

Forum:[Hack Forums](#)

Thread: [From C to Mips32 help pleaseeeeeeeee!!!!](#)

AEJO - July 1, 2009, 11:35 am

I have done this program in c but i have difficulties to translate it to Mips32. I used gcc -march=mips32 somefile.c but doesn't work. PLEASE CAN ANYONE DO THIS FOR ME, help me pleaseeeee.

I am not so good in linux,,,,,i translated this program in using gcc -s somefile.c but in assembly x86. I want Mips32. Pleaseeeeeeee help meeeeeeeeeeeeeeeeeeeee.

```
#include <stdio.h>
#include <stdlib.h>
#define base 2
```

```
int sequenza(int);
int proba(int);
int main(){
```

```
printf("\n  L'applicazione consiste nel determinare\n ");
printf("    la probabilit  su uno lancio di moneta \n\n");
printf("L'applicazione scommette che nei prossimi n lanci non usciranno piu di due croci consecutive\n");
```

```
char c;
do{
```

```
    int numero, successo, i, seq = 1;
printf("\n\n->inserire un numero superiore o uguale a zero :");
```

```
/*inserimento numero di lancio*/
scanf("%d", &numero);
```

```
/*controllo se il numero inserito   negativo*/
while(numero < 0){
printf("Attenzione il numero inserito non   valido! inserirlo di nuovo > ");
scanf("%d", &numero);
}
```

```
/*calcolo du numero de sequenza possibile*/
seq = sequenza(numero);
```

```
/*calcolo numero di sequenza di successo*/
successo = proba(numero);
```

```
printf("\nprobabilit  di successo = %d / %d\n\n", successo, seq);
printf(" Digitare c per continuare oppure w per uscire ");
c=getchar();
```

```
}while((getchar())!='w');
```

```

return 0;
}

```

```

int proba(int n){
    if(n < 3)
        return sequenza(n);
    else
        return proba(n - 1) + proba(n - 2)+ proba(n - 3);
}

```

```

int sequenza(int a){
    int i, seq = 1;
    for(i = 1; i<= a;i++)
        seq = seq * base;
    return seq;
}

```

assembler x86

```

.file "main.c"
.section .rodata
.align 4
.LC0:
.string "\n  L'applicazione consiste nel determinare\n "
.align 4
.LC1:
.string "    la probabilit\303\240 su uno lancio di moneta \n"
.align 4
.LC2:
.string "L'applicazione scommette che nei prossimi n lanci non usciranno piu di due croci consecutive"
.align 4
.LC3:
.string "\n\n->inserire un numero superiore o uguale a zero : "
.LC4:
.string "%d"
.align 4
.LC5:
.string "Attenzione il numero inserito non \303\250 valido! inserirlo di nuovo > "
.align 4

```

```

.LC6:
.string "\nprobabilit  di successo = %d / %d\n\n"
.align 4
.LC7:
.string " Digitare c per continuare oppure w per uscire "
.text
.globl main
.type main, @function
main:
    leal 4(%esp), %ecx
    andl $-16, %esp
    pushl -4(%ecx)
    pushl %ebp
    movl %esp, %ebp
    pushl %ecx
    subl $52, %esp
    movl $.LC0, (%esp)
    call printf
    movl $.LC1, (%esp)
    call puts
    movl $.LC2, (%esp)
    call puts
.L4:
    movl $1, -24(%ebp)
    movl $.LC3, (%esp)
    call printf
    leal -12(%ebp), %eax
    movl %eax, 4(%esp)
    movl $.LC4, (%esp)
    call scanf
    jmp .L2
.L3:
    movl $.LC5, (%esp)
    call printf
    leal -12(%ebp), %eax
    movl %eax, 4(%esp)
    movl $.LC4, (%esp)
    call scanf
.L2:
    movl -12(%ebp), %eax
    testl %eax, %eax
    js .L3
    movl -12(%ebp), %eax
    movl %eax, (%esp)
    call sequenza
    movl %eax, -24(%ebp)
    movl -12(%ebp), %eax
    movl %eax, (%esp)
    call proba
    movl %eax, -16(%ebp)
    movl -24(%ebp), %eax
    movl %eax, 8(%esp)
    movl -16(%ebp), %eax

```

```

movl %eax, 4(%esp)
movl $.LC6, (%esp)
call printf
movl $.LC7, (%esp)
call printf
call getchar
movb %al, -5(%ebp)
call getchar
cmpl $119, %eax
jne .L4
movl $0, %eax
addl $52, %esp
popl %ecx
popl %ebp
leal -4(%ecx), %esp
ret
.size main, .-main
.globl proba
.type proba, @function
proba:
pushl %ebp
movl %esp, %ebp
pushl %ebx
subl $20, %esp
cmpl $2, 8(%ebp)
jg .L7
movl 8(%ebp), %eax
movl %eax, (%esp)
call sequenza
movl %eax, -8(%ebp)
jmp .L8
.L7:
movl 8(%ebp), %eax
subl $1, %eax
movl %eax, (%esp)
call proba
movl %eax, %ebx
movl 8(%ebp), %eax
subl $2, %eax
movl %eax, (%esp)
call proba
addl %eax, %ebx
movl 8(%ebp), %eax
subl $3, %eax
movl %eax, (%esp)
call proba
addl %eax, %ebx
movl %ebx, -8(%ebp)
.L8:
movl -8(%ebp), %eax
addl $20, %esp
popl %ebx
popl %ebp

```

```

ret
.size proba, .-proba
.globl sequenza
.type sequenza, @function
sequenza:
pushl %ebp
movl %esp, %ebp
subl $16, %esp
movl $1, -8(%ebp)
movl $1, -4(%ebp)
jmp .L11
.L12:
sall -8(%ebp)
addl $1, -4(%ebp)
.L11:
movl -4(%ebp), %eax
cmpl 8(%ebp), %eax
jle .L12
movl -8(%ebp), %eax
leave
ret
.size sequenza, .-sequenza
.ident "GCC: (Ubuntu 4.3.3-5ubuntu4) 4.3.3"
.section .note.GNU-stack,"",@progbits

```

g4143 - July 1, 2009, 1:05 pm

If your doing this with GCC then you need to set up a cross compiler that will compile into MIPS. try this link:

<http://linux.bytesex.org/cross-compiler.html>

<http://www.linuxselfhelp.com/HOWTO/MIPS-HOWTO-9.html>

...G4143

AEJO - July 2, 2009, 12:14 am

thank you very much.i am working with this information but is a little difficult.If anyone has done this in linux could compile it for me to mips32????? pleaseeeeeeeee