1.2.3. FBD-мова

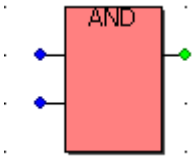
Мова Функціональних блокових діаграм FBD (Functional Block Diagram) або функціональних блоків призначена для інженерів-технологів, які розв'язують задачі керування технологічними процесами. FBD-мова дозволяє створити програмну одиницю практично будь-якої складності на основі стандартних «цеглинок» (арифметичні, тригонометричні та логічні блоки, ПІД-регулятори і т.д.), які сполучаються між собою за допомогою входів і виходів.

Коріння мови з'ясувати складно, проте більшість фахівців вважають, що це не що інше, як перенесення ідей мови релейно-контактних схем на іншу елементну базу. Дійсно, замість реле в цій мові використовуються функції, функціональні блоки, а також стандартні оператори у вигляді функцій, які знаходяться у бібліотеках. Додатково користувач може створювати власні блоки, які він будує із стандартних елементів або С-мовою і оформляє їх, як об'єкти бібліотек.

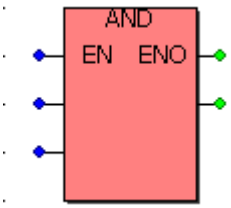
1.2.3.1. Синтаксис FBD-мови

FBD – програма (діаграма) – це блок-схема з певної кількості елементарних функцій та функціональних блоків, які зображаються прямокутниками.

Функції – це комбінаційні схеми. Вони не мають внутрішніх умов, тому при кожному їх виконанні одному і тому ж значенню вхідних величин відповідають ті ж самі значення вихідної величини, як це трапляється, наприклад, з функцією складання двох величин. Функція має декілька входів і тільки один вихід. Входи завжди розташовані зліва, виходи - справа. Назва функції записується усередині прямокутника. Графічне зображення функції має вигляд:



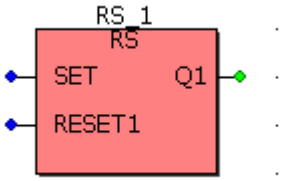
У деяких функціях перший вхід або перший вихід не є булевим. Тому в процесі програмування при необхідності можна штучно створити логічний вхід '**EN**' і логічний вихід '**ENO**'**.**



Тоді алгоритм, визначений функцією, буде виконуватися, коли вхід '**EN**' дорівнює TRUE. Після виконання алгоритму без помилок значення виходу '**ENO**' автоматично стає рівним одиниці. Коли виникають помилки при виконанні алгоритму, значення '**ENO**' стає рівним нулю.

Функціональні блоки – це програмні одиниці з багатьма вхідними та вихідними параметрами і оперативною пам'яттю. Тому вони можуть мати різний стан, завдяки чому при неодноразових виконаннях функціональних блоків одним і тим же значенням вхідних величин можуть відповідати різні

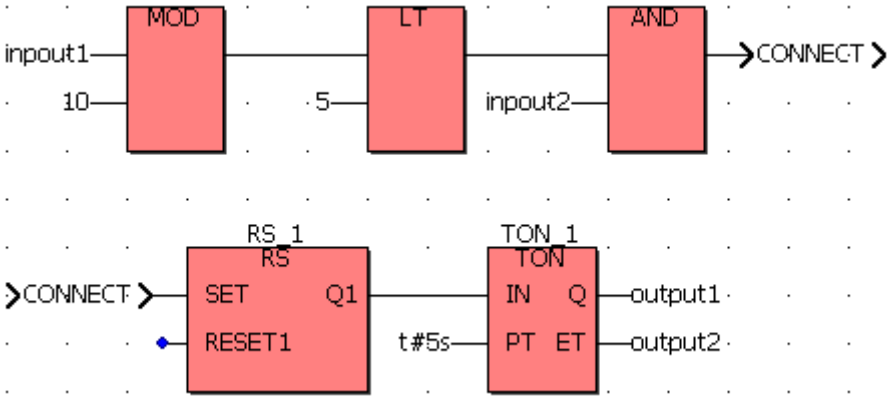
значення на виході. Наприклад, значення виходу лічильника збільшується, тобто має бути різним при незмінних одиничних вхідних імпульсах. Графічне зображення функціонального блока має вигляд:



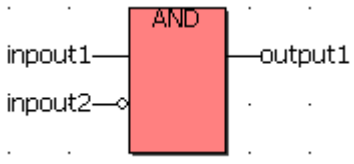
Назва функціонального блока, а також назви усіх входів і виходів записуються всередині прямокутника. Над блоком показується назва екземпляра функціонального блока.

Входом FBD- блока може бути константа, будь-яка внутрішня вхідна або вихідна змінна. Виходом – будь-яка внутрішня або вихідна змінна, або ім'я функції (тільки для функцій).

Вхідні та вихідні змінні сполучаються з відповідними входами та виходами блок. При цьому тип кожної змінної такий самий, як і тип відповідного входу або виходу. Вхідні і вихідні змінні, входи і виходи блоків сполучаються разом лініями або з'єднувачами. З'єднувачі (connectors) використовуються при обмеженій ширині екрана, коли є необхідність розірвати сполучення і перенести його в наступну мережу. Тому FBD-програма не обов'язково зображає собою велику єдину схему, частіше вона має вигляд декількох мереж, які послідовно виконуються.



Одинична лінія правим кінцем, приєднаним до входу блока, може закінчуватися логічним запереченням. Заперечення зображається маленьким кільцем.



(* ST еквівалент: *)

output1 := input1 AND NOT (input2).

Коли використовується логічне заперечення, лівий і правий кінці лінії зв'язку повинні мати тип BOOL.

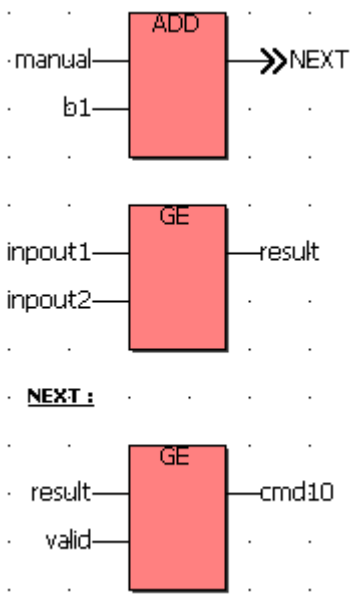
Виконання FBD-програми здійснюється зліва направо, зверху вниз. Блок починає обчислюватися тільки після того, як будуть знайдені значення усіх

його входів. Подальші обчислювання не будуть продовжені поки значення на всіх виходах не будуть обчислені. Обчислення мережі програми є закінченим тільки після обчислення значень на виходах усіх складових елементів.

Послідовність виконання FBD-програми можна примусово змінити, використовуючи мітки та переходи, які обов'язково мають однакові імена.

Мітка з двома крапками вставляється попереду будь-якої мережі і стає її назвою, а перехід на мітку у вигляді стрілки показується біля виходу блока, з якого треба змінити хід програми. Перехід обов'язково зв'язаний з логічною змінною і виконується, коли вона має значення TRUE. Для створення безумовного переходу використовується константа TRUE, що зв'язана з переходом.

Наприклад:

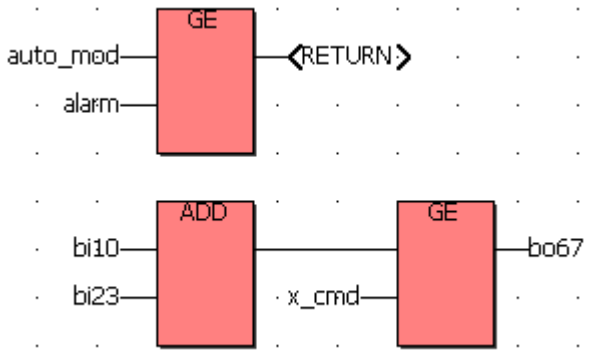


(* IL еквівалент: *)

```
LD      manual
ADD     b1
JMPC    NEXT
LD      input1
OR      input2
ST      result
LD      result
OR      valid
ST      cmd10
```

Для виходу із програми використовується ключове слово <RETURN>.

Якщо вихід блока, сполученого з оператором <RETURN>, має тип TRUE, решта частини діаграми не виконується[4,7-9].



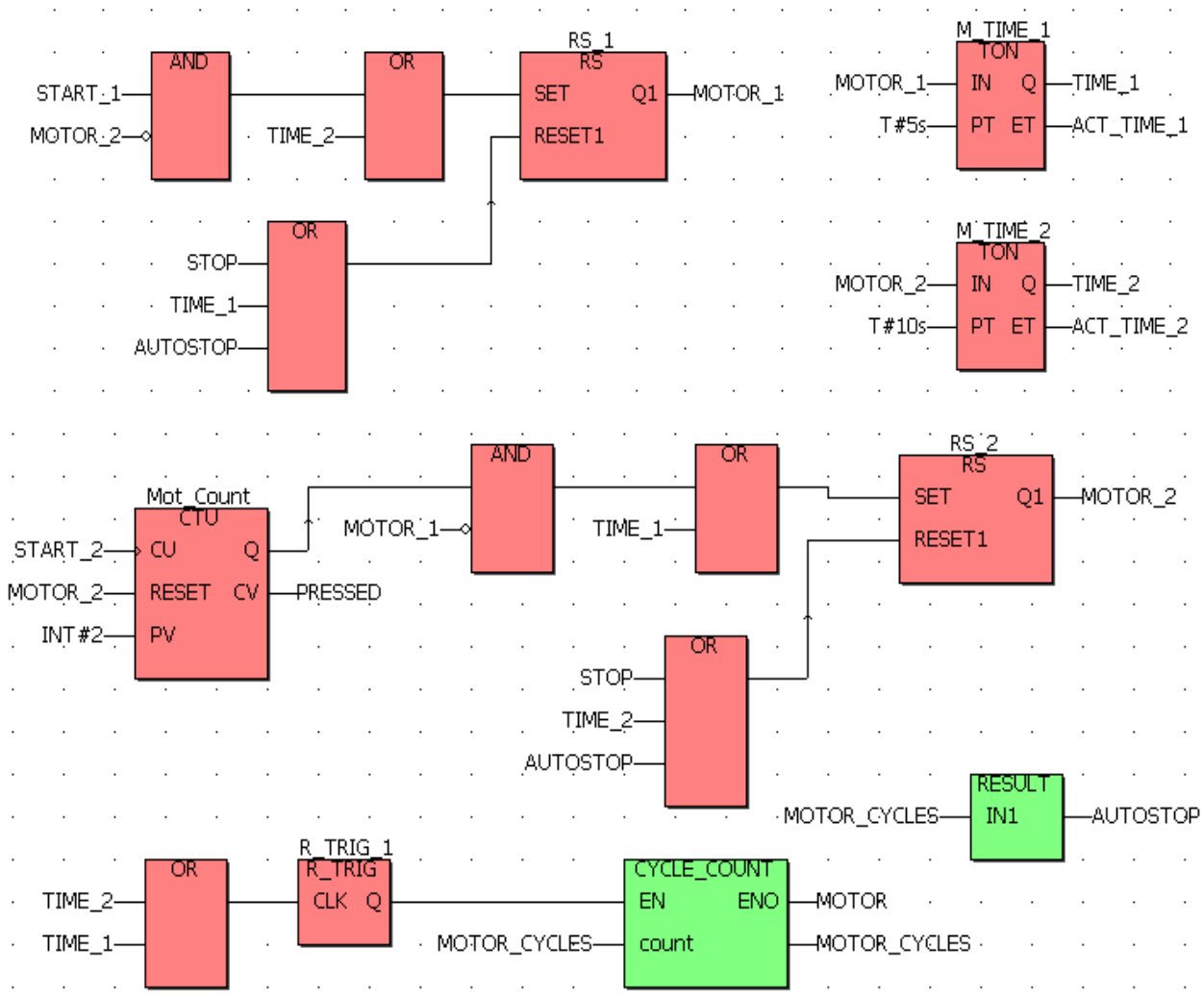
(* ST еквівалент: *)

```
If auto_mode OR alarm Then
Return;
End_if;
bo67 := (bi10 ADD bi23) OR x_cmd;
```

1.2.3.2. Приклад програмування FBD - мовою

Як приклад програмування FBD – мовою розробимо програму керування роботою двох двигунів згідно з алгоритмом наведеним у параграфі 1.2.2.2.

Проектний код FBD-мовою має вигляд:



Тут:

START_1 і START_2 - кнопки початкового пуску двигунів;

MOTOR_1 і MOTOR_2 – перший і другий двигуни;

M_TIME_1 і M_TIME_2 – екземпляри функціонального блока таймера TON, які відлічують час роботи двигунів;

Mot_Count – екземпляр функціонального блока лічильника CTU, який підраховує кількість натисків на кнопку START_2;

RS_1 і RS_2 – екземпляри функціонального блока детектора імпульсу переднього фронту R_TRIG.

CYCLE_COUNT(*CYCLE_COUNT:=count+1;)*–функція користувача, яка підраховує активізації двигунів;

RESULT*(RESULT:=IN1=10;)* – функція користувача, яка порівнює поточну кількість вмикань двигунів з уставкою 10;

STOP – кнопка непередбаченої зупинки двигунів.