

# **DIRECTLOGIC**

## **PROGRAMMING EXAMPLES**

---



### **In This Chapter...**

Programming Examples Overview .....	8-2
Load and Run a Pulse Profile .....	8-3
Dynamic Positioning Profile .....	8-4
Dynamic Velocity Profile .....	8-5
Velocity Mode .....	8-6
Run to Position Mode .....	8-7
Dynamic Positioning Profile .....	8-8
System Functions Examples Overview .....	8-9
Simulating Retentive Counter .....	8-10
Reading CTRIO Internal Registers .....	8-11

## Programming Examples Overview



**NOTE:** The programming examples on the following pages are provided “as is” without a guarantee of any kind. This Chapter is provided by our technical support group to assist others. We do not guarantee the examples are suitable for a particular application, nor do we assume any responsibility for them in your application. Chapter 6 “Program Control” contains flowcharts that provide detailed steps needed to execute a pulse profile or System Functions command.

The *Direct*LOGIC programming examples provided on the following pages are simple examples that are intended to assist you in the basics of loading and running various output pulse profiles. The examples are complete enough to load a profile, process the command and load the Parameter registers necessary to execute the profile. Two System Functions examples are also provided.

### **Load and Run a Pulse Profile example:**

You will need to have a Trapezoid, S-Curve, Symmetrical S-Curve or Home Search profile configured using the Configure I/O dialog. You will need to have the appropriate Pulse Profile Table File Number (decimal) stored in V3000 for this example. You must turn C0 on to load and run the pulse profile. C2 controls the pulse output direction.

### **Dynamic Positioning Profile example:**

You will need to have a Dynamic Positioning profile configured as Table File Number 1 using the Configure I/O dialog. You will need to have the appropriate Target Pulse Count Position (signed decimal) stored in V3000 for this example. You must turn C0 on to initialize the settings. Then turn C2 on to Go to Position.

### **Dynamic Velocity Profile example:**

You will need to have a Dynamic Velocity profile configured as Table File Number 1 using the Configure I/O dialog. You will need to have the appropriate Target Velocity (signed decimal) stored in V3000 for this example. You must turn C0 on to initialize the settings and enable the output.

### **Velocity Mode, Run to Limit Mode and Run to Position Mode examples:**

No CTRIO Pulse Profile Tables are necessary to execute these profiles, but the Outputs need to be configured for Step/Direction or CW/CCW using the Configure I/O dialog. All parameters are stored in V-memory as shown in the examples. You must turn C0 on to initialize the settings and to run the pulse profile. C2 controls the pulse output direction.

### **Simulating Retentive Counter example:**

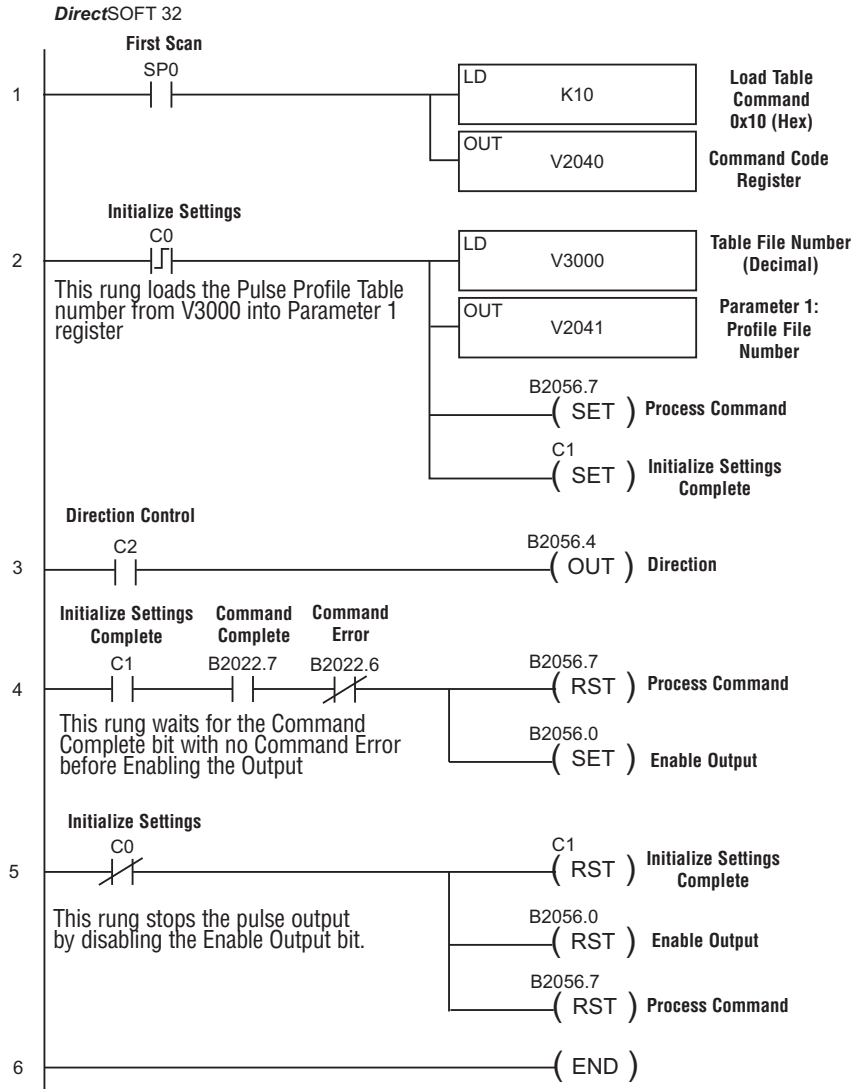
This Systems Functions example uses the Write to Intelligent (WT) instruction to write the current count stored in the PLC's retentive memory to the CTRIO's current count register on a power cycle or a RUN-STOP-RUN PLC mode change. No permissive bits are required to be turned on in this example.

### **Reading CTRIO Internal Registers example:**

This Systems Functions example uses the Write to Intelligent Module (WT) and Read from Intelligent Module (RD) instructions to read all of the CTRIO's internal registers every 900ms. You must turn C0 on to initialize the settings to perform the Read routine.

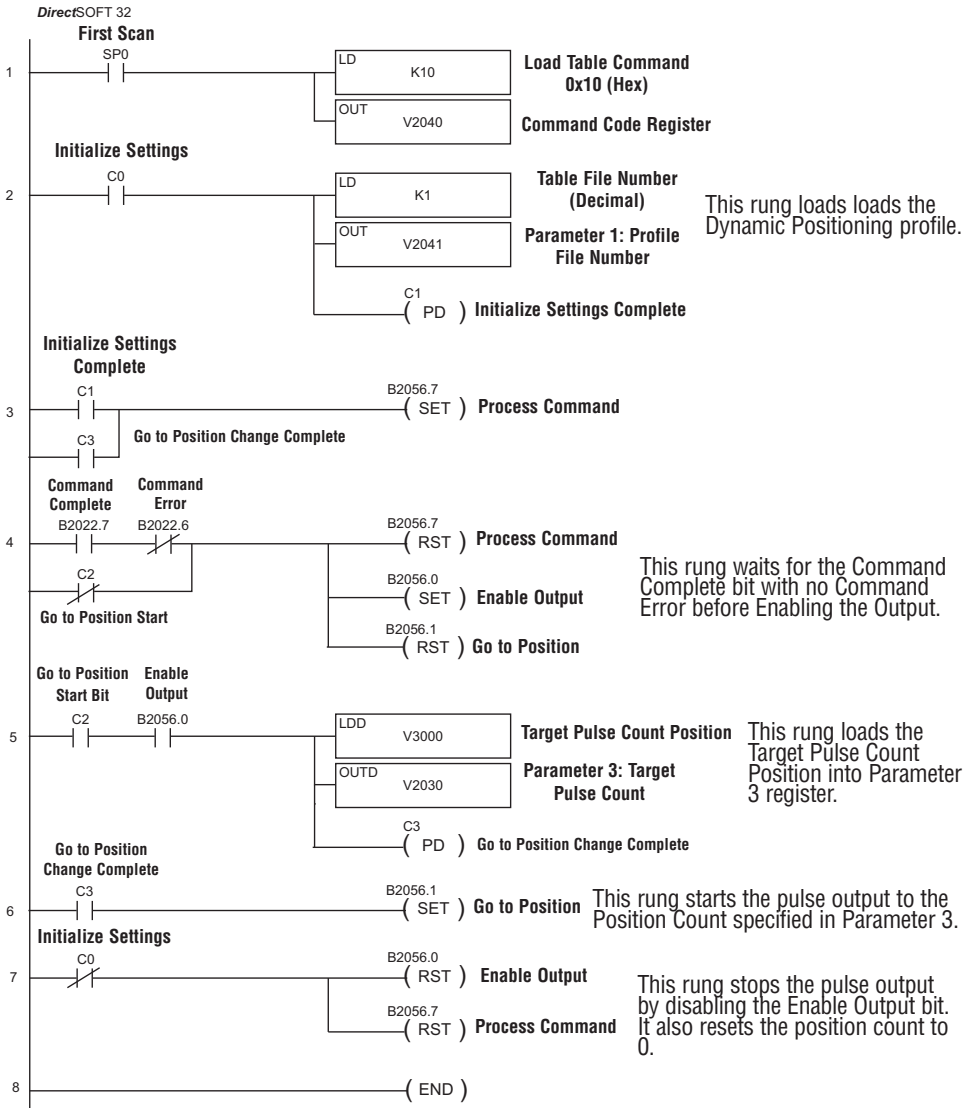
## Load and Run a Pulse Profile

The following example program loads and executes a Pulse Profile that was created using CTRIO Workbench Pulse Profiles dialog . This example can be used for Trapezoid, S-Curve, Symmetrical S-Curve and Home Search profiles (Home Search requires that CTRIO inputs C and/or D are configured for Limit Out 0 and/or Limit Out 2). The Pulse Profile number is stored in V3000 for this example. Turning on C0 will load and run the pulse profile.



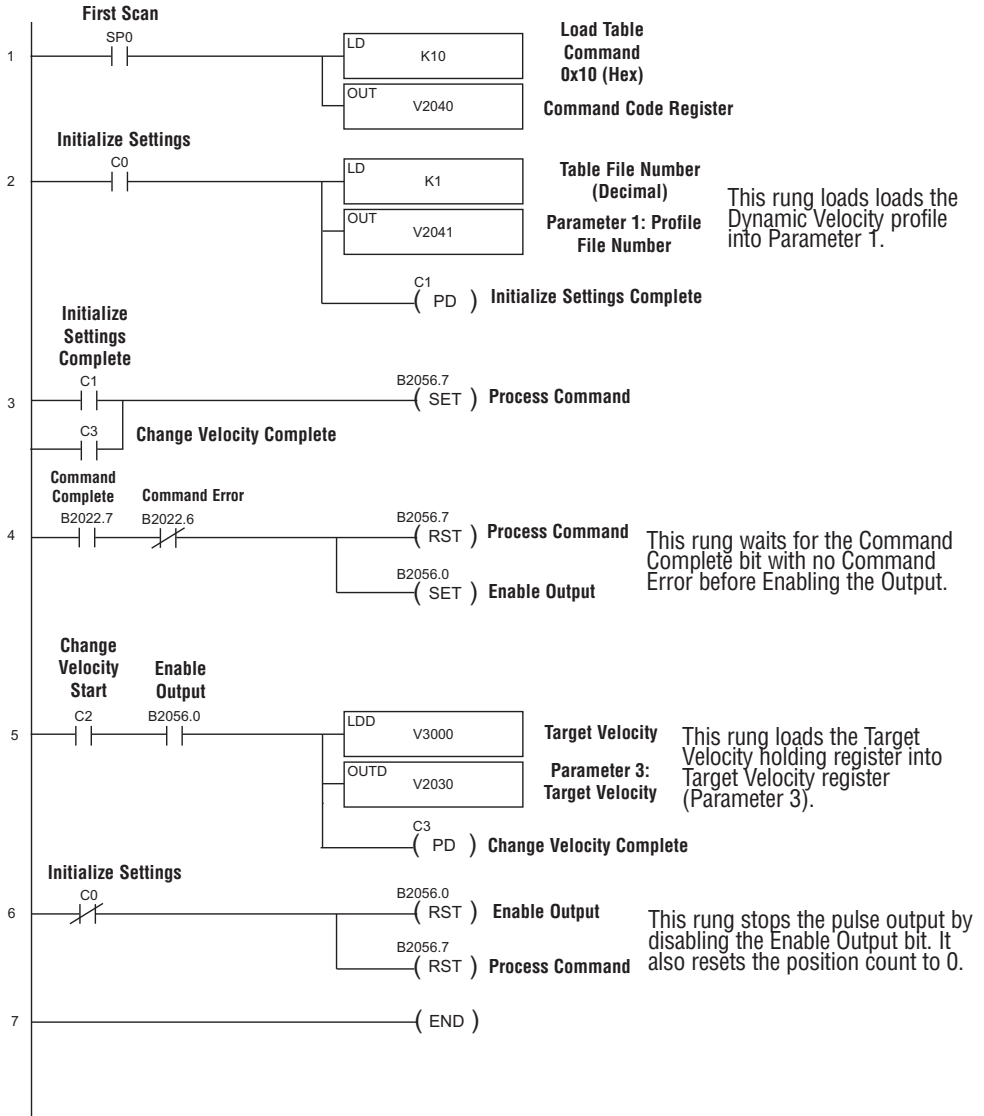
## Dynamic Positioning

The following example program runs a Dynamic Positioning pulse profile. Turn on C0 to load the profile number and process the command. Turning on C2 will start the pulse output to position specified in Parameter 3. The sign of the value in the target pulse count register controls the pulse output direction. When a new position is specified, the CTRIO moves to the new position relative to its previous position as long as the Enable Output bit remains set. Clearing the Enable Output bit will disable output pulsing and reset the current position to 0.



# Dynamic Velocity

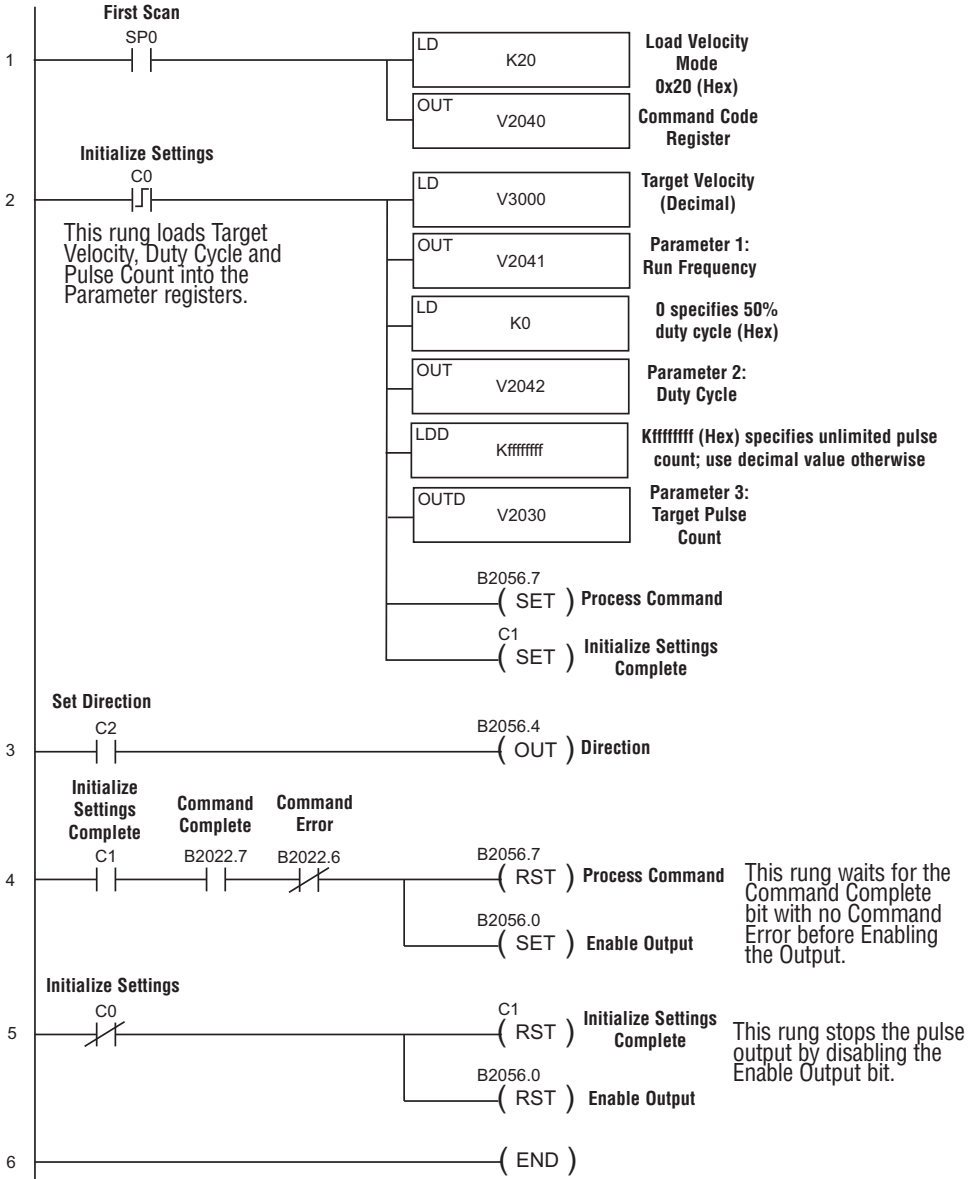
The following example program executes a Dynamic Velocity pulse profile. Turn C0 on to load the profile number, process the command and enable the output. The Target Velocity needs to be specified in V3000. The velocity can be changed “on the fly” by entering a different value into V3000. The sign of the value in the target velocity register controls the pulse output direction. Clearing the Output Enable bit will always suspend pulsing.



# Velocity Mode

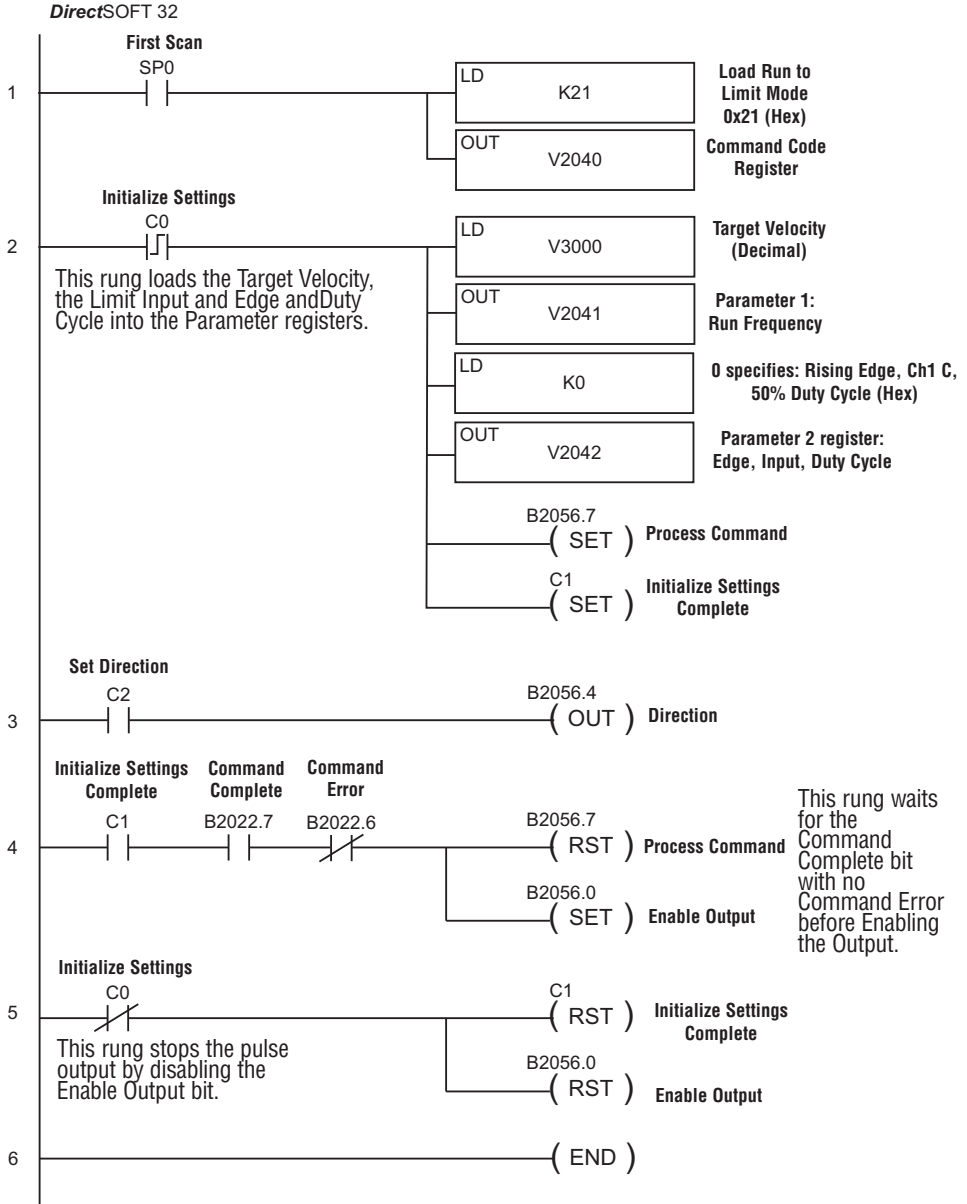
The following example program loads and executes a Velocity Mode pulse profile. For Parameter 3, a specific number of pulse output counts can be specified or if set to “ffffff” Hex, the pulse output will remain ON at the specified Target Velocity until the output is disabled.

DirectSOFT 32



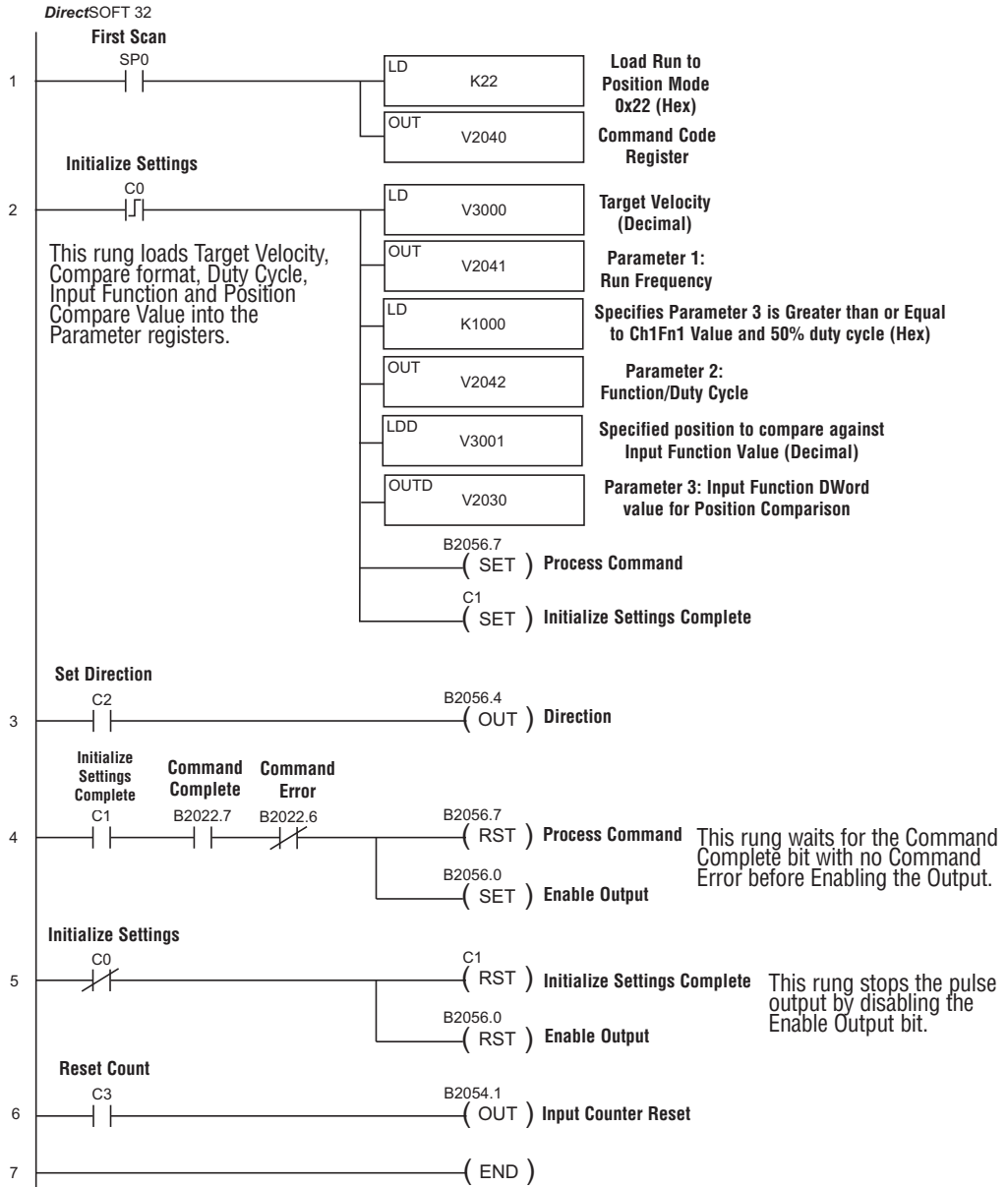
## Run to Limit Mode

The following example program loads and executes a Run to Limit Mode pulse profile. Turn on C0 to run the profile. CTRIO input C or D must be assigned to Limit for this profile.



## Run to Position Mode

The following example program loads and executes a Run to Position Mode pulse profile. Turn on C0 to run the pulse profile.



## System Functions Examples Overview




---

**NOTE:** System Functions are supported only when the CTRIO module is installed in the same base as the *DirectLOGIC* CPU.

---

The Systems Functions examples on the following pages use the *DirectLOGIC* Write to Intelligent Module (WT) and/or Read from Intelligent Module (RD) instructions to write to or read from the CTRIO's internal registers.

### Reading From CTRIO Internal Memory

Reading the CTRIO's internal memory consists of several steps. Step one is using the WT instruction to send a Systems Function's command to the CTRIO telling it to put its internal register values into the CTRIO's "shared RAM". Step two is processing the request for the internal register values using the Process Command bit. Step three is using the RD instruction to read the values from the CTRIO's "shared RAM" memory into PLC V-memory.

Steps 1 and 2: WT instruction and Process Command

```

PLC V-memory ==> CTRIO's Shared RAM
CTRIO's Shared RAM ==> Process Command to internal processor
CTRIO's Shared RAM <== Internal data values

```

Step3: RD instruction

```

PLC V-memory <== CTRIO's Shared RAM

```

### Writing to CTRIO Internal Memory

Writing to the CTRIO's internal registers is basically a two step process. Step one is using the WT instruction to send a System Function's command and the desired data values to the CTRIO's "Shared RAM". Step two is using the Process Command bit to tell the CTRIO to process the command and data values that are in the CTRIO's Shared RAM. This moves the data values from the Shared RAM into the CTRIO's internal registers.

Steps 1 and 2: WT instruction (command and data) and Process Command Bit:

```

PLC V-memory ==> CTRIO Shared RAM
CTRIO Shared RAM ==> Process Command to internal processor
CTRIO Shared RAM ==> internal data registers

```




---

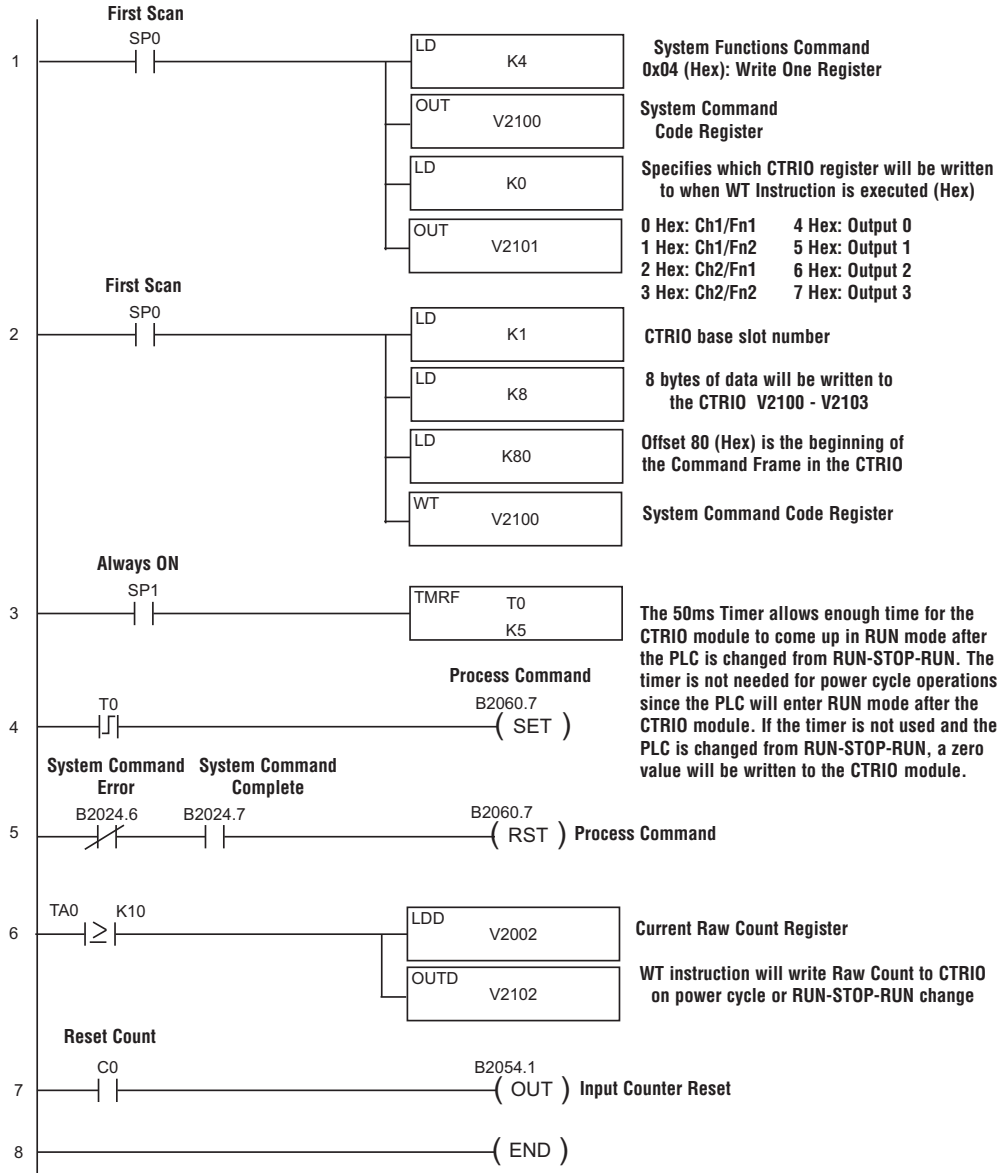
**NOTE:** This function is not available when the CTRIO module is installed in a EBC expansion base that also has an ERM installed in it.

---

# Simulating Retentive Counter

The following Systems Functions example uses the Write to Intelligent (WT) instruction to write the current count stored in the PLC's retentive memory to the CTRIO's current count register on a power cycle or a RUN-STOP-RUN PLC mode change.

DirectSOFT 32



8

# Reading CTRIO Internal Registers

The following Systems Functions example uses the Write to Intelligent Module (WT) and Read from Intelligent Module (RD) instructions to read all of the CTRIO's internal registers every 900ms and place the data starting at V2200.

