PRUEBAS DE ACEPTACIÓN Y SISTEMA

Evaluador de microcontroladores para misiones espaciales

Gonzalo Nahuel Vaca

Facultad de Ingeniería
Universidad de Buenos Aires
Argentina
21 de noviembre de 2021

1. Introducción

Este documento tiene el objetivo de planificar los ensayos de nivel de sistema y de nivel de aceptación para un módulo del proyecto "Evaluador de microcontroladores para misiones espaciales".

El módulo seleccionado es el "Validador de periférico SPI", quién deberá determinar si el integrado aún cuenta con un sistema SPI funcional.

El código del módulo es el siguiente:

```
bool validate_SPI()
{
    uint8_t spi_write[5] = {1, 2, 3, 4, 5};
    uint8_t spi_read[5] = {6, 7, 8, 9, 10};
    uint8_t status = NORMAL;
    SPI0_WriteRead(&spi_write[0], 5, &spi_read[0], 5);
    for(uint8_t i = 0; i < 5; ++i)
    {
        if(spi_write[i] != spi_read[i])
        {
            status = ERROR;
            break;
        }
    }
    return status;
}
```

2. Pruebas a nivel sistemas

Las pruebas de este nivel se realizan considerando un conocimiento pleno sobre el código fuente. Por este motivo se realizarán bajo el esquema de *Elementary Comparison Test*.

C1 & C2 & C3 & C4 & C5	NORMAL	ERROR
1.1.1.1.1	+	
1.1.1.1.0		+
1.1.1.0.1		+
1.1.0.1.1		+
1.0.1.1.1		+
0.1.1.1.1		+

Para realizar estas pruebas se necesitará realizar un *mock* de la función de escritura y lectura del periférico SPI.

3. Pruebas a nivel aceptación

Las pruebas de este nivel se realizan considerando al módulo como una caja negra. Por este motivo se realizarán bajo el esquema de *Classification-tree Method*. En la figura 1 se puede observar el diagrama con los dos casos de prueba. El caso "1" consiste en dejar conectado del *loopback* del periférico, mientras que el caso "2" se realiza desconectado la conexión entre la entrada y salida del módulo SPI.

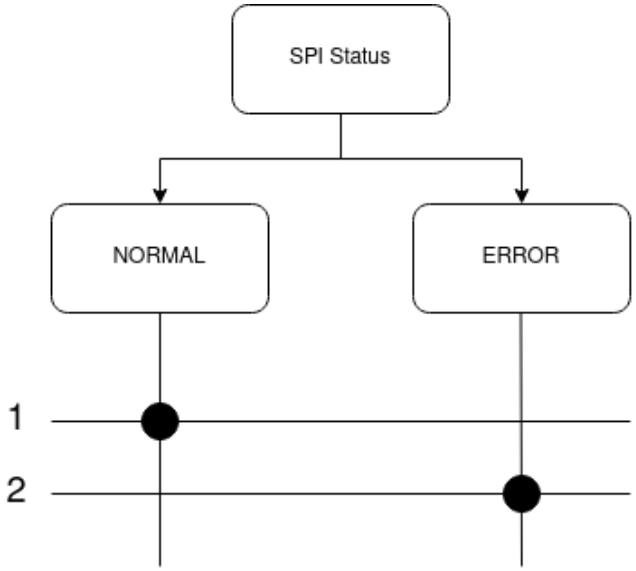

Figura 1: Diagrama *Classification-tree Method*.

La pruebas se realizarancon el siguiente script de *Python*:

```
from serial import Serial
import signal, os

def handler(signum, frame):
    print("")
    print("Terminal ended by user")
    exit()

def main():
    with Serial('/dev/ttyACM0', 9600, timeout=1) as s:
        while True:
            x = s.read()
            if x == b'R':
                f = ord(s.read())
                c = int.from_bytes(s.read(4), 'little')
                print("FLAGS:", f, "COUNTER:", c, end='\r')
```

```
if __name__ == "__main__":
    signal.signal(signal.SIGINT, handler)
    main()
```