PRÁCTICA Nro. 1

CUESTIONARIO

Ejercicio No 1: Repaso de lenguaje C

a) Investigue sobre los distintos tipos de variables en C y sus modificadores: static, volatile, register, const. ¿Cuáles son los tipos de datos por defecto en el compilador que utilizaremos?

b) Investigue sobre las sentencias del pre-procesador de C, entre ellas: #include, #define, #ifdef y typedef.

c) ¿Qué es una constante de carácter? ¿qué es una cadena de caracteres?

d) ¿Cuál es la diferencia entre una variable local y una global? ¿Por qué utilizaría una u otra?

e) Describa todos los operadores lógicos de C. ¿cúal es la diferencia entre los operadores && y &, || y |? ¿Qué es una máscara de bits?

f) ¿Qué es un prototipo de función en C? ¿Cuáles son las alternativas para pasar argumentos a una función? ¿Cómo se retorna un valor desde una función?

g) Repase el concepto de punteros y arreglos. Explique con ejemplos la relación entre ambos.

h) Investigue sobre los tipos de variables struct y union (estructuras y uniones) en C. De un ejemplo de cada caso.

i) ¿Qué son los campos de bit de una estructura?

Ejercicio No 2: Familia de microntroladores AVR

a) Investigue sobre los diferentes modelos de la familia Atmega AVR. Tabule los periféricos, la cantidad de RAM y de FLASH que poseen las distintas versiones. En particular detalle las características de los modelos Atmega328P y Atmega2560 utilizados en las plataformas Open-Source Arduino UNO y MEGA.

b) Detalle las características de la CPU AVR, incluyendo: Arquitectura de la CPU (Realice un diagrama en bloques), modelo de programación, modos de direccionamiento y resumen del conjunto de instrucciones.

c) Puertos de Entrada/Salida: Describa qué registros se utilizan para controlar los puertos de entrada y salida. Realice un diagrama en bloques de un terminal de entrada y salida y explique su funcionamiento.

d) Investigue cual es la capacidad de corriente que puede manejar un terminal individual configurado como salida. ¿Depende del estado lógico? ¿cuál es la capacidad de corriente que puede manejar el microcontrolador con todos los puertos funcionando simultáneamente?

e) Indique cuales son las dos posibilidades de conexión de un LED a un terminal de entrada y salida. Calcule la resistencia serie para que la corriente máxima por cada LED sea de 5mA. Muestre como configurar el terminal como salida y como modificar su estado lógico.

f) Se desea conectar un pulsador a una entrada digital del MCU. Investigue los posibles esquemas de conectar un pulsador y determine el algoritmo más adecuado para detectar en cada caso cuando el pulsador se presiona y se suelta. ¿Qué es una resistencia de pull-up? ¿y de pull-down? ¿Importa su valor? Investigue sobre cómo utilizar los pull-up internos del MCU.

EJERCICIOS DE REPASO (*):

a) Dado un número N de 8 bits sin signo, realice una función en C que devuelva la suma de los números consecutivos de 1 hasta N. Analice el tipo de variable que retorna más conveniente. Ensaye la función en un programa sobre el compilador estandar.

b) Implemente una función que reciba como parámetros un arreglo (vector) y su tamaño en bytes. La función debe reordenar el vector de manera inversa, es decir: el primer valor pasará a ser el último y así siguiendo. Los parámetros de entrada deben pasarse por referencia y por valor respectivamente.

c) Realice una función de C para convertir un número binario de 8 bits sin signo en un conjunto de caracteres ASCII que correspondan a los dígitos de dicho número. El número a convertir se debe pasar como parámetro y los dígitos ASCII resultantes deberán retornarse en una cadena de caracteres. Por ejemplo, si el dato de entrada es 124 (0x7C) los 3 dígitos ASCII serán '1','2' y '4'. Utilice la interfaz de entrada estándar del compilador para para pasarles los argumentos a la función.

d) Realice una función en C, que reciba como parámetro un número N de 8 bits y lo envíe de manera serie por la salida estándar a razón de un bit cada 1s comenzando por el menos significativo. Utilice las bibliotecas estándar del compilador GNU para controlar el tiempo de ejecución.

e) Uso del compilador Microchip (**). Explique qué hacen las siguientes sentencias de C: DDRC=0x0F; PORTC=0x0C;

f) Explique qué hace la siguiente sentencia de C: PORTC |= (1<< PORTC0) | (1<< PORTC2) | (1<< PORTC3); ¿Cuál es la diferencia con la última del ejercicio anterior. ¿Qué representa PORTCx? ¿Dónde y cómo está definido?

g) Escriba ahora una única sentencia para establecer el nivel lógico de PORTC0, PORTC2 y PORTC3 a nivel bajo sin afectar el estado lógico del resto de los bits del puerto.

EJERCICIO DE SIMULACIÓN:

El objetivo principal del siguiente ejercicio es abordar la programación en C de dispositivos AVR, comenzando por programas simples que utilizan los puertos de entrada-salida, y el uso del conjunto de herramientas de desarrollo para abarcar el proceso completo de edición, compilación, simulación del dispositivo, simulación de la aplicación o la placa de entrenamiento y finalmente la programación del dispositivo real.

a) Analice el programa ejemplo de la Fig.1 y comprenda su funcionamiento línea a línea.

b) Explique cómo funciona la función _delay_ms() de la bibliotecas de AVR Libc. *AVR Libc Reference Manual*

c) Cree un proyecto nuevo en Microchip Studio 7 y copie el programa del ejemplo en el editor, guarde el archivo ".c" y compile.

d) Analice el resultado de la compilación: errores, advertencias, memoria de datos y de programa utilizada.

e) Analice las dependencias externas: examine y comprenda la función que cumple la línea #include<avr/io.h>.

f) Simule el programa paso por paso con el simulador integrado Microchip Studio 7. Examine las ventanas de registros del MCU, de ciclos de reloj consumidos y el comportamiento de los puertos I/O utilizados.

g) Realice en Proteus ISIS, un circuito que incluya el MCU, ocho pulsadores y ocho leds para simular el programa desarrollado. Simule y verifique el comportamiento del mismo. Utilice el archivo ".elf" para depurar el programa paso a paso.

EJERCICIO PARA ENTREGAR

Se especificará en un documento separado próximamente.

NOTAS:

(*) Para repasar el lenguaje C y sus aplicaciones puede utilizar la herramienta on-line gratuita https://www.onlinegdb.com/online_c_compiler donde podrá editar, compilar y ejecutar programas en lenguaje C.

(**)Para programar el microcontrolador ATMEGA328p en lenguaje C utilizaremos las herramientas de Microchip disponibles en https://www.microchip.com/en-us/tools-resources/develop/microchip-studio

```c
/* Inclusión de bibliotecas de código */
#include <avr/io.h> // Registros del microcontrolador
#define F_CPU 16000000UL // Defino la frecuencia de oscilador en 8MHz
#include <util/delay.h> // Retardos por software

/* Función main */
int main (void)
{
	/* Setup */
	DDRD = 0xFF; // Configuro Puerto D como salida

	DDRC &= ~(1<<PORTC0); // Configuro bit0 puerto C como entrada
	PORTC |= (1<<PORTC0); // Habilito Pull-Up en bit0 puerto C

	/* Loop */
	while(1)
	{
		if (PINC & (1<<PINC0))
		{
			PORTD = 0b10101010; // Escribo Port D con patrón de bits
			_delay_ms(100); // Delay de 100 ms
			PORTD = 0x00; // Escribo Port D con todos 0
			_delay_ms(100); // Delay de 100 ms
		}
		else
		{
			PORTD = 0b01010101; // Escribo Port D con otro patrón de bits
			_delay_ms(100); // Delay de 100 ms
			PORTD = 0x00; // Escribo Port D con todos 0
			_delay_ms(100); // Delay de 100 ms
		}
	}
	/* Punto de finalización del programa (NO se debe llegar a este lugar) */
	return 0;
}
```

Fig. 1 - Programa Ejemplo