**Structures and Enumerations**

|  |  |  |
| --- | --- | --- |
| **ST\_VelPlan\_t** | | |
|  | The ST\_VelPlan\_t object contains all configuration parameters of the Velocity Plan component | |
|  | **typedef struct** {  /\* Input variable \*/  // Control Bits  *bool ENB;*  *bool RES;*  /\* Output variable \*/  *\_iq24 VelEnd;*  *\_iq24 AccLim;*  *\_iq20 JrkLim;*  // Information varaiable  *ST\_PlanStatus\_e STATUS;*  *unit16\_t CurState;*  *unit16\_t CurTran;*  *ST\_PlanFsmState FsmState;*  *int32\_t Timer\_ticks;*  *unit16\_t ERR\_ID;*  *ST\_PlanErroe\_t CgfError;*  /\* Internal variables \*/  *unit32\_t s0[26];*  } ST\_VelPlan\_t; | //!< Enable bit {false: disable; true: enabled}  //!< Reset bit {false: not reset; true: reset}  //!< Velocity setpoint {unit:[pu/s], value range:[-1.0, 1.0]}  //!< Acceleration limit {unit:[pu/s^2], value range:[0.001, 120.0]}  //!< Jerk Limit {unit:[pu/s^3], value range:[0.0005,2000.0]}  //!< Plan status  //!< Current state of the state machine  //!< Current transition of the state machine  //!< State of the sate machine  //!< State Timer{unit:[tick]}  //!< Error ID  //!< Error decoding structure  // Structure for SpinTAC Velocity plan |

|  |  |  |
| --- | --- | --- |
| **ST\_PlanError\_t** | | |
|  | The ST\_PlanError\_t object contains error debugging information for the Velocity  Plan component | |
|  | **typedef struct** {  *uint16\_t ERR\_idx;*  *unit16\_t ERR\_code;*  } ST\_PlanError\_t; | //!< Plan component index that caused the error  //!< Function specific condition that caused the error  // Structure for SpinTAC Plan Error |

|  |  |  |
| --- | --- | --- |
| **ST\_PlanStatus\_e** | | |
|  | Enumeration for the Plan Status States | |
|  | **typedef enum** {  *ST\_PLAN\_IDLE=0,*  *ST\_PLAN\_INIT,*  *ST\_PLAN\_BUSY,*  *ST\_PLAN\_HALT,*  *ST\_PLAN\_WAIT,*  }ST\_PlanStatus\_e; | //!< idle state, holding velocity  //!< init state, initialize internal states  //!< busy state, running state machine  //!< halt state, return to the speed reference in the idle state  //!< wait state, holding the speed reference in idle state |

|  |  |  |
| --- | --- | --- |
| **ST\_PlanCond\_e** | | |
|  | Enumeration for the Plan Condition option | |
|  | **typedef enum** {  *ST\_COND\_NC=0,*  *ST\_COND\_FC,*  *ST\_COND\_AND,*  *ST\_COND\_OR,*  } ST\_PlanCond\_e; | //!< Transition/Action with no condition satisfied  //!< Transition/Action with first condition satisfied  //!< Transition/Action with both condition satisfied  //!< Transition/Action with either condition satisfied |

|  |  |  |
| --- | --- | --- |
| **ST\_PlanComp\_e** | | |
|  | Enumeration for the Plan Compare option | |
|  | **typedef enum** {  *ST\_COMP\_NA=0,*  *ST\_COMP\_EQ,*  *ST\_COMP\_NEQ,*  *ST\_COMP\_GT,*  *ST\_COMP\_EGT,*  *ST\_COMP\_LW,*  *ST\_COMP\_ELW,*  *ST\_COMP\_In,*  *ST\_COMP\_EIn,*  *ST\_COMP\_InE,*  *ST\_COMP\_EInE,*  *ST\_COMP\_Out,*  *ST\_COMP\_Eout,*  *ST\_COMP\_OutE,*  *ST\_COMP\_EoutE,*  } ST\_PlanComp\_e; | //!< No comparison  //!< Compares, VarIdx == Value1 OR VarIdx1 == VarIdx2  //!< Compares, VarIdx != Value1 OR VarIdx1 != VarIdx2  //!< Compares, VarIdx > Value1 OR VarIdx1 > VarIdx2  //!< Compares, VarIdx >= Value1 OR VarIdx1 >= VarIdx2  //!< Compares, VarIdx < Value1 OR VarIdx1 < VarIdx2  //!< Compares, VarIdx <= Value1 OR VarIdx1 <= VarIdx2  //!< Compares, Value1 <= VarIdx <= Value2  //!< Compares, Value1 < VarIdx <= Value2  //!< Compares, Value1 <= VarIdx < Value2  //!< Compares, Value1 < VarIdx < Value2  //!< Compares, Value1 >= VarIdx >= Value2  //!< Compares, Value1 > VarIdx >= Value2  //!< Compares, Value1 >= VarIdx > Value2  //!< Compares, Value1 > VarIdx > Value2 |

|  |  |  |
| --- | --- | --- |
| **ST\_PlanActOptn\_e** | | |
|  | Enumeration for the Plan Action Option | |
|  | **typedef enum** {  *ST\_ACT\_EQ=0,*  *ST\_ACT\_ADD,*  } ST\_PlanActOptn\_e; | //!< Action will set the variable equal to a value  //!< Action will add a value to the variable |

|  |  |  |
| --- | --- | --- |
| **ST\_PlanActTrgr\_e** | | |
|  | Enumeration for the Plan Action trigger point types | |
|  | **typedef enum** {  *ST\_ACT\_ENTR=0,*  *ST\_ACT\_EXIT,*  } ST\_PlanActTrgr\_e; | //!< Action will be considered when entering the state  //!< Action will be considered when exiting the state |

|  |  |  |
| --- | --- | --- |
| **ST\_PlanFsmState\_e** | | |
|  | Enumeration for the Plan FSM | |
|  | **typedef enum** {  *ST\_FSM\_STATE\_STAY=0,*  *ST\_FSM\_STATE\_COND,*  *ST\_FSM\_STATE\_TRAN*  } ST\_PlanFsmState\_e; | //!< Plan is staying in a state until the timer is up  //!< Plan is waiting for a transition condition to be true  //!< Plan is in transition with a motion profile |

|  |  |  |
| --- | --- | --- |
| **ST\_PlanVar\_e** | | |
|  | Enumeration for the Plan variable types | |
|  | **typedef enum** {  *ST\_VAR\_INOUT=0,*  *ST\_VAR\_IN,*  *ST\_VAR\_OUT,*  } ST\_PlanVar\_e; | //!< Variable type input/output  //!< Variable type input  //!< Variable type output |

**STVELPLAN Get Function**

|  |  |
| --- | --- |
| **STVELPLAN\_getVelocitySetpoint()** | |
|  | **Static inline \_iq24 STVELPLAN\_getVelocitySetpoint(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Velocity Setpoint (VelEnd) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | \_iq24 VelEnd Velocity setpoint {unit: [pu/s], value range: [-1.0, 1.0]} |

|  |  |
| --- | --- |
| **STVELPLAN\_getAccelerationLimit()** | |
|  | **Static inline \_iq24 STVELPLAN\_getAccelerationLimit (ST\_VELPLAN\_Handle handle)** |
|  | Gets the Acceleration Limit(AccLim) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | \_iq24 AccLIm Acceleration limit {unit: [pu/s^2], value range: [0.001, 120.0]} |

|  |  |
| --- | --- |
| **STVELPLAN\_getJerkLimit()** | |
|  | **Static inline \_iq20 STVELPLAN\_getJerkLimit(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Jerk Limit (JrkLim) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | \_iq20 JrkLim Jerk Limit {unit: [pu/s^3], value range: [0.0005, 2000.0]} |

|  |  |
| --- | --- |
| **STVELPLAN\_getEnable()** | |
|  | **static inline bool STVELPLAN\_getEnable(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Enable signal (ENB) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | bool ENB Enable bit { false: disable; true: enable |

|  |  |
| --- | --- |
| **STVELPLAN\_getReset()** | |
|  | **static inline bool STVELPLAN\_getReset(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Reset signal (RES) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | RES Reset bit { false: reset; true: not reset } |

|  |  |
| --- | --- |
| **STVELPLAN\_getStatus()** | |
|  | **static inline ST\_PlanStatus\_e STVELPLAN\_getStatus(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Status value (STATUS) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | ST\_PlanStatus\_e STATUS Status { ST\_VEL\_ID\_IDLE, ST\_VEL\_ID\_INIT, ST\_VEL\_ID\_BUSY } |

|  |  |
| --- | --- |
| **STVELPLAN\_getCurrentState()** | |
|  | **static inline uint16\_t STVELPLAN\_getCurrentState(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Current State (CurState) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | uint16\_t CurState Current state of the state machine |

|  |  |
| --- | --- |
| **STVELPLAN\_getCurrentTransition()** | |
|  | **static inline uint16\_t STVELPLAN\_getCurrentTransition(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Current Transition (CurTran) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | uint16\_t CurTran Current transition of the state machine |

|  |  |
| --- | --- |
| **STVELPLAN\_getFsmState()** | |
|  | **static inline ST\_PlanFsmState\_e STVELPLAN\_getFsmState(ST\_VELPLAN\_Handle handle)** |
|  | Gets the State Machine State (FsmState) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | ST\_PlanFsmState\_e FsmState Current operation of the state machine |

|  |  |
| --- | --- |
| **STVELPLAN\_getCurrentTimerValue\_tick()** | |
|  | **static inline int32\_t STVELPLAN\_getCurrentTimerValue\_tick(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Current Time Value (Timer\_tick) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | int32\_t Timer\_tick State Timer { unit: [tick] } |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgError()** | |
|  | **static inline uint16\_t STVELPLAN\_getCfgError(ST\_VELPLAN\_Handle handle, uint16\_t \*ERR\_idx, uint16\_t \*ERR\_code)** |
|  | Gets the Configuration Error for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **ERR\_idx:** | Component index where error occurred { 0: no index; others: see error code } |
| **ERR\_code:** | Specific configuration error { 0: no additional information; others: see error code } |
| **Return:** | uint16\_t ERR\_ID Error ID { 0: no error; others: see error code } |

|  |  |
| --- | --- |
| **STVELPLAN\_getErrorID()** | |
|  | **static inline uint16\_t STVELPLAN\_getErrorID(ST\_VELPLAN\_Handle handle)** |
|  | Gets the Error ID for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Return:** | uint16\_t ERR\_ID Error ID { 0: no error; others: see error code } |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgStateNum()** | |
|  | **void STVELPLAN\_getCfgStateNum(ST\_VELPLAN\_Handle handle, uint16\_t \*StateNum)** |
|  | Returns the number of configured States in SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **StateNum:** | The number of configured States in SpinTAC Velocity Plan |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgVarNum()** | |
|  | **void STVELPLAN\_getCfgVarNum(ST\_VELPLAN\_Handle handle, uint16\_t \*VarNum)** |
|  | Returns the number of configured Variables in SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VarNum:** | The number of configured Variables in SpinTAC Velocity Plan |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgCondNum()** | |
|  | **void STVELPLAN\_getCfgCondNum(ST\_VELPLAN\_Handle handle, uint16\_t \*CondNum)** |
|  | Returns the number of configured Conditions in SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **CondNum:** | The number of configured Conditions in SpinTAC Velocity Plan |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgTranNum()** | |
|  | **void STVELPLAN\_getCfgTranNum(ST\_VELPLAN\_Handle handle, uint16\_t \*TranNum)** |
|  | Returns the number of configured Transitions in SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **TranNum:** | The number of configured Transitions in SpinTAC Velocity Plan |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgActNum()** | |
|  | **void STVELPLAN\_getCfgActNum(ST\_VELPLAN\_Handle handle, uint16\_t \*ActNum)** |
|  | Returns the number of configured Actions in SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **ActNum:** | The number of configured Actions in SpinTAC Velocity Plan |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgCond()** | |
|  | **void STVELPLAN\_getCfgCond(ST\_VELPLAN\_Handle handle, uint16\_t CondIdx, uint16\_t \*VarIdx, ST\_PlanComp\_e \*Comp, \_iq24 \*Value1, \_iq24 \*Value2)** |
|  | Returns a Condition from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **CondIdx:** | Index of the Plan Condition to return |
| **VarIdx:** | Index of the Plan Variable that will be compared |
| **Comp:** | Type of comparison that will be done |
| **Value1:** | First value to use in the comparison |
| **Value2:** | Second value to use in the comparison |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgVarCond()** | |
|  | **void STVELPLAN\_getCfgVarCond(ST\_VELPLAN\_Handle handle, uint16\_t CondIdx, uint16\_t \*VarIdx1, uint16\_t \*VarIdx2, ST\_PlanComp\_e \*Comp)** |
|  | Returns a Variable Condition from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **CondIdx:** | Index of the Plan Condition to return |
| **VarIdx1:** | Index of the first Plan Variable that will be compared |
| **VarIdx2:** | Index of the second Plan Variable that will be compared |
| **Comp:** | Type of comparison that will be done |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgTran()** | |
|  | **void STVELPLAN\_getCfgTran(ST\_VELPLAN\_Handle handle, uint16\_t TranIdx, uint16\_t \*FromState, uint16\_t \*ToState, ST\_PlanCond\_e \*AndOr, uint16\_t \*CondIdx1, uint16\_t \*CondIdx2, \_iq24 \*AccLim, \_iq20 \*JrkLim)** |
|  | Return a Transition from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **TranIdx:** | Index of the Transition to modify |
| **FromState:** | Index of the from State |
| **ToState:** | Index of the to State |
| **AndOr** | Condition option for the Transition |
| **CondIdx1:** | Index of the first Condition to consider for the Transition |
| **CondIdx2:** | Index of the second Condition to consider for the Transition |
| **AccLim:** | Acceleration limit for the Transition { unit: [pu/s^2], value range: [0.001, 120.0] } |
| **JrkLim:** | Jerk limit for the Transition { unit: [pu/s^3], value range: [0.0005, 2000.0] } |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgAct()** | |
|  | **void STVELPLAN\_getCfgAct(ST\_VELPLAN\_Handle handle, uint16\_t ActIdx, uint16\_t \*State, ST\_PlanCond\_e \*AndOr, uint16\_t \*CondIdx1, uint16\_t \*CondIdx2, uint16\_t \*VarIdx, ST\_PlanActOptn\_e \*Opt, \_iq24 \*Value, ST\_PlanActTrgr\_e \*EnterExit)** |
|  | Returns an action from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **ActIdx:** | Index of the Action to modify |
| **State:** | Index of the State the Action happens in |
| **AndOr:** | Condition option for the Action |
| **CondIdx1:** | Index of the first Condition to consider for the Action |
| **CondIdx2:** | Index of the second Condition to consider for the Action |
| **VarIdx:** | Index of the Varaible to be acted upon |
| **Opt:** | Type of Action to perform on the Varaible |
| **Value:** | Value to use in the Action |
| **EnterExit:** | State event that will trigger the Action |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgVar()** | |
|  | **void STVELPLAN\_getCfgVar(ST\_VELPLAN\_Handle handle, uint16\_t VarIdx, ST\_PlanVar\_e \*Type, \_iq24 \*Value)** |
|  | Returns a variable from the SpinTAC Velocity Plan configuration |
| **Handle:** | handle The handle for the SpinTAC Velocity Plan Object |
| **VarIdx:** | VarIdx Index of the Variable to return |
| **Type:** | Type of Variable |
| **Value:** | Initial value of the Variable |

|  |  |
| --- | --- |
| **STVELPLAN\_getVar()** | |
|  | **void STVELPLAN\_getVar(ST\_VELPLAN\_Handle handle, uint16\_t VarIdx, \_iq24 \*Value)** |
|  | Gets the value of a SpinTAC Velocity Plan variable |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VarIdx:** | Index of the Variable to get |
| **Value:** | Value to get from the Variable |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfg()** | |
|  | **void STVELPLAN\_getCfg(ST\_VELPLAN\_Handle handle, \_iq24 \*T\_sec, bool \*LoopENB)** |
|  | Gets the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **T\_sec:** | Sample Time { unit: [sec], value range: (0, 0.01] } |
| **LoopENB:** | Sets if SpinTAC Velocity Plan should continuously run after it reaches the end { false: Do not continuously run Plan; true: Continuously run Plan } |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgHaltState()** | |
|  | **void STVELPLAN\_getCfgHaltState(ST\_VELPLAN\_Handle handle, \_iq24 \*VelEnd, \_iq24 \*AccLim, \_iq20 \*JrkLim, int32\_t \*Timer\_tick)** |
|  | Gets the SpinTAC Velocity Plan Halt state configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VelEnd:** | Velocity setpoint during Halt state { unit: [pu/s], value range: [-1.0, 1.0] } |
| **AccLim:** | AccLim Acceleration limit for Halt state { unit: [pu/s^2], value range: [0.001, 120.0] } |
| **JrkLim:** | Jerk limit for Halt state { unit: [pu/s^3], value range: [0.0005, 2000.0] } |
| **Timer\_tick:** | Minimum amount of time to stay in Halt state { unit: [tick] } |

|  |  |
| --- | --- |
| **STVELPLAN\_getCfgState()** | |
|  | **void STVELPLAN\_getCfgState(ST\_VELPLAN\_Handle handle, uint16\_t StateIdx, \_iq24 \*VelEnd, int32\_t \*Timer\_tick)** |
|  | Returns a state from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **StateIdx:** | Index of the State to modify |
| **VelEnd:** | Velocity setpoint for State { unit: [pu/s], value range: [-1.0, 1.0] } |
| **Timer\_tick:** | Minimum time to stay in State { unit: [tick] } |

**STVELPLAN Set Function**

|  |  |
| --- | --- |
| **STVELPLAN\_setEnable()** | |
|  | **static inline void STVELPLAN\_setEnable(ST\_VELPLAN\_Handle handle, bool enb)** |
|  | Sets the Enable signal (ENB) for SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **enb:** | Enable bit { false: disable; true: enable } |

|  |  |
| --- | --- |
| **STVELPLAN\_setReset()** | |
|  | **static inline void STVELPLAN\_setReset(ST\_VELPLAN\_Handle handle, bool res)** |
|  | Sets the Reset signal (RES) for SpinTAC Velocity Plan |
| Handle: | The handle for the SpinTAC Velocity Plan Object |
| res: | Reset bit { false: reset; true: not reset } |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfgCond()** | |
|  | **void STVELPLAN\_setCfgCond(ST\_VELPLAN\_Handle handle, uint16\_t CondIdx, uint16\_t VarIdx, ST\_PlanComp\_e Comp, \_iq24 Value1, \_iq24 Value2)** |
|  | Modifies a Condition in the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **CondIdx:** | Index of the Plan Condition to modify |
| **VarIdx:** | Index of the Plan Variable that will be compared |
| **Comp:** | Type of comparison that will be done |
| **Value1:** | First value to use in the comparison |
| **Value2:** | Second value to use in the comparison |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfgVarCond()** | |
|  | **void STVELPLAN\_setCfgVarCond(ST\_VELPLAN\_Handle handle, uint16\_t CondIdx, uint16\_t VarIdx1, uint16\_t VarIdx2, ST\_PlanComp\_e Comp)** |
|  | Modifies a Varaible Condition to the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **CondIdx:** | Index of the Plan Condition to modify |
| **VarIdx1:** | Index of the first Plan Variable that will be compared |
| **VarIdx2:** | Index of the second Plan Variable that will be compared |
| **Comp:** | Type of comparison that will be done |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfgTran()** | |
|  | **void STVELPLAN\_setCfgTran(ST\_VELPLAN\_Handle handle, uint16\_t TranIdx, uint16\_t FromState, uint16\_t ToState, ST\_PlanCond\_e AndOr, uint16\_t CondIdx1, uint16\_t CondIdx2, \_iq24 AccLim, \_iq20 JrkLim)** |
|  | Modifies a Transition in the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **TranIdx:** | Index of the Transition to modify |
| **FromState:** | Index of the from State |
| **ToState:** | Index of the to State |
| **AndOr:** | Condition option for the Transition |
| **CondIdx1:** | Index of the first Condition to consider for the Transition |
| **CondIdx2:** | Index of the second Condition to consider for the Transition |
| **AccLim:** | Acceleration limit for the Transition { unit: [pu/s^2], value range: [0.001, 120.0] } |
| **JrkLim:** | Jerk limit for the Transition { unit: [pu/s^3], value range: [0.0005, 2000.0] } |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfgAct()** | |
|  | **void STVELPLAN\_setCfgAct(ST\_VELPLAN\_Handle handle, uint16\_t ActIdx, uint16\_t State, ST\_PlanCond\_e AndOr, uint16\_t CondIdx1, uint16\_t CondIdx2, uint16\_t VarIdx, ST\_PlanActOptn\_e Opt, \_iq24 Value, ST\_PlanActTrgr\_e EnterExit)** |
|  | Modifies an action in the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object\ |
| **ActIdx:** | Index of the Action to modify |
| **State:** | Index of the State the Action happens in |
| **AndOr:** | Condition option for the Action |
| **CondIdx1:** | Index of the first Condition to consider for the Action |
| **CondIdx1:** | Index of the second Condition to consider for the Action |
| **VarIdx:** | Index of the Varaible to be acted upon |
| **Opt:** | Type of Action to perform on the Varaible |
| **Value:** | Value to use in the Action |
| **EnterExit:** | State event that will trigger the Action |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfgVar()** | |
|  | **void STVELPLAN\_setCfgVar(ST\_VELPLAN\_Handle handle, uint16\_t VarIdx, ST\_PlanVar\_e Type, \_iq24 Value)** |
|  | Modifies a variable in the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VarIdx:** | Index of the Variable to modify |
| **Type:** | Type of Variable |
| **Value:** | Initial value of the Variable |

|  |  |
| --- | --- |
| **STVELPLAN\_setVar()** | |
|  | **void STVELPLAN\_setVar(ST\_VELPLAN\_Handle handle, uint16\_t VarIdx, \_iq24 Value)** |
|  | Sets the value of a SpinTAC Velocity Plan variable |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VarIdx:** | Index of the Variable to set |
| **Value:** | Value to set to the Variable |

|  |  |
| --- | --- |
| **STVELPLAN\_setUnitProfDone()** | |
|  | **void STVELPLAN\_setUnitProfDone(ST\_VELPLAN\_Handle handle, bool ProDON)** |
|  | This should be done in the main ISR. This function needs to be called when SpinTAC Velocity Move has completed a profile. This is to alert SpinTAC Velocity Plan that we have reached the goal speed that it provided to SpinTAC Velocity Move. This should be placed after the function call for SpinTAC Velocity Move  Sets a flag in SpinTAC Velocity Plan to indicate if the unit profile is complete |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **ProDON:** | Indicates if the unit profile is completed |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfg()** | |
|  | **void STVELPLAN\_setCfg(ST\_VELPLAN\_Handle handle, \_iq24 T\_sec, bool LoopENB)** |
|  | Configures SpinTAC Velocity Plan |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **T\_sec:** | Sample Time { unit: [sec], value range: (0, 0.01] } |
| **LoopENB:** | Sets if SpinTAC Velocity Plan should continuously run after it reaches the end { false: Do not continuously run Plan; true: Continuously run Plan } |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfgHaltState()** | |
|  | **void STVELPLAN\_setCfgHaltState(ST\_VELPLAN\_Handle handle, \_iq24 VelEnd, \_iq24 AccLim, \_iq20 JrkLim, int32\_t Timer\_tick)** |
|  | Configures the SpinTAC Velocity Plan Halt state |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VelEnd:** | Velocity setpoint during Halt state { unit: [pu/s], value range: [-1.0, 1.0] } |
| **AccLim:** | Acceleration limit for Halt state { unit: [pu/s^2], value range: [0.001, 120.0] } |
| **JrkLim:** | Jerk limit for Halt state { unit: [pu/s^3], value range: [0.0005, 2000.0] } |
| **Timer\_tick:** | Minimum amount of time to stay in Halt state { unit: [tick] } |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfgState()** | |
|  | **void STVELPLAN\_setCfgState(ST\_VELPLAN\_Handle handle, uint16\_t StateIdx, \_iq24 VelEnd, int32\_t Timer\_tick)** |
|  | Modifies a state in the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **StateIdx:** | Index of the State to modify |
| **VelEnd:** | Velocity setpoint for State { unit: [pu/s], value range: [-1.0, 1.0] } |
| **Timer\_tick:** | Minimum time to stay in State { unit: [tick] } |

|  |  |
| --- | --- |
| **STVELPLAN\_setCfgArray()** | |
|  | **void STVELPLAN\_setCfgArray(ST\_VELPLAN\_Handle handle, uint32\_t \*cfgArray, const size\_t numBytes, uint16\_t MaxActNum,uint16\_t MaxCondNum, uint16\_t MaxVarNum, uint16\_t MaxTranNum, uint16\_t MaxStateNum)** |
|  | Prepares SpinTAC Velocity Plan configuration array |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **cfgArray:** | Pointer to the SpinTAC Velocity Plan configuration array |
| **numBytes:** | The number of bytes in the SpinTAC Velocity Plan configuration array |
| **MaxActNum:** | Number of Actions |
| **MaxCondNum:** | Number of Conditions |
| **MaxVarNum:** | Number of Variables |
| **MaxTranNum:** | Number of Transitions |
| **MaxStateNum:** | Number of States |

**STVELPLAN Add Function**

|  |  |
| --- | --- |
| **STVELPLAN\_addCfgCond()** | |
|  | **void STVELPLAN\_addCfgCond(ST\_VELPLAN\_Handle handle, uint16\_t VarIdx, ST\_PlanComp\_e Comp, \_iq24 Value1, \_iq24 Value2)** |
|  | Adds a Condition to the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VarIdx:** | Index of the Plan Variable that will be compared |
| **Comp:** | Type of comparison that will be done |
| **Value1:** | First value to use in the comparison |
| **Value2:** | Second value to use in the comparison |

|  |  |
| --- | --- |
| **STVELPLAN\_addCfgVarCond()** | |
|  | **void STVELPLAN\_addCfgVarCond(ST\_VELPLAN\_Handle handle, uint16\_t VarIdx1, uint16\_t VarIdx2, ST\_PlanComp\_e Comp)** |
|  | Adds a Variable Condition to the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VarIdx1:** | Index of the first Plan Variable that will be compared |
| **VarIdx2:** | Index of the second Plan Variable that will be compared |
| **Comp:** | Type of comparison that will be done |

|  |  |
| --- | --- |
| **STVELPLAN\_addCfgTran()** | |
|  | **void STVELPLAN\_addCfgTran(ST\_VELPLAN\_Handle handle, uint16\_t FromState, uint16\_t ToState, ST\_PlanCond\_e AndOr, uint16\_t CondIdx1, uint16\_t CondIdx2, \_iq24 AccLim, \_iq20 JrkLim)** |
|  | Adds a Transition to the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **FromState:** | Index of the from State |
| **ToState:** | Index of the to State |
| **AndOr:** | Condition option for the Transition |
| **CondIdx1:** | Index of the first Condition to consider for the Transition |
| **CondIdx2:** | Index of the second Condition to consider for the Transition |
| **AccLim:** | Acceleration limit for the Transition { unit: [pu/s^2], value range: [0.001, 120.0] } |
| **JrkLim:** | Jerk limit for the Transition { unit: [pu/s^3], value range: [0.0005, 2000.0] } |

|  |  |
| --- | --- |
| **STVELPLAN\_addCfgAct()** | |
|  | **void STVELPLAN\_addCfgAct(ST\_VELPLAN\_Handle handle, uint16\_t State, ST\_PlanCond\_e AndOr, uint16\_t CondIdx1, uint16\_t CondIdx2, uint16\_t VarIdx, ST\_PlanActOptn\_e Opt, \_iq24 Value, ST\_PlanActTrgr\_e EnterExit)** |
|  | Adds an action to the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **State:** | Index of the State the Action happens in |
| **AndOr:** | Condition option for the Action |
| **CondIdx1:** | Index of the first Condition to consider for the Action |
| **CondIdx2:** | Index of the second Condition to consider for the Action |
| **VarIdx:** | Index of the Varaible to be acted upon |
| **Opt:** | Type of Action to perform on the Varaible |
| **Value:** | Value to use in the Action |
| **EnterExit:** | State event that will trigger the Action |

|  |  |
| --- | --- |
| **STVELPLAN\_addCfgVar()** | |
|  | **void STVELPLAN\_addCfgVar(ST\_VELPLAN\_Handle handle, ST\_PlanVar\_e Type, \_iq24 Value)** |
|  | Adds a variable to the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **Type:** | Type of Variable |
| **Value:** | Initial value of the Variable |

|  |  |
| --- | --- |
| **STVELPLAN\_addCfgState()** | |
|  | **void STVELPLAN\_addCfgState(ST\_VELPLAN\_Handle handle, \_iq24 VelEnd, int32\_t Timer\_tick)** |
|  | Adds a State to the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VelEnd:** | Velocity setpoint for State { unit: [pu/s], value range: [-1.0, 1.0] } |
| **Timer\_tick:** | Minimum time to stay in State { unit: [tick] } |

**STVELPLAN Del Function**

|  |  |
| --- | --- |
| **STVELPLAN\_delCfgCond()** | |
|  | **void STVELPLAN\_delCfgCond(ST\_VELPLAN\_Handle handle, uint16\_t CondIdx)** |
|  | Deletes a Condition from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **CondIdx:** | Index of the Plan Condition to delete |

|  |  |
| --- | --- |
| **STVELPLAN\_delCfgVarCond()** | |
|  | **void STVELPLAN\_delCfgVarCond(ST\_VELPLAN\_Handle handle, uint16\_t CondIdx)** |
|  | Deletes a Variable Condition from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **CondIdx:** | Index of the Plan Condition to delete |

|  |  |
| --- | --- |
| **STVELPLAN\_delCfgTran()** | |
|  | **void STVELPLAN\_delCfgTran(ST\_VELPLAN\_Handle handle, uint16\_t TranIdx)** |
|  | Deletes a Transition from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **TranIdx:** | Index of the Transition to delete |

|  |  |
| --- | --- |
| **STVELPLAN\_delCfgAct()** | |
|  | **void STVELPLAN\_delCfgAct(ST\_VELPLAN\_Handle handle, uint16\_t ActIdx)** |
|  | Deletes an action from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **ActIdx:** | Index of the Action to delete |

|  |  |
| --- | --- |
| **STVELPLAN\_delCfgVar()** | |
|  | **void STVELPLAN\_delCfgVar(ST\_VELPLAN\_Handle handle, uint16\_t VarIdx)** |
|  | Deletes a Variable from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **VarIdx:** | Index of the Variable to delete |

|  |  |
| --- | --- |
| **STVELPLAN\_delCfgState()** | |
|  | **void STVELPLAN\_delCfgState(ST\_VELPLAN\_Handle handle, uint16\_t StateIdx)** |
|  | Deletes a state from the SpinTAC Velocity Plan configuration |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |
| **StateIdx:** | Index of the State to delete |

**STVELPLAN Reset Function**

|  |  |
| --- | --- |
| **STVELPLAN\_reset()** | |
|  | **void STVELPLAN\_reset(ST\_VELPLAN\_Handle handle)** |
|  | Resets the Velocity Plan component |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |

**STVELPLAN Run Function**

|  |  |
| --- | --- |
| **STVELPLAN\_run()** | |
|  | **void STVELPLAN\_run(ST\_VELPLAN\_Handle handle)** |
|  | The primary function is STVELPLAN\_run(ST\_VELPLAN\_Handle handle), where handle is a pointer to a specific ST\_VelPlan\_t object. This function runs SpinTAC Velocity Plan. This function can be run in the main loop of the program  Runs SpinTAC Velocity Plan calculation |
| **Handle:** | The handle for the Velocity Plan structure |

|  |  |
| --- | --- |
| **STVELPLAN\_runTick()** | |
|  | **void STVELPLAN\_runTick(ST\_VELPLAN\_Handle handle)** |
|  | The function for SpinTAC Velocity Plan that handles decrementing the state timer is STVELPLAN\_runTick(ST\_VELPLAN\_Handle handle), where handle is a pointer to a specific ST\_VelPlan\_t object. This function decrements the state times for SpinTAC Velocity Plan. This function must be run at the same rate as SpinTAC Velocity Control.  Updates the SpinTAC Velocity Plan Timer |
| **Handle:** | The handle for the SpinTAC Velocity Plan Object |

**STVELPLAN Init Function**

|  |  |
| --- | --- |
| **STVELPLAN\_init()** | |
|  | **ST\_VELPLAN\_Handle STVELPLAN\_init(void \*pMemory, const size\_t numBytes)** |
|  | Initializes the SpinTAC Velocity Plan component |
| **pMemory:** | Pointer to the memory for ST\_VelPlan\_t |
| **numBytes:** | The number of bytes in the ST\_VelPlan\_t |
| **Return:** | ST\_VELPLAN\_Handle handle The handle for the SpinTAC Velocity Plan Object |

**Planning Sequence**

It is important that the configuration of SpinTAC Velocity Plan is done in this order. If the configuration is not done in this order it could cause a configuration error

* States
* Variables
* Conditions
* Transitions
* Actions

**States**

States describe the steady operation of the profile. The user specifies the end speed (SpinTAC Velocity Plan) or position step (SpinTAC Position Plan) and the minimum time that SpinTAC Velocity Plan should remain in a state before transitioning to another state.

***STVELPLAN\_addCfgState(Velocity Plan Handle, Speed Setpoint [pu/s], Time in State [ISR ticks])***

This function adds a state into the motion sequence. It is configured by setting the speed that you want the motor to run during this state and with the minimum time it should remain in this state.

**Variables**

Variables allow SpinTAC Velocity Plan to interact with the rest of the project. There are three types of variables in SpinTAC Velocity Plan: input, output, and input-output. Input variables are used to receive values from outside SpinTAC Velocity Plan, and to evaluate conditions. Output variables are used to interact with the rest of the system. Output variables can be modified by SpinTAC Velocity Plan, but will not be used by SpinTAC Velocity Plan to check conditions. Note: The user must write the code that performs the event associated with the output variable. Input-output variables are typically used as counters or timers and are used by actions or conditions.

***STVELPLAN\_addCfgVar(Velocity Plan Handle, Variable Type, Initial Value)***

This function establishes a variable that will be used in the motion sequence. The variable type determines how SpinTAC™ Velocity Plan can use this variable. The initial value is the value that should be loaded into this variable initially. The variable can be up to a 32-bit value.

**Conditions**

Conditions provide logical checks within transitions or action. A transition or action may have a maximum of two conditions. The condition(s) must be satisfied before the motor can transition from one state to the next or for the action to occur. To determine whether a condition is satisfied, a variable is compared against a specific value or value range. This returns a true or false value based on the criteria.

***STVELPLAN\_addCfgCond(Velocity Plan Handle, Variable Index, Comparison, Comparison Value 1, Comparison Value 2)***

This function sets up a condition to be used in the motion sequence. This will be a fixed comparison of a variable against a value or value range. The variable index describes which variable should be compared. The comparison should be used to describe the type of comparison to be done. Comparison values 1 & 2 are used to establish the bounds of the comparison. If a comparison only requires one value it should be set in comparison value 1 and comparison value 2 should be set to 0.

**Transition**

Transitions define the allowable moves between states. They establish the connections between the states. Transitions allow the move between states to occur if a condition has been fulfilled. The user specifies the initial and target states, the profile limits, and the conditions to be evaluated prior to the transition.

***STVELPLAN\_addCfgTran(Velocity Plan Handle, From State, To State, Condition Option, Condition Index 1, Condition Index 2, Acceleration Limit [pu/s^2], Jerk Limit [pu/s^3])***

This function establishes the transitions between two states. The From State and To State values describe which states this transition is valid for. The condition option specifies if a condition needs to be evaluated prior to the transition. The condition index 1 & 2 specify which conditions should be evaluated. If no conditions or one condition needs to be evaluated, set the not used values to 0. Acceleration limit sets the acceleration to use to transition between the From State speed and the To State speed. This value cannot exceed the acceleration max that is configured for the motion sequence. The jerk limit sets the jerk to be used in the speed transition. This value should not exceed the jerk max that is configured for the motion sequence.

**Actions**

Actions change the value of variables. Actions set a variable equal to a value, or add a value to a variable. Actions may take place within a specified state, or when SpinTAC VelocityPlan enters or exits that state. Actions may have associated conditions. This allows an action to occur only when the condition is satisfied. If an action is configured as an ENTER action, SpinTAC Plan will start evaluating the conditions of that action upon entering the state. The action will take place once, only after the conditions are satisfied. Similarly, if an action is configured as an EXIT action, SpinTAC Plan will start evaluating the conditions of that action when leaving the state. When the conditions are satisfied, the actions take place once.

***STVELPLAN\_addCfgAct(Velocity Plan Handle, State Index, Condition Option, Condition Index 1, Condition Index 2, Variable Index, Operation, Value, Action Trigger)***

This function adds an action into the motion sequence. The state index describes which state the action should take place in. The condition option specifies if a condition needs to be evaluated prior to the action. The condition index 1 & 2 specify which conditions should be evaluated. If no conditions or one condition needs to be evaluated, set the not used values to 0. The variable index indicates which variable the action should be done to. The operation determines what operation should be done to the variable, the only available options are to add a value or set a value. The value is what should be added or set to the variable. The action trigger indicates if the action should be performed when entering or exiting the state.

**Velocity Plan Element Limits**

SpinTAC Velocity Plan does not have a maximum number of elements that can be configured. The limit is based on the amount of memory that you would like to commit to the configuration of SpinTAC Velocity Plan. This is done in order to be both efficient as to how SpinTAC Velocity Plan is using the system memory and be flexible to allow for custom configurations of the element. Each element of SpinTAC Velocity Plan has a different memory foot print.

This additional flexibility requires you to declare a configuration array whose address needs to be passed into SpinTAC Velocity Plan. This configuration array needs to be sized according to how many elements are in your Plan. It is a best practice to declare enumerations for the Plan elements that you wish to use. This makes it simple to calculate the amount of memory that is required for the configuration array.

|  |  |
| --- | --- |
| **Plan Element** | **SpinTAC Velocity Plan (double words)** |
| States | 6 |
| Variables | 2 |
| Conditions | 3 |
| Transitions | 5 |
| Actions | 5 |
| **Table: Memory Requirements for SpinTAC Velocity Plan Elements** | |

**Troubleshooting SpinTAC Velocity Plan**

**ERR\_ID**

ERR\_ID provides an error code for users to identify the specific SpinTAC Velocity Plan function that caused the error. A list of ERR\_IDs defined in SpinTAC Velocity Plan

|  |  |
| --- | --- |
| **ERR\_ID** | **Plan Function** |
| 3000 | STVELPLAN\_addCfgCond |
| 3001 | STVELPLAN\_delCfgCond |
| 3002 | STVELPLAN\_setCfgCond |
| 3003 | STVELPLAN\_getCfgCond |
| 3004 | STVELPLAN\_addCfgTran |
| 3005 | STVELPLAN\_delCfgTran |
| 3006 | STVELPLAN\_setCfgTran |
| 3007 | STVELPLAN\_getCfgTran |
| 3008 | STVELPLAN\_addCfgAct |
| 3009 | STVELPLAN\_delCfgAct |
| 3010 | STVELPLAN\_setCfgAct |
| 3011 | STVELPLAN\_getCfgAct |
| 3012 | STVELPLAN\_addCfgVar |
| 3013 | STVELPLAN\_delCfgVar |
| 3014 | STVELPLAN\_setCfgVar |
| 3015 | STVELPLAN\_getCfgVar |
| 3016 | STVELPLAN\_addCfgState |
| 3017 | STVELPLAN\_delCfgState |
| 3018 | STVELPLAN\_setCfgState |
| 3019 | STVELPLAN\_setVar |
| 3020 | STVELPLAN\_getVar |
| 3021 | STVELPLAN\_setCfg |
| 3022 | STVELPLAN\_setCfgHaltState |
| 3023 | STVELPLAN\_setCfgArray |
| 3024 | STVELPLAN\_addCfgVarCond |
| 3025 | STVELPLAN\_delCfgVarCond |
| 3026 | STVELPLAN\_setCfgVarCond |
| 3027 | STVELPLAN\_getCfgVarCond |
| 4001 | STVELPLAN\_run (Invalid SpinTAC license. Use the chip with valid license for SpinTAC.) |
| 4003 | STVELPLAN\_run (Invalid ROM version. Use a chip with a valid ROM version or use the SpinTAC library that is compatible with the current ROM version.) |

**Configuration Errors**

The configuration errors are reported via the CfgError structure included in the main SpinTAC Velocity Plan structure. This structure contains elements that store additional information about the error. The elements are described below:

* + - * CfgError.ERR\_idx: Identifies the instance of configured element at which the error occurred.
      * CfgError.ERR\_code: Identifies the specific error condition that caused the error.

The ERR\_code for a specific condition remains the same for all Plan functions. A list of ERR\_codes and conditions defined in SpinTAC Velocity Plan.

|  |  |  |
| --- | --- | --- |
| **ERR\_code** | **Description** | **Solution** |
| 1 | SpinTAC Plan is running | Place SpinTAC Plan into the idle status prior to running the configuration. |
| 2 | Maximum State number exceeded | The maximum number of States has been configured. |
| 3 | Maximum Condition number exceeded | The maximum number of Conditions has been configured. |
| 4 | Maximum Transition number exceeded | The maximum number of Transitions has been configured. |
| 5 | Maximum Action number exceeded | The maximum number of Actions has been configured. |
| 6 | Maximum Variable number exceeded | The maximum number of Variables has been configured. |
| 7 | Invalid sample time value | Set sample time, cfg.T\_sec, within (0, 0.01]. |
| 8 | Invalid VelMax value | Choose VelMax within (0, 1]. |
| 9 | Invalid AccMax value | Choose AccMax within [0.001, 120]. |
| 10 | Invalid JrkMax value | Choose JrkMax within [0.0005, 2000]. |
| 11 | Invalid LoopENB value | Choose LoopENB within { false, true }. |
| 12 | Invalid VelEnd value | Choose VelEnd within [(0, VelMax]. |
| 13 | Invalid AccLim value | Choose AccLim within [0.001, AccMax]. |
| 14 | Invalid JrkLim value | Choose JrkLim within [0.0005, JrkMax]. |
| 15 | Invalid Timer\_tick value | Choose a positive integer value. |
| 16 | Invalid State index | The index should be for a configured State index. |
| 17 | Invalid Condition index | The index should be for a configured Condition index. |
| 18 | Invalid Transition index | The index should be for a configured Transition index. |
| 19 | Invalid Action index | The index should be for a configured Action index. |
| 20 | Invalid Variable index | The index should be for a configured Variable index. |
| 21 | Invalid Variable type | Choose variable type from the values in ST\_PlanVar\_e. |
| 22 | Invalid value of Comparison | Choose comparison from the values in ST\_PlanComp\_e. |
| 23 | Invalid Operation | Choose operation from the values in ST\_PlanActOptn\_e. |
| 24 | Invalid AndOr value | Choose AndOr from the values in ST\_PlanCond\_e. |
| 25 | Improper Variable type | ST\_VAR\_OUT Variables cannot have a value set to them. ST\_VAR\_OUT Variables cannot be used in Conditions.  ST\_VAR\_IN Variables cannot be used in Actions. |
| 26 | Improper values in Comparison | Value1 should be less than or equal to Value2. |
| 27 | Improper State index | In Transitions FromState cannot be equal to ToState, and these States must be equal to a configured State. |
| 28 | Improper Condition index in Transition | In Transitions: CondIdx1 cannot be equal to CondIdx2, and these Conditions must be equal to a configured Condition |
| 29 | Improper EnterExit value | Choose EnterExit from the values in ST\_PlanActTrgr\_e |
| 30 | Improper AndOr during Variable deletion | The AndOr value conflicts with the value of VarIdx. When deleting a Variable, it causes a configuration error in a Transition. |
| 31 | Cannot delete Variable as an Action depends on it | Remove Variable from Action configuration before deleting the Variable. |
| 37 | Plan Configuration array declared is too small for plan elements | Remove an Element from the configuration or declare a larger configuration array. |
| 38 | Cannot delete a State as a Transition depends on it | Remove State from Transition configuration before deleting the State. |
| 39 | Cannot delete a State as an Action depends on it | Remove State from Action configuration before deleting the State. |
| 40 | Improper values for variable comparison | Variable comparison conditions cannot have comparison enum greater than ST\_COMP\_ELW. |
| 41 | Cannot compare a variable to itself | Ensure that the variable indexes passed to the function are different and valid. |
| 42 | Cannot get a variable based Condition from the index of a value based type of Condition | Pass an index that is known to contain a variable based Condition. |
| 43 | Cannot delete a Condition as a Transition depends on it | Remove Condition from Transition configuration before deleting the Condition. |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*