LABORATORIO

ASEMBLER 8086

LABORATORIO DE ARQUITECTURA DE COMPUTADORAS

Curso de Arquitectura de computadoras

Instituto de Computación

Facultad de Ingeniería

OBJETIVOS

El presente trabajo obligatorio tiene como objetivo introducir al estudiante al conocimiento práctico de la arquitectura 8086. Para ello, este trabajo propone el desarrollo de una calculadora con entrada en notación polaca inversa (*RPN* por sus siglas en inglés).

El laboratorio propone un acercamiento a la arquitectura 8086 desde su lenguaje assembler incluyendo la compilación manual desde C de estructuras de datos y control y la interacción con el sistema de entrada y salida (E/S) mediante la técnica de E/S programada.

Descipción de la tarea

Se desea construir un programa ArquiCalc que modela una sencilla calculadora RPN.

La notación polaca inversa es un método alternativo de introducción de datos en el cual para evaluar expresiones matemáticas primero se introducen los operandos y luego el operador. Por ejemplo la siguiente expresión algebráica **10 – 7** se escribe **10 7 –** y un ejemplo un poco más complejo, **14 / (3 + 4)** puede reescribirse en notación RPN así: **14 3 4 + /**.

Las calculadoras que emplean notación polaca inversa utilizan una pila (*stack*) para mantener los operandos en memoria hasta que sean requeridos por la siguiente operación. Una vez que los operandos se sacan de la pila (para operaciones binarias primero se quita el lado **derecho** y luego el **izquierdo**) se realiza la operación y el resultado se coloca en el tope de la pila.

Así en el ejemplo de la expresión **14 / (3 + 4)** y suponiendo que la pila inicialmente está vacía se introducen los tres operandos

4 3 14	<- Tope de la pila

Luego al procesar el operador **+** se retiran los operandos **4** y **3** de la pila, se computa 4+3 dando como resultado **7** y la pila queda de esta manera:

7 14	<- Tope de la pila

El siguiente operador a tomar es **/** por lo que se toman los operandos **7** (divisor) y **14** (dividendo) de la pila y se calcula la división dando como resultado **2** y la pila queda de esta manera:

2	<- Tope de la pila

Los usuarios de la calculadora pueden intercalar operandos y operadores de tal manera que el resultado de una expresión se utilice como operando en otra expresión. Por ejemplo si quiero que el resultado del ejemplo anterior se utilice como operando de la expresión ***resultado_anterior + 1*** basta con agregar **1 +** a la entrada de datos anterior, quedando la secuencia **14 3 4 + / 1 +**. Luego de evaluar hasta la división lo siguiente es agregar el valor 1 al tope de la pila:

1 2	<- Tope de la pila

Y al procesar el operador **+** se toman los operandos y se calcula 1 + 2 dando como resultado **3** que queda en la pila

3	<- Tope de la pila

Características de la ArquiCalc

La calculadora a implementar deberá tener la siguiente funcionalidad:

- Recibir la secuencia de entrada de acuerdo al formato especificado más adelante en la subsección **Formato de entrada**
- Manejar una pila de trabajo de hasta **31** operandos enteros de 16 bits representados en complemento a 2
- Evaluar las siguientes operaciones aritméticas binarias: +, -, *, /, %
- Evaluar las siguientes operaciones booleanas binarias: &, |, <<, >>
- Evaluar las siguientes operaciones aritméticas unarias: NEG, FACT
- Exponer el tope de la pila en un puerto seleccionado: Top
- Realizar un volcado de la pila en un puerto seleccionado: Dump
- Duplicar el tope de la pila: DUP
- Intercambiar el tope de la pila con el elemento debajo de él: SWAP
- Sumar todos los elementos de la pila: SUM

Bitácora de ejecución

La calculadora debe mantener una bitácora de ejecución a medida que va procesando cada comando. Se utilizará el puerto de salida definible por el usuario a tales efectos que deberá funcionar de la siguiente forma:

- Antes de procesar un comando se debe mandar el código 0 seguido del comando a procesar (incluyendo los parámetros, una palabra por cada dato)
- Luego de procesar el comando se deberá mandar:
    - El código 16 si la operación se pudo realizar con éxito
    - el código 8 si la operación falló por falta de operandos en la pila
    - el código 4 si la operación fallo por desbordamiento de la pila (ya hay 31 operandos)
    - el código 2 si no se reconoce el comando (comando inválido)

El puerto por defecto para la bitácora debe estar definido en la constante **PUERTO_LOG_DEFECTO**

Formato de entrada

La entrada al sistema se realizará leyendo el puerto de entrada/salida de 16 bits de solo lectura ENTRADA con el formato ***Comando [Parámetro]*** dónde cada comando tiene predefinidos si requiere o no un parámetro. Tanto los comandos como los parámetros son de 16 bits.

La siguiente tabla muestra la codificación requerida para cada comando:

Comando	Parámetro	Codigo	Descripción
Num	Número	1	Agrega Numero a la pila
Port	Puerto	2	Setea el puerto de salida
Log	Puerto	3	Setea el puerto de la bitácora
Top		4	Muestra el tope de la pila en el puerto de salida (no retira del la pila)
Dump		5	Realiza un volcado de la pila en el puerto de salida (no retira de la pila)
DUP		6	Duplica el tope de la pila
SWAP		7	Intercambia tope de la pila con el elemento debajo
Neg		8	Calcula el opuesto
Fact		9	Calcula el factorial
Sum		10	Calcula la suma de todos los elementos de la pila (borrando la pila) y deja el resultado en el tope de la pila. Si no hay elementos deja 0 en el tope.
+,-,*,/,%,&,|,<<,>>		11 a 19	Realiza operación binaria
Clear		254	Borra todo el contenido de la pila
Halt		255	Detiene el procesamiento

Observaciones: +, -, *, /, % realizan operaciones enteras. & es el *and* bit a bit, | es el *or* bit a bit, << es el desplazamiento a la izquierda y >> desplazamiento a la derecha.

Todas las operaciones aritméticas (incluyendo la multiplicación) son de 16 bits y dejan como resultado en el tope de la pila los 16 bits menos significativos de la operación.

El comando Top muestra el tope de la pila en el puerto de salida actual. Esta operación no quita el operando del tope de la pila. El puerto de salida por defecto estará dado por una constante **PUERTO_SALIDA_DEFECTO** y puede ser cambiado con el comando Port.

Cada operación realizada deja registro en el puerto de la bitácora que puede ser cambiado mediante el comando Log.

Cuando se desea realizar una operación binaria pero solamente se encuentra 1 elemento en la pila (el operando derecho), además de dejar constancia en la bitácora se debe eliminar el elemento de la pila (quedará vacía).

El comando Dump realiza un volcado de la pila en el puerto de salida comenzando desde el tope de la misma. Esta operación no modifica la pila.

El comando Clear elimina todo el contenido de la pila.

El comando Halt detiene el procesamiento de la calculadora (se deja de leer el puerto de ENTRADA).

La implementación del factorial (utilizada en el comando Fact) debe ser una rutina recursiva.

Inicialmente la pila debe estar vacía.

Para el caso del ejemplo (calcular **14 / (3 + 4)** y luego a ese resultado sumarle **1**), visto en la sección anterior, la secuencia de entrada en el puerto ENTRADA suponiendo que se quiere mostrar el resultado final en el puerto 15 podría ser la siguiente: **2 15 1 14 1 3 1 4 11 14 1 1 11 4 255.**

Breve descripción del procesamiento de esa secuencia teniendo en cuenta el siguiente formato para la pila: { tope pila, siguiente, …, base de la pila }:

Comando (y parámetro)	Descripción	Comando (y parámetro)	Descripción
2 15	Se setea el puerto de salida	14	Se toma 7 y 14 de la pila, se realiza 14 / 7 y se agrega 2 al tope de la pila { 2 }
1 14	Se agrega 14 al tope de la pila { 14 }	1 1	Se agrega 1 al tope de la pila { 1, 2 }
1 3	Se agrega 3 al tope de la pila { 3, 14 }	11	Se toma 1 y 2 de la pila, se suman y se agrega 3 al tope de la pila { 3 }
1 4	Se agrega 4 al tope de la pila { 4, 3, 14 }	4	Se muestra 3 en el puerto 14 { 3 }
11	Se toma 4 y 3 de la pila, se suman y se agrega 7 al tope de la pila { 7, 14 }	255	Se deja de leer la entrada

Herramientas

Se utiliza un ambiente simulado como plataforma de programación con el objetivo de minimizar las dependencias con el sistema operativo y el sistema computador, y fomentar la comprensión de los elementos y mecanismos de hardware de la arquitectura 8086. En 2014 la Facultad de Ingeniería se propone el desarrollo de un simulador como herramienta educativa para la enseñanza de la arquitectura 8086, iniciando así el desarrollo del simulador ArquiSim [1]. El simulador (ArquiSim) permite el desarrollo de software en ensamblador, el ensamblado del mismo y la simulación de la ejecución sobre una arquitectura 8086. Para el desarrollo de software el simulador propone estructurar el código en cuatro tipos de secciones: una sección para definir datos, una sección para código de usuario, una sección para interrupciones y una sección para interactuar con entrada/salida. Para la presente entrega no será necesario configurar la sección de interrupciones.

El simulador se distribuye como un paquete java (archivo jar) y se recomienda usar JRE 8 para ejecutarlo.

Información sobre la entrega

Aplicación

La aplicación a desarrollar debe ser compatible con el simulador ArquiSim v1.3.7. Se deberá entregar un programa escrito en C que modele el problema y un archivo en lenguaje ensamblador que lo implemente.

Para evaluar el funcionamiento de la aplicación debe establecerse la sección ports con la secuencia de comandos a ejecutar. En la sección de laboratorio del EVA podrán encontrar varios casos de prueba presentados como secciones ports. Para evaluar el correcto funcionamiento y el desempeño del sistema desarrollado se debe incorporar la sección ports al resto de las secciones (code, data). Ver el manual de usuario de Arquisim [1] para más información sobre el manejo de secciones.

El número de puerto de E/S **ENTRADA** y los números de puerto por defecto **PUERTO_SALIDA_DEFECTO** y **PUERTO_LOG_DEFECTO** deberán definirse mediante una constante (*EQU*) en el cabezal del programa.

Forma de entrega

La entrega de la tarea debe realizase a través del formulario habilitado en la plataforma EVA para dicho fin. La entrega del obligatorio debe ser un archivo empaquetado de nombre obligatorio.tar.gz u obligatorio.zip que debe contener los siguientes archivos:

1. un archivo C de nombre “obligatorio.c” con la implementación de alto nivel de programa
2. un archivo fuente de nombre “obligatorio.asm”, donde se considerarán los comentarios que aparezcan con el fin de facilitar la comprensión del mismo. Este archivo fuente contendrá el código assembler del programa.
3. un documento de nombre “obligatorio.pdf” con la documentación sobre la implementación.

A continuación se presentan un conjunto de pautas que podrán ser consideradas al momento de elaborar el informe:

1. Carátula (Título, datos del estudiante, etc)
2. Índice
3. Descripción del problema
4. Descripción de la solución incluyendo detalle de las estructuras de datos y constantes utilizadas
5. Experimentación y problemas encontrados
6. Conclusiones
7. Mejoras a futuro
8. Referencias

Una clara, concisa y descriptiva documentación es clave para la evaluación de los trabajos entregados. No deje la documentación para lo último.

Los trabajos deberán ser entregados indefectiblemente antes del domingo 6 de noviembre a las 23.30 horas. No se aceptará ningún trabajo pasada la citada fecha y hora. En particular, no se aceptarán trabajos enviados por correo electrónico a los docentes del curso, ni entregados en medios físicos en el instituto. El sistema de entregas soporta múltiples entregas. Pruebe de realizar una entrega con tiempo, a los efectos de verificar que su sistema le permite entregar correctamente.

Bibliografía

[1] ArquiSim: manual de usuario. https://eva.fing.edu.uy/mod/resource/view.php?id=61253. Accedido: 12-10-2021.