DCC/ICEX/UFMG

Curso: Bacharelado em Ciência da Computação

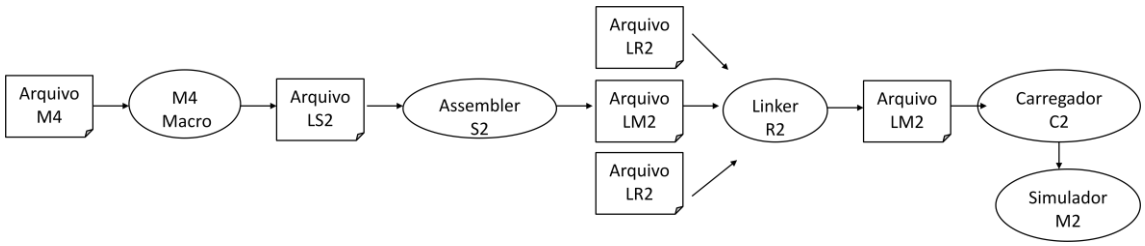
Disciplina: Software Básico

2º. Semestre de 2017

TRABALHO PRÁTICO 1

Este trabalho consiste no desenvolvimento de um mini-sistema de programação capaz de executar programas escritos em linguagem de montagem (assembler). Para tanto foi definida uma máquina virtual simples, M2 (Máquina do Marcos).

O sistema terá a seguinte estrutura:



O trabalho, detalhado a seguir, consiste no seguinte:

A) Implementar um simulador de M2 e um carregador de programa (C2).
B) Implementar um montador (S2) que traduza um programa da linguagem simbólica LS2 para a linguagem de máquina M2.
C) Expandir a linguagem de máquina e o montador de forma a facilitar o acesso a elementos de vetores.
D) Expandir a linguagem de máquina e o montador de forma a permitir a inclusão de funções e subrotinas internas.
E) Utilizar um processador de macro para fazer a definição e expansão de macros.
F) Implementar um editor de ligação (R2) para permitir a tradução de módulos em separado e a realocação de programas.

Na implementação do sistema o tratamento de erros deverá ser o mínimo possível, ou seja, pode-se considerar que o programa tratado esteja correto.

Parte A – Simulador M2

A máquina M2 possui as seguintes características:

-unidade endereçável: palavra (correspondente a um inteiro).

-dados: somente inteiros

-registradores: PC (Program Counter) e AC (acumulador).

-formato da instrução: Operação| deslocamento

-endereçamento: relativo ao contador de programa

Endereço= (PC)+ deslocamento

Conjunto de instrução

(Notação: x= endereço; (x)= conteúdo)

Código	Símbolo	Nome	Ação
01	LAD	Carga AC direto	AC<- (M)
02	SAD	Armazena AC direto	(M) <- AC
03	ADD	Soma	AC<- AC+(M)
04	SUB	Subtrai	AC<- AC-(M)
05	INP	Entrada	(M) <- inteiro lido
06	OUT	Saída	Escreve (M)
07	JMP	Desvio	PC <- M
08	JGZ	Desvio-maior	Se AC>0 então PC <-M
09	JLZ	Desvio-menor	Se AC<0 então PC <-M
10	JZE	Desvio-igual	Se AC==0 então PC <-M
11	HLT	pare	

A.1 Implementar um programa (interpretador) que simula a máquina M2, ou seja, que executa programas em linguagem de máquina de M2 (para a execução de um programa, este deve estar na memória de M2 e PC deve conter o endereço da primeira instrução executável).

A.2 Implementar um programa (carregador C2) que ,dado um arquivo contendo um programa em linguagem de máquina de M2 e o endereço de carga, carregue o programa na memória de M2 e inicialize PC.

A.3 Testar o simulador com o seguinte programa que lê dois números e imprime o menor deles. Estão mostrados o código “assembler” e o código objeto.

simbólico	endereço	conteúdo
INP A	00	05 18
INP B	02	05 17
LAD A	04	01 14
SAD C	06	02 14
SUB B	08	04 11
JLZ P	10	09 04
LAD B	12	01 07
SAD C	14	02 06
P OUT C	16	06 04

HLT	18	11 00
A DC 0	20	0
B DC 0	21	0
C DC 0	22	0
END		

Obs: O endereçamento é relativo ao contador de programa (PC). O PC sempre aponta para a próxima instrução a ser executada. No estágio IF (instruction fetch), a instrução é lida da memória e o PC é incrementado. Então, na primeira linha do programa temos PC=2 e o endereçamento é 18+PC=20.

Parte B – Montador (assembler)

A linguagem simbólica de M2, LS2, é bastante simples:

-formato fixo.

-símbolos (rótulos) de apenas um caracter.

-operando pode ser apenas constante ou símbolo.

-pseudo-operadores: DC ("define constant") e END (fim de programa).

B.1 Implementar um programa (S2) que traduza um programa na linguagem simbólica LS2 para a linguagem de máquina LM2.

B.2 Traduzir e executar o programa exemplo mostrado na parte A.

B.3 Escrever um programa em LS2 para calcular o resto da divisão de um número inteiro positivo por outro inteiro positivo. Traduzir e executar esse programa.

Parte C – Repetição e endereçamento indexado

Para facilitar a repetição de um conjunto de instruções podem ser incluídos um registrador contador RC, uma instrução de carga desse registrador e uma instrução de desvio condicional com decremento automático. O acesso a elementos de vetores pode ser feito através de um registrador índice RX. Assim, devem ser acrescentados à máquina M2 os registradores RC e RX, juntamente com as seguintes instruções:

Código	Símbolo	Nome	Ação
12	LXD	carrega RX direto	RX<- (M)
13	SXD	armazena RX direto	(M) <- RX
14	LAX	Carrega AC indexado	AC<- (RX); RX<-RX+1
15	SAX	Armazena AC indexado	(RX)<- AC; RX<-RX+1
16	LCD	Carrega RC direto	RC <-(M)
17	JCC	Conta e desvia	RC <-RC-1; se RC>0 então PC<-M

Acrescentar à linguagem simbólica LS2 os pseudo-operadores DS x (“define storage”) e DA x (“define address”).

DS 4 4 palavras

DA X endereço de X (absoluto)

C1. Fazer as devidas alterações no montador e no simulador.

C.2 Escrever um programa em LS2 para zerar um conjunto de n posições.

Parte D – Subprogramas

A máquina M2 deve permitir também a utilização de subprogramas (funções e subbrotinas). A forma adotada de comunicação com subprogramas é a seguinte:

- endereço de retorno: colocado no registrador RX.

- endereço de cada parâmetro: colocado logo após a instrução de chamada do subprograma.

- resultado de função: colocado no acumulador.

Para isso devem ser adicionadas as seguintes instruções à máquina M2:

Código	Símbolo	Nome	Ação
18	CAL	Chama subprograma	RX<-PC; PC <-M
19	RET	retorna	PC <-RX
20	LAI	Carrega AC indireto	AC<-((RX)); RX<-RX+1
21	SAI	Armazena AC indireto	((RX))<- AC; RX<-RX+1

D.1 Traduzir e executar o programa seguinte, que implementa a função dobro(x).

```
   INP A
   CAL S
   DA A
   SAD D
   OUT A
   OUT D
   HLT
A  DC 0
D  DC 0
S  LAI  ;AC<-A
   SAD T ;T<-A
```

```
   ADD T ;AC<-A+A
L  RET ; retorna
T  DC 0
   END
```

D.2 Escrever um subprograma para calcular o resto da divisão de um número inteiro positivo por outro inteiro positivo, e um programa que utiliza este subprograma; traduzir e executar o programa resultante.

Deverá ser usado o método da divisão chinesa, que é baseado nas operações dobro(x), metade(x) e par(x). Para isso devem ser acrescentadas à maquina M2 as instruções DOB n, MET n e JPA x (desvio se ACC par).

Parte E – Processador de Macros

Um processador de macros deve ser incorporado ao sistema. Recomenda-se usar o m4 (UNIX).

E.1 Definir as macro INC(x), DEC(x) e CLR(x), para incrementar, decrementar e zerar o valor de uma variável.

E.2 Escrever um programa em LS2, usando as macro acima, para ler pares de números inteiros positivos e calcular o quociente da divisão do primeiro número pelo segundo. Traduzir e executar esse programa.

E.3 Usar um macro processador para traduzir programas escritos em pseudo-linguagem estilo C para a linguagem de moantgem da máquina M2. A linguagem deve ter pelo menos as palavras chaves: ler, escrever, inc, dec, clr, menor, igual, maior, begin, end, end_programa, soma, sub, se maior, se menor, se igual, end_se, para,end_para.

Parte F- Editor de ligação

O sistema de programação de M2 deve também permitir:

- Relocação de programas: endereço de carga do programa definido somente após a tradução.
- Tradução separada: módulos traduzidos separadamente e depois combinados para formar um único programa objeto.

F.1 Alterar o programa tradutor S2 de forma a gerar informações para relocação e ligação de programas (formato relocável, LR2).

F.2 Implementar um editor de ligação (R2) que combine módulos objeto relocáveis em um único módulo executável.

F.3 Reescrever o programa do item D.2 em módulos separados. Traduzir, ligar e executar o programa, para diferentes endereços de carga.

F.4 Se o endereço de carga real (fornecido ao carregador) for diferente do endereço de carga fornecido quando da geração do código executável (pelo editor de ligação), o programa do item anterior não será executado corretamente. Reescrever esse programa para que isto não aconteça, ou seja, para que o programa seja auto-relocável. Executar o programa para diferentes endereços de carga.

Formato da entrega

Os trabalhos práticos devem ser entregues contendo: um relatório, o código fonte da solução implementada, sendo que esse código deve ser acompanhado por uma makefile que compile os programas de acordo com o especificado abaixo:

Simulador: M2

Assembler: S2

Linker : R2

Formato de Entrada e Saída

Parte A

Para a parte A seu programa deverá receber dois parâmetros, o primeiro especificando o programa em linguagem de máquina e um segundo especificando o local de carregamento desse. Além disso, os comandos INP e OUT devem ler e escrever os resultados na entrada e saída padrão, respectivamente. (OBS: Nada além do valores de OUT devem ser impressos, sendo o valor de OUT sempre seguido por uma quebra de linha `/n`).

Exemplo de entrada (onde o programa utilizado é o apresentado no inicio deste documento)

./M2 programa.m2 10 < entradas.txt

Exemplo de saída (tendo como entrada 40\n 80)

40

Parte B

A parte B receberá dois parâmetros de entrada. O primeiro deles é um arquivo contendo o programa em linguagem de montagem o segundo é o arquivo de saída que será escrito com a linguagem de máquina reconhecida por M2.

Exemplo de execução (Um exemplo exemplo de entrada e saída é apresentado no inicio do deste documento)

./S2 programa.s2 programa.m2

Parte C, D e E

As partes C, D e E tem um formato similar a parte A

Parte F

A parte F deverá receberá um conjunto de arquivos de entrada de entrada e um arquivo de saída. Sendo os arquivos de entrada no formato retornado pelo programa S2 e a saída no formato da máquina M2.

Exemplo de execução

./R2 programa1.m2 programa2.m2 programa3.m2 programasaida.m2

Informações Importantes

- O trabalho deve ser feito individualmente, podendo ser discutido entre os colegas desde que não haja cópia ou compartilhamento do código fonte.
- A data de entrega será especificada através de uma tarefa no Moodle.
- Os trabalhos poderão ser entregues até às 23:55 do dia especificado para a entrega. O horário de entrega deve respeitar o relógio do sistema Moodle. Haverá uma tolerância de 5 minutos de atraso, de forma que os alunos podem fazer a entrega até às 0:00. A partir desse horário, os trabalhos já estarão sujeitos a penalidades. A fórmula para desconto por atraso na entrega do trabalho prático é:

  Desconto = 2d/0.32 %

  onde d é o atraso em dias úteis. Note que após 5 dias úteis, o trabalho não pode ser mais entregue.
- O trabalho deve ser implementado obrigatoriamente na linguagem C.
- Deverá ser entregue o código fonte com os arquivos de dados necessários para a execução e um arquivo Makefile que permita a compilação do programa nas máquinas Linux do DCC.
- Além disso, deverá ser entregue uma pequena documentação contendo todas as decisões de projeto que foram tomadas durante a implementação, sobre aspectos não contemplados na especificação, assim como uma justificativa para essas decisões. Esse documento não precisa ser extenso (mínimo 3 e máximo de 10 páginas). A documentação deve indicar o nome dos aluno. O código fonte não deve ser incluído no arquivo PDF da documentação.
- Todas as dúvidas referentes ao trabalho serão esclarecidas por meio do fórum disponível no ambiente Moodle da disciplina.
- A entrega do trabalho deverá ser realizada pelo Moodle, na tarefa criada especificamente para tal. A entrega deverá ser feita no seguinte formato:
  - O trabalho a ser entregue deverá estar contido em um único arquivo compactado, em formato “.zip”, com o nome no formato “tp1_aluno.zip”
  - O arquivo .zip definido deverá ter três pastas:

- “src”: Essa pasta deverá conter o código-fonte do montador implementado, juntamente do arquivo Makefile (OBS.: Não devem ser incluídos arquivos .o nem executáveis nessa pasta.)
- “tst”: Essa pasta irá conter os arquivos de entrada desenvolvidos para testar esse projeto.
- “doc”: Essa pasta deverá conter o arquivo da documentação, em formato PDF. Caso julgue necessário incluir quaisquer outros arquivos a parte, esses deverão ser justificados em um arquivo texto com o nome README.

Atenção: Trabalhos que descumprirem o padrão definido acima serão penalizados