

The Application Note is pertinent to CTNet Systems

Tripping the Drive upon a CTNet Loss

This application note is pertaining to an example using a User Define Function Block (UDFB) to trip a drive if the CTNet link for that pertaining node goes down. This is very helpful in applications where you are passing critical data. The UDFB will also take into account the bootup time of the processor and when to start checking it. This code should be put into every drive on the CTNet Network.

In the DPL Editor screen you can create your own function blocks if you wish to, they will show up under the category column of User-Defined FB's. If you have multiple UDFB's you may want to create your own Library which you can share between your colleagues. The only drawback of using a Library is if someone doesn't have the Library that will not be able to compile your program (sometimes this could be considered an advantage as well as it protects your code from being modified and possibly disrupted).

In this example we will put the actual code in and is only an example of a way to handle this.

Step 1: Creating the CTNet Loss UDFB by going to Insert / User Define Function Block. You must be in the DPL editor screen. You will get a screen as shown below. The FB Name must begin with an underscore such as `_CTNetLoss`. Some thought must be put into how many inputs you may want and how many outputs you may want. To insert an Input click on the 0 of the inputs and type in a name in the Name Tab, then use the drop down to make it a Float or an Integer. If you type in `Input%` in the Name Tab it will automatically make it an Integer for you. Add in as many Inputs as you wish. Next do the same for the Outputs.

Before Populated


After Populated

As you can see in the After Populated we have 1 input and 3 outputs and we have named the UDFB _CTNetLoss.

Step 2: Writing the code for your UDFB. In every UDFB you have initialization code as well as the FBbody. The initialization code is similar to the Initial Task, it only runs once. Where the FBbody runs is based on which Task you put the code into (ie: Background, Clock, Pos0). In this UDFB we set a BootUp bit in the initialization task, when the program moves down into the FBbody we have a 3 second **Timer** (TON) which is used because all drives processors may not come up at the same time, such as when common bus is being used for example.

Then we have a basic **If / Then / Else** statement where we monitor the input (the input will be the CTNet Status (XX.36), so when the CTNet status goes negative we set a bit called StartTimeOut. If the StartTimeOut is set (1) this is used for a trigger to start another 3 Second Timer, if the StartTimeOut is on for 3 seconds we set a bit called TripQ. The last output we have is TimeMs which is the elapsed time .

```

// CTNet Error Check
 (TimeMs%, TripQ%, Active%) = _CtNetLoss (Input%){

// initialisation code
BootUp% = 1

FBbody
// main body code

// Boot up Timer
Active% = TON(BootUp%, 3000)

// If CTNet Messages Per second is < 0 then start to time out
IF Input% < 0 AND Active% = 1 THEN

    StartTimeOut% = 1

ELSE

    StartTimeOut% = 0

ENDIF

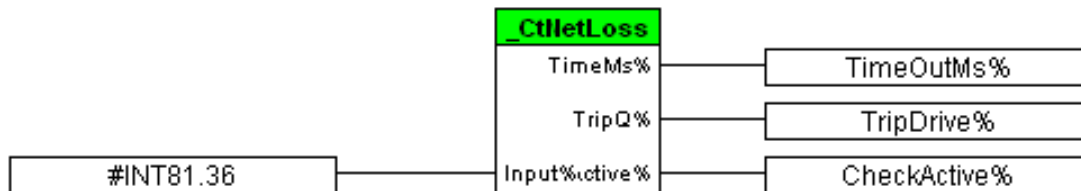
// Trip Timer
(TripQ%, TimeMs%) = TON2(StartTimeOut%, 3000)

} //(TimeMs%, TripQ%, Active%) = _CtNetLoss (Input%)

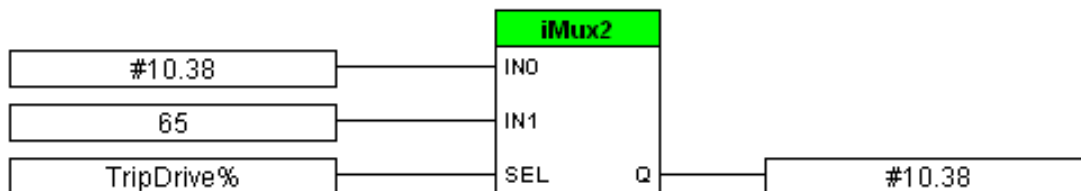
```

Step 3: We will use this UDFB in the Clock Task since it is deterministic. So after you have inserted a clock task, insert a LD/FB diagram. When you insert a Function Block, goto the category called User-Defined FB's and you should see your FB that you have created. In this example when we lose the CTNet for this node we trip the drive on a User Trip 65 (tr 65). The variable Input% is monitoring #81.36. It should be noted that you can do this if you only have one Application module in the drive as it makes it universal regardless of which slot one places the Apps Plus module. If you are using multiple Application Modules in one drive you will have to address it discretely via #15.36, #16.36 or #17.36.

Check CtNet Diagnostic Parameter xx36.
 If <0 Then start delay timer. After 3 secs
 set drive trip output. (NOTE function will
 only become active 3 secs after boot-up)



If a CtNet error is detected and the TripDrive bit
 is set Then trip drive on a Tr065 fault.



Tr 65 = CTNet Loss

When you create such User Trip such as we have here, it becomes imperative that one documents what each Trip represents in the User's Manual for the End Customer. It is always a good idea to include this trip list as a single sheet that can be taped to the cabinet inside door. If a customer calls in for Technical Support we would have no clue as to the nature of what this trip is without having the actual source code listing.

Questions ?? Ask the Author:

Author: Tim Micklich

e-mail : <mailto:tim.micklich@emerson.com>

