# sum\_list - Sum of elements in a linked list

#author : HackerM

.pos 0

irmovq stack, %rsp

call sum\_list

halt

.align 8

ele1:

.quad 0x00a

.quad ele2

ele2:

.quad 0x0b0

.quad ele3

ele3:

.quad 0xc00

.quad 0

sum\_list:

xorq %rax,%rax

irmovq ele1,%rdi

loop:

andq %rdi,%rdi

je done

mrmovq (%rdi),%rsi

addq %rsi,%rax

mrmovq 0x8(%rdi),%rdi

jmp loop

done:

ret

.pos 0x200

stack:

xorq %rax,%rax # count = 0;

andq %rdx,%rdx # len <= 0?

jle Done # if so, goto Done:

iaddq $-1,%rdx

jle Loop2

Loop1:

mrmovq (%rdi), %r10 # read val from src...

mrmovq 8(%rdi),%r9

rmmovq %r10, (%rsi) # ...and store it to dst

rmmovq %r9, 8(%rsi)

xorq %rcx, %rcx

andq %r10, %r10 # val <= 0?

irmovq $1, %r10

cmovg %r10, %rcx

addq %rcx, %rax # count++ or not

xorq %rcx, %rcx

andq %r9, %r9 # val <= 0?

cmovg %r10, %rcx

addq %rcx, %rax # count++ or not

iaddq $0x10, %rdi # src+=2

iaddq $0x10, %rsi # dst+=2

iaddq $-2, %rdx # len-=2 && setCC

jg Loop1 # if so, goto Loop1:

jl Done

Loop2:

mrmovq (%rdi), %r10

rmmovq %r10, (%rsi)

xorq %rcx, %rcx

andq %r10, %r10

irmovq $1, %r10

cmovg %r10, %rcx

addq %rcx, %rax

irmovq $1, %rbx

xorq %rax,%rax # count = 0;

andq %rdx,%rdx # len <= 0?

jle Done # if so, goto Done:

iaddq $-3, %rdx

jle Re

Loop1:

mrmovq (%rdi), %r8 # read val from src...

mrmovq 8(%rdi),%r9 # ...and store it to dst

rmmovq %r8, (%rsi)

rmmovq %r9, 8(%rsi)

xorq %rcx, %rcx

andq %r8, %r8 # val <= 0?

cmovg %rbx, %rcx

addq %rcx, %rax # count++ or not

xorq %rcx, %rcx

andq %r9, %r9 # val <= 0?

cmovg %rbx, %rcx

addq %rcx, %rax # count++ or not

mrmovq 0x10(%rdi), %r10

mrmovq 0x18(%rdi), %r11

rmmovq %r10, 0x10(%rsi)

rmmovq %r11, 0x18(%rsi)

xorq %rcx, %rcx

andq %r10, %r10 # val <= 0?

cmovg %rbx, %rcx

addq %rcx, %rax # count++ or not

xorq %rcx, %rcx

andq %r11, %r11 # val <= 0?

cmovg %rbx, %rcx

addq %rcx, %rax # count++ or not

iaddq $0x20, %rdi # src+=4

iaddq $0x20, %rsi # dst+=4

iaddq $-4, %rdx # len-=4 && setCC

jg Loop1 # if so, goto Loop1:

Re:

iaddq $3, %rdx

je Done

Loop2:

mrmovq (%rdi), %r10

rmmovq %r10, (%rsi)

xorq %rcx, %rcx

andq %r10, %r10

cmovg %rbx, %rcx

addq %rcx, %rax

iaddq $8, %rdi

iaddq $8, %rsi

iaddq $-1, %rdx

jg Loop2

jmp S1

S2:

iaddq $-5, %rdx

jle Re

Loop1:

mrmovq (%rdi), %r8

mrmovq 8(%rdi),%r9

mrmovq 0x10(%rdi), %r10

mrmovq 0x18(%rdi), %r11

mrmovq 0x20(%rdi), %r12

mrmovq 0x28(%rdi), %r13

rmmovq %r8, (%rsi)

rmmovq %r9, 8(%rsi)

rmmovq %r10, 0x10(%rsi)

rmmovq %r11, 0x18(%rsi)

rmmovq %r12, 0x20(%rsi)

rmmovq %r13, 0x28(%rsi)

iaddq $0x30, %rdi # src+=4

iaddq $0x30, %rsi # dst+=4

andq %r8, %r8

jle LE1

iaddq $1, %rax

LE1:

andq %r9, %r9

jle LE2

iaddq $1, %rax

LE2:

andq %r10, %r10

jle LE3

iaddq $1, %rax

LE3:

andq %r11, %r11

jle LE4

iaddq $1, %rax

LE4:

andq %r12, %r12

jle LE5

iaddq $1, %rax

LE5:

andq %r13, %r13

jle LE6

iaddq $1, %rax

LE6:

iaddq $-6, %rdx # len-=4 && setCC

jg Loop1 # if so, goto Loop1:

Re:

iaddq $5, %rdx

je Done

Loop2:

mrmovq (%rdi), %r10

rmmovq %r10, (%rsi)

xorq %rcx, %rcx

andq %r10, %r10

cmovg %rbx, %rcx

addq %rcx, %rax

iaddq $8, %rdi

iaddq $8, %rsi

iaddq $-1, %rdx

jg Loop2

jmp Done

S1:

irmovq $1, %rbx

xorq %rax,%rax

andq %rdx,%rdx # len <= 0?

jg S2

jmp S1

S2:

iaddq $-5, %rdx

jle Re

L1:

mrmovq (%rdi), %r8

mrmovq 8(%rdi),%r9

mrmovq 0x10(%rdi), %r10

mrmovq 0x18(%rdi), %r11

mrmovq 0x20(%rdi), %r12

mrmovq 0x28(%rdi), %r13

rmmovq %r8, (%rsi)

rmmovq %r9, 8(%rsi)

rmmovq %r10, 0x10(%rsi)

rmmovq %r11, 0x18(%rsi)

rmmovq %r12, 0x20(%rsi)

rmmovq %r13, 0x28(%rsi)

iaddq $0x30, %rdi # src+=6

iaddq $0x30, %rsi # dst+=6

andq %r8, %r8

jle LE1

iaddq $1, %rax

LE1:

andq %r9, %r9

jle LE2

iaddq $1, %rax

LE2:

andq %r10, %r10

jle LE3

iaddq $1, %rax

LE3:

andq %r11, %r11

jle LE4

iaddq $1, %rax

LE4:

andq %r12, %r12

jle LE5

iaddq $1, %rax

LE5:

andq %r13, %r13

jle LE6

iaddq $1, %rax

LE6:

iaddq $-6, %rdx # len-=6 && setCC

jg L1 # if so, goto Loop1:

Re:

iaddq $5, %rdx

je Done

L2:

mrmovq (%rdi), %r10

iaddq $8, %rdi

andq %r10, %r10

rmmovq %r10, (%rsi)

jle LE

iaddq $1, %rax

LE:

iaddq $8, %rsi

iaddq $-1, %rdx

jg L2

jmp Done

S1:

irmovq $1, %rbx

xorq %rax,%rax

andq %rdx,%rdx # len <= 0?

jg S2