

OMAC Packaging Working Group

How to implement PackML / TR88 ?

Sept. 28, 2015

Carl Bostrom, Principal Engineer

Bosch Rexroth

PackML Template Options

GAT PackML

- Special case of our Generic Application Template (“GAT”)
- Wizard-driven
- Automatic code generation
- Modular
- Extremely powerful, but structured text only

OMAC Implementation

- RIL_PackML_Toolkit compiled library
- Closely follows OMAC Implementation Guide
- ISA88 Hierarchy
- Modular, self-registering
- Self-registering event handling
- Supports LD, FBD, ST, etc.

PackML Toolkit: OMAC Implementation Guide

The screenshot shows a Library Manager window for 'MLC_L65: Logic: Application'. The tree view on the left shows the following structure:

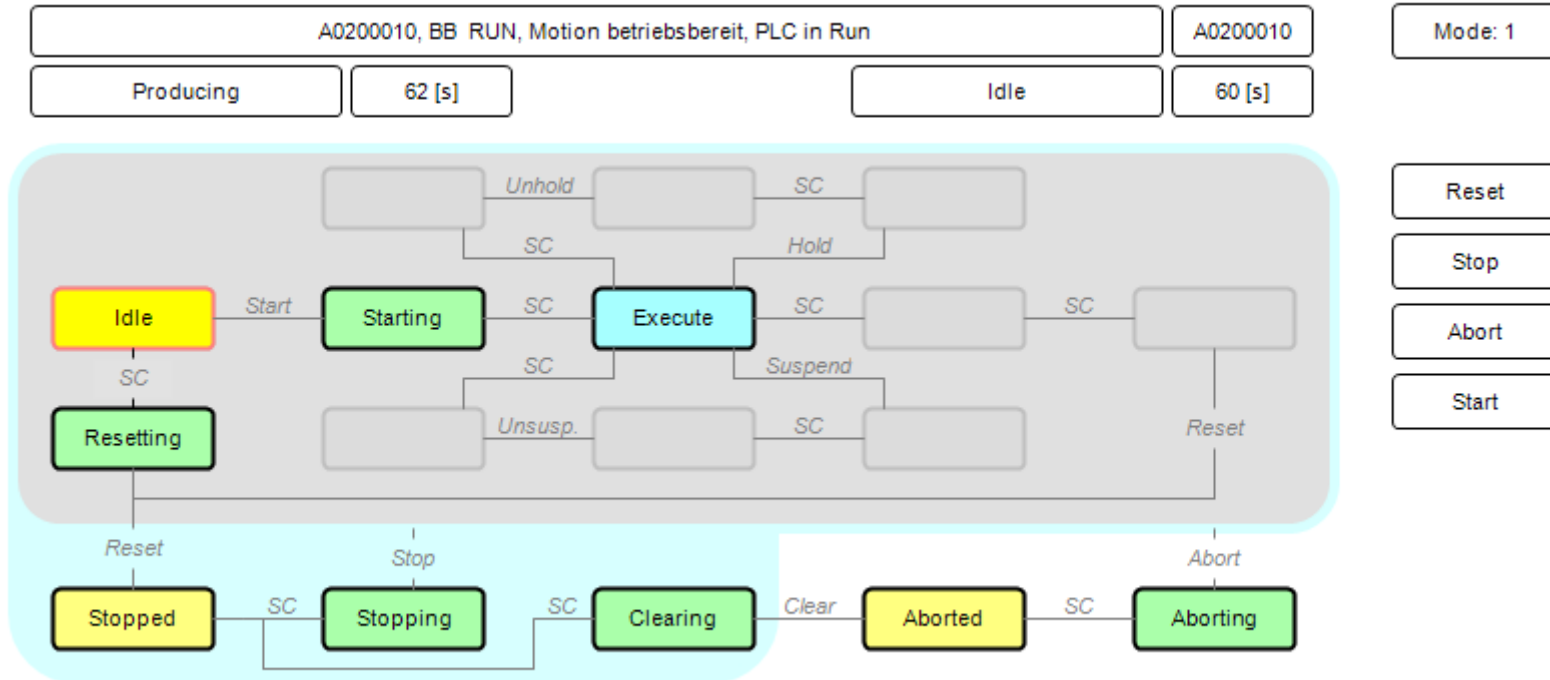
- ril_packml_toolkit
 - Data Types
 - POUs
 - EventHandling
 - IL_CM_Event_Type01
 - IL_CM_Event_Type02
 - IL_EventHistory_Type01
 - IL_EventManager_Type02
 - IL_EventSummation_Type01
 - Internal
 - StateCommands
 - IL_PackML_CommandClient_Type01
 - IL_PackML_CommandManager_Type01
 - StateMachine
 - IL_PackML_StateTimer_Type02
 - IL_PackML_Type02
 - Project Information
 - PackML_StateMachine_LG
 - PackML_StateMachine_SM

The detailed view of **IL_PackML_Type02** is as follows:

| Property | Type | Value |
|--------------------|------------------------|-------------------------------|
| Enable | BOOL | Active |
| Mode | DINT | State |
| Reset | BOOL | DINT StateCurrent |
| Start | BOOL | STRING StateCurrent_Name |
| Stop | BOOL | BOOL StateChangeInProgress |
| Hold | BOOL | DINT UnitMode |
| Suspend | BOOL | DINT UnitModeCurrent |
| Abort | BOOL | STRING UnitModeCurrent_Name |
| Clear | BOOL | BOOL UnitModeChangeInProgress |
| StateComplete | BOOL | UDINT DisabledStates |
| audModeTransitions | ARRAY[0..31] OF UDINT | BOOL Error |
| audDisableStates | ARRAY[0..31] OF UDINT | ERROR_CODE ErrorID |
| astrModeNames | ARRAY[0..31] OF STRING | ERROR_STRUCT ErrorIdent |
| astrStateNames | ARRAY[0..17] OF STRING | |

Library functions have OMAC “look and feel.”

RIL_PackML_Toolkit: State Handling



- Full support for all PackML states.
- States may be enabled or disabled by configuring state handler.

RIL_PackML_Toolkit: Mode Management

- Mode management is handled by IL_PackML_Type02
- A mode transition is allowed in a given state provided the bit associated to this state is true for both `audModeTransitions[current mode]` and `audModeTransitions[target mode]`.

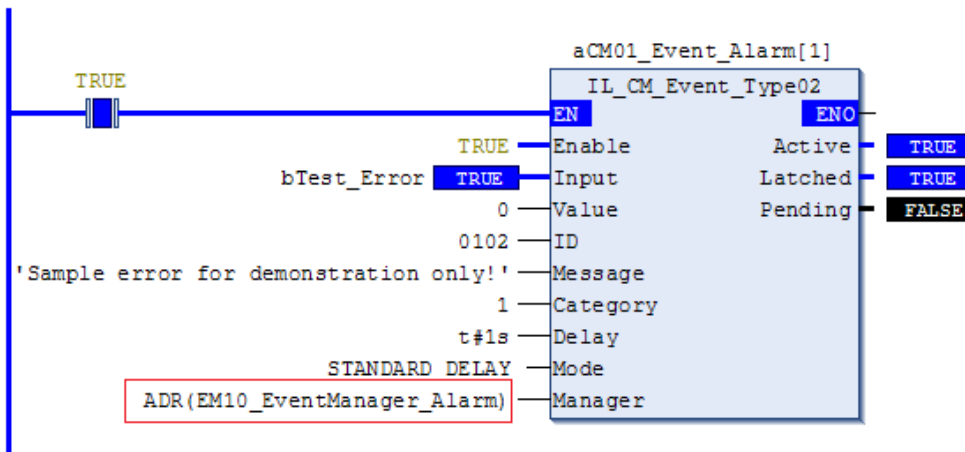
Example: If the machine is currently in mode 1, state 9 (aborted), then we may change to mode 3 provided:

`audModeTransitions[1] = bxxxx xx1x xxxx xxxx,`

`audModeTransitions[3] = bxxxx xx1x xxxx xxxx`

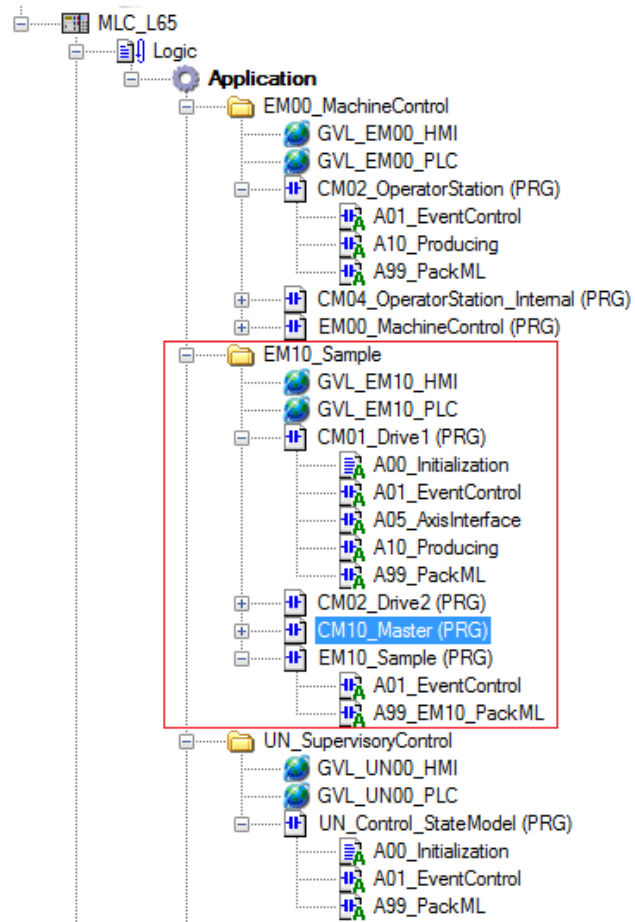
RIL_PackML_Toolkit: Event Handling

| Visu_AlarmHistory[MLC_L65: Logic: Application] | | | | | |
|--|-----|---|-----------------------|-----------------------|----------|
| | ID | Message | TimeEvent | TimeAck | Category |
| 0 | 102 | EM10:Sample error for demonstration only! | DT#2013-9-19-17:24:50 | DT#2013-9-19-17:25:18 | 1 |



- Events automatically register themselves with their respective manager! Library handles Alarm and AlarmHistory structures.

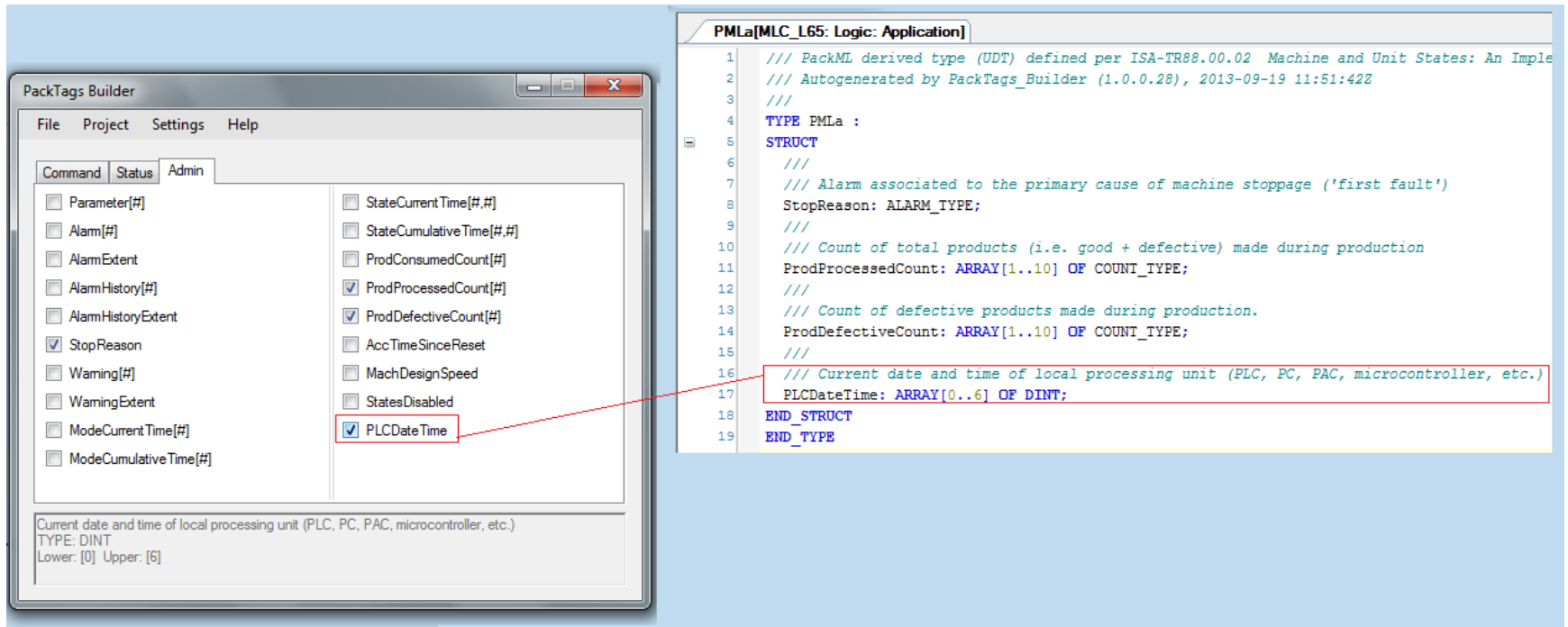
ISA88 Physical Hierarchy



ISA88 modularity is strictly followed:

- Equipment modules manage their own global/interface variables.
- Equipment modules **register themselves** with the unit machine. No code is required at the global level to add an EM to a project!

PackTags: PackTags_Builder 2.0



- Free software tool to facilitate implementation of PackTags. Revised per ISA-TR88.00.02-2015!

Implementation tips

- The interface is the priority. PackTags, especially the minimal set required for compliance, can be implemented easily.
- Details are critical. Data are only useful if they can be interpreted correctly.
- Leverage expertise from the technology provider.

Thanks for your attention!

