

Forum:[Hack Forums](#)

Thread: [Missing Information in AMD64 Architecture Vol 3?](#)

DracoRexifer - March 12, 2019, 9:13 am

Page 224 shows the code for the following instructions:

MOV AL, moffset8 A0 Move 8-bit data at a specified memory offset to the AL register.
MOV AX, moffset16 A1 Move 16-bit data at a specified memory offset to the AX register.
MOV EAX, moffset32 A1 Move 32-bit data at a specified memory offset to the EAX register.
MOV RAX, moffset64 A1 Move 64-bit data at a specified memory offset to the RAX register.
MOV moffset8, AL A2 Move the contents of the AL register to an 8-bit memory offset.
MOV moffset16, AX A3 Move the contents of the AX register to a 16-bit memory offset.
MOV moffset32, EAX A3 Move the contents of the EAX register to a 32-bit memory offset.
MOV moffset64, RAX A3 Move the contents of the RAX register to a 64-bit memory offset.

In all cases, there is only one opcode byte and no values to specify an offset.
How can this be?

Infamia - March 14, 2019, 4:09 am

the opcode byte tells the processor what to expect from the incoming fetches, the opcode byte is small because it can encode operand size and SIB and other encoding specific details in that short bit field, otherwise data comes afterwards in the respective endianness

DracoRexifer - March 14, 2019, 3:22 pm

(Mar 14, 2019 04:09 AM)Infamia Wrote: [\[CITE\]](#) the opcode byte tells the processor what to expect from the incoming fetches, the opcode byte is small because it can encode operand size and SIB and other encoding specific details in that short bit field, otherwise data comes afterwards in the respective endianness

The manual is saying that the whole instruction consists of just a single byte. Though it talks about an offset, none is given in the instruction format. Just an 8-bit opcode cannot contain a 64-bit offset (or even a 32-bit offset). It's impossible. Something has to follow the opcode but the manual shows that nothing follows the opcode.

Infamia - March 15, 2019, 3:43 am

(Mar 14, 2019 03:22 PM)DracoRexifer Wrote: [\[CITE\]](#) The manual is saying that the whole instruction consists of just a single byte. Though it talks about an offset, none is given in the instruction format. Just an 8-bit opcode cannot contain a 64-bit offset (or even a 32-bit offset). It's impossible. Something has to follow the opcode but the manual shows that nothing follows the opcode.

quote the manual on this, given that x86 is a variable length instruction set i'm pretty sure you can fit full 8/16/32/64 bit offsets into an instruction, only for archs like arm where you need additional

calculation in order to store a larger number

DracoRexifer - March 15, 2019, 6:13 am

(Mar 15, 2019 03:43 AM)Infamia Wrote: [\[CITE\]](#) quote the manual on this, given that x86 is a variable length instruction set i'm pretty sure you can fit full 8/16/32/64 bit offsets into an instruction, only for archs like arm where you need additional calculation in order to store a larger number

Most of my original post was a quote that came directly from the manual. So in the following:

MOV AL, moffset8 A0 Move 8-bit data at a specified memory offset to the AL register.

The A0 is by itself. But for other instructions like the one listed below, additional data is given for the rest of the instruction. But not for A0~A3. This is why my post said "Missing information"
In the following instruction, the code is specified as "C6 /0 ib"

MOV reg/mem8, imm8 C6 /0 ib Move an 8-bit immediate value to an 8-bit register or memory operand.

But for the first instruction, all that is given is "A0". Certainly there is missing information.

Do you see what I mean or are you still not understanding my original question?