Erik Delanois Dallas Delaney

delanos2 ddelane2

ECE 391 – AD2

Problem 1:

EAX – return value

EBX – arg

ECX – i loop counter

EDX – bit

ESI- bit mask

Dispatch:

PUSHL %EBP #push on EBP

MOVEL %ESP, %EBP #set new ebp

PUSHL %ESI

PUSHL %EBI

PUSHL %EBX

MOVL 4(%EBP,$0) ,% ESI #get the bitmask from the stack

MOVL 8(%EBP,$0), %EBX #get arg from stack

MOVL $0,% ECX # set loop counter to 0

MOVL $1, %EDX # set bit to one

LOOP:

MOVL 4(%EBP,$0) , % ESI #get the bitmask from the stack since it will be

ANDL %EDX, % ESI # and the two values

CMPL $0,% ESI

jne REVAL # if not zero then match

SHLL %EDX # shift bit

INCL %ECX #increment loop count

CMPL $32,%ECX #compare loops done to 32

jb LOOP # if ECX less than 32 then do loop again

MOVL $0, %EAX # if the code gets here return 0

jmp DONE

REVAL:

PUSHL %EBX #push arg onto the stack

jmp \*jump\_table(,%ECX,4) #call to function

#EAX now also has the correct return value

POPL %EBX #pop pushed arguments

DONE:

POPL -4(EBP),%ESI

POPL -8(EBP),%EBI

POPL -12(EBP),%EBX

LEAVE

RET

Problem 2:

int calculate( int ecx, int eax){

if(ecx > 2)

return eax;

if(ecx == 0)

return eax\*eax;

if(ecx == 1)

return –eax;

if(ecx == 2)

return eax + 128;

return eax;

}