CONTROL DEL OSCILADOR MEDIANTE C++

EDUARDO TRIANA MOYANO[1]

"No es la soledad la que nos asusta, sino las voces que la habitan"
Víctor Hugo

Resumen

El lenguaje C, dispone de un conjunto de funciones para acceder al reloj del sistema, pudiéndose leer el contador de ticks, visualizar la fecha y hora actual o mediante acceso al vector de interrupciones controlar el tiempo transcurrido entre un evento y otro, como también conocer la información horaria asociada con un archivo.

Palabras claves: reloj, piezoelectricidad, frecuencia cristal.

Abstract.

The language C, has a group of functions to consent to the clock of the system, being able to read the ticks accountant, to visualize the date and current hour or by means of access to the vector of interruptions to control the time lapsed among an event and other, as well as to know the information temporally associated with a file.

Key words: clock, piezoelectricity, frequency glass.

Contextualización Teórica

Básicamente el oscilador es un circuito generador de pulsos eléctricos de manera constante, el cual se implementa a partir de un amplificador retroalimentado que una la salida con la entrada.

La señal de oscilación, es un producto de la acción de la piezoelectricidad que evidencia el cristal de cuarzos, al aplicarse tensión en sus extremos y de la interacción con un conjunto de compuertas lógicas, que definen su funcionalidad.

Los chips 8284A, 82284 y 8253, son reconocidos fácilmente por el ingeniero, pues gracia a ellos se genera en el microprocesador la denominada señal de reloj (CLK), sin su presencia sería imposible obtener la sincronización de reinicialización, la sincronización de la señal de listo (READY), o la señal periférica de reloj de nivel TTL, como tampoco sin el temporizador 8253, se podría obtener la producción de sonidos, la generación de números aleatorios, la temporización del disco, o el refresco de la memoria RAM por acción directa del DMA.

El generador de reloj, integra el conjunto siguiente de señales: control del canal, frecuencia de cristal, entrada de cristal, CLK o reloj, salida del oscilador, entrada y salida de reinicialización, sincronización de reloj y reloj periférico, ésta última actúa sobre el equipamiento operacional del sistema y equivale a la sexta parte de la frecuencia del cristal, amén de incorporar el conjunto de registros que mantienen la feb ay hora.

El reloj calendario del sistema, maneja para la fecha y hora 32 BITS (16 para cada una), la fecha direcciona el formato AAAA MMM DDD (A = año, M = mes, D = día), y la hora opera con el formato binario HHH MMM SSS (H = hora, M = minuto, S = segundo), información que se referencia en el trabajo con archivos de operacionalizar las múltiples estructuras de control existente, tales como:

- **Estructura FFBLK**

```
Char  ff_reserved[21];
Char ff_attrib;
Int ff_time;
Int ff_date;
Long ff_size;
Char ff_name;
```

[1] Docente Programa de Ingeniería de Sistemas
Universidad Autónoma de Colombia

- **Estructura FTME**

Unsigned ft_sec;
Unsigned ft_min;
Unsigned ft_hour;
Unsigned ft_day;
Unsigned ft_month;
Unsigned ft_year;

- **Estructura STAT**

Dev_t st_dev;
Inv_t st_inv;
Unsigned short st_mode;
Short st_nlink;
Short st_uid;
Short st_gid;
Den_t st_rden;
Off_t st_size;
Time_t st_atime;
Time_t st_mtime;
Time_t st_ctime;

- **Estructura TM**

Int tm_sec;
Int tm_min;
Int tm_hour;
Int tm_day;
Int tm_mon;
Int tm_year;
Int tm_wday;
Int tm_yday;
Int tm_isdst;

El acceso al reloj, puede ser realizado, al manipular las interrupciones, 1AH servicios 2 y 4 para hora y fecha respectivamente o con la interrupción 23H servicios 2AH y 2CH para fecha y hora, o mediante el manejo de funciones catalogadas que leen el contador de ticks y lo modifican según necesidad, el chip generador libera cada hora 65536, por lo que 1 segundo es equivalente a 182044 ticks.

Se presenta a continuación ejemplos de programas que actúan sobre el oscilador del sistema.

Interacción oon el Reloj

- **Lectura de la hora**

Con ayuda de la interrupción 1AH, servicio O2, se lee el valor actual.

```
#include <bios.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
main ()
{
    unio regs uac;
    clrscr ();
    uac.h.ah.OX2;
    int86(&X1A, &UAC, &UAC);
    gotoxy (20,12);
printf ("son las: %x:%x:%x", uac.h.ch, uac.h.cl, uac.h.dh);
    getchar ();
    return (o);
}
```

- **Acceso al temporizador**

El tick es la medida del número de pulsos por segundo, el reloj genera 18204 pulsos / segundo y cada 24 horas se resetea, es decir, se coloca en cero, el siguiente programa muestra el número de pulsos emitidos por el circuito oscilador a partir de la media noche.

```
#include <bios.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
main ()
{
    clrscr ();
    gotoxy (20,12);
    printf ("se han generado %id pulsos o ticks", biostime (0,0));
getchar ();
    return (0);
}
```

- **Control de duración**

Operando sobre el vector de interrupciones y validando el descriptor de nivel número 8 que actúa sobre el reloj, se puede obtener el registro de tiempo transcurrido durante una actividad.

```
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
int total = 0;
void interrupt (*dos_interrupot) (void);
```

```
void interrupt reloj ();
{
    total = total+1;
    (*dos_interrupt)();
}
main ()
{
    char uac [50];
dos_interrupt = getuect (8);
disable ();
setuect (8, reloj);
enable ();
clascr ();
gotoxy (20,12);
printf (“Digite una línea de texto”);
gotoxy (20,14);
gets (uac);
gotoxy (20,16);
printf (“Usted tardó %d ticks”, total”);
disable ();
setuect (8, dos_interrupt);
enable ();
getchar ();
return (0);
    }
```

- **Información de hora y fecha de creación**

Empleando la estructura STAT, se puede obtener la información asociada con la feb ay hora de creación del archivo que se especifica como parámetro.

```
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys\stat.h>
#include <sys\types.h>
main (int argc, char * argv[])
{
```

```
    int jaysha;
    struct stat keyla;
if (argc != 2)
    {
        clrscr ();
        gotoxy (20,12);
        printf (“Error al digitar repita”);
        getchar ();
        abort ();
    }
else
{
    while (*(++argv))
    {
        if ((jaysha = open (xargv, ordonly)==-1)
        {
            clrscr ()
            gotoxy (20, 12);
            printf (“No se puede accesar el archivo %s”, x argv);
            Getchar ();
            Exit (1);
        }
        Else if (fstat (uac, &keyla)==-1)
            {
                clrscr ();
                Gotoxy (20, 12);
                Printf (“No se puede operar archivo %s”, xargv);
                Getchar ();
                Exit (1);
            }
    else
    {
        clrscr ();
        gotoxy (20, 12);
        printf (“Archivo = %s”, xargv);
        gotoxy (20, 14);
        printf (“Longitud = %ld” keyla_st_atime);
        getchar ();
        close (jaysha);
    }
            }
        }
    return (0);
}
```

Bibliografía

Brey Barry. Los microprocesadores Intel. Editorial Prentice Hall, 2002.

Jamsa Kris. Microsoft c: Secrets, shorcuts and solutions. Editorial Microsoft Press, 1997.

Martín, José María. Hardware Microinformático. Editorial Alfa Omega, 2002.