

**UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO**

FACULTAD DE INGENIERÍA

**PROYECTO 01**

**RELOJ CRONÓMETRO**

Estructura y programación de computadoras

Grupo: 02

Profesor:

Duran Arenas Luis Sergio

Alumnos:

Esparza Fuentes Jorge Luis

Mora González Alan Francisco

Semestre 2021-1

18 de diciembre del 2020

**PROYECTO 01**

**RELOJ CRONÓMETRO**

INTRODUCCIÓN.

Un reloj es un dispositivo que permite realizar la medición del tiempo y segmentarlo en unidades (segundos, minutos, horas, etc.). El mecanismo del reloj consiste en desarrollar un movimiento de características uniformes, regulándolo con un péndulo. Dicho movimiento se comunica, a través de ruedas, con las agujas o manecillas que indican la hora. Se denomina reloj digital al que indica la hora mediante números, por oposición al reloj analógico que lo hace mediante manecillas. El funcionamiento de los relojes digitales es electrónico normalmente, si bien existen emuladores informáticos que se pueden presentar en la pantalla de una computadora tanto relojes analógicos como digitales.

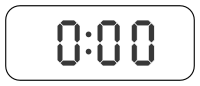
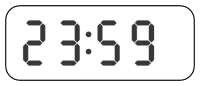
 

Figura 1. Reloj digital.

La palabra cronómetro es un neologismo de etimología griega: Χρόνος Cronos es el Titán del tiempo, μετρον -metron es hoy un sufijo que significa aparato para medir. Un cronómetro es un reloj de precisión que se emplea para medir fracciones de tiempo muy pequeñas. A diferencia de los relojes convencionales que se utilizan para medir los minutos y las horas que rigen el tiempo cotidiano, los cronómetros suelen usarse en competencias deportivas y en la industria para tener un registro de fracciones temporales más breves, como milésimas de segundo.



Figura 2. Cronómetro digital.

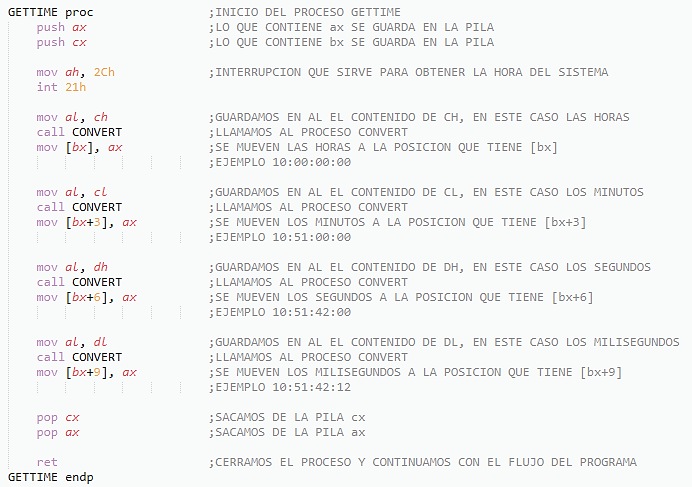
En este proyecto se planteó la programación en lenguaje ensamblador, para la arquitectura Intel x86, un reloj y un cronometro digital, además de dejar como extra una opción para dar la fecha del sistema.

Los aspectos a considerar son:

* Se deberá imprimir en pantalla la opción que según seleccione el usuario; si el usuario selecciona la opción 1, imprimirá la hora del sistema en pantalla. Si el usuario selecciona la opción 2, imprimirá la fecha del sistema en pantalla. Si el usuario selecciona la opción 3, abrirá un sub menú que corresponderá a la manipulación de un cronometro digital. Si el usuario selecciona la opción 4, el programa se cerrará ya que habrá seleccionado la opción de salir.
* Si el usuario selecciona la opción Reloj, el programa deberá mostrar la hora del sistema, en formato HH:MM:SS (HH-horas, MM-minutos, SS-segundos) y cada segundo se debe actualizar. Para regresar al menú, el usuario debe presionar cualquier tecla.
* Si el usuario selecciona la opción Cronómetro, el programa deberá funcionar como un cronómetro y mostrarlo en pantalla en formato MM:SS.mmm. El cronómetro debe tener 3 opciones que pueden seleccionarse por el usuario a través del teclado: Iniciar/continuar, detenerse (pausa) y reiniciar.
* El cronómetro debe iniciar en ceros. Si el usuario selecciona la opción de Iniciar/continuar, el cronómetro arranca y empieza a contar, el cronómetro se debe actualizar todo el tiempo. Si mientras el cronómetro está corriendo, el usuario selecciona la opción Detener, el cronómetro se detiene, y en cualquier momento si el usuario vuelve a seleccionar la opción Iniciar/continuar el cronómetro continúa desde el punto en el que se quedó. Por último, si el usuario selecciona la opción Reinicia el cualquier momento, el cronómetro vuelve a su estado inicial.
* Para regresar el menú, el usuario puede presionar cualquier otra tecla.
* Para finalizar la ejecución del programa, el usuario debe seleccionar la opción Salir en el menú principal.

DESARROLLO.

Para la solución al problema planteado para la impresión del reloj se ocupó un proceso programado por nosotros llamado GETTIME, nos funcionaba ya que nos era más fácil llamar a un proceso que el ciclo se repitiera una y mil veces:

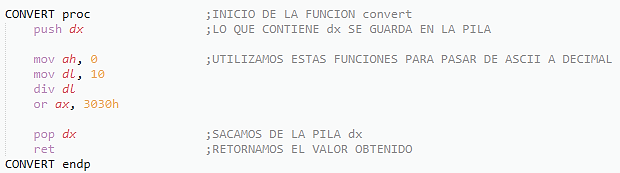


Primero guardamos en la pila los registros ax y cx, ya que los utilizaremos después en el flujo del programa y no queríamos utilizar auxiliares.

Después utilizamos la interrupción 21h, habiendo declarado antes ah como 2Ch, esto nos servirá para obtener la hora del sistema. Sabiendo así que dentro de los registros CX y DX encontraremos la información repartida de la siguiente manera:

|  |  |
| --- | --- |
| Horas | ch |
| Minutos | cl |
| Segundos | dh |
| Milisegundos | dl |

Una vez sabiendo esto solo lo mandamos a imprimir, pero primero tendríamos que convertirlo de ASCII a decimal, es por eso que utilizamos otro proceso llamado CONVERT, y este lo estaremos llamando para cada uno de los valores:



Después de convertir los valores los guardábamos en la dirección de bx según nos conviniera, ¿Por qué en el registro bx y no en una variable?, bueno pues antes de entrar al loop hicimos uso de una instrucción:



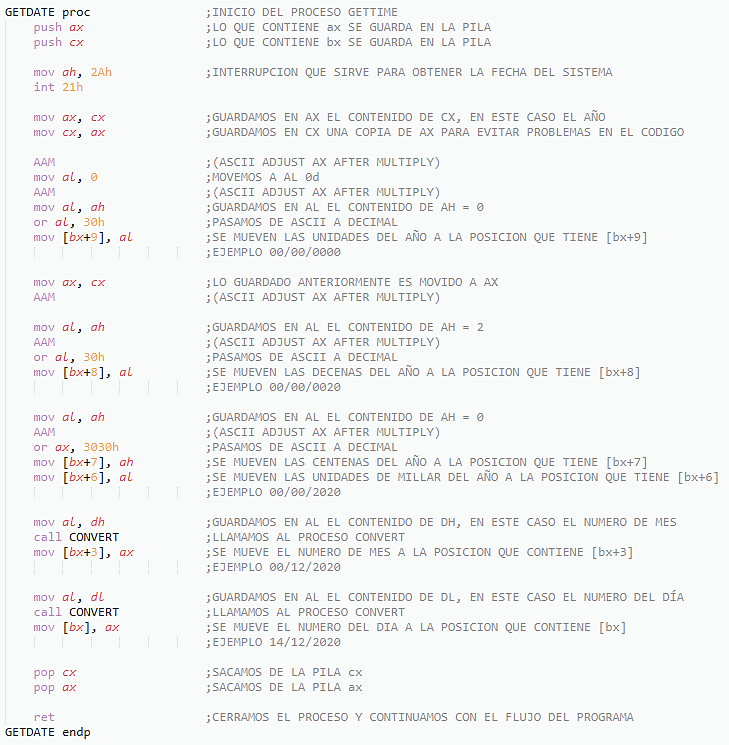
Utilizamos la palabra “transportar” porque si modificamos el registro bx, automáticamente se modificaría la variable timer declarada con anterioridad:



Por último, hicimos que esto se repitiera infinitamente hasta que el usuario hiciera uso de cualquier tecla:



Para imprimir la fecha utilizamos un proceso llamado GETDATE:



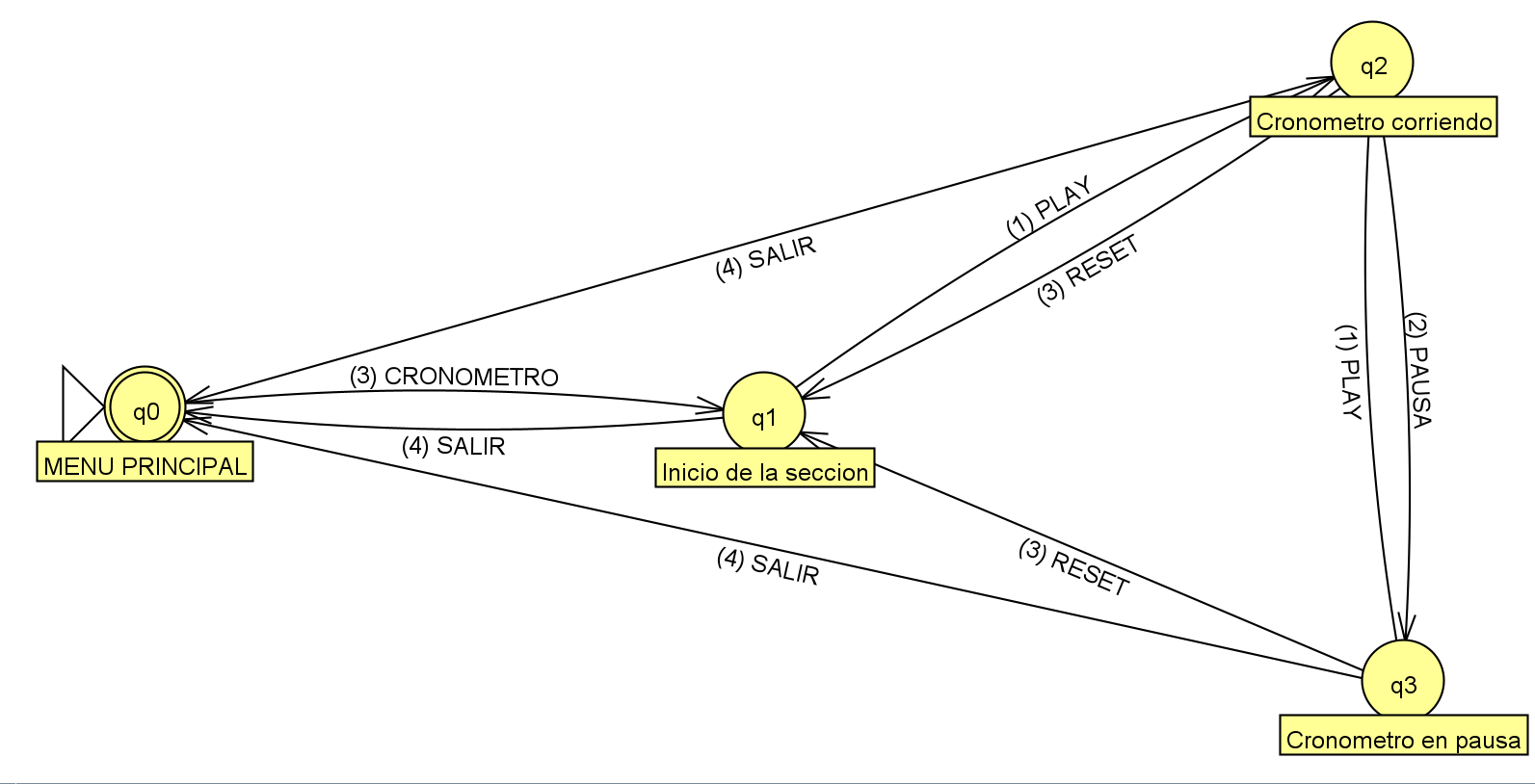
En este proceso tratamos el año muy diferente, ya que era un numero de cuatro dígitos, y no podíamos utilizar el proceso CONVERT ya que este proceso única y exclusivamente servía para números de dos dígitos, entonces el único que cambio que implementamos en este punto fue dividir el numero con ayuda del operador AAM.

En el caso del cronómetro, se presentaron muchos problemas al momento del diseño de la solución del problema, ya que debido a la muy poca información disponible en internet, se tuvo que realizar varias pruebas para saber cómo tal la información que nos aporta CX : DX al ingresar la instrucción int 1Ah opción 00h, no obstante, con la ayuda del código de ejemplo proporcionado, se puedo concluir de manera óptima la realización de esta sección.

Sin embargo previamente a todo esto, se diseñó el primer boceto de un algoritmo el cual consiste en que si sabiendo que una iteración de un loop infinito cuesta prácticamente un ciclo de reloj, entonces ese loop iba a servirnos como los ticks necesarios, de tal forma que, cada mil iteraciones equivaldría aproximadamente 1000 ciclos de reloj por lo que se podría traducir a 1000 milisegundos  lo que esto equivale a un segundo entonces si este loop lo dejábamos correr dentro de un loop “maestro”, entonces solo bastaría con solo aumentar periódicamente hasta que los segundos valgan 60, en ese instante se le incrementa a los minutos una unidad y se reinician los segundos en 0, así sucesivamente hasta que los minutos valgan 60, los segundos 0 y los milisegundos 0 también.

 Este algoritmo en primera instancia sonaba prometedor, sin embargo, su complejidad en cuanto a la programación era un tanto innecesaria. Entonces se optó por la instrucción propuesta por los rubros del proyecto. en cierto modo su complejidad en términos de O(n) es lineal pero multiplicada por una constante equivalente a todas las instrucciones del algoritmo, en cambio la complejidad del algoritmo anterior que pretendía resolver el problema era cuadrático.

Claramente la diferencia entre ambos aparentemente es similar, sin embargo, la instrucción 1AH es constante, y esta hace que se acceda al reloj del procesador por lo que no hay ningún coste algorítmico extra, en contraste con el primer algoritmo planteado el cual hacíamos nosotros en este 1000 iteraciones dentro de un loop infinito, por lo que se llegó a la conclusión que es mejor utilizar la instrucción.

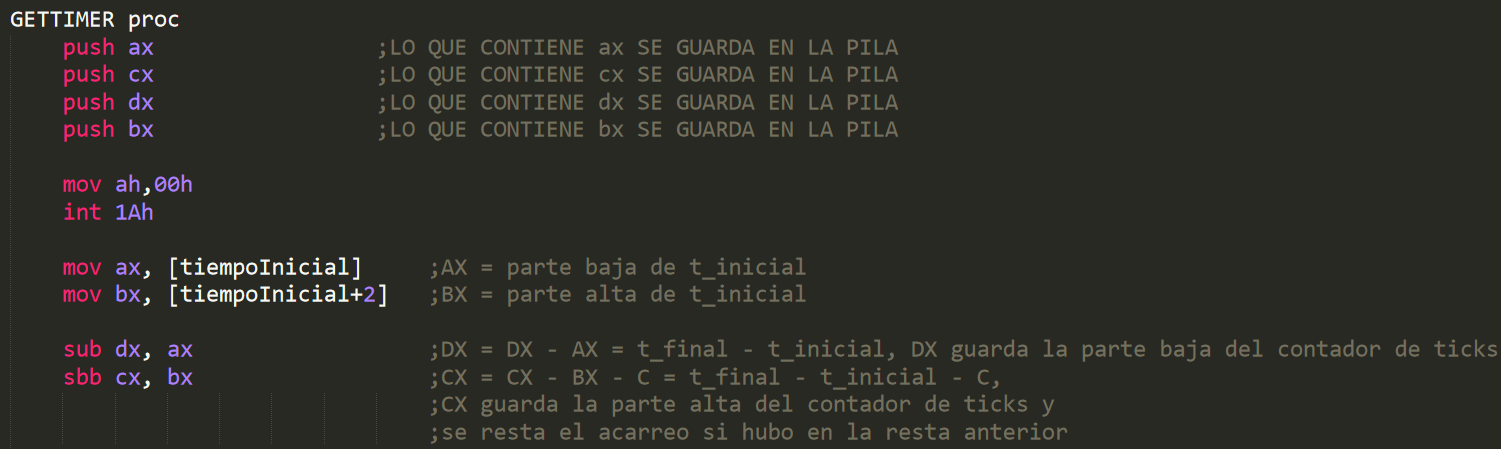


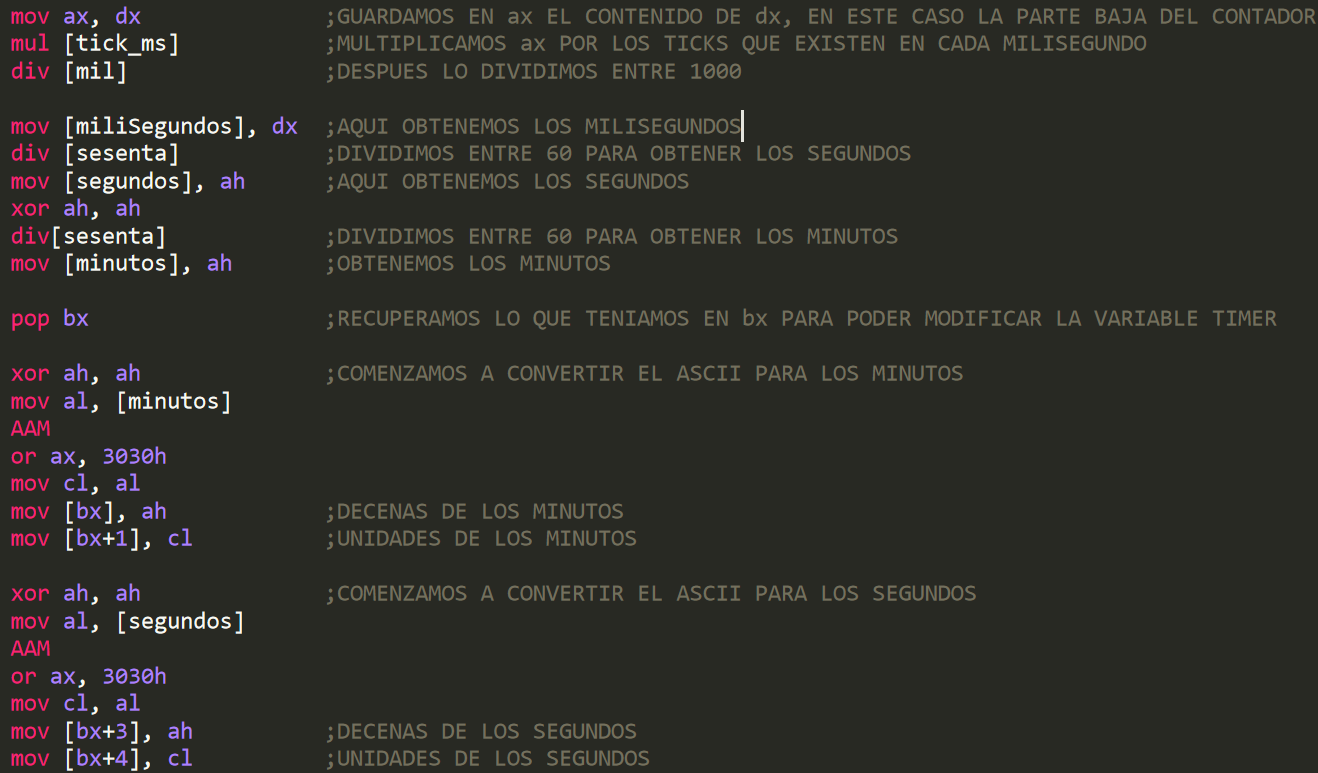
Ahora bien, la sección del cronómetro está constituido por 4 estados tomando como estado inicial al menú principal, al estado 1 como el inicio de la sección del cronómetro, al estado 2 cuando el cronómetro está corriendo y el estado 3 el cual es la pausa del cronómetro.

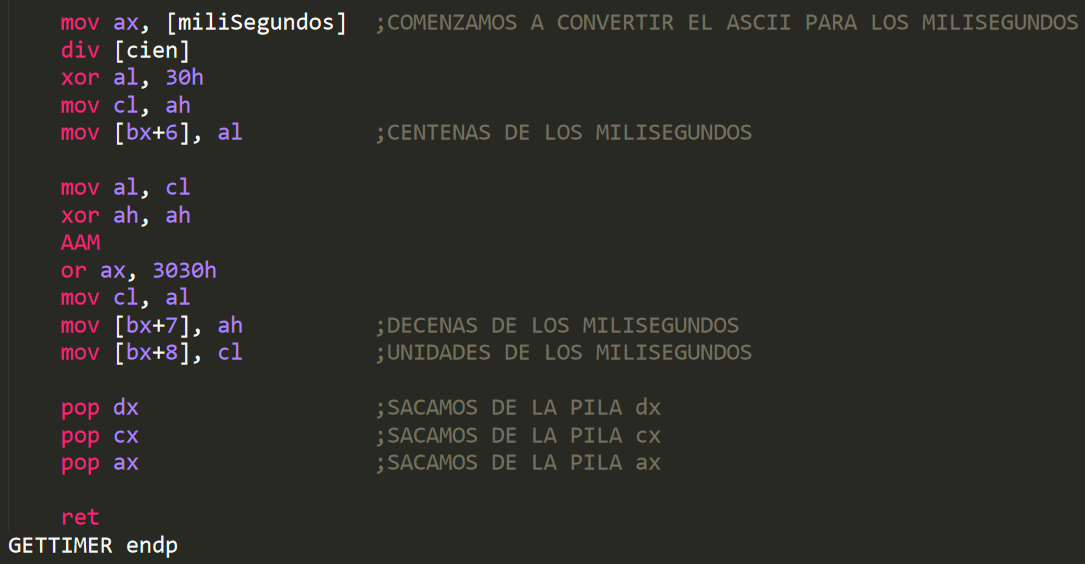
A continuación, se presentan las posibles opciones para cada estado

* **Q0** Menú principal
  + **(3)** Cronómetro, al pulsar tres se salta al estado q1
* **Q1** Inicio de la sección
  + **(1)** Iniciar, se salta al estado q2, los contadores empiezan a correr
  + **(4)** Salir, se salta al estado q0
* **Q2** Cronómetro corriendo
  + **(2)** Pausar, se salta al estado q3
  + **(3)** Reiniciar, salta al estado q1, reiniciando todos los contadores a 0
  + **(4)** Salir, salta al estado q0
* **Q3** Cronómetro en pausa
  + **(1)** Reanudar, se salta al estado q1, los contadores siguen corriendo
  + **(3)** Reiniciar, salta al estado q1, reiniciando todos los contadores a 0
  + **(4)** Salir, salta al estado q0

El procedimiento principal que realiza todo el procesamiento de obtener los registros dada la instrucción 1Ah opción 00h, convertirlos a ascii y guardarlos en la cadena timer es la siguiente:







El procedimiento GETTIMER se divide en 3 secciones principales:

1. **Obtención de datos de los registros DX y CX**

Sabiendo que int 1Ah opción 00h es la instrucción para obtener el tiempo real del sistema, retornando estos datos, la parte alta en CX y la baja en DX, entonces se obtienen por primera vez en el estado 1 (inicio de la sección) guardándolas en la variable ***tiempoInicial***  después ya estando en el estado 2 el cual es un loop que va ir actualizando lo que muestra en pantalla y llamando constantemente al procedimiento GETTIMER, una vez dentro del procedimiento ahora si esta es la primera sección la cual consiste en volver a realizar la interrupción 1Ah, 00h, obteniendo nuevamente CX y DX  pero esta vez le restamos a estos dos registros la parte alta y baja de la variable ***tiempoInicial***, con el objetivo de obtener finalmente con DX los milisegundos.

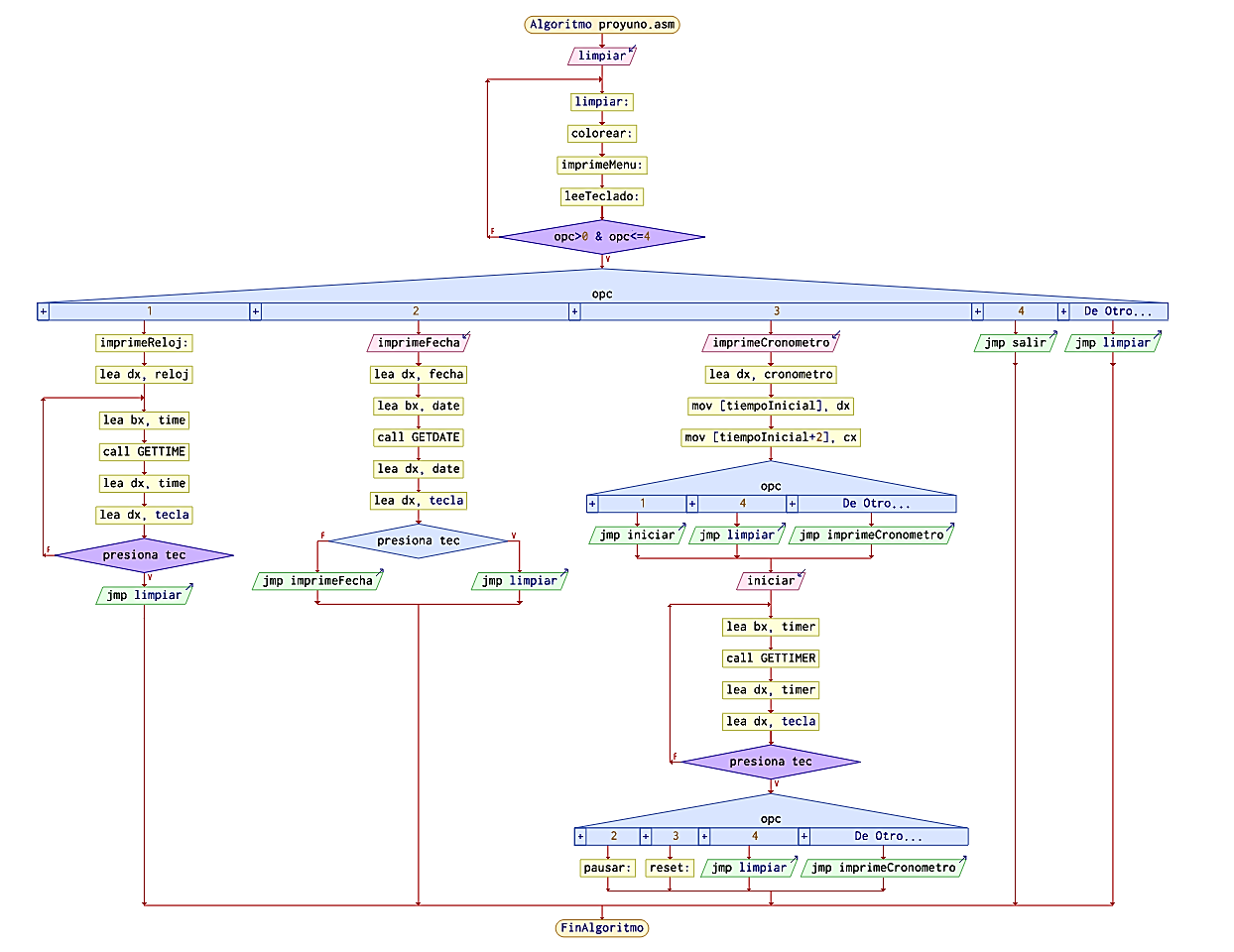
1. **Conversión y procesamiento de los datos a ascii y a su vez a milisegundos, segundos y minutos.**

Una vez obtenido DX se multiplica por el número de ticks en un segundo para realizar la equivalencia en segundos después se divide entre mil para obtener los milisegundos, finalmente se realizan las conversiones necesarias para transformar los datos adquiridos al ASCII.

1. **Guardado de los datos procesados a la cadena timer**

se van a ir guardando milisegundos, segundos y minutos en cada posición correspondiente de la cadena ***timer*** de este modo se ahorran varios pasos cómo imprimir elemento a elemento.

Diagrama de flujo



Pruebas de escritorio

**Prueba 1**

1. Inicio
2. Mientras se espera lectura de teclado hacer:
   1. Se limpia pantalla
   2. Se colorea pantalla
   3. Imprime menú, mostrando opciones: 1) Reloj, 2) Fecha, 3) cronometro o 4) salir
   4. Leer teclado
3. Se ingresa 1
4. Se salta a la etiqueta imprime reloj
5. Mientras se espera lectura de teclado hacer:
   1. Obtener el tiempo del reloj del sistema
   2. Transformarlo y asignarlo a la variable time
   3. Se obtiene como salida, la impresión en pantalla de la variable time
   4. Leer teclado
6. Limpiar pantalla
7. fin

**Prueba 2**

1. Inicio
2. Mientras se espera lectura de teclado hacer:
   1. Se limpia pantalla
   2. Se colorea pantalla
   3. Imprime menú, mostrando opciones: 1) Reloj, 2) Fecha, 3) cronometro o 4) salir
   4. Leer teclado
3. Se ingresa 2
4. Se salta a la etiqueta *imprime fecha*
5. Obtener del sistema la fecha
6. Transforma los datos obtenidos y asignarlos a la variable date
7. Se obtiene como salida, la impresión en pantalla de la variable date
8. Limpiar pantalla
9. Fin

**Prueba 3**

1. Inicio
2. Mientras se espera lectura de teclado hacer:
   1. Se limpia pantalla
   2. Se colorea pantalla
   3. Imprime menú, mostrando opciones: 1) Reloj, 2) Fecha, 3) cronometro o 4) salir
   4. Leer teclado
3. Se ingresa 3
4. Se salta a la etiqueta *imprime cronómetro*
5. se obtienen el tiempo inicial del sistema
6. imprime el cronometro inicializado en 0’s , en el caso de reinicio es la función reset
7. imprime menú de sección, mostrando las opciones de 1) Iniciar 2) Salir
8. Lee el teclado
9. Se ingresa 1
10. Salta a la etiqueta *loop1*
11. Mientras se espera lectura del teclado hacer:
    1. obtener del sistema otra vez el tiempo del sistema, restar con el inicial y transformas a ASCII
    2. guardar en la variable timer
    3. Se obtiene como salida, la impresión en pantalla de la variable timer
    4. imprime menú de sección, mostrando las opciones de 2) pausar 3) reiniciar 4) salir
    5. leer teclado
12. Se ingresa 2
13. Salta a la etiqueta *pausaloop*
14. Mientras se espera lectura del teclado hacer:
    1. Se obtiene como salida, la impresión en pantalla de la variable timer en la última iteración realizada
    2. imprime menú de sección, mostrando las opciones de 1) Iniciar 3) reiniciar 4) salir
    3. Lee el teclado
15. Se ingresa 3
16. Se salta a la etiqueta *imprime cronómetro*
17. Se obtiene como salida, la impresión en pantalla de la variable timer inicializada en 0’s
18. imprime menú de sección, mostrando las opciones de 1) Iniciar 2) Salir
19. Lee el teclado
20. Se ingresa 1
21. Salta a la etiqueta *loop1*
22. Mientras se espera lectura del teclado hacer:
    1. obtener del sistema otra vez el tiempo del sistema, restar con el inicial y transformas a ASCII
    2. guardar en la variable timer
    3. Se obtiene como salida, la impresión en pantalla de la variable timer
    4. imprime menú de sección, mostrando las opciones de 2) pausar 3) reiniciar 4) salir
    5. leer teclado
23. Se ingresa 2
24. Salta a la etiqueta *pausaloop*
25. Mientras se espera lectura del teclado hacer:
    1. Se obtiene como salida, la impresión en pantalla de la variable timer en la última iteración realizada
    2. imprime menú de sección, mostrando las opciones de 1) Iniciar 3) reiniciar 4) salir
    3. Lee el teclado
26. Se ingresa 4
27. Limpiar pantalla
28. Se obtiene como salida, el salto a la etiqueta *imprime menu*
29. Fin

**Prueba 4**

1. Inicio
2. Mientras se espera lectura de teclado hacer:
   1. Se limpia pantalla
   2. Se colorea pantalla
   3. Imprime menú, mostrando opciones: 1) Reloj, 2) Fecha, 3) cronometro o 4) salir
   4. Leer teclado
3. Se ingresa 3
4. Se salta a la etiqueta *imprime cronómetro*
5. se obtienen el tiempo inicial del sistema
6. imprime el cronometro inicializado en 0’s , en el caso de reinicio es la función reset
7. imprime menú de sección, mostrando las opciones de 1) Iniciar 2) Salir
8. Lee el teclado
9. Se ingresa 1
10. Salta a la etiqueta *loop1*
11. Mientras se espera lectura del teclado hacer:
    1. obtener del sistema otra vez el tiempo del sistema, restar con el inicial y transformas a ASCII
    2. guardar en la variable timer
    3. Se obtiene como salida, la impresión en pantalla de la variable timer
    4. imprime menú de sección, mostrando las opciones de 2) pausar 3) reiniciar 4) salir
    5. leer teclado
12. Se ingresa 3
13. Se salta a la etiqueta *imprime cronómetro*
14. Se obtiene como salida, la impresión en pantalla de la variable timer inicializada en 0’s
15. imprime menú de sección, mostrando las opciones de 1) Iniciar 2) Salir
16. Lee el teclado
17. Se ingresa 1
18. Salta a la etiqueta *loop1*
19. Mientras se espera lectura del teclado hacer:
    1. obtener del sistema otra vez el tiempo del sistema, restar con el inicial y transformas a ASCII
    2. guardar en la variable timer
    3. Se obtiene como salida, la impresión en pantalla de la variable timer
    4. imprime menú de sección, mostrando las opciones de 2) pausar 3) reiniciar 4) salir
    5. leer teclado
20. Se ingresa 4
21. Limpiar pantalla
22. Se obtiene como salida, el salto a la etiqueta *imprimeMenu*
23. Fin

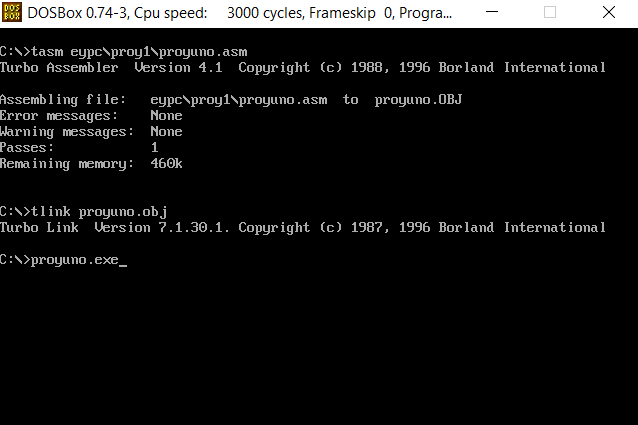
**Prueba 5**

1. Inicio
2. Mientras se espera lectura de teclado hacer:
   1. Se limpia pantalla
   2. Se colorea pantalla
   3. Imprime menú, mostrando opciones: 1) Reloj, 2) Fecha, 3) cronometro o 4) salir
   4. Leer teclado
3. Se ingresa 3
4. Se salta a la etiqueta *imprime cronómetro*
5. se obtienen el tiempo inicial del sistema
6. imprime el cronometro inicializado en 0’s , en el caso de reinicio es la función reset
7. imprime menú de sección, mostrando las opciones de 1) Iniciar 2) Salir
8. Lee el teclado
9. Se ingresa 4
10. Limpiar pantalla
11. Se obtiene como salida, el salto a la etiqueta *imprimeMenu*
12. Fin

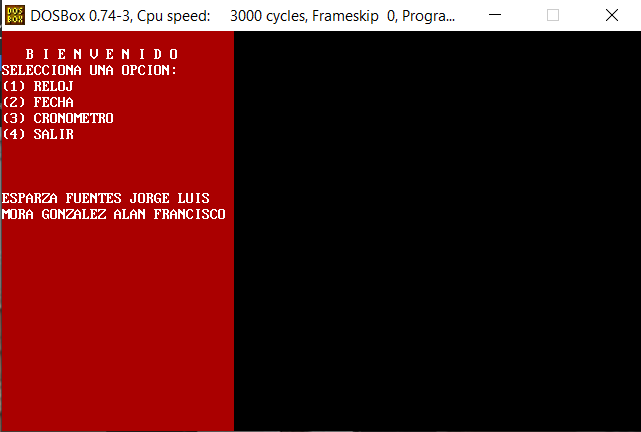
**Prueba 6**

1. Inicio
2. Mientras se espera lectura de teclado hacer:
   1. Se limpia pantalla
   2. Se colorea pantalla
   3. Imprime menú, mostrando opciones: 1) Reloj, 2) Fecha, 3) cronometro o 4) salir
   4. Leer teclado
3. Se ingresa 4
4. Limpiar pantalla
5. Se obtiene como salida, el salto a la etiqueta *salir*
6. Se le regresa el control al sistema con la int 21h, opción 4C00h
7. Fin

Capturas de pantalla



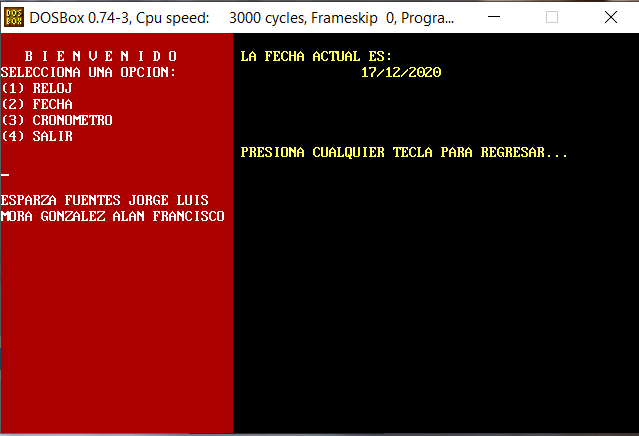
Se confirma que el código fue correctamente ensamblado y enlazado, por lo que se procede a ejecutar el código



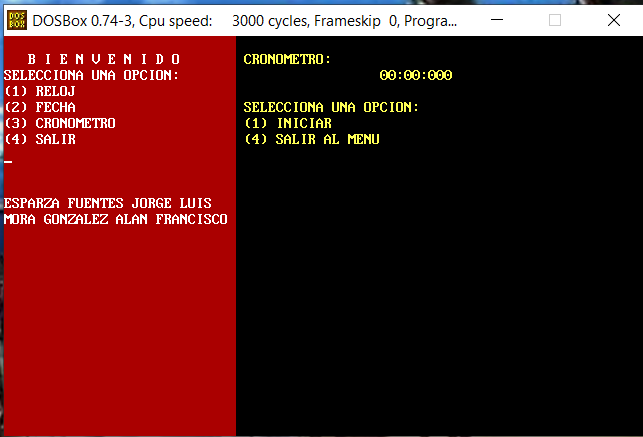
Como se planteó en el algoritmo y se testeó el algoritmo con base en pruebas de escritorio, se confirma que se accede al menú principal, ahora bien, primero se ingresa al reloj.

Insertar captura de pantalla del reloj

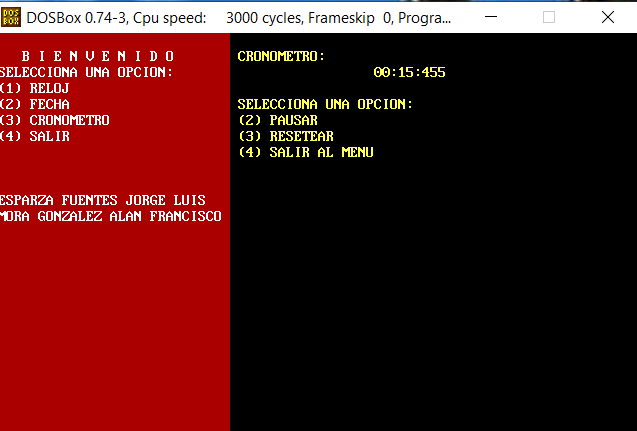
Como se puede observar, se confirma lo esperado del diseño y las pruebas de escritorio, ahora se procede a acceder a la fecha



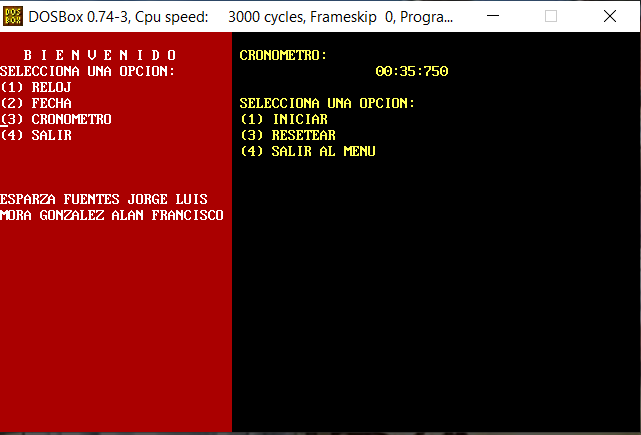
Se confirma lo que se diseñó en esta implementación, tambien las pruebas de escritorio concuerdan, finalmente se procede a acceder al cronómetro.



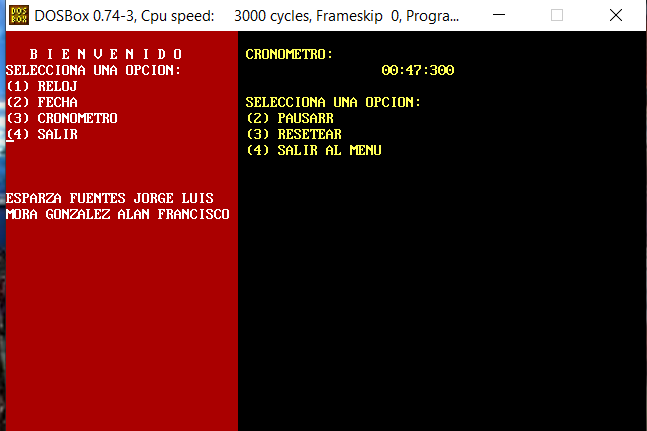
Como se puede observar se accedió según el diseño y el diagrama de estados, al estado q1 el cual consiste en mostrar el menú de esta sección y el cronómetro inicializado en 0’s, se procede a iniciar el cronometro.



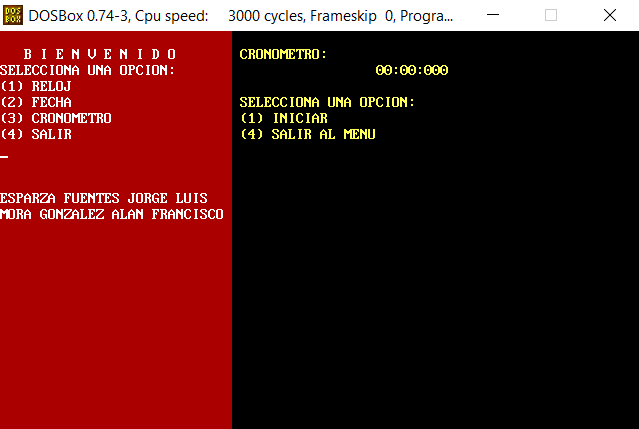
Como se puede ver se pasó al estado q2 el cual consiste en mostrar en pantalla el cronómetro corriendo y las opciones de pausar, reiniciar y salir al menú, en este caso se procede a pausar el cronómetro.



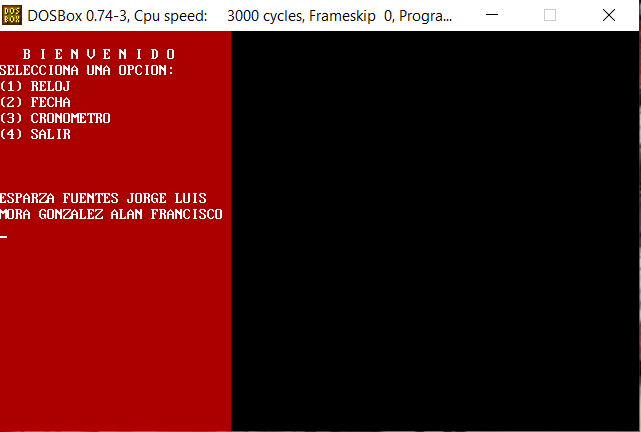
Cuando el cronómetro esta en pausa se confirma que se pasó al estado q3 el cual muestra como ya se mencionó el cronómetro pausado y las opciones de iniciar, restear y salir al menú. En este caso se procede a reanudar el cronómetro.



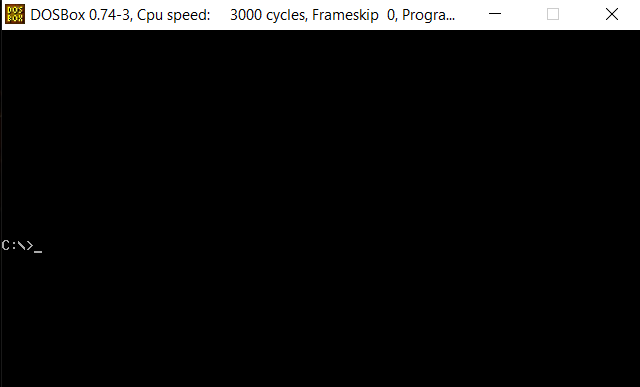
Como se puede ver se paso al estado q2 de nuevo por lo que el cronómetro se reanuda y sigue corriendo. Ahora se procede a reiniciar el cronómetro.



Nótese que el cronómetro se reinicia bien por que se igualan a 0’s el cronómetro como tal y se muestra en pantalla las mismas opciones que del estado q1 por lo que se confirma que se saltó a ese estado correctamente como se había diseñado. Finalmente se procede a salir al menú principal y después salir del programa.



Efectivamente se limpio el fondo negro y se accede al menú principal (fondo rojo) por lo que se confirma que se pudo saltar al estado inicial q0 del diseño.



Final mente el sistema vuelve tomar control después de haber ingresado la opción 4 que es salir y de haber limpiado pantalla.

Conclusiones

**Esparza Fuentes Jorge Luis**

Se presentaron muchas complicaciones a lo largo del tiempo de elaboración del proyecto, sin embargo, se logro resolver cada problema que surgió.

Se obtuvo el diseño de la solución del problema, así como tambien su implementación usando el lenguaje de programación Assembly para arquitectura de un procesador Intel x86. Por lo que se puede concluir que se obtuvieron las salidas esperadas con base en las pruebas de escritorio, al ingresar las entradas planteadas en esta misma sección, tambien se obtuvo la complejidad del algoritmo para obtener el cronómetro, el cual era el más complicado y enredado de hacer.

Por lo que, en otras palabras, se analizó el problema, se planteó su solución, se diseñó la solución y se realizaron pruebas de escritorio para que finalmente se implementara la solución basados en el diseño.

**Mora González Alan Francisco**

**REFERENCIAS**

Pérez, J & Gardey, A. (2009). *Definición de Reloj*. diciembre 16, 2020, de Definición.de Sitio web: <https://definicion.de/reloj/>

Autor anónimo. (2020). *Reloj digital*. diciembre 16, 2020, de Wikipedia Sitio web: <https://es.wikipedia.org/wiki/Reloj_digital>

Figura 1 tomada de: <https://upload.wikimedia.org/wikipedia/commons/thumb/6/6a/Reloj_digital_2359.svg/200px-Reloj_digital_2359.svg.png>

Pérez, J & Merino, M. (2011). *Definición de cronómetro*. diciembre 17, 2020, de Definición.de Sitio web: <https://definicion.de/cronometro/>

Figura 2 tomada de: <https://www.google.com/url?sa=i&url=http%3A%2F%2Fdiccionariofacil.org%2Fdiccionario%2FC%2Fcronometro-1.html&psig=AOvVaw1pdKmRsdrDmL9gTZ2SPLMU&ust=1608305545046000&source=images&cd=vfe&ved=0CAIQjRxqFwoTCNid4Zur1e0CFQAAAAAdAAAAABAD>