มาโครโปรแกรม ของ Fanuc

Fanuc Macro Programming

บทแปล โดย

สืบศักดิ์ สุขการเพียร

จากผู้เขียน

บทความฉบับนี้ จัดทำขึ้นจากความต้องการส่วนตัวของผู้เขียน ที่อยากทำเอกสารภาษาไทยเกี่ยวกับการสร้าง/เขียนโปรแกรมให้กับเครื่องจักรซีเอ็นซี ในรูปแบบมาโคร (Custom Macro) เพื่อ

1. การเผยแพร่ความรู้ต่อสาธารณะ และเป็นวิทยาทาน
2. ทำให้เอกสารต่างประเทศ อ่าน และเข้าใจได้ง่ายขึ้น

สำหรับบทความนี้ มีความเป็นมาจากการที่มีผู้มาปรึกษา เรื่องเครื่องจักรที่มีมาโครควบคุมการสั่งงานเครื่องเจียรซีเอ็นซี 5 แกน (5 axis CNC Tool Re-Grinding Machine) และชุดควบคุมเป็น ฟานัค (FANUC 16M) และเอกสารที่มากับเครื่องเป็นภาษาญี่ปุ่น ประกอบกับพอจะมีความรู้อยู่บ้าง จึงอยากแปลเป็นภาษาไทยเผยแพร่

โดยเนื้อหาภายใน ผู้เขียนได้แปลตาม คู่มือ FANUC Operating Manual 16 M (B-62454E/04) บทที่ 16 CUSTOM MACRO ซึ่งเป็นคู่มือของเครื่องกัดมิลลิ่ง

อย่างไรก็ดี หากพบว่าความหมายต่างๆ และหรือ การให้คำจำกัดความของผู้เขียน อาจไม่ตรงกับหนังสืออื่นที่ท่านผู้อ่านได้เคยพบเห็น ก็ขอความกรุณาแจ้งให้ทราบด้วย เนื่องจากเป็นการแปล/ถ่ายทอด ด้วยความรู้และประสบการณ์ส่วนตัว

สุดท้ายนี้ ต้องขอขอบคุณ

- คุณวิรัตน์ สุภาษิต (Seal Precision Co) เจ้านายคนแรกที่ให้โอกาสรู้จักเครื่องCNC.
- คุณชุวิทย์ เหมาะประไพพันธ์ ผู้ทำให้รู้จักมาโครเป็นรายแรก (เขียนมาโครได้เก่งมาก)
- คุณชาญวิทย์ เลิศเจริญชัย (CNC Parts Co.) (ปัจจุบันเสียชีวิตแล้ว) ทำให้ต้องมาโครอีกครั้ง
- คุณพรพิมนต์ พุ่มอุไร (PTS Precision Tooling Co.) ที่มาขอคำปรึกษาเรื่องเครื่อง
- คุณธนวัตร มีกลิ่นหอม / คมสัน สุดตะกู (Submit Engineering Co) ที่ให้ถ่ายสำเนาคู่มือ
- สมาชิกเวบบอร์ด ของ www.9engineer.com
- และอีกมากมายหลายท่าน ในแวดวงอุตสาหกรรมที่ให้ประสบการณ์ ความรู้ และกำลังใจ

หวังว่าท่านผู้ที่ได้รับ หรือ นำไปอ่านคงได้ประโยชน์ ในการนำไปใช้หรือ นำไปเป็นแนวทางในการศึกษาต่อเพิ่มเติม

สืบศักดิ์ สุขการเพียร

ตุลาคม 2548

สารบัญ

บทนำ

CUSTOM MACRO เป็นฟังชั่น สำหรับการเขียนโปรแกรมสั่งงานเครื่องจักรซีเอ็นซี (CNC machine) ฟังชั่นนี้มีมากับชุดควบคุมเครื่องจักร (Control) ของฟานัค (Fanuc) (บางรุ่น เป็น Option โปรดตรวจสอบก่อนใช้งาน)

Macro Program สามารถเขียนและจัดเก็บได้เฉกเช่น NC Program ตามปกติ มีการใช้งานในลักษณะคล้ายคลึงกับ โปรแกรมย่อย (Sub Program) โปรแกรมมาโคร สามารถเขียนสั่งงานได้ทั้งในแบบธรรมดา ไปจนถึงระดับสูง ที่สร้างโปรแกรมควบคุมการทำงานทั้งหมด ของเครื่อง

โปรแกรมมาโคร แบ่งเป็น 2 ส่วน คือ

1. บรรทัดคำสั่งเรียก (Custom Macro Instruction – G65, G66)
2. เนื้อโปรแกรมมาโคร (Custom Macro Body)

1. Custom Macro Instruction (G65,G66)
   เป็นบรรทัดคำสั่ง ในโปรแกรมหลัก ที่ใช้เรียกมาโครโปรแกรม โดยปกติจะเป็นบรรทัดคำสั่งที่ใช้ G65, G66 เพื่อเรียกโปรแกรมย่อย ซึ่งเป็น Macro Body

2. Custom Macro Body
   โปรแกรมคำสั่ง (NC Program) ที่มีชื่อเหมือนโปรแกรมปกติ (Oxxxx) แต่ภายในบรรจุ คำสั่ง การเคลื่อนที่ (Tool Path) คำสั่งอื่น ตัวแปร ฯลฯ ซึ่งผู้เขียนโปรแกรมได้บัญญัติขึ้น โดยที่ค่าของตัวแปรในโปรแกรมสามารถที่จะส่งค่าจากภายนอก (จากโปรแกรมหลัก) หรือ เปลี่ยนแปลงจากการคำนวณภายใน ความยากง่าย ละเอียด ความซับซ้อน ของโปรแกรมขึ้นอยู่กับความสามารถของผู้เขียน/ทำโปรแกรม
   การเรียกใช้ Macro Program จะมีลักษณะคล้ายกับการเรียกโปรแกรมย่อย (Sub Program) แต่มีข้อแตกต่างตรงที่ เราสามารถส่งค่าที่ต้องการเข้าสู่ Sub Program ที่เป็น Macro Body ได้ ตัวอย่างเช่น

Main Program	Macro body
O0001	O9010
:	#1 = #18/2
G65 P9010 R50 L2	G01 G42 X#1 Y#1 F0.3
:	G02 X#1 Y-#1 R#1
M30	M99

(R = #18 รายละเอียดอธิบายในบทหลังๆ)

บทที่ 1 ตัวแปร (Variable)

ในคำสั่ง NC code (G-code) ที่ใช้สั่งงานเครื่องจักรนั้น ระยะการเคลื่อนที่และค่าอ้างอิงต่างๆ จะเป็นค่าตัวเลขจำนวนจริง หรือ จำนวนเต็ม เช่น G01 X100.0

แต่หากเป็นแบบมาโคร (Macro Program) ค่าตัวเลขจำนวนจริงเหล่านั้น จะแทนที่ด้วยตัวแปร ตามที่ผู้เขียนต้องการ และกำหนดขึ้น เมื่อจะใช้งานจะมีการกำหนดค่า หรือ ส่งค่าเข้าสู่ตัวแปรนั้นๆ

ค่าของตัวแปรเหล่านี้ สามารถที่จะมีการเปลี่ยนแปลงได้ในระหว่างการทำงาน หรือ ส่งค่าเข้าโปรแกรมจากหน้า MDI (MDI Mode) (ใช้ได้ในบางกรณี)

1.1 การกำหนดตัวแปร

ตัวแปรที่ใช้งาน มีลักษณะคือ มีสัญญาลักษณ์ # นำหน้า ตามด้วยตัวเลขเท่านั้น เช่น #1 , #10, #100 ซึ่งจะต่างจากระบบคอมพิวเตอร์ที่สามารถตั้งชื่อให้กับตัวแปรได้หลายแบบมากกว่า

หากจะมีการกำหนดเลขตัวแปรใหม่ โดยใช้การคำนวณเปลี่ยน สามารถทำได้โดยสูตรคำนวณต้องอยู่ในวงเล็บใหญ่ เช่น #[#1+#2-12] และผลการคำนวณต้องได้เป็นค่าจำนวนเต็มบวก และอยู่ในเกณฑ์ที่กำหนดเลขตัวแปรได้ เท่านั้น

1.2 ชนิดของตัวแปร

ตัวแปรแบ่งเป็น 4 ชนิด ตามตัวเลขที่ตามหลังเครื่องหมาย # คือ

Number	Type	Function
#0	Null	ตัวแปรนี้จะไม่มีค่า (ไร้ค่า) เสมอ และไม่สามารถเปลี่ยนแปลงค่าได้
#1 - #33	Local Variables	ตัวแปรภายใน (Local Variables) เป็นตัวแปรที่เก็บค่าและถูกใช้เฉพาะภายในโปรแกรมนั้นๆ และค่าจะหายไป (เป็น Null) เมื่อปิดเครื่อง (Power Off), เราสามารถกำหนดค่าให้ตัวแปรด้วยอากิวเมนท์ (Argument) เพื่อเรียกใช้มาโคร
#100-#149 (#199) #500-#531 (#999)	Common Variables	ตัวแปรทั่วไป (Common Variables) เป็นตัวแปรที่จะคงค่าอยู่ และสามารถเรียกใช้ได้จากทุกๆมาโครโปรแกรม เมื่อปิดเครื่อง (Power off) ตัวแปร #100 - #149 จะไร้ค่า (Null) ตัวแปร #500 - #531 จะยังคงค่าไว้แม้ปิดเครื่อง สำหรับตัวแปร #150 - #199 และ #532 - #999 เป็น option
#1000 - ขึ้นไป	System Variables	ตัวแปรระบบ (system variables) เป็นตัวแปรที่เก็บ และบันทึกค่าในระบบของเครื่องจักร ซึ่งเกี่ยวกับการใช้งาน เช่น ค่าตำแหน่งแกนปัจจุบัน (current position) ค่าชดเชยของเครื่องมือตัด (tool compensation)

1.3 พิสัยของค่าตัวแปร

พิสัยของค่าตัวเลขจำนวนจริง หรือ จำนวนเต็ม ของ ตัวแปรภายใน (Local) และ ตัวแปรทั่วไป(Common) สามารถเป็น 0 หรือ อยู่ระหว่าง -10^{47} ถึง -10^{-29} หรือ 10^{-29} ถึง 10^{47}

หากผลการคำนวณเกิน หรือ ต่ำกว่าค่าในพิสัย จะเกิด P/s Alarm No.111 (ค่าไม่อยู่ในพิสัย)

1.4 เมื่อไม่มีจุดทศนิยม

หากมิได้เติมจุดทศนิยม เมื่อให้ค่ากับตัวแปร ค่านั้นจะเป็นจำนวนเต็ม ตัวอย่างเช่น

#1 = 123 ค่าจริงเมื่อเรียกใช้ #1 คือ 123.000

1.5 การอ้างอิงตัวแปร

การอ้างอิงตัวแปรในโปรแกรมนั้น จะต้องกำหนดรหัสคำสั่งตามด้วยตัวแปร

เช่น G01 X#1 Y#3 F#3 ;

เมื่อต้องการทำการคำนวณ จำเป็นจะต้องใส่ไว้ในเครื่องหมายวงเล็บใหญ่ [ ]

เช่น G01 X[#1+#2] F#3 ;

ค่าของตัวแปรจะทำการปัดขึ้นโดยอัตโนมัติ ตามคำสั่งการเคลื่อนที่ครั้งสุดท้าย

เช่น หากการเคลื่อนที่ครั้งสุดท้าย อยู่ในระดับ 1/1000 mm (0.001) และมีการ กำหนดค่า #1 = 12.3456 (ไม่ว่าจากการป้อนหรือผลการคำนวณ)

คำสั่ง G00 X#1 จะเป็น G00 X12.346

หากต้องการให้กลับทิศทาง (ค่าเป็นลบ) ให้เติมเครื่องหมายลบ “-“ ก่อน #

เช่น G00 X-#1

กรณีที่มิได้กำหนดค่าให้กับตัวแปร (null) ค่านั้นๆจะไม่ถูกเรียกใช้ไม่ว่าจะตามอักษรตัวแปรใดๆ เช่น #1 = 0 , #2 = null

คำสั่ง G00 X#1 Y#2 จะเป็น G00 X0 ;

1.6 ตัวแปรที่ไม่กำหนดค่า

เมื่อมิได้กำหนดค่าให้กับตัวแปร ค่าของตัวแปรนั้นจะไม่มี(ไร้ค่า/null)

ตัวแปร 0 (#0) เป็นตัวแปรที่ไร้ค่า และไม่สามารถกำหนดให้ได้ แต่อ่านได้ โดยมีการใช้งานที่แตกต่างคือ

1. เมื่อมีการกำหนด แต่ไม่ให้ค่า ตัวแปรนั้นจะไม่ถูกเรียกใช้ เช่น

   #1 = 0 , #2 = null คำสั่ง G00 X#1 Y#2 จะเป็น G00 X0 ;

2. ขณะทำงาน (เรียกใช้ในโปรแกรม)

   <ค่าว่าง> (Vacant) มีค่าเท่ากับ 0 ยกเว้นแทนที่ด้วย < ค่าว่าง > (vacant)

เมื่อ #1 = <vacant> ว่าง	เมื่อ #1 = 0
#2 = #1 จะเป็น #2 = <vacant>	#2 = #1 จะเป็น #2 = 0
# 2 = #1*5 จะเป็น #2 = 0	#2 = #1*5 จะเป็น #2 = 0
#2 = #1 + #1 จะเป็น #2 = 0	#2 = #1 + #1 จะเป็น #2 = 0

3. เมื่อใช้แบบเงื่อนไข

<ค่าว่าง> (Vacant) แตกต่างกับ 0 เฉพาะกับ EQ และ NE

เมื่อ #1 = <vacant> ว่าง	เมื่อ #1 = 0
#1 EQ #0 ทำงาน	#1 EQ #0 ไม่ทำงาน
#1 NE 0 ทำงาน	#1 NE 0 ไม่ทำงาน
#1 GE #0 ทำงาน	#1 GE #0 ทำงาน
#1 GT 0 ไม่ทำงาน	#1 GT 0 ไม่ทำงาน

```
VARIABLE                                        01234 N12345

   NO.          DATA          NO.          DATA
   100          123.456       108
   101            0.000       109
   102                        110
   103                        111
   104                        112
   105                        113
   106                        114
   107                        115

ACTUAL POSITION  (RALATIVE)
         X      0.000                  Y      0.000
         Z      0.000                  B      0.000

MEM  **** *** ***                  18:42:15
[ MACRO ]  [ MENU ]    [  OPR  ]    [      ]    [(OPRT)]
```

เมื่อช่อง DATA ว่างเท่ากับไม่มีค่า ตัวแปรนั้นจะไร้ค่า (null)

สัญญาลักษณ์ **** *** *** หมายถึง ค่าตัวเลขไม่อยู่ในพิสัย (overflow)

ค่าสูงเกิน (มากกว่า 99999999) หรือต่ำกว่าพิสัย (น้อยกว่า 0.000000001)

1.7 ข้อจำกัด (ห้ามใช้)

ตัวเลขของ หมายเลขโปรแกรม(Program Number), เลขลำดับการทำงาน (Sequence Number) และ เลขลำดับสั่งข้ามบล็อก (Optional Block Skip Number) ไม่สามารถนำค่าตัวแปรมาใช้ได้แทนได้

ตัวอย่าง เช่น การใช้ตัวแปรเหล่านี้ จะใช้ไม่ได้

```
O#1 ;
/#2 G00 X100.0 ;
N#3 Y200.0 ;
```

บทที่ 2 ตัวแปรระบบ (System Variable)

ตัวแปรระบบ หมายถึง ตัวแปรที่เก็บบันทึกค่า ที่เกี่ยวกับระบบเครื่องจักร เช่น ตำแหน่งปัจจุบัน (current position) ค่าชดเชยเครื่องมือตัด (tool compensation values) ตัวแปรระบบนี้สามารถอ่านและเขียนทับได้ ยกเว้น บางกลุ่มที่ไม่สามารถเขียนทับได้ แต่อ่านได้อย่างเดียว

ตัวแปรระบบนี้ เป็นเครื่องมือที่ทำให้ผู้สร้างโปรแกรม สามารถเขียนได้อย่างซับซ้อน และหรือทำงานได้เองโดยอัตโนมัติ (หลังจากกำหนดค่าเริ่มต้น)

2.1 สัญญาณ Interface Signal

สัญญาณ ของชุดควบคุมต่างๆ (PMC) สามารถอ่านขียนไปมากับ มาโครโปรแกรมได้ โดย

Variable Number	Function
#1000 - #1015 #1032	ใช้เรียกอ่านสัญญาณ 16 บิท จาก PMC เข้าในโปรแกรมมาโคร โดย ตัวแปร #1000 ถึง #1015 จะอ่านค่าสัญญาณ บิทต่อบิท ส่วน #1032 จะอ่านค่าทั้ง 16 บิทในครั้งเดียว
#1100 - #1115 #1132	ใช้เขียนค่าให้แก่ PMC จากค่าในโปรแกรมมาโคร โดยที่ตัวแปร #1100 ถึง #1115 จะเขียนค่าแบบ บิทต่อบิท ส่วน #1132 จะเขียนครั้งเดียว
#1133	ตัวแปรนี้ (#1133) จะทำการเขียนค่าสัญญาณ 32 บิท จากมาโคร ให้แก่ PMC ในครั้งเดียว (ค่าอยู่ระหว่าง –99999999 ถึง +99999999)

รายละเอียด อยู่ในคู่มือ Connection Manual (B-62443E-1)

2.2 ค่าชดเชยเครื่องมือตัด (Tool compensation values)

ค่าชดเชยเครื่องมือตัด (Tool compensation values) ก็สามารถอ่านหรือเขียนได้ โดยใช้ร่วมกับตัวแปรระบบ ตามปกติตัวเลขลำดับจะคู่กันระหว่าง ค่าชดเชยกับเลขตัวแปร ไม่ว่าจะเป็นค่าชดเชยขนาด(Geometric Compensation) ค่าชดเชยการสึกกร่อน(wear compensation) หรือ ค่าความยาวเครื่องมือ (Tool Length) และขนาดของเครื่องมือ (cutter compensation) กรณีที่จำนวนของค่าชดเชยรวมแล้วเกิน 200 ชุด จะเพิ่มการใช้ตัวแปร #2001 ถึง #2400

ตัวแปรระบบของค่าชดเชยของเครื่องมือ (tool compensation) แบบ A

Compensation number	System variable
1	#10001 (#2001)
:	:
200	#10200 (#2200)
:	:
999	#10999

ตัวแปรระบบของค่าชดเชยของเครื่องมือ (tool compensation) แบบ B

Compensation number	Geometry compensation	Wear compensation
1 : 200 : 999	#11001 (#2201) : #11200 (#2400) : #11999	#10001 (#2001) : #10200 (#2200) : #10999

ตัวแปรระบบของค่าชดเชยของเครื่องมือ (tool compensation) แบบ C

	Tool Length compensation (H)		Cutter compensation (D)	
Compensation number	Geometry compensation	Wear compensation	Geometry compensation	Wear compensation
1 : 200 : 999	#11001 (#2201) : #11201 (#2400) : #11999	#10001 (#2001) : #10201 (#2200) : #10999	#13001 : : #13999	#12001 : : #12999

(ข้อสังเกตุ กรุณาจำลำดับที่ 200 ของแบบ C นี้ด้วย เนื่องจากไม่แน่ใจว่าหนังสือพิมพ์ผิดหรือไม่)

2.3 Macro Alarms

Variable Number	Function
#3000	เมื่อป้อนค่าให้กับ #3000 (ค่าระหว่าง 0 ถึง 200) จะทำให้เครื่องหยุดการทำงาน พร้อมแสดงข้อความเตือน โดยข้อความนั้นไม่ควรเกิน 26 ตัวอักษร หน้าจอจะแสดงตัวเลข 3000 บวกด้วยค่าที่ป้อนให้กับ #3000

ตัวอย่างเช่น

#3000 = 1(Tool Not Found)

จะเกิด alarm เครื่องหยุดและหน้าจอแสดงข้อความ “3001 (Tool Not Found) “

การใช้ตัวแปรนี้ ปกติใช้เพื่อการตรวจสอบเงื่อนไขที่เราสร้างขึ้น เมื่อไม่ตรงกับที่กำหนดไว้ ก็จะป้อนค่าเข้าสู่ตัวแปร โดยตัวแปรระบบจะสั่งให้เครื่องหยุด ทั้งนี้เพื่อเป็นการป้องกันความผิดพลาดที่อาจเกิดขึ้นได้ ในระหว่างเดินโปรแกรม

2.4 ข้อมูลเวลา (Time information)

ข้อมูลเวลาในเครื่อง ก็สามารถอ่านและเขียนได้ด้วยตัวแปรระบบ เช่นกัน

ตัวแปร	การทำงาน
#3001	ตัวแปรนี้จะเป็นค่าของเวลาที่ละเอียด 1 มิลลิวินาที (millisecond) นับเพิ่มต่อเนื่องจาก 0 เมื่อเริ่มเปิดเครื่อง ไปเรื่อยๆ และเปลี่ยนกลับเป็น 0 เมื่อนับถึง 2147483648
#3002	ตัวแปรนี้จะนับเวลาทีละ 1 ชั่วโมง เริ่มต้นเมื่อสัญญาณไฟ cycle start ติด (เริ่มสั่งเดินเครื่อง) และจะเก็บค่าไว้แม้ว่าจะปิดเครื่อง และจะเปลี่ยนเป็น 0 เมื่อนับถึง 9544.371767
#3011	ตัวแปรนี้ใช้เพื่อการอ่านค่า วันเดือนปี ปัจจุบันในเครื่อง โดยจะทำการแปลงค่าเป็นตัวเลข เช่น September 28,1994 จะเป็น 19940928
#3012	ตัวแปรนี้ใช้อ่านค่าเวลาปัจจุบัน เป็น ชั่วโมง/นาที/วินาที แล้วแปลงค่าเป็นตัวเลข เช่น 3:34:56 pm. (15:34:56) จะเป็นเลข 153456

2.5 Automatic operation control

ตัวแปร ที่เกี่ยวข้องกับ การควบคุมระหว่างการเดินเครื่องอัตโนมัติ

#3003	Single block	M-S-T function
0	ทำงาน	รอ
1	ไม่ทำงาน	รอ
2	ทำงาน	ไม่รอ
3	ไม่ทำงาน	ไม่รอ

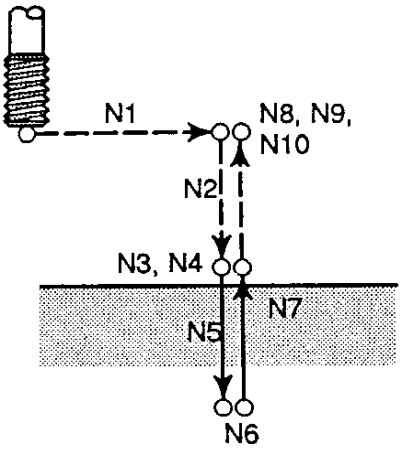
- ค่าของตัวแปรเป็น 0 เมื่อเปิดเครื่อง
- เมื่อสั่งให้ single block ไม่ทำงาน เครื่องจะไม่หยุดที่ละบรรทัด ถึงแม้ว่า สวิทซ์ single block จะเปิด (On)
- เมื่อสั่งให้ M-S-T ไม่รอ (จนทำงานเสร็จ) โปรแกรม จะทำงานตามคำสั่งในบรรทัดถัดไป โดยไม่หยุดรอจน function นั้นทำจนเสร็จ และจะไม่สัญญาณ output ของ DEN (Distribution completion signal) ด้วยเช่นกัน เช่น การสั่งเรียก tool มาเตรียมไว้ เครื่องจะไม่หยุดทำงาน ถึงแม้ว่า tool นั้นจะยังหมุนมาไม่ตรงตำแหน่งรอเปลี่ยน

#3004	Feed hold	Feedrate override	Exact stop
0	ทำงาน	ทำงาน	ทำงาน
1	ไม่ทำงาน	ทำงาน	ทำงาน
2	ทำงาน	ไม่ทำงาน	ทำงาน
3	ไม่ทำงาน	ไม่ทำงาน	ทำงาน
4	ทำงาน	ทำงาน	ไม่ทำงาน
5	ไม่ทำงาน	ทำงาน	ไม่ทำงาน
6	ทำงาน	ไม่ทำงาน	ไม่ทำงาน
7	ไม่ทำงาน	ไม่ทำงาน	ไม่ทำงาน

- ค่าตัวแปรนี้ จะเป็น 0 เมื่อเปิดเครื่อง

- กรณีที่กำหนด feed hold ไม่ทำงาน (disabled)
    1. เมื่อกดปุ่ม feed hold ค้างไว้, เครื่องจะหยุดในลักษณะของ single block stop mode ทำงานทีละบรรทัดคำสั่ง แต่หาก #3003 กำหนดค่าแบบไม่ทำงานเครื่องก็จะไม่หยุดทีละบรรทัด (single block stop not perform)
    2. กดปุ่ม feed hold แล้วปล่อย ไฟสัญญาณติด, เครื่องจะไม่หยุด แต่จะทำงานต่อไปเรื่อยๆ จนกว่าจะพบบรรทัดคำสั่ง ที่เปิดให้ feed hold ทำงาน (enabled)
- กรณีกำหนดให้ feed rate override ไม่ทำงาน (disable) ค่าของ feed override จะเป็น 100% ตลอดโดยไม่สามารถปรับลดที่ปุ่มควบคุม(feed rate override switch) ที่แผงเครื่องได้
- กรณีที่กำหนดให้ exact stop ไม่ทำงาน(disable) เครื่องจะไม่ตรวจสอบตำแหน่ง แม้ว่าในบรรทัดคำสั่งจะไม่มีคำสั่งในการกัด (cutting) (Exact stop จะสั่งเครื่องหยุดเพื่อตรวจสอบตำแหน่งของ tool)

ตัวอย่างโปรแกรม สำหรับการ Tapping




```
O001;
N1 GOO G91 X#24 Y#25;
N2 Z#18
   G04
N3 #3003=3 ;
N4 #3004=7 ;
N5 G01 Z#26 ;
N6 M04 ;
N7 G01 ;
Z-[ROUND[#18]+ROUND[#26] ;
  G04 ;
N8 #3004=0 ;
N9 #3003=0 ;
N10 M03 ;
M99 ;
```

2.6 ค่า Setting

ค่า setting สามารถอ่าน/เขียน โดยตัวแปรจะทำการแปลงค่าเลขฐานสอง (binary) ให้เป็นค่าตัวเลข (decimal number)

#3005

	#15	#14	#13	#12	#11	#10	#9	#8
Setting							FCV	
	#7	#6	#5	#4	#3	#2	#1	#0
Setting			SEQ			INI	ISO	TVC

#9 (FCV) : การใช้ FS15 รูปแบบของการอ่านเทป

#5 (SEQ) : กำหนดให้ใส่เลขบรรทัด โดยอัตโนมัติ

#2 (INI) : กำหนดให้หน่วยทำงานเป็นแบบระบบ นิ้ว หรือ เมตริก

#1 (ISO) : กำหนดให้ใช้รหัส EIA หรือ ISO สำหรับรับ-ส่งข้อมูล

#0 (TVC) : กำหนดให้ใช้/ไม่ใช้ TV Check (ตรวจสอบสัญญาณรับ-ส่ง)

2.7 Mirror Image

ค่าสถานะ mirror image ของแต่ละแกนอ้างอิง (axis) จะบันทึกใน ตัวแปร #3007 โดยจะแปลงค่าเลขฐานสอง(binary) เป็นค่าตัวเลขฐานสิบ (decimal)

#3007								
	#7	#6	#5	#4	#3	#2	#1	#0
Setting	8th axis	7th axis	6th axis	5th axis	4th axis	3rd axis	2nd axis	1st axis
ค่าตัวเลข	128	64	32	16	8	4	2	1

ในแต่ละบิทของข้อมูล ค่า 0 = ไม่เปิด / 1 = เปิด ค่าตัวเลขเป็นค่า 2 ยกกำลัง แปลงเป็นเลขฐานสิบ

ตัวอย่าง เช่น หากค่าของ #3007=3 หมายถึง เปิด Mirror image ให้กับแกนแรก และแกนที่สองอยู่ (ค่านี้เป็นค่าผลรวมของบิทที่มีข้อมูล)

- เมื่อมีการสั่ง mirror image ให้กับแกนใด ไม่ว่าโดยสวิทซ์ หรือ คำสั่ง ค่าของสถานะจะถูกบันทึกในแต่ละบิทของแกน และส่งค่าผลรวมให้ตัวแปร#3007

ตัวแปรระบบ #3007 นี้ เป็นตัวแปรที่อ่านได้อย่างเดียว (read only) หากมีการพยายามเขียนค่าเข้าสู่ตัวแปร เครื่องจะเกิด P/S Alarm “WRITE PROTECTED VARIABE”

2.8 จำนวนของชิ้นงาน

Variable Number	Function
#3901	จำนวนของชิ้นงานที่ทำเสร็จ
#3902	จำนวนของชิ้นงานที่ต้องการ (กำหนด)

ข้อควรระวัง ไม่ควรใส่ค่าเป็นจำนวน ลบ

2.9 Modal Information

คำสั่งแบบ Modal เป็นคำสั่งที่คงอยู่ จนกว่าจะมีคำสั่งที่เหมือนกันในกลุ่มมาสั่งแทน ในตัวแปรระบบ จะมีข้อมูลที่สามารถอ่านค่าได้ ตามตาราง

Variable Number	Function
#4001	G00, G01, G02, G03, G33 (Group 01)
#4002	G17, G18, G19 (Group 02)
#4003	G90, G91 (Group 03)
#4004	(Group 04)
#4005	G94, G95 (Group 05)
#4006	G20, G21 (Group 06)
#4007	G40, G41, G42 (Group 07)
#4008	G43, G44, G49 (Group 08)
#4009	G73, G74, G76, G80 – G89 (Group 09)
#4010	G98, G99 (Group 10)
#4011	G50, G51 (Group 11)
#4012	G65, G66, G67 (Group 12)
#4013	G96, G97 (Group 13)
#4014	G54 – G59 (Group 14)
#4015	G61 – G64 (Group 15)
#4016	G68, G69 (Group 16)
:	:
#4022	(Group 22)

Variable Number	Function
#4102	B code
#4107	D code
#4109	F code
#4111	H code
#4113	M code
#4114	Sequence number
#4115	Program number
#4119	S code
#4120	T code
#4130	P code (number of the currently selected additional workpiece coordinate system)

ตัวอย่าง

เมื่อกำหนด #1 = #4001 ผลลัพธ์ของ #1 จะเป็น 0, 1, 2 ,3, หรือ 33

ในกรณีที่กำหนดตัวแปรระบบ เพื่อให้อ่านข้อมูลคำสั่ง Modal โดยที่กลุ่มของ G code นั้นห้ามใช้ (ไม่มีให้ใช้) จะเกิด P/S alarm

2.10 ตำแหน่งปัจจุบัน (Current Position)

ค่าของตำแหน่งปัจจุบัน จะไม่สามารถเขียนได้ แต่อ่านและนำไปใช้ได้ ซึ่งจะมีตัวแปรระบบที่บันทึกค่าคือ

Variable Number	Position information	Coordinate system	ค่าชดเชย Tool compensation	อ่านค่าได้ขณะเคลื่อนที่
#5001 - #5008	ตำแหน่งสิ้นสุดคำสั่ง (Block end point)	พิกัดของชิ้นงาน (workpiece coordinate)	ไม่รวม	อ่านได้
#5021 - #5028	ตำแหน่งปัจจุบัน (Current position)	พิกัดของเครื่องจักร (Machine coordinate)	รวม	อ่านไม่ได้
#5041 - #5048	ตำแหน่งปัจจุบัน (Current position)	พิกัดของชิ้นงาน (workpiece coordinate)		
#5061 - #5068	ตำแหน่งขณะ Skip (Skip signal position)			อ่านได้
#5081 - #5088	ค่าชดเชยความยาว (Tool length offset)			อ่านไม่ได้
#5101 - #5108	ตำแหน่งที่คลาดเคลื่อนของ Servo			

- ค่าหลักสุดท้าย (1 – 8) แทน แกนแต่ละแกน
- #5081 – 5088 จะบันทึกค่าชดเชยความยาวเครื่องมือ ที่ถูกเรียกใช้งานในขณะนั้น
- เมื่อใช้คำสั่ง G31 (Skip function) ค่าตำแหน่งสิ้นสุด (block end point) จะถูกเก็บใน #5061 - #5068 เมื่อหยุดคำสั่ง G31 ค่าตำแหน่งสิ้นสุด (block end point) จะถูกเก็บในตัวแปรของตนเอง (#5001 - #5008)
- กรณีที่อ่านค่าไม่ได้ในขณะเคลื่อนที่ (disabled) หมายถึง ตำแหน่งขณะนั้นอ่านไม่ได้ จากการที่ค่าจริงยังอยู่ใน buffer อันเป็นผลจากคำสั่งที่เรียกใช้ขณะนั้น (เครื่องยังเคลื่อนที่ หรือทำคำสั่งใน Block ไม่เสร็จสิ้น)

2.11 ค่าชดเชยตำแหน่งศูนย์ของชิ้นงาน (workpiece zero point offset values)

ค่าชดเชยนี้ (workpiece zero point offset values) สามารถอ่านได้จากตัวแปรระบบกลุ่มนี้

Variable Number	Function
#5201 : #5208	ตำแหน่งแกนแรก ของ External workpiece zero offset : ตำแหน่งแกนที่แปด ของ External workpiece zero offset
#5221 : #5228	ตำแหน่งแกนแรก ของ G54 : ตำแหน่งแกนที่แปด ของ G54
#5241 : #5248	ตำแหน่งแกนแรก ของ G55 : ตำแหน่งแกนที่แปด ของ G55
#5261 : #5268	ตำแหน่งแกนแรก ของ G56 : ตำแหน่งแกนที่แปด ของ G56
#5281 : #5288	ตำแหน่งแกนแรก ของ G57 : ตำแหน่งแกนที่แปด ของ G57
#5301 : #5308	ตำแหน่งแกนแรก ของ G58 : ตำแหน่งแกนที่แปด ของ G58
#5321 : #5328	ตำแหน่งแกนแรก ของ G59 : ตำแหน่งแกนที่แปด ของ G59
#7001 : #7008	ตำแหน่งแกนแรก ของ G54.1 P1 : ตำแหน่งแกนที่แปด
#7021 : #7028	ตำแหน่งแกนแรก ของ G54.1 P2 : ตำแหน่งแกนที่แปด
:	:
#7941 : #7948	ตำแหน่งแกนแรก ของ G54.1 P48 : ตำแหน่งแกนที่แปด
#14001 : #14008	ตำแหน่งแกนแรก ของ G54.1 P1 : ตำแหน่งแกนที่แปด
#14021 : #14028	ตำแหน่งแกนแรก ของ G54.1 P2 : ตำแหน่งแกนที่แปด
:	:
#19980 : #19988	ตำแหน่งแกนแรก ของ G54.1 P300 : ตำแหน่งแกนที่แปด

** เมื่อใช้ตัวแปร #5201 - #5328 จะต้องใช้ ตัวแปรที่เป็น Option ด้วย โดยที่ #7001 – 7948 (G54.1 P1 – G54.1 P48) เป็น Option สำหรับระบบชิ้นงานแบบ 48 ชิ้น และ #14001 – 19988 (G54.1 P1 – G54.1 P300) เป็น Option สำหรับระบบชิ้นงานแบบ 300 ชิ้น และยังใช้ตัวแปร #7001 – 7948 ได้เช่นกัน

บทที่ 3 การคำนวณและลอจิก

ตารางต่อไปจะเป็นลักษณะของการคำนวณ และลอจิก ที่สามารถใช้ในโปรแกรมมาโครได้ การคำนวณสามารถที่จะใช้ค่าคงที่ที่บันทึกในแต่ละค่าตัวแปร หรือ การนำค่าของตัวแปรแต่ละตัวมาทำการคำนวณให้เกิดเป็นค่าใหม่ หรือ สร้างตัวแปรใหม่จากผลการคำนวณ อักษร j, k จะเป็นค่าคงที่ หรือ สูตรคำนวณก็ได้ ส่วน อักษร I จะเป็นตัวเลขลำดับของตัวแปร

Function	Format	Remarks
เท่ากับ / เท่ากัน	#I = #j	
บวก (Sum) ลบ (Difference) คูณ (Product) หาร (Quotient)	#I = #j + #k ; #I = #j - #k ; #I = #j * #k ; #I = #j / #k ;	
Sine Cosine Tangent Arctangent	#I = SIN[#j] ; #I = COS[#j] ; #I = TAN[#j] ; #I = ATAN[#j] / [#k] ;	ค่ามุมกำหนดเป็นองศา เช่น มุม 90 องศา 30 ลิปดา เขียนเป็น 90.5
รากกำลังสอง (Square root) จำนวนเต็มบวก (Absolute value) ไม่ปัดเศษทศนิยม (Round off) ปัดเศษลง (Round down) ปัดเศษขึ้น (Round up)	#I = SQRT[#j] ; #I = ABS[#j] ; #I = ROUND[#j] ; #I = FIX[#j] ; #I = FUP[#j] ;	
หรือ (OR) และหรือ (XOR) และ (AND)	#I = #j OR #k ; #I = #j XOR #k ; #I = #j AND #k ;	การทำงานของลอจิก จะตรวจสอบ แบบบิทต่อบิท
แปลงค่าจาก BCD เป็น BIN แปลงค่าจาก BIN เป็น BCD	#I = BIN[#j] ; #I = BCD[#j] ;	ใช้แปลงค่าสัญญาณเพื่อรับส่งกับ PMC

หมายเหตุ

BCD (binary character data) หมายถึง ตัวอักษรที่ใช้แทนค่าของเลข ฐานสอง (1, 0)

BIN (binary data) หมายถึง สัญญาณ binary มีสถานะ on หรือ off

3.1 หน่วยของมุม

หน่วยของค่ามุมที่จะใช้กับฟังชั่น SIN, COS, TAN และ ATAN จะเป็นค่าองศาในแบบทศนิยม

ตัวอย่างเช่น 90 องศา 30 ลิปดา = 90.5

3.2 ฟังชั่น ATAN

กรณีของฟังชั่น ATAN จะต้องกำหนดความยามสองส่วน คั่นด้วยเครื่องหมายหาร (/) โดยที่ผลลัพธ์ จะอยู่ระหว่าง 0 ถึง 360

ตัวอย่างเช่น

#1 = ATAN[1] / [-1] ผลลัพธ์ของ #1 = 135.0

3.3 ฟังชั่นปัดเศษ Round

กรณีที่ฟังชั่นนี้ ใช้กับการคำนวณ หรือ ลอจิก (IF หรือ WHILE) ฟังชั่นนี้ จะทำการปัดเศษให้เป็นจำนวนเต็ม

เช่น หาก #2 มีค่าเท่ากับ 1.2345

#1 = ROUND(#2) ผลลัพธ์ของ #1 = 1.0

กรณีที่ฟังชั่นนี้ ใช้ใน NC statement (G-code) ฟังชั่นจะทำการปัดเศษที่ทศนิยมหลักสุดท้าย ของหน่วยที่ละเอียดสุดของการสั่งเคลื่อนที่ เช่น นิ้ว = 0.0001" เมตริก = 0.001 mm.

เช่น

#1 = 1.2345

#2 = 2.3456

G00 G91 X-#1 ; เคลื่อนที่ 1.235 มม.

G01 X-#2 F300 ; เคลื่อนที่ 2.346 มม.

G00 X[#1+#2] ; เคลื่อนที่ 3.580 มม. (1.2345 + 2.3456 = 3.5801)

ซึ่งตำแหน่งจะผิดไป เนื่องจาก 1.235 + 2.346 = 3.581 มม. ดังนั้นในบรรทัดสุดท้ายควรจะเขียนเป็น

G00 X-[ROUND[#1] + ROUND[#2]] เพื่อให้ทำการปัดเศษก่อนทำการคำนวณ

3.4 การตัด/ปัดเศษ จำนวนจริง

การใช้ฟังชั่นปัดเศษขึ้น (round up / FUP) หรือ ปัดลง (round down / FIX) ในการสั่งงาน CNC ควรจะมีการระมัดระวัง ตลอดจนการให้ค่าเป็น ลบ

เช่น

#1 = 1.2 , #2 = -1.2

#3 = FUP[#1] ผลลัพธ์ #3 จะเท่ากับ 2

#3 = FIX[#1] ผลลัพธ์ #3 จะเท่ากับ 1

#3 = FUP[#2] ผลลัพธ์ #3 จะเท่ากับ -2

#3 = FIX[#2] ผลลัพธ์ #3 จะเท่ากับ -1

3.5 อักษรย่อของฟังชั่น

เมื่อจะเลือกใช้ฟังชั่นในโปรแกรม สามารถใช้อักษรย่อได้ โดยใช้อักษรสองตัวแรกของฟังชั่น เช่น

ROUND -> RO

FIX -> FI

3.6 ลำดับของการทำงาน

ลำดับแรก จะเป็นเครื่องหมาย คูณ หาร เงื่อนไข "และ" (* , / , AND)

ลำดับถัดไป เป็น เครื่องหมาย บวก ลบ เงื่อนไข "หรือ" "และหรือ" (+ , - , OR, XOR)

เช่น

#1 = #2 + #3 * SIN[#4]

-- 1 --

------ 2 --------

--------- 3 ----------

ลำดับ 1 , 2 และ 3 ของการทำงาน

3.7 วงเล็บ และลำดับทำงาน

วงเล็บใหญ่ ([ ]) สามารถใช้เพื่อกำหนดลำดับการคำนวณด้วย เช่นเดียวกับการคำนวณตามปกติ แต่วงเล็บสามารถใช้ได้เพียง 5 ชั้นของการคำนวร นับรวมถึงการคำนวณที่ปิดฟังชั่น หากเกิน 5 ชั้น จะเกิด P/S Alarm No.118

#1 = SIN[[[#2+#3]*#4+#5]*#6]

--- 1 –

----- 2 -----

---------- 3 -------

---------- 4 ----------

---------- 5 ----------

3.8 ข้อจำกัด (Limitations)

วงเล็บใหญ่ วงเล็บใหญ่ [ ] ใช้สำหรับ ปิดการคำนวณฟังชั่น ส่วนวงเล็บเล็ก ( ) ใช้สำหรับคำอธิบาย หรือ comments

ข้อผิดพลาด อาจเกิดข้อผิดพลาด (error) ในบางกรณีต่อไปนี้

Operation	Average error	Maximum error	ชนิดของ error
a = b*c	1.55×10^{-10}	4.66×10^{-10}	Relative error (1) $\lvert \varepsilon/a \rvert$
a = b / c	4.66×10^{-10}	1.88×10^{-9}	
a = √b	1.24×10^{-9}	3.73×10^{-9}	
a = b + c a = b – c	2.33×10^{-10}	5.32×10^{-10}	(2) Min $\lvert \varepsilon/b \rvert$ ” $\lvert \varepsilon/c \rvert$
a = SIN[b] a = COS[b]	5.0×10^{-9}	1.0×10^{-8}	Absolute error (3) $\lvert \varepsilon \rvert$ degree
a = ATAN[b] / [c] (*4)	1.8×10^{-6}	3.6×10^{-6}	

1. Relative error จะขึ้นอยู่กับผลลัพธ์ของ การคำนวณ (operation)
2. ผลลัพธ์ได้ค่าน้อยกว่า error ทั้งสองแบบ
3. ผลลัพธ์เป็นค่าคงที่ ขึ้นอยู่กับผลการคำนวณ
4. ฟังชั่น TAN จะคำนวณโดย SIN/COS

ความละเอียดของ ค่าตัวแปร ประมาณ 8 หลัก หากมีการให้ค่ามากเกิน อาจจะได้ผลลัพธ์ที่ผิดออกไป เช่น

#1 = 9876543210123.456

#2 = 9876543277777.777

ผลที่ได้คือ

#1 = 9876543200000.000

#2 = 9876543300000.000

กรณีนี้ เมื่อกำหนด #3 = #2 - #1 ผลลัพธ์ #3 = 100000.000

(การที่ผลลัพธ์ในการคำนวณไม่ถูกต้อง เนื่องจากเป็นการคำนวณในรูปแบบของ เลขฐานสอง)

ตลอดจนโปรดระวัง ความผิดพลาดที่อาจเกิดขึ้นกับ การทำงานแบบเงื่อนไข โดยกำหนดค่าของเงื่อนไขเข้าใกล้ จุดต่ำสุดของพิกัด (low limit value) หรือ ข้อจำกัดของเครื่องในการเคลื่อนที่ เช่น

IF [ABS[#1 - #2] LT 0.001

ซึ่งในทางปฏิบัติงาน เครื่องอาจไม่พบความแตกต่างนี้จริงๆ ก็ได้ ถึงแม้ว่าค่าทางการคำนวณจะสามารถรับรู้ได้ว่าแตกต่าง แต่ผลทางปฏิบัติเครื่องอาจรับรู้ว่าเท่ากันก็ได้

และโปรดระวังการปัดเศษลง โดยร่วมกับการตัดเศษ ที่ผลของการคำนวณบางกรณี ทำให้ค่าเกิดความผิดพลาดไปได้ เช่น

เมื่อกำหนด #2 = #1 * 1000 ในขณะที่ ค่าของ #1 = 0.002

ผลลัพธ์ที่ได้ #2 อาจไม่เท่ากับ 2 แต่เป็น 1.99999997

และเมื่อสั่ง #3 = FIX[#2] ยิ่งทำให้ผลลัพธ์ผิดออกไปอีก โดยแทนที่ค่าจะ = 2 แต่จะเป็น = 1 แทน

หารด้วยศูนย์ กรณีตัวหารเป็นศูนย์ หรือ TAN[90] จะเกิด P/S Alarm No.112

บทที่ 4 ลักษณะคำสั่งมาโคร (Macro statement)

บรรทัดของคำสั่ง (block) ใน NC Program ต่อไปนี้ จะหมายถึง คำสั่งมาโคร (Macro statement)

- ในบรรทัดมีการคำนวณ หรือ ลอจิกเงื่อนไข (=)
- ในบรรทัดมีคำสั่งควบคุมสั่งการ เช่น GOTO, DO, END
- ในบรรทัดมีคำสั่งเรียกใช้มาโคร เช่น G65, G66, G67 หรือ G-Code, M-code อื่น (ที่กำหนดในพารามิเตอร์)

1. ความต่างจากคำสั่ง NC

   มาโครจะไม่มีการหยุดทำงานทีละบรรทัด (single block mode) (single block จะทำงานต่อเมื่อ ค่า SMB ในพารามิเตอร์ No.6000 = 1)

   บรรทัดคำสั่งมาโคร จะไม่เหมือนกับบรรทัด NC ที่ไม่มีการเคลื่อนที่ ในขณะที่มีการเรียกค่าชดเชยเครื่องมือ (cutter compensation mode) (รายละเอียดในบทที่ 7)

2. คำสั่ง NC ที่เหมือนมาโคร

   บรรทัดคำสั่ง NC ที่มีการเรียกโปรแกรมย่อย (subprogram call) เช่น M98, M หรือ T code อื่น) และไม่มีรหัสคำสั่งอื่น ยกเว้น O, N, L จะมีคุณสมบัติเหมือน คำสั่งมาโคร

   บรรทัดคำสั่ง (block) ที่ไม่มีรหัสคำสั่งอื่น ยกเว้น O, N, P หรือ L จะมีคุณสมบัติเหมือน คำสั่งมาโคร

บทที่ 5 การข้าม การวนรอบ

ลำดับการทำงานในโปรแกรม สามารถควบคุมได้ด้วย การใช้ GOTO, IF โดยมีลักษณะ ของลำดับการทำงาน หรือ ทำซ้ำอยู่ 3 แบบ คือ

- GOTO ข้ามบรรทัดไปยังบรรทัดที่กำหนด
- IF ลำดับทำงานแบบมีเงื่อนไข
- WHILE ทำงานซ้ำๆ จนครบจำนวนรอบ หรือ เงื่อนไขที่กำหนด

5.1 คำสั่ง GOTO

คำสั่งนี้ ใช้เพื่อการข้ามไปทำงานยังบรรทัดที่กำหนด โดยการกำหนดเลขลำดับบรรทัดที่ต้องการ เลขลำดับบรรทัดนี้จะต้องอยู่ระหว่าง 1 – 99999 หากผิดไปจากนี้ จะเกิด P/S Alarm No.128 เลขลำดับนี้ กำหนดเป็นตัวเลขโดยตรง หรือ จากการคำนวณก็ได้

รูปแบบ GOTO n ; n = sequence number (1 – 99999)

เช่น

```
GOTO1 ;
GOTO#10 ;
```

5.2 คำสั่ง IF

คำสั่งนี้ ต้องกำหนดเงื่อนไข ตามหลัง IF หากตรงเงื่อนไขก็จะทำงานในบรรทัดที่กำหนด หากไม่ตรงเงื่อนไขก็จะข้ามไปทำงานในบรรทัดอื่น

เช่น

```
IF [#1 GT 10] GOTO2 ;  (หากตรงเงื่อนไข ค่า #1 มากกว่า 10 ข้ามไปทำงานใน บรรทัด N2)
: (หากไม่ตรงเงื่อนไข ทำในบรรทัดต่อไป )
N2 G00 G91 X10.0 ;
:
```

การกำหนดเงื่อนไข

1. เงื่อนไขที่กำหนด จะต้องเขียนอยู่ในเครื่องหมายวงเล็บใหญ่ ([,]) สามารถใช้การคำนวณด้วยได้
2. อักษรของเงื่อนไข จะใช้ตัวอักษร 2 ตัวแทนข้อกำหนด เท่ากับ, มากกว่า, น้อยกว่า ไม่สามารถใช้รูปแบบของสัญญาลักษณ์ได้

อักษรเงื่อนไข	ความหมาย
EQ	เท่ากับ
NE	ไม่เท่ากับ
GT	มากกว่า
GE	มากกว่า หรือเท่ากับ
LT	น้อยกว่า
LE	น้อยกว่า หรือเท่ากับ

ตัวอย่างโปรแกรม เป็นโปรแกรมที่กำหนดเงื่อนไข ตรวจสอบค่ามากกว่า 10

```
O9500
#1 = 0
#2 = 1
N1 IF[#2 GT 10] GOTO2 ; ( กำหนดเงื่อนไข #2 มากว่า 10 จึงไปทำงานบรรทัด N2 )
#1=#1+#2   ( เพิ่มค่าให้กับ #1 )
#2=#2+#1   ( เพิ่มค่าให้กับ #2 )
GOTO1      ( ย้อนกลับไปตรวจเงื่อนไข )
N2 M30;    ( End of Program )
```

5.3 การทำซ้ำ WHILE

คำสั่ง WHILE ใช้สำหรับการทำงานแบบวนรอบ (ทำซ้ำ) ซึ่งหากตรงเงื่อนไข ก็จะทำงานตามบรรทัดคำสั่งที่อยู่ระหว่าง DO ถึง END หากไม่ตรงเงื่อนไขก็จะข้ามไปทำงานในบรรทัดคำสั่งที่ต่อจาก END

รูปแบบ

```
WHILE [ เงื่อนไข ] DO m ; ( m = 1,2,3)
:  ทำงานในบรรทัดจนถึง END เมื่อตรงเงื่อนไข
END m;
:  ทำงานในบรรทัดถัดจาก END เมื่อไม่ตรงเงื่อนไข
```

เลขลำดับตามหลัง DO และ END ใช้ได้เพียง 1, 2, 3 หากใช้เลขอื่น จะเกิด P/S Alarm No.126

ข้อกำหนดของ loop (Nesting)

กำหนดให้ใช้ 1 – 3 หลัง DO และ END สามารถเรียกหรือ สร้าง loop ได้ตามที่ต้องการโดยไม่จำกัด ยกเว้น การทำซ้อนกันได้ไม่เกิน 3 ระดับ แต่ไม่สามารถที่จะตัดกันได้ หากมีการตัดกัน จะเกิด P/S Alarm No.124

ตัวอย่าง

1. เลขลำดับ 1,2,3 เรียกใช้ไม่จำกัด

```
WHILE [ …. ] DO1;
:
END1
WHILE [ … ] DO1 ;
:
END1;
```

2. วงรอบ (Loop) ไม่สามารถตัดกันได้

```
WHILE [ … ] DO1;
:
WHILE […] DO2 ;
:
END1 ;
:
END2 ;
```

3. วงรอบ (loop) ทำได้เพียง 3 ระดับ

```
WHILE [ … ] DO1;
:
    WHILE […] DO2;
    :
        WHILE […] DO3;
        :
        END3 ;
    :
    END 2 ;
:
END1 ;
```

4. สามารถใช้คำสั่งอื่น ให้ออกนอกวงได้

```
WHILE [ … ] DO1;
:
IF […] GOTO n ;
:
END1 ;
:
Nn
```

5. ไม่สามารถสั่งให้กระโดดเข้าในวงได้

```
IF [ … ] GOTO n;
:
WHILE […] DO1 ;
:
Nn
:
END1 ;
```

- หากลืมกำหนดเงื่อนไขของ WHILE บรรทัดคำสั่งระหว่าง DO และ END นั้นจะไม่ทำงาน
- การใช้ GOTO ข้ามไปมา จะทำงานช้ากว่า การใช้ WHILE เนื่องจากเสียเวลาค้นหาเลขลำดับ
- เงื่อนไข EQ และ NE ค่าศูนย์ (zero) และไร้ค่า (vacant) ไม่เหมือนกัน ส่วนเงื่อนไขอื่นมีค่าเหมือนกัน คือ vacant มีค่าเท่ากับ ศูนย์ (zero)

ตัวอย่างโปรแกรม ทำงานเมื่อค่าไม่เกิน 10 (น้อยกว่าหรือเท่ากับ)

```
O0001
    #1 = 0 ;
    #2 = 1 ;
    WHILE [#2 LE 10] DO1 ;
    #1=#1+#2 ;
    #2=#2+#1 ;
    END1 ;
    M30;
```

บทที่ 6 การเรียกโปรแกรมมาโคร

การเรียกโปรแกรมมาโคร สามารถเรียกได้หลายแบบ คือ

- แบบง่าย (G65)
- แบบ Modal (G66, G67)
- แบบใช้ G code
- แบบใช้ M code
- โปรแกรมย่อย เรียกโดย M code
- โปรแกรมย่อย เรียกโดย T code

ข้อแตกต่างระหว่างมาโคร (macro) กับโปรแกรมย่อย (sub program)

คำสั่ง G65 ที่เรียกใช้มาโคร จะต่างจาก การเรียกโปรแกรมย่อย ด้วย M98 คือ

- G65 สามารถกำหนด อากิวเมนต์ (ส่งค่าให้กับตัวแปรในมาโคร) ได้ แต่ M98 ทำไม่ได้
- ในบรรทัดที่มี M98 และมีคำสั่ง NC อื่นๆอยู่ เครื่องจะเรียกโปรแกรมย่อยหลังจากทำคำสั่งเสร็จ (ตัวอย่างเช่น G01 X100.0 M98 Pp) ส่วน G65 จะไม่ทำงาน ไม่เรียกมาโครมาทำงาน
- ในบรรทัดที่มี M98 และมีคำสั่ง NC อื่นๆอยู่ สามารถใช้ single block ได้ ในขณะที่ G65 ไม่รับการสั่งงานแบบ single block และไม่หยุดทีละบรรทัด (block)

6.1 การเรียกมาโครแบบง่าย (Simple Call, G65)

เมื่อเรียกมาโครโดย G65 ต้องตามหลักด้วย P และหมายเลขของโปรแกรมมาโคร โดยที่ค่าข้อมูลตัวแปรที่เรียก อากิวเมนต์ (argument) สามารถกำหนดเพื่อส่งค่าเข้าสู่ตัวแปรในมาโครได้

รูปแบบ G65 Pp L l <argument-specification>

: p = หมายเลขโปรแกรมมาโคร

l = จำนวนรอบที่ทำซ้ำ (ปกติเป็น 1)

argument = ข้อมูลตัวเลขที่ส่งเข้าสู่ตัวแปรภายใน ของมาโคร

ตัวอย่าง

O001 ;	O9010
:	#3 = #1+#2
G65 p9010 L2 A1.0 B2.0	IF [#3 GT 360] GOTO9
:	G00 G91 X#3 ;
M30 ;	N9 M99 ;

<u>การเรียกมาโคร</u>

1. หลังคำสั่ง G65 ต้องตามด้วย Pและหมายเลขโปรแกรมที่บันทึกมาโคร
2. เมื่อต้องการทำซ้ำ ให้เติมเลขจำนวนครั้ง 1 – 9999 หลัง L หากไม่กำหนด L ค่าจะเป็น 1
3. เมื่อมีการกำหนด อากิวเมนต์ จะเป็นการส่งค่าเข้าสู่ ตัวแปรภายใน (มีข้อกำหนดของอากิวเมนต์เฉพาะ)
4. เมื่อเรียกมาโครโดย G65 ระดับของตัวแปรภายใน (Local Variable) จะมีการเปลี่ยนระดับ แต่ M98 ตัวแปรภายใจจะไม่เปลี่ยน (มีอธิบายถัดไป)

การกำหนดอากิวเมนต์

อากิวเมนต์ (argument) เป็นตัวอักษรที่ใช้ในการกำหนดค่าตัวแปร เพื่อส่งข้อมูลหรือ ค่าตัวเลข ให้กับตัวแปรภายในมาโคร ซึ่งมาโครจะรับรู้ความแตกต่างของแต่ละแบบ โดยอัตโนมัติ อากิวเมนต์ มี 2 แบบ เบ่งเป็น

อากิวเมนต์ แบบที่ 1 (TYPE I)

Address	Variable Number
A	#1
B	#2
C	#3
D	#7
E	#8
F	#9
H	#11

Address	Variable Number
I	#4
J	#5
K	#6
M	#13
Q	#17
R	#18
S	#19

Address	Variable Number
T	#20
U	#21
V	#22
W	#23
X	#24
Y	#25
Z	#26

- อากิวเมนต์ แบบที่ 1 นี้ อักษร G, L, N, O, P จะไม่สามารถใช้ได้ (เนื่องจากจะไปซ้ำกับรหัสคำสั่ง)
- ค่าของตัวแปรอื่น ที่มิได้กำหนดค่าอากิวเมนต์ จะเป็น Null (ไร้ค่า)

อากิวเมนต์ แบบที่ 2 (TYPE II)

อากิวเมนต์ แบบนี้จะใช้ A,B,C และ I,J,K รวม 10 ชุด มักใช้ส่งค่าในแบบสามมิติ(3-Dimension)

Address	Variable Number
A	#1
B	#2
C	#3
I_1	#4
J_1	#5
K_1	#6
I_2	#7
J_2	#8
K_2	#9
I_3	#10
J_3	#11

Address	Variable Number
K_3	#12
I_4	#13
J_4	#14
K_4	#15
I_5	#16
J_5	#17
K_5	#18
I_6	#19
J_6	#20
K_6	#21
I_7	#22

Address	Variable Number
J_7	#23
K_7	#24
I_8	#25
J_8	#26
K_8	#27
I_9	#28
J_9	#29
K_9	#30
I_{10}	#31
J_{10}	#32
K_{10}	#33

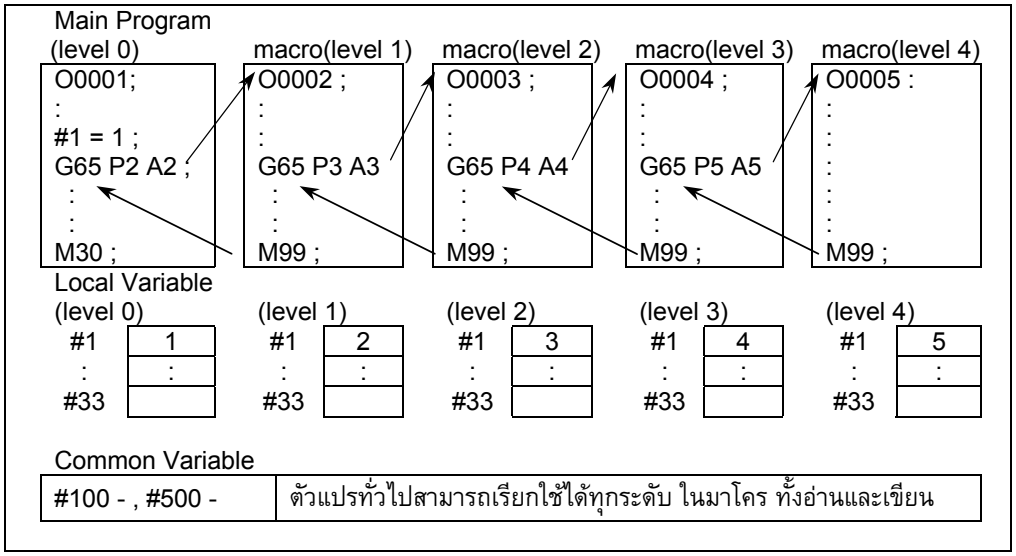
ข้อกำหนด

1. รูปแบบ คือ G65 ตามด้วย อากิวเมนต์
2. อากิวเมนต์ แต่ละแบบไม่สามารถใช้ผสมกันได้ หากมีการกำหนดผสมกัน อากิวเมนต์แบบหลังจะทำงาน
3. การใช้ หรือ ลืมกำหนดจุดทศนิยม ของค่าตำแหน่ง หรือ ค่าใดๆ ของอากิวเมนต์ ขึ้นอยู่กับระบบของเครื่องที่ตั้งไว้ หรือค่าสุดท้ายที่ป้อนให้กับการเคลื่อนที่ อย่างไรก็ดี แนะนำว่าไม่ควรลืมการใส่ทศนิยม เช่น ค่า 2 ควรใส่ 2.0 มิฉะนั้นอาจเป็น 0.002
4. การเรียกโปรแกรมย่อย หรือ ซ้อน (Nesting) ในมาโครนี้ สามารถทำได้ไม่เกิน 4 ระดับ โดยใช้ G65 หรือ G66 แต่ไม่นับรวมการเรียกโดย M98

ระดับของตัวแปรภายใน

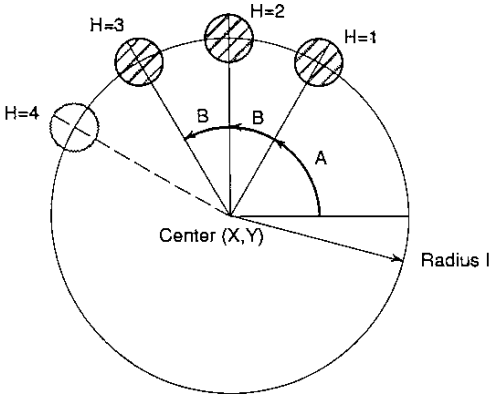
1. ระดับของตัวแปรภายใน จะมีการเปลี่ยนได้จาก 0 – 4 ในขณะที่มีการเรียกโปรแกรมย่อย (ซ้อน)
2. ระดับของโปรแกรมหลัก (Main Program) จะเป็น 0
3. ทุกครั้งที่มีการเรียกโปรแกรมย่อย(ซ้อน) โดย G65 หรือ G66 ระดับของตัวแปรภายในจะมีการเปลี่ยน โดยของเดิมจะถูกเก็บไว้ในหน่วยความจำของเครื่อง โดยเพิ่มทีละ 1

4. เมื่อพบคำสั่ง M99 จะกลับไปสู่โปรแกรมที่เรียกมา และระดับของตัวแปรภายใน ก็จะลดลง 1 ระดับ เช่นกัน โดยจะเรียกค่าตัวแปรที่เก็บไว้ก่อนหน้ามาใช้งาน



ตัวอย่างโปรแกรม

โปรแกรมมาโครนี้ เป็นโปรแกรมสำหรับ การเจาะรู หลายรู (H) ด้วยรัศมี (I) จากมุมเริ่มต้น (A) และเลื่อนไปตามองศาที่กำหนด (B) มีจุดศูนย์กลางที่ (X,Y)



รูปแบบคำสั่ง

G65 P9100 Xx Yy Zz Rr Ff Ii Aa Bb Hh ;

X = พิกัดตำแหน่งของ จุดศูนย์กลางวงกลม ในแกน X (#24)

Y = พิกัดตำแหน่งของ จุดศูนย์กลางวงกลม ในแกน Y (#25)

Z = ความลึกของการเจาะ (#26)

R = ตำแหน่งจุดที่จะทำการเจาะครั้งแรก (#18)

F = อัตราป้อนในการเจาะ (#9)

I = รัศมีของวงกลม (#4)

A = มุมของจุดเริ่มต้นที่จะเจาะ (#1)

B = องศาที่เพิ่มขึ้นของแต่ละรูเจาะ (#2) (หากค่าเป็นลบ จะนับตามเข็มนาฬิกา)

H = จำนวนรูที่จะเจาะ

โปรแกรมหลักที่เรียกมาโคร (Main Program)

```
O0002;
G90 G92 X0 Y0 Z100.0;
G65 P9100 X100.0 Y50.0 R30.0 Z-50.0 F500 I100.0 A0 B45.0 H5;
M30;
```

ส่วนของมาโครโปรแกรม (ส่วนที่เรียกใช้งาน)

```
O9100;
  #3=#4003;                (บันทึกรหัส G กลุ่ม 3 ลงในตัวแปร )
  G81 Z#26 R#18 F#9 K0     (คำสั่งวัฐจักรเจาะรู /  ใช้ L0 แทน K0 ก็ได้ )
  IF[#3 EQ 90] GOTO1;      (ข้ามไปบรรทัด N1 หากอยู่ใน Mode G90)
  #24=#5001+#24;           (ใช้ตัวแปรระบบ คำนวณตำแหน่งจุดศูนย์กลางในแกน X)
  #25=#5002+#25;           (ใช้ตัวแปรระบบ คำนวณตำแหน่งจุดศูนย์กลางในแกน Y)
N1 WHILE[#11 GT 0]DO1;     ( ทำคำสั่งต่อไปจนจำนวนรูเจาะเป็น 0 )
  #5=#24+#4*COS[#1];       ( คำนวณตำแหน่งจุดที่จะเจาะในแกน X )
  #6=#25+#4*SIN[#1];       ( คำนวณตำแหน่งจุดที่จะเจาะในแกน X )
  G90 X#5 Y#6;             ( เคลื่อนที่ไปทำการเจาะ ณ ตำแหน่งจุดที่คำนวณได้ )
  #1=#1+#2                 ( คำนวณองศาของจุดเจาะใหม่ )
  #11=#11-1                ( ลดค่าจำนวนรูที่จะเจาะ )
 END1;
  G#3 G80;                 ( คืนค่ารหัส G กลับไปเช่นเดิม )
 M99;
```

ความหมายของตัวแปร	
#3	บันทึกค่ารหัส G ของกลุ่ม 3 (G-code Group 3)
#5	ค่าในแกน X ของรูที่จะทำการเจาะถัดไป
#6	ค่าในแกน Y ของรูที่จะทำการเจาะถัดไป

6.2 การเรียกแบบ Modal Call, G66 G67

MODAL มาจากภาษาลาติน คือ Modus หมายถึง ลักษณะ หรือ ประเภท (manner) ใน CNC Program จะหมายถึง คำสั่งที่เป็นกลุ่ม และคงอยู่ จนกว่าจะมีการคำสั่งอื่นมาทำการเปลี่ยนแปลงคำสั่งนั้นๆ

G66 เป็นคำสั่งสำหรับใช้เรียกมาโครที่มีคำสั่งในกลุ่ม Modal เพื่อการสั่งเคลื่อนที่ในแกนต่างๆ คำสั่งนี้ยกเลิกโดย G67

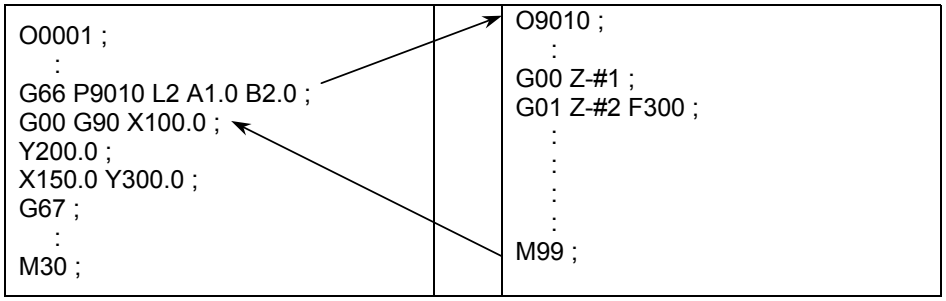
รูปแบบ

G66 Pp L l <agrument-specification>

p : หมายเลขโปรแกรมที่เรียก

l : จำนวนรอบที่ต้องการ (หากไม่เติมจะเป็น 1)

agrument : ข้อมูลที่จะส่งเข้าสู่โปรแกรม



คำอธิบาย

- คำสั่ง G66 จะเรียกมาโคร โดยกำหนดหมายเลขโปรแกรมหลัง P
- ค่าตัวเลข (1 – 9999) คือ จำนวนรอบที่จะทำงานซ้ำๆ
- สามารถส่งค่าตัวแปร เข้าในมาโคร ได้เหมือน G65
- G67 ใช้เพื่อยกเลิก การเรียกมาโครแบบ modal

การเรียกซ้อน (Nesting)

การเรียกซ้อน ทำได้ลึกถึง 4 ระดับ รวมถึงการเรียกแบบง่ายด้วย G65 และเรียกแบบ Modal โดย G66 ซึ่งจะไม่รวมถึงการเรียกแบบโปรแกรมย่อย (subprogram) M98

Modal Call Nesting (เรียกซ้อนในโหมด Modal)

แบบ Modal นี้สามารถเรียกซ้อนกัน โดยใช้คำสั่ง G66

โปรแกรมตัวอย่าง

ตัวอย่างโปรแกรมนี้ เป็น วัฐจักรการเจาะส่วาน

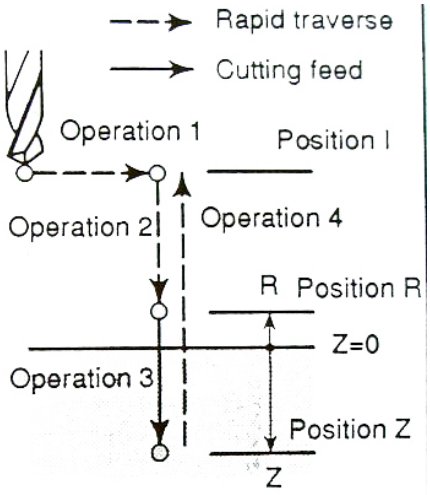
The canned cycle consists of the following basic operations:

Operation 1:
Positioning along the X–axis and Y–axis

Operation 2:
Rapid traverse to point R

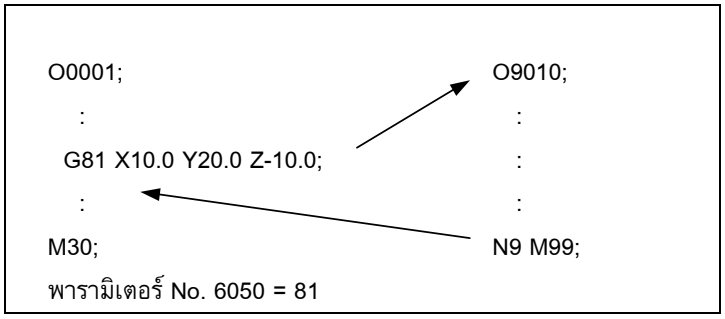
Operation 3:
Cutting feed to point Z

Operation 4:
Rapid traverse to point R or I



6.3 การเรียกโดยรหัส G

มาโครสามารถเรียกได้โดยการกำหนดตัวเลข ที่จะใช้เรียกในพารามิเตอร์ เพื่อเป็น G พิเศษสำหรับเรียกมาโคร ซึ่งการทำงานจะเหมือนกับคำสั่ง G65



คำอธิบาย

เมื่อทำการปรับเปลี่ยนค่าระหว่าง 1 – 9999 ในพารามิเตอร์ 6050 – 6059 เมื่อใช้ G ตามด้วยเลขนั้นๆ จะทำการเรียกมาโครโปรแกรม O9010 – O9019 (ปกติไม่ควรใส่ค่าตัวเลขที่เหมือนกับ G-code ปกติที่มีอยู่) ซึ่งการเรียกจะเหมือนกับการใช้ G65

เช่น เมื่อป้อนค่า 81 ให้กับพารามิเตอร์ 6050 เมื่อสั่ง G81 จะทำให้เกิดการเรียกมาโคร O9010 มาใช้งาน

โปรแกรมและพารามิเตอร์ที่เกี่ยวข้องกัน

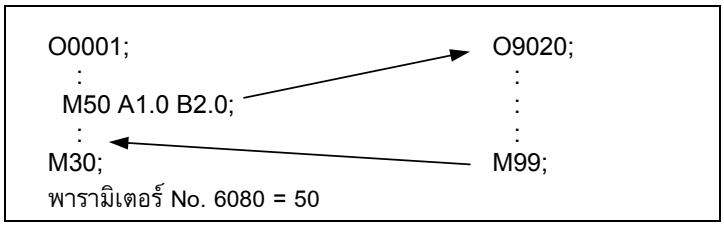
Program Number	Parameter Number
O9010	6050
O9011	6051
O9012	6052
O9013	6053
O9014	6054
O9015	6055
O9016	6056

O9017	6057
O9018	6058
O9019	6059

- การทำซ้ำๆ สามารถสั่งได้โดยกำหนดค่าหลัง L จำนวน 1 – 9999 ครั้ง
- การกำหนดอากิวเมนต์ สามารถทำได้ทั้งสองแบบ เช่นกัน เหมือนกับการเรียกปกติ
- ข้อจำกัด คือ การเรียกแบบซ้อน (Nesting) โดยเรียกมาโครซ้อนอีกไม่ได้ แต่สามารถเรียกโปรแกรมย่อยได้ โดย M หรือ T code ถึงแม้ว่าการกำหนดโดยวิธีนี้จะเหมือนกับ G-code ทั่วไป แต่เป็นการเรียกมาโครโปรแกรมแบบตรงตัว จึงเรียกมาโครโปรแกรมหมายเลขอื่น ซ้อนอีกไม่ได้

6.4 การเรียกโดยรหัส M

เราสามารถเรียกมาโครโปรแกรมด้วยคำสั่ง M ได้เช่นเดียวกับคำสั่ง G65 เช่นกัน



โดยกำหนดค่าให้กับพารามิเตอร์ 6080 – 6089 ด้วยค่าตัวเลขระหว่าง 1 – 99999999 เพื่อใช้เรียกมาโครโปรแกรมหมายเลข 9020 – 9029 โดยการเรียกจะเหมือนกับการใช้ G65

โปรแกรมและพารามิเตอร์ที่เกี่ยวข้องกัน

Program Number	Parameter Number
O9020	6080
O9021	6081
O9022	6082
O9023	6083
O9024	6084
O9025	6085
O9026	6086
O9027	6087
O9028	6088
O9029	6089

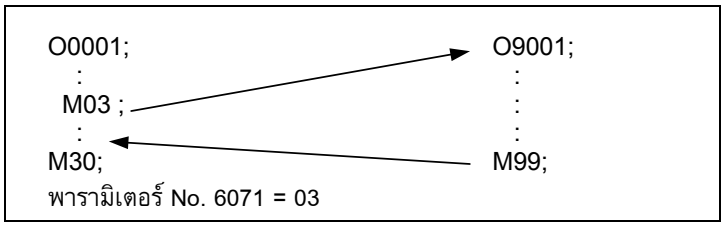
- การทำซ้ำๆ สามารถสั่งได้โดยกำหนดค่าหลัง L จำนวน 1 – 9999 ครั้ง
- การกำหนดอากิวเมนต์ สามารถทำได้ทั้งสองแบบ เช่นกัน เหมือนกับการเรียกปกติ

ข้อจำกัด

- คำสั่ง M นี้จะต้องอยู่เป็นคำสั่งแรก บนบรรทัดนั้นๆ
- ไม่สามารถเรียก มาโครหมายเลขอื่นได้ ถึงแม้จะมีการสั่งงานเหมือน G code

6.5 การเรียกแบบโปรแกรมย่อย ด้วยรหัส M

คำสั่ง M แบบนี้จะเป็นการสั่งเพื่อเรียกโปรแกรมย่อย เหมือน M98 โดยที่โปรแกรมย่อยมีเนื้อความเป็นมาโครโปรแกรม



โดยทำการกำหนดค่า ระหว่าง 1 ถึง 99999999 ให้กับพารามิเตอร์ 6071 – 6079 ซึ่งจะไปเรียกโปรแกรมย่อยหมายเลข O9001 – O9009 และมีการทำงานเหมือนกับเรียกโปรแกรมย่อย (Sub Program) M98

โปรแกรมและพารามิเตอร์ที่เกี่ยวข้องกัน

Program Number	Parameter Number
O9001	6071
O9002	6072
O9003	6073
O9004	6074
O9005	6075
O9006	6076
O9007	6077
O9008	6078
O9009	6079

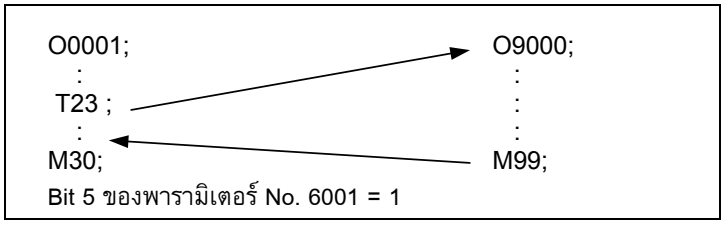
- การทำซ้ำๆ สามารถสั่งได้โดยกำหนดค่าหลัง L จำนวน 1 – 9999 ครั้ง
- การกำหนดอากิวเมนต์ ทำไม่ได้
- คำสั่ง M ที่เรียกมาโครย่อยนี้ จะเหมือนกับการเรียกโปรแกรมย่อยปกติ

ข้อจำกัด

- กรณีที่เรียกมาโครย่อย ด้วย G code หรือ M หรือ T จะไม่อนุญาติให้เรียก มาโครโปรแกรมย่อย ด้วยคำสั่ง M แบบนี้เพิ่มซ้อนเข้าไปได้อีก ถึงแม้จะมีการรับรู้คำสั่งเหมือน G code ปกติก็ตาม

6.6 การเรียกจากโปรแกรมย่อย ด้วยรหัส T

เมื่อกำหนดค่าให้กับพารามิเตอร์ที่เกี่ยวข้อง เราสามารถที่จะเรียกโปรแกรมย่อยด้วย คำสั่ง T ได้ ตามจำนวนครั้งที่เรียกในโปรแกรมหลัก



- การกำหนดให้ทำการเรียกมาโครโปรแกรม O9000 ทำโดยการเปลี่ยนค่า บิทที่ 5 ของพารามิเตอร์ TCS หมายเลข 6001 ให้เป็น 1 และทำการกำหนดเลขรหัสที่จะใช้ในตัวแปรหมายเลข #149 เช่น ตามตัวอย่าง #149 = 23

ข้อจำกัด

- กรณีที่เรียกมาโครย่อย ด้วย G code หรือ M หรือ T จะไม่อนุญาติให้เรียก มาโครโปรแกรมย่อย ด้วยคำสั่ง T แบบนี้เพิ่มซ้อนเข้าไปได้อีก ถึงแม้จะมีการรับรู้คำสั่งเหมือน T code ปกติก็ตาม

บทที่ 7 การทำงานของคำสั่งมาโคร

ปกติส่วนควบคุมเครื่องจักร จะทำการอ่านบรรทัดคำสั่งล่วงหน้า เพื่อรักษาความต่อเนื่องของการทำงาน ซึ่งเราเรียกว่า Buffering

ในกรณีที่เป็นการทำงานในโหมด ชดเชยขนาด (Compensation G41, G42) เครื่องจะอ่านคำสั่งล่วงหน้าประมาณ 2 - 3 บรรทัด ของคำสั่งที่มีการเคลื่อนที่ หรือ มีจุดพิกัดที่กำหนดตำแหน่ง

ส่วนมาโครจะแตกต่างกัน คือ หากเป็นบรรทัดที่เป็นการคำนวณ หรือ กำหนดค่า จะทำการอ่านบรรทัดคำสั่งและทำไปเรื่อยๆ โดยไม่มีการหยุด จนกว่าจะพบบรรทัดที่มีคำสั่งในการเคลื่อนที่

บรรทัดที่มีคำสั่ง M00, M01, M02, M30, M-code อื่นๆ ที่กำหนดในพารามิเตอร์ 3411 - 3420, และ G31 จะไม่มีการอ่านล่วงหน้า

- กรณี ที่บรรทัดคำสั่งต่อไป เป็น Buffer
- กรณี ที่คำณวนต่อจนพบ บรรทัดคำสั่ง แบบอื่น

บทที่ 8 การบันทึกมาโครโปรแกรม

มาโครโปรแกรม จะถูกบันทึกเช่นเดียวกันกับ โปรแกรมย่อย การกำหนดเลขโปรแกรมก็ไม่ต่างกับโปรแกรมหลัก ซึ่งจะมีการใช้พื้นที่ ในหน่วยความจำในเครื่องเช่นเดียวกัน ตลอดรวมจนถึงการแก้ไขหน้าเครื่องด้วย

บทที่ 9 ข้อจำกัด

มาโครจะมีข้อกำหนดต่างๆ คือ

- โหมด MDI — ใน MDI สามารถเรียกมาโครโปรแกรมได้ แต่ในระหว่างทำงานในแบบอัตโนมัติ คงเป็นไปไม่ได้ ที่จะมีการสับสวิทซ์ ไปยัง MDI เพื่อเรียกโปรแกรมมาโคร
- ค้นหาเลขบรรทัด (Sequence number)
- Single Block
- Optional Block skip
- การทำงานใน Edit Mode
- Reset
- Feed Hold
- ค่าคงที่ใน สูตรคำนวณ (expression) — ค่าคงที่ ที่สามารถใช้ได้ในมาโคร คือ

  +0.0000001 ถึง +99999999

  -99999999 ถึง -0.0000001

  ค่ากำหนดได้เพียงทศนิยม 8 หลัก หากค่าที่กำหนดไม่อยู่ในพิสัย จะเกิด P/S Alarm No.003

บทที่ 10 การส่งค่าออกจากเครื่อง

ในส่วนนี้จะเป็นคำสั่งที่ ผู้ใช้สามารถส่งค่าออกจากเครื่อง (Output Command to External) โดยส่วนใหญ่ เนื้อหาอธิบายผลลัพธ์ที่จะเกิดเมื่อต่อกับ เครื่องพิมพ์เทป

ซึ่งจะมีคำสั่งที่ใช้เหล่านี้

- BPRINT
- DPRINT
- POPEN
- PCLOS

ซึ่งในปัจจุบันไม่ค่อยมีใช้ เครื่อง Reader/Puncher แล้ว เพราะเป็นการใช้เทปกระดาษ

ดังนั้นจึงขอข้ามเนื้อหาในส่วนนี้

บทที่ 11 การแทรกซ้อนโปรแกรม (Interruption Type)

บทส่งท้าย

การจะเขียนหรือ สร้างโปรแกรมมาโคร ในระดับสูงและซับซ้อนได้นั้น ขอแนะนำให้ผู้อ่านได้ศึกษาเพิ่มเติม เรื่อง วิธีการเขียนโปรแกรมภาษาของระบบคอมพิวเตอร์ ควบคู่กันไปด้วย เช่น ภาษาเบสิค ภาษาซี โคบอล เป็นต้น ขอให้เป็นโปรแกรมที่ต้องมีการเขียนตัวภาษา และโครงสร้างเอง โดยมีสิ่งต่างๆที่ควรทราบ เพื่อเสริมสร้างความรู้ความเข้าใจเพิ่มเติม อันมาจากวิธีการเขียนโปรแกรมคอมพิวเตอร์ คือ

1. การวิเคราะห์ โครงสร้าง (System Analyzes)
2. การสร้าง Flow Chart
3. ระบบตัวแปร (Variable)
4. Loop และ อัลกอริทึม
5. การสร้าง-กำหนด เงื่อนไข (If cause)
6. ระบบเลขฐานสอง ฐานแปด ฐานสิบหก
7. รหัส ASC II
8. อื่นๆ

การเรียนรู้เรื่อง Computer Programming นี้ อาจดูเหมือนไม่จำเป็น แต่จากประสพการณ์ส่วนตัวจากการที่ตนเองได้เรียนรู้สิ่งต่างๆข้างต้นมาก่อน จนสามารถเขียนโปรแกรมคอมพิวเตอร์ เพื่อใช้เองในองค์กรได้ เมื่อต้องมาเรียนรู้ การเขียน G-Code และ มาโคร ปรากฏว่าสามารถเข้าใจได้ง่ายและรวดเร็ว เพราะคุ้นเคยศัพท์เทคนิคและข้อจำกัดหรือกำหนดต่างๆเหล่านั้นมาแล้ว เป็นอย่างดี เพียงแค่ปรับเปลี่ยนความรู้สึกเล็กน้อย แต่หลักการไม่แตกต่าง

บทแปลนี้ มีความประสงค์จะทำเป็นวิทยาทาน และขออภัยที่ต้องทำการ lock ต้นฉบับ ไม่ให้แก้ไขเองได้ (แม้ว่าบางคนอาจจะเปิด lock ได้ก็ตาม)

ขอให้ทุกท่านได้นำไปศึกษาหาความรู้เพิ่มเติม หากมีข้อผิดพลาด หรือไม่ถูกต้องประการใด เมื่อนำไปลองเขียนใช้งาน ขอความกรุณาแจ้งให้ทราบด้วย เพื่อทำการแก้ไขต้นฉบับ และ จัดส่งกลับไปให้ผู้ได้รับท่านอื่นต่อไป