PfcWdOrch ← doTask (consumer : Consumer &) : void

process till consumer.m_toSync.end()

m_toSync : SyncMap

map<string, KeyOpFieldsValuesTuple>

"Ethernet0"

SET_COMMAND

std::vector<FieldValueTuple>

"action" "drop" — PfcWdAction::PFC_WD_ACTION_DROP

action : PfcWdAction

"detection_time" "200" — detectionTime : uint32_t

"restoration_time" "200" — restorationTime : uint32_t

CreateEntry

PfcWdSwOrch — startWdOnPort (port, detectionTime, restorationTime, action)

registerInWdDb (port, detectionTime, restorationTime, action)

sai_port_api→get_port_attribute (port.m_port_id, 1, &attr)

attr.id = SAI_PORT_ATTR_PRIORITY_FLOW_CONTROL

attr.value.u8 returns the TC mask under PFC

PortsOrch

- m_portList : map<string, Port>

first

second

.m_type

PHY

LAG

VLAN

.m_alias.c_str()

"Ethernet0"

.m_port_id

sai_object_id_t

"Ethernet32"

getPort (alias : string, port : Port &)

.m_queue_ids

losslessTC : set<uint8_t>

3

4

generate key from port.m_port_id ——— "pfc_WD:0:d:0x[0 0 0 0 0 0 0 xx]"

serialise entries in C_portStatIds to string, separated by ","

for each entry, call sai_serialize_port_stat() → sai_serialize_enum(counter: sai_port_stat_t,
sai_port_stat_t

&sai_metadata_enum_sai_port_stat_t)

→ valuescount ——— totalCnt

Do linear search on the metadata for counter

values[i] ——— SAI_PORT_STAT_PFC_3_RX_PKTS

valuesnames[i] — "SAI_PORT_STAT_PFC_3_RX_PKTS

Filter out non-lossless TCs by checking against losslessTC.count(tc)

"PORT_COUNTER_ID_LIST"

m_flexCounterTable → set(key, fieldValues)

"SAI_PORT_STAT_PFC_3_RX_PKTS, SAI_PORT_STAT_PFC_4_RX_PKTS, ..."