20 Язык инструкций IL (Instuctions List)

20.1 Введение

Программа на языке **IL** представляет собой список команд и предназначена для создания небольших по размеру высокоэффективных процедур.

Программа должна начинаться с ключевого слова `Program` за которым следует имя программы, и заканчиваться ключевым словом `End_Program`.

Блок объявления типов данных может начинаться с ключевых слов `Var` , `Var_Input`, `VAR_Output, VAR_In_Out` и заканчиваться ключевым словом `End_`Var.

Каждая команда должна начаться с новой строки, и может содержать оператор с дополнительными модификаторами.

Команды всегда касаются только текущего результата (регистр IL). Оператор указывает операцию, которая должна быть сделана между текущим результатом и операндом.
Результат операции снова сохраняется в текущем результате.

Команде может предшествовать метка с двоеточием (:). Метки должны соответствовать следующим правилам:
- длина не может превышать 16 символов;
- первым символом должна быть буква латинского алфавита;
- далее могут следовать буквы, цифры или символы подчеркивания.

Имя метки может использоваться один раз в программе. Метка может иметь такое же имя, как переменная.

Комментарий должен быть последним элементом строки. Между командами могут быть вставлены пустые строки.
Примеры полей команды

```
Метка       Оператор    Операнд      Комментарий
Start:      LD   IX1   (* проверить кнопку start *)
            ANDN MX5   (* команда не запрещена *)
      ST    QX2   (* start motor *)
```

20.2 Операторы, модификаторы и операнды

В синтаксисе языка IL имеются следующие модификаторы:

N	двоичное отрицание операнда
(	Отсроченная операция
C	Условная операция

Символ модификатора должен заканчивать имя оператора без знаков пробела между ними.

Модификатор **`'N'`** вызывает двоичное отрицание операнда. Например, команда `ORN IX12` интерпретируется как:

```
result := result OR NOT (IX12)
```

Модификатор **`круглая скобка '('`** указывает, что оценка команды должна быть отсрочена до заключительной круглой скобки ')'.

Модификатор **`'C'`** указывает, что команда должна быть выполнена, если текущий результат имеет двоичное значение `TRUE` (или 1 для небулевых значений).

Модификатор **`'C'`** может быть объединен с модификатором **`'N'`**, чтобы указать, что команда должна быть выполнена, если текущий результат имеет двоичное значение `FALSE` (или 0 для небулевых значений).

20.2.1 Операторы работы с памятью

Оператор LD загружает значение в текущий результат

Допустимые модификаторы N
Операнд константы, переменные

Примеры:

```
     LD   false     (* результат:=boolean constant *)
     LD   123       (*результат := integer constant *)
     LD   123.1     (*результат := float constant *)
     LD   t#3ms     (*результат := time constant *)
     LD   boo_var1  (*результат := boolean variable *)
     LD   tmr_var1  (*результат := timer variable *)
     LDN  boo_var2  (*результат := NOT ( boolean variable ) *)
```

Оператор ST сохраняет текущий результат в переменной, текущий результат не изменяется этой операцией.
Допустимые модификаторы N
Операнд переменные
Примеры:

```
STboo:    LD   false
     ST   boo_var1  (* boo_var1 := FALSE *)
     STN  boo_var2  (* boo_var2 := TRUE *)
STana:    LD   123
     ST   ana_var1  (* ana_var1 := 123 *)
STtmr:    LD   t#12s
     ST   tmr_var1  (* tmr_var1 := t#12s *)
```

20.2.2 Операторы управляющих конструкций

Оператор S устанавливает значение TRUE в переменной типа Boolean, если текущий результат имеет значение TRUE. Операция не выполняется, если текущий результат FALSE. Текущий результат не изменяется этой операцией.
Допустимые модификаторы нет
Операнд переменные типа Boolean

Примеры:

```
SETex:    LD   true (* текущий результат := TRUE *)
     S    boo_var1  (* boo_var1 := TRUE *)
               (*текущий результат не модифицируется*)
     LD   false     (* current result := FALSE *)
     S    boo_var1  (* операция не выполняется *)
```

Оператор R устанавливает значение FALSE в переменной типа Boolean, если текущий результат имеет значение TRUE. Операция не выполняется, если текущий результат FALSE. Текущий результат не изменяется этой операцией.
Допустимые модификаторы нет
Операнд переменные типа Boolean

Примеры:

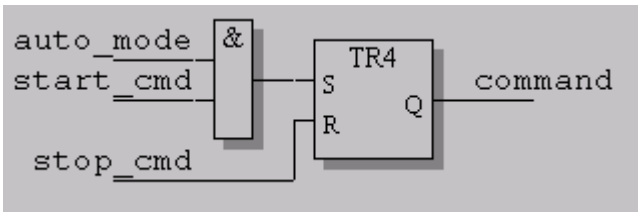
```
RESETex:  LD   true (*текущий результат := TRUE *)
     R    boo_var1  (* boo_var1 := FALSE *)
               (* current result is not modified *)
     ST   boo_var2  (* boo_var2 := TRUE *)
```

```
     LD   false     (*текущий результат:= FALSE *)
     R    boo_var1  (*текущий результат не модифицируется *)
```

Оператор CAL вызов внешних подпрограмм
Допустимые модификаторы C, C N
Операнд Название внешней подпрограммы.
Входные параметры подпрограмм должны быть назначены перед вызовом, используя последовательность операций ST.

```
     LD   auto_mode
     AND  start_cmd
     ST   TR4.S
     LD   stop_cmd
     ST   TR4.R
     CAL  TR4
     LD   TR4.Q
     ST   command
```

(* FBD эквивалент : *)



Оператор JMP переход к указанной метке
Допустимые модификаторы C, C N
Операнд Метка, определенная в программе IL

Пример:

```
( * следующий пример проверяет значение селектора (0 или 1) *)
( *, чтобы установить один из 2-х выходов типа boolean. *)
JMPex: LD selector
     JMPC test2
     LD   true
     ST   bo0
     JMP  JMPend
test2: LD true
     ST   bo1
JMPend:(* end of the IL program *)
```

Оператор RET заканчивает текущий список команды. Если IL последовательность - подпрограмма, текущий результат возвращается к вызывающей программе
Допустимые модификаторы C, C N
Операнды нет
Пример

```
(*Следующий пример проверяет значение аналогового селектора (0 или
1 или 2)   *)
( *, чтобы установить один из 3-х выходов типа boolean *)
JMPex: LD selector(* селектор 0 или 1 or 2 *)
     JMPC test1     (* если селектор = 0 тогда *)
     LD   true
```

```
     ST   bo0  (* bo0 := true *)
     RET       (* end - return 0 *)
               (* уменьшаем селектор *)
test1:    LD   selector
     SUB  1    (*селектор 0 или 1 *)
     JMPC test2     (* если селектор = 0 тогда *)
     LD   true
     ST   bo1  (* bo1 := true *)
     LD   1    (* загрузка значения селектора *)
     RET       (* end - return 1 *)
               (*  *)
test2: RETNC   (* возвращаемое значение селектора *)
               (* как недействительное значение *)
     LD   true
     ST   bo2  (* bo2 := true *)
     LD   2    (* load real selector value *)
               (* end - return 2 *)
```

Оператор) выполняет отсроченную операцию, которая была отмечена символом '('
Модификаторы нет
Операнды нет

Пример:

```
(*Следующая программа демонстрирует отсроченные операции: *)
(* res := a1 + (a2 * (a3 - a4) * a5) + a6; *)
Delayed: LD    a1   (* результат := a1; *)
     ADD( a2   (* отсрочено ADD - результат:= a2; *)
     MUL( a3   (*отсрочено MUL - результат := a3; *)
     SUB  a4   (*отсрочено := a3 - a4; *)
     )         (* выполнение отсроченного MUL - результат := a2 *
(a3-a4); *)
     MUL  a5   (*результат := a2 * (a3 - a4) * a5; *)
     )         (*выполнение отсроченного ADD *)
               (*результат:= a1 + (a2 * (a3 - a4) * a5); *)
     ADD  a6   (*результат := a1 + (a2 * (a3 - a4) * a5) + a6; *)
     ST   res  (* запоминание результата в переменной res *)
```

20.2.3 Логические операторы

Оператор AND (&) логическое И текущего результата и операнда
Допустимые модификаторы N, (
Операнды переменные и константы типа Boolean

Пример:

```
     LD   auto_mode
     AND  start_cmd
     JMPC start
     JMP  exit_cmd
Start:
(*программа START*)
     ...
Exit_cmd: Ret
```

Оператор OR логическое ИЛИ текущего результата и операнда
Допустимые модификаторы N, (
Операнды переменные и константы типа Boolean

Пример:

```
     LD   stop_cmd
     OR   err
     JMPC stop
Work:
(*работа*)
…
Stop:
(*останов*)
```

Оператор XOR логическое исключающее ИЛИ текущего результата и операнда
Допустимые модификаторы N, (
Операнды переменные и константы типа Boolean

20.2.4 Арифметические операторы

Все арифметические операторы качестве операндов могут использовать переменные **SHORTINT, WORD, DWORD, INTEGER, TIMER, FLOAT**. Переменные каждой отдельной операции должны иметь одинаковый тип.

Операция SQRT использует в качестве операнда переменные типа FLOAT.

Предупреждение !

При арифметических операциях учитывайте ожидаемые диапазоны результатов вычислений, которые не должны выходить из допустимых значений переменных.

Оператор ADD арифметическое сложение
Допустимые модификаторы N, (

Оператор SUB арифметическое вычитание
Допустимые модификаторы N , (

Оператор MUL арифметическое умножение
Допустимые модификаторы N , (

Оператор DIV арифметическое деление
Допустимые модификаторы N , (

Оператор SQRT извлечение квадратного корня
Допустимые модификаторы N, (

20.2.5 Операторы сравнения

Все операции сравнения в качестве операндов могут использовать переменные типа **SHORTINT, WORD, DWORD, INTEGER, TIMER , FLOAT**. Переменные каждой отдельной операции должны иметь одинаковый тип. Результатом операций сравнения является переменные типа **BOOLEAN.** После инициализации программы эти переменные принимают значение **FALSE**, кроме тех глобальных переменных, значение которых при инициализации явно объявлены как **TRUE**.

Оператор GT Сравнение больше чем

Модификатор (
Пример

```
LD   cmd
GT   cur_value
JMPC ex_GT          (*переход , если больше*)
```

Оператор GE Сравнение больше или равно
Модификатор (
Пример

```
LD   cmd
GE   cur_value
JMPC ex_GE          (*переход , если больше или равно*)
```

Оператор EQ Сравнение равно
Модификатор (
Пример

```
LD   cmd
EQ   cur_value
JMPC ex_EQ          (*переход , если равно*)
```

Оператор NE Сравнение не равно
Модификатор (
Пример

```
LD   cmd
NE   cur_value
JMPC ex_NE          (*переход , если не равно*)
```

Оператор LE Сравнение меньше или равно
Модификатор (
Пример

```
LD   cmd
LE   cur_value
JMPC ex_LE          (*переход , если меньше или равно*)
LT   cur_value
JMPC ex_LT          (*переход , если меньше*)
```

Оператор LT Сравнение меньше чем
Модификатор (
Пример

```
LD   cmd
LT   cur_value
JMPC ex_LT          (*переход , если меньше*)
```