# General System Structure

An assembly line comprises one or more stations where the produced parts resides while they are operated on or where they just temporarily are stored. The parts will be moved from station to station when each operation is completed. The transportation can be done using several different types of equipment such as robots, roller beds, lift-and-carry and other types of conveyors. The flow of parts between stations can be strictly consecutive or split into more flows and again merged into one flow. At the first station and commonly also in other downstream stations, parts will be added and at the end of the line be output as a subassembly or a finished product. Typically, many subassemblies will be fed into one main flow.

## Hardware design

One common part of a line is power supply such as electrical power and sometimes also compressed air and hydraulics. In some cases, you will also need cooling water.

Another important part is the control system, which usually consists of one or more PLCs collecting sensor data and issuing commands to motors, valves and other actuators. There is usually also a Manufacturing Execution System (MES) and a Supervisory Control and Data Acquisition (SCADA) system the feeds the PLCs with production control data and retrieves information from the line about the produced parts regarding quantity and quality and the equipment regarding disturbances, faults and alarms.

Another important task for the control system is to secure the safety aspects of the production line. In the past this was handled by electromechanical components. Today it is preferably handled by safety PLCs. This is a great advantage since different types of lines no longer need to be built with safety relays configured with electrical connections. Instead you will use safety PLCs configured by software.

Now you can design a set of standard power distribution panels and control panels. The power distribution panels will have standard sets of circuit breakers that will fit many types of lines. The panels will have a set of connection points (connectors) commonly required by your type of production lines.

Only a few types of control panels will be required for different sizes of lines. The DC power supplies are preferably included in the control panels. The circuit breakers should be monitored by the PLC.

The power distribution and control panels can be used as components when you create the electrical design.

Another great thing is that, today you can buy most of the additional electrical component with an IP54 or better encapsulation, connected with standard sensor, actuator, power and control bus cables with connectors. This will reduce the electrical design to connecting components with standard cables. No detailed circuit diagrams are required; only single line connection diagrams showing cables with connectors or standard connection to screw terminals (such as L1, L2, L3, N, PE for power). This will simplify and reduce the electrical documentation a lot.

### Safety

Many assembly lines have dangerous movements and needs to be monitored regarding safety and perhaps enclosed by a perimeter guarding (a fence). They can consist of only one station, but usually they comprise many stations. All dangerous movements must be monitored, and you must be able to shut off the power in a safe way when a fault is detected or if an emergency stop button is pressed. If a stop is activated, you do not want to stop more equipment than necessary, so it is convenient to divide the line into groups of stations. We can call these zones.

Many components like valve terminals and motor controllers can be powered off individually so they are easy to group and control in respect of zones. There are however some parts of the safety control system that will belong to a zone, such as emergency stop buttons, gate switches on zone access gates and light curtains between zones. You may also need a main shut off valve for the zone to fulfill the redundancy requirements for safety.

### Line and Zones

All zone related components shall be documented on a zone connection diagram together with the connections to all stations belonging to the zone. Note, the connections within the stations will not be included. The stations will be shown as a box on this type of drawing. A drawing for the line will also be required showing connections between zones and external connections to the line.

### Stations

The next level will be stations. Each station will comprise a limited number of components. Each station can have its own set of connection diagrams showing the connections between components within the station and to the zone level. Some stations only have a few clamping devises and sensors and only need a DC power and bus connection from/to the zone level.

## Components

Most components also need documentation. The supplier of components you have bought offers documentation of their products while control panels you have designed yourself need to be documented by you. This level of documentation need to be done in much more detail in a traditional way.

## Naming

The goal is also to use international standards as far as possible. We will use ISO/IEC 81346 Industrial systems, installations and equipment and industrial products – Structuring principles and reference designations – Part1: Basic rules and Part 2: Classification of objects and codes for classes.

Part 1 tells you to use a hierarchical designation i.e. one part belonging to a part will be prefixed by the designation of the part it belongs to. It also defines different aspects for a designation such as functional (=), location (+) and product (-). Each part of the designation is prefixed by a character, defining the aspect. If we define the line as a function, we can call it =L1. If we look at station S10 as a location within the line function, we can give it the designation =L1+S10. If we have a component (product) C1 in station +S10 we will give it the designation =L1+S10-C1. Part 1 of the standard give us a great deal of freedom how to build each level of the designation. It can be a number, some upper-case letters or some upper-case letters followed by a number. A number can however not be followed by letters. Designations built up from several levels having the same aspect only need one initial aspect character if each level comprises letters followed by a number. E.g. A sensor B1 on a safety gate unit -RUB1 in station +S010 will be   
+S010-RUB1B1.

Part 2 is a bit more specific. It defines three letter codes for object classes (components). You can use one letter, two letters or all three letters, as you see fit, followed by a number to differentiate between more objects of the same class on the same level. You may use any number, but sequential numbers are preferred.

### Lines

A list of unique names shall be created for each type of lines. The name should preferably be two or three uppercase letters followed by a number indicating the generation of line (product year model). E.g. =AAA1.

### Stations

Stations should be named +S and a number. You need to decide the format. A three-digit number is recommended. Depending on type of line a numbering scheme could be to use increments of 10 for the main flow and use the intermediate numbers for infeed stations.

### Zones

Zones should be named +Z and a number. It is recommended to use the same number as the first station in the zone.

Note! A line will comprise stations and zones. Zones will be considered a group of stations and not a hierarchy level between line and stations.

### Components

Components shall be named in accordance with ISO/IEC 81346 Part 2.

E.g. UCA1 is a control panel

Note that components can be defined on different levels in the hierarchy. Control panels can exist on the line level =AAA-UAC1 (power distribution panel), on zone level =AAA1+Z010-UCA1 (safety control panel) and station level =AAA1+S010-UCA1 (safety gate control panel, manual operator station).

### Software

See Software Design, Naming.

## Software design

The basic idea is to structure the control system as an image of the hardware it controls.

We will also adopt a type of Object Based Programming which is a simple form of Object-Oriented-Programming (OOP) i.e. without Inheritance, which so far is rarely supported by PLC software developments systems. This means that we will try to represent each physical object and logical function with a function block (abstraction and encapsulation). We can control the object via the FB inputs, and we will be able to read the objects status from the FB outputs. The function block will hold data and code for the object, and it will also monitor the object and set alarms if a fault is detected.

### Program Structure

In the Siemens PLC you have organizational blocks (OB) that is run by the system. OB1 is the cyclic module that runs continuously and from where you call all other standard program modules. There is also a Safety Runtime Environment with an organizational block OBnnn. This is run at fixed time intervals by the system (default every 100ms) and will interrupt other tasks. You call the safety programs for respective zone from this OB.

There are some functions common to the line, such as Power Distribution and Main control panel interface, Portable HMI interface (if used) and control bus (ProfNet) supervision. Three function blocks will be created in the root folder for these functions.

To mimic the hardware, we will create one program for each main part of the line such as zones and stations. Unfortunately, Siemens do not follow the IEC 61131-3 standard, so Program is not implemented. Instead we will use single instance function blocks for this purpose.

|  |
| --- |
| Function blocks are also not correctly implemented. In the IEC standard, an instance of a function block consists of both data and code. In the Siemens implementation, the instance is represented only by the data part (Instance Data Block) of the function block, which have a lot of bad consequences. |
| * The code block needs to be visible in the Project tree, so that you can open it for editing and monitoring, instead of being listed under system blocks. (Code block should be reached via the instance data block instead.) * The version handling of library modules has a naming problem. When you import an FB from the project library there will be a link to this in the project tree. The name of this link will by default get the same name as the FB in the library, but it is not bound to the name in the library so it can be named anything you like. If you have used two versions of the same FB in your project, the links need to have different names, the second with \_1 appended to it as default. This would not have been a problem if this was forced and hidden in the system block folder. * When you open an instance of a function block for editing or monitoring the name on the tab shows the FB name instead of the instance name. * You can only open and monitor one instance of a function block at a time. * The IDB is associated with the FB instead of the other way around. This makes it difficult for the system to keep track of the dependencies. If you define an FB and call more than one instance of that in the definition of another FB you will run into a problem if you call more than one instance of the outer FB in a third FB. If you open and monitor the outer FB and then open and monitor the inner FB, the system will lose track of the instances, so this will not be correctly monitored and the system will not tell you it is wrong. * The system needs to have a “Call environment” so that the programmer can help the system to select the instance you want to monitor. This is the only way to fix the previous problem. A proper implementation would not require this. * You cannot change the value of a variable when monitoring an FB (only possible in an IDB). * If you change the code and want to monitor, it will often loose its association to the IDB, so you need to close the editor and reopen, or fix it in “Call environment”, before you can monitor. |
| If Siemens had made a proper implementation of function blocks none of above problems would occur. The instance data block would have had a link to the FB (code block) so the code could be displayed when opening the IDB (same view as when you open the FB in present implementation) but always show the proper data for that instance. |

We will create a folder structure to hold the programs, starting with one folder for each zone (group of stations).

In the zone folder we will create one single instance function block for the zone standard program, one single instance function block for the safety program and one folder for each station in that zone.

In the station folder we will create one single instance function block for the station. We will also create one SFC (GRAPH) function block for the automatic program for the station.

All single instance function blocks except the SFCs will be called from OB1. The SFC function block will be called from the station function block. The safety function block will be called from the Safety OB. All FBs will also have a companion Instance Data Block in the same folder as its FB.

There are also some other general functions required to make the system complete. We will create four folders for these. These will hold constants, HMI data exchange data blocks, library modules and MES data exchange data blocks.

### Naming

For basic naming rules see chapter Naming. According to the standard IEC 61131 part 3, special characters are not allowed in PLC identifiers. This means that the aspect characters used in the hardware designators should not be used in the PLC. The Siemens editor allows this, but since we have committed to follow international standard as far as possible, we will not use these characters in the PLC. Normally we can drop these characters without any problems even if the PLC names will not fully match the hardware names.

#### PLC

The PLC is always named KEB1. The full name will be =AAA1-UCA2KEB1 since it is placed in the main control panel -UAC2. If there are more PLCs, they will be prefixed with a different control panel name.

Note! To support the principles of Object Based Programming, the instance data block will have a name corresponding to the object it represents and the function (code) block name will have an extension to that name. The Siemens default naming convention is unfortunately the opposite, which will cause some inconvenience when naming instance data blocks.

#### Line Function Blocks

The function blocks and instance data blocks for common line functions shall be named

|  |  |  |
| --- | --- | --- |
| FB | IDB | Description |
| GENprg | GEN | General functions for the line |
| HMIprg | HMI | Portable HMI interface |
| PNprg | PN | ProfNet communication supervision |

#### Safety program

The main safety program and instance data block shall be named:

|  |  |  |
| --- | --- | --- |
| FB | IDB | Description |
| Sprg | S | Main safety program |

#### Zone programs

Zone program shall be placed in folders named \_Zzzz where zzz is the zone number. You may add some descriptive text to the folder name. The initial underscore is used to make zone program folders be sorted first. Zone programs and instance data blocks shall be named according to below example:

|  |  |  |
| --- | --- | --- |
| FB | IDB | Description |
| Zzzzprg | Zzzz | Zone standard program |
| ZzzzSprg | ZzzzS | Zone safety program |

#### Station programs

Station programs shall be placed in folders named Ssss where sss is the station number. You may add some descriptive text to the folder name. Station programs and instance data blocks shall be named according to below example:

|  |  |  |
| --- | --- | --- |
| FB | IDB | Description |
| Ssssprg | Ssss | Station program |
| Ssssseq | Ssssau | Automatic sequence control program |

#### Other Folders

The following folders are required to hold additional general function blocks and data

|  |  |
| --- | --- |
| Folder name | Description |
| Constants | Constant defining common names for controllable objects in PLC and HMIs |
| HMI | PLC/HMI data exchange data blocks |
| Lib | Tree structure for local library modules including links to Project library |
| MES | MES data exchange data blocks for stations and robots |

### Safety Programs

If a person presses an emergence stop button placed close to the border between two zones, you do not know in which zone the person has detected a dangerous situation. A common solution is to stop the zone to which the button belongs and adjacent zones.

To handle this, we will have a two-level safety control system; line and zone. The zone programs will have a few inputs and outputs which will be used to interlock the zone programs with each other when called in the line program. The ES\_Ok output from one zone program will be connected the ES\_Next input on the program for previous zone and to ES\_Prev on the program for next zone.