```
module ieee-dot1q-cnc-config {
  yang-version "1.1";
  namespace urn:ieee:std:802.1Q:yang:dot1q-cnc-config;
  prefix dot1q-cnc-config;
  import ieee802-dot1q-tsn-types {
    prefix tsn;
    reference
      "48.6.3 of IEEE Std 802.1Q";
  }
  organization
    "IEEE 802.1 Working Group";
  contact
    "WG-URL: http://ieee802.org/1/
    WG-EMail: stds-802-1-l@ieee.org

    Contact: IEEE 802.1 Working Group Chair
    Postal: C/O IEEE 802.1 Working Group
           IEEE Standards Association
           445 Hoes Lane
           Piscataway, NJ 08854
           USA

    E-mail: stds-802-1-chairs@ieee.org";
  description
    "This module supports management of a Time-Sensitive Networking (TSN)
    Centralized Network Configuration (CNC) component, its Configuration
    Domains - each supporting one or more Centralized User Configuration (CUC)
    components, and the Streams created by those CUCs. RPCs support
    path computation and resource allocation across all or specified
    Configuration Domains, CUCs, and Streams.

    Copyright (C) IEEE (2024).
  This version of this YANG module is part of IEEE Std 802.1Q; see
  the standard itself for full legal notices.";
revision 2024-01-25 {
  description
    "Published as part of IEEE Std 802.1Qdj-2024.

    The following reference statement identifies each referenced IEEE
    Standard as updated by applicable amendments.";
  reference
    "IEEE Std 802.1Q Bridges and Bridged Networks:
     IEEE Std 802.1Q-2022, IEEE Std 802.1Qcz-2023, IEEE Std 802.1Qcw-2023,
     IEEE Std 802.1Qcj-2023, IEEE Std 802.1Qdj-2024.";
}
container cnc-config {
  description
    "Top-level container for the CNC module.";
  list domain {
    key "domain-id";
    description
      "A list of Configuration Domains, each supporting one or more CUCs.
      Access to Streams and resources associated with particular Configuration
      Domains can be restricted.";
    leaf domain-id {
      type string;
      description
        "The DomainID uniquely identifies a Configuration Domain.";
      reference
        "46.2.2.1 of IEEE Std 802.1Q";
    }
    leaf cnc-enabled {
      type boolean;
  default "false";
  description
    "cnc-enabled is used to enable or disable CNC functionality. If TRUE,
    the CNC functionality is enabled. If FALSE, the CNC functionality is
    disabled.";
  reference
    "46.2.2.3 of IEEE Std 802.1Q";
}
list cuc {
  key "cuc-id";
  description
    "A list of CUCs in the Configuration Domain, each with its associated
    Streams. Access to  Streams and resources associated with particular
    CUCs can be restricted.";
  leaf cuc-id {
    type string;
    description
      "The CucID uniquely identifies the CUC.";
    reference
      "46.2.2.2 of IEEE Std 802.1Q";
  }
  list stream {
    key "stream-id";
    description
      "A list of Streams created by the CUC, with their status and
      configuration (talker and listener parameters) in the CNC.";
    leaf stream-id {
      type tsn:stream-id-type;
      description
        "The StreamID uniquely identifies a Stream.";
    }
    leaf stream-status {
```

```
  type enumeration {
    enum planned {
      value 0;
      description
        "The Stream has been requested but has not yet been
        configured by the CNC.";
    }
    enum configured {
      value 1;
      description
        "The Stream has been computed and configured by the
        CNC.";
    }
    enum modified {
      value 2;
      description
        "The Stream has been configured but Stream parameters
        have been modified after configuration.";
    }
  }
  config false;
  description
    "The stream-status is the status of the Stream in the CNC.";
  reference
    "46.2.3.8 of IEEE Std 802.1Q";
}
container talker {
  description
    "The Talker container for the Stream comprises the following:
    - The Talker’s behavior (how/when frames are transmitted).
    - The Talker’s network requirements.
    - The Talker's interface(s) TSN capabilities.";
  uses tsn:group-talker;
  uses tsn:group-status-talker-listener {
    refine "accumulated-latency" {
      config false;
    }
    refine "interface-configuration" {
      config false;
    }
  }
}
list listener {
  key "index";
  description
    "A list of parameters for each of the Stream's Listeners,
     each list entry comprising:
    - The Listener's interface(s).
    - The Listener’s network requirements.
    - The Listener’s interface(s) TSN capabilities.";
  leaf index {
    type uint32;
    description
      "This index is provided in order to provide a unique key
      per list entry.";
  }
  uses tsn:group-listener;
  uses tsn:group-status-talker-listener {
    refine "accumulated-latency" {
      config false;
    }
    refine "interface-configuration" {
      config false;
    }
    }
  }
  uses tsn:group-status-stream {
    refine "status-info" {
      config false;
    }
    refine "failed-interfaces" {
      config false;
    }
  }
}
action remove_streams {
  description
    "Removes the selected Streams.";
  reference
    "46.2.8.1 of IEEE Std 802.1Q";
  input {
    list stream-list {
      key "stream-id";
      description
        "List of stream-ids for the Streams to be removed.";
      leaf stream-id {
        type tsn:stream-id-type;
        description
          "The stream-id uniquely identifies a Stream.";
      }
    }
  }
  output {
    leaf result {
      type string;
      description
```

```
              "Returns status information indicating if Stream removal
              has been successfully started.";
          }
        }
      }
    }
  }
}

// RPCs
rpc compute_streams {
  description
    "Starts computation of path and resource allocation for one or more
    Streams, each identified by its domain-id, cuc-id, and stream-id.
    This RPC can compute paths and resource allocations for both new and
    modified Streams.";
  input {
    list domain {
      key "domain-id";
      description
        "A list of Configuration Domains containing the Streams for which
        paths and resource allocations are to be computed.";
      reference
        "46.2.7.1 of IEEE Std 802.1Q";
      leaf domain-id {
        type leafref {
          path '/dot1q-cnc-config/domain/domain-id';
        }
        description
          "The referenced DomainID uniquely identifies a Configuration
          Domain.";
      }
    list cuc {
     key "cuc-id";
     description
       "A list of CUCs in the Configuration Domain, each with Streams whose
       paths and resource allocations are to be computed. ";
     leaf cuc-id {
       type leafref {
         path '/dot1q-cnc-config/domain/cuc/cuc-id';
       }
       description
         "The referenced CucID uniquely identifies a CUC.";
     }
     list stream-list {
       key "stream-id";
       description
         "A list of Streams, created by the CUC, whose paths and resource
         allocations are to be computed.";
       leaf stream-id {
         type leafref {
           path '/dot1q-cnc-config/domain/cuc/stream/stream-id';
         }
         description
           "The referenced StreamID uniquely identifies a Stream.";
       }
     }
    }
  }
}
output {
  leaf result {
    type string;
    description
```

```
        "Only returns status information indicating if the computation
        has been started. It does not return status information on the
        success or failure of the actual Stream computation. A
        notification can be used to inform the caller of this RPC on the
        results of Stream computation after the computation has
        finished.";
      }
    }
  }
  rpc compute_planned_and_modified_streams {
    description
      "Starts computation of path and resource allocation for all Streams
      that are in specified CUCs in specified domains, and that have not been
      computed (i.e., that have a Stream status of planned or modified).";
    reference
      "46.2.7.2 of IEEE Std 802.1Q";
    input {
      list domain {
        key "domain-id";
        description
          "A list of Configuration Domains containing the CUCs with Streams
          whose paths and resource allocations are to be computed.";
        leaf domain-id {
          type string;
          description
            "A unique identifier of a Configuration Domain. It is used to
            identify the Configuration Domain a CUC belongs to.";
        }
        list cuc {
          key "cuc-id";
          description
            "A list of CUCs, in the Configuration Domain, that have Streams
            whose paths and resource allocations are to be computed.";
          leaf cuc-id {
            type string;
            description
              "The CucID uniquely identifies the CUC.";
          }
        }
      }
    }
    output {
      leaf result {
        type string;
        description
          "Only returns status information indicating if the computation
          has been started. It does not return status information on the
          success or failure of the actual Stream computation. A
          notification can be used to inform the caller of this RPC of the
          results of Stream computation after the computation has
          finished.";
      }
    }
  }
  rpc compute_all_streams {
    description
      "Starts computation of path and resource allocation for all Streams
      that are in specified CUCs in specified domains.";
    reference
      "46.2.7.3 of IEEE Std 802.1Q";
    input {
      list domain {
        key "domain-id";
        description
```

```
   "A list of Configuration Domains containing the CUCs with Streams
   whose paths and resource allocations are to be computed";
  leaf domain-id {
   type string;
   description
    "The DomainID uniquely identifies a Configuration Domain.";
  }
  list cuc {
   key "cuc-id";
   description
    "A list of CUCs in the Configuration Domain, each with Streams whose
    paths and resource allocations are to be computed.";
   leaf cuc-id {
    type string;
    description
     "The CucID uniquely identifies the CUC.";
   }
  }
 }
}
output {
 leaf result {
  type string;
  description
   "Only returns status information indicating if the computation
   has been started. It does not return status information on the
   success or failure of the actual Stream computation. A
   notification can be used to inform the caller of this RPC of the
   results of Stream computation after the computation has
   finished.";
 }
}
```

```
}
rpc request_domain_id {
  description
    "Returns the DomainID of the Configuration Domain to which a specified
    CUC belongs.";
  reference
    "46.2.7.4 of IEEE Std 802.1Q";
  input {
    leaf cuc-id {
      type string;
      description
        "The CucID uniquely identifies the CUC.";
    }
  }
  output {
    leaf result {
      type string;
      description
        "Returns the DomainID of the Configuration Domain for the
        specified CUC.";
    }
  }
}
rpc request_free_stream_id {
  description
    "Returns a free (i.e., available) StreamID for use by a specified
    CUC in a specified Configuration Domain.";
  reference
    "46.2.7.5 of IEEE Std 802.1Q";
  input {
    leaf domain-id {
      type string;
      description
        "The DomainID uniquely identifies the Configuration Domain.";
    }
    leaf cuc-id {
      type string;
      description
        "The CucID uniquely identifies the CUC.";
    }
  }
  output {
    leaf result {
      type string;
      description
        "Returns a free (i.e., available) StreamID for use by the
        specified CUC in the specified Configuration Domain.";
    }
  }
}

// Notifications
notification compute_streams_completed {
  description
    "Notifies the caller of an RPC or action that initiated computation for
    one or more Streams, that the computation is complete. Returns information
    on the success or failure of computation for each of those Streams.";
  reference
    "46.2.9.1 of IEEE Std 802.1Q";
  list domain {
    key "domain-id";
    description
      "The list of Configuration Domains for which computation was requested,
      with each list entry specifying the CUCs, and the Streams for each of
      those CUCs for which computation was requested, with the result of the
      computation (success or failure) for each Stream.";
    leaf domain-id {
      type string;
      description
        "The DomainID uniquely identifies a Configuration Domain.";
    }
    list cuc {
      key "cuc-id";
      description
        "The list of CUCs for the specified Configuration Domain, with each
        list entry specifying the Streams for which computation was requested
        and the result of the computation for each Stream.";
      leaf cuc-id {
        type string;
        description
          "The CucID uniquely identifies the CUC.";
      }
      list stream {
        key "stream-id";
        description
          "The list of Streams, for which computation was requested, for the
          specified Configuration Domain and CUC, with the result of the
          computation (success or failure) for each Stream.";
        reference
          "46.2.3 of IEEE Std 802.1Q";
        leaf stream-id {
          type tsn:stream-id-type;
          description
            "The StreamID uniquely identifies a Stream.";
        }
        leaf failure-code {
          type uint8;
          description
            "A code that indicates if the computation for the Stream
            was successful (0) or not. In the case of a failure a code
            is returned to indicate what kind of failure occurred.";
        }
      }
    }
  }
}
notification configure_streams_completed {
  description
    "Notifies the caller of an RPC or action that initiated computation for
    one or more Streams, that computation and configuration is complete.
    Returns information on the success or failure information of computation
    and configuration for each of those Streams.";
  reference
    "46.2.9.2 of IEEE Std 802.1Q";
  list domain {
    key "domain-id";
    description
      "The list of Configuration Domains for which computation was requested,
      with each list entry specifying the CUCs, and the Streams for each of
      those CUCs for which computation was requested, with the result of the
      computation (success or failure) for each Stream.";
    leaf domain-id {
      type string;
      description
        "The DomainID uniquely identifies a Configuration Domain.";
    }
    list cuc {
      key "cuc-id";
  description
    "The list of CUCs for the specified Configuration Domain, with each
    list entry specifying the Streams for which computation was requested
    and the result of the computation for each Stream.";
  leaf cuc-id {
    type string;
    description
      "The CucID uniquely identifies the CUC.";
  }
  list stream {
    key "stream-id";
    description
      "The list of Streams, for which computation was requested, for the
      specified Configuration Domain and CUC, with the result of the
      computation (success or failure) for each Stream.";
    reference
      "46.2.3 of IEEE Std 802.1Q";
    leaf stream-id {
      type tsn:stream-id-type;
      description
        "The StreamID uniquely identifies a Stream.";
    }
    leaf failure-code {
      type uint8;
      description
        "A code that indicates if the computation and configuration
        for the Stream was successful (0) or not. In the case of a
        failure a code is returned to indicate what kind of failure
        occurred.";
    }
  }
}
```

```
 }
}
notification remove_streams_completed {
 description
  "Notifies the caller of an RPC or action that initiated computation for
  one or more Streams, that the removal is complete. Returns information
  on the success or failure of removal for each of those Streams.";
 reference
  "46.2.9.3 of IEEE Std 802.1Q";
 list domain {
  key "domain-id";
  description
   "A list of Configuration Domains with each list entry specifying the
   CUCs, and the Streams for each of those CUCs for which removal was
   requested, with the result of the removal attempt (success or failure)
   for each Stream.";
  leaf domain-id {
   type string;
   description
    "The DomainID uniquely identifies a Configuration Domain.";
  }
  list cuc {
   key "cuc-id";
   description
    "The list of CUCs for the specified Configuration Domain, with each
    list entry specifying the Streams for which removal was requested
    and the result of the removal attempt for each Stream.";
   leaf cuc-id {
    type string;
    description
     "The CucID uniquely identifies the CUC.";
   }
```

```
        list stream {
          key "stream-id";
          description
            "The list of Streams, for which removal was requested, for the
            specified Configuration Domain and CUC, with the result of the
            removal (success or failure) for each Stream.";
          reference
            "46.2.3 of IEEE Std 802.1Q";
          leaf stream-id {
            type tsn:stream-id-type;
            description
              "The StreamID uniquely identifies a Stream.";
          }
          leaf failure-code {
            type uint8;
            description
              "A code that indicates if the removal of the Stream was
              successful (0) or unsuccessful (1).";
          }
        }
      }
    }
  }
}
```