Self Training Guide

DLP Module 03 – Testing Digitals, Setting Digitals and DLP Timers

Introduction

One of the most basic and common uses for a DLP is to delay (or debounce) digital signals before they are transmitted back to the base station.

This document discusses the DLP timer function and how to use it.

Before reading this document, you should have read the previous module(s) and be comfortable with the concepts discussed within. This document also assumes that you be familiar with the Q90 configuration software, and have successfully installed the DLP IDE software.

Additional details on the syntax of all DLP commands can be found in the online help.

In this document any DLP commands are presented in BLUE TYPEFACE while all DLP system variables and IO registers are in RED TYPEFACE.

The .ASM file for any DLP shown in this document is available separately.

This module contains help on:

- TCE and TCD
- EC and ECC
- SETDIG and RESDIG
- TMR
- System variables TIMERS*n* , TIMERD*n* and TIMERC*n*.

QTech Data Systems Ltd
Unit 6 Amuri Park
404 Barbadoes Street
Christchurch
New Zealand

Phone: +64 3 3663713
Fax: +64 3 3652815
E-mail: sales@qtech.co.nz
Web www.qtech.co.nz

TCE and TCD

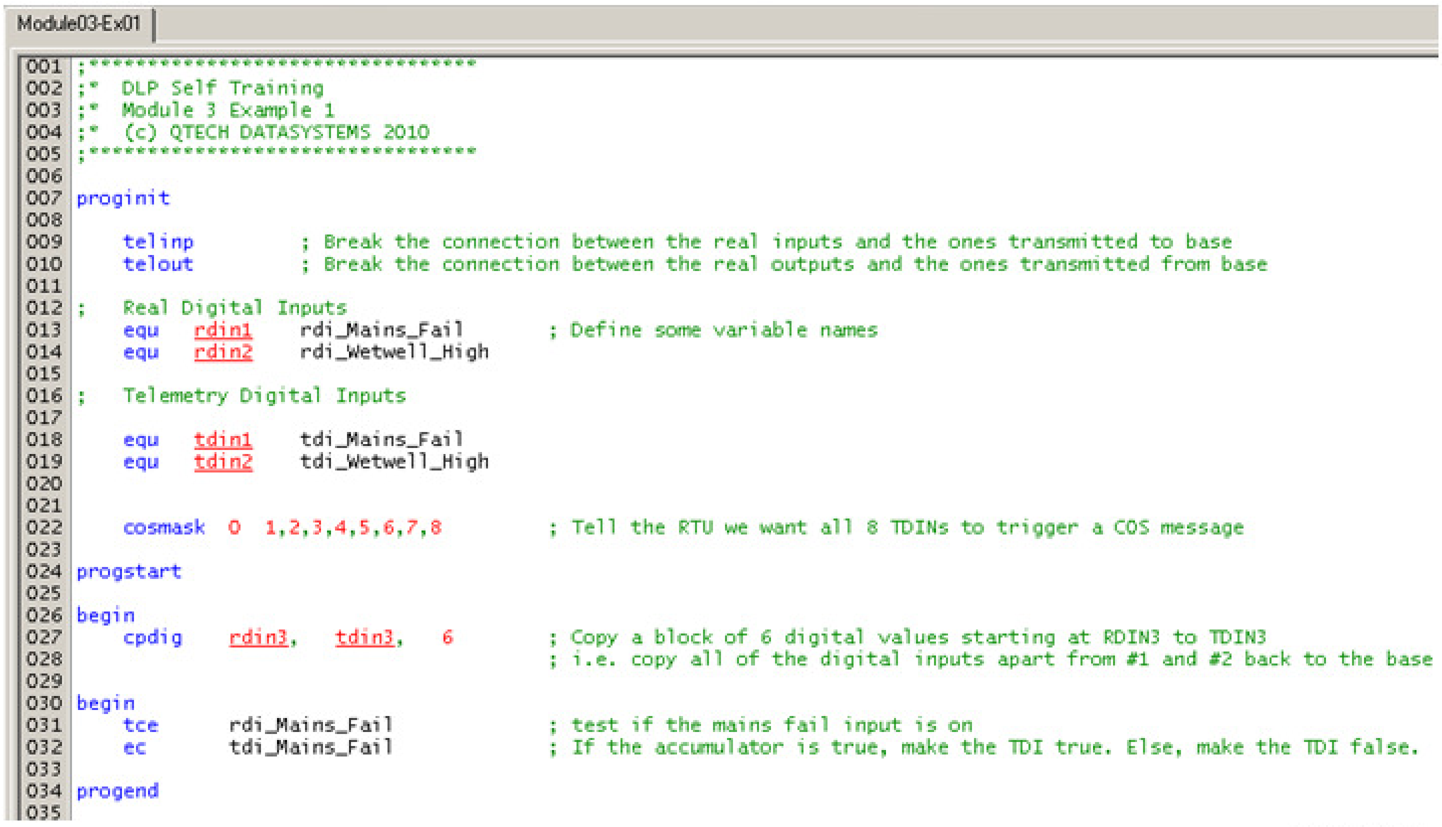
The most basic need for any DLP is to be able to evaluate the current state of a digital input and then to make choices depending on the state of that input. The TCE and TCD commands are used to do just that.

TCE stands for “Test if contact is energised”, in other words “Test to see if the digital register is currently ON”.

TCD is very similar, but subtly different. “Test if contact is de-energised” or “Test to see if the digital register is currently OFF”.

Once the status of the register in question has been evaluated, the logic accumulator (as discussed in Module 02) will be loaded with the result of the evaluation. Note that the result of the TCE and TCD commands also relies on the state of the logic accumulator as well as the state of the register that is being evaluated. It is therefore more accurate to think of TCE as “Test to see if the logic accumulator is true, AND if the digital register is <u>on</u>, and then load the result into the logic accumulator.” Likewise, TCD is “Test to see if the logic accumulator is true, AND if the digital register is <u>off</u>, and then load the result into the logic accumulator.”

Once the logic accumulator is set, the result then affects the behaviour of subsequent commands in the DLP.



```
001 ;**********************************
002 ;*  DLP Self Training
003 ;*  Module 3 Example 1
004 ;*  (c) QTECH DATASYSTEMS 2010
005 ;**********************************
006
007 proginit
008
009     telinp          ; Break the connection between the real inputs and the ones transmitted to base
010     telout          ; Break the connection between the real outputs and the ones transmitted from base
011
012 ;   Real Digital Inputs
013     equu    rdin1     rdi_Mains_Fail         ; Define some variable names
014     equ     rdin2     rdi_Wetwell_High
015
016 ;   Telemetry Digital Inputs
017
018     equ     tdin1     tdi_Mains_Fail
019     equ     tdin2     tdi_Wetwell_High
020
021
022     cosmask  0  1,2,3,4,5,6,7,8          ; Tell the RTU we want all 8 TDINs to trigger a COS message
023
024 progstart
025
026 begin
027     cpdig     rdin3,    tdin3,    6       ; Copy a block of 6 digital values starting at RDIN3 to TDIN3
028                                           ; i.e. copy all of the digital inputs apart from #1 and #2 back to the base
029
030 begin
031     tce       rdi_Mains_Fail              ; test if the mains fail input is on
032     ec        tdi_Mains_Fail              ; If the accumulator is true, make the TDI true. Else, make the TDI false.
033
034 progend
035
```

Example 1

Example 1 above shows most of the features that have been discussed so far in other modules, plus a section that makes use of the TCE and the EC command (more on EC in the next section)

Lines 30 to 32 of this DLP test to see if the real digital input for the mains fail is on, and then writes an ON or and OFF value to the telemetry depending on the result of the TCE command using EC

EC and ECC

Now that we know how to check for the status of various digital values, we need to know how to actually use that information to *do* something. This is where EC and ECC come in.

EC stands for “Energise Contact”, or “Turn the named register ON”.

Both EC and ECC copy the status of the Logic Accumulator to the named register. This means that the register will be turned ON if the logic accumulator is TRUE, and will be turned off if the logic accumulator is FALSE.

The difference between EC and ECC is that EC is designed to be used at the end of a logic rung, and will reset the logic accumulator to TRUE after it has executed. In 90% of all situations this will be suitable, however sometimes you will want to use EC to write to a register but then for the logic rung to continue. This is where ECC comes in. (“Energise contact and then continue”). ECC will copy the state of the logic accumulator to the nominated digital register, but will leave the state of the logic accumulator unchanged so the rung may continue.

If we refer to DLP example #1 above, lines 30 to 32 could be written in English:

“Test to see if the logic accumulator is TRUE and if the input we have named rdi_Mains_Fail has the value ON/TRUE. After this has been evaluated, copy the result into the register we have named tdi_Mains_Fail, and reset the logic accumulator to TRUE.”

Be careful when using EC (or ECC) in conjunction with TCD as the result may not be what you were expecting.
If example #1 used TCD instead of TCE, the value that was written to the telemetry input would be the OPPOSITE of the real input, not the same!

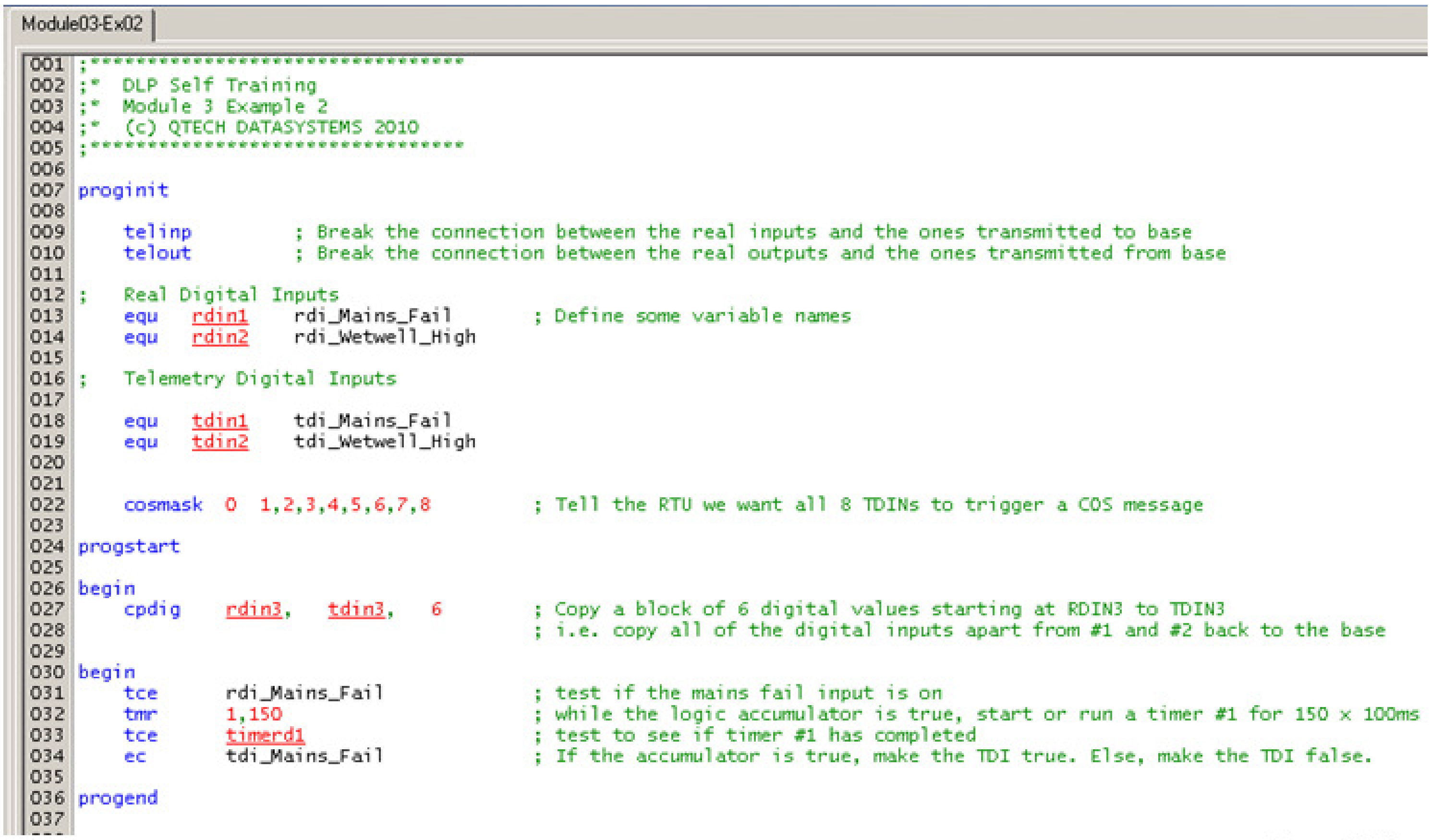
SETDIG and RESDIG

The SETDIG and RESDIG commands are similar to EC except that they only take action when the logic accumulator is TRUE.

SETDIG will cause the nominated digital register to be set to ON/TRUE

RESDIG will cause the nominated digital register to be set to OFF/FALSE

If the logic accumulator is FALSE, then both SETDIG or RESDIG will do nothing.



Module03-Ex02

```
;**********************************
;*  DLP Self Training
;*  Module 3 Example 2
;*  (c) QTECH DATASYSTEMS 2010
;**********************************

proginit

    telinp          ; Break the connection between the real inputs and the ones transmitted to base
    telout          ; Break the connection between the real outputs and the ones transmitted from base

;   Real Digital Inputs
    equ   rdin1     rdi_Mains_Fail        ; Define some variable names
    equ   rdin2     rdi_Wetwell_High

;   Telemetry Digital Inputs

    equ   tdin1     tdi_Mains_Fail
    equ   tdin2     tdi_Wetwell_High


    cosmask  0  1,2,3,4,5,6,7,8           ; Tell the RTU we want all 8 TDINs to trigger a COS message

progstart

begin
    cpdig    rdin3,   tdin3,    6         ; Copy a block of 6 digital values starting at RDIN3 to TDIN3
                                          ; i.e. copy all of the digital inputs apart from #1 and #2 back to the base

begin
    tce      rdi_Mains_Fail               ; test if the mains fail input is on
    tmr      1,150                        ; while the logic accumulator is true, start or run a timer #1 for 150 x 100ms
    tce      timerd1                      ; test to see if timer #1 has completed
    ec       tdi_Mains_Fail               ; If the accumulator is true, make the TDI true. Else, make the TDI false.

progend

```

Example 2

TMR and TMV (DLP Timers)
Now, to the heart of the matter: How to run a timer in a DLP.

Example 2 builds on Example 1 but now we can see that the Tmr function has been used and the DLP reads the TimerD1 system register before setting the telemetry at the base station.

The DLP has a stack of 128 timer registers available for the programmer to use for various timing functions throughout the DLP.

Syntax :	TMR	[Timer Register Number],	[Timer Duration x 100 ms]
	TMV	[Timer Register Number],	[Analog Register x 100 ms]

TMR runs a timer using a constant as the timer duration, TMV runs a timer but allows the programmer to pass a variable to the timer function so the duration can be changed at runtime.

Three things can happen when the TMR or TMV command is executed:

- If the timer is not already running, and the Logic Accumulator is TRUE, the timer will be started and the counter duration will be set.
- If the timer has already started, and the Logic Accumulator is TRUE, the timer will be maintained and the timer counter will decrement accordingly.
- If the Logic Accumulator is FALSE, the timer will be stopped and reset.

Each timer has three system variables associated with it. These are TIMERD1, TIMERS1 and TIMERC1, where 1 is the number of the counter register, so can be from 1 to 128.

TIMERS1 is a digital value (Timer Started). This register becomes true when the timer starts and remains true until the timer is reset by TMR or TMV command being executed when the Logic Accumulator is false.

TIMERD1 is also a digital value (Timer Done). This register becomes true when the timer has been running long enough for the timer counter to expire. It will remain true until the timer is reset by TMR or TMV command being executed when the Logic Accumulator is false.

TIMERC1 is an analogue register (Timer Count). This register is loaded with the timer duration when the timer is started and will be decremented by the RTU firmware until it reaches zero, and is set. It will remain at zero until the next time the timer is started.

So, as a step-by-step English description of the DLP above:

- Test to see if the mains fail input is ON
- If it is on, start (or maintain) a timer on timer register #1. We want it to run for 15 seconds. If the mains fail input is off, shut down the timer and reset it immediately.
- Test to see if TIMERD1is true, in other words, test to see if the timer has run to completion.
- If the timer has completed, set the telemetry back to the base station for the Mains Fail input, else if the timer has not yet completed OR it is not running at all, then set the telemetry to FALSE.

The behaviour this will cause is that the mains fail input will be delayed in its transmission back to the base station by 15 seconds when it becomes TRUE, but will transmit immediately back to the base station when it becomes false.

This is the simplest and most widely used form of debounce in a DLP, and will be suitable for most applications, such as Mains Fail, Float Switch, Pressure Switch and Door Switch Inputs.

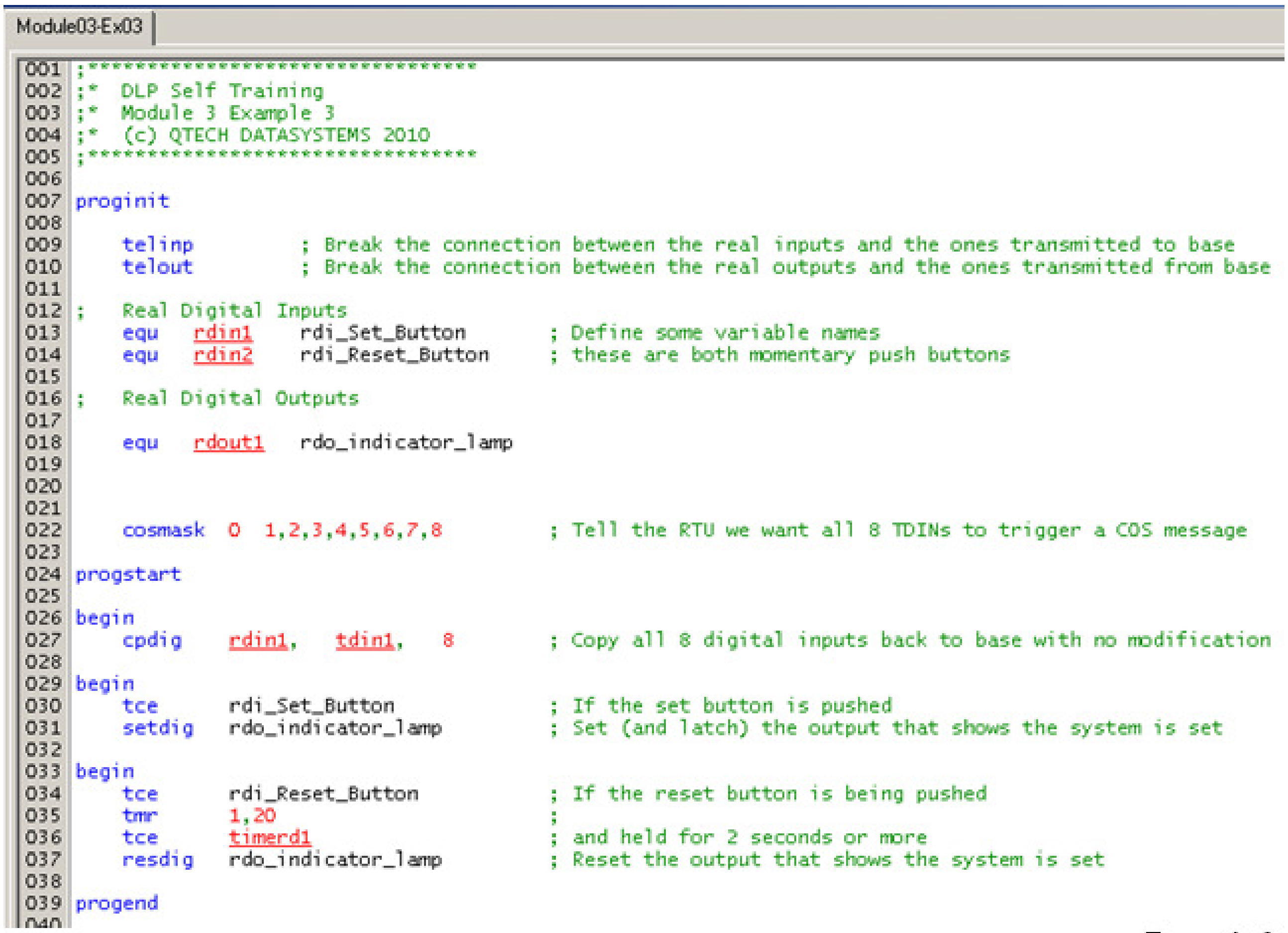
More on SETDIG and RESDIG

Example 3 shows the use of SETDIG and RESDIG to take an action based on the value of a digital input to either set or reset a digital output on the RTU. In this particular case, RDIN1 and RDIN2 both have momentary pushbuttons installed. RDIN1 is used to set the output on the RTU, while RDIN2 is used to reset the same output.

This DLP also requires that the reset button be held for a length of time before the output is reset to prevent accidental switching.

This raises the question of contention, since we now have two different routines that will need to write to the same digital output, potentially at the same time. One might wonder which will take precedence?

The DLP executes sequentially from the top to the bottom, and the state the DLP has modified are only written at the end of the DLP execution. So, the LAST operation in the DLP that writes to a specific output will have the control over what that output does, even if several other parts of the DLP have also written to it previously.



```
;**********************************
;*  DLP Self Training
;*  Module 3 Example 3
;*  (c) QTECH DATASYSTEMS 2010
;**********************************

proginit

    telinp          ; Break the connection between the real inputs and the ones transmitted to base
    telout          ; Break the connection between the real outputs and the ones transmitted from base

;   Real Digital Inputs
    equu   rdin1    rdi_Set_Button      ; Define some variable names
    equ    rdin2    rdi_Reset_Button    ; these are both momentary push buttons

;   Real Digital Outputs

    equ    rdout1   rdo_indicator_lamp



    cosmask  0  1,2,3,4,5,6,7,8         ; Tell the RTU we want all 8 TDINs to trigger a COS message

progstart

begin
    cpdig    rdin1,    tdin1,    8      ; Copy all 8 digital inputs back to base with no modification

begin
    tce      rdi_Set_Button             ; If the set button is pushed
    setdig   rdo_indicator_lamp         ; Set (and latch) the output that shows the system is set

begin
    tce      rdi_Reset_Button           ; If the reset button is being pushed
    tmr      1,20                       ;
    tce      timerd1                    ; and held for 2 seconds or more
    resdig   rdo_indicator_lamp         ; Reset the output that shows the system is set

progend
```

Example 3