Autor: Francisco Javier Bolívar Expósito

Práctica 2: Calcular la media de una lista de N enteros

Versión 5.1 *Sumar N enteros sin signo de 32bits sobre dos registros de 32bits usando uno de ellos como acumulador de acarreos (N=16)*

Código media.s

```
.section .data
lista:    .int 0xffffffff, 0x00000001
longlista:        .int   (.-lista)/4
resultado:        .quad   0
  formato:        .asciz   "suma = %lu = 0x%lx hex\n"


.section .text
main: .global  main

          mov     $lista, %rbx
          mov  longlista, %ecx
          call suma                 # == suma(&lista, longlista);
          mov  %eax, resultado
          mov  %edx, (resultado+4)

          mov   $formato, %rdi
          mov   resultado,%rsi
          mov   resultado,%rdx
          mov         $0,%eax     # varargin sin xmm
          call  printf            # == printf(formato, res, res);

          mov  resultado, %edi
          ret

suma:
          mov  $0, %eax
          mov  $0, %rsi
          mov  $0, %edx
bucle:
          add  (%rbx,%rsi,4), %eax
          jnc   etiqueta
          inc   %edx
etiqueta:
    inc   %rsi
          cmp   %rsi,%rcx
          jne    bucle
    ret
```

Versión 5.2 *Sumar N enteros sin signo de 32bits sobre dos registros de 32bits mediante extensión con ceros (N=16)*

Código media.s

```
.section .data
#ifndef TEST
#define TEST 9
#endif
   .macro linea
   #if TEST==1
         .int 1,1,1,1
   #elif TEST==2
         .int 0x0fffffff, 0x0fffffff, 0x0fffffff, 0x0fffffff
   #elif TEST==3
         .int 0x10000000, 0x10000000, 0x10000000, 0x10000000
   #elif TEST==4
         .int 0xffffffff, 0xffffffff, 0xffffffff, 0xffffffff
   #elif TEST==5
         .int -1, -1, -1, -1
   #elif TEST==6
         .int 200000000, 200000000, 200000000, 200000000
   #elif TEST==7
         .int 300000000, 300000000, 300000000, 300000000
   #elif TEST==8
         .int 5000000000, 5000000000, 5000000000, 5000000000
   #else
      .error "Definir TEST entre 1..8"
   #endif
      .endm

lista:    .irpc i, 1234
               linea
          .endr
longlista:       .int   (.-lista)/4
resultado:       .quad   0
formato:         .asciz   "suma = %lu = 0x%lx hex\n"


.section .text
main: .global  main

          mov     $lista, %rbx
          mov  longlista, %ecx
          call suma                  # == suma(&lista, longlista);
          mov  %eax, resultado
          mov  %edx, (resultado+4)
```

```
        mov   $formato, %rdi
        mov   resultado,%rsi
        mov   resultado,%rdx
        mov         $0,%eax     # varargin sin xmm
        call  printf            # == printf(formato, res, res);

        mov  resultado, %edi
        ret

suma:
        mov  $0, %eax
        mov  $0, %rsi
        mov  $0, %edx
bucle:
        add (%rbx,%rsi,4), %eax
        adc $0, %edx

   inc  %rsi
        cmp   %rsi,%rcx
        jne    bucle

   ret
```

Resultado de la batería de tests

T#1suma = 16 = 0x10 hex
T#2suma = 4294967280 = 0xfffffff0 hex
T#3suma = 4294967296 = 0x100000000 hex
T#4suma = 68719476720 = 0xfffffff0 hex
T#5suma = 68719476720 = 0xfffffff0 hex
T#6suma = 3200000000 = 0xbebc2000 hex
T#7suma = 4800000000 = 0x11e1a3000 hex
T#8suma = 11280523264 = 0x2a05f2000 hex

Versión 5.3 *Sumar N enteros con signo de 32bits sobre dos registros de 32bits (mediante extensión de signo, naturalmente) (N=16)*

Código media.s

```
.section .data
#ifndef TEST
#define TEST 9
#endif
    .macro linea
    #if TEST==1
            .int -1, -1, -1, -1
    #elif TEST==2
            .int 0x04000000, 0x04000000, 0x04000000, 0x04000000
    #elif TEST==3
            .int 0x08000000, 0x08000000, 0x08000000, 0x08000000
    #elif TEST==4
            .int 0x10000000, 0x10000000, 0x10000000, 0x10000000
    #elif TEST==5
            .int 0x7fffffff, 0x7fffffff, 0x7fffffff, 0x7fffffff
    #elif TEST==6
            .int 0x80000000, 0x80000000, 0x80000000, 0x80000000
    #elif TEST==7
            .int 0xF0000000, 0xF0000000, 0xF0000000, 0xF0000000
    #elif TEST==8
            .int 0xF8000000, 0xF8000000, 0xF8000000, 0xF8000000
    #elif TEST==9
            .int 0xF7FFFFFF, 0xF7FFFFFF, 0xF7FFFFFF, 0xF7FFFFFF
    #elif TEST==10
            .int 100000000, 100000000, 100000000, 100000000
    #elif TEST==11
            .int 200000000, 200000000, 200000000, 200000000
    #elif TEST==12
            .int 300000000, 300000000, 300000000, 300000000
    #elif TEST==13
            .int 2000000000, 2000000000, 2000000000, 2000000000
    #elif TEST==14
            .int 3000000000, 3000000000, 3000000000, 3000000000
    #elif TEST==15
            .int -100000000, -100000000, -100000000, -100000000
    #elif TEST==16
            .int -200000000, -200000000, -200000000, -200000000
    #elif TEST==17
            .int -300000000, -300000000, -300000000, -300000000
    #elif TEST==18
            .int -2000000000, -2000000000, -2000000000, -2000000000
    #elif TEST==19
            .int -3000000000, -3000000000, -3000000000, -3000000000
    #else
        .error "Definir TEST entre 1..19"
    #endif
        .endm
```

```
lista:    .irpc i, 1234
                linea
          .endr
longlista:        .int   (.-lista)/4
resultado:        .quad   0
formato:          .ascii    "suma  = %ld   = 0x%016lx hex\n"

.section .text
main: .global  main
          mov     $lista, %rbx
          mov  longlista, %ecx

          call suma                  # == suma(&lista, longlista);
          mov  %eax, resultado
          mov  %edx, (resultado+4)

          mov   $formato, %rdi
          mov   resultado,%rsi
          mov   resultado,%rdx
          mov         $0,%eax
          call  printf               # == printf(formato, res, res);

          mov  resultado, %edi
          ret

suma:
          mov  $0, %rsi
          mov  $0, %eax
          mov  $0, %edx
          mov  $0, %r8d
          mov  $0, %r9d
bucle:
          add (%rbx,%rsi,4), %eax
          CDQ
          add %eax, %r8d
          adc %edx, %r9d

          mov $0, %eax
          mov $0, %edx

          inc   %rsi
          cmp   %rsi,%rcx
          jne    bucle

          mov %r8d, %eax
          mov %r9d, %edx
    ret
```

Resultado de la batería de tests

T#1suma = -16   = 0xfffffffffffffff0 hex
T#2suma = 1073741824   = 0x0000000040000000 hex
T#3suma = 2147483648   = 0x0000000080000000 hex
T#4suma = 4294967296   = 0x0000000100000000 hex
T#5suma = 34359738352   = 0x00000007ffffff0 hex
T#6suma = -34359738368   = 0xfffffff800000000 hex
T#7suma = -4294967296   = 0xffffffff00000000 hex
T#8suma = -2147483648   = 0xffffffff80000000 hex
T#9suma = -2147483664   = 0xffffffff7ffffff0 hex
T#10suma = 1600000000   = 0x000000005f5e1000 hex
T#11suma = 3200000000   = 0x00000000bebc2000 hex
T#12suma = 4800000000   = 0x000000011e1a3000 hex
T#13suma = 32000000000   = 0x0000000773594000 hex
T#14suma = -20719476736   = 0xfffffffb2d05e000 hex
T#15suma = -1600000000   = 0xffffffffa0a1f000 hex
T#16suma = -3200000000   = 0xffffffff4143e000 hex
T#17suma = -4800000000   = 0xfffffffee1e5d000 hex
T#18suma = -32000000000   = 0xfffffff88ca6c000 hex
T#19suma = 20719476736   = 0x00000004d2fa2000 hex

Versión 5.4 *Media y resto de N enteros con signo de 32bits calculada usando registros de 32bits (N=16)*

Código media.s

```
.section .data
#ifndef TEST
#define TEST 20
#endif
        .macro linea
    #if TEST==1
            .int 1, 2, 1, 2
    #elif TEST==2
            .int -1, -2, -1, -2
    #elif TEST==3
            .int 0xfFFFFFFF, 0xfFFFFFFF, 0xfFFFFFFF, 0xfFFFFFFF
    #elif TEST==4
            .int 0x80000000, 0x80000000, 0x80000000, 0x80000000
    #elif TEST==5
            .int 0xffffffff, 0xffffffff, 0xffffffff, 0xffffffff
    #elif TEST==6
            .int 2000000000, 2000000000, 2000000000, 2000000000
    #elif TEST==7
            .int 3000000000, 3000000000, 3000000000, 3000000000
    #elif TEST==8
            .int -2000000000, -2000000000, -2000000000, -2000000000
    #elif TEST==9
            .int -3000000000, -3000000000, -3000000000, -3000000000
```

```
    #elif TEST>=10 && TEST<=14
            .int 1, 1, 1, 1
    #elif TEST>=15 && TEST<=19
            .int -1, -1, -1, -1
    #else
        .error "Definir TEST entre 1..19"
    #endif
        .endm

        .macro linea0
    #if TEST>=1 && TEST<=9
            linea
    #elif TEST==10
            .int 0, 2, 1, 1
    #elif TEST==11
            .int 1, 2, 1, 1
    #elif TEST==12
            .int 8, 2, 1, 1
    #elif TEST==13
            .int 15, 2, 1, 1
    #elif TEST==14
            .int 16, 2, 1, 1
    #elif TEST==15
            .int 0, -2, -1, -1
    #elif TEST==16
            .int -1, -2, -1, -1
    #elif TEST==17
            .int -8, -2, -1, -1
    #elif TEST==18
            .int -15, -2, -1, -1
    #elif TEST==19
            .int -16, -2, -1, -1
    #else
        .error "Definir TEST entre 1..19"
    #endif
        .endm

lista:          linea0
            .irpc i, 123
                linea
            .endr

longlista:      .int   (.-lista)/4
media:          .int 0
resto:          .int 0
formato:        .ascii    "media \t = %10d \t resto \t = %10d\n"
                .asciz  "\t\t = 0x%08x \t\t = 0x%08x \n\n"
```

```
.section .text
main: .global  main

        mov     $lista, %rbx
        mov  longlista, %ecx

        call suma                   # == suma(&lista, longlista);

        mov  %eax, media
        mov  %edx, resto

        mov   $formato, %rdi
        mov   media,%rsi
        mov   resto,%rdx
        mov   media,%rcx
        mov   resto,%r8
        mov   $0,%eax# varargin sin xmm
        call  printf                # == printf(formato, media, resto, media, resto);
        ret

suma:
        mov  $0, %rsi
        mov  $0, %eax
        mov  $0, %edx
        mov  $0, %r8d
        mov  $0, %r9d
bucle:
        add (%rbx,%rsi,4), %eax
        CDQ
        add %eax, %r8d
        adc %edx, %r9d

        mov $0, %eax
        mov $0, %edx

        inc   %rsi
        cmp   %rsi,%rcx
        jne    bucle

        mov %r8d, %eax
        mov %r9d, %edx

        idiv %ecx

    ret
```

Resultado de la batería de tests

T#1media	= 1	resto	= 8
	= 0x00000001		= 0x00000008
T#2media	= -1	resto	= -8
	= 0xffffffff		= 0xfffffff8
T#3media	= -1	resto	= 0
	= 0xffffffff		= 0x00000000
T#4media	= -2147483648	resto	= 0
	= 0x80000000		= 0x00000000
T#5media	= -1	resto	= 0
	= 0xffffffff		= 0x00000000
T#6media	= 2000000000	resto	= 0
	= 0x77359400		= 0x00000000
T#7media	= -1294967296	resto	= 0
	= 0xb2d05e00		= 0x00000000
T#8media	= -2000000000	resto	= 0
	= 0x88ca6c00		= 0x00000000
T#9media	= 1294967296	resto	= 0
	= 0x4d2fa200		= 0x00000000
T#10media	= 1	resto	= 0
	= 0x00000001		= 0x00000000
T#11media	= 1	resto	= 1
	= 0x00000001		= 0x00000001
T#12media	= 1	resto	= 8
	= 0x00000001		= 0x00000008
T#13media	= 1	resto	= 15
	= 0x00000001		= 0x0000000f
T#14media	= 2	resto	= 0
	= 0x00000002		= 0x00000000
T#15media	= -1	resto	= 0
	= 0xffffffff		= 0x00000000

```
T#16media     =       -1              resto    =       -1
              = 0xffffffff                     = 0xffffffff

T#17media     =       -1              resto    =       -8
              = 0xffffffff                     = 0xfffffff8

T#18media     =       -1              resto    =       -15
              = 0xffffffff                     = 0xfffffff1

T#19media     =       -2              resto    =       0
              = 0xfffffffe                     = 0x00000000
```

Versión 5.5 Media y resto de N enteros calculada en 32 y en 64 bits (N=16)

Código media.s

```
.section .data
# La definición de los tests es la misma que en la versión 5.4
# ….
longlista:      .int   (.-lista)/4
media:          .int 0
resto:          .int 0
formato:        .ascii   "\t 32bits\n \t media \t = %10d \t resto \t = %10d\n"
                .asciz  "\t\t = 0x%08x \t\t = 0x%08x \n\n"

formatoq:       .ascii   "\t 64bits\n \t media \t = %10d \t resto \t = %10d\n"
                .asciz  "\t\t = 0x%08x \t\t = 0x%08x \n\n"

.section .text
main: .global  main

        mov     $lista, %rbx
        mov  longlista, %ecx

        #Calcular en 32bits
        call suma               # == suma(&lista, longlista);

        mov  %eax, media
        mov  %edx, resto

        push %rcx
        mov   $formato, %rdi
        mov   media,%rsi
        mov   resto,%rdx
        mov   media,%rcx
        mov   resto,%r8
        mov   $0,%eax# varargin sin xmm
        call  printf            # == printf(formato, media, resto, media, resto);
        pop %rcx
```

```
        #Calcular en 64bits
        call sumaq

        mov  %eax, media
        mov  %edx, resto

        mov   $formatoq, %rdi
        mov   media,%rsi
        mov   resto,%rdx
        mov   media,%rcx
        mov   resto,%r8
        mov   $0,%rax           # varargin sin xmm
        call  printf            # == printf(formato, media, resto, media, resto);
        ret

suma:
        mov  $0, %rsi
        mov  $0, %eax
        mov  $0, %edx
        mov  $0, %r8d
        mov  $0, %r9d
bucle:
        add (%rbx,%rsi,4), %eax
        CDQ
        add %eax, %r8d
        adc %edx, %r9d

        mov $0, %eax
        mov $0, %edx

        inc   %rsi
        cmp   %rsi,%rcx
        jne    bucle

        mov %r8d, %eax
        mov %r9d, %edx

        idiv %ecx
        ret

sumaq:
        mov  $0, %rsi
        mov  $0, %rax
        mov  $0, %rdx
        mov  $0, %r8
```

```
bucleq:
        add (%rbx,%rsi,4), %eax
        CDQE
        add %rax, %r8

        mov $0, %rax

        inc   %rsi
        cmp   %rsi,%rcx
        jne    bucleq

        mov %r8, %rax

        CQO
        idiv %rcx

        ret
```

Resultado de la batería de tests

T#1	32bits				
	media	=	1	resto	= 8
		= 0x00000001			= 0x00000008
	64bits				
	media	=	1	resto	= 8
		= 0x00000001			= 0x00000008
T#2	32bits				
	media	=	-1	resto	= -8
		= 0xffffffff			= 0xfffffff8
	64bits				
	media	=	-1	resto	= -8
		= 0xffffffff			= 0xfffffff8
T#3	32bits				
	media	=	-1	resto	= 0
		= 0xffffffff			= 0x00000000
	64bits				
	media	=	-1	resto	= 0
		= 0xffffffff			= 0x00000000
T#4	32bits				
	media	= -2147483648		resto	= 0
		= 0x80000000			= 0x00000000

	64bits				
	media	= -2147483648	resto	=	0
		= 0x80000000		=	0x00000000
T#5	32bits				
	media	= -1	resto	=	0
		= 0xffffffff		=	0x00000000
	64bits				
	media	= -1	resto	=	0
		= 0xffffffff		=	0x00000000
T#6	32bits				
	media	= 2000000000	resto	=	0
		= 0x77359400		=	0x00000000
	64bits				
	media	= 2000000000	resto	=	0
		= 0x77359400		=	0x00000000
T#7	32bits				
	media	= -1294967296	resto	=	0
		= 0xb2d05e00		=	0x00000000
	64bits				
	media	= -1294967296	resto	=	0
		= 0xb2d05e00		=	0x00000000
T#8	32bits				
	media	= -2000000000	resto	=	0
		= 0x88ca6c00		=	0x00000000
	64bits				
	media	= -2000000000	resto	=	0
		= 0x88ca6c00		=	0x00000000
T#9	32bits				
	media	= 1294967296	resto	=	0
		= 0x4d2fa200		=	0x00000000
	64bits				
	media	= 1294967296	resto	=	0
		= 0x4d2fa200		=	0x00000000
T#10	32bits				
	media	= 1	resto	=	0
		= 0x00000001		=	0x00000000

64bits
media = 1 resto = 0
= 0x00000001 = 0x00000000

T#11 32bits
media = 1 resto = 1
= 0x00000001 = 0x00000001

64bits
media = 1 resto = 1
= 0x00000001 = 0x00000001

T#12 32bits
media = 1 resto = 8
= 0x00000001 = 0x00000008

64bits
media = 1 resto = 8
= 0x00000001 = 0x00000008

T#13 32bits
media = 1 resto = 15
= 0x00000001 = 0x0000000f

64bits
media = 1 resto = 15
= 0x00000001 = 0x0000000f

T#14 32bits
media = 2 resto = 0
= 0x00000002 = 0x00000000

64bits
media = 2 resto = 0
= 0x00000002 = 0x00000000

T#15 32bits
media = -1 resto = 0
= 0xffffffff = 0x00000000

64bits
media = -1 resto = 0
= 0xffffffff = 0x00000000

T#16 32bits
media = -1 resto = -1
= 0xffffffff = 0xffffffff

64bits

media = -1 resto = -1
= 0xffffffff = 0xffffffff

T#17 32bits
media = -1 resto = -8
= 0xffffffff = 0xfffffff8

64bits
media = -1 resto = -8
= 0xffffffff = 0xfffffff8

T#18 32bits
media = -1 resto = -15
= 0xffffffff = 0xfffffff1

64bits
media = -1 resto = -15
= 0xffffffff = 0xfffffff1

T#19 32bits
media = -2 resto = 0
= 0xfffffffe = 0x00000000

64bits
media = -2 resto = 0
= 0xfffffffe = 0x00000000