# Multi-Tasking Allows Simultaneous Operation of Motion & Non-Motion Tasks

**Objective**
Simultaneously process non-motion tasks in addition to performing dedicated motion profiles to provide a solution for integrated application tasks within the same control environment.

**Overview**
Many applications require user created “non-motion related” or “background” task functionality to solve applications. The PTi-210 offers a multi-tasking environment that allows up to four separate tasks to run at the same time. The processor in the PTi-210 effectively splits its available processing resources between the various user initiated tasks in addition to the associated motion and communications related overhead tasks that occur normally. The example below illustrates the PTi-210’s multi-tasking capabilities in the form three separate programs that run independently on separate tasks that allow the user to convince themselves that the PTi-210 is multi-tasking the three programs. The PTi-210 can run up to four separate tasks simultaneously.

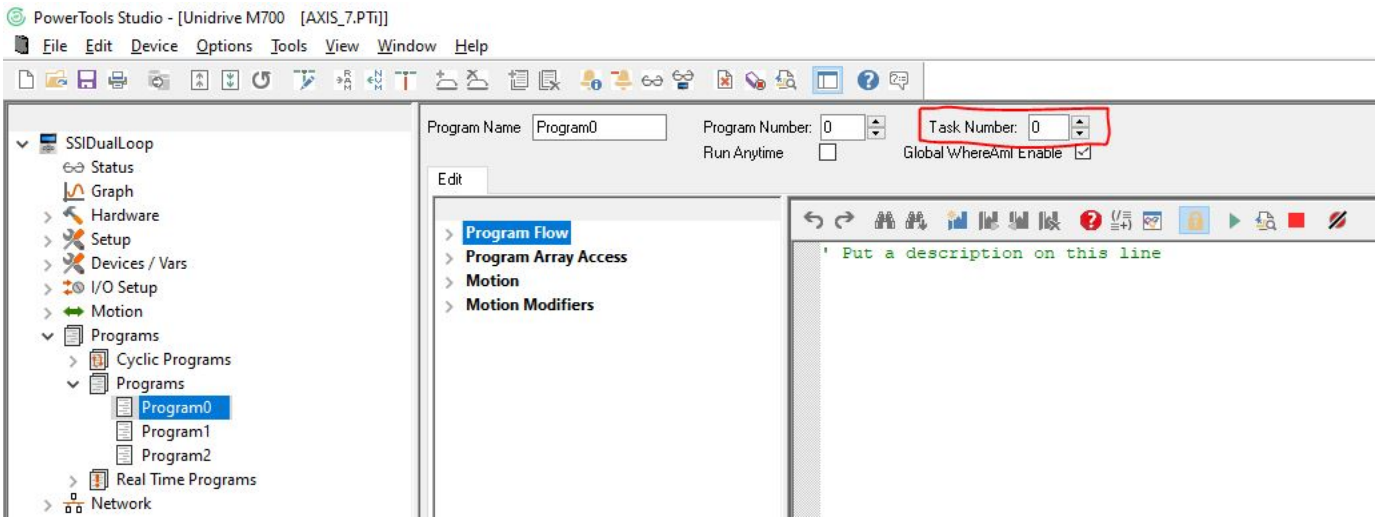
**Example – 3 Programs Running on 3 Separate Tasks**

Program1 – is a program that initiates Index0 followed by a 0.5 second dwell. DriveInput4 initiates Program1.

Program2 – is a program that counts the number of times that Index0 completes. DriveInput5 initiates Program2.

Program3 – is a program that turns DriveOutput1, DriveInput5, and DriveOutput3 on and off. DriveInput6 initiates Program3.

**Solution Summary**
The solution summary for this Example is very simple. The three programs detailed above are created and assigned to separate tasks. (For numbering clarity relative to the DriveInput numbers (4-6) used, Program0 is not used in the solution.) By default, all PTi-210 programs are designated to run on Task0. This can be verified in the PowerTools Studio programming screen as shown here:

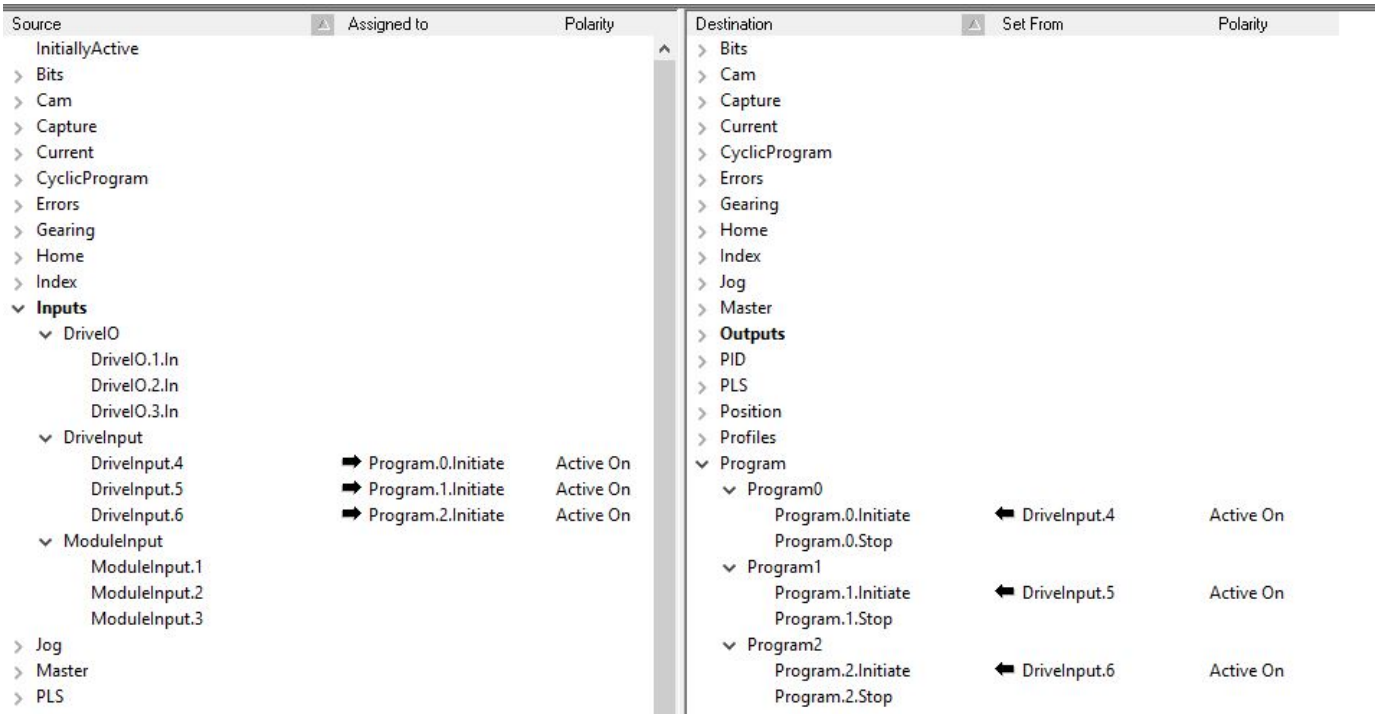



Control Techniques a division of Nidec Motor Corporation
7078 Shady Oak Road, Eden Prairie, MN 55344-3505 USA
T: +1 952 995-8000 www.controltechniques.us

If all PTi-210 programs are assigned to the same task number, the PTi-210 is not being used in a multi-tasking fashion. To create a multi-tasking environment, separate task numbers must be assigned to the various program numbers that are to be run at the same time (multi-tasked). This is done very easily in the program view window by changing the task number through the use of the arrow buttons or by typing a number (0-3) for the Task Number you wish to assign to a given program.

### I/O Assignments

Only three assignments are used in this example. DriveInputs 4, 5 & 6 are used to initiate Programs 1, 2 & 3.




### Program Code Used

```
'------------------------------------------------------------------------
'Description- Multi-tasking Motion and Non-Motion Tasks                 -
'Program 1 (Running on Task 1)                                          -
'Revision 1 – 3/30/2021, Created using PowerTools Studio               -
'Min. Software Level Required:  PowerTools Studio v01.01.00.20         -
'Min. Firmware Required: v01.01.00.20                                  -
'------------------------------------------------------------------------

'Initialize Variables
Index.0.Dist=10.000

'Main Program
Do While DriveInput.4 = ON
    Index.0.Initiate
    Wait For Index.AnyCommandComplete
    Dwell For Time 0.5 'seconds
Loop

End
```

```
'--------------------------------------------------------------------
'Description- Multi-tasking Motion and Non-Motion Tasks              -
'Program 2 (Running on Task 2)                                      -
'Revision 1 – 3/30/2021, Created using PowerTools Studio            -
'Min. Software Level Required:  PowerTools Studio v01.01.00.20      -
'Min. Firmware Required: v01.01.00.20                               -
'--------------------------------------------------------------------

'Initialize Variables
Var.Indx0Complts=0

'Main Program
Do While DriveInput.5 = ON
    Wait For Index.0.CommandComplete
    Var.Indx0Complts=Var.Indx0Complts + 1
    Index.1.Dist=Var.Indx0Complts
Loop

End
```

```
'--------------------------------------------------------------------
'Description- Multi-tasking Motion and Non-Motion Tasks              -
'Program 3 (Running on Task 3)                                       -
'Revision 1 – 3/30/2021, Created using PowerTools Studio            -
'Min. Software Level Required:  PowerTools Studio v01.01.00.20       -
'Min. Firmware Required: v01.01.00.20                                -
'--------------------------------------------------------------------

'Main Program
Do While DriveInput.6 = ON
    DriveIO.1.Out = ON
    Dwell For Time 0.2 'seconds
    DriveIO.1.Out = OFF
    DriveIO.2.Out = ON
    Dwell For Time 0.2 'seconds
    DriveIO.2.Out = OFF
    DriveIO.3.Out = ON
    Dwell For Time 0.2 'seconds
    DriveIO.3.Out = OFF
Loop

End
```

**Description of Program Code**

Program1 – is the only program running on Task1 and it is initiated by turning DriveInput4 ON. The Index0 distance is initialized to ten revolutions at the start of the program. The rest of the program consists of an index and dwell loop that runs continuously as long as DriveInput4 stays ON. A “Do While Loop” is used to implement the looping structure that checks the status of DriveInput4 to verify it is ON before starting the next index and dwell loop. When DriveInput4 turns OFF, the loop discontinues and the program ends.

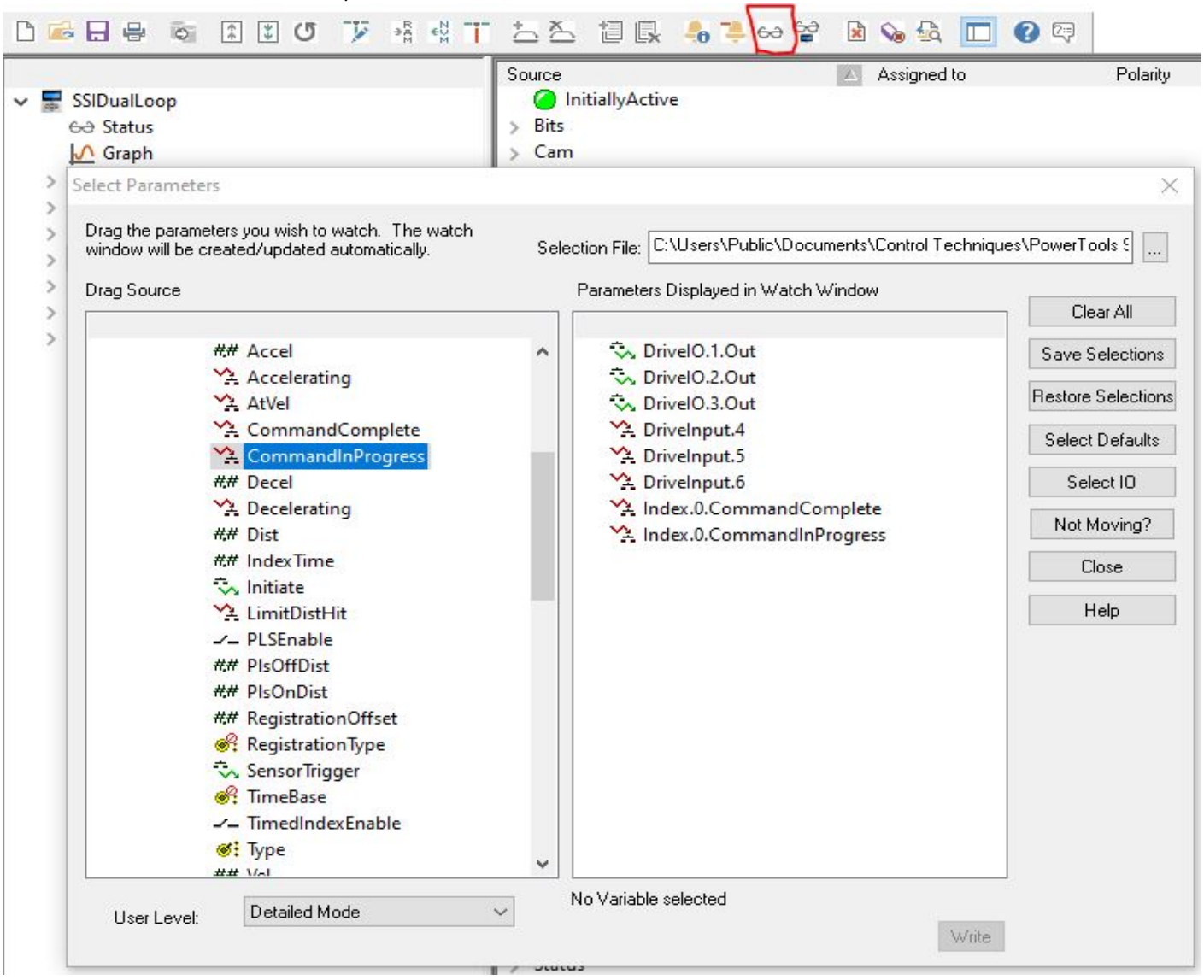
Program2 – is the only program running on Task2 and it is initiated by turning DriveInput5 ON. The user variable to count the number of times Index0 completes (`Indx0Complts`) is initialized to zero at the start of the program. The rest of the program is consists of a wait and count loop that runs continuously as long as Drive Input2 stays ON. As in Program1, a “Do While Loop” is used to run a loop that runs as long as DriveInput5 is ON. After checking DriveInput5, program execution waits until the Index.0.CommandComplete parameter is ON. This will occur as a result of the Index0 in Program1 completing its profile. The counter variable (`Indx0Complts`) is incremented by one and then the value of that variable is then stored in the Index.1.Dist parameter because that parameter can be accessed from the PTi-210 Keypad/Display.

Program3 – is the only program running on Task3 and it is initiated by turning DriveInput6 ON. The program consists of a turn outputs ON and OFF loop that runs continuously as long as DriveInput6 stays ON. As in Program1&2, a “Do While Loop” is used to run a loop that runs as long as DriveInput6 is ON. The loop itself turns on DriveOutputX ON, dwells for 0.2 seconds, and then turns DriveOutputX OFF. (X =1,2, or 3)

**Testing**

A combination of tools can be used to verify the multi-tasking operation of this example. First, turn ON DriveInputs 4,5, & 6 and leave them on. The easiest way to verify that Program1/Task1 is running is to watch the motor shaft. The motor will continuously index and dwell as long as long as Program1/Task1 is running. The easiest way to verify that Program2 is running is to read the keypad display value for the number of completed indexes. This value is stored in the Index.1.Dist parameter and will be incrementing by one for every time the motor stops if Program1/Task1 is running and will not change if Program1/Task1 is stopped. The easiest way to verify that Program3/Task3 is running it to watch the status of DriveOutputs 1,2, & 3. The outputs will turn on and off one at a time in order if Program3/Task3 is running and will not change at all if Program3/Task3 is not running.

Once that you have convinced yourself that all three programs/tasks are running, alternate turning one or two the Drive Inputs off in various combinations to further reinforce the multitasking operation. Another easy alternative to verify all three tasks are running is to look for the same results described above while using the Watch Window Feature in PowerTools Studio software. See the screen below for the list of parameters to view using the Watch Window for this example.




| Watch Window | |
|---|---|
| Axis Name | SSIDualLoop |
| Axis Address | 1 |
| | |
| DriveIO.1.Out | False |
| DriveIO.2.Out | False |
| DriveIO.3.Out | False |
| DriveInput.4 | False |
| DriveInput.5 | False |
| DriveInput.6 | False |
| Index.0.CommandComplete | False |
| Index.0.CommandInProgress | False |