

IEC 61499 Execution Model

Definitions

The IEC 61499 standard defines a distributed model for splitting different parts of an industrial automation process and complex machinery control into functional modules called function blocks. These function blocks can be distributed and interconnected across multiple controllers.

System: A collection of devices interconnected and communicating with each other by means of a communication network consisting of segments and links.

Device: An independent physical entity capable of performing one or more specified functions in a particular context and delimited by its interfaces.

Resource: A functional unit having independent control of its operation, and which provides various services to applications including scheduling and execution of algorithms.

Application: A software functional unit that is specific to the solution of a problem in industrial-process measurement and control. An application may be distributed among devices and may communicate with other applications.

Function block: A software functional unit that is the smallest element of a distributed control system. It utilizes an execution control chart (ECC) state machine to control the execution of its algorithms.

Overview

An Execution Model represents parts included in a function block execution mechanism. Figure 1 shows these parts of a execution mechanism. Each function block execution follows a specific mechanism.

A function block is a functional unit of software comprising an individual instance or copy within a resource. The algorithms contained within a function block are hidden from the outside of the function block and are scheduled according to the Execution Control Chart state machine (ECC).

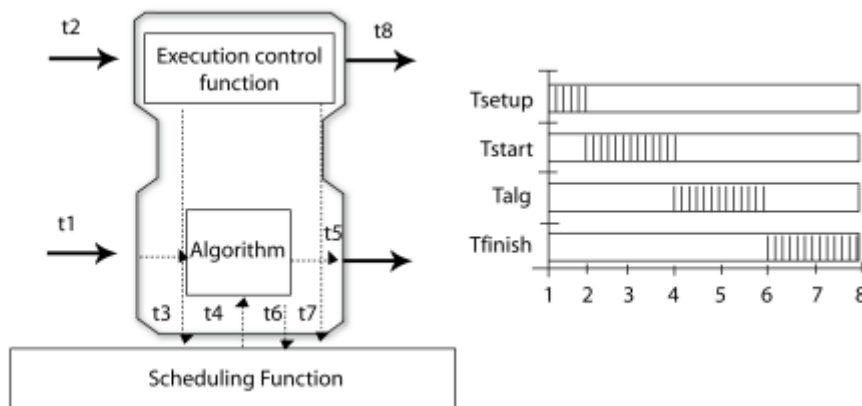


Figure 1: IEC 61499 Execution Model

Event input (t2) and event output (t8) are used to synchronize function blocks within an application and to schedule the algorithms within the function block.

Data input (t1) and data output (t5) are the interface with the external of the function block since internal data is hidden. The data may be part of the algorithms and may also be state information for the ECC.

Function blocks can be created by defining their ECC, input and output signals, and programming their algorithms. These function blocks are called Basic function block. The ECC is a state machine processing events and scheduling algorithms. The ECC defines the behavior of the function block upon receiving events. The algorithms operate on internal variable values, input values, and output values. Each Basic function block can run on any resource.

In a Basic function block, timing is important. Data inputs are received first (t1), then the event inputs (t2) are either received at the same time or next. When event inputs trigger the ECC execution the function block must have stable data inputs. Otherwise, erroneous behavior occurs. At t3, the resource schedules the execution of the algorithms related to the event. At t4, the algorithms start running and process the input data signals. Upon completion, the algorithm outputs the data signal (t5), then the resource is notified (t6) and the ECC takes over at t7. The ECC outputs event signals (t8) and the execution of the function block is completed until the reception of a new event.

When function block algorithms and the control of their execution are expressed entirely in terms of interconnected function blocks, these are called Composite function block. These are created by interconnecting existing Basic and Composite function blocks. No ECC and algorithms are created. A Composite function block may run on any resource. Each function block within the Composite function block runs on the same resource as the Composite function block. Function blocks within of a composite function block cannot be individually distributed across multiple resources.

The execution of a Composite function block differs from a Basic function block; A Composite function block does not have an ECC or algorithms. At some point when breaking down a Composite function block, each function block contained within is a Basic function block and executes as a standard Basic function block. The overall timing delay depends on the execution time of each of these internal function blocks.

In **ISaGRAF**, you can create applications using custom function blocks or function blocks from libraries. You can also create Basic function blocks.

The **ISaGRAF** ECC is a state machine built with the SFC editor. However, the ECC has a different execution behavior from IEC 61131 SFC. All STEPS execute in one virtual machine cycle until a FALSE transition occurs. Algorithms may use any of the IEC61131-3 programming languages as well as the flow chart language provided with the **ISaGRAF** toolset. The available programming languages are the following: Sequential Flow Chart (SFC), Function Block Diagram (FBD), Ladder (LD), Instruction List (IL), and Structured Text (ST).

Composite function blocks can also be created with the **ISaGRAF** toolset using the Composite FB editor. **ISaGRAF** enables the creation of composite function blocks by adding any available Basic and Composite function block to the function block network.

The creation of a new function block (basic or composite) makes it available for use in any application and may be configured to run on any Resource or Device making up the system.

Figure 2 shows the execution mechanism used by an **ISaGRAF** control engine. After reading IO inputs and bound variables, the logic is processed. Then bound variables, retain variables and output IO are written. This cycle starts over once the delay is over.

1. Scan input devices
2. Consume bound variable (t1, t2)
3. Execute TIC code (t3, t4, t6, t7)
4. Production bound variables (t5, t8)
5. Update Output devices
6. Save Retained Values
7. Sleep until next cycle

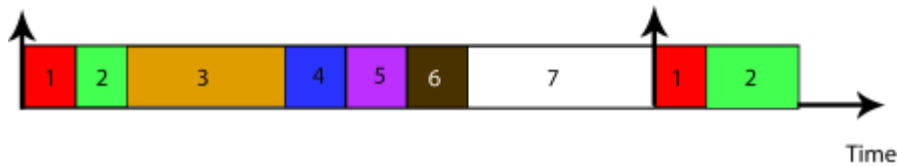


Figure 2: ISaGRAF Basic Function Block Execution Model

The **ISaGRAF** Basic function block execution model operates as defined in the IEC 61499 standard.

When the **ISaGRAF** resource consumes bound data, the resource reads the event and data input values (WITH qualifier). This is IEC 61499 t1 and t2 time (figure 1). Then the resource starts the execution of the function block ECC (t3 - figure 1) as it starts executing the TIC code. The algorithm is then running (t4 - figure 1) and upon completion (t5 - figure 1), the algorithm writes the output data values. The resource (t6 - figure 1) returns execution to the ECC (7). This one writes output event values. Then the resource writes bound data values (WITH qualifier). This action generates the event and data signals simultaneously.

References

International Electrotechnical Commission: Function Blocks Part 1 - Architecture (61499-1 © CEI:200X).

ICS Triplex ISaGRAF Inc.: ISaGRAF User's Guide. November 2005.