

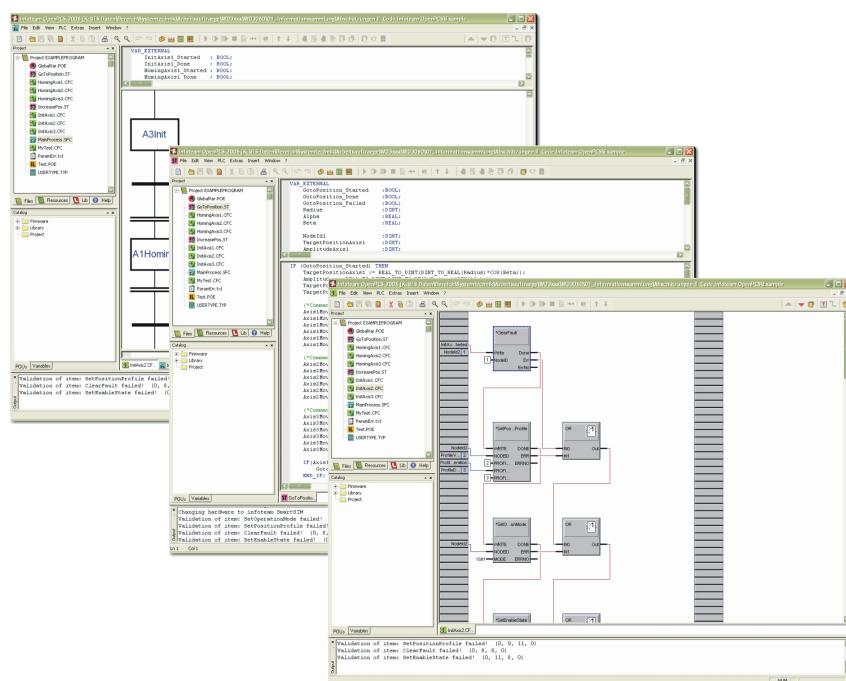
EPOS2 P Programming Examples	Application Example Simple Motion Sequence	Edition April 2010
------------------------------	---	--------------------

# EPOS2 P

## Programmable Positioning Controller

### Application Example Simple Motion Sequence

Edition April 2010



maxon document: rel1492

maxon motor		
EPOS2 P Programming Examples	Application Example Simple Motion Sequence	Edition April 2010

## Table of contents

Table of contents .....	2
1 Table of figures.....	3
2 Introduction.....	4
2.1 Required Tools .....	4
2.2 Required Devices .....	4
2.3 References .....	4
3 Functionality .....	5
3.1 Main State Machine.....	5
3.2 Error Handling State Machine .....	6
4 How to use Example Program .....	8
4.1 How to Watch Variables in the Source Code.....	8
4.2 How to Watch State Changes .....	9
4.3 How to Trigger and Acknowledge Errors .....	10

maxon motor		
EPOS2 P Programming Examples	Application Example Simple Motion Sequence	Edition April 2010

## 1 Table of figures

Figure 1: Main State Machine.....	5
Figure 2: ErrorHandling State Machine .....	6
Figure 3: StateMachine Interactions for more than one process .....	7
Figure 26: Project Explorer.....	8
Figure 27: Monitor/Edit.....	8
Figure 28: Watch variable in source code .....	8
Figure 29: Project Explorer Resources .....	9
Figure 30: Watch Variables for State Changes .....	9
Figure 31: Project Explorer Resources .....	10
Figure 32: Watch Variables for ErrorHandling.....	10

maxon motor		
EPOS2 P Programming Examples	Application Example Simple Motion Sequence	Edition April 2010

## 2 Introduction

The example “SimpleMotionSequence” shows how to move between two positions.

### 2.1 Required Tools

Tool	Minimal Version	Description
maxon motor EPOS Studio	1.42	Configuration, Programming

**Note:** Freely available at <http://www.maxonmotor.com> category <Service & Downloads> or in the maxon motor e-shop <http://shop.maxonmotor.com>. They are also part of the EPOS Positioning Controller DVD.

### 2.2 Required Devices

Device	Quantity	Firmware Version	Description
EPOS2 P 24/5	1	0x0200	Programmable Positioning Controller

### 2.3 References

Document	Version	Description
EPOS2 P Programming Reference	Edition April 2010	Programming
EPOS2 P Firmware Specification	Edition April 2010	EPOS2 P Functionality
EPOS2 Firmware Specification	Edition April 2010	EPOS2 Functionality

**Note:** Freely available at <http://www.maxonmotor.com> category <Service & Downloads> or in the maxon motor e-shop <http://shop.maxonmotor.com>. They are also part of the EPOS Positioning Controller DVD.

### 3 Functionality

The SimpleMotionSequence example consists of two state machines. One state machine implements the application process and the second implements the error handling. The main state machine moves between two positions.

#### 3.1 Main State Machine

The main state machine is an example for a motion sequence application process.

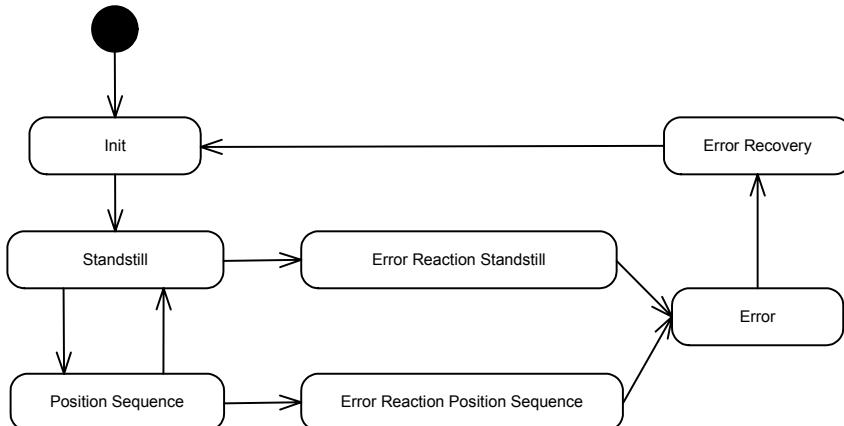


Figure 1: Main State Machine

State	Description
Standstill	This is an initialisation state, which resets the EPOS2.
Position Sequence	This is a process state, which is entered after the init state or after a position sequence.
Error Reaction Standstill	This is an error reaction state, which is entered when an error is detected in State Standstill.
Error Reaction Position Sequence	This is an error reaction state, which is entered when an error is detected in State Position Sequence.
Error	After an error reaction the process remains in the Error state, until the error is acknowledged.
Error Recovery	After acknowledgment the process is recovered and restarted.

Transitions		Description
Init	→ Standstill	Done = Reset is done and Timer of 5s has elapsed
Standstill	→ Position Sequence	Done = Timer of 2s has elapsed
Position Sequence	→ Standstill	Done = The two sequential relative positioning operations are completely finished
Standstill	→ Error Reaction Standstill	ErrorHandling State = Error Reaction
Position Sequence	→ Error Reaction Position Sequence	ErrorHandling State = Error Reaction
Error Reaction Standstill	→ Error	Done = Timer of 1s has elapsed
Error Reaction Position Sequence	→ Error	Done = Timer of 1s has elapsed
Error Recovery	→ Init	Done = Reset is done and Timer of 5s has elapsed

### 3.2 Error Handling State Machine

The ErrorHandling state machine monitors the process and coordinates the error reactions and error recoveries of the processes. The error handling is implemented as a separate state machine to allow the implementation of more than one process state machine.

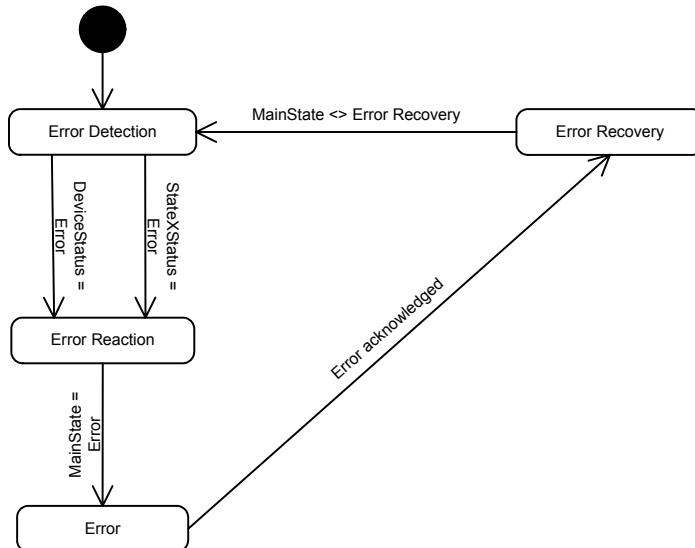


Figure 2: ErrorHandling State Machine

State	Description
Error Detection	This state monitors the main state machine and all involved devices.
Error Reaction	This state signals to the main state machine, that an error reaction has to be started. The error handling state machine remains in this state, until all processes have accomplished their error reactions.
Error	This state is entered when the main state machine has finished the error reaction and shows all error information.
Error Recovery	This state signals to the main state machines, that the error is acknowledged and the process can be restarted. The error handling state machine remains in this state, until all processes have been restarted.

Transitions		Description
Error Detection	→	Error Reaction
Error Reaction	→	Error
Error	→	Error Recovery
Error Recovery	→	Error Detection

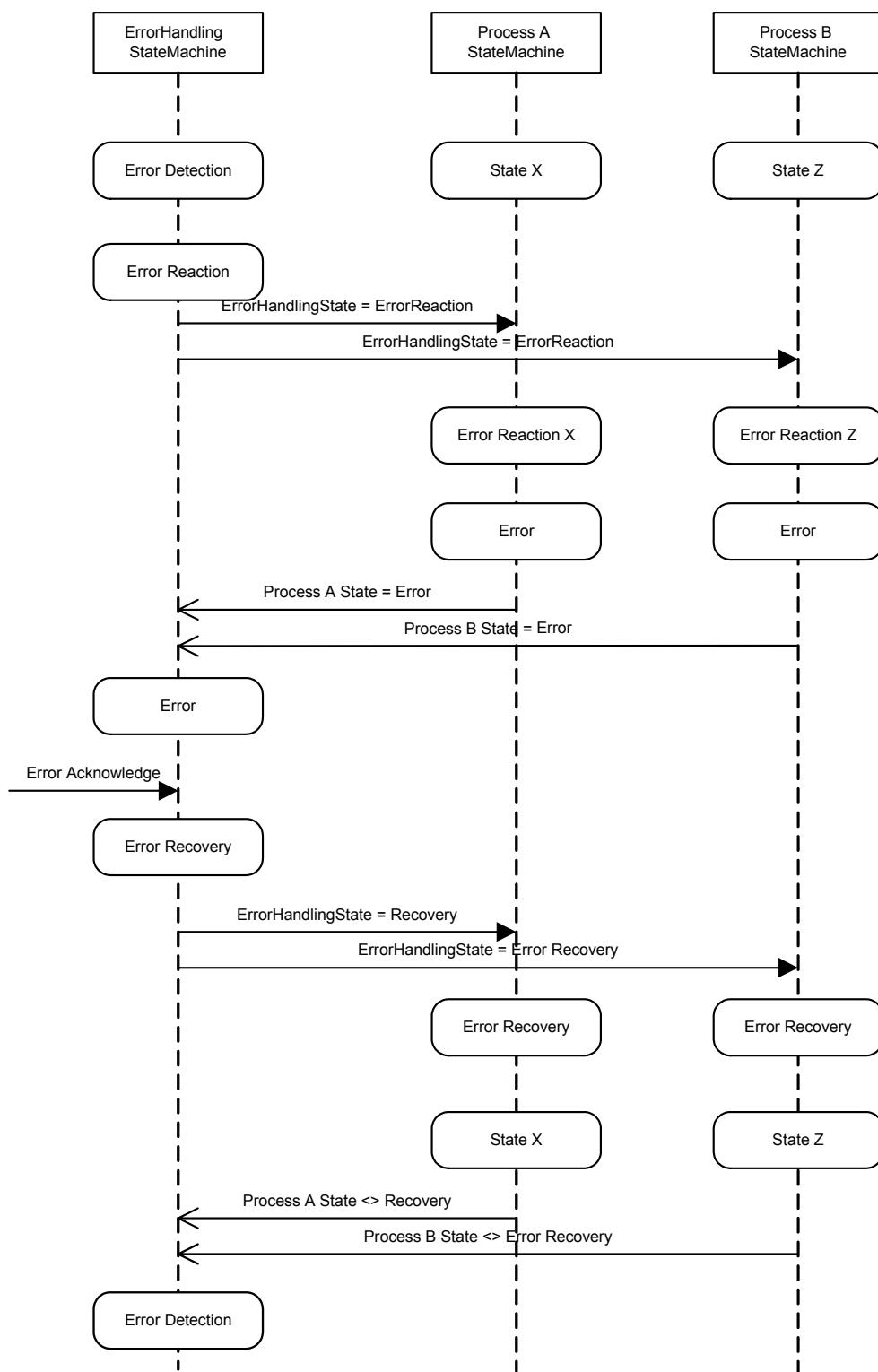


Figure 3: StateMachine Interactions for more than one process

## 4 How to use Example Program

### 4.1 How to Watch Variables in the Source Code

#### 1. Open Source Code

- Double click on a file in the project explorer (e.g. PROG\_MAIN.ST).

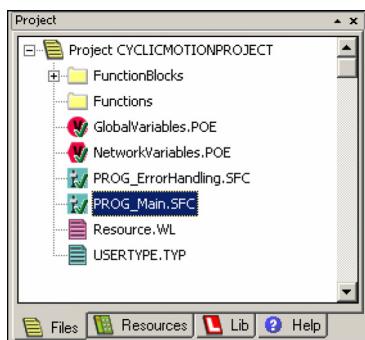


Figure 4: Project Explorer

- Click on the button 'Monitor/Edit' in the toolbar to switch the source code window into debugging mode.



Figure 5: Monitor/Edit

- Move the cursor over a variable in the source code to see the actual value.

```

IF eStateMain = MAIN_Init THEN
  fbInit(MAIN_STATE eStateMain = 0) execute := oExecuteInit;
  oExecuteInit := TRUE;

  IF fbInit.Done THEN
    oExecuteInit := FALSE;
    eStateMain := MAIN_StateA;
  END_IF;

```

The screenshot shows the source code editor with the variable 'eStateMain' underlined with a red dotted line, indicating it is being watched.

Figure 6: Watch variable in source code

## 4.2 How to Watch State Changes

### 2. Select Tab 'Resources'

- d) Select the tab 'Resources' in the project explorer.

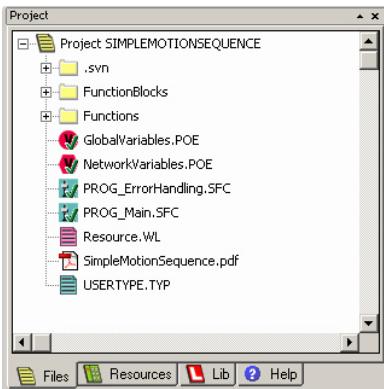


Figure 7: Project Explorer Resources

### 3. Add Watch Variables

- e) Add the following variables to the watch window. Right click on the items in the 'Resources' window and click 'Add To Watchlist'

Instancepath	Name	Value	Type	Address	Force	Comment
PROG_ERRORHANDLING.FBEH_ERROR	OERRORRECOVERY	FALSE	BOOL			
PROG_MAIN.FBSTATECYCLICMOVEMENT	DPOSITION2	0	DINT			
PROG_MAIN.FBSTATECYCLICMOVEMENT	DPOSITION1	0	DINT			
PROG_MAIN	AXISO_ERRORCODE	0	DINT			
PROG_MAIN	AXISO_ERRORFLAG	FALSE	BOOL			
PROG_MAIN	MAIN_ERRORSTATE	0	ENUM			
PROG_MAIN	MAIN_ERRORID	0	DINT			
PROG_MAIN	MAIN_ERRORFLAG	FALSE	BOOL			
PROG_ERRORHANDLING	ESTATEERRORHANDLING	0	ENUM			
PROG_MAIN	ESTATEMAIN	0	ENUM			

Figure 8: Watch Variables for State Changes

### 4. Watch State Changes

- f) Have a look at the state variables to see the active state.

## 4.3 How to Trigger and Acknowledge Errors

### 5. Select Tab 'Resources'

- g) Select the tab 'Resources' in the project explorer.

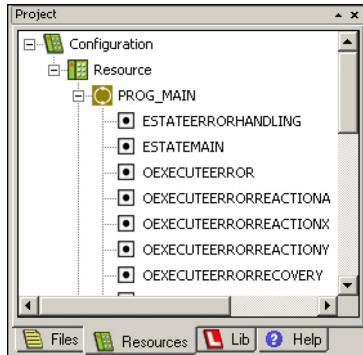


Figure 9: Project Explorer Resources

### 6. Add Watch Variables

- h) Add the following variables to the watch window. Right click on the items in the 'Resources' window and click 'Add To Watchlist'.

Instancepath	Name	Value	Type	Address	Force	Comment
PROG_ERRORHANDLING.FBEH_ERROR	OERRORRECOVERY	FALSE	BOOL			
PROG_MAIN.FBSTATECYCLICMOVEMENT	DPOSITION2	0	DINT			
PROG_MAIN.FBSTATECYCLICMOVEMENT	DPOSITION1	0	DINT			
PROG_MAIN	AXISO_ERRORCODE	0	DINT			
PROG_MAIN	AXISO_ERRORFLAG	FALSE	BOOL			
PROG_MAIN	MAIN_ERRORSTATE	0	ENUM			
PROG_MAIN	MAIN_ERRORID	0	DINT			
PROG_MAIN	MAIN_ERRORFLAG	FALSE	BOOL			
PROG_ERRORHANDLING	ESTATEERRORHANDLING	0	ENUM			
PROG_MAIN	ESTATEMAIN	0	ENUM			

Figure 10: Watch Variables for ErrorHandling

### 7. Trigger Error

- i) Open the EPOS Studio tool 'Profile Position Mode' of the EPOS [Internal].
- j) Reduce the parameter 'Max Following Error' to 20 qc.
- k) Move or block the motor axis to cause a following error ( Code 0x8611).
- l) Have a look at the watch variables AXIS0\_ERRORFLAG,  
AXIS0\_ERRORCODE.
- m) Increase the parameter 'Max Following Error' to the original value.

### 8. Acknowledge Error

- n) Wait until the Main and ErrorHandling State Machine have changed to the state 'Error'.
- o) Double click the variable 'oErrorRecovery' and set the value to TRUE.
- p) See how the Main State Machine is restarted.

#### Note:

Make sure the variables are set to FALSE before changing. Only a positive edge of the signal triggers a reaction.