6

Drum Instruction Programming

(DL450 CPU only)

In This Chapter. . . .

Introduction

Purpose

✗	✗	✔
430	440	450

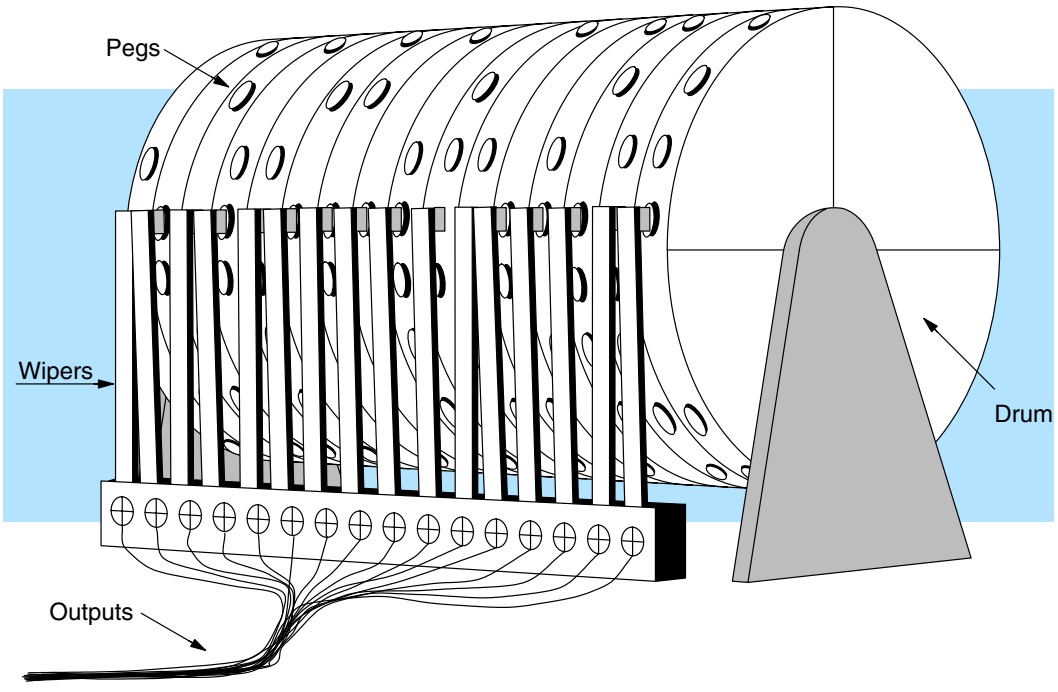
The four drum instructions available in the DL450 CPU electronically simulate an electro-mechanical drum sequencer. The instructions offer slight variations on the basic principle, which we describe first.

Drum Terminology

Drum instructions are best suited for repetitive processes that consist of a finite number of steps. They can do the work of many rungs of ladder logic with elegant simplicity. Therefore, drums can save a lot of programming and debugging time.

We introduce some terminology associated with drum instructions by describing the original electo-mechanical drum pictured below. The mechanical **drum** generally has pegs on its curved surface. The pegs are populated in a particular **pattern**, representing a set of desired actions for machine control. A motor or solenoid rotates the drum a precise amount at specific times. During rotation, stationary wipers sense the presence of pegs (present = on, absent = off). This interaction makes or breaks electrical contact with the wipers, creating electrical **outputs** from the drum. The outputs are wired to devices on a machine for On/Off control.

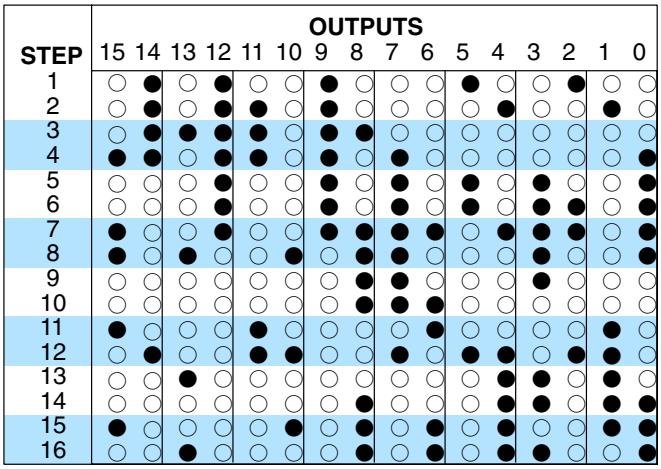
Drums usually have a finite number of positions within one rotation, called **steps**. Each step represents some process step. At powerup, the drum **resets** to a particular step. The drum rotates from one step to the next based on a **timer**, or on some external **event**. During special conditions, a machine operator can manually increment the drum step using a **jog** control on the drum's drive mechanism. The contact closure of each wiper generates a unique on/off pattern called a **sequence**, designed for controlling a specific machine. Because the drum is circular, it automatically repeats the sequence once per rotation. Applications vary greatly, and a particular drum may rotate once per second, or as slowly as once per week.



Electronic drums provide the benefits of mechanical drums and more. For example, they have a **preset** feature that is impossible for mechanical drums: The preset function lets you move from the present step *directly* to any other step on command!

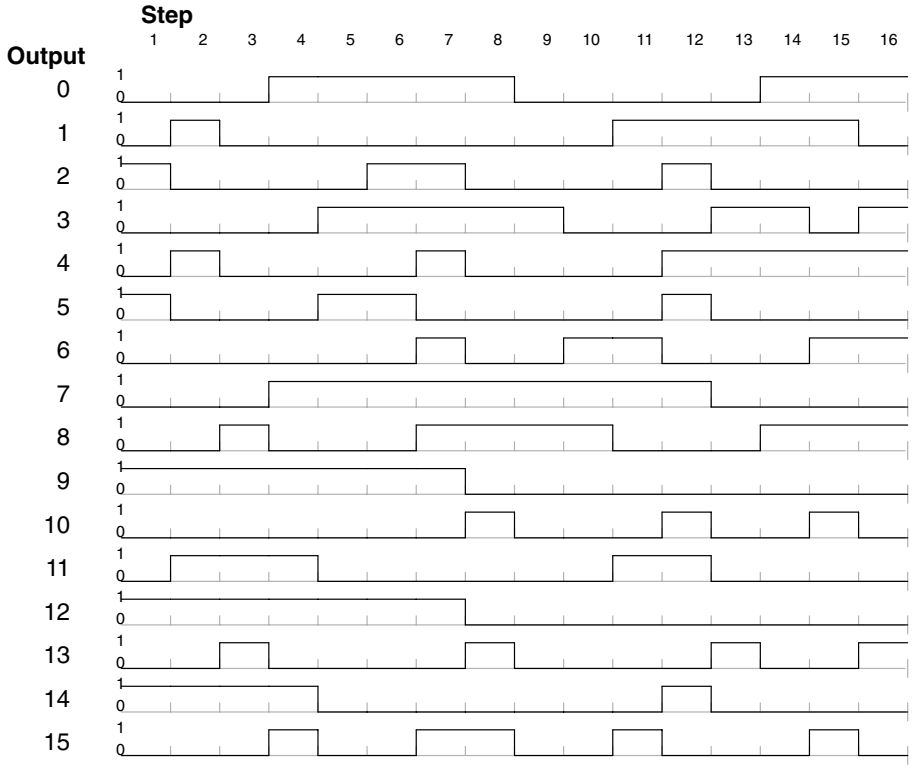
Drum Chart Representation

For editing purposes, the electronic drum is presented in chart form in ***Direct***SOFT32 and in this manual. Imagine slicing the surface of a hollow drum cylinder between two rows of pegs, then pressing it flat. Now you can view the drum as a chart as shown below. Each row represents a step, numbered 1 through 16. Each column represents an output, numbered 0 through 15 (to match word bit numbering). The solid circles in the chart represent pegs (On state) in the mechanical drum, and the open circles are empty peg sites (Off state).



Output Sequences

The mechanical drum sequencer derives its name from sequences of control changes on its electrical outputs. The following figure shows the sequence of On/Off controls generated by the drum pattern above. Compare the two, and you will find that they are equivalent! If you can see their equivalence, you are well on your way to understanding drum instruction operation.



Step Transitions

Drum Instruction Types

Drum instructions in the DL450 CPU consist of four types:

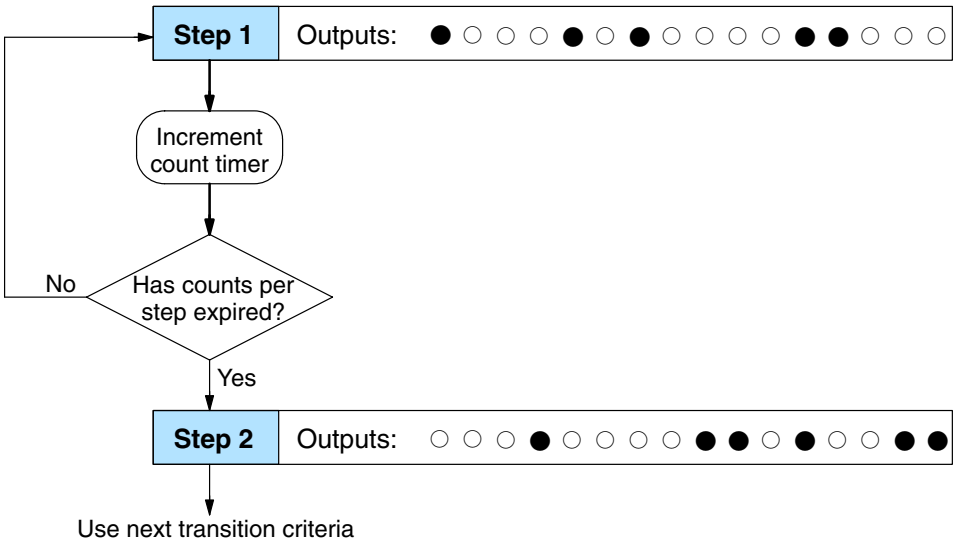
- Timed Drum with Discrete Outputs (DRUM)
- Time and Event Drum with Discrete Outputs (EDRUM)
- Masked Event Drum with Discrete Outputs (MDRMD)
- Masked Event Drum with Word Output (MDRMW)

The four drum instructions all include time-based step transitions, and three include event-based transitions as well. Other options include outputs defined as a single word or as individual bits, and an output mask (individual output disable/enable).

Now we will discuss how drum instructions work. Each drum has 16 steps, and each step has 16 outputs. Refer to the figure below. Each output can be either an X, Y, or C coil, offering a lot of programming flexibility. We assign Step 1 an arbitrary unique output pattern (○= Off, ●= On) as shown. When programming a drum instruction, you also determine both the output assignment and the On/Off state (pattern) at that time. All steps use the same output assignment, but each step may have its own unique output pattern.

Timer-Only Transitions

Drums move from step to step based on time and/or an external event (input). All four drum types offer timer step transitions, and three types also offer events. The figure below shows how timer-only transitions work.



The drum stays in each step for a specific duration (user-programmable). The timebase of the timer is programmable, from 0.01 seconds to 99.99 seconds. This establishes the resolution, or the duration of each “tick of the clock”. Each step uses the same timebase, but has its own unique counts per step, which you program. The drum spends a specific amount of time in each step, given by the formula:

Time in step = 0.01 seconds X Timebase x Counts per step

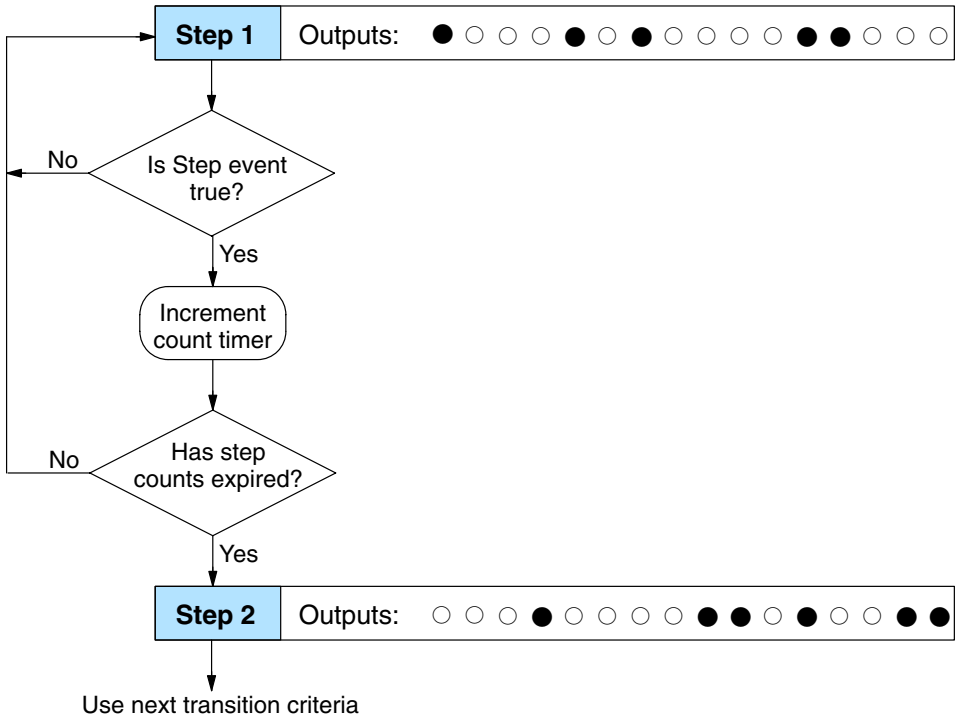
For example, if you program a 5 second time base and 12 counts for Step 1, then the drum will spend 60 seconds in Step 1. The maximum time for any step is given by the formula:

Max Time per step = 0.01 seconds X 9999 X 9999
= 999,800 seconds = 277.7 hours = 11.6 days

NOTE: When first choosing the timebase resolution, a good rule of thumb is to make it about 1/10 the duration of the shortest step in your drum. Then you will be able to optimize the duration of that step in 10% increments. Other steps with longer durations allow optimizing by even smaller increments (percentage-wise). Also, note that the drum instruction executes once per CPU scan. Therefore, it is pointless to specify a drum timebase that is much faster than the CPU scan time.

Timer and Event Transitions

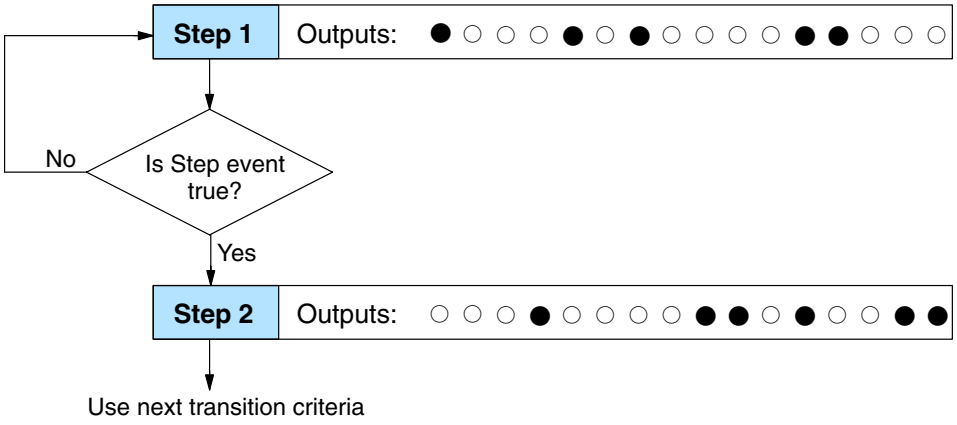
Time and Event Drums move from step to step based on time and/or external events. The figure below shows how step transitions work for these drums.



When the drum enters Step 1, it sets the output pattern as shown. Then it begins polling the external input programmed for that step. You can define event inputs as X, Y, or C discrete point types. Suppose we select X0 for the Step 1 event input. If X0 is off, then the drum remains in Step 1. When X0 is On, the event criteria is met and the timer increments. The timer increments as long as the event remains true. When the counts for Step 1 have expired, then the drum moves to Step 2. The outputs change immediately to match the new pattern for Step 2.

Event-Only Transitions

Time and Event drums do not have to have both the event and the timer criteria programmed for each step. You have the option of programming just one of the two, and even mixing transition types among all the steps of the drum. For example, you might want Step 1 to transition on an event, Step 2 to transition on time only, and Step 3 to transition on both time and an event. Furthermore, you may elect to use only part of the 16 steps, and only part of the 16 outputs.



Counter Assignments

Each drum instruction uses the resources of four counters in the CPU. When programming the drum instruction, you select the first counter number. The drum also uses the next three counters automatically. The counter bit associated with the first counter turns on when the drum has completed its cycle, going off when the drum is reset. These counter values and counter bit precisely indicate the progress of the drum instruction, and can be monitored by your ladder program.

Suppose we program a timer drum to have 8 steps, and we select CT10 for the counter number (remember, counter numbering is in octal). Counter usage is shown to the right. The right column holds typical values, interpreted below.

Counter Assignments

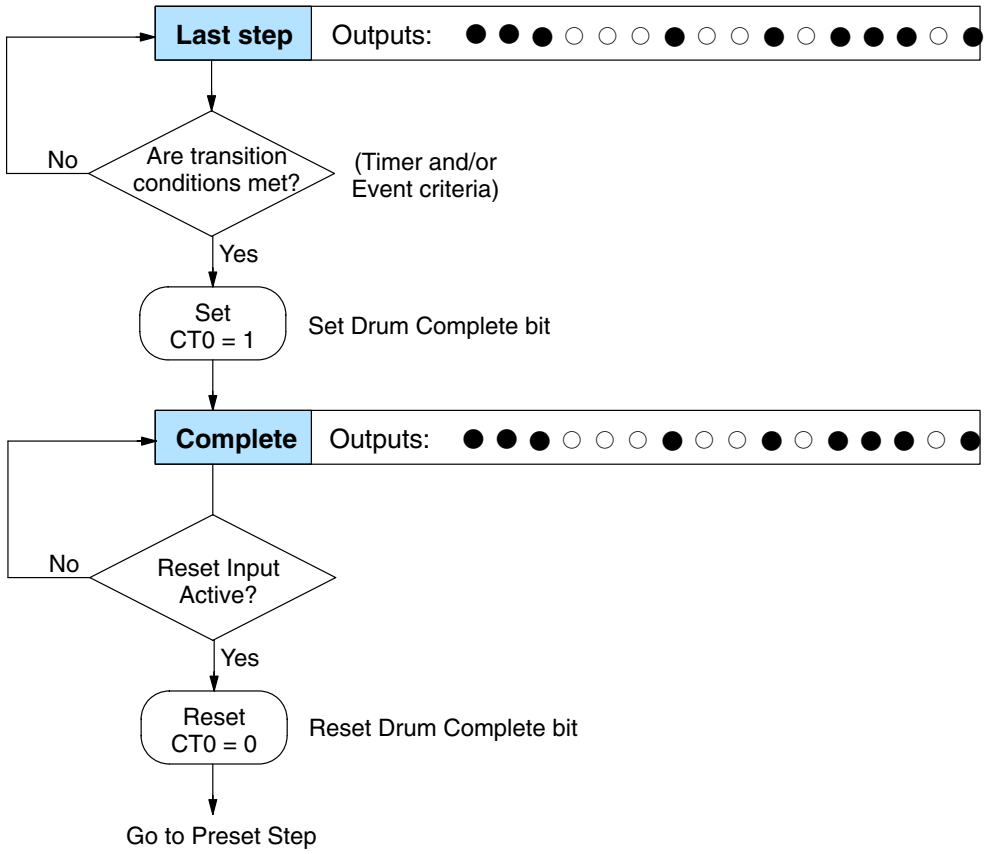
CT10	Counts in step	V1010	1528
CT11	Timer Value	V1011	0200
CT12	Preset Step	V1012	0001
CT13	Current Step	V1013	0004

CT10 shows that we are at the 1528th count in the current step, which is step 4 (shown in CT13). If we have programmed step 4 to have 3000 counts, then the step is just over half completed. CT11 is the count timer, shown in units of 0.01 seconds. So, each least-significant-digit change represents 0.01 seconds. The value of 200 means that we have been in the current count (1528) for 2 seconds (0.01 x 100). Finally, CT12 holds the preset step value which was programmed into the drum instruction. When the drum's Reset input is active, it presets to step 1 in this case. The value of CT12 does not change without a program edit. Counter bit CT10 turns on when the drum cycle is complete, and turns off when the drum is reset.

Last Step Completion

The last step in a drum sequence may be any step number, since partial drums are valid. Refer to the following figure. When the transition conditions of the last step are satisfied, the drum sets the counter bit corresponding to the counter named in the drum instruction box (such as CT0). Then it moves to a final “drum complete” state. The drum outputs remain in the pattern defined for the last step (including any output mask logic). Having finished a drum cycle, the Start and Jog inputs have no effect at this point.

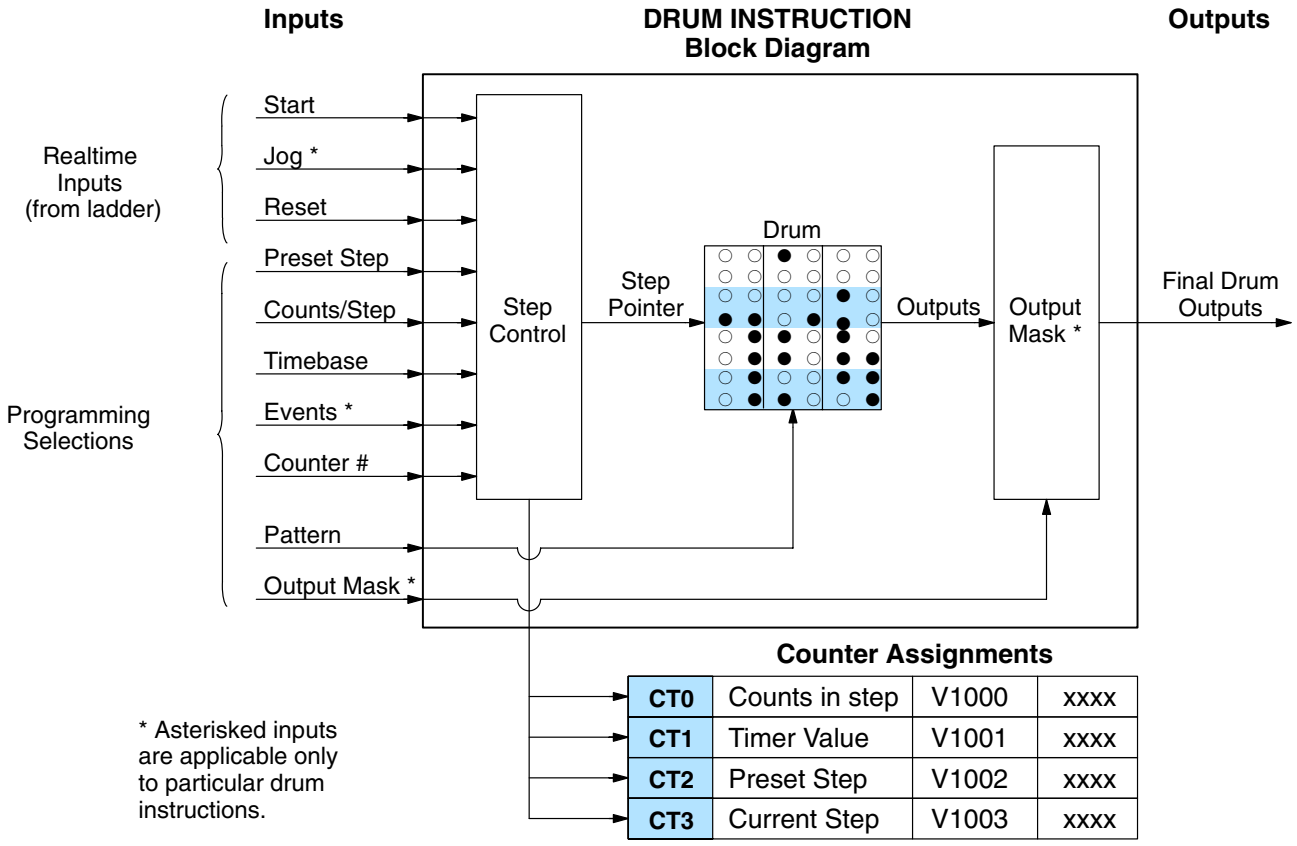
The drum leaves the “drum complete” state when the Reset input becomes active (or on a program-to–run mode transition). It resets the drum complete bit (such as CT0), and then goes directly to the appropriate step number defined as the preset step.



Overview of Drum Operation

Drum Instruction Block Diagram

The drum instruction utilizes various inputs and outputs in addition to the drum pattern itself. Refer to the figure below.



Counter Assignments

CT0	Counts in step	V1000	xxxx
CT1	Timer Value	V1001	xxxx
CT2	Preset Step	V1002	xxxx
CT3	Current Step	V1003	xxxx

The drum instruction accepts several inputs for step control, the main control of the drum. The inputs and their functions are:

- **Start** – The Start input is effective only when Reset is off. When Start is on, the drum timer runs if it is in a timed transition, and the drum looks for the input event during event transitions. When Start is off, the drum freezes in its current state (Reset must remain off), and the drum outputs maintain their current on/off pattern.
- **Jog** – The jog input is only effective when Reset is off (Start may be either on or off). The jog input increments the drum to the next step on each off-to-on transition. Note that only the basic timer drum does not have a jog input.
- **Reset** – The Reset input has priority over the Start input. When Reset is on, the drum moves to its preset step. When Reset is off, then the Start input operates normally.
- **Preset Step** – A step number from 1 to 16 that you define (typically is step 1). The drum moves to this step whenever Reset is on, and whenever the CPU first enters run mode.

- **Counts/Step** – The number of timer counts the drum spends in each step. Each step has its own counts parameter. However, programming the counts/step is optional on Timer/Event drums.
- **Timer Value** – the current value of the counts/step timer.
- **Counter #** – The counter number specifies the first of four consecutive counters which the drum uses for step control. You can monitor these to determine the drum's progress through its control cycle.
- **Events** – Either an X, Y, C, GX, GY, S, C, CT, or SP type discrete point serves as step transition inputs. Each step has its own event. However, programming the event is optional on Timer/Event drums.

WARNING: The outputs of a drum are enabled any time the CPU is in Run Mode. The Start Input **does not** have to be on, and the Reset input does not disable the outputs. Upon entering Run Mode, drum outputs automatically turn on or off according to the pattern of the preset step. This includes any effect of the output mask when applicable.

Powerup State of Drum Registers

The choice of the starting step on powerup and program-to-run mode transitions are important to consider for your application. Please refer to the following chart. If the counter memory is configured as non-retentive, the drum is initialized the same way on every powerup or program-to-run mode transition. However, if the counter memory is configured to be retentive, the drum will stay in its previous state.

Counter Number	Function	Initialization on Powerup	
		Non-Retentive Case	Retentive Case
CT(n)	Current Step Count	Initialize = 0	Use Previous (no change)
CT(n + 1)	Counter Timer Value	Initialize = 0	Use Previous (no change)
CT(n + 2)	Preset Step	Initialize = Preset Step #	Use Previous (no change)
CT(n + 3)	Current Step #	Initialize = Preset Step #	Use Previous (no change)

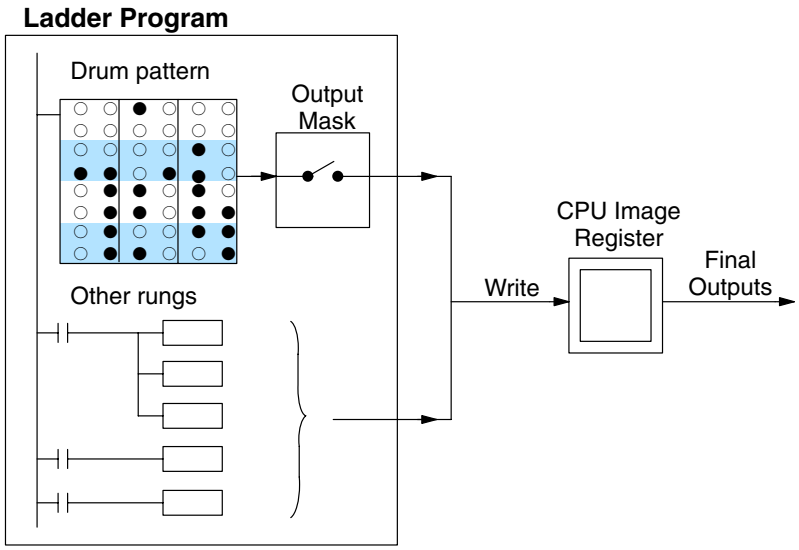
Applications with relatively fast drum cycle times typically will need to be reset on powerup, using the non-retentive option. Applications with relatively long drum cycle times may need to resume at the previous point where operations stopped, using the retentive case. The default option is the retentive case. This means that if you initialize scratchpad V-memory, the memory will be retentive.

Output Mask Operation

Sometimes we need more flexibility in controlling outputs than standard drum output patterns provide. The output mask feature lets you disable drum pattern control of selected outputs on selected steps, allowing those outputs to be controlled by other ladder logic. Two of the four drum instructions have the “output mask” feature:

- **MDRMD** – Masked Event Drum with Discrete Outputs
- **MDRMW** – Masked Event Drum with Word Output

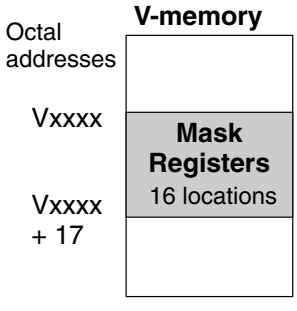
The output mask is simply a bit-by-bit enable/disable control for the drum writing to the image register of the sixteen outputs. Refer to the figure below. The image register contains the official current status of all I/O points. At the end of each PLC scan, the CPU uses the image register status to write to the actual output points.



Practical applications for drum output masking include:

- **Nested Sequence** – a particular output can perform a specialized sequence “inside” a particular step, while the other drum outputs remain static. Rather than consume additional steps, we just mask off the output and control it elsewhere in ladder logic during the step duration.
- **Manual Override** – occasionally we need to do manual control of some output(s) in a particular step. Masking the appropriate drum outputs will manual inputs to take over the control through ladder logic.

Each step has its own mask word! Each bit of the word masks the corrresponding output point. A 16-register table in V-memory will contain the mask values as shown to the right. In the drum instruction, you specify the starting location of the table. For example, a table which begins at V2000 will extend to V2017. Multiple MDRUMD or MDRUMW drums must have separate mask tables.

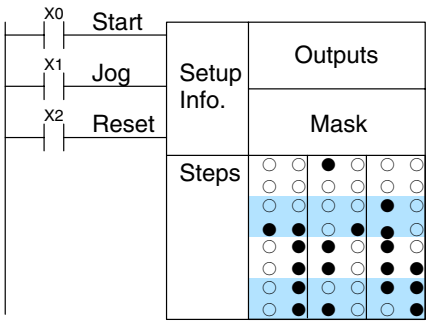


When a mask bit = 1, the drum controls the output point. when the mask bit =0, the drum cannot write to the image register, *so the output remains in its current state.*

Drum Control Techniques

Drum Control Inputs

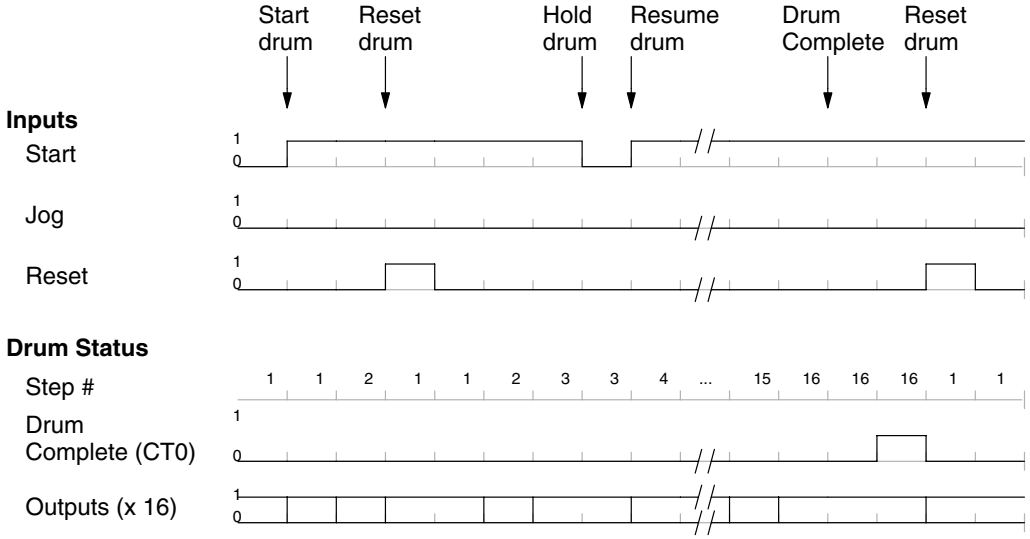
Now we are ready to put together the concepts on the previous pages and demonstrate general control of the drum instruction box. The drawing to the right shows a simplified generic drum instruction. Inputs from ladder logic control the Start, Jog, and Reset Inputs. The first counter bit of the drum (CT0, for example) indicates the drum cycle is done.



The timing diagram below shows an arbitrary timer drum input sequence and how the drum responds. As the CPU enters run mode it initializes the step number to the preset step number (typically is Step 1). When the Start input goes high the drum beginns running, looking for an event and/or running the count timer (depending on the drum type and setup).

After the drum enters Step 2, Reset turns On while Start is still On. Since Reset has priority over Start, the drum goes to the preset step (Step 1). Note that the drum is *held* in the preset step during Reset, and that step *does not run* (respond to events or run the timer) until Reset turns off.

After the drum has entered step 3, the Start input goes off momentarily, halting the drum's timer until Start turns on again.

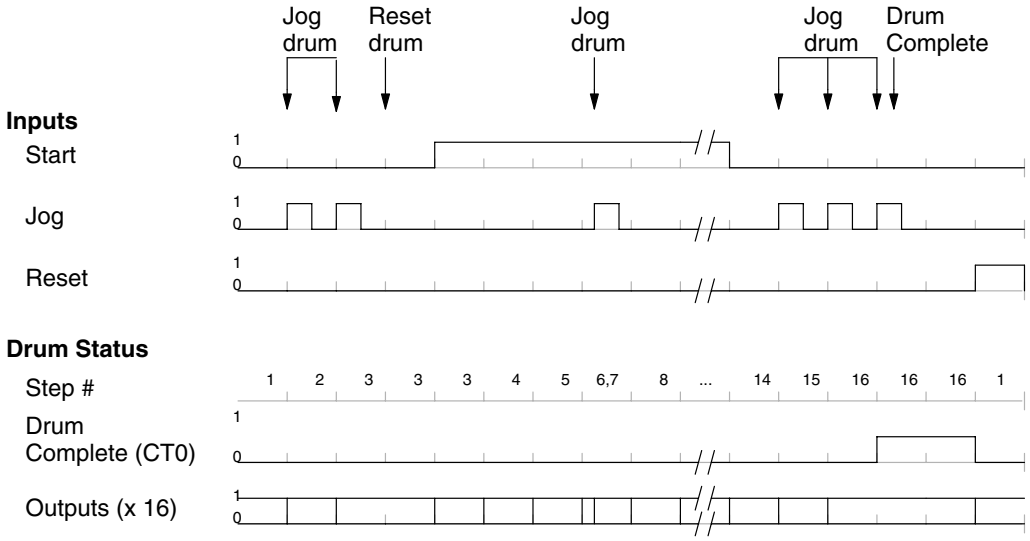


When the drum completes the last step (Step 16 in this example), the Drum Complete bit (CT0) turns on, and the step number remains at 16. When the Reset input turns on, it turns off the Drum Complete bit (CT0), and forces the drum to enter the preset step.

NOTE: The timing diagram shows all steps using equal time durations. Step times can vary greatly, depending on the counts/step programmed.

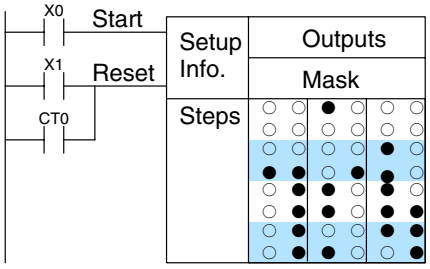
In the figure below, we focus on how the Jog input works on event drums. To the left of the diagram, note that the off-to-on transitions of the Jog input increments the step. Start may be either on or off (however, Reset must be off). Two jogs takes the drum to step three. Next, the Start input turns on, and the drum begins running normally. During step 6 another Jog input signal occurs. This increments the drum to step 7, setting the timer to 0. The drum begins running immediately in step 7, because Start is already on. The drum advances to step 8 normally.

As the drum enters step 14, the Start input turns off. Two more Jog signals moves the drum to step 16. However, note that a third Jog signal is required to move the drum through step 16 to “drum complete”. Finally, a Reset input signal arrives which forces the drum into the preset step and turns off the drum complete bit.



Self-Resetting Drum

Applications often require drums that automatically start over once they complete a cycle. This is easily accomplished, using the drum complete bit. In the figure to the right, the drum instruction setup is for CT0, so we logically OR the drum complete bit (CT0) with the Reset input. When the last step is done, the drum turns on CT0 which resets itself to the preset step, also resetting CT0. Contact X1 still works as a manual reset.



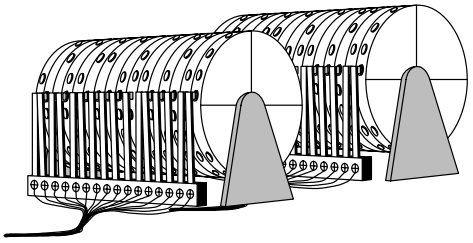
Initializing Drum Outputs

The outputs of a drum are enabled any time the CPU is in run mode. On program-to-run mode transitions, the drum goes to the preset step, and the outputs energize according to the pattern of that step. If your application requires all outputs to be off at powerup, there are two approaches:

- Make the preset step in the drum a “reset step”, with all outputs off.
- Or, use a drum with an output mask. Initialize the mask to “0000” on the first scan using contact SP0, and LD K000 and OUT Vxxx instructions, where Vxxxx is the location of the mask register.

Cascaded Drums Provide More Than 16 Steps

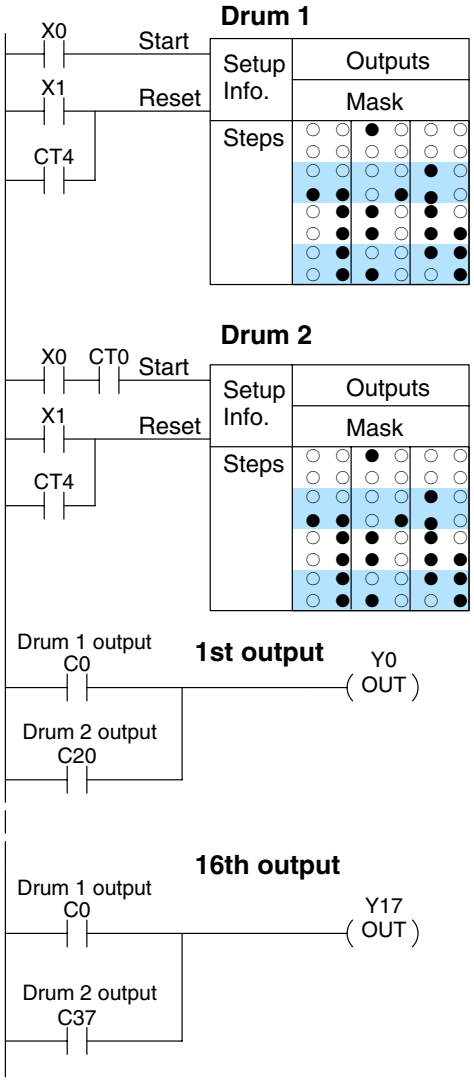
Occasionally the need arises for a drum with more than 16 steps. The solution is to use two or more drums that are logically cascaded. When the first drum finishes, the second one starts, and so on. Remember that a drum instruction writes to the outputs on every scan, even when its start input is off. So, two drums using the same output points will be in conflict. The way around this is to use separate control relays contacts (CRs) for each drum's outputs, and logically OR them together to control the final outputs.



Refer to the figure to the right. The two drums behave as one 32–step drum. The procedure is:

- Use the drum cycle done bit of the first drum for the start input of the next drum (CT0 in the example).
- Use the last drum's cycle done bit for the reset input of all drums (CT4 in the example).
- OR a manual reset contact with the reset contact above, if needed (is X1 in the example).
- Use the same V-memory address for the output mask of both drums, if your drum application requires a mask.
- Use different control relay (CR) output coils for each drum, but OR them together in ladder logic as shown.

Now, Y0 is the final output from the combined drums. Note that each drum must have an "idle" step in which its CR outputs are off, while the other drum(s) operate (will typically be step 1).



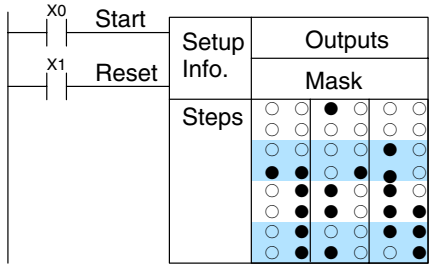
Handheld Programer Drum Mnemonics

The DL450 drum instructions may be programmed using either ***Direct***SOFT32 or a handheld programmer. This section covers entry via the handheld programmer (Refer to the ***Direct***SOFT32 manual for drum instruction entry using that tool).

NOTE: Drum editing requires Handheld Programmer firmware version 5.5 or later.

First, enter Store instructions for the ladder rungs controlling the drum's ladder inputs. In the example to the right, the timer drum's Start and Reset inputs are controlled by X0 and X1 respectively. The required keystrokes are listed beside the mnemonic.

These keystrokes *precede* the Drum instruction mnemonic. Note that the ladder rungs for Start and Reset inputs are *not* limited to being single–contact rungs.



Handheld Programmer Keystrokes

Store X0	STR	X(IN)	0	⏎
Store X1	STR	X(IN)	1	⏎

After the Store instructions comes the drum instruction itself. For example, let's program a basic timer drum (its mnemonic is DRUM). Also, we will use counters CT0, CT1, CT2, and CT3. The required keystrokes are:

Handheld Programmer Keystrokes

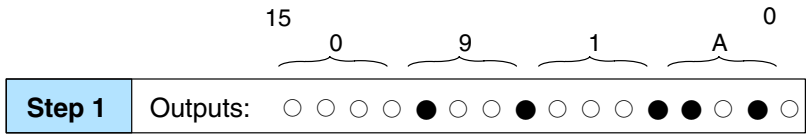
DRUM CNT0	SHFT	D	R	U	M	SHFT	CNT	0	⏎

After entering the DRUM mnemonic as above, the handheld programmer creates an input form that specific matches the drum type you entered. The input form consists of approximately fifty or more default mnemonic entries containing DEF (define) statements. The default mnemonics are already "input" for you, so they appear automatically. Use the NXT and PREV keys to move forward and backward through the form. Only the editing of default values is required, thus eliminating many keystrokes. The entries required for the basic timer drum are in the chart below.

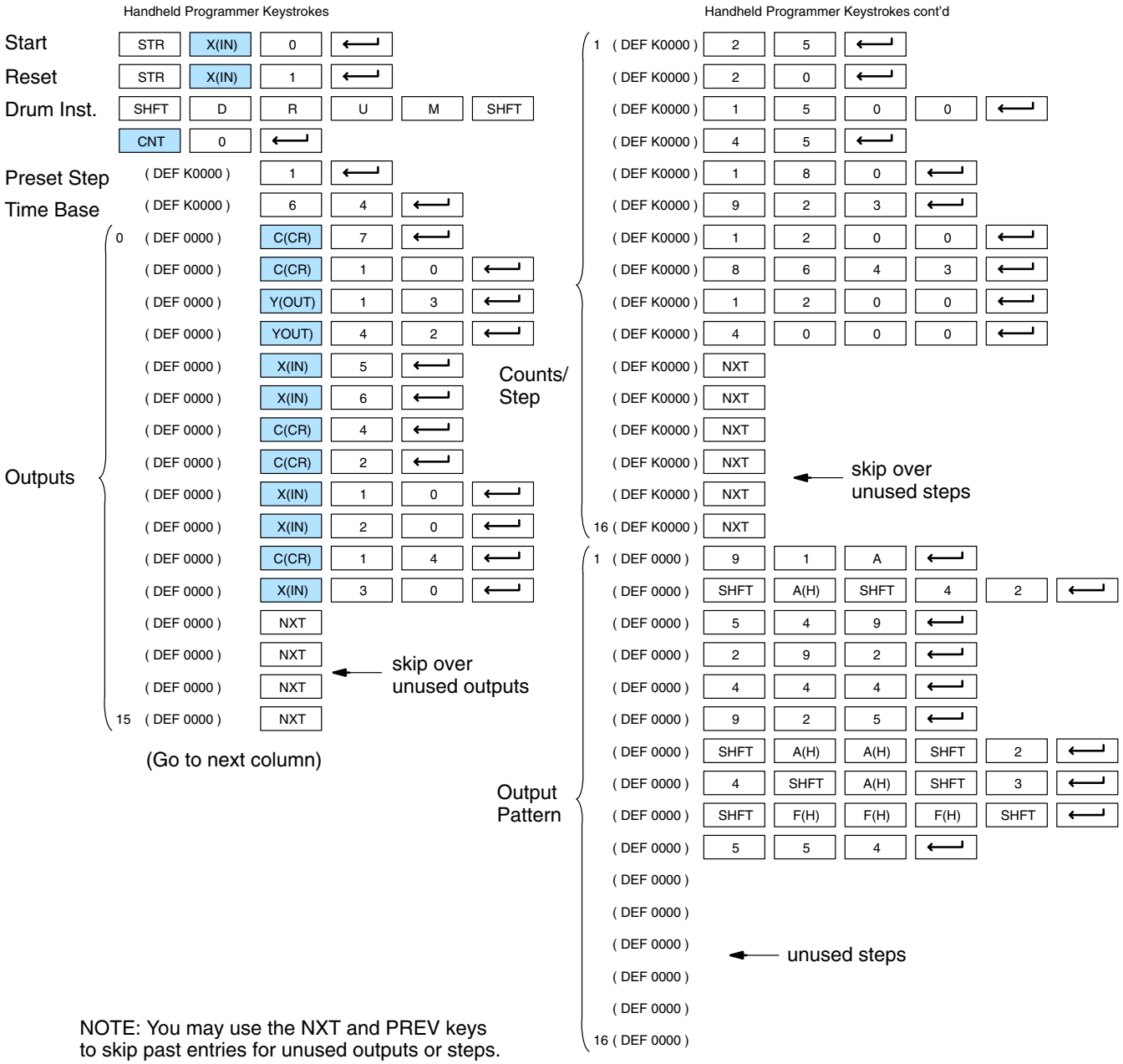
Drum Parameters	Multiple Entries	Mnemonic / Entry	Default Mnemonic	Valid Data Types	Ranges
Start Input	–	STR (plus input rung)	–	–	–
Reset Input	–	STR (plus input rung)		–	–
Drum Mnemonic	–	DRUM CNT aaaa	–	K	0 – 777
Preset Step	1	bb	DEF K0000	K	1 – 16
Timer base	1	cc	DEF K0000	K	0 – 9999
Output points	16	Fffff	DEF 0000	X, Y, C *	see page 3–42
Counts per step	16	dddd	DEF K0000	K	se page 3–42
Output pattern	16	gggg	DEF K0000	K	see page 3–42

NOTE: Default entries for output points are "DEF 0000", which means they are unassigned. If you need to go back and change an assigned output as unused again, enter "K0000". The entry will again show as "DEF 0000".

Using the DRUM entry chart on the previous page, we show the method of entry for the basic timer drum instruction. First, we convert all the step patterns to the equivalent hex number, as shown in the following example.



The next figure shows the required handheld programmer keystrokes to enter an example DRUM instruction. The default entries of the form are in parenthesis. After the drum instruction entry, the remaining keystrokes shown over-write the numeric portion of each default DEF statement.

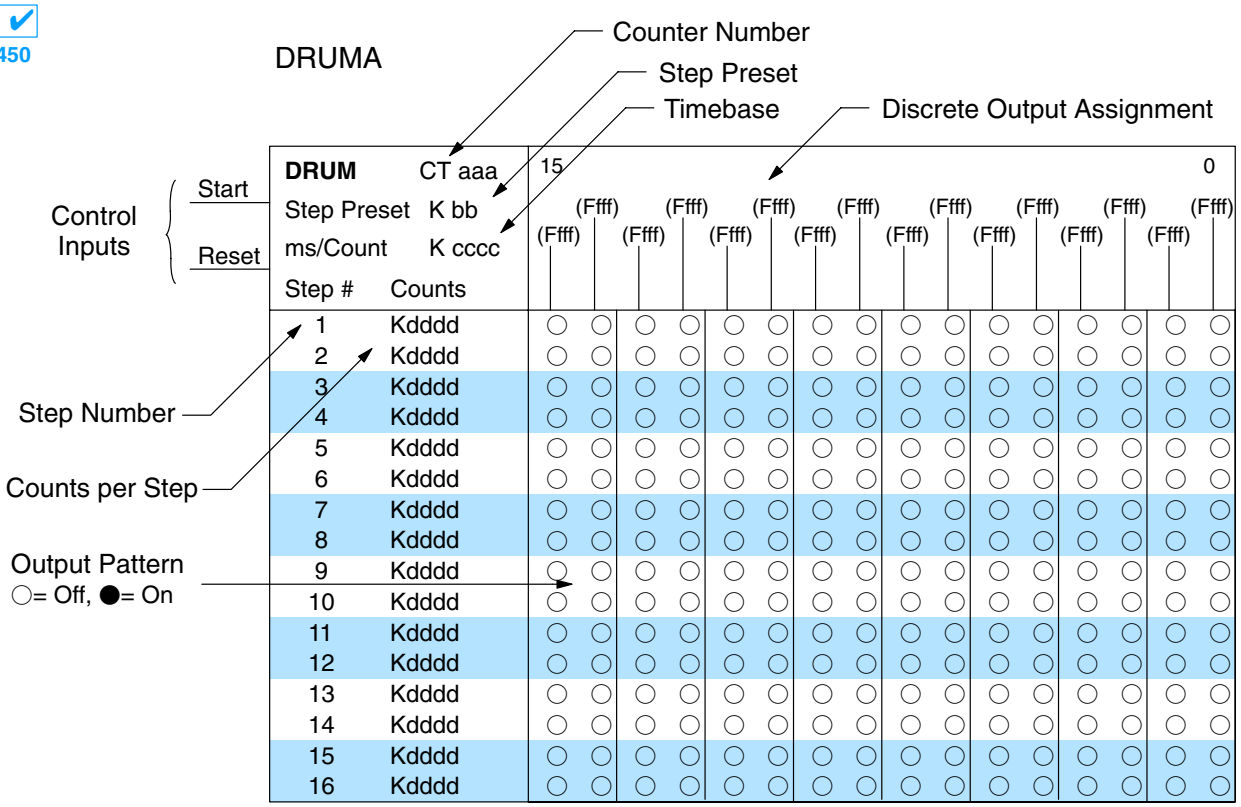


Drum Instructions

Timed Drum with Discrete Outputs (DRUM)

X	X	✔
430	440	450

The Timed Drum with Discrete Outputs is the most basic of the DL450's drum instructions. It operates according to the principles covered on the previous pages. Below is the instruction in chart form as displayed by ***Direct***SOFT32.



The Timed Drum features 16 steps and 16 outputs. Step transitions occur only on a timed basis, specified in counts per step. Unused steps must be programmed with "counts per step" = 0 (this is the default entry). The discrete output points may be individually assigned as X, Y, or C types, or may be left unused. The output pattern may be edited graphically with ***Direct***SOFT32.

Whenever the Start input is energized, the drum's timer is enabled. It stops when the last step is complete, or when the Reset input is energized. The drum enters the preset step chosen upon a CPU program-to-run mode transition, and whenever the Reset input is energized.

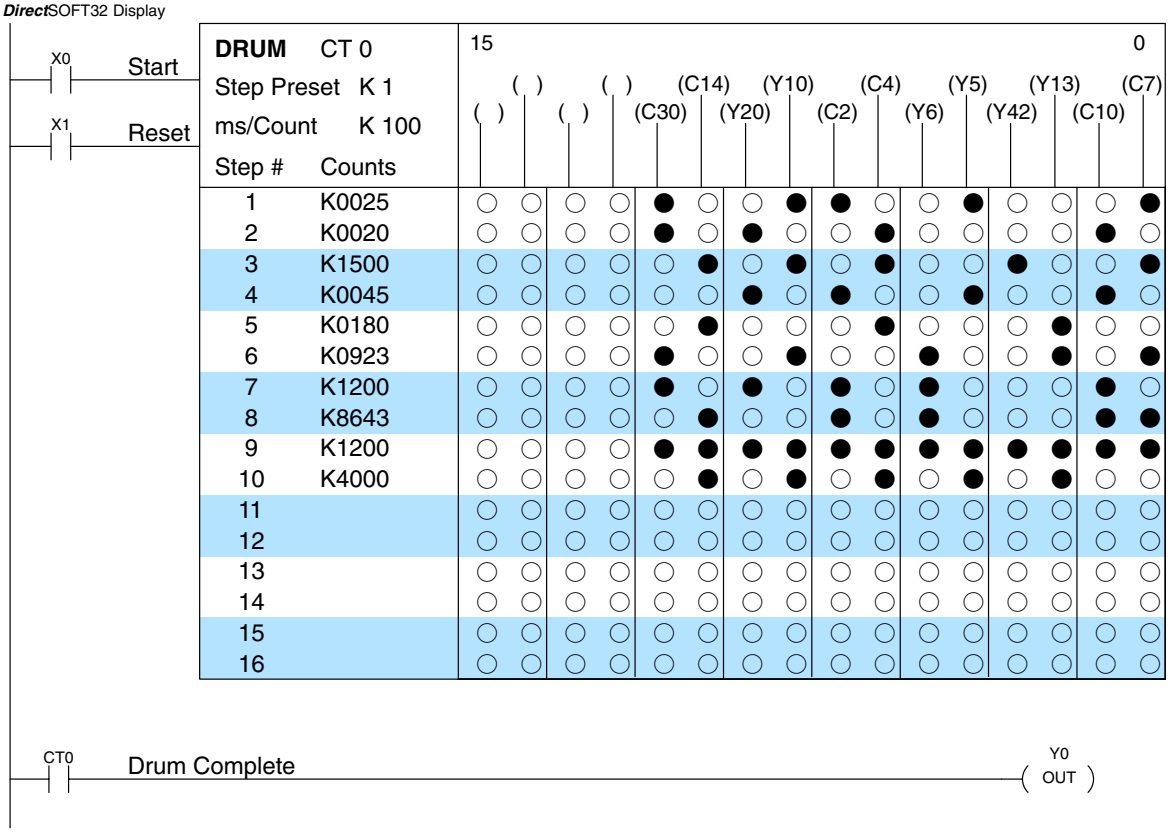
Drum Parameters	Field	Data Types	Ranges
Counter Number	aaa	–	0 – 377
Preset Step	bb	K	1 – 16
Timer base	cc	K	0 – 99.99 seconds
Counts per step	dddd	K	0 – 9999
Discrete Outputs	Fffff	X, Y, C, GX, GY *	see page 3–42

Drum instructions use four counters in the CPU. The ladder program can read the counter values for the drum's status. The ladder program may write a new preset step number to CT(n+2) at any time. However, the other counters are for monitoring purposes only.

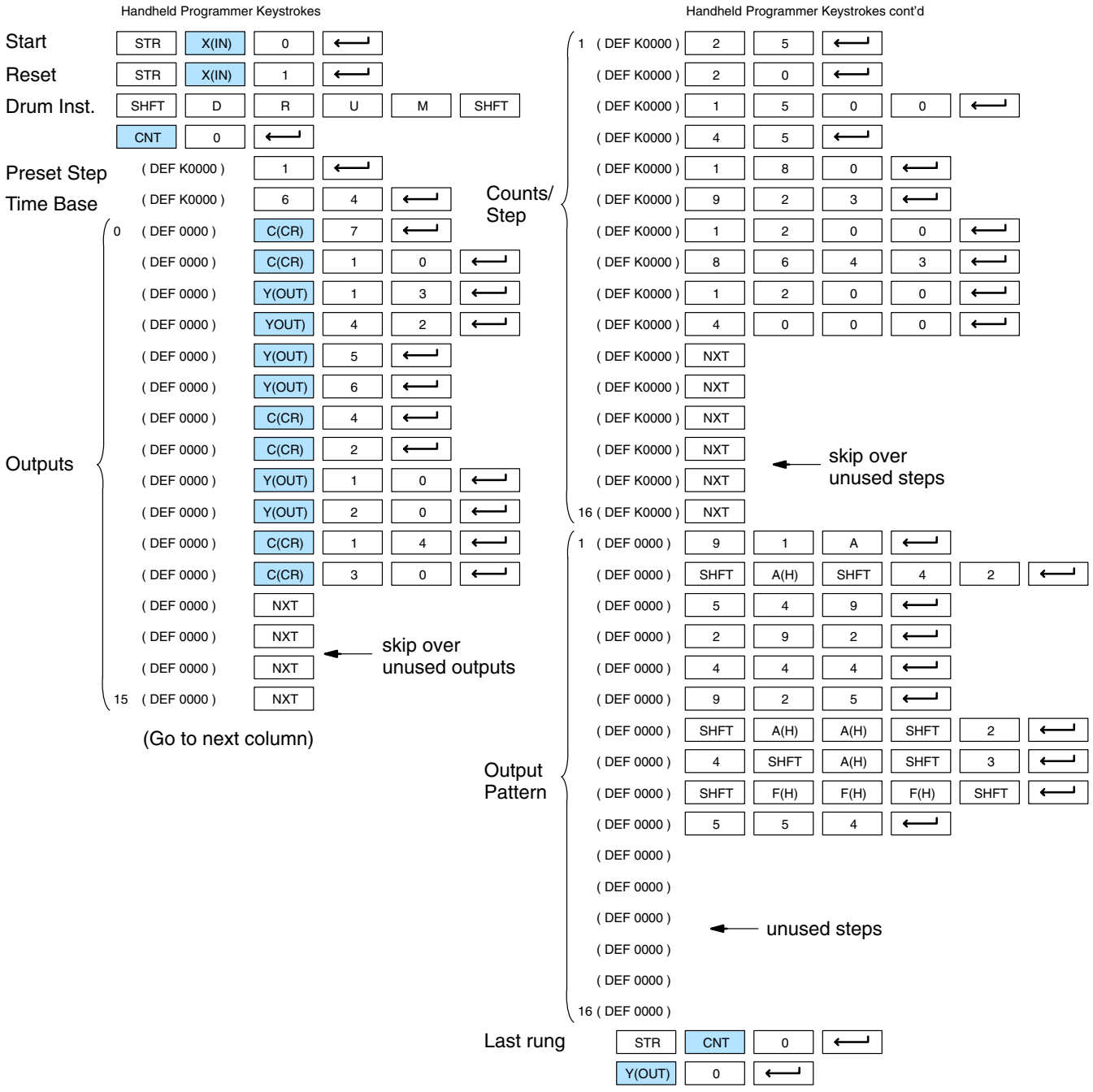
Counter Number	Ranges of (n)	Function	Counter Bit Function
CT(n)	0 – 374	Counts in step	CTn = Drum Complete
CT(n+1)	1 – 375	Timer value	CT(n+1) = (not used)
CT(n+2)	2 –376	Preset Step	CT(n+2) = (not used)
CT(n+3)	3 –377	Current Step	CT(n+1) = (not used)

The following ladder program shows the DRUM instruction in a typical ladder program, as shown by ***Direct***SOFT32. Steps 1 through 10 are used, and twelve of the sixteen output points are used. The preset step is step 1. The timebase runs at 100 mS per count. Therefore, the duration of step 1 is (25 x 0.1) = 2.5 seconds. In the last rung, the Drum Complete bit (CT0) turns on output Y0 upon completion of the last step (step 10). A drum reset also resets CT0.

DirectSOFT32 Display



The handheld programmer can also enter or edit drum instructions. The diagram below lists the keystrokes for entering the drum example on the previous page. **NOTE**: Drum editing requires Handheld Programmer firmware version 5.5 or later.

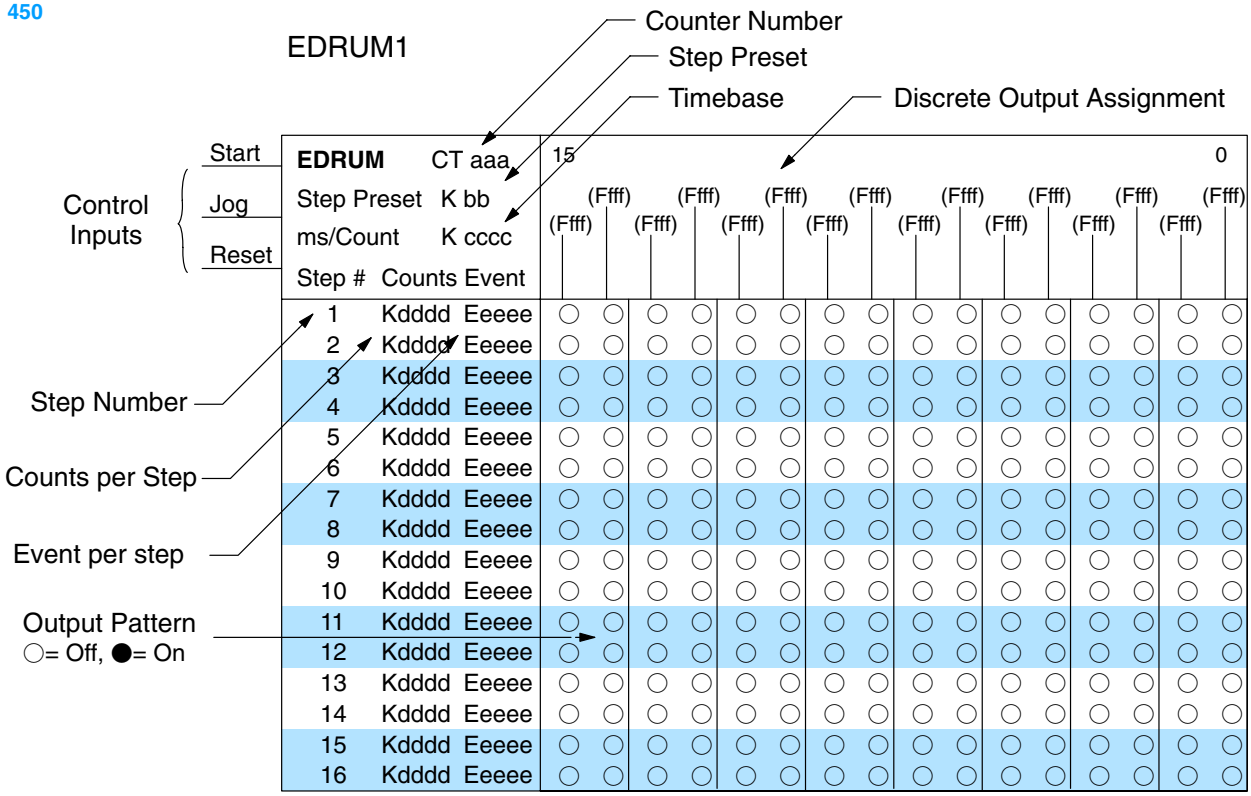


NOTE: You may use the NXT and PREV keys to skip past entries for unused outputs or steps.

Event Drum with Discrete Outputs (EDRUM)

430	440	450
✗	✗	✔

The Event Drum with Discrete Outputs has all the features of the Timed Drum, plus event-based step transitions. It operates according to the general principles of drum operation covered in the beginning of this section. Below is the instruction in chart form as displayed by ***Direct***SOFT32.



The Event Drum with Discrete Outputs features 16 steps and 16 outputs. Step transitions occur on timed and/or event basis. The jog input also advances the step on each off-to-on transition. Time is specified in counts per step, and events are specified as discrete contacts. Unused steps must be programmed with “counts per step” = 0, and event = “0000”. The discrete output points may be individually assigned. The output pattern may be edited graphically with ***Direct***SOFT32.

Whenever the Start input is energized, the drum’s timer is enabled. As long as the event is true for the current step, the timer runs during that step. When the step count equals the counts per step, the drum transitions to the next step. This process stops when the last step is complete, or when the Reset input is energized. The drum enters the preset step chosen upon a CPU program-to-run mode transition, and whenever the Reset input is energized.

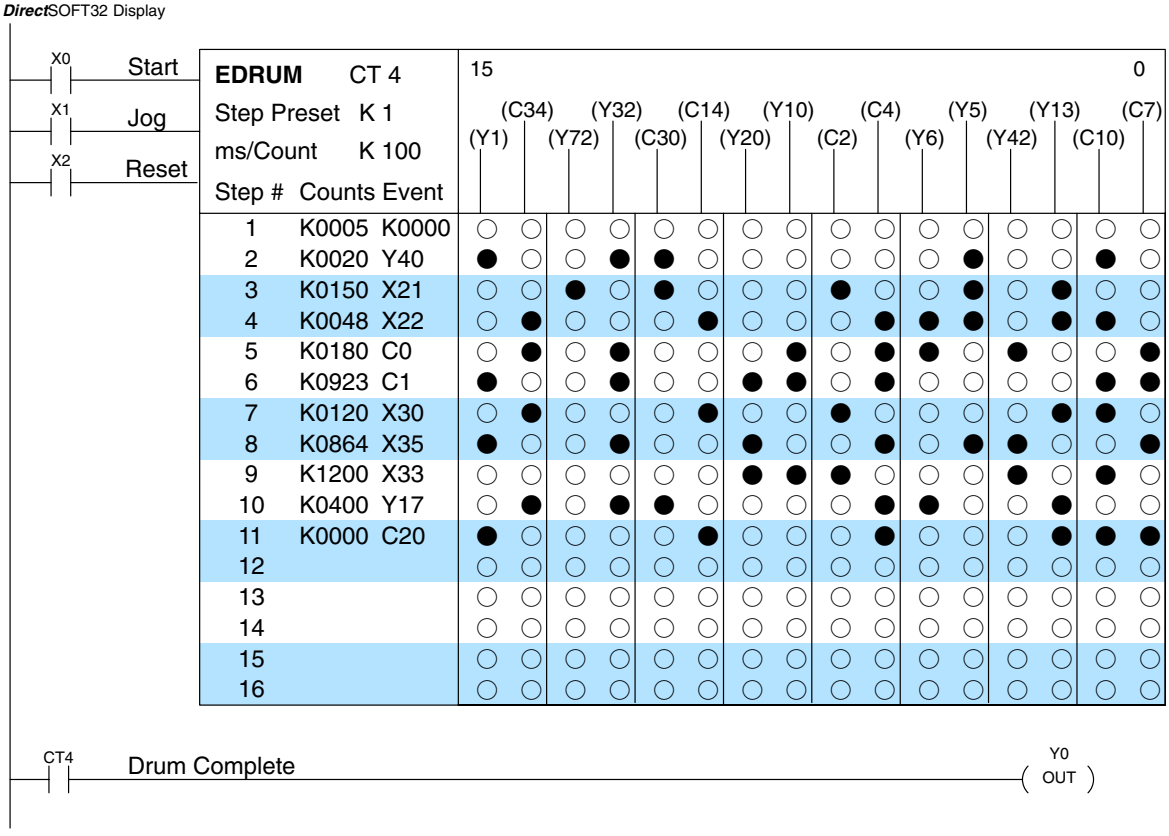
Drum Parameters	Field	Data Types	Ranges
Counter Number	aaa	–	0 – 377
Preset Step	bb	K	1 – 16
Timer base	cc	K	0 – 99.99 seconds
Counts per step	dddd	K	0 – 9999
Event	eeee	X, Y, C, GX, GY, S, T, ST	see page 3–42
Discrete Outputs	Fffff	X, Y, C , GX, GY*	see page 3–42

Drum instructions use four counters in the CPU. The ladder program can read the counter values for the drum's status. The ladder program may write a new preset step number to CT(n+2) at any time. However, the other counters are for monitoring purposes only.

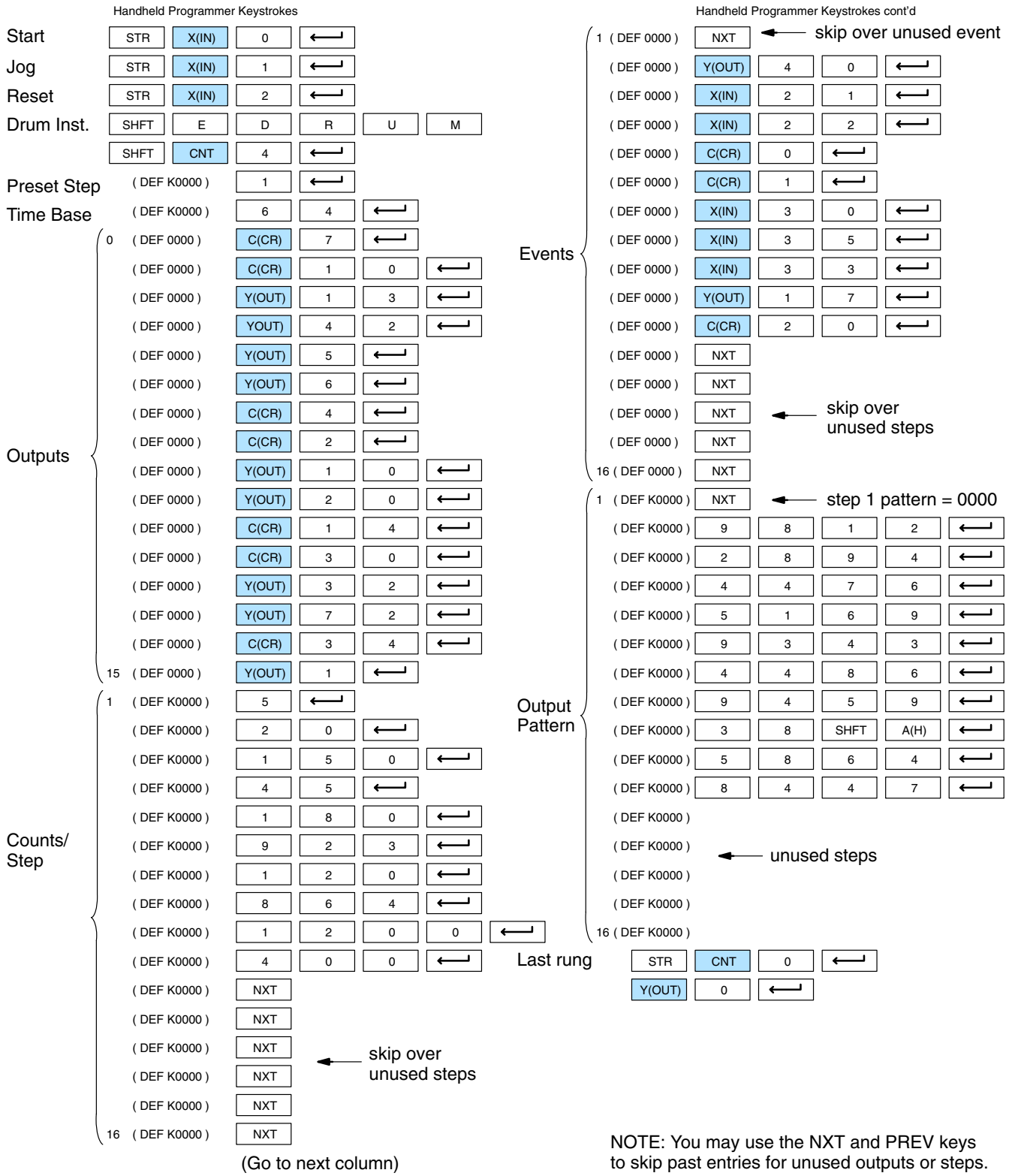
Counter Number	Ranges of (n)	Function	Counter Bit Function
CT(n)	0 – 374	Counts in step	CTn = Drum Complete
CT(n+1)	1 – 375	Timer value	CT(n+1) = (not used)
CT(n+2)	2 –376	Preset Step	CT(n+2) = (not used)
CT(n+3)	3 –377	Current Step	CT(n+1) = (not used)

The following ladder program shows the EDRUM instruction in a typical ladder program, as shown by ***Direct***SOFT32. Steps 1 through 11 are used, and all sixteen output points are used. The preset step is step 1. The timebase runs at 100 mS per count. Therefore, the duration of step 1 is (5 x 0.1) = 0.5 seconds. Note that step 1 is time-based only (event = "K0000"). And, the output pattern for step 1 programs all outputs off, which is a typically desirable powerup condition. In the last rung, the Drum Complete bit (CT4) turns on output Y0 upon completion of the last step (step 10). A drum reset also resets CT4.

DirectSOFT32 Display



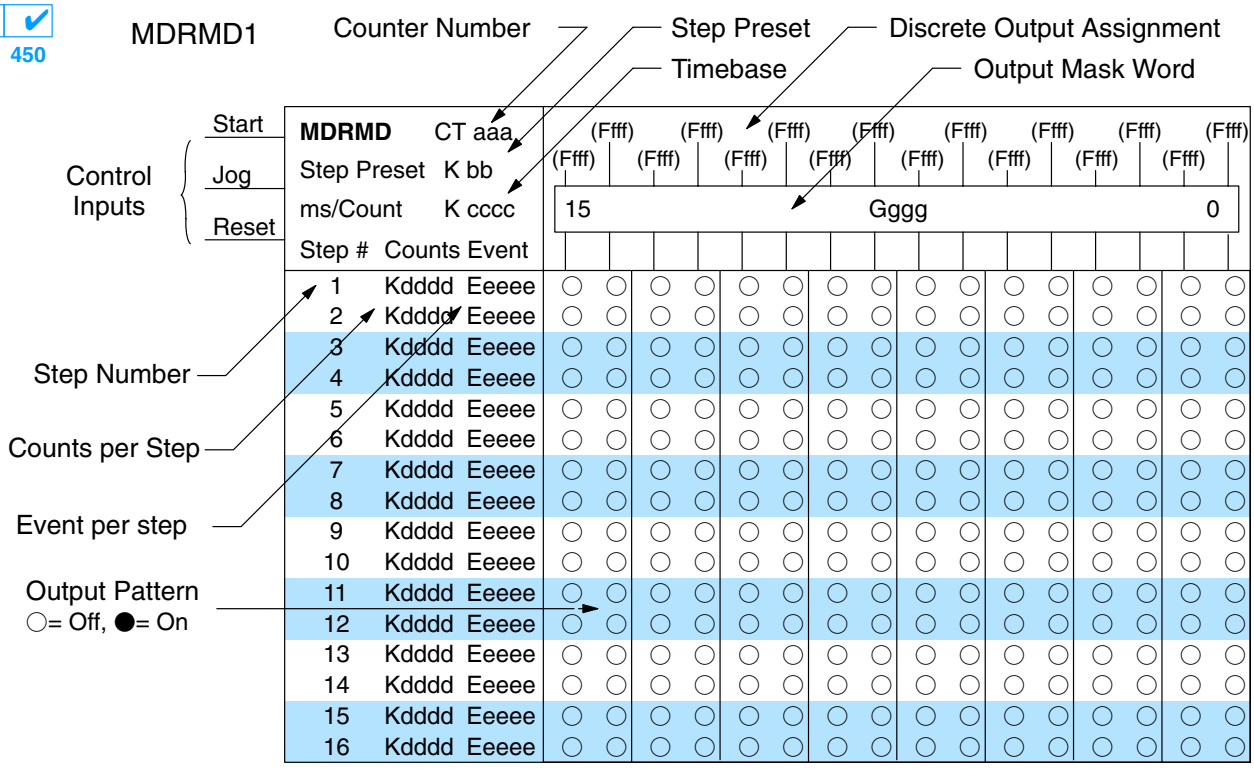
The handheld programmer can also enter or edit drum instructions. The diagram below lists the keystrokes for entering the drum example on the previous page. **NOTE**: Drum editing requires Handheld Programmer firmware version 5.5 or later.



Masked Event Drum with Discrete Outputs (MDRMD)

✗	✗	✔
430	440	450

The Masked Event Drum with Discrete Outputs has all the features of the basic Event Drum plus final output control for each step. It operates according to the general principles of drum operation covered in the beginning of this section. Below is the instruction in chart form as displayed by ***Direct***SOFT32.



The Masked Event Drum with Discrete Outputs features sixteen steps and sixteen outputs. Drum outputs are logically ANDed bit-by-bit with an output mask word for each step. The Gggg field specifies the beginning location of the 16 mask words. Step transitions occur on timed and/or event basis. The jog input also advances the step on each off-to-on transition. Time is specified in counts per step, and events are specified as discrete contacts. Unused steps must be programmed with “counts per step” = 0, and event = “0000”.

Whenever the Start input is energized, the drum’s timer is enabled. As long as the event is true for the current step, the timer runs during that step. When the step count equals the counts per step, the drum transitions to the next step. This process stops when the last step is complete, or when the Reset input is energized. The drum enters the preset step chosen upon a CPU program-to-run mode transition, and whenever the Reset input is energized.

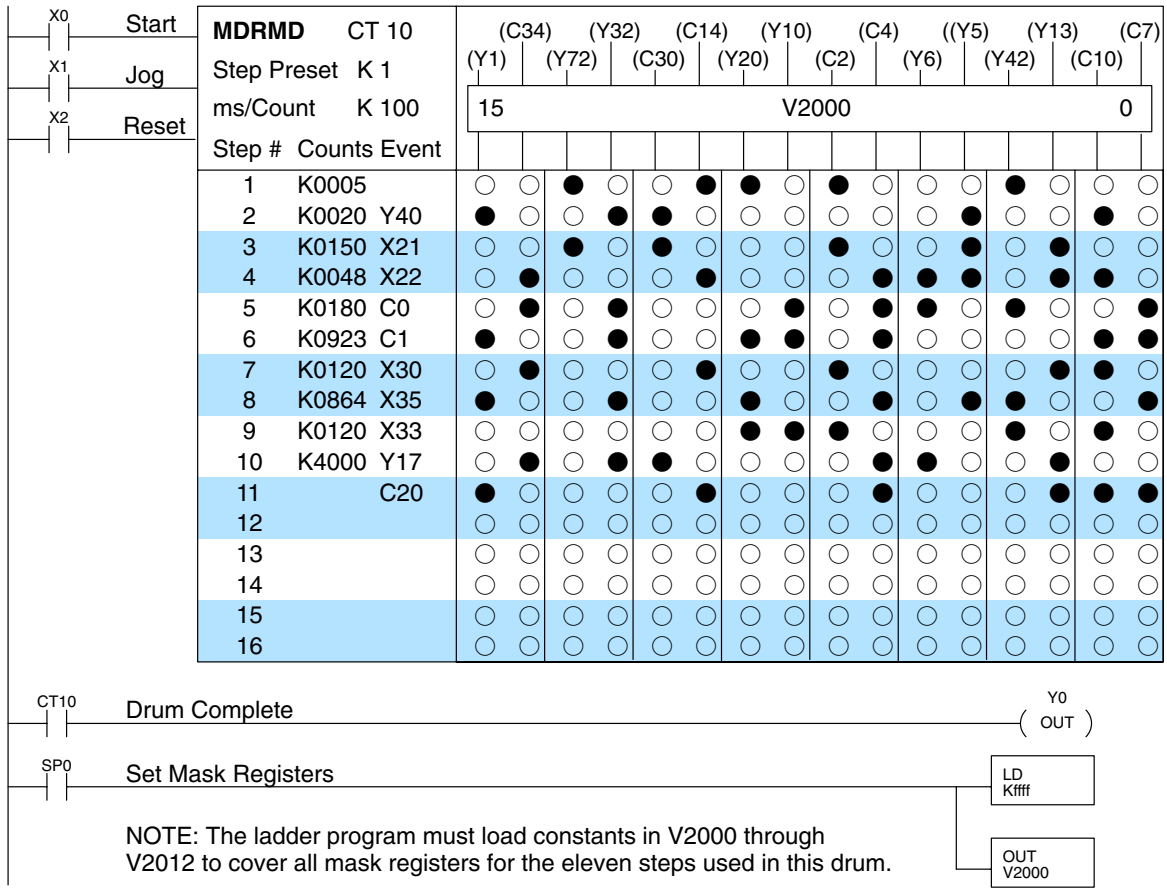
Drum Parameters	Field	Data Types	Ranges
Counter Number	aaa	–	0 – 177
Preset Step	bb	K	1 – 16
Timer base	cc	K	0 – 99.99 seconds
Counts per step	dddd	K	0 – 9999
Event	eeee	X, Y, C, GX, GY, S, T, ST	see page 3–42
Discrete Outputs	Fffff	X, Y, C, GX, GY *	see page 3–42
Output Mask	Gggg	V	see page 3–42

Drum instructions use four counters in the CPU. The ladder program can read the counter values for the drum's status. The ladder program may write a new preset step number to CT(n+2) at any time. However, the other counters are for monitoring purposes only.

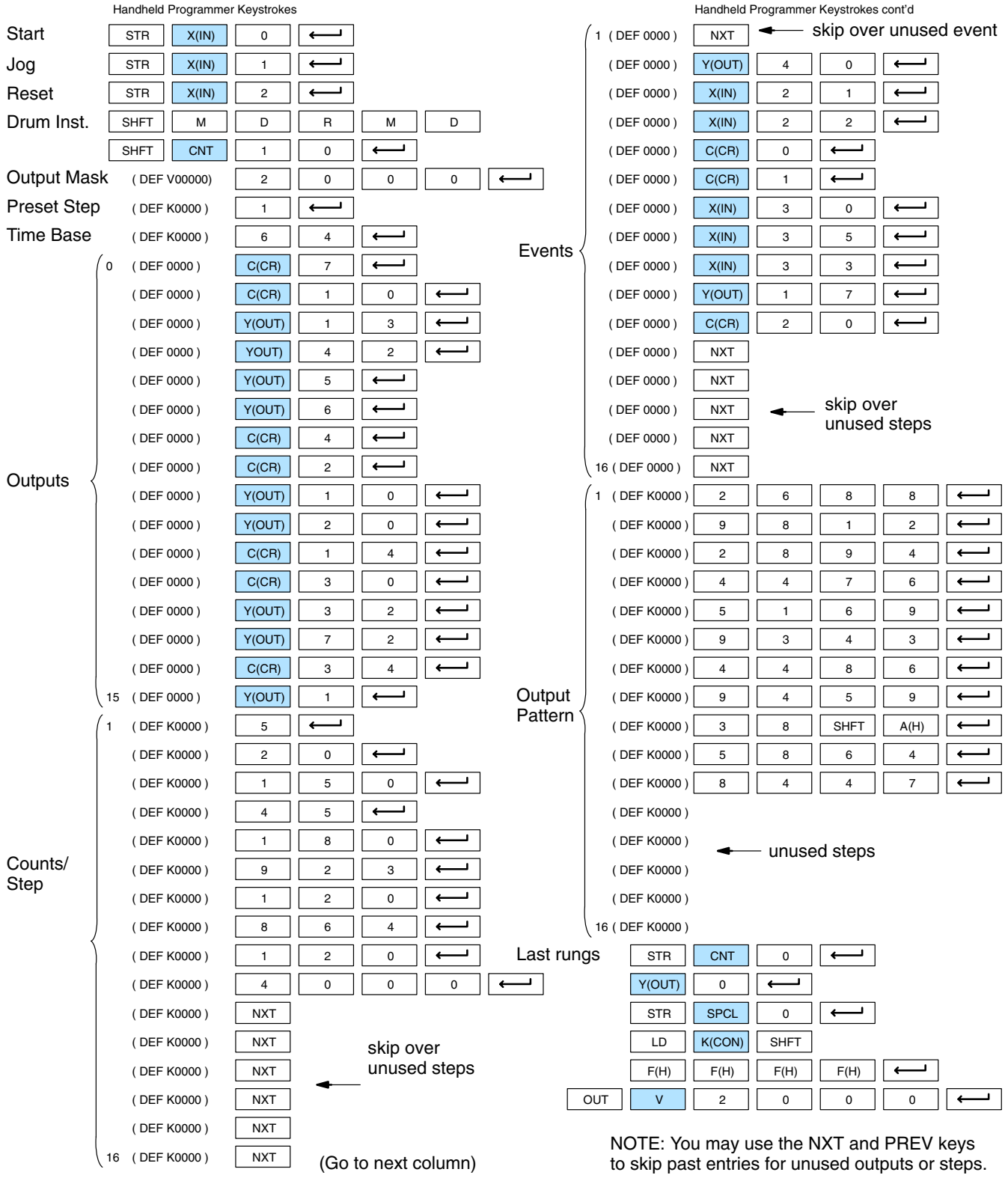
Counter Number	Ranges of (n)	Function	Counter Bit Function
CT(n)	0 – 374	Counts in step	CTn = Drum Complete
CT(n+1)	1 – 375	Timer value	CT(n+1) = (not used)
CT(n+2)	2 –376	Preset Step	CT(n+2) = (not used)
CT(n+3)	3 –377	Current Step	CT(n+1) = (not used)

The following ladder program shows the MDRMD instruction in a typical ladder program, as shown by ***Direct***SOFT32. Steps 1 through 11 are used, and all 16 output points are used. The output mask word is at V2000. The final drum outputs are shown above the mask word as individual bits. The data bits in V2000 are logically ANDed with the output pattern of the current step in the drum. If you want all drum outputs to be off after powerup, just write zeros to V2000 on the first scan. Ladder logic may update the output mask at any time to enable or disable the drum outputs. The preset step is step 1. The timebase runs at 100 mS per count. Therefore, the duration of step 1 is (5 x 0.1) = 0.5 seconds. Note that step 1 is time-based only (event – "K0000"). In the last rung, the Drum Complete bit (CT10) turns on output Y0 upon completion of the last step (step 10). A drum reset also resets CT10.

DirectSOFT32 Display



The handheld programmer can also enter or edit drum instructions. The diagram below lists the keystrokes for entering the drum example on the previous page. **NOTE**: Drum editing requires Handheld Programmer firmware version 5.5 or later.

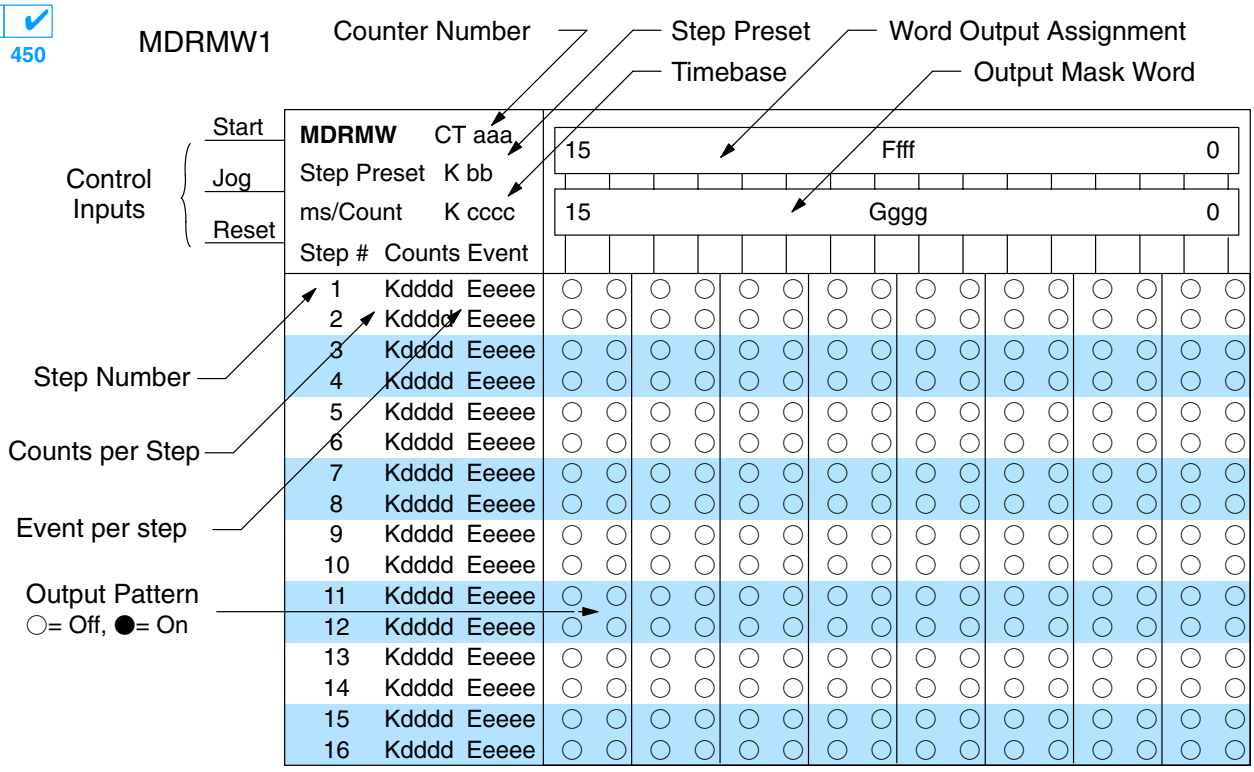


NOTE: You may use the NXT and PREV keys to skip past entries for unused outputs or steps.

Masked Event Drum with Word Output (MDRMW)

430	440	450
✗	✗	✔

The Masked Event Drum with Word Output features outputs organized as bits of a single word, rather than discrete points. It operates according to the general principles of drum operation covered in the beginning of this section. Below is the instruction in chart form as displayed by ***Direct***SOFT32.



The Masked Event Drum with Word Output features sixteen steps and sixteen outputs. Drum outputs are logically ANDed bit-by-bit with an output mask word for each step. The Gggg field specifies the beginning location of the 16 mask words, creating the final output (Ffff field). Step transitions occur on timed and/or event basis. The jog input also advances the step on each off-to-on transition. Time is specified in counts per step, and events are specified as discrete contacts. Unused steps must be programmed with "counts per step" = 0, and event = "0000".

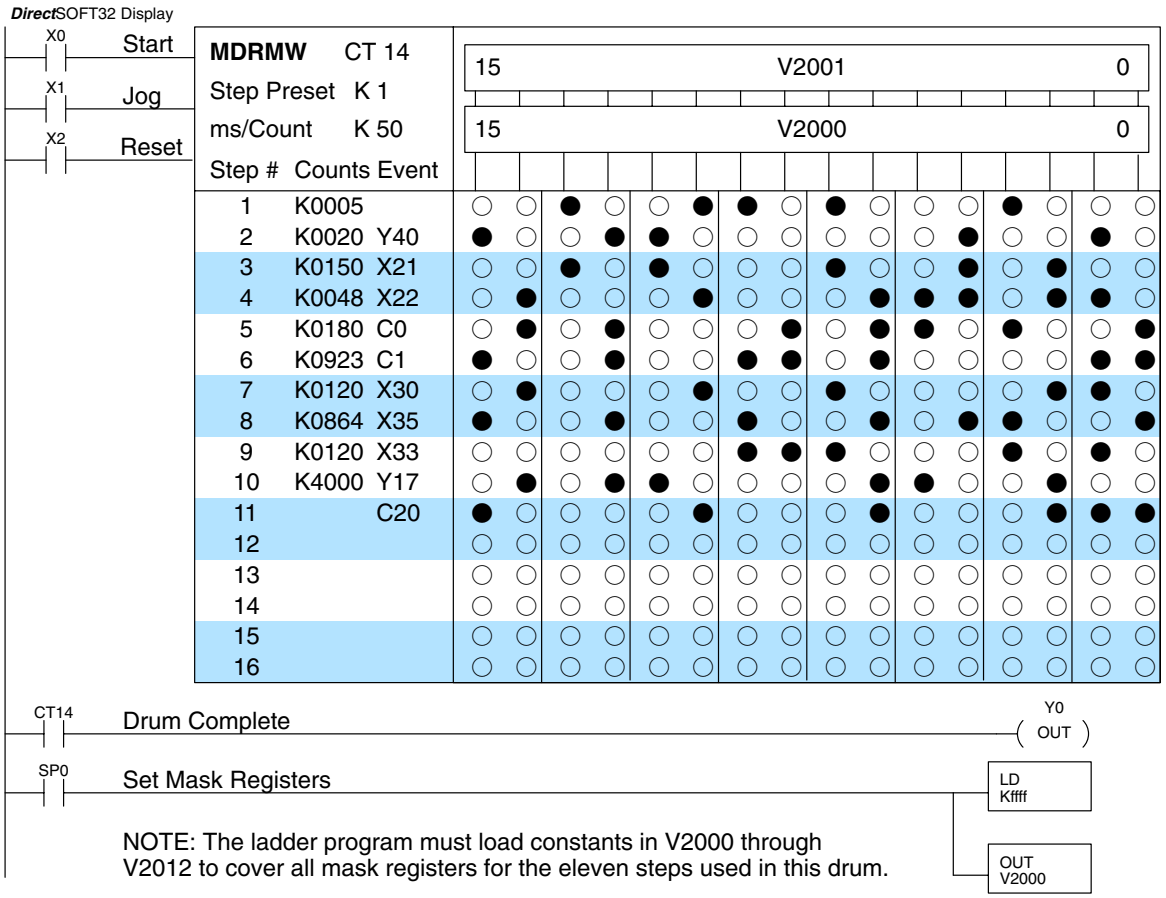
Whenever the Start input is energized, the drum's timer is enabled. As long as the event is true for the current step, the timer runs during that step. When the step count equals the counts per step, the drum transitions to the next step. This process stops when the last step is complete, or when the Reset input is energized. The drum enters the preset step chosen upon a CPU program-to-run mode transition, and whenever the Reset input is energized.

Drum Parameters	Field	Data Types	Ranges
Counter Number	aaa	–	0 – 177
Preset Step	bb	K	1 – 16
Timer base	cc	K	0 – 99.99 seconds
Counts per step	dddd	K	0 – 9999
Event	eeee	X, Y, C, GX, GY, S, T, ST	see page 3–42
Word Output	Fffff	V	see page 3–42
Output Mask	Gggg	V	see page 3–42

Drum instructions use four counters in the CPU. The ladder program can read the counter values for the drum's status. The ladder program may write a new preset step number to CT(n+2) at any time. However, the other counters are for monitoring purposes only.

Counter Number	Ranges of (n)	Function	Counter Bit Function
CT(n)	0 – 374	Counts in step	CTn = Drum Complete
CT(n+1)	1 – 375	Timer value	CT(n+1) = (not used)
CT(n+2)	2 –376	Preset Step	CT(n+2) = (not used)
CT(n+3)	3 –377	Current Step	CT(n+1) = (not used)

The following ladder program shows the MDRMD instruction in a typical ladder program, as shown by ***Direct***SOFT32. Steps 1 through 11 are used, and all sixteen output points are used. The output mask word is at V2000. The final drum outputs are shown above the mask word as a word at V2001. The data bits in V2000 are logically ANDed with the output pattern of the current step in the drum, generating the contents of V2001. If you want all drum outputs to be off after powerup, just write zeros to V2000 on the first scan. Ladder logic may update the output mask at any time to enable or disable the drum outputs. The preset step is step 1. The timebase runs at 50 mS per count. Therefore, the duration of step 1 is (5 x 0.1) = 0.5 seconds. Note that step 1 is time-based only (event – "K0000"). In the last rung, the Drum Complete bit (CT14) turns on output Y0 upon completion of the last step (step 10). A drum reset also resets CT14.



The handheld programmer can also enter or edit drum instructions. The diagram below lists the keystrokes for entering the drum example on the previous page. **NOTE**: Drum editing requires Handheld Programmer firmware version 5.5 or later.