**Práctica 1 (Oficial)**

**Fecha de entrega: 17 de septiembre de 2015 (antes de las 7:00 pm)**

**Modo de entrega: Demostrativa en clase y enviada por mail.**

**Tipo de entrega: En equipos**

**Propósito: El alumno se adecue al código y diseño hecho por otra persona además resolver con satisfacción los problemas de recursos limitados del micro controlador.**

**Descripción:**

**-Utilizar 4 archivos (practica1.c, practica1.h, practica1\_int.h, main.c)**

**Reglas de los archivos:**

**\* Practica1\_int.h**

**\*No puede ser modificado**

**\* Pratica1.h**

**\*Pueden Agregar solo macros para evitar números mágicos o largas secuencias de caracteres.**

**\*Pueden declarar nuevos tipos de datos (ojo, solo tipos de datos, no variables/constantes/registros)**

**\*No pueden declarar variables/constantes/registros ni exportar/importar funciones/variables/constantes/files.**

**\*Practica1.c**

**\*No pueden modificar prototipos**

**\*Solo pueden incluir practica1\_int.h y practica1.h**

**\*No pueden agregar nuevas funciones**

**\*No pueden declarar nuevas variables/constantes/registros globales**

**\*El cuerpo de las funciones que se exportan en practica1\_int.h no puede ser modificado.**

**\*Pueden o no pueden utilizar el resto de funciones**

**\*Pueden agregar las variables locales que gusten, máximo 5 bytes en todo el archivo (por cada byte de mas, se restará 1 punto de la calificación total) aparte de las variables que son argumentos de función.**

**\*Pueden cambiar el nombre a las estructuras y enums que se llamen auxiliares. Más no pueden modificar tamaños.**

**El cuerpo de las funciones privadas que tiene completa la descripción no puede ser modificado**

**\* En algunas funciones sin descripción pero con un comentario de un número de líneas, el alumno deberá de implementar el código con exacto número líneas de las que pide el comentario, así como llenar la descripción de la función.**

**En algunas funciones el cuerpo de la función está vacío, pero la descripción está completa, se deberá de llenar la función y esta debe de hacer únicamente lo que dice la descripción.**

**\* main.c**

**\*Este archivo deberá de tener todo lo necesario para que los leds cambien de color.**

**\*Solo podrá tener un include aparte de los archivos de default de code warrior, y este debe de ser el practica1\_int.h**

**\* No deberá de tener código auxiliar que ayude al archivo practica1.c en su funcionalidad.**

**Funcionalidad:**

* **Se tendrá una variable global llamada ruw\_AP0\_1 que cambiará su valor cada segundo y donde su máximo valor es 4095.**
* **Hay 5 rangos que se deberán de tener de acuerdo al valor de ruw\_AP01:**
  + **Rango 1: X>0 y X<928**
  + **Rango 2: X>=928 y X<1258**
  + **Rango 3: X>=1258 y X<=2518**
  + **Rango 4: X>2518 y X<=3708**
  + **Rango 5: X>3078 y X<=4095**
* **Se tendrá un contador de 10 segundos llamado rub\_operative\_time**
* **Se tendrá un contador de 5 segundos llamado rub\_demature\_time**
* **Para hacer valido el nuevo valor de ruw\_AP0\_1, primero deberá de llegar en el mismo rango por un mínimo de 2 segundos.**
* **Si se encuentra en Rango 5, el LED deberá permanecer apagado**
* **Si se encuentra en Rango 1, el LED tendrá color azul.**
* **Si se encuentra en Rango 2, el LED tendrá color azul.**
* **Si se encuentra en Rango 3, el LED tendrá color blanco.**
* **Si se encuentra en Rango 4, el LED tendrá color verde.**
* **Consideraciones especiales:**
  + **Cuando haya una transición del rango 1 o 2 al 4 o 5, el contador rub\_operative\_time empezará a contar.**
  + **Si hay una transición del rango 1 o 2 al rango 3 el LED cambiará a color blanco.**
  + **Si estando en rango 4 el contador rub\_operative\_time expira, el LED deberá de togglear (pueden elegir cualquier color).**
  + **Si estando en el rango 5 el contador rub\_operative\_time expira, el LED deberá de permanecer apagado.**
  + **Si estando en el rango 5 hay una transición al rango 4. Se tomará en cuenta el estado de rub\_operative\_time:**
    - **Rub\_operative\_time expire: LED toggle**
    - **Rub\_operative\_time no ha expirado: LED color verde.**
  + **Si hay una transición del rango 4 o 5 al rango 3. El contador rub\_demature\_time empezará a contar y el led dependerá el contador rub\_operative\_time:**
    - **Rub\_operative\_time expire: LED toggle**
    - **Rub\_operative\_time no ha expirado: LED color blanco.**
  + **Si hay una transición del rango 3, 4 o 5 al rango 1 o 2. Ambos contadores deberán de detenerse.**
  + **Si estando en el rango 3 el contador rub\_operative\_time expira el LED debe permanecer en color blanco**
  + **Si estando en el rango 3 el contador rub\_demature\_time expira el LED cambiará a color blanco.**
  + **Si estando en el rango 3 el contador rub\_demature\_time expira ambos contadores son detenidos.**
  + **Si hay una transición del rango 3 al rango 4 o 5 y el contador rub\_demature\_time está detenido. El contador rub\_operative\_time empezará a contar.**
  + **Si hay una transición del rango 3 al rango 4 y el contador rub\_demature\_time NO está detenido:**
    - **Led toggle si rub\_operative\_time expiró.**
    - **Led verde si rub\_operative\_time no ha expirado.**

**La secuencia de números será dada por el maestro el día de la revisión de la práctica, el alumno deberá de compilar con la nueva secuencia y mostrar el kinetics funcionando.**

**El formato será el siguiente:**

**10 (15) 900 (5) . . . . . dónde: El numero fuera del paréntesis es la que tomara el valor la variable ruw\_AP0\_1 y el numero dentro del paréntesis es la duración en segundos en el que permanecerá.**

**El total de números y segundos no será mayor a 50.**