bajo nivel.

La Figura 4 muestra un diagrama funcional del controlador de bajo nivel, mientras que la Figura 5 ilustra el funcionamiento secuencial del sistema.

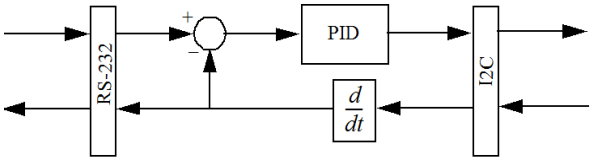


Figura 4: Esquema funcional del controlador de bajo nivel.

Tras la inicialización comienza un bucle sin fin. Dentro del mismo, en primer lugar, se atiende al buffer del puerto serie, con el fin de conocer los valores de referencia para las velocidades de las ruedas. Si no hay datos en el buffer se mantienen las referencias anteriores, si los hay, las referencias se actualizan. Seguidamente, el modulo PID calcula el error entre los valores deseados y los medidos con anterioridad, y determina los valores de la acción de control a ejecutar sobre las ruedas. Estos valores se envían a través del bus I2C. A continuación, se solicita al esclavo la lectura de los valores leídos por los encoders. Merece la pena resaltar que los encoders proporcionan información incremental, es decir, informan del número de pulsos leídos desde la última vez que fue solicitada la lectura. Los valores de las velocidades angulares de los motores se calculan dividiendo el número de pulsos por el tiempo transcurrido desde la última medida. De ello se encarga el mismo módulo del control de bajo nivel, que posee un contador de tiempo que determina el lapso temporal entre dos medidas consecutivas. Los valores de las velocidades así estimados son enviados al PC a través del puerto RS-232, a la vez que serán utilizados por el PID para establecer futuras acciones de control.

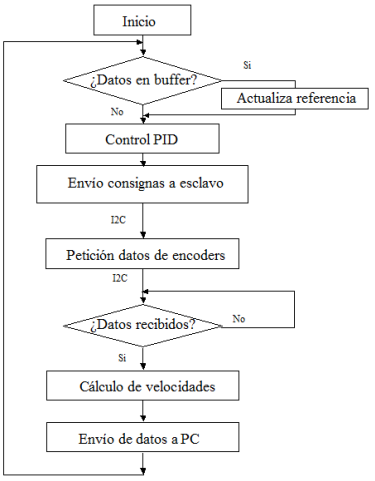


Figura 5: Diagrama de flujo que describe el funcionamiento del controlador de bajo nivel.

Módulo de inserción de fallos.

El modulo de inserción de fallos se ha diseñado para provocar de forma controlada y reversible un mal funcionamiento en el sistema de bajo nivel. Se trata de un módulo reconfigurable, para permitir probar en el futuro distintas anomalías que pudieran desvelar más vulnerabilidades. En este artículo se describe la configuración correspondiente al caso de estudio considerado: la perturbación del bus I2C.

En la Figura 6 se muestra el diagrama de flujo que representa el funcionamiento de este módulo, el cual ejecuta su tarea en paralelo con el resto de módulos que contiene la FPGA. Tras la inicialización de todos los componentes del bloque, la ejecución comienza con un bucle infinito, que mantiene al sistema a la espera de recibir la orden de ejecución de la perturbación. Una vez recibida ésta, se procede a leer un conjunto de conmutadores que contienen la información relativa a la perturbación (en este caso la dirección de esclavo sobre el que se quiere realizar la

perturbación y, opcionalmente, el registro cuya escritura se va a perturbar). Seguidamente, el sistema se mantiene escuchando y a la espera de detectar que se ha realizado una transmisión con las direcciones seleccionadas. En ese momento, el módulo procede a insertar la perturbación.

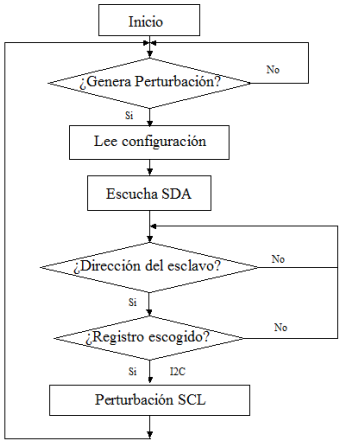


Figura 6: Diagrama de flujo que describe el funcionamiento del módulo de inserción de fallos.

Más concretamente la perturbación provocada consiste en transmitir a las líneas del bus I2C un estado de baja impedancia. En la Figura 7 se muestra un esquema funcional de este módulo. Dentro se representan las etapas de salida de los buffer triestado que permiten la escritura. Las perturbaciones así provocadas son totalmente reversibles, ya que, según la norma, las líneas del bus I2C trabajan en configuración "*pull-up*" y una baja impedancia no supone un cortocircuito en la línea.

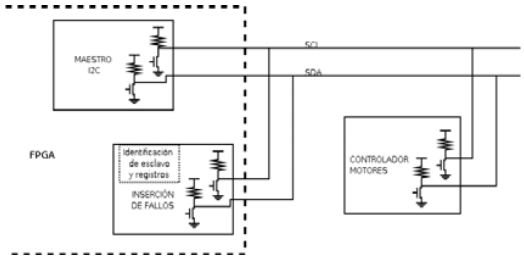


Figura 7: Esquema funcional del módulo de inserción de fallos

En definitiva, el módulo de inserción de fallos tiene un comportamiento parecido al de un módulo maestro, con la peculiaridad de que al escribir, fuerza en las líneas del bus un nivel bajo durante el tiempo que dura la anomalía provocada.

Sistema de interconexión y monitorización de señales

En la Figura 8 se muestra como se interconecta la FPGA con el resto de elementos de la plataforma. En primer lugar, se distinguen las conexiones que permiten programar y alimentar la placa. En segundo lugar, destacadas en rojo, se señalan las utilizadas para la comunicación con el controlador de alto nivel, y con el esclavo I2C estándar. Finalmente, las zonas destacadas en azul identifican las conexiones y elementos utilizados para la monitorización de señales.

En este sentido, uno de los principales objetivos de la plataforma es facilitar la monitorización de las señales involucradas en el proceso de control, con el fin de verificar el comportamiento obtenido. Para ello, se han utilizado dos tipos diferentes de monitorización: las señales cuyo cambio es realizado a alta frecuencia (señales de comunicación entre el dispositivo FPGA y el resto de componentes, que se visualizarán utilizando el analizador lógico); las señales cuyo cambio es realizado a baja frecuencia (los estados de configuración del sistema, que se visualizarán utilizando diodos LEDs).

Las fases de operación consideradas han conllevado la monitorización de los grupos de señales de alta frecuencia, que se corresponden con:

- Inserción de vulnerabilidades.
- Comportamiento del controlador MD23.
- Comportamiento del maestro del protocolo I2C.
- Captura de los datos de los encoders
- Comportamiento de la comunicación RS232 con el PC.
- Comportamiento del protocolo I2C.

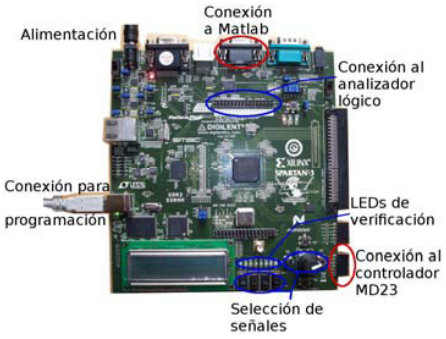


Figura 8: Conexión de la placa de desarrollo con los elementos restantes de la plataforma de experimentación.

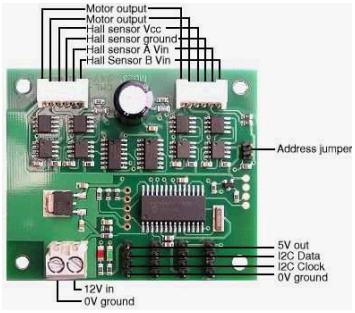
Además, la utilización del analizador lógico implica la generación de una señal de disparo ('trigger'). Dicha señal entra dentro de la categoría de alta frecuencia, y se utiliza para indicar al analizador el momento de comienzo de captura de datos, de forma que éste registre el número de eventos adecuados, monitorizando el comportamiento completo de las fases anteriores.

No obstante, una de las principales limitaciones de los dispositivos FPGAs es su limitado número de conexiones, el cual está aun más limitado en las placas de desarrollo. Como consecuencia de ello, no es posible monitorizar al mismo tiempo todas las señales enumeradas anteriormente. Por tanto, se ha utilizado una estrategia de multiplexación mediante selección, para permitir el acceso a la monitorización de las señales según se hayan escogido. Esto se logra mediante un proceso de configuración que involucra el uso de dos pulsadores y cuatro conmutadores. Un primer pulsador se utiliza para activar el valor de la señal de disparo. Un segundo pulsador se utiliza para activar la selección de las señales a monitorizar. Dicha selección se configura mediante los cuatro conmutadores, que conforman un valor de 4 bits. Este valor determina: el grupo de señales a monitorizar y el momento de activación de la señal de disparo. Dicha configuración es monitorizada correspondientemente por un grupo de diodos LEDs.

2.4. Esclavo I2C estándar y motores

El controlador de motores utilizado en la plataforma experimental es el MD23 (mostrado en la Figura 9 a), que ha sido diseñado para interactuar con el modelo de motor EMG30

(mostrado en la Figura 9 b). Dicho controlador dispone de una única dirección I2C, pudiendo controlar hasta dos motores (correspondientes a las ruedas derecha e izquierda respectivamente).



(a)

(b)

Figura 9: Control de bajo nivel: (a) controlador MD23; (b) motor EMG30.

La propiedad más destacable del motor EMG30 es que integra un sistema de 'encoders' que proporcionan 360 cuentas por vuelta. Dichos sensores están unidos a una caja reductora de 30:1. Las especificaciones más interesantes de dicho motor son: alimentación 12v; velocidad comprendida entre 1.5rpm y 200rpm; par máximo de 1.5kg/cm; corriente sin carga 150mA; corriente máxima con carga 530 mA.

Cabe destacar que la interacción desde el dispositivo FPGA hacia el controlador MD23 (y por tanto a los motores) se realiza mediante el acceso a registros internos del controlador. Concretamente, cada motor tiene asociado un registro, de tal forma que, para enviar una acción de control sobre un motor u otro se escribe en la dirección del registro correspondiente. Igualmente se procede para la lectura de los encoders. Los registros con los que se ha interactuado en la experimentación son los indicados en la Tabla 1.

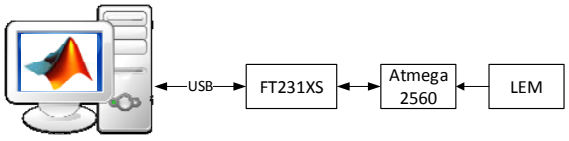
Tabla 1: Registros del controlador MD23 utilizados en la plataforma experimental

Registro	Misión
0x00	Velocidad del motor 1
0x01	Velocidad del motor 2
0x02-0x05	Lectura de los cuatro bytes del encoder del motor 1
0x06-0x09	Lectura de los cuatro bytes del encoder del motor 2
0x10	Deshabilitar la parada a los dos segundos

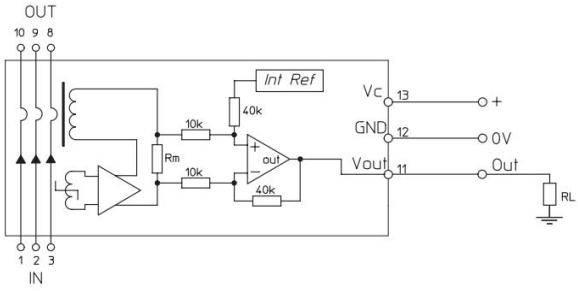
2.1. Sistema de instrumentación.

El sistema de instrumentación utilizado se puede dividir en dos grandes grupos: instrumentación digital y analógica. El bloque de instrumentación digital está compuesto por un analizador lógico y la electrónica que da acceso a las señales que deben ser monitorizadas. Este bloque de instrumentación ya ha sido tratado en el apartado 2.3.

El sistema de medición analógico se ha implementado con objeto de tomar medidas del consumo de ambos motores. Con ello se ha pretendido supervisar el efecto que cualquiera de los fallos insertados pudiera provocar en el desempeño del sistema de tracción. Aunque el controlador MD23 permite la medición de la potencia consumida, dicha medición no puede ser continua en el tiempo, puesto que requiere la lectura de los registros adecuados. Por tanto, se ha diseñado un sistema de instrumentación externo, que captura las señales del consumo, siendo registradas éstas por el PC, a la misma vez que se ejecuta el programa principal. Dicho sistema está compuesto por un microcontrolador ATMEGA2560, un transceiver (FT231XS) y un transductor de corriente (LEM CAS-6NP), como se muestra en la Figura 10 a). El transductor de corriente es un dispositivo electrónico que posee una entrada, que alimenta el primario de una bobina, y una salida en baja impedancia, que suministra una corriente proporcional y amplificada a la que circula por el primario. El esquema eléctrico del transductor se muestra en la Figura 10 b). El primario de cada transductor se conecta en serie con el circuito de alimentación de cada motor, de forma que en cada experimento es posible registrar el consumo de ambos motores.



(a)



(b)

Figura 10: Instrumentación analógica: (a) diagrama de bloques; (b) esquema del transductor de corriente.

3. Caso de estudio: Vulnerabilidad del bus I2C

En esta sección se presenta el estudio realizado sobre la comunicación entre el sistema de control de bajo nivel y los motores, presentándola como ejemplo de uso de la plataforma para el estudio de una vulnerabilidad concreta. Especialmente, con esta investigación se ha pretendido analizar los efectos que tiene la inyección de fallos en la señal de reloj que controla la comunicación en el bus I2C.

Esta interferencia puede aparecer en diversas fases del proceso de comunicación. De todos ellas, la tranmisión de órdenes desde el maestro hacia el esclavo es la fase en la que, sin duda, este tipo de perturbación puede tener un efecto más negativo. Más concretamente, en la plataforma diseñada se han interferido los comandos enviados por parte del maestro I2C al esclavo

controlador de motores. Con la idea de estudiar la aparición de fallos, tantos fortuitos como provocados intencionadamente, se han explorado varias posibilidades, interfiriendo la comunicación con un solo motor, con los dos motores, de forma temporal o de forma permanente.

Como se muestra en la sección de experimentación, según sean las condiciones en que se producen estas perturbaciones, el curso de robot puede verse afectado en mayor o menor medida, poniendo en peligro la integridad de la aplicación robótica.

3.1. Vulnerabilidad del bus I2C

El protocolo I2C se basa en la transmisión de dos señales, generalmente llamadas SDA (línea de datos) y SCL (línea de reloj) y eventualmente una señal de tierra (todos los elementos deben tener la misma referencia de voltaje). Todos los módulos incluidos en el sistema de comunicaciones están conectados a las mismas líneas del bus, como se muestra en la Figura 11 a). Existen dos tipos de módulos: un módulo principal o maestro que genera la señal SCL y controla las transmisiones; y uno o varios módulos esclavos que serán la fuente (operaciones de lectura) o destino (operaciones de escritura) de la información. El comportamiento de una operación de escritura usando un protocolo I2C se ilustra en la Fig. 11 b) y está ampliamente descrito en la literatura científica y técnica (Hamblen, 2013). La señal SDA es generada tanto por el maestro como por los módulos esclavos cuando se produce la comunicación. Con el fin de que diferentes módulos sean capaces de manipular una misma señal sin problemas (colisión de información), los valores lógicos en el protocolo I2C son: tierra, para el nivel bajo (común a todos los módulos); y alta impedancia, para el nivel alto.

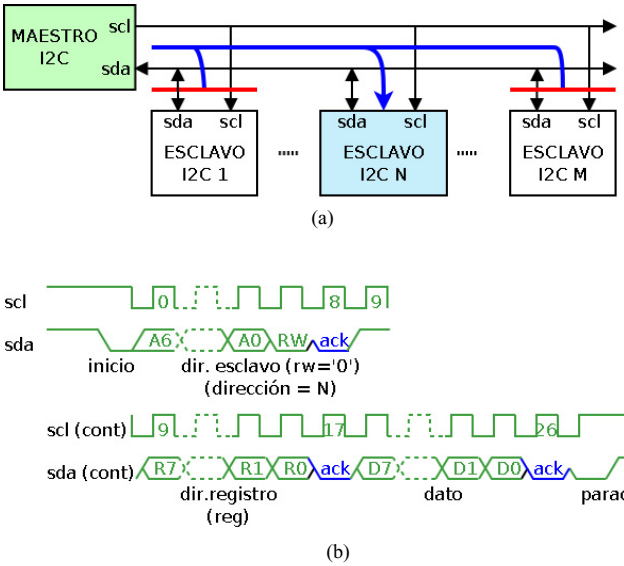


Figura 11: Bus I2C: (a) arquitectura de un sistema basado en el protocolo I2C; (b) señales del procedimiento de escritura en el protocolo I2C.

La utilización de alta impedancia tiene una doble implicación. En primer lugar, en caso de colisión de información (ocurre cuando simultáneamente un nodo trata de escribir un '1' lógico y otro nodo un '0' lógico) resulta dominante el '0' lógico (evitando un valor lógico inespecífico). En segundo lugar, los diferentes módulos no tienen que utilizar la misma fuente de polarización y, por lo tanto, los módulos que necesitan alta tensión pueden conectarse al mismo bus que los módulos necesitados de baja tensión. Desde el punto de vista del análisis de la vulnerabilidad, estas implicaciones representan un inconveniente manifiesto. Si una colisión de información (bien fortuita o intencionada) tiene lugar en la señal SCL, la comunicación con el controlador esclavo queda totalmente inhibida. Aun cuando ésta se detectase (por cualquiera de los controladores de alto o bajo nivel), sería imposible comunicar al esclavo una orden de parada, ya que éste no tendría referencia temporal para monitorizar los diferentes bits de la transmisión.

Con el fin de estudiar la repercusión de esta situación de fallo en el proceso de navegación del robot, se ha utilizado el módulo de inserción de fallos presentado en la sección 2.3. Mediante el mismo, se ha provocado, de forma controlada, una colisión de información en la línea SCL, en los momentos en que el maestro I2C se comunica con el esclavo para transmitir una acción de control sobre uno o ambos motores. Un fallo de inserción típico se puede apreciar en la Figura 12. La gráfica se encuentra dividida en tres grandes sectores:

- Cronograma de estados del módulo de Inserción de fallos (fondo gris).
- Señal de inserción de fallos (fondo crema): CLK_Insert (activación de fallo en SCL).
- Señales del bus (fondo blanco): SDA y SCL MASTER (generadas por el maestro); SCL CONTROLLER (señal leída por el MD23).

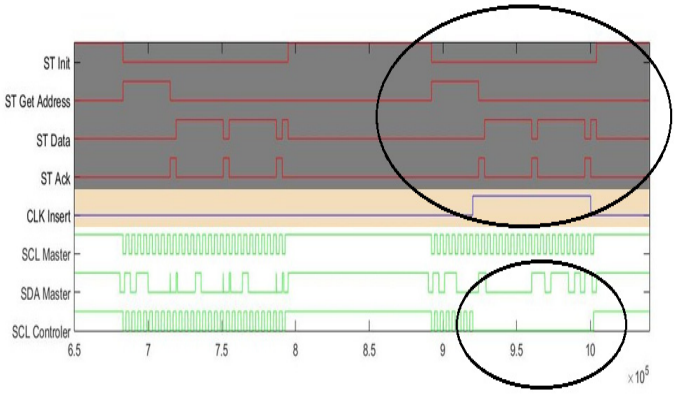


Figura 12: Graficas de las señales obtenidas con el analizador lógico durante el estudio de la inserción de un fallo.

En un principio, y tras ser activado, el módulo de inserción se encuentra en un estado inicial (St Ini a nivel alto) a la espera de la llegada de una comunicación. Cuando esto sucede, el modulo pasa al estado de lectura de la dirección del esclavo (ST Get address). Una vez el maestro escribe dicha dirección (ver señal SDA Master) el módulo de inserción la lee, y evoluciona al estado de identificación de datos (St Data) y posteriormente al de identificación de reconocimiento (ST Ack). Cuando llega el fin de la transmisión, el módulo vuelve al estado inicial (St Ini vuelve a nivel alto). La inserción de fallo se producirá cuando la dirección del esclavo incluido en la transmisión sea la misma que la previamente seleccionada, independientemente del registro con el que se establece la comunicación. Esta situación se observa en la zona marcada de la Figura 12, cuando la señal CLK Insert pasa a nivel alto. Puede observarse que en la primera transmisión no se ha activado el fallo, mientras que en la segunda sí. En la zona

marcada inferiormente, se observa el comportamiento ante el fallo. El maestro genera la señales de forma adecuada (SDA Master y SCL Master), mientras que la señal SCL recibida por el esclavo (SCL controller) no es la adecuada ya que se mantiene a nivel bajo durante la mayor parte de la transmisión, instante en el que el maestro trata de transmitir el valor de la velocidad al motor.

3.2. Metodología Experimental.

La operatividad para realizar los experimentos ha sido la siguiente. A partir de un escenario particular se ha definido una trayectoria que el robot ha de seguir con precisión. Dicha trayectoria es entregada al programa de control de alto nivel. Este programa se ejecuta según lo detallado en la sección 2.2.

En los casos que se presentan a continuación, la inserción de fallos en la línea de reloj se realiza en un sentido descendente, es decir, cuando se envía un dato hacia el esclavo. En la Figura 13, se muestra el recorrido de los datos en la plataforma experimental. Más concretamente, el interfaz del programa de control del robot permite seleccionar cuándo se desea insertar un fallo en la línea del reloj. En el momento así seleccionado, cuando el maestro trata de comunicar con el motor que se desea afectar, el módulo de inserción actúa sobre la señal de reloj. Durante el tiempo que dura la orden de inserción la etapa de salida del módulo hace que la línea SCL tome el valor 0 voltios. Debido a esto, el módulo esclavo no percibe transiciones en la línea del reloj, y por tanto, no entiende que se le esté transmitiendo ningún dato, manteniendo de esta manera el valor de velocidad que anteriormente le ha sido ordenado.

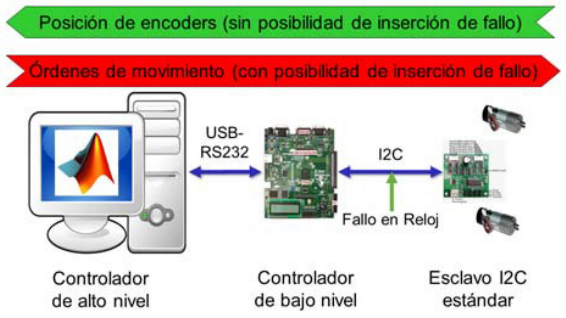


Figura 13: Dirección de la inserción de fallos en la plataforma.

La inserción de fallos propuesta, puede tener distintos orígenes. Bien puede ser provocada por una causa fortuita (sería el caso de una avería que provoque un cortocircuito en la conexión del esclavo a la línea SCL). O incluso puede ser generada de forma intencional, con el fin de implementar un ataque hardware sobre el robot. Esta situación se describe en: Gomez-Bravo *et al*., (2015) y Gomez-Bravo *et al*. (2016). En el primero de los casos, la anomalía tendría muchas probabilidades de ser permanente, lo que contribuiría a una fácil detección de la misma. Por el contrario, en el segundo caso, la anomalía sería de carácter temporal, lo cual dificultaría la caracterización del problema y además aparecería, con alta probabilidad, en aquellos momentos en los que la vulnerabilidad afectara con mayor grado a la fiabilidad de la aplicación robótica. Ambas posibilidades se exploran en la sección de resultados experimentales.

En los experimentos propuestos, tras insertar el fallo, el controlador de alto nivel identifica que el robot no sigue la trayectoria correcta. Éste, siempre tratará de corregir el comportamiento anómalo. No obstante, mientras dure la inserción estas correcciones nunca llegan a los motores. Un posible intento de corrección por parte del mismo MD 23 está descartado, porque dicho elemento, al no recibir señal de reloj, no identifica que se quieren comunicar con él. Por tanto, la detección del comportamiento anómalo, bien sea por parte del controlador de alto nivel o por otro módulo externo al esclavo, no permite contrarrestar las consecuencias negativas del fallo insertado. No parecen ser productivas las soluciones en ese sentido; por contra si parece más apropiado el diseño de un esclavo, con cierta inteligencia, que sea capaz de detectar este tipo de fallos.
En la siguiente sección se muestran distintas situaciones en las que se ha procedido a insertar este tipo de fallo. Como se verá, la vulnerabilidad propuesta afecta con distintos grados de intensidad a la fiabilidad del robot, de manera que las conclusiones alcanzadas, van más allá de lo que una simple predicción pudiera prever.

4. Resultados Experimentales

Se han realizado numerosos experimentos con varios escenarios. La idea ha sido caracterizar un conjunto de circunstancias en las que la inserción de fallos provoca en el robot comportamientos no deseables, poniendo de manifiesto la envergadura de la vulnerabilidad estudiada. En la Figura 14, se muestra el resultado obtenido con la plataforma experimental en un experimento de seguimiento de una trayectoria sin que se produzca la inserción de ningún fallo.

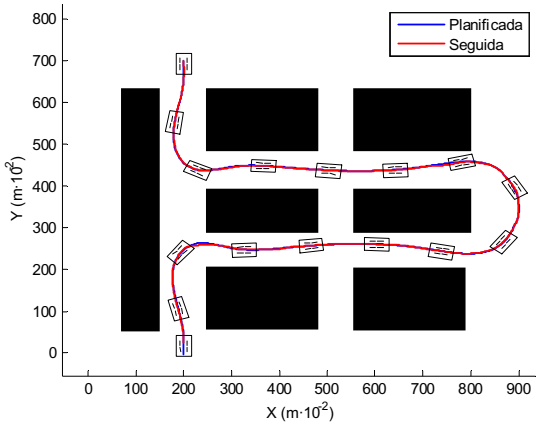
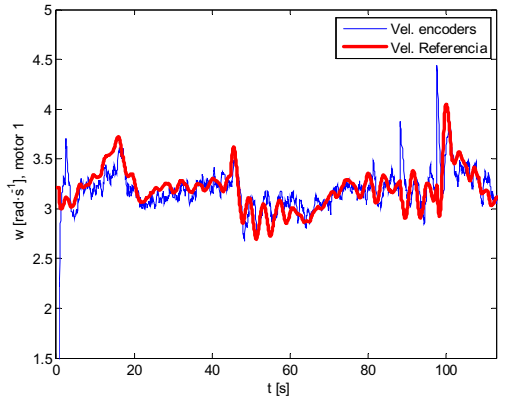


Figura 14: Trayectoria realizada por el robot sin inserción de fallos.

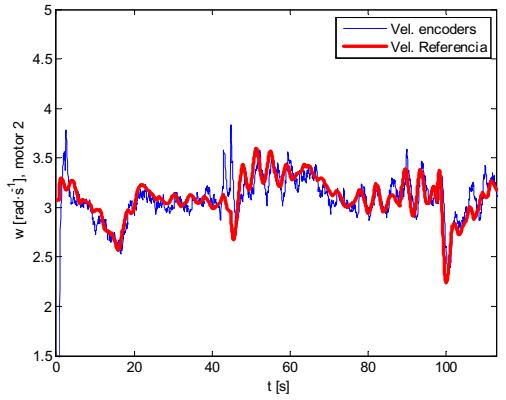
En esta figura se presentan: la trayectoria de referencia (azul), la trayectoria seguida por el robot (rojo) en el entorno definido por el mapa, así como el contorno del robot en distintos puntos del experimento. Como era de esperar, la trayectoria seguida prácticamente coincide con el camino de referencia. En la Figura 15 se ilustran, para cada motor: las referencias de velocidad angular entregadas al controlador de bajo nivel (línea gruesa roja); la evolución de dichas velocidades medidas a partir del las señales de los encoders (línea fina azul); y el consumo de intensidad de cada motor.

Nótese que, al no haber inserción de fallos, la evolución de la velocidad de las ruedas sigue con bastante precisión la referencia generada por el controlador de alto nivel. Obviamente, la inserción de un fallo permanente en la señal de reloj durante el proceso de escritura en ambos motores, tiene una repercusión

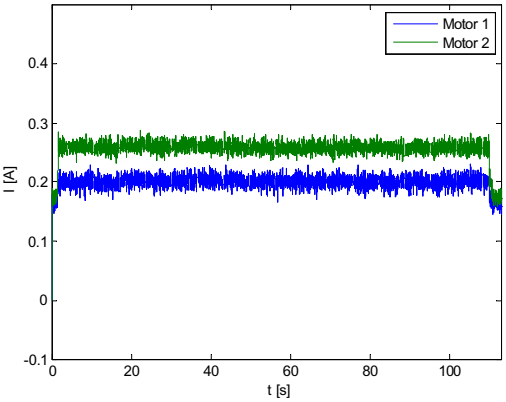
dramática sobre la integridad del robot. Aun cuando éste se produzca en una zona en la que el vehículo viaja a lo largo de una sección más o menos recta, la falta de control sobre los motores genera una trayectoria que, con mucha probabilidad, acaba en colisión con los obstáculos que definen el escenario, ver Figuras 16 a) y b). En ambas gráficas se muestran dos experimentos en los que se inserta un fallo, de forma permanente, en diferentes partes de la trayectoria.



(a)



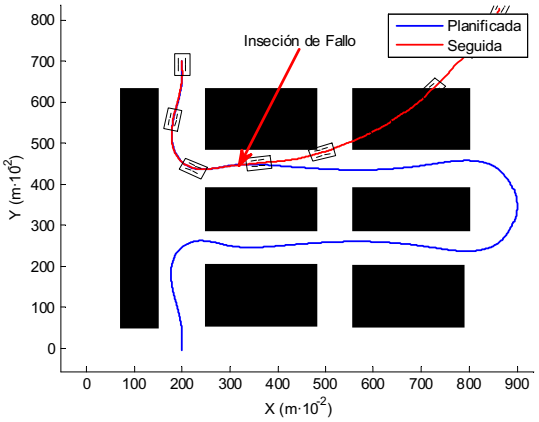
(b)



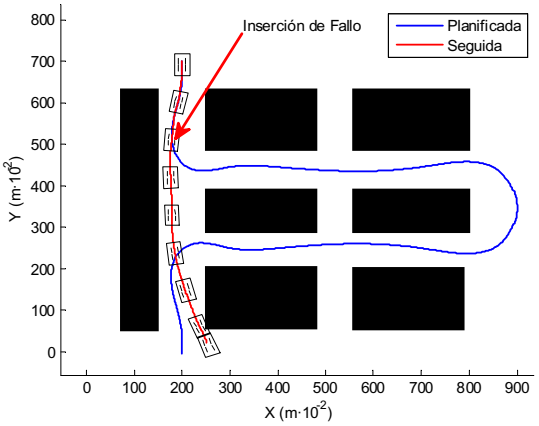
(c)

Figura 15: Experimento sin inserción de fallo: a) referencia y velocidad angular del motor 1; b) referencia y velocidad angular del motor 2; c) intensidades de ambos motores.

En la Figura 17 se representan la evolución de las velocidades medidas de los motores y sus referencias para el experimento de la Figura 16 b). Adicionalmente se incluye la evolución del estado del sistema de inserción de fallo (línea negra gruesa). Obsérvese que esta última señal se mantiene a bajo nivel hasta que, en un determinado instante, cuando se produce el fallo, adquiere un nivel lógico alto (los valores asociados a los niveles lógicos se han elegido para facilitar la representación).



(a)



(b)

Figura 16: a) y b): Trayectorias con fallo permanente al escribir en ambos motores.

A partir del momento en que se produce la inserción del fallo, la Figuras 17 a) y b) muestran cómo ambos motores dejan de seguir la referencia procedente del controlador de alto nivel, manteniendo aproximadamente el valor de velocidad que tenían en el momento de aparecer el fallo en la línea del reloj. Lo que ocurre, es que durante el tiempo en que tiene lugar la inserción, el esclavo del bus I2C mantiene la velocidad de los motores con los últimos valores que ha registrado, entendiendo que todo funciona correctamente ya que al no recibir señal de reloj, piensa que no hay comunicación en curso.

Otro dato importante a resaltar es que la intensidad consumida por los motores (Figura 17 c) no presenta ninguna anomalía, ni comportamiento diferente antes o después de la inserción del fallo. Lo que indica, que esta medida no puede ser utilizada para detectar el fallo estudiado en este trabajo.

Una situación diferente y muy interesante aparece cuando la inserción de fallo afecta sólo a un motor. Esta situación podría darse si cada motor estuviese controlado por un MD23 distinto, y una anomalía apareciera en la sección de la línea SCL que se conecta a uno de los controladores. Cuando se reprodujo esta situación, el controlador de alto nivel si fue capaz de compensar la perturbación provocada por el fallo insertado (ver Figura 18).

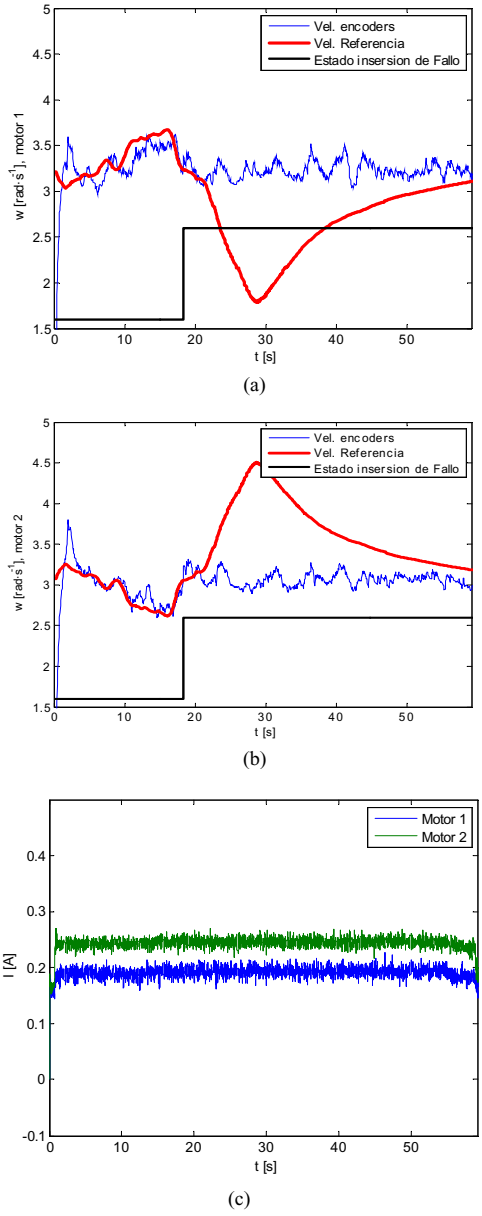


Figura 17: Experimento con inserción de fallo en ambos motores: a) velocidad, referencia e inserción de fallo en motor 1; b) velocidad, referencia e inserción de fallo en motor 2; c) intensidades consumidas por los motores.

Teniendo en cuenta la estimación de la posición y realizando la acción de control sobre un solo motor, el controlador calcula las velocidades, de manera que mantiene el valor de la velocidad para el motor no controlado y ajusta la del motor no perturbado, de manera que el robot sigue el camino, con un error no despreciable, pero con suficiente éxito como para hacer que el vehículo llegue hasta el punto objetivo final.

En la Figura 19 se puede observar cómo la velocidad del motor 1 evoluciona siguiendo su referencia y cómo el motor 2, que presenta un fallo insertado de forma permanente (desde el principio del experimento), mantiene un valor de velocidad aproximadamente constante, sin obedecer a las consignas de la referencia.

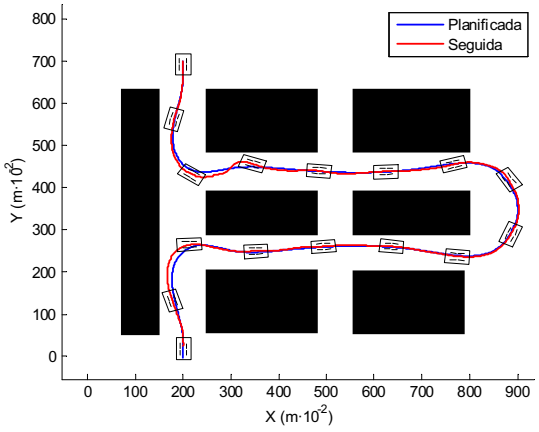


Figura 18: Trayectoria realizada por el robot insertando un fallo permanente en motor 2.

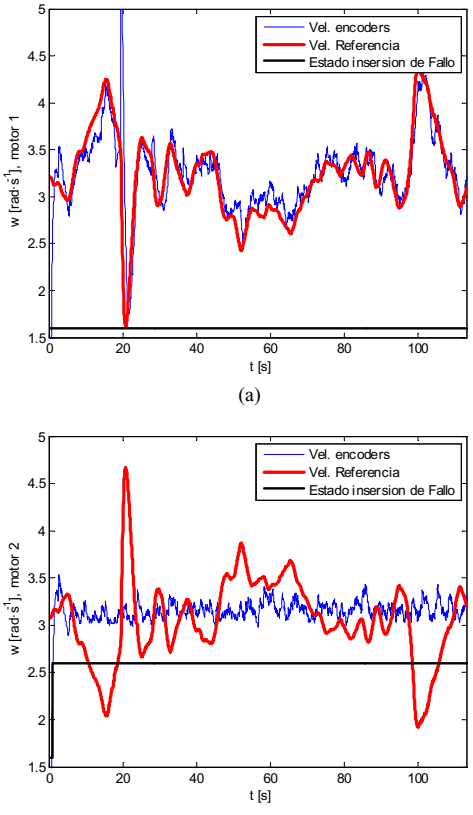


Figura 19: Fallo en un solo motor: a) referencia y velocidad angular en motor 1 (sin fallo); b) Referencia y velocidad angular motor 2 (con fallo).

Estos resultados permiten concluir que disponer de un esclavo distinto para cada motor puede suponer una ventaja, ya que la aparición de fallos en solo uno de los motores no impide la ejecución de la navegación, aunque la precisión de la misma se devalúe.

Otra situación cuyo análisis proporciona interesantes resultados aparece cuando la inserción de fallos se realiza sobre ambos motores, pero temporalmente. Esta situación podría ser consecuencia de un ataque hardware tal y como se comentó anteriormente. En este caso, la inserción dura unos breves instantes, y es efectuada en lugares estratégicos. Si la perturbación es efectuada con precisión, es posible modificar el curso del robot de forma que, sin causar daños en el mismo, y sin que aparentemente exista causa externa, el robot, por ejemplo, evite visitar una zona determinada. Para ilustrar esta situación se ha escogido el mismo escenario que en los experimentos anteriores. Así, en la Figura 20 se muestran las zonas donde se inicia y donde termina la inserción del fallo. Igualmente se muestra la trayectoria recorrida por el robot. Obsérvese que dicha trayectoria solo coincide con parte del camino inicialmente definido.

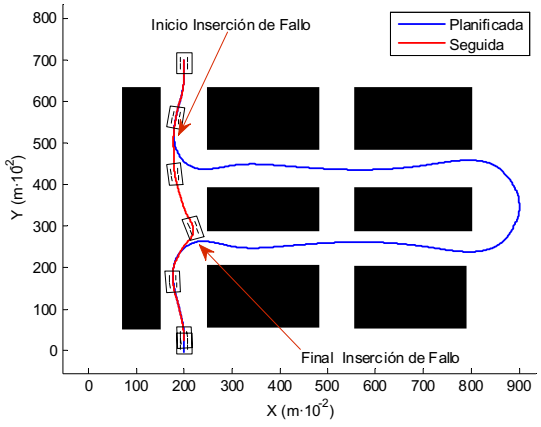
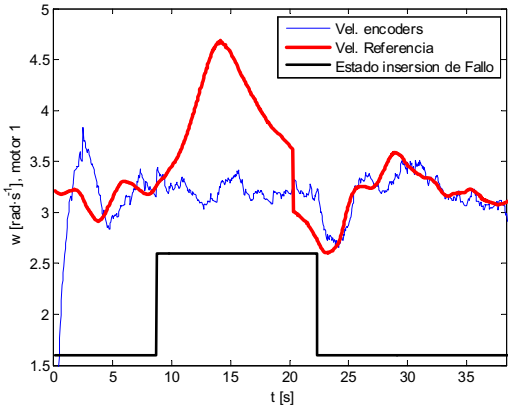


Figura 20: Trayectoria del robot con una inserción temporal selectiva.

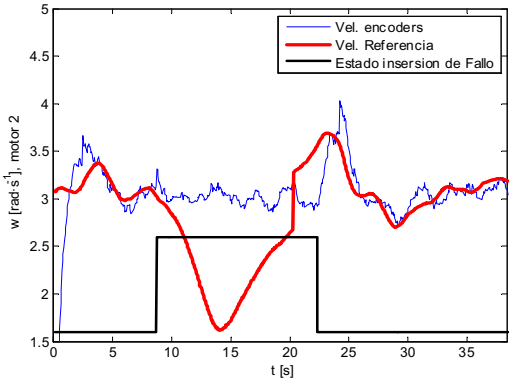
En la Figura 21 (a) y (b), se muestra cómo la señal de control generada para los dos motores (ambas gráficas en rojo), intenta obligar a la plataforma a cambiar el sentido de la marcha. Esto lo realiza el controlador de alto nivel bajando el valor de la velocidad deseada para el motor 2 y aumentando de forma considerable el valor de la velocidad deseada para el motor 1. Sin embargo, los motores hacen caso omiso. Posteriormente, cuando el fallo insertado desaparece, el controlador de alto nivel es capaz de hacer que el robot converja hacia la sección del camino original que queda más próxima al vehículo. Es por este motivo que el robot se mueve hasta alcanzar la sección final del mismo. Si esta perturbación hubiese sido generada de forma intencionada, habría conseguido que el robot visitara una parte de la trayectoria de referencia. Obsérvese en la Figura 21 (a) y (b), que a partir del cese de la inserción del fallo, la velocidades de los motores siguen con bastante precisión la señal de referencia.

5. Conclusiones y futuros trabajos

En este trabajo se presenta con detalle, el desarrollo de una plataforma para el estudio de la vulnerabilidad hardware y el comportamiento de los controladores de bajo nivel en el ámbito



(a)



(b)

Figura 21: Inserción temporal selectiva: a) referencia y velocidad angular del motor 1 (con fallo temporal); b) referencia y velocidad angular del motor 2 (con fallo temporal).

de la robótica móvil. La plataforma ha sido diseñada en base al concepto de hardware configurable, de tal forma que los elementos hardware que intervienen en el proceso pueden ser monitorizados, y, si es el caso, alterado su comportamiento durante el funcionamiento del robot, simulando de esta forma una situación de fallo. La plataforma dispone de un sistema de instrumentación que permite la realización de múltiples medidas digitales y analógicas que posibilitan la caracterización de los comportamientos anómalos de los elementos vulnerados. Particularmente, el artículo presenta un caso de estudio en el que se analiza la vulnerabilidad del bus I2C en situaciones en las que la señal de reloj se ve alterada. Los resultados experimentales alcanzados con la plataforma ponen de manifiesto la importancia de la vulnerabilidad estudiada. Un descubrimiento destacable consiste en la bondad de controlar cada motor con un esclavo distinto, de manera que las perturbaciones sufridas por uno pueden ser compensadas por el control realizado por el otro.

En la actualidad, los autores se encuentran trabajando en una metodología que permita detectar, justo el instante en el que se produce el tipo de fallo considerado. En este sentido, un futuro trabajo es el desarrollo de un sistema que permita establecer una reacción segura por parte del robot.

English Summary

Experimental Platform for Studying Hardware Vulnerabilities on Mobile Robots: I2C Bus, a Case of Study.

Abstract

This paper presents an experimental platform for studying the effects of hardware vulnerabilities in mobile robots. The platform is designed so that the hardware, involved in the navigation process, can be monitored during the robot operation and, if desired, their behavior can be altered, allowing the simulation of a failure. The paper shows a particular case of study: the I2C Bus vulnerability when some anomalies appear in the clock signal. A set of experimental results confirm the interest of the studied vulnerabilities and the applicability of the developed platform.

Keywords:

Mobile robot, hardware vulnerability, FPGA, configurable hardware.

Referencias

Alkalai, L., CHAU, S. N., Tai, A. T., 2006. Fault-tolerant communication channel structures. U.S. Patent No 7,020,076.

Anderson, R., Kuhn, M., 1996. Tamper Resistence a Cautionary Note", 2nd USENIX Workshop on Electronic Commerce Proceeding, 1-11.

Arora, A., Telang, R., & Xu, H., 2008. Optimal policy for software vulnerability disclosure. Management Science, 54(4), 642-656.

Ashenden, P. J., 1990. The VHDL cookbook. Department of Computer Science, University of Adelaide.

Basu, P., & Redi, J., 2004. Movement control algorithms for realization of fault-tolerant ad hoc robot networks. Network, IEEE, 18(4), 36-44.

Brown, D. W., 1981. A state-machine synthesizer—SMS. In Proceedings of the 18th Design Automation Conference, 301-305.

Bruschi, D., Cavallaro, L., Lanzi, A., 2005. Replay Attack in TCG Specification and Solution. In Proceedings of the 21st Annual Computer Security Applications Conference, IEEE Computer Sociecity, 127–137. DOI: 10.1109/CSAC.2005.47.

Cañas, N., Hernández, W., González, G., Sergiyenko, O., 2014. Controladores multivariables para un vehículo autónomo terrestre: Comparación basada en la fiabilidad del software. Revista Iberoamericana de Automática e Informática Industrial RIAI, 11(2), 179-190. DOI: 10.1016/j.riai.2014.02.002

Chapman, K., 2006. Initial Design for Spartan-3E Starter Kit (LCD Display Control). Xilinx Ltd 16th February.

Chen, C. Y., Shih, B. Y., Shih, C. H., & Chou, W. C. 2012. RETRACTED: The development of autonomous low-cost biped mobile surveillance robot by intelligent bricks. Journal of Vibration and Control, 18(5), 577-586.

Carbone, G., Gomez-Barvo, F. 2015. Motion and Operation Planning of Robotic Systems. Springer International Publishing. Switzerland. DOI: 10.1007/978-3-319-14705-5.

Cuesta, F., Gómez-Bravo, F., & Ollero, A., 2004. Parking maneuvers of industrial-like electrical vehicles with and without trailer. Industrial Electronics, IEEE Transactions on, 51(2), 257-269. DOI: 10.1109/TIE.2004.824855.

Ferruz, J., Vega, V. M., Ollero, A., & Blanco, V., 2011. Reconfigurable control architecture for distributed systems in the HERO autonomous helicopter. Industrial Electronics, IEEE Transactions on, 58(12), 5311-5318. DOI: 10.1109/TIE.2010.2046003.

Fukuhara, R., Day, L., Luong, H. H., Rasmussen, R., & Chau, S. N., 2004. I2C bus protocol controller with fault tolerance. U.S. Patent No. 6,728,908. Washington, DC: U.S. Patent and Trademark Office.

Garcia-Cerezo, A., Mandow, A., Martinez, J. L., Gómez-de-Gabriel, J., Morales, J., Cruz, A., Seron, J., 2007. Development of ALACRANE: A mobile robotic assistance for exploration and rescue missions. In Safety, Security and Rescue Robotics, 2007. SSRR 2007. IEEE International Workshop on, pp 1-6. DOI: 10.1109/SSRR.2007.4381269.

Gomez-Bravo, F., Naharro, R. J., García, J. M., Galán, J. G., & Raya, M. S., 2015. Sobre la vulnerabilidad de los robots móviles frente a los ataques hardware. XXXVI Jornadas de Automática, pp. 358-365.

Gomez-Bravo, F., Naharro, R. J., García, J. M., Galán, J. G., & Raya, M. S. (2016). Hardware Attacks on Mobile Robots: I2C Clock Attacking. In Robot 2015: Second Iberian Robotics Conference, pp. 147-159.

Gómez, J. V., Vale, A., Garrido, S., & Moreno, L., 2015. Performance analysis of fast marching-based motion planning for autonomous mobile robots in ITER scenarios. Robotics and Autonomous Systems, 63, 36-49.

Hamblen, J.O., van Bekkum, G.M.E., 2013. An Embedded Systems Laboratory to Support Rapid Prototyping of Robotics and the Internet of Things, Education, IEEE Transactions on, 56 (1), 121-128.

Heelan, S. (2011). Vulnerability detection systems: Think cyborg, not robot. IEEE Security & Privacy, (3), 74-77. DOI: 10.1109/MSP.2011.70.

Huang, A., (2003). Hacking the Xbox: An Introduction to Reverse Engineering, No Starch Press.

Jardón, A., Giménez, A., Correal, R., Martinez, S., & Balaguers, C., 2008. Asibot: Robot portátil de asistencia a discapacitados. Concepto, arquitectura de control y evaluación clínica. Revista Iberoamericana de Automática e Informática Industrial RIAI, 5(2), 48-59. DOI: 10.1016/S1697-7912(08)70144-4

Kachouie, R., Sedighadeli, S., Khosla, R., Chu, M. T. 2014. Socially assistive robots in elderly care: a mixed-method systematic literature review. International Journal of Human-Computer Interaction, 30(5), 369-393.

Karaklajic, D, Verbauwhede, I., 2013. Hardware Designer's Guide to Fault Attacks. IEEE Transactions on Very Large Scale Integration Systems, 21, 2295-2306. DOI: 10.1109/TVLSI.2012.2231707.

Ladd, A. M., Bekris, K. E., Rudys, A. P., Wallach, D. S., & Kavraki, L. E., 2004. On the feasibility of using wireless ethernet for indoor localization. IEEE Transactions on Robotics and Automation, 20(3), 555-559.

Marques, C., Cristóvão, J., Alvito, P., Lima, P., Frazão, J., Ribeiro, I., Ventura, R. 2007. A search and rescue robot with tele-operated tether docking system. Ind. Robot: An International Journal, 34(4), 332-338.

Minguez, J., Montesano, L., Montano, L. 2004. An architecture for sensor-based navigation in realistic dynamic and troublesome scenarios. In Proceedings of the Intelligent Robots and Systems International Conference on, Vol. 3, pp. 2750-2756. DOI: 10.1109/IROS.2004.1389825.

Moreno, H. A., Saltaren, R., Carrera, I., Puglisi, L., Aracil, R., 2012. Índices de desempeño de robots manipuladores: una revisión del estado del arte. Revista Iberoamericana de Automática e Informática Industrial RIAI, 9(2), 111-122. DOI: 10.1016/j.riai.2012.02.005.

Morales, J., Martínez, J. L., Martínez, M. A., Mandow, A. 2009. Pure-pursuit reactive path tracking for nonholonomic mobile robots with a 2D laser scanner. Journal on Advances in Signal Processing, 2009, 3. DOI:10.1155/2009/935237.

Nakhaeinia, D., Payeur, P., Hong, T. S., Karasfi, B. 2015. A hybrid control architecture for autonomous mobile robot navigation in unknown dynamic environment. In 2015 IEEE International Conference on Automation Science and Engineering (CASE) (pp. 1274-1281).

Nobile, C., 2012. Robots Vulnerable to Hacking. http://www.roboticsbusinessreview.com/article/robots_vulnerable_to_hacking/

Ollero, A., Heredia, G., 1995. Stability analysis of mobile robot path tracking. In Intelligent Robots and Systems 95. In Proceedings of the Human Robot Interaction and Cooperative Robots. IEEE/RSJ International Conference on,Vol. 3, pp. 461-466. DOI: 10.1109/IROS.1995.525925

Ollero, A., Mandow, A., Muñoz, V. F., & De Gabriel, J. G., 1994. Control architecture for mobile robot operation and navigation. Robotics and computer-integrated manufacturing, 11(4), 259-269.

Ollero, A., Arrue, B. C., Ferruz, J., Heredia, G., Cuesta, F., López-Pichaco, F., & Nogales, C., 1999. Control and perception components for autonomous vehicle guidance. Appliction to the ROMEO vehicles. Control Engineering Practice, 7(10), 1291-1299. DOI: 10.1016/S0967-0661(99)00091-X.

Park, J., Jeong, W., Lee, H. K., Won, J. 2013. An efficient path planning method for a cleaning robot based on ceiling vision. In 2013 IEEE International Conference on Consumer Electronics (ICCE).

Prieto, J., Ramos, O., Delgado, A., 2007. Diseño de un gene digital en FPGA y MATLAB con aplicaciones en robótica móvil. XIII Taller Iberchip IWS-2007, Lima, 14.

Sheppard, B., Thompson, T., (2014).Cyber Security for Robots: Scenarios. http://www.roboticsbusinessreview.com/article/cyber_security_for_robots_scenarios_for_2030.

Tehranipoor, M., Koushanfaar, F., 2010. "A Survey of Hardware Trojan Taxonomy and Detection". IEEE Design and Test of Computers, 27(1), 10-25. DOI: 10.1109/MDT.2010.