ОНЛАЙН-ОБРАЗОВАНИЕ

Меня хорошо слышно && видно?

Напишите в чат, если есть проблемы!

Ставьте + если все хорошо

OTUS

Представление данных

Команды пересылки

Пакулов Артур

1

Команды пересылки данных

Обозначения команд

Процессор 8086

Местонахождение операнда	Обозначение	Запись в ЯА
В команде	i8, i16, i32	Константное выражение
В регистре общего назначения	r8, r16	Имя регистра
В сегментном регистре	sr	cs,ds,ss,es
В ячейке памяти	m8, m16, m32	Адресное выражение

Команды пересылки данных

```
MOV op1, op2          ;        op1 = op2
```

Заносит содержимое op2 в op1.

```
Q DW 1234h
Mov ax, Q          ;ah = 12h, al = 34h
```

op1	*op2*	
r8	i8, r8, m8	пересылка байтов
m8	i8, r8	
r16	i16, r16, sr, m16	пересылка слов
sr (кроме CS)	r16, m16	
m16	i16, r16, sr	

Если op – в памяти, то команда автоматически переворачивает взятые байты

Оператор уточнения типа

```
PTR                 ;       <новый тип> PTR <выражение>

Mov [si], 0         ;       Возникает неопределённость типов операндов

                            mov byte ptr [si], 0
```

```c
#include <Windows.h>


int main()
{
    void * a;
    int *b;

    *a = 123;
    *b = 123;

    return 0;
}
```

Уточняем тип

```c
#include <Windows.h>


int main()
{
    void * a;
    int *b;

    *(BYTE *)a = 123;
    *b = 123;

    return 0;
}
```

Оператор уточнения типа

OTUS

```
PTR                    ;       <новый тип> PTR <выражение>
```

Помним про обратный порядок байт

```
org 100h

mov al, byte ptr [x]

x dd 0x11223344

ret
```

x уже в перевёрнутом виде в памяти

```
0700:0104  44 33 22 11 C3 90 90 90-90 90 90 90 90 90 90 90   D3"◄├PPPPPPPPF
0700:0114  90 90 90 90 90 90 90 90-90 F4 00 00 00 00 00 00   PPPPPPPPP⌠....
0700:0124  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
0700:0134  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

➔ В al запишется 0x44

Оператор уточнения типа

```
PTR                    ;       <новый тип> PTR <выражение>
```

Если op – в памяти, то команда автоматически переворачивает взятые байты

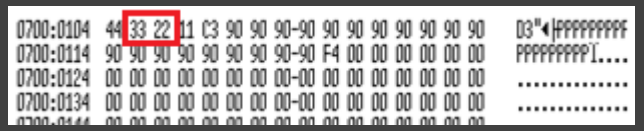
```
org 100h

mov ax, word ptr [x+1]

x dd 0x11223344

ret
```

x уже в перевёрнутом виде в памяти



➔ В ax запишется 0x2233

Оператор уточнения типа

```
PTR                    ;       <новый тип> PTR <выражение>
```

Если op – в памяти, то команда автоматически переворачивает взятые байты

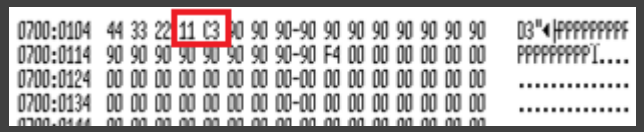
```
org 100h

mov ax, word ptr [x+3]

x dd 0x11223344

ret
```

x уже в перевёрнутом виде в памяти



➔ В ax запишется 0xC311

Оператор уточнения типа

```
PTR                    ;        <новый тип> PTR <выражение>
```

Если op – в памяти, то команда автоматически переворачивает взятые байты

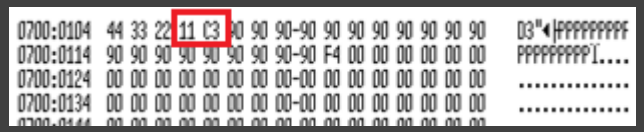
```
org 100h

mov ax, word ptr [x+3]

x dd 0x11223344

ret
```

x уже в перевёрнутом виде в памяти



➔ В ax запишется 0xC311

Команды пересылки данных

```
xchg op1, op2       ;
```

Меняет местами содержимое регистров

xchg op1,op2

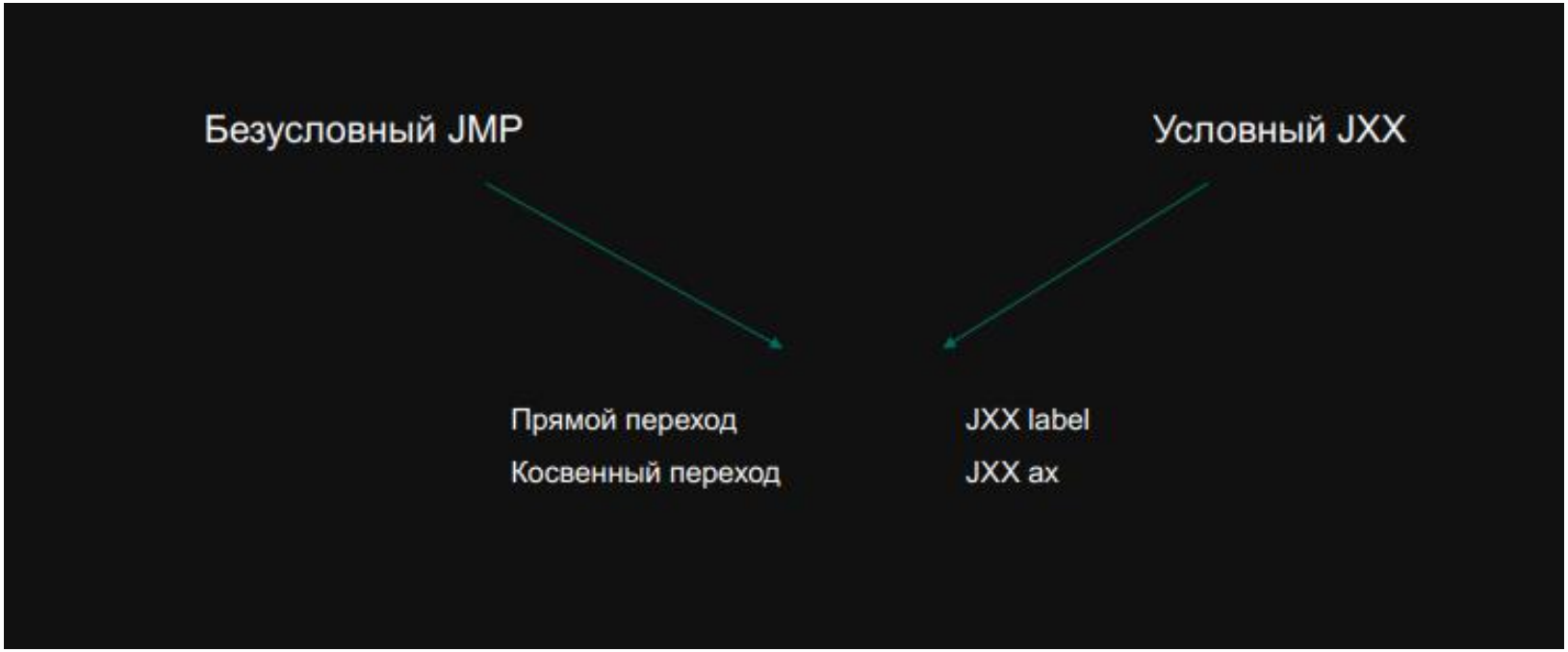
op1	*op2*	
r8	r8, m8	перестановка байтов
m8	r8	
r16	r16, m16	перестановка слов
m16	r16	

На флаги не влияет

2

Команды перехода и изменения флагов

Команды перехода



Команды изменения флагов

Есть флаги состояний и флаги условий

На ассемблере	Алгебраически
CMP op1, op2	op1-op2
Test op1, op2	op1 & op2

Результат не записывается!
Изменяются флаги!

Команды изменения флагов

Есть флаги состояний и флаги условий

Команда	Значение
Stc/clc	Установить/сбросить CF флаг
Cmc	Инвертировать CF
Sti/cli	Установить/сбросить IF флаг
Std/cld	Установить/сбросить DF флаг
Sahf/Lahf	Сохранить/Загрузить флаги (*) из/в ah

```
AH = EFLAGS(SF,ZF,0,AF,0,PF,1,CF);
```

Команды перехода

Циклы

Loop Label

```
    MOV CX, 10
    XOR AX, AX
L:  INC AX
    LOOP L
```

Cx – счётчик повторов

Jcxz after – проверка регистра cx

Досрочный выход из цикла:

Loope/loopz – по счётчику и пока == 0

3

Представление данных

Представление данных

- Целые числа без знака
- Целые числа со знаком
- Вещественные числа
- Символы

Знаковые и беззнаковые числа

OTUS

```
N = 2^k
```

Беззнаковое число использует все разряды типа

Знаковое число занимает старший бит под знак

Дополнительный код

Отрицательные числа записываются процессором в виде дополнительного кода

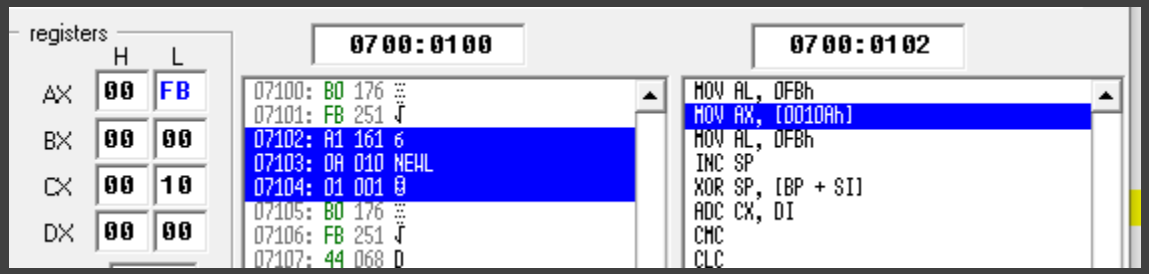
ДК = обратный код + 1

Пример:

```
-5 = 00000101b                ; прямой код

Not 00000101b = 11111010      ; обратный код

11111010 + 1 =  0xFB          ; дополнительный код
```



4

Представление команд

Представление команд

OTUS

- Регистр – регистр (2 байта)
- Регистр – память (2-4 байта)
- Регистр – непосредственный операнд (3-4 байта)
- Память – непосредственный операнд (3-6 байт)

Представление команд

Регистр – регистр (2 байта)

Reg1 = Reg1 x Reg2
Reg2 = Reg2 X Reg1

w= { 1 - слова ; 0 - байты } d= { 1 - reg1:=reg1*reg2 ; 0 - reg2:=reg2*reg1 }

Представление команд

Регистр – регистр (2 байта)

Reg1 = Reg1 x Reg2
Reg2 = Reg2 X Reg1

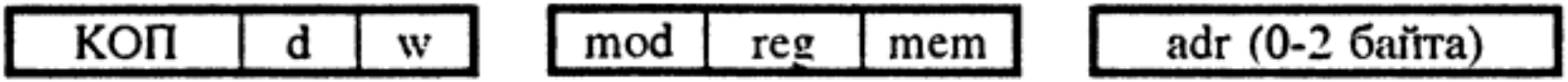
reg	*w=1*	*w=0*
000	AX	AL
001	CX	CL
010	DX	DL
011	BX	BL

reg	*w=1*	*w=0*
100	SP	AH
101	BP	CH
110	SI	DH
111	DI	BH

Представление команд

Регистр – память (2-4 байта)

Reg = Reg*addr
Addr = Addr*Reg



mem \ mod	*00*	*01*	*10*
000	[BX]+[SI]	[BX]+[SI]+a8	[BX]+[SI]+a16
001	[BX]+[DI]	[BX]+[DI]+a8	[BX]+[DI]+a16
010	[BP]+[SI]	[BP]+[SI]+a8	[BP]+[SI]+a16
011	[BP]+[DI]	[BP]+[DI]+a8	[BP]+[DI]+a16
100	[SI]	[SI]+a8	[SI]+a16
101	[DI]	[DI]+a8	[DI]+a16
110	a16	[BP]+a8	[BP]+a16
111	[BX]	[BX]+a8	[BX]+a16

Представление команд

Регистр – непосредственный операнд (3-4 байта)

Reg = Reg x Im

Представление команд

Память – непосредственный операнд (3-4 байта)

Addr = Addr * Im

Представление команд

Память – непосредственный операнд (3-4 байта)

Addr = Addr * Im

Используем Capstone

Дизассемблируем инструкции

```
# test1.py
from capstone import *

CODE = b"\x55\x48\x8b\x05\xb8\x13\x00\x00"

md = Cs(CS_ARCH_X86, CS_MODE_64)
for i in md.disasm(CODE, 0x1000):
    print("0x%x:\t%s\t%s" %(i.address, i.mnemonic, i.op_str))
```

```
$ python test1.py

0x1000: push    rbp
0x1001: mov     rax, qword ptr [rip + 0x13b8]
```

https://www.capstone-engine.org/lang_python.html
https://www.capstone-engine.org/documentation.html

OTUS

Вопросы???

Пакулов Артур

A.Pakulov.Otus@Gmail.com
Inst: @ArturPakulov

Спасибо
за внимание!