Project 1

Hazael Magino

2/23/2024

VV12417

The overall scope of this project is to show the development of how a hamming code can be determined using x86 architecture.

Milestone 1

Hamming Distance:

- This is the concept of determining the difference between two strings of equal length. Specifically, this measures the number of positions at which the corresponding characters in the string differ. This is meant to represent the minimum number of errors that could have occurred between two communicating computers.

Shortest length String:

- Since hamming distance is compared between two equal length strings having differences in length poses a problem. If a length of a string is shorter than the other the hamming distance would be incorrect due to the fact that we are counting one less than what is supposed to be the value of the hamming distance. One way to deal with this dilemma is to pad the remaining lengths of the string with 0s until all register bits are fully occupied, thus being equal in length. Another way of dealing with a shorter string is that we only calculate the hamming distance up to the length of the shorter string. This enables us to get accurate results as if we were to calculate after padding registers with 0s.

Successive SHR instruction:

- Finding the hamming distance between two strings using SHR or SHL is a useful process as it allows us to determine the hamming distance between two strings with unequal lengths. This technique allows successive right shifts or left on one of the numbers in the binary sequence of each strings simultaneously. With this we can XOR the result and analyze the XOR result to determine the full hamming distance.

Milestone 2:

Here is the general program I used:

Note this program has a Segfault because of me using printf to output the distance. I don’t know how to produce a value without it.

Project 1

Hazael Magino

2/23/2024

VV12417

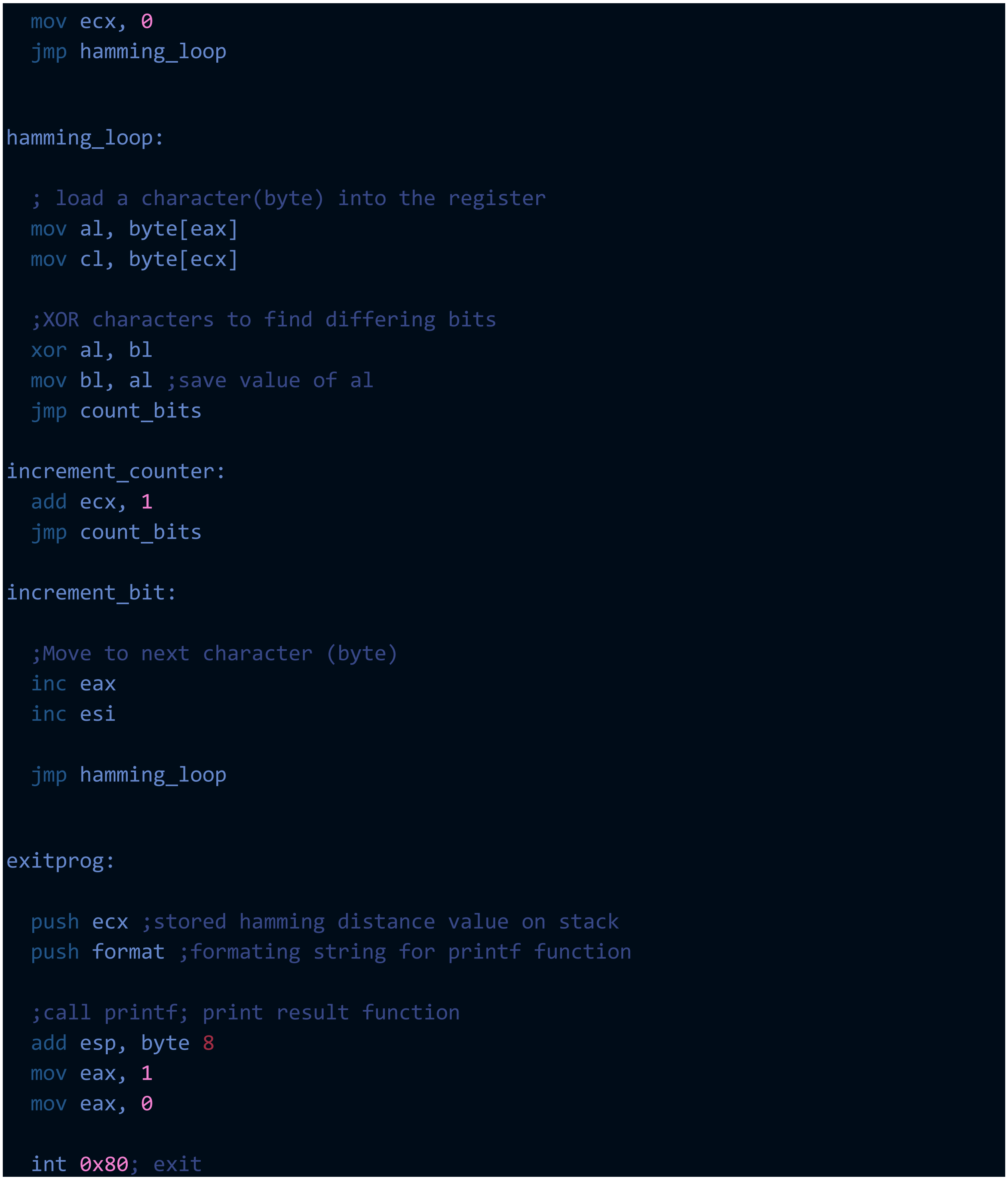
```
section .data
  msg1 dd 'chess', 0x0A ; words to determine hamming distance
  msg2 dd 'minute', 0x0A

  format: db "The Hamming distance between msg1 and msg2 is:",10,"%d",10,0


  count1 equ $-msg1
  count2 equ $-msg2


section .text
  extern printf
  global main

main:

  ;initialize all registers to 0, avoid garbage data

  xor eax, eax
  xor ebx, ebx
  xor ecx, ecx
  xor edx, edx


  ;Check if msgs are stored
  mov eax, 4 ;sys write
  mov ebx, 1 ; std output
  mov ecx, msg1  ;store first input
  mov edx, count1 ;store length of first input
  int 0x80


  mov eax, 4 ;sys write
  mov ebx, 1 ;std output
  mov edi, msg2 ;store second input
  mov esi, count2 ; store length of input
  int 0x80
```

Project 1

Hazael Magino

2/23/2024

VV12417



```
  mov ecx, 0
  jmp hamming_loop


hamming_loop:

  ; load a character(byte) into the register
  mov al, byte[eax]
  mov cl, byte[ecx]

  ;XOR characters to find differing bits
  xor al, bl
  mov bl, al ;save value of al
  jmp count_bits

increment_counter:
  add ecx, 1
  jmp count_bits

increment_bit:

  ;Move to next character (byte)
  inc eax
  inc esi

  jmp hamming_loop


exitprog:

  push ecx ;stored hamming distance value on stack
  push format ;formating string for printf function

  ;call printf; print result function
  add esp, byte 8
  mov eax, 1
  mov eax, 0

  int 0x80; exit
```

Project 1

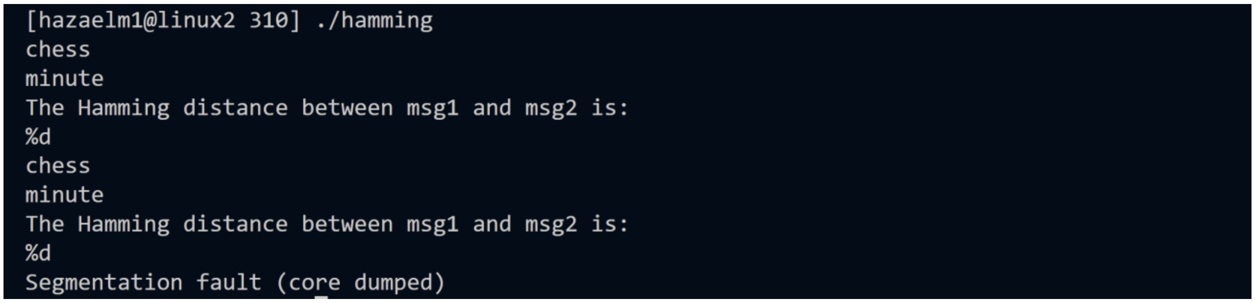
Hazael Magino

2/23/2024

VV12417

[redacted]

Result:



```
[hazaelm1@linux2 310] ./hamming
chess
minute
The Hamming distance between msg1 and msg2 is:
%d
chess
minute
The Hamming distance between msg1 and msg2 is:
%d
Segmentation fault (core dumped)
```