F&eIT Series

Isolated Counter Module

CNT24-2(FIT)GY

User's Manual

CONTEC CO.,LTD.

Check Your Package

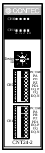
Thank you for purchasing the CONTEC product.

The product consists of the items listed below.

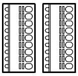
Check, with the following list, that your package is complete. If you discover damaged or missing items, contact your retailer.

Product Configuration List
- Module …1
- First Step Guide …1
- CD-ROM [F&eIT Series Setup Disk] *1 …1
- Interface connector plugs …2

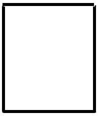
*1 The CD-ROM contains various software and User’s Manual (this manual)



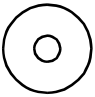
Module



Interface connector plugs



First step guide



CD-ROM
[F&eIT Series Setup Disk]

Copyright

CONTEC CO., LTD. makes no commitment to update or keep current the information contained in this document.
The information in this document is subject to change without notice.

All relevant issues have been considered in the preparation of this document. Should you notice an omission or any questionable item in this document, please feel free to notify CONTEC CO., LTD.

Regardless of the foregoing statement, CONTEC assumes no responsibility for any errors that may appear in this document or for results obtained by the user as a result of using this product.

Trademarks

F&eIT is a registered trademark or trademark of CONTEC CO., LTD. Other company and product names that are referred to in this manual are generally trademarks or registered trade trademark.

Table of Contents

1. Introduction

Congratulations on your recent purchase of an Insulator Counter Module.
The CNT24-2(FIT)GY counts high-speed pulses so that the pulses can be processed within a F&eIT-series controller module
<CPU-CAxx(FIT)GY, CPU-SBxx(FIT)GY etc>. As such, the
CNT24-2(FIT)GY can be used for position alignment control in combination with an encoder. The insulation between external signals and the Controller Module permits the use of the Controller Module without compromising the communications features of the latter.

Please read this manual carefully to create application programs and configure the system, such as setting the switches and connecting it to external devices.

Features

-The CNT24-2(FIT)GY can perform 24-bit up and down counting covering two channels.

-The CNT24-2(FIT)GY can count two-phase signals, such as rotary encoders and linear gauges.

-The CNT24-2(FIT)GY is equipped with general-purpose input signals, with one point per channel.

-A rotary switch allows you to set device IDs to help you keep track of device numbers.

-The system incorporates a screw less connector plug that allows you to easily attach and detach wires without using any special tools.

-Similar to other F&eIT series products, the system, in the module itself, incorporates a 35mm DIN rail mounting mechanism as a standard item. A connection to a controller module can be effected on a lateral, stack basis in a unique configuration, which permits a simple, smart system configuration without the need for a backplane board.

Functions and control method by controller connected

The CNT24-2(FIT)GY can be connected to a variety of controllers.

Micro Controller Unit	: CPU-SBxx(FIT)GY
I/O Controller Module	: CPU-CAxx(FIT)GY
Monitoring & Control Server Unit	: SVR-MMF2(FIT)
Monitoring & Control Server Unit	: SVR-MMF(FIT)GY
Isolated Counter Module for USB	:CNT24-2(USB)GY
I/O Controller Module with USB	: CPU-CA10(USB)GY

The functions and control of the CNT24-2(FIT)GY vary with the controller to which the CNT24-2(FIT)GY is connected.

Functions available with each controller connected

	CPU-SBxx(FIT)GY	CPU-CAxx(FIT)GY	SVR-MMF2(FIT)	SVR-MMF(FIT)GY	CNT24-2(USB)GY	CPU-CA10(USB)GY
2-phase input; asynchronous clear; multiplication by 1/2/4	o	o	*1	*1	o	o
2-phase input synchronous clear, multiplication by 1/2/4	o	o			o	o
Single-phase input, asynchronous clear, multiplication by 1	o	o			o	o
Single-phase input with gate control, multiplication by 1/2	o	o			o	o
Digital filter	o	o			o	o
Interrupt function	o					
Programmable timer	o					
Count match pulse output	o	o			o	
General-purpose digital input	o	o			o	o
Device ID setting range	0 - 7	0 - 7	0 - 7	0 - 7	1 - 3	0 - 7

*1 For the function available, refer to the reference manual for the SVR-MMF2(FIT), SVR-MMF(FIT)GY.

Control method by controller connected

		CPU-SBxx(FIT)GY	CPU-CAxx(FIT)GY	SVR-MMF2(FIT)	SVR-MMF(FIT)GY	CNT24-2(USB)GY	CPU-CA10(USB)GY
Control using the I/O address map		o					
Control using the memory address map			o				
Control via the Windows driver *	FIT Protocol		o				
	API-CAP(W32)		o				
	API-SBP(W32)	o					
	API-USBP(WDM)					o	o
Control over the web				o	o		

* The API-SBP(W32) is included in the development kit DTK-SBxx(FIT)GY; the other drivers are bundled with each controller.

Control using the I/O address map

When connected to the CPU-SBxx(FIT)GY, the CNT24-2(FIT)GY can receive I/O instructions directly from the controller module. For details, see Chapter 4 “Using the I/O Address Map”.

Control using the memory address map

When connected to the CPU-CAxx(FIT))GY, the CNT24-2(FIT)GY can be accessed from the host computer over the network.

The CNT24-2(FIT)GY is assigned with its device ID in the memory managed by the controller module. The application running on the host computer controls the module by reading/writing the memory managed by the controller module. For details, see Chapter 5 “Using the Memory Address Map”.

Control via the Windows driver

For the functions and settings available when using the Windows driver, refer to the reference manual and online help for each module.

Control over the web

You can monitor collected data and manage the log over the web. You can use your familiar browser to easily make various settings. For details, refer to the reference manual for the SVR-MMF2(FIT), SVR-MMF(FIT)GY.

Limited One-Year Warranty

CONTEC F&eIT series products are warranted by CONTEC CO., LTD. to be free from defects in material and workmanship for up to one year from the date of purchase by the original purchaser.

Repair will be free of charge only when this product is returned freight prepaid with a copy of the original invoice and a Return Merchandise Authorization to the distributor or the CONTEC group office, from which it was purchased.

This warranty is not applicable for scratches or normal wear, but only for the electronic circuitry and original products. The warranty is not applicable if the device has been tampered with or damaged through abuse, mistreatment, neglect, or unreasonable use, or if the original invoice is not included, in which case repairs will be considered beyond the warranty policy.

How to Obtain Service

For replacement or repair, return the device freight prepaid, with a copy of the original invoice. Please obtain a Return Merchandise Authorization Number (RMA) from the CONTEC group office where you purchased before returning any product.

*** No product will be accepted by CONTEC group without the RMA number.**

Liability

The obligation of the warrantor is solely to repair or replace the product. In no event will the warrantor be liable for any incidental or consequential damages due to such defect or consequences that arise from inexperienced usage, misuse, or malfunction of this device.

Handling Precautions

Take the following precautions when handling this module.

-Do not modify the module. CONTEC will bear no responsibility for any problems, etc., resulting from modifying this module.

-Do not use or store the equipment in a hot or cold place, or in a place that is subject to severe temperature changes. (Operating temperature range: 0 - 50°C)

-Do not use or store the equipment in a place subject to direct sunlight or near a heating device, such as a stove.

-Do not use or store the equipment in a dusty or humid place. (Operating humidity range: 10 - 90%RH, No condensation)

-As this product contains precision electronic components, do not use or store in environments subject to shock or vibration.

-Do not use or store the product near equipment generating a strong magnetic field or radio waves.

-If you notice any strange odor or overheating, please unplug the power cord immediately.

-In the event of an abnormal condition or malfunction, please consult the dealer from whom the equipment was purchased.

-To avoid electric shock, please do not touch the system with a wet hand.

-Do not open the module casing. CONTEC will disclaim any responsibility for equipment whose casing has been opened.

-To prevent damage, please do not subject the module to impact or bend it.

-To prevent contact malfunction, please do not touch the metallic pins on the external module connector.

-The module contains switches that need to be properly set. Before using the module, please check its switch settings.

-To avoid malfunction, please do not change the module switch settings in an unauthorized manner.

-Do not operate the device module when the power for the Controller Module is on. To avoid malfunction, please be sure to turn off the power for the Controller Module.

FCC PART 15Class A Notice

NOTE

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in commercial environment.
This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference at his own expense.

WARNING TO USER

Change or modifications not expressly approved the manufacturer can void the user's authority to operate this equipment.

About the Manual

This manual consists of the following chapters:

Chapter 1	Introduction
Chapter 2	Module Nomenclature and Settings Explains the nomenclature of the components of the Module and their switch settings.
Chapter 3	Connecting to an External Device Explains interface connectors and external I/O circuits.
Chapter 4	Using the I/O Address Map Explains I/O port bit assignments and the definitions of the bits when the Module is used as a CPU-SBxx(FIT)GY module.
Chapter 5	Using the Memory Address Map Explains the module settings area, the information area, and the I/O data area when the Module is used as a CPU-CAxx(FIT)GY module.
Chapter 6	System Reference Explains module specifications and circuit block diagrams.

2. Module Nomenclature and Settings

Nomenclature of Module Components

Figure 2.1. shows the names of module components. In the figure, the indicated switch settings represent factory settings.

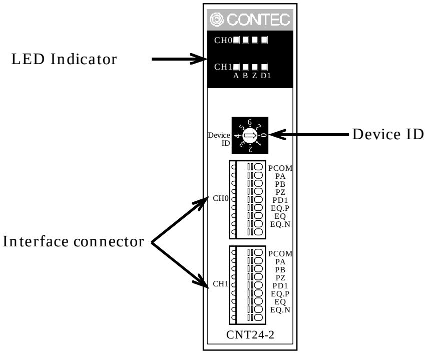


Figure 2.1.   Names of Module Components

Setting a Device ID

The controller module distinguishes and keeps track of the modules that are connected to it by assigning device IDs to them.
Each module, therefore, should be assigned a unique ID.

A Device ID can be assigned in a 0 - 7 range, so that a maximum of eight modules can be distinguished.

To connect the CNT24-2(FIT)GY to the CNT24-2(USB)GY, assign a device ID between 1 and 3.

The factory setting for the Device ID is [0].

Setup Method

A Device ID can be set by turning the rotary switch that is located on the module face.
A Device ID can be assigned by turning the switch.

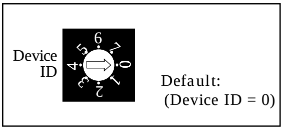


Figure 2.2. Setting a Device ID (SW1)

LED Indicator

A: Turns on when a current flows from pin PA
(low level, negative logic). (green)

B: Turns on when a current flows from pin PB
(low level, negative logic. (green)

Z: Depends on the logic setting on the Z phase.

Z phase set to positive logic: turns on when pin PZ is open (high level, positive logic). (green)
Z phase set to negative logic: turns on when a current flows from pin PZ. (low level, negative logic).(green)

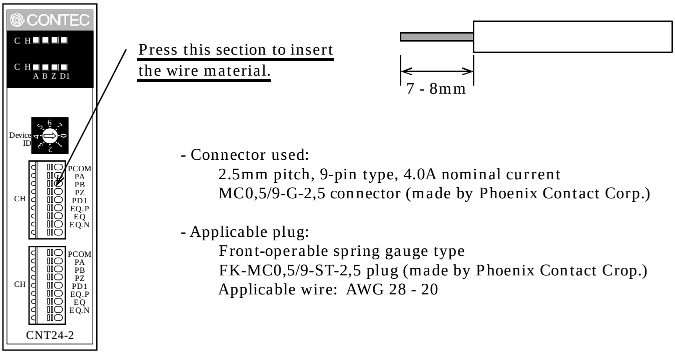
DI: turns on when a current flows from pin PD.
(low level, negative logic).(green)

3. Connecting to an External Device

Interface Connector

How to Connect an Interface Connector

When connecting the Module to an external device, you can use the supplied connector plug. When wiring the Module, strip off approximately 7 - 8 mm of the covering for the cable, and insert the bare wire by pressing the orange button on the connector plug. Releasing the orange button after the wire is inserted to fix the cable. Compatible wires are AWG 28 - 20.



- Connector used:
  2.5mm pitch, 9-pin type, 4.0A nominal current
  MC0,5/9-G-2,5 connector (made by Phoenix Contact Corp.)

- Applicable plug:
  Front-operable spring gauge type
  FK-MC0,5/9-ST-2,5 plug (made by Phoenix Contact Crop.)
  Applicable wire: AWG 28 - 20

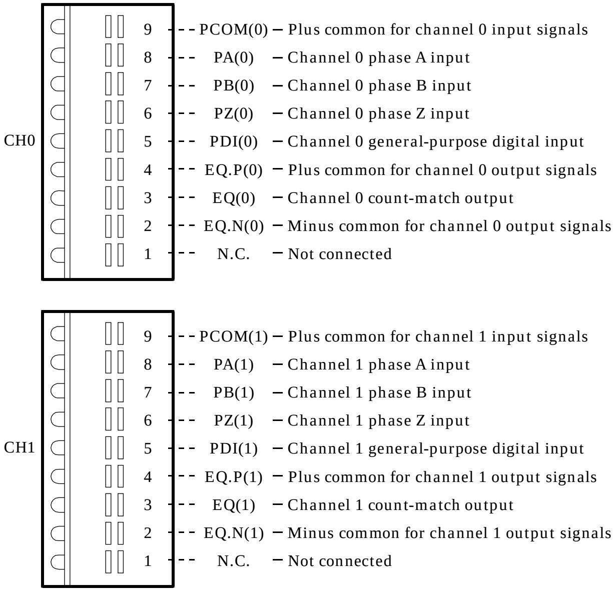
Figure 3.1. Connecting an Interface Connector and Connectors That Can Be Used

Note!

Removing the connector plug by grasping the cable can break the wire.

Signal Layout on the Interface Connector

The Module can be connected to an external device using a 9-pin (1 group) connector that is provided on the Module face.



CH0

9	PCOM(0)	–	Plus common for channel 0 input signals
8	PA(0)	–	Channel 0 phase A input
7	PB(0)	–	Channel 0 phase B input
6	PZ(0)	–	Channel 0 phase Z input
5	PDI(0)	–	Channel 0 general-purpose digital input
4	EQ.P(0)	–	Plus common for channel 0 output signals
3	EQ(0)	–	Channel 0 count-match output
2	EQ.N(0)	–	Minus common for channel 0 output signals
1	N.C.	–	Not connected

CH1

9	PCOM(1)	–	Plus common for channel 1 input signals
8	PA(1)	–	Channel 1 phase A input
7	PB(1)	–	Channel 1 phase B input
6	PZ(1)	–	Channel 1 phase Z input
5	PDI(1)	–	Channel 1 general-purpose digital input
4	EQ.P(1)	–	Plus common for channel 1 output signals
3	EQ(1)	–	Channel 1 count-match output
2	EQ.N(1)	–	Minus common for channel 1 output signals
1	N.C.	–	Not connected

Figure 3.2. Signal Layout on the Interface Connector

Connecting an External Signal

Connecting to a Opto-Isolated Input Circuit

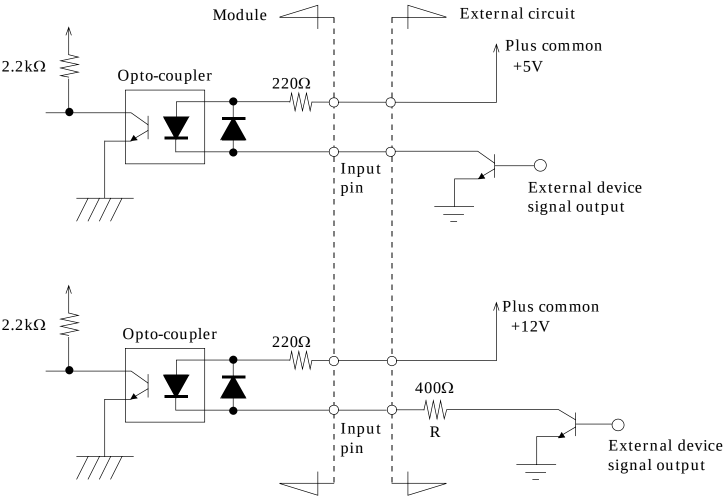


Figure 3.3. Isolated input Circuit and an Example of a Connection

Notes!

-The general-purpose input signal also has a similar circuit configuration.

-When an external power supply other than 5V is used, insert a current-limiting resistor at position R. If PV denotes an external power supply, the current-limiting resistor R can be calculated as follows:

$$\frac{P-5}{20} < Rk\Omega < \frac{P-5}{15}$$

For example, P = 12V will require the following resistance:

350Ω < R < 470Ω.

Output Circuit and an Example Connection

When there is a match between a channel count and a specified value, a one-shot (one pulse) match signal is output to the outside. The signal output section has an open collector configuration based on opto-isolation. Driving the output of this module requires an external power supply.

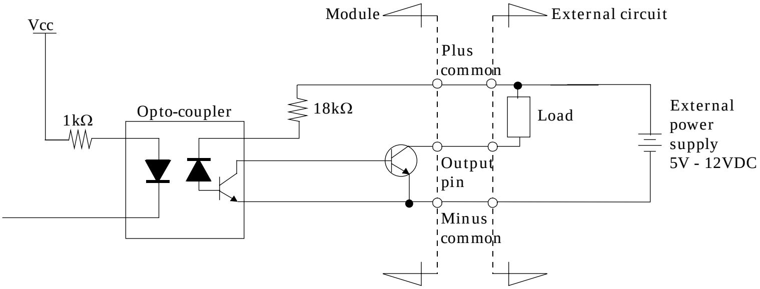


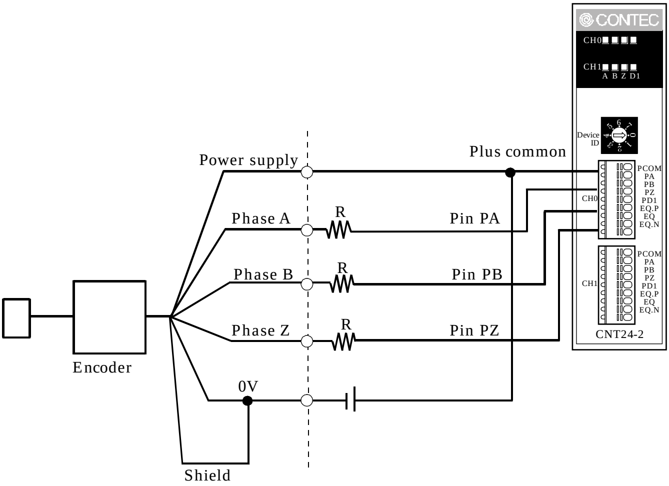
Figure 3.4. Connecting to an Output Circuit

Note!

A surge voltage protection circuit is not provided on the output transistors for this Module. Therefore, when driving relays, lamps, and other induction loads using this Module, a surge voltage countermeasure should be provided on the load side.

For a description of how to deal with surge voltages, see "Surge Voltage Countermeasures".

Connecting to a Rotary Encoder



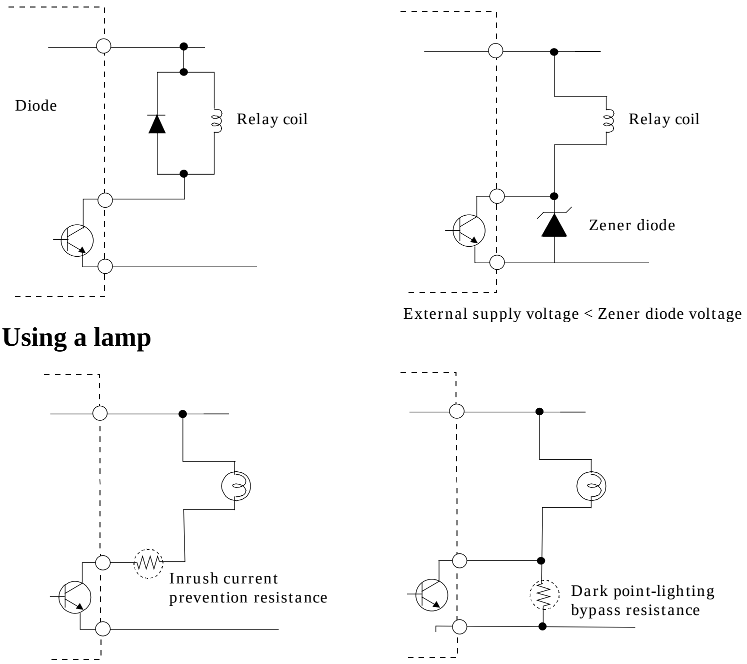
* A 400Ω resistor is required at position R when used at 12V, but not at 5V.

Figure 3.5. Connecting to a Rotary Encoder

Surge Voltage Countermeasures

When connecting a load that generates surge voltages and inrush currents, such as an induction load (relay coil) or an incandescent light bulb, to the one-shot pulse output, appropriate protection must be provided in order to prevent damage to the output stage or a malfunction due to noise. The rapid shutoff of a coil, such as a relay, generates a sudden high-voltage pulse. If this voltage exceeds the voltage tolerance level of the output transistor, it can cause the transistor to gradually deteriorate, or even completely damage the transistor. Therefore, when driving an induction load, such as a relay coil, you should always connect a surge-absorbing device. The following illustrates a surge voltage countermeasure that can be employed:

Using a relay coil



Using a lamp

Figure 3.6. Surge Voltage Countermeasure

Note!

In order for a protection circuit to operate effectively, it must be connected within 50 cm of a load and a contact point.

4. Using the I/O Address Map

Starting I/O Address

When connected to a CPU-SBxx(FIT)GY, the CNT24-2(FIT)GY can directly receive I/O commands from the controller module. Depending on how the Device ID is set, the I/O addresses indicated below will be used exclusively by the CNT24-2(FIT)GY.

Because the address bus on which the I/O address space is specified is not fully decoded in continued 16 bits, four starting I/O addresses exist in each Device ID.

If the Device ID is set to 0h, one of the four addresses (0800h, 0840h, 0880h, or 08C0h) will be used as a starting I/O address.

Table 4.1. List of Starting I/O Addresses

ID No.	Occupied I/O address			
0	0800h - 081Fh(recommend)	0840h - 085Fh	0880h - 089Fh	08C0h - 08DFh
1	1800h - 181Fh(recommend)	1840h - 185Fh	1880h - 189Fh	18C0h - 18DFh
2	2800h - 281Fh(recommend)	2840h - 285Fh	2880h - 289Fh	28C0h - 28DFh
3	3800h - 381Fh(recommend)	3840h - 385Fh	3880h - 389Fh	38C0h - 38DFh
4	4800h - 481Fh(recommend)	4840h - 485Fh	4880h - 489Fh	48C0h - 48DFh
5	5800h - 581Fh(recommend)	5840h - 585Fh	5880h - 589Fh	58C0h - 58DFh
6	6800h - 681Fh(recommend)	6840h - 685Fh	6880h - 689Fh	68C0h - 68DFh
7	7800h - 781Fh(recommend)	7840h - 785Fh	7880h - 789Fh	78C0h - 78DFh

For detailed specifications on the I/O space that is controlled by the controller module, see the controller module manual.

List of I/O Address Maps

Input Port

Starting I/O address	D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0
	Product Category				Revision Data			
+0 (00h)	0	0	1	1	Revision Data3	Revision Data2	Revision Data1	Revision Data0
	Product ID Number							
+1 (01h)	0	0	0	0	0	0	0	0
	Interrupt Status							
+2 (02h)	Enable	Status	0	0	0	IRQ9	IRQ7	IRQ5
+3 (03h)	(Not available)							
⋮								
+16 (10h)								
	Interrupt Status							
+17 (11h)	Data07	Data06	Data05	Data04	Data03	Data02	Data01	Data00
+18 (12h)	(Not available)							
⋮								
+31 (1Fh)								

Figure 4.1. Input Port

Output Port

Starting I/O address	D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0
+0 (00h)	(Not allowed)							
+1 (01h)								
	Interrupt Data							
+2 (02h)	Enable	N/A	N/A	N/A	N/A	IRQ9 Data	IRQ7 Data	IRQ5 Data
+3 (03h)	(Not allowed)							
⋮								
+15 (0Fh)								
	Command							
+16 (10h)	Command Data7	Command Data6	Command Data5	Command Data4	Command Data3	Command Data2	Command Data1	Command Data0
	Setting Data							
+17 (11h)	Setting Data7	Setting Data6	Setting Data5	Setting Data4	Setting Data3	Setting Data2	Setting Data1	Setting Data0
+18 (12h)	(Not allowed)							
⋮								
+31 (1Fh)								

Figure 4.2. Output Port

Specifications Common to F&eIT Products

The starting I/O addresses from +0h - +Fh are common to all modules in the F&eIT series.

Product Information

Starting I/O address	D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0
input	Product Category				Revision Data			
+0 (00h)	0	0	1	1	Revision Data3	Revision Data2	Revision Data1	Revision Data0
	Product ID Number							
+1 (01h)	0	0	0	0	0	0	0	0

Figure 4.3. Product Information

-Revision Data [D3 - D0]:
This is product update information, subject to change without notice, that is controlled by Contec.

-Product Category [D7 - D4]:
This is a module function classification code.
For the CNT24-2(FIT)GY, the code is "3h".

Code	Function
0	Extention BUS
1	Digital input-output
2	Analog input-output
3	Counter
4	Serial communication
5	GPIB
6-F	Reserved

Figure 4.4. Product Category

-Product ID Number [D7 - D0]:
This is a product ID number within the same Product ID Number.
For the CNT24-2(FIT)GY, the ID is “0h”.

Following are examples demonstrate the initialization of the device by high-level languages:

Microsoft C   Microsoft QBASIC

```
ProductID = inp( ADR+1 );        ProductID = INP( ADR+1 )
```

* ADR indicates the starting I/O address for the module.

Interrupt Status

This is a port on which the interrupt status generated by the Module can be verified. Although in this example values are assigned centered on the status concerning interrupt levels, information on interrupt sources varies from module to module.

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
input	Interrupt Status							
+2 (02h)	Enable	Status	0	0	0	IRQ9	IRQ7	IRQ5

Figure 4.5. Interrupt Status

-Enable [D7]:
This verifies the interrupt source enabled/disabled status.
The value "1" indicates that a hardware interrupt on the controller module is enabled.

-Status [D6]:
This bit indicates an interrupt request status in the module.
When IRQ5, IRQ7, or IRQ9 is "1", this bit will also be "1".

-IRQ* [D2 - D0]:
These bits allow you to verify the interrupt level that is currently set. The current interrupt level is indicated as "1".

Following are examples of the initialization that is performed in high-level languages:

Microsoft C

```
IrqStatus = inp( ADR+2 );
```

Microsoft QBASIC

```
IrqStatus = INP( ADR+2 )
```

Setting an Interrupt Level

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
output	Interrupt Data							
+2 (02h)	Enable	N/A	N/A	N/A	N/A	IRQ9 Data	IRQ7 Data	IRQ5 Data

Figure 4.6. Setting an Interrupt Level

-Enable [D7]:
This bit enables an interrupt source.

-IRQ* [D2 - D0]:
The interrupt level used by the module is set in these bits.

Following are examples of initialization settings that can be affected in high-level languages.
The interrupt level to be used is assigned to IRQ5.

Microsoft C

```
outp( ADR+2, 0x81 );
```

Microsoft QBASIC

```
OUT ADR+2, &H81
```

Bit Assignments for I/O Ports

The I/O ports for the counting function have command-section and data-section registers.

A register can be set as follows: First, a command is issued (OUT) to a port located at *starting I/O address + 16* to make the register available for setting. For output, data is assigned (OUT) to the register at port *+17*; for input, the +17 port is read. In other words, for both input and output, a command is issued (OUT) to the output port +16 to make registers available for setting, and either data is issued (OUT) to the port +17 to set the register or the register is read at the +17 port.

When setting a register even when using the same command, the command should be output to the output port +16 each time.

Figures 6.7. and 6.8. show I/O port bit assignments.

Starting I/O address	D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0
input +16 (10h)	(Not available)							
input +17 (11h)	Data/Status							
	Data07	Data06	Data05	Data04	Data03	Data02	Data01	Data00

Figure 4.7. Input Port

Starting I/O address	D 7	D 6	D 5	D 4	D 3	D 2	D 1	D 0
output +16 (10h)	Command							
	Command Data7	Command Data6	Command Data5	Command Data4	Command Data3	Command Data2	Command Data1	Command Data0
output +17 (11h)	Setting Data							
	Setting Data7	Setting Data6	Setting Data5	Setting Data4	Setting Data3	Setting Data2	Setting Data1	Setting Data0

Figure 4.8. Output Port

Flow of Count Values

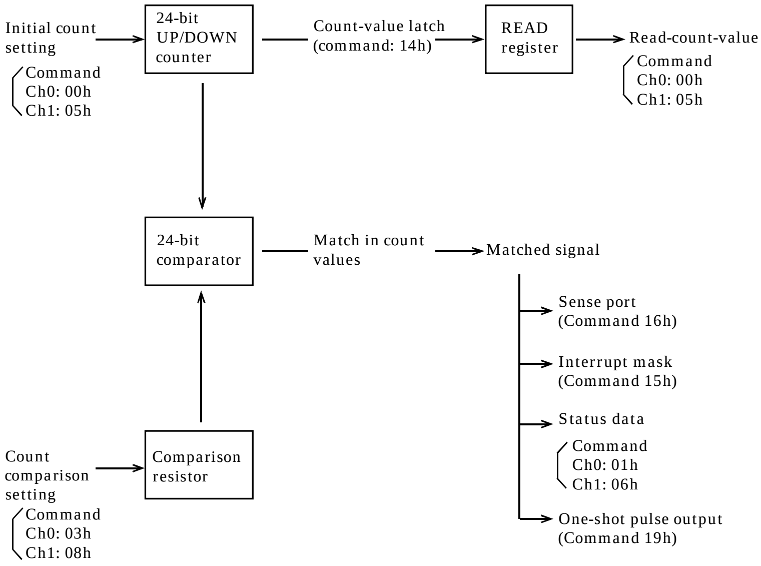


Figure 4.9. Flow of Count Values

Operation Commands

Table 4.2. Output Commands

Command (h)	D7	D6	D5	D4	D3	D2	D1	D0	Function	Data size
00	CH0 initial count value								CH0 initial count value	24-bit
01	$\overline{\text{RESET}}$	SEL	ZSEL	UD/$\overline{\text{AB}}$	DIR	SEL2	SEL1	SEL0	CH0 mode setting	8-bit
02	Not allowed					ZE1	ZE0	0	CH0 phase Z CLR input	3-bit
03	CH0 count comparison value								CH0 count comparison value	24-bit
04	Not allowed				Digital filter				CH0 digital filter	4-bit
05	Same as CH0								CH1 initial count value	24-bit
06	Same as CH0								CH1 mode setting	8-bit
07	Same as CH0								CH1 phase Z CLR input	3-bit
08	Same as CH0								CH1 count comparison value	24-bit
09	Same as CH0								CH1 digital filter	4-bit
14	Not allowed						CH1LT	CH0LT	Count-value latch	2-bit
15	Not allowed			TIME	Not allowed		CH1	CH0	Interrupt mask	3-bit
16	Not allowed			TIME	Not allowed		CH1	CH0	Sense reset	3-bit
17	Programmable timer settings data								Timer data	8-bit
18	Not allowed							START	Timer start	1-bit
19	One-shot pulse width data								One-shot pulse	8-bit

Table 4.3. Input Commands

Command (h)	D7	D6	D5	D4	D3	D2	D1	D0	Function	Data size
00	CH0 initial count value								CH0 count value	24-bit
01	AI	Z	A	B	1	U/D	$\overline{\text{EQ}}$	U	CH0 status	8-bit
02	CH1 initial count value								CH1 count value	24-bit
03	AI	Z	A	B	1	U/D	$\overline{\text{EQ}}$	U	CH1 status	8-bit
15	0	0	0	TIME	0	0	CH1	CH0	Interrupt mask	3-bit
16	0	0	0	TIME	0	0	CH1	CH0	SensePort	3-bit

Description of Input Commands

Reading a Count (command CH0: 00h, CH1: 05h)

By reading the contents of the READ register for a corresponding channel, this command reads a count value. By issuing the command to the output port +16 and reading the input port +17 three times, you can read low, middle, and high count values.

In the initial state, the contents of the READ register are undefined.

Following are programs that read a count value from CH0:

Microsoft C

```
outp( ADR+16, 0x0 );
LowerData = inp( ADR+17 );
MiddleData = inp( ADR+17 );
UpperData = inp( ADR+17 );
```

Microsoft QBASIC

```
OUT ADR+16, &H0
LowerData = INP( ADR+17 )
MiddleData = INP( ADR+17 )
UpperData = INP( ADR+17 )
```

In this case, the count value latch for CH0 must be pre-set to [1].

Status Data (command CH0: 01h, CH1: 06h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
01h/06h	AI	Z	A	B	1	U/D	$\overline{\text{EQ}}$	U

By issuing the command to the output port +16 and reading the input port +17, you can monitor signal inputs, the direction of counting, any count matches, and the state of any abnormal input.

AI When phases A and B change simultaneously during two-phase input or simultaneous changes in UP and DOWN pulses are detected during UP/DOWN input, the event is flagged as abnormal pulse input, and this bit is set to [1].

1: Abnormal input detected 0: No abnormal input detected

Z Indicates the input status of phase Z.

<Positive logic>

1: Phase Z input status [1] 0: Phase Z input status [0]

<Negative logic>

1: Phase Z input status [0] 0: Phase Z input status [1]

A Indicates the input status of phase A.

1: Phase A input status [1] 0: Phase A input status [0]

B Indicates the input status of phase B.

1: Phase B input status [1] 0: Phase B input status [0]

U/D Count direction operating status

This status bit indicates the counting direction of the current UP/DOWN counter.

0: Counting in UP direction

1: Counting in DOWN direction

$\overline{\text{EQ}}$ Match detection output

0: Count value matches the contents of the compare register.

1: Count value does not match the contents of the compare register

U Indicates the status of generic input

1: Generic input status [1] 0: Generic input status [0]

Notes!

- *The initial state changes with the external connection state.*
- *The status for phases A, B, and Z represents the data that is in effect after the filtering function is processed, and thus involves a delay of 4 setting cycle clocks. The generic input indicates the state of external input "as is".*
- *The logic for phase Z is set in [ZSEL] in the operating mode settings.*

Interrupt Mask (command 15h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
15h	0	0	0	TIME	0	0	CH1	CH0

This command allows you to monitor the status of the current interrupt mask that was set using the interrupt mask command. TIME is associated with the programmable timer, and CH1 - CH0 are associated with channels. When these bits are [1], the interrupt is masked, and no interrupt signals are output.

You can monitor the masking status by issuing the value 15h to the output port +16 and by reading the input port +17.

0: Not masked 1: Masked

Note!

Initial state: 1FH

Sense Port (command 16h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
16h	0	0	0	TIME	0	0	CH1	CH0

This command indicates any matching counts in the channels and any time-up status.

You can monitor the status of the sense port by issuing the value 16h to the output port +16 and reading the input port +17.

-TIME This bit is set to [1] when the time limit that is set on the programmable timer has elapsed.

-CH1 - CH0

With regard to channels CH1 - CH0, when there is a match between the contents of the UP/DOWN counter and the compare register, the bits associated with the channels become [1]. In this manner, when counting operations are performed on multiple channels and an interrupt is generated upon the generation of a match signal, you can determine from which channel the interrupt is generated.

0: No match in counts; pending time-up

1: A match in counts; time-up

Notes!

- *Initial state: 00h*
- *If an interrupt is generated with the interrupt option set, and if one of the bits in TIME or CH1 to CH0 is set to [1], before another interrupt can be generated, the affected sense bit must be reset. For a description of how to reset a sense bit, see "Sense Reset" on output ports.*

Description of Output Commands

Initial Count Value (command CH0: 00h, CH1: 05h)

The command is issued to the output port +16, and an initial count value is set on the output port +17. Because count data consists of 24 bits, count values are output three times in low, middle, and high order, 8 bits each time. When the third (high 8 bits) count value is output, data consisting of 24 bits is loaded simultaneously by the counting process.

Following are program examples that set a count value 100(64h) on CH0:

Microsoft C

```
outp( ADR+16, 0x0 );
outp( ADR+17, 0x64 );
outp( ADR+17, 0x0 );
outp( ADR+17, 0x0 );
```

Microsoft QBASIC

```
OUT ADR+16, &H0
OUT ADR+17, &H64
OUT ADR+17, &H0
OUT ADR+17, &H0
```

Operating Mode (command CH0: 01h, CH1: 06h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
01h/06h	$\overline{\text{RESET}}$	SEL	ZSEL	UD/$\overline{\text{AB}}$	DIR	SEL2	SEL1	SEL0

The command is issued to the output port +16, and the operating mode is set on the output port +17.

$\overline{\text{RESET}}$ Clears the 24-bit UP/DOWN counter to "000000h".
No counting is performed during the period RESET = 0.
0: Counter cleared 1: Counting
After a zero-clear operation is performed when Z-phase input is enabled one time, and when a RESET = 0 operation is performed, the "Z-phase enabled one time" option is set again.

SEL Switches pulse inputs into counters.
0: Opto-isolated input (fixed)
1: Not available

ZSEL Selects Z-phase input logic (positive/negative).
0: Positive logic (HIGH active)
1: Negative logic (LOW active)

UD/$\overline{\text{AB}}$, SEL2 to 0

Selects a counter operation mode.

Table 4.4. Counter Operation Mode

UD/$\overline{\text{AB}}$	DIR	SEL2	SEL1	SEL0	Operating mode to be set
0	See Table 6.5.	0	0	0	2-phase input, synch clear, 1x mode
0		0	0	1	2-phase input, synch clear, 2x mode
0		0	1	0	2-phase input, synch clear, 4 mode
0		1	0	0	2-phase input, asynch clear, 1x mode
0		1	0	1	2-phase input, asynch clear, 2x mode
0		1	1	0	2-phase input, asynch clear, 4 mode
1		0	1	1	mono-phase input, synch clear, 1x mode
0		0	1	1	Mono-phase input with gate control, asynch clear, 1x mode
0		1	1	1	Mono-phase input with gate control, asynch clear, 2x mode

DIR Switches the counting directions of the rotary encoder.

Table 4.5. Switching Counting Directions

DIR	Direction of rotation of rotary encoder	
	Clockwise	Counterclockwise
0	DOWN	UP
1	UP	DOWN

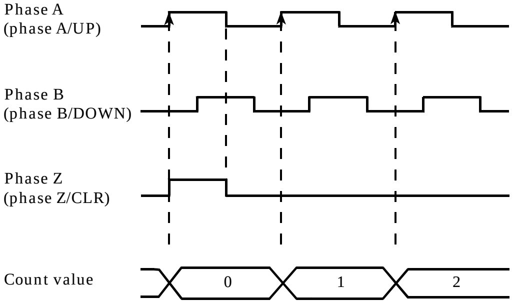
Note!

Initial state: 00h for all channels

Counter Operating Modes

2-Phase Input

2-phase pulse input refers to the input of two pulses, A phase (fast signal) and B phase (slow signal) that differ in phase by 90°. If Z phase is provided (reference position signal), the counter can be cleared using 2-phase pulse input.

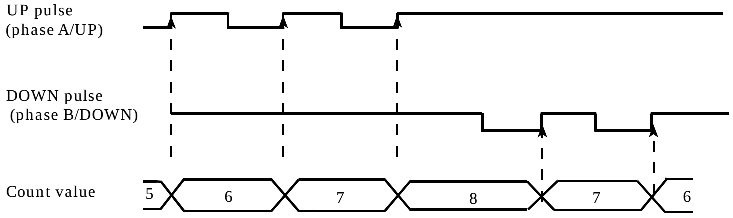


* The figure above illustrates the counting operation when DIR = 1 is specified.
If DIR = 0, a DOWN count commences when phase A rises.

Figure 4.10. Example of Counting with 2-Phase Input

Mono-phase input

During mono-phase input, the system counts up upon the input of an UP pulse, and counts down upon the input of a DOWN pulse. A count fails if UP and DOWN pulses occur simultaneously or both pulses happen to be LOW.

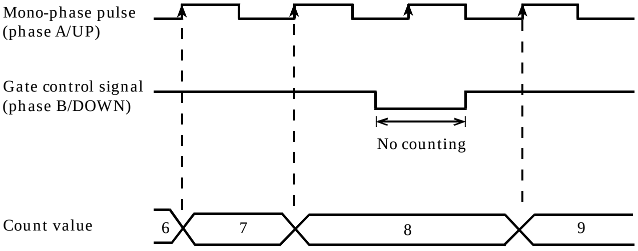


* The figure above illustrates the counting operation when DIR = 1 is specified.
If DIR = 0, a DOWN count commences when a phase A pulse rises, and an UP count commences when a phase B pulse falls.

Figure 4.11. Example of Counting with Mono-Phase Input

Mono-Phase Input with Gate Control

The counter can be started/stopped according to the gate control signal that is input together with a string of mono-phase pulses. The clear signal zero-clears the counter value.

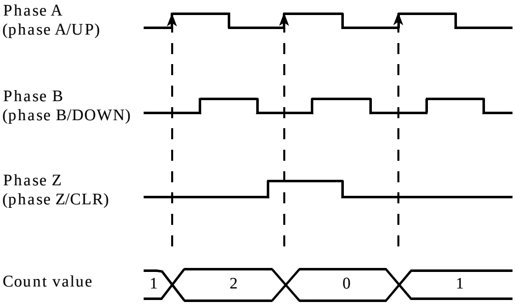


* The figure above illustrates the counting operation when DIR = 1 is specified.
If DIR = 0, a DOWN count commences when the gate control signal (phase B/DOWN) is HIGH and when a mono-phase pulse example (phase A/UP) rises; the counting stops when the gate control signal is LOW.

Figure 4.12. Example of a Mono-Phase Input Count with Gate Control

Synchronous Clear

If DIR=1 and ZSEL=0, the counter is zero-cleared when A phase rises with the B-phase input LOW an the Z-phase input HIGH; the counting process is started when A phase rises after the Z-phase input turns LOW.

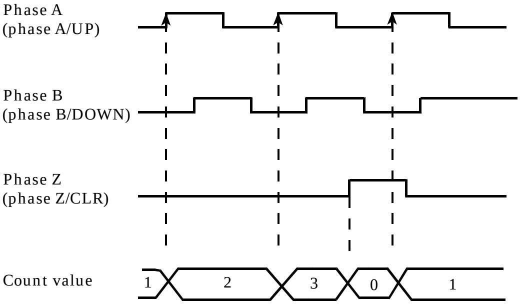


* If DIR = 0, a DOWN count commences when phase A rises with phase B LOW.
if ZSEL= 1, the operation is enabled when phase Z input is LOW.

Figure 4.13. Example of Counting with a Synchronous Clear

Asynchronous Clear

If DIR=1 and ZSEL=0, the counter is zero-cleared when Z phase turns HIGH, irrespective of the input state of phase A or B. The counting process is started at the next rise of A phase, irrespective of the input state of Z phase.

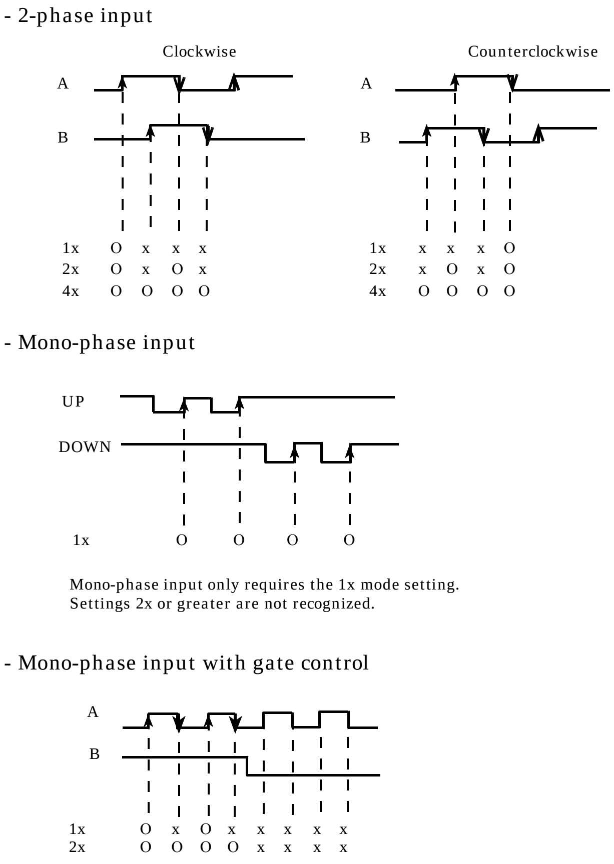


* If DIR = 0, a DOWN count commences when phase A rises with phase B LOW.
if ZSEL= 1, the operation is enabled when phase Z input is LOW.

Figure 4.14. Example of Counting with an Asynchronous Clear

Count Input Multiplier

Detailed control can be effected by setting the count input multiplier to 2 or 4.



- 2-phase input

Clockwise

1x	O	x	x	x
2x	O	x	O	x
4x	O	O	O	O

Counterclockwise

1x	x	x	x	O
2x	x	O	x	O
4x	O	O	O	O

- Mono-phase input

1x	O	O	O	O

Mono-phase input only requires the 1x mode setting.
Settings 2x or greater are not recognized.

- Mono-phase input with gate control

1x	O	x	O	x	x	x	x	x
2x	O	O	O	O	x	x	x	x

Figure 4.15. Example of Counting with a Count Input Multiplier Set

Z Phase /CLR Input (command CH0: 02h, CH1: 07h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
02h/07h	Not used					ZE1	ZE0	0

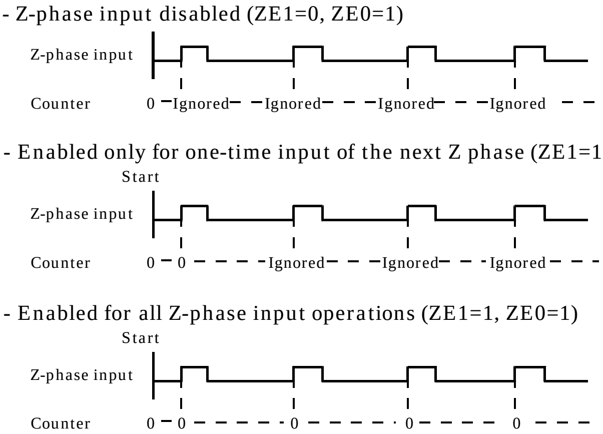
The command is issued to the output port +16, and the number of Z-phase input operations is assigned to the output port +17.

ZE1, ZE0 Selecting a Z phase input mode

Table 4.6. Z phase Input Mode

	ZE1	ZE0	Specified state
	0	1	Z-phase input disabled. Use this value when there is no Z phase.
*	1	0	Enabled only for one-time input of the next Z phase.
	1	1	Enabled for all Z-phase input operations.

* When in the initial condition


- Z-phase input disabled (ZE1=0, ZE0=1)

- Enabled only for one-time input of the next Z phase (ZE1=1, ZE0=0)

- Enabled for all Z-phase input operations (ZE1=1, ZE0=1)

Figure 4.16. Effective Z Phase Count ZSEL=0 (Positive Logic)

Notes!

- *The initial setting is "effective for only one input of the next Z phase" (04h).*
- *If ZSEL=1 (negative logic), the Z phase input is effective only when it is LOW.*
- *If the Z phase /CLR input command is not used, the following settings must be specified: ZE1=0, ZE0=1 (Z phase input disabled).*

-When the Z phase input is effective only once and if a zero-clear is performed upon the input of Z phase and the count value is cleared by setting $\overline{\text{RESET}}$=0, the "Z-phase effective one time" setting is effected again when $\overline{\text{RESET}}$=0 is set.

Compare Register (command CH0: 03h, CH1: 08h)

This command compares the count value for a corresponding channel with the value of a compare register, and if there is a match, the command sets '$\overline{\text{EQ}}$" for the status bit to "0" (the "0" value remains in effect for the duration of the match). The initial condition for this register is 0h. The command is issued to the output port +16, and a count comparison value is set on the output port +17. Because the count comparison value is 24-bit long, the data is output three times, in low, middle, and high order. Following are example programs that assigns a count comparison value 1000 (3E8h) to channel CH1:

Microsoft C

```
outp( ADR+16, 0x3 );
outp( ADR+17, 0xe8 );
outp( ADR+17, 0x3 );
outp( ADR+17, 0x0 );
```

Microsoft QBASIC

```
OUT ADR+16, &H3
OUT ADR+17, &HE8
OUT ADR+17, &H3
OUT ADR+17, &H0
```

Depending on the settings, a once-shot pulse can also be output to an external device (see the section on One-Shot Pulse).

Digital Filter (command CH0: 04h, CH1: 09h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
04h/09h	Not used				Clock data for digital filter			

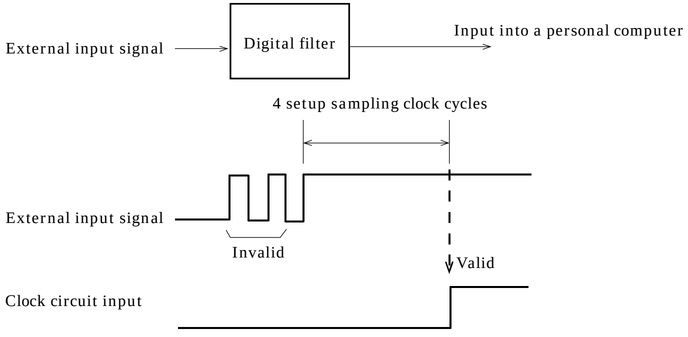
The digital filter is designed to ensure that the counter will operate normally even in the presence of noise in the pulse input to the counter and in phase A, B, and Z signals. Digital filter clock setting data determines the sampling clock cycle for the digital filter.

When detecting 4 clocks of continuous HIGH (or LOW) signals by sampling the input signals with this sampling clock, the digital filter outputs a HIGH (or a LOW), and transmits the signal to the counter circuit.

The command is issued to the output port +16, and a sampling cycle is set on the output port +17. The allowable range is 0.1sec - 1056.1sec.

Notice that because all external input signals (with the exception of general-purpose input signals) are directed into the internal counter through the digital filter, they are read with a delay of 4 sampling clock cycles.

In the initial condition, external input signals are read with a delay of 0.4sec.



* Same as the LOW level

Figure 4.17. Digital Filter

Table 4.7. Setting a Digital Filter Clock

D3	D2	D1	D0	Digital filter clock cycle	Input frequency
0	0	0	0	0.1µsec	Approx. 1MHz max.
0	0	0	1	6.5µsec	Approx. 15kHz max.
0	0	1	0	25.7µsec	Approx. 3.5kHz max.
0	0	1	1	32.1µsec	Approx. 3kHz max.
0	1	0	0	204.9µsec	Approx. 480Hz max.
0	1	0	1	211.3µsec	Approx. 470Hz max.
0	1	1	0	230.5µsec	Approx. 430Hz max.
0	1	1	1	236.9µsec	Approx. 420Hz max.
1	0	0	0	819.3µsec	Approx. 122Hz max.
1	0	0	1	825.7µsec	Approx. 121Hz max.
1	0	1	0	844.9µsec	Approx. 118Hz max.
1	0	1	1	851.3µsec	Approx. 117Hz max.
1	1	0	0	1024.1µsec	Approx. 97Hz max.
1	1	0	1	1030.5µsec	Approx. 96Hz max.
1	1	1	0	1049.7µsec	Approx. 95Hz max.
1	1	1	1	1056.1µsec	Approx. 94Hz max.

Precision is approximately ±1/10000 relative to a cycle setting.

Notes!

- *In the initial condition, the clock is set at 0.1µsec (which is the default).*
- *Some noise signals can cause a delay greater than 4 clock cycles.*
- *Any change in level occurring at a frequency faster than a set sampling clock cycle is invalidated and the level is not correctly counted. Therefore, signals less than the input frequency must be entered.*

Count Value Latching (command 14h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
14h	Not used						CH1 LT	CH0 LT

This command latches a count value on a corresponding channel to the READ register. The command issues the value "14h" to the output port +16, and sets a data latch on the output port +17. The command latches the count value by setting the applicable bit to "1". All channels can be latched simultaneously by setting all applicable bits to "1".

Note!

The initial condition is no-latch (00h).

Interrupt Mask (command 15h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
15h	Not used			TIME	Not used		CH1	CH0

This command issues the value "15h" to the output port +16, and sets a mask on the output port +17. The generation of interrupt signals is disabled when an applicable bit is set to "1".

TIME Setting the value "1" disables any interrupt after the time that is set on the programmable timer has elapsed. Setting the value "0" resets the disabled condition.

CH1 to CH0

These bits are associated with the respective channels. Setting the value "1" to any of these bits disables a count-match interrupt on the associated channel. Setting the value "0" resets the disabled condition.

Notes!

- *In the initial condition, all channels are timer-masked (1Fh).*
- *Even in the masked state, on all channels the count match and the timer-up conditions change.*

Sense Reset (command 16h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
16h	Not used			TIME	Not used		CH1	CH0

The interrupt sense sets an applicable bit to "1" when a count-match or timer-up condition is detected on a given channel. When the applicable bit is "1", an interrupt signal is not generated when another count-match or timer-up condition arises. Issuing the value "+16h" to the output port +16 and the value "1" to the applicable bit on the output port clears the sense bit, and enables the generation of another interrupt signal.

TIME Sets the sense bit when the programmable timer is up.

CH1 to CH0

Resets the sense bit when counts match on the respective channels.

Programmable Timer (command timer data: 17h, timer start: 18h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
17h	Programmable timer settings data							

	D7	D6	D5	D4	D3	D2	D1	D0
18h	Not used							START

The programmable timer can generate interrupts in cycles that are compatible with 32-bit settings data. To operate the timer, you need to set 32-bit data and assign the value "1" to the START bit.

The programmable timer can be set by issuing the value "17h" to the output port +16, and by setting timer data on the output port +17. Because it is 32-bit long, the timer data must be set 8 bits at a time sequentially from the lowest bit, for a total of 4 times.
The allowable range is 1msec - 200sec.

The timer starts when the timer start command "18h" is issued to the output port +16, and the applicable bit is set to "1" on the output port +17; setting this bit to "0" stops the timer.

Table 4.8. shows the relationship between programmable timer setup data and timer interrupt cycles. Following are example programs that set the programmable timer to 1 sec.

Microsoft C

```
outp( ADR+16, 0x17 );
outp( ADR+17, 0xff );
outp( ADR+17, 0x2c );
outp( ADR+17, 0x31 );
outp( ADR+17, 0x1 );
outp( ADR+16, 0x18 );
outp( ADR+17, 0x1 );
```

Microsoft QBASIC

```
OUT ADR+16, &H17
OUT ADR+17, &HFF
OUT ADR+17, &H2C
OUT ADR+17, &H31
OUT ADR+17, &H1
OUT ADR+16, &H18
OUT ADR+17, &H1
```

Note!

In the initial condition, the timer is stopped.

Table 4.8. Relationship between Programmable Timer Setup Data and Interrupt Cycles

$$(\underset{\underset{\text{32-bit (converted into decimal)}}{\downarrow}}{\boxed{\text{timer data}}} \text{ x } 50 + 50) \text{ x } 10^{-6} = \text{time interrupt cycle [msec]}$$

Programmable timer-setting data [h]								Timer interrupt cycle
High byte ←							→ Low byte	
0	0	0	0	4	E	1	F	1msec
0	0	0	3	0	D	3	F	10msec
0	0	1	E	8	4	7	F	100msec
0	1	3	1	2	C	F	F	1sec
0	2	6	2	5	9	F	F	2sec
0	3	9	3	8	6	F	F	3sec
0	4	C	4	B	3	F	F	4sec
0	5	F	5	E	0	F	F	5sec
0	7	2	7	0	D	F	F	6sec
0	8	5	8	3	A	F	F	7sec
0	9	8	9	6	7	F	F	8sec
0	A	B	A	9	4	F	F	9sec
0	B	E	B	C	1	F	F	10sec
1	1	E	1	A	2	F	F	15sec
1	7	D	7	8	3	F	F	20sec
2	3	C	3	4	5	F	F	30sec
2	F	A	F	0	7	F	F	40sec
3	B	9	A	C	9	F	F	50sec
7	7	3	5	9	3	F	F	100sec
B	2	D	0	5	D	F	F	150sec
E	E	6	B	2	7	F	F	200sec

Precision is approximately ±1/10000 relative to a cycle setting

One-Shot Pulse (command 19h)

Starting I/O address	D7	D6	D5	D4	D3	D2	D1	D0
19h	One-shot pulse width data							

When there is a match between a count value on a channel and a count compare value, a one-shot pulse is output separately to the channels. The pulse width is common to all channels, and is determined by setup data. The allowable range is 0 - 104.45msec.

This command issues the value +19h" to the output port +16, and sets a pulse width on the output port +17.

The width of a one-shot pulse can be determined according to the following expression:

Table 4.9. Relationship between One-Shot Pulse Setup Data and the Pulse Width

$\boxed{\text{pulse width-setting data}}$ x 409.6 = pulse width [msec]

↓

8-bit (converted into decimal)

	One_shot_pulse width setting data									Pulse width
	D7	←→						D0	[h]	
*	0	0	0	0	0	0	0	0	00h	0
	0	0	0	0	0	0	0	1	01h	Approx. 409.6µsec
	0	0	0	0	0	0	1	0	02h	Approx. 819.2µsec
	0	0	0	0	0	0	1	1	03h	Approx. 1.23msec
	0	0	0	0	1	1	0	0	0Ch	Approx. 4.92msec
	0	0	0	1	1	0	0	1	19h	Approx. 10.24msec
	0	0	1	1	0	0	0	1	31h	Approx. 20.07msec
	0	1	0	0	1	0	0	1	49h	Approx. 29.9msec
	0	1	1	0	0	0	1	0	62h	Approx. 40.14msec
	0	1	1	1	1	0	1	0	7Ah	Approx. 49.97msec
	1	0	0	1	0	0	1	1	93h	Approx. 60.2msec
	1	0	1	0	1	0	1	1	ABh	Approx. 70.04msec
	1	1	0	0	0	1	0	0	C4h	Approx. 80.28msec
	1	1	0	1	1	1	0	0	DCh	Approx. 90.11msec
	1	1	1	1	0	1	0	1	F5h	Approx. 100.35msec
	1	1	1	1	1	1	1	1	FFh	Approx. 104.45msec

*represents the initial state

Notes!

- *The initial condition is pulse width = 0 (no output) (00h).*
- *The pulse width is subject to some variable depending upon connection load specifications.*

Initialization

When the power is turned on or the system is reset, the following initial settings are effected:

Table 4.10. Initial Settings

Item	Initial settings
Operation mode	00h
Z-phase CLR input	04h (enabled only for Z-phase one-time input)
COMPARE register	0
READ register	Not available
24-bit UP/DOWN counter	000000h
Digital filter	00h (0.1msec)
Status data	7Bh (when not connected to an external device)
Count-value latch	00h
Interrupt mask	1Fh (all interrupts disabled)
Sense port	00h
Timer data	00000000h
Timer start	00h (timer stopped)
One-shot pulse	00h (no output)

Examples

2-Phase Pulse Count (no interrupts)

Flowchart

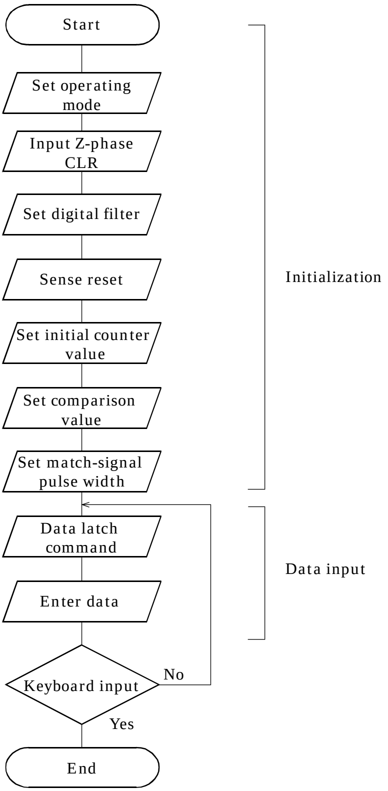


Figure 4.18. 2-Phase Pulse Count (No Interrupts)

Sample Program

```
/*======================================================
Sample program 1

   DEVICE ID:            0
   Mode:                 2-Phase, Asynchronous Clear, Normal Count
   Initial Data:         000000H
   Compare Data:         000064H
   Channel:              0ch
   Programmable Timer:   N/A
   Digital Filter        6.5 used
   Interrupt:            N/A
======================================================*/
#include <stdio.h>
#include <conio.h>

/* ----- Constant ------------------------------------- */
#define  ADR            0x0800            /* I/O address */
#define  CH             2                     /* channel */

/* ----- Declarations --------------------------------- */
struct   REGS08 {                               /* 8-bit */
   unsigned char  lower;                        /* lower */
   unsigned char  middle;                      /* middle */
   unsigned char  upper;                        /* upper */
   unsigned char  reserved;                  /* reserved */
};

struct   REGS24 {                              /* 24-bit */
   unsigned long  count;
};

union ACCESS {
   struct         REGS08      Byte[CH];        /*  8-bit */
   struct         REGS24      Whole[CH];       /* 24-bit */
};

   struct         REGS08      Byte[CH];
   struct         REGS24      Whole[CH];       /* 24-bit */
   union ACCESS   CountData;
   union ACCESS   CompData;

/* ----- Prototype ------------------------------------ */
void  main( void );
void  Initialize( unsigned char );       /* initialize */
void  ReadData( unsigned char, unsigned long *, unsigned char * );
                                          /* read data */

/* ----- Initialize ----------------------------------- */
void  Initialize( unsigned char ch )
{
   outp( ADR+0x10, ch*5+1 );                 /* mode set */
   outp( ADR+0x11, 0x8c );
   outp( ADR+0x10, ch*5+2 );              /* z-pulse set */
   outp( ADR+0x10, ch*5+4 );       /* digital filter set */
   outp( ADR+0x11, 0x1 );
   outp( ADR+0x10, 0x16 );               /* sense reset */
   outp( ADR+0x11, 0xff );
   outp( ADR+0x10, ch*5 );          /* initial data set */
   outp( ADR+0x11, CountData.Byte[ch].lower );  /* lower */
   outp( ADR+0x11, CountData.Byte[ch].middle );/* middle */
   outp( ADR+0x11, CountData.Byte[ch].upper );  /* upper */
   outp( ADR+0x10, ch*5+3 );        /* compare data set */
   outp( ADR+0x11, CompData.Byte[ch].lower );   /* lower */
   outp( ADR+0x11, CompData.Byte[ch].middle ); /* middle */
   outp( ADR+0x11, CompData.Byte[ch].upper );   /* upper */
   outp( ADR+0x10, 0x19 );         /* one-shot pulse set */
   outp( ADR+0x11, 0xff );
}
```

```
/* ----- Read Data ------------------------------------- */
void  ReadData( unsigned char ch, unsigned long *data,
unsigned char *sts )
{
   outp( ADR+0x10, 0x14 );                 /* data latch */
   outp( ADR+0x11, 0xf );
   outp( ADR+0x10, ch*5 );                  /* read data */
   CountData.Byte[ch].lower = (unsigned
char)inp( ADR+0x11 );                           /* lower */
   CountData.Byte[ch].middle = (unsigned
char)inp( ADR+0x11 );                          /* middle */
   CountData.Byte[ch].upper = (unsigned
char)inp( ADR+0x11 );                           /* upper */
   *data = CountData.Whole[ch].count &= 0xffffff;
   outp( ADR+0x10, ch*5+1 );                    /* status*/
   *sts = (unsigned char)inp(ADR+0x11);
}

/* ------ main ----------------------------------------- */
void  main( void )
{
   unsigned char ch, sts;
   unsigned long data;

   ch = 0;
   CountData.Whole[ch].count = 0x000000;   /* count data */
   CompData.Whole[ch].count = 0x000064;  /* compare data */
   Initialize( ch );
   while( !kbhit() ) {
      ReadData( ch, &data, &sts );
      printf("%01dch  %08ld  Status %02x \n", ch, data, sts );
                                              /* display */
   }
}

/* ------------------------------------- End of file --- */
```

2-Phase Pulse Count (with interrupts)

Flowchart

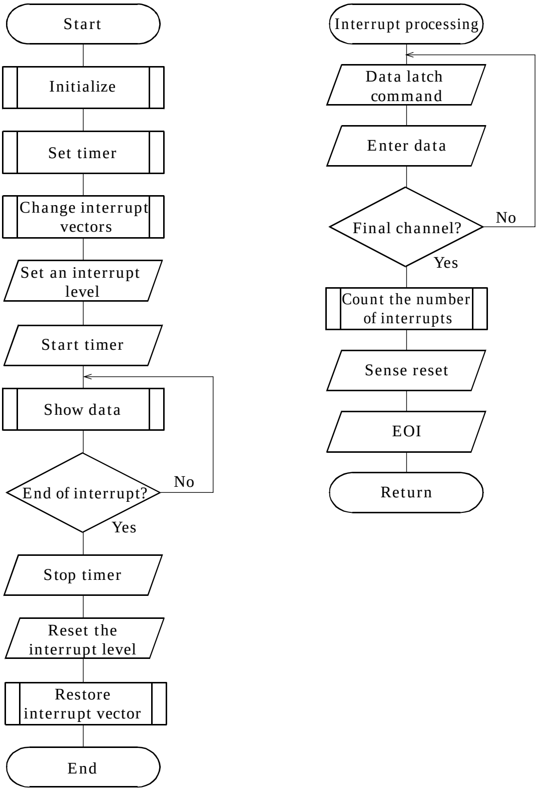


Figure 4.19. 2-Phase Pulse Count (with Interrupts)

Sample Program

```
/* =======================================================
Sample program 2

   DEVICE ID:           0
   Mode:                2-Phase, Asynchronous Clear, Normal Count
   Initial Data:        000000H
   Compare Data:        000064H
   Channel:             0 to 1ch
   Programmable Timer:  1 sec
   Digital Filter       6.5 used
   Interrupt:           IRQ5 10 times

======================================================= */
#include <stdio.h>
#include <conio.h>
#include <dos.h>

/* ----- Constant ------------------------------------ */
#define  ADR         0x0800                 /* I/O address */
#define  CH          2                          /* channel */
#define  IRQ5        0                             /* IRQ5 */
#define  IRQ7        1                             /* IRQ7 */
#define  IRQ9        2                             /* IRQ9 */

/* ----- Declarations -------------------------------- */
struct   REGS08 {                                 /* 8-bit */
      unsigned char  lower;                       /* lower */
      unsigned char  middle;                     /* middle */
      unsigned char  upper;                       /* upper */
      unsigned char  reserved;                 /* reserved */
};

struct   REGS24 {                                /* 24-bit */
      unsigned long  count;
};

union ACCESS {
      struct      REGS08      Byte[CH];           /* 8-bit */
      struct      REGS24      Whole[CH];         /* 24-bit */
};

volatile union    ACCESS      CountData;
volatile union    ACCESS      CompData;
         struct   REGS08      Byte[CH];           /* 8-bit */
         struct   REGS24      Whole[CH];         /* 24-bit */

volatile int      intcnt = 0;     /* interrupt counter */
volatile int      IrqLevel = IRQ5;    /* interrupt level */
         int      OrgMasterImr, OrgSlaveImr;
                                        /* original IMR */
         unsigned char  IntVector[3] = { 0x0d, 0x0f, 0x71 };
                                    /* interruput vector */
         unsigned char  PicMask[3] = { 0xdf, 0x7f, 0xfd };
                                             /* mask bit */
         unsigned char  IsrClear[3] = { 0x65, 0x67, 0x61 };
                                            /* ISR clear */
         unsigned char  IntEnable[3] = { 0x81, 0x82, 0x84 };
                                     /* interrupt enable */

/* ----- Prototype ----------------------------------- */
void  main( void );
void  Initialize( unsigned char );         /* initialize */
void  Interval( void );                     /* timer set */
void  ChgVect( void );                  /* change vector */
void  Display( void );                        /* display */
void  ResVect( void );                 /* restore vector */
void  _interrupt _far inthandler( void );
                                    /* interrupt handler */
void  ( _interrupt _far *OrgVect)();
                             /* original interrupt vector */



/* ----- Initialize ---------------------------------- */
void  Initialize( unsigned char ch )
{
   outp( ADR+0x10, ch*5+1 );                /* mode set */
   outp( ADR+0x11, 0x8c );
   outp( ADR+0x10, ch*5+2 );             /* z-pulse set */
   outp( ADR+0x11, 0x4 );
   outp( ADR+0x10, ch*5+4 );      /* digital filter set */
   outp( ADR+0x11, 0x1 );
   outp( ADR+0x10, 0x16 );               /* sense reset */
   outp( ADR+0x11, 0xff );
   outp( ADR+0x10, ch*5 );          /* initial data set */
   outp( ADR+0x11, CountData.Byte[ch].lower );  /* lower */
   outp( ADR+0x11, CountData.Byte[ch].middle );/* middle */
   outp( ADR+0x11, CountData.Byte[ch].upper );  /* upper */
   outp( ADR+0x10, ch*5+3 );        /* compare data set */
   outp( ADR+0x11, CompData.Byte[ch].lower );   /* lower */
   outp( ADR+0x11, CompData.Byte[ch].middle ); /* middle */
   outp( ADR+0x11, CompData.Byte[ch].upper );   /* upper */
   outp( ADR+0x10, 0x19 );        /* one-shot pulse set */
   outp( ADR+0x11, 0xff );
}

/* ----- timer set ----------------------------------- */
void  Interval( void )
{
   outp( ADR+0x10, 0x15 );                /* mask clear */
   outp( ADR+0x11, 0xf );
   outp( ADR+0x10, 0x17 );                 /* timer set */
   outp( ADR+0x11, 0xff );                     /* 1 sec */
   outp( ADR+0x11, 0x2c );
   outp( ADR+0x11, 0x31 );
   outp( ADR+0x11, 0x1 );
}

/* ----- change vector ------------------------------- */
void  ChgVect( void )
{
   OrgVect = _dos_getvect( IntVector[IrqLevel] );
   _disable();
   _dos_setvect( IntVector[IrqLevel], inthandler );
   if ( IrqLevel > IRQ7 ) {       /* IMR and mask clear */
      outp( 0x21, ( OrgMasterImr = inp( 0x21 ) ) & 0xfb );
      outp( 0xa1, ( OrgSlaveImr = inp( 0xa1 ) ) & PicMask[IrqLevel] );
      outp( 0x20, 0x62 );         /* ISR clear (master) */
      outp( 0xa0, IsrClear[IrqLevel] );
                                   /* ISR clear (slave) */
   } else {                       /* IMR and mask clear */
      outp( 0x21, ( OrgMasterImr = inp( 0x21 ) ) & PicMask[IrqLevel] );
      outp( 0x20, IsrClear[IrqLevel] );    /* ISR clear */
   }
   _enable();                                 /* enable */
}

/* ----- display ------------------------------------- */
void  Display( void )
{
   int           i, j;

   j = intcnt;
   for (i = 0; i < CH; i++) {
      printf("%01dch  %06lXH   ", i, CountData.Whole[i].count );
   }
   printf("interrupt count = %03d \n", intcnt);
}

/* ------ restore vector ----------------------------- */
void  ResVect( void )
{
   _disable();                               /* disable */
   if ( IrqLevel > IRQ7 ) {              /* restore IMR */
      outp( 0x21, OrgMasterImr );
      outp( 0xa1, OrgSlaveImr );
   } else
      outp( 0x21, OrgMasterImr );
   _dos_setvect( IntVector[IrqLevel], OrgVect );
                                     /* restore orgvect */
   _enable();                                 /* enable */
}

/* ----- interrupt handler --------------------------- */
void  _interrupt _far inthandler( void )
```

```
{
   unsigned char i;

   _enable();                                 /* enable */

   for (i = 0; i < CH; i++) {                /* channel */
      outp( ADR+0x10, 0x14 );             /* data latch */
      outp( ADR+0x11, 0xf );
      outp( ADR+0x10, i*5 );               /* read data */
      CountData.Byte[i].lower = (unsigned char)inp( ADR+0x11 );
                                               /* lower */
      CountData.Byte[i].middle = (unsigned char)inp( ADR+0x11 );
                                              /* middle */
      CountData.Byte[i].upper = (unsigned char)inp( ADR+0x11 );
                                               /* upper */
      CountData.Whole[i].count &= 0xffffff;
   }
   intcnt++;                         /* count interrupt */
   outp( ADR+0x10, 0x16 );               /* sense reset */
   outp( ADR+0x11, 0x10 );
   _disable();                               /* disable */

   if ( IrqLevel > IRQ7 ) {                      /* EOI */
      outp( 0xa0, 0x20 );
      outp( 0xa0, 0x0b );
         if ( !inp( 0xa0 ) )   outp( 0x20, 0x20 );
   } else        outp( 0x20, 0x20 );
}

/* ------ main --------------------------------------- */
void  main( void )
{
   unsigned char i;

   for (i = 0; i < CH; i++) {
      CountData.Whole[i].count = 0x000000; /* count data */
      CompData.Whole[i].count = 0x000064;/* compare data */
      Initialize(i);                      /* initialize */
   }

   Interval();
   ChgVect();                          /* change vector */
   outp( ADR+0x2, IntEnable[IrqLevel] );/* interrupt level */
   outp( ADR+0x10, 0x18 );               /* timer start */
   outp( ADR+0x11, 0x1 );

   while( intcnt < 10 )
      Display();                            /* display  */

   Display();                               /* display  */
   outp( ADR+0x10, 0x18 );                /* timer stop */
   outp( ADR+0x11, 0x0 );
   outp( ADR+0x2, 0x0 );             /* interrupt level */
   ResVect();                         /* restore vector */

}

/* ------------------------------------ End of file --- */
```

5. Using the Memory Address Map

When connected to a CPU-CAxx(FIT)GY, the CNT24-2(FIT)GY can be accessed by a host computer through a network. In addition, the Module can be allocated to the memory controlled by the Controller Module according to a given Device ID. Applications running on the host computer control the I/O modules by reading/writing the memory that is controlled by the Controller Module.

For detailed specifications on the memory controlled by the Controller Module, see the Controller Module manual

Following is an explanation of the memory areas necessary for the use of the CNT24-2(FIT)GY: the "module settings area", the "module information area", and the "basic output data area.

Module Settings Area

This area controls the settings and how the module is started.

The module becomes available when the necessary settings are written into this area and the module activation option is set in the [module startup register].

Module Information Area

The current module settings are stored in this area.

When the Module is started, the contents of the Module Settings Area are copied to the Module Information Area. By reading this area, you can verify the current module settings.

Basic Input Data Area

Basic input data is read in this area.

Basic Output Data Area

Basic output data is written in this area.

Module Settings Area

The module settings area is a 128-byte (80h) area beginning at address 301000h, and corresponds to a given Device ID.

The starting address can be determined according to the following expression:

Starting address = 301000h + 80h × (Device ID)

Table 5.1. Module Settings Area ***< 1 / 2 >***

Address(h)	Area	Item	Size	Access type	Initial value(h)	Initial settings
Starting address+00	Module-specific information	Module type (category)	1	R	03	CNT24-2(FIT)GY
Starting address+01		Module type (serial No.)	1	R	00	
Starting address+02		System-reserved (revision No.)	1	R	None	
Starting address+03		Supported functions	1	R	03	Basic input
Starting address+04		Number of basic input channels	1	R	02	2channels
Starting address+05		Basic input data size	1	R	08	8bytes
Starting address+06		Number of basic output channels	1	R	02	2channels
Starting address+07		Basic output data size	1	R	08	8bytes
Starting address+08		Input channel settings address	1	R	20	20h
Starting address+09		Input channel settings data size	1	R	18	24bytes
Starting address+0A		Output channel settings address	1	R	20	20h
Starting address+0B		Output channel settings data size	1	R	18	24bytes
Starting address+0C - Starting address+0F		Reserved	4	R	None	
Starting address+10	Common to modules	Module startup register	1	R/W	00	
Starting address+11		Error status	1	R	00	
Starting address+12		Counter resolution	1	R	18	24bits counter resolution (fixed)
Starting address+13		One-shot pulse settings	1	R/W	00	One-shot pulse output off
Starting address+14 - Starting address+17		One-shot pulse width	4	R/W	00000000	
Starting address+18		Start registers by channels	1	R/W	00	
Starting address+19		Start register mask configuration by channels	1	R/W	00	
Starting address+1A - Starting address+1F		Reserved	6	R	None	

Table 5.1. Module Settings Area *< 2 / 2 >*

Address(h)	Area	Item		Size	Access type	Initial value(h)	Initial settings
Starting address+20		CH0	Setting the channel reset operation	1	R/W	01	Channel reset operation enabled
Starting address+21			Setting an initial count	1	R/W	01	Put the initial count into effect
Starting address+22 - Starting address+25			Initial count	4	R/W	00000000	Initial count = 00000000h
Starting address+26			Pulse input mode	1	R	00	Opto-isolation
Starting address+27			Digital filter settings	1	R/W	00	Do not use a digital filter
Starting address+28 - Starting address+29			Digital filter value	2	R/W	00	1MHz
Starting address+2A			Function	1	R/W	01	2-phase counter
Starting address+2B			Counter input multiplier	1	R/W	00	1x
Starting address+2C			Setting the Clear Signal Synchronization	1	R/W	01	Synchronous clear
Starting address+2D			Counting direction	1	R/W	00	Phase A down/counterclockwise
Starting address+2E			Clear Signal Input Logic	1	R/W	00	Positive logic (HIGH active)
Starting address+2F			Setting the Clear Signal Operation	1	R/W	02	Enabled only once for the next clear signal input
Starting address+30 - Starting address+33			Preset count values	4	R/W	00000000	
Starting address+34 - Starting address+37			Reserved	4	R	None	
Starting address+38	Channel settings	CH1	Setting the channel reset operation	1	R/W	01	Channel reset operation enabled
Starting address+39			Setting an initial count	1	R/W	01	Put the initial count into effect
Starting address+3A - Starting address+3D			Initial count	4	R/W	00000000	Initial count = 00000000h
Starting address+3E			Pulse input mode	1	R	00	Opto-isolation
Starting address+3F			Digital filter settings	1	R/W	00	Do not use a digital filter
Starting address+40 - Starting address+41			Digital filter value	2	R/W	00	1MHz
Starting address+42			Function	1	R/W	01	2-phase counter
Starting address+43			Counter input multiplier	1	R/W	00	1x
Starting address+44			Setting the Clear Signal Synchronization	1	R/W	01	Synchronous clear
Starting address+45			Counting direction	1	R/W	00	Phase A down/counterclockwise
Starting address+46			Clear Signal Input Logic	1	R/W	00	Positive logic (HIGH active)
Starting address+47			Setting the Clear Signal Operation	1	R/W	02	Enabled only once for the next clear signal input
Starting address+48 - Starting address+4B			Preset count values	4	R/W	00000000	
Starting address+4C - Starting address+4F			Reserved	4	R	None	
Starting address+50 - Starting address+7F		Reserved		48	R	None	

Module-Specific Information

-Module type (category)
The CNT24-2(FIT)GY belongs to the counter module category (03h).

-Module type (serial No.)
The CNT24-2(FIT)GY is a counter module with a serial No. 0 (00h).

-Supported functions
The CNT24-2(FIT)GY supports the basic input function and the basic output function (03h).
Basic input data comprises a count value, a comparison match detected, abnormal input detected, and general-purpose input.
Basic output data comprises a count comparison value, compare-match detection reset, and abnormal input detection reset.

-Number of basic input channels
The CNT24-2(FIT)GY has 2 (02h) basic input channels.
Two counter channels are provided.

-Basic input data size
In the CNT24-2(FIT)GY, the size of basic input data is 8 bytes (08h).
A count value requires 4 bytes, and the comparison match detected, abnormal input detected, general-purpose input, and reserves areas require 1 byte each.

-Number of basic output channels
The CNT24-2(FIT)GY has 2 (02h) basic output channels.
Two counter channels are provided.

-Basic output data size
In the CNT24-2(FIT)GY, the size of basic output data is 8 bytes (08h).
A count comparison value requires 4 bytes; the compare-match detection reset, and abnormal input detection reset areas require 1 byte each; and the reserved area requires 2 bytes.

-Input channel settings address
This item indicates the address of the area in which input channel settings are stored in terms of an offset address within the module settings area.
In the CNT24-2(FIT)GY, input channel settings are stored beginning with "20h".

-Input channel settings data size
Indicates the data size of the area in which input channel settings are stored.
In the CNT24-2(FIT)GY, the data size of input channel settings is 24 bytes (18h).

-Output channel settings address
This item indicates the address of the area in which output channel settings are stored in terms of an offset address within the module settings area.
In the CNT24-2(FIT)GY, both input channel and output channel are stored beginning with "20h".

-Output channel settings data size
Indicates the data size of the area in which output channel settings are stored.
In the CNT24-2(FIT)GY, the data size of output channel settings is 24 bytes (18h).

Items Common to Modules

-Module startup register

Setting the module startup option (01h) causes the device module to be started. Setting the module startup option when the module is being started causes the module to be restarted.

The CNT24-2(FIT)GY does not contain a module shutdown function.

00h: No operation

01h: Module startup

-Error status

The error status bits, which are not reflected in the module settings area, always remain [00h].

The error status on a module is stored in the module information area.

-Counter resolution

In the CNT24-2(FIT)GY, the counter resolution is fixed at 24 bits (18h).

-One-shot pulse settings

The CNT24-2(FIT)GY can produce one-shot pulse output of a count-match signal when a count value is equal to a count comparison value. The comparison between a count value and a count comparison value is performed on a channel-by-channel basis.

The use of the one-shot pulse option is common to all modules; it cannot be set on a channel-by-channel basis.

00h: One-shot pulse on

01h: One-shot pulse off

-One-shot pulse width

This item sets the width of one-shot pulse output for count-match signals.

One-shot-pulse width (h)	Pulse width
00000000	0 (Output Off)
0000019A	Approx. 409.6µsec
000004CD	Approx. 1.23msec
00002800	Approx. 10.24msec
00019800	Approx. 104.45msec

One-shot pulse widths are common to all modules; they cannot be set on a channel-by-channel basis.

Note!

The pulse width is subject to some error depending upon the connected load.

- Start registers by channels / Start register mask configuration by channels

	D7	D6	D5	D4	D3	D2	D1	D0
+18h	-	-	-	-	-	-	CH1	CH0
+19h	-	-	-	-	-	-	CH1	CH0

Start registers by channels (+18h)

CH1 to CH0 Turns ON/OFF for each channel.

0 : Stop

1 : Start

If started from the module start register, each channel is enabled (1 is turned on). If any one of the channels is ON, the module start register is also made active.

Start register mask configuration by channels (+19h)

The corresponding bit 1 reflects the channel for the start register value by channel.

0 : Not processed.

1 : Start register value by channel is reflected.

Channel settings

-Setting the channel reset operation

This item sets the channel reset operation that is to be performed when the module is started.

The channel reset operation clears the count value to "00000000h" and clears the comparison match detection and the abnormal input detection to "00h". If the clear-operation setting is "enabled only once", the next clear signal input will be valid once.

During the channel reset operation, the operation of the counter pauses, and the counting process is restarted from the initial condition after the channel settings are put into effect.

00h: Channel reset operation disabled
01h: Channel reset operation enabled

Note!

When modifying channel settings, you should perform a channel reset operation so that the new settings will be safely put into effect.

-Setting an initial count

This item puts the initial count into effect when the module is started.

00h: Do not put the initial count into effect.
01h: Put the initial count into effect.

-Initial count

The initial count is stored in Little Endian.

	D7	D6	D5	D4	D3	D2	D1	D0
+0h	C7	C6	C5	C4	C3	C2	C1	C0
+1h	C15	C14	C13	C12	C11	C10	C9	C8
+2h	C23	C22	C21	C20	C19	C18	C17	C16
+3h	0	0	0	0	0	0	0	0

-Pulse input mode

The pulse input mode is fixed at opto-isolated input (00h).

-Digital filter settings

This item specifies the use of a digital filter.

00h: Do not use a digital filter.
01h: Use a digital filter.

-Digital filter value

This item sets a digital filter value.

Digital filter value (h)	Input frequency	Sampling clock cycle
0000	Disabled digital filter (and above)	0.1μsec
005E	94Hz and above	1056.1μsec
01A4	420Hz and above	236.9μsec
0BB8	3kHz and above	32.1μsec
3A98	15kHz and above	6.5μsec

Precision: ±1/10000 relative to the cycle settings

The digital filter is designed to cut noise, such as the chattering of A, B, and Z-phase input signals (a digital filter for generic input signals is not provided). The value of a digital filter should be selected by considering the frequency of input signals. Digital filter values determine the sampling clock cycle for the digital filter. When detecting 4 clocks of continuous HIGH (or LOW) signals by sampling the input signals with this sampling clock, the digital filter outputs a HIGH (or a LOW), and transmits the signal to the counter circuit. As a result, external input signals are read with a delay of 4 sampling clock cycles.

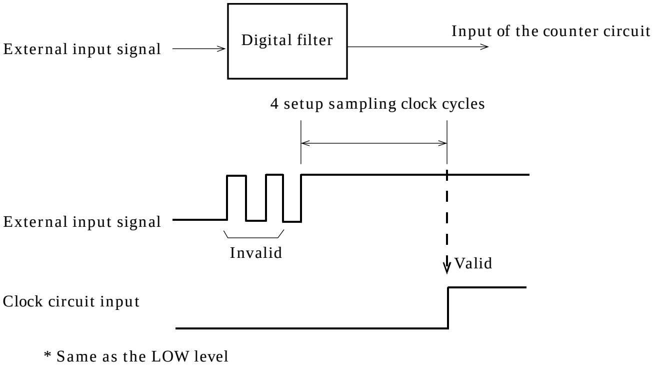


Figure 5.1. Digital Filter

Function

This item sets a counter function.

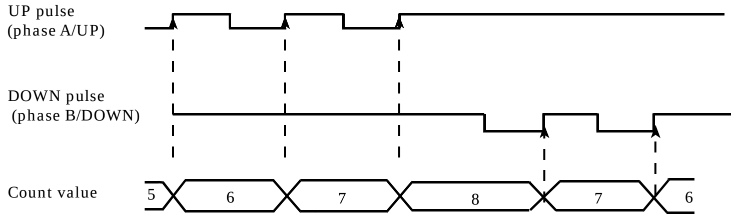
00h: Mono-phase counter

01h: 2-phase counter

02h: Mono-phase counter with gate control

-Mono-phase counter

In the case of the mono-phase counter, the system counts up upon the input of an UP pulse, and counts down upon the input of a DOWN pulse. The counting process fails if UP and DOWN pulses occur simultaneously or both pulses happen to be LOW.



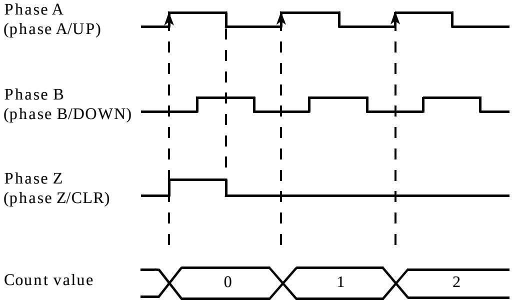
* The figure above illustrates the counting operation when a count direction = 01h is specified.
If count direction = 00h, a DOWN count commences when phase A pulse rises, and an UP count commences when a phase B pulse rises.

Figure 5.2. Mono-Phase Counter

-2-phase counter

The 2-phase counter counts the 2-phase pulse inputs of phases A and B that have a 90° phase difference. This option is principally used to count pulse inputs from an incremental rotary encoder.

If both phases A and B change simultaneously, the condition is treated as an input error and cannot successfully be counted.

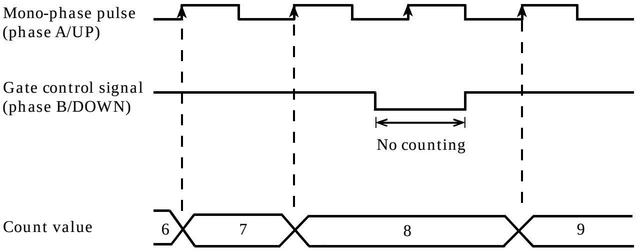


* The figure above illustrates the counting operation when a count direction = 01h is specified.
If count direction = 00h, a DOWN count commences when phase A rises.

Figure 5.3. 2-Phase Counter

-Mono-phase counter with gate control

This option can start/stop the counting of mono-phase pulse strings according to gate control signals.



* The figure above illustrates the counting operation when a count direction = 01h is specified.
If count direction = 00h, a DOWN count commences when the gate control signal (phase B/DOWN) is HIGH and when a mono-phase pulse example (phase A/UP) rises; the counting stops when the gate control signal is LOW.

Figure 5.4. Example of a Mono-Phase Counter with Gate Control

Different functions require different counter input multipliers and clear signal synchronization settings.

Function	Counter input multiplier			Clear signal synchronization settings	
	1x	2x	4x	Asynchronous clear	Synchronous clear
Mono-phase counter	O	×	×	O	×
2-phase counter	O	O	O	O	O
Mono-phase counter with gate control	O	O	×	O	×

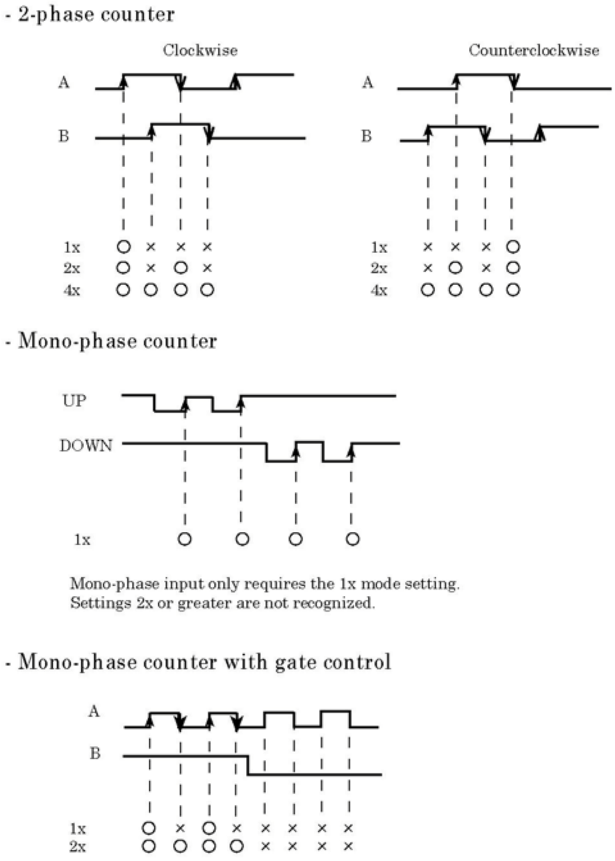
Note!

If invalid functions, counter input multipliers, or clear signal synchronization settings, a module startup error results, and the module cannot be started.

Counter input multiplier

This item sets the counter multiplication processing.

00h: 1x
01h: 2x
02h: 4x



Mono-phase input only requires the 1x mode setting.
Settings 2x or greater are not recognized.

Figure 5.5. Example of a Counting Operation with a Count Input Multiplier Set

Setting the Clear Signal Synchronization

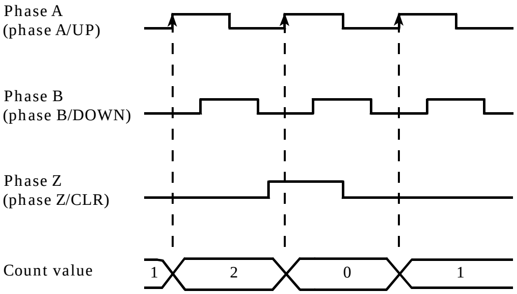
This item sets the synchronization of the count value clear operation and phase A signals based on clear signals (phase Z).

00h: Asynchronous clear

01h: Synchronous clear

-Synchronous clear

If counting direction =01h and clear signal input logic =00h, the counter is zero-cleared when A phase rises with the B-phase input LOW an the Z-phase input HIGH; the counting process is started when A phase rises after the Z-phase input turns LOW.

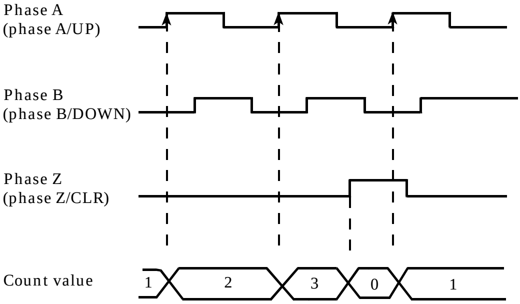


* If count direction = 00h, a DOWN count commences when phase A rises with phase B being LOW; if clear signal input logic = 01h, the operation is enabled with the phase Z input is LOW.

Figure 5.6. **Example of a Synchronous Clear Counting Operation**

-Asynchronous clear

If counting direction =01h and clear signal input logic =00h, the counter is zero-cleared when Z phase turns HIGH, irrespective of the input state of phase A or B. The counting process is started at the next rise of the A phase, irrespective of the input state of Z phase.



* If count direction = 00h, a DOWN count commences when phase A rises with phase B being LOW; if clear signal input logic = 01h, the operation is enabled with the phase Z input is LOW.

Figure 5.7. **Example of an Asynchronous Clear Counting Operation**

Counting direction

This item specifies the direction in which counting is to be performed.

00h: Phase A down/counterclockwise

01h: Phase A up/clockwise

Counting direction	Mono-phase counter	2-phase counter	Mono-phase counter with gate control
Phase A DOWN /counterclockwise	Counts DOWN on phase A input; counts UP on phase B input.	Counts DOWN when the rotary encoder rotates clockwise.	Counts DOWN on phase A input with phase B being HIGH.
Phase A UP /clockwise	Counts UP on phase A input; counts DOWN on phase B input.	Counts UP when the rotary encoder rotates clockwise.	Counts UP on phase A input with phase B being HIGH.

Clear Signal Input Logic

Selecting clear signal (phase Z) input logic

00h: Positive logic (HIGH active)

01h: Negative logic (LOW active)

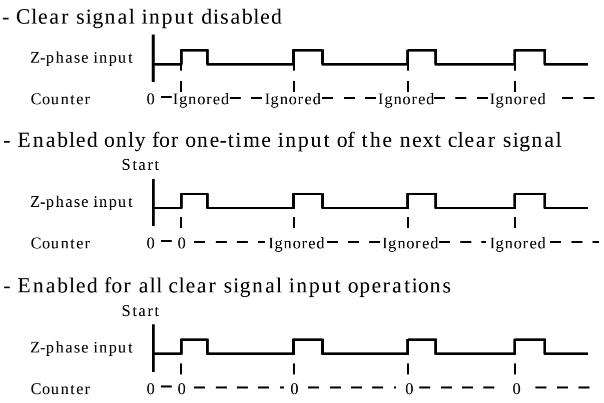
Setting the Clear Signal Operation

This item specifies the operation of clear signal (phase Z) input

01h: Clear signal input disabled

02h: Enabled only once for the next clear signal input

03h: Enabled for all clear signal input operations



- Clear signal input disabled

- Enabled only for one-time input of the next clear signal

- Enabled for all clear signal input operations

Figure 5.8. Setting a Clear Signal Operation (Clear Signal Input Logic = 00h)

Note!

When not using the clear signal (phase Z) option, choose the "Clear signal input disabled" setting.

Preset count values

Entering the value allows the counter value to be set as a preset value.

The function works only when the value is set in this area only.

Module Information Area

The module information area is a 128-byte area beginning with address 300000h and corresponding to a given Device ID.

The starting address can be determined according to the following expression:

Starting address = 300000h + 80h × (Device ID)

Table 5.2. Module Information Area ***< 1 / 2 >***

Address(h)	Area	Item	Size	Access type	Initial value (h)
Starting address+00	Module-specific information	Module type (category)	1	R	03
Starting address+01		Module type (serial No.)	1	R	00
Starting address+02		System-reserved (revision No.)	1	R	None
Starting address+03		Supported functions	1	R	03
Starting address+04		Number of basic input channels	1	R	02
Starting address+05		Basic input data size	1	R	08
Starting address+06		Number of basic output channels	1	R	02
Starting address+07		Basic output data size	1	R	08
Starting address+08		Input channel settings address	1	R	20
Starting address+09		Input channel settings data size	1	R	18
Starting address+0A		Output channel settings address	1	R	20
Starting address+0B		Output channel settings data size	1	R	18
Starting address+0C - Starting address+0F		Reserved	4	R	None
Starting address+10	Common to modules	Module startup register	1	R/W	00
Starting address+11		Error status	1	R	00
Starting address+12		Counter resolution	1	R	18
Starting address+13		One-shot pulse settings	1	R/W	00
Starting address+14 - Starting address+17		One-shot pulse width	4	R/W	00000000
Starting address+18		Start registers by channels	1	R/W	00
Starting address+19		Start register mask configuration by channels	1	R/W	00
Starting address+1A - Starting address+1F		Reserved	6	R	None

Table 5.2. Module Information Area *< 2 / 2 >*

Address(h)	Area	Item		Size	Access type	Initial value (h)
Starting address+20	Channel settings	CH0	Setting the channel reset operation	1	R	01
Starting address+21			Setting an initial count	1	R	01
Starting address+22 - Starting address+25			Initial count	4	R	00000000
Starting address+26			Pulse input mode	1	R	00
Starting address+27			Digital filter settings	1	R	00
Starting address+28 - Starting address+29			Digital filter value	2	R	00
Starting address+2A			Function	1	R	01
Starting address+2B			Counter input multiplier	1	R	00
Starting address+2C			Setting the Clear Signal Synchronization	1	R	01
Starting address+2D			Counting direction	1	R	00
Starting address+2E			Clear Signal Input Logic	1	R	00
Starting address+2F			Setting the Clear Signal Operation	1	R	02
Starting address+30 - Starting address+33			Preset count value	4	R	00000000
Starting address+34 - Starting address+37			Reserved	4	R	None
Starting address+38		CH1	Setting the channel reset operation	1	R	01
Starting address+39			Setting an initial count	1	R	01
Starting address+3A - Starting address+3D			Initial count	4	R	00000000
Starting address+3E			Pulse input mode	1	R	00
Starting address+3F			Digital filter settings	1	R	00
Starting address+40 - Starting address+41			Digital filter value	2	R	00
Starting address+42			Function	1	R	01
Starting address+43			Counter input multiplier	1	R	00
Starting address+44			Setting the Clear Signal Synchronization	1	R	01
Starting address+45			Counting direction	1	R	00
Starting address+46			Clear Signal Input Logic	1	R	00
Starting address+47			Setting the Clear Signal Operation	1	R	02
Starting address+48 - Starting address+4B			Preset count value	4	R	00000000
Starting address+4C - Starting address+4F			Reserved	4	R	None
Starting address+50 - Starting address+7F		Reserved		48	R	None

When the module is started, the contents of the module settings area are stored in the module information area, with the exception of the [Module Startup Register] and the [Error Status].

-Module startup register

This register holds the module operating status.

The CNT24-2(FIT)GY does not contain a module shutdown function. Therefore, the fact that the module is shut down simply indicates that the module has not been started.

00h: Module shutdown

01h: Module operating

-Error status

This register stores the error status of the module.

The error status register is reset when the module is restarted.

00h: Normal

32h: Conflicting settings

Conflicting setting (32h) is an error status that indicates that the module was started using settings that are invalid for the

CNT24-2(FIT)GY. When a setting conflict is generated, the module remains shut down. To resolve the problem, see the settings in "Function", "Counter Input Multipliers", and "Clear Signal Synchronization Settings".

Basic Input Data Area

The basic input data area, which is a 128-byte area beginning with address 304000h, corresponds to a given Device ID.

The starting address can be determined according to the following expression:

Starting address = 304000h + 80h × (Device ID)

Table 5.3. Basic Input Data Area

Address(h)	Area	Item	Size	Access type
Starting address+00 - Starting address+03	CH0	Counter value	4	R
Starting address+04		Compare-match detected	1	R
Starting address+05		Abnormal input detected	1	R
Starting address+06		General-purpose input	1	R
Starting address+07		Reserved	1	R
Starting address+08 - Starting address+0B	CH1	Counter value	4	R
Starting address+0C		Compare-match detected	1	R
Starting address+0D		Abnormal input detected	1	R
Starting address+0E		General-purpose input	1	R
Starting address+0F		Reserved	1	R
Starting address+10 - Starting address+7F	Reserved		112	R

-Count value

A count is stored in Little Endians.

	D7	D6	D5	D4	D3	D2	D1	D0
+0h	C7	C6	C5	C4	C3	C2	C1	C0
+1h	C15	C14	C13	C12	C11	C10	C9	C8
+2h	C23	C22	C21	C20	C19	C18	C17	C16
+3h	0	0	0	0	0	0	0	0

Count values are stored simultaneously in 4 bytes.

Notes!

- *When the module is running, valid data is stored as a count value. When the module is stopped, the count value is undefined.*
- *The count value is 4 bytes per channel. To maintain data integrity, 4 bytes of data should be loaded per READ operation.*

-Compare-match detected

When a match between a count value and a count comparison value is detected, the indicator "comparison match detected" is stored.

To reset this condition, either output a value to the "comparison match detection reset" bit or perform a "channel reset" operation.

00h: No comparison match detected
01h: Comparison match detected

-Abnormal input detected

When abnormal input is detected, the system stores an "abnormal input detected" value.

In the case of a mono-phase counter, the simultaneous input of UP and DOWN pulses causes the detection of abnormal input.

In the case of a 2-phase counter, if both phases A and B change simultaneously, abnormal input is detected as an abnormal state transition.

To reset this condition, either output a value to the "abnormal input detection reset" bit or perform a "channel reset" operation.

00h: No abnormal input detected
01h: Abnormal input detected

-General-purpose input

Indicates the status of general-purpose input.

00h: General-purpose input status "LOW"
01h: General-purpose input status "HIGH"

Basic Output Data Area

The basic output data area, which is a 128-byte area beginning with address 305000h, corresponds to a given Device ID.

The starting address can be determined according to the following expression:

Starting address = 305000h + 80h × (Device ID)

Table 5.4. Basic Output Data Area

Address(h)	Area	Item	Size	Access type
Starting address+00 - Starting address+03	CH0	Count value	4	R/W
Starting address+04		Compare-match detected reset	1	W
Starting address+05		Abnormal input detected reset	1	W
Starting address+06 - Starting address+07		Reserved	2	None
Starting address+08 - Starting address+0B		Count comparison value	4	R/W
Starting address+0C	CH1	Compare-match detected reset	1	W
Starting address+0D		Abnormal input detected reset	1	W
Starting address+0E - Starting address+0F		Reserved	2	None
Starting address+10 - Starting address+7F		General-purpose input	112	None

-Count comparison value

Stores a count comparison value in a Little Endian.

	D7	D6	D5	D4	D3	D2	D1	D0
+0h	C7	C6	C5	C4	C3	C2	C1	C0
+1h	C15	C14	C13	C12	C11	C10	C9	C8
+2h	C23	C22	C21	C20	C19	C18	C17	C16
+3h	0	0	0	0	0	0	0	0

-Comparison-match detection reset

Resets the comparison match detection status.

00h: No operation

01h: Resets comparison-match detection.

-Abnormal input detection reset

Resets the abnormal input detection status.

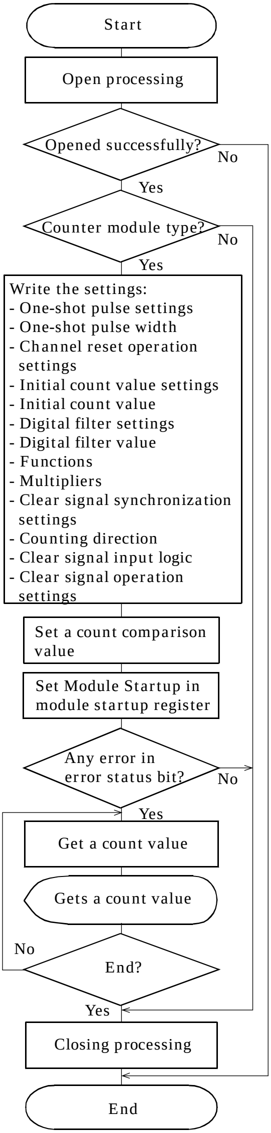
00h: No operation

01h: Resets the abnormal input detection.

Examples

Flowchart

The example below illustrates the case where the CNT24-2(FIT)GY is installed at Device ID: 0.


Start
Open processing
Opened successfully?
No
Yes
Counter module type?
No
Yes
Write the settings:
- One-shot pulse settings
- One-shot pulse width
- Channel reset operation settings
- Initial count value settings
- Initial count value
- Digital filter settings
- Digital filter value
- Functions
- Multipliers
- Clear signal synchronization settings
- Counting direction
- Clear signal input logic
- Clear signal operation settings
Set a count comparison value
Set Module Startup in module startup register
Any error in error status bit?
No
Yes
Get a count value
Gets a count value
No
End?
Yes
Closing processing
End

Determine category by reading address:301000h.
For counter-type module, category is "03h".

Specify counter operation-related settings.
There are common settings and channel-specific settings.
Write any values that must be changed from the initial settings.
Any values that are used in their initial settings need not be overwritten.
Example; Setting a CH0 function to the 2-phase counter:
Write "01h" to address: 30102Ah.

When using the count comparison match function, write a count comparison value.
Example: To write 1000 (decimal) as a CH0 comparison value:
writing "01h" to address: 301010h start the module.

Write "00000003E8h" at the 4th byte from address: 305000h.
The new value takes effect when the module is started so that a count value can be obtained.

Error status can be checked by reading address: 300011h.
If there is a setting problem, the error status will be "32h"; otherwise, it is "00h".

A CH0 count value can be obtained by reading 4 bytes from address: 304000h.
A CH1 count value can be obtained by reading 4 bytes from address: 304008h.

Sample Program

```
/*======================================================
F&eIT I/F Sample Program

   DEVICE ID:        0
   Channel:          0ch
====================================================== */
#include <windows.h>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include "Fit.h"

/* Address(common) */
#define FIT_IO                          (0x00300000)
#define FIT_IO_DEVICE_INFOR             (0x0000)
#define FIT_IO_DEVICE_CONFIG            (0x1000)
#define FIT_IO_INPUT                    (0x4000)
#define FIT_IO_OUTPUT                   (0x5000)

#define FIT_IO_DEVICE_SIZE              (0x0080)

#define FIT_PRODUCT_CATEGORY            (0x00)

#define FIT_MODULE_START                (0x10)
#define FIT_ERROR_STATUS                (0x11)

/* Information(Common) */
#define FIT_PRODUCT_DIGITAL             (0x01)
#define FIT_PRODUCT_ANALOG              (0x02)
#define FIT_PRODUCT_COUNTER             (0x03)

#define FIT_MODULE_START_OFF            (0x00)
#define FIT_MODULE_START_ON             (0x01)

/* Address(CNT) */
#define FIT_CNT_BIT                     (0x12)
#define FIT_CNT_OUTPUT_SIGNAL           (0x13)
#define FIT_CNT_OUTPUT_WIDTH            (0x14)

#define FIT_CNT_CH_RESET                (0x00)
#define FIT_CNT_INITIAL_MODE            (0x01)
#define FIT_CNT_INITIAL_DATA            (0x02)
#define FIT_CNT_INPUT_MODE              (0x06)
#define FIT_CNT_FILTER_SET              (0x07)
#define FIT_CNT_FILTER_VALUE            (0x08)
#define FIT_CNT_FUNCTION                (0x0A)
#define FIT_CNT_RESOLUTION              (0x0B)
#define FIT_CNT_CLEAR_MODE              (0x0C)
#define FIT_CNT_DIRECTION               (0x0D)
#define FIT_CNT_CLEAR_LOGIC             (0x0E)
#define FIT_CNT_CLEAR_INPUT             (0x0F)

/* Sample */
#define  FIT_SAMPLE_IP_ADDRESS          Note!"192.168.132.211"
#define FIT_SAMPLE_PORT                 (0x5007)
#define FIT_SAMPLE_DEVICE_ID            (0)
#define FIT_SAMPLE_CH                   (0)

int main(void)
{
   DWORD dwIpAddress;
   DWORD dwVaBase;
   DWORD dwVaOffset;
   DWORD dwVaChOffset;
   WORD  hHandle;
   WORD  wStatus;
   BYTE  byCategory;
   BYTE  byModuleStart;
   BYTE  byData[0x80];
   BYTE  byErrorStatus;
   /* Open */
      dwIpAddress = FIT_IpChenge((BYTE 
*)FIT_SAMPLE_IP_ADDRESS);
      hHandle = FIT_Open((BYTE *)&dwIpAddress, FIT_SAMPLE_PORT, 
NULL);
      if (hHandle == 0) {
              printf("Error! FIT_Open = %04X(H)\n", 
hHandle);
              return 1;
      }

   /* Offset Address */
   dwVaOffset = FIT_IO_DEVICE_SIZE * FIT_SAMPLE_DEVICE_ID;

   /* Read 'Category' */
   dwVaBase = FIT_IO + FIT_IO_DEVICE_CONFIG;
   wStatus = FIT_Read(hHandle, dwVaBase + dwVaOffset + 
FIT_PRODUCT_CATEGORY, 1, &byCategory);
   if (wStatus != 0) {
      printf("Error! FIT_Read = %04X(H)\n", wStatus);
      FIT_Close(hHandle);
      return 1;
   }
   if (byCategory != FIT_PRODUCT_COUNTER) {
      printf("Error! Category = %02X(H)\n", byCategory);
      FIT_Close(hHandle);
      return 1;
   }

   /* Channel Offset */
   dwVaChOffset = 0x20 + 0x18 * FIT_SAMPLE_CH;

   /* Read 'Channel Configuration' */
   wStatus = FIT_Read(hHandle, dwVaBase + dwVaOffset + 
dwVaChOffset, 0x18, &byData[0]);
   if (wStatus != 0) {
      printf("Error! FIT_Read = %04X(H)\n", wStatus);
   }

   /* Configuration Data Set */
   byData[FIT_CNT_FUNCTION] = 0x01;         /* Two Phase */

   /* Write 'Channel Configuration' */
   wStatus = FIT_Write(hHandle, dwVaBase + dwVaOffset + 
dwVaChOffset, 0x18, &byData[0]);
   if (wStatus != 0) {
      printf("Error! FIT_Write = %04X(H)\n", wStatus);
   }

   /* Write 'Compare Data' */
   dwVaBase = FIT_IO + FIT_IO_OUTPUT;
   dwVaChOffset = 8 * FIT_SAMPLE_CH;
   byData[0] = 0xE8;
   byData[1] = 0x03;
   byData[2] = 0x00;
   byData[3] = 0x00;
   wStatus = FIT_Write(hHandle, dwVaBase + dwVaOffset + 
dwVaChOffset, 4, (BYTE *)&byData[0]);
   if (wStatus != 0) {
      printf("Error! FIT_Write = %04X(H)\n", wStatus);
   }

   /* Write 'Module Start' */
   dwVaBase = FIT_IO + FIT_IO_DEVICE_CONFIG;
   byModuleStart = FIT_MODULE_START_ON;
   wStatus = FIT_Write(hHandle, dwVaBase + dwVaOffset + 
FIT_MODULE_START, 1, &byModuleStart);
   if (wStatus != 0) {
      printf("Error! FIT_Write = %04X(H)\n", wStatus);
   }
```

```
    /* Read 'Error Status' */
    dwVaBase = FIT_IO + FIT_IO_DEVICE_INFOR;
    wStatus = FIT_Read(hHandle, dwVaBase + dwVaOffset +
FIT_ERROR_STATUS, 1, &byErrorStatus);
    if (wStatus != 0) {
        printf("Error! FIT_Read = %04X(H)\n", wStatus);
        FIT_Close(hHandle);
        return 1;
    }
    if (byErrorStatus != 0x00) {
        printf("Error! Error Status = %02X(H)\n", byErrorStatus);
        FIT_Close(hHandle);
        return 1;
    }

    /* Read 'Counter Data' */
    dwVaBase = FIT_IO + FIT_IO_INPUT;
    printf("- Hit Any Key:Stop -\n");
    while (!kbhit()) {
        wStatus = FIT_Read(hHandle, dwVaBase + dwVaOffset +
dwVaChOffset, 8, (BYTE *)&byData[0]);
        if (wStatus != 0) {
            printf("Error! FIT_Read = %04X(H)\n", wStatus);
        }
        printf("Counter CH%d Data:%02X%02X%02X%02X\r",
            FIT_SAMPLE_CH, byData[3], byData[2], byData[1], byData[0]);
    }

    /* Close */
    FIT_Close(hHandle);

    return 0;
}
```

6. System Reference

Block Diagram

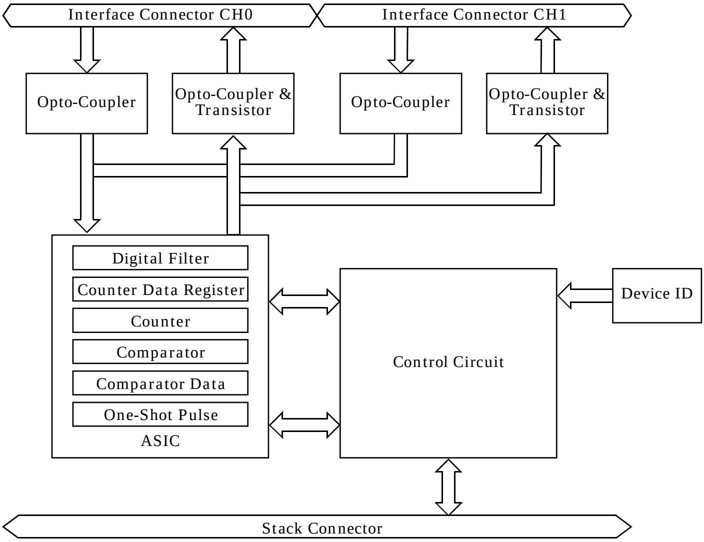


Figure 6.1. Circuit Block Diagram

Specifications

Table 6.1. Specifications

Item	Specification
Counter input section	
Channel count	2
Counting system	Up/down counting
Max. count	FFFFFFH (binary data)
Input type	Opto-isolated input (for current sinking output)
Input signal	Phase A/UP Phase B/DOWN Phase Z/CLR General-purpose input
Internal resistance	220Ω and above
Input ON current	15 - 25mA
Input protection circuit	Not avilable
Response frequency	500kHz duty 50% (Max.)
External power	5V - 12VDC ±10% 400mA (Min.)
Digital filter	0.1μsec - 105.6μsec
Interrupt	IRQ5 or IRQ7 or IRQ9 *1
Programmable timer	1msec - 200sec *1
Match-signal output section *2	
Output count	One x 2 channels
Output type	Opto-isolated open collector output (current sinking type) (negative logic)
Rated output	35VDC 50mA (Max.)
Pulse width	0 - 104.45msec
Output signal width	Not avilable
External power	5V - 12VDC ±10%
Common section	
Internal power consumption	5VDC±5% 150mA (Max.)
Maximum distance of signal extension	30m
External dimensions (mm)	25.2 (W) x 64.7 (D) x 94.0 (H) (exclusive of protrusions)
Weight (module itself)	100g
Module connection method	Stack connection by the connector that is provided with the side of module
Module installation method	One-touch connection to 35mm DIN rails (standard connection mechanism provided in the system)
Applicable wire	AWG 28 - 20
Applicable plug	FK-MC 0,5/9-ST-2,5 plug (made by Phoenix Contact Corp.)

*1 Available only when the CNT24-2(FIT)GY is connected to the CPU-SBxx(FIT)GY.

*2 Not supported if connected to a CPU-CA10(USB)GY

Notes!

- *When connecting the Module to a controller module, the internal power consumption should be taken into account. If the total current exceeds the capacity of the power supply unit, the integrity of the operation cannot be guaranteed. For further details, please see the Controller Module manual.*
- *Depending upon the specific controller module that is used, some of the functions are not supported.*

Table 6.2. Installation Environment Requirements

Parameter		Requirement description
Operating temperature		0 - 50°C
Storage temperature		-10 - 60°C
Operating humidity		10 - 90% RH (No condensation)
Floating dust particles		Not to be excessive
Corrosive gases		None
Line-Noise resistance	Line-noise	AC line/2kV, Signal line/1kV (IEC1000-4-4Level 3, EN61000-4-4Level 3)
	Static electricity resistance	Contact discharge/4kV (IEC1000-4-2Level 2, EN61000-4-2Level 2) Atmospheric discharge/8kV (IEC1000-4-2Level 3, EN61000-4-2Level 3)
Vibration resistance	Sweep resistance	10 - 57Hz/semi-amplitude 0.15mm, 57 - 150Hz/2.0G 80minutes each in X, Y, and Z directions (JIS C0040-compliant, IEC68-2-6-compliant)
Impact resistance		15G, half-sine shock for 11ms in X, Y, and Z directions (JIS C004-compliant, IEC68-2-27-compliant)

External Dimensions

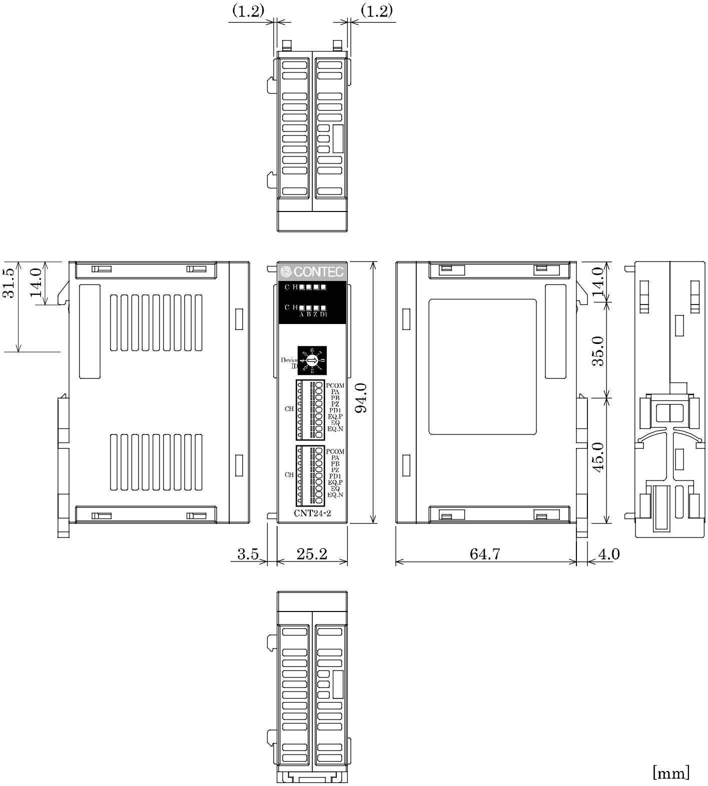


Figure 6.2. External Dimensions

CNT24-2(FIT)GY

User’s Manual

CONTEC CO., LTD. November 2006 Edition

3-9-31, Himesato, Nishiyodogawa-ku, Osaka 555-0025, Japan

Japanese http://www.contec.co.jp/

English http://www.contec.com/

Chinese http://www.contec.com.cn/

[08062001] Management No. A-40-610

[11152006_rev7] Parts No. LZU3892