Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ingeniería Eléctrica
SD20A Seminario de Diseño

Guía Experiencia Nº3
Microcontroladores

Profesor: Javier Ruiz del Solar

Ayudantes: Mauricio Correa
Carlos Fernández
Isao Parra

1. Introducción.

Hoy en día, hacia donde se mire se encuentran artefactos eléctricos, pero que a diferencia de las antiguas lavadoras o refrigeradores, pueden elegir que hacer dependiendo de su uso. La gente frente a esto dice: “ahh lo hace el computador que tiene dentro, que moderno ehh” pero lo que nos aqueja es qué es este computador interno, que ahora uno puede encontrar en la mayoría de los artefactos, incluso en un excusado. La respuesta a esto es el Microcontrolador, que es básicamente un Circuito integrado que en su interior incluye una CPU, Memoria y Puertos de I/O, es decir es un computador simple todo incluido en un integrado. Esa es su principal característica la de funcionar con un mínimo de cosas extras externas.

Las funciones que puede hacer un Microcontrolador (uC) son tan diversas que hace que no permite que exista un solo tipo, por lo que existen en diversos sabores, lo que implica que cada modelo incluya diferentes tipos de Entradas, Salidas, Características especiales, más o menos Memoria o Arquitecturas diferentes.

Durante el Curso la marca de uC que usaremos serán los de Microchip, y usaremos uno que incluirá características suficientes para la mayoría de los proyectos: “PIC16F873A”.

1.1. Descripción.

Básicamente para el entendimiento, es una plataforma donde uno puede crear programas secuénciales y no multiprocesos, donde se pueden realizar cálculos a variables que se almacenan en la RAM y obtener o enviar datos a los PUERTO que son la manera de comunicarse al exterior del uC. Para poder escribirse el programa hay que hacerlo en un lenguaje de bajo nivel que se llama assembler el cual es directamente las instrucciones que le llegan al procesador. De esta manera de programar puede ser muy engorrosa y difícil de entender, además de ser totalmente específica ya que para cada uC se necesita programar con diferentes valores. Para evitar esto se utilizará un lenguaje de más alto nivel el cual se llama picC que es un lenguaje C orientado a PICs. La ventaja de esto es que es mas claro ver lo que se quiere hacer, y es mas parecido a los lenguajes aprendidos anteriormente, como Basic, C o java. Salvo que con lo reducido del sistema hay que olvidarse de interfases graficas o cosas por el estilo, por lo que hay que escribir el código lo mas conciente de lo que se hace, y por lo limitado de memoria, no crear variables para cualquier cosa.

El PICs tienen 3 tipos de memoria: RAM, Memoria de Programa y EEPROM:
RAM en ella se encuentra las variables, esta es la memoria que se va modificando a medida que se ejecuta un programa. Esta memoria se pierde cada vez que se reinicia o apaga el PIC.

Memoria de Programa aquí es donde se graba el programa que uno ejecuta, esta se puede grabar y borrar muchas veces, y se hace mediante un grabador o programador. Esta memoria no se borra al reiniciar o apagar, en caso de requerirse (aquí es prohibitivo) se puede bloquear para no se pueda borrar de nuevo.
EEPROM es una memoria que sirve para almacenar datos durante la ejecución del programa en si, y que al apagar o reiniciar no se pierda. Típicamente tiene un limite de ciclos de borrado y grabado, así que al programar hay que cuidar de no sobre exigirlo. Ya que si se hace muy seguido se le acorta la vida útil, típicamente de 1 millón de ciclos.

Los PUERTOS son la representación interna a un grupo de patas del integrados, estas pueden se de Entrada o Salida, lo que permite enviar datos desde el uC hacia fuera por ejemplo para encender un LED, o desde afuera hacia al uC por ejemplo un botón para activar cierta rutina. Típicamente existen dos puertos, cada uno de 8bits, el A y B. a cada pata se enumeran desde 0 a 7.

Existen otros PINs del PIC que son “esenciales” para su funcionamiento.

Vdd: es la alimentación del PIC, esta debe ser +5V, hay que ser muy cuidadoso con esto, nunca darle mas.
Vss: es la tierra del PIC, o sea 0V. Al igual que el Vdd si es que hay más de un PIN de Vss se debe conectar.
MCLR: este es el PIN reset del PIC, en caso de estar configurado, para estar funcionando tiene que estar alimentado a Vdd, y para resetear debe ser bajado a Vss. Para esto hay que usar una resistencia, un condensador y un interruptor de manera de no provocar un Corto Circuito.

Oscilador: si es el PIC no tiene oscilador interno se requiere un oscilador externo, típicamente un cristal. Su función es la generar la señal que crea los ciclos para operar el PIC, suelen ser dos PIN y se requiere para el cristal dos condensadores de 15 a 30 pF conectados a tierra y el cristal entre ambos PINs. Antes de cualquier conexión de este tipo, revisar datasheet del uC.

El programa que se utilizará para compilar el programa es el PIC C COMPILER en uno puede escribir el programa y compilarlo a la vez, este se encuentra en la sección de material de la pagina de Mecatrónica. Una buena herramienta de autoaprendizaje es la de revisar la ayuda del programa y los archivos de ejemplo, ellos se presentan ejemplos de casi todas las funciones especiales de los PICs. A continuación se presentara la estructura típica de un programa en base a un ejemplo simple:

```
#include <16F873A.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=20000000)

void main()
{
        delay_ms(3000);
        while(TRUE)
        {
                output_high(PIN_B0);
                delay_ms(1000);
                output_low(PIN_B0);
                delay_ms(1500);
        }
}
```

El código anterior lo que hace es: al comenzar la ejecución espera durante 3000ms (3 segundos) luego entra a un ciclo infinito, donde pone en HIGH el PIN B0 luego espera 1 segundo y lo pone en LOW, luego espera 1,5 segundos y vuelve a comenzar.
Entonces este programa genera una señal cuadrada de periodo 2,5 segundos con 1 segundo en HIGH muy similar al circuito del 555 de la experiencia anterior.
Ahora los encabezados que se muestrean se explican a continuación:
#include <16F873A.h>

Especifica que PIC se usara, de esta manera el compilador sabe que propiedades tiene y que módulos, y si uno lee este archive puede encontrar los nombres de las partes del PIC, por ejemplo que uno se refiera al PIN_B0 en verdad no se refiere a eso sino a un numero, en este caso 48, pero para facilitar la memoria, uno puede referirse al nombre.

#fuses HS,NOWDT,NOPROTECT,NOLVP

Esta línea especifica la línea de configuración del PIC, ya que si uno quiere usar algunos módulos en vez de otros, o quiere usar otro tipo de oscilador aquí es donde se debe cambiar, ya que en este controla como se debe comportar el PIC a la partida, en este caso se especifica que use HS (HighSpeed Oscillator), NOWDT(NO WATCH DOG TIMER), NOPROTECT(NO PROTECCION A LA MEMORIA), NOLVP(NO LOW VOLTAGE PROGRAMMING). Las otras opciones también pueden ser encontradas en el archivo .h agregado anteriormente. El NOPROTECT es OBLIGATORIO, ya que si lo quitan no se puede volver a cambiar el programa del PIC.

#use delay(clock=20000000)

Indica a que velocidad va a correr el oscilador, esto se hace para que las funciones que tienen que calcular tiempos como la función delay dure lo que debe durar, ya que si no esta sufre una alteración proporcional al tamaño que se indica con respecto al real. En este caso indica que esta corriendo a 20Mhz.

void main(){ }

Esta es la función main. Todo programa en PIC C parte ejecutando la función main, al igual que java esta devuelve una nada y no recibe parámetros. Otras funciones pueden ser creadas, donde al comienzo se indican el tipo de la variable a devolver, luego el nombre y entre paréntesis redondos cuales son los parámetros, y entre llaves el código, y deben terminar en un return donde entregan el valor a devolver.

delay_ms(3000);

Esta es una función que esta predefinida, la cual espera en este caso 3000ms y luego retorna, es decir espera 3000ms, también existe la función delay_us pero espera microsegundos. Hay que tener en cuenta que si a esta función se le pasa una variable el máximo valor llega a 255 y si se le pasa una constante llega hasta 65535.

while(TRUE){}

Esta es comando condicional que permiten hacer control de flujo, en esta caso es similar al de java, lo que esta entre corchetes se hace mientras se cumpla la condición dentro del paréntesis, en este caso se repite siempre. También existen los commandos if-else, do-while, for, etc.

output_high(PIN_B0);
output_low(PIN_B0);

Estos son comando para habilitar la salida como HIGH o LOW. En este caso como parámetro lleva PIN_BO. También existen OUTPUT_B(valor) donde en vez de levantar solo un PIN del puerto B se levanta todo el puerto B con el valor en binario ingresado. Otras funciones del mismo estilo es INPUT(pin) donde devuelve el valor del pin preguntado, siendo 0 o 1 la respuesta, de la misma manera existe INPUT_B() que devuelve el valor en un int de todo el puerto B.

Los diferentes tipos de datos con los que opera PIC C son diferentes a los usados en lenguajes como java. Acá por ejemplo no existe los booleanos, se usan ints donde 0 es FALSE y cualquier otra cosa es TRUE. A continuación una tabla con los tipos de datos que existe:

int1	define un numero de 1 bit
int8	define un numero de 8 bit
int16	define un numero de 16 bit
int32	define un numero de 32 bit
char	define un numero de 8 bit
float	define un numero de 32 bit de punto flotante
short	Por defecto lo mismo que un int1
Int	Por defecto lo mismo que un int8
long	Por defecto lo mismo que un int16
void	Ningún tipo en especifico

Otra de las funciones típicas de un PIC es la comunicación Serial, la gracia de este tipo de comunicación es que solo requiere tres cables, una tierra, una data transmisión y una data recepción. Esta caracterizada por ciertos parámetros como es la velocidad de transmisión, y si tiene control de errores o no. ¿Pero para que nos podría servir?, típicamente comunicación entre dispositivos, esto incluye a los computadores. Pero el problema para comunicarse con un PC es que la señal serial del PC va entre -15V y +15V, y la del PIC es de 0V a 5V, por eso para comunicarlos entre si hay que agregar una interfase que pueda convertir ambas comunicaciones (MAX232).

2. PARTE PRÁCTICA.

1- Escriba un programa que encienda y apague el B0 partiendo con 3segundos de periodo y reduciendo de a 100ms hasta llegar a 0 y entonces encienda B1 durante 5 segundos y vuelva a comenzar. Si tiene dudas pregúntele al auxiliar o a la ayuda del programa. Luego pídale al auxiliar que le corrija.
2- Haga diferentes funciones que usando el puerto B, enciendan diferentes números usando un Display de 7 Segmentos, acuérdese de cómo funcionaba el Display y ponga las resistencias donde corresponda, antes de conectar pida que le revisen.
3- Diseñe un circuito usando una resistencia y un pulsador que al apretarlo entregue 5V o 0V, vea que le conviene mas.
4- Usando la función INPUT(x) y un pin del puerto A haga una función que al apretar el botón llame a una función.
5- Integre todo para hacer un programa que al apretar el botón empiece una cuenta regresiva desde 9 a 0 y al terminar encienda un LED en el puerto A.
6- Pida que le revisen el código y el circuito, en caso que este bien se le dejara programar el PIC y probar su sistema.