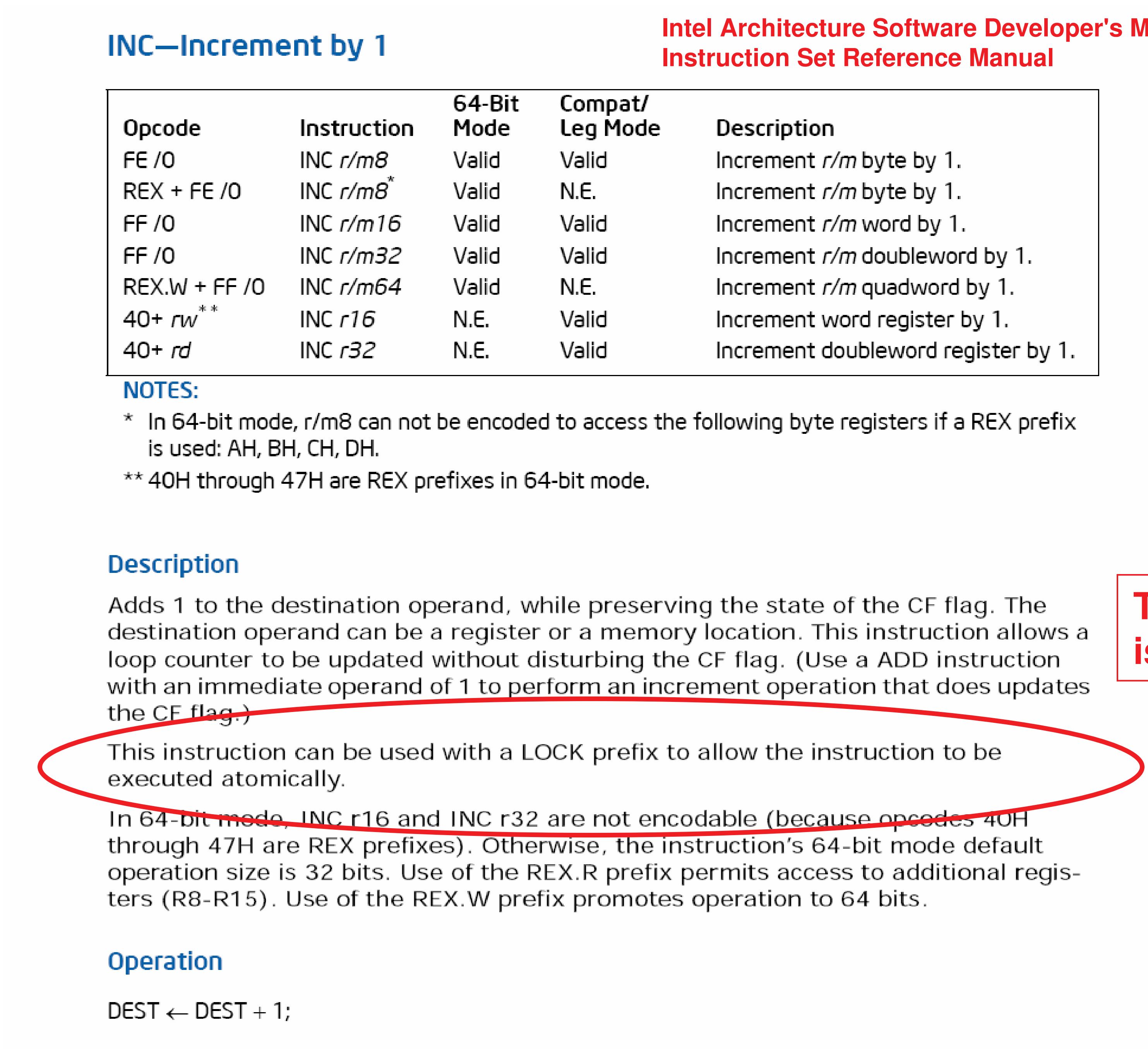
```c
/////////////////////////////////////////////////////////////////////////////
//  Adder - code executed by one of the threads
/////////////////////////////////////////////////////////////////////////////
void *Adder(void *arg)   {
    int     i;
    for (i = 0; i < LoopCount; i++)   {
        GlobalVariable++;
    }            // End of for
}                // End of Adder
```

```
.gcc ThreadsRunAmock.c -S
```

```
.globl Adder
        .type   Adder, @function
Adder:
        pushq   %rbp
        movq    %rsp, %rbp
        movq    %rdi, -8(%rbp)
        movl    $0, -12(%rbp)
.L10:
        movl    -12(%rbp), %eax
        cmpl    LoopCount(%rip), %eax
        jl      .L13
        jmp     .L11
.L13:
        incl    GlobalVariable(%rip)
        leaq    -12(%rbp), %rax
        incl    (%rax)
        jmp     .L10
.L11:
        leave
        ret
```

So why does ThreadsRunAmock.c give such strange behavior on a multiprocessor??

Intel Architecture Software Developer's Manual, Volume 2: Instruction Set Reference Manual



INC—Increment by 1

Opcode	Instruction	64-Bit Mode	Compat/ Leg Mode	Description
FE /0	INC *r/m8*	Valid	Valid	Increment *r/m* byte by 1.
REX + FE /0	INC *r/m8*[*]	Valid	N.E.	Increment *r/m* byte by 1.
FF /0	INC *r/m16*	Valid	Valid	Increment *r/m* word by 1.
FF /0	INC *r/m32*	Valid	Valid	Increment *r/m* doubleword by 1.
REX.W + FF /0	INC *r/m64*	Valid	N.E.	Increment *r/m* quadword by 1.
40+ *rw*[**]	INC *r16*	N.E.	Valid	Increment word register by 1.
40+ *rd*	INC *r32*	N.E.	Valid	Increment doubleword register by 1.

NOTES:

* In 64-bit mode, r/m8 can not be encoded to access the following byte registers if a REX prefix is used: AH, BH, CH, DH.

** 40H through 47H are REX prefixes in 64-bit mode.

Description

Adds 1 to the destination operand, while preserving the state of the CF flag. The destination operand can be a register or a memory location. This instruction allows a loop counter to be updated without disturbing the CF flag. (Use a ADD instruction with an immediate operand of 1 to perform an increment operation that does updates the CF flag.)

This instruction can be used with a LOCK prefix to allow the instruction to be executed atomically.

In 64-bit mode, INC r16 and INC r32 are not encodable (because opcodes 40H through 47H are REX prefixes). Otherwise, the instruction's 64-bit mode default operation size is 32 bits. Use of the REX.R prefix permits access to additional registers (R8-R15). Use of the REX.W prefix promotes operation to 64 bits.

Operation

DEST $\leftarrow$ DEST + 1;

This instruction is NOT atomic!!