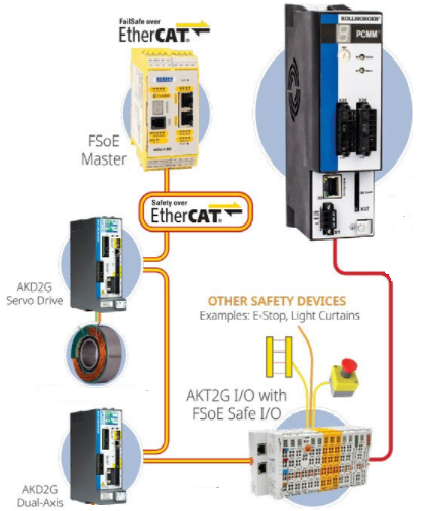
**KAS/SafePLC2/AKD2G - Application Note**

****

**This application note details a safety application with:**

* the safety parameters defined in the SCU Controller
* the PLCOpen motion engine

*Version 1c Oct 5, 2022*

**Table of Contents**

[**Introduction**](#Introduction)

[**Supported Safety Functions**](#Supported_Safety_Functions)

[**FW/SW/IDE Considerations**](#FW_SW_IDE_Considerations)

[**Meeting Safety Ratings**](#Meeting_Safety_Ratings)

[**Minimum Wiring**](#Minimum_Wiring)

[**PCMM Controller Setup**](#PCMM_Controller_Setup)

[**AKT2G Safety IO setup**](#AKT2G_Safety_IO_Setup)

[**SafePLC2 Programming Basics**](#SafePLC2_Programming_Basics)

[**Building a Safety Program**](#Building_a_Safety_Program)

[**SCU - PCMM Communications During System Operation**](#SCU_PCMM_Communications_During_System)

[**Running a SCU Safety program**](#Running_a_SCU_Safety_Program)

[**Safety Parameter Transfer (SafetyPLC2 based safety configuration)**](#Safety_Parameter_Transfer)

[**Monitoring a Running Safety Project - Project Animation**](#Monitoring_a_Running_Project)

[**AKD2G/ Workbench Setup**](#AKT2G_Safety_IO_Setup)

[**Verification**](#Verification)

[**Safety Brake (to be added)**](#Safety_Brake)

[**Safety Feedback**](#Safety_Feedback)

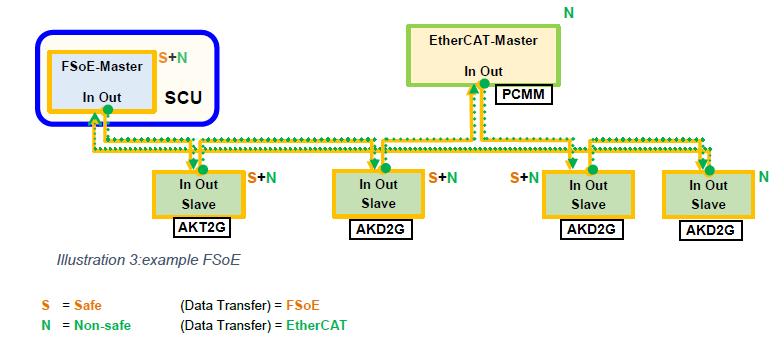
**Introduction**

The KAS machine control system can be configure to meet safety standards needed in some application

This application note details the foundational parts of a KAS Safety System utilizing:

* PCMM Motion/Machine Controller
* KAS Machine/Motion control project including safety **initialization and monitoring**
* Machine control PLC code
* Servo axis definitions
* the EtherCAT Master which serves as a conduit to transfer information between the AKD2G, **AKT2G Safety IO** **and the SCU Safety Controller**
* The AKD2G Servo Drive
* the servo controls
* **Safety Actuation Functionality (FS2 or FS3 functionality)**
* Overall drive setup/configuration for the motor and load
* **The WorkBench environment that displays the safety Configuration**
* AKT2G IO
* **AKT2G-SDI-004, 4 channel fail-safe digital inputs**
* **AKT2G-SD0-004, 4 channel fail-safe digital outputs**
* SCU Safety Controller
* **SCU works with the AKD2G and AKT2G to achieve functional safety  
  The SCU master assembly receives the data from the PCMM, AKD2G, and AKT2G and analyzes the received data.**
* **After data analysis, the SCU can implement safety functionality either directly via the Safety digital outputs or via the outputs of the slave PDO assemblies (through ECAT).**
* **SCU controller contains the values of the AKD2G Safety Parameters and download the parameters when the ECAT network initializes**
* Note: Kollmorgen only supports BBH model SCU-1-EC
* BBH Online Manuals
  + Installation: <http://www.bbh-products.de/files/content/downloads/Handbuecher/en/HB-37450-810-01-10F-EN%20COM%20Installation%20Manual.pdf>
  + Programming: <http://www.bbh-products.de/files/content/downloads/Handbuecher/en/HB-37500-820-10-09F-EN%20SCU%20Programming%20manual.pdf>
* Ethercat Network
* **Connects the drives , remote IO, and other machine devices to the machine controller**
* **Through the FSOE channel, connects the AKD2G drive to the SCU Safety controller**
* **Example Network:**



* FSoE (FailSafe over EtherCAT) provides safety related data transmission in parallel with standard data over the same Ethernet cable, FSoE utilizes the Black Channel for the communication meeting the IEC 61784-3 international standard. From the FS0E safety documentation: The safety data is embedded in the standard process data as a container combined with additional data for its integrity. The safety connection between the FSoE master and an FSoE slave is fully monitored in each safety cycle: The checksum of the safety frames, the connection ID and a watchdog time for each FSoE frame transmission are all checked

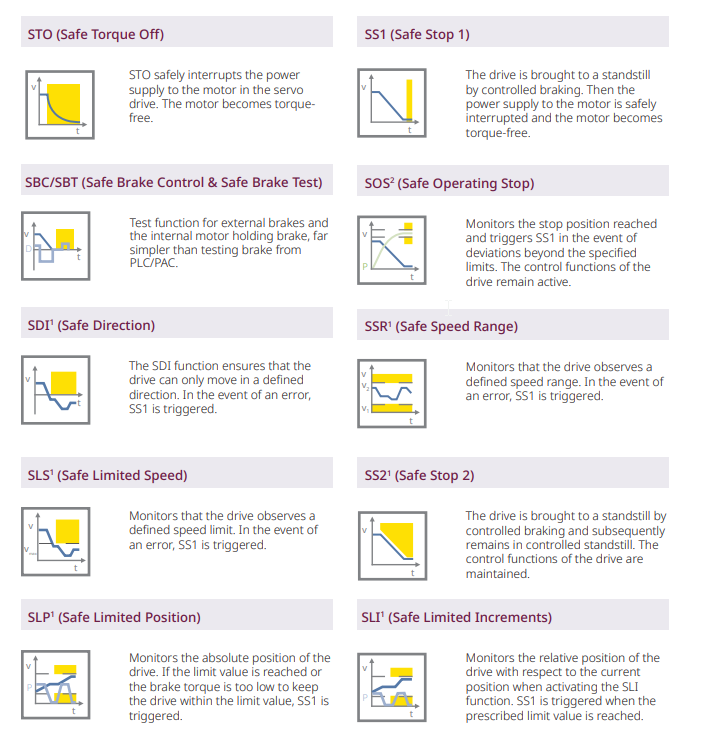
See <https://www.ethercat.org/download/documents/Safety-over-EtherCAT_Introduction.pdf> for more detail.

**KAS Supported Safety Functions**

**Table

Description automatically generated Text

Description automatically generated**

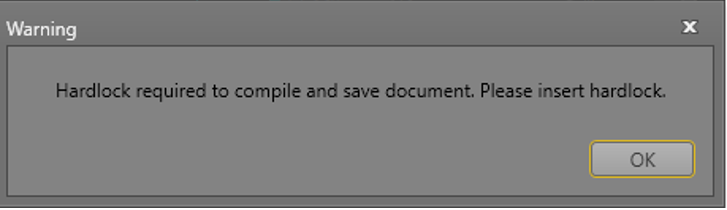
****

FS3

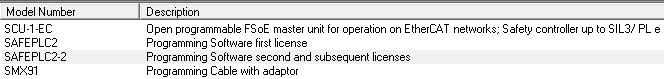
FS2 or FS3

**FW/SW/IDE Considerations**

* Controller – PCMM
* IDE/FW version 3.05 or greater
* Safety Controller SCU
* Programming SW version 1.7.1.8126 or greater
  + - SafePLC2. Installs separately on your computer. As with KVB this SW is accessible from within the KAS project, and the Safety PLC project is saved as part of a .kas project
    - Key dongle (hardlock) required for full operation, If you start the program without the dongle, the following message appears:



* + - Click “OK“ and insert the dongle into the USB port. The dongle is recognized, and all functions of SafePLC2 can be used. If you remove the dongle while working with SafePLC2,functions get lost, and you can neither compile nor save programs. When you reconnect the dongle to the USB port, all functions are available again.
    - Part Numbers:



* AKD2G Servo Drive
  + FW 2.07.01
  + FS2 or FS3. The FS1 model does not support FSOE

**Meeting Safety Ratings**

Safety functionality allows equipment to be partly operational and not completely shut down

* For maintenance
* To let people into the work area

It’s as much about allowing safe access to machinery as it is about protecting people or equipment or the environment. Functional safety influences machine design and will tend to reduce guards and mechanical interlocks but increases safety controls, sensors, and actuators

A safety audit /risk assessment (FSA – Functional Safety Assessment) of the machine is conducted by the machine manufacturer and with the safety level required by the machine the machine manufacturer decides what safety features in the KAS system will need to be implemented.

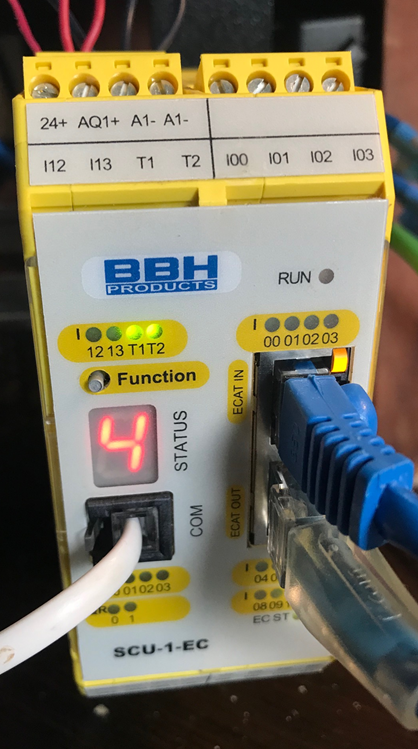
Task of the machine builder (simplified list):

* Identify risks – possibly using formal techniques.
* Determine the SILs or Pls required for each safety function to reduce the risks to acceptable levels.
* Ensuring the safety function performs to the design intent, including under conditions of incorrect operator input and failure modes. This will involve having the design and lifecycle managed by qualified and competent engineers carrying out processes to a recognized functional safety standard.
* Verification that the system meets the assigned SIL or Pl
* Certification by a notified body (if sold in EU).
* Conduct functional safety audits to examine and assess the evidence that the appropriate safety lifecycle management techniques were applied consistently and thoroughly in the relevant lifecycle stages of product.

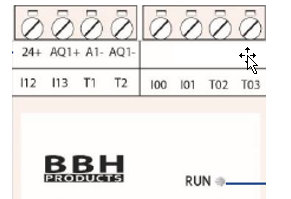
**Minimum Wiring**

Below is the minimal wiring to get a system up and running:

* SCU-1-EC



* + Logic Power: 24 VDC- Wire Screw terminal 4 pins: 1 and 2 24VDC, 3 and 4 common



* + Programming port – RJ11 connector
  + ECAT communication – RJ45 connector - ECAT IN, ECAT OUT ports
* PCMM
  + Logic Power: conn X1 – plug in connector
  + Programming port: conn X32 – RJ45
  + ECAT communication: X6 - RJ45 - ECAT OUT port
* AKD2G
  + Logic Power: conn X1 – plug in connector
  + Programming port: conn X20 – RJ45
  + ECAT communication: RJ45 connector – X12 - CAT IN, X11- ECAT OUT ports
* AKT2G I/O
  + Logic Power: 24 volts applied to pins 1 and 2, common to pins 3 and 5
  + ECAT communication AKT2G -ECT -000

**PCMM Controller Setup**

A PCMM based .kas motion/machine project with Safety contains the following:

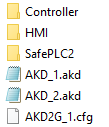
* PCMM
* Servo Drives
* Remote IO
* Safety IO
* SCU Safety controller

Software used in the project could includes

* KAS IDE (Integrated development Environment)
* SafePLC2 Safety Programming SW
* KVB (Kollmorgen Visualization Builder)

**Integration of Safety in a .kas project**

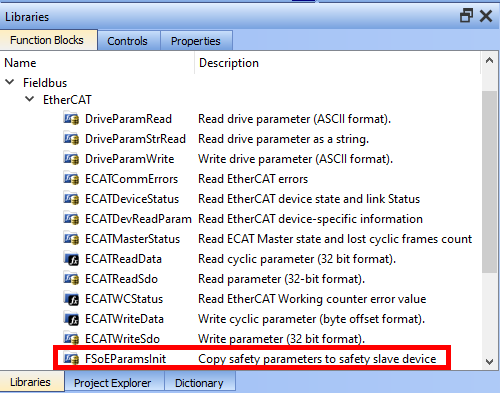
Safety programming is accomplished by first separately installing the SafePLC2. Then opening up the SafePLC2 programming environment from inside the IDE, similar to opening KVB. When SafePLC2 is opened, IDE project Safety elements will be imported into SafePLC2 project programing environment. When leaving the SafePLC2 programing environment the safety program can be saved and become a part of the .kas project. In the development PC the .kas project storage will be:



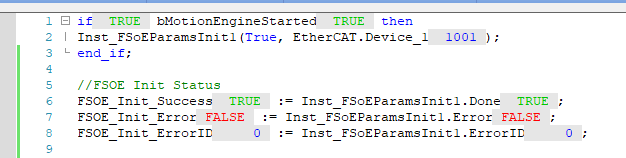
Location: C:\Users\USERNAME\AppData\Local\Kollmorgen\KAS\IDE\_Instances\INSTALL No\PROJECT NAME

**Transfer of Safety Parameters**

In the PDMM, the function block FSoEParameterInit is used to transfer Safety Parameters from the SCU Safety PLC to AKD2G drives. Typically, FSoEParameterInit is executed right after the ECAT motion Network has been initiated (to run mode) during project startup.

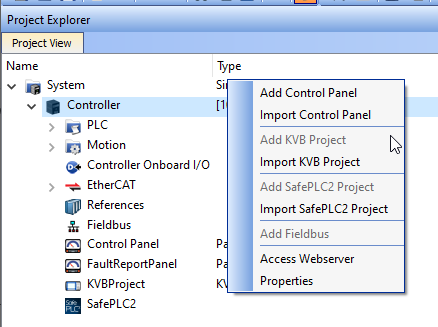


Code example:



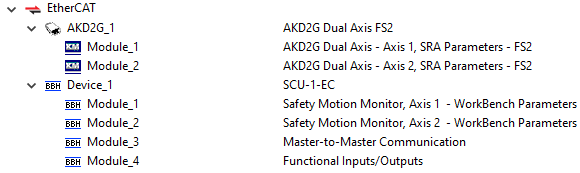
**Adding SCU Controller**

In the project tree, right click on Controller to add or import a SafePLC2 Project



**Ethercat Section (of the .kas Project)**

After the AKD2G drive(s) with Safety Modules (1 for each axis) and the SCU

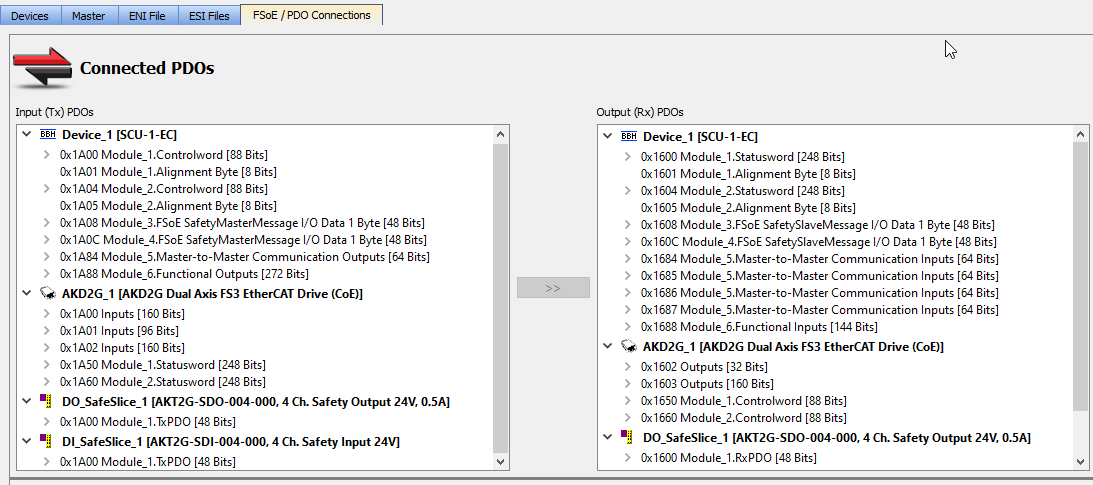
****

*Note: AKD2G safety module “SRA Parameter must be selected. Note: SCU Safety Motion Axis - WorkBench Parameters is the correct setting even when Safety parameters are setup in the SCU*

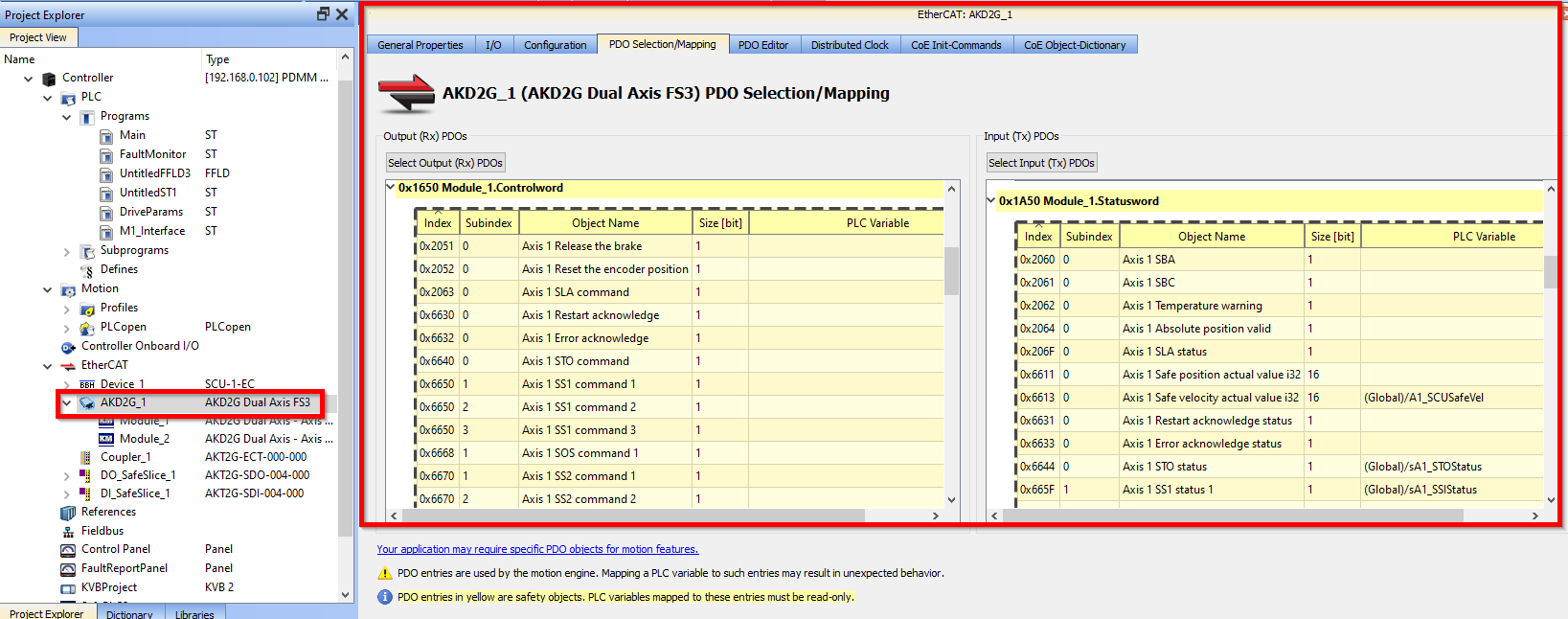
*The AKD2G safety IO if used in the application also needs to be added. KAS supports 2 safety slices, models: AKT2G-IN-004 and AKT2G-OUT-004. A AKT2G-ECT-000 coupler is required. Non-Safety slices can be intermixed with the Safety slices in an IO rack assembly.*

**PDO Safety Communications**

During machine operation, The PDMM (the Ethercat Master) facilitates the transfer of data between the SCU safety controller, Remote I/O, PDMM and the AKD2G drives via FSoE. Once a SCU and AKD2G Safety Drive are added to the KAS project a set of PDO are available for safety information transfer:

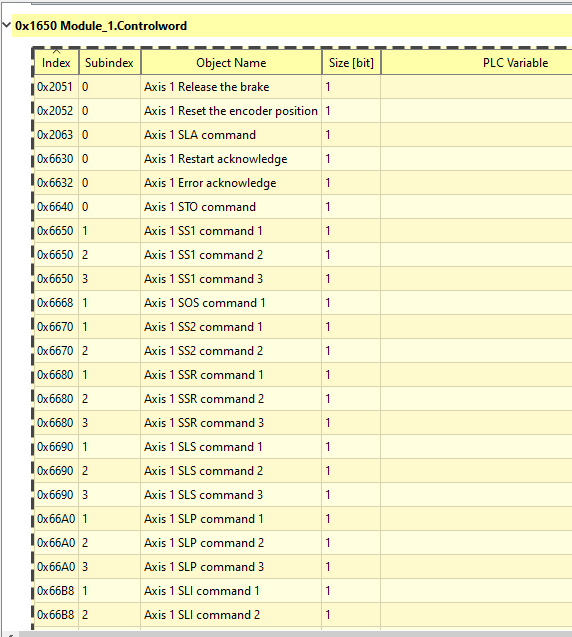
****

Safety Parameters are automatically transferred between the AKD2G drive and SCU safety controller each ECAT cycle. PDO Safety I/O objects are colored Yellow. Once the AKD2Gs and SCU is setup in the project tree, the PDO Selection/Mapping tab allows mapping KAS Project PLC variables to SCU/AKD2G safety functionality

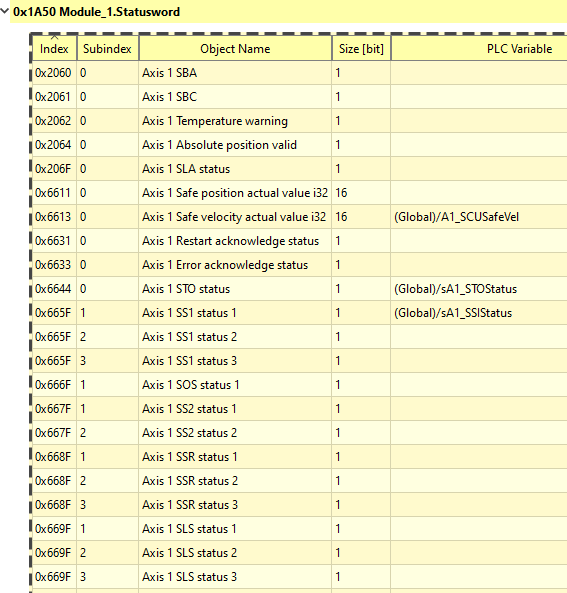


PDO Parameter details:

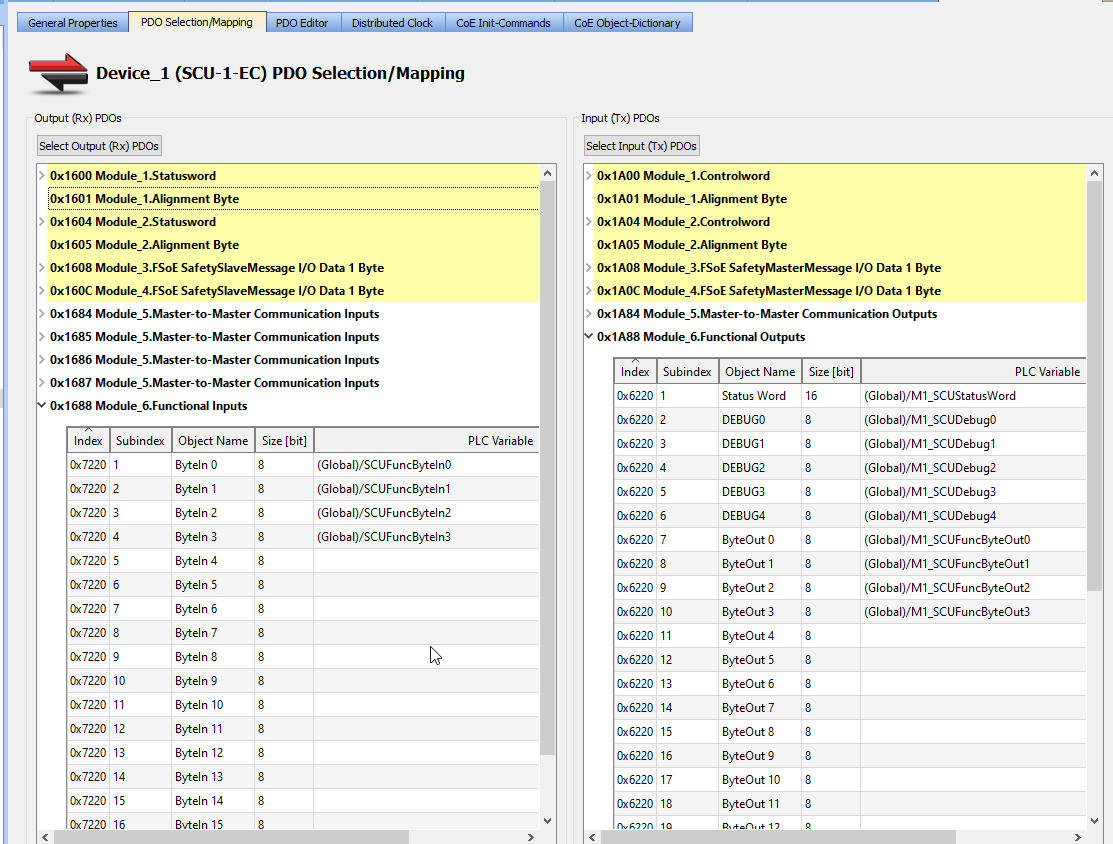
* AKD2G Safety Control - Allows linking FSOE ECAT safety control objects to KAS program variables that can be used in the PCMM PLC code to activate safety functions

****

* AKD2G Safety Status - Allows linking FSOE ECAT status objects to KAS program variables that can be used in the PCMM PLC code to read the status of a particular safety function



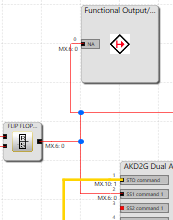
* SCU Function Ins and Outs - Allows the user to read or write values at specific points in the SCU safety program.

****

Functional Inputs

Functional Outputs

Example: SafetPLC2 Function Input and Function Output blocks



Note: Setup of Safety variables can also be done in the AKD2G PDO setup which contains the same set of parameters. Setting up in both areas is not required.

**AKD2G / Workbench Setup**

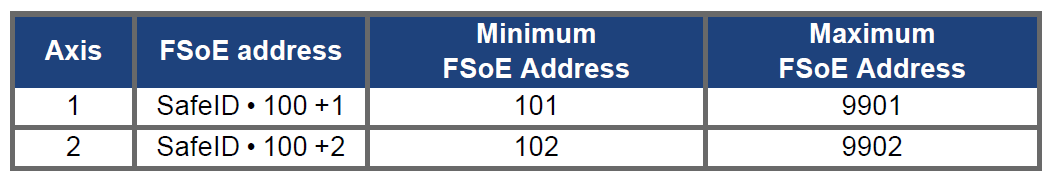
This application note details a configuration in which the drives safety properties are define in the SCU safety PLC. An STO hardware connection into the AKD2G is optional as well as the use of other AKD2G hardware safety IO.

**Workbench Safety Parameter list**

Note: there will l be a safety parameter listing in the AKD2G, reflecting what is downloaded from the SCU controller on ECAT network initialization.

**Setting up the AKD2G Safety card**

The unique FSoE address for every axis is derived from the SafeID address set by the rotary switch on the drives front face.**um**

**FSoE Address**

**Maximum**

Example for FSoE address setting:

Rotary switches setting: S1 = 2, S2 = 3 → SafeID = 10 • 2 + 3 = 23

Resulting FSoE address for axis 1 : 23 • 100 + 1 = **2301**

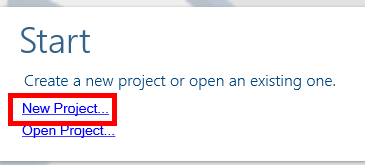
Resulting FSoE address for axis 2 : 23 • 100 + 2 = **2302**

**Default:** With default SafeID setting (01) the resulting FSoE addresses are 1001 / 1002.

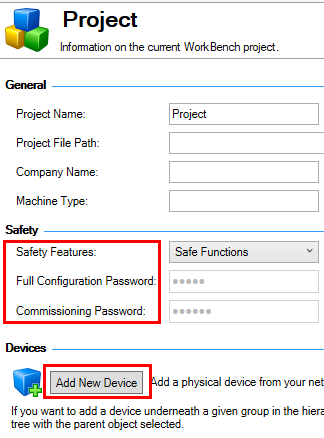
Safety Password is then set up in Workbench

**Creating a new Project - AKD2G Project Screen**

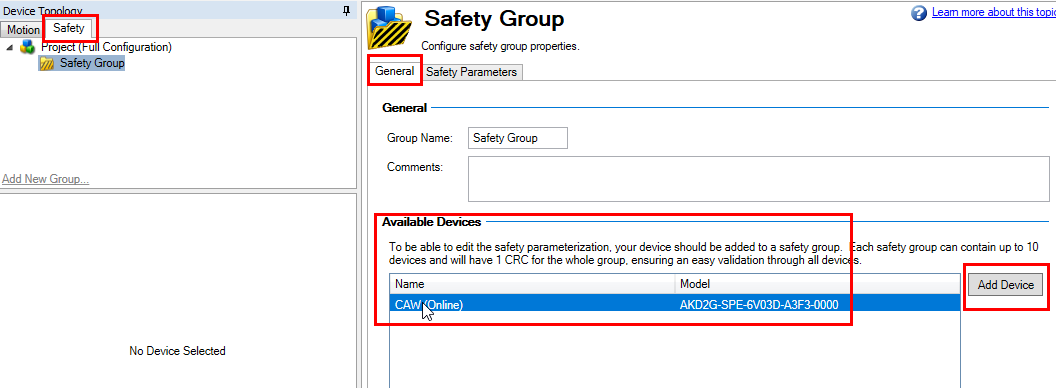
Open Workbench (Standalone version, outside of the IDE) and select new project:



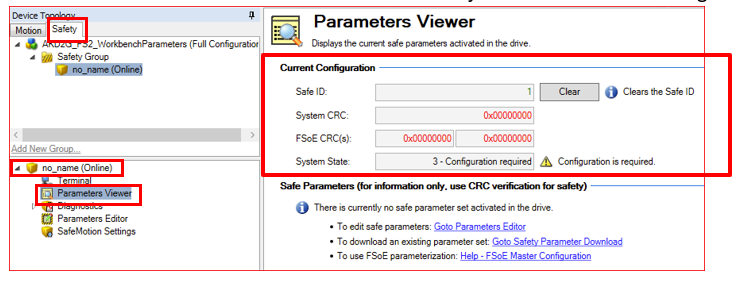
Setup up Safety Features box for Safe Functions, create passwords, then select Add New Devices and add the drive(s) to the project:



Then select Safety Group within the added Safety tab and Add Device to tie the axis into the safety profile



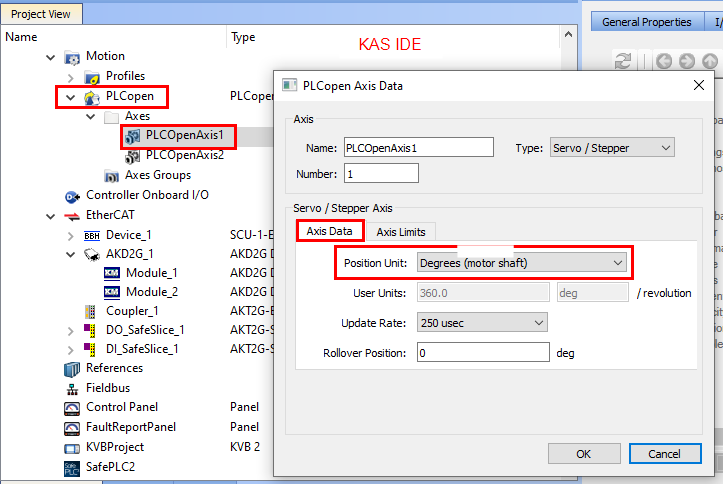
**Clearing any previous safety setup**

Any previous Safety Configuration in the AKD2G should be invalidated and the SMM (safety motion module) will be expecting the safety configuration from the Safety Master SCU-1-EC. Verify the Safety Parameters Viewer in Workbench and ensure the System State is  "3-Configuration required". See below.  
 

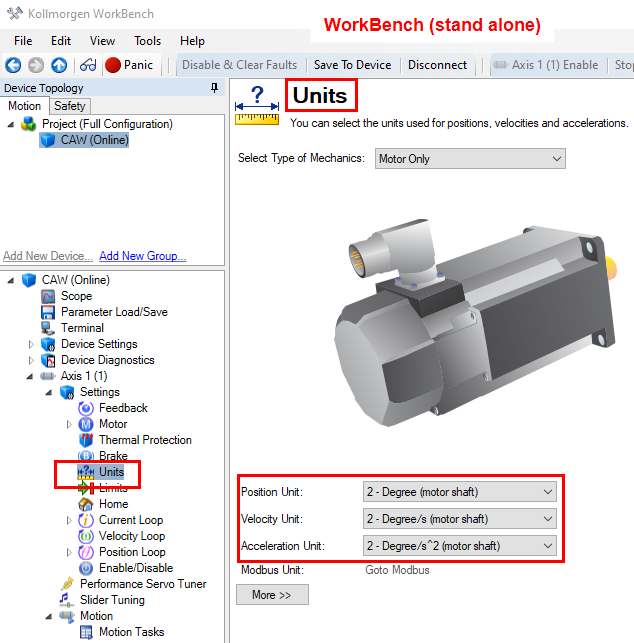
**User Units**

With the PxMM and AKD2G User Units setup in the IDE are automatically transferred to the drive. Example: Position Units: degrees, Velocity Unit: degrees/sec, and Acceleration: degrees/sec2

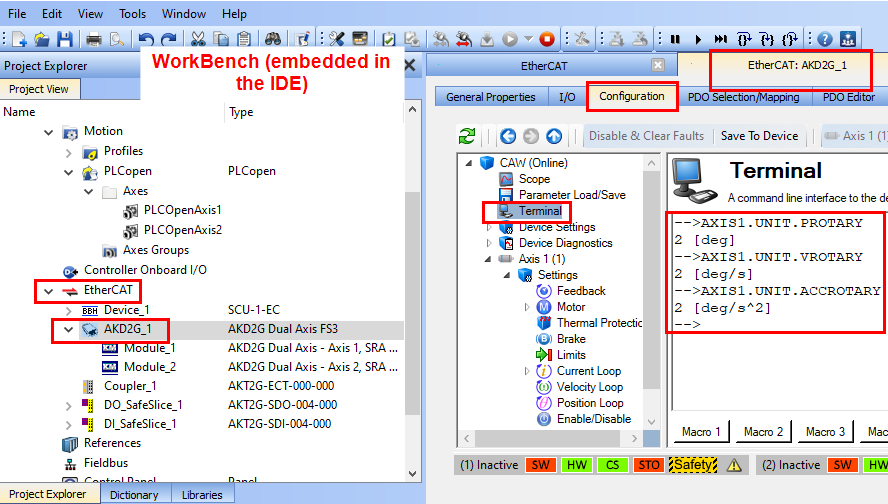
As shown in the IDE:



As shown in WorkBench(Stand alone version):

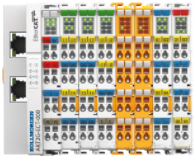


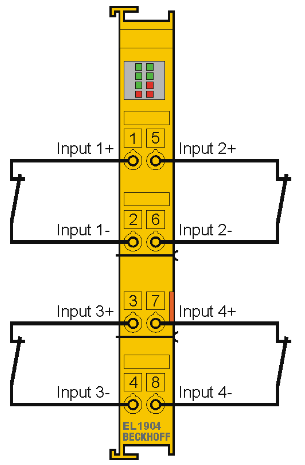
As shown in WorkBench(embedded in the IDE):

****

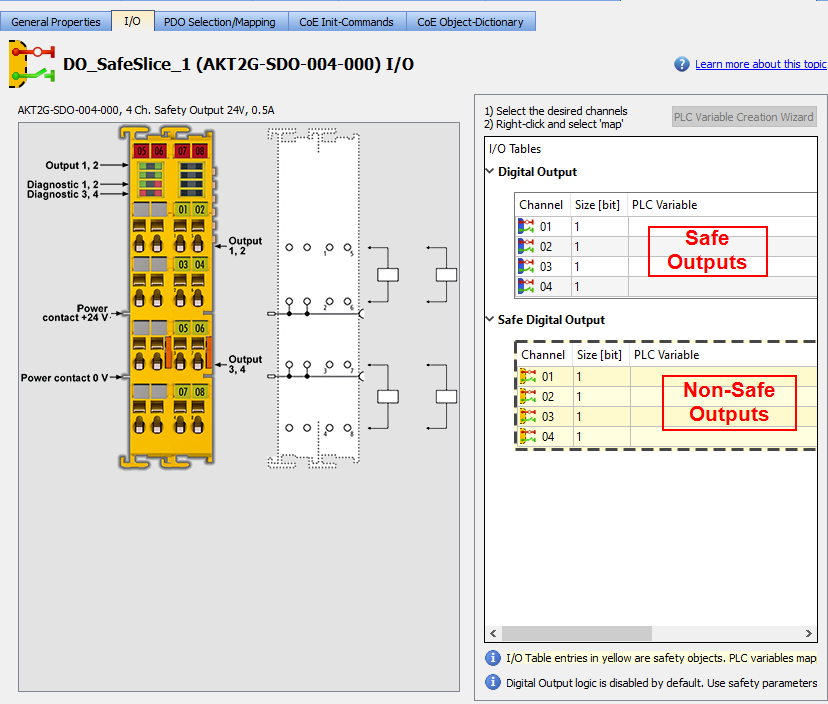
**AKT2G Safety IO Setup**

**AKT Specifics**

* The AKT2G supports 2 safety slices. input module AKT2G-SDI-004 and output module AKT2G-SD0-004 that can be integrated with non-safety slices, then connected to a AKD2G -ECT- 000-000 Ethercat coupler. Example (Safety slices in yellow):
* 
* Unique features of the Safety IO Slices verses non safety IO
  + - AKT2G-SDI-004, 4 channel fail-safe digital inputs. For active high requires a close (SC) connection between 2 inputs terminals. Example:

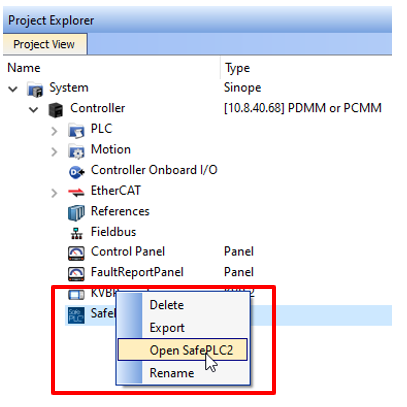


* + - AKT2G-SD0-004, includes 4 channel fail-safe digital outputs and 4 non-safe digital outputs

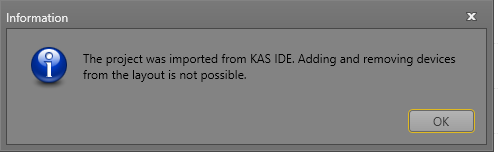


**SafePLC2 Programming Basics**

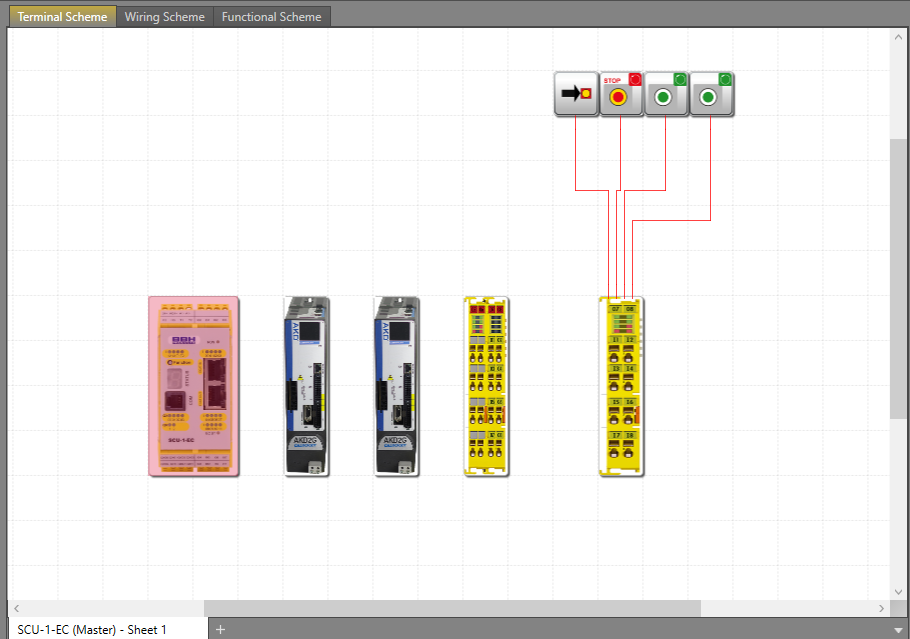
The SafePLC2 user software is opened from within the KAS IDE project tree after it has been added to the project. To open:

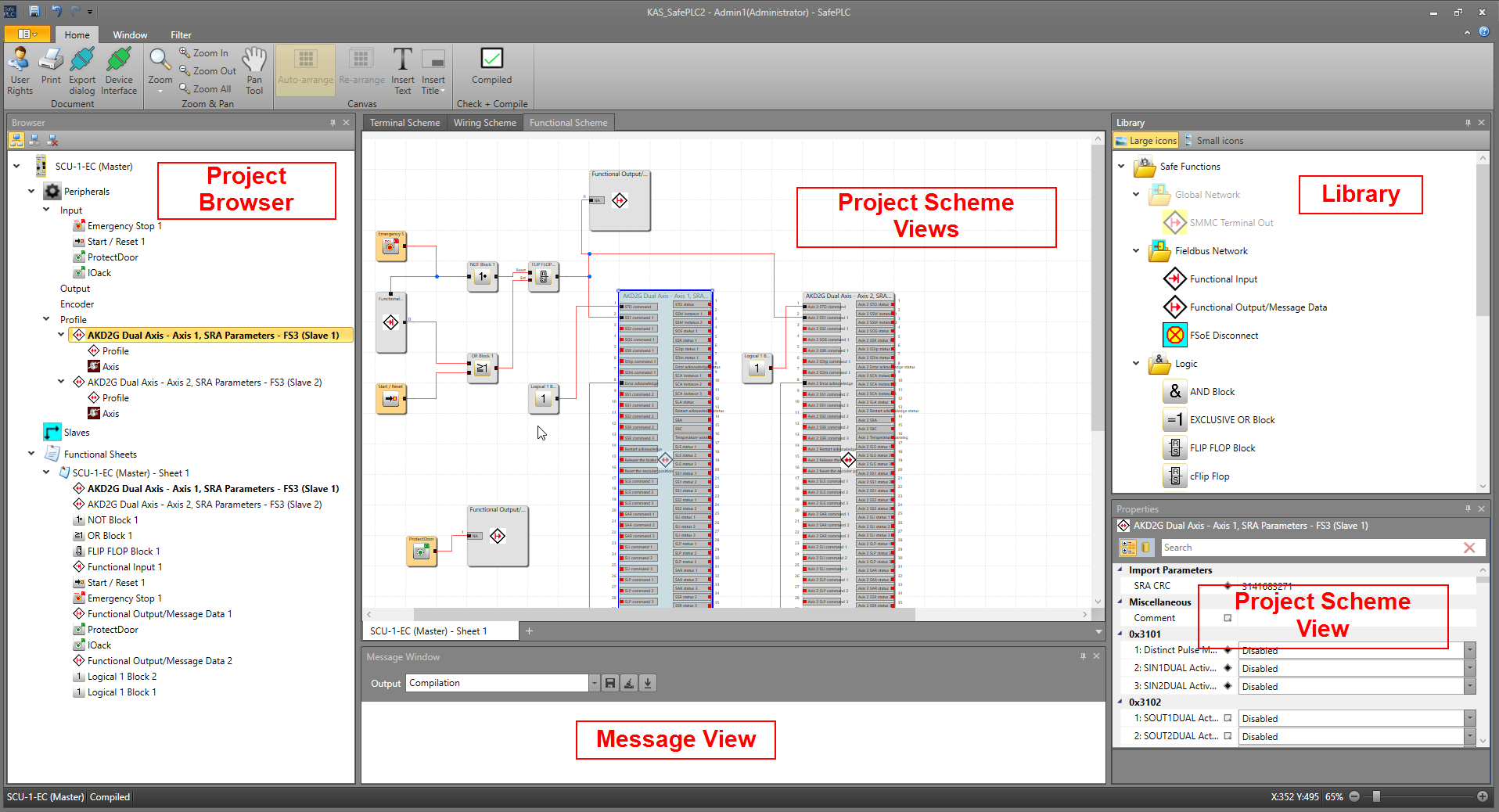


Once the SafePLC2 application is opened the following statement appears stating that the AKD2Ga axis and AKT2G safety IO have been added into the SafePLC2 project..



When the SW opens, in the Terminal and Wiring Scheme views, the AKD2G Axis and AKT2G Safety IO in the .kas IDE have been automatically added to the SafePLC2 software thus simplifying the setup. Example:

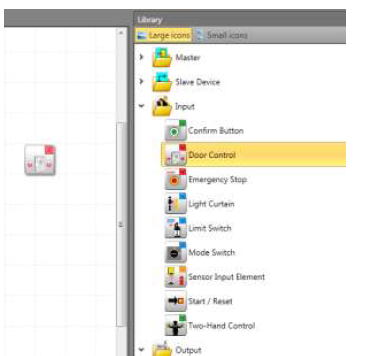


The key parts of the SafetyPLC2 layout are:  


**Object Properties**

**Project Browser** - Includes all objects that have been included to the project

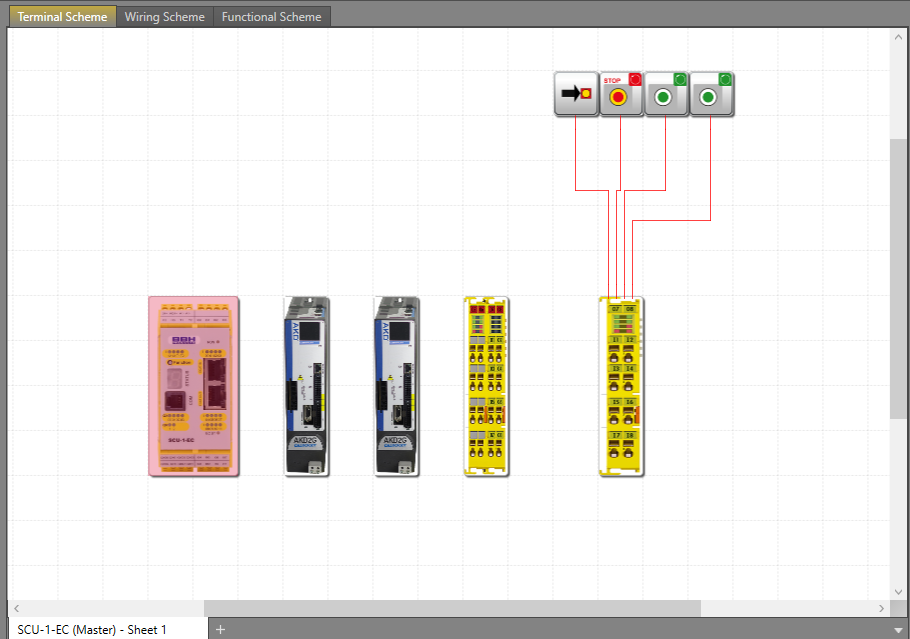
**Library** – List of elements that can be added to the project.



To insert an element into a document, the user must drag and drop the element from the

Library window to the work surface.

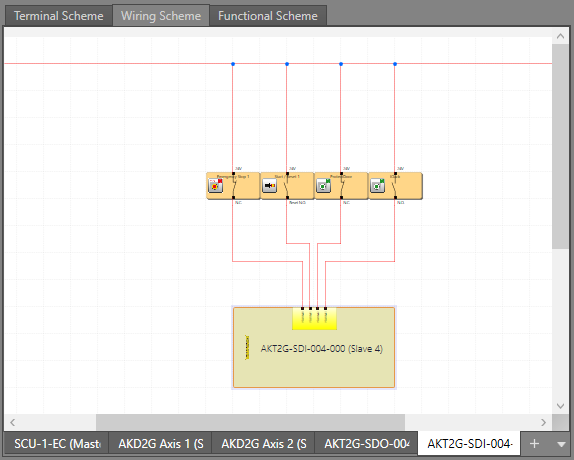
**Project Scheme - Terminal View –** shows a simplified view of selected devices and peripheral devices of the safety system.



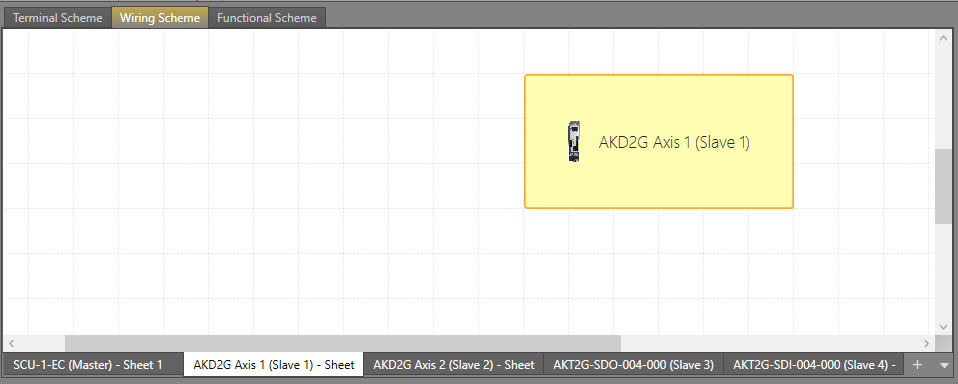
Additional elements inserted into the connection diagram (shown in the red box) are automatically coupled with the device. If several devices are indicated in the connection diagram, the user must add peripheral devices to the corresponding device. Otherwise, the dialogue for device selection appears.

**Project Scheme - Wiring View** - The assignments of SCU external connections to the selected sensor and actuator elements are shown. Each product (SCU, AKD2G Axis, AKT2G safety slice) has a different view. AKD2G axis will show up as a block with no input or output connections because drive setup parameters are part of the AKD2G drive parameter file. Other items include basic wiring. In this view I/O devices can then be added to the project from the library.

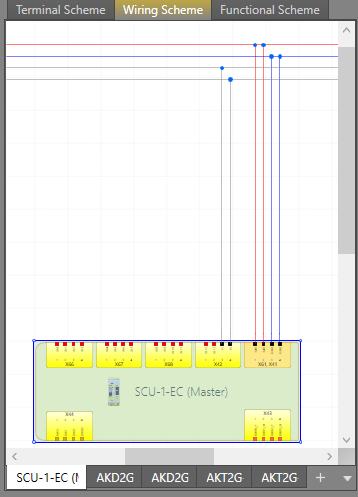
* AKT2G Digital Input Module

****

* AKD2G Drive Safety IO, 1 block for each slice. No connections are shown or need to be setup.

****

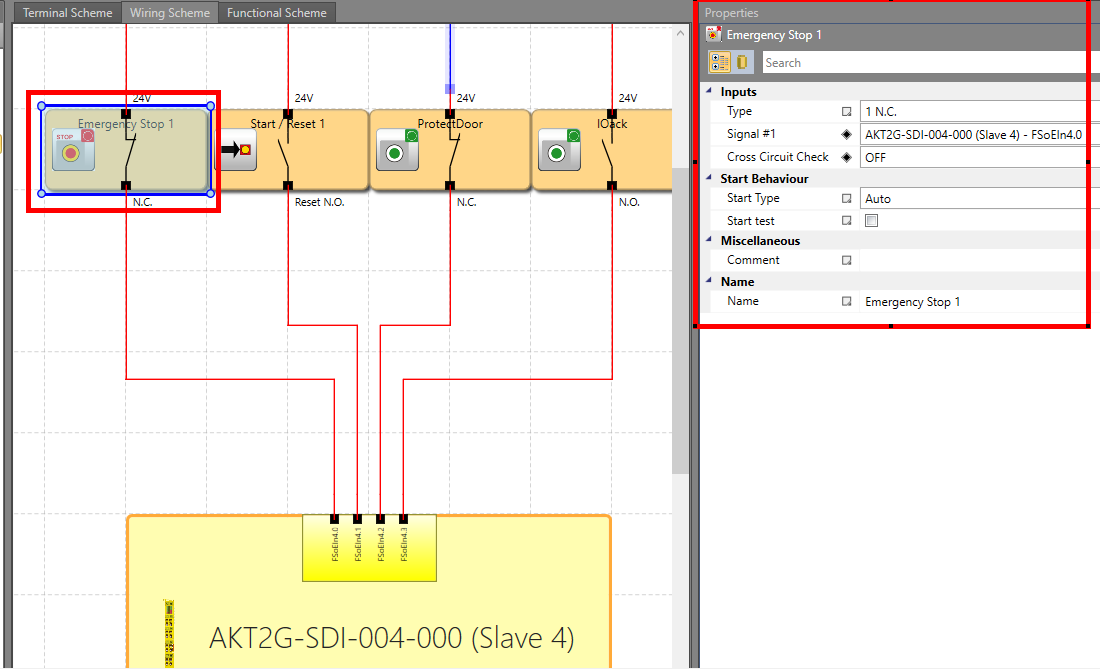
* SCU Safety Controller -Will include at a minimum the 24 VDC connections plus any connections to the SCU onboard IO.

****

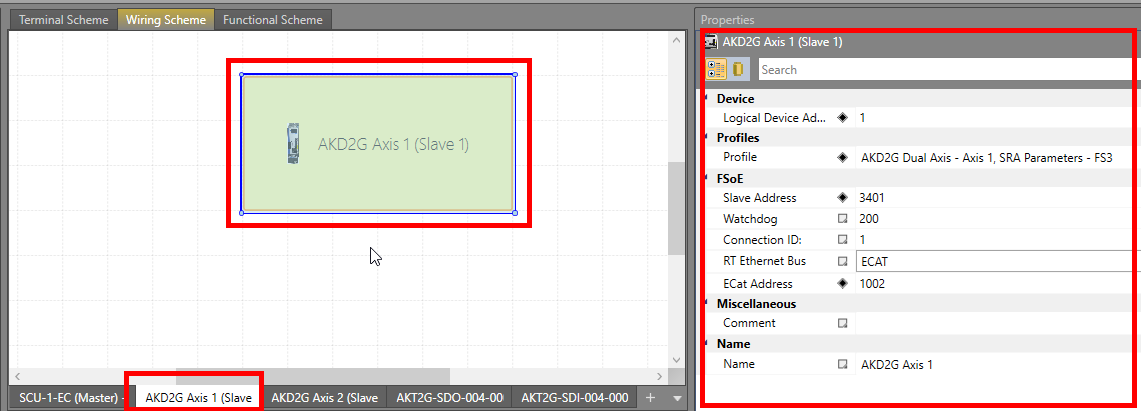
**Initial Configuration**

Clicking on an object in the Wiring Scheme view or in the Project tree will open the property box for the item allowing changes to the default configuration to be made.

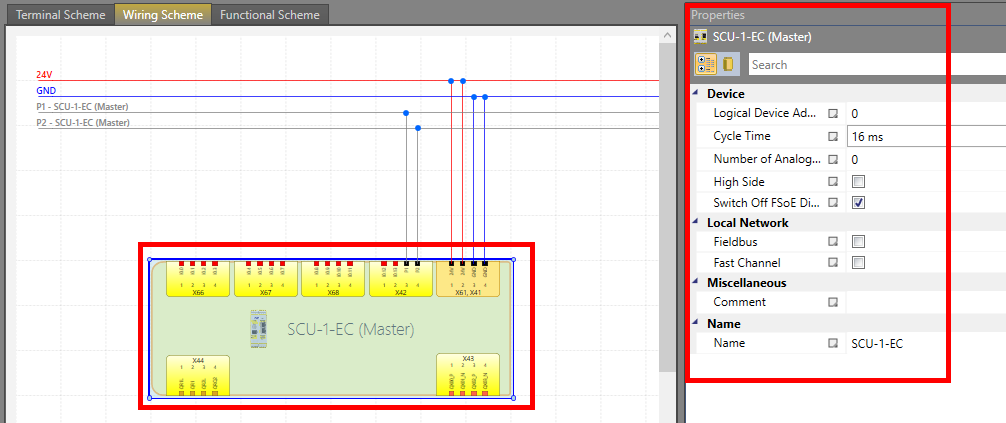
Example – Emergency Stop Button**:**

****

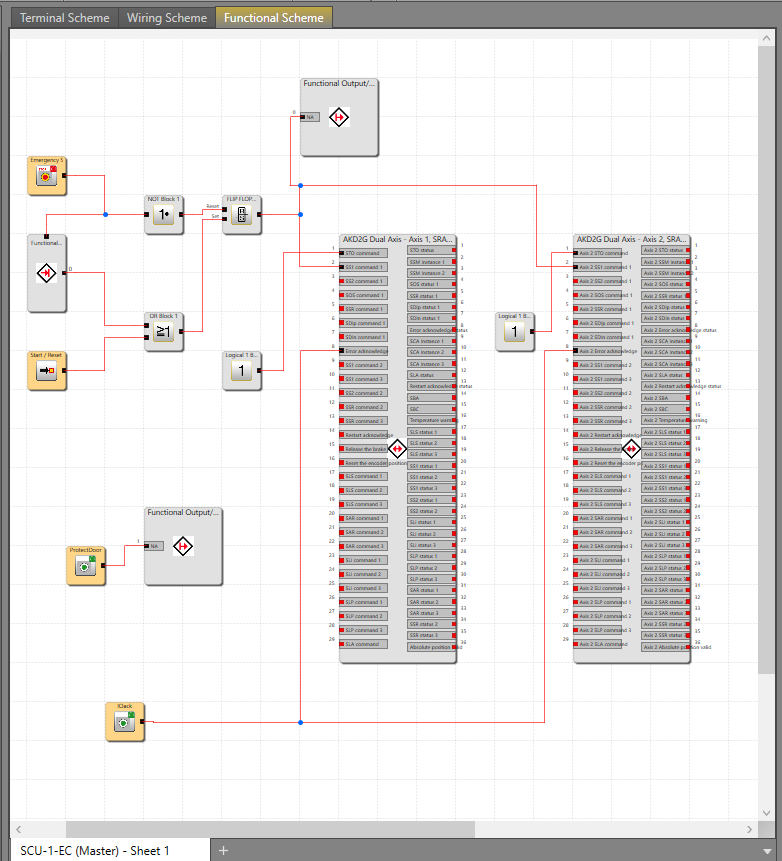
Example – AKD2G



Example: SCU

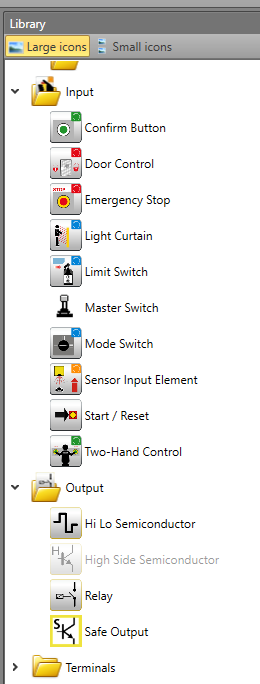


**Project Scheme - Functional View** - Details the safety logic in the SafePLC2 controller the user creates to meet the machines safety requirements In this view, the connections are made between inputs blocks (machine switches, drives inputs, PCMM inputs, AKT2G inputs), control blocks, output blocks (drive safety function activation), and logical blocks. To develop a clearly structured function block diagram, connections must be defined. When animating (project running) a red line indicates “no power” and yellow indicates power



**Adding HW IO**

To add HW IO into the project go to the Terminal or Wiring Screen. THe Library will then contain the options to add.



**Adding Functional Input/Output (Only from/to Functional Scheme)**

These functions can be used in the function diagram and allow information to be moved in and out of a functional diagram from the PCMM controller via PDO mapping of the safety parameters. Note: Many varial have a dedicated PDO mapping, these Function blocks are information not accessible via the dedicated variable mapping

Graphical user interface, application

Description automatically generated

**Logic**

The library also contains logic functions to use when creating the control logic

Graphical user interface, application

Description automatically generated

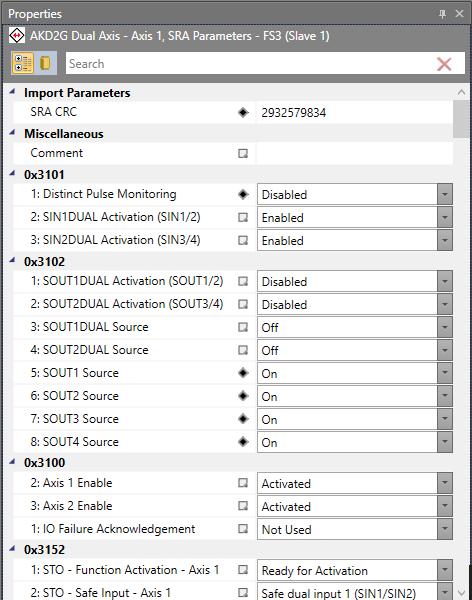
**Object Properties - Parameter setup**

Defines the parameters for the following drive safety functions used in the project.

**Graphical user interface, table

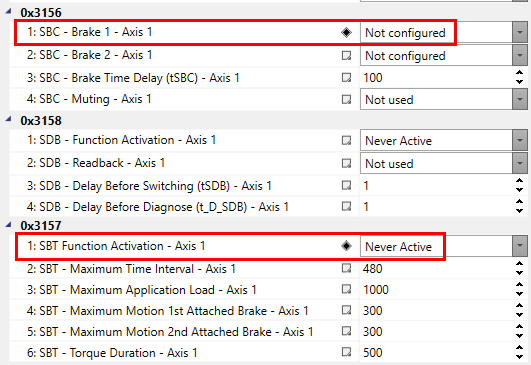
Description automatically generated**

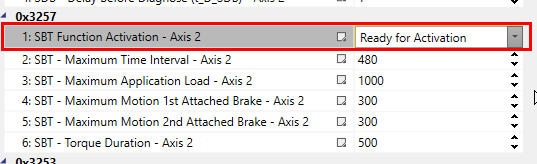
For applications with SCU based safety parameters the drive safety parameters for the AKD2G are defined in this screen. These parameter values are stored in the SCU safety controller and typically are downloaded to the AKD2G Safety drive when the FSOEInit function block is called in the KAS application program.



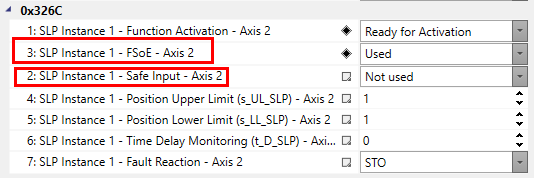
**Notes:**

Axis1 SBC may be configured by default, if so and the axis does not have a brake set to not configured

****

****

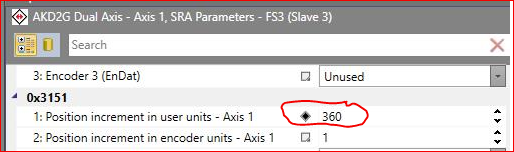
**Source selection** xxxxxxxx



**SafetPLC or AKD2G**

**Present Work around**: AKD2G FS3 drives SRA Parameters through the BBH SafePLC2.

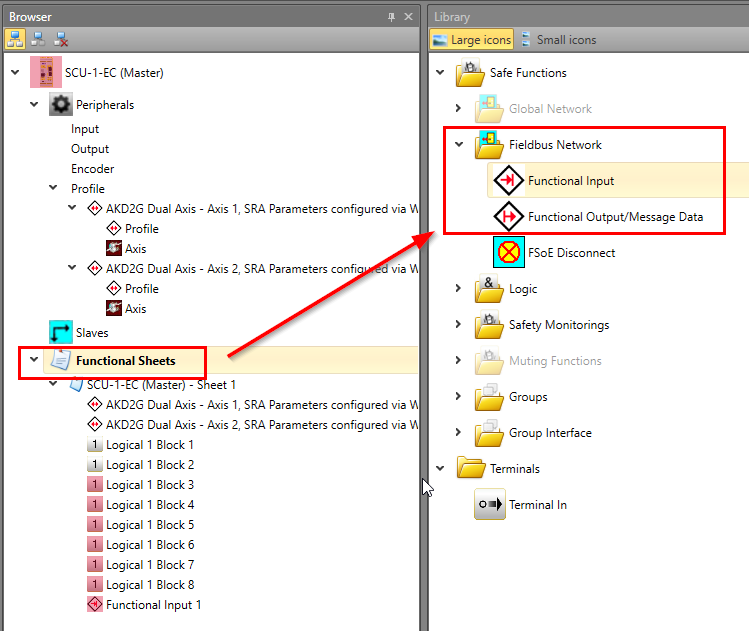
1. The parameter on the object 0x3151:1 (If you have a 2 Axis, then  3251:1 for second axis) should be changed to 360 instead of the default value 3600.This is a mismatch between AKD2G SMM and the ESI file default value.



1. When negative numbers are used for signed Parameters in SafePLC2, the limits are not working correctly. BBH is currently looking into this.

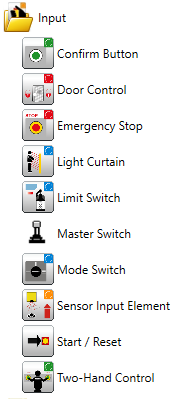
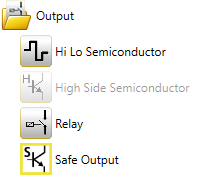
**Adding Functional Inputs and Outputs**

Click on Function Sheets in the Project tree for the Library to show Fieldbis function input and output blocks



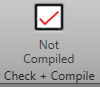
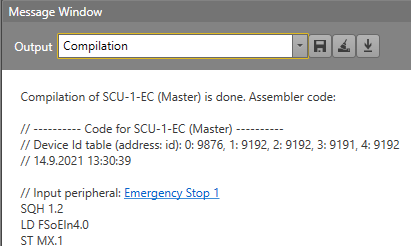
**SCU Hardware IO**

The SCU contains Hardware IO that can be used in a safety project that can be connected to various IO elements:

Each IO has a configuration screen in SafePLC2. Example: for a xxxxxxx the configuration screen is as follows

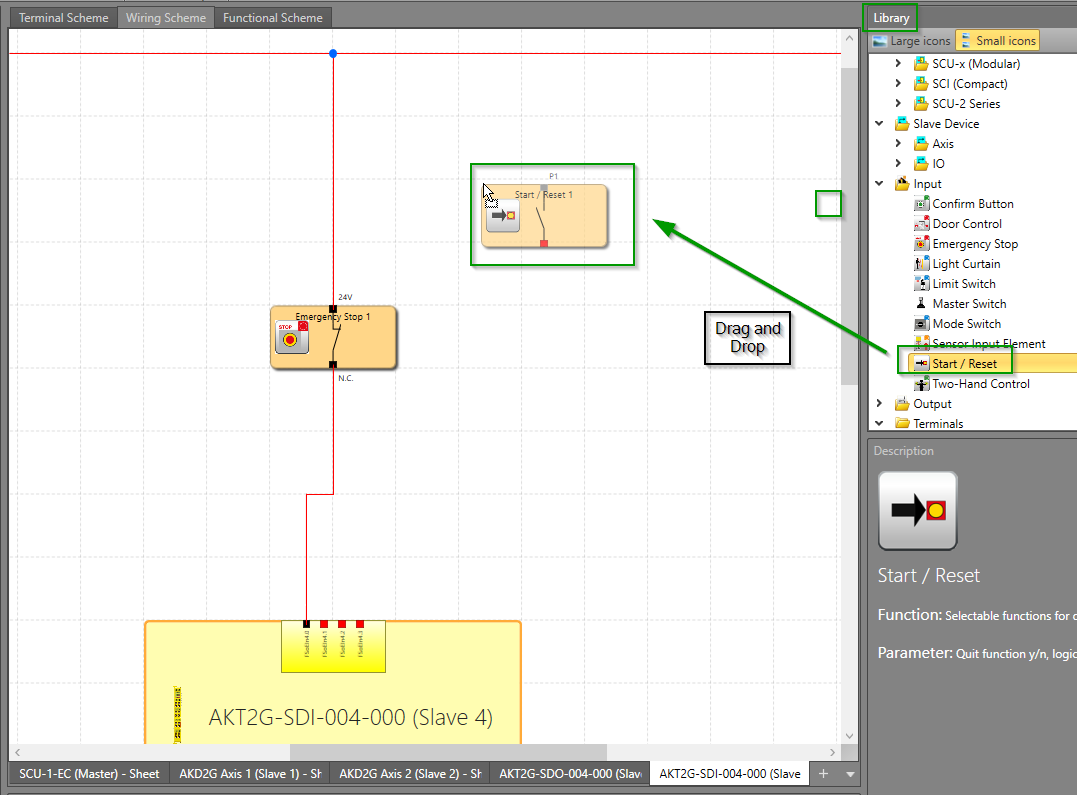
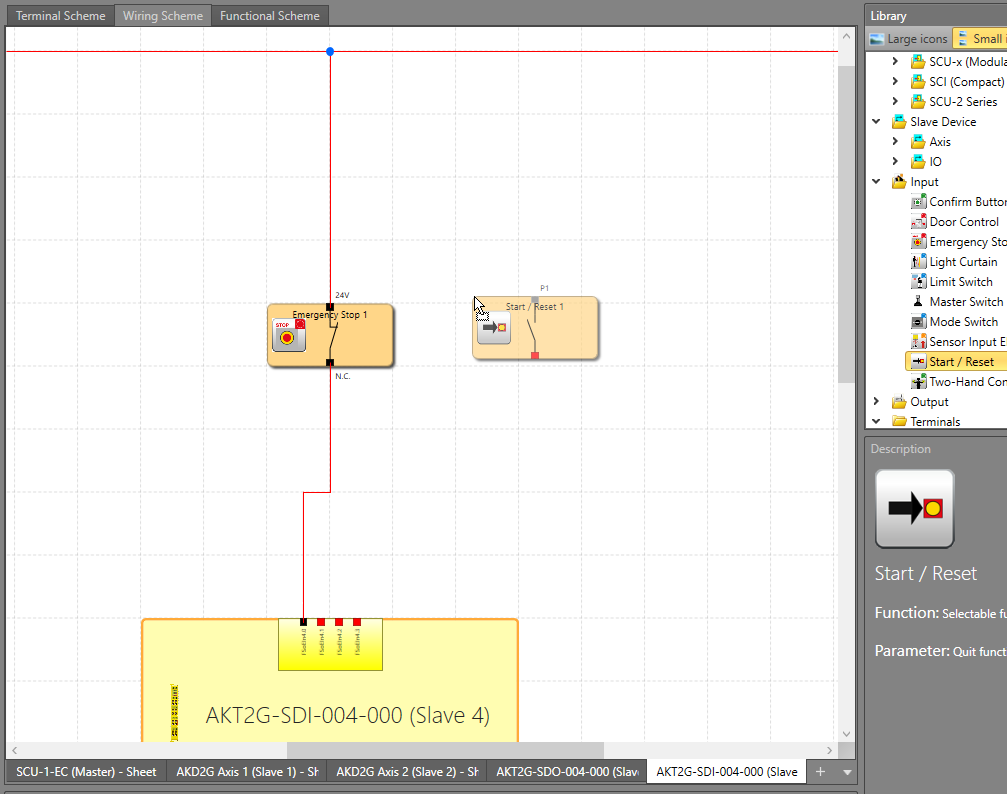
**Compiling a SafePLC2 Project.** Compile your project using the compile button on the top ribbon. Results will be shown in the Compile View

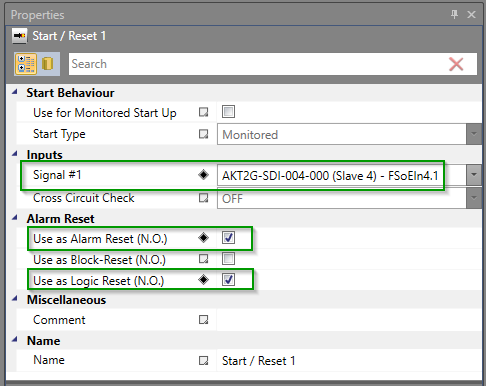
**Building a Safety Program**

Wiring Scheme: Example - Adding Safety Devices

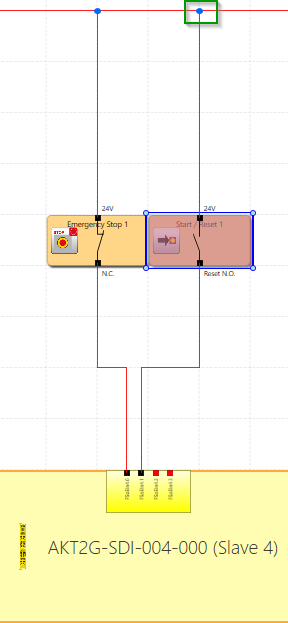
Add safety switches, lights, etc. by first adding to the Wiring scheme. This can be done thru Drag and drop from the Library view into either the Wiring Scheme or Terminal Scheme

 ****

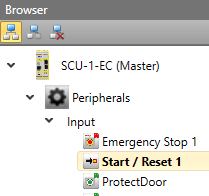
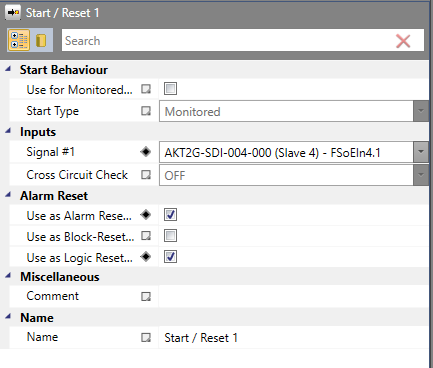
Each device will have a property screen with elements that need to be defined. Example: the configuration of the Start/Reset Switch is done in the property box

****

When the Alarm Reset setup is complete the view will add a connection from the switch to logic voltage

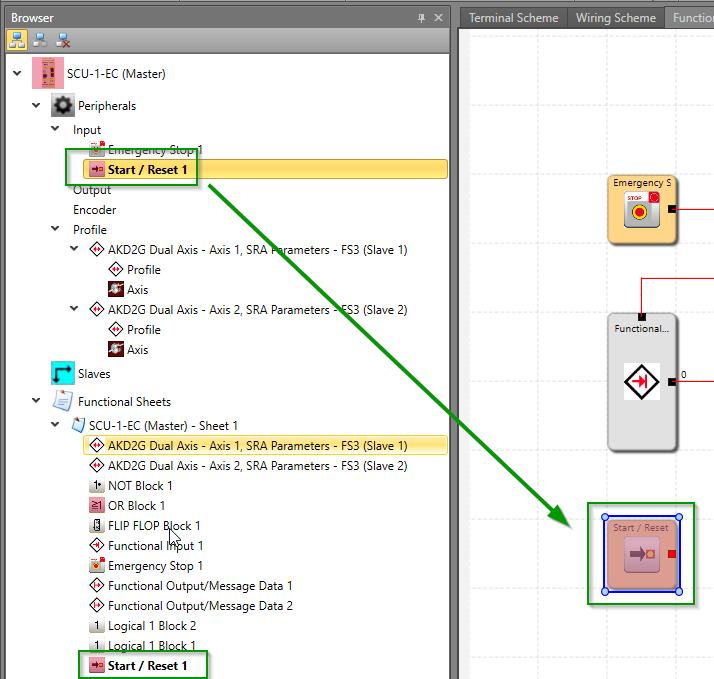
****

Another example is the setup for a Start/Rest Safety Input

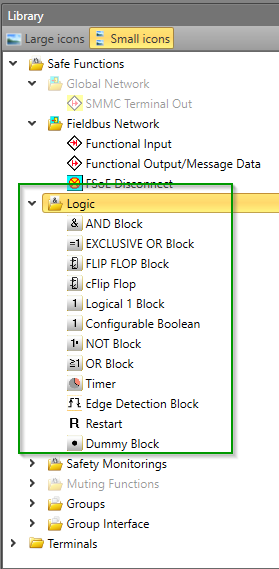
** **

**Functional Scheme: Example - Adding Code**

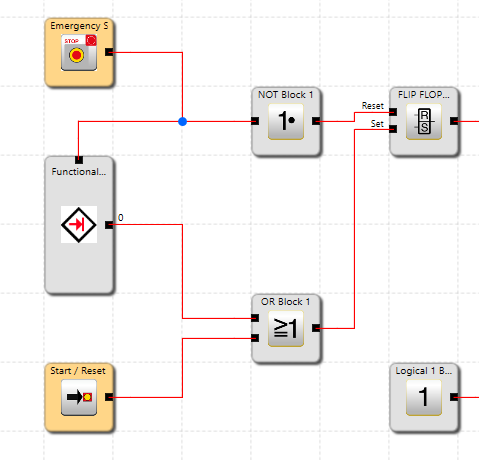
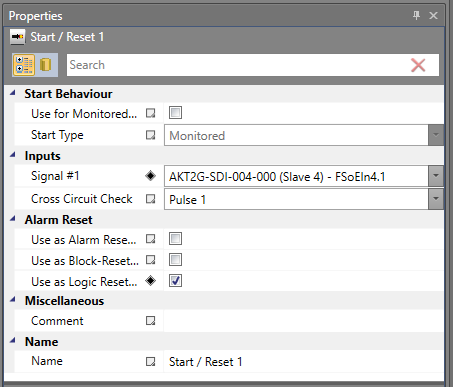
The Start / Reset function can then be added to the function scheme view.

****

Logic functions can be added by drag and drop from the Library

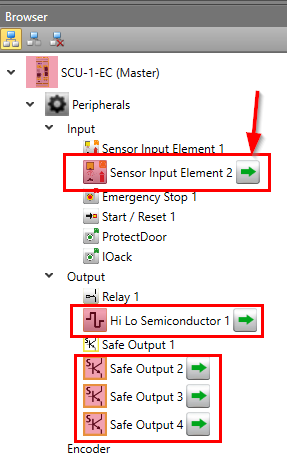


Through this a safety functional scheme can be built. Example:

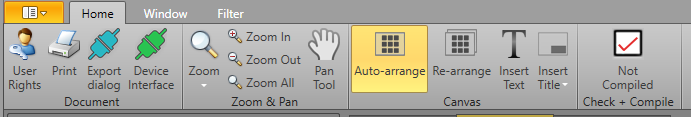
** **

**Connected verses unconnected**

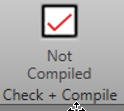
* ** - Indicates not presently used**

****

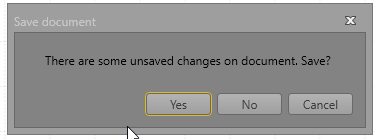
**Compiling a project**



Click on:

****

You will be prompted to save unsaved changes.



Compiled results: If successfully the test code version of the safety code created in the functional screen window will be created. You do not have to do anything with this code. In most cases it is for reference only



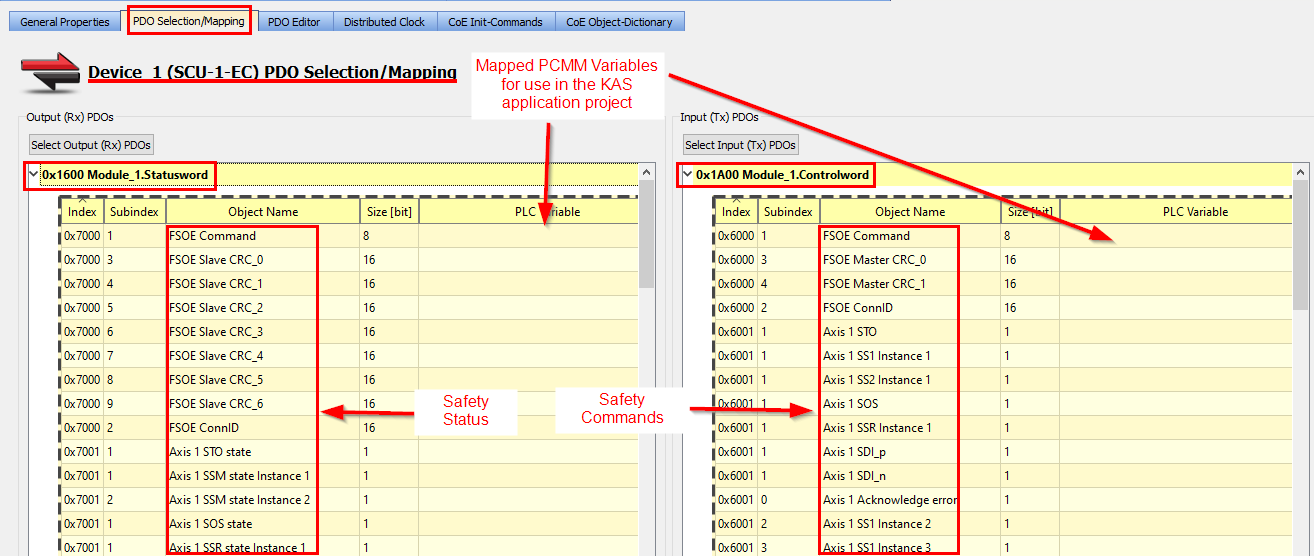


**SCU - PCMM Communications During System Operation**

A set of PDO objects is used for this communication. The following information can be transmitted:

* Direct Safety commands from the PCMM to the SCU
* Status information from the SCU to the PDMM
* Additional Functional Inputs and Outputs to and from the SCU and PCMM

**SCU Safety Commands and Status**



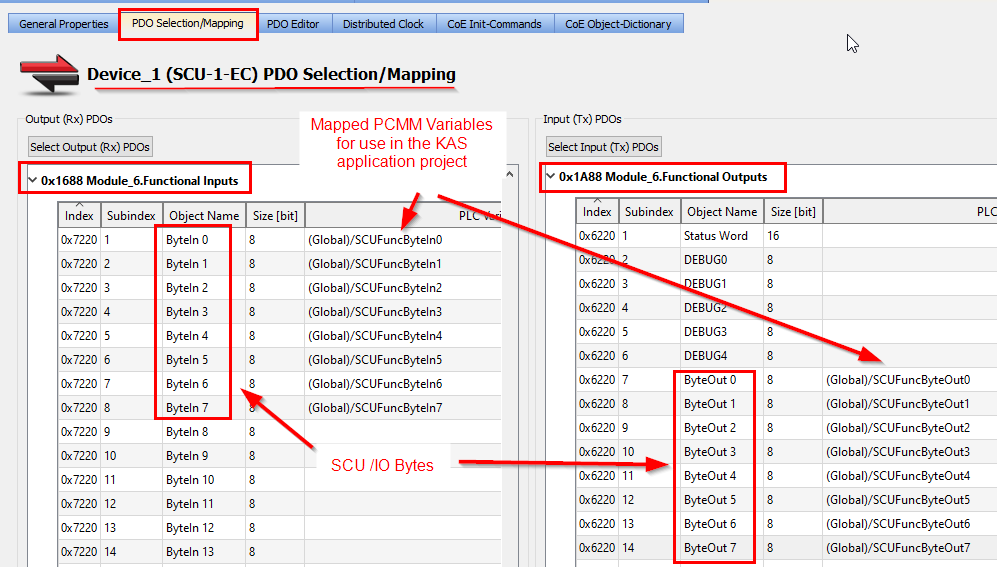
**SCU Functional Inputs and Output PDO Mapping**

Additional Information can be transferred between the PCMM and SCU controller over Ethercat via Functional I/O. On the SCU side Function Input and output blocks are used:

SCU Output Block SCU Input Block

**** ****

On the PCMM side the information (ByteIn and ByteOut) can be mapped to application program via PLC variable mappings down on the SCU PDO definitions

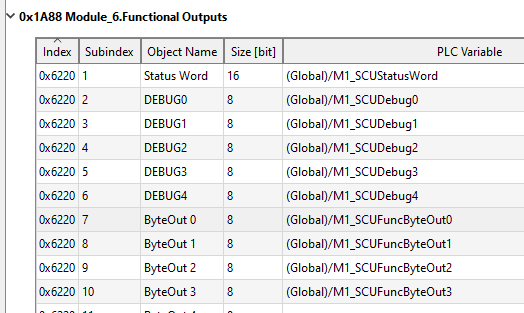


Reset Cmd

SCU Inputs

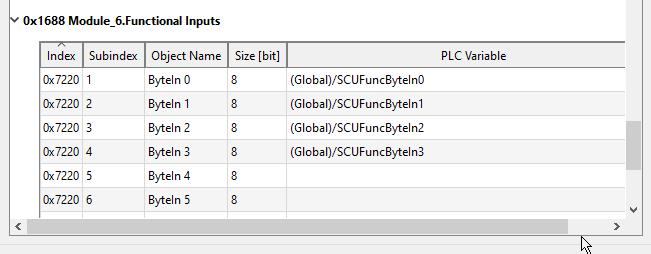
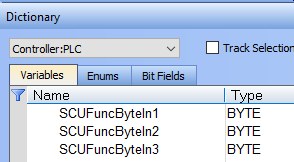
SCU outputs

**Outputs from Functional Scheme**

****

|  |  |  |
| --- | --- | --- |
| **SCU Functional Output No** | **SCU Function Outputs PDO (0x1A88)**  ByteOutx Values: 0 to 127 | **KAS Program Variables** |
| 0 to 7 | ByteOut0 | SCUFunctByteOut0.0 to  SCUFunctByteOut0.7 |
| 8 to 15 | ByteOut1 | SCUFunctByteOut1 |
| 16 to 23 | ByteOut2 | SCUFunctByteOut2 |
| 24 to 31 | ByteOut3 | SCUFunctByteOut3 |
| 32 to 39 | ByteOut4 | SCUFunctByteOut4 |
| to | to | to |
| 208 to 215 | ByteOut26 | SCUFunctByteOut26 |

**Inputs Functional Scheme**

**** ****

|  |  |  |
| --- | --- | --- |
| **SCU Functional Input No** | **SCU Function Inputs PDO**  **(0x1688)**  ByteInx Values: 0 to 127 | **KAS Program Variables** |
| 0 to 7 | ByteIn0 | SCUFuncByteIn0.0 to SCUFuncByteIn0.7 |
| 8 to 15 | ByteIn1 | SCUFuncByteIn1.0 to SCUFuncByteIn1.7 |
| 16 to 23 | ByteIn2 | SCUFuncByteIn2.0 to SCUFuncByteIn2.7 |
| 24 to 31 | ByteIn3 |  |
| 32 to 39 | ByteIn4 |  |
| to…… | to…… |  |
| 136 to 145 | ByteIn17 |  |

**Running a SCU Safety Program**

After successfully compiling the project it can be downloaded into the SCU -1-EC and run.

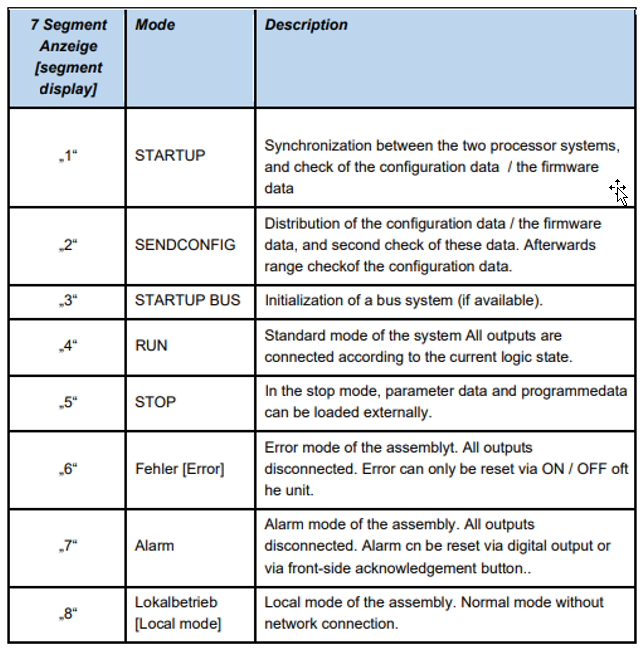
**Downloading to the SCU Controller**

The cable to download is Model SMX-91



has a RJ11 connection into the SCU and a USB connection into the Development PC. Power the SCU. In a successful startup the front face LED display with transition from 1, to 2, to 3, to 8.

Description of the SCU front face single digit LED values:

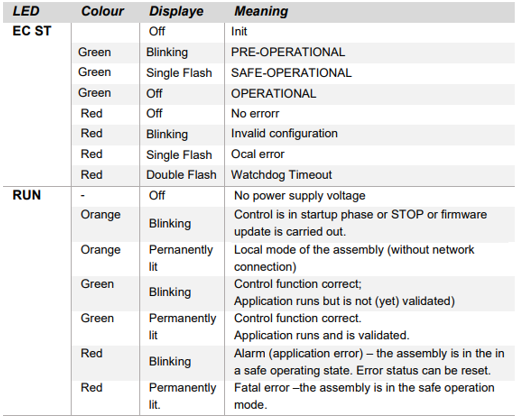


Run mode with Ethercat Network UP

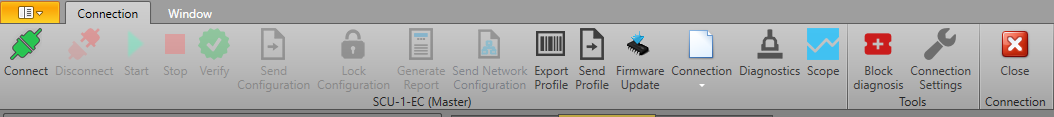
Run mode without Ethercat

Power Up

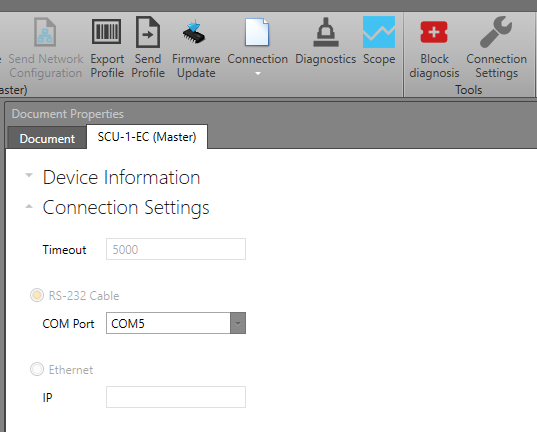
Description of the SCU LED lights:



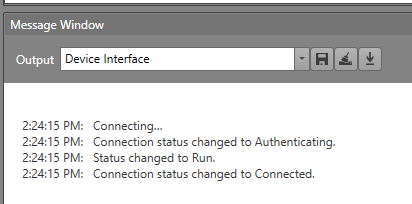
Click on Device Interface  and the Connection window appears:



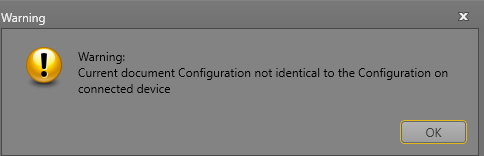
From there, set up the connections  selecting an available comm port



Then click connect  . The Message View provides the status of making the connection.



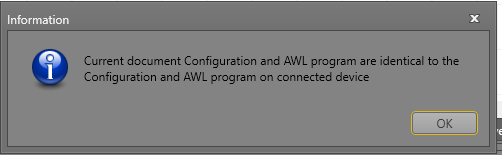
Click on  and if the safety project you are working on is different from what is presently in the SCU , this message will come up:



Download the Project to the SCU Safety Controller  . Bottom tray will indicate the status of the domnload:



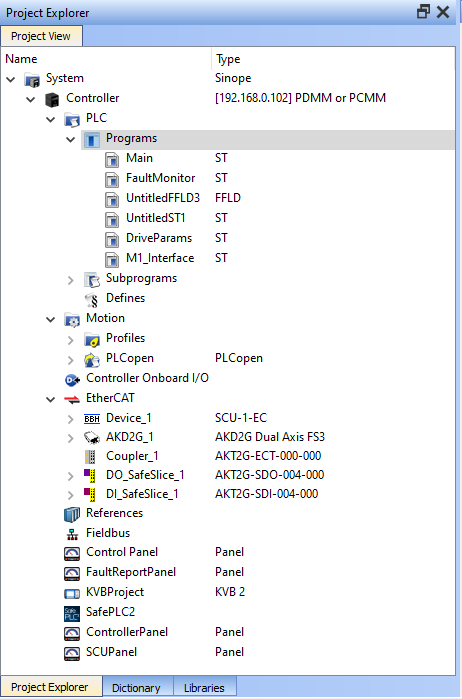
During downloading, theLED will change to a flashing “5”. Then verify samenessby clicking on . If successful the following will come up:



SCU Run LED – Blinking green – when application up and running but not yet validated

**Project Example**

The project View in the IDE

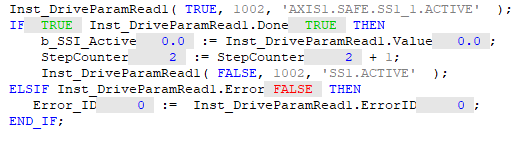
****

*Startup: After downloading cycle logic power to the SCU and AKD2G*

*To manually start:*

Start the project with this button:  . The SCU’s front face LED Display will go from 5 to 8. This is the ready state waiting for the ECAT network to start. The dot in the display after the “8.” will blink during the safety startup

Next Start of the IDE project. When the ECAT network is in run mode the SCU LED display will display a value of “4” and the SCU safety parameters will be downloaded to the AKD2G drive(s) via the FSOEINit function block execution in the .kas project



**Safety Parameter Transfer (SafetyPLC2 based safety configuration)**

When AKD2G/SMM is powered on, it loads the default values for the axis safety parameters to its memory and computes the CRC. The same default safety parameters values are imported into the SafePLC2 when it is opened via the IDE. The user using SafePLC2 then configures the safety parameters as needed for the application. A SafePLC2 safety file is created.

When the FSOEinit function block is executed in the KAS program all SafePLC2 parameters not set to default are down loaded into the AKD2G via the FSOE safety channel. After all parameters are loaded the AKD2G/SMM checks whether the CRC received from the FSoE master (SCU) matches the new calculated AKD2G/SMM CRC value. If the CRCs matches, then SMM state is changed to "Process Data".



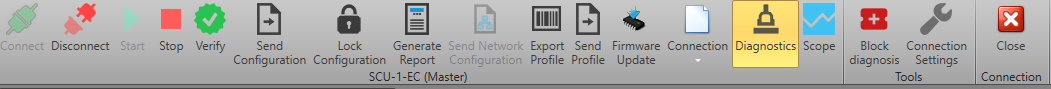
If there is not a match the AKD2G/SMM remains in the following state:  
 

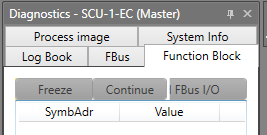
**Monitoring a Running Safety Project - Project Animation**

Operational Status of the Project is available thru:

* + SafePLC2 SW
  + IDE SW
  + Workbench SW

**SafePLC2 Connection Tab:**

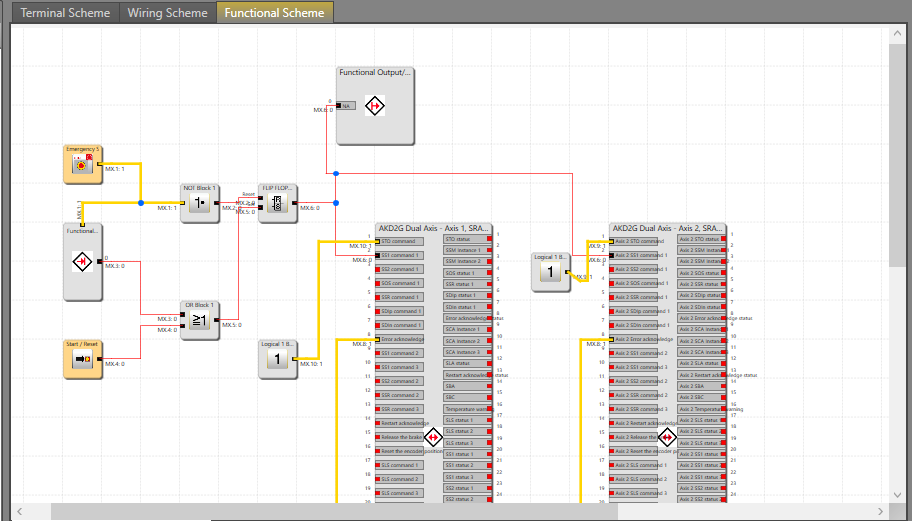


****

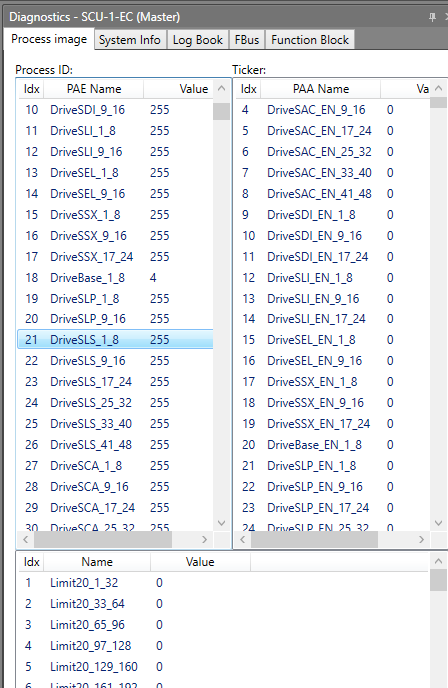
**Animation in SafePLC2**

Animation of the Functional Scheme shows the status of the SCU Safety Logic:

* Yellow/range line = high signal
* Red Line = Low signal

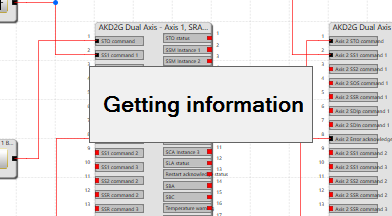


Drive Safety parameters animation:

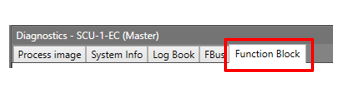


With this a user can see safety status and trouble shoot their application. To animate click on: 

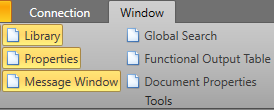
You will see this:

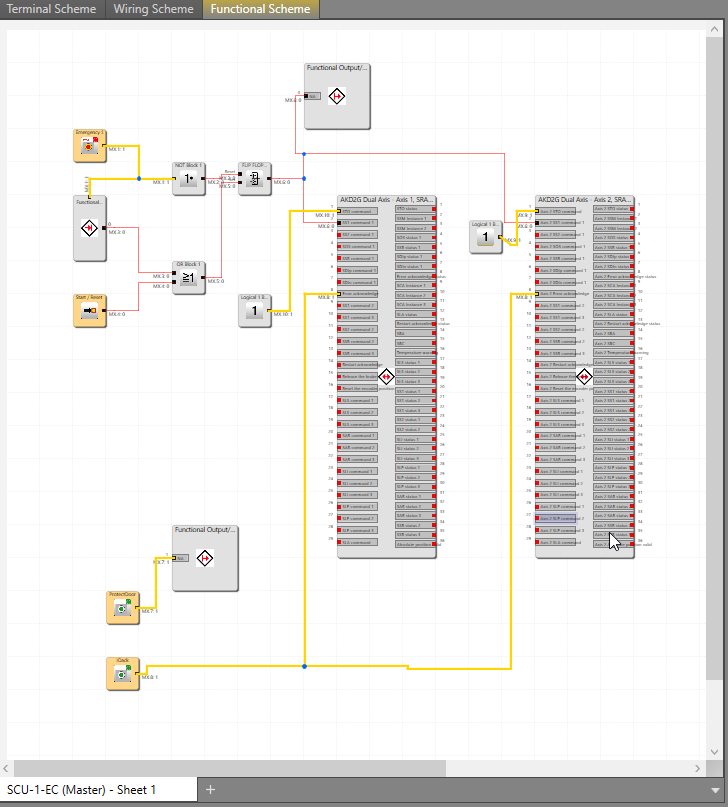


Then click on the Function Block tab:



Note: You may need to toggle the “Document Properties” button to un-highlight it





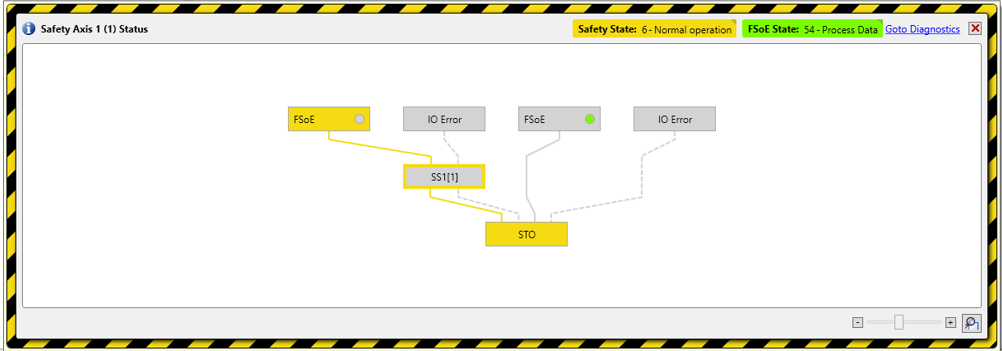
**Startup**

*Note: To view Workbench safety configuration click on  in the bottom tray*

1. Workbench - Pre parameter download from SCU

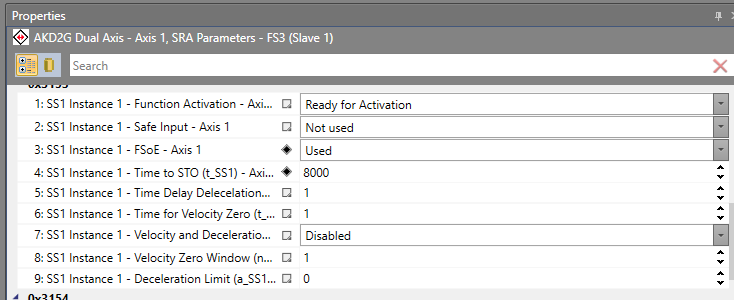


1. Post Parameter download

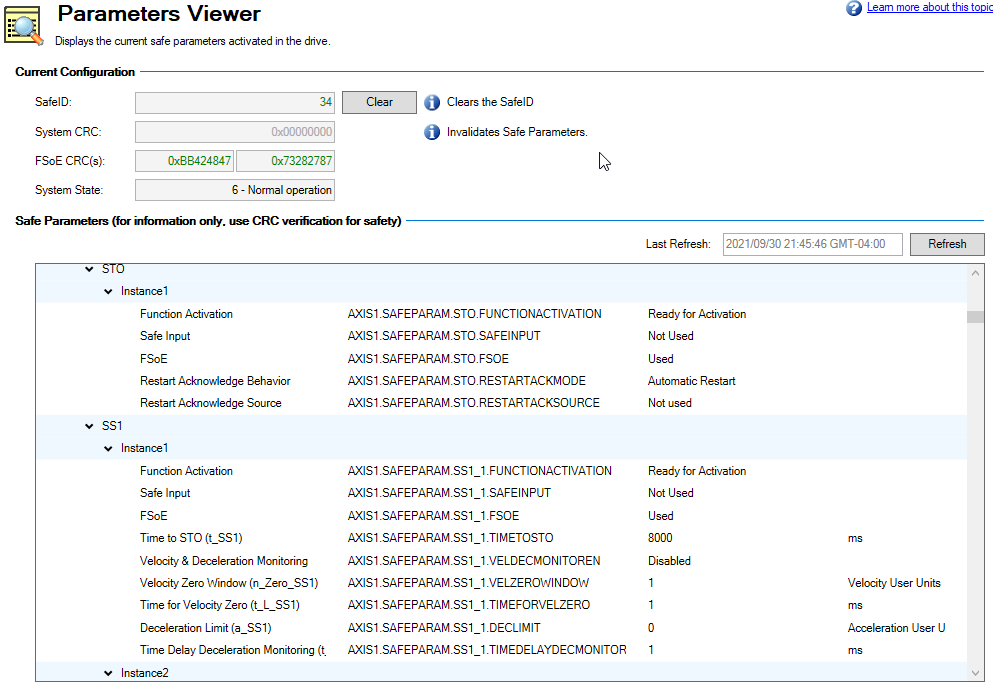




All the safety properties set up in the SafePLC2:



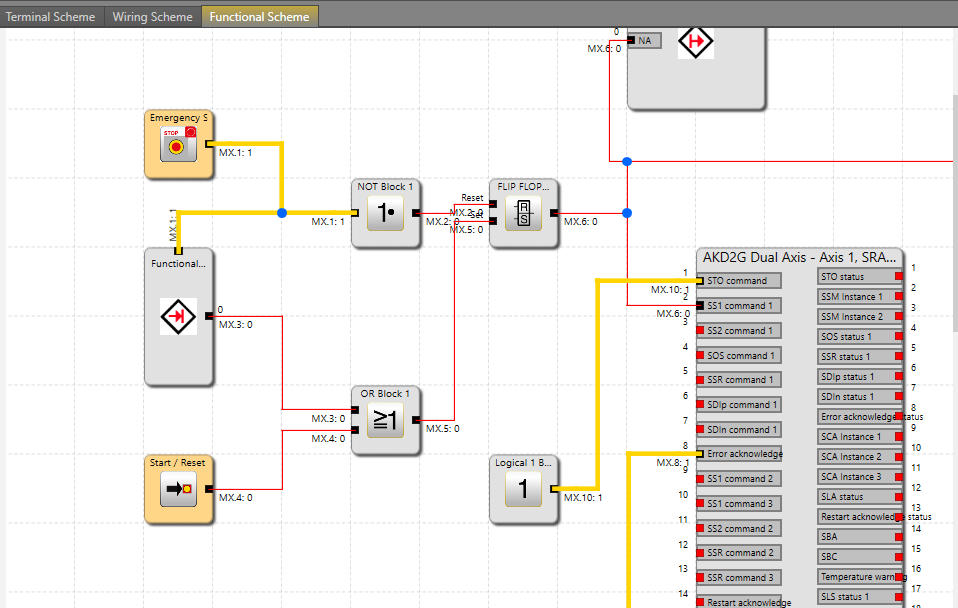
are now reflected in the Workbench Safety Parameter Viewer:



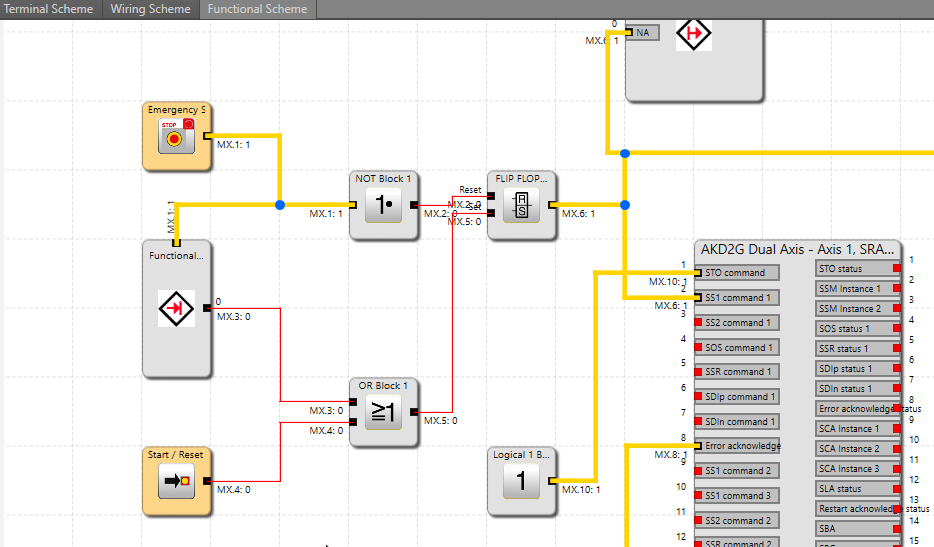
1. Startup

the SSI Safety Stop 1 input (red line) to the AKD2G being low

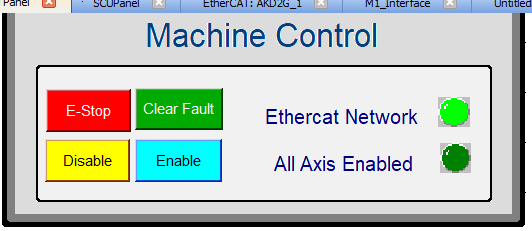


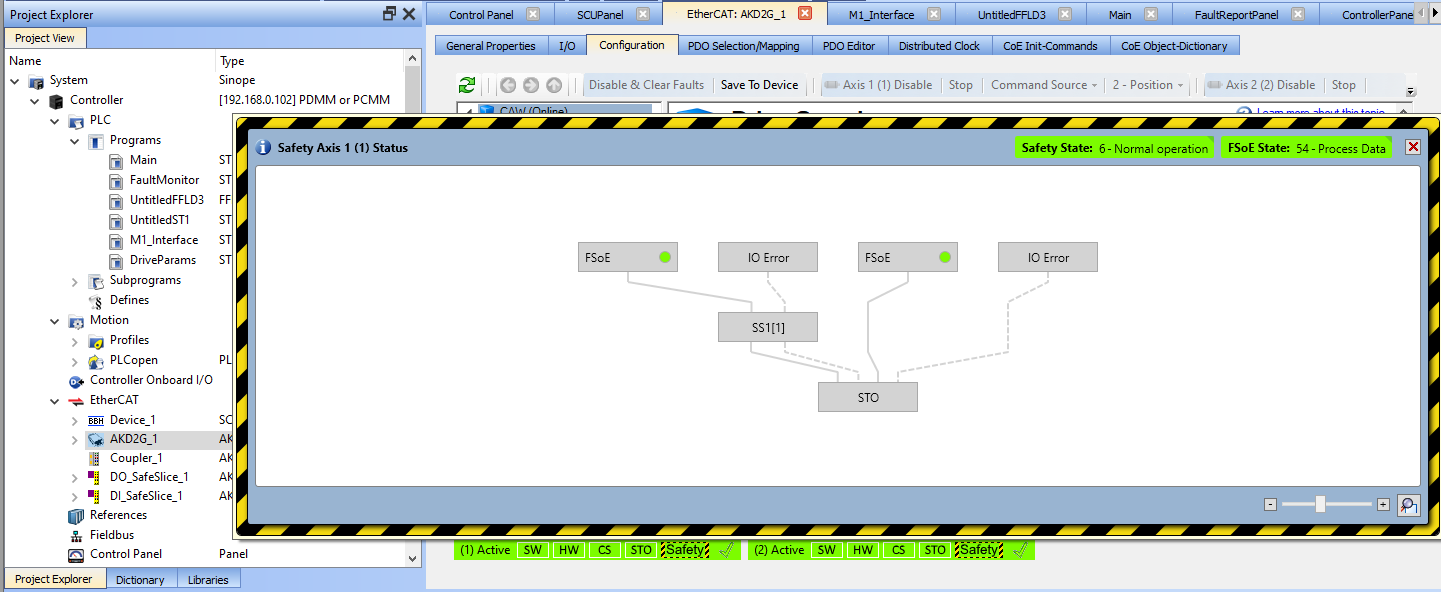
1. Toggle the Start/Reset Switch and get the following. Note the SS1 command is high going into the AKD2G FB: 

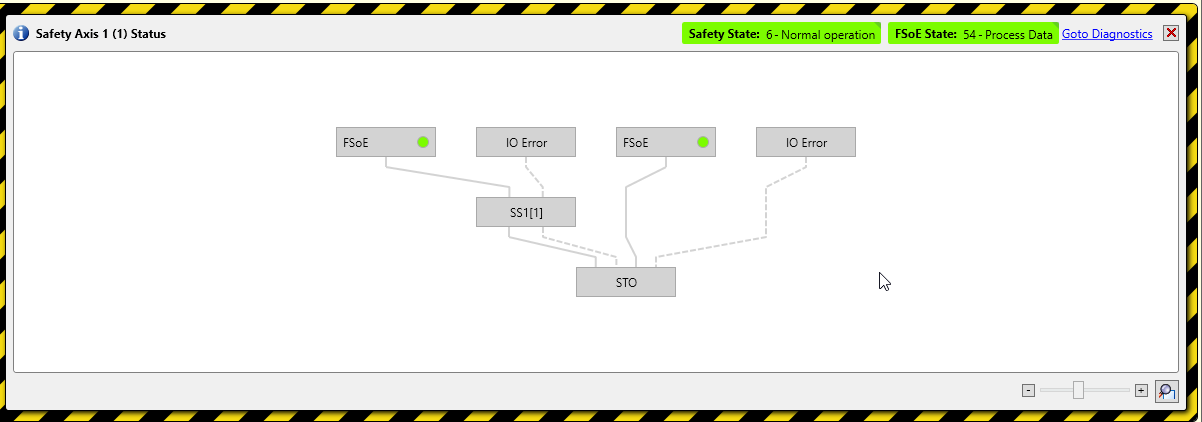


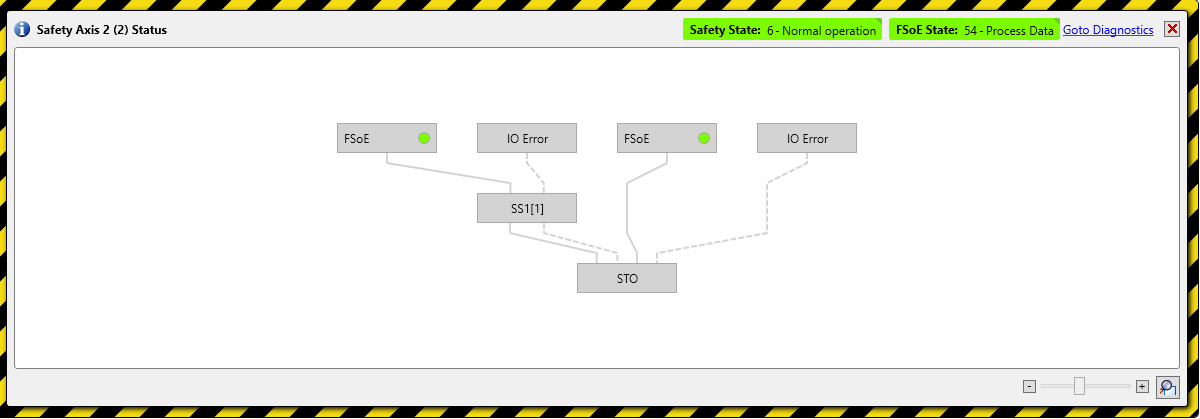
ESTOP Present

1. Next enable the drive from the .kas program

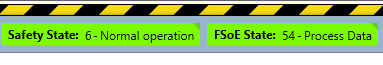
In Workbench embedded in the IDE (or WorkBench directly), double click on the Safety (Axis 1 and Axis 2). Axis1 will look like this:  


or directly in WorkBench - Axis 1: 

Axis 2: 

Bottom tray bar: 

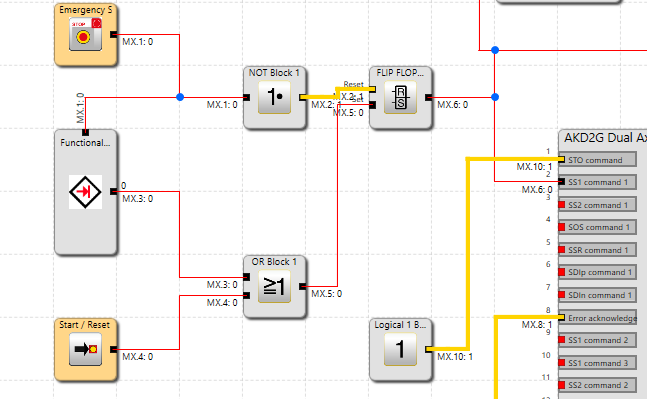
Note the Status information:



Then changing the state of the ESTOP button, triggers: (1) FSOE monitoring SS1, which triggers (2) SS1, which when complete triggers (3) the STO and the drive disabled

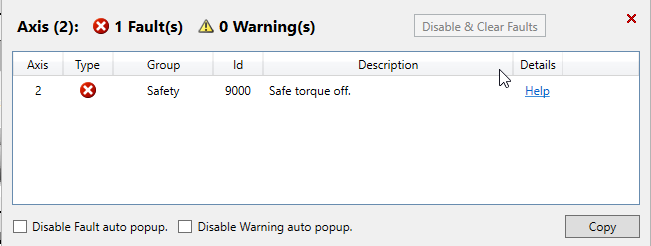






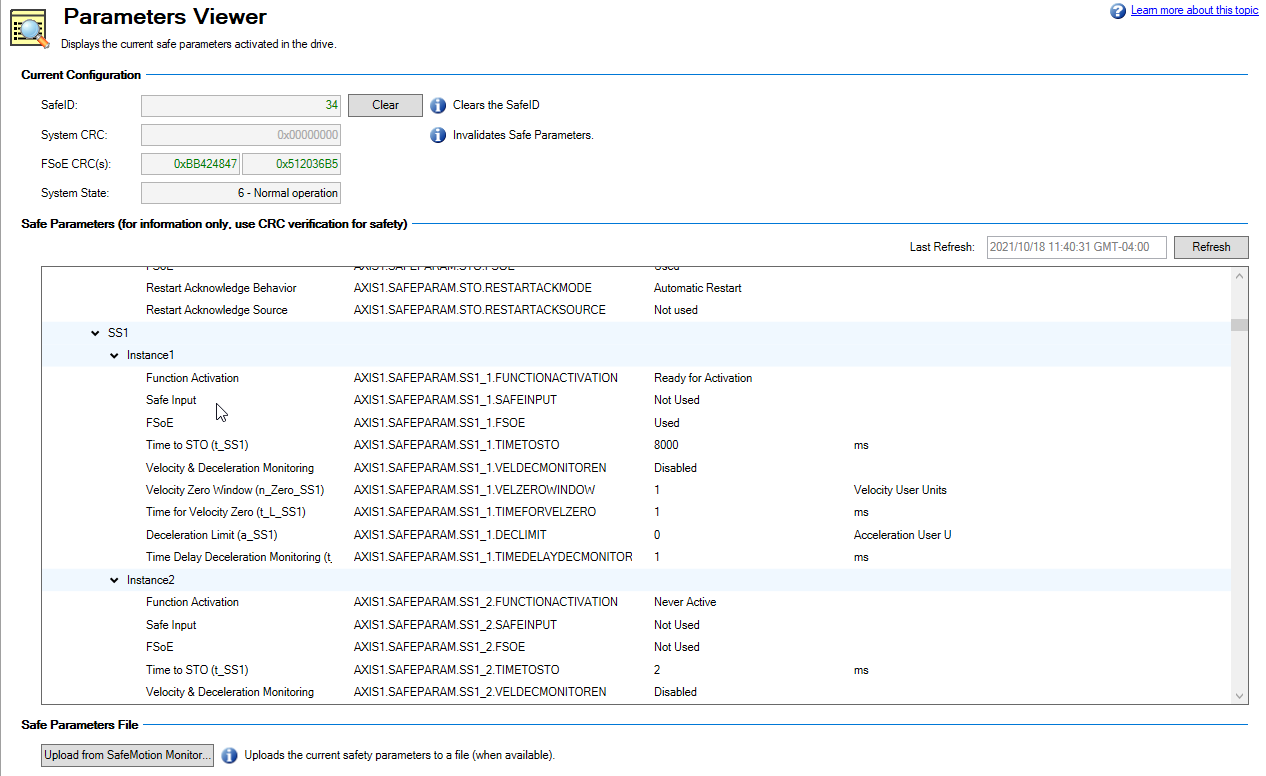
ESTOP Removed

Resulting Drive Fault:



6. Note the WorkBench Parameter viewer also shows the Safety parameters setup by the SafeMotion2 SW.

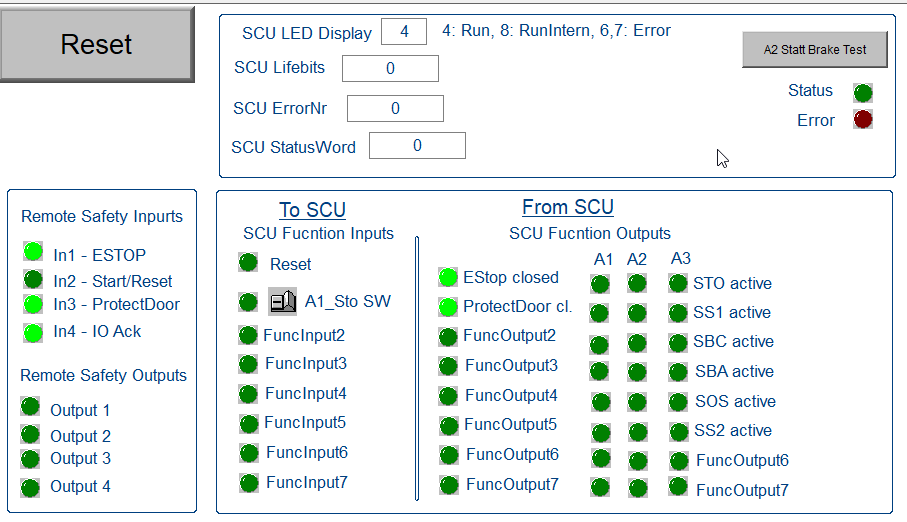
FSOE CRC – First number does not change after new params sent from SCU, 2nd one changed from0x512036B5 to 0x241594EC



7. KAS IDE Controls

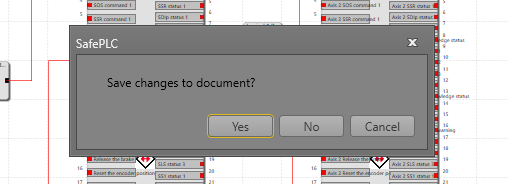


IDE Control Panel screen displays safety operation status



**Close Connection**

To close the connection with the SCU, click on . Then clicking yes will save the SafePLC2 project when later the complete IDE project is saved.

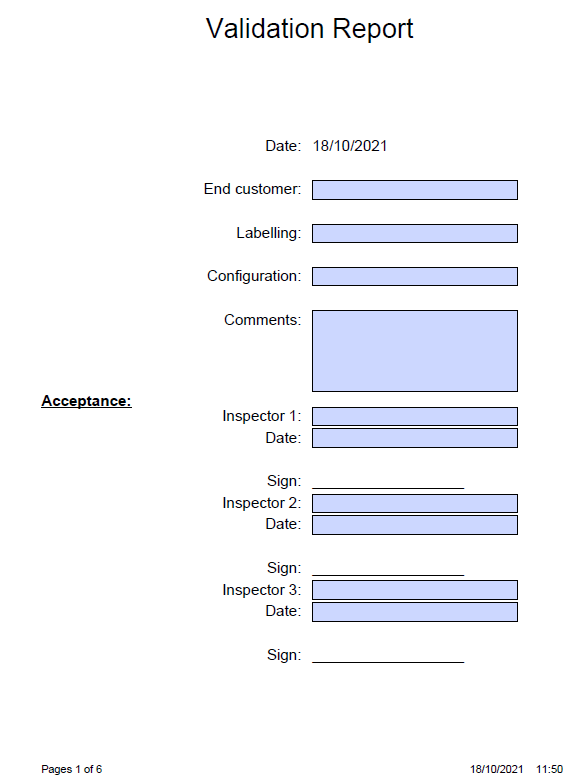
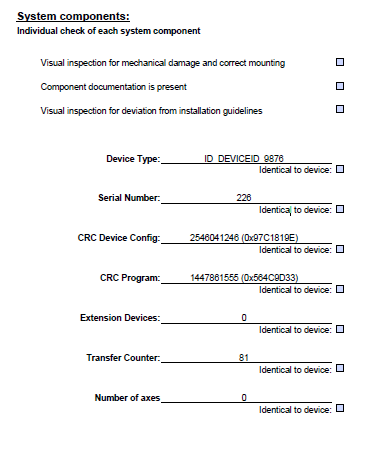


**Verification**

To ensure the implemented safety functions, the user must check and document the parameters and links after commissioning and parameterization. This is supported by the SafePLC2 parameterization software. (see additionally programming manual "HB-37500-820-10-xxF-EN Programming manual SafePLC2"). 5.7.1.



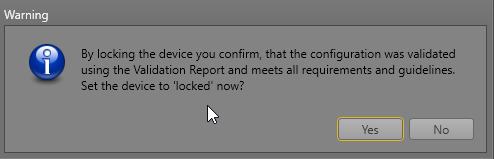
**Generate Validation Report**

General information about the application is given on the first two pages. After that all the functions used are printed with their parameters as an itemization of the safety-related test. Note: After the transfer of the configuration data and the programmed data to the SCU assembly, the status LED flashes GREEN. This indicates that the configuration data have not yet been validated.

**Lock Configuration**

At the conclusion of the verification process clicking on lock the configuration and SCU Run LED turns to solid GREEN.

****

**Safety Brake (*to be added*)**

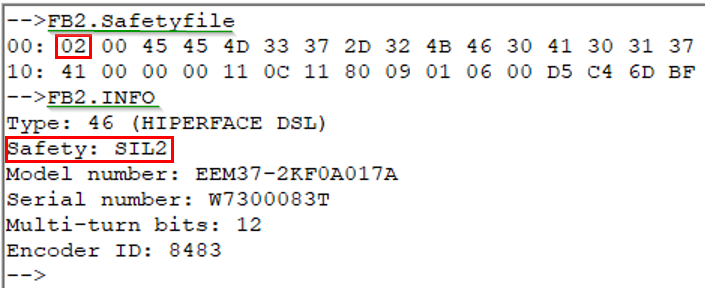
**Parameter setup**

**Application Screen shots**

**Safety Feedback**

A certified safe feedback device is needed to implement many safety functions. Presently Kollmorgen motors supports Hiperface DSL model EEM37-2KFOA017A, Using the WorkBench terminal the following commands provide information on a safety encoder:

First line must start with an “02” and Safety: SIL2 included

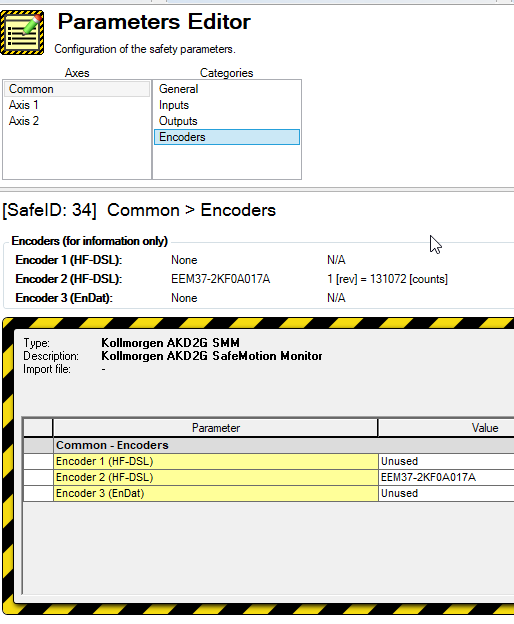


Safety rated

Safety rated

**Setup**

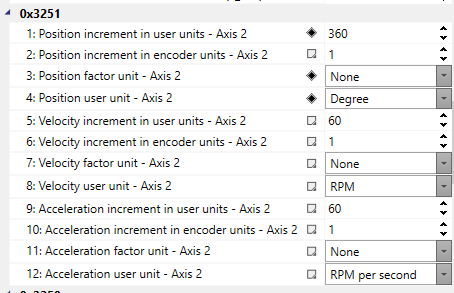
Define Encoders: Encoder 1 – for Axis 1 , Encoder 2 – for Axis 2

****

**Encoder FSOE Setup**

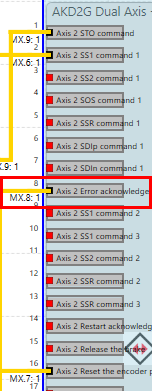
****

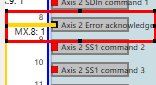
Encoderunits are set with FSOE Objects 0x3151/0x3251

****

The axis must be “homed” at the beginning of operation. This is done byAKD2G Homing functionality or the Motion controller (Kollmorgen PCMM or third party product) The value entered here “Absolute Position Reset Value” is the same value that is assigned to AXISx.PL.FB when homing is complete.

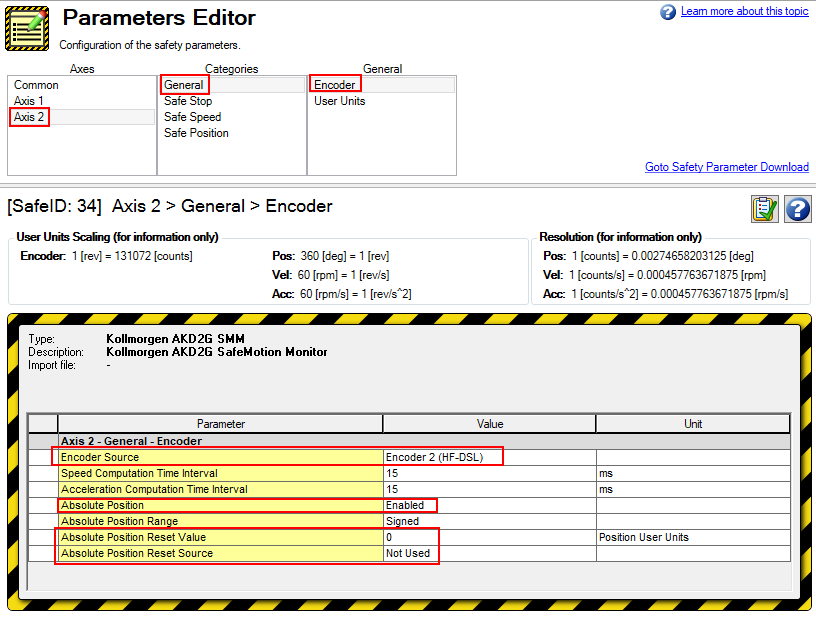
Reset Position



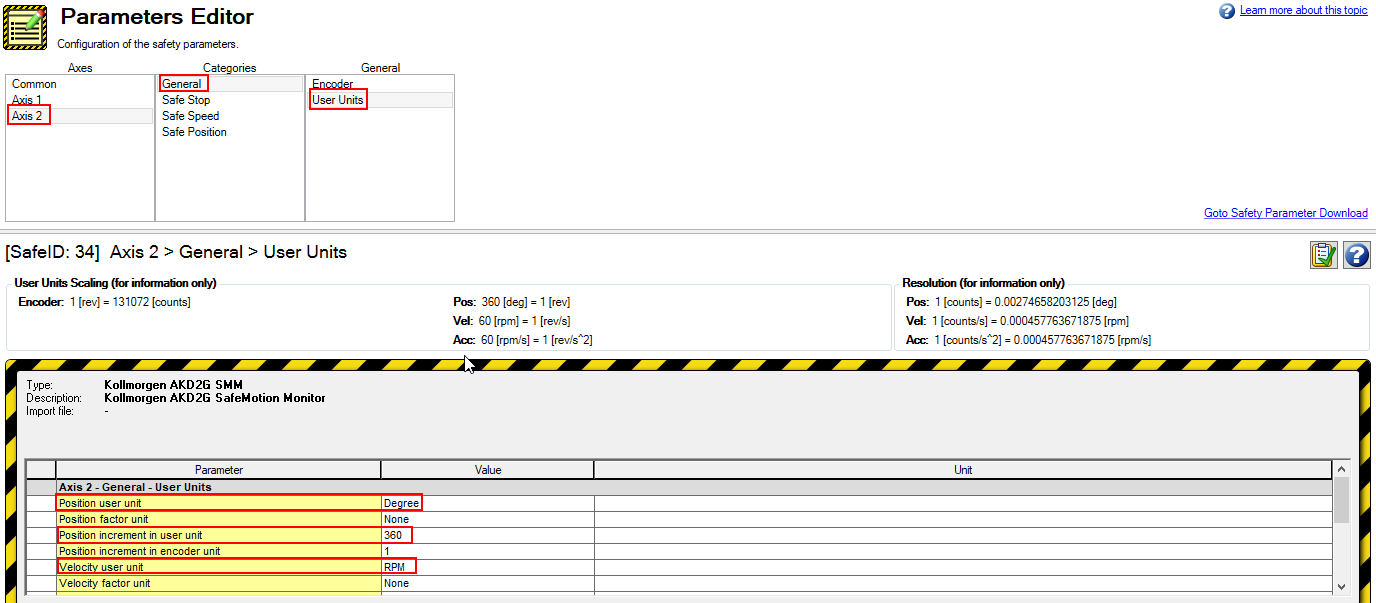


The object:

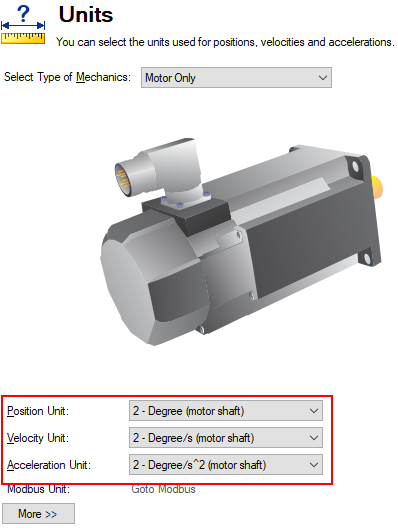
****

****

**User units are next setup**

****

**Typically, these would be the same as set here:**

****