Pulse Control LSI

PCD/PCL/G series

Guidebook

Impress, not just satisfy

Nippon Pulse Motor Co., Ltd.

INDEX

1. Introduction

This document describes the features and the basic functions of Pulse Control LSI (=Motion Control LSI), PCD/PCL/G series, provided by Nippon Pulse Motor Co., Ltd. (NPM).

In recent years, developing and designing engineers in equipment manufacturers have been decreased, so we hope this manual would help you to understand how quickly and easily you can build a desired motion control in a short period of time.

(Hereinafter, NPM-made pulse control LSIs are collectively referred to as "PCL".)

Please note that this document is made in order to understand the basics of pulse control LSIs, and the detailed explanations such as restrictions and operations that may differ depending on the settings are excluded.

Refer to the user's manual if you want the more detail information on how to set the functions.

2. Outlines

2-1. Acceleration/Deceleration control

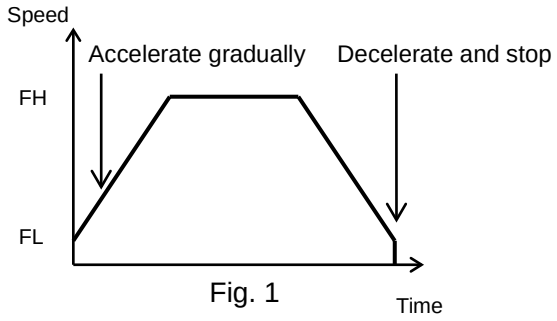


Fig. 1

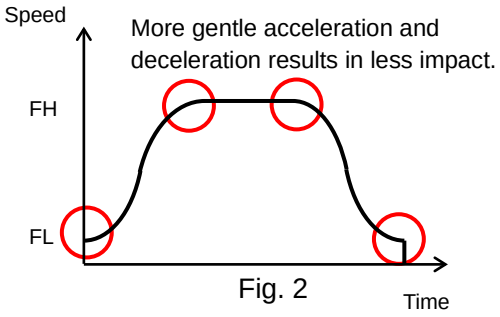


Fig. 2

First, we explain about “acceleration/deceleration control” which is the MUST to understand PCLs. If you have already known this, you can skip this section.

Stepping motors are so called “Give up easily type of motors”. If you try to move a heavy load quickly, they would easily give up before trying to say “I cannot move such a heavy load at such a high speed. (This is called “out-of-step” phenomenon.)

When compared to an automobile, an automobile with a manual shift (which has recently been few) can stall if you try to start driving at the 5th speed all of a sudden.

In order to increase the speed, you need to accelerate gradually in steps from 1st, 2nd, 3rd...

A stepping motor works in the same way. If you want to move a load somewhat at faster speed, you need to accelerate from low to high speed.

On the other hand, a stepping motor cannot stop suddenly when running at high speed. The moment of inertia overshoot the motor from the position that you want to stop. To stop at the exact position, you must also decelerate to an instantaneous stop speed [=FL speed(starting speed, speed to stop)].

Acceleration/deceleration is required to perform positioning as quickly and accurately as possible.

Additionally, if acceleration/deceleration characteristics can be S-curve rather than straight.... it can be gentler and give less impact to the moving load.

Please imagine to move a table with a glass of water on left and right horizontally.

The operation pattern shown in Fig. 1 is with linear acceleration/deceleration.

When starts moving, ends acceleration, starts deceleration and stops moving, the “angle” can cause the water to shake significantly due to the impact.

To prevent water from shaking as much as possible, you might want to use S-curve acceleration/deceleration as shown in Fig. 2.

Stepping motors are used to transport a variety of things. Some can be handled somewhat roughly, but others, for example semi-conductor wafers, should be handled “slowly and softly. Failures to handle them slowly and softly can give them an impact and may ruin the costly wafers.

S-curve acceleration/deceleration is the ability to move things more gently than ever.

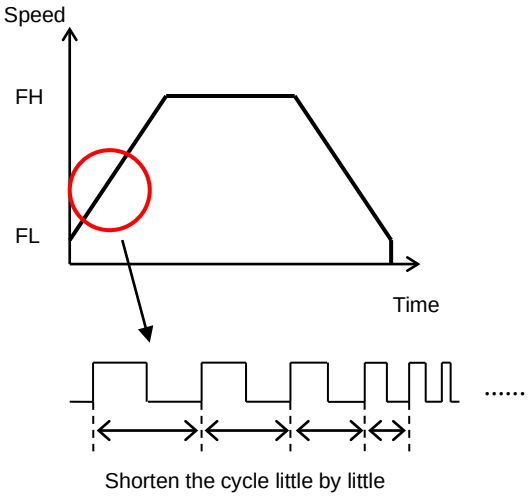
2-2. Advantage of using PCL

Controlling a stepping motor requires, of course, devices or circuits that produce pulse signals.

Motor control (motion control) is often created by "CPUs (1-chip microcomputers)" or "FPGA". So can program them to perform acceleration/deceleration speed controls. But please wait before doing it.
Do you know about dedicated LSIs specialized in motor control?
They are called "Pulse Control LSIs (PCL series)".
There are various names to call them, such as motor control ICs, motion control LSIs or pulse generators. However they all are the same thing basically.

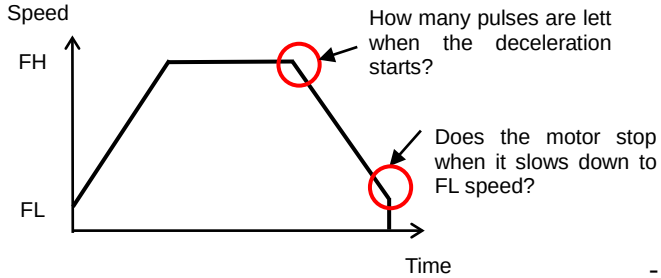
<A story from a customer>
★ Have you had any experiences to control motors with devices other than pulse control LSIs? -
No, not here.
Why? The answer is very simple; "PCLs are very convenient and make things easier."
For example, if only a CPU is used to create pulse signals, we heard that I need to create the programs to control counters and timers by making the ports ON and OFF.
If we only create a program using a constant speed control, it can be easy. However, it's a little troublesome to create a program using acceleration and deceleration controls.
★ A high-performance CPU is also required, isn't it? -
Recent CPUs are becoming high-performance in relatively inexpensive, and linear accelerations and decelerations may be possible. However, performing S-curve acceleration/deceleration can be a different subject. To create a program where acceleration/ deceleration rates changes all the time is very difficult, and we do not want to create such a troublesome program!
If you want to do it with a low-cost CPU, processing speed of other controls than motor controls become very slow. Therefore, it is necessary to use a somewhat high-performance CPU. Maybe....
However, if you use PCL chips, all you need to do is simply to write the setup data and to send the commands from the CPU, do it is very easy.
★What are the exact advantages of using PCL? What makes you comfortable? -
When considering the cost and development period in total, it is more advantageous to use PCL products. Not only to reduce costs, but also to develop and deliver products as soon as possible with limited number of development and design engineers, which is very important.

2-3. Is pulse control difficult with a CPU (1-chip microcomputer)?



When a pulse control is performed by a CPU, what is difficult?
Take an acceleration/deceleration speed control as an example...
It's a little professional spoken, but "higher speed = shorter pulse cycles." "To accelerate" means making a program so that the pulse cycles become gradually shorter.

When creating a program, simply shortening the pulse cycle is not enough. You need to calculate and program them correctly. Since the setting is related to acceleration and deceleration ratios, you have to set them very precisely.
Even linear acceleration/deceleration is "relatively difficult", so S-curve acceleration/deceleration can be "super difficult".



However, you have tried and overworked for days to write a program, and finally becomes able to output acceleration/ deceleration pulses using a CPU. But still you have many other things to consider and cannot take a break at this point.

For example,

● What will be the total number of pulses to output? and the pulses need to be counted.

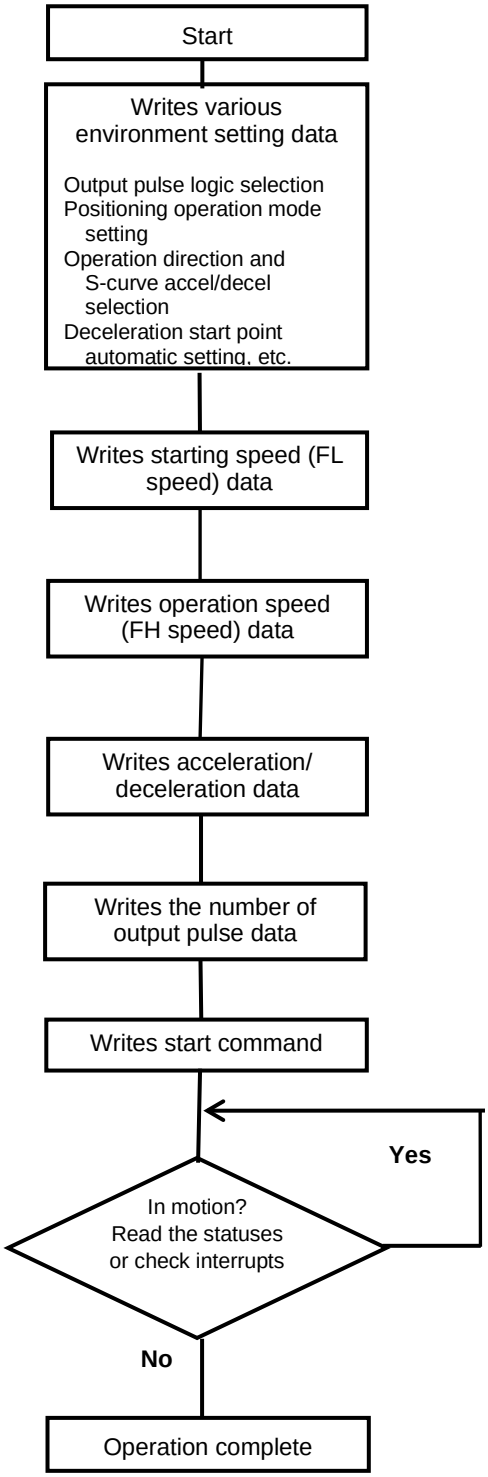
● Acceleration has started successfully, but at how many pulses remains should we start decelerating?

● Although a deceleration has started successfully, can we stop precisely when we decelerate to starting speed (FL speed)?

● The current position cannot be managed only by the number of output pulse, so a current position counter may be required.

●If any forced stop by an external signal is required, how should we process the signal?

We need to consider the above things.



As described above, when creating the software using CPUs (1-chip microcomputer), we believe that programming and processing speed settings are the most annoying. PCL can reduce such annoying in the development process.

If PCLs are used, for example, basic positioning controls can be performed with s-curve acceleration/deceleration as shown in the flow chart in the left.

That is everything you need to do. Isn't it simple?

You can perform positioning operation with S-curve acceleration/ deceleration, and you can also know the current position at any locations by reading the up/down counter.

When the operation is completed, an interrupt is issued to a CPU to inform that the operation is completed. Since the CPU can assign motor control work to PCL, the CPU can do other work during that time.

Also, in order to forcibly stop the operation by an external signal, a dedicated terminal can be used, so it is very easy.

It might be easy to understand the relationship if you consider CPU as the boss and PCL as the subordinate.

The boss (CPU) assigns a task (motor control) by giving data and instructions to the subordinate (PCL) with absolute confidence, and the boss can do other work (processing). After the task is completed, the subordinate informs the boss that it has been completed, and then wait for the next task from the boss.

Writing and Reading data

Sending data from CPU to PCL is called "writing data" to PCL.

There are two types of data to write: registers and commands.

Conversely, when CPU checks the current status of PCL (what numerical values have been set, what status has it been in, etc.), the process is called "reading data" from PCL.

The types of data to be read include registers, commands, and statuses.

In addition to reducing development man-hours, there are advantages as follows:

- High frequency pulses can be output.
- Motor control tasks are left to PCL, so the load to CPU becomes less.
- Specializing in motor controls ensures high operational reliability.

Now, we summarize the advantages and disadvantages of each control device as follows:

CONTROL DEVICE	ADVANTAGE	DISADVANTAGE
CPU/ 1-chip microcontroller	・ Low cost in low-end applications	・ It is difficult to create programs. ・ High-speed operations and complicated operations are difficult to perform. ・ It is necessary to test whether or not the actual operation will be realized. ・ High-speed CPU is required for complex operations. (or a dedicated CPU is required.) ・ Burden is applied due to the motor controls, and processing speed for other process becomes slow.
FPGA	・ Functions suited to the application can be created freely. ・ Programs can be rewritten, and correction is relatively easy.	・ It is very difficult to create programs. ・ It can freely create functions suitable for the application, so the functions tend to be limited only for the application. (Since unnecessary functions are not created,) it is difficult to use for other applications. ・ Relatively expensive and not suitable for mass production
Pulse control LSI (PCL)	・ The development man-hours can be shortened because the functions related to motor controls are built- in as the standard. (The operation has been verified in the factory.) ・ Low-end CPU can be used due to the less CPU burden. Motor control tasks are left to this dedicated LSI. ・ High frequent pulses can be output. ・ The outputting pulse cycle during acceleration or deceleration changes linearly, not by steps.	・ Depending on the CPU to be compared, the cost is slightly higher.

We have a lineup of PCL models that can be used for servo motors if the servo motor drivers are pulse signal input types.

3. Examples of basic configuration, application, and operation pattern using PCL.

NPM's pulse control LSI (PCL) was designed to control motors, and since its release in 1985, it has a long history in the market and has been supported by many customers a long period of time.

- Divides the incoming reference clock to create pulse signals at various frequencies suitable for motor control.
  ⇒ Controls the rotation speed of a motor by generating pulse signals at specified frequencies.
- Motor control can be "left" simply by giving operation pattern data and instructions (commands) from CPU.
  ⇒ The load on CPU can be reduced.
- Required number of axis/ functions can be selected out of five different series.

3-1. CPU Interface

PCL does not function by itself. You must have a source to control it. The source can be a one-chip microcomputer or a PC; anything is OK as long as data settings or a start instruction can be made. The source CPU can realize complicated motor controls by a simple data operation.

3-1-1. Parallel bus interface (8-bit, 16-bit)

Parallel bus interface is built-in to all models except PCD2112A and G9103C (described later).

- CS (Chip select signal Input)
- WR (Write enable signal input)
- RD (Read enable signal input)
- INT (Interrupt signal output)
- WRQ (Wait signal output)
- A0 to A2 [Address bus input (the number varies depending on the model)]
- D0 to D15 (data input/output)

Other terminals are used to perform operations that match the type of the source CPU.
User-friendly interface can be selected either for 68000, H8, 8086 or Z80 series CPU.

3-1-2. Serial bus interface

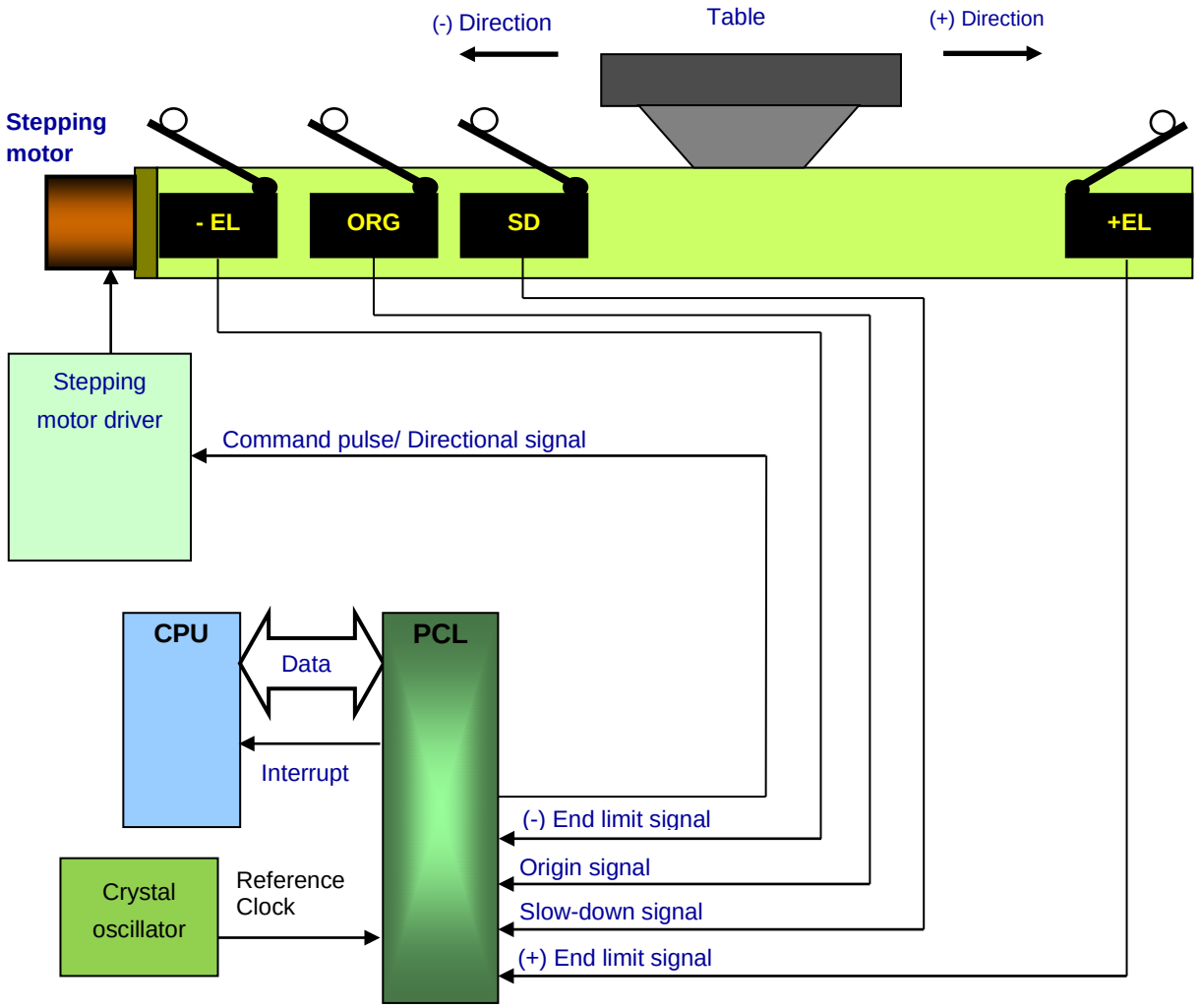
Recently, the number of CPUs that does not have parallel bus interface is increasing.
So, PCD46x1A series, PCD2112A, and PCL61x5 series (described later) have 4-wire synchronous serial bus interfaces to cope with it.
Up to four LSIs can be connected by one slave select signal (SS).
The connected LSIs can be identified by the device selection data set to DS0 and DS1 terminals.

- SCK (Serial Clock): Serial bus interface clock terminal.
- SS (Slave Select): Input terminal for slave(LSI) selection.
- MOSI (Master Output Slave Input): Input terminal from the master (CPU) to the slave (LSI).
- MISO(Master Input Slave Output): Outputs terminal from the slave to the master.

3-2. Examples of basic configuration with PCL

The basic configurations are shown in the figures below.

3-2-1. Connections using a stepping motor



The following signals can be input as mechanical position detection signals.

1) End limit signal

Stops immediately or decelerates to stop when the signal is turned ON in the moving direction and remains stopped even if the signal is turned OFF.

2) Slow-down signal

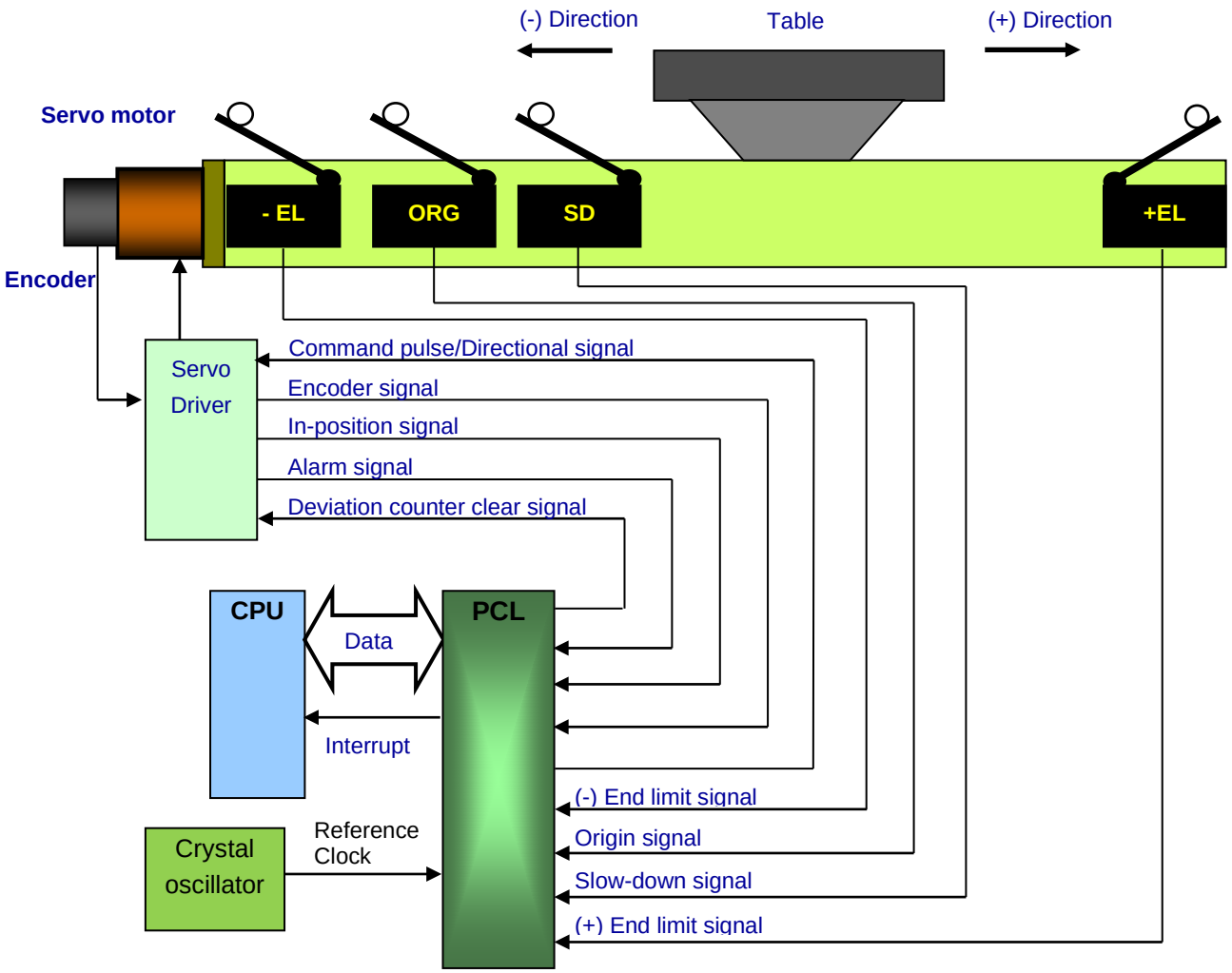
While this signal is enabled, if the signal turns ON, the speed is reduced to FL. After that, it will accelerate again when it turns OFF.

3) Origin signal

Used for an origin return operation.

Some models can decelerate to stop when the origin signal turns ON without using the slow-down signal.

3-2-2. Connections using a servo motor



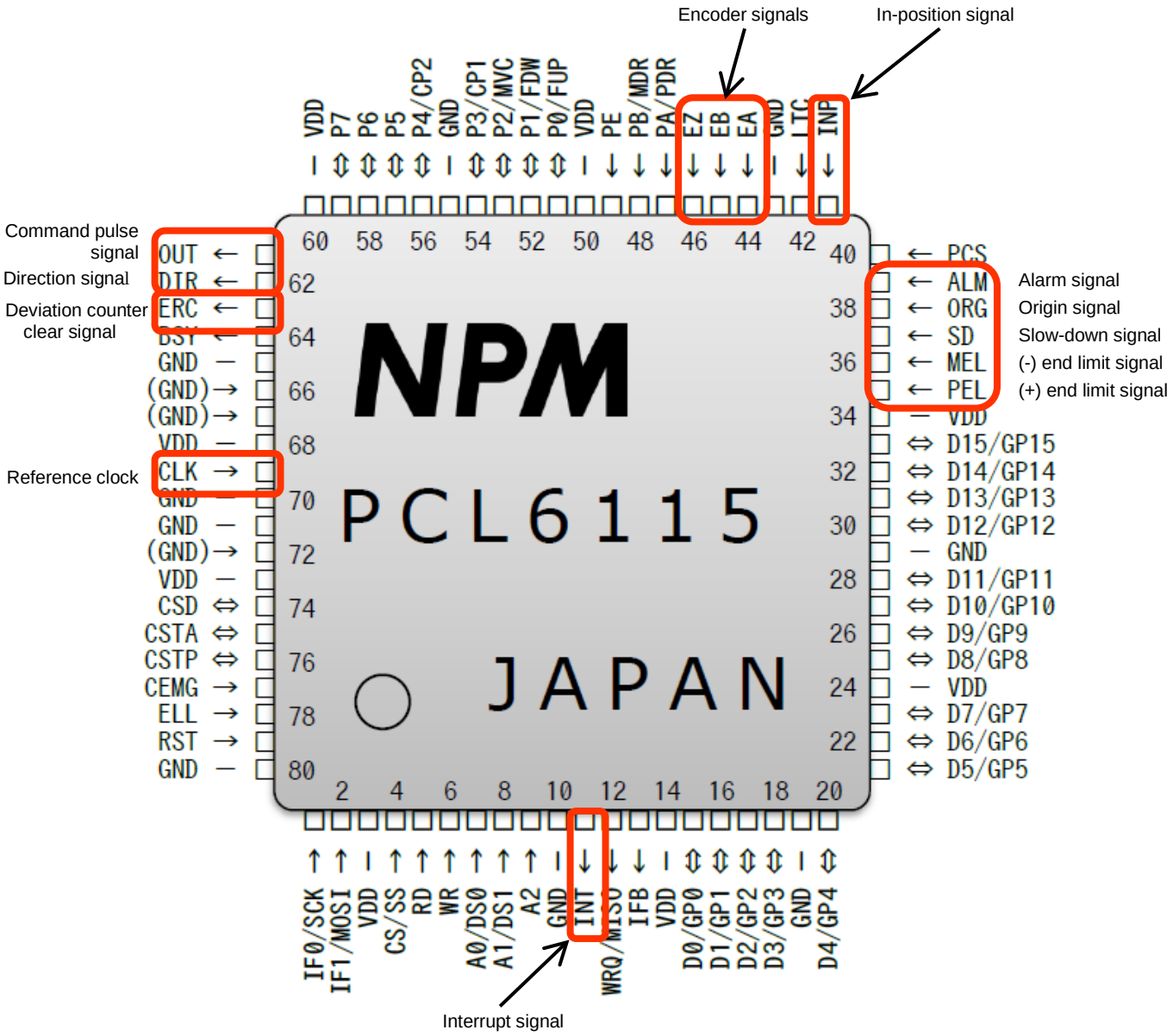
The mechanical position detection signals are the same as for a stepping motor. However, when using a servo motor, there are encoder signals, positioning completion signals, alarm signals, and deviation counter clear signals, which are described later.

* Hereinafter, the signal names and terminal names are described as follows.

- Output pulse signal to a motor driver
  Command pulse signal / direction signal (OUT / DIR)
- End limit signal
  Positive direction: (+) EL signal (PEL), Negative direction: (-) EL signal (MEL)
  (If no direction is specified: EL signal)
- Slow-down signal
  SD signal (SD)
- Origin signal
  ORG signal (ORG)
- Encoder signal
  A-phase signal (EA), B-phase signal (EB), Z-phase signal (EZ)
  (Refer to 9-2-7-2. about A-phase, B-phase and Z-phase signals)
- In-position signal
  INP signal (INP) (Refer to 9-2-5-2. In-position (INP) signal)
- Deviation counter clear signal
  ERC signal (ERC) (Refer to 9-2-5-3. Deviation counter clear (ERC) signal)
- Alarm signal
  ALM signal (ALM) (Refer to 9-2-5-4. Alarm (ALM) signal)

3-3. Example of terminal assignment diagram

The signals described in the previous page are assigned with terminals in the figure below (PCL6115 as an example).



3-4. Application

There is a lot of application experiences as follows:

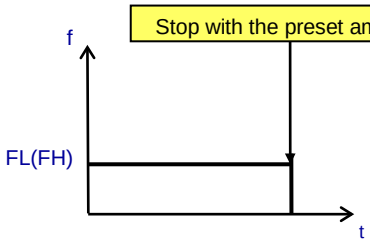
FA equipment	**Semiconductor and LCD production equipment**	**Medical, Health, and Bio-related equipment**	**Security, OA, others**
Injection molding machine Mounter Laser processing machine Coil winding machine Dispenser X-Y stage Knitting machine Paper processor Taping machine Food processing machine Robot Packaging machine Auto-soldering device	Exposure device Deposition system Etching Washer Probing Polishing Dicing Bonding LSI tester Handler Molding Visual inspection device Dimensional measuring device LCD glass processing	Hematology analyzer Liquid dispenser CT scan MRI Sampling device X-ray irradiation device Reagent unloading Analytical process device Microscope Electron microscope Care support	Monitoring camera Entry/exit control Parking control Professional-use 3D printers Multifunction device Laser printer Printing machine Labelling machine Card transport Bank ATM Locker Sorting device Fluid control Amusement device Home-automation

3-5. Examples of operation patterns

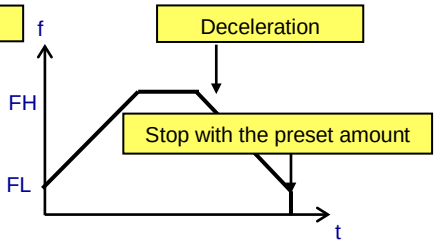
This section provides the overview of features and main functions of PCLs.

3-5-1. Preset operation (positioning operation)

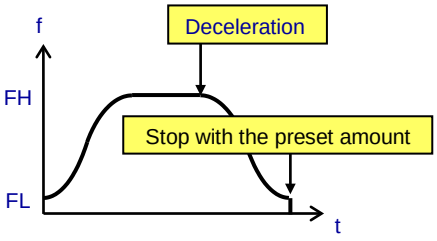
A motor stops when a predetermined feeding amount (= preset amount) is output.



Constant speed



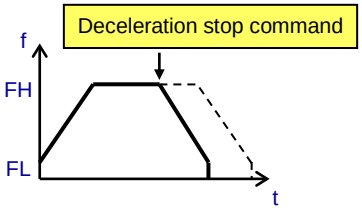
Linear acceleration/deceleration



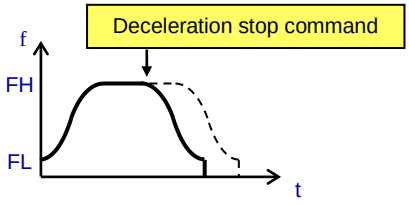
S-curve acceleration/deceleration

3-5-2. Deceleration stop operation

When the deceleration stop command or deceleration signal is entered, deceleration starts from that point, and stops when it reaches the stopping speed.



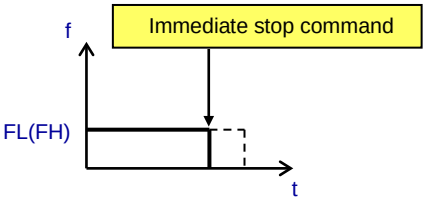
Linear acceleration/deceleration



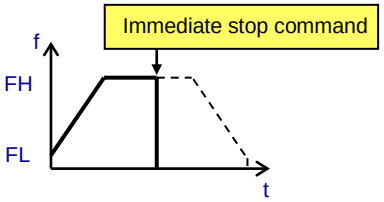
S-curve acceleration/deceleration

3-5-3. Immediate stop operation

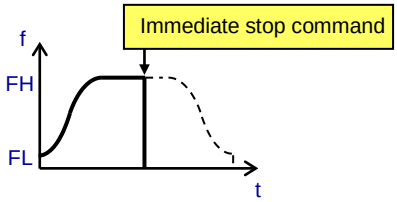
When the immediate stop command or immediate stop signal is entered, the LSI immediately stops regardless of the operation status.



Constant speed operation

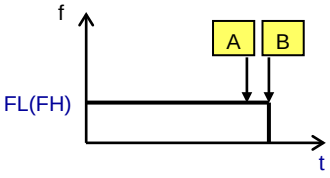


Linear accel/decel operation

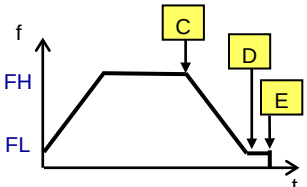


S-curve accel/decel operation

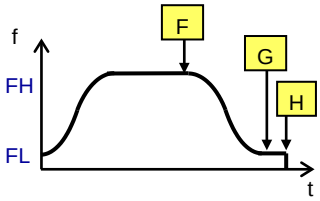
3-5-4. Origin return operation



Constant speed operation



Linear accel/decel operation



S-curve accel/decel operation

The origin return method can be selected from conditions such as the locations of ORG sensor, SD sensor, and EL sensor, and the combined use of Z-phase signal.

The following are the examples of origin return operations.

・Constant speed operation:

(1) Z-phase signal starts counting when ORG signal turns ON from OFF at point A, and Z-phase signal counts up and stops at point B.

(2) While Z-phase signal is not used, the operation stops when ORG signal turns ON from OFF at point B.

・Linear (S-curve) acceleration/deceleration operations:

(3)Deceleration starts when SD signal input turns ON from OFF at point C (F). After decelerating to FL speed at point D (G), the operation stops when ORG signal turns ON from OFF at point E (H)

(4) Deceleration starts when SD signal input turns ON from OFF at point C (F). When ORG signal turns ON from OFF at point D (G), Z-phase signal counting starts. At point E (H), Z-phase signal counts up and the operation stops.

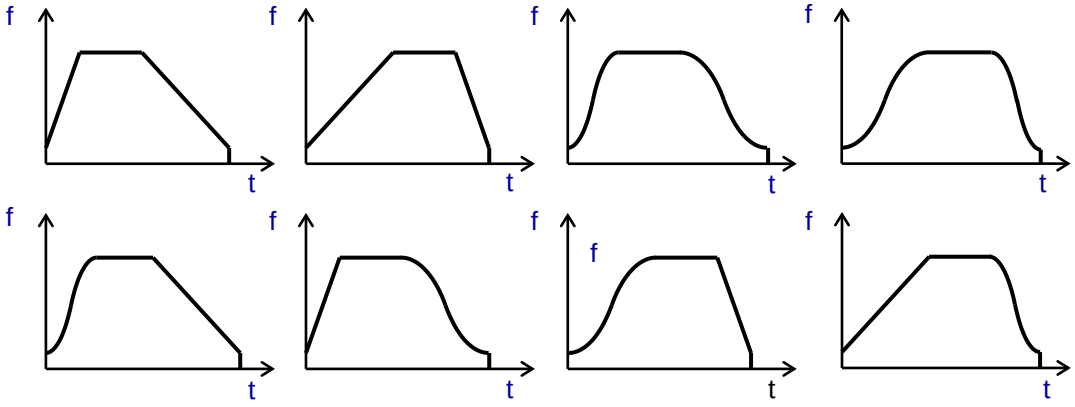
(5) Deceleration starts when ORG signal input turns ON from OFF at point C (F), and it decelerates to FL speed at point D (G), and the operation stops.

(6) Deceleration starts when ORG signal input turns ON from OFF at point C (F), and also Z-phase signal counting starts. At point E (H), Z-phase signal counts up and the operation stops.

PCL60xx series and G9103C have various other origin return methods such as using EL signals.
For details, refer to 9-2-4. Origin return operation.

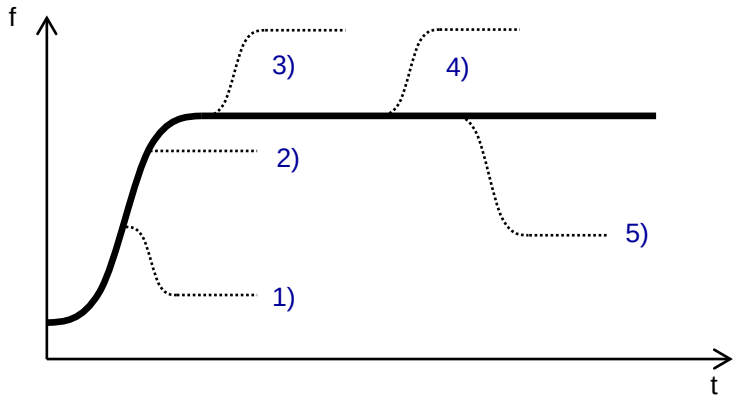
3-5-5. Examples of acceleration/deceleration pattern

Various acceleration/deceleration patterns can be performed by setting the speed pattern as shown in the figure below.



※ For PCD46x1A series, the acceleration and deceleration patterns cannot be set individually.

3-5-6. Change in speed patterns by changing speed (during S-curve acceleration/deceleration operation)



Although a continuous operation has been set as the bold line beforehand as shown in the figure, it is available to change the speed (acceleration or deceleration during operation) at any points as shown in 1) to 5).

4. Introduction of PCL series

4-1. PCD46x1A Series-

Low-cost version exclusively for stepping motors

- PCD4611A (1-axis)
- PCD4621A (2-axis)
- PCD4641A (4-axis)

This series LSIs are equipped with the basic functions that are generally necessary for stepping motor's open-loop control. Considering the recent CPU circumstances, both 8-bit parallel bus and 4-wire serial bus interfaces can be used to expand the available CPU options. The smallest QPP-type package size in the industry is adopted; 10×10 mm for 2-axis type and 14×14 mm for 4-axis type.
Multi-axis controllers for a stepping motor can be developed "in compact", "inexpensive" and “quick”.

◆ Supports the 8-bit parallel bus and the 4-wire serial bus
The parallel bus terminals can be used for general-purpose I/Os when the serial interface is used.
Up to four LSIs can be connected extendedly with a single slave selection signal.
◆ Maximum output frequency: 2.4 Mpps
◆ Linear/ S-curve acceleration/deceleration
◆ Simultaneous start/stop function
◆ Built-in 1 current position counter per axis
◆ Small package (industry's smallest for QFP type)
◆ Ambient temperature: it can be used in -40 to +85°C

● Slow-down point automatic setting function
● Idling pulse output function (1 to 7 pulses)
● Speed override during operation
● Operation mode (four types available)
● Interrupt signals can be output by three types of factors (event factors can be selected)

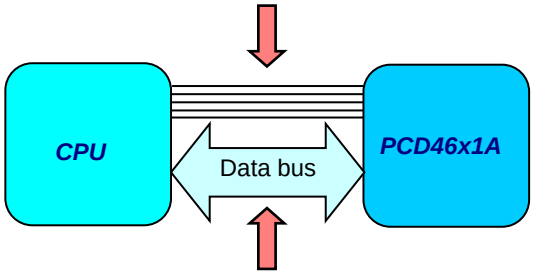
<Ideal for these requests!>
◆ I want to use a small, fewer pin, and inexpensive CPU.
◆ We are trying to use PCL for the first time for stepping motor control.
◆ Since we are having difficulties in designing a software using a PC, we want to design it more easily and quickly.
◆ No need to be sophisticated. Only need to support basic operations.

Either can be used

<Parallel bus method>

The main feature is that access from the CPU is fast.

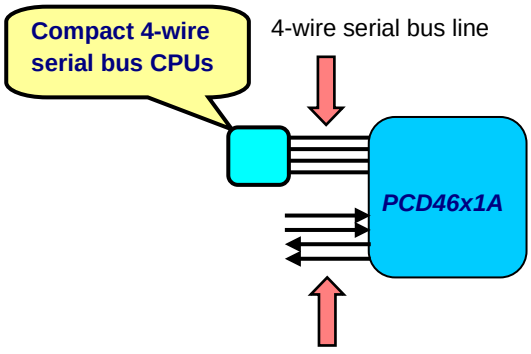
CS A0 to A3 RD WR each signal line



D0 to D7 data bus signal lines
General-purpose I/O port terminals are available for each axis.

<Serial bus method>

Need a little more time to access from the CPU, but fewer pin, smaller size, and lower cost CPU can be selected.



D0 to D5 terminals can be used as general-purpose I/O port terminals (6-pin). Other general-purpose I/O port terminals are provided for each axis.

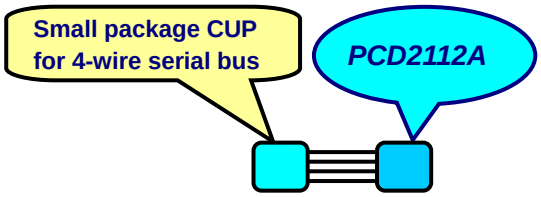
4-2. PCD2112A

Small package dedicated for 4-wire serial bus

Small packages (molded part: 7x7mm) exclusively for 4-wire serial bus and small CPUs with less terminals can be connected, so the entire board size can be miniaturized.

Encoder inputs, up/down counters, and servo motor interfaces are built-in, so closed-loop control of a stepping motor or a servo motor is also supported.

< Serial bus method >



◆ Connecting to CPUs via 4-wire serial bus

- This LSI can be used even with a CPU that has no external bus terminals.
- For a CPU with sharing terminals for external buses, the number of usable general-purpose I/Os can be adjusted.

◆ Optimization of control data allocation and block transfer method

Thus, it is possible to minimize the transmission time.

◆ New operation mode "Stand-alone system modes" that can be controlled without a CPU

CPU-less operation is enabled by attaching an external EEPROM to which up to 32 types of operation patterns have been written.

- Maximum output frequency: 5 Mpps (when reference clock 20 MHz is used)
- Pulse output type: selectable from 12 types; pulse signal outputs and excitation sequence outputs for 2-phase stepping motors
- 32-bit up/down counter is built-in
- Extensive operation mode (11 operation modes are available).
- Equipped with manual pulser input terminals (no multiplication or division functions)
- Interrupt signals can be output per 11 factors (event factor can be selected by the register).

<Ideal for these requests>

◆ We want to intelligently control a motor with a CPU that has few terminals.
◆ We want to make a small motor control board anyway.
◆ We want to use the it stand-alone during operation without connecting to a CPU.
◆ We want more functions than PCD46x1A series.

4-3. PCL61x5 series

Enhanced version of CPU interface with two modes: parallel bus and 4-wire serial bus

- PCL6115 (1-axis)
- PCL6125 (2-axis)
- PCL6145 (4-axis)

With built-in pre-register (one stage), up/down counter with two comparator systems per axis, linear interpolation function, encoder input, and servo motor interface, etc., it has functions that can sufficiently cope with general open-loop as well as general closed-loop controls. Considering recent CPU circumstances, both 8 and16-bit parallel bus and 4-wire serial bus can be used to expand the available CPU choices.

The maximum output frequency is up to 15 Mpps, and the up/down counter is 32-bit, which enables linear motor control with high resolution and long strokes.

◆ Supports 8/16-bit parallel bus and 4-wire serial bus
The parallel bus terminals can be used for general-purpose I/O ports in a serial interface.
Up to four LSIs can be connected with a single slave select signal.

◆ Allows linear interpolation between any 2-4 -axis.

◆ Maximum output frequency: 15 Mpps

◆ Up/down counter: 2 per axis built-in (32-bit)

◆ Comparator: 4 per axis built-in (32-bit) (2 of them are exclusively for software limit function)
The following operations can be performed by using comparator and counters.

- Interrupt output, external output of comparison result
- Ring count function
- Start by internal synchronous signal
- Additional software limit function

◆ Speed and target position can be overridden during operations

◆ Built-in pre-register to write the next operation data (feeding amount, speed, operation mode, etc.) during the current operation (1 stage). Since the data of the next operation is stored in advance, it is possible to smoothly transfer to the next operation without stopping when the current operation is completed.

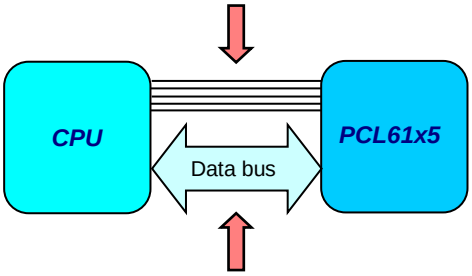
● A variety of operation modes (12 operation modes)
● Equipped with manual pulser inputs terminals (no multiplication and division functions)
● Interrupt signal can be output with 11 types of error factors and 21 types of event factors. (Event factors can be selected by a register.)

Either can be used

<Parallel bus method>

The access from the CPU is fast.

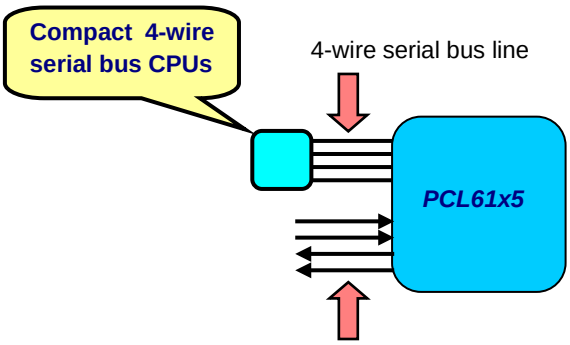
CS A0 to A4 each RD/WR lines



D0 to D15 data bus signal lines
8 general-purpose I/O port terminals are provided for each axis.

<Serial bus method>

It takes a little time to access from the CPU, but you can choose a fewer terminal pin, smaller size, and lower cost CPU.



D0 to D15 terminals pins can be used as general-purpose I/O ports (16 terminals). In addition, there are 8 general-purpose I/O port terminals for each axis.

4-4. PCL60xx series

The most advanced version of PCL

- PCL6025B (2-axis)
- PCL6045BL (4-axis QFP)
- PCL6046 (4-axis BGA)

They are equipped with various functions such as linear/circular interpolation, overriding speed or target position during operations, triangle drive elimination function, backlash correction, vibration restriction and software limit during stopping, direct input of operation switches, various origin return sequences, mechanical input and servo motor interface, etc. These versatile features make it easy to build a complex motion control system.
By adapting BGA package, PCL6046 enables the board to be downsized.

◆ Circular interpolation between any 2 axes; linear interpolation between any 2 to 4 axes
Allows a linear interpolation of 5 axes or more between two or more LSIs (3 axes or more for PCL6025B).

◆ Pre-register is used to realize continuous interpolation operations from circular to linear to circular interpolations...

◆ Maximum output frequency: Up to 6.5 Mpps (PCL6046: 10 Mpps)

◆ Up/down counter: 4 per axis built-in

PCL6046: 32-bit × 3 and 16-bit for deviation × 1

PCL6045BL/PCL6025B: 28-bit × 3 and 16-bit for deviation × 1

All counters can be latched and reset by a signal input, operating condition fulfillment, and command write, so they can be used for a wide variety of applications.

◆ Comparator: 5 pcs per axis built-in

- PCL6046: 32-bit × 5
  PCL6045BL/PCL6025B: 28-bit × 5

The following operations can be performed by using comparators and counters:

- Interrupt output, external output of comparison result
- Start by Internal synchronous signal
- Immediate stop or deceleration stop of operations
- Automatic speed change during operations
- Software limit function
- Out-of-step detection function for stepping motor
- Synchronous signal output
- Ring count function

◆ Speed and target position overrides during operation

◆ Built-in pre-register to write the next-operation data (feeding amount, speed, operation mode, etc.) during current operation (2 stages). Since the data of the next operation and the following operation can be stored in advance, it is possible to smoothly move to the next operation without stopping when the current operation is completed.

● You can access to the registers directly without going through I/O buffer (PCL6046 only)
● Variety of operation modes (24 operation modes)
● Constant synthetic speed control during Interpolation operations
● Equipped with manual pulser input terminal (with 32 multiplication and 2048 division function)
● 17 types of error factors and 20 types of event factors can be used to output interrupt signals. (The event factors can be selected by the register.)

4-5. G9103C

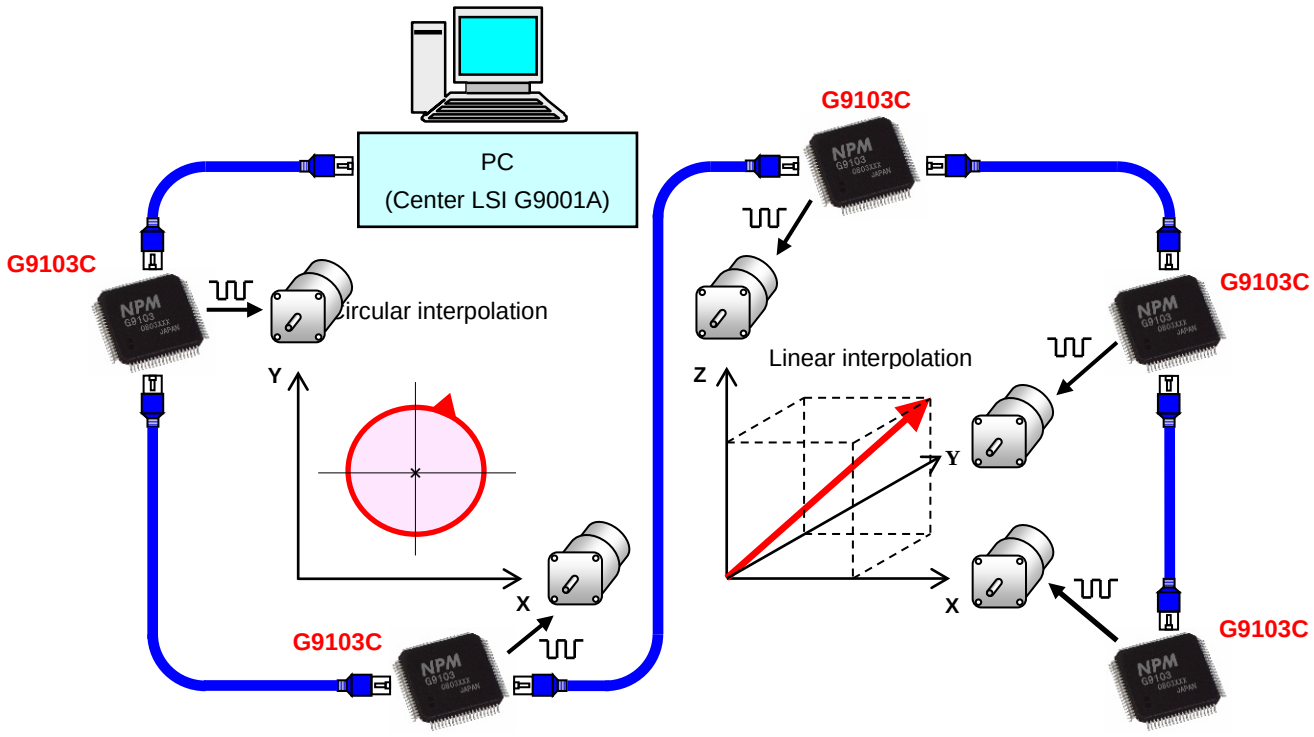
Interpolated control between separate local boards (Motionnet dedicated high-function 1−axis control)

This model is not a direct-connection type with CPUs, but a 1-axis control (pulse-control) local device for Motionnet, which is NPM's serial communication system. It has almost the same specifications as in the single axis of PCL60xx series, which is the most advanced series of PCLs.
By using more than one LSI, circular interpolations can be performed by two axes via Motionnet, and linear interpolations can be performed by two or more axes.
Since It is equipped with various functions such as overriding speed and target position during operation, triangular drive elimination, backlash correction, vibration restriction while stopping, software limits, various origin return sequences, mechanical inputs, and servo motor interface, it will be easy to build complex motion control systems.
You can connect the required number of G9103C to the center device,"G9001A", through a multi-drop configuration. The cyclic communication constantly communicates with the center device for the status information of general-purpose I/O and axis control of the devices, and the data communication reads and writes the status information of the axis control commands and registers.

◆ Up to 64 units can be connected per line.
◆ Circular interpolations for any two axes via Motionnet, and linear interpolations for any two axes or more
◆ Maximum output frequency: Up to 6.66Mpps
◆ Up/down counter: 3 pcs built-in (28-bit × 2 and 16-bit for deviation × 1)
◆ Comparator: 3 pcs per axis built-in
The following operations can be performed by using comparators and counters.
・ Interrupt output, external output of comparison result
・ Immediate stop or deceleration stop of operations
・ Software limit function
・ Out-of-step detection function for stepping motors
・ Synchronous signal output
◆ Overriding speed and target position during operation

● Number of general-purpose I/O: General-purpose I/O, 1 port (1 port = 8-bit): I/O can be set for each bit
● Communication data length: 1 to 4 words per frame (1 word = 16-bit)
● Communication method: cyclic communication (I/O port and status information), data communication (register data and commands, etc.)
● Pulse output types: Selectable from 12 types of pulse signal output and excitation sequence output for 2-phase stepping motors
● A variety of operation mode (44 operation modes)
● Built-in pre-register (one stage) to write and store the data (feeding amount, starting speed, operation speed, acceleration rate, deceleration rate, speed magnification, slow-down point, operation mode, s-curve acceleration range, s-curve deceleration range and interpolation related operations) for the next operation during the current operation.
● Equipped with manual pulser input terminals (with 32 multiplication and 2048 division function)
● Interrupt signal can be output with 22 types of error factors and 18 types of event factors (event factors can be selected by the register).

G9103C interpolation control image: Interpolation control between separate boards is available.



5. Register

This section explains PCL61x5 series. (Regarding other models, calculation formulas may be different, so please check the user's manual.)

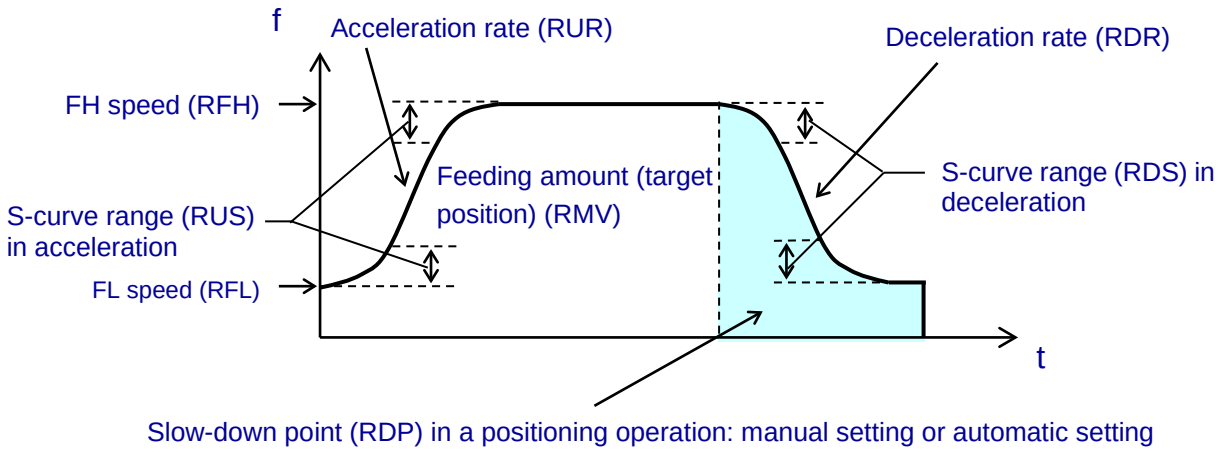
5-1. Speed pattern setting

How to determine the output frequency

Output frequency = Speed register value × speed magnification rate

For the models with the more step number for speed setting, the finer speed setting can be made.

Examples of the parameters of the model that S-curve acceleration/deceleration and S-curve range setting can be made. ("(Rxx)" in the figure is a register name to be described later.)



5-2. Registers for speed pattern settings

5-2-1. RMV: feeding amount (target position) setting register

- Sets the number of positioning pulses

5-2-2. RFL: FL speed setting register

- Sets the FL speed (starting speed. stopping speed)

  ⇒ The actual output frequency is obtained by multiplying this setting value by the speed magnification rate.

5-2-3. RFH: FH speed setting register

- Sets the FH speed (operating speed)

  ⇒The actual output frequency is obtained by multiplying this setting value by the speed magnification rate.

5-2-4. RUR: Acceleration rate setting register

- Sets the value to determine the acceleration rate

  ⇒ Substitutes this value into the given formula yields the acceleration time.

  < Example of calculation >

$$\text{Acceleration time} = \frac{(\text{RFH-RFL}) \times (\text{RUR+1}) \times 2}{\text{Reference clock frequency}}$$

$$\text{RUR} = \frac{\text{Reference clock frequency} \times \text{Acceleration time}}{(\text{RFH-RFL}) \times 2} - 1$$

5-2-5. RDR: deceleration rate setting register

- Sets the value to determine the deceleration speed.

  ⇒ Substitutes this value into the given formula yields the deceleration time.

  < Example of calculation >

$$\text{Deceleration time} = \frac{(\text{RFH-RFL}) \times (\text{RDR+1}) \times 2}{\text{Reference clock frequency}}$$

$$\text{RDR} = \frac{\text{Reference clock frequency} \times \text{Deceleration time}}{(\text{RFH - RFL}) \times 2} \text{ -1}$$

5-2-6. RMG: Speed magnification setting register

- Substitutes this set value into the given formula yields the magnification rate.

  ⇒ RFL or RFH setting value multiplied by this rate yields the actual output frequency.

  < Example of calculation >

$$\text{Speed magnification} = \frac{\text{Reference clock frequency}}{(\text{RMG+1}) \times 16384}$$

$$\text{RMG} = \frac{\text{Reference clock frequency}}{\text{Speed magnification} \times 16384} \text{ -1}$$

<Example of RMG setting >>> When reference clock frequency = 19.6608 MHz

3999	0.3 times	0.3 to 4,914.9 pps
1199	1 time	1 to 16,383 pps
599	2 times	2 to 32,766 pps
119	10 times	10 to 163,830 pps
11	100 times	100 to 1,638,300 pps
1	600 times	600 to 9,829,800 pps

5-2-7. RDP: Slow-down point setting register

- Sets the number of residual pulses to start decelerating in a positioning operation.

5-2-8. RUS: S-curve range setting register in an acceleration

- Sets the S-curve range in a S-curve acceleration.

  ⇒ Only when right after starting at FL speed, and right before reaching FH speed, partial S-curve can be performed, and the middle part can be a straight line.
  This can lower the acceleration rate compared to the maximum acceleration rate of the S-curve without a straight line.

5-2-9. RDS: Deceleration S-curve range setting register

- Sets the S-curve rage in a S-curve deceleration.

  ⇒ Only when right after deceleration starts from FH speed, and right before reaching FL speed, partial S-curve can be performed, and the middle part can be a straight line.
  This can lower the deceleration rate compared to the maximum deceleration rate of the S-curve without a straight line.

5-3. Registers related to environment settings

5-3-1. RMD: Operation mode setting register

- This register is used to set the operation mode (example below).
  - Select the type of operation (continuous operation, positioning operation, origin return operation, linear interpolation, circular interpolation, pulser operation, or operation directions, etc.)
  - Acceleration/Deceleration (linear/ S-curve)
  - Operation completion timing (normal/ early)
  - Slow-down point (auto/ manual)
  - Triangle drive elimination (enable/ disable)

5-3-2. RENV1 to RENV4: Environment setting registers

- These registers are used to set various conditions other than speed pattern or operation mode setting, which should be determined prior to the operation:
  - Selects the output type of pulse signal (2 pulses of CW/CCW, pulse and direction signals, or 90-degree phase difference signal)
  - Sets the logic of SD signal, ORG signal or ALM signal, etc.
  - Switches the functions of terminals
  - Sets the input specification of encoder signals or pulser signals (90-degree phase difference or 2-phase of CW/CCW)
  - Sets the origin return method
  - Sets what will be counted by a counter (output pulses, encoder inputs.)
  - Sets the comparator compare method (<, >, =)

5-4. Counting register

5-4-1. RCUN1/RCUN2: Counter register

- This register sets the counter and obtains the counting value.

5-4-2. RCMP1/RCMP2: Comparison data (comparator) register

- This register sets the comparison values for comparators.
- There are also RCMP3 and RCMP4, but they are only used to set the software limit values.

5-4-3. RLTC1 to RLTC4: Latched data register (read-only)

- This register is used to obtain the counting data latched by the counter latch signal input from outside.

5-5. Interrupt register

5-5-1. RIRQ: Event interrupt factor setting register

- This register sets a non-error(=event) interrupt factor (under which condition the interrupt signal to be output to CPU).
  ⇒ When the operation stops, acceleration ends, deceleration starts, deceleration ends, comparator condition is fulfilled, ORG signal is turned ON, or SD signal is turned ON, etc.

5-5-2. REST: Error interrupt factor check register

- This register is used to obtain the error interrupt factor.
- The content of interrupt factor can be checked such as when stopped by EL signal is turned ON, by ALM signal is turned ON, by emergency stop signal ON, by encoder signal input errors, or by software limit signal is detected, etc.

5-5-3. RIST: Event interrupt factor check register

- This register is used to obtain the event interrupt factor.
- The factor can be checked such as when stopped, acceleration ends, deceleration starts, deceleration ends, comparator condition is satisfied, ORG signal is turned ON, or SD signal is turned ON, etc.

6. Command

The following is the commands of PCL61x5. (There are more.)
Writes the command code (xxh) below in the given location (buffers for commands).

6-1. Operation command

(1) Start command

- FL constant speed start (50h) "Move at FL speed without acceleration"
- FH constant speed start (51h) "Move at FH speed from the beginning without acceleration"
- Acceleration/deceleration start (53h) "Start by accelerating from FL speed"

(2) Start command with residual pulses

- Residual pulses FL constant speed start (54h) "Positioning operation is paused in the middle, but start by the residual pulses at FL speed".
- Residual pulses FH constant speed start (55h) "Positioning operation is paused in the middle, but start by the residual pulses at FH speed".
- Residual pulses accel/decel start (57h) "Positioning operation is paused in the middle, but start by the residual pulses by accelerating or decelerating from FL speed."

(3) Simultaneous start command (06h) "Do not start before you receive an external signal."

(4) Speed change command (42h) "Decelerates from current operation speed to FL speed."

(5) Stop command

- Immediate stop (49h) "Stop immediately while operating."
- Deceleration stop (4Ah) "Decelerate now and stop when reaching FL speed."

(6) Emergency stop command (05h) “Stop immediately regardless of pulse output width or pulse cycle completion.”

6-2. General-purpose output bit control command (10h to 1Fh)

"Turn ON the specified I/O terminal."

6-3. Control command

1) Software reset (04h) “Stop signal outputs and clear the contents of I/O buffers and registers.”
2) Counter reset (24h, 25h) “Reset the specified counter”

6-4. Register control command

There are writing commands (when writing data to each register) and reading commands (when reading data from each register).
(e.g.) RMV: Feeding amount register

- Writing command : 90h “Set the I/O buffer data into RMV (feeding amount) register.”
- Reading command : D0h “Set the current RMV register data into I/O buffer.”

7. Status

The current operating status of PCL and the input status of external signals (EL, ORG, etc.) input to PCL can be monitored from CPU.

CPU check the state of PCL, and instructs the PCL what to do next. Also instructs other PCLs and devices what to do. .

As an example, the statuses of PCL61x5 series include "Main status", "Sub status", and "Extension status obtaining register".

The current status of PCL can be checked by reading each status from CPU.

The read result is shown by "xxh" in hexadecimal code. (Status is read-only)

Some sample phrases in words are shown as below:

- "Start command has been written and it is operating."
- "Pulse output has stopped and the operation is on hold."
- "Error interrupt occurred."
- “Event interrupt occurred.”
- "Comparator condition is met.”
- "Target position failed to be overridden."
- "Currently it is accelerating.”
- "Now operating at FL speed or FH speed at "constant speed"."
- "Alarm (ALM) signal turns ON."
- "(+) EL signal [(-) EL signal] turns ON."
- "ORG signal turns ON."
- "SD signal turns ON."
- “INP signal turns ON.”

The status can be checked as to which bit is set to "1" in binary as follows:

< Examples >

(1) Bit 0 of main status is "1" "Start command has been written and is operating"

(2) Bit 8 of sub-status is "1" “Currently accelerating"

- RSTS: Extension status obtaining register

The level monitoring of various signals, direction of the current operation and counter-latch history can be obtained.
by this register. Those are not in the bit contents of the main status or the sub-status (some are overlapped).

8. Basic specifications and technical terms

8-1. Basic specification

Model (Series) / Specification	PCD46x1A	PCD2112A	PCL61x5	PCL60xx		G9103C
Number of controllable axis	1 (PCD4611A) 2 (PCD4621A) 4 (PCD4641A)	1	1 (PCL6115) 2 (PCL6125) 4 (PCL6145)	2 (PCL6025B) 4 (PCL6045BL)	4 (PCL6046)	1
Reference clock (Std, Max.)	4.9152 MHz (Max. 10 MHz)	9.8304 MHz (Max. 20 MHz)	19.6608 MHz (Max. 30 MHz)	19.6608 MHz (Max. 20 MHz)	19.6608 MHz (Max. 30 MHz)	80 or 40 MHz
Maximum output frequency (Std, Max.)	2.4 Mpps (Max. 5 Mpps)	2.4 Mpps (Max. 5 Mpps)	9.8 Mpps (Max. 15 Mpps)	6.5 Mpps	6.5 Mpps (Max. 10 Mpps)	6.66 Mpps
Number of speed setting registers	2 (FL、FH)	2 (FL、FH)	2 (FL、FH)	3 [FL, FH, FA (for adjustment)]	3 [FL, FH, FA (for adjustment)]	3 [FL, FH, FA (for adjustment)]
Step number for speed setting	1 to 8,191 (13-bit)	1 to 8,191 (13-bit)	1 to 16,383 (14-bit)	1 to 65,535 (16-bis)	1 to 65,535 (16-bit)	1 to 100,000 (17- bit)
Speed magnification setting range	1 - 300 times	0.5 to 300 times	0.3 to 600 times	0.1 to 100 times	0.1 to 152.5 times	0.1 to 66.6 times
Acceleration rate setting range	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)
Deceleration rate setting range	Same as acceleration	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)	1 to 65,535 (16-bit)
Positioning pulse number setting range	0 to 16,777,215 (24-bit)	0 to 268,435,455 (28-bit)	-2,147,483,648 to +2,147,483,647 (32-bit)	-134,217,728 to +134,217,727 (28-bit)	-2,147,483,648 to +2,147,483,647 (32-bit)	-134,217,728 to +134,217,727 (28-bit)
CPU Interface	8/16-bit parallel 4-wire serial	4-wire serial	8/16-bit parallel 4-wire serial	8/16-bit parallel	8/16-bit parallel	G9000 communication Interface
Slow-down point setting range	0 to 16,777,215 (24-bit)	0 to 16,777,215 (24-bit)	0 to 16,777,215 (24-bit)	0 to16,777,215 (24-bit)	0 to 16,777,215 (24-bit)	0 to 16,777,215 (24-bit)
Package	44 pin QFP (PCD4611A) 64 pin QFP (PCD4621A) 100 pin QFP (PCD4641A)	48 pin QFP	80pin QFP (PCL6115) 128pin QFP (PCL6125) 176pin QFP (PCL6145)	128 pin QFP (PCL6025B) 176 pin QFP (PCL6045BL)	208 pin BGA	80 pin QFP
Package dimension (Molded part) (mm)	7×7 (PCD4611A) 10×10 (PCD4621A) 14×14 (PCD4641A)	7×7	12×12 (PCL6115) 14×14 (PCL6125) 24×24 (PCL6145)	20×14 (PCL6025B) 24×24 (PCL6045BL)	12×12	12×12
Power-supply voltage	+3.0 to +3.6V	+3.0 to +3.6V	+3.0 to +3.6V	+4.5 to +5.5V and +3.0 to +3.6V 2 power supplies (PCL6025B) +3.0 to +3.6V (PCL6045BL)	+3.0 to +3.6V	+3.0 to +3.6V
Ambient temperature	-40 to +85℃	-40 to +85℃	-40 to +85℃	-40 to +85℃	-40 to +85℃	-40 to +85℃

8-2. Explanations of technical terms in catalogues and user's manuals

Number of controllable axis	The number of axis that can be controlled by one chip.
Reference clock (Standard, Max.)	The frequency of reference clock input to the pulse control LSI. A frequency other than the standard frequency can be input, but the output frequency may not be divisible.
Maximum output frequency	Maximum frequency of output pulse.
Number of speed setting registers	There are FL speed (starting speed) and FH speed (operation speed). The speeds can be changed by rewriting FH speed register value during an operation.
Step number for speed setting	Indicates the "number of steps" that can be used for speed setting. The more bits, the more detail the frequency can be set.
Speed magnification setting	The output pulse frequency is obtained by multiplying the value in speed setting register by the magnification rate.
Acceleration rate setting	Set the inclination (acceleration) during acceleration. Acceleration time can be calculated from this setting value.
Deceleration rate setting	Set the inclination (deceleration) during deceleration. Deceleration time can be calculated from this setting value.
Positioning pulse number setting	Set the number of feeding pulses to be output in positioning operation.
CPU Interface	User's manual lists the typical types of CPUs that can be interfaced. Some models support 4-wire serial bus.
Slow-down point setting	Set the number of residual pulses from when deceleration starts in positioning operation.

9. Function

The following chart describes the features and functions equipped with PCD/PCL series. .

9-1. List of functions

Model (Series) / Function	PCD46x1A	PCD2112A	PCL61x5	PCL6025B PCL6045BL	PCL6046	G9103C
S-curve acceleration/deceleration	●	●	●	●	●	●
S-curve range setting		●	●	●	●	●
Triangle drive elimination function (Automatic FH correction function)		●	●	●	●	●
Origin return operation	● (2 types)	● (4 types)	● (4 types)	● (13 types)	● (13 types)	● (13 types)
Origin search, Origin escape		● (Origin escape only)		●	●	●
Origin return operation with feeding amount limit	●	●				
Limit positioning operation				●	●	●
Limit escape operation		●		●	●	●
Servo motor interface		● (※1)	●	●	●	●
Stepping motor interface (CDW signal)		●				●
Encoder inputs (Up to 4- multiplication available)		●	● (each axis)	● (each axis)	● (each axis)	●
Encoder Z-phase used origin return		●	● (each axis)	● (each axis)	● (each axis)○	●
Slow-down point auto setting function	●	●	●	● (※2)	● (※2)	● (※2)
Up/down counter (Current position counter)	● (each axis) 24-bit × 1	● 32-bit × 1	● (each axis) 32-bit × 2	● (each axis) 28-bit × 3 16-bit × 1 (*3)	● (each axis) 32-bit × 3 16-bit × 1 (*3)	● 28-bit × 2 16-bit × 1 (*3)
Up/down counter zero return (Software origin automatic return)				●	●	●
Counter hard latch			●	●	●	●
Comparator			● (each axis) 32-bit × 4 (2 for software limit only.)	● (each axis) 28-bit × 5	● (each axis) 32-bit × 5	● 28-bit × 3
Mechanical external signal input	●	●	● (each axis)	● (each axis)	● (each axis)	●
Direct access to internal registers					●	
Interrupt signal output	● (3 factors)	● (11 factors)	● (32 factors)	● (37 factors)	● (37 factors)	● (40 factors)
Interrupt factor setting	●	●	●	●	●	●
Interrupt status (Interrupt factor monitor)	●	●	●	●	●	●
Status	● (26 types)	● (38 types)	● (48 types)	● (54 types)	● (57 types)	● (45 types)
Pre-buffer for the next operation (Pre-register)			● (1 stage)	● (2 stages)	● (2 stages)	● (1 stage)
Next operation auto start control			●	●	●	●
Command buffer monitor	●	●	●	●	●	●
Output pulse logic selection	●	●	●	●	●	●
Output pulse mode selection	●	●	●	●	●	●
Excitation sequence outputs for 2-phase stepping motors	●	●				●
Monitor signal output terminal	● (1 type)	● (2 types)	● (6 types per axis)	● (9 types per axis)	● (9 types per axis)	● (10 types)
Pulser input		● (No multiplication/ no division functions	●(each axis) (No multiplication/ no division functions)	●(each axis) (32 multiplication and 2048 division functions)	●(each axis) (32 multiplication and 2048 division functions)	● (32 multiplication and 2048 division functions)
Pulser synchronous positioning		●	●	●	●	●

Model (Series) / Function	PCD46x1A	PCD2112A	PCL61x5	PCL6025B PCL6045BL	PCL6046	G9103C
Linear interpolation			● (※4)	●	●	● (※4)
Circular interpolation				●	●	● (※5)
Continuous interpolation function			●	●	●	●
Target position override			●	●	●	●
1-pulse output				●	●	●
Idling pulse output	● (0 to 7 pulses)	● (0 to 7 pulses)		● (0 to 7 pulses)	● (0 to 7 pulses)	● (0 to 7 pulses)
Output pulse width control			●	●	●	●
Simultaneous start/ stop	●	●	●	●	●	●
External start/ stop	●	●	●	●	●	●
Stepping motor out-of-step detection function				●	●	●
I/O port (General-purpose I/O terminal)	● (Per axis I/O x 4 Input only x 2 output only x 1) (※6)	● (I/O x 2 output only x 2)	● (8 points per axis) (※7)	● (8 points per axis)	● (8 points per axis)	● (8 points)
Operation switch input terminal		●	●	●	●	
Ring count function			●	●	●	●
Backlash correction function				●	●	●
Software limit function			●	●	●	●
Timer operation	●	●	●	●	●	●
Synchronous signal output			●	●	●	●
Vibration restriction function				●	●	●
Stand-alone operation modes		●				
5V interface support	●	●	●	●	●	●

(※1) Alarm signal cannot be connected on PCD2112A.

(※2) When (deceleration time) ≦ (acceleration time × 2), slow-down point auto setting can be performed in PCL60xx series and G9103C.

(※3) PCL60xx and G9103C are also equipped with counters that can be used as deviation counters.

(※4) A Linear interpolation can be performed by connecting more than one PCL6115s or G9103Cs.

(※5) A circular interpolation can be performed by connecting more than one G9103Cs.

(※6) It indicates the number of I/O points when the excitation sequence output function for 2-phase stepping motor is not used.
When this function is used, only one output is provided for each axis. When using 4-wire serial bus mode, D0 to D5 terminals (data bus terminals in parallel bus mode) can be used as general-purpose I/O port terminals. (6 pcs.)

(※7) When using 4-wire serial bus mode, D0 to D15 terminals (data bus terminals in parallel bus mode) can be used as general-purpose I/O port terminals. (16 pcs.)

9-2. Function description

This chapter provides additional explanations to some functions listed in the previous section.
Please note that the following symbols indicate the following meanings.

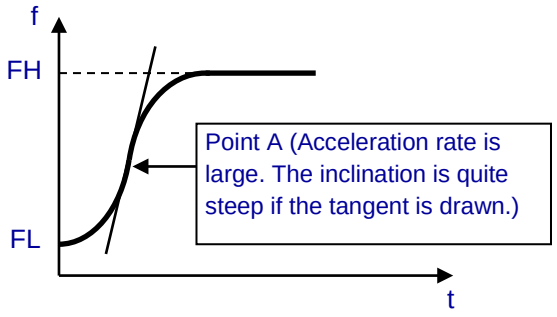
PCD46x1A	:	Functions exist in PCD46x1A series
PCD2112A	:	Functions exist in PCD2112A
PCL61x5	:	Functions exist in PCL61x5 series
PCL60xx	:	Functions exist in PCL60xx series
G9103C	:	Functions exist in G9103C

9-2-1. S-curve acceleration/deceleration operation

PCD46x1 PCD2112A PCL61x5 PCL60xx G9103C

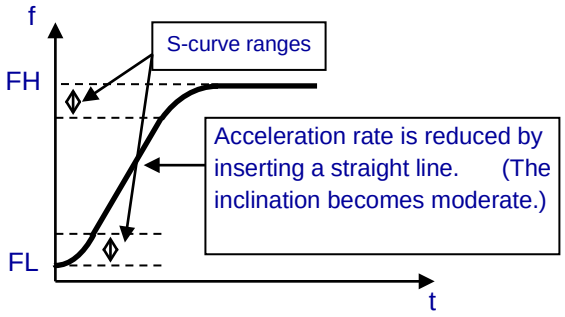
This operation performs acceleration/deceleration in S-curve. It is useful for vibration restriction of mechanisms that used to perform a linear acceleration/deceleration. (The effects of vibration restriction vary depends on the conditions such as the motor, the mechanism, and the operation pattern.)

9-2-2. Acceleration/deceleration S-curve range setting



This function allows you to insert a straight line between the beginning and the end of S-curve to reduce the duration of the s-curve acceleration/deceleration. By setting the range of the S-curve, the start and end portions of acceleration and deceleration become S-curves, and the middle range become a straight line.

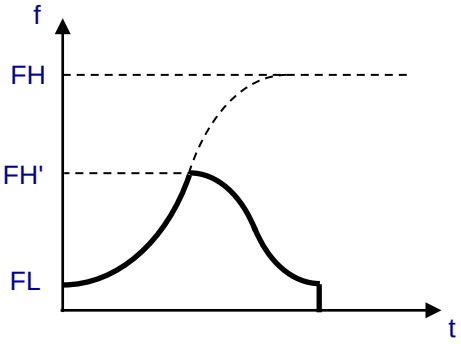
In a normal S-curve acceleration, the acceleration rate increases (the inclination becomes steep) in the middle of S-curve (Point A in the left figure). Therefore, the operation may fail into out-of-step with stepping motors.



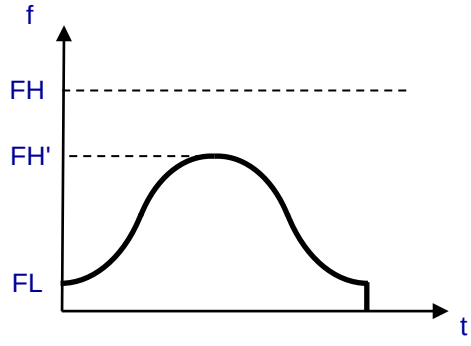
The middle part of S-curve acceleration can be straight (set the S-curve ranges) in PCD2112A, PCL61x5/PCL60xx series and G9103C. It can reduce the acceleration rate, so that it can prevent out-of-step from occurring.

9-2-3. Triangular motion elimination function (Auto FH correction function)

PCD2112A PCL61x5 PCL60xx G9103C



If FH speed is too high for a feeding amount or a feeding amount is too small for FH speed, the operation arrives at the slow-down point before reaching the set FH speed, and a deceleration starts from there. This is called a triangular motion.
If a motor is operated with a triangular motion, vibrations may affect the mechanisms.



Triangle drive avoidance function automatically lower the initially set FH-speed to avoid a triangular motion.
The lowered FH speed can also be calculated manually, e.g. when the operating time needs to be determined accurately.

9-2-4. Origin return operation

For origin return operations with acceleration/deceleration, a slow-down start (SD) sensor was located right before the origin (ORG) sensor in previous systems. So, the slow-down sensor (SD) turns ON and it decelerates to FL speed sufficiently before the ORG sensor turns ON.

Currently, you can start a deceleration stop when ORG sensor turns ON without using the SD sensor. Furthermore, in some models, an origin return operation can be done only by using an EL sensor, not using an ORG sensor.

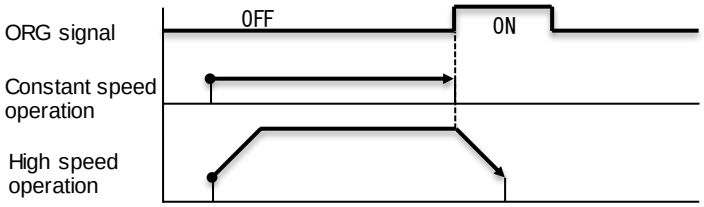
This section explains the types of origin return operations that are available for each series.

Please note that the notations of speed type in an origin return operation are defined as follows:

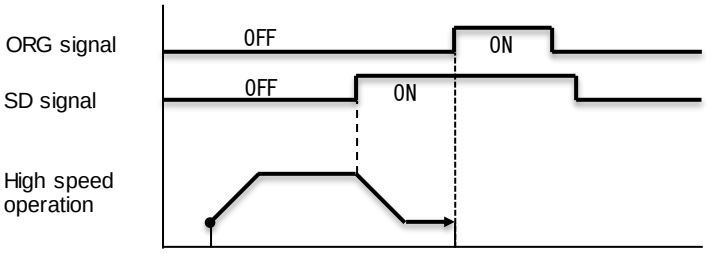
- Constant speed at FL speed or FH speed: Constant speed operation
- Acceleration from FL speed to FH speed: High speed operation.
- Constant speed for various correction operations: FA speed
- Deceleration starts when ORG signal turns ON, and the operation stops when reaching FL speed: Deceleration stop

9-2-4-1. Origin return operation 0

PCD46x1A | PCD2112A | PCL61x5 | PCL60xx | G9103C



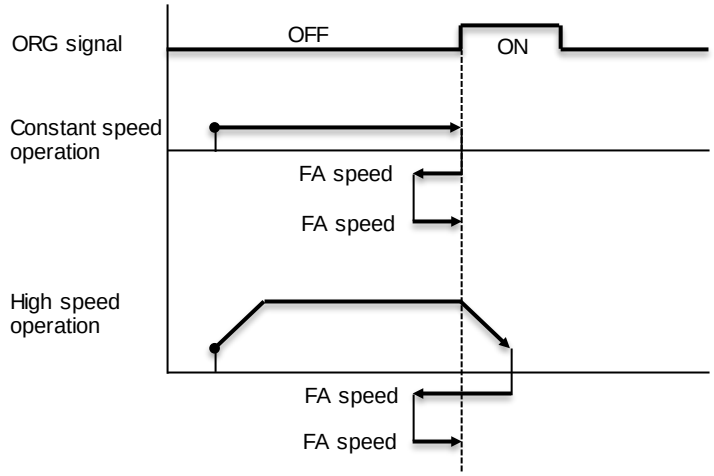
This is the most basic origin return method.
During constant speed operations, it performs an immediate stop when ORG signal turns ON.
During a high-speed operation, it performs deceleration stop when ORG signal turns ON.



When using SD signal in high speed operation, deceleration starts when SD signal turns ON, and it performs immediate stop when ORG signal turns ON.

9-2-4-2. Origin return operation 1

PCL60xx | G9103C

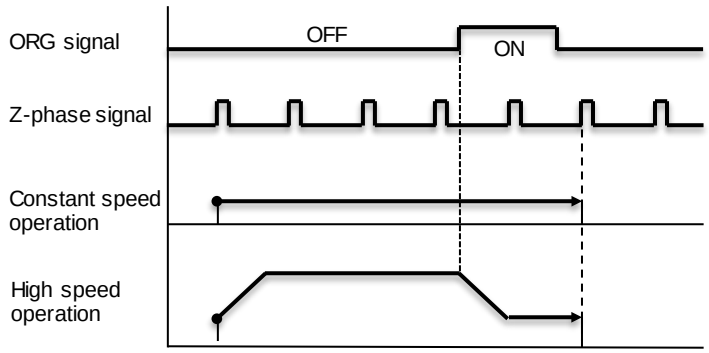


During a constant speed operation, it performs immediate stop when ORG signal turns ON, and then starts in the opposite direction at FA speed until ORG signal turns OFF. Then it moves in the initial direction at FA speed, and then immediately stops when ORG signal turns ON.

During a high-speed operation, it performs deceleration stop when ORG signal turns ON, then starts in the opposite direction at FA speed until ORG signal turns OFF. Then moves in the initial direction at FA speed, and immediately stops when ORG signal turns ON.

9-2-4-3. Origin return operation 2

PCD2112A | PCL61x5 | PCL60xx | G9103C

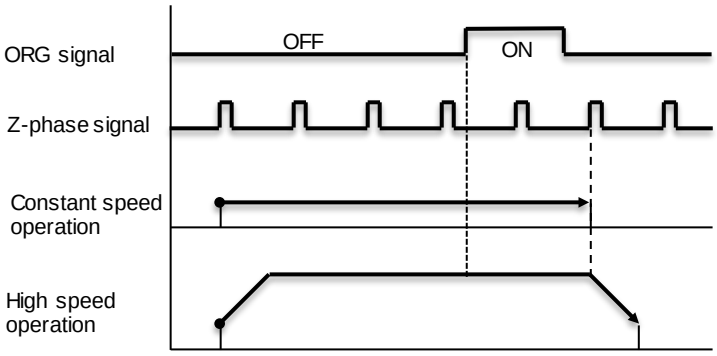


During a constant speed operation, Z-phase signal counting starts when ORD signal turns ON, and it immediately stops when Z-phase signal counting completes the preset number of times (it is called “Z-phase signal count up”).

During a high-speed operation, deceleration starts when ORG signal turns ON, and Z-phase signal counting also starts at the same time. Then, it immediately stops when Z-phase signal is counted up. (If the speed has never reached at FL when the Z-phase signal is counted up, the motor stops at the speed higher than the FL speed.)

9-2-4-4. Origin return operation 3

PCD2112A | PCL61x5 | PCL60xx | G9103C

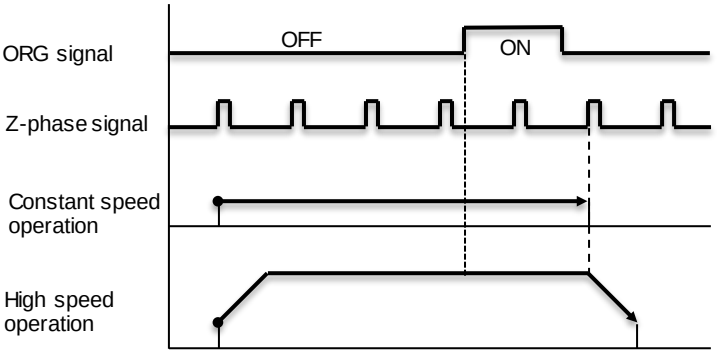


During a constant speed operation, Z-phase signal counting starts when ORG signal turns ON, and it performs immediate stop when Z-phase signal is counted up.

During a high-speed operation, Z-phase signal counting starts when ORG signal turns ON, and it performs deceleration stop when the Z-phase signal is counted up.

9-2-4-5. Origin return operation 4

PCL60xx | G9103C

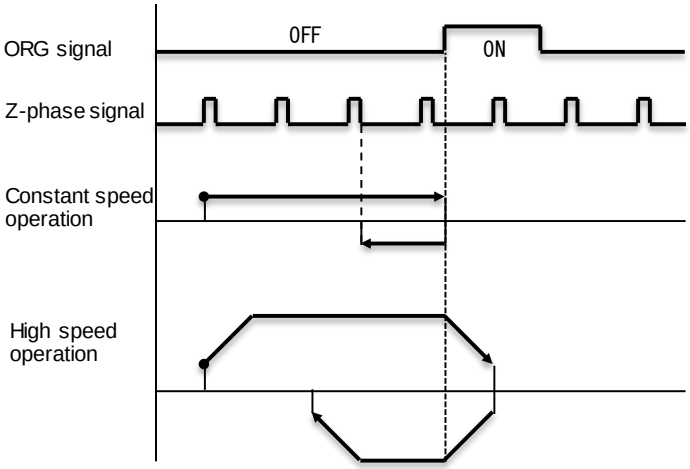


During a constant speed operation, it performs immediate stop when ORG signal turns ON, and then starts in the opposite direction at FA speed. Z-phase signal counting starts when ORG signal turns OFF, and it performs immediate stop when Z-phase signal is counted up.

During a high-speed operation, it performs deceleration stop when ORG signal turns ON, and starts in the opposite direction at FA speed. Z-phase signal starts counting when ORG signal turns OFF, and then it performs immediate stop when Z-phase signal is counted up.

9-2-4-6. Origin return operation 5

PCL60xx G9103C

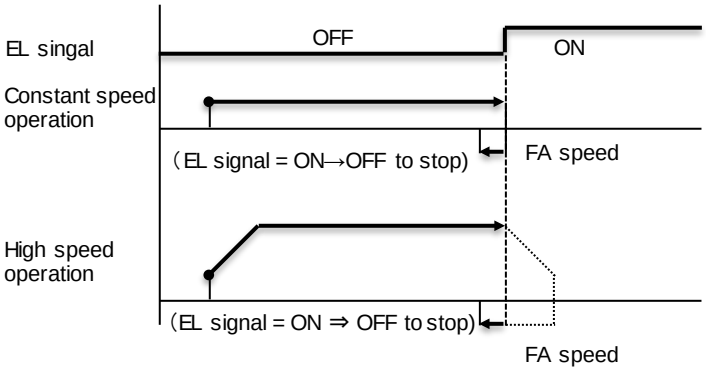


During a constant speed operation, it performs immediate stop when ORG signal turns ON, and then starts in the opposite direction at FL-speed. Z-phase signal counting starts when ORG signal turns OFF, and it immediately stops when Z-phase signal is counted up.

During a high-speed operation, it performs deceleration stop when ORG signal turns ON, and starts in the opposite direction while accelerating from FL speed to FH speed. Z-phase signal starts counting when ORG signal turns OFF, and then it performs deceleration stop when Z-phase signal is counted up.

9-2-4-7. Origin return operation 6

PCL60xx G9103C

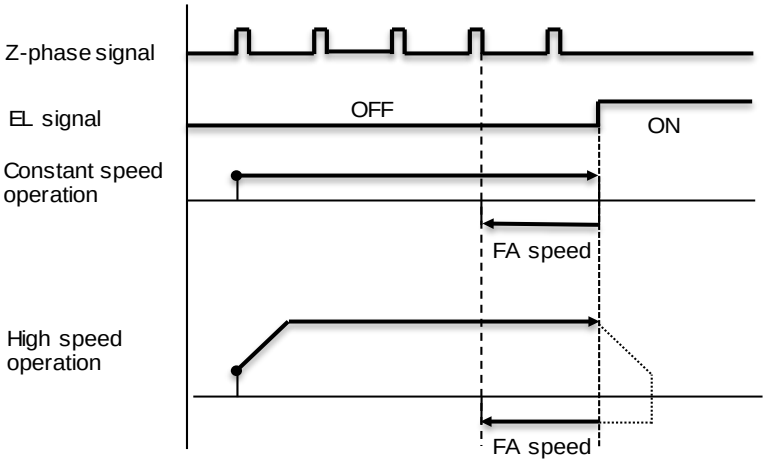


During a constant speed operation, it performs immediate stop when EL signal turns ON, and then starts in the opposite direction at FA speed. It performs immediate stop when EL signal turns OFF.

During a high-speed operation, when EL signal turns OFF, it performs immediate stop if EL signal is set as "immediate stop". It performs deceleration stop if the signal is set as “deceleration stop”. Then, it automatically starts in the opposite direction at FA speed, and performs immediate stop when EL signal turns OFF.
In this method, origin return can be performed without using neither ORG signal nor SD signal.

9-2-4-8. Origin return operation 7

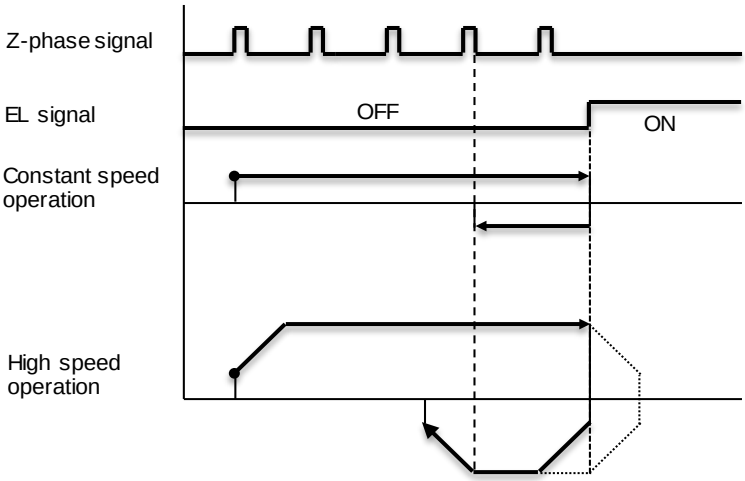
PCL60xx G9103C



During a constant speed operation, it performs immediate stop when EL signal turns ON, and then starts in the opposite direction at FA speed. Z-phase signal counting starts when EL signal turns OFF, and immediately stops when Z-phase signal is counted up. During a high-speed operation, when EL signal turns ON, it performs "immediate stop" if EL signal is set as “immediate stop”, and performs deceleration stop If set as "deceleration stop". Then it automatically starts in the opposite direction at FA speed, and when EL signal turns OFF, Z-phase signal counting starts and immediately stops when Z-phase signal is counted up. In this method, origin return can be performed without using neither ORG signal nor SD signal.

9-2-4-9. Origin return operation 8

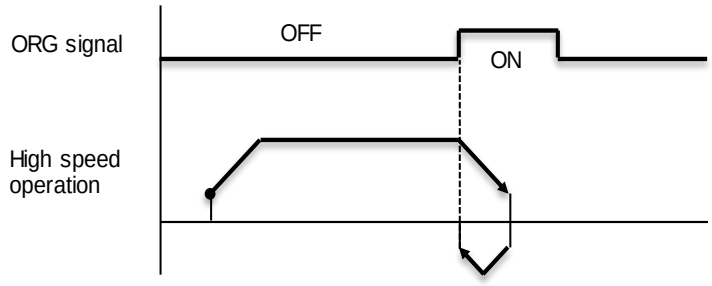
PCL60xx | G9103C



During a constant speed operation, it performs immediate stop when EL signal turns ON, and Z-phase signal counting starts when EL signal turns OFF. It performs immediate stop when Z-phase signal is counted up.
During a high-speed operation, when EL signal turns ON, it performs "immediate stop" if EL signal is set as “immediate stop”, and it performs deceleration stop if it is set as "deceleration stop". Then it automatically starts in the reverse direction while accelerating from FL to FH speed. When EL signal turns OFF, Z-phase signal counting starts and performs deceleration stop when Z-phase signal is counted up.
In this method, origin return can be performed without using neither ORG signal nor SD signal.

9-2-4-10. Origin return operation 9

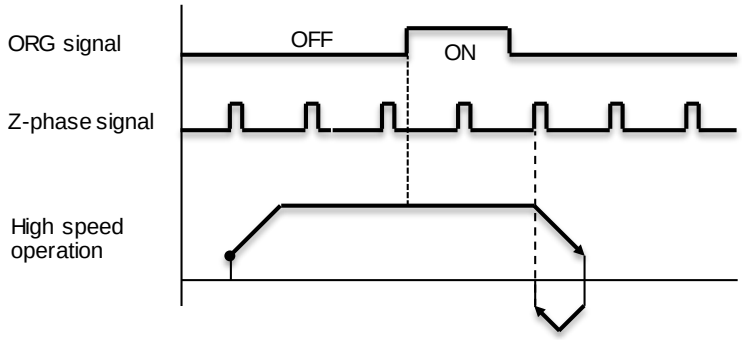
PCL60xx | G9103C



When origin return operation 0 is performed, it keeps operating until the specified counter becomes 0.

9-2-4-11. Origin return operation 10

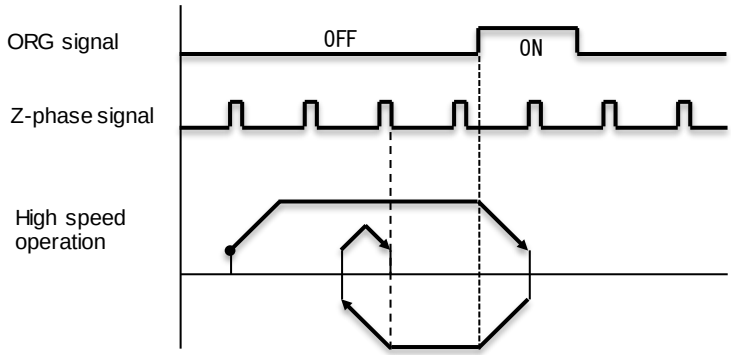
PCL60xx | G9103C



When origin return operation 3 is performed, it keeps operating until the specified counter reaches 0.

9-2-4-12. Origin return operation 11

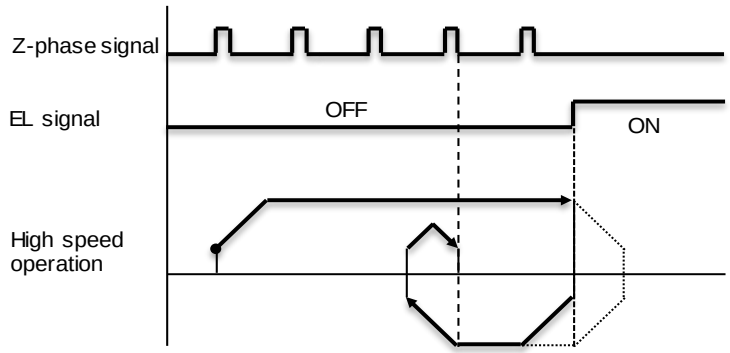
PCL60xx | G9103C



When origin return operation 5 is performed, it keeps operating until the specified counter becomes 0.

9-2-4-13. Origin return operation 12

PCL60xx | G9103C



When origin return operation 8 is performed, it keeps operating until the specified counter reaches 0.

9-2-4-14. Origin return with feeding amount limit

PCD2112A

This function stops the operation when reaching at the pre-register setting in prior to completion of origin return.
When an ORG sensor is disconnected or fails, an ORG signal may not be input to PCL in order to stop the operation.
So, this is the safety measure to stop the operation.
This function is implemented in PCD2112A, but the same function can be performed by other models as follows:

- PCD46x1A series
  It is activated by setting the bit of ORG enabled and positioning enabled in the control mode command.
- PCL60xx/PCL61x5 series
  The similar function can be performed by using software limit function.

9-2-4-15. Origin escape operation

PCD2112A | PCL60xx | G9103C

Even if ORG signal is ON, it is not always at the origin position. So it operates and stops until ORG signal turns OFF once. If there is a width while ORG signal is ON, and it is not exactly at the origin position, it can be used to run (= escape) until ORG signal OFF and return to the origin again.

9-2-4-16. Origin search operation

PCL60xx | G9103C

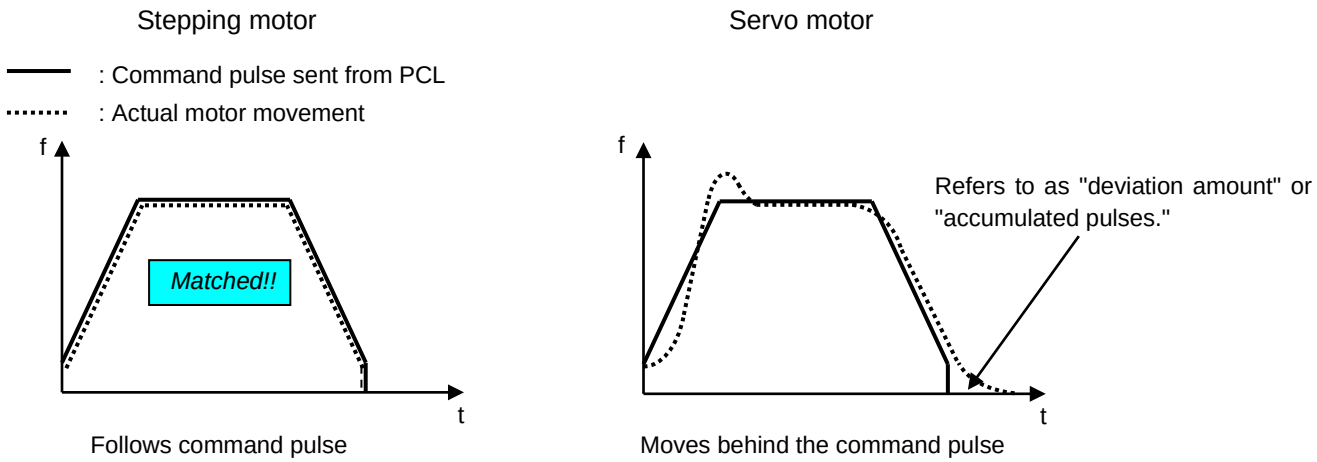
Origin search, for example, is an operation to move in-between (-)EL sensor and (+)EL sensor, and finally return to origin position from the specified direction. Internally, it is a combinations of several origin return operations and origin escape operations, and it is operated continuously automatically.

9-2-5. Servo motor interface

9-2-5-1. Servo motor control overview

Please skip this section if you have already known it.

The following figure shows the difference in operation between a stepping motor and a servo motor in response to a command pulse:



As shown, a stepping motor is synchronized (“follows” in more precisely) to the command pulse signal from a pulse generator such as PCL. However, a servo motor moves with a delay. So, when PCL finishes outputting the command pulse for the set feeding amount, the encoder does not return the pulses for that amount, and the motor is still running.

The difference is called "deviation amount" or "accumulated pulses", and the counter that counts the difference is called "deviation counter."

The deviation counter is generally built into a servo driver, and is controlled by the servo driver so that it stops when the value reaches 0.

Now, the following describes the timing when the PCL’s operation is completed and the timing when the motor stops.

“Operation completion” of PCL refers to either one of the following cases.

- When BSY signal changes from L level to H level
- When Stop flag of an event interrupt becomes 1
- When Operation stop flag is cleared to 0 in Status.

For example, if you want to start Y-axis after checking if the operation of X-axis is completed, use one of the above methods to check if the operation of X-axis is completed by CPU or hardware before starting Y-axis .

In a positioning operation, command pulse stopping equals an operation completion for a stepping motor. However, a servo motor is still running when the command pulse stops. So starting Y-axis when command pulse stops may cause a system failure. So Y-axis must start after the X-axis motor stops.

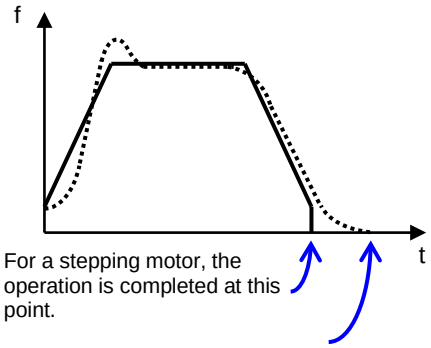
There are three types of servo motor control signals as follows:

- In-position (INP) signal
- Deviation counter clear (ERC) signal
- Alarm (ALM) signal

These are collectively referred to as "servo motor interface ."

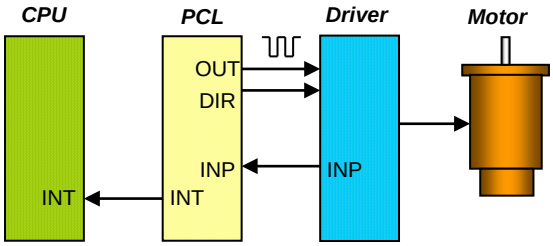
9-2-5-2. In-position (INP) signal

PCD2112A | PCL61x5 | PCL60xx | G9103C



For a servo motor, the operation is completed when the deviation amount of pulses have been output.

This signal is output from a servo motor driver when the deviation value reaches 0 during a positioning operation.
When in-position (INP) signal input is set to "enabled" in PCD/PCL/G sides, the operation completion timing can be delayed until this signal is input to PCL instead of operation completion by command pulse stops.
However, when the motor stops after EL signal or ALM signal turns ON, or when origin return is completed, the operation completion timing by INP signal is not delayed.
INP signal is not used for a stepping motor. Please set INP signal to "disable".

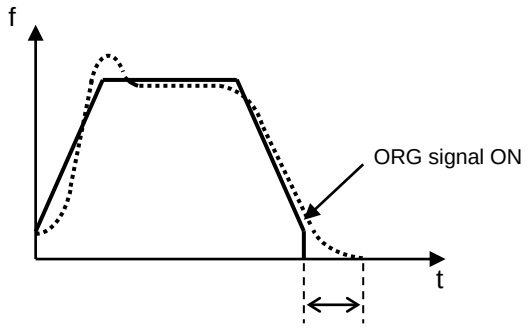


The operation of PCL is completed when an INP signal is input from the servo driver.

Stop interrupt (INT) signal to a CPU is also output when the operation is completed by the input of an INP signal.

9-2-5-3. Deviation counter clear (ERC) signal

PCD2112A | PCL61x5 | PCL60xx | G9103C

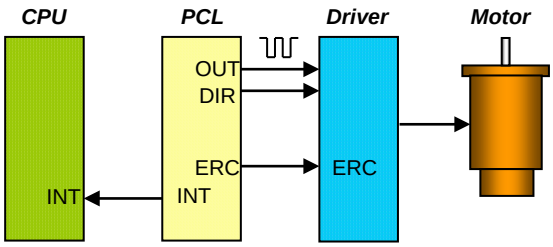


For example, even if the motor is stopped by turning ON the ORG signal in an origin return operation, the motor will move for the pulse amount of deviation.

Outputs a one-shot signal to clear the built-in deviation counter in a servo driver from PCL.
While INP signal "delays" the completion of PCL operation until the motor stops, the deviation counter clear (ERC) signal is used to forcibly stop the motor even if the deviation pulse amount still remains when the command pulse stops.

In the case of a positioning operation, the motor cannot be positioned unless it is moved until it is equal to the command pulse count. However, in some cases, the motor has to "immediately" stop when the command pulse stops.

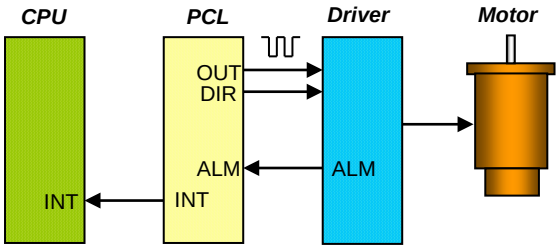
Specifically, this is when an immediate stop or an emergency-stop command is written, EL signal, ORG signal, or ALM signals is turned ON. In particular, if an emergency stop occurs or EL signal turns ON when an error occurs, it is necessary to stop the operation immediately even if the deviation pulse amount remains. Also, in the origin return operation, if the deviation remains, it will move further than the origin position, and the origin cannot be accurately determined.



PCL outputs an ERC signal to the servo driver when the command pulse stops by inputting the immediate stop/emergency stop commands or EL/ORG/ALM signals.

To stop the operation immediately even with the deviation pulse amount remains, deviation counter needs be 0, i.e., to clear the deviation counter. This is a ERC signal.
Interrupt (INT) signal is output to the CPU to inform that “the motor has stopped.”
You can confirm the cause of the stop by reading the status register or the interrupt factor register.

9-2-5-4. Alarm (ALM) signal PCL61x5 PCL60xx G9103C



When an ALM signal is input from a servo driver, PCL outputs an interrupt signal to CPU regardless of whether it is operating or stopped.
If the ALM signal is not turned OFF, it will never start even if start command is written.

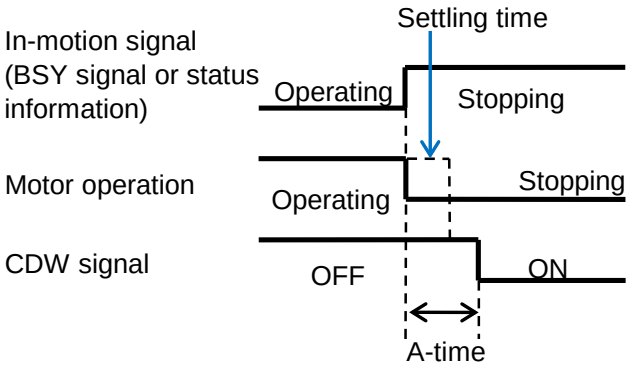
Alarm (ALM) signal is output from a servo driver as the same as INP signal.
It is output when an error occurs in the servo driver or the servo motor. For example:

- Excessive deviation, e.g. due to excessive heavy loads.
- When overcurrent flows to the motor (instantaneous & a certain duration of time)
- Temperature error
- Abnormal power supply voltage

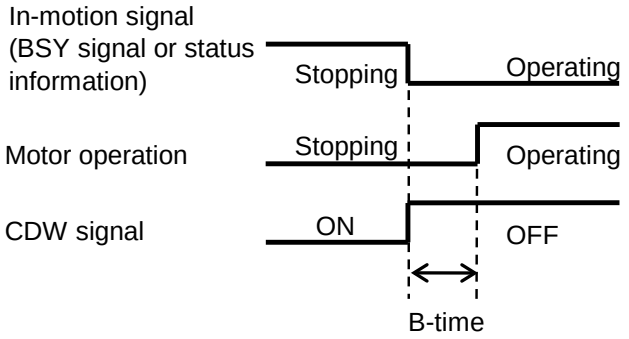
Generally, the ALM signal output from the servo driver remains ON until it is cleared.
For safety reasons, PCL cannot start even if a start command is written before the ALM factor in the servo driver is cleared and the ALM signal is turned OFF (= the error has eliminated). (There is no release method in PCL side.)
If a start command is written while the ALM signal is ON, only an interrupt (INT) signal is output without doing any operation.
The input logic of ALM signal can be set.

※ PCD2112A has an INP signal input terminal and an ERC signal output terminal, but an ALM signal cannot be connected (no terminal for ALM signal).

9-2-6. Stepping motor interface
9-2-6-1. Current-down (CDW) signal PCD2112A G9103C



Current down (CDW) signals can be output from ERC/CDW terminals to a stepping motor driver by setting an environment setting register. (Since ERC signal and ERC signal share the terminal, ERC signal is output for servo motors)
As for the time from the completion of current operation to CDW signal turns ON (A-time in the left figure), and the time from when CDW signal turns OFF until the next operation starts (B-time in the left figure), you can select from 8 types each(G9103C has only 4 types of A-time).



・ A-Time A: Current-down timer
This is the time from when an operating signal (BSY signal or status information) changes from “active” to “stopping” until CDW signal turns ON.
Specifically, it is set as a reference time from the completion of command pulse output (timing when the operating signal indicates “stopping”) to the end of the settling time of the stepping motor (the time from rotor vibration to damping to stop).
The purpose of this is to maintain the drive current (= torque)

・ B-time: Current up timer
This is the time from when CDW signal is ON (= during current down) until a motor starts operating.
Specifically, the time until the motor starts operating from the operating signal changes from “stopping” to “operating” (= the timing of a start command) is set as a reference time.
The purpose of this is to prevent “out of stop” by increasing the current from the current down state to the drive current before the first command pulse is input to the stepping motor driver.

* If this control is performed on a model other than PCD2112A or G9103C, you can use one of the general-purpose output terminals, but A-time and B-time are controlled by CPU.

9-2-6-2. PH1 to PH4 signals (Excitation sequence signal for 2-phase stepping motor)

PCD46x1A | PCD2112A | G9103C

2-phase stepping motor has unipolar drive or bipolar drive as the drive method, and 2-2 phase excitation or 1-2 phase excitation as the excitation method.
Since these four combinations of sequence signals can be output from PH1 to PH4 terminals, a 2-phase stepping motor driver circuit can be constructed simply by connecting a power FET at the next stage.

- Unipolar drive 2-2 phase excitation (full step)
- Unipolar drive 1-2 phase excitation (half step)
- Bipolar drive 2-2 phase excitation (full step)
- Bipolar drive 1-2 phase excitation (half step)

Refer to the user's manual for each model for the terminal status of PH1 to PH4 versus Step.

* Sequence for micro-stepping is not supported.

9-2-7. Encoder input PCD2112A | PCL61x5 | PCL60xx | G9103C

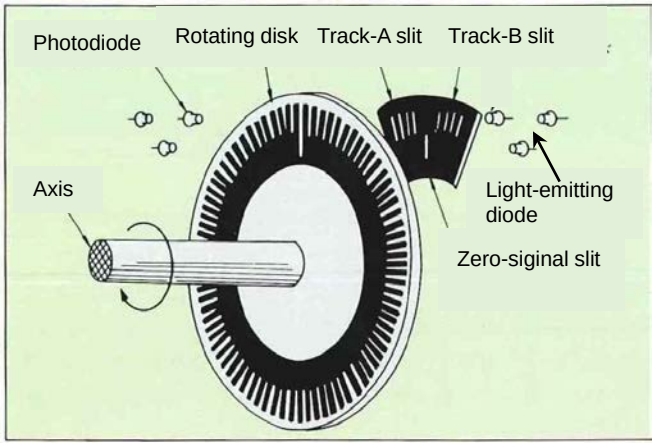
9-2-7-1. Introduction of encoders

This section briefly describes the encoder types and the output signals. Please skip reading this section if you have already known it.
Encoder is a type of pulse generators, and it is a position detector to know the current position. There are two types of encoders: "incremental type" and "absolute type".
Generally, “a rotary-type encoder” is attached to the rear side of a rotary motor, but recently, “linear-type encoders”, which are aligned parallel to the side of linear motors or linear stages, are also widely used.
We will introduce rotary types as the examples as follows:

(1) Incremental type

・ Features

The pulse corresponding to a rotation angle is output only when the disk is rotating, and is not output when the disk is stopped. Therefore, it is necessary to count the output pulses with a counter, and the rotation amount is detected by the count number.
In order to know the revolution amount from a given axis position, the pulse signals cannot be distinguished individually by this method. Therefore, the number of pulses from that position must be cumulatively added and subtracted by counters.
One of the features of this encoder is that the rising and falling edges of waveforms can be used to increase the frequency of pulse generation by a factor of two or four, thereby electrically increasing resolution. (This is referred to as 2-times or 4-times multiplication.)



・ Structure

As seen from the side of the encoder, the light from the light-emitting diode (light-emitting element) passes through the slits (small gaps) ⇒ disk, and reaches the photodiode (light-receiving element).
When the disk rotates and the black pattern blocks the light from the slit, the photodiode does not operate, and it operates when the light arrives.
In the case of an incremental type, the number of rotations can be determined by counting the number of operations.

This incremental encoder is compatible with the encoder inputs of PCL series.

.

(2) Absolute type

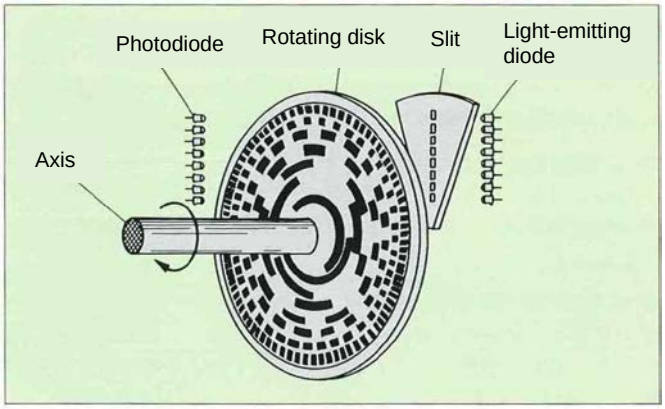
・ Features

With this method, absolute position signals corresponding to the rotation angle are output in parallel in a code, even when the disc is stopped.

Therefore, no counters are required, and the rotation position can always be checked.

When the encoder is installed in a machine, the zero position of the input rotation axis is fixed, and the rotation angle with the zero position as the coordinate origin is constantly digitally output.

Even when the power is turned off, and is tuned ON again, the correct rotation angle can be indicated.



・ Structure

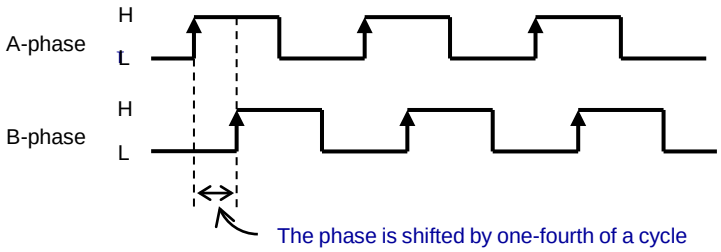
As shown in the figure, the structure of an absolute type is the same as that of the incremental type, but the difference is that the pattern of the rotating disk is a combination-coded pattern.

9-2-7-2. A-, B- and Z-phase signals

In the incremental encoder diagram in the previous section, the signal passing through "track A slit" is output as an A-phase signal, the signal passing through "track B slit" is output as a B-phase signal, and the signal passing through "zero signal slit" is output as a Z-phase signal.

The A-, B-, and Z-phase signals of the incremental encoder are explained as follows:

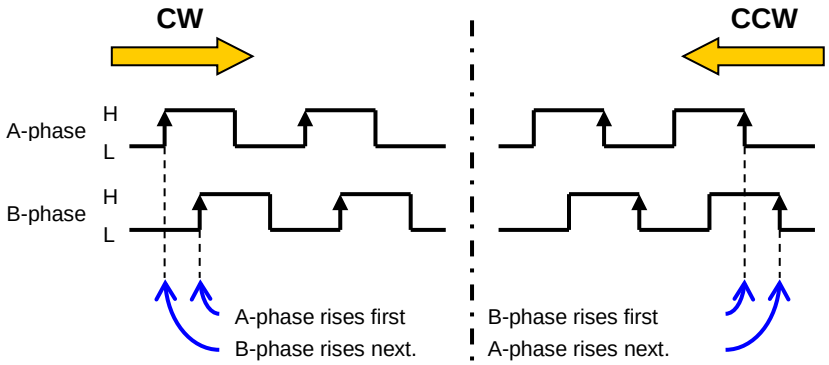
・ A-phase and B-phase signals



It is commonly referred to as an "A-phase signal, B-phase signal" or "A/B-phase signal."

When drawing with a square wave diagram, the pulse signals of two same cycles (one cycle = 360 °) are shifted by 1/4 cycle, so it is also called “90° phase difference signal”.

As a result, the rotation will be determined as if A-phase signal rises before B-phase signal, it is “CW direction”, and if it is reversed and B-phase signal rises first, it is “CCW direction.



・ Z-phase signal

Z-phase signal is output only once per revolution. (= like the reference position of an encoder), and the signal is used as the formal origin position for a high-precision origin return operation.

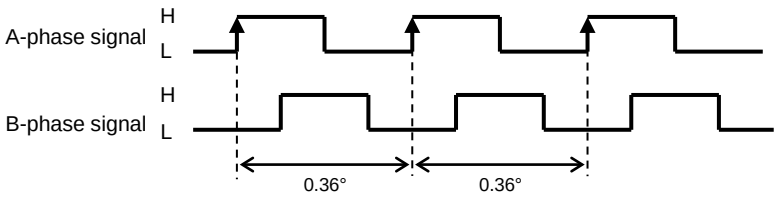
Since an ORG sensor has some error in the position to stop, Z-phase signal is combined to use by counting several times (the number of counting is set in advance) while the ORG sensor is ON, to determine the more precise origin position during

high precision origin return operations.
These A-, B-, and Z-phase signals are input to PCL. (Corresponding terminal names of PCL are EA, EB, EZ)
Either A/B-phase signal (90° phase difference signal) or 2-pulse [(+) direction pulse or (-) direction pulse] can be selected as the input type.

9-2-7-3. What is multiplication (x) ?
One cycle of a A-phase or B-phase signal is the basic resolution of an encoder, and the resolution of an encoder can be finely set by the reading methods of the rising and the falling edges of each signal.
For example, an encoder of 1000 pulses per revolution (=1000 ppr) can be used as:

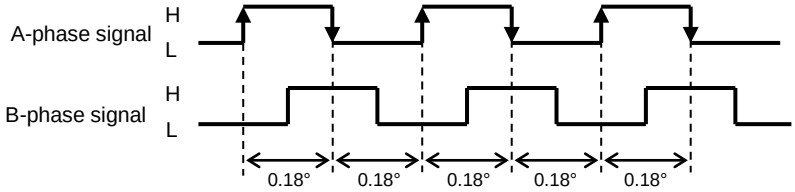
- 1 X (basic) : 1000 ppr = 0.36°/pulse
- 2 X : 2000 ppr = 0.18°/pulse
- 4 X : 4000 ppr = 0.09°/pulse.
  (The multiplication is set by the driver.)

- 1 x



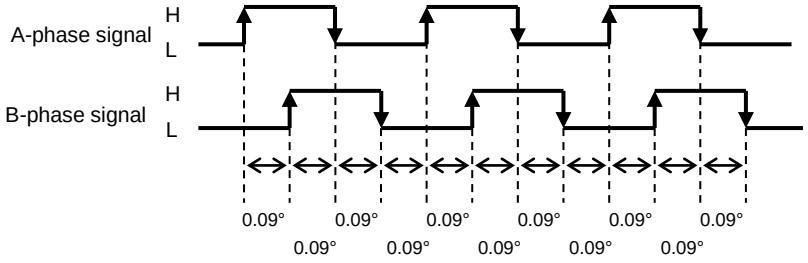
Counts at the rising edge of A-phase signal, and the direction is determined by which of A-phase or B- phase signal rises first.

- 2 x



Counts at the rising and falling edges of A-phase and B-phase signals. The direction is determined by which of A-phase or B-phase signal rises first.

- 4 x



Counted at the rising edges and the falling edges of A-phase signals and B-phase signals. The direction is determined by which of A-phase or B-phase signal rises first.

9-2-8. Up/down counter PCD46x1A PCD2112A PCL61x5 PCL60xx G9103C
This counter can be used for current position management. Output pulse (Command pulse), Encoder A/B-phase, and pulser A/B-phases signal, etc., can be counted. (Counts up in (+) direction and count down in (-) direction.)
This counter;

- can be disabled or can be stopped counting on the way.
- can count 2-pulse type signal [pulse for (+) direction and pulse for (-) direction] additionally.

The outlines of counter functions of each series is as follows.

- PCD46x1A series : 1 per axis (dedicated to count the output pulses)
- PCD2112A : 1 (counts the output pulse and encoder/pulser signals)
- PCL61x5 series : 2 per axis (typically used for counting the output pulses and encoder/pulser signals)
- PCL60xx series : 4 per axis (counts the output pulse and encoder/pulser signals, the deviation count, the general-purpose count such as external scale signals, etc.)
- G9103C: 3 (counts the output pulses and encoder (pulser) signals, the deviation count)

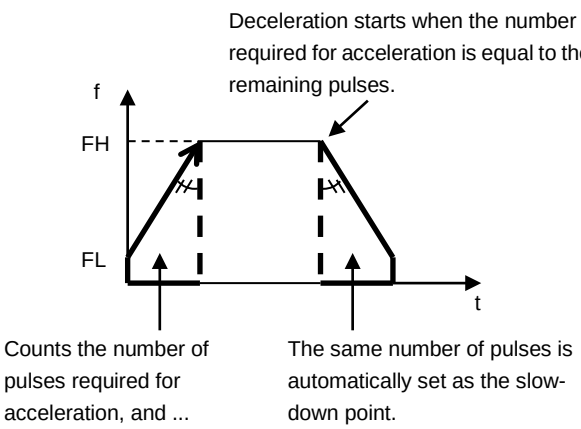
9-2-9. Slow-down point auto setting function

PCD46x1A | PCD2112A | PCL61x5 | PCL60xx | G9103C

Slow-down point refers to the deceleration start point in a positioning operation with acceleration/deceleration. Specifically, it refers to "the number of residual pulses when starting deceleration."

When this auto setting function is enabled, the number of pulses required for acceleration or the number of pulses in the numerical operation result is automatically set to the slow-down-down point setting register.

All PCL models have this function, so that it is not necessary to write data to the slow-down point register every time.



In PCD46x1A, PCD2112A, and PCL61x5 series, the number of pulses required for acceleration (the number of pulses from FL speed to FH speed) has been counted by the built-in dedicated counters, and when the residual number of pulses becomes equal to the number of pulses for acceleration, the deceleration is automatically started with the number of pulses as slow-down point. (Please note that the condition is "acceleration time = deceleration time".)

PCL60xx series and G9103C are designed to calculate slow-down point automatically. If the condition is

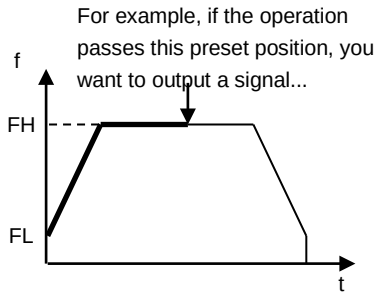
(deceleration time) ≦ (acceleration time × 2),

the acceleration time and deceleration time can be set automatically even in patterns where the acceleration time and deceleration time are different.

9-2-10. Comparator

PCL61x5 | PCL60xx | G9103C

The comparators included in PCL series are all comparison circuits, which are used to compare the comparator register (RCMP) with the counter (RCUN).



CP terminal can be turned ON or OFF, or an interrupt signal can be output to CPU when RCUN are compared with the values set in advance in RCMP, and the comparison condition set in an environment setting register is fulfilled.

Stepping motor out-of-step detection function or software limit function, which is described later, can also be performed by applying the comparator function.

The content of comparator functions differs depending on the model.

- PCL61x5 : 4 per axis
  Comparator 1 can be compared with counter 1 (CUN1); Comparator 2 can be compared with Counter 2 (RCUN2). (Comparison counter is fixed.)
  CP1 terminal turns ON or OFF when the Comparator 1 condition is fulfilled. CP2 terminal turns ON or OFF when the Comparator 2 condition is fulfilled.
  Comparator 3 and 4 are dedicated to the software limit function, which is described later.
- PCL60xx series : 5 per axis
- G9103C : 3

As mentioned above, the comparator function of PCL61x5 is rather simple; comparison counter is fixed, CP terminal turns ON/OFF and interrupt signal is output when the condition is fulfilled. However, the comparator functions of PCL60xx series and G9103C are considerably multi-functional.

- Processing when conditions are fulfilled
  - Immediate stop or deceleration stop of the axis
  - Change the operation data to the pre-register value (e.g. change the speed when a condition is fulfilled).
- Comparison counter
  - This is not fixed, and can be combined freely, such as Comparator 2 and Counter3 or Comparator 3 and Counter1.

9-2-11. Pre-register (pre-buffer for the next operation)

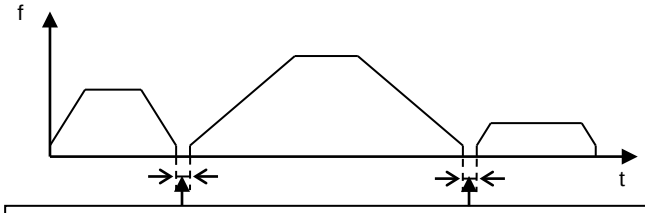
PCL61x5 PCL60xx G9103C

The pre-register is a "pre-stage register for the following operation," which looks like a waiting room in which register data for the following velocity patterns are stored in advance. The pre-register is existed in registers to determine the speed patterns [feeding amount (target position), FL-speed, FH-speed and acceleration/deceleration rate, etc.] and in Start command.

By setting the next operation data to the pre-register and writing the next start command data during current operation, the following operation can automatically start, and the different operation patterns can be continuously performed without interval.

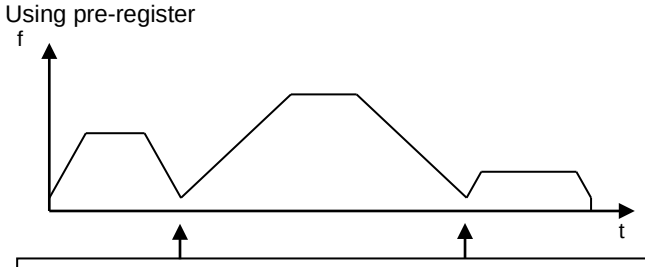
PCL61x5 series and G9103C have one pre-register stage. PCL60xx series has two stages, so that you can draw by connecting fine pseudo-curves.

Without pre-register



There are slight stopping times to confirm if the previous operation has stopped, and each register value and command for the next operation are written.

Using pre-register



The pre-register can be used to prepare the pattern data for the next operation during the current operation, so that the stopping times between the operations can be eliminated.

9-2-12. Pulser input

9-2-12-1. What is pulser ?

A pulser is a rotary encoder equipped with a manual wheel on its axis. It is commonly seen in processing machines and machine tools.

The wheel has a scale, and it looks like a dial-type "safe" key. It ticks and turns (typically moves (+) direction in CW rotation).

There is an operation panel in which the operator drives the motor by pressing (+) direction or (-) direction button to adjust the position while actually looking at the workpiece on the stage or slightly shifting the position. However, in the case of fine position tuning, pulser seems to be better for workability.

9-2-12-2. Pulser input PCD2112A PCL61x5 PCL60xx G9103C

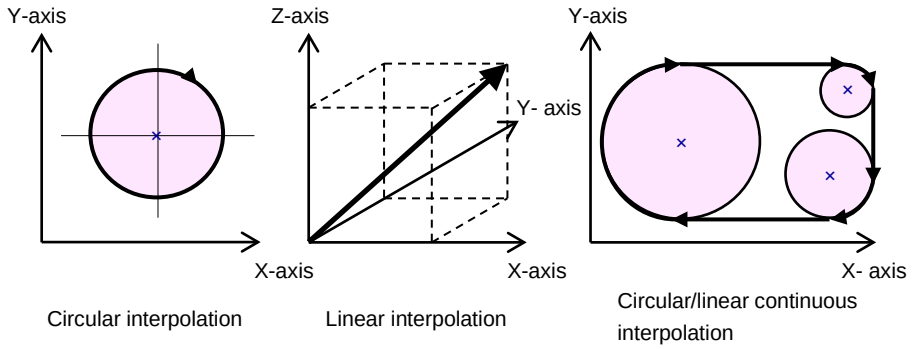
Receiving signals from a pulser, command pulse corresponding to the rotational speed is transmitted to a motor driver. Since a motor moves as much as the wheel is turned, it is also possible to control the current position by operating the up/down counter. PA/PB terminals are provided for the pulser signal inputs.

Either A/B-phase signal (90° phase difference signal) or 2-pulse, (+) pulse and (-) pulse, can be selected as the input format. Similar to encoder inputs, you can select 1, 2, or 4 x multiplication for A/B-phase input.

For stepping motors, turning the wheel too fast may cause “out-of-step” issue, so you can restrict the high speed (output frequency) operations.

Also PCL60xx series and G9103C can output a command pulse that is a constant multiple of the input signal from a pulser by using 32-multiplication and 2048-division function.

9-2-13. Interpolation function



Circular interpolation | Linear interpolation | Circular/linear continuous interpolation

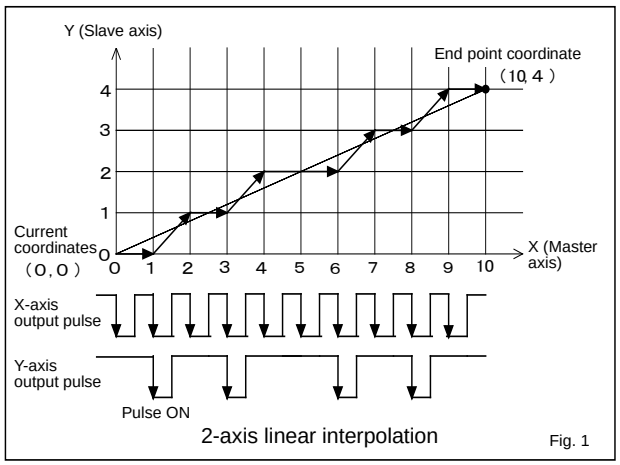
PCL61x5 series can perform linear interpolations by any number of axis. Linear interpolation can also be performed by using more than one LSI.

In addition to linear interpolation, PCL60xx series and G9103C can perform circular interpolations between any two axis.

Since these models also have built-in pre-registers, it can perform "continuous interpolation" in which linear interpolation or circular interpolation is continuously performed as a continuous operation.

9-2-13-1. Linear interpolation

PCL61x5 | PCL60xx | G9103C



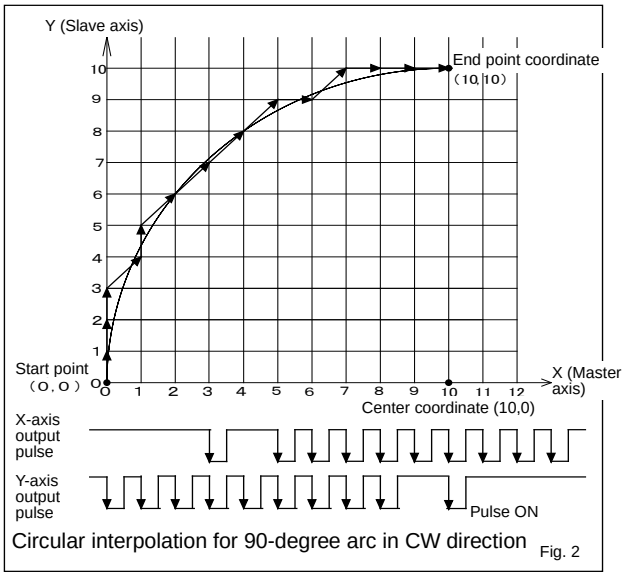
A linear interpolation is performed by setting the end point coordinate and linear interpolation movement to the current position. The end point position is relative to the current position, and the number of output pulses is set to each axis.

The axis of the largest feeding amount is automatically recognized as the master axis, and the other axis becomes the slave axis. The master axis always outputs pulses, and the slave axis outputs intermittent pulses of the master axis according to the interpolation calculation result.

The figure on the left shows a 2-axis linear interpolation in which X-axis and Y-axis are used, and the end point coordinate is set to (10, 4).

9-2-13-2. Circular interpolation

PCL60xx | G9103C



Circular interpolation is performed by setting the current position as the start point (coordinate position 0,0), setting the center coordinates and end point coordinates of the arc relative to the start point, and setting the CW or CCW circular interpolation movement.

Specify the coordinates of the center and end coordinates relative to the current position (start point).

CW circular interpolation draws an arc in the clockwise direction from the current coordinate to the end point coordinate, centered on the center coordinate, and CCW circular interpolation in the counterclockwise direction.

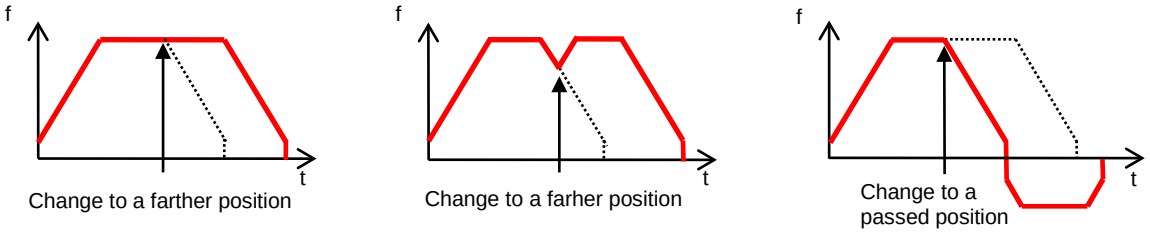
The figure on the left shows an example of CW circular interpolation with a 90° arc drawn on the X and Y axes.

9-2-14. Override the target position

PCL61x5 | PCL60xx | G9103C

The target position (feeding amount) can be changed during a positioning operation.

If the new position has already passed, the motor rotates in the reverse direction after deceleration stop (immediate stop in a constant speed operation) is made. It can also be stopped by outputting the set number of pulses by an external signal input timing.



9-2-15. Simultaneous start/simultaneous stop

PCD46x1A | PCD2112A | PCL61x5 | PCL60xx | G9103C

This function starts or stops more than one axis at the same time with 2-axis or more LSIs.

When you want to start more than one axis at the same time, a time lag should occur if start command is simply written to the operation axis sequentially.

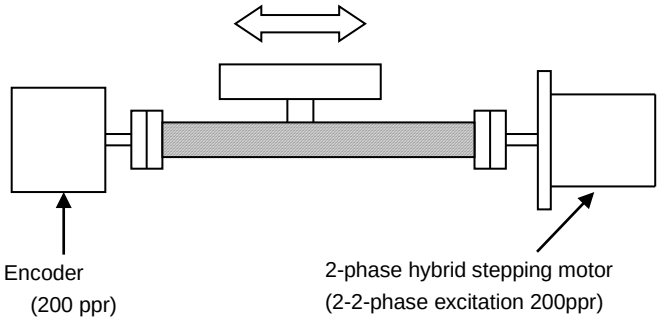
The simultaneous start function is not performed by writing immediate start commands. Instead, it is a function to hold the start of all axes and start them simultaneously by a simultaneous start command or an external input signal.

The simultaneous stop function stops all axes at once in the same way.

The setting method slightly differs depending on the model.

9-2-16. Stepping motor out-of-step detection function

PCL60xx | G9103C



It detects an out-of-step state of a stepping motor.

To use this function, install an encoder with the same resolution as the stepping motor coaxially as shown in the picture.

In applications of the comparator function, the number of pulses output by PCL and the number of pulses returned from the encoder are compared, and when the number of pulses differs by more than the set value (= the condition of comparator is satisfied), it is regarded as "out-of-step".

Since this function is performed using a deviation counter and the comparator function, you can select the process “when the comparator condition is fulfilled.” (Immediate stop, deceleration stop, or interrupt signal outputs, etc.)

9-2-17. I/O port (General-purpose I/O terminal)

PCD46x1A | PCD2112A | PCL61x5 | PCL60xx | G9103C

Depending on the setting, the terminals can be used as general-purpose inputs or outputs.

For example, it can be used for an excitation ON/OFF signal or a current down signal of a stepping motor driver if it is set to output.

In PCD46x1A and PCL61x5 series, D0 to D7 (D15) for parallel interface can be used as general-purpose I/O terminals when serial-bus interface with CPU is selected. .

The number of I/O points differs depending on the model.

9-2-18. Ring counter function

PCL61x5 | PCL60xx | G9103C

For an application of the comparator function, a counter can be set to a ring counter to control a rotary table.

If any comparator is set to a ring counter, the counter returns to 0 after counting up to the comparator value.

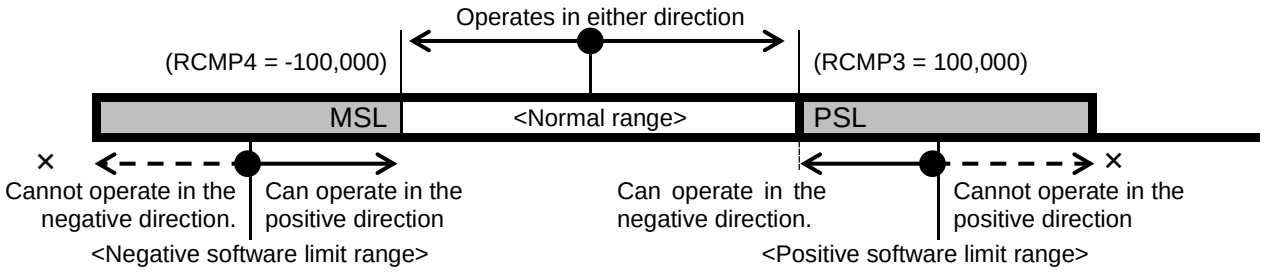
Conversely, if the counter counts down when it is 0, it becomes the comparator value and then start counting down.

9-2-19. Software limit function PCL61x5 PCL60xx G9103C

Software limit can be configured using the Comparator 2 circuitry.

Software limit is not a hardware end limit using an EL sensor, but sets the both end values using two comparators. Basically, it operates within the "range" of two comparator data, so it can be used like a software end limit.

"Software limit" causes an immediate stop or a deceleration stop when the operation enters into the software limit range (a comparable condition is fulfilled), the operation cannot move further in the same direction, but can move in the opposite direction.



For setting methods or the detail information of this function, please refer to the user's manual.

<At the end>

Most of the current PCL series have upgraded their base models (initial versions), and the functions described here have been built-in from the base models to the current models. They have had a lot of sales results, and will be kept selling for a long period of time from now on.

In addition, all models, of course, are compatible with RoHS2, so you do not have to warry about RoHS issue. .

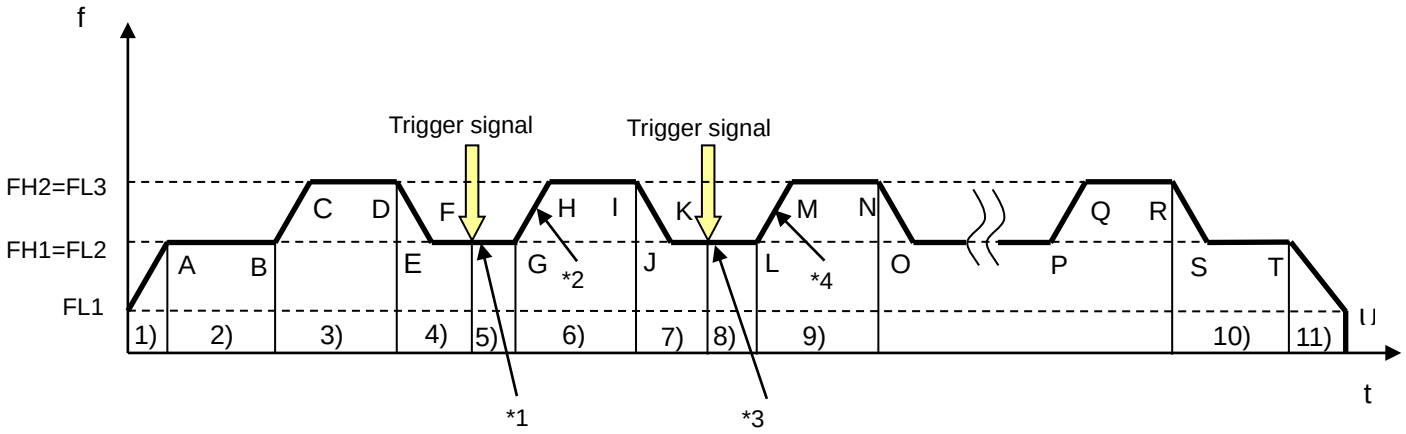
It would be appreciated if you would check on NPM’s pulse control LSIs when you have a hard time with one-chip microcomputers, FPGAs, etc. in the field of motion controls.

Reference

<Appendix> Continuous operation in PCL61x5

Refer to the following three examples below:

- Continuous operation with a pre-register
- Override 1 in the target position
- Override 2 in the target position



1. Outline of the operation

- It appears that the speed is changed several times during an operation, however, a positioning operation is performed
  in each block (1 to 11), and they are all connected and operated continuously.
- For each point (point A to point U), it is necessary to operate without a break.
- For 4 to 5 and 7 to 8, the target positions are changed during the operations.
  (Trigger signals are input during the operations, so that positioning operations to Point G (and Point L) are performed.)
- For operations in 6 and 9, the target positions are changed during the operations.

2. Advance preparation

Input the trigger signals to PCS terminal of PCL.

3. Common settings for each operation

* As shown below, the specific bit of the register is expressed by "register name". "bit name".

(1) Operation mode (PRMD)

- Except for 11), they are all positioning operations. (PRMD.MOD = 41h)
- Accelerations and decelerations are shown all linear in the figure, but actually they all S-curves. (PRMD.MSMD=1).
- Each operation is completed when the cycle is completed. (PRMD.METM = 0)
- Slow-down point is always set to “manual”. (PRMD.MSDP= 1)
  (If it is set to auto (0), for example, the operation toward point B in the above figure at FH1 speed can start decelerating before reaching point B.)
- Except for 4) to 5) and 7) to 8) operations, set all other operations to PRMD.MPCS = "0".
- Set the sequence number to the each operation. (PRMD.MSN1 to 0)
- Set the other bits to 1 as needed.

(2) Set the required items in environment setting 1 to 3 (RENV1 to 3), event interrupt factor setting (RIRQ), RCUN1 to 2 (Counter 1 to 2), and RCMP1 to 2 (Comparator 1 to 2).

4. Details of operations

1) 2) Start ⇒ Point A ⇒ Point B

Accelerate from FL1 speed to FH1 speed, and operate toward Point B at FH1 speed.

- Write the following data to pre-register before starting the next program.
  PRMV (number of pulse from start to point B)
  PRMD (operation mode)
  PRMG (speed magnification), PRFL(FL1 speed data), PRFH(FH1 speed data), PRUR (acceleration rate), PRDR (deceleration rate), PRUS (S-curve range during acceleration), PRDP (Slow-down point) = 0
  ⇒ If PRDP = 0, operation will be completed at FH1 speed without decelerating.
- The operation starts when the start command (53h) is written, and the operation completes when the number of pulses equal to PRMV register setting is output.

- During this operation, write the data of the next operation 3) to pre-register.
  PRMV (number of pulse from point B to point D)
  PRMD.MSN1 to 0 = 00 (sequence number)
  The other bits of PRMD are as shown in 3. Common settings for each operation, (1) in the previous page.
  PRFL(FL2 speed data = value of FH1 speed data), PRFH(FH2 speed data)
  *PRMG (speed magnification rate), PRUR (acceleration rate), PRDR (deceleration rate) and PRUS (s-curve acceleration range) are not required to be set if they are the same as 1) and 2).
  PRDP (slow-down point) is also need to stay at 0.
  Write the start command (53h) of the next operation 3).

3) Point B ⇒ Point C ⇒ Point D

Accelerate from FH1 speed (=FL2 speed) to FH2 speed, and operates to point D at FH2 speed.
During operations 1) and 2), the register data and start command for operation 3) have been written, so the operation 3) starts without stopping when operation 1) and 2) are completed.
This operation is completed when the number of pulses for operation 3) is all output.

- During this operation, the data of the following operation 4) and 5) should be written to the pre-register.
  PRMV (a large number of pulses from point D to point G for the time being)
  PRMD.MPCS=1 (positioning operation starts by PCS-signal input.)
  PRMD.MSN1 to 0 = 01 (Sequence number)
  The other bits of PRMD are as shown in 3. Common settings for each operation, (1) in the previous page.
  PRFL(FL3 speed data = value of FH2 speed data), PRFH(FH1 speed data)
  *PRMG (speed magnification rate), PRUR (acceleration rate), PRDR (deceleration rate), and PRUS (s-curve acceleration range) are not required to be set if they are the same as 1) and 2).
  PRDP (slow-down point) is also need to stay at 0.
  Write the start command (53h) of the following operations 4) + 5).

  * Since the operation in 4) is FL3 > FH1, it will perform a reverse acceleration (= deceleration) from FL3 speed to FH1 speed.

4) 5) Point D ⇒ Point E ⇒ Point F ⇒ Point G

As described above, it "decelerates" from FL3 speed to FH1 speed, and operates at FH1 speed.
The same as the previous section, operations 4) and 5) start without stopping when operation 3) is completed.
Immediately after a trigger signal (PCS signal) is input at point F, the number of pulses from point F to point G will be determined. ⇒ Positioning operation from PCS signal = ON (point F) to point G will be performed.

Since PRMD.MPCS=1 when starting from point D, the positioning control counter is stopped until a PCS signal is input. The positioning control counter starts when the PCS signal is input. (Override 2 of the target position)

If it is required to check whether or not the operation 4) and 5) are being performed, read the main status, MSTSW.SSC1 to 0, and check if they are 01.

PCL tries to output the number of pulses that have been previously written into PRMV [that is a large number of pulses from point D to point G for the time being written during the operation 3) is performed], so you need to change “RMV” to “the number of pulses determined for point F to point G” at the point of “*1”.

(*) RMV is not pre-register (PRMV) but is currently operating register.

This operation is completed when the pulse count of the changed RMV value has been output.

・ During this operation, the next operation 6) should be written to pre-register:

PRMV (A large number of pulses from point G to point I for the time being)

Return PRMD.MPCS to "0".

PRMD.MSN1 to 0 = 10 (Sequence no.)

The other bits of PRMD are as shown in 3. Common settings for each operation, (1) in the previous page.

PRFL(FL2 speed data = FH1 speed data), PRFH(FH2 speed data)

*If PRMG (speed magnification rate), PRUR (acceleration rate), PRDR (deceleration rate), and PRUS (s-curve acceleration range) are the same as 1) and 2), they are not required to be set.

PRDP (slow-down point) also remains at 0.

Write the start command (53h) of operation 6).

6) Point G ⇒ Point H ⇒ Point I

Same as the previous section, operation of 6) starts without stopping when the operations of 4) and 5) are completed. After starts, change "RMV" to "fixed number of pulses from point G to point I" at the point "*2".

(Target position override 1)

(*) RMV is not a pre-register (PRMV) but is currently operating register.

If it is required to check whether or not the operation 6) is being performed beforehand, read the main status MSTSW.SSC1 to 0, and check if they are 10.

This operation is completed when the number of pulses corresponding to the changed RMV value has been output.

・ During this operation, the data for the following operation 7) and 8) should be written to pre-register.

PRMV (a large number of pulses from point I ⇒ point L for the time being)

PRMD.MPCS = 1 (Positioning is started when PCS-signal is input.)

PRMD.MSN1 to 0 = 11 (Sequence no.)

The other bits of PRMD are as shown in 3. Common settings for each operation, (1) in the previous page.

PRFL(FL3 speed data = FH2 speed data), PRFH(FH1 speed data)

*If PRMG (speed magnification rate), PRUR (acceleration rate), PRDR (deceleration rate), and PRUS (s-curve acceleration range) are the same as 1) and 2), they are not required to be set.

PRDP (slow-down point) also remains at 0.

Write the start command (53h) of the following operation 7) and 8).

7) and 8) and the later are the same as 4), 5) and 6) above.

10) Point R ⇒ Point S ⇒ Point T

・When the operation of point P ⇒ point Q ⇒ point R starts, the data of the next operation 10) should be written to pre-register.

PRMV (the number of pulses from point R to point U or the number of pulses farther than point U)

PRMD is as shown in 3. Common settings for each operation (1). Operation mode (PRMD)

PRFL(FL3 speed data = FH2 speed data), PRFH(FH1 speed data)

*If PRMG (speed magnification rate), PRUR (acceleration rate), PRDR (deceleration rate), and PRUS (s-curve acceleration range) are the same as 1) and 2), they are not required to be set.

PRDP (slow-down point) also remains at 0.

Write the start command (53h) of the next operation 11).

11) Point T ⇒ Point U

If FL3 speed is changed to FL1 speed and a deceleration stop command (4Ah) is issued during operation 10), a deceleration starts by the command and the operation stops when it reaches at FL1 speed.

*For operations of 10) and 11), the feeding amount is not sufficient in the above example.

If you want to set the feeding amount from point R to point U, set as follows:

PRMV (the number of pulse from point R to point U)
PRFL = FL3 speed
PRFH = FH1 speed
PRDP = Calculate and set (the number of pulse from point T to point U)
Start command (53h)

This changes FL3 speed to FL1 speed during the operation 10) (from point R to point T).

* Reading statuses and interrupt factors that are not required in the above explanation are omitted.

December 6, 2019
Initial issue
No. 68S-018

Impress, not just satisfy
Nippon Pulse Motor Co., Ltd.