Лекція №16

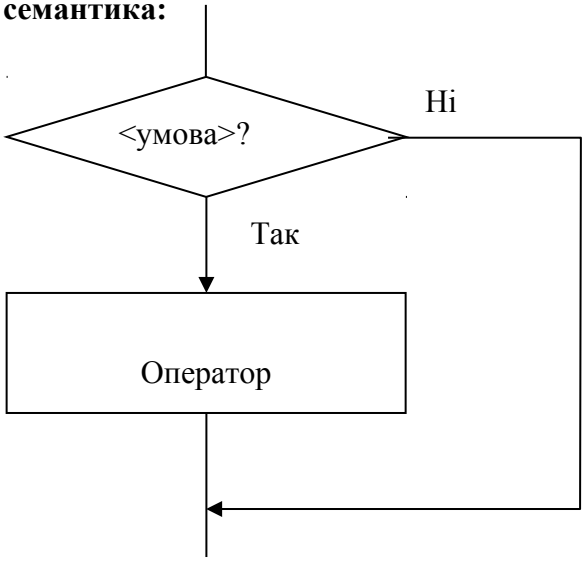
Генерація коду для управляючих конструкцій

Генерація коду для неповної умовної конструкції if-then

Синтаксис:

1. <умовна конструкція> → if <умова> then <оператор>
2. <умова> → <вираз1><операція порівняння><вираз2>
3. <операція порівняння> → >
4. <операція порівняння> → >=
5. <операція порівняння> → <
6. <операція порівняння> → <=
7. <операція порівняння> → =
8. <операція порівняння> → <>

Неформальна семантика:



Команди умовного переходу мови асемблера Intel86, що відповідають операціям порівняння:

>	→	JG	<мітка>
≥	→	JGE	<мітка>
<	→	JL	<мітка>
≤	→	JLE	<мітка>
=	→	JE	<мітка>
<>	→	JNE	<мітка>

Приклад.

Розглянемо семантичне визначення для неповної умовної конструкції if-then.

if a>b then x:=y

Припустимо, що всі змінні мають тип integer.

```
        MOV  AX,a   }
        MOV  BX,b   }  a>b
        CMP  AX,BX  }
        JLE  ?L0      ; якщо a<=b, то перехід ?L0
        MOV  AX,y   }
        MOV  x,AX   }  x:=y
?L0:    NOP
```

Зовнішні мітки – це мітки, які програміст вказує в тексті вхідної програми.

Внутрішні мітки не вказуються в початковому тексті, а генеруються компілятором для реалізації управляючих конструкцій.
? – символ в іменах асемблера Intel

Структура внутрішньої мітки (приклад):

?<буква><рядок цифр>

'? ' + 'L' + '0' → ?L0

'? ' + 'M' + '21' → ?M21

Розглянемо порядок обходу дерева розбору і генерації коду для умовної конструкції if-then:

1. Спуск по піддереву нетерміналу <умова> правила 1 (* SPR(k2) *) під час якого генеруються команди:

   MOV AX,<вираз1>

   MOV BX,<вираз2>

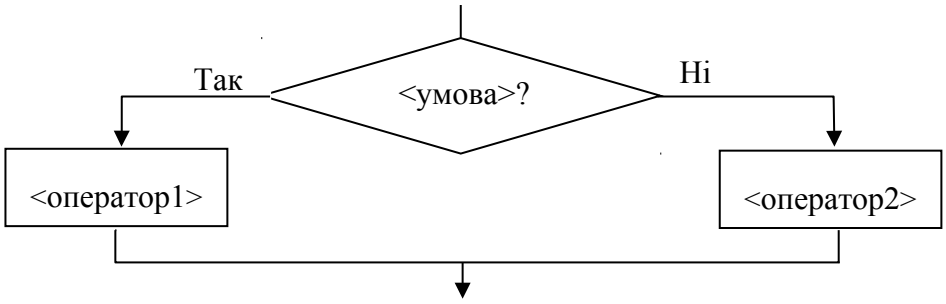
   CMP AX,BX
2. Генерація внутрішньої мітки, наприклад, ?L0 .
3. Генерація команди умовного переходу за умовою, оберненою заданій, на внутрішню мітку: JLE ?L0
4. Спуск по піддереву нетерміналу <оператор> правила 1 (* SPR(k1) *), під час якого генерується команди реалізації цього оператора.
5. Генерація команди NOP з внутрішньою міткою ?L0: ?L0: NOP

Генерація коду для повної умовної конструкції if-then-else

Синтаксис:

1. <умовна конструкція> → if <умова> then <оператор1> else <оператор2>
2. <умова> → <вираз1><операція порівняння><вираз2>
3. <операція порівняння> → >
4. <операція порівняння> → >=
5. <операція порівняння> → <
6. <операція порівняння> → <=
7. <операція порівняння> → =
8. <операція порівняння> → <>

Неформальная семантика:



Приклад.
Розглянемо семантичне визначення для повної умовної конструкції if-then-else.

```
if a>b then x:=y
       else  y:=x

        MOV  AX,a
        MOV  BX,b      } a>b
        CMP  AX,BX     }
        JLE  ?L1       }
        MOV  AX,y      } x:=y
        MOV  x,AX      }
        JMP  ?L2
?L1:    NOP
        MOV  AX,x      } y:=x
        MOV  y,AX      }
?L2:    NOP
```

Розглянемо порядок обходу дерева розбору і генеріції коду для повної умовної конструкції if- then-else:

1. <умовна конструкція> → if <умова> then <оператор>elsw<оператор>
   {[3] Jxx ?L1 [2] JMP ?L2 ?L1:NOP [1] ?L2:NOP}
2. <умова> → <вираз><операція порівняння><вираз>
   {Reg:=’AX’ [3] Reg:=’BX’ [1] [2] CMP AX,BX}
3. <операція порівняння> → >
4. <операція порівняння> → >=
5. <операція порівняння> → <
6. <операція порівняння> → <=
7. <операція порівняння> → =
8. <операція порівняння> → <>

   } {Zn=’символи операції’}
9. <вираз> → <змінна>{[1] MOV Reg,BF}
10. <змінна> → a
11. <змінна> → b
12. <змінна> → x
13. <змінна> → y

    } {BF:=’змінна’}
14. <оператор> → <змінна>:=<вираз>{Reg:=’AX’ [1] [2] MOV BF,Reg}

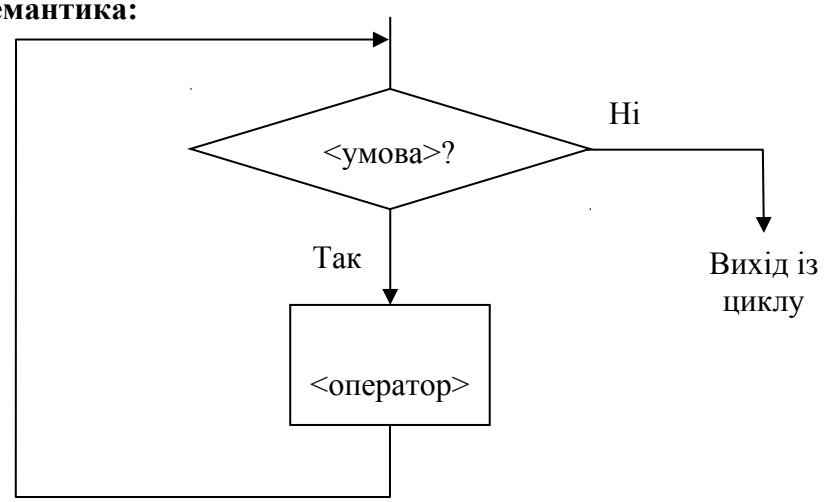
1. Спуск по піддереву <умова> (* SPR(k3) *), під час якого генеруються команди:
   ```
   MOV AX,<вираз1>
   MOV BX,<вираз2>
   CMP AX,BX
   ```
2. Генерація внутрішньої мітки, наприклад, ?L1 .
3. Генерація команди умовного переходу на внутрішню мітку ?L1, наприклад, JLE ?L1
   Вибір команди порівняння залежить від <операції порівняння>.
4. Спуск по піддереву <оператор1> гілки then (* SPR(k2) *), під час якого генеруються команди реалізації цього оператора:
   ```
   MOV  AX,y
   MOV  x,AX
   ```
5. Генерація внутрішньої мітки для виходу із умовного оператора, наприклад ?L2 .
6. Генерація команди безумовного переходу JMP на мітку ?L2: JMP ?L2
7. Генерація команди NOP з внутрішньою міткою ?L1: ?L1: NOP
8. Спуск по піддереву <оператор2> гілки else (* SPR(k1) *), під час якого генеруються команди реалізації цього оператора:
   ```
   MOV AX,x
   MOV y,AX
   ```
9. Генерація команди NOP з внутрішньою міткою ?L2: ?L2: NOP

Генерація коду для конструкції циклу з передумовою while

Синтаксис:

1. <цикл-while> → while <умова> do <оператор>
2. <умова> → <вираз1><операція порівняння><вираз2>
3. <операція порівняння> → >
4. <операція порівняння> → >=
5. <операція порівняння> → <
6. <операція порівняння> → <=
7. <операція порівняння> → =
8. <операція порівняння> → <>

Неформальна семантика:



Приклад.
Розглянемо семантичне визначення для конструкції циклу з передумовою while.

while a>b do begin x:=y; y:=z end

```
?L1:   NOP
       MOV  AX,a
       MOV  BX,b
       CMP  AX,BX
       JLE   ?L2
       MOV  AX,y
       MOV  x,AX
       MOV  AX,z
       MOV  y,AX
       JMP   ?L1
?L2:   NOP
```

Розглянемо порядок обходу дерева розбору і генерації коду для конструкції циклу з передумовою while:

1. Генерація внутрішньої мітки, наприклад, ?L1 .
2. Генерація команди NOP з внутрішньою міткою ?L1: ?L1: NOP
3. Спуск по піддереву <умова> (* SPR(k2) *), під час якого генеруються команди обчислення цієї умови:
   ```
   MOV  AX,<вираз1>
   MOV  BX,<вираз2>
   CMP  AX,BX
   ```

4. Генерація внутрішньої мітки для виходу із циклу, наприклад, ?L2 .
5. Генерація команди умовного переходу на внутрішню мітку ?L2 за умовою, оберненою заданій, наприклад, JLE ?L2.
6. Спуск по піддереву <оператор> (* SPR(k1) *), під час якого генеруються команди реалізації цього оператора:

   ```
   MOV  AX,y
   MOV  x,AX
   MOV  AX,z
   MOV  y,AX
   ```
7. Генерація команди безумовного переходу JMP на початок циклу, тобто на мітку ? L1:

   ```
   JMP  ?L1
   ```
8. Генерація команди NOP з внутрішньою міткою ?L2: ?L2: NOP