Prácticas de Lenguaje Máquina

E. Montseny, S. Romani y D. Puig
Universitat Rovira i Virgili
Dep. Enginyeria Informàtica
Ctra Salou s/n 43006 Tarragona
{emontsen, sromani, dpuig}@etse.urv.es

Resumen. *En este trabajo se presentan un conjunto de prácticas diseñadas para que el alumno alcance un buen conocimiento del nivel de L.M. y entienda como se ejecuta un programa escrito en un lenguaje de alto nivel en un ordenador. En estas prácticas el alumno debe implementar un programa en L.M. a partir de un algoritmo dado. La principal novedad que se presentan es la metodología, que se basa en: 1) cada ejercicio se realiza en una sesión de 2 horas; 2) al principio de cada sesión, el profesor de prácticas repasa (en 30 minutos) los conceptos teóricos que se aplican en el ejercicio; 3) cada ejercicio se puede realizar independientemente de los anteriores.*

Introducción

Las prácticas de lenguaje máquina pertenecen a una asignatura de primer curso, que puede impartirse en el primer o segundo cuatrimestre. Esta asignatura tiene como requisito previo, que el alumno tenga conocimientos de la Máquina Sencilla o una máquina pedagógica equivalente.

El temario de esta asignatura, que esta dirigido tanto a alumnos de ingeniería técnica informática de sistemas, ingeniería técnica de informática de gestión, e ingeniería informática, es el siguiente:

Programa de la asignatura

1. Estructura de los L.M.
   - Introducción
   - Registros L.M.
   - Modos de direccionamiento
   - Instrucciones
   - Interrupciones
2. Programación en L.M.
   - Tipos de datos
   - Estructuras algorítmicas

Lenguaje ensamblador

3. Arquitectura del I-8086
   - Introducción al I-8086
   - Registros
   - Modos de direccionamiento
   - Instrucciones
4. Subrutinas
   - Tipos de subrutinas
   - Paso de parámetros
   - Retorno de resultado
   - Bloque de activación
5. Entrada y Salida
   - Subsistemas de E/S
   - Sincronización de las operaciones de E/S
6. Lenguaje ensamblador

El primer objetivo de esta asignatura es que el alumno alcance un buen conocimiento del nivel de L.M. del ordenador. De este modo, en el tema 1 se le introducen todos los elementos básicos que caracterizan un L.M. y, en el tema 5, los relacionados con E/S. Cada uno de estos elementos se relaciona con los mecanismos del nivel de hardware que los hacen posible.

El segundo objetivo de la asignatura es conseguir que el alumno se forme una idea de la apariencia que tendrá, en L.M., su programa escrito en un lenguaje de alto nivel. De este modo, en el tema 2 se introduce la implementación de los distintos tipos de datos y estructuras algorítmicas, mientras que en el tema 4 se introducen los conceptos relacionados con la implementación de subrutinas.

La ordenación de los temas de esta asignatura obedece a la necesidad de introducir lo antes posible los conceptos imprescindibles para que el alumno pueda empezar las clases prácticas. De este modo, en el tema 3 se explica un L.M. ejemplo con el que se realizan los ejercicios.

Esta asignatura se imparte en 2 horas de teoría y 2 horas de práctica a la semana. Durante las 3 primeras semanas solo se imparten clases de teoría (4 horas a la semana).

Programa de prácticas

El objetivo de las prácticas es conseguir que el alumno refuerce los conceptos que se le han introducido en las clases de teoría. Para ello, durante las clases prácticas, el alumno debe implementar en L.M. un programa que le obligue a utilizar los diversos mecanismos explicados en clase de teoría.

Los ejercicios que se proponen en las 10 clases prácticas abarcan los conceptos siguientes:

1 - Introducción al entorno del L.M. del I-8086: TurboDebugger y Norton Guides.

Se presenta la interficie que ofrece el TD y se identifican los distintos elementos (registros, flags, memoria) que hay que conocer desde el punto de vista del nivel de LM. En concreto se

utiliza el TD como banco de pruebas para introducir, ensamblar y ejecutar instrucciones o pequeños programas directamente.
Se estudia cuáles son los registros del i8086 y cómo se puede acceder a cada uno de ellos, el significado de los flags, y la segmentación de memoria en una zona de código, otra de datos y otra para la pila.
También se explica la utilidad de las NG como manual de consulta rápida de las características del i8086.

2.- Modos de direccionamiento e instrucciones de salto.

Se utiliza el TD para repasar los modos de direccionamiento del i8086 introduciendo directamente diversos ejemplos representativos de cada uno de ellos. También se ven ejemplos que ilustran una utilización incorrecta de los modos de direccionamiento debido a errores conceptuales que puedan tener los alumnos, o a limitaciones físicas del procesador.
Con ayuda del TD se estudian las instrucciones de salto analizando que condiciones se evalúan, es decir, que flags se tienen en cuenta para realizar o no el salto.

3.- Programación de estructura algorítmicas. Introducción al proceso de edición, ensamblado y montaje de un programa en L.M.

Se implementan los primeros programas sencillos en lenguaje ensamblador utilizando tres programas básicos para la programación en L.M.: un editor, un ensamblador y un montador.
Se hace énfasis en la necesidad de programar de forma estructurada y para ello se practica la implementación de estructuras algorítmicas en lenguaje ensamblador siguiendo las pautas que se han dado en las clases de teoría. Concretamente se realiza un programa que sustituye cada aparición de un determinado elemento de un vector por otro elemento.

4.- Programación de algoritmos que utilizan datos estructurados: acceso a matrices.

Siguiendo con los criterios de programación introducidos en la sesión anterior, en esta sesión se implementa un programa que realiza el producto de dos matrices, con lo que se repasan nociones de implementación de estructuras algorítmicas, así como de acceso a tipos de datos estructurados.

5.- Introducción a las subrutinas: utilización de la pila y concepto de programa de pruebas.

En primer lugar, utilizando el TD, se repasa el concepto de pila: instrucciones de acceso, evolución del puntero a la pila, evolución del contenido de memoria de pila, y registros de acceso a la pila.
A continuación, se implementa una subrutina para calcular el máximo común múltiplo de dos números, realizando el paso de parámetros y la devolución de resultados a través de registros.
En este ejemplo se utiliza la pila para guardar la dirección de retorno de la subrutina y para salvar los registros que utiliza la subrutina.

6.- Utilización del bloque de activación: paso de parámetros.

Se realiza un ejercicio que consiste en la implementación de una subrutina para mostrar un string por pantalla.
En primer lugar se construye el bloque de activación para esta subrutina identificando cada uno de sus componentes. Se hace énfasis en la necesidad de realizar correctamente el paso de

parámetros y cual es el mecanismo más adecuado para la realización del mismo dependiendo del tipo de parámetro.
Por último se realiza un pequeño programa de prueba para evaluar el funcionamiento de la subrutina.

7.- Utilización del bloque de activación: retorno del resultado.

En esta sesión se implementa un subrutina para leer un string.
En primer lugar se construye el bloque de activación para esta subrutina identificando cada uno de sus componentes. Se hace énfasis en la devolución de resultados de la subrutina y cual es el mecanismo más adecuado para la misma.
Por último se realiza un pequeño programa principal de prueba que evalúa el funcionamiento de la subrutina en diversos casos.

8.- Subrutina de acceso a la pantalla.

En primer lugar de explica de forma breve el mecanismo de acceso a pantalla en modo texto. A continuación se implementa una subrutina para escribir un carácter por pantalla. Se realiza también un pequeño programa de prueba que escribe un rectángulo por pantalla formado por asteriscos.

9.- Subrutina de acceso al teclado: sincronización por encuesta.

Se explica brevemente el funcionamiento del controlador de teclado y cuáles son los puertos de E/S que es necesario utilizar para realizar la sincronización por encuesta con el teclado. A continuación se implementa una subrutina para leer un carácter desde teclado. Por último se realiza un programa para probar esta subrutina.

10.-Implementación de una RSI de reloj: sincronización por interrupción.

Se explica brevemente el funcionamiento del timer y cuáles son los puertos de E/S que es necesario utilizar para realizar sincronización por interrupción con el timer. Se implementa un rutina de servicio a la interrupción de reloj que permita contar tiempo en minutos y segundos. Se realiza un programa de prueba que realiza una tarea independiente, mientras que la RSI de reloj escribe por pantalla el valor del contador de tiempo cada vez que pasa un segundo.

Metodología

Las clases prácticas se realizan en grupos de 50 alumnos, los cuáles disponen de un ordenador por cada dos personas. Hay dos profesores por clase, uno de ellos realiza una introducción a la práctica y el otro apoya en la atención a las consultas de los alumnos. Esta introducción tienen el propósito de refrescar los conceptos teóricos que se utilizaran en esa clase, explicar en detalle el programa a realizar y el entorno de trabajo (herramientas, rutinas de apoyo, etc.) y recalcar los puntos donde se debe aplicar la teoría en el ejercicio práctico de la sesión. Dichas explicaciones no superaran los 30 minutos, puesto que el tiempo restante de las 2 horas de la clase es necesario para que los alumnos realizan el ejercicio propuesto.

La idea principal que promovió estas prácticas es el hecho de que el alumno pueda verificar por si mismo si sabe poner en práctica los conceptos explicados en clase de teoría. Así, en el momento de plantear la solución le asaltaran múltiples dudas, algunas relacionadas con la parte conceptual y otras simplemente con el entorno de trabajo. Los profesores resolverán cualquier duda sobre el entorno, pero solo guiaran al alumno en la resolución de las primeras. Una vez terminado el programa, ensamblado y listo para ejecutar, seguramente surgirán diversos errores de funcionamiento. Los profesores ayudaran al alumno a buscar técnicas para descubrir esos errores, y verificar el correcto funcionamiento del programa para múltiples casos (es decir, en el diseño de un juego de pruebas). De este modo, el alumno se de cuenta los conceptos que no ha llegado a comprender, como consecuencia directa de observar si su solución funciona o no funciona. Esto debería suponer la principal fuente de motivación para el alumno.

Además del correcto funcionamiento del programa, se exige que el diseño se estructurado, que el programa esté bien comentado y que se realizan las pruebas necesarias. Es interesante hacer observar a los alumnos las ventajas que esto supone.

Una vez que un grupo ha concluido su ejercicio se procede a la evaluación del mismo, teniendo en cuenta, además de su correcto funcionamiento, los puntos expresados en el párrafo anterior. Uno de los principales objetivos de la evaluación, en clase, de los ejercicios es conseguir un mejor aprovechamiento de la clase por parte del alumno.
Esta información influirá en la nota final de cada alumno, en función del numero de ejercicios finalizados correctamente. En tanto que estos ejercicios se puedan corregir con mayor rigor permitirían una aproximación a la evaluación continuada. El porcentaje en la nota final de la asignatura puede ser mínimo, pero hemos observado que, a pesar de ello, supone un gran aliciente para la asistencia a estas clases.

Para completar la evaluación del nivel práctico, los alumnos tienen que entregar a final de curso un programa mas complejo de los vistos en clase, pero que abarca todos los conceptos de interés para la asignatura. Las rutinas de E/S que se proponen en las clases prácticas están pensadas para que se reutilizen en el trabajo final. Los alumnos están advertidos de ello, y se les propone que agrupen en un módulo separado todas estas rutinas. Así se promueve el diseño modular de programas, la compilación separada, y la reutilización de código.

Como consecuencia positiva observable es que los alumnos se familiarizan con el entorno del lenguaje máquina, es decir, instrucciones, modos de direccionamiento, subrutinas, proceso de ensamblado y montaje de programas, proceso de detección de errores, etc. Así, una vez realizadas las prácticas, no presenta grandes dificultades escribir, ensamblar y probar un programa en L.M., y no son necesarias

las clases de problemas puesto que han realizado los ejercicios suficientes para alcanzar un buen conocimiento del nivel de L.M.

Resultados y conclusiones

Este método de impartir las prácticas de la asignatura de Computadores II (Lenguaje Máquina) ha sido probado durante dos cursos. Al final del segundo curso se verificó un incremento notable en:

- La asistencia de los alumnos a las clases prácticas.
- El número de aprobados (de 100 a 160).

Después de la experiencia en la aplicación de este método se puede concluir que la evaluación de las prácticas en clase:

- Motiva enormemente a los alumnos.
- Permite realizar un mejor seguimiento de cada alumno.
- Ayuda a mejorar el nivel de la asignatura y el rendimiento del alumno.