**Pruebas unitarias individuales (para cada tipo de actuador y sensor) y de aplicación.**

Para las pruebas hemos empleado caja negra y caja gris.

En caso de que se detecten errores, se procedió a revisar el componente HW y SW relacionado (ver trazas) y se realizó debugging para la aplicación involucrada (SMA-LAMP, SMA-COMP o ambos) y se realizará la prueba de nuevo. Cuando se halla la solución se indica en este documento. Si la incidencia es suficientemente grande se indica en la sección de incidencias.

No fueron contadas las pruebas de Estructura Mecánica (ya que se han realizado por los de Industriales) ni del CA (salvo pruebas de interfaces, ya se diseñó por conocidos del profesor). Las pruebas entre SMA\_LAMP y SMA\_COMP que son comunes solo se definen en ambos lados por claridad a la hora de realizar las pruebas, centrándose en las diferencias entre ambas partes.

Para los I2C, si hubiera dado tiempo, habríamos comprobado que no hubiese problemas con las resistencias de pull-up de 4k7.

Acá quedan las trazas:

|  |  |  |
| --- | --- | --- |
| **Requisito** | **Componente HW** | **Componente SW** |
| ENC-10 | SMA-LAMP, ordenador Linux | SMA-LAMP, SMA-COMP |
| ENC-20 | Sensor Temperatura, Humedad, Ruido, CO2 y Luminosidad, SMA-LAMP, USART y ordenador Linux. | Sensor Temperatura, Humedad, Ruido, CO2 y Luminosidad; SMA-LAMP y SMA-COMP |
| ENC-30 | EEPROM/Flash | Memoria |
| AP-10 | SMA-LAMP (registros/mem volátil) | SMA-LAMP |
| AP-20 | Ordenador Linux/USART | SMA-COMP |
| AP-30 | Ordenador Linux/USART | SMA-COMP |
| MO-10 | CAD | Sensor Ruido |
| MO-20 | Timer | Sensor Ruido |
| MO-30 | Timer | Sensor Temperatura, Humedad, Ruido, CO2 y Luminosidad |
| MO-40 | I2C (C02 y Luminosidad), ADC (el resto) | Sensor Temperatura, Humedad, Ruido, CO2 y Luminosidad |
| AC-10 | I2C, USART | Memoria |
| AC-20 | PWM, USART | PWMVentilador |

Y acá el conjunto de pruebas (indicando aquellas que no pudimos realizar a tiempo):

|  |  |  |
| --- | --- | --- |
| Dispositivo | Prueba | Resultado (Pasa/No Pasa) |
| **Estructura mecánica**  (ya construido) | N/A | N/A |
| **Componente CA**  (ya construido) | (interfaces ya existentes, realizado por compañeros de Norberto) | N/A |
| Comprobación de malas soldaduras: | 1º Al soldar se comprueba que se puede conectar el componente a la pista (probar con multímetro que la zona más distante de la pista correcta sí conecta)  2º Al soldar no se comunican pistas que no deberían estar conectados (ver con multímetro que las pistas cercanas no se conectan con el componente). Esto incluye asegurarse que los tornillos no se cargan. | Pasa (No hay problemas).  Pasa (No hay problemas). |
| Comprobación de carga (PICKIT-3) | Se puede cargar un programa a la placa con éxito (en este caso un programa de prueba de usart empleado anteriormente) | Pasa. |
| Sensor de luminosidad VEML7700 | Comprobar si sube la luminosidad al incrementar valor (poner lámpara o taparlo con un dedo) y que no interfiera con los resultados del sensor de CO2. Esto se hace con una versión simplificada del programa final con solo los I2C, primero solo para este sensor y luego para ambos sensores. | -----NO DIO TIEMPO----- |
| Sensor de CO2 iAQ-Core | Comprobar si cambian valores de C02 en diferentes lugares de la universidad (o mejor, al echar vaho), y que el sensor de luminosidad no interfiere. Esto se hace con una versión simplificada del programa final con solo los I2C, primero solo para este sensor y luego para ambos sensores. | ----- NO DIO TIEMPO ----- |
| Sensor de humedad relativa HIH4000 | Comprobar si el valor de la humedad varia (al echar vaho). Utilizaremos una versión del programa de adc para ello. | El sensor funciona. |
| Sensor de temperatura LM35-LP | Comprobar si la temperatura incrementa adecuadamente (poner mano caliente o algún objeto frío). Utilizaremos una versión del programa de adc para ello. | El sensor funciona. |
| 1. Nivel de ruido. | Comprobamos la señal analógica de ruido (haremos ruidos con objetos o la voz para probarlo) y que no interfiera con la de humedad relativa ni temperatura. Utilizaremos una versión del programa de adc para ello y luego una versión del programa final con solo lo de ADC, necesita de UART. | Adc: Pasa.  Nota: posteriormente nos dejó de funcionar por motivos desconocidos, tras analizarlo por varias horas y no encontrar la causa SW se postergó a otras pruebas. |
| 1. SCL. | El bus I2C conecta el sensor de humedad y el de CO2 y funciona a 100kHz. | ----- NO DIO TIEMPO ----- |
| 1. SDA. | El bus I2C conecta el sensor de humedad y el de CO2 y funciona a 100kHz | ----- NO DIO TIEMPO ----- |
| 1. Humedad relativa. | Comprobamos la señal analógica y que no interfiera con temperatura ni ruido. Utilizaremos una versión del programa de adc para ello (con) y luego una versión del programa final con solo lo de ADC. | El sistema lee, pero los valores no se relacionan con nada.  Nota: posteriormente nos dejó de funcionar por motivos desconocidos, tras analizarlo por varias horas y no encontrar la causa SW se postergó a otras pruebas. |
| 1. Temperatura | Comprobamos la señal analógica y que no interfiera con ruido ni humedad. Utilizaremos una versión del programa de adc para ello y luego una versión del programa final con solo lo de ADC. | El sistema lee, pero los valores no se relacionan con nada.  Nota: posteriormente nos dejó de funcionar por motivos desconocidos, tras analizarlo por varias horas y no encontrar la causa SW se postergó a otras pruebas. |
| 1. PWM ventilador | Comprobar si aumenta o disminuye la velocidad del ventilador (CCP1). Utilizaremos primero el programa del pwm de prácticas y luego una versión del programa final con el pwm. | Adc: Pasa\*  \*Evento: hay que darle un pequeño golpecito para que arranque a veces -> Resistencia estática muy grande o polvo en el motor.  El profesor dijo que eso no es de nuestra competencia, y además luego ya se resolvió. |
| 1. SCK. (Línea de control de un bus SPI. Emisor luz lámpara)   Sugerencia, como el pin de I2C y SPI son el mismo, simulamos SPI por SW. | Comprobar nivel de luminosidad de la lampara al recibir la señal del bus SPI. | Transmitir un código de ejemplo según el datasheet con máxima iluminancia (31) no encendía ningún LED. Ejecutamos debug para comprobar por qué, descubrimos un error en el for que no hacía pasar ni una sola instrucción de estas, tras resolverlo ya sí se podían encender pero por algún motivo solo el primer led de la tira y débilmente. Procedíamos a comprobar por debug de nuevo cuando se acabó el tiempo. -- NO SE COMPLETÓ POR FALTA DE TIEMPO -- |
| 1. SDO. (Línea de salida de datos hacia un bus SPI. Emisor luz lámpara).   Sugerencia, como el pin de I2C y SPI son el mismo, simulamos SPI por SW. | Comprobación niveles de luz por la lampara. | Ídem del anterior -- NO SE COMPLETÓ POR FALTA DE TIEMPO -- |
| USART | Comprobar que se pueden enviar y recibir datos (se usa el mismo programa de la usart que el de las prácticas) | Pasa. |
| **Componente SMA** | | |
| SMA-LAMP | Comprobamos que es capaz de alterar la velocidad del ventilador mediante la señal PWM, que varían los colores de la tira de leds (bus SPI) y que guarda los estados en la memoria (EEPROM/Flash) (según las trazas descritas anteriormente) y que es capaz de enviar datos por la USART | Pasa. |
| ENC-10: SMA-LAMP se arranca correctamente | Hubo problemas con SPI (ver explicación arriba), aparte de eso todos los otros sistemas se iniciaban correctamente. |
| ENC-20: se envían los datos de los sensores disponibles y los no listos a SMA-COMP | ----- NO DIO TIEMPO ----- |
| ENC-30: al modificar la configuración de LEDes y ventilador, la siguiente vez que se arranca, sale esa misma configuración y no la de por defecto | Pasa. |
| AP-10: apagar SMA-COMP directamente no debe crashear SMA-LAMP | ----- NO DIO TIEMPO ----- |
| AP-20: SMA-LAMP envía un pong (señal “B”) a los ping (“b”) de SMA\_COMP por USART | ----- NO DIO TIEMPO ----- |
| AP-30: se puede dar la orden de apagar SMA-LAMP por USART (señal “e”), debe salir de su bucle. | ----- NO DIO TIEMPO ----- |
| MO-10: SMA-LAMP traduce todas las señales de ruido a las categorías Bajo [0, 400], intermedio (400, 900] y alto (900, infinito) | ----- NO DIO TIEMPO A VERIFICAR EN EL MODELO REAL ----- |
| MO-20: el ruido debe monitorizarse cada 10 ms sin interferencias de los otros ADC (debe cumplirse el sample and hold) + solo debe actualizarse cada segundo dando la categoría de ruido más alta por segundo | La simulación indicaba que se ejecutaría en ese período de tiempo. No dio tiempo a verificar en el modelo real. |
| MO-30: humedad, temperatura, CO2 y luminosidad deben leerse cada 5 segundos y mandar cada 5 segundos valores actualizados. | La simulación indicaba que se ejecutaría en ese período de tiempo. No dio tiempo a verificar en el modelo real. |
| MO-40: en el printf el SMA-LAMP envía la cadena con los datos indicados en la categorías a/m/b (ruido), ºC (temperatura), % (humedad relativa), ppm (para CO2) y lx (para luminosidad) | Se comprobó que efectivamente puede enviarlos por minicom, pero no dio tiempo a comprobar en SMA-COMP puesto que aún no se había instalado en el computador.  ----- NO DIO TIEMPO A VERIFICAR EN EL MODELO REAL ----- |
| AC-10: al recibir por USART la señal de cambio de LED (señal “d”) los siguientes 4 caracteres recibidos se entenderán como el nivel de R, G, B y luminosidad, que se enviarán a los LEDes y por ENC-30, a la memoria. | ----- NO DIO TIEMPO ----- |
| AC-20: al recibir por USART la señal de cambio de PWM (señal “c”) el siguiente carácter recibido se entenderá como el porcentaje del duty cycle del ventilador, que se enviarán al CCP1 y por ENC-30, a la memoria. | ----- NO DIO TIEMPO ----- |
| SMA-COMP | Comprobamos que la interfaz programada funciona correctamente, realizando las operaciones oportunas, siendo capaz de leer los datos de los sensores, codificar los estados de los leds y de la señal PWM y apague la lampara cuando sea necesario (según las trazas descritas anteriormente). | La interfaz programada funciona, pero no dio tiempo a montar en el sistema de pruebas. |
| ENC-20: se reciben los datos de los sensores disponibles y los no listos de la SMA-LAMP y se tratan para ser legibles con el formato empleado (p.ej. en ventanas) | ----- NO DIO TIEMPO ----- |
| AP-20: Apagar SMA-LAMP supone que SMA-COMP de un mensaje de que se ha perdido la conexión y deberá cerrarse SMA-COMP. Esto se hace por comunicación por USART donde SMA\_COMP manda a la lámpara un ping periódico por USART (señal “b”), y si no responde en un tiempo determinado se apaga. Comprobar que no ocurren falsos positivos. | ----- NO DIO TIEMPO ----- |
| AP-30: SMA-COMP debe tener un botón para mandar una orden de apagado al SMA-LAMP que envíe por la USART el comando “e”) | Efectivamente tiene el botón, pero no se puedo probar su funcionamiento por las causas mencionadas anteriormente. |
| MO-10: las señales interpretables como “bajo/medio/alto” 1, 2, 3 de SMA-LAMP deben traducirse a “alto”, “medio”, “bajo” para el usuario final | Esta prueba se alteró para explicarse a nivel del manual de usuario (ver sección “MANUAL DE USUARIO”) |
| MO-20: el SMA-COMP refleja las actualizaciones de ruido cada segundo. | ----- NO DIO TIEMPO ----- |
| MO-30: humedad, temperatura, CO2 y luminosidad deben recibirse de la USART y actualizarse por pantalla cada 5 segundos | ----- NO DIO TIEMPO ----- |
| MO-40: los datos recibidos desde SMA-LAMP debo recibirlos en categorías a/m/b (ruido), ºC (temperatura), % (humedad relativa), ppm (para CO2) y lx (para luminosidad) | Comprobado, el SMA-LAMP al final envía los datos directamente sin que SMA-COMP necesite traducirlos a unidades específicas. |
| AC-10: la aplicación del ordenador debe tener un botón que permita enviar una cadena del tipo “d0000” dónde d es el indicador del comando LED y 0000 son 4 caracteres que indican el R, G, B y luminosidad, por supuesto éstos se hacen legible al usuario y a prueba de errores (no menos de 0, no más de 255 (o 31 para luminosidad)). | Efectivamente tiene el botón, pero no se puedo probar su funcionamiento por las causas mencionadas anteriormente. |
| AC-20: la aplicación del ordenador debe tener un botón que permita enviar una cadena del tipo “c0” dónde c es el indicador del comando PWM y 0 es el carácter que indica porcentaje del duty cycle, por supuesto legible al usuario y a prueba de errores (no menos de 0, no más de 100). | Efectivamente tiene el botón, pero no se puedo probar su funcionamiento por las causas mencionadas anteriormente. |

Ejemplo de caso de prueba modelo: encendemos el dispositivo SMA-LAMP y vemos que el ventilador está apagado y la luz es blanca, leemos sensores por SMA-COMP, tras unos segundos escribimos PWM, tras eso leemos de nuevo y comprobamos PWM y LEDes. Unos segundos después escribimos PWM, leemos datos y cerramos el SMA-COMP. Volvemos a abrir SMA-COMP y vemos que sigue funcionando. Ahora ordenamos apagar la SMA-LAMP por comando de la SMA-COMP y debemos ver como SMA-COMP nos dice que ya no recibe ping de la LAMP y se para. Ahora iniciamos de nuevo y vemos que el SMA-LAMP debe tener la configuración de PWM y LED pre-apagado.

Listado de pruebas que han tenido éxito (o sea, detectan defectos) -> Y solución

Dev/ttyUSB0

2.3.2. Interfaz computador personal  
Para ganar en flexibilidad se ha decidido que el interfaz de monitorización y ajuste del LMDE, que se ofrezca al usuario final, se despliegue en un computador personal ubicado en la misma mesa de estudio que la lámpara.

El SMA deberá ofrecer un interfaz UART (no bluetooth) para conectarse con el computador personal el cual ofrecerá un interfaz USB. Para ello se utilizará el conversor TTL-232R-RPi (FTDI).

2.3.3. Interfaz de usuario  
El interfaz de usuario debe desarrollarse para una máquina LINUX. A nivel de prototipo puede aceptarse un interfaz que funcione en modo terminal de línea (el profesor nos lo puede dar o uno nuestro). El interfaz definitivo debe ser un interfaz de ventanas, pero el hacerlo bonito se hace al final.

El sistema debe mostrar en pantalla el estado de las variables ambientales

* nivel de ruido (usar práctica 3)
* nivel de CO2
* humedad relativa (usar práctica 3)
* temperatura
* luminosidad

También debe permitir cambiar el color y la intensidad de la luz emitida, y la velocidad del ventilador (usar Práctica 4 para señal PWM).  
Detalles más concretos se indican en el apartado de requisitos específicos.

2.4. Funciones del sistema

**Monitorización** El sistema debe monitorizar una serie de parámetros ambientales que tienen interés para determinar si el ambiente de estudio es adecuado. Fundamentalmente son:

* Nivel de ruido ambiente
* Nivel de CO2
* Nivel de temperatura
* Nivel de humedad relativa
* Nivel de luminosidad.

**Ajuste del comportamiento**: El sistema debe permitir ajustar el color de luz emitida de la lámpara, la iluminancia de la luz emitida y la velocidad de rotación de un ventilador sujeto al soporte de la lámpara.

2.5. Restricciones  
1. Por la configuración seleccionada para el interfaz de usuario, el componente SMA se dividir ́a en dos subcomponentes.  
SMA-LAMP Estar ́a constituido por el hardware y software ubicado en la lámpara destinado a recibir la información de los sensores, tratar dicha información y enviarla al computador de sobremesa. Del mismo modo atender a los comandos que lleguen desde el computador de sobremesa para actualizar el comportamiento de los leds as ́ı como el del ventilador (IRSE VIENDO COMO LEER DE USART, ESCRITURA EN UN TERMINAL, LECTURA EN OTRO, AJUSTA COMO EN SO PARA QUE AJUSTE EN EL FICHERO).  
SMA-COMP Será un componente software desplegado en el computador de sobremesa, que ofrecerá el interfaz de la lámpara con el usuario.  
2. Por condicionantes externos, el microcontrolador a utilizar en el componente SMA-LAMP debe ser el PIC16F886.  
3. Las dimensiones de la placa del componente SMA no pueden ser superiores a 6 cm x 6 cm.  
4. Es necesario reservar espacio en la placa para cuatro tornillos (o nylon) de sujeción, de 4mm de diámetro (+ 1 de salvaguarda en caso de poner tornillos para evitar cortocircuitos), cerca de las esquinas de la misma. El margen necesario debe ser de 5mm de diámetro para poder alojar la cabeza de los tornillos.  
3. Requisitos específicos  
Lo mínimo es hacer las trazas. Trazar componentes del diseño contra especificación.

Al profesor le gustan ciclos de vida en espiral (suele poner el atributo de ciclo de desarrollo en los requisitos; primero los más claros e indispensables y luego los menos claros y opcionales).

No cambies el número de los requisitos.

Pon una lista de requisitos vinculados a otros requisitos.

**Encendido del sistema**: Este caso de actividad contempla las acciones que deben realizarse al encender el sistema.  
\*ENC-10 Por defecto, se arrancará en primer lugar SMA-LAMP, alimentando la lámpara, para a continuación arrancar SMA-COMP, en el computador de sobremesa.  
\*ENC-20 En el momento de arrancar se ofrecerá la información de los sensores disponibles y se  
indicar ́a el estado de aquellos que requieran un proceso arranque más prolongado. NOTA: el sensor de CO2 tarda mucho en arrancar, sus datos iniciales son basura, no nuestro error.  
\*ENC-30 (mejor en el primer ciclo de desarrollo, pero el profe lo admite en el segundo haciendo que siempre arranque con luz blanca y ventilador apagado) La lámpara debe arrancar respetando la última configuración utilizada en lo referente a leds y ventiladores. En el caso de ser la primera vez que se enciende se optará por una opción intermedia de luz blanca y ventilador apagado. NOTA: Requiere uso de memoria EEPROM, pon una marca que indique si existe la información y arranca desde EEPROM.

**Apagado del sistema**: Este caso de actividad contempla las acciones que deben realizarse ante los diferentes escenarios de apagado del sistema.  
\*AP-10 Si se apaga el sistema SMA-COMP, el sistema SMA-LAMP debe seguir funcionando con normalidad según la ́ultima configuración establecida.

\* AP-20 Si se apaga SMA-LAMP, SMA-COMP debe indicar que SMA-LAMP se ha desconectado. En este caso, SMA-COMP no debe volver a operar hasta después de que la aplicación se haya reinicializado. NOTA: suele exigir un HeartBeat, comunica algo periódicamente -> Puede ser simplemente datos de sensores sin recibir.  
\* AP-30 SMA-COMP debe ofrecer alguna alternativa para apagar la aplicación. NOTA: p.ej al cerrar la ventana  
**Monitorización de variables ambientales** Este caso de actividad contempla las acciones que deben realizarse en relación con la monitorización de variables ambientales.

MO-10 El nivel de ruido es una señal en voltaje. Deben considerarse las siguientes categorías para el ruido:  
- Ruido bajo La lectura CAD está en el intervalo [0,400].  
- Ruido intermedio La lectura del CAD entre [401, 900].  
- Ruido alto La lectura del CAD encima de 900.

MO-20 El nivel de ruido debe monitorizarse cada 10 ms, pero en el interfaz humano solo debe mostrarse la categoría de ruido m ́as alta alcanzada cada segundo (situación laxa).

MO-30 El nivel de CO2, humedad relativa, temperatura y la iluminancia, deben ser muestreados cada 5 segundos y actualizarse en pantalla con el mismo periodo.

MO-40 La unidades a utilizar para las magnitudes muestreadas (salvo el ruido, que ya ha sido categorizado) deben ser:  
- PPM (partes por millón) para el nivel de CO2 (Por I2C ya nos los da).  
- % (tanto por ciento) para la humedad relativa (interpreta curvas fabricante).  
- ºC (grados centígrados) para la temperatura.  
- lx (Lux) para la iluminancia (Por I2C ya nos los da).

**Ajuste de actuadores** Este caso de actividad contempla las acciones a tomar relacionados con el comportamiento de los LEDs y el ventilador.

AC-10 SMA-COMP debe ofrecer al usuario la posibilidad de cambiar el comportamiento de los LEDs. Para ellos debe suministrar la posibilidad de especificar el color con un código RGB, en el que se permita un valor entre 0 y 255 para cada color. Al mismo tiempo debe permitir codificar el nivel de brillo con un código que pueda tomar valores entre 0 y 31 (eso es decisión del fabricante).  
AC-20 SMA-COMP debe ofrecer al usuario la posibilidad de cambiar la velocidad del ventilador. Dicha velocidad se codifica en tanto por ciento con valores posibles entre 0 % y 100 %