DESARROLLO E IMPLEMENTACION DE UN GENERADOR DE ENSAMBLADORES

Ingeniero Iván E. Ramos C.

Universidad del Valle - Depto. de Electricidad
A.A. 25360 Cali - Colombia

RESUMEN

El presente trabajo es el resultado de investigaciones en el desarrollo y dominio de Herramientas de Hardware y Software, como parte integral del trabajo en la Línea de Investigación en Arquitectura y Concepción de Sistemas Computacionales, definido por el Departamento de Electricidad como una de sus áreas de interés.

Este trabajo se deriva de los resultados obtenidos en el proyecto ***"Desarrollo del Algoritmo Optimo que permita Generar Código Objeto a partir de un Programa Escrito en Lenguaje Ensamblador"*** (Proyecto XASM), primero que abordó este objeto y cuyo resultado es un Algoritmo Optimo de Ensamblaje (presentado en el V Encuentro Nacional de Informática Universitaria). La investigación se relaciona con la posibilidad de desarrollo de una herramienta universal que permita generar, en forma automática, ensambladores cruzados, y que sea aplicable tanto a procesadores y coprocesadores comercialmente disponibles como también a aquellos "armados, como sucede con los del tipo Bit-Slice".

Los avances en la tecnología digital han permitido el reemplazo masivo de sistemas analógicos por sistemas basados en microprocesadores, y también el que los nuevos sistemas de mando y regulación se conciban ya desde una tecnología digital con el microprocesador como elemento fundamental.

El desarrollo de aplicaciones basadas en microprocesadores requiere herramientas lo suficientemente ágiles y flexibles que permitan la competencia y adaptabilidad de estos sistemas a los nuevos adelantos. No es técnico ni práctico el desarrollar aplicaciones sobre el hardware mismo; los recursos son limitados y la tarea se vuelve tediosa y de bajo rendimiento. La alternativa es el desarrollo de "Talleres de Software" en los cuales podamos simular y evaluar las aplicaciones antes de su implementación, y así disponer de grados de confiabilidad y rentabilidad apreciables.

Adicionalmente, cada día se producen nuevos microprocesadores con arquitecturas diferentes como resultado de nuevos adelantos comerciales y también como prototipos de investigación y desarrollo. Estos crecimientos solamente son prácticos cuando se dispone de las herramientas de diseño apropiadas que faciliten su empleo en aplicaciones. Estas herramientas están disponibles comercialmente y se las conoce como Meta-ensambladores, que son Paquetes de Software costosos, difíciles de manejar y de acceso restringido, precisamente por los niveles de competencia que la industria de la computación ha impuesto.

El resultado de este trabajo de investigación es un Generador Automático de Ensambladores Cruzados, que a partir de una descripción del procesador produce el Ensamblador que permitirá el desarrollo de las aplicaciones. Este ensamblador responde no solamente a las características propias del procesador sino que también utiliza los nemónicos definidos por el fabricante.

Esta temática es de actualidad en el avance de aplicaciones que, dado el nivel de estandarización del hardware y la alta disponibilidad de éste, requieren adelantos apreciables en software. Prácticamente, el problema del desarrollo de aplicaciones es, en alto porcentaje, un problema de software y hacia allá apunta este trabajo, a contribuir a su solución.

INTRODUCCION

No se puede negar el papel que ha jugado el computador en el desarrollo de las ciencias donde, de no haber sido por el gran auge de los sitemas computacionales, sus niveles de conocimiento no habrían avanzado a los estadios actuales. Es que el computador le ha dado a la ciencia una tercera forma de interrogación, adicionalmente a la observación y la experimentación: la simulación.

Esta verdad no le es ajena tampoco a la investigación y desarrollo en los sistemas computacionales. Hoy, antes de construir físicamente un sistema, ya se han hecho similitudes, en condiciones muy cercanas a la realidad, gracias a elaborados algoritmos de simulación, a avances en las interfases gráficas y a disponibilidad de sistemas de alta velocidad que permiten evaluar estos algoritmos en tiempos reducidos y, por tanto, la realización de una simulación de estos en condiciones de tiempo real.

El investigador, al igual que el desarrollador de aplicaciones, requiere herramientas ágiles y confiables para su trabajo, tanto a nivel de hardware (emuladores) como de software (simuladores, depuradores y ensambladores). Sin embargo, las herramientas para la producción de software no son de la misma naturaleza que las de hardware. En efecto, poseen, entre otras, dos características que les son propias:

- Necesitan un medio de cálculo, y con frecuencia el microprocesador considerado no está en capacidad de soportarlo.
- Son totalmente dependientes del procesador y, por consiguiente, cada procesador debe poseer sus propias herramientas de desarrollo.

La primera característica necesita los servicios de un computador de gran capacidad para soportar las herramientas de desarrollo; se dice entonces que el sistema de desarrollo es *cruzado*.

La segunda característica, asociada a hechos como la proliferación de nuevos procesadores y coprocesadores en el mercado, y que prácticamente es posible "armar" el procesador a partir de procesadores en pedazos (Bit-Slice), hace necesario automatizar el proceso de producción de nuevas herramientas de desarrollo, como por ejemplo: simuladores, depuradores y ensambladores.

Estas dos características son las que han impulsado el desarrollo de ambientes de programación independientes de la máquina, como base fundamental para el desarrollo de aplicaciones.

La situación presentada plantea dos problemas; el primero, relacionado con la ingeniería de base, que plantea la necesidad de desarrollar tantos ensambladores como microprocesadores existan, y el segundo, relacionado con los usuarios, quienes deben adaptarse, con cada cambio de microprocesador, a un nuevo ensamblador. Estos problemas se ven incrementados por la complejidad y magnitud crecientes, tanto de los microprocesadores como de las aplicaciones basadas en ellos, y también por la necesidad de disponer de métodos y herramientas (ayudas) para su desarrollo de implantación.

Este trabajo se ha orientado a encontrar una solución que, por un lado, facilite y automatice la producción de ensambladores y, por otro, defina una sintaxis general para los ensambladores que permita operar independiente del microprocesador, pero respetando las especificaciones del fabricante.

ANTECEDENTES

Un primer proyecto en este campo se relaciona con el desarrollo de un Ensamblador Cruzado. Concretamente, la investigación se enfocó en el "Desarrollo del Algoritmo Optimo para Generar Código Objeto a partir de Programas Escritos en Lenguaje Ensamblador". Se trabajó en la definición de un método y en la búsqueda de un algoritmo óptimo para el desarrollo de ensambladores cruzados. Se desarrollaron ensambladores para el Z80, para las familias 8048 y 8051. Los resultados son altamente positivos, no solamente desde el punto de vista del proceso de ensamblaje sino también en factores que se relacionan con el tiempo de ensamblaje, pues se consideran estudios ya realizados, respecto de la frecuencia de uso de las instrucciones (estos estudios, realizados en la Universidad de California, dieron origen a las arquitecturas RISC y CISC), para optimizar el proceso aumentando el rendimiento en el número de líneas ensambladas por minuto.

Los resultados de este primer proyecto son la base para continuar la investigación con el desarrollo de un *"generador de Ensambladores Cruzados"*, pensando en la concepción de un software que sea realmente una herramienta universal de desarrollo.

Todo sistema de software incluye al menos un ensamblador que traduce un programa fuente, escrito en Lenguaje Ensamblador, en código objeto ejecutable por la máquina. Todo nuevo procesador (de gestión, numérico, de entrada/salida, etc.) requiere, por consiguiente, la realización de un nuevo ensamblador. Las herramientas de producción automática facilitan mucho esta tarea, y reducen el tiempo necesario y el costo. Sin embargo, los generadores comerciales, denomiados Meta-Ensambladores, son costosos, difíciles de manejar y no son universales, es decir, se identifican con una determinada familia de microprocesadores.

En los Meta-Ensambladores comerciales es importante notar que la mayoría utiliza técnicas de traducción y no de ensamblaje, es decir, la generación del código ejecutable la realizan a partir de tablas de correspondencia nemónicas - código objeto. Este trabajo no corresponde a este tipo de ensamblaje sino que utilizará técnicas de ensamblaje, lo cual significa que el código binario se "ensambla" a partir de la descripción de la máquina y en términos de características particulares, como por ejemplo: número de acceso a bits, etc. Necesariamente este tipo de ensamblador es más eficiente en el uso de la memoria y requiere menos tiempo para generar el código, dado que éste se deduce, mientras que en el otro caso corresponde a una búsqueda, que puede ser larga si el nemónico está al final de la tabla.

LAS HERRAMIENTAS

Aun cuando en los actuales momentos se habla bastante acerca del auge de los lenguajes modernos, incluso los de Cuarta Generación (4GL), el Ensamblador es un lenguaje que se seguirá utilizando, dado su nivel de acercamiento al hardware específico de la máquina. Cuando se trata de aplicaciones industriales en tiempo real, aplicaciones en las cuales se programan directamente las interfases de entrada/salida y se debe acceder a direcciones específicas, el lenguaje utilizado es el de máquina, y más propiamente el ensamblador. Este factor determina una restricción, y es que los programas escritos en Lenguaje Ensamblador *no son independientes* de la máquina en la cual se ejecutarán, por ejemplo: tamaño de la memoria, dirección de los periféricos, velocidad del procesador, microprocesador base, y otros elementos propios de la arquitectura, sobre la cual se implantará la aplicación. Estos factores hacen necesaria la disposición de **Herramientas de Desarrollo** que permitan el avance del software de aplicación en máquinas diferentes de la que finalmente ejecutará el programa.

Existen dos tipos de herramientas: Residentes y Cruzadas.

Residente significa que las herramientas de desarrollo producirán código objeto que será ejecutado en el mismo procesador para el cual ellas mismas fueron escritas.

Por Cruzado se entiende que las herramientas corriendo en un procesador A producen código para un procesador B.

HIPOTESIS

El resultado de XASM, el algoritmo de ensamblaje, se probó para varios procesadores de la familia INTEL y se mostró que es posible la definición de un núcleo genérico, a partir del cual se haría crecer el ensamblador particular. Con base en lo anterior, y teniendo en cuenta que un programa escrito en Lenguaje Ensamblador sigue una sintaxis representada en la escritura de los operandos y en la estructura de las líneas fuente, es posible definir un Ensamblador Universal (que se acomode a cualquier microprocesador) constituido por unos elementos fijos y otros redefinibles por el usuario al momento de generar el ensamblador particular.

XGEN

El Generador Automático de Ensambladores (XGEN) está constituido por los siguientes dos elementos fundamentales:

Lenguaje de Descripción. Permite la descripción del procesador, de tal manera que pueda encadenarse con el motor del generador. Sus entradas son: Número de Bits, Número de Registros, Modos de Direccionamiento, Espacio de Memoria, Bits de Estado, Modos de Interrupción, etc.

Núcleo del Generador. Es un conjunto de programas y rutinas comunes a todo ensamblador. Esta base, encadenada con la descripción, produce el Ensamblador Cruzado de la máquina descrita.

La Figura No. 1 presenta un esquema de la organización del generador de ensambladores.

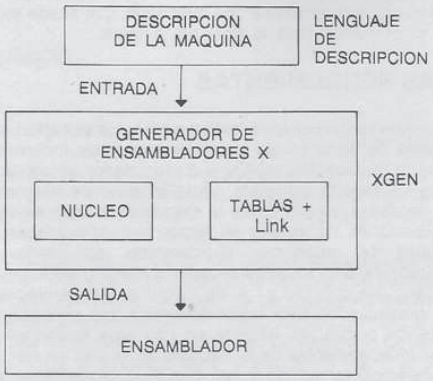


Figura No. 1 Estructura del Generador de Ensambladores

La descripción del procesador se hace por medio de un *Lenguaje de Descripción* que utiliza una herramienta proveniente de UNIX: Yacc (Yet Another Compiler-Compiler), que es un generador de analizadores sintácticos (parser). La entrada de Yacc son las reglas para controlar el proceso de entrada, código a ser invocado cuando estas reglas sean reconocidas y las rutinas de entrada básicas estén a bajo nivel. La salida es una función para controlar el proceso de entrada. Esta función es la que en compilación se denomina parser, es decir, el analizador sintáctico. Yacc genera el analizador en Lenguaje C.

El Generador de Ensambladores, tal como lo presenta la Figura No. 1, está constituido de un núcleo y de una parte complementaria y específica, generada por la descripción, y corresponde a Tablas. Adicionalmente existe un módulo de enlace *(Link)*, que es el encargado del encadenamiento con el sistema operacional, en este caso el MS-DOS, pero bien podría ser otro (Unix).

Para generar un Ensamblador particular lo que se hace es una compilación de los diferentes módulos, tanto del Núcleo como del Link, juntamente con la descripción generada por el analizador. Esta compilación genera un ejecutable que corresponde al ensamblador particular y de allí en adelante se utiliza este como un programa aplicativo independiente.

La Figura No. 2 presenta el procedimiento de operación con el Ensamblador generado por XGEN. La entrada es un programa fuente escrito en Ensamblador (Nemónicos) del procesador particular y la salida es un programa objeto (OP CODES).

Tomando como base el resultado de Yacc, y por razones de compatibilidad y portabilidad, el software (módulos) que constituye XGEN, se codificó en Lenguaje C.

Es importante puntualizar las razones para el desarrollo del software en Lenguaje C:

-C incluye las ventajas de un Lenguaje de Alto Nivel, lo cual permite la escritura de programas más claros y con menor posibilidad de errores, además el mantenimiento de los programas es simple.

-Con C se tiene la mayor cercanía al Lenguaje Ensamblador y al de Máquina y, a diferencia de éste, es más manejable y permite un mejor control sobre el código ejecutable.

-El código generado por los compiladores de C es rápido y eficiente. El C está siendo usado para programar aplicaciones que antes eran obligadas en Ensamblador y hoy día con mayor razón, dada la variedad y complejidad de los "Chips" disponibles.

-El lenguaje está disponible para todo tipo de máquina, desde los microcontroladores hasta los grandes equipos como los CRAYs. El fuente C necesita ser escrito una sola vez, con la sola compilación (con modificaciones menores) se puede transportar el código de una máquina a otra.

-C es un lenguaje moderno y estructurado y la técnica "Top-Down" resulta apropiada para el desarrollo de aplicaciones, debido precisamente a su estructuración y facilidades de modularización. Los procedimientos pueden complicarse separadamente y conformar una biblioteca propia.

-La comunicación entre C y Ensamblador es relativamente fácil, lo cual permite optimizar la implantación definitiva de una aplicación a nivel de una máquina particular (ejemplo: PC).



Figura No. 2 Proceso de Ensamblaje

CONCLUSIONES

El trabajo está en fase de verificación y se han generado ensambladores para procesadores tanto de 8 como de 16 bits y de familias diferentes (Motorola, Intel, Zilog, etc.) con el objeto de experimentar con diferentes arquitecturas y lenguajes ensambladores (de nemónicos). Los resultados son satisfactorios y se espera agotar esta fase del proyecto hacia una refinación del proceso de generación de ensambladores en forma más automática e interactiva, dado que, actualmente, la descripción es un archivo en disco y con las posibilidades de los sitemas computacionales actuales se podría obtener a partir de un diálogo con el usuario.

Complementario a lo anterior, la parte más compleja del sistema corresponde al lenguaje de descripción y al proceso de compilación. Realmente el éxito de la aplicación radica en que podamos comunicar la descripción del procesador, contando únicamente con la hoja de especificaciones que da el fabricante, es decir, que no se requiera un nivel de expertismo para poder hacer esta descripción. Esta es, posiblemente, la parte que más cuidado requiere.

Para complementar lo anterior se ha realizado un trabajo paralelo hacia la generación de programas fuente a partir de lenguajes Seudo-Código. Existen dos desarrollos: *intergen*, aplicación desarrollada en Basic y que genera programas Basic a partir de Seudo-Código castellano. Este resultado se presentó en el marco del V Encuentro de Informática Universitaria. Una versión mejorada, denominada *Ambrosia*, se implementó en Prolog y genera fuente en Basic para el microcontrolador 8052AH. Esta segunda versión es más flexible y nos ha permitido avanzar en el tratamiento del lenguaje con posibilidades importantes que se relacionan con ligar XGEN con AMBROSIA.

El Generador Automático es un resultado práctico esperado, pero adicionalmente es importante notar que el trabajar en el desarrollo de un Ensamblador Cruzado, luego de un Interpretador-Generador de Programas (Intergen), y después en la Generación Automática de Ensambladores para diferentes procesadores, es una experiencia importante pensando en que el ensamblador es la fase final de cualquier compilador y la interpretación una de sus primeras fases. En este orden de ideas, los logros de estos proyectos contribuirán al conocimiento sobre compilación, que es una área que la Universidad debe abordar si desea desarrollos propios en Informática y en el campo de los Lenguajes Formales.

Pensamos que en una etapa avanzada será posible disponer de un Ejecutivo de Tiempo Real para la programación de autómatas secuenciales, tal como se insinúa en la Figura No. 3.

BIBLIOGRAFIA

- GAGE, un Systeme de Production d'Assembleurs.
  H. Leroy et Y. Bekkers. IRISA, RENNES (France), 1983.
- Instruction Set and Beyond: Computers, Complexity, and Controversy.
  Colwell, Hitchock, Jensen, Spunt and Kollar.
  Computer, September 1985.
- Yacc, Yet Another Compiler-Compiler.
  Stephen C. Johnson. AT&T Bell Laboratories.
- Compilers. Principles, Techniques and Tools.
  Aho, Sethi, and Ullman.
  Addison-Wesley, 1986.
- Théorie des Langages et Compilation. Notes de Cours
  C. Boiret.
  Révision Octobre 1987. INPG Grenoble
- Ingeniería del Software. Un Enfoque Práctico. 2a. Edición
  Pressman Roger.
  McGraw-Hill Book. México 1988.
- Sistemas Operativos. Conceptos y Diseño.
  Milenkovic Milan.
  McGraw-Hill Book. Madrid, 1988.
- XASM: Cross-Ensamblador Z80
  Eduardo F. Calcedo B., Iván E. Ramos C.
  V Encuentro Nacional de Informática Universitaria.
  Popayán, julio de 1988
- Ambientes de Programación Independientes de la Máquina.
  Iván E. Ramos C. XI Congreso Nacional de ACIEM.

INSTRUMENTACIONES CIENTIFICAS LTDA.
Distribuciones Industriales - Equipos de Laboratorio

Representantes exclusivos de:

AGITADORES INDUSTRIALES Y DE LABORATORIO
A UNIT OF GENERAL SIGNAL GS
LIGHTNIN
neptune MEASUREMENT COMPANY
MEDIDORES DE FLUJO

jamesbury corp. VALVULAS DE BOLA
SPRAYING SYSTEMS CO. SPRAY NOZZLES & ACCESSORIES
BOQUILLAS ASPERSORAS PARA LA INDUSTRIA Y LA AGRICULTURA

MEDIDORES INDUSTRIALES Y DE PETROLEO
Smith Meter Inc.
A Moorco Company
MOORCO

Calle 25 Norte No. 6N-75
Tels.: 681324/613241/622490/91
FAX: 672492. Télex: 55479. A.A. 9477
Cali - Colombia

ALMACEN
SOLUCIONES ELECTRICAS LTDA.

TODO EN ELECTRICOS
ILUMINACION EN GENERAL

MATERIALES ELECTRICOS
Industriales y Residenciales
Nacionales e Importados
Distribuidores de:

Breter
Motores
Estabilizadores
Ventiladores
Pirómetros
Siemens
3M Telemecanique
Colmena
Pavco
Ergon
Electrocontrol

Carrera 6 No. 18-77
80 03 67 - 80 96 95
83 73 82 - 83 34 05
Telefax (923) 833405