

BECKHOFF



Calling FBs with a CASE Instruction

Calling FBs with a CASE Instruction

BECKHOFF

- When calling Function Blocks from a CASE instruction it is important to remember that only one step is run per PLC cycle.

Calling FBs with a CASE Instruction

- In the below code step 0 will Initialize the timer
- Step 1 sets the input to TRUE and sets iStep to a value of 2
- Step 2 is monitoring the output of the timer, however the timer itself is never being called and will no longer update

```
CASE iStep OF

0: (*Init*);
    fbTON(IN:=FALSE);
1: (*Start TON*);
    fbTON(IN:=TRUE, PT:=T#1S);
    iStep := 2;
2: (*Wait*);
    IF fbTON.Q THEN
        xTimerComplete := TRUE;
    END_IF
END_CASE
```

Calling FBs with a CASE Instruction

- The function block instance along with the parentheses () must be called in each step that uses the function block.

```
CASE iStep OF

0: (*Init*);
    fbTON(IN:=FALSE);
1: (*Start TON*);
    fbTON(IN:=TRUE, PT:=T#1S);
    iStep := 2;
2: (*Wait*);
    fbTON();
    IF fbTON.Q THEN
        xTimerComplete := TRUE;
    END_IF
END_CASE
```

Calling FBs with a CASE Instruction

BECKHOFF

- Another option is to call the function block from outside of the CASE instruction
- In this example the timer is called every PLC cycle
- The steps in the CASE instruction simply control a variable that is evaluated by the timer

```
CASE iStep OF

0: (*Init*);
    xStartTimer := FALSE;
1: (*Start TON*);
    xStartTimer := TRUE;
    iStep := 2;
2: (*Wait*);
    IF fbTON.Q THEN
        xTimerComplete := TRUE;
    END_IF
END_CASE

fbTON(IN:=xStartTimer, PT:=T#1S);
```

- Calling a Function Block outside of a CASE instruction is slightly less efficient.
- When the Function Block is outside of a CASE instruction, it is updated every PLC cycle.
- When the Function Block is only called from within a CASE instruction then it will only be updated when the step is active.
- Calling the Function Block from within a CASE instruction requires careful programming practices to ensure that the Function Block is updated properly.

Calling FBs with a CASE Instruction

BECKHOFF

- When calling a Function Block from within a CASE instruction it is important to call the FB in each step as needed.

Calling FBs with a CASE Instruction

BECKHOFF

- The Execute input of a Function Block from the Tc2_MC2.lib requires a rising edge for proper operation.

```
CASE istep OF
```

```
0: (*Init*);
```

```
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
```

```
10: (*Move*);
```

```
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
```

```
    fbMC_MoveRel (Axis:= Axis1,
```

```
        Execute:= TRUE,
```

```
        Distance:= 100.0,
```

```
        Velocity:= 10.0);
```

```
    iStep := iStep + 1;
```

```
11: (*Wait Complete*);
```

```
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
```

```
    IF NOT fbMC_MoveRel.Error THEN
```

```
        IF fbMC_MoveRel.Done THEN
```

```
            fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
```

```
            iStep := 0;
```

```
        END_IF
```

```
    ELSE
```

```
        iErrStep := iStep;
```

```
        iStep := 999;
```

```
    END_IF
```

```
999: (*Error Step*);
```

```
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
```

```
    iStep := 0;
```

```
ELSE
```

```
    iErrStep := iStep;
```

```
    iStep := 999;
```

```
END_CASE
```

Calling FBs with a CASE Instruction

BECKHOFF

- To ensure the rising edge is handled properly the FB is first called with a FALSE at the Execute and then again called with a TRUE at the Execute.
- iStep is immediately and unconditionally changed.

```
CASE istep OF

0: (*Init*);
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
10: (*Move*);
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
    fbMC_MoveRel (Axis:= Axis1,
        Execute:= TRUE,
        Distance:= 100.0,
        Velocity:= 10.0);
    iStep := iStep + 1;
11: (*Wait Complete*);
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
    IF NOT fbMC_MoveRel.Error THEN
        IF fbMC_MoveRel.Done THEN
            fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
            iStep := 0;
        END_IF
    ELSE
        iErrStep := iStep;
        iStep := 999;
    END_IF
999: (*Error Step*);
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
    iStep := 0;
ELSE
    iErrStep := iStep;
    iStep := 999;
END_CASE
```

Calling FBs with a CASE Instruction

BECKHOFF

- The FB is called again with the Execute set to FALSE, this calls the Function Block so that it will be updated in the step, and also resets the Execute
- The FB is then monitored for both an Error status and a Done status.

```
CASE istep OF

0: (*Init*);
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
10: (*Move*);
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
    fbMC_MoveRel (Axis:= Axis1,
        Execute:= TRUE,
        Distance:= 100.0,
        Velocity:= 10.0);
    iStep := iStep + 1;
11: (*Wait Complete*);
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
    IF NOT fbMC_MoveRel.Error THEN
        IF fbMC_MoveRel.Done THEN
            fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
            iStep := 0;
        END_IF
    ELSE
        iErrStep := iStep;
        iStep := 999;
    END_IF
999: (*Error Step*);
    fbMC_MoveRel (Axis:= Axis1, Execute:= FALSE);
    iStep := 0;
ELSE
    iErrStep := iStep;
    iStep := 999;
END_CASE
```

Beckhoff Automation GmbH & Co. KG

Headquarters
Huelshorstweg 20
33415 Verl
Germany

Phone: +49 5246 963-0
Fax: +49 5246 963-198
E-Mail: info@beckhoff.com
Web: www.beckhoff.com

© Beckhoff Automation GmbH & Co. KG

All images are protected by copyright. The use and transfer to third parties is not permitted.

Beckhoff®, TwinCAT®, EtherCAT®, Safety over EtherCAT®, TwinSAFE®, XFC® and XTS® are registered trademarks of and licensed by Beckhoff Automation GmbH. Other designations used in this presentation may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

The information provided in this presentation contains merely general descriptions or characteristics of performance which in case of actual application do not always apply as described or which may change as a result of further development of the products. An obligation to provide the respective characteristics shall only exist if expressly agreed in the terms of contract.

Beckhoff Automation GmbH & Co. KG

Unternehmenszentrale
Hülshorstweg 20
33415 Verl
Deutschland

Telefon: +49 5246 963-0
Fax: +49 5246 963-198
E-Mail: info@beckhoff.de
Web: www.beckhoff.de

© Beckhoff Automation GmbH & Co. KG

Alle Bilder sind urheberrechtlich geschützt. Die Weitergabe und Nutzung durch Dritte ist nicht gestattet.

Beckhoff®, TwinCAT®, EtherCAT®, Safety over EtherCAT®, TwinSAFE®, XFC® und XTS® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH. Die Verwendung anderer in dieser Präsentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Kennzeichen führen.

Die Informationen in dieser Präsentation enthalten lediglich allgemeine Beschreibungen bzw. Leistungsmerkmale, welche im konkreten Anwendungsfall nicht immer in der beschriebenen Form zutreffen bzw. welche sich durch Weiterentwicklung der Produkte ändern können. Die gewünschten Leistungsmerkmale sind nur dann verbindlich, wenn sie bei Vertragsabschluss ausdrücklich vereinbart werden.