# Start-Up Manual

# FMC32

# Compact Controller with Integrated Driver





— Blank Page —

## <u>Index</u>

1.	Preface	1
2.	Outline	2
	2-1. Operation sequence to be designed this time	2
	2-2. Environment requirements	3
	2-3. Connection examples	3
	2-4. Order to power on	4
	2-5. Outline of operation procedure	4
3.	Prepare operation	5
	3-1. Start a program	5
	3-2. Prepare a project	6
	3-3. Design operation patterns	7
	3-3-1. "Design the data of pattern" window	7
	3-3-2. Design and registration of Origin return operation	7
	3-3-3. Design and register operation to output 50 pulses in CW direction	10
	3-3-4. Design and register operation to output 100 pulses in CW direction	12
	3-3-5. Design and register operation to output 2000 pulses in CCW direction	15
	3-3-6. Save the registered operation pattern data	17
	3-3-7. Check registered data	8
	3-4. Operation procedure programming	19
	3-4-1. Show editor window	19
	3-4-2. Programming flow	20
	3-4-3. Confirm operation of program ( Simulation )	30
	3-4-4. Save a program	34
	3-4-5. Confirm operation of program (Actual operation)	35
	3-4-6. Write data to EEPROM	37
	3-4-7. Operation in stand-alone mode	37

#### 1. Preface

This manual shows explanations for basic operation and control of motors using the FMC32. Prior installation of hardware and software is required.

For how to use the FMC32, please see the following user's manual (described below) together.

-Compact controller with integrated driver, FMC32 Hardware, User's manual (Document No. YA 7162)

-FMC32 Control software, compact controller with built-in driver, User's manual (Document No. YA7163)

- USB to 4-wire serial conversion unit PUSB-3503, User's manual (Document No. YA7164)

- Pulse Control LSI PCD2112 for serial bus control, User's manual (Document No. DA70115)

#### 2. Outline

#### 2-1. Operation sequence to be designed this time

This manual describes the procedure to simply control the FMC32. The series of operation for the FMC32 is called an operation procedure (or operation sequence). The work to prepare an operation procedure is called "programming" and operation procedure data is called "program".



Outline of operation:

After power on, the FMC32 searches the origin point and the machine moves to the start point that is 50 pulses far from the origin point.

Then, after it repeats three sets of operation from positive direction to negative direction, it repeats operation to search the origin point again.

Other conditions:

The initial speed is 10pps and the maximum speed is 500pps.

The acceleration time and deceleration time is both 500ms.

The constant speed is 200pps.

Between each operation within one set operation, 200ms waiting time is inserted.

#### 2-2. Environment requirements

The following environment is needed to operate the operation sequence created according to this manual.

Item	Contents	Notes
PC	It is used for creating an operation pattern, running a program simulation and writing operation data to FMC32. WindowsXP SP3 or Windows 7 is necessary as OS.	Indispensable
PUSB-3503	It is our product used to control FMC32 from a PC.	Indispensable
FMC32	It is a product to be operated according to this manual.	Indispensable
Power supply device for FMC32	It is a power supply device to provide power to FMC32. It can apply 6.5V to 40V. It is used to drive the connected motor. Prepare one that is compatible with the motor to be used.	Indispensable
Stepper motor	A motor to be driven by FMC32. PFCU25-24C1G (our product), etc.	
Origin switch	It is used to notice an origin point and you can use a toggle switch as a substitute for this. Ideal one is a slider switch with an origin point sensor.	
Power supply for origin switch	It is a 5V power supply to drive an origin return switch and photo-coupler of FMC32.	

Items that are not marked as "Indispensable" in the Notes field, are not necessary for basic operation of the device. Even if you do not prepare these items, you can confirm the programmed operation by simulation.

#### 2-3. Connection examples

Connect the devices as follows.



The products surrounded by dots in the above figure do not have to be prepared if you cannot prepare them.

Make the origin switch OFF. The FMC32 recognizes that a point where the origin switch turns on is an origin point.

#### <u>2-4. Order to power on</u>

Please power the products on in the following order.

Connect FMC32 with PUSB-3503 and FMC32 with an origin switch and a motor. Power on FMC32 and the origin switch (5V). You can power on whichever first. Connect PUSB-3503 with PC through an USB cable (PUSB-3503 is applied power from USB bus-power).

Please power them off in the reverse order.

#### 2-5. Outline of operation procedure

The design for operation procedure to FMC32 is classified to three steps as follows.



#### 3. Prepare operation

#### 3-1. Start a program

Double-click the following icon.



The software starts and the following main window appears.

F	MC32 prog	ram developme	nt					X
F	ile ( <u>F</u> ) Tool	(T) EEPROM (E	) (	Others	$(\bigcirc)$			
	RENV2	00000000 h		R	STS	27	2756889	
	RENV1	00000000 h		F	UST		0000 h	
	RDS	0000 h		R	IOP		0003 h	
	RUS	0000 h		R	DUN	00	000000 h	
	RDR	0000 h		RI	DWC	00	000000 h	
	RUR	0000 h		R	SDP		0000 h	
	RMG	0000 h						
	RDP	000000 h						
	RFL	0000 h		00	h		h	
	RFH1	0000 h			00 h	00 h		
	RFH2	0000 h			01 h	00 h		
	RMV	00000000 h			02 h	00 h		
	RMD	0000 h		Dete	a		ite.	
	RCOM	00 h		Deta		VVr	ne	

[Main window]

#### 3-2. Prepare a project

First, prepare a project to save data.

1. Select "New Project" from the "File" menu.



The following window appears.

New project making	×
Project Name	
Project Location SB2SPI¥FMC32_PCD2112¥MyDLL_FMC32_MakeDATA	Browse
cancel	ок

Click the "Browse" button and select the folder where you want to save the project.
 A new folder that has the same name as the "Project Name" specified, is made under the folder you select. The operation data is saved in this folder.

フォルダの参照
Select a folder
MyDLL_FMG32_MakeDAT
OK         キャンセル

3. Input a project name.

After you input a project name, click the "OK" button.

New project making	×
Project Name test	
Project Location PI¥FMC32_PCD2112¥MyDLL_FMC32_MakeDATA¥test	Browse
cancel	ОК

#### 3-3. Design operation patterns

"Operation pattern" means a operation such as positioning operation and origin return operation, etc. You can set the factors such as "initial speed", "maximum speed", "acceleration time", "deceleration data" and "feeding amount" that are necessary for this positioning operation while you drive a motor actually.

#### 3-3-1. "Design the data of pattern" window

Select "Design the data of pattern " from the "Tool" menu.

FMC32	program development			
File ( <u>F</u> )	Tool (T) EEPROM (E) Others (O)			
REN	RSTS window (S) RIST window (D)	i		
REN	FMC32 program development         File (E)       Tool (T)       EEPROM (E)       Others (Q)         REN       RSTS window (S)       RIST window (P)       Others       O         REN       Command sekect (Q)       Design the data of pattern (D)       D         RU       Design the program (E)       Manual operation (M)       Search USB target (N)       Search PCD2112 (E)			
RD	Design the data of pattern ( <u>D</u> )	1		
RU	Design the program ( $\underline{E}$ )	ī		
RD	Manual operation ( <u>M</u> )	Ī		
RU	Search USB target ( <u>N</u> ) Search PCD2112 (F)			
RM				

The following window appears.

Design the data of pattern	×
File (E) Setting (S) Tool (T)	
2000 pulse 2000 pulse x 1 1000 pps 1000 pps 749 pulse 749 pulse	CLK 98304 MHz Constant speed S-curve accel/decel Magnification of one or less is permitted RMV = 00007D0 h RFL = 01F4 h
500 pps 500 pps 500 pps 999.96 ms 999.96 ms 999.96 ms	RFHT = U3E8 h RUR = 2665 h RDR = 2665 h RMG = 4AF h RDP = 2ED h
Pattern Number up 0 v down select MODE Origin return operation	
Direction 🚰 COMMAND STAUD1 (43h) 💌 Register this data Comment Origin return	execute
O	lose

#### 3-3-2. Design and registration of Origin return operation

Check the "Constant speed" checkbox in the programming window.

CLK 9.8304	MHz
Constant spee	ed
🗖 S-curve acce	l/decel
Magnification or less is per	of one mitted

The programming window will display as follows.

Design the data o	f pattern		8
File (E) Setting (S)	Tool (I)		
2	2000 pulse 2000	pulse × 1	CLK 9.8304 MHz
pps	2000 pul: 4000.00	90 ms	Constant speed
500 pps 500 pps	ma	1000 ms	Magnification of one or less is permitted RMV = 00007D0 h RFL = 01F4 h RFH = 03E8 h RUR = 0 h RDR = 0 h RDR = 0 h RDP = 0 h
Pattern Number	p 0 💌 down select	MODE Origin return operation	
Direction 🛃 C	OMMAND STAFL (40h) 💌	Register this data	Auto execute
Comment Origin return			<b>10</b>
			Close

Select "Origin return operation" from the pull-down menu of the mode selection.

MODE	In position	ing operation			•
	Continuous	s operation			
Regi	Origin retu Origin esca Escape fro In position Timer oper	m operation ape operation m EL ing operation ation			
			Clo	se	

Then, change the value in the following column to "200".



Next, click the "Direction" icon to change it to "-".



With that, operation pattern for origin return operation is prepared.

Please input your comment if possible. It helps you to make subsequent programs.



[Check operation]

If a motor is actually connected with the FMC32, you can operate the motor. This allows you to check operation pattern.

Click the "Execute" button.

After the motor starts to operate, the motor will stop when the ORG switch turns on.



Note 1: Do not uncheck the "Auto" checkbox.

Note 2: You cannot check an origin return operation when there is not an ORG switch.

Note 3: If a motor is not connected with FMC32, you can check operation condition by RCUN register value's increasing and decreasing.

[ Check pattern number ]

Please confirm that the pattern number is "0".



[Register operation pattern]

Register the prepared operation pattern. If you do not register it, you cannot use it with the program you will use. Please register the data after you confirm that there are no problems.

Up to 32 operation patterns can be registered.

Please click the "Register this data" button.



The following window appears.

	Mode	Direction	Comment	^
)	Origin return	Minus	Origin return	
2				
}				
5				
;				
-				
1				
,				
0				
1				
2				۰.,
-		_		~

"Operation mode", "Direction" and "Comment" columns appears.

Other information is not shown. Therefore, we recommend that you input a name that you can identify the operation in the "comment" field.

Click the "Set" button" to register origin return operation as patter No.0.

#### 3-3-3. Design and register operation to output 50 pulses in CW direction

Keep the "constant speed" check box checked in "Design of the data of pattern" window.



Next, select "In positioning operation" from the pull-down menu next to "Mode".



The value in the following does not change from "200".



Change the following value to 50. This is the number of pulses to be output.



Then, click the icon next to "Direction" to make it from "-" to "+". Input comments as necessary.

Pattern Number	Comment
Direction 🚰 DOMN	Civ +30 7 200pps
Comment	

With that, you designed an operation to output 50 pulses in CW direction.

Next, you will register this designed pattern. Before you register it, click the "up" button next to "Pattern Number" to change register number to "1". As the following figure shows, the gray background color of the field means that operation is not registered in this pattern number.

Pattern Number	up 1 💌 down	select
Direction 🛃	COMMAND STAFL	(40h) 🔽



Click the following "Register this data" button.

Register this data

Click the "Set" button to register this operation to pattern No.1.

Sele	ct the save numb	er of EEPI	ROM (test.edt)	
	Mode	Direction	Comment	^
0	Origin return	Minus	Origin return	
1	Origin return	Minus	cw +50 / 200pps	
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				~
_				_
	Load deta from file	Sav	e data to file Cancel ESet	

#### 3-3-4. Design and register operation to output 100 pulses in CW direction

Remove the check in the "Constant speed" check box.



Next, select "In positioning operation" from the pull-down menu next to "Mode".

MODE Regis	In positioning operation Continuous operation Origin return operation Origin escape operation Escape from EL In positioning operation Timer operation		
		Close	

#### Set an initial speed as follows.



Put the following value in the maximum speed field.



Set the number of pulses to be output.



Set an acceleration time.



Set a deceleration time.



Next, click the "Direction" icon to change it to "+". Input comments.



With that, you designed the operation to output 1000 pulses in CW direction.

Then, you will register this created pattern. Before register it, click the "up" button next to "Pattern Number" to change register number to "2".

Pattern Number	up 2 🗸 down select
Direction 🛃	COMMAND STAUD1 (43h)

The following "Register this data" button.



Click the "Set" button to register this operation to pattern No.2.

	Mode	Direction	Comment	^
)	Origin return	Minus	Origin return	
1	In positioning	Puls	см +50 / 200pps	
2	In positioning	Minus	cw +1000 / 10pps -> 500pps	
1				
1	1			
5	1			
5	1			_
7	1			- 1
k				_
2		_		- 1
10	-			_
		_		
10		_		_
Z	-			~

[ Confirmation value to be set ]

If the work has gone correct, the "design the data of pattern" window should show as follows.

FMC uses PCD2112 and values are set to each register automatically so as that this PCD2112 operates using the specified values.

The field bounded by the blue line in the following figure shows the actual value that is set to PCD2112 registers.



The value bounded by green lines in the upper line shows how PCD 2112 operates using the value set in registers actually.

PCD2112 does not necessarily operate in accordance with the value you set. The reason is because some errors are observed between the result calculated in PCD2112 and the setting value under PCD2112 specifications. For example, if you input 500ms for acceleration time, acceleration time will be 500.05ms in calculation.

#### 3-3-5. Design and register operation to output 2000 pulses in CCW direction

Keep the "Constant speed" check box unchecked in the "Design the data of pattern" window.



Next, select "In positioning operation" from the pull-down menu next to "Mode".

MODE Regis	In positioning operation Continuous operation Origin return operation Origin escape operation Escape from EL In positioning operation Timer operation		
		Close	

#### Initial speed is not changed.



Maximum speed is not changed.



Set the number of pulses to be output.



Acceleration time is not changed.



Deceleration time is not changed.



Then, click the icon next to "Direction" to make it from "+" to "-". Change comments.



With that, you designed an operation to output 2000 pulses in CCW direction.

Then, you will register this created pattern. Before you register it, click the "up" button next to "Pattern Number" to change register number to "3".

Pattern Number 【	up 3 🗸 Jown s	elect
Direction 🧮	COMMAND STAUD1 (43	h) 🔽

Click the following "Register this data" button.

Register this data
--------------------

Click the "Set" button to register this operation to pattern No.3.

	Mode	Direction	Comment	^
)	Origin return	Minus	Origin return	
	In positioning	Puls	см +50 / 200рря	
2	In positioning	Puls	cw +1000 / 10pps -> 500pps	
}	In positioning	Minus	ccw -2000 / 50 -> 500pps	
1				
;				
;				
,				
;				
)				
0				
1		_		
2				_
-	1			~

#### 3-3-6. Save the registered operation pattern data

Save registered data as a file. If you exit the program without saving, all the registered operation patterns are lost.

There are two ways to save data. Select one to proceeded.

[Operation pattern save procedure 1]

Select "Save Pattern data" from the "File" menu of the menu bar in "Design the data of pattern" window.



#### [Operation pattern save procedure 2]

Select "Save Project" from the "File" menu of the menu bar in the main window.



#### 3-3-7. Check registered data

You can check what operation is registered in which pattern No. Click the "select" button.

vn	select	MC
		1

Registration window appears and the outlines of the operation pattern appear in the pattern number field that operation has already registered.

	Mode	Direction	Comment	<u>^</u>
0	Origin return	Minus	Origin return	<u></u>
1	In positioning	Puls	cw +50 / 200pps	
2	In positioning	Puls	cw +1000 / 10pps -> 500pps	
3	In positioning	Minus	ссн -2000 / 50 -> 500рря	
4				
5				
6				
7				
8				
9				
10				
11				
12				
-	-			×

All information is not shown. Therefore, we recommend you put descriptive comments to help you easily identify operations.

#### 3-4. Operation procedure programming

Next, the operation procedure will be designed by arranging the designed operation patterns. You can check designed operation procedure to be processed in the order you want.

#### 3-4-1. Show editor window

Select "Design the program" from the "Tool" menu. The following editor window appears.



Comment information is already described by default. Do not delete them. Do not change the content of the first line. (You can change the contents of lines below the first line.)

All lines with "#" head are handled as comments. Though "#" is in the middle of lines, character strings that follows "#" are handled as comments if there is a "#" that is separated with more than one space.

The character strings other than comments is handled as program.

FMC32 processes the specified operation in order from the first line of the program in the stand alone mode after it is powered on

#### 3-4-2. Programming flow

The following shows the operation procedure described in "2-1. Operation sequence to be designed this time".

[Label setting 1]

This operation procedure repeats the same operation continuously.

The jump point for repeated operation is specified as "Label"(character string).



Click the last line of the editor and put the cursor in the last line.

Input "Loop\_1" using keyboard. Do not forget to attach a head colon (:).

Attaching a (:) colon at the head of line shows that the character string is a label.

Label characters are composed of only alphanumeric characters and underscores.

🕲 test.sdc
ファイル (E) プロジェクト (P) ツール (T)
🖻 🖬 🖹 📵 🔺 🗹 🕞 🕽 🛯 🖉 🔛 🦉
# FMC32 #
# Made by *****
# # EEROM data file name is
# [test.dpg]
*
:LOOP_1

[Origin return]

Put the cursor under the label and right-click there to show the following pop-up menu. Select "Select the pattern number..." from the pop-up menu.

Select the patt	ern number ( <u>P</u> )	
Wait time setti	ng ( <u>W</u> )	۲
Change excitat	tion ( <u>E</u> )	۲
Set the variable	e register ( <u>R</u> )	۲
Decrement the	variable register ( <u>D</u> )	۲
Selection of Ju	imp processing (J)	۲
HALT (H)		
label (L)		
Break On/Off	( <u>B</u> )	
close (C)		

The list of the operation patterns that have been designed and registered appears. In this list, select 0 (Origin return) and click the "OK" button.

	Mode	Direction	Comment	^
0	Origin return	Minus	Origin return	
1	In positioning	Puls	cw +50 / 200pps	
2	In positioning	Puls	cw +1000 / 10pps -> 500pps	1
3	In positioning	Minus	ccw -2000 / 50 -> 500pps	1
6	1			1
5	1			
5	1			
7				
3				
,				
10				
11	1			
12	-			1.
-	4	_		~

The program for selected operation pattern (pattern No. 0) is added as follows.

The line shows that "P0" operates pattern No. 0.

The character string that follows "#" is a comment to be input when operation pattern is designed.



[ Operation to output 50 pulses in CW direction ]

Put the cursor under the line of "P0" and right-click there to show the following pop-up menu.

Select "Select the pattern number..." from the pop-up menu.

The list of the operation patterns appears. In this list, select the first "In positioning" and click the "OK" button.

	Mode	Direction	Comment	^
D	Origin return	Minus	Origin return	
1	In positioning	Puls	cw +50 / 200pps	
2	In positioning	Pule	cw +1000 / 10pps -> 500pps	
3	In positioning	Minus	ccw -2000 / 50 -> 500pps	
4				
5				
5				
7				
В				
9				
10				
11				
12				
-				×

"P1" and the comment is added as follows.

🕲 test.sdc	
File ( <u>F</u> ) Project ( <u>F</u>	) Tool (T)
🛩 🖪 📑 🖻	🛛 🗹 🖂 🛄 🔃 🖉 🛄 🖉
# FMC32 # # Program file to co	ntrol FMC32
# Made by ****	kokok
# #EEROM data file r # [test.dpg] #	name is
LOOP_1	
P0 #Ori	gin return →50 / 200≂cc
#cw	+50 / 200pps

[Set repeating time]

Set the repeating time so as to repeat one set operation from positive direction to negative direction three times. Put the cursor under "P1" and right-click it. Select "Set the variable register" from the pop-up menu and select "Register number 0".

PO P1	# Origin return # cw +50 ∕ 200pps		
	Select the pattern number ( <u>P</u> ) Wait time setting ( <u>W</u> ) Change excitation ( <u>E</u> )	;	
	Set the variable register ( <u>R</u> )	•	Register number 0 ( <u>0</u> )
	Decrement the variable register ( <u>D</u> )		Register number 1 ( <u>1</u> )
	Selection of Jump processing ( <u>J</u> )		Register number 2 (2)
	HALT ( <u>H</u> )		Registernumber 3 ( <u>3</u> )
	label (L)		Registernumber 4 ( <u>4</u> )
-			Registern umber 5 (5)
_	Break On/Off (B)		Registern umber 6 ( <u>6</u> )
	close ( <u>C</u> )		Register number 7 (7)
-			Registernumber 8 ( <u>8</u> )
			Registernumber 9 ( <u>9</u> )
			Register number 10 ( <u>A</u> )

The following window to set register initial value appears. Input "3" in this field and click the "OK" button. The program that assigns "3" to REG0 register is added.

Input data		×
	Setting of register initial value (1255)	
	3	
	Cancel 0	K 🔤

🖲 test.sdc	
File ( <u>F</u> ) Proje	ect (P) Tool (T)
🖂 🖪 📑	• 🔁 🔺 🖂 🕞 🕽 🔃 🔘 🜆 🖉
#FMC32 # #Program file # Made b # #EEROM data # [test.dpg] #	to control FMC32 <sub>by</sub> ***** file name is
:LOOP_1 P0 P1 REG0 3	# Origin return # cw +50 / 200pps

#### [Label setting 2]

Set the repeating time so as to repeat one set operation from positive direction to negative direction three time. The jump point for reported operation is specified as "Label".

Put the cursor under "REGO 3" line and begin a new line. (This operation is to improve appearance and is not indispensable.)

Next input ":LOOP\_2". Input the head ":" without fail.

🕲 test.sdc	
File ( <u>F</u> ) Project	t (P) Tool (T)
🖂 🖪 📑	8 🛛 🖂 🖂 🔍 📜 🚺 🖉 🏧
#FMC32 # #Program file to #Made by # #EEROM data fil #[test.dpg] #	control FMC32 ***** le name is
:LOOP_1 P0 # P1 # REG0 3 :LOOP_2 I	Origin return cw +50 ∕ 200pps

[Set one operation set from positive direction to negative direction]

Set the repeating time so as to repeat one operation set from positive direction to negative direction three time. The setting method is the same as "origin return" or "operation to output 50 pulses" described in the above.

The window should be as follows after programming.

🕲 test.s	de 🗖 🗖 🔀
File ( <u>F</u> ) F	Project (P) Tool (T)
📂 🖫	👔 🔁 🔺 🖂 🕞 🕽 🛯 🖉 🖉 🚺
# FMC32 #	
# Program 1	file to control FMC32
# Ma	de by *****
#	
# EEROM o	iata file name is
# Ltest.dp	je j
+	
LOOP 1	
PO	# Origin return
P1	# cw +50 / 200pps
REG0 3	
:LOOP_2	
P2	# cw +1000 / 10pps -> 500pps
P2	# cw +1UUU / 1Upps -> 5UUpps

There are two "P2" lines and one "P3" line.

[Put the waiting time]

This operation is to execute the next operation after each operation of one operation set from positive direction to negative direction is completed and 200ms waiting time is passed over. Click the cursor the second "P2" line to be put.



Right-click here and select "wait time setting" from the pop-up menu and select "Set time "x10ms).

2 2 2	# cw +1000 / 10pps -> 500pps 	
·	Wait time setting ( <u>W</u> )	<ul> <li>Set time (x10ms) (S)</li> </ul>
	Change excitation ( <u>E</u> )	•
	Set the variable register (R)	+
	Decrement the variable register (D)	→
	Selection of Jump processing (J)	→
	HALT (H)	
	label (L)	
	Break On/Off ( <u>B</u> )	
	close (C)	

The "wait time input" field appears. Input "20" in this field and click the " OK" button. Waiting time (that is 10 times of the input value in this field) is taken.

Input data		×
	Wait time input (x10ms)	
	20	
	Cancel	ок 🛛

The line specified by the cursor comes down and the program "WAIT 20" is inserted.

🛞 test.si		
File ( <u>F</u> ) F	roject (P) Tool (T)	
📂 🖫	🛨 🖬 🔟 🖂 🖂 🔍 📜 🚺 🖉 🚾 🗾	
#FMC32 # #Program f # Mai #EEROM d #EEROM d # [test.dp #	ile to control FMC32 de by ***** ata file name is ε]	
:LOOP_1 P0 P1 REG0 3	# Origin return # cw +50 ∕ 200pps	
:LOOP_2 P2 WAIT 20	# cw +1000 / 10pps -> 500pps	ŀ
P2 P3	# cw +1000 / 10pps -> 500pps # ccw -2000 / 50 -> 500pps	

Insert the waiting time by the similar operation and edit the program as follows.

🕲 test.sd	
File ( <u>F</u> ) Pr	oject (P) Tool (T)
📂 🖪 🚦	* 6 🛛 A 🖂 > 🖵 🛯 🞯 🖊 🚾 🌌
#FMC32 # #Program fi # Mad # EEROM da # [test.dpg #	le to control FMC32 e by ***** ta file name is ; ]
:LOOP_1 P0 P1 REG0 3	# Origin return # cw +50 ∕ 200pps
:LOOP_2 P2 WAIT_20 P2	# cw +1000 / 10pps -> 500pps # cw +1000 / 10pps -> 500pps
WAIT 20 < P3	# ccw -2000 / 50 -> 500pps

[ Control to repeat operation three time ]

Decrement the REG0 value and operate repetition control by confirming whether or not this result is 0.

First decrement REG0.

Click the line under the third "WAIT 20" to move the cursor.

Right-click here and select "Decrement the variable register" from the pop-up menu and select "Register number 0".



Decrement command of REG0 is inserted.

🛞 test.sd	lc 🔲	
File ( <u>F</u> ) Pr	roject (P) Tool (T)	
	🕇 🖬 🛛 🗠 🗩 🗩 🗶 🖿 🖉	
# FMC32 # # Program fi # Mad # # EEROM da # [test.dpg #	ile to control FMC32 Je by ***** ata file name is g]	
LOOP_1: P0 P1 REG0 3	# Origin return # cw +50 / 200pps	
Loop_2: P2 WAIT 20 P2 WAIT 20 P3	# cw +1000 / 10pps -> 500pps # cw +1000 / 10pps -> 500pps # ccw -2000 / 50 -> 500pps	
WAIT 20 DEC0 <del>4</del>		

Next, jump command is described.

Make the cursor put under the inserted "DEC0" line and right-click. Select "Selection of Jump processing" from the pop-up menu and select "Jumps if the register value is not zero" and "Register number 0".

WAIT 20 DEC0 Select the pattern number (P) Wait time setting (W) Change excitation (E) Set the variable register (R) Decrement the variable register (D) Selection of Jump processing (J) HALT (H) label (L) Break On/Off (B) close (C)	register value is not zero ② al jump (№) interrupt is generated @
--	---

"Jump point Label input window" appears. Input "LOOP\_2" in this field. The head (:) is unnecessary to be input.

Input data				×
	Jump point	Label input		
	LOOP_2			
		Cancel	OK	

A jump command is added.

🕲 test.sd	c 🔲 🗖 🗙
File ( <u>F</u> ) Pi	roject (P) Tool (T)
	16 🛛 🖂 🖂 🖵 🔃 🔘 🔤 🜌 💆
# FMC32 # # Program fi # Mad # # EEROM da # [ test.dps #	ile to control FMC32 le by ***** ata file name is g]
:LOOP_1 P0 P1 REG0 3	# Origin return # cw +50 / 200pps
:LOOP_2 P2 WAIT 20 P2	# cw +1000 / 10pps -> 500pps # cw +1000 / 10pps -> 500pps
WAIT 20 P3 WAIT 20 DEC0	# ccw −2000 / 50 -> 500pps
JNZ REGO	LOOP_2

[ Process to jump to the head of the program ]

After the repetition operation is completed three times, jump to the head of the program to operate origin return operation again.

Place the cursor under "JNZ REG0 Loop\_2" and right-click, and select "Selection of Jump processing" from the pop-up menu and select "Jumps if the register value is not zero."

DEC0 JNZ	REG0 LOOP_2		
	Select the pattern number (P) Wait time setting (W) Change excitation (E) Set the variable register (R) Decrement the variable register (D) Selection of Jump processing (J) HALT ( <u>H</u> ) Iabel (L)	Jumps if the register v Jnconditional jump ( <u>N</u> ) Jumps if the interrupt	alue is not zero (Z) → is generated (D)
	Break On/Off ( <u>B</u> )		
	close (C)		

"Jump point Label input" window appears. Input "LOOP\_2" in this field. The head (:) is unnecessary to be input.

Input data			×
	Jump poi	int Label input	
	LOOP_1		
		Cancel	ОК

A jump command is added.



With this, the program is completed.

#### 3-4-3. Confirm operation of program (Simulation)

Confirm whether or not the created program is operated in the order you want.

First, confirm the operation without using a motor.

In the simulation of operation confirmation, commands are not sent to the motor driver. The purpose of simulation is to confirm whether the designed program is executed in the order you want.

#### [Assemble program]

Click the following "Assemble" button.

🔮 test.sdc	
File (F) Project (P) Tool (T)	
	Auto Real 🏂

If there are no errors in the designed program, the part of the program (the character string but comment and labels) is highlighted in yellow and the first line of the program is displayed inverted.

<b>2</b> 🗐	1 8   A 🗹 🕨 J 🛯 🖉 🏧 🔤 🗾
	Line = [ 22] [ 挿入 ]
# FMC32 #	1 · · · · · · · · · · · · · · · · · · ·
#Program 1 # Mai	ile to control FMU32 le by ######
# 113	
# EEROM d	ata file name is
# [test.dp	e ]
#	
:LOOP 1	
PO	# Origin return
P1	# cw +50 / 200pps
REG0 3	
100P 2	
P2	# cw +1000 / 10pps -> 500pps
WAIT 20	# WAIT 200 msec
P2	# cw +1000 / 10pps -> 500pps
	# WAIT 200 msec
WAIT 20	# cw =1000 / 10pps => 500pps
WAIT 20 P3 WAIT 20	# WATT 200 msec
WAIT 20 P3 WAIT 20 DEC0	# WAIT 200 msec
WAIT 20 P3 WAIT 20 DEC0 JNZ REG0	# WAIT 200 msec
WAIT 20 P3 WAIT 20 DEC0 JNZ REG0 JMP LOOP	# WAIT 200 msec LOOP_2 _1

[ In the case that there is an error to assemble ]

If there is an error in the program, the following message appears

	<b>X</b>
Mistake in the 3rd code	
Close	
	Mistake in the 3rd code

This message shows that the third character string has an error. Click the "Close" button. The line with error is highlighted as follows.

🕲 test.sd	c 📃 🗖 🔀
File ( <u>F</u> ) P	roject (P) Tool (T)
📂 🖪	* 🖻 🔺 🗹 🗩 🕽 💷 🖉 🖉
# FMC32 # # Program fi # Mad # # EEROM da # [test.dpu #	le to control FMC32 le by ***** ata file name is g]
:LOOP_1 P0 P1 REG0 3	# Origin return # cw +50 / 200pps
:LOOP_2 P2 WAIT_20	# cw +1000 / 10pps -> 500pps
P2 WAIT 20 P3 WAIT 20	ж см +1000 / 10pps -> 500pps # ссм −2000 / 50 -> 500pps
JNZ REGO JMP LOOP	LOOP <u>22</u> 1

The third character string "JNZ REG0 LOOP\_22" is wrong. The line should be "LOOP\_22". Like the above example, if the program has any errors, correct the errors by taking a cue from the part of highlighted display.

[ Operation confirmation by step execution]

Click the following "Step execution" button.



The line that is highlighted inverted is processed and the highlighted line moves to the next program line to be processed.



Continue to click the "step execution" button and check whether program is arranged in the order you want.

[ Confirmation operation by continuous exaction ] Click the following "Run" button.



The program to be executed is highlighted in order.

The simulation is processed at the speed you can follow.

Waiting time of WAIT command is ignored at this time. You do not wait actually.

[Stop continuous execution]

Click the "Stop" button.

🕲 test.sdc		
File ( <u>F</u> ) Project ( <u>P</u> )	Tool (I)	
		) 🖉 🔤 🔤

If you click "Run" button again, program starts to be processed from the location that the process stopped.

If you want to restart from the beginning, click the "Assemble" button again.

#### 3-4-4. Save a program

Save the created program as a file.

If you exit program without saving data, the designed program is lost.

There are two kinds of data save procedures. Select one to be processed.

#### [Program save procedure 1]

Select "Save Pattern data " from the "File" menu of the menu bar in the editor window.



Or you can save data by clicking the following save button.



#### [Program save procedure 2]

Select "Save Project" from the "File" menu of the menu bar in the main window.



#### 3-4-5. Confirm operation of program (Actual operation)

You can check the designed program operation while connecting a motor actually. First, assemble the program and make the window as follows.

ファイル ( <u>E</u> )	
	上ine = [22] [挿入]
#FMC32 # #Programfi # Mad # EEROMda #EEROMda #EEROMda	le to control FMC32 le by ***** sta file name is g]
LOOP_1	
PO	# Origin return
P1	# cw +50 / 200pps
REGU 3	
LOOP 2	
P2	# cw +1000 / 10pps -> 500pps
WAIT 20	# WAIT 200 msec
P2	# cw +1000 / 10pps -> 500pps
WAIT 20 Po	# WALL 200 msec
NATT 20	# CW = 1000 / 10pps = 2 000pps # WATT 200 msec
DECO	
JNZ REGO	LOOP_2
JMP LOOP	1

Click the following excitation switch button.

🕲 test.sdc		
File ( <u>F</u> ) Project ( <u>P</u> )	Tool (II)	
	🖌 🗹 🕨 🕽 🚺 🔘 Auto 🔤	

After switching the excitation condition, the button changed as follows.



If the excitation switch button is not changed, the message: "Excitation is not changed" appears. In this case, FMC32 is not detected as connected. Check the condition of the connection to the FMC32 and the condition of the power supply.

Caution: In this start-up guide, a button is used for switching excitation. Actually, you can use an excitation switch command and change the status automatically. The "Ex\_ON" command can make excitation on and the "Ex\_OFF" command can make it off.

After the excitation switch button is changed, click the following "REAL" (Switch to a real operation mode) button.



If the REAL button change as follows, the mode is the actual operation mode.

🖗 test.sdc		
File ( <u>F</u> ) Operation ( <u>O</u> )	Tool (T)	
	4 😒 🗖 🗐 😡	Auto Real 🗾

Subsequent procedure is the same as operation confirmation by simulation.

#### 3-4-6. Write data to EEPROM

Write designed operation pattern and the program using the operation pattern to EEPROM on FMC32. Select "Write to EEPROM" from the EEPROM menu on the Main window.

n d	evelopment/	展示会_基本動作
2	EEPROM (E)	Others ( <u>O</u> )
	Write to EEF	PROM ( <u>E</u> )
Read data from EEPROM (R)		

If an operation pattern is being written, the following message appears.

It is not written to the operation number that operation pattern is not registered. Comments are also not written.



If writing operation pattern is completed normally, writing program starts. All writing is completed, the following message appears.

Writing in EEPROM was completed	
Close	
	Writing in EEPROM was completed

Caution: To Write created program, assemble it before writing program.

Check that the program does not have any errors by assembling it.

If you write unassembled program, the following error message appears and the program is not written. (Operation pattern is written.)

Unsaved program is not written. Please save it before completing this step.

Message example Please assemble the program. The program has been changed. Please assemble it. Please save the program. Please save the data of pattern.

#### 3-4-7. Operation in stand-alone mode

First, unplug a USB cable and power off the FMC32 and origin switch. Separate the four-wire serial communication cable from the PUSB-3503 to the FMC32.

Finally, turn the origin switch on and power on the FMC32.

After powering on, the FMC32 operates in stand-alone mode according to the written data.

According to each dip switch of "STA external input" and "MON external input", there are cases that FMC32 does not operate.

FMC32 should operate if "STA external input" is on and "MON external input" is on.

CAUTION	The descriptions in this manual may be changed without prior notice to improve performance or quality.
---------	--

### **NPM** Nippon Pulse Motor Co., Ltd.

Head Office:No.16-13, 2-chome, Hongo, Bunkyo-ku, Tokyo, 113-0033, JapanTEL: 81-3-3813-8841FAX: 81-3-3813-8665Web: <a href="http://www.pulsemotor.com">http://www.pulsemotor.com</a>E-mail: <a href="int-l@npm.co.jp">int-l@npm.co.jp</a>

Issued in September 2011