softMC Training – Module 7

Motion

Topics

- Motion Elements
- Single Axis
- Groups
- Enabling Motion
- Element Properties
- System Properties
- Modal/Nodal Properties
- Motion Types
- Motion Commands
- Motion Blending

MOTION ELEMENTS

Motion Elements (also called Elements)

- Motion element is any object on which a motion command can be executed
- Motion element refers to both axes and groups.

- Axis – a single motion axis (drive + motor and associated properties)
- Group – a collection of axes

- Motion elements are defined in Config.prg or Terminal

Element Properties

- Properties are used primarily for defining the characteristics of the motion
- Axis Properties
- Group Properties
- Many property names (and commands) are used for both axes and groups.

Element Properties

- Syntax

```
<element>.<property> = <value>
```

- Examples (Properties)

```
Axis.Acceleration = 100
Joint.Jerk = 1000
Robot.Deceleration = 500
```

Element Properties

- Example – Axis Motion

```
A1.Acceleration = 100
A1.VelocityCruise = 1000
A1.Deceleration = 50
Move A1 1567
         ’1567 is the target position
```

- Example – Group Motion

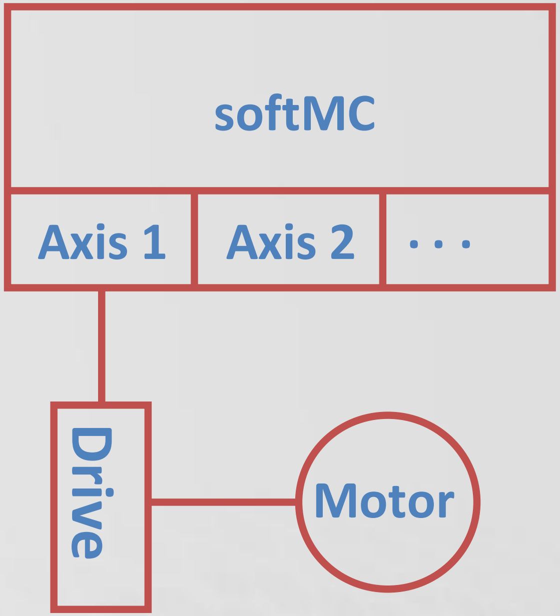
```
MyGroup.Acceleration = 100
MyGroup.VelocityCruise = 1000
MyGroup.Deceleration = 50
Move MyGroup {10,20,35,50}
      `target positions of each axis

         `in the group
```

SINGLE AXIS

What is an Axis?

- What is an axis?
  - Motor
  - Drive
  - Components of softMC



Defining Axes

- System default axis names: A1, A2, A3 ...
- Recommended to give axes meaningful names , using **<axis>.AxisName** property
- Axes are defined/renamed in config.prg
- Must be the first command, before Program
- Syntax

```
Sys.Naxes = <number>
…
Program
…
End Program
```

- Example

```
Sys.Naxes = 2
…
Program
A1.AxisName = Xaxis
A2.AxisName = Lift
…
End Program
```

Axis Setup Procedure

- Axis setup procedure detailed in wiki: http://softmc.servotronix.com/wiki/Axis_Setup_Procedure

Properties - Axis Setup Subroutine

```
sub SetRotAxis(ax as generic axis, byval minval
   as double , byval maxval as double)
  with ax
    attach
      En = 0
      AxisType = 1
      PositionFactor = pos_unit/360
      VelocityFactor = PositionFactor /1000
      AccelerationFactor = VelocityFactor /1000
      Jerkfactor = AccelerationFactor /1000
      VelocityMax = 1000
      AccelerationMax = 10000
      DecelerationMax= 10000
      JerkMax = 20*amax
      VelocityCruise = 0.5*VelocityMax
      Acceleration = AccelerationMax
      Deceleration = DecelerationMax
      Jerk = JerkMax
      PrfType = -1
      Smooth = -1
      VelocityOverspeed = 1.2*VelocityMax
      VelocitySafetyLimit = 10*VelocityMax
      PositionErrorDelay = 2
      PositionErrorMax  = 1
      PositionMax = maxval
      PositionMin = minval
      PositionMaxEn = 1
      PositionMinEn = 1
      PositionRolloverEnable = 0
    detach
  end with
end sub
```

GROUPS

What Is A Group?

- A group is a combination of axes that are setup to work together.
  - XYTable (A1, A2)
  - Robot (A1, A2, A3)
- Groups enable control of multiple axes as a single mechanism.

- No limit to number of axes in a group.
- All axes in the system may be part of one group.
- One axis can belong to several groups but only one group can be active (attached) at a time.

Create a Group

- A group can be defined only in Config.prg or Terminal
- To set up a Group, first set up each of the axes that will be in the group.
- Then, form the group by using the command **Common Shared ... as Group**

- Syntax

```
Common Shared <group> as Group AxisName = <axis> {AxisName = <Axis>}
```

- Example

```
Common Shared MyGroup as Group AxisName = A1  AxisName = A2
```

Group Properties

- Once a goup is defined, group properties can be set
- Properties of a group are set like properties of a single axis

Properties – Group Setup Subroutine

```
sub SetGroup(gr as generic group)
  dim i as long
  with gr
    attach
      en = 0
      sleep 10
      vfac = 1/1000
      afac = vfac/1000
      jfac = afac/1000
      positionerrorsettle = 2
      pemax  = 1
      abs = 1

      a1.axistype = 0
      a2.axistype = 0
      a3.axistype = 0

      configgroup

      vmax = 1000 '  mm/sec
      amax = 1500 '  1500 mm/sec^2
      dmax = 1500 '  1500 mm/sec^2
      jmax = omega*amax
      vcruise = vmax
      acc = amax
      dec = dmax
      jerk = jmax

      vmtran = vmax
      amtran = amax
      jmtran = jmax
      vtran = vmtran
      atran = amtran
      dtran = amtran
      jtran = jmtran

      vmrot = vmax
      amrot = amax
      jmrot = jmax
      vrot = vmrot
      arot = amrot
      drot = amrot
      jrot = jmrot

      xmax = 300
      xmin = 0

      ymax = 300
      ymin = 0

      zmax =  300
      zmin =  0

      rmin = 0
      rmax = 4000

      smooth = -1
      prftype = profile

      j1.pmin = 0
      a1.pmin = 0

      j2.pmin = 0
      a2.pmin = 0

      j3.pmin = 0
      a3.pmin = 0

      j1.pmax = 300
      a1.pmax = 300

      j2.pmax = 300
      a2.pmax = 300

      j3.pmax = 300
      a3.pmax = 300

      Print "group ";elementname;" set."
      detach
  end with
end sub
```

ENABLING MOTION

Prerequisites for Motion – Attach

- Attaching prevents other tasks from attempting to control the axis
- An axis can only be attached to one task at a time
- To **attach** the axis to the controlling task

```
Program
Attach <axis>
```

- To **detach** the axis:

```
Detach <axis>
```

- All axes are automatically detached at the end of a task
    - If Detach is issued while profile motion for the axis is in progress, Detach is not executed until that motion is complete

- Motion commands from the Terminal implicitly attach/detach motion elements
    - For example, Move command from Terminal implicitly attaches the element before the move and implicitly detaches it after the command has executed.
- Stop/Proceed mechanism temporarily takes control of the motion element

Prerequisites for Enabling Motion

- Physical drive
  - Drive enabled

- System
  - System enabled
  - System logical motion flag On

- Motion element
  - Element enabled
  - Element logical motion flag On

Prerequisites for Enabling Motion – Drive

- Remote enable must be configured
  - INMODE set to define use of remote enable
  - Remote Enable signal is not a perquisite for a motion bus system

- Software enable (signal from motion bus)
  - No faults
  - Software Enable = EtherCAT controlword

- STO – Safe Torque Off
  - An electrical signal to the drive. If removed, drive cannot be enabled.

Prerequisites for Enabling Motion – System

- System enabled
  - **System.Enable = 0** immediately disables all motion elements
  - **System.Enable = 1** allows all motion elements to be explicitly enabled

- Motion flags must be turned on
  - **System.Motion = 0** inhibits motion
  - **System.Motion = 1** allows motion

- Motion flag is cleared (set to 0) when an unhandled error occurs
- Motion flag can be set only from Terminal

Prerequisites for Enabling Motion – Element

- Element enabled
  - **<*element*>.Enable = 0** immediately disables the motion element
  - **<*element*>.Enable = 1** enables the motion element

- If <*element*> is a group, enabling the group enables all axes in the group

- Motion flags must be turned on
  - **<*element*>. Motion = 0** inhibits motion
  - **<*element*>. Motion = 1** allows motion

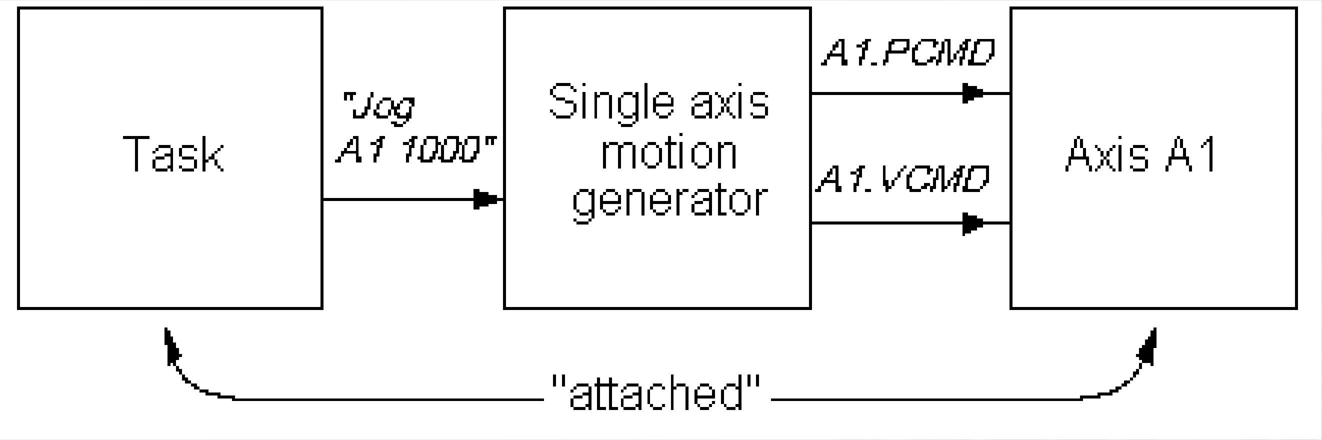
- Element motion flags are not affected by the error handler

3 Phases of Motion Generation

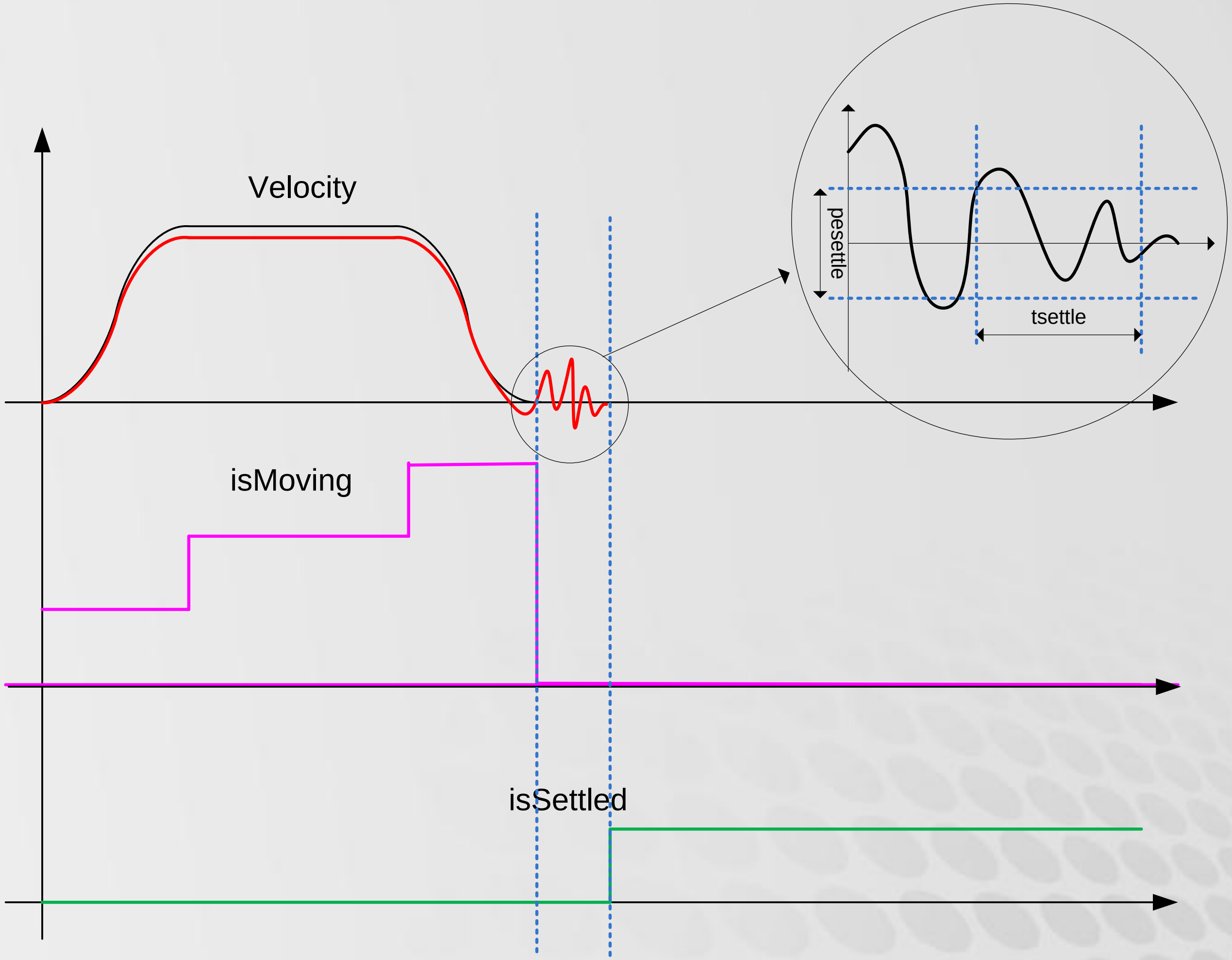
- Task issues an instruction for the movement.
- Motion manager task prepares the motion for execution.
- At every sample the next point of the path/trajectory is computed, and sent via motion bus to the drive.

- Example

```
      Move A1 1000 VelocityCruise = 10
```



End of Motion



MOTION ELEMENT PROPERTIES
by Category

- Applicable to all Motion Elements

Properties – Profile

Long Form	Short Form
<element>.**Acceleration**	*<element>*.ACC
<element>.**AccelerationMax**	*<element>*.AMAX
<element>.**AccelerationRate**	*<element>*.ARATE
<element>.**Deceleration**	*<element>*.DEC
<element>.**DecelerationMax**	*<element>*.DMAX
<element>.**DeclerationRate**	*<element>*.DRATE
<element>.**SmoothFactor**	*<element>*.SMOOTH
<element>.**VelocityCruise**	*<element>*.VCRUISE
<element>.**VelocityMax**	*<element>*.VMAX

Properties – Jerk

- Jerk can be active only if <*element*>.**smooth = -1**

Long Form

- <*element*>.**Jerk**
- <*element*>.**JerkMax**
- <*element*>.**JerkRate**

Short Form

- <*element*>.JERK*
- <*element*>.JMAX
- <*element*>.JRATE

- **same as long form*

Properties – Velocity Vector

Long Form

- *<element>*.**VelocityCommand**
- *<element>*.**VelocityFeedback**

Short Form

- *<element>*.VCMD
- *<element>*.VFB

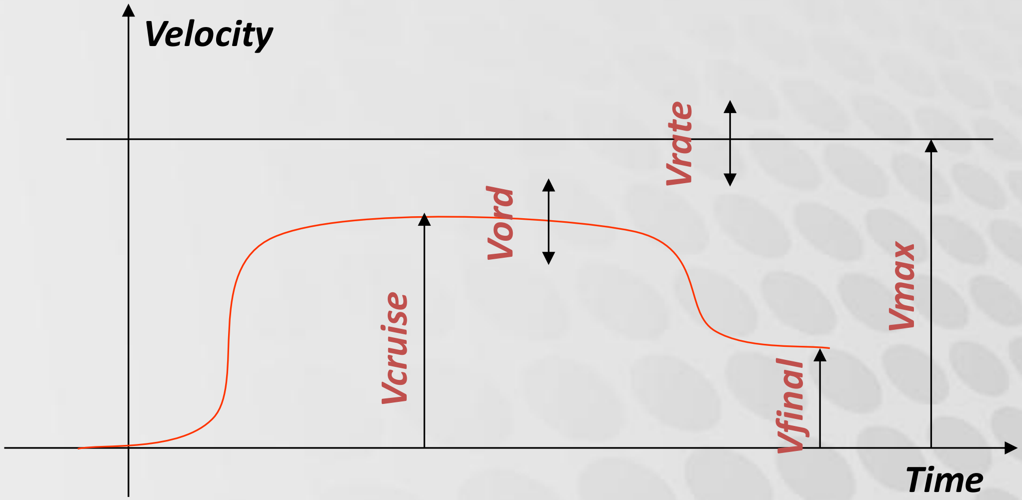
Properties – Velocity Scalar

Long Form

- *<element>*.**VelocityCruise**
- *<element>*.**VelocityFinal**
- *<element>*.**VelocityMax**
- *<element>*.**VelocityOverride**
- *<element>*.**VelocityRate**
- *<element>*.**VelocityOverSpeed**

Short Form

- *<element>*.VCRUISE
- *<element>*.VFINAL
- *<element>*.VMAX
- *<element>*.VORD
- *<element>*.VRATE
- *<element>*.VOSPD



Properties – Units

Long Form

- *<element>*.**VelocityFactor**
- *<element>*.**AccelerationFactor**
- *<element>*.**JerkFactor**

Short Form

- *<element>*.VFAC
- *<element>*.AFAC
- *<element>*.JFAC

Properties – Position Vector

Long Form

- *<element>*.**PositionCommand**
- *<element>*.**PositionFeedback**
- *<element>*.**PositionToGo**

Short Form and Examples

- *<group>*.PCMD

```
Move MyGroup {5.5,6.5}
```

- *<group>*.PFB

```
?MyGroup.PFB
{5.5,6.5}
```

- *<group>*.PTOGO

```
?MyGroup.PTOGO
{1.324,2.324}
```

Properties – Position Scalar

Long Form

- *<element>*.**PositionError**
- *<element>*.**PositionErrorSettle**
- *<element>*.**TimeSettle**
- *<element>*.**TimeSettleMax**
- *<element>*.**Absolute**

Short Form

- *<element>*.PE
- *<element>*.PESETTLE
- *<element>*.TSETTLE
- *<element>*.TSETTLEMAX
- *<element>*.ABS

MORE MOTION ELEMENT PROPERTIES

- Applicable to all Motion Elements

Properties – Motion Attributes

- **Absolute (Abs)** – specifies whether movement is absolute or incremental

```
<element>.absolute
      'absolute motion
<element>.absolute = 0                          'relative motion
```

- Syntax

```
<element>.ProfileType
```

- Examples

```
<element>.ProfileType = -1                       'sine acceleration;
   S-curve
<element>.ProfileType = 1                       'trapezoidal
   velocity
<element>.ProfileType = 2                       'trapezoidal
   acceleration; S-curve
```

Properties – Motion Monitoring

- <*element*>.isMoving (0,1,2,3)
- <*element*>.isSettled (0,1)
- <*element*>.iType

Properties – StartType

- **StartType** defines the condition under which an **element** starts a move
  - Immediate (1) – after 5 samples (5 communication bus cycles)
  - InPosition (2) – previous motion has settled
  - GeneratorCompleted (3) – previous motion profile has completed
  - SyncStart (4) – used with SYNC command, to synchronize multiple motion elements
  - SuperImmediate (5) – immediately

- Syntax

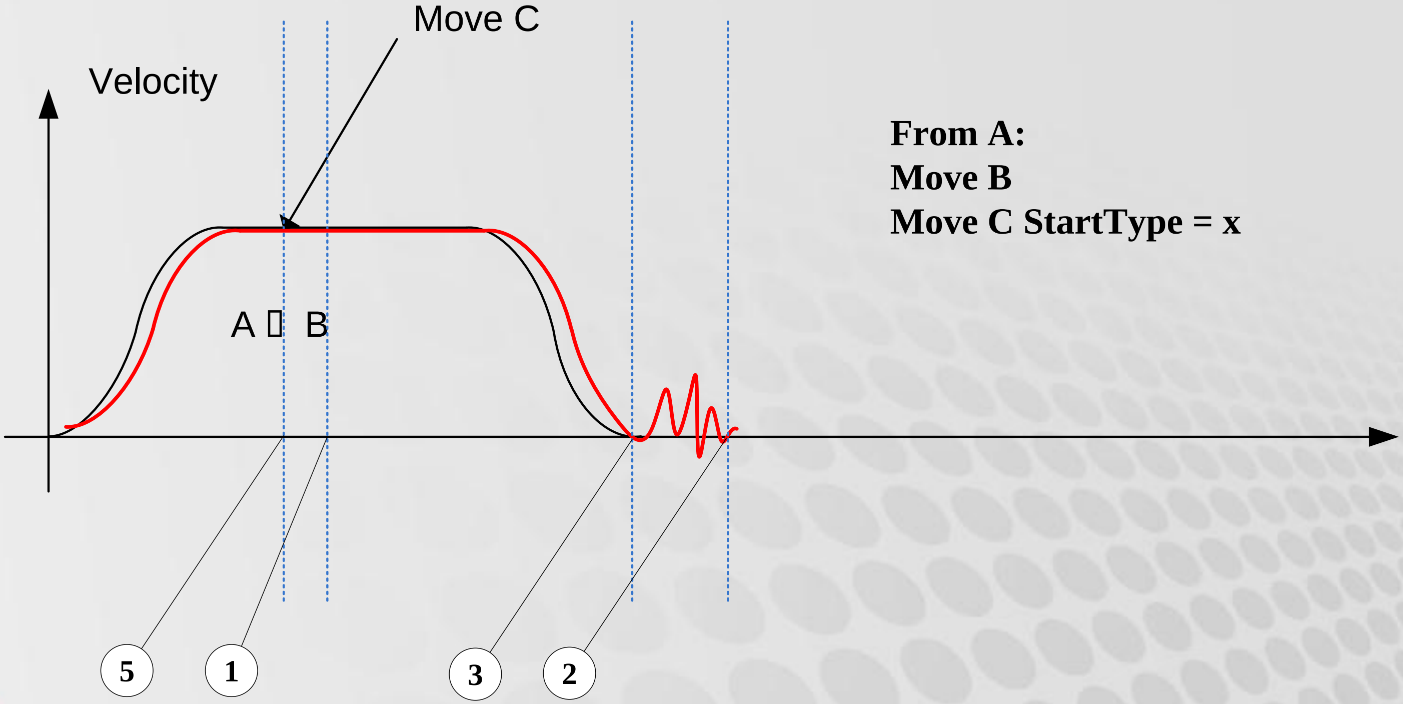
```
<element>.StartType
```

- Example

```
MyGroup.StartType = Immediate
MyGroup.StartType = 1
```

Properties – StartType Illustration

- Movement of A to B is interrupted by Move C command according to StartType value:
  - Immediate (1)
  - InPosition (2)
  - GeneratorCompleted (3)
  - SuperImmediate (5)



From A:
Move B
Move C StartType = x

Properties – StopType

- **StopType** defines how an **element** responds to the **Stop** command
  - Immediate (1)
  - OnPath (2)
  - EndMotion (3)

- Syntax

```
<element>.StopType
```

- Example

```
MyGroup.StopType = OnPath
MyGroup.StopType = 2
```

Properties – ProceedType

- **ProceedType** defines how **an element** responds to the **Proceed** command
  - Continue (1)
  - NextMotion (2)
  - ClearMotion (3)

- Syntax

```
<element>.ProceedType
```

- Example

```
MyGroup.ProceedType = Continue
MyGroup.ProceedType = 1
```

SYSTEM PROPERTIES

System Properties that Affect Motion

- Sys.Vord
- Sys.Motion
- Sys.En
- Sys.vRate, Sys.dRate, Sys.aRate, Sys.jRate *

* Note that the term “**Rate**” in this context **indicates a limiting value**, and not a rate of change,

MODAL and NODAL SETTINGS

Modal vs. Nodal Settings

- **Modal** Setting – a fixed value
  - Assigned explicitly in a program line:

    ```
    MyAxis.Acceleration = 100
    ```

  - Value persists until the property is assigned a different value
  - Value is used for motion commands unless specified otherwise by a nodal setting

- **Nodal** Setting – a temporary value, used for one motion command only
  - Overrides the modal setting
  - Included in the motion command

- Example

  ```
  MyAxis.Acceleration = 100
        'modal setting  (100)
  Jog MyAxis 5000 Acceleration = 10  'nodal override (10)
  ```

Modal vs. Nodal Settings

- Modal setting (MyGroup is a group compised of 4 axes)

```
...
MyGroup.Acceleration=1000
...



Move MyGroup{10,10,-10,0}
...
```

- Nodal setting

```
Move MyGroup{10,10,-10,0}  Acceleration = 500
...
```

MOTION TYPES
AND
COMMANDS

Motion Types – Interpolation

- **Move** – joint (natural) interpolation
- **MoveS** – straight, world-space interpolation
- **Circle** – circular (arc) interpolation
- **Jog** – unlimited axis movement at constant speed
- **Delay** – movement delay
- **SyncStart** – synchronized start of several elements

Move – Joint Interpolation

- Joint interpolated move
- Velocity determined by the “slowest” axis
- VCRUISE, ACC, DEC

Circle – Circular Motion

- Circular motion defined by angle, center and plane
  - helical motion - possible
- Circular motion defined by start, target and arc-point

Jog

- Axis only
- Pure velocity move
- Time can be limited or endless
- Used for conveyer motors and similar applications

SyncStart

- Two or more motion elements begin to move at the same time (synchronized movement)

- Syntax

```
SyncStart a1, g2, scara1
```

Delay

- DELAY – delays the motion sequence but not the program sequence (“dwell”)
- SLEEP – delays program execution, but not motion

- Syntax

```
DELAY <element> <time in ms>
```

MOTION BLENDING

Motion Blending

- A process in which motion is not stopped between motion commands
- 4 types of blending:
  - Concatenation - specifies a non-zero value for Vfinal (velocity at end of motion)
  - Continuous path (CP) – two motions are combined in a controlled manner
  - Superposition (SP) – two motions are superimposed on each other
  - Advanced interpolation (AI) – blending performed on entire array of points

- http://softmc.servotronix.com/wiki/Motion_Blending

END