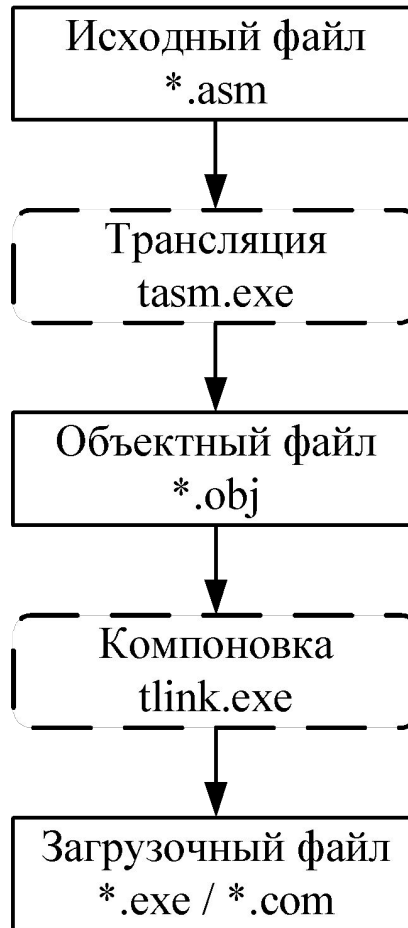


# Ассемблер Intel 8086

## Выполнение трансляции и компоновки



# Ассемблер Intel 8086

## Компоненты и структура программы

Пример 1. Программа читает с клавиатуры три символа, уменьшает их коды на 1 и отображает на экране результат преобразования.

```
dosseg
.model small
.stack 200h
.data
    DisplayString db 13, 10
    ThreeChars db 3 dup(?)
                db '$'
.code
Begin:
    mov ax, @Data
    mov ds, ax

    mov bx, offset ThreeChars
    mov ah, 1
    int 21h
```

```
    dec al
    mov [bx], al
    inc bx
    int 21h
    dec al
    mov [bx], al
    inc bx
    int 21h
    dec al
    mov [bx], al
    mov dx, offset DisplayString
    mov ah, 9
    int 21h
```

```
    mov ax, 4C00h
    int 21h
```

```
end Begin
```

# Ассемблер Intel 8086

## Сегментные директивы

Упрощённые директивы:

DOSSEG – определяет порядок следования сегментов

.MODEL – задание модели памяти

.DATA – сегмент данных

.CODE – сегмент кода

.STACK – определяет размер сегмента стека

# Ассемблер Intel 8086

## Сегментные директивы: модели памяти

код данные	< 64 КБайт	> 64 КБайт
< 64 КБайт	tiny small	medium
> 64 КБайт	compact	large huge

Примечания:

- 1) **tiny** – код и данные располагаются в одном сегменте, **small** – код и данные могут располагаться в разных сегментах;
- 2) **large** – массивы не могут быть больше 64 Кбайт, **huge** – массивы могут значительно превышать размер 64 КБайта.

# Ассемблер Intel 8086

## Режимы адресации данных

Режим	Формат операнда	Регистр сегмента	Примеры
1. Непосредственный	константа	не используется	mov ax, 1
2. Прямой	метка, смещение	DS	dec cnt
3. Регистровый	регистр	не используется	mov ds, ax
4. Регистровый косвенный	[BX], [SI], [DI], [BP]	DS DS (ES) SS	mov al, [bx] inc [di] mov cl, [bp]
5. Регистровый относительный	[BX+смещение], [SI+смещение], [DI+смещение], [BP+смещение]	DS DS DS (ES) SS	mov ah, [bx+6]

# Ассемблер Intel 8086

## Режимы адресации данных

Режим	Формат операнда	Регистр сегмента	Примеры
6. Базовый индексный	[BX+SI] [BX+DI] [BP+SI] [BP+DI]	DS DS SS SS	mov [bx+di], dx
7. Относительный базовый индексный	[BX+SI+смещение] [BX+DI+смещение] [BP+SI+смещение] [BP+DI+смещение]	DS DS SS SS	mov al, [bp+si+ChStr+2]

# Ассемблер Intel 8086

## Инициализация данных: директивы

**DB** – 1 байт

**DW** – 1 слово (2 байта)

**DD** – двойное слово (4 байта)

**DF, DP** – 6 байтов (для i386 и старше)

**DQ** – 8 байтов

**DT** – 10 байтов

# Ассемблер Intel 8086

## Инициализация данных: примеры

### 1. Инициализация массивов:

а) массив из 8 элементов типа «двойное слово»:

```
DArray DD 0, 1, 2, 3, 4  
        DD 5, 6, 7
```

б) массив из ста нулей:

```
WArray DW 100 DUP(0)
```

в) массив из 50 кодов '0':

```
BArray DB 50 DUP('0')
```

г) массив из 19 любых элементов:

```
SArray DW 19 DUP(?)
```



# Ассемблер Intel 8086

## Инициализация данных: примеры

### 2. Инициализация строки

```
String1 DB 'A', 'B', 'C', 'D'
```

```
String2 DB 'ABCD'
```

```
; String1 = String2
```

```
String3 DB 'Line', 0Dh, 0Ah, '$'
```

# Ассемблер Intel 8086

## Именованные области памяти

Типы меток:

- |          |                 |
|----------|-----------------|
| 1) BYTE  | 2) WORD         |
| 3) DWORD | 4) FWORD, PWORD |
| 5) QWORD | 6) TBYTE        |
| 7) NEAR  | 8) FAR          |
| 9) PROC  | 10) UNKNOWN     |

# Ассемблер Intel 8086

## Именованные области памяти: примеры

1. KeyBuffer LABEL BYTE  
DB 20 DUP(?)

2. .Data

WordVar LABEL WORD

ByteVar DB 1, 2

.Code

mov AX, [WordVar] ;AH = 2, AL = 1

mov DL, [ByteVar]