UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"

PARSER

Grupo: BRUNO VEDOVETO
BIANCA PRIVATI
GIOVANNA CAZELATO
WESLEY OTTO

Profº: MAURÍCIO DIAS

RIO CLARO, 22 DE JUNHO DE 2015

INSTRUÇÕES MIPS

FUNÇÕES EM ASSEMBLY/BINÁRIO E BINÁRIO/ASSEMBLY

RIO CLARO
2015

RESUMO

Desenvolvemos um Parser que lê um arquivo que contenha um conjunto de instruções de linguagem de máquina (Assembly) e o transforma em um arquivo contendo seu respectivo código binário. O Parser também lê um arquivo em código binário e o transforma para Assembly.

INTRUÇÕES

O programa “Parser” deve ser executado dentro de uma pasta contendo os arquivos: “AssBin.h”, “binary.h”, “BinAss.h”, “constsOP_Fun.h”, “constsReg.h”, “file.h”, “useful.h” e o arquivo de texto(.txt) contendo as instruções em Assembly que será convertida para binário com o nome “AssBin”, ou um arquivo de texto(.txt) contendo o binário que será convertido para Assembly com o nome “BinAss”.

- Objetivo:

O objetivo deste trabalho é transformar, com sucesso, binário em Assembly e vice-versa.

TEORIA

As funções principais neste trabalho são as que estão no arquivo “useful.h”, são elas:

- Função POS - int pos(char* string, char search, int occurrence)

Utilizamos esta função para encontrar a posição de um certo caractere em uma string, dependendo de sua ocorrência(retorna um inteiro).

Foi útil no trabalho para encontrarmos a posição que representa o começo e o fim de cada parte de uma instrução como funções, registradores e/ou constantes. Por exemplo:

Dada a instrução (string) “**ADD $s0, $s1, $zero**”

```
pos(instruction, $, 1)     // nos retorna a posição 4
pos(instruction, $, 2)     // nos retorna a posição 9
```

- Função SUBSTRING - void substring(char* result, char* string, int start, int length)

Utilizamos esta função para recortar parte de uma string.

Utilizando as posições encontradas pela função pos(), fomos capazes de recortar cada parte de uma instrução. Por exemplo:

Dada a instrução(string) “**ADD $s0, $s1, $zero**”

substring(result, instruction, 0, 3) // nos retorna a string “ADD” na variável result.

- Função TRIM - char* trim (char *s)

Utilizada para retirar espações desnecessários no começo e/ou/ fim da string.

Existe a possibilidade de uma instrução possuir espações antes ou depois. Tais espaços devem ser desconsiderados, por isso utilizamos esta função. Por exemplo:

Dada a instrução (string) “ **ADD $s0, $s1, $zero** “

trim(instruction) // nos retorna a string “ADD $s0, $s1, $zero”

Definimos nossos labels como binários de 6 bits.

Em “AssBin.h”, para identificarmos um label utilizamos a função pos(). Caso exista o caractere “**:**” em uma linha, podemos admitir que se trata de um label.

Em “BinAss.h”, para identificarmos um label, se o binário possuir apenas 6 bits, podemos admitir que se trata de um label.

Em nosso programa temos uma lista que guarda o binário e o nome do label. Ao identificarmos um label, procuramos o mesmo nesta lista. Caso exista, apenas retornamos seu binário (ou nome). Caso contrário, geramos um novo binário (ainda não existente na lista) e acrecentamos em nossa lista.

DISCUSSÃO E CONCLUSÃO

O trabalho foi feito com a ajuda do GitHub, GitBash e o Kanbanflow. Com a cooperação de todos, conseguimos completar em tempo e com êxito a primeira parte do trabalho. Caso queira conferir nosso ambiente de trabalho segue o link do nosso repositório: https://github.com/UNESP/UNESP.