UNIVERSIDAD SIMÓN BOLÍVAR DEPARTAMENTO DE COMPUTACION Y DE TECNOLOGIA DE INFORMACION

CI3815: ORGANIZACIÓN DEL COMPUTADOR

Septiembre – Diciembre 2025

INFORME DE LABORATORIO: PROYECTO 1

Profesor:

Mora Fernando

Autores:

Renata Colon, **Carnet:** 18-10649

Ricardo García, **Carnet:** 20-10274

Sartenejas, 21 de noviembre de 2025

INTRODUCCION

Con el objetivo de desarrollar habilidades en el manejo de MARS, se solicitó implementar un programa en MIPS que funcione como una caja registradora sencilla. Este sistema debe ser capaz de generar facturas según el producto que el cliente desee comprar y la cantidad indicada, mostrando el total de la compra en dólares. Además, al finalizar la jornada, debe informar el dinero recaudado y los productos restantes en el inventario.

El desarrollo de este proyecto no solo busca afianzar el conocimiento en programación en ensamblador, sino también poner en práctica conceptos fundamentales de organización y arquitectura de computadores, tales como el uso de estructuras de datos dinámicas, la gestión de memoria, el control de registros y el manejo de operaciones aritméticas en punto flotante. Asimismo, se pretende simular un escenario real de aplicación, en el que la interacción entre el usuario y el sistema requiere validar entradas, actualizar inventarios y mantener un registro coherente de las transacciones realizadas.

En este informe se declaran las decisiones tomadas para lograrlo, las estructuras adicionales utilizadas, las dificultades encontradas y los aspectos que no pudieron mejorarse o no dieron el resultado esperado. También se reflexiona sobre las limitaciones del diseño y las posibles mejoras que podrían implementarse en futuras versiones, con el fin de optimizar tanto la eficiencia del programa como la claridad de su salida en la terminal.

Diseño de la estructura de datos para almacenar el inventario y la compra

El inventario se carga mediante el comando .include, que permite importar un archivo externo. Los productos se almacenan en un arreglo de etiquetas, cada una con sus atributos: código, stock, parte entera del precio, parte decimal y nombre. La principal limitación de este diseño es que, para verificar la existencia de un producto, se debe recorrer toda la estructura. En el peor de los casos, la búsqueda es de complejidad O(n), donde n es 14 (el número total de productos definidos). Si el inventario creciera demasiado, ya que si es muy grande, se pueden tener problemas a la hora de cargar un producto.

Para registrar las compras del día se implementó una lista enlazada, donde cada nodo creado en tiempo de ejecución ocupa 16 bytes y representa un producto adquirido. Cada nodo está conformado por:

- Precio**:** un valor en punto flotante calculado a partir de la parte entera y decimal del precio en la etiqueta del producto. Se convierte cada número entero a float, se divide la parte decimal entre 100 y se suma a la parte entera para obtener el precio real.
- Nombre**:** un puntero al string que identifica el producto.
- Código**:** el primer word de la etiqueta, que representa el identificador del producto.
- Next**:** un puntero al siguiente nodo; inicialmente es cero.

La primera compra del usuario establece el puntero head en el primer nodo, que al inicio coincide con tail y también con medio. Este último se utiliza para señalar el inicio de cada compra, lo que permite recorrer la lista desde medio hasta tail para calcular el total de la compra actual, y desde head hasta tail para obtener el acumulado de todas las compras del día.

Dificultades Encontradas en la implementación u otras decisiones posteriores

El uso de MARS exige un manejo cuidadoso de los registros. En varias ocasiones se sobrescribieron registros que no debían cambiar, ya que eran necesarios en otras funciones. Esto obligó a revisar repetidamente el código, pues aunque inicialmente funcionaba, al avanzar en las pruebas aparecían fallos inesperados.

Tal como crear un bucle en el main que siempre pidiera datos por el terminal guardando en un buffer que tuviera suficiente espacio por si el codigo dado era muy largo, ya que las operaciones que se pueden hacer en la caja registradora no son mas largas que estos valores.

También fue necesario replantear varias veces la implementación (puede haber indicios de esto en el programa) principalmente en la elección de estructuras y en el desarrollo de las funciones de la caja registradora:

- Multiplicación: Se necesitaba convertir el precio en flotante y almacenarlo de alguna forma para siempre tenerlo a disposición, luego el valor de n pasado al hacer *n es convertido en entero para luego ser

un float y realizar la multiplicación entre n y el precio para finalmente colocarse como el nuevo precio que tiene el nodo, ya que después como se recorre la lista necesitaremos el verdadero total. En esta función también se procura que dependiendo de n se reste del inventario la cantidad señalada.

- Suma: Al ingresar “+”, se suman los precios desde el nodo señalado por medio hasta tail, mostrando el mensaje “Total compra:” con el monto que debe pagar el cliente. Inicialmente no se había previsto el puntero medio, pero su incorporación permitió separar las compras y calcular correctamente el total.
- Resta: Se simula una liberacion de nodos de la lista dependiendo de la cantidad ingresada n en -n eliminado a partir de tail hacia atrás permitiendo que el anterior a tail ya no apunte a este sino a cero y esto hace que no sea tomado en cuenta a la hora de sumar el total de la compra, ademas se suma al stock correspondiente el producto que anteriormente había sido eliminado o reducido del inventario.
- División: La operación de división consiste en recorrer la lista enlazada y acumular el valor de cada nodo para mostrar al usuario el total correspondiente. Además, se presenta el estado del inventario con los productos que fueron adquiridos durante la jornada y, finalmente, se procede a cerrar el programa. Sin embargo, en esta funcionalidad persisten errores relacionados con la reducción del inventario: por cada código ingresado en la terminal se descuenta automáticamente una unidad del stock en el arreglo de productos, y posteriormente se vuelve a restar la cantidad indicada en la operación *n. Esto provoca que el stock no se reduzca en la cantidad exacta solicitada, sino en n+1, generando inconsistencias en el control de existencias.
- s: Es un comando que puede mostrar el inventario actual almacenado en el programa, sin necesidad de esperar por colocar “/” y cerrar el programa en consecuencia.

Otra de las dificultades encontradas es la salida en la terminal, ya que no se imprimen con un espacio intermedio que haga que se vea perfectamente alineado, sino que dependiendo de un mismo espaciado se imprime cada linea y aspecto especifico requerido para la presentación visual.

Conclusiones

Trabajar en MARS requiere atención constante a los detalles: verificar que los registros no se sobrescriban indebidamente, asegurarse de que cada instrucción sea la adecuada y comprender la organización de las estructuras en memoria (.space, .word, .asciiz). Es fundamental contar con el manual de referencia para resolver problemas durante la implementación.

A pesar de estos cuidados, el programa aún presenta aspectos por mejorar:

- Ajustar la salida para que los precios y nombres se impriman alineados en columnas.
- Corregir el manejo del stock para que al usar *n se reduzcan exactamente n unidades, y no n+1.
- Facilitar la ampliación o reducción del inventario, permitiendo modificar mas fácilmente el tamaño del arreglo de productos y su referencia en el código.
- Se podría implementar en vez de un arreglo para almacenar y manipular el inventario una tabla de Hash, donde la clave sea el código del producto y las demás características sus valores, pero aunque fue considerado se complicó por la creación total de ello y su uso.

En conclusión, aunque el programa todavía tiene detalles por perfeccionar, cumple con las funciones principales: calcular el precio total de las compras y mostrar el dinero recaudado al cierre del día. Esto lo convierte en una herramienta útil para simular el funcionamiento básico de una caja registradora.