Technical Questions

PCL Series Chips

The PCL61xx series does not have an out-of-step detection function for stepper motors. Is there any good method for detecting this condition?

The PCL61xx series has simplified versions (economy versions) of the functions on the PCL6025/6045B. As you have noted, this series does not have an out-of-step detection function.
One method for detecting an out-of-step condition in the PCL61xx series is as follows:

1. Count the output pulses using the COUNTR1 function.
2. Count the encoder signal (EA/EB input) using the COUNTER2 function.
   <To check during operation>
3. During operation, latch COUNTER1 and COUNTER2 using the latch commands. Then check for a difference using an application program. (If you want to check it with an interrupt, you should check at certain intervals using the comparator.)
   <To check when stopped>
4. When stopped, check the values of COUNTER1 and COUNTER2. See if there is any problem with the difference.

As said above, there is no method for evaluating an out-of-step condition except by comparing the output pulses and the feedback pulses.

The motor will turn in a CW direction, but it does not turn in a CCW direction. What are some possible causes for this?

Check the following:

1. Is an EL signal being input?
2. Is specification of the direction appropriate?
3. Is the selection of output method (2-pulse mode or common pulse/direction signal mode) appropriate for the driver?

We want to set the acceleration and deceleration rates independently. Which series can handle this?

All PCL models have separate acceleration and deceleration rate registers, which can be set independently.

We want to calculate the corrected FH speed and total feed time using S-curve acceleration/deceleration, when we turn ON the FH correction function (automatic operation speed correction function) to make the PCL provide smooth acceleration/deceleration by restricting the amount of triangle shape driving.

The purpose of the "FH correction function" that is integrated into the PCL series is to ease the software design burden on users and to smooth the top of the acceleration/deceleration curve. It is not designed to calculate the precisely corrected FH speed or total operating time.

If you need to obtain a precise FH speed and total operation time, we recommend that you turn OFF the FH correction function and calculate the FH speed (top of the triangle shape) and the total operation time using the formulas described in each user's manual.

The calculation formulas for the triangle operation pattern for each model are included in an Excel file (to calculate the register setting values) on each product page.

Please note that even when each register value for a speed pattern, such as the feed amount (preset amount), FL speed, FH speed, acceleration/deceleration rate, S-curve range, and ramp-down point are the same, the corrected FH speed and total operation time will be different when the FH correction function is enabled. (Unfortunately, if the FH correction function is enabled, the total operation time must actually be measured.)

When the FH correction function is enabled, and all the register values for the speed pattern are kept the same, the corrected FH speed and total feed time will also produce the same results.

Can we write data to the PCD/PCL even when the reference clock is stopped?

The PCD/PCL cannot read or write data when the reference clock is stopped.

We are planning to connect an LSI in the pulse control PCL series to an H8 series CPU using 8-bit bus mode. The manual says that we should use a 16-bit bus with the PCL6045B. Is it possible to connect it to an 8-bit bus such as the Z80 family? Is an 8-bit bus is available for other chips in the PCL series?

It is possible to connect an 8-bit bus such as a Z80 family CPU to the PCL6045B. However, when the Z80 family interface is selected, the PCL6045B employs little-endian addressing so that you have to be carefully when accessing the bus with words and long words.

The other chips in the PCL series can be connected to an 8-bit bus.

How should unused terminals be handled?

Unused terminals should be handled as follows.

1. Input-group active-low terminals (with a bar over the terminal name) should be connected to a 5V terminal.
2. Input-group active-high terminals (without a bar over the terminal name) should be connected to a GND terminal.
3. Output terminals should be left open

This does not apply to input terminals connected to a CPU (such as D0 to D7, RD, WR, CS, A1, and A0)
For details, see the PDF file called "How to handle unused terminals."

When a PCL runs out and keeps outputting pulses, does the EL signal function reliably? (Are the +-EL signals independent from the control circuit?)

The PCL series is configured with hard logic circuits, so there is possibility of "overrunning" the internal software.

The EL signal is a level signal and is connected to a reset terminal on the division circuit. Therefore, the EL signal can definitely stop operation.

By supplying an EL signal, the start control circuit is reset and the start signal is turned OFF. By turning OFF the start signal, the variable division circuit and multiplication division circuit are reset, and the output circuit is turned OFF.

When the motor is stopped by turning ON the EL signal, how do you release the EL signal?

In order to release the EL, feed the motor in the opposite direction from that used by the EL to stop. When the signal EL is input by a pulse signal, the current output clock will stop. However, the motor can be restarted without taking any further action.

How is it possible to start a zero return operation while the ORG signal is ON?

The PCD series does not start the zero return. (The zero return operation is delayed.) When the ORG signal is released, they will start to return to the zero position.

LSIs with the "zero position leave" function, such as the PCL 60xx series, can be operated even if the ORG signal is ON by using certain setting methods.

Can we supply a signal to +-EL directly from a switch or sensor?

If the signal on the line is normally open and active LOW, it can be input directly. In order to prevent problems from electrical noise, insert an isolation router on the line.

We want to know the results when both the ORG and Z phase signals are input at the same time. In the following cases, does the PCL count the Z phase signal?
(When the ORG signal is active LOW, and the Z phase signal is active HIGH)

1. **The ORG signal rising edge and the Z phase signal falling edge occur at the same time.**
2. **The ORG signal is input while the Z phase signal is ON.**
3. **The ORG signal falling edge and the Z phase signal falling edge occur at the same time.**

Strictly speaking, the Z phase signal also has rising and falling edge timings. It is not defined whether the Z phase signal will be counted by turning ON the ORG signal on the rising or falling edges of the Z phase signal. (Because these occur in nanoseconds and are difficult to measure.)

If the ORG signal is input while the Z phase signal is ON, the PCL will not count this Z phase signal, and will start counting from the next Z phase signal. However, due to the reason given above, it is better if you understand that it is undefined whether the PCL will count the Z phase signal (once the ORG signal goes ON) from the time the Z phase signal rises until it falls again. Therefore, set the operation to turn the ORG signal ON when the Z phase signal is definitely OFF.

Is it true that during acceleration/deceleration the LSI refreshes the pulse cycle for each pulse that is output, regardless of the operation pattern selected (linear or S-curve)? Or, does the LSI create a step shape pattern at certain multiplication rates (which keeps the same speed for some interval and the curve will be a staircase shape) while accelerating/decelerating?

All of the LSIs in the PCL series use acceleration characteristics, which do not allow the same cycle (speed) pulse train to continue even for a short time. They do not accelerate/decelerate using a staircase pattern (unless they are intentionally set to do so).

We want to synchronize the operation of a motor by receiving an encoder signal from a main axis under control. Is it possible with this model?

The PCL60xx and PCL61xx series have a pulsar input terminal. This terminal can be used to input external pulses, including an A/B phase signal from an encoder. The input signals can be output through the POUT (OUT) or PDIR (DIR) terminals. The pulsar input terminal can handle bi-directional pulses, positive and negative pulses and 90-degree phase difference signals.

We want to set a free acceleration curve other than a sine wave or a 2-dimensional curve.

The PCL60xx and 61xx series can set any free acceleration/deceleration curve using the pre-register function. The pre-register function is used to write the details for the next operation. The next operation is automatically started right after the previous operation completes.

We recommend the following methods:

1. Initialize
2. Set the operation completion timing to "when the output frequency cycle is complete" (= 0) in the operation mode buffer.
3. Set preset counter operation ON (=0) in the control mode buffer. Set the rampdown point to manual (=0), the acceleration/deceleration characteristic to linear (=0), and the next operation auto start function to ON.

	1st operation	2nd operation	3rd operation	4th operation	5th operation
Feed amount	P1	P2	P3	P4	P5
Start frequency	Fl	f1	f2	f3	f4
Target frequency	f1	f2	f3	f4	Fl
Acceleration curve	A1	A2	A3	A4	A5
Ramp-down point	0	0	0	0	0

4. Prepare the data as shown above. From the 2nd operation and after, write the data into the pre-buffer whenever the interrupt indicates that the previous data have been transferred.
5. When writing the data for the 5th operation, note that f4>fL and the size of the frequency will be reversed.
6. Since there is no next operation data, the PCL will stop the motor. This method is effective since the ramp-down point data is 0.

It is easier if we can use linear acceleration/deceleration. However, we need extreme precision in the output frequency settings. (Example of frequency settings: 11827.2pps, 23654.4pps) So, we would like to use higher frequencies with fractional numbers. Is there an NPM chip that can handle the settings above? In addition, we are using DC servomotors and counting pulses with encoders.

To set frequencies in 0.1 pps units, setting the multiplication rate register to 0.1x will be the easiest method. However, using that multiplier, the setting ranges of our PCL60xx series chips are 0.1 to 6553.5pps. Therefore, we cannot meet your required frequency range using this method.

If you use values other than our recommended values for the reference clock, or use a value that is not evenly divisible by the multiplication rate register, our chip will output a frequency with a fractional part. However, the exact frequencies are unlikely to have precisely 0.1 pps units. Therefore the setting will be very complicated.

Is there any model that can accept input from an absolute encoder (gray code)?

All of PCD/PCL series and CCL series will only accept incremental type encoder signals there is no model that accepts absolute encoder signals.

Is the change in speed always linear, even if the acceleration time is set quite long (2 to 3 seconds)?

The change is always linear, regardless of the amount of acceleration/deceleration time.

What are some connection examples for CPUs?
Although we often receive this kind of request, we cannot show you connection examples for all the CPUs on the market. The user's manuals for each model contain block diagrams for 6800 series, H8 series, 8086 series, and Z80 series CPU interfaces. We hope that you can design the circuits you need by referring to these block diagrams.

Is it possible to use these LSIs at ambient temperatures of 100 deg? How about operation at lower temperatures?

The LSIs cannot be operated at temperatures of 100 degrees. However, the temperature conditions vary a little with each model. For details, see the user's manual for each model, and operate the chip within the specified temperature range.

Why do we set bit 3 (stop an acceleration/deceleration operation in the middle)to 1 in the output mode command?

When bit 3 in the output mode command is 0, the PCD will execute a normal ramp-up/down. If it is set to 1:

1. The PCL will halt a ramp-up/down the instant that the bit is set to 1, and it will hold the current pulse frequency.
2. If the PCD has stopped outputting pulses, even if an FH high-speed command is written it will feed at FL speed without a ramp-up.

When relationship of the values in registers R1 (FL) and R2 (FH) is R1>R2, what will the operation be like?

The PCD/PCL series changes the speed from FL to FH speed. Therefore, when R1>R2, it will "decelerate" from FL to FH speed. If it is currently operating at FH speed, it will accelerate from FH to FL speed.

In preset operation, when the LSI reaches the ramp-down point while feeding at FH speed, it will accelerate to FL speed. Therefore, the shape will be an inverted trapezoid.

We want to use a 30 kpps frequency. Is it possible?

It will be possible if you select the 4x mode for the multiplication rate. (Set R4 (multiplication register) to 150.) If the reference clock is 4.9152MHz, and if the maximum value of the speed register range is "8191," the maximum output frequency in 4x mode will be:
8191 x 150 = 32764pps

By increasing the multiplication rate, it can output several hundred kpps.

Compared with the PCL series, the PCD series has a smaller limit for the following settings:

- The number of bits in the acceleration/deceleration rate register
- The number of bits in the ramp-down point setting register

Therefore, when the FH speed is high, or you want to use a long acceleration time, the PCD series may not be able to handle the necessary numbers.

In preset operation, we set R0 = 0 and trigger a start. In this condition, the PCL does not generate any pulses. Then, we set PCL bit 3 = 1 (stop the count in the preset counter) in the register select command, and trigger a start. How come?

Regardless of setting of bit 3 in the register select command, the PCL will not start as long as R0 = 0. However, when bit 5 is set equal to 1 (output INT when stopped) in the start mode command, the PCL will output the INT signal.

We write a start command while the EL signal is ON. Then, we turn OFF the EL signal. Is it normal for the PCL to start operation at this point?

The stop circuit using the EL signal stops the start circuit by ORing "detecting the OFF to ON edge and latched signal of the EL" with the "EL level signal." By turning ON the EL signal, the latched signal is cleared by a reset command (08H). Therefore, after turning ON the EL signal, write a reset command, and a start command. Then turn OFF the EL signal. The PCD will start operation. (This is basically the same for the PCL series.)