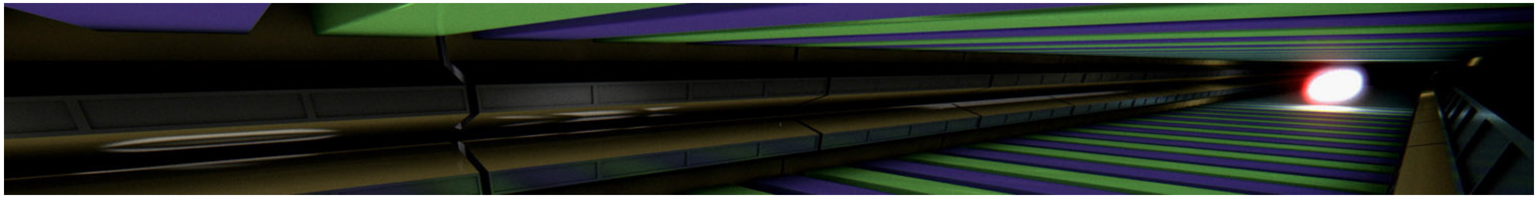


# Integration of BECKHOFF control systems

1st Meeting of European XFEL Accelerator Consortium

17 April 2012,

Hamburg

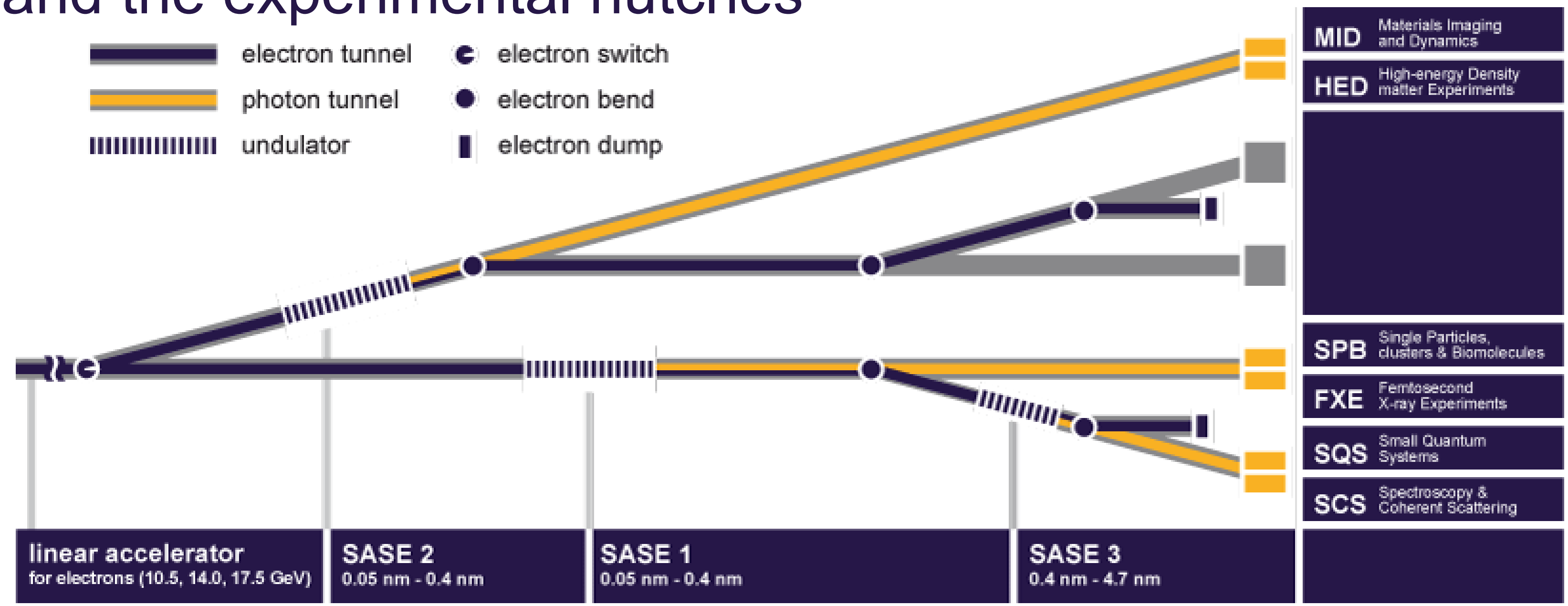
N. Coppola, WP76

# Outline

- Definition of control system
  - Beam-line + Diagnostic control
  - Experiments' control
- Why BECKHOFF
- BECKHOFF PLC development tools
  - Programmatic generation of configuration
  - TwinCAT 2.11 (in the future TwinCAT3)
- How our implementation looks like
- Current work
- Where will we use it
- Outlook and Conclusion

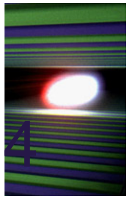
# Topology of PBS control system

In this talk we will deal with the Yellow part of the tunnels and the experimental hutches


electron tunnel
photon tunnel
undulator
electron switch
electron bend
electron dump
MID Materials Imaging and Dynamics
HED High-energy Density matter Experiments
SPB Single Particles, clusters & Biomolecules
FXE Femtosecond X-ray Experiments
SQS Small Quantum Systems
SCS Spectroscopy & Coherent Scattering
linear accelerator for electrons (10.5, 14.0, 17.5 GeV)
SASE 2 0.05 nm - 0.4 nm
SASE 1 0.05 nm - 0.4 nm
SASE 3 0.4 nm - 4.7 nm

Note: the undulator control is done by WP71, we need to monitor undulator quantities via a software interface to their global control system

# Why BECKHOFF



We need something with intelligence that must run also when no internet is present, needed near-to 100% reliability:

- SPS/PLC firmware on Real-Time CPU(s);
- PLC under Beckhoff runs also with operating system blue screen;
- Beckhoff is COTS;
- with EtherCAT bus (with real-time-Ethernet protocol);
- Beckhoff EL-system allows for redundant loops, stars and trees topologies;

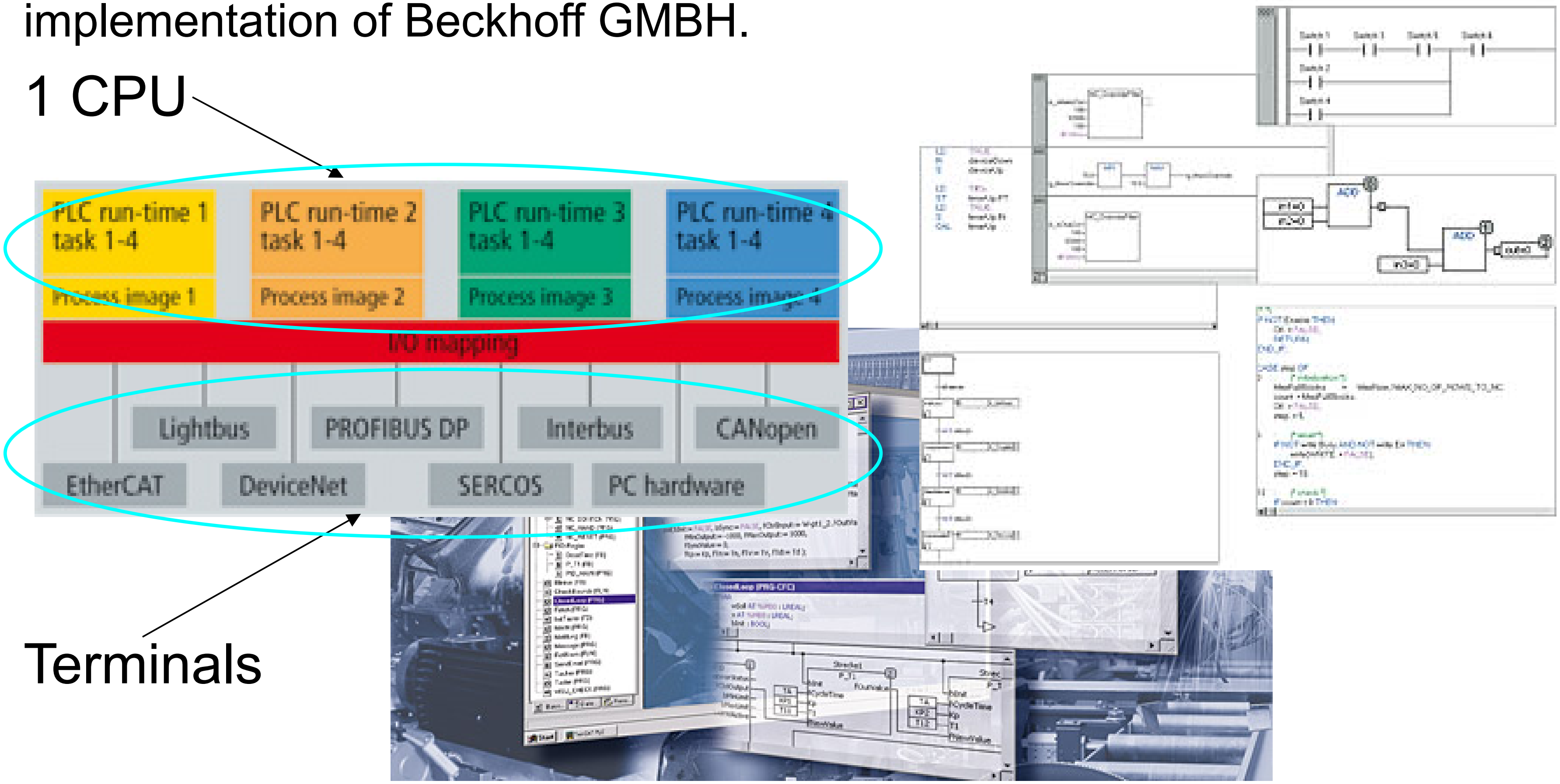
PLC programs on Beckhoff CPUs

- Windows CE (also Windows XP and/or Wondows 7 are possible)
- contains: FSM, main business logic;
- allows H/W synchronization (example: axes coupling);
- can be connected to software Device running in linux-box (via standard Tcp/Ip);
- allows some local logging (remanent and persistent variables);
- administrational tasks (also *high* level ones).

# What is a SPS/PLC

Speicherprogrammierbare System (SPS) and IEC 61131-3 software Programmable Logic Controller (PLC)

implementation of Beckhoff GMBH.



1 CPU

Terminals

# Definition of photon beam line control system

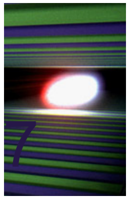
Beam line control concerns controlling hardware in the photon beam lines and experiment hutches:

- vacuum: pumps, gauges and valves
- offset mirror actuators, shutters…
- motors for screen insertion,
- K-monochromator stages…
- micro and nano positioning devices...
- sample injectors...

The aim is for

- a simple, durable, sustainable and easily maintainable solution;
- a simple but scalable system with as small as possible types of devices, to help easy operations, interchangeability, spares etc;
- a system that is easily interfaced to other control systems to allow “external” control and interactions of the accelerator operators.

# Photon beam line control system

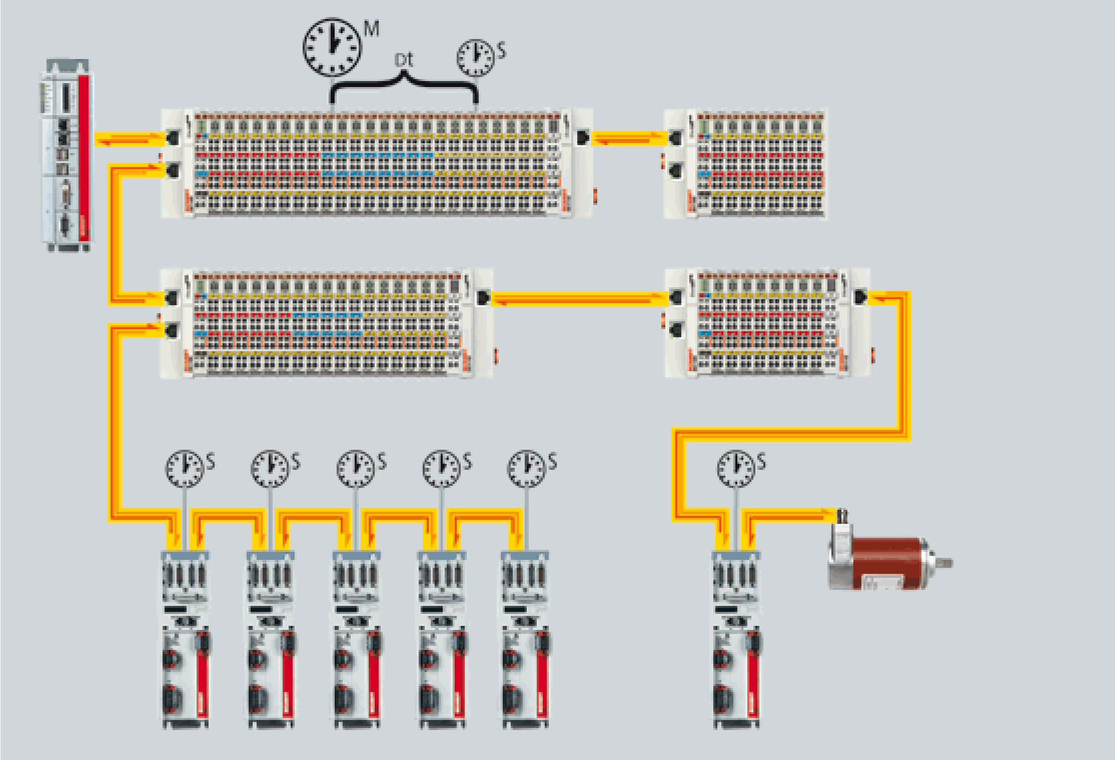


Currently our implementation model assumes that all components can be controlled via Beckhoff EtherCAT bus. Why?

- compatible with undulator (WP71) implementation decision = mutual benefits;
- all other control systems (ALBA, DIAMOND, ESRF, LCLS, FLASH, PETRA3…) reviewed use similar (WAGO, B&R, Siemens…) rail/terminal/PLC systems;
- Beckhoff is COTS and should reduce FTE and hardware costs

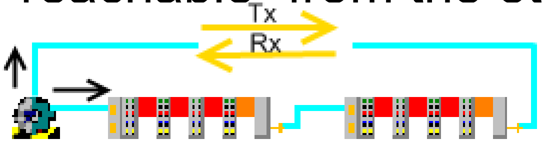
We will no doubt find that a few % or implementations cannot be supported = require special solutions

# Freedom of EtherCAT bus topology



Possible EtherCAT topology(ies):
Closed loops are also possible

Nice features:
terminals' local clocks;
auto failure detections;
capability to run terminals up to the point with failure;
in case a loop (redundant topology) the rest of the terminals is "reachable" from the other direction



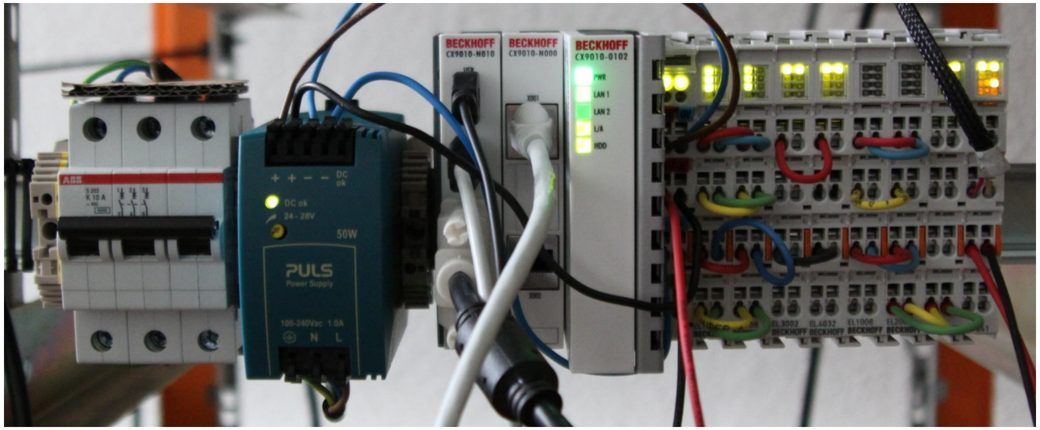
Possibility to use IP67 Field-boxes where needed.

using "intelligent" couplers (like EK1101) even if one branch has a problem (or needs maintenance work) the rest can run w/o interruptions

# Our PLC development is similar to ALBA's

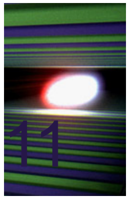
- PLC Code generation *a là* Alba:
  - Written bash scripts to generate code starting from emulated DB reports, as csv input-files
- Structures *a là* Alba implemented, tailored and widened to allow for motor control
- PLCs on Beckhoff, what we have (not complete list):
  - administrative PLC to manage other PLCs, CRC check and versioning;
  - simple but generic state machine;
  - example firmware to configure and control Quantum pulse generator;
  - firmware to control: Varian Ion pumps, MKS Vacuum gauge Pfeiffer MaxiGauge controller(s) via serial interface;
  - firmware to control stepper axes;
  - firmware to control piezo motors (SmarAct).

# Current developments

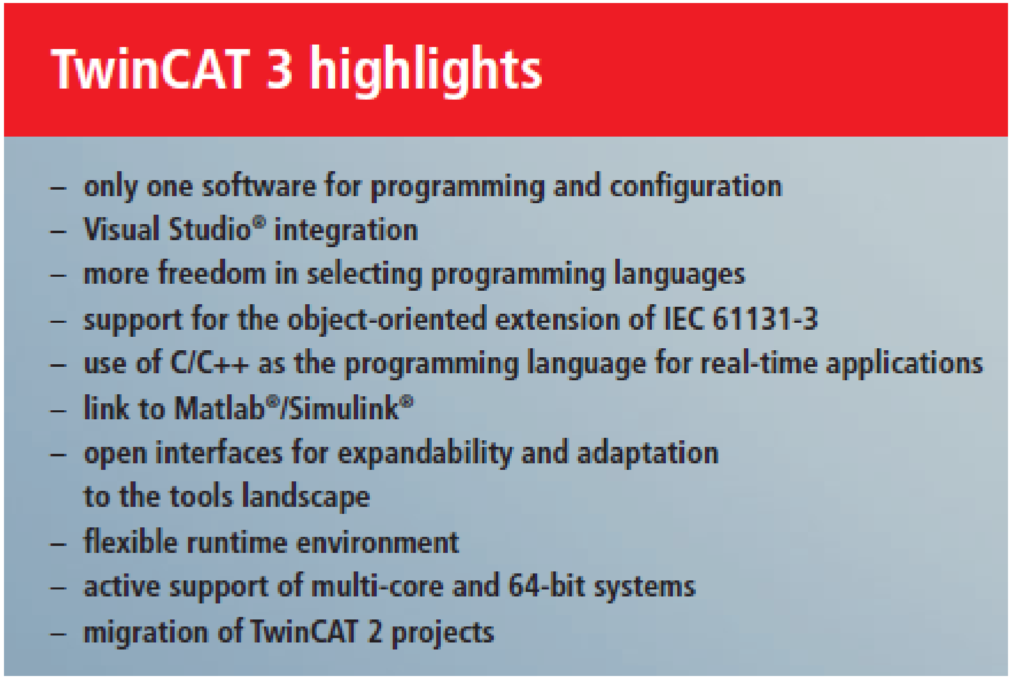


- Development of control of K-mono prototype
  - synchronous movement of motors;
- Photoelectron Spectrometer;
- Mockup stage of pop-in cameras;
- Offset mirror movement
  - synchronous movement of motors;
- Pump in a cart pumping movable stand to start pump down operations;
- Control of Vacuum sections;
- SPB (and SQS): planning of experiment hutch and racks' room;
- 2-dim imaging detector test setup (for example pnCCD);
- Interface to eXFEL Timing board for Train Nr tagging;
- Interface to MPS

# PLC configurations and future developments



- Programmatic generation of low level Terminals/Rails configurations et al.:
  - started developing generation of configuration files from simple demo programs (source: vb, c++, c#, & .net written by Beckhoff experts), (software project managers have promised direct support in case of problems)
  - We have been BETA-testers of version of TwinCAT 3 since October 2011
    we plan in a *short time* to switch from TC2.11 to TC3
    - March 19th 2012 TwinCAT3 officially released;
    - promised @end Q2/2012 serial interface support;
    - possibility to insert or use C++/Simulink code directly;
    - unified administration of firmware coding & configurations;
    - capability to feed xml files (ie reports from KDS) to create configuration files.



TwinCAT 3 highlights

- only one software for programming and configuration
- Visual Studio® integration
- more freedom in selecting programming languages
- support for the object-oriented extension of IEC 61131-3
- use of C/C++ as the programming language for real-time applications
- link to Matlab®/Simulink®
- open interfaces for expandability and adaptation to the tools landscape
- flexible runtime environment
- active support of multi-core and 64-bit systems
- migration of TwinCAT 2 projects

# Cable and Equipment layout tool

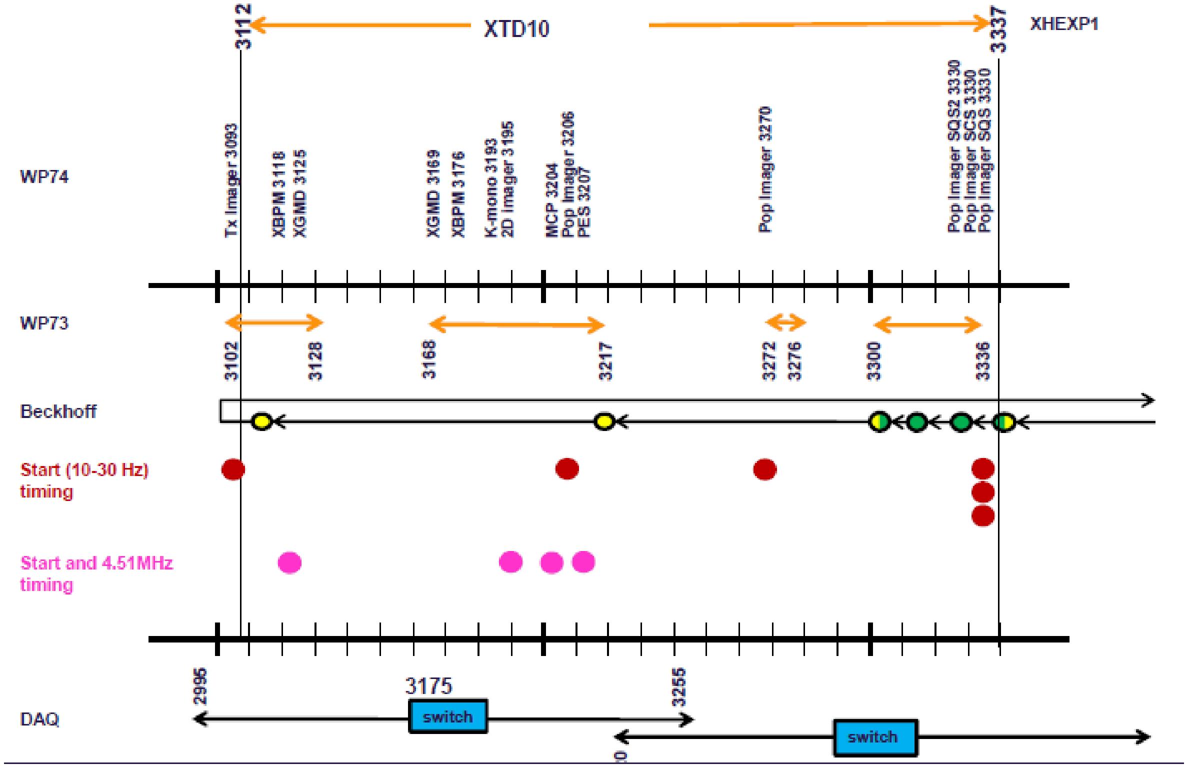
- Investigating KDS (Command) of FNT-GmbH
  - intend to use it
- Updates (for PLC implementation):
  - Implemented place holders *for a rail in a crate*
  - Implemented relevant EtherCAT coupler
  - There are WAGO terminals (but we do not use them) therefore inserting Beckhoff equivalent  must be easy
- Main problem: missing manpower
  - position advertisement closed;
  - selection process starting in a short time.

# Conclusions

- We have selected BECKHOFF EL-rail system and TwinCAT given:
  - experts' support (in Lübeck and in Verl)
  - guaranteed long term availability ∝20 years
- Cable and equipment layout tool
- PLC Firmware developments going on:
  - increasing scope of applications;
  - TwinCAT 3 (programmatic management of configurations)
- Longer term Aim:
  - Complete integration Beckhoff's world into the homogeneous software framework.

# Backup Slides

# Topology and interconnections of control system



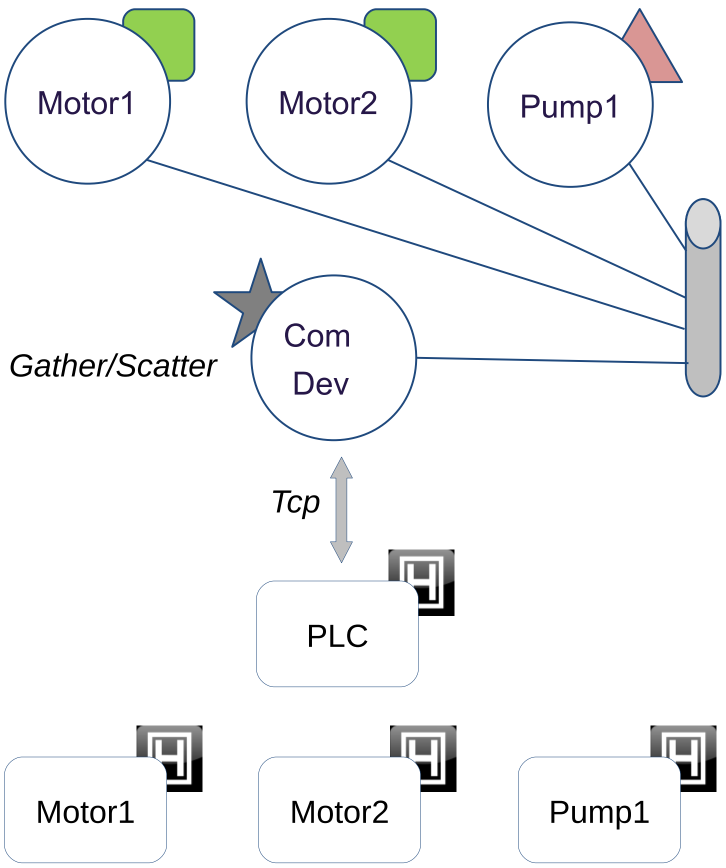
SASE3 – XTD10 equipment and network planning as it is currently understood.

Equipment locations defined by WP73 and WP74 component list

There are currently 4 network types present

- Beckhoff beam line control
- Train start timing (10-30Hz)
- DAQ timing (start and 4.51MHz bunch clock)
- DAQ readout IP network

# Integrating Beckhoff into the control system



*“Motor1”, “signalPlcWrite” --- “ComDev”, “slotPlcWrite”*

*“ComDev”, “signalPlcRead” --- “Motor1”, “slotPlcRead”*

Beckhoff PLCs can run several hardware “pieces”

Communication is limited to a single entry point (PLC server)

Modularity of different PLC setups should be reflected and easily implemented on C++ side

Burkhard Heisen

# Beckhoff cabling and topology in the tunnels

We aim at geographical segmenting the system in subsystems, one per SASE tunnel and one per experimental hutch.

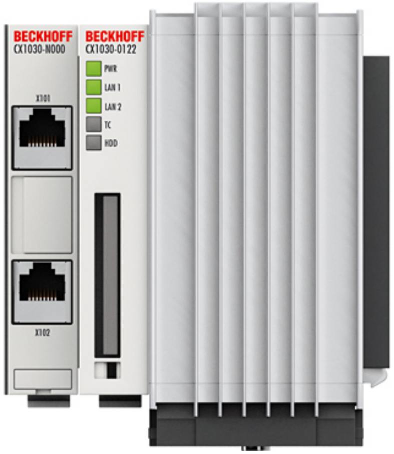
We plan 3-loops per tunnel (most probably >3-loops per hutch), i.e. splitting

- movements (motors and encoders)
- vacuum (pumps and gauges) *and the rest*
- special cases (testing new or possibly faulty systems)

these will be interconnected according to needs (either via TCP/IP or via EtherCAT bridges)

# Beckhoff control system components and locations

We plan to place the controlling CPU part of the system in the *balconies*.

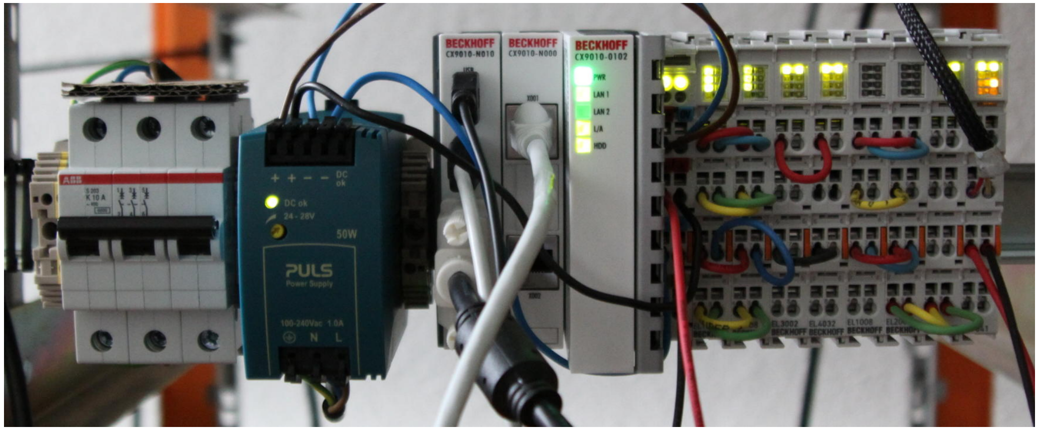


The EL rails and EL terminals in the tunnels and in the experimental hutches, as nearby as possible to the device they are connected to.

We use CAT5 (or better) copper cables, where distances are below ~90 m, or multi mode fibers covering up to ~2 km.

# PLC development

- *Few* CPUs (Windows XP and Windows CE) to develop PLC and widen tests
  - DAC && ADC of *simple* quantities (V; I; Temp; Pressure...)
  - Motor steering
  - Vacuum pump(s) and gauge(s)
  - Terminals like:
    bridges,
    PT100, counter,
    PROFIBUS, …
  - Direct interface to Clock & Control timing system for synchronization purposes



# Implementation of control system

## Multi layered system allows to swap pieces w/o too much effort

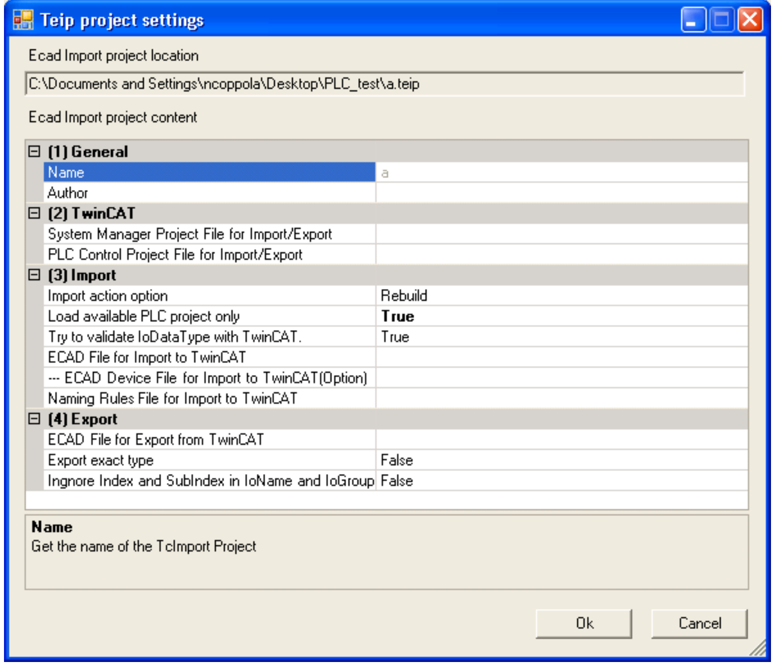
- GUIs (top, K.Weger in charge since 1/11/2011): C++ Qt/PyQt Tango/Mango Papaja/kiwi ….
- exfelSuite (aka ☺M*A*S*H): DeviceServer(s) and CompositeServer (C++) contains (middle layer): FSM, some business logic, inter-connectivity, messaging, logging, archive (once DB FTE is hired), configuration(s).
Same framework as for scientific computing and data acquisition.
- SPS Firmware on Real-Time CPU with EtherCAT bus:
PLC programs on Beckhoff (bottom layer, direct connection to devices to be controlled) contains: FSM, main business logic, connected to CompositeServer, some local logging (remanent and persistent variables), versioning and CRC of Firmware and configuration, must run also when no internet is present, needed 100% (or almost 100%) reliability. Beckhoff system allows for redundant loops, stars and trees topologies. Plan is to put CPUs on private network.
- Micro controller on final devices (pumps, RGA, vacuum gauges...) physically connected to Beckhoff bus

European XFEL

# PLC configuration developments

- Started contact with Automation INTERFACE experts at Beckhoff to decouple from proprietary configuration tool (aim: move to non-operator scripts)
- To this aim:
    - we had received test version of ECAD utility
      *“Software package for importing XML files from the ECAD systems.”*

      *the utility helps to generate PLC firmware and configuration files using xml vocabularies*



**Teip project settings**

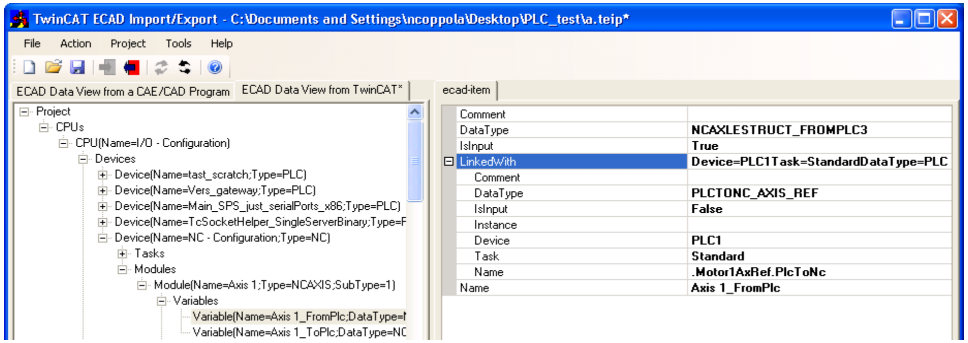
Ecad Import project location

C:\Documents and Settings\ncoppola\Desktop\PLC_test\a.teip

Ecad Import project content

| | |
|---|---|
| ⊟ **(1) General** | |
| Name | a |
| Author | |
| ⊟ **(2) TwinCAT** | |
| System Manager Project File for Import/Export | |
| PLC Control Project File for Import/Export | |
| ⊟ **(3) Import** | |
| Import action option | Rebuild |
| Load available PLC project only | **True** |
| Try to validate IoDataType with TwinCAT. | True |
| ECAD File for Import to TwinCAT | |
| --- ECAD Device File for Import to TwinCAT(Option) | |
| Naming Rules File for Import to TwinCAT | |
| ⊟ **(4) Export** | |
| ECAD File for Export from TwinCAT | |
| Export exact type | False |
| Ingnore Index and SubIndex in IoName and IoGroup | False |

**Name**
Get the name of the TcImport Project



**TwinCAT ECAD Import/Export - C:\Documents and Settings\ncoppola\Desktop\PLC_test\a.teip***

File
Action
Project
Tools
Help
ECAD Data View from a CAE/CAD Program
ECAD Data View from TwinCAT*
ecad-item

```
⊟ Project
   ⊟ CPUs
      ⊟ CPU(Name=I/O - Configuration)
         ⊟ Devices
            ⊞ Device(Name=tast_scratch;Type=PLC)
            ⊞ Device(Name=Vers_gateway;Type=PLC)
            ⊞ Device(Name=Main_SPS_just_serialPorts_x86;Type=PLC)
            ⊞ Device(Name=TcSocketHelper_SingleServerBinary;Type=F
            ⊟ Device(Name=NC - Configuration;Type=NC)
               ⊞ Tasks
               ⊟ Modules
                  ⊟ Module(Name=Axis 1;Type=NCAXIS;SubType=1)
                     ⊟ Variables
                        Variable(Name=Axis 1_FromPlc;DataType=I
                        Variable(Name=Axis 1_ToPlc;DataType=NC
```

| | |
|---|---|
| Comment | |
| DataType | **NCAXLESTRUCT_FROMPLC3** |
| IsInput | **True** |
| ⊟ LinkedWith | **Device=PLC1Task=StandardDataType=PLC** |
| Comment | |
| DataType | **PLCTONC_AXIS_REF** |
| IsInput | **False** |
| Instance | |
| Device | **PLC1** |
| Task | **Standard** |
| Name | **.Motor1AxRef.PlcToNc** |
| Name | **Axis 1_FromPlc** |