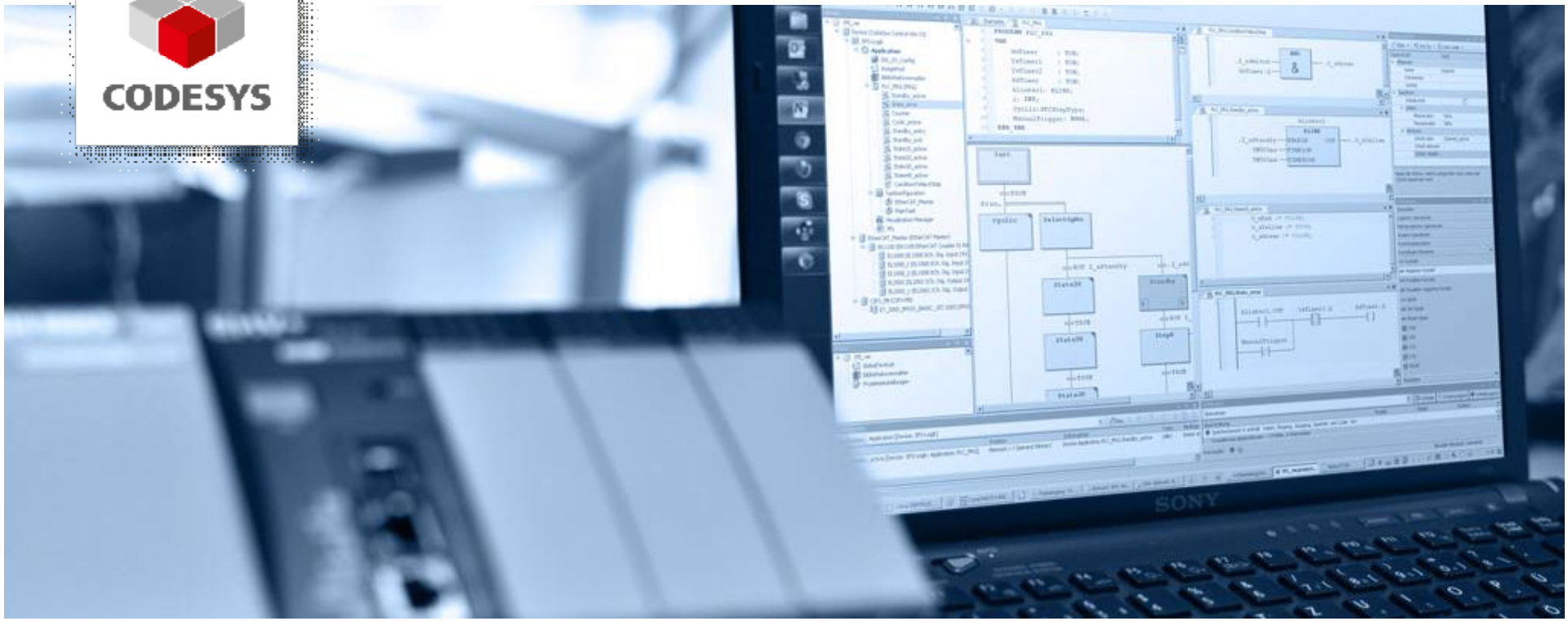


IEC 61131-3 Edition 3

CODESYS Users‘ Conference 2013

1 Что такое МЭК 61131-3

2 История стандарта

3 Новинки

4 Планы на будущее

Что это за стандарт?

- Стандарт для программирования контроллеров и систем автоматизации
- Строгое разделение объявлений данных и кода программы
- Разнообразные подходы к программированию – графические и текстовые языки программирования
- Простые и сложные типы данных
- Различные виды модулей – функции, функциональные блоки и программы

История успеха

- Революционные новшества
- Символьное программирование
- Неограниченое количество экземпляров функциональных блоков
- Язык структурного программирования (ST)
- Практически все системы программирования ПЛК соответствуют стандарту
- Распространение в новых областях
- Автоматизация зданий
- Мобильные системы (строительные машины, краны, подъемники и т.д.)

История

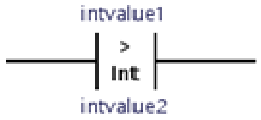
- Первая редакция 1994
- Вторая редакция 2003 : исправление ошибок, небольшие дополнения
- С 2008 года разрабатывается третья редакция
- Международные встречи для разработки стандарта
- Фирмы-участники: Siemens, Panasonic, Pilz, Elau, Schneider, Bosch Rexroth, KW, LogiCALs, Mitsubishi, Rockwell, 3S и т.д.

„Сокращения“ в стандарте

- Восьмеричная система исчисления (8#52)
- Оператор TRUNC заменен на REAL_TRUNC_DINT
- Использование МЭК адресов (%MW123) в коде
  Теперь только в разделе объявлений
- Язык IL

„Дополнения“ в стандарте

- Использование констант при инициализации переменных (i : INT := CONST_VALUE;)
- Использование констант при объявлении массива (arr : ARRAY [0..COUNT-1];)
- Переменные в качестве меток case
- Правила неявного преобразования типов (Например: dint_var := int_var;)
- Новые элементы LD:



- Оператор continue для циклов

Новые типы данных

- CHAR, WCHAR
- LTIME, LDATE, LDATE_AND_TIME, LTIME_OF_DAY (количество наносекунд, начиная с 1.1.1970).

Доступ к частям битовой строки

- Чтение и запись
- <Byte_var>.%X1
- <Word_var>.%B1
- <DWord_var>.%W1
- <LWord_var>.%D1

Структуры с явным расположением элементов

```
TYPE
  Com1_data: STRUCT
   head       AT %B0:       INT;
   length     AT %B2:       USINT := 26;
   flag1      AT %X3.0:     BOOL;
   end        AT %B25:      BYTE;
  END_STRUCT;
   Com2_data: STRUCT OVERLAP
   head       AT %B0:       INT;
   length     AT %B2:       USINT;
   flag2      AT %X3.3:     BOOL;
   data1      AT %B5:       BYTE;
   data2      AT %B5:       REAL;
   end        AT %B19:      BYTE;
  END_STRUCT;
END_TYPE
```

Массивы с переменным размером

- В качестве входов функции или VAR_IN_OUT параметров функционального блока

```
FUNCTION SUM: INT;
VAR_IN_OUT A: ARRAY [*] OF INT; END_VAR;
VAR i, sum2 : DINT; END_VAR;

sum2:= 0;
FOR i:= LOWER_BOUND(A,1) TO UPPER_BOUND(A,1)
  sum2:= sum2 + A[i];
END_FOR;
SUM:= sum2;
END_FUNCTION
```

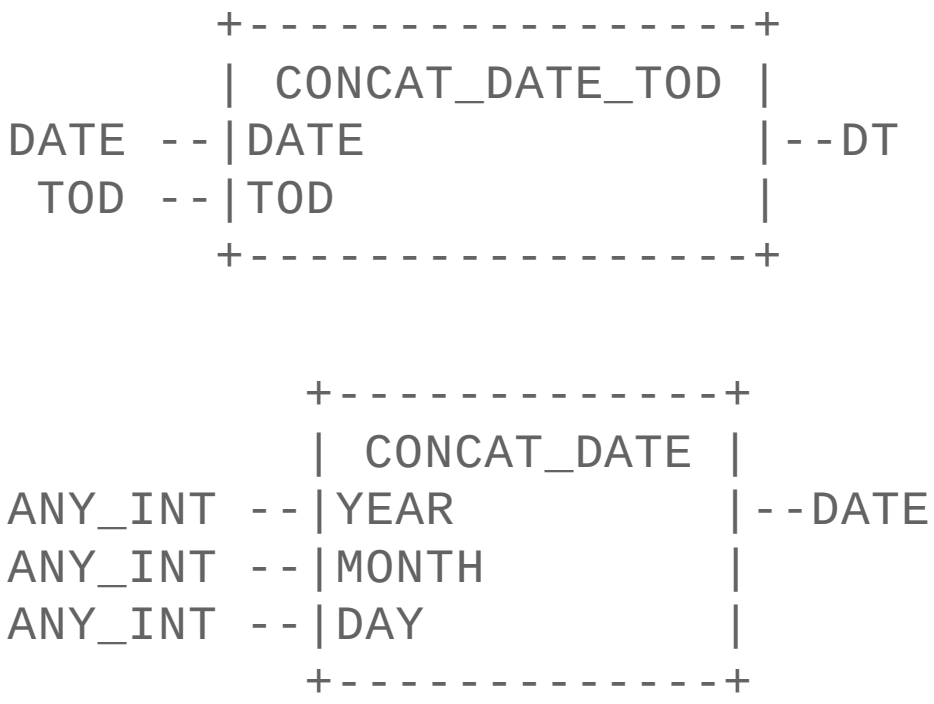
Ссылки (reference)

- myRefInt: REF_TO INT := REF(myInt);
- Ссылки типизированы
- Арифметика не поддерживается
- Оператор для получения ссылки на переменную

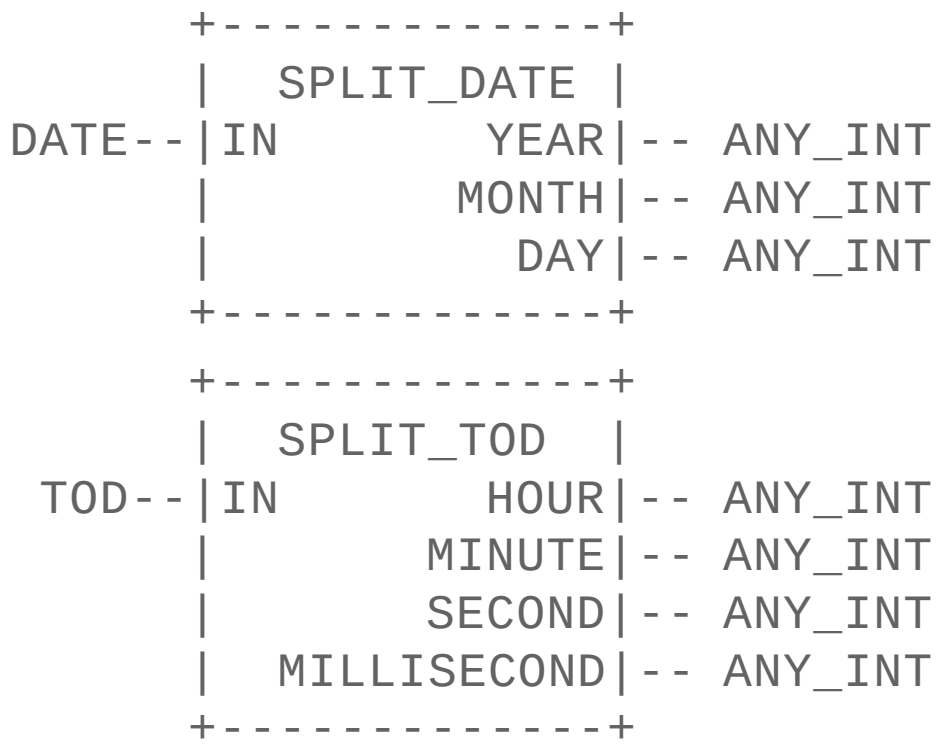
Новые функции преобразования типов

- A := TO_REAL(B); // ранее: INT_TO_REAL
- A := LREAL_TRUNC_INT(B); // ранее : TRUNC
- A := TRUNC_INT(B); // ранее : TRUNC
- A := BCD_TO_INT(B); // ранее : WORD_BCD_TO_INT
- A := TO_BCD_WORD(B);// ранее : INT_TO_BCD_WORD

Новые функции для типов DATE и TIME (1)



Новые функции для типов DATE и TIME (2)



Объектно-ориентированное программирование

- Объектно-ориентированный функциональный блок или класс?
- Преимущества использования ФБ:
  - Возможно постепенное введение новых возможностей
  - Класс включает понятие функционального блока
  - Легкость перехода
  - Использование старых проектов
  - Для библиотек: внешне обычный функциональный блок, но разработан с помощью приемов ООП

Объектно-ориентированное программирование

- Преимущества использования классов:
  - Хорошо известны специалистам по программированию

- => Компромисс: использование как классов, так и функциональных блоков

Объектно-ориентированное программирование. Возможности

- Ключевые слова из Java:
  (METHOD, EXTENDS, IMPLEMENTS, THIS, SUPER и т.д.).
- Концепция также взята из Java:
  - Нет множественного наследования
  - Любое количество реализуемых интерфейсов

Объектно-ориентированное программирование. Возможности

- Методы могут быть
  - Private, Protected, Public, Internal (Namespaces!)
- Наследование (Extends)
- Переопределение методов (Override)
- Абстракция интерфейсов
- Реализация интерфейсов (Implements)
- Абстрактные классы

Объектно-ориентированное программирование. Планы на будущее

- Конструктур/диструктор
- Свойства (как в C#)
- Статические переменные в классах
- Статические методы в классах
- Функции с переменным числом входов (как в C#: PARAMS OF),
- Перегрузка методов (Overloading)

Третья редакция принята в январе 2013 года.

Работа над стандартом продолжается...

Inspiring Automation Solutions

Спасибо за внимание