Національний технічний університет
«Харківський Політехнічній Інститут»

Факультет Інтегрованих технологій і хімічної техніки

Програмне забезпечення *мікропроцесорних систем*

Лекция 9

Основы применения языка SFC в CoDeSys. Реализация многозадачности

Лысаченко И.Г.

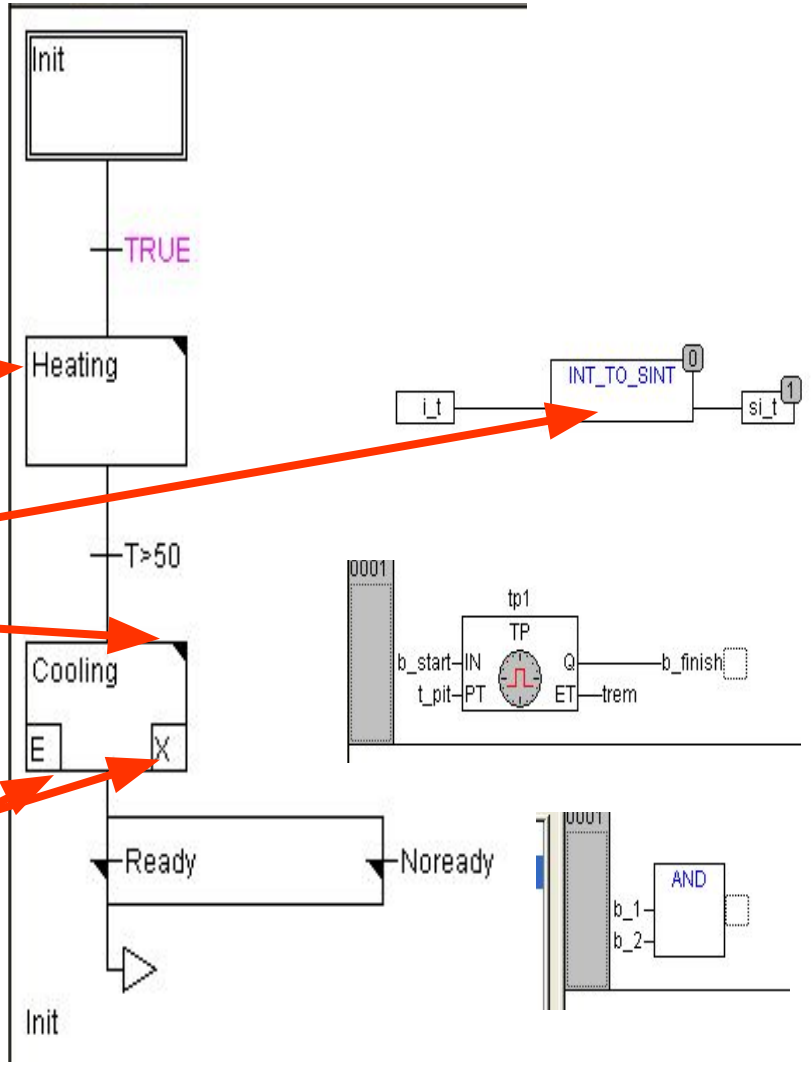
2012

Вопросы лекции

- Элементы языка последовательных функциональных схем (SFC)
  - упрощенный SFC
- Многозадачность в проектах

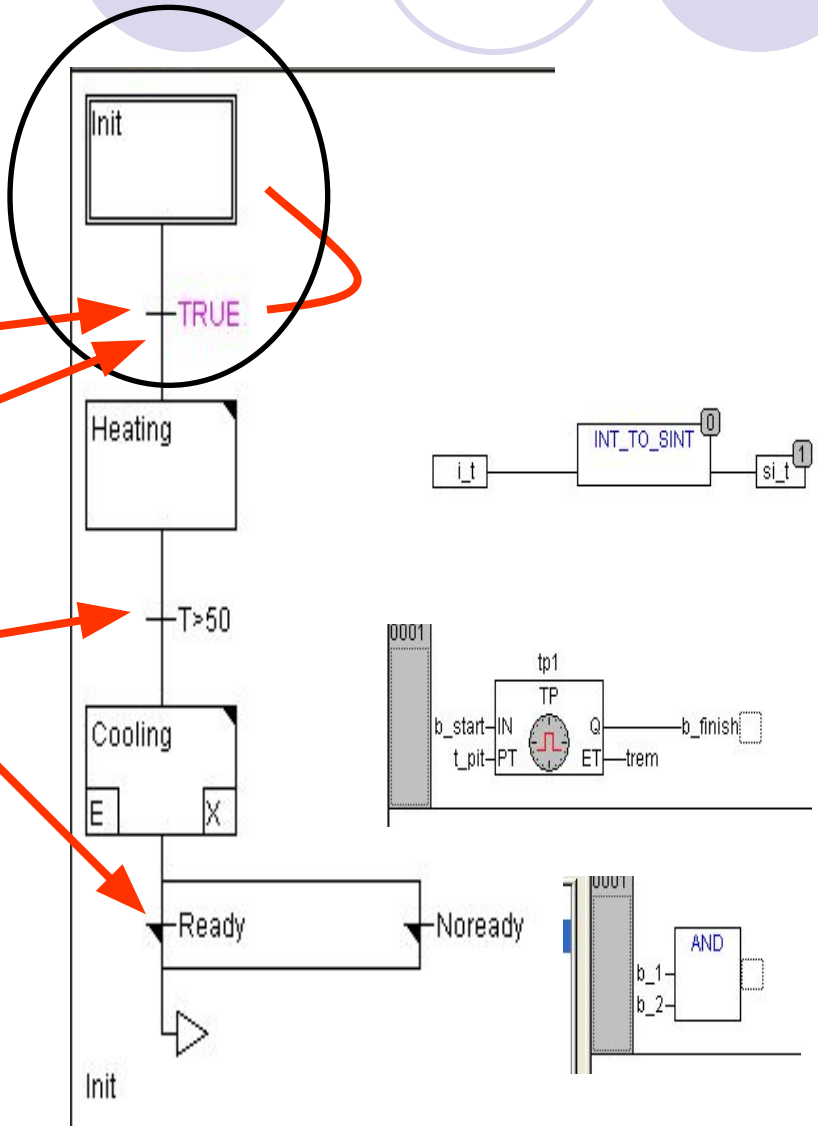
Элементы языка SFC…

- любая схема включает
  - шаги
    - в прямоугольниках
      - название
      - коментарий
    - действие - в отдельном окне
      - обозначено черным треугольником справа-вверху шага
      - действие может быть входным и выходным
    - могут быть пустыми – ожидание перехода



Элементы языка SFC…

- любая схема включает
  - условия перехода
    - рядом с чертой, ниже шага – относится к шагу сверху
    - условие перехода (в схеме только на языке LD)
      - константа
      - логическое выражение
      - логическая переменная
      - прямой адрес
    - сложные условия в отдельных окнах
      - IL, ST, LD, FBD



Элементы языка SFC …

- переход выполняется, если
  - переход разрешен (шаг активный)
  - условие имеет значение TRUE

  Наличие сложного условия определяется по идентификатору – закрашенному углу перехода

Инструменты языка SFC…

- *'Вставка' 'Шаг-переход (сверху)'*
- *'Вставка' 'Шаг-переход (снизу)'*

  Шаг можно удалить, только выделив его вместе с предшествующим или последующим переходом
  - для этого сделайте выделение вокруг шага вместе с переходом и дайте команду 'Правка' 'Очистить', либо нажмите клавишу <Del>

Инструменты языка SFC…

- *'Вставка' 'Альтернативная ветвь (справа)'*
- *'Вставка' 'Альтернативная ветвь (слева)'*
- *'Вставка' 'Параллельная ветвь (справа)'*
- *'Вставка' 'Параллельная ветвь (слева)'*
- *'Вставка' 'Безусловный переход'*
  - вставляет произвольный безусловный переход (jump) в конец ветви, к которой принадлежит выделенный блок

Инструменты языка SFC…

- *'Вставка' 'Переход-Безусловный переход'*
  - вставляет переход вместе со следующим после него произвольным переходом (jump) в конец выбранной параллельной ветви
- *'Вставка' 'Добавить входное действие'*
  - шаг с входным действием имеет букву "E" в левом нижнем углу
- *'Вставка' 'Добавить выходное действие'*
  - шаг с входным действием имеет букву "X" в правом нижнем углу

Реализация многозадачности…

- в любом проекте всегда существует, как минимум, одна задача
  - по умолчанию это *циклическое задание,* которое вызывается в каждом рабочем цикле ПЛК
  - минимальное время привязки задач 10 мс
- каждая задача обладает определенным приоритетом
  - приоритет определяется числом от 0 до 32
  - любая задача, даже более приоритетная, дает доработать текущую задачу до конца одного рабочего цикла (*невытесняющая многозадачность*)

Работа с конфигуратором задач

- Создать циклическую задачу
- Создать задачу, выполняемую по событию
- Создать свободно-выполняемую задачу
- Создать программы – счетчики числа запусков задач
- Проследить за выполнением свободно-выполняемой задачи, изменяя параметры других задач

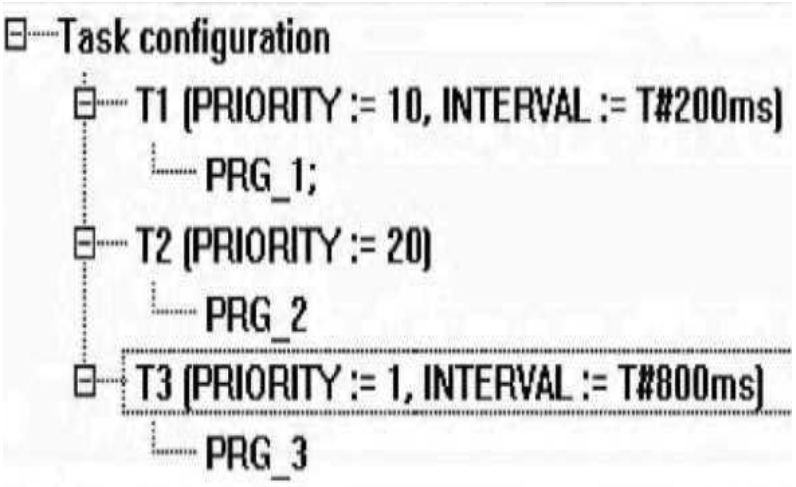
Конфигурирование задач

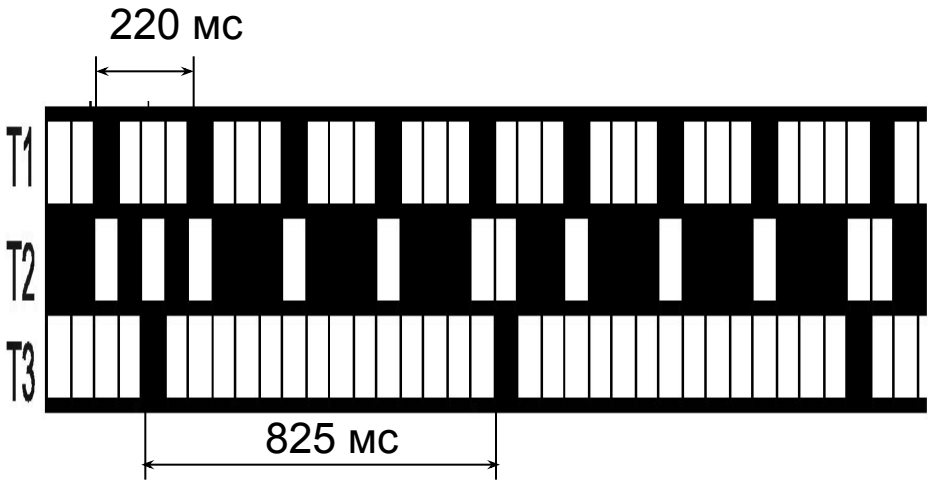
- Задачи выполняются по событию или циклически
- Имеют приоритет
- Вызывают программы
- Есть свободно-выполняемые задачи (аналог idle)

Задачи в ПЛК…

- Каждая задача должна иметь собственный уникальный *идентификатор*
  - Циклическая
    - выполняется через заданные интервалы времени
  - Разовая (single)
    - выполнение *разовой задачи* запускается по фронту логической триггерной переменной
      - каждая задача может включать вызов одной или нескольких программ
      - если программа имеет входные параметры **(Var_input)**, то они задаются в описании задачи
      - все программы одной задачи выполняются в одном рабочем цикле ПЛК

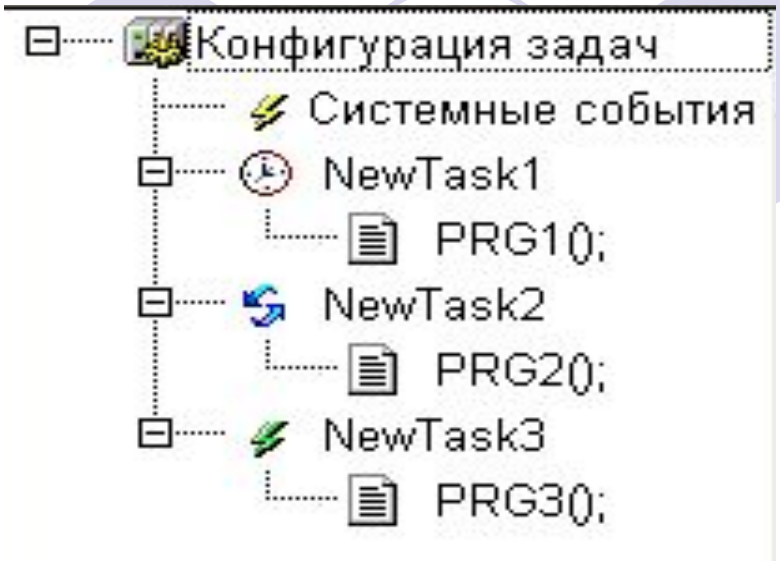
Реализация многозадачности…





- показаны три задачи
  - для ПЛК с временем рабочего цикла около 55 мс (такой цикл дает системный таймер Windows)
- диаграмма выполнения задач

Пример



- VAR_GLOBAL
  w1, w2, w3: WORD;
  in1 AT %IX0.0.0:BOOL;
  END_VAR
- PRG_1 – циклическая, T#1s0ms
  - w1:=w1+1;
- PRG_2 – свободная
  - w2:=w2+1;
- PRG_3 – по событию, IN1
  - w3:=w3+1;