Trabajo Practico De Organización Del Computador

Integrantes Del Grupo:

Jorge Gastón Sanchez Salinas

Leandro Javier Campanella Miranda

Profesores:

Viviana Nakatsuka

Cristian Ciarallo

Comisión 05 turno mañana

Fecha De Entrega: 15/6/21

Introducción

En el siguiente trabajo se nos pidió realizar una aplicación en la arquitectura ARM que nos permitiera cifrar o descifrar mensajes ingresados por el usuario a través de la consola, teniendo en cuenta una clave o pista ingresada junto con el mensaje. El programa esta codificado en lenguaje ensamblador, gracias a esta aplicación sabremos que nos quieren informar en un mensaje encriptado o encriptar nuestros propios mensajes.

Para poder realizar de manera eficaz la elaboración del trabajo se recurrió a llamadas grupales frecuentes y mantenerse en constante comunicación sobre los cambios realizados con el fin de mantener a todos los miembros del grupo al tanto de nuevas modificaciones.

Para iniciar la codificación del programa se fueron implementando las rutinas principales mencionadas en el enunciado del trabajo practico, asegurándose del correcto funcionamiento de cada una de ellas antes de comenzar a implementar la siguiente. Luego de tener un programa base que leyera lo que ingresa el usuario y lo dividiera en partes para su manipulación individual, se procedió a la codificación o decodificación según lo requiera el usuario y por último a informar por pantalla la cantidad de letras procesadas durante la ejecución del programa. Tomamos en consideración todas las pautas del enunciado para poder realizar trabajo, a la hora de llevarlo a cabo fuimos implementando cada una de ellas y gracias a esto se cumplió con lo pedido en el enunciado.

Dificultades

Las dificultades encontradas durante este trabajo fueron varias, principalmente la complicada manera en la que se escribe en lenguaje ensamblador, ya que un error en la modificación de un registro podía hacer que el programa dejara de funcionar por completo, por eso se recurrió a comentar todo lo que fuera necesario y de esa manera poder seguir de manera mucho más simple y clara la ejecución del programa y el contenido de los registros. Otra dificultad se dio durante las conversiones de variables numéricas a tipo carácter ASCII y viceversa, por eso se limitó el rango de opciones que podía escoger el usuario al momento de ingresar la clave quedando entre 0 y 9. Una dificultad similar se dio al momento de mostrar cuantas letras había

procesado el programa al final, luego de mucho pensar e intentar distintas formas se pudo llegar a una solución, pero en el mientras se dificulto notablemente. Por último, una dificultad no relacionada con el código, pero si con la manera de trabajar fue la disponibilidad horaria ya que teniendo materias que cursar en la mañana y trabajo durante la tarde el tiempo para coincidir no era mucho, pero con la predisposición de los miembros del grupo se pudo superar esa dificultar organizando en su mayoría de veces llamadas y tiempo de codificación en la noche.

Funcionamiento del programa

Codificación: El programa mostrara un mensaje de bienvenida, seguidamente el usuario debe ingresar un mensaje a encriptar, una clave numérica comprendida entre 0 y 9 que marcara el desplazamiento de los caracteres y la letra C ya sea minúscula o mayúscula que determinara que se quiere codificar. Todo lo ingresado debe estar separado por ";".

Ej: *"hola buenas tardes;6;c"*

Luego de la ejecución, el programa mostrara el mensaje encriptado, un bit de paridad que depende de la cantidad de letras procesadas siendo 0 si es par y 1 si es impar y finaliza informando la cantidad de letras procesadas.

Ej: *"Su mensaje procesado es: nurg haktgy zgxjky 0*

Cantidad de letras procesadas: 16"

Decodificación: El programa mostrara un mensaje de bienvenida, seguidamente el usuario debe ingresar un mensaje a desencriptar, una pista comprendida entre 0 y 9 caracteres que marcará el desplazamiento, la letra D ya sea minúscula o mayúscula que determinará que se quiere decodificar y por último un bit de paridad que será 0 o 1 dependiendo si la cantidad de letras a decodificar es par o impar respectivamente. Todo lo ingresado debe estar separado por ";".

Ej: *"nurg haktgy zgxjky;ocungs;d;0"*

-Si el bit de paridad ingresado no coincide con la paridad del mensaje el programa mostrara un mensaje de error

Ej: "Error!, La paridad de su mensaje no coincide"

-Luego de la ejecución, si la paridad ingresada es correcta, el programa mostrara el mensaje desencriptado y finaliza informando la cantidad de letras procesadas.

Ej: *“Su mensaje procesado es: hola buenas tardes*

Cantidad de letras procesadas: 16”

Conclusión

Para finalizar con una conclusión, el trabajo y las dificultades encontradas durante su elaboración nos dieron la experiencia necesaria para trabajar en equipo de forma constante, priorizando la comunicación y la organización. En cuanto a habilidades destacamos la mejora en el manejo de conversión de bases numéricas, las instrucciones de la arquitectura ARM, el uso de condiciones para la decisión del orden de ejecución, una mejor comprensión del paso a paso durante la ejecución de un programa y la importancia de comentar lo que ocurre en cada subrutina para seguir de manera simple lo que sucede en el programa. Creemos que todas estas habilidades serán de suma utilidad durante el progreso de la carrera y la vida profesional ya que entender los fundamentos y bases detrás de la programación es clave para avanzar hacia experiencias más complejas.

Pseudocodigo

```
main() {
        saludar= mostrar un saludo al usuario
        leerInput= leer input del usuario
        mostrarInput = Muestra el input que ingreso el usuario

        extraerMensaje= extraer mensaje de input ingresado
        extraerClave= extraer clave de input ingresado
        extraerOpcion= extraer opción de input ingresado
        extraerParidad= extrae la paridad ingresada por el usuario si corresponde

        mostrarMensaje= mostrar solamente el mensaje
        mostrarClave= mostrar solamente la clave
        mostrarOpcion= mostrar solamente la opcion
```

```
opcionElegida: { /*comparar la opcion con una c o una d

if opcion=='c'
        asciiAEntero= Convertir la clave de caracter a decimal
        mostrarMensajeProc= codificar el mensaje y contar las letras procesadas
        bitDeParidad = guardar y mostrar un 0 o un 1 dependiendo de la cantidad de letras procesadas
        cantidadLetrasProc= guardar la cantidad de letras procesadas en forma de caracter
        Imprimir mostrarMensajeProc
        Imprimir bitDeParidad
        Imprimir cantidadLetrasProc
        finPrograma

if opcion=='d'
        mostrarMensajeProc= decodificar el mensaje y contar las letras procesadas
        cantidadLetrasProc= guardar la cantidad de letras procesadas en forma de caracter
        paridadAEntero= convierte la paridad ingresada por el usuario en un decimal
        bitDeParidadDeco = guardar un 0 o un 1 dependiendo de la cantidad de letras procesadas

        if paridadAEntero == bitDeParidadDeco
                Imprimir mostrarMensajeProc
                Imprimir cantidadLetrasProc
        else mostrarErrorParidad
        finPrograma
}
}
```