JVL Add On Instructions for RockWell CompactLogix series

The JVL Add on functions help integration of the JVL stepper series in the CompactLogix PLC series programming environment.

Basically the motor is controlled using reading and writing to a group of registers in the motor. However this can sometimes be confusing and tideous since these registers and the values used has to be looked up in a register list.

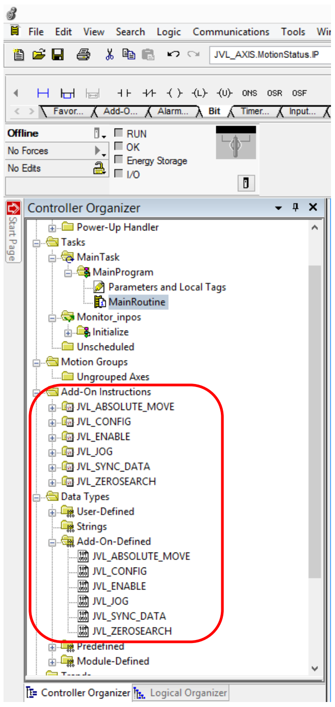
The add on functions are based on a linear motion, such as a linear actuator, spindle, belt etc.

The Add on functions makes controlling the motor simple since all motions are related to the real world mechanical setup instead of internal register values.

These functions are constantly developed and new functions will be added.

The JVL AOI can be downloaded from the JVL download page and imported into a project.

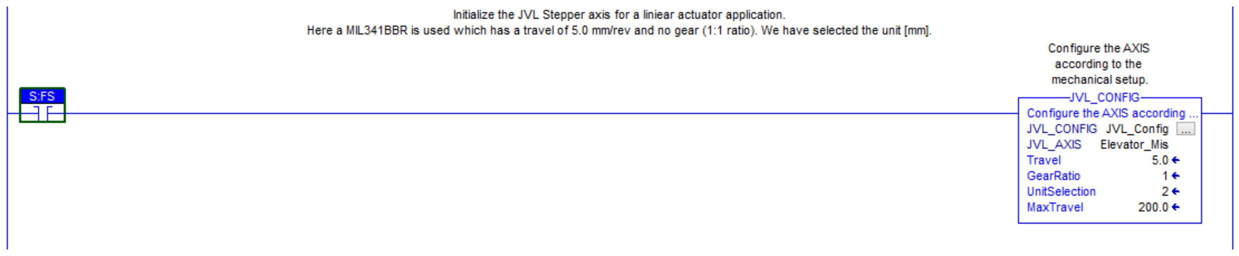
The following AOI are developed so far:



1. JVL_ABSOLUTE_MOVE
2. JVL_CONFIG
3. JVL_ENABLE
4. JVL_JOG
5. JVL_SYNC_DATA
6. JVL_ZEROSEARCH

JVL_CONFIG

Configures the axis with the basic mechanical parameters such as total length of travel, gearing ratio, travel length pr. Rev. and the unit to be used. After this function has been executed the motion commands will be related to the mechanical setup just configured using this function.



Initialize the JVL Stepper axis for a liniear actuator application.
Here a MIL341BBR is used which has a travel of 5.0 mm/rev and no gear (1:1 ratio). We have selected the unit [mm].

Configure the AXIS according to the mechanical setup.

```
S:FS
JVL_CONFIG
Configure the AXIS according ...
JVL_CONFIG   JVL_Config
JVL_AXIS     Elevator_Mis
Travel              5.0
GearRatio             1
UnitSelection         2
MaxTravel         200.0
```

Parmarameters

Travel	*[REAL], Travel/rev. MUST be in [mm]/rev.*
GearRation	*[REAL], In case a gearbox is mounted the ratio can be entered here, in case no gearbox is mounted, use 1 as ratio 1:1 is used.*
UnitSelection	*the following possibilitied are available:*
	0: No scaling is used, default motor units are used.
	1: [µm], and [µm/s] for velocity indications
	2: [mm], and [mm/s] for velocity
	3: [m] , and [m/s] for velocity
	4: [inch], and [inch/s] for velocity

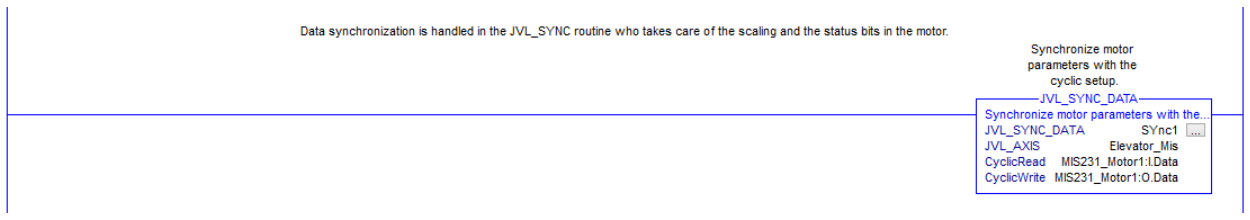
JVL_SYNC

This function is a "helper" function that keeps the axis parameters updated with the cyclic data exchange with the motor. Scaling is also handled in this function and other "household" –function needed. This function MUST be updated in each scan period. The function takes the following parameters:

Axis that is handled

CyclicRead words

CyclicWrite words

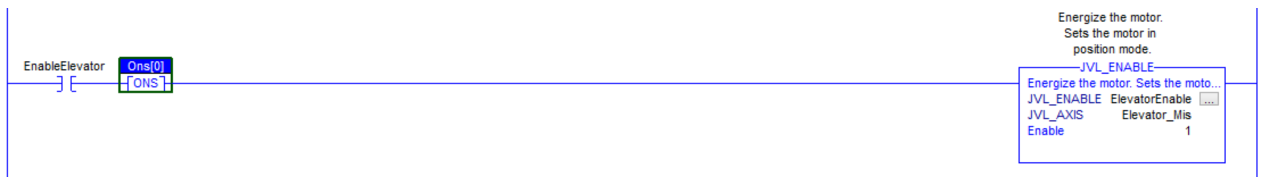

Data synchronization is handled in the JVL_SYNC routine who takes care of the scaling and the status bits in the motor.

Synchronize motor parameters with the cyclic setup.

```
JVL_SYNC_DATA
Synchronize motor parameters with the...
JVL_SYNC_DATA          SYnc1
JVL_AXIS          Elevator_Mis
CyclicRead   MIS231_Motor1:I.Data
CyclicWrite  MIS231_Motor1:O.Data
```

JVL_DRIVE_ENABLE

Energizes the motor and sets the motor in "Position" –mode ready for the motion command to be executed. This function takes a Boolean parameter that enables or disables the motor.


EnableElevator Ons[0] ONS

Energize the motor. Sets the motor in position mode.

```
JVL_ENABLE
Energize the motor. Sets the moto...
JVL_ENABLE  ElevatorEnable
JVL_AXIS          Elevator_Mis
Enable                       1
```

JVL_ABSOLUTE_MOVE

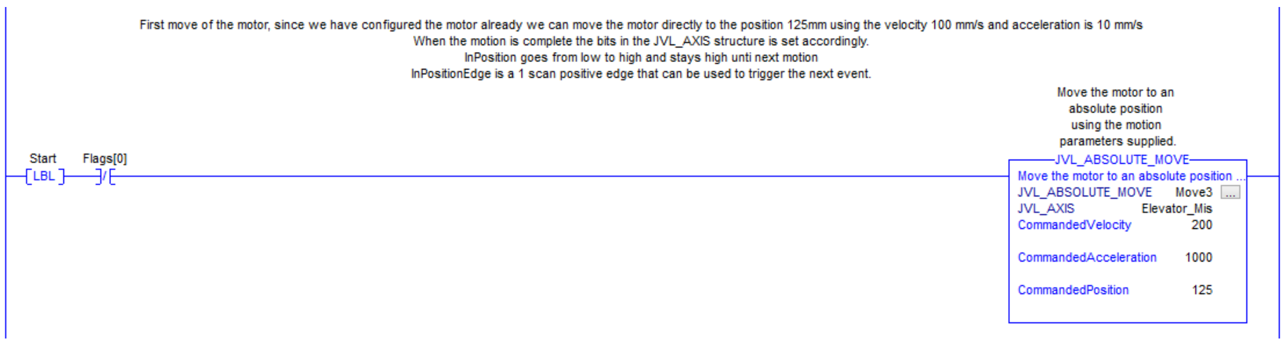
This command will execute a motion to a specific position using the unit configured in the ***JVL_CONFIG*** *command. A typical scenario would be that the actuator has a reference position achieved by a zerosearch or the motor could be equipped with a multiturn encoder configuration that will eliminated the need for a zerosearch procedure after powerup.*

The function takes the AXIS as a parameter and the following motion specific parameters:

Commanded velocity *related to the unit configured in* ***JVL_CONFIG*** *(um/s, mm/s, m/s or inch/s)*

Commanded acceleration *related to the unit configured in* ***JVL_CONFIG*** *(um/s, mm/s, m/s or inch/s)*

Commanded position *related to the unit configured in* ***JVL_CONFIG*** *(um, mm, m or inch)*


First move of the motor, since we have configured the motor already we can move the motor directly to the position 125mm using the velocity 100 mm/s and acceleration is 10 mm/s
When the motion is complete the bits in the JVL_AXIS structure is set accordingly.
InPosition goes from low to high and stays high unti next motion
InPositionEdge is a 1 scan positive edge that can be used to trigger the next event.

Start LBL Flags[0]

Move the motor to an absolute position using the motion parameters supplied.

```
JVL_ABSOLUTE_MOVE
Move the motor to an absolute position ...
JVL_ABSOLUTE_MOVE       Move3
JVL_AXIS          Elevator_Mis
CommandedVelocity          200
CommandedAcceleration     1000
CommandedPosition          125
```

2 bits in the JVL_AXIS structure indicates if the motor is in progress of a motion.

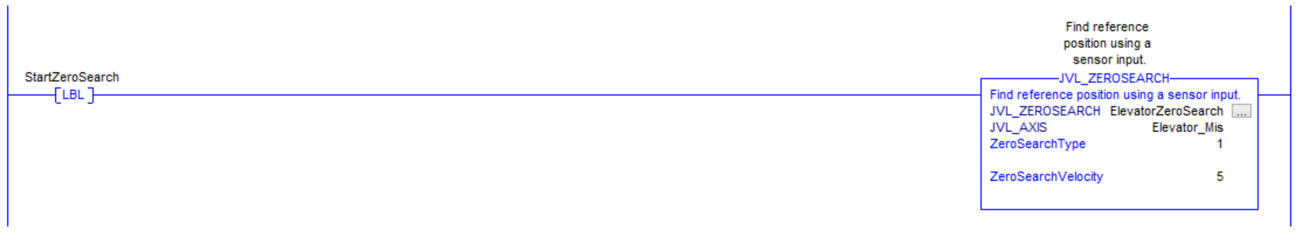
[JVL_AXIS].IP indicates if the axis is in progress of a motion

[JVL_AXIS].PC Indicates that the motion process is completed.

JVL_ZEROSEARCH

The JVL_ZEROSEARCH function will initiate a zerosearch procedure in the motor. This procedure starts a motion with the specific velocity and starts searching for a sensor input. 2 different zerosearch types can be used.

The 2 zerosearch types are described in the manual.



```
Find reference position using a sensor input.
JVL_ZEROSEARCH   ElevatorZeroSearch
JVL_AXIS                Elevator_Mis
ZeroSearchType                     1

ZeroSearchVelocity                 5
```

2 bits in the JVL_AXIS structure is used to indicate the state of the zerosearch.

[JVL_AXIS].AxisHomed Who is set when the zerosearch procedure is completed.

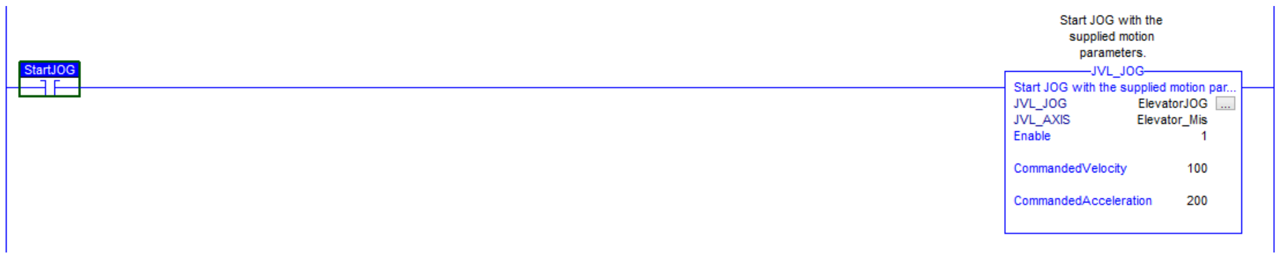
[JVL_AXIS].AxisHoming who is set during the zerosearch procedure.

JVL_JOG

This method is used solely for jogging purpose, the motor will move forward or backward depending on the velocity parameter supplied. The process is started when calling the method with the enable parameter set. For stopping the method needs to be called with the enable parameter cleared.

The parameters are:

Enable	Enable=1 will start the JOG, Enable = 0 will stop the JOG
CommandedVelocity,	velocity to set. The units used will be configured in the JVL_CONFIG -method
CommandedAcceleration,	acceleration used. The units used will be configured in the JVL_CONFIG -method

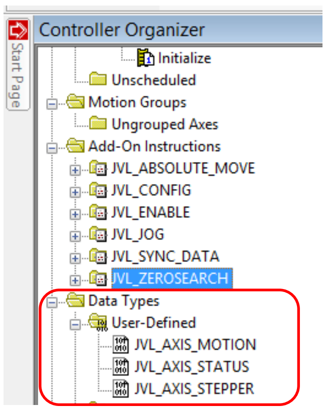


```
Start JOG with the supplied motion par...
JVL_JOG                ElevatorJOG
JVL_AXIS               Elevator_Mis
Enable                           1

CommandedVelocity              100

CommandedAcceleration          200
```

The following datastructures are used for the JVL AOI.



JVL_AXIS_MOTION

JVL_AXIS_STATUS

JVL_AXIS_STEPPER

JVL_AXIS_MOTION

Holds databits indicating a certain motion state.

Accelerating	*[BOOL] Motor is accelerating*
Decelerating	*[BOOL] Motor is decelerating*
Inposition	*[BOOL] Motor is inposition*
InPositionEdge	*[BOOL] Motor is inposition, rising edge, high for 1 scan*
IP	*[BOOL] Motion is in progress,*
AxisHoming	*[BOOL] Motor is performing a zerosearch and is not finished yet*
PC	*[BOOL] Motion is finished, the motor is ready for a new motion command*
AxisHomed	*[BOOL] Motor is done with a zerosearch procedure.*

JVL_AXIS_STATUS

This structure is mainly used internally and shall not be used externally.

JVL_AXIS_STEPPER

This is the main structure class that holds information regarding the actual motor state.

MotionStatus	*[JVL_AXIS_MOTION] See specification for this structure previously described.*
AxisStatus	*[JVL_AXIS_STATUS] See specification for this structure previously described.*
ActualPosition	*[DINT], Holds the actual position of the motor shaft, updated through the JVL_SYNC_DATA object.*
AverageVelocity	*[DINT] Average filtered velocity, reserved not for stepper series.*
ActualVelocity	*[DINT] Actual velocity*
ActualAcceleration	*[DINT] Reserved, not for stepper series.*
CommandPosition	*[DINT], Requested position, set from AOI*
CommandVelocity	*[DINT], Requested velocity, set from AOI*
CommandAcceleration	*[DINT], Requested acceleration, set form AOI*
Input	*[DINT], HW input status IO1-8, only stepper series.*
Output	*[DINT], HW output status IO1-8, only stepper*
CyclicRead	*[DINT8], cyclic input registers exchanged with the motor.*
CyclicWrite	*[DINT8], cyclic output registers exchanged with the motor.*
ZeroSearchVelocity	*[DINT], Velocity used for zerosearch.*
ZeroSearchPosition	*[DINT], ZeroSearch position, this position is used at the zerosearch sensor position.*
TravelConst	*[REAL], calculated constants used for scaling purposes, don't change this value after calling JVL_CONFIG method.*
VelConst	***[REAL]**], calculated constants used for scaling purposes, don't change this value after calling JVL_CONFIG method.*

How to….

Application example.

We have a MIL341BBR linear actuator.

MIL341BBR-EIH385F

Power 260W

Travel/rev	5.0	[mm]
Stroke	200	[mm]

Since this linear actuator is equipped with a multiturn encoder, we have no need to do a reference zerosearch.

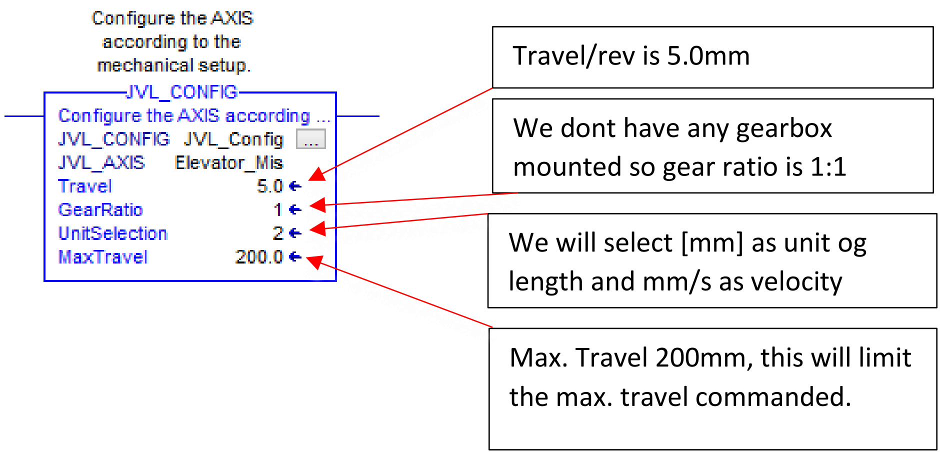
The motor must move between 2 points whenever a material is positioned at a certain input.

The lower position is located at 10mm and the upper position is located at 125mm.

So the PLC program task with this motor is to move to the upper position when a sensor is detected and move back to the lower position after 5s. to the lower position.

For the initialization we will will setup the JVL_AXIS_STEPPER class for this actuator model, the **MIL341BBR**.

We call the **JVL_CONFIG** method with the following parameters:



Configure the AXIS according to the mechanical setup.

```
JVL_CONFIG
Configure the AXIS according ...
JVL_CONFIG   JVL_Config  ...
JVL_AXIS     Elevator_Mis
Travel               5.0
GearRatio              1
UnitSelection          2
MaxTravel          200.0
```

Travel/rev is 5.0mm

We dont have any gearbox mounted so gear ratio is 1:1

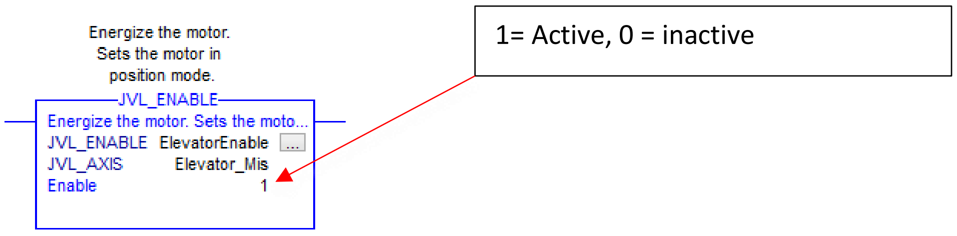
We will select [mm] as unit og length and mm/s as velocity

Max. Travel 200mm, this will limit the max. travel commanded.

So now we can control the motor very precise using mm and mm/s as velocity indications.

Next we will set the motor into an active mode, we will energize the motor.

We assume that the motor has been mounted correctly so position 0 is at the right reference level.


Energize the motor. Sets the motor in position mode.

JVL_ENABLE

Energize the motor. Sets the moto...

```
JVL_ENABLE  ElevatorEnable ...
JVL_AXIS    Elevator_Mis
Enable                   1
```

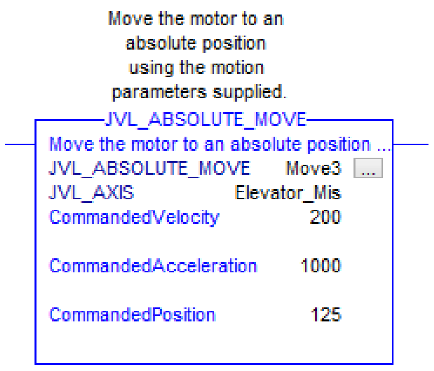
Now the motor is ready to receive the motion commands required.

When the sensor has been detected we will call the **JVL_ABSOLUTE_MOVE** method, with the following parameters:

Velocity: 200mm/s

Acceleration: 1000mm/s

Position 125mm


Move the motor to an absolute position using the motion parameters supplied.

JVL_ABSOLUTE_MOVE

Move the motor to an absolute position ...

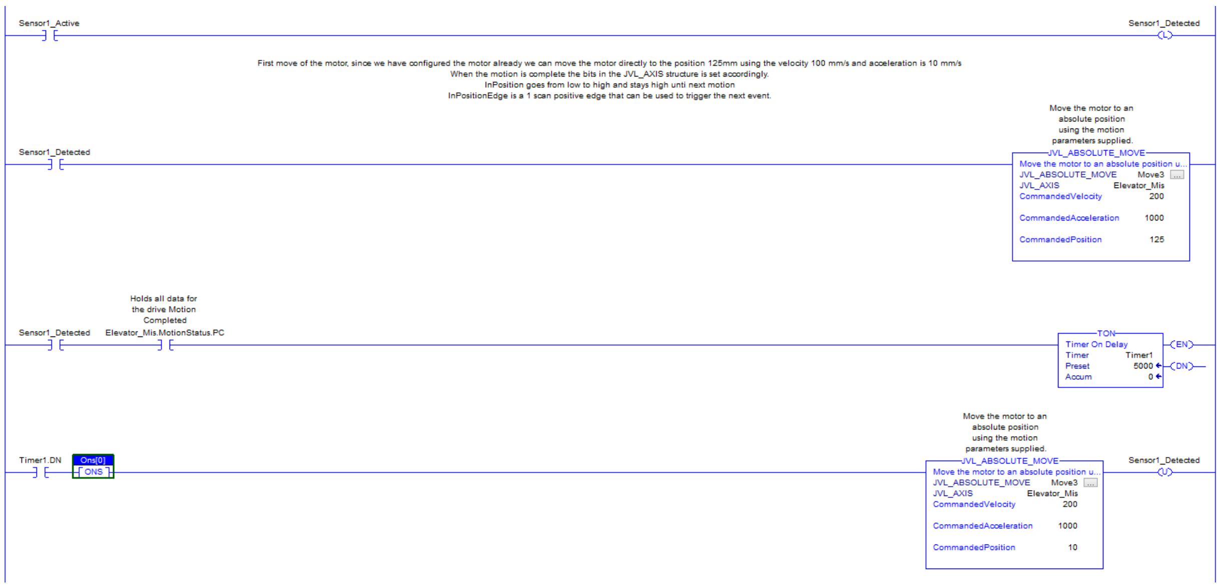
```
JVL_ABSOLUTE_MOVE       Move3 ...
JVL_AXIS          Elevator_Mis
CommandedVelocity          200

CommandedAcceleration     1000

CommandedPosition          125
```

After 5s, we will call the **JVL_ABSOLUTE_MOVE** method for returning to position 10mm.

Program example simplified.


First move of the motor, since we have configured the motor already we can move the motor directly to the position 125mm using the velocity 100 mm/s and acceleration is 10 mm/s
When the motion is complete the bits in the JVL_AXIS structure is set accordingly.
InPosition goes from low to high and stays high unti next motion
InPositionEdge is a 1 scan positive edge that can be used to trigger the next event.

Sensor1_Active
Sensor1_Detected
L
Sensor1_Detected
Move the motor to an absolute position using the motion parameters supplied.
JVL_ABSOLUTE_MOVE
Move the motor to an absolute position u...
JVL_ABSOLUTE_MOVE Move3
JVL_AXIS Elevator_Mis
CommandedVelocity 200
CommandedAcceleration 1000
CommandedPosition 125
Holds all data for the drive Motion Completed
Sensor1_Detected
Elevator_Mis.MotionStatus.PC
TON
Timer On Delay
Timer Timer1
Preset 5000
Accum 0
EN
DN
Timer1.DN
Ons[0]
ONS
Move the motor to an absolute position using the motion parameters supplied.
JVL_ABSOLUTE_MOVE
Move the motor to an absolute position u...
JVL_ABSOLUTE_MOVE Move3
JVL_AXIS Elevator_Mis
CommandedVelocity 200
CommandedAcceleration 1000
CommandedPosition 10
Sensor1_Detected
U

JVL Industri Elektronik A/S
Bregnerødvej 127
DK-3460 Birkerød
Denmark
Tlf. +45 45 82 44 40
Fax. +45 45 82 55 50
e-mail: jvl@jvl.dk
Internet: http://www.jvl.dk