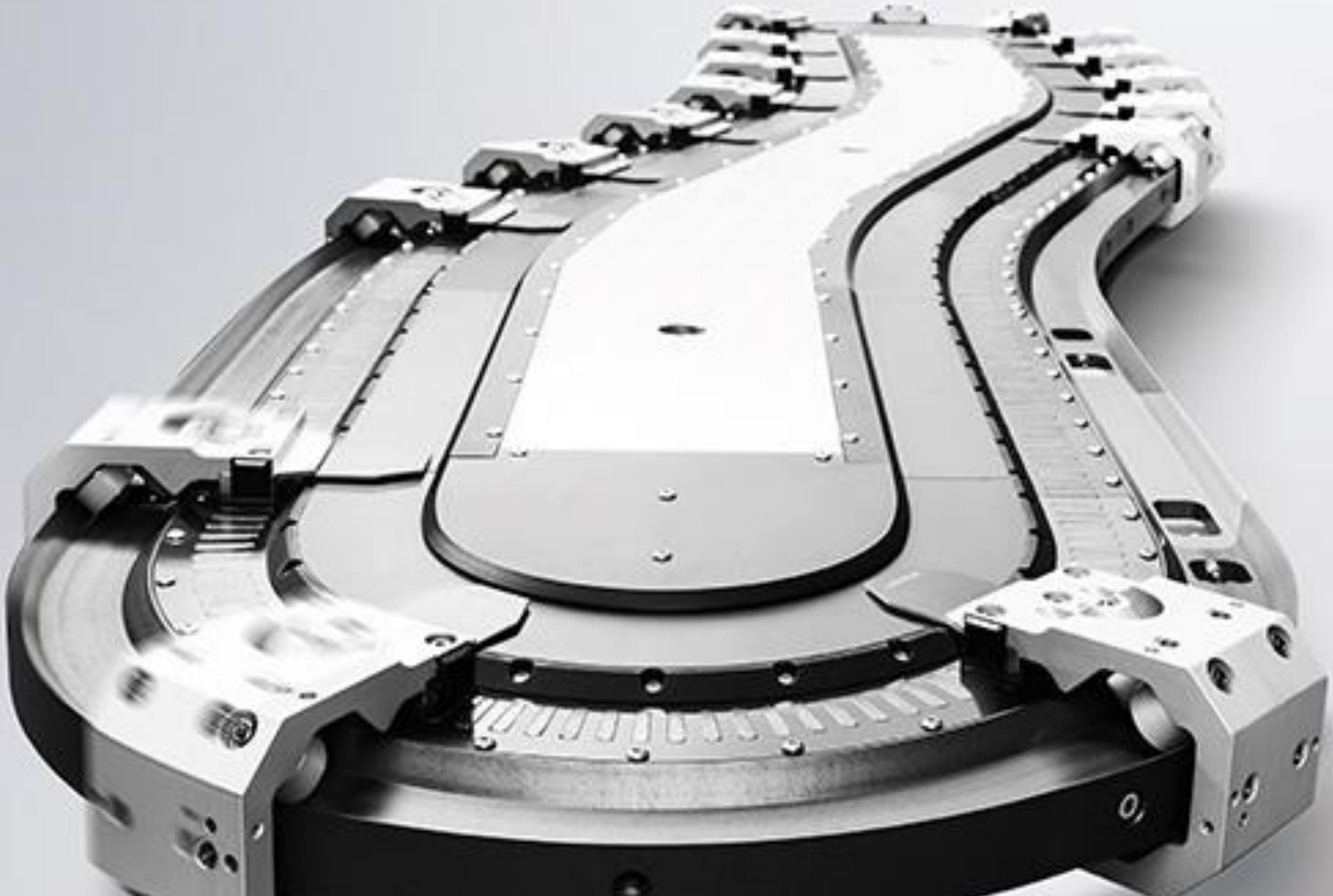
XTS TRANSPORT LAYER – a station based approach

BECKHOFF



XTS TRANSPORT LAYER (HAUD)

1. Requirements
   - XtsTransport (main control)
   - Xpu (XTS Processing Unit)
   - CaGroup (Collision Avoidance)
   - Mover (MC and CA)
   - Station (process handshake)

2. Design
   - use with any cyclic runtime
   - use with non cyclic software

3. License

1. Requirements

- XtsTransport
  - Access to CA group function blocks (interface pointer)
  - Access to Stations (interface pointer)
  - Access to Movers (interface pointer)
  - Cyclic interface for access from extern control
    - Ctrl (write): command
    - State (read): response to command
      - information from Xpu
      - Information from CA Group

1. Requirements

- Xpu (XTS Processing Unit)
  - Check Init Parameter
  - Check Online Parameter
  - Get Module Info Data
  - Connect TcCOM Objects to instances from XTS_Utility.lib function blocks
  - Cyclic plausibility checks
    - Mover ID detection after init
  - Cyclic interface for access from main control
    - Ctrl (write): command
    - State (read): response to command
    - Info (read): details from cyclic checks

1. Requirements

- CaGroup
  - Access to group function blocks
  - Access to movers for group commands
  - Get Group Info Data
  - Implements interface pointer

1. Requirements

- Mover
  - Access to MC function blocks
  - Access to CA function blocks
  - Cyclic interface for access from extern control
    - Ctrl (write): command
    - Data (write): command parameter
    - State (read): response to command
  - Interface pointer for access from:
    - TransportUnit
    - Station

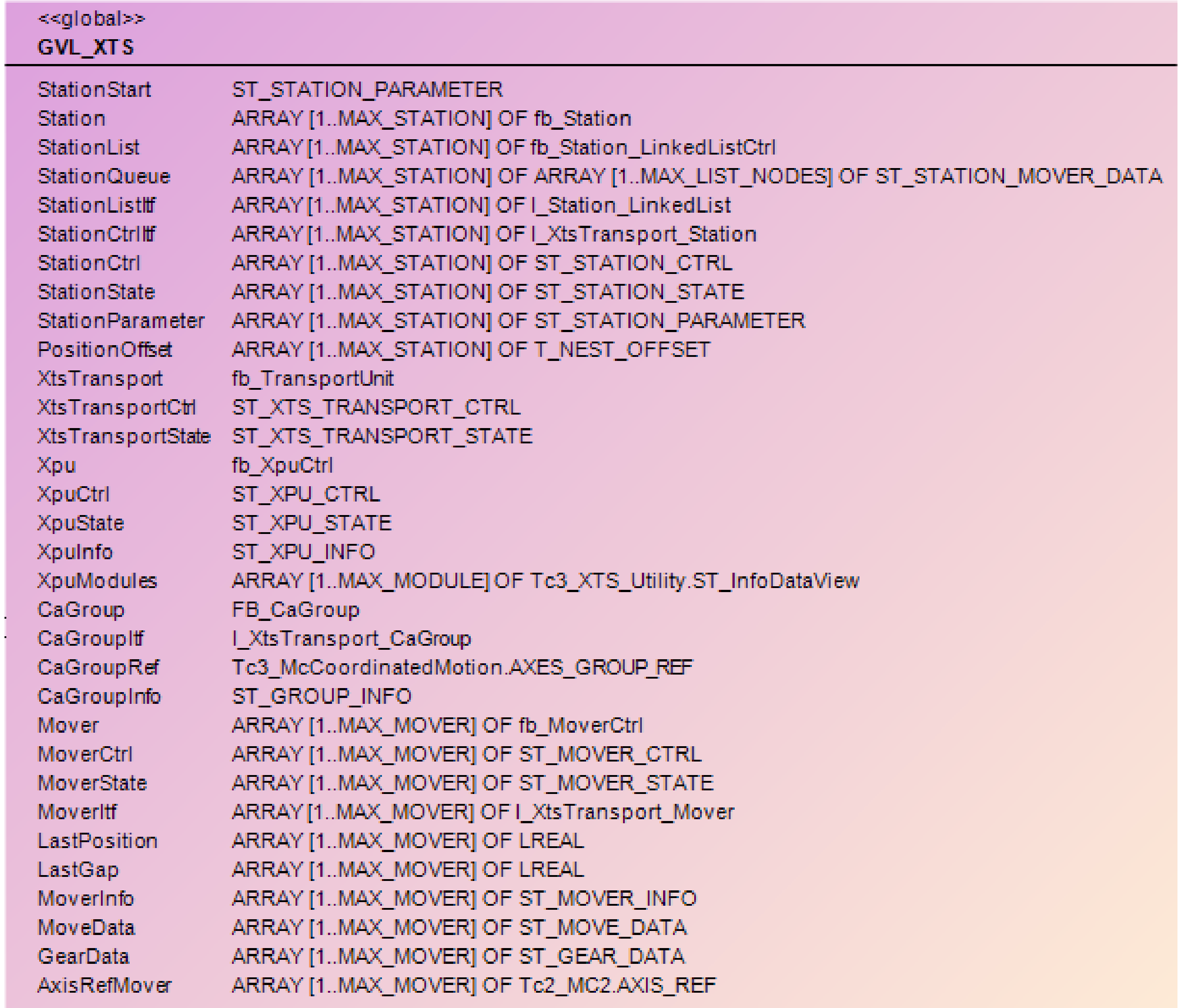
1. Requirements

BECKHOFF

- Station
  - Handshake mover transport with extern control
  - Close observation of movements with feedback to extern control
  - List for movers in queue
  - Cyclic interface for access from extern control
    - Ctrl (write): command and parameter
    - State (read): response to command and information about mover and queue
  - Uses Mover interface pointer

2. Design

- Namespace GVL_XTS
  - Station
    - Handshake with Process for mover transport
  - TransportUnit
    - Main command interface to extern control
    - XpuCtrl
      - Access to TcCOM Objects
      - Cyclic plausibility checks
    - CaGroup
      - Access to CA library
  - Fb_MoverCtrl
    - Access to MC and CA library



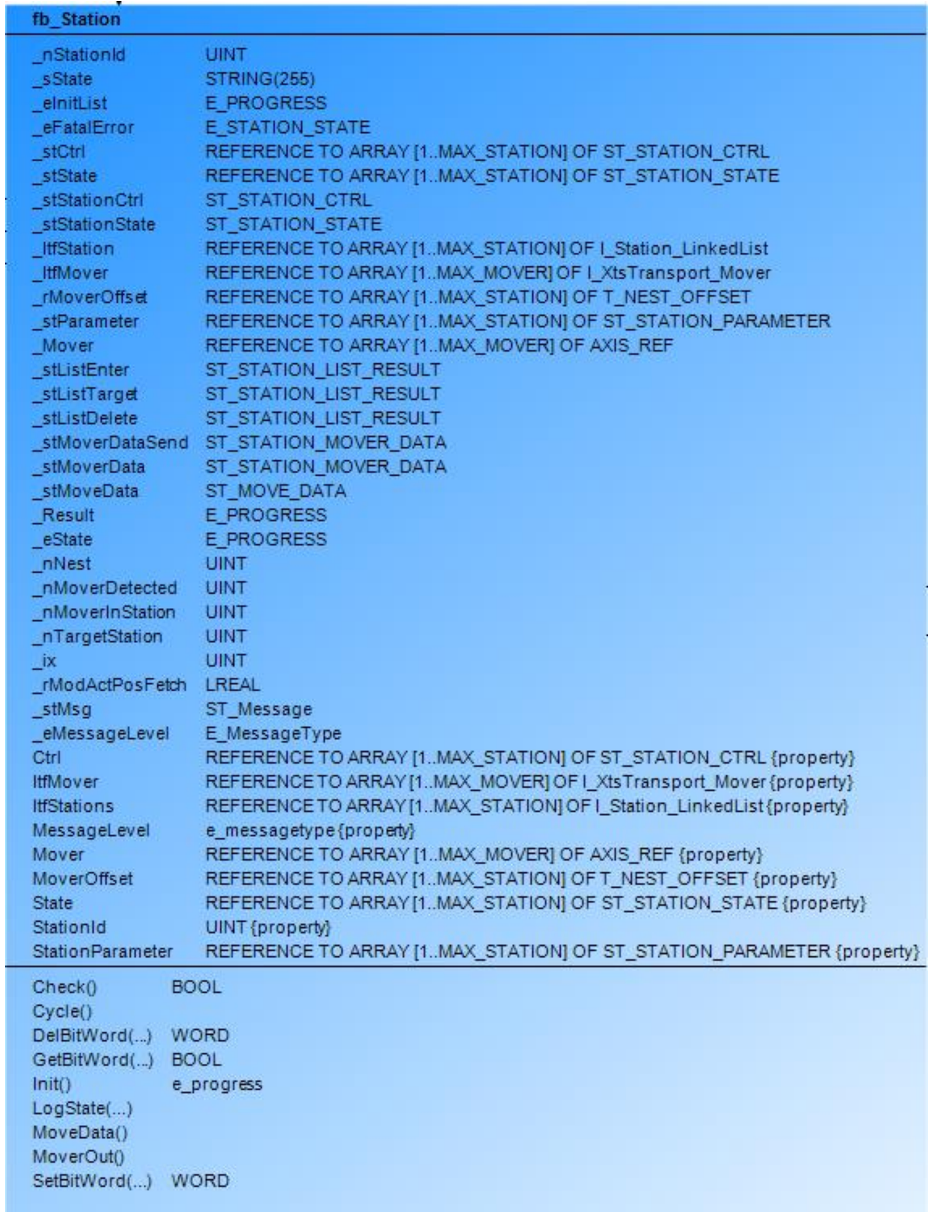
<<global>>

GVL_XTS

StationStart	ST_STATION_PARAMETER
Station	ARRAY [1..MAX_STATION] OF fb_Station
StationList	ARRAY [1..MAX_STATION] OF fb_Station_LinkedListCtrl
StationQueue	ARRAY [1..MAX_STATION] OF ARRAY [1..MAX_LIST_NODES] OF ST_STATION_MOVER_DATA
StationListItf	ARRAY [1..MAX_STATION] OF I_Station_LinkedList
StationCtrlItf	ARRAY [1..MAX_STATION] OF I_XtsTransport_Station
StationCtrl	ARRAY [1..MAX_STATION] OF ST_STATION_CTRL
StationState	ARRAY [1..MAX_STATION] OF ST_STATION_STATE
StationParameter	ARRAY [1..MAX_STATION] OF ST_STATION_PARAMETER
PositionOffset	ARRAY [1..MAX_STATION] OF T_NEST_OFFSET
XtsTransport	fb_TransportUnit
XtsTransportCtrl	ST_XTS_TRANSPORT_CTRL
XtsTransportState	ST_XTS_TRANSPORT_STATE
Xpu	fb_XpuCtrl
XpuCtrl	ST_XPU_CTRL
XpuState	ST_XPU_STATE
XpuInfo	ST_XPU_INFO
XpuModules	ARRAY [1..MAX_MODULE] OF Tc3_XTS_Utility.ST_InfoDataView
CaGroup	FB_CaGroup
CaGroupItf	I_XtsTransport_CaGroup
CaGroupRef	Tc3_McCoordinatedMotion.AXES_GROUP_REF
CaGroupInfo	ST_GROUP_INFO
Mover	ARRAY [1..MAX_MOVER] OF fb_MoverCtrl
MoverCtrl	ARRAY [1..MAX_MOVER] OF ST_MOVER_CTRL
MoverState	ARRAY [1..MAX_MOVER] OF ST_MOVER_STATE
MoverItf	ARRAY [1..MAX_MOVER] OF I_XtsTransport_Mover
LastPosition	ARRAY [1..MAX_MOVER] OF LREAL
LastGap	ARRAY [1..MAX_MOVER] OF LREAL
MoverInfo	ARRAY [1..MAX_MOVER] OF ST_MOVER_INFO
MoveData	ARRAY [1..MAX_MOVER] OF ST_MOVE_DATA
GearData	ARRAY [1..MAX_MOVER] OF ST_GEAR_DATA
AxisRefMover	ARRAY [1..MAX_MOVER] OF Tc2_MC2.AXIS_REF

2. Design

- Station
  - fb_Station[].Cycle
    - State machine for handshaking with extern control
      - Init (clears everything in station)
      - Enable
      - Mover Enter
      - Stop Position(s)
      - Mover Out
      - Empty
    - Control writes ticket for mover
      - MoverId
      - TargetStation
      - Mask
      - Offset



fb_Station

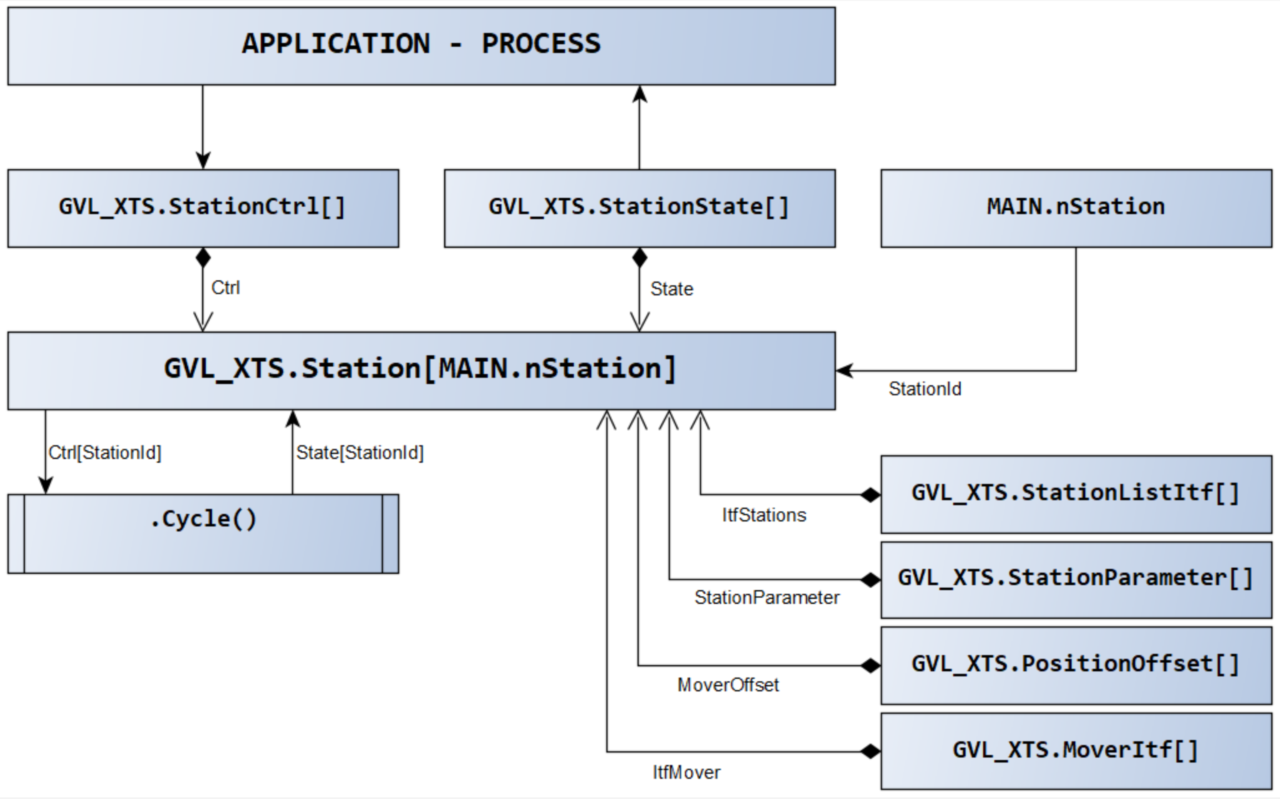
_nStationId	UINT
_sState	STRING(255)
_eInitList	E_PROGRESS
_eFatalError	E_STATION_STATE
_stCtrl	REFERENCE TO ARRAY [1..MAX_STATION] OF ST_STATION_CTRL
_stState	REFERENCE TO ARRAY [1..MAX_STATION] OF ST_STATION_STATE
_stStationCtrl	ST_STATION_CTRL
_stStationState	ST_STATION_STATE
_ItfStation	REFERENCE TO ARRAY [1..MAX_STATION] OF I_Station_LinkedList
_ItfMover	REFERENCE TO ARRAY [1..MAX_MOVER] OF I_XtsTransport_Mover
_rMoverOffset	REFERENCE TO ARRAY [1..MAX_STATION] OF T_NEST_OFFSET
_stParameter	REFERENCE TO ARRAY [1..MAX_STATION] OF ST_STATION_PARAMETER
_Mover	REFERENCE TO ARRAY [1..MAX_MOVER] OF AXIS_REF
_stListEnter	ST_STATION_LIST_RESULT
_stListTarget	ST_STATION_LIST_RESULT
_stListDelete	ST_STATION_LIST_RESULT
_stMoverDataSend	ST_STATION_MOVER_DATA
_stMoverData	ST_STATION_MOVER_DATA
_stMoveData	ST_MOVE_DATA
_Result	E_PROGRESS
_eState	E_PROGRESS
_nNest	UINT
_nMoverDetected	UINT
_nMoverInStation	UINT
_nTargetStation	UINT
_ix	UINT
_rModActPosFetch	LREAL
_stMsg	ST_Message
_eMessageLevel	E_MessageType
Ctrl	REFERENCE TO ARRAY [1..MAX_STATION] OF ST_STATION_CTRL {property}
ItfMover	REFERENCE TO ARRAY [1..MAX_MOVER] OF I_XtsTransport_Mover {property}
ItfStations	REFERENCE TO ARRAY [1..MAX_STATION] OF I_Station_LinkedList {property}
MessageLevel	e_messagetype {property}
Mover	REFERENCE TO ARRAY [1..MAX_MOVER] OF AXIS_REF {property}
MoverOffset	REFERENCE TO ARRAY [1..MAX_STATION] OF T_NEST_OFFSET {property}
State	REFERENCE TO ARRAY [1..MAX_STATION] OF ST_STATION_STATE {property}
StationId	UINT {property}
StationParameter	REFERENCE TO ARRAY [1..MAX_STATION] OF ST_STATION_PARAMETER {property}

Check()	BOOL
Cycle()	
DelBitWord(...)	WORD
GetBitWord(...)	BOOL
Init()	e_progress
LogState(...)	
MoveData()	
MoverOut()	
SetBitWord(...)	WORD

2. Design

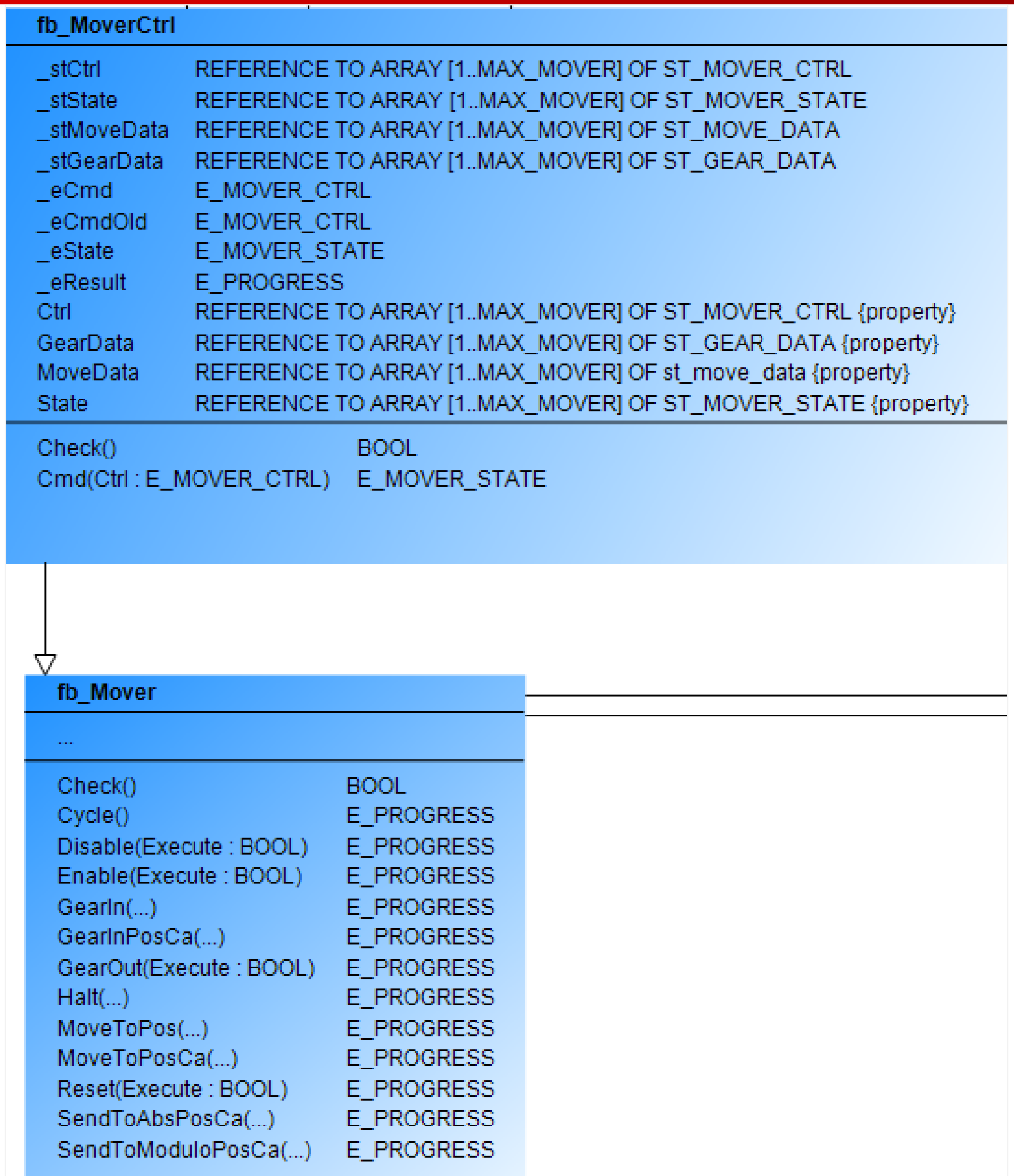
BECKHOFF

- Station
  - Station index is passed as value from caller
  - Global datafields are passed as references (REF=) into fb_Station properties
    - Ctrl / State: handshakes
    - ItfStations: interface pointer to linked list methods for getting and setting of mover data
    - StationParameter: Coordinates and dynamic constraint of XtsStation
    - MoverOffset: correction values for every mover in every station with every nest (StopPos[])
    - ItfMover: interface pointer to CA movements



2. Design

- fb_MoverCtrl:
  - Inherits fb_Mover
    - Access to MC function blocks in library
    - Implements Interface for use in other classes
  - Contains cyclic interface
    - Ctrl datafield for setting commands
    - State data field for checking responses
    - Parameter datafields for using motion functions



fb_MoverCtrl

_stCtrl	REFERENCE TO ARRAY [1..MAX_MOVER] OF ST_MOVER_CTRL
_stState	REFERENCE TO ARRAY [1..MAX_MOVER] OF ST_MOVER_STATE
_stMoveData	REFERENCE TO ARRAY [1..MAX_MOVER] OF ST_MOVE_DATA
_stGearData	REFERENCE TO ARRAY [1..MAX_MOVER] OF ST_GEAR_DATA
_eCmd	E_MOVER_CTRL
_eCmdOld	E_MOVER_CTRL
_eState	E_MOVER_STATE
_eResult	E_PROGRESS
Ctrl	REFERENCE TO ARRAY [1..MAX_MOVER] OF ST_MOVER_CTRL {property}
GearData	REFERENCE TO ARRAY [1..MAX_MOVER] OF ST_GEAR_DATA {property}
MoveData	REFERENCE TO ARRAY [1..MAX_MOVER] OF st_move_data {property}
State	REFERENCE TO ARRAY [1..MAX_MOVER] OF ST_MOVER_STATE {property}

Check()	BOOL
Cmd(Ctrl : E_MOVER_CTRL)	E_MOVER_STATE

fb_Mover

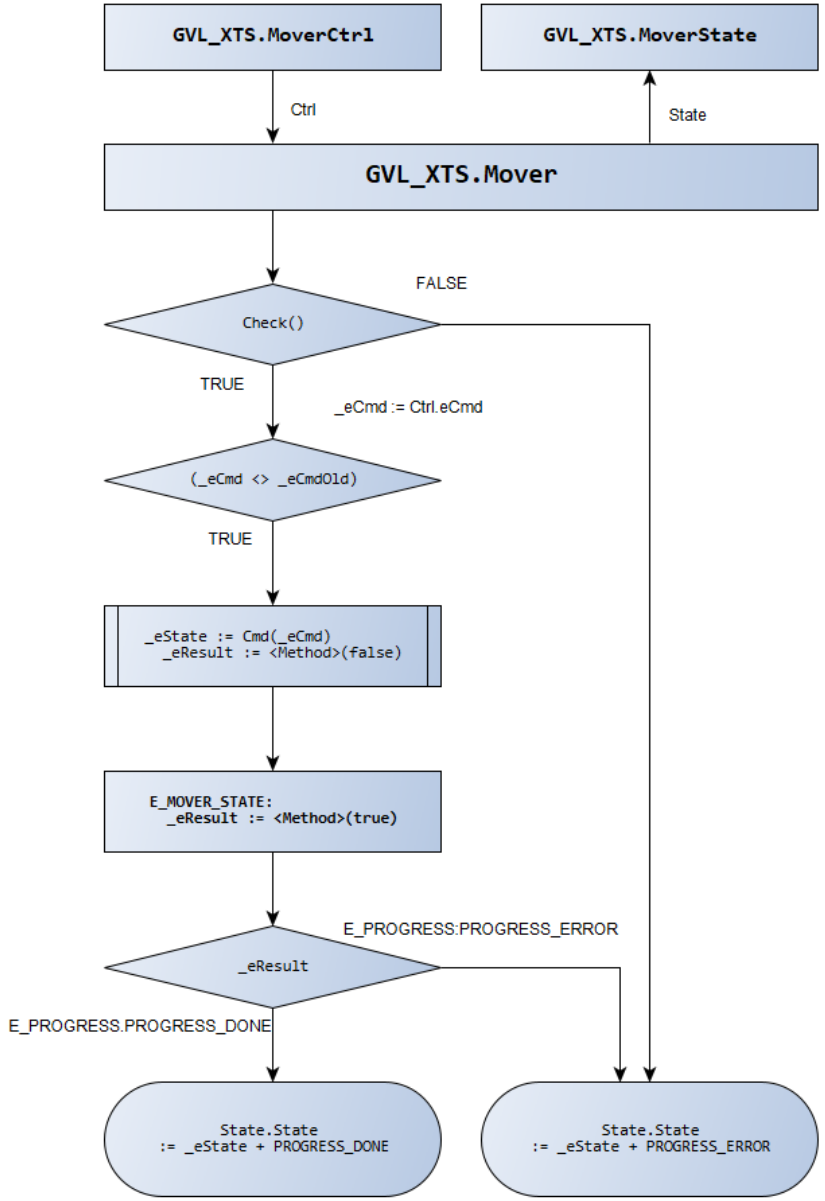
...

Check()	BOOL
Cycle()	E_PROGRESS
Disable(Execute : BOOL)	E_PROGRESS
Enable(Execute : BOOL)	E_PROGRESS
GearIn(...)	E_PROGRESS
GearInPosCa(...)	E_PROGRESS
GearOut(Execute : BOOL)	E_PROGRESS
Halt(...)	E_PROGRESS
MoveToPos(...)	E_PROGRESS
MoveToPosCa(...)	E_PROGRESS
Reset(Execute : BOOL)	E_PROGRESS
SendToAbsPosCa(...)	E_PROGRESS
SendToModuloPosCa(...)	E_PROGRESS

2. Design

BECKHOFF

- fb_MoverCtrl:
  - Mover index is passed as value from caller
  - Global datafields are passed as references (REF=) into fb_MoverCtrl properties
    - Ctrl / State: handshakes
    - standard return value for method
    - Log LastPosition on CA/MC function execute
    - Log LastGap on CA function execute


GVL_XTS.MoverCtrl
GVL_XTS.MoverState
Ctrl
State
GVL_XTS.Mover
FALSE
Check()
TRUE
_eCmd := Ctrl.eCmd
(_eCmd <> _eCmdOld)
TRUE
_eState := Cmd(_eCmd)
_eResult := <Method>(false)
E_MOVER_STATE:
_eResult := <Method>(true)
E_PROGRESS:PROGRESS_ERROR
_eResult
E_PROGRESS.PROGRESS_DONE
State.State
:= _eState + PROGRESS_DONE
State.State
:= _eState + PROGRESS_ERROR