THE CONFIGURABLE SOFTWARE INTERLOCK SYSTEM FOR HLS-II*

Y. Song, K. Xuan, G. Liu†, NSRL, USTC, Hefei, Anhui 230029, China

Abstract

The interlock system is an essential component for an accelerator facility. A configurable software interlock system(SIS) is designed for Hefei Light Source II (HLS-II), which complements the hardware interlock system to ensure equipment and operators' safety. The system is developed using Python under the EPICS framework with the method of separating the configuration file from the interlock program. The interlock logic is completely determined by the configuration file and its nested tree structure is easy to expand. The test results indicate that the new software interlock system is reliable, flexible and convenient to operate. This paper will describe the design and the construction of HLS-II SIS.

INTRODUCTION

The Hefei Light Source (HLS) is the first dedicated synchrotron radiation facility in China. HLS was upgraded from 2010 to 2015 to improve its performance. The upgraded light source is renamed as Hefei Light Source II (HLS-II) [1]. The HLS-II was fully open to users in January 2016. The control system of the HLS-II is a distributed system based on EPICS. The interlock system is one of the critical components of the control system. The main purpose of an interlock system is to protect the machine and enhance safety during the accelerator operation.

In general, the interlock system consist of two parts: the hardware interlock system (HIS) and the software interlock system (SIS). The HIS usually uses PLC and FPGA technologies [2, 3]. It is designed to response within a few microseconds. The SIS is not a hard real-time system and the response time amounts to several seconds [4, 5]. So it's used as the necessary complement of the hardware interlock system when there is no strict requirement for response time, but easier to implement complex logic among different subsystems.

The original SIS of HLS-II was developed by the State Notation Language (SNL) running in the EPICS IOC. The code has to be rewrote and compiled after the interlock logic is updated, which is not convenient enough in the operation and maintenance. The new HLS-II SIS is developed using Python under the EPICS framework. The system adopts the method of separating the configuration file from the interlock program, the interlock logic can be updated by modifying the configuration file simply. The interlock logic is completely determined by the JSON configuration file and its nested tree structure is easy to expand. The system was put into operation in February 2017. When a fault occurs, the interlock actions are triggered, the alarm information is sent to operators and the interlock event is logged.

* Work supported by National Natural Science Foundation of China (No.11375186)

† Corresponding author, gfliu@ustc.edu.cn

HLS-II SIS ARCHITECTURE

Figure 1 shows the overall structure the HLS-II SIS, which is divided into two layers, the SIS core and pyEPICS. The SIS core is composed of a list of interlock trees. Each tree, which is associated with a subsystem, has a similar structure and is an independent logical unit. There are two types of nodes in the tree structure: the leaf node and trunk node. The structure of the interlock tree is determined by an external configuration file (config.json).

The SIS core is connected with IOCs through the PyEPICS extension interface [6]. Through this interface, the SIS core reads / writes the process variable (PV) to monitor / control the devices. The SIS program processes each interlock tree recursively to achieve interlock protection.

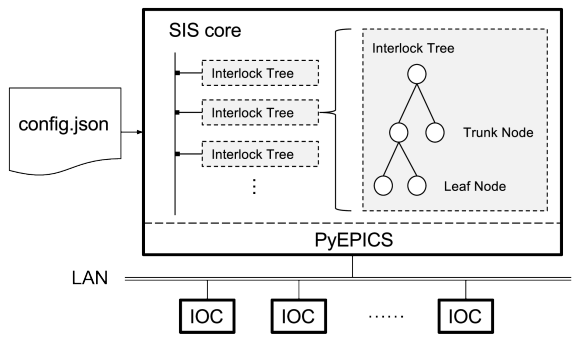

Figure 1: Architecture of the HLS-II SIS.

Leaf Node

The leaf node is the lowest node in the interlock tree, and is directly related to the controlled device. The leaf node detects the state of the controlled device and provides an interlock signal for the whole system. The state will be tested based on the *compare_operator* and *design_value* pre-defined in the leaf node. If the device is in a normal state, the test is successful and he state value of the leaf node is FALSE by default. However, when a fault occurs, the test is not successful, the state value of the leaf node is TRUE and an interlock event is generated in the leaf node and processed by the upper nodes. The usual operators are supported, including <, >, <=, >=, ==, ! = etc.

Trunk Node

The other nodes above the leaf nodes are all trunk nodes in the interlock tree. Each trunk node collects all the state values of its sub-nodes, and use these values to calculate the its own state value according to the pre-defined logic expression. Logic expressions support a variety of operators, including AND, OR, NOT, >, <, etc., to implement complex interlock logic. The trunk node can be configured with a action list. The actions in the action list are triggered when

the state value is True. If the parent nodes exist, the state value is also collected by the upper nodes.

Action List

The trunk node can be linked with an action list when necessary. The action list contains a series of actions which are executed in sequence when the linked trunk node value equals "True". At the present time, two types of actions, the "set" and "delay" are supported. The "set" action set the value "set_point" to the designed PV, and the "delay" action delays for "delay_time" seconds. The action type is extensible, more actions will be supported in the future.

Interlock Masking

A boolean value can be defined in every node and action unit of the interlock tree, which is called the interlock mask. The masking mechanism is designed to shield the interlock signal from the lower children branches and allow operators to ignore its interlock event. This mechanism makes it convenient for the operators to maintain the machine.

HLS-II SIS SOFTWARE DEVELOPMENT

The SIS is developed using Python. The interlock logic is defined in the JSON configuration file. JSON's nested syntax makes it easier to define interlock trees and each node is expressed as an object.

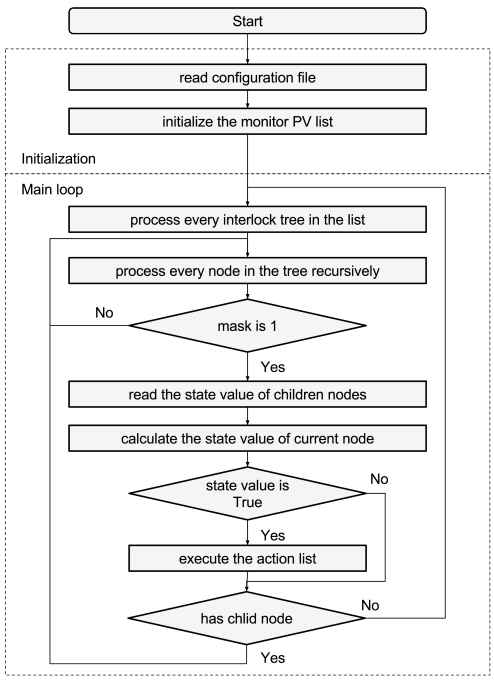

Figure 2: Flow chart of HLS-II SIS.

Figure 2 shows the flow chart of the SIS program. The program enters the initialization part first after it starts. The configuration file is read by the program and constructed to a list consisting of multiple interlock trees. At the same time, the PVs defined in the leaf nodes are parsed and the program begins to monitor the PVs' values.

Then the system enters the main loop part and processes each interlock tree one by one recursively. If the mask is 0, the program ignores the node and processes the next node. The PV value is used as an input signal iexn the leaf node. The state value of the upper node is determined by the lower nodes. If the state value is True, the action list will be executed.

HLS-II SIS TEST

In order to verify the SIS function and measure the response time, a test system is set up. Figure 3 is a logic diagram for SIS test, which contains only one interlock logic tree. The white rectangles represent the leaf nodes, and the values of PV_IN_1 - PV_IN_4 are monitored in the leaf node. The gray rectangles represent the trunk nodes and logical expressions, such as AND, fault_count >= 2, are defined in them. The dashed rectangles represent the action lists.

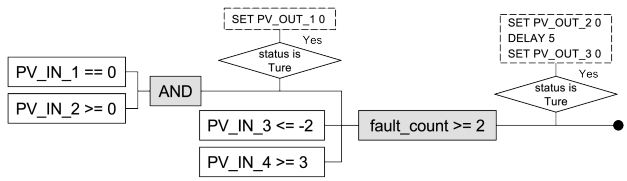

Figure 3: The logic diagram for SIS Test.

The logic in Fig. 3 can be transformed into the JSON configuration file shown in Fig. 4. The "node_type" key identifies two node types: trunk_node and leaf_node. The "mask" key for each node defines the mask value and the default value is 1. The "compare_operator" and "design_value" are set in the leaf node. Within the trunk node, the "child" key contains the list of all child nodes and the "expression" key defines the expression which is used to calculate the state values of the current node. The two action types, "set_point" and "delay_time", are defined in the action key.

Using the striptool to monitor the values of the PVs, we get the operation result of the SIS test as shown in Fig. 5. When the interlock condition is satisfied, the actions are triggered according to the predefined action list. Multiple measurement results show that the response time of the SIS is in the order of 100 ms.

OPERATION

The HLS-II SIS has been put into operation to protect the machine and enhance safety. At present, HLS-II SIS consists four parts: the vacuum interlock of the storage ring, the power supply interlock of the correction magnet, the magnet gap interlock of the insertion device and the injection interlock. Each part represents an independent interlock tree in the configuration file.

Taking the vacuum interlock as an example, 20 vacuum measuring points are distributed on the storage ring and each measurement point corresponds to a PV in EPICS control system. The SIS monitors the value of these PVs. The

ISBN 978-3-95450-182-3

```
{
   "demo":{
      "node_type":"trunk_node","mask" : 1,"expression":"fault_count>=2",
      "child":[
         {
            "node_type":"trunk_node","mask":1,"expression":"and",
            "child":[
               { "node_type":"leaf_node","mask":1,"pv_name":"PV_IN_1","compare_operator":"==","design_value":0},
               { "node_type":"leaf_node", "mask":1, "pv_name":"PV_IN_2", "compare_operator":">=", "design_value":1}
            ],
            "action_list" : [
               { "mask" : 1, "action_type":"set", "pv_name" : "PV_OUT_1", "set_point" : 0}
            ]
         },
         { "node_type":"leaf_node","mask" : 1,"pv_name":"PV_IN_3","compare_operator":"<=","design_value":-2},
         { "node_type":"leaf_node","mask" : 1,"pv_name":"PV_IN_4","compare_operator":">=","design_value":3}
      ],
      "action_list" : [
         {"mask" : 1,"action_type":"set", "pv_name" : "PV_OUT_2","set_point" : 0 },
         { "mask" : 1, "action_type":"delay", "delay_time" : 5},
         { "mask" : 1, "action_type":"set", "pv_name" : "PV_OUT_3", "set_point" : 0 }
      ]
   }
}
```


Figure 4: The configuration file of SIS test.

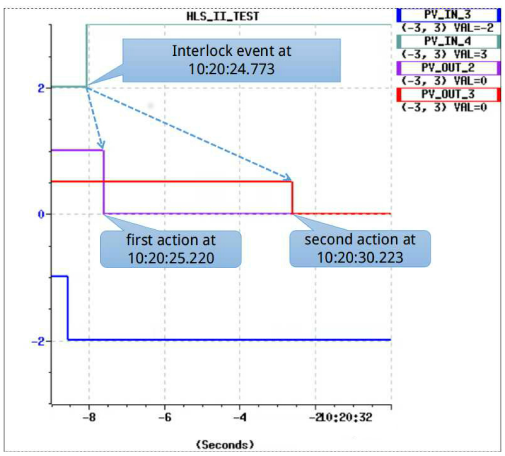

Figure 5: Test result of SIS. An interlock event occurs at 10:20:24.773, and two actions are executed in sequence at 10:20:25:220 and 10:20:30:223.

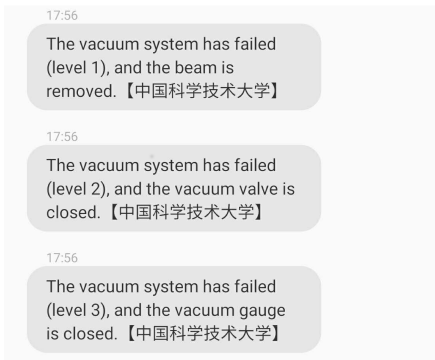

Figure 6: SMS screenshots received by operators.

vacuum pressure value of the storage ring is less than 1×10^{-6}Pa in normal state. When the fault occurs, the pressure value changes significantly. Depending on the severity level of the fault, different interlock actions are triggered. Figure 6 shows the SMS screenshots received by the operators.

SUMMARY

In order to improve the performance of HLS-II SIS, we developed a new software interlock system using the method of separating the configuration file from the interlock program. The interlock logic is completely determined by the configuration file. Multiple measurement results show that the response time of the SIS is in the order of 100 ms, which achieves the current interlock needs. The test results indicate that the new software interlock system is reliable, flexible and convenient to operate. The system was put into operation in February 2017.

ACKNOWLEDGEMENT

The authors would like to thank Prof. Jingyi Li of the National Synchrotron Radiation Laboratory for his help.

REFERENCES

[1] J.Y. Li, W. Xu, K. Xuan et al., "Operational Status of HLS-II", Proc. IPAC'16, paper THPOY028, p. 2664.

[2] W.J. Chou, D.Y. Zhou, J.G. Chen et al., "Machine Interlock and Protection System based on PLC for the SSRF Linac", Nuclear Techniques, Vol. 31, No. 7, 2008, p. 506.

[3] M. Kago, T. Matsushita, N. Nariyama et al., "Design of the Accelerator Safety Interlock System for XFEL in Spring-8", Proc. ICALEPCS'09, paper WEP096, p. 588.

[4] J. Wozniak, V. Baggiolini, D. Garcia Quintas, et al., "Software Interlocks System", Proc. ICALEPCS'07, paper WPPB03, p. 403.

[5] J. Wozniak, M. Polnik, and G. Kruk, "Groovy as Domain-Specific Language in the Software Interlock System", Proc. ICALEPCS'13, paper MOPPC142, p. 443.

[6] EPICS Channel Access for Python, `http://cars9.uchicago.edu/software/python/pyepics3/`