

HƯỚNG DẪN SỬ DỤNG BRAIN TRONG MACH3

Mục lục

I. Tổng quát về Brain trong Mach3.....	3
1. Giới thiệu về Brain trong Mach3.....	3
2. Brain hoạt động như thế nào?.....	3
3. Cấu trúc một Brain	3
4. Ví dụ thực tế	3
5. Ứng dụng thực tế của Brain.....	4
6. Cách tạo Brain	4
7. Lưu ý.....	4
II. Thao tác với Brain.....	5
I.1. Chức năng “Add”.....	5
I.1.1. Thêm một dòng xử lý mới vào Brain.....	5
I.1.2. Add thêm một xử lý logic vào dòng xử lý đã có.....	17
I.2. Chức năng Delete	18
I.3. Chức năng Terminate Lobe	18
I.3.1. INPUTS.....	19
I.3.2. OUTPUTS	19
I.3.3. DRO's	19
I.3.4. Modbus	19
I.3.4. SendString	20
I.3.4. Local Var	20
I.3.4. ButtonPress	21
I.3.4. User LED's.....	23
I.3.4. Mach Control.....	23
III. Một số dự án thực tế với Brain	24
III.1. Dự án tạo tay cầm ModBUS MPG.....	24
III.1.1. Bật chế độ cho phép sử dụng ModBus trên Mach3	25
III.1.2. Thiết lập cấu hình Serial Modbus Control.....	26

III.1.3. Thiết lập Brain xử lý dữ liệu nhận được từ ModBus để điều khiển Mach3	28
III.1.4. Kích hoạt cho Brain tạo ở bước 3 chạy trong Mach3	40
III.2. Dự án điều khiển các van kẹp phôi tự động	42
II.2.1. Ý tưởng của dự án.....	42
III.2.2. Lô gic xử lý vấn đề của dự án:	43
III.2.3. Tiến hành tạo Brain	43
III.2.4. Kích hoạt cho brain chạy	45

I. Tổng quát về Brain trong Mach3

1. Giới thiệu về Brain trong Mach3

Brain trong Mach3 là một **hệ thống lập trình logic dạng khối**, cho phép bạn xử lý tín hiệu đầu vào/đầu ra (I/O), điều kiện và trạng thái hệ thống **mà không cần viết code VBScript hay plugin**.

Nó giống như một **PLC mini** được tích hợp sẵn, giúp người dùng xử lý các chức năng tự động hoặc đặc biệt – chẳng hạn như bật/tắt relay, xử lý cảm biến, công tắc, nút nhấn, điều kiện chạy máy, v.v.

2. Brain hoạt động như thế nào?

Một **Brain** gồm nhiều **"logic blocks"** (khối logic), mỗi khối làm nhiệm vụ:

- **Lấy dữ liệu từ đầu vào hoặc biến hệ thống**
- **Xử lý dữ liệu (so sánh, logic, toán học...)**
- **Xuất ra đầu ra số, analog hoặc tác động trạng thái hệ thống**

Các khối này được nối với nhau như **sơ đồ mạch điều khiển**.

3. Cấu trúc một Brain

Mỗi Brain bao gồm nhiều dòng (line), mỗi dòng là một **chuỗi xử lý logic**.

Một dòng Brain có thể gồm các phần sau:

Thành phần	Vai trò
Input	Đọc tín hiệu (ví dụ: từ công tắc, cảm biến, hoặc biến hệ thống như IsMoving, THC On,...)
Function	Các hàm xử lý logic (AND, OR, Compare, Flip Flop, Timer,...)
Output	Tác động đầu ra (ví dụ: bật Output #1, set DRO, bật MPG mode,...)

4. Ví dụ thực tế

Mục tiêu: Bật Output #1 nếu Input #3 đang ở mức cao và máy đang chạy (IsMoving = true)

1. Input: **Port 1 Pin 3** (ngõ vào số 3)
2. Function: **AND**

3. Input khác: **IsMoving**
4. Output: **Output #1**

→ Sơ đồ logic:

Input#3 —┐ AND → Output#1
IsMoving ┘

5. Ứng dụng thực tế của Brain

- Tự động điều khiển đèn, còi, quạt làm mát
- Giao tiếp với cảm biến hành trình, báo lỗi
- Điều khiển tốc độ quạt/cắt theo thời gian thực
- Điều kiện an toàn: không cho phép chạy nếu chưa bật hút bụi
- Tự reset lỗi khi có tín hiệu ngoài

6. Cách tạo Brain

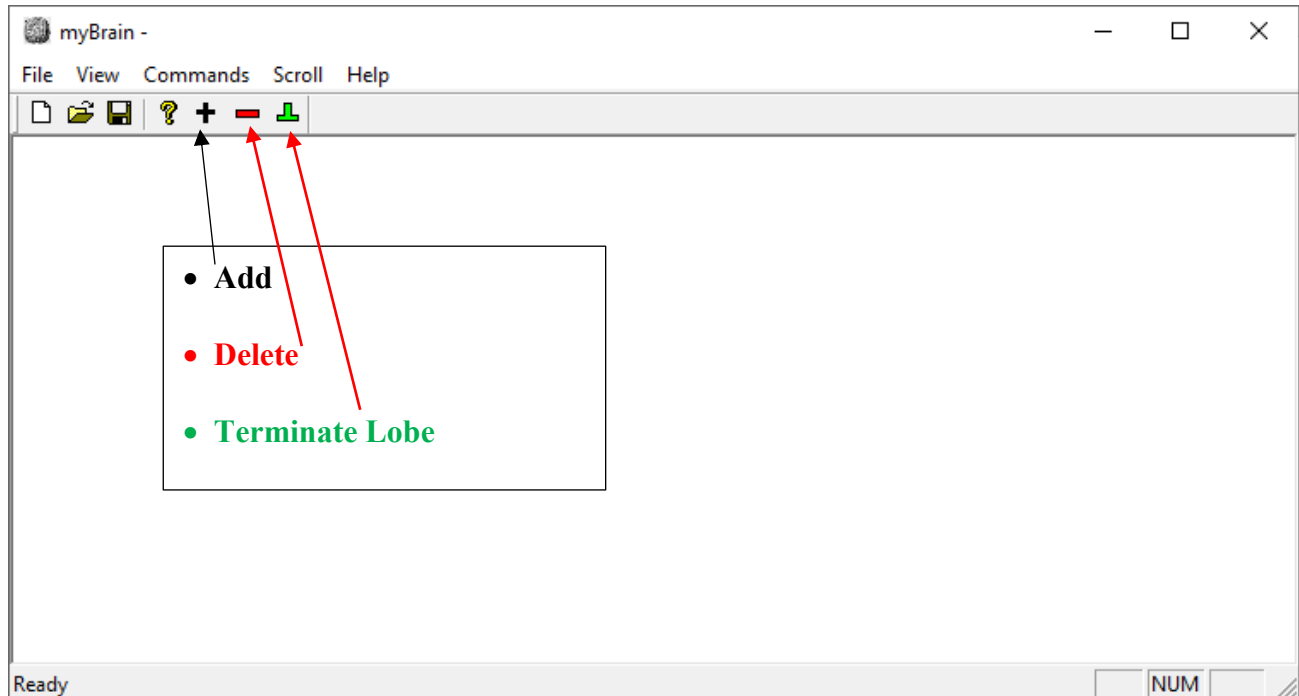
1. Mở **Mach3**
2. Vào menu **Operator > Brain Editor**
3. Tạo mới hoặc mở Brain hiện có
4. Thêm các dòng xử lý (Input > Function > Output)
5. Lưu lại với đuôi .brn
6. Vào **Operator > Brain Control**, tải Brain vào và kích hoạt.

7. Lưu ý

- Brain chạy **song song và theo thời gian thực** với hệ thống chính.
- Rất nhẹ, ổn định và không cần reset như macro.
- Khó debug hơn so với code, nhưng đơn giản nếu dùng đúng.
- Có thể kết hợp macro + brain để xử lý các tình huống phức tạp.

II. Thao tác với Brain

Màn hình làm việc của Brain chỉ có 3 nút công cụ cơ bản là Add (thêm), Delete (Xóa) và Terminate Lobe, do vậy không quá khó để nhớ đến ba nút này.



Hình 1. Màn hình chính của trình soạn thảo Brain

- ☐ **Add:** Thêm một dòng xử lý (lobe) mới vào Brain hoặc thêm một xử lý logic vào dòng hiện có.
- ☐ **Delete:** Xóa dòng xử lý đang được chọn.
- ☐ **Terminate Lobe:** Kết thúc dòng xử lý (giống như kết thúc nhánh xử lý tại điểm đó).

I.1. Chức năng “Add”

I.1.1. Thêm một dòng xử lý mới vào Brain

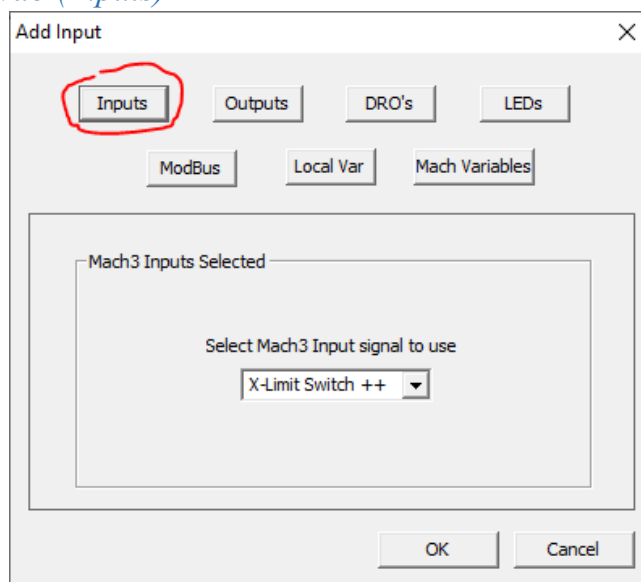
Khi bấm **Add** để tạo một dòng mới, cửa sổ hình 2 - Add Input hiện lên. Dữ liệu xem bảng 1.

Bảng 1. Tổng hợp các dữ liệu đầu vào/ra dùng cho Brain

TT	Nhóm tín hiệu	Mô tả chức năng	Danh sách chi tiết
1	INPUTS	Đọc giá trị các đầu vào	Bảng 2
2	OUTPUTS	Đọc giá trị các đầu ra	Bảng 3
3	DRO's	Đọc các giá trị số hiển thị trên giao diện người dùng, như tọa độ trục, tốc độ trục chính, v.v.	Bảng 4
4	LEDs	Đọc trạng thái các đèn báo trạng thái trên giao diện người dùng, như trạng thái chạy, dừng, lỗi, v.v.	Bảng 5

5	Modbus	Đọc các tín hiệu từ thiết bị Modbus.	
6	Local Var	Đọc các biến cục được sử dụng trong Brains để lưu trữ và xử lý dữ liệu	Có 100 biến từ V0 ~ V99
7	Mach3 System Variables	Đọc các biến hệ thống của Mach3	Bảng 6

1. Nhóm các biến đầu vào (Inputs)

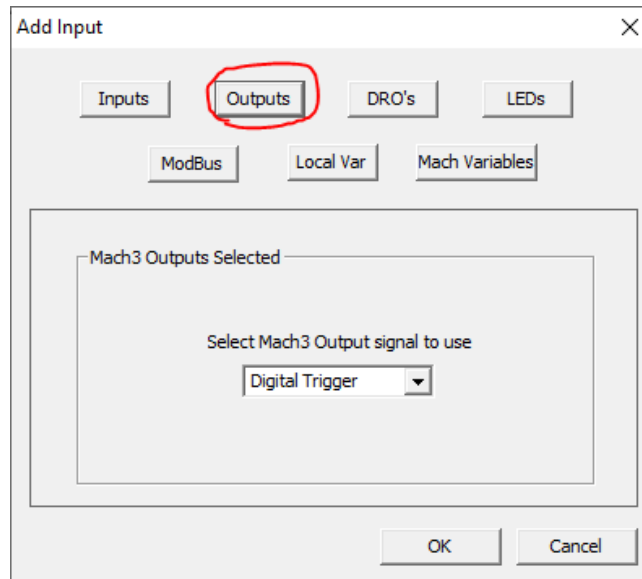


Hình 2. Màn hình giao diện Add Input

Bảng 2. Nhóm tín hiệu đầu vào (Inputs)

X-Limit Switch ++	B-Home Switch	THC Down Command	OEMTRIGGER #14
X-Limit Switch --	C-Limit Switch ++	OEMTRIGGER #1	OEMTRIGGER #15
X-Home Switch	C-Limit Switch --	OEMTRIGGER #2	OEMTRIGGER #16
Y-Limit Switch ++	C-Home Switch	OEMTRIGGER #3	OEM Input Slot
Y-Limit Switch --	INPUT #1	OEMTRIGGER #4	Jog X ++
Y-Home Switch	INPUT #2	OEMTRIGGER #5	Jog X --
Z-Limit Switch ++	INPUT #3	OEMTRIGGER #6	Jog Y ++
Z-Limit Switch --	INPUT #4	OEMTRIGGER #7	Jog Y --
Z-Home Switch	Probe Switch	OEMTRIGGER #8	Jog Z ++
A-Limit Switch ++	Spindle Index	OEMTRIGGER #9	Jog Z --
A-Limit Switch --	Limit Switch OverRide	OEMTRIGGER #10	Jog A ++
A-Home Switch	Estop	OEMTRIGGER #11	Jog A --
B-Limit Switch ++	Torch Out	OEMTRIGGER #12	
B-Limit Switch --	THC Up command	OEMTRIGGER #13	

2. Nhóm các tín hiệu đầu ra (Outputs)



Bảng 3. Nhóm các biến trong tín hiệu đầu ra (Outputs)

Charge pump #1	OUTPUT1	OUTPUT19
Charge pump #2	OUTPUT10	OUTPUT2
Current Hi/Low	OUTPUT11	OUTPUT20
Digital Trigger	OUTPUT12	OUTPUT3
Enable1	OUTPUT13	OUTPUT4
Enable2	OUTPUT14	OUTPUT5
Enable3	OUTPUT15	OUTPUT6
Enable4	OUTPUT16	OUTPUT7
Enable5	OUTPUT17	OUTPUT8
Enable6	OUTPUT18	OUTPUT9

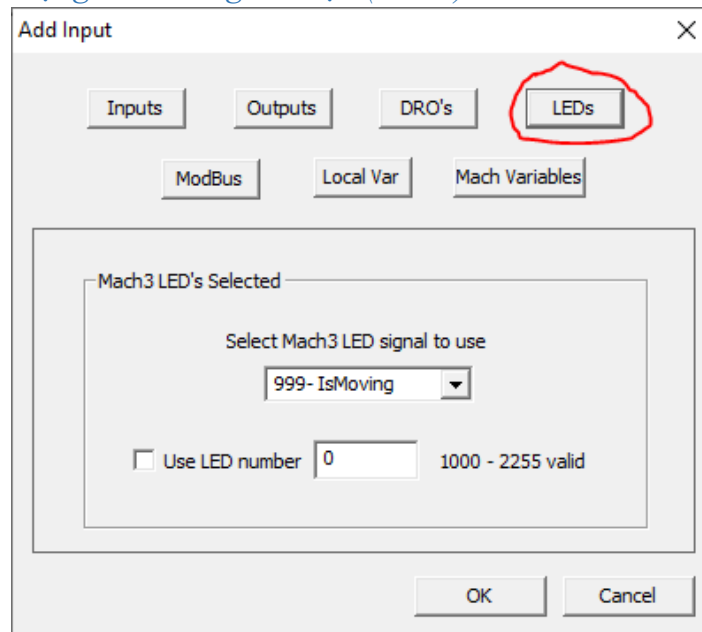
3. Nhóm tín hiệu hiển thị số trên giao diện (DROs)

Bảng 4. Nhóm tín hiệu hiển thị số (DROs)

842-X Position	36-Homed Z	97-MPG1 Pos	160-SoftMin B
843-Y Position	37-Homed Off	98-MPG3 Pos	161-SoftMin C
844-Z Position	38-Homed Off	100-MPG3 Pos	166-Torch Speed
845-A Position	39-True Spindle RPM	101-MPG4 Count	167-Cond SpindleRPM
846-B Position	40-Tool offset	102-MPG1 Count	170-Encoder 0 Cnt
847-C Position	42-Tool offset	103-MPG2 Count	171-Encoder 1 Cnt
800-X Feedrate	43-Tool Dia	104-Rapid Rate	172-Encoder 2 Cnt
801-Y Feedrate	44-Tool Radius	105-Tool Dia	173-Encoder 3 Cnt
802-Z Feedrate	46-Coordinate system	106-Tool Direction	174-Encoder 4 Cnt
803-A Feedrate	47-Fix Y off	107-Tool Radius	175-Encoder 5 Cnt
804-B Feedrate	48-Fix X off	110-T2 Wear	192-Freeze Delay
805-C Feedrate	49-Fix Z off	111-T2 Wear	193-ToGoX
806-F Feedrate	50-Fix A off	112-Insert Angle	194-ToGoY
807-S Feedrate	51-Fix B off	113-MPG0 Velocity	195-ToGoZ
808-F Feedrate	52-Fix C off	114-MPG1 Velocity	196-ToGoA
810-D Feedrate	53-CVU Speed	115-MPG2 Velocity	197-ToGoB
811-E Feedrate	54-Soft Z	116-MPG3 Velocity	198-ToGoC
813-CBlend Feedrate	55-Feedrate	117-CRC rotation	199-ToGoU
1-JobSlice Inc	56-Current Pulley	118-G68 R or G100	200-ToGoG

2-PulseFreq	57-Pulley max speed	119-Current Pulley M	202 - Spin RPM - Ovr%
3-Jogging	58-Feed per rev	127-Encoder LAG X	203 - Encoder Ovr. P
4-XJob Min	59-X Scale	128-Encoder LAG Y	204 - Encoder Ovr. P
5-YJob Min	60-Y Scale	129-Encoder LAG Z	205 - Encoder Ovr. P
6-ZJob Min	61-Z Scale	130-Encoder LAG A	206 - Encoder Ovr. P
7-AJob Min	62-A Scale	131-Encoder LAG B	207 - Encoder Ovr. P
8-BJob Min	63-B Scale	132-Encoder LAG C	208 - Encoder Ovr. P
9-CJob Min	64-C Scale	133-Encoder LAG U	210 - JobMaxX ABS.
10-XJob Max	65-THCMin	134-Pulsecount X	211 - JobMaxY ABS.
11-YJob Max	66-Thread Pentry ang	135-Pulsecount Y	212 - JobMaxZ ABS.
12-ZJob Max	74-Spindle %	136-Pulsecount Z	213 - JobMinX ABS.
13-AJob Max	77-Laser X grid	137-Pulsecount A	214 - JobMinY ABS.
14-BJob Max	78-Spare	138-Pulsecount B	215 - JobMinZ ABS.
15-CJob Max	79-Init Bit	139-Pulsecount C	216 - SpindleCur ABS.
16-G92X Off	80-Init Bit	140-Pulsecount U	217 - Spindle Cur %
17-G92Y Off	81-Capture Pos	141-EncoderCnt 0	818 - Current Line
18-G92Z Off	82-G53G0 Pos	142-EncoderCnt 1	819 - Current Mode
19-G92A Off	83-G53G1 Pos	143-EncoderCnt 2	821 - Feed Ord
20-G92B Off	84-G53G2 Pos	144-EncoderCnt 3	822 - Current Tool
21-G92C Off	85-G53G3 Pos	145-EncoderCnt 4	823 - FeedRate%
22-Queue Depth	86-G53G4 Pos	146-EncoderCnt 5	824 - B Radius
23-TimeSlice	87-G53G5 Pos	150-SoftMax X	825 - C Radius
24-Spindle Speed	88-G53G6 Pos	151-SoftMax Y	826 - G Radius
25-THC Pos	89-Feed Factor	152-SoftMax Z	827 - P Radius
28-BufferLoad	90-Spin Spindle	153-SoftMax A	830 - Planned Inc.
29-MPG0 Pos	91-G76s Input	154-SoftMax B	831 - X Jog Feed%
30-MPG1 Pos	92-Knife Lift angle	155-SoftMax C	832 - Y Jog Feed%
31-Tool offset	93-Knife Lift vel	156-SoftMin X	833 - Z Work Offset
33-THC Offset	94-Fedrate	157-SoftMin Y	834 - A Work Offset
34-Homed X	95-CEM Rdo	158-SoftMin Z	835 - C Work Offset
35-Homed Y	96-MPG0 Pos	159-SoftMin A	

4. Nhóm tín hiệu đèn trạng thái trên giao diện (LEDs)

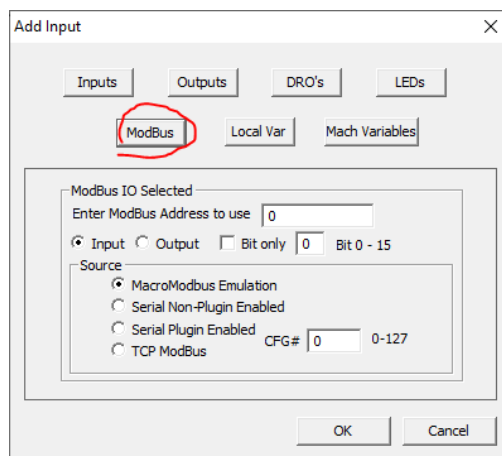


Bảng 5. Nhóm tín hiệu các đèn báo (LEDs)

0 - IsMoving	30 - 4x Scaled	60 - UnHide	101 - Using Formulas
1 - Spindle On	31 - 5x Scaled	61 - Spindle Preload	111 - Partial line hold
2 - M1 Cmd	32 - 6x Scaled	62 - Hardware EStop Trig	112 - Reverse run
3 - Jog Inc.	33 - 7x Scaled	63 - Non-GURLEC	162 - Macro running
4 - Jog Cont.	34 - 8x Scaled	64 - Spindle Stable	163 - Mcode waiting
5 - Program Coord	35 - G91 Mode	65 - UNcode ABS	164 - Spindle CW on
6 - Machine Coord	36 - G90 Mode	66 - TMode PRC	165 - Spindle CCW on
7 - Distance Mode	37 - G95 Mode	67 - TMode PSS	166 - Jogging
8 - Feedhold	38 - Threading	68 - THead fire open	167 - Override active
9 - STOP	39 - Laser Trigger On	69 - G28 Off	169 - Spindle M3
10 - EStop	40 - Inhibit	70 - Tangential On	170 - Single surface
11 - No A Radius	41 - Ignore M6	71 - 2s Angle Step	171 - External Device
12 - No B Radius	42 - CV On	72 - Jog On	172 - File loaded
13 - No C Radius	43 - CV On M6	73 - AJog On	173 - Scripts running
14 - SoftLimits On	44 - Repeat On	74 - OVmode	174 - Probe trig decel
15 - THC On	45 - Exact Stop	85 - x-Enabled	800 - MPG On
16 - Spindle Decel	46 - G83 Called	86 - y-Enabled	801 - Inch
17 - Spindle Inc	47 - MPG Mode	87 - z-Enabled	802 - mm
18 - Tool In Spindle	48 - Threading	88 - a-Enabled	803 - RUN
19 - No Tool	49 - Laser Trigger On	89 - b-Enabled	804 - STOP
20 - Tool Is Offset	50 - Inhibit	90 - c-Enabled	805 - Planner Paused

21 - Not In G94	51 - Ignore M6	91 - d-Enabled	806 - ToolChange wait
22 - Not In G95	52 - CV On	92 - Tangential Mode	807 - X Homed
23 - Auto Limits Overrd	53 - CV On M6	94 - HotKeys On	808 - Y Homed
24 - Un-Limit On	54 - Repeat On	95 - Units Per Minute	809 - Z Homed
25 - Units Per Rev	55 - Exact Stop	96 - Units Per Rev	810 - A Homed
26 - Units Per Min	56 - Mpx Jog	97 - G95 On	811 - B Homed
27 - 1x Scaled	57 - Mpx Continuous Jog	98 - G96 On	812 - C Homed
28 - 2x Scaled	58 - Optional Stop	99 - G97 On	813 - Dweel in effect
29 - 3x Scaled	59 - Block Delete	100 - Rapid Override	815 - Soft limit on

5. Nhóm tín hiệu Modbus



ModBus IO Selected:

Enter ModBys Address to use: là nơi nhập địa chỉ thanh ghi cần lấy dữ liệu. Nếu mỗi Cfg# trong Function Cfg's -> Setup Serial Modbus Control chỉ lấy 1 thanh ghi từ Slave thì trong cửa sổ này chỉ cần nhập số 0 (tức là luôn lấy thanh ghi đầu tiên của Cgf# và cũng là thanh ghi duy nhất).

Input: là đọc dữ liệu từ các thanh ghi của Slaver

Output: là ghi dữ liệu vào các thanh ghi của Slaver

Bit Only: là đọc/ghi từng Bit rời của thanh ghi

Trong phần Source:

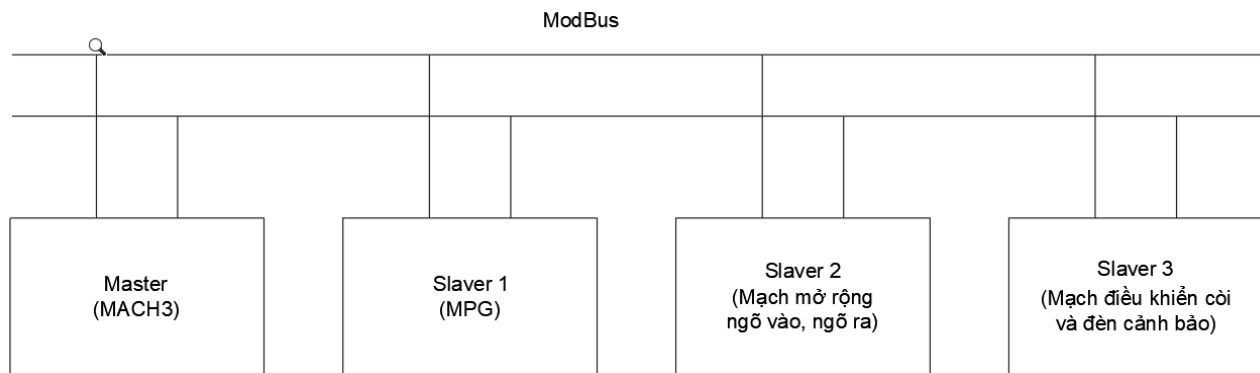
- MacroModbus Emulation: Giả lập ModBus bằng Macro;
- Serial Non-Plugin Enable: Sử dụng ModBus RTU không dùng Plugin;

- Serial Plugin Enabled: Sử dụng Plugin Modbus RTU (thường chúng ta sẽ sử dụng tùy chọn này);

- TCP Modbus: Sử dụng ModBus giao thức TCP qua cáp mạng Lan;

CFG#xxx là sử dụng Config số mấy. Tùy thuộc vào thiết lập cấu hình trong **Function Cfg's** -> **Setup Serial Modbus Control** có những cấu hình Cfg #x nào và đang muốn lấy dữ liệu từ Cfg nào mà sẽ phải điền giá trị tương ứng.

Modbus là giao thức truyền thông giữa các thiết bị dựa trên kiến trúc master-slave. Nó sử dụng các giao diện RS-485, RS-422 và RS-232, cũng như mạng Ethernet TCP/IP (giao thức Modbus TCP) để truyền dữ liệu. Trong mạng Modbus luôn có một thiết bị chủ (Master) sẽ làm mọi nhiệm vụ đọc, ghi dữ liệu vào mạng và kiểm soát mạng, MACH3 chính là Master của mạng Modbus. Các thiết bị còn lại là thiết bị đầy tớ (Slaver), làm nhiệm vụ cung cấp dữ liệu hoặc nhận dữ liệu từ Master yêu cầu. Các thiết bị tớ ví dụ như Bo mạch mở rộng các cổng vào ra cho Mach3, Tay cầm MPG, hệ thống đèn và chuông báo



Hình 2. Ví dụ về một mạng Modbus kết nối các thiết bị ngoại vi với Mach3

Trong một mạng Modbus có thể có tối đa 247 thiết bị tớ (Slaver), mỗi Slaver có một địa chỉ định danh khác nhau, không trùng lặp và thay đổi từ 1 đến 247.

a. Cấu trúc khung dữ liệu truyền thông trong mạng Modbus

Thông điệp Modbus RTU bao gồm địa chỉ của thiết bị SlaveID, mã hàm (function code) và dữ liệu đặc biệt tùy theo mã hàm, cùng với CRC để kiểm tra lỗi

SlaveID	Function code	Special data	CRC
---------	---------------	--------------	-----

Trong đó:

- Trường **SlaveID** (1 byte): Xác định thiết bị slave đích. Giá trị nằm trong khoảng từ 1 đến 247, trong đó 0 được dành cho truyền phát (broadcast) còn các địa chỉ từ 248 đến 255 được dự trữ chưa sử dụng.

- Trường **Function code** (1 byte): là mã chức năng của khung truyền, chỉ định loại hành động mà master yêu cầu, chẳng hạn như đọc hoặc ghi dữ liệu:

Function Code	Tên chức năng	Giải thích	Tài liệu tham khảo
01	Read Coils	Đọc trạng thái nhiều cuộn ngõ ra số (Discrete Output Coils)	Trang 12 của giao thức
02	Read Discrete Inputs	Đọc trạng thái nhiều tiếp điểm đầu vào số (Discrete Input Contacts)	Trang 13 của giao thức
03	Read Output Holding Registers	Đọc nhiều thanh ghi đầu ra (Analog Output Holding Registers)	Trang 15 của giao thức
04	Read Input Registers	Đọc nhiều thanh ghi đầu vào (Analog Input Registers)	Trang 16 của giao thức
05	Write Single Coil	Ghi một cuộn ngõ ra số (Discrete Output Coils)	Trang 17 của giao thức
06	Write Single Holding Register	Ghi giá trị một thanh ghi đầu ra (Analog Output Holding Registers)	Trang 19 của giao thức
15	Write Multiple Coils	Ghi nhiều cuộn ngõ ra số (Discrete Output Coils)	Trang 29 của giao thức
16	Write Multiple Holding Registers	Ghi nhiều thanh ghi đầu ra (Analog Output Holding Registers)	Trang 30 của giao thức

- Trường **Special data** (n byte): chứa thông tin liên quan đến hàm, bao gồm địa chỉ thanh ghi và giá trị. Chi tiết từng lệnh **Function Code**, các byte dữ liệu đi kèm và cách Slaver phản hồi tham khảo chi tiết hơn tại tài liệu giao thức ứng dụng Modbus ([\files\MACH3_MODBUS\Modbus_Application_Protocol_V1_1b.pdf](#)).

- Trường **CRC** (2 bytes): Dùng để kiểm tra lỗi bằng thuật toán kiểm tra dư theo chu kỳ (CRC - Cyclic Redundancy Check).

b. Tổ chức dữ liệu trong thiết bị Slaver

Dữ liệu trong module Slaver được tổ chức lưu trữ các thanh ghi. Thanh ghi được phân thành 4 bảng tùy theo kiểu dữ liệu mà nó chứa. Trong đó Hai bảng là chỉ đọc (read-only) và hai bảng cho phép đọc-ghi (read-write). Mỗi bảng chứa 9999 giá trị.

REGISTER NUMBER (Số của thanh ghi)	REGISTER ADDRESS HEX (Các địa chỉ của thanh ghi)	TYPE (Kiểu thanh ghi)	NAME (Tên thanh ghi)	TYPE (Loại thanh ghi)
1~9999	0000 to 270E	read-write (Đọc và Ghi)	Discrete Output Coils (Đầu ra kiểu tiếp điểm rời rạc từng bit)	DO (Đầu ra kỹ thuật số)
10001~19999	0000 to 270E	read (Chỉ đọc)	Discrete Input Contacts (Đầu vào kiểu điểm đầu rời rạc từng bit)	DI (Đầu vào kỹ thuật số)

30001~39999	0000 to 270E	read (Chỉ đọc)	Analog Input Registers (Đầu vào kiểu tương tự, nhiều bit)	AI (Đầu vào tương tự)
40001~49999	0000 to 270E	read-write (Đọc và Ghi)	Analog Output Holding Registers (Đầu ra kiểu tương tự, nhiều bit)	AO (Đầu ra tương tự)

Giải thích ngắn:

Discrete Output Coils: Các đầu ra điều khiển dạng bật/tắt (ON/OFF), ví dụ điều khiển relay.

Discrete Input Contacts: Đầu vào kỹ thuật số, thường là tín hiệu từ công tắc, cảm biến dạng ON/OFF.

Analog Input Registers: Nhận tín hiệu analog từ cảm biến (ví dụ: nhiệt độ, áp suất).

Analog Output Holding Registers: Gửi tín hiệu analog để điều khiển (ví dụ: điều khiển tốc độ động cơ).

Một thiết bị Slaver có thể có một hoặc nhiều thanh ghi lưu trữ các loại dữ liệu khác nhau, mỗi thanh ghi gồm 2 Byte, mỗi Byte gồm 8 bits. Cấu trúc thanh ghi như sau

Cấu trúc của 1 thanh ghi															
Byte 1								Byte 0							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Mỗi bit dữ liệu có thể chứa thông tin 0 hoặc 1

c. Ví dụ khung truyền dữ liệu truy vấn từ Master và phản hồi từ Slaver

Ví dụ một khung dữ liệu của Master gửi đi (hệ thập lục phân): 11 03 00 6B 00 03 76 87, phân tích chi tiết vào cấu trúc khung như sau:

SlaveID	Function code	Special data	CRC
Địa chỉ của Slave	Mã chức năng	Dữ liệu	Checksum để kiểm tra lỗi truyền dữ liệu
11	03	00 6B 00 03	76 87

Ý nghĩa của khung dữ liệu trên như sau:

SlaveID = 11 nghĩa là địa chỉ Slave cần liên lạc là 11

Function code = 03 nghĩa là Read Output Holding Registers (Đọc thanh ghi đầu ra Analog Output Holding Registers)

Special data đi kèm là 00 6B 00 03

Tra tài liệu giao thức mã Function code 03 ([Trang 15 của giao thức](#)) ta thấy theo sau mã code 03 sẽ yêu cầu 2 byte để xác định địa chỉ bắt đầu của thanh ghi cần đọc, tiếp theo là 2 byte để xác định số lượng thanh ghi cần đọc. Như vậy:

Starting Address = 006B (hệ thập lục phân, HEX) = 107 (Hệ thập phân, DEC);

Quantity of Registers = 0003 (HEX) = 3 (DEC)

CRC = 8776 (Hex) là mã Checksum được tính toán để Slave kiểm tra xem dữ liệu nhận được có CRC tính toán bằng với giá trị CRC này không. Nếu bằng là dữ liệu được bảo toàn, không bị lỗi do đường truyền.

Như vậy, khung trên có nghĩa là cần đọc 3 thanh ghi, bắt đầu từ địa chỉ 107 của thiết bị Slaver có địa chỉ 11. Do mỗi thanh ghi luôn có kích thước là 2 byte, như vậy cần đọc 6 bytes dữ liệu.

6.3 03 (0x03) Read Holding Registers

This function code is used to read the contents of a contiguous block of holding registers in a remote device. The Request PDU specifies the starting register address and the number of registers. In the PDU Registers are addressed starting at zero. Therefore registers numbered 1-16 are addressed as 0-15.

The register data in the response message are packed as two bytes per register, with the binary contents right justified within each byte. For each register, the first byte contains the high order bits and the second contains the low order bits.

Request

Function code	1 Byte	0x03
Starting Address	2 Bytes	0x0000 to 0xFFFF
Quantity of Registers	2 Bytes	1 to 125 (0x7D)

Response

Function code	1 Byte	0x03
Byte count	1 Byte	2 x N*
Register value	N* x 2 Bytes	

*N = Quantity of Registers

Hình ảnh trích xuất trang 15 của tài liệu giao thức ứng dụng Modbus về lệnh 03

Khi nhận được truy vấn bên trên, thiết bị Slaver có địa chỉ 11 phản hồi lại như sau:

Ví dụ: 11 03 06 AE 41 56 52 43 40 49 AD

Từ tài liệu giao thức phản hồi của lệnh 03 ta có

SlaveID	Function code	Special data						CRC
		Số byte dữ liệu	Thanh ghi thứ 107	Thanh ghi thứ 108	Thanh ghi thứ 109			
11	03	06	AE 41	56 52	43 40			49 AD

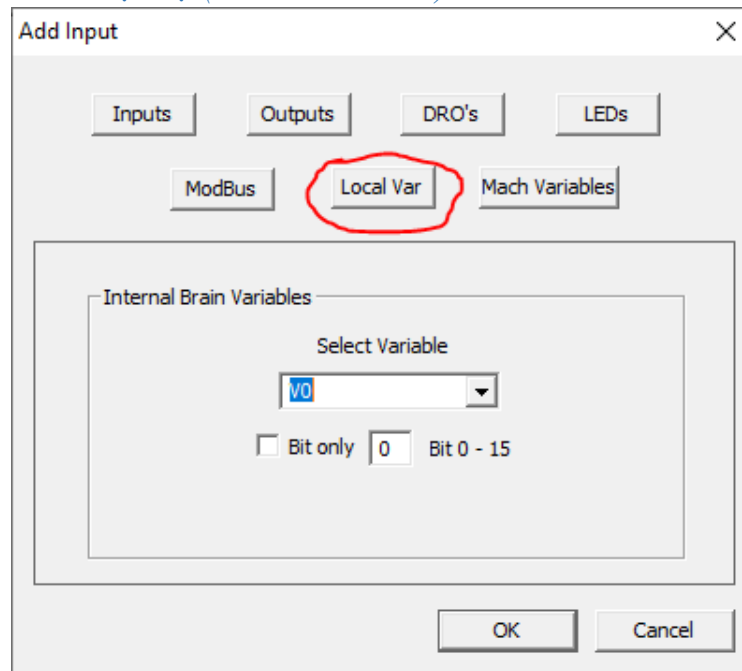
Như vậy ta có dữ liệu đọc về của ba thanh ghi từ địa chỉ 107 là

Registry_Value[107] = 0xAE41 (Hex) = 44609 (DEC)

Registry_Value[108] = 0x5652 (Hex) = 22098 (DEC)

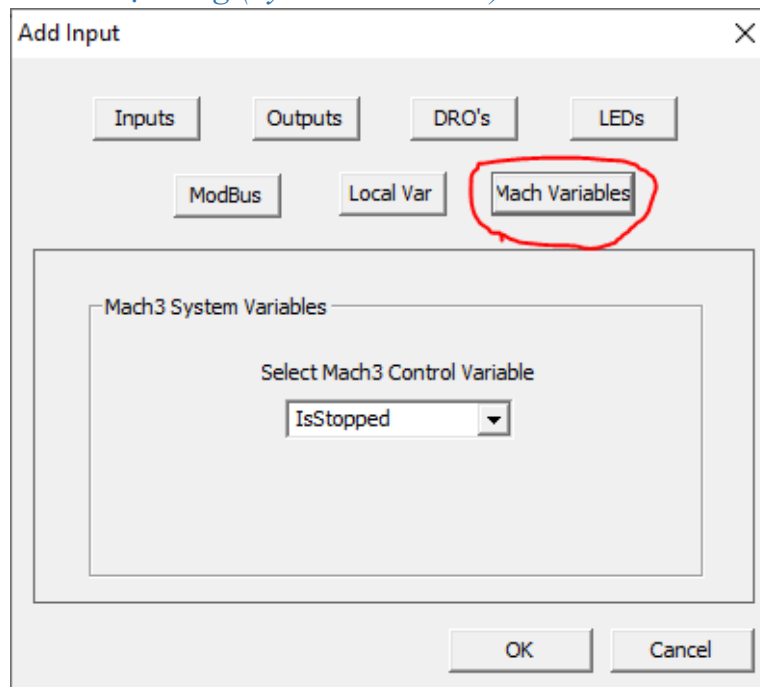
Registry_Value[109] = 0x4340 (Hex) = 17216 (DEC)

6. Nhóm tín hiệu các biến cục bộ (Local Variables)



Các biến cục bộ được dùng để lưu và sử dụng dữ liệu trong khi Brain chạy, lập trình brain có thể sử dụng các biến này làm trung gian xử lý dữ liệu.

7. Nhóm tín hiệu các biến hệ thống (System Variables)



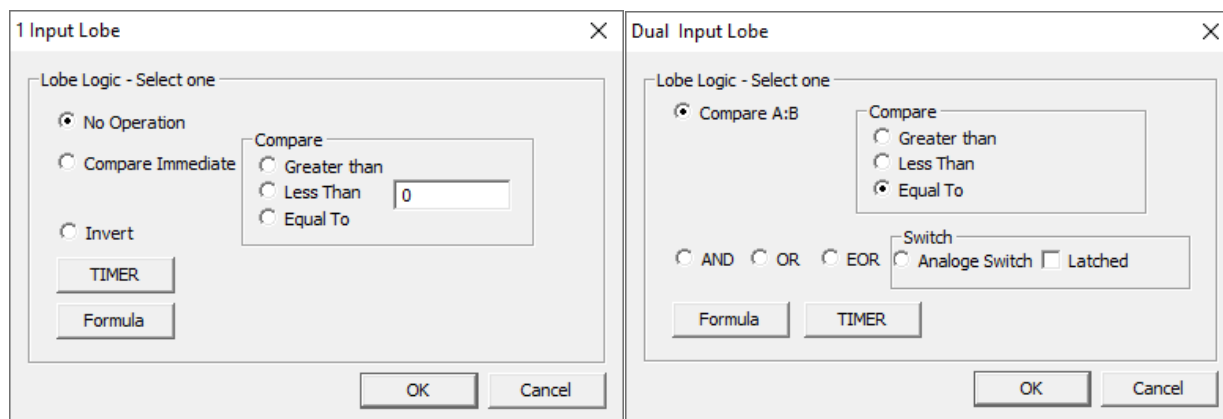
Biến hệ thống là các biến điều khiển của chương trình mach3.

Bảng 6. Nhóm tín hiệu các biến hệ thống (System Variables)

TT	Biến trạng thái	Mô tả
1	- IsStopped	☐ Ý nghĩa: Trả về true nếu hệ thống đang ở trạng thái dừng hoàn toàn (tức là không đang chạy G-code, không di chuyển). ☐ Ứng dụng: Kiểm tra xem có thể bắt đầu lệnh mới hay không.
2	- IsMoving	☐ Ý nghĩa: Trả về true nếu máy đang di chuyển , bao gồm khi thực hiện các lệnh G-code hoặc điều khiển bằng tay. ☐ Ứng dụng: Có thể dùng để chờ cho đến khi di chuyển kết thúc (ví dụ: While IsMoving).
3	- TWaitState	☐ Ý nghĩa: Trạng thái chờ của trục quay (spindle) hoặc M-code . ☐ Chi tiết: Biến này thường phản ánh trạng thái tạm dừng do lệnh chờ M3/M5 hoặc chờ tín hiệu từ thiết bị ngoài. ☐ Giá trị: thường là số nguyên đại diện cho từng trạng thái chờ khác nhau.
4	- THC On	☐ Ý nghĩa: Trạng thái Torch Height Control (THC) – hệ thống điều chỉnh chiều cao đầu cắt plasma theo thời gian thực. ☐ Giá trị: true nếu THC đang bật, false nếu tắt. ☐ Ứng dụng: Dùng trong các ứng dụng plasma để kiểm tra xem THC có đang hoạt động hay không.
5	- CV On	☐ Ý nghĩa: Trạng thái Constant Velocity (CV) – cho biết chế độ chạy G-code đang là CV (di chuyển mượt, không dừng giữa các đoạn). ☐ Giá trị: true nếu đang bật chế độ CV, false nếu đang ở chế độ Exact Stop. ☐ Ứng dụng: Kiểm tra xem đường chạy có được làm mượt hay không.
6	- MPG TickCount	☐ Ý nghĩa: Tổng số xung MPG nhận được trên một đơn vị thời gian. ☐ Ứng dụng: Theo dõi tổng số bước quay đã được xử lý kể từ lần đọc trước, dùng để xác định mức độ dịch chuyển.
7	- MPG1 Count	☐ Ý nghĩa: Bộ đếm xung riêng của MPG1 . ☐ Chi tiết: Dùng để theo dõi số lượng xung (bước quay) được phát hiện từ đầu vào MPG1. ☐ Ứng dụng: Có thể dùng để xử lý riêng từng tay quay nếu sử dụng nhiều MPG.
8	- MPG2 Count	Tương tự MPG1
9	- MPG3 Count	Tương tự MPG1

I.1.2. Add thêm một xử lý logic vào dòng xử lý đã có

Giao diện màn hình chức năng xử lý tín hiệu logic của Brain

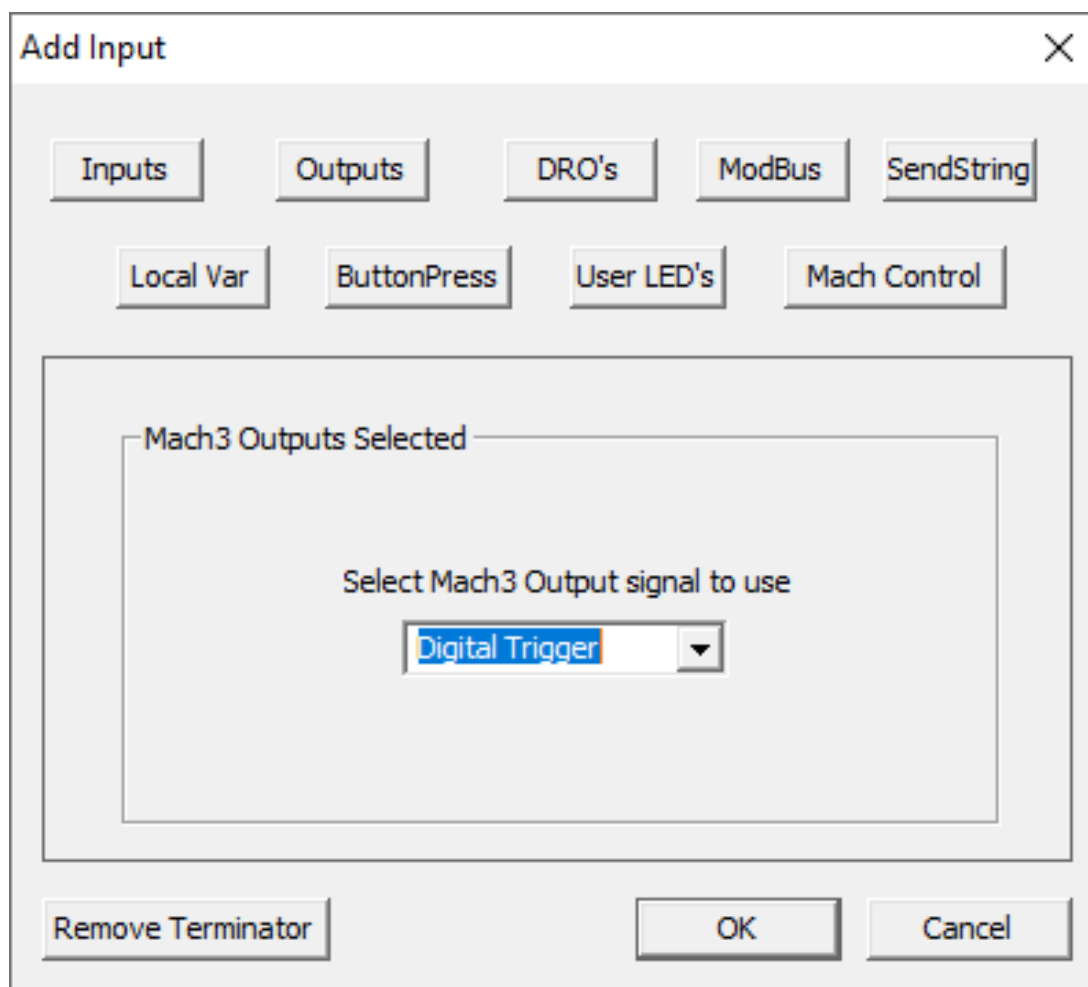


I.2. Chức năng Delete

Chức năng Delete cho phép xóa các Lobe đã tạo ra.

I.3. Chức năng Terminate Lobe

Khi bấm **Terminate Lobe** kết thúc một dòng, cửa sổ hình 2 Add Input hiện lên. Dữ liệu có thể chọn xem bảng 8.



Bảng 8. Tổng hợp các dữ liệu đầu vào/ra dùng cho Brain

TT	Nhóm tín hiệu	Mô tả chức năng	Danh sách chi tiết
1	INPUTS	Thiết lập giá trị cho các đầu vào	Bảng 2
2	OUTPUTS	Thiết lập giá trị cho các đầu ra	Bảng 3
3	DRO's	Thiết lập giá trị cho các DRO	Bảng 4
4	Modbus	Thực hiện đọc hoặc ghi dữ liệu MODBUS	
5	SendString	Gửi dữ liệu dạng chuỗi tối đa 80 ký tự ra MODBUS	
6	Local Var	Thiết lập giá trị cho các biến cục bộ	Có 100 biến từ V0 ~ V99
6	ButtonPress	Thực hiện giả lập bấm các nút	Bảng 9
5	User LED's	Thiết lập giá trị cho các đèn LED của người dùng	Bảng 10
6	Mach Control	Thiết lập giá trị cho các biến hệ thống của Mach3	Bảng 6

I.3.1. INPUTS

Xem danh sách chi tiết tại bảng 2

I.3.2. OUTPUTS

Xem danh sách chi tiết tại bảng 3

I.3.3. DRO's

Xem danh sách chi tiết tại bảng 4

I.3.4. Modbus

Add Input

Inputs Outputs DRO's **ModBus** SendString

Local Var ButtonPress User LED's Mach Control

ModBus IO Selected

Enter ModBus Address to use 0

☒ Input ☐ Output ☐ Bit only 0 Bit 0 - 15

Source

☒ MacroModbus Emulation

☐ Serial Non-Plugin Enabled

☐ Serial Plugin Enabled CFG# 0 0-127

☐ TCP ModBus

Remove Terminator OK Cancel

I.3.4. SendString

Cho phép gửi đến Slave một chuỗi văn bản có độ dài tối đa tới 80 ký tự

Add Input

Inputs Outputs DRO's ModBus **SendString**

Local Var ButtonPress User LED's Mach Control

Send String Message to PLC

String to send 80 chars max

Register Address 0

☒ 0 - 128 serial

☐ 0 - 1023 TCP

☐ 0 - 127 SP Cfg # 0

Remove Terminator OK Cancel

I.3.4. Local Var

Các biến cục bộ được sử dụng trong nội bộ chương trình Brain có 100 biến từ V0 đến V99. Khi lập trình có thể sử dụng các biến này lưu trữ dữ liệu

I.3.4. ButtonPress

Bảng 9. Danh sách các nút bấm có thể thực hiện bởi Terminate Lobe

1000 - CycleStart	148 - Correction On/Off	219 - Z Inhibit off	293- MPG 1 axis a
1001 - FeedHold	149 - AutoLimit OverRide Toggle	221 - AntiDive toggle	294- MPG 1 axis b
1002 - Rewind File	151 - SingleActive toggle	222 - AntiDive On	295- MPG 1 axis c
1003 - Stop File	153 - UNITEX/UNTMIN toggle	223 - AntiDive Off	296 - Soew Dev. Calibrate
1004 - SingleStep	156 - Set Next Line	224 - Flood On	297- Stop Dwell
1005 - ResumeFile	157 - Jog Follow Toggle	225 - Flood Off	299- Bypass Feedrate on/off
1007 - Zero All Axis	158 - Joystick On	226 - Mist On	300- Front Rear post toggle
1008 - Zero X	159 - Joystick Off	227 - Mist Off	302- MPG 1 MODE increment
1009 - Zero Y	160 - Regen Toolpath	228 - Load Teach File	303- MPG 1 MODErO
1010 - Zero Z	162 - G90/G91 toggle	229 - ConfigToolPath	304- MPG 1 MODE: 1
1011 - Zero A	163 - Dnc SpinSpeed	230 - Select Wizard	305- MPG1 MODE: 2
1012 - Zero B	164 - Dec SpinSpeed	231 - Load Default Screens	306- MPG 1 MODE: 3
1013 - Zero C	165 - Laser Trigger On/Off	232 - Tab screen on/off	307- Jog X - right
1014 - Reset FRO	167 - ZInhibit on/off	233 - Mist Off	308- JogX - Left
1015 - Simulate	168 - Tool Ignore on/off	240 - All homes deref.	309- Jog Y - right
1016 - Run At...	169 - CloseFile	241 - Tangential On/Off	310- Jog Y - Left
1017 - Goto Zero's	170 - Load Last File	242 - Copy position to G59P253	311- Jog 2 - right
1018 - Display Machine Coord	171 - Jog incr increment	243 - Move to G53 from G59P25	312- Jog z - Left
1019 - Verify	172 - ErrorClear	246 - Force Axis to read as home	313- Jog A - right
1021 - RESET	173 - Spindle Off	247- cv Toggle A	314- Jog A - Left
1022 - Home X	174 - Jog type conf/mpg1	248- GV Off	315- Calibrate MPG
1023 - Home Y	175 - MPG1 Axis inc	249- cv On	316 - SaveTools
1024 - Home A	176 - Block Delete Toggle	25C- Toggle X axis onhne/offline	317- Save Fixtures
1025 - Home B	177 - Optional Stop toggle	251- Toggle Y axis onhne/offline	318- Zoomstate on/off
1026 - Home C	178 - Offline Toggle	252- Toggle 2 axis onhne/offline	319- PanState O n/off
1027 - Joystick Toggle	179 - Machine/Work Coord display	253- Toggle A axis online/offline	320- Select Wizard
1028 - Softlimits Toggle	180 - Display Work Coord - unload	254- Toggle B axis online/offline	321- Call Newfnagled Wizards

1030 - Execute Gcode	181 - Display Work Coord - locked	255- Toggle c axis online/offline	322- Stop all actions.
1032 - Execute Button Script	182 - Use Spindle Feedback toggle	257- Online	323- Tab control in/out
1034 - Jog Incr ++	183 - Reset Spindle Over	258- Offline	324- TurnToudi X
1040 - Jog Incr --	184 - Lathe Home	259- MPG 1 X control	325- TorchVolts
1041 - Init Interp	185 - MPG1 Jog X	26C-MPG1Y control	326- TurnTouch z
1042 - Jog On/Off	186 - MPG1 Jog Y	261- MPG 12 control	327- MPG Jog On
1044 - GotoSafeZ	187 - MPG1 Jog Z	262- MPG 1 A control	329 - MPG2 axis toggle
105 - Refill	188 - MPG1 Jog A	263- MPG 1 B control	329- Jog B right
106 - English/Metric Toggle	189 - MPG1 Jog B	264- MPG 1 c control	330- Jog 6 left
107 - Machine Coordinate Display	190 - MPG1 Jog C	265 -Select MPG Ino 1	331- Jog c right
108 - FRO UP	191 - Select Jog Incr 1	266- Select MPG Incr 2	332- Jog c left
109 - FRO DN	192 - Select Jog Incr 2	267- Select MPG Ino 3	333- ToGo display on/off
110 - Spindle On/Off	193 - Select Jog Incr 3	268- Select MPG Incr 4	334- Jog Off X
111 - Jog% UP	194 - Select Jog Incr 4	269- Select MPG Ino 5	335- Jog Off Y
112 - Jog% DN	195 - Select Jog Incr 5	270- Select MPG Ino 6	336- Jog Off Z
113 - Flood Toggle	196 - Select Jog Incr 6	271- Select MPG Incr 7	337- Jog Off A
114 - Mist Toggle	197 - Select Jog Incr 7	272- Select MPG Incr 8	338- Jog Off B
115 - Edit File	198 - Select Jog Incr 8	273- Select MPG Ino 9	339- Jog Off C
116 - Zero Radius A	199 - Select Jog Incr 9	274- Select MPG Ino 10	340- SetSoftMin
117 - Zero Radius B	200 - Select Jog Incr 10	275- Jog cont.	341- SetSoftMax
118 - Zero Radius C	201 - JoyThrottle for Jog %	276- Jog incremental	342- Restore Soft limits
119 - Softlimits on/off	202 - JoyThrottle for FRO	277- Feedrate+ +	343- ABS/INC DRO readout
120 - Tool Touch	203 - Jog Mode Cont.	278- Feedrate -	347- Cv DiSt on/aff
121 - Tool Table Call	204 - Jog Mode Incr.	279- Run in Reverse	348- MenuOff
122 - Fixture Table Call	205 - Joy Mode Incr.	280- Autoload Wizard	349- MenuOn
123 - THC Toggle	206 - JoyStick On	281- MPG2 axis increment	350- Spin Inoement
124 - Calibrate THC	207 - Joystick Off	282- Taper mode on/off	351- Spin Decrement
125 - DNC ->ENC X	208 - Tool Length to zero	283- DualMPG O n/off.	356- Ignore M3 toggle
126 - ENC ->DRO X	209 - Tool X to zero	284- Shuttle Mode on/off	357- All Slaves O ff
127 - DRC ->ENC Y	210 - Move to Home	285- Return to..	358- All Slaves On
128 - ENC ->DRO Y	211 - Move to Home	286- Remember Pos..	360- Jog Probe On/Off
129 - DRC ->ENC Z	212 - Lathe1 to Home	287- RapidOverride toggle	361- Jog Probe o f f
130 - ENC ->DRO Z	213 - Lathe2 to Home	287- RapidOverride toggle	362- Jog Probe On
131 - MillaToggle	214 - Load recent file	288- stop G04 Dwell A	363- A axis toggle display
132 - ToolPath On/Off	215 - Display History	289- Set Formulas	364- ISO Enable
136 - G43 Toggle	216 - Load Gcode File	290- MPG 1 axis X	365- ISO Disable
137 - GoTo Home Loc.	217 - Flip table tool to top	291- MPG 1 axis y	
147 - Throttle Function Toggle	218 - Z Inhibit on	292- MPG 1 axis z	

I.3.4. User LED's

Bảng 9. Danh sách các đèn LED có thể bật/tắt bởi Terminate Lobe

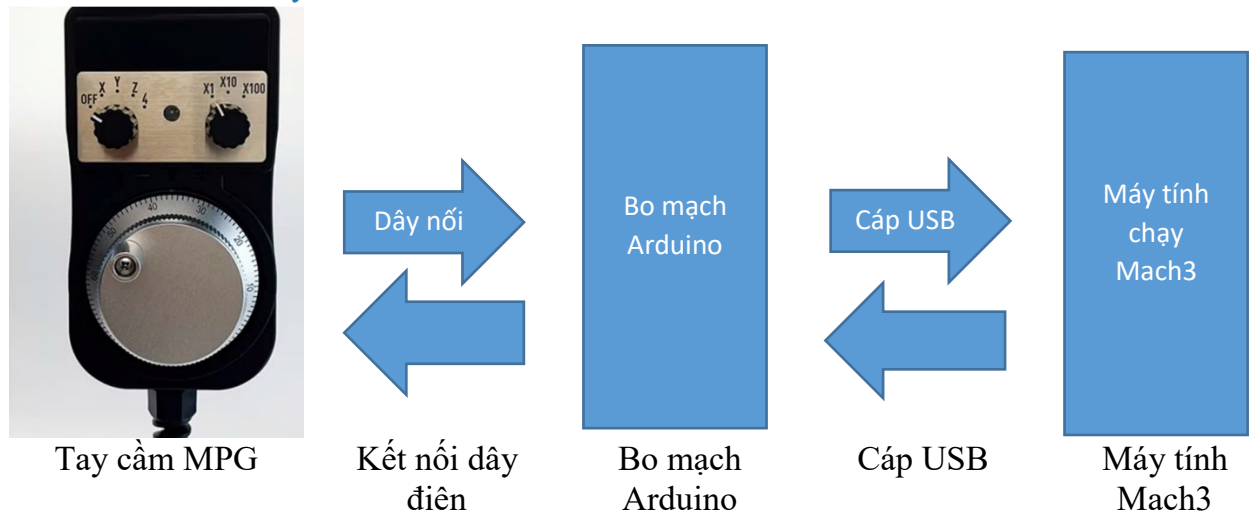
999- IsMoving	42 - Y Scaled	75- G92 off	165- Spindle ccw on
11- Spindle ON	43- 2 Scaled	8 1 -Tangentai On	166- Normal On
12- Mist On	44- A Scaled	8 2 -Single Step	169- Ignore M3
13- Flood On	45- B Scaled	83- Jog On	170- Slaving in effiec
14- Jog Cent.	46 - c Scaled	84- CV ON	171- External Device
15- Jog Ino.	48- G9CMode	85-SlowMode	172- Fife 'oadec
16 - Machine Coord	49- G91Mode	86- X enabled	173- Scripts running
17- Feed Override on	50- Threading	87- Y enabled	174- Probe jog decel in effect
18- Simulating	51- Laser Trigger On	88- 2 enabled	800-ESTOP
19- ESTOP	52- 2 Inhibit	89- A enabled	801- Inch
20- No A Radius	53- Ignore M6	90- 6 enabled	802- mm's
21- No B Radius	54- cv On	9 1-c enabled	804- RUN
22- No c Radius	55- Repeat On	9 2 -Diameter Mode	803- STOP
23- SoftLimits On	56 -Exact Stop	94- HotKeys On	805- Planner Paused
24- IN C On	57- MPG Jog	95- Units Per Minute	806- Toolchange wa
25- Spindle Accel	58- Cont shift Jog	96- Units Per Rev.	807- X Homed
26- Spindle Decel	65- Optional stop	97- Reverse running	808- Y Homed
27- Toolpath On	66- Block Delete	107- G96 On	809- 7 Homed
28- Tool Is Offset	67- Offline	108- G68 on	810- A Homed
29- Not in GS4	68- Spindle Feedbad	109- Rapid Ovrtn	811- B Homed
30- Throttle O ff	69- Thread wait Trig	110- Using Formulas	812- c Homed
33- Auto Limits Ovr	70- NoPlungeTHC	111- Partial line holdi	813- Dwell in effect
34- No Limits On	71- Spindle Stable	112- Reverse runnin	815- Softiimits On
39- Units Per Rev	72- IJMode ABS	162- Maaro running	
40- Units Per Min	73- IJ mode INC	163- ToGo mode on	
41- X Scaled	74- Tech File open	164- Spindle cw on	

I.3.4. Mach Control

Mach control bao gồm các biến hệ thống có thể được điều khiển bởi Terminate Lobe, biến hệ thống đã được giới thiệu trong bảng 6 mục I.1.1

III. Một số dự án thực tế với Brain

III.1. Dự án tạo tay cầm ModBUS MPG



Sơ đồ kết nối hệ thống MPG Modbus sử dụng Arduino

Ví dụ tay cầm MPG trong dự án [MPG Modbus sử dụng Arduino](#) và dự án [MPG SENTROL AC09-RX](#) kèm theo tài liệu này đã chọn địa chỉ Slaver là 1, Có 3 thanh ghi **Analog Output Holding Registers** chứa dữ liệu với địa chỉ như sau:

Dữ liệu trong các thanh ghi **Analog Output Holding Registers** của tay cầm MPG

Tên thanh ghi	Địa chỉ thanh ghi	Dữ liệu chứa	Ý nghĩa
4001	0	MPG_Count	Bộ đếm của núm xoay Encoder
4002	1	MPG_TickCount	Tốc độ quay của núm xoay
4003	2	MPG_Switch	Giá trị bật/tắt của các công tắc chọn trục X, Y Z, A và công tắc khuếch đại x1, x10, x100

Trong đó:

MPG_Count: là bộ đếm xung của núm xoay Encoder, kích thước 2 byte, kiểu dữ liệu interger, giá trị thay đổi từ -32768 đến 32767 tương ứng vị trí của trục. Khi quay thuận mỗi xung tăng bộ đếm lên 1 đơn vị. Khi quay ngược, mỗi xung sẽ giảm bộ đếm 1 đơn vị.

Thanh ghi MPG_Switch	
Byte 1 (Byte cao)	Byte 0 (Byte thấp)

MPG_TickCount: là giá trị tốc độ quay của núm xoay, tính theo số xung tạo ra trong 1 giây, kích thước 2 byte, kiểu dữ liệu interger, giá trị thay đổi từ -32768 đến 32767 tương ứng tốc độ và chiều quay của núm xoay.

Thanh ghi MPG_TickCount	
Byte 1 (Byte cao)	Byte 0 (Byte thấp)

MPG_Switch: Là trạng thái bật/tắt hiện tại của các công tắc chọn trục X Y Z A và công tắc khuếch đại x1 x10 x100. Kích thước 2 byte (16 bits), kiểu dữ liệu unsigned integer. Khi núm xoay về vị trí nào thì bit tương ứng của vị trí đó đặt thành 1, các bit tương ứng của các vị trí còn lại đặt thành 0.

Thanh ghi MPG_Switch															
Byte 1 (Byte cao)								Byte 0 (Byte thấp)							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	-	-	-	-	-	-	-	X100	X10	X1	A	Z	Y	X

Ví dụ: khi công tắc vặn về vị trí x100 thì bit 6 của Byte 0 trong thanh ghi MPG_Switch sẽ được đặt thành 1, khi công tắc rời đi nó sẽ được đặt thành 0 và tương tự cho các vị trí công tắc khác.

Trình tự để tạo một chương trình Brain đọc được tay cầm MPG modbus này như sau:

Bước 1: Bật chế độ cho phép sử dụng ModBus trên Mach3

Bước 2: Thiết lập cấu hình Serial Modbus Control để đọc dữ liệu từ ModBus vào Mach3

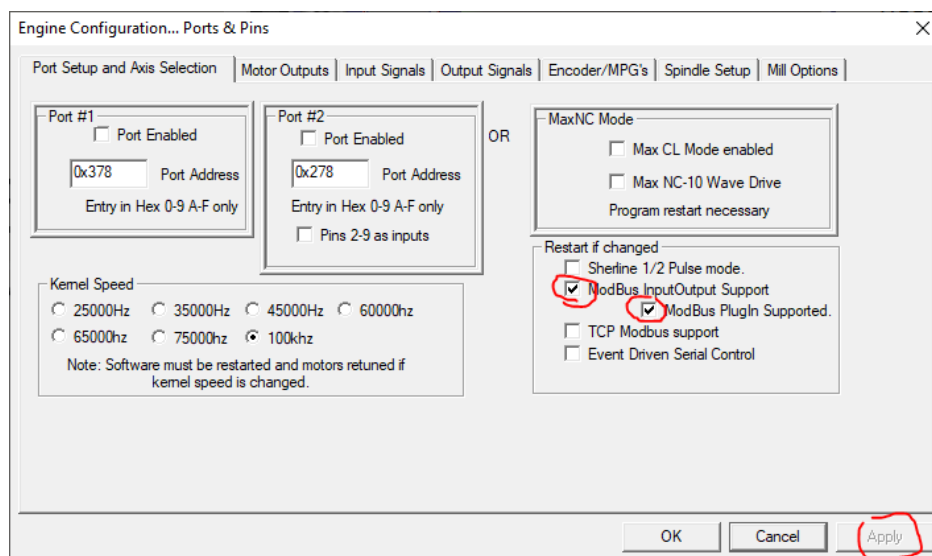
Bước 3: Thiết lập Brain xử lý dữ liệu nhận được từ ModBus ở bước 2 để điều khiển Mach3

Bước 4: Kích hoạt cho Brain tạo ở bước 3 chạy trong Mach3

Dưới đây sẽ đi vào chi tiết cách làm từng bước:

III.1.1. Bật chế độ cho phép sử dụng ModBus trên Mach3

- Vào Config -> Ports and Pins -> thẻ Port Setup and Axis Selection-> tích chọn như hình dưới-> bấm Apply.

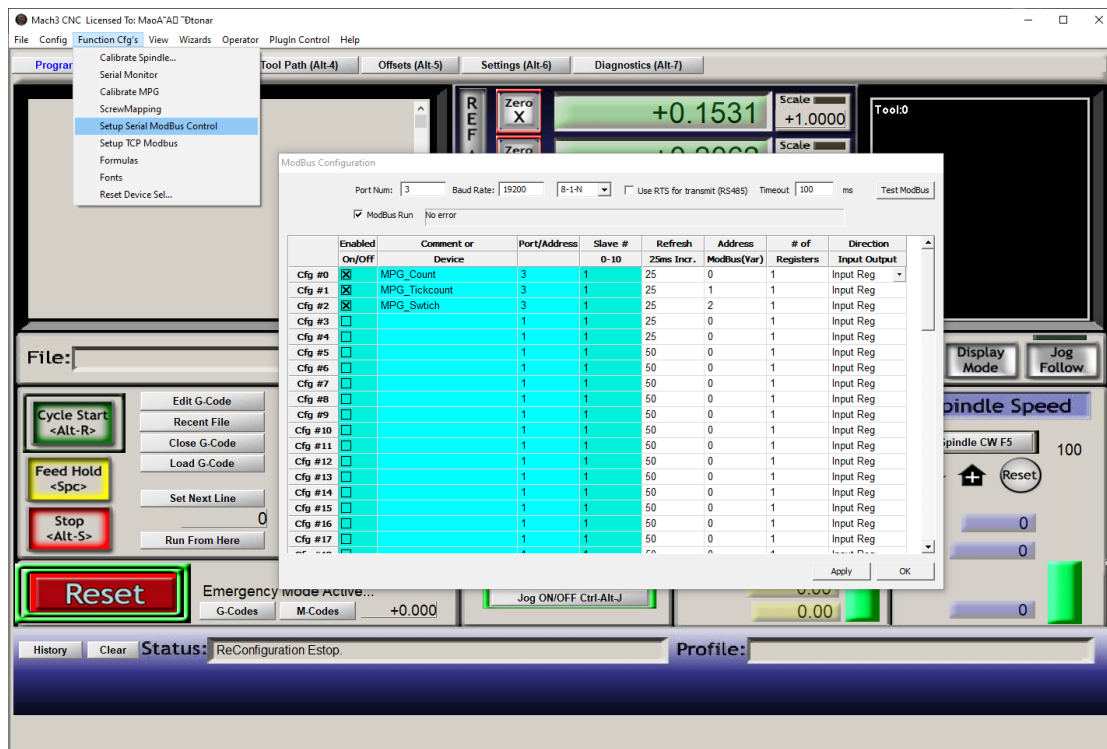


- Tắt phần mềm Mach3 sau đó mở lại để phần mềm nhận cài đặt cho phép sử dụng ModBus.

III.1.2. Thiết lập cấu hình Serial Modbus Control

Cắm thiết bị MPG modbus vào cổng USB của máy tính và kiểm tra xem máy tính đã nhận thiết bị vào cổng COM mấy, ví dụ COM3.

Vào Function Cfg's -> Setup Serial Modbus Control.



Ý nghĩa của các tham số trên màn hình **ModBus Configuration** như sau:

Port Num: Số cổng COM của máy tính đã kết nối với Modbus Arduino (ví dụ này đang là COM3);

Baurate: Là tốc độ truyền dữ liệu RS232 của cổng COM, trong mã nguồn Arduino đã đặt là 19200.

8-1-N: là chuẩn dữ liệu UART sử dụng, chuẩn này đã được thiết lập trong mã nguồn Arduino

ModBus Run: Nếu tích chọn là cho phép giao tiếp với thiết bị, hủy tích là ngắt kết nối.

Tên cấu hình: Các Cfg #0 Cfg #63 là các cấu hình có thể sử dụng để thiết lập, trong ví dụ này sẽ sử dụng 3 vị trí cấu hình là Cfg #0, Cfg #1 và Cfg #2 (Cfg #0 cho giá trị của MPG_Count, Cfg #1 cho giá trị MPG_TickCount, Cfg #2 cho giá trị MPG_Switch).

Enabled On/Off: Cho phép hoặc không cho phép Cfg tương ứng được chạy.

Comment or Device: Đặt tên bất kỳ cho dễ hiểu, ví dụ này đặt

Port/Address: Địa chỉ cổng COM máy tính kết nối đến thiết bị Arduino USB MPG (ví dụ này đang là COM3)

Slave #: Địa chỉ Slave, mã nguồn Arduino trong ví dụ này đã chọn địa chỉ là 1 cho MPG Slave

Refresh: Tần số cập nhật dữ liệu tính theo mili giây, chọn 25ms.

Address ModBus (Var): Địa chỉ của thanh ghi cần đọc trong Slaver

of Registry: Số lượng thanh ghi cần đọc ra, do chúng ta tạo 3 cấu hình Cfg cho 3 thanh ghi nên chỉ cần đọc ra 1 thanh ghi cho mỗi cấu hình.

Direction Input Output: Chọn kiểu đọc dữ liệu là Input Registry

Tạo 3 cấu hình lấy dữ liệu tay cầm MPG qua Modbus với các thông số như trong bảng sau đây:

Bảng tổng hợp cấu hình Serial ModBus MPG

Cấu hình	Enabled On/Off	Comment or Device	Port/Address	Slave #	Refresh	Address ModBus (Var)	# of Registry	Direction Input Output
Cfg #0	x (bật)	MPG_Count	3	1	25	0	1	Input Registry
Cfg #1	x (bật)	MPG_TickCount	3	1	25	1	1	Input Registry
Cfg #2	x (bật)	MPG_Switch	3	1	25	2	1	Input Registry

ModBus Configuration

Port Num: 3 Baud Rate: 19200 8-1-N ☐ Use RTS for transmit (RS485) Timeout: 100 ms Test ModBus

☒ ModBus Run No error

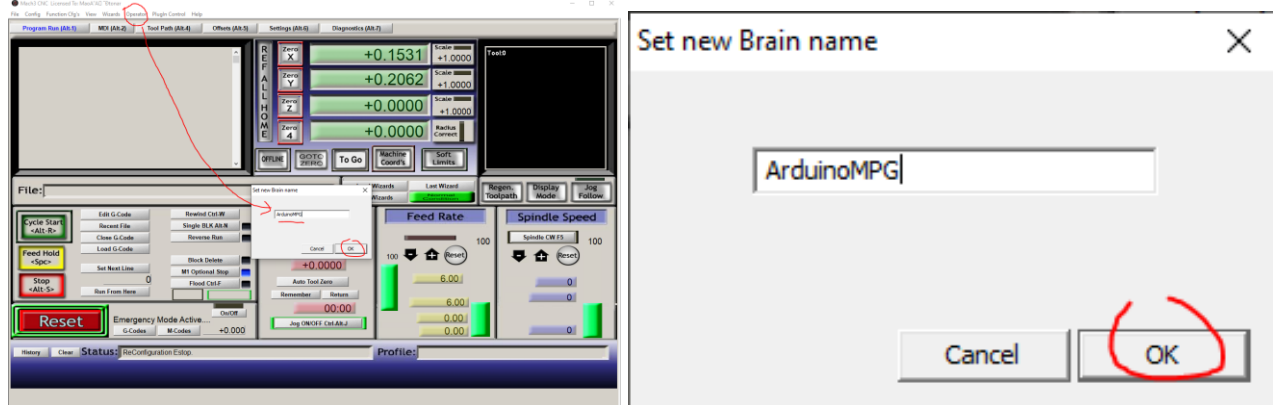
	Enabled On/Off	Comment or Device	Port/Address	Slave #	Refresh	Address	# of	Direction
				0-10	25ms Incr.	ModBus(Var)	Registers	Input Output
Cfg #0	☒	MPG_Count	3	1	25	0	1	Input Reg
Cfg #1	☒	MPG_Tickcount	3	1	25	1	1	Input Reg
Cfg #2	☒	MPG_Switch	3	1	25	2	1	Input Reg
Cfg #3	☐		1	1	25	0	1	Input Reg
Cfg #4	☐		1	1	25	0	1	Input Reg
Cfg #5	☐		1	1	50	0	1	Input Reg
Cfg #6	☐		1	1	50	0	1	Input Reg
Cfg #7	☐		1	1	50	0	1	Input Reg
Cfg #8	☐		1	1	50	0	1	Input Reg
Cfg #9	☐		1	1	50	0	1	Input Reg
Cfg #10	☐		1	1	50	0	1	Input Reg
Cfg #11	☐		1	1	50	0	1	Input Reg
Cfg #12	☐		1	1	50	0	1	Input Reg
Cfg #13	☐		1	1	50	0	1	Input Reg
Cfg #14	☐		1	1	50	0	1	Input Reg
Cfg #15	☐		1	1	50	0	1	Input Reg
Cfg #16	☐		1	1	50	0	1	Input Reg
Cfg #17	☐		1	1	50	0	1	Input Reg

Apply OK

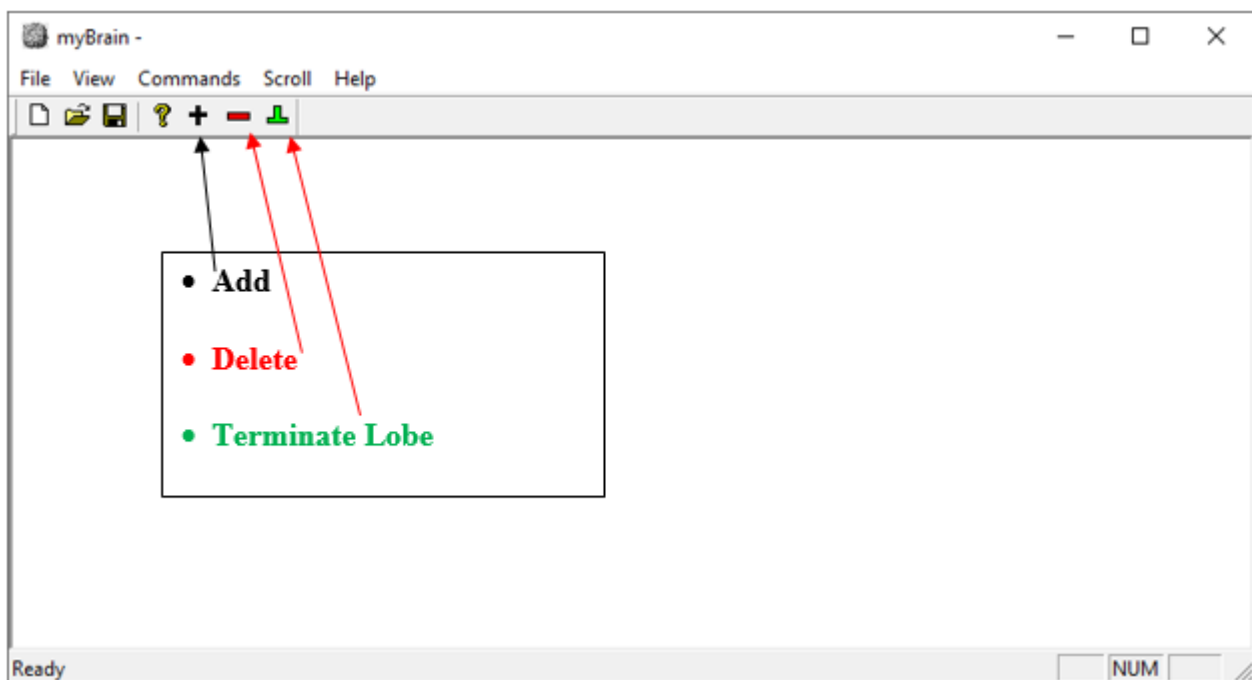
Kể từ thời điểm này, dữ liệu từ MPG ModBus đã được đẩy vào các ngăn chứa của Cfg#0, Cfg#1, Cfg#2.

III.1.3. Thiết lập Brain xử lý dữ liệu nhận được từ ModBus để điều khiển Mach3

Brain có nhiệm vụ đọc dữ liệu đã lấy về từ ModBus để điều khiển các chức năng của Mach3. Trước tiên cần tạo mới một Brain bằng cách vào Operator -> Brain Editor...



Đặt một tên cho Brain, ví dụ **ArduinoMPG** sau đó bấm OK, màn hình biên soạn Brain hiện ra như sau

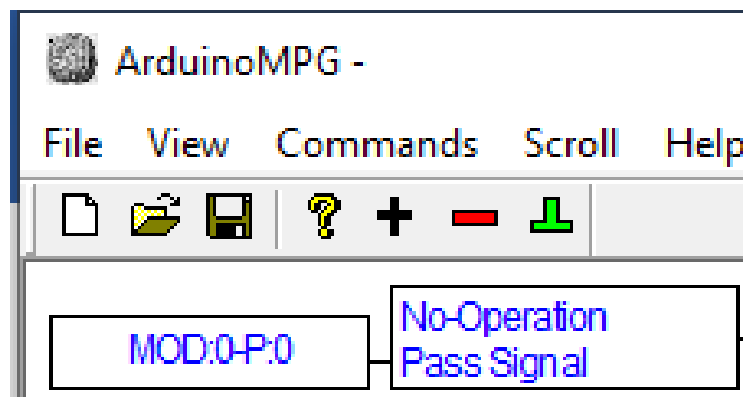
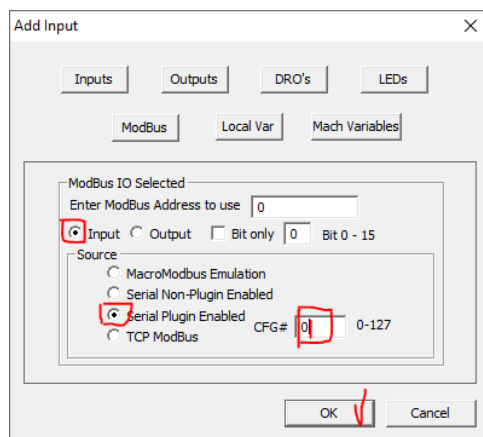


Tiến hành tạo các đầu vào cho các dữ liệu sau

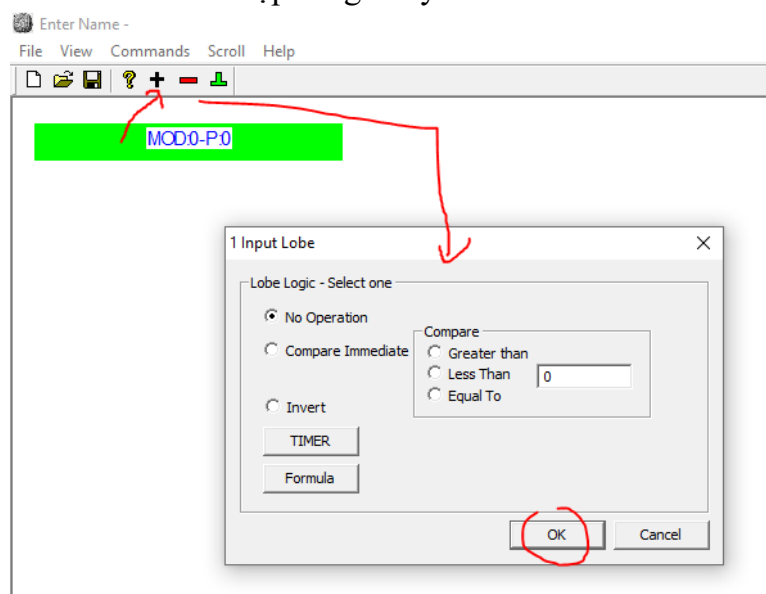
Các đầu vào	Nơi lấy dữ liệu	Chức năng
MPG_Count	Cfg #0 -> địa chỉ 0	Bộ đếm xung của núm xoay
MPG_Tickcount	Cfg #1 -> địa chỉ 0	Tốc độ xoay xung/s
MPG_X	Cfg #2 -> địa chỉ 0 -> bit 0	Chọn trục X
MPG_Y	Cfg #2 -> địa chỉ 0 -> bit 1	Chọn trục Y
MPG_Z	Cfg #2 -> địa chỉ 0 -> bit 2	Chọn trục Z
MPG_A	Cfg #2 -> địa chỉ 0 -> bit 3	Chọn trục A
MPG_x1	Cfg #2 -> địa chỉ 0 -> bit 4	X1
MPG_x10	Cfg #2 -> địa chỉ 0 -> bit 5	X10
MPG_x100	Cfg #2 -> địa chỉ 0 -> bit 6	X100

Thực hiện bằng cách

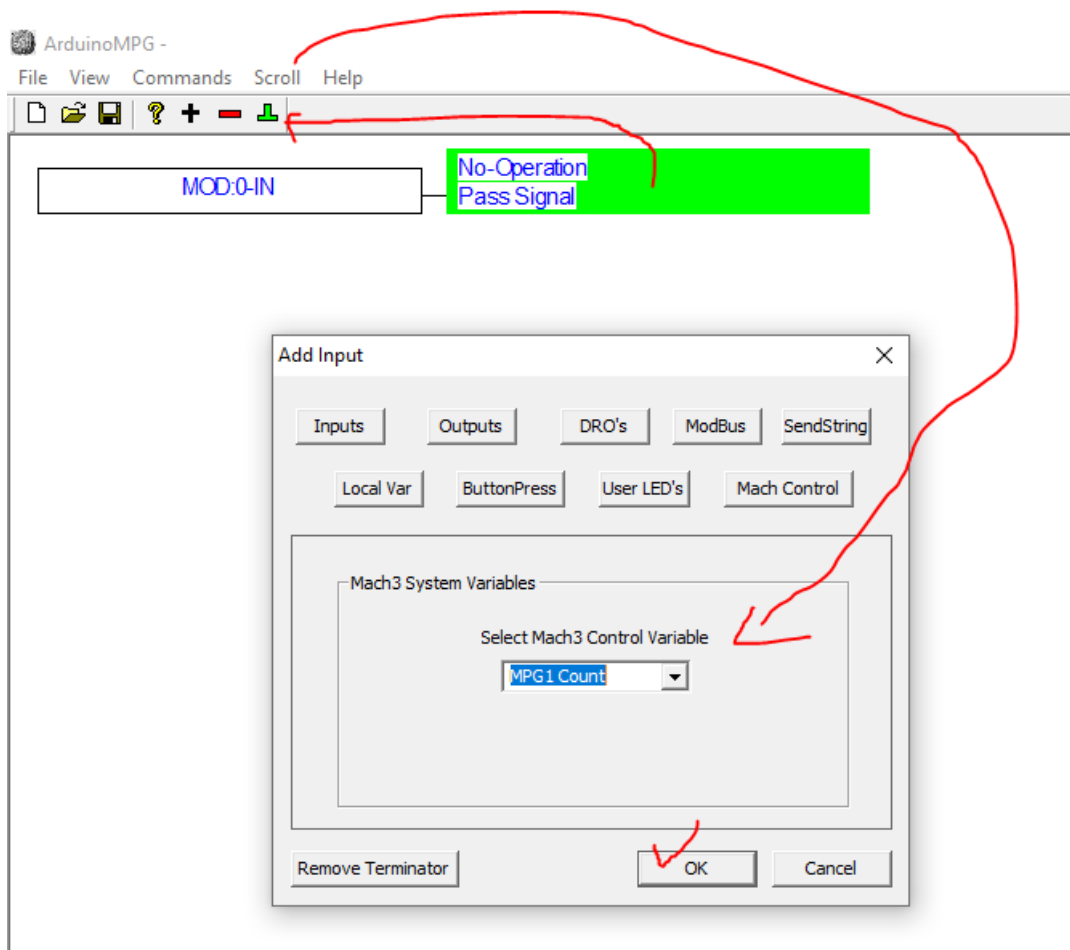
a. MPG_Count: Bấm Add -> ModBus -> chọn như hình dưới sau đó bấm OK, một đầu vào MOD:0-P:1 xuất hiện lên màn hình.



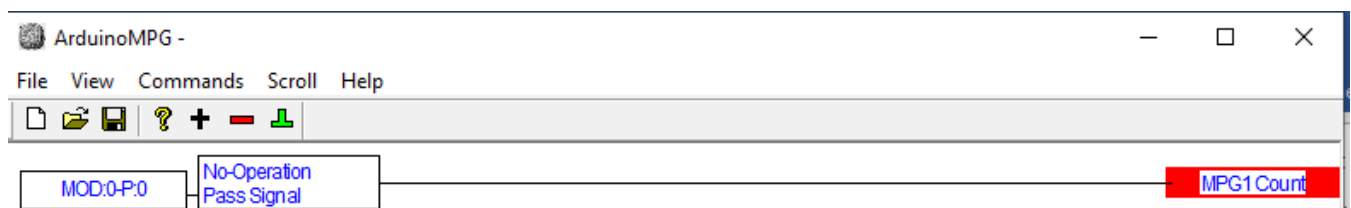
Bấm vào **MOD:0-P:0** để hộp sáng chuyển màu Xanh -> Bấm Add -> **OK**



Lúc này một đối tượng No-Operation Pass Signal xuất hiện. Bấm vào để nó chuyển màu xanh sau đó bấm vào biểu tượng Terminate Lobe -> Chọn Mach Control -> Chọn **MPG1 Count** -> OK

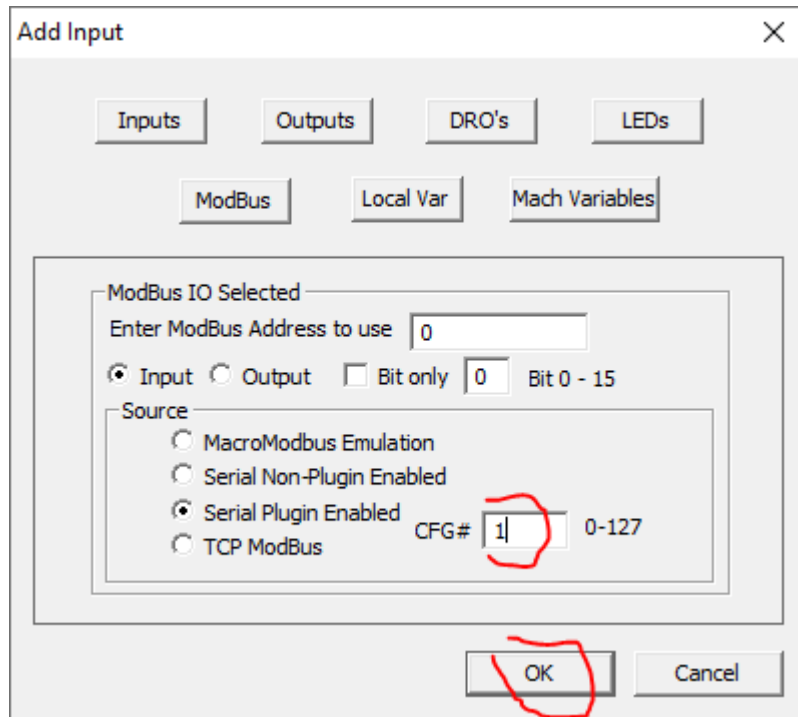


Lúc này một luồng xử lý cho **MPG_Count** đã hoàn thiện



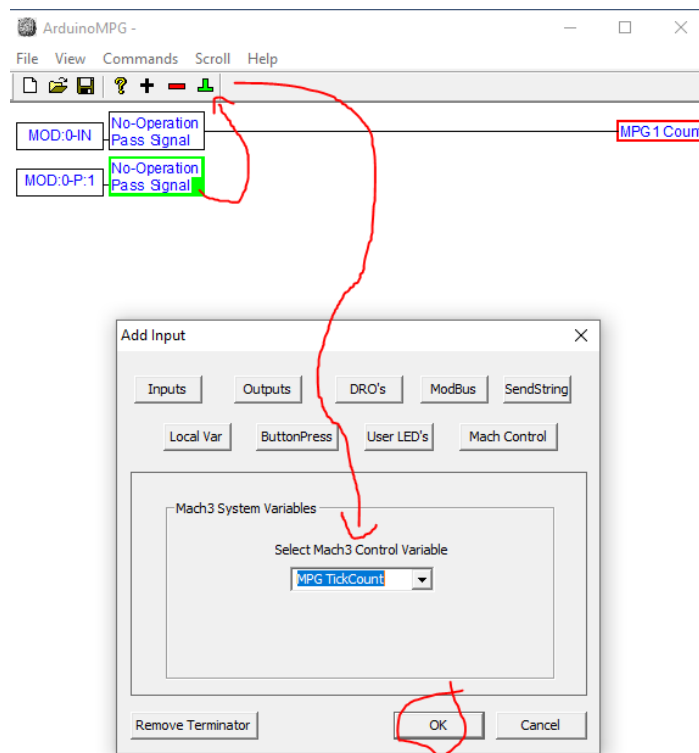
b. MPG_TickCount:

Chọn Add -> ModBus -> chọn CFG# = 1 -> OK

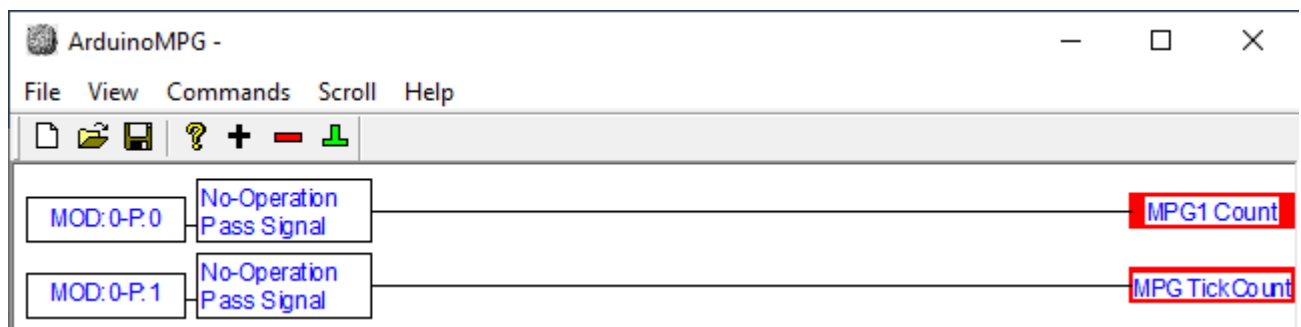


Bấm vào MOD:0-P:1 để hộp sáng chuyển màu Xanh -> Bấm Add -> OK để tạo ra tác vụ xử lý dữ liệu **No-Operation Pass Signal**

Bấm vào **Terminate Lobe** -> Chọn **Mach Control** -> Chọn **MPG_TickCount** -> OK



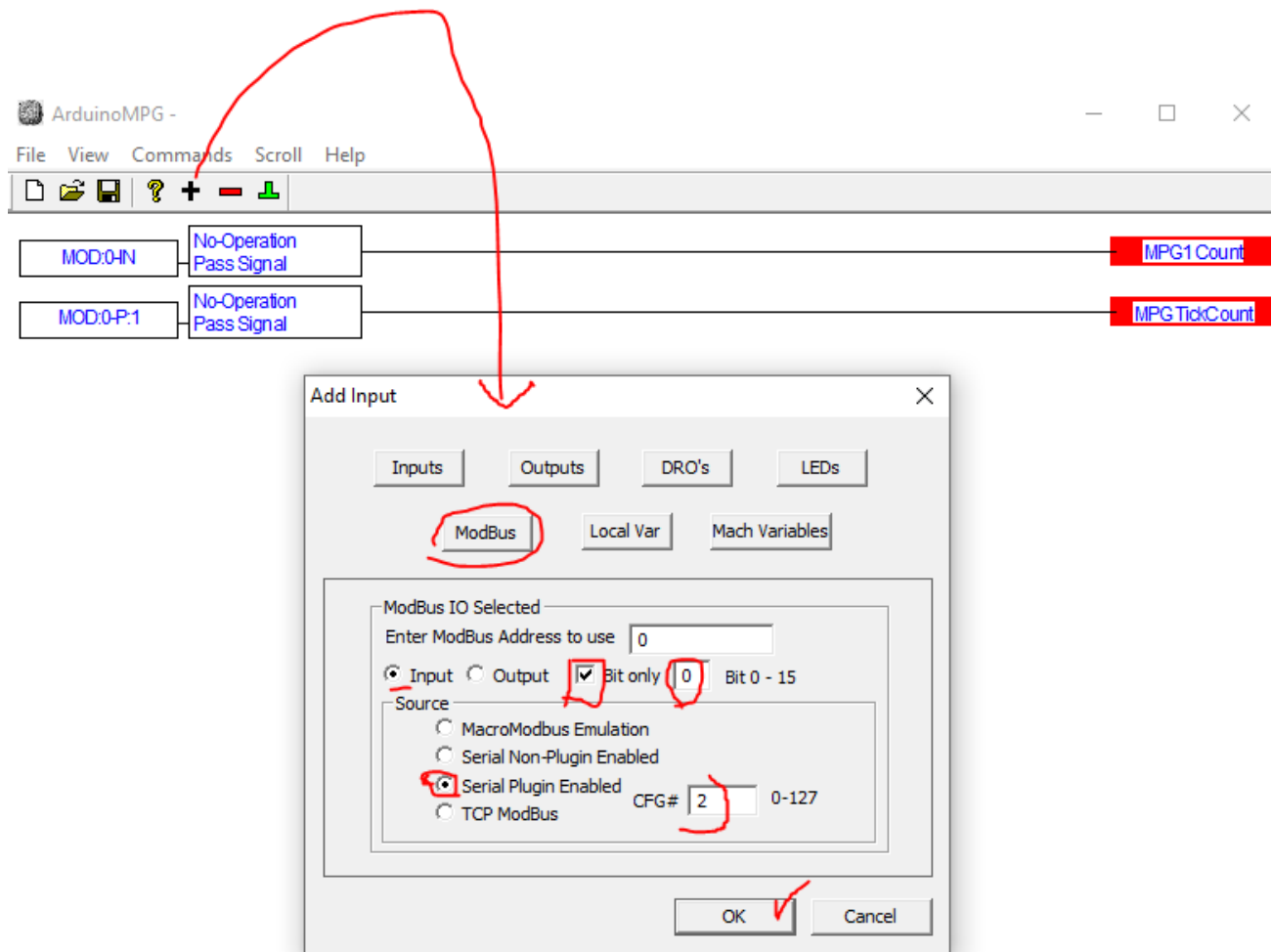
Lúc này màn hình đã có được hai luồng xử lý dữ liệu MGP_Count và MPG_TickCount



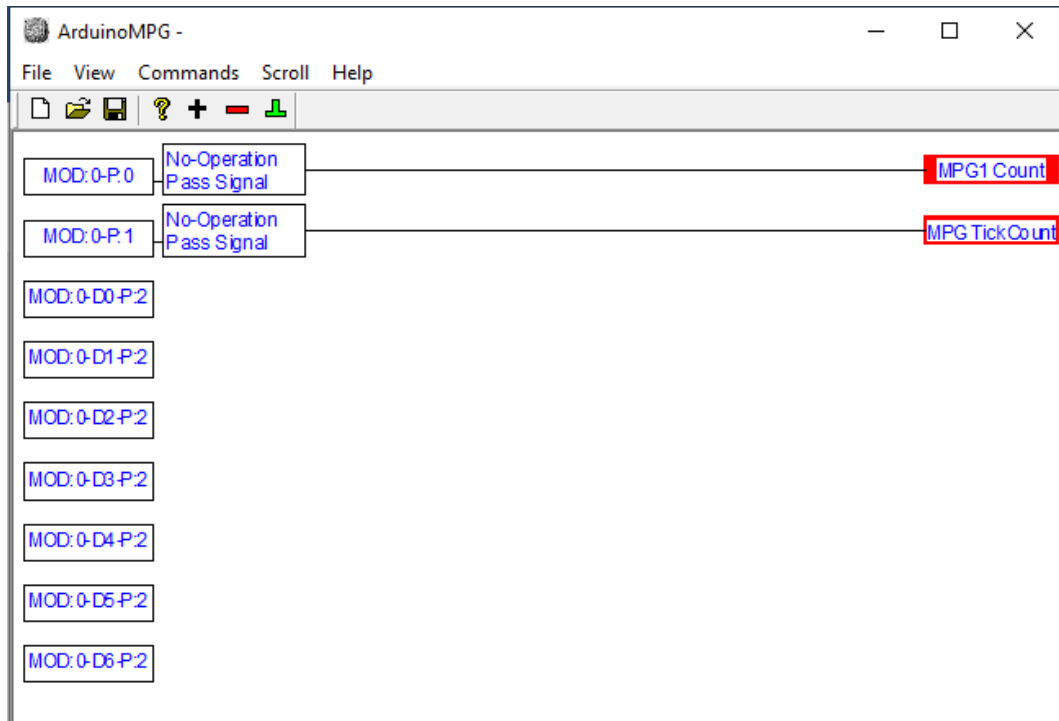
Tiếp tục thực hiện với Cfg#2 để thêm luồng xử lý dữ liệu cho các công tắc chọn chế độ

c. MGP_Switch

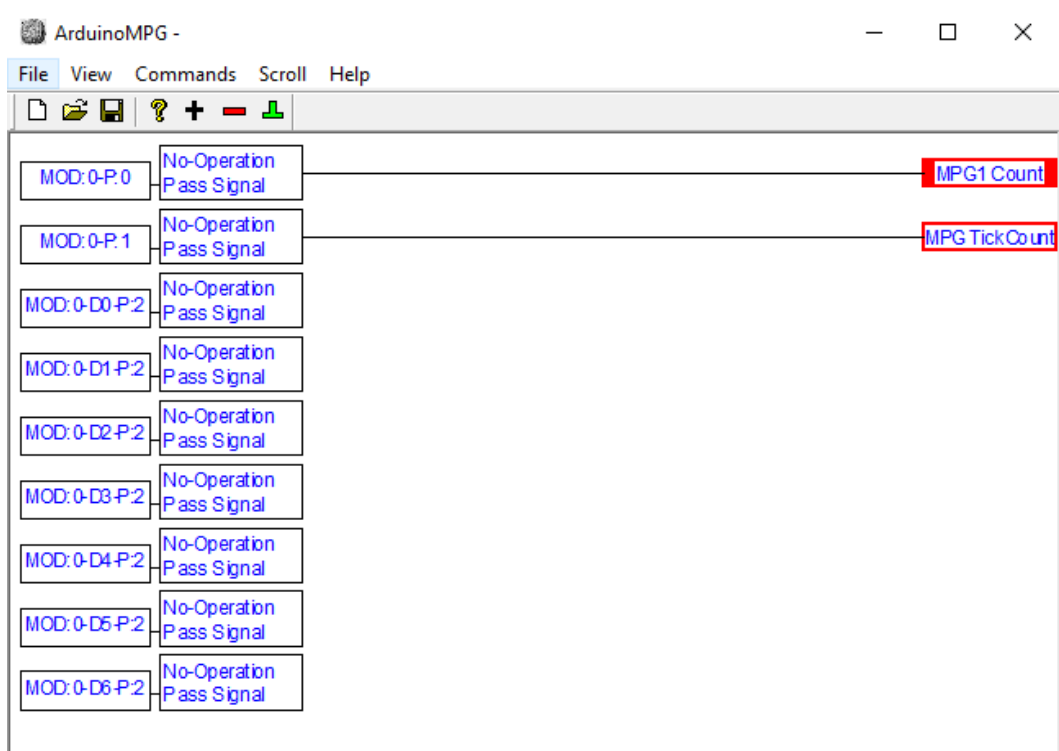
Chọn **Add** -> **ModBus** -> chọn **CFG# = 2**. Tích vào ô **Bit only**, điền 0 (bit 0 là chọn trục X) và bấm **OK** như hình



Làm tương tự lần lượt với mục Bit only là 1 (chọn trục Y), 2 (chọn trục Z), 3 (chọn trục A), 4 (chọn x1), 5 (chọn x10), 6 (chọn x100);



Bấm chọn lần lượt từng MOD:0-D~~x~~-P:2 trong hình cho nó chuyển màu xanh, sau đó bấm Add -> Ok để tạo các No-Operation Pass Signal (tức là không xử lý gì dữ liệu nhận được)



Tại các dòng làm như sau:

MOD:0-D0-P:2 bấm **No-Operation Pass Signal** cho hộp sáng chuyển sang màu xanh-> Terminate Lobe -> Button Press -> 259-MPG1 Xcontrol -> OK

MOD:0-D1-P:2 bấm **No-Operation Pass Signal** cho hộp sáng chuyển sang màu xanh -> Terminate Lobe -> Button Press -> 260-MPG1 Ycontrol -> OK

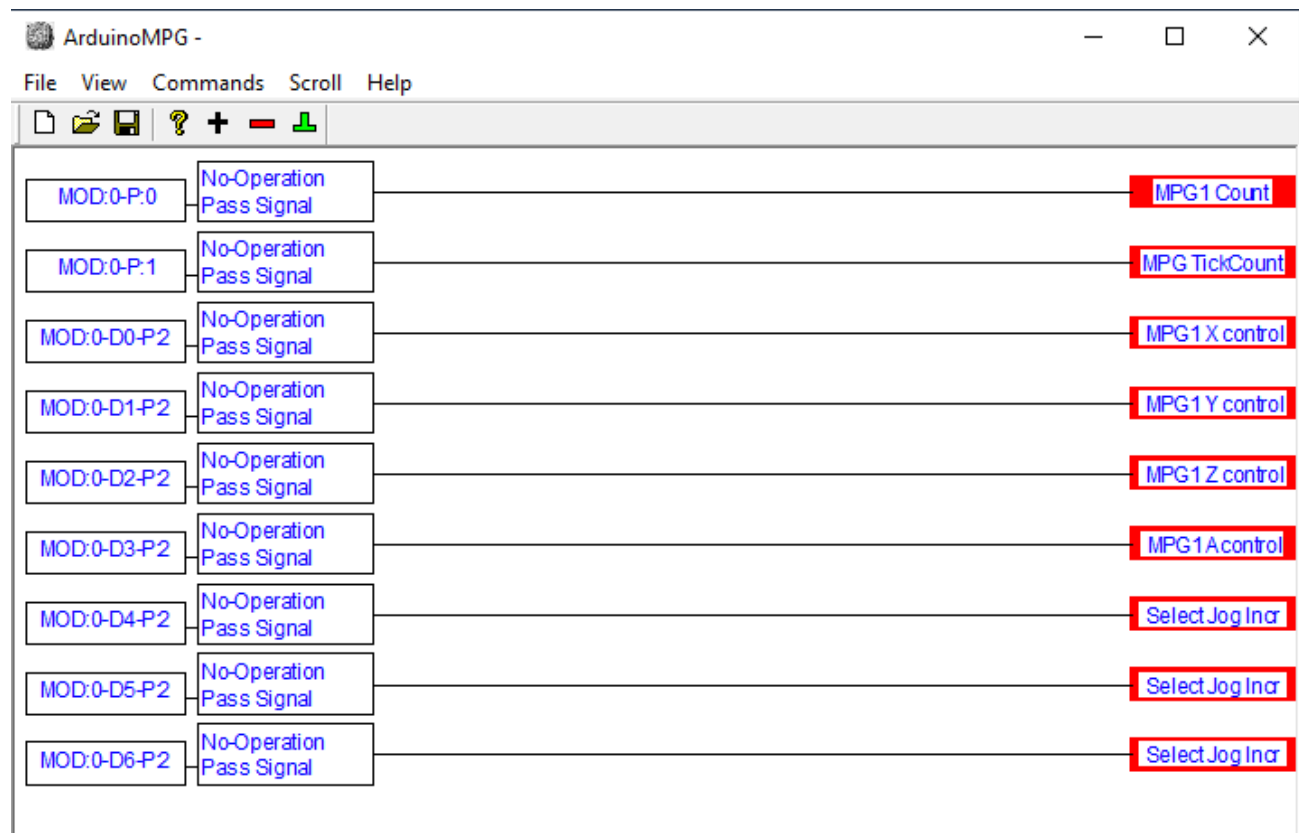
MOD:0-D2-P:2 bấm **No-Operation Pass Signal** cho hộp sáng chuyển sang màu xanh -> Terminate Lobe -> Button Press -> 261-MPG1 Zcontrol -> OK

MOD:0-D3-P:2 bấm **No-Operation Pass Signal** cho hộp sáng chuyển sang màu xanh -> Terminate Lobe -> Button Press -> 262-MPG1 Acontrol -> OK

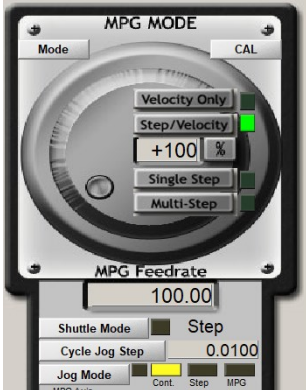
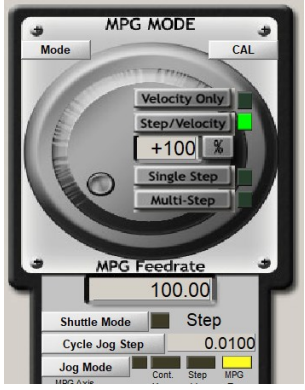
MOD:0-D4-P:2 bấm **No-Operation Pass Signal** cho hộp sáng chuyển sang màu xanh -> Terminate Lobe -> Button Press -> 191- Select Jog Incr 1 -> OK

MOD:0-D5-P:2 bấm **No-Operation Pass Signal** cho hộp sáng chuyển sang màu xanh -> Terminate Lobe -> Button Press -> 192- Select Jog Incr 2 -> OK

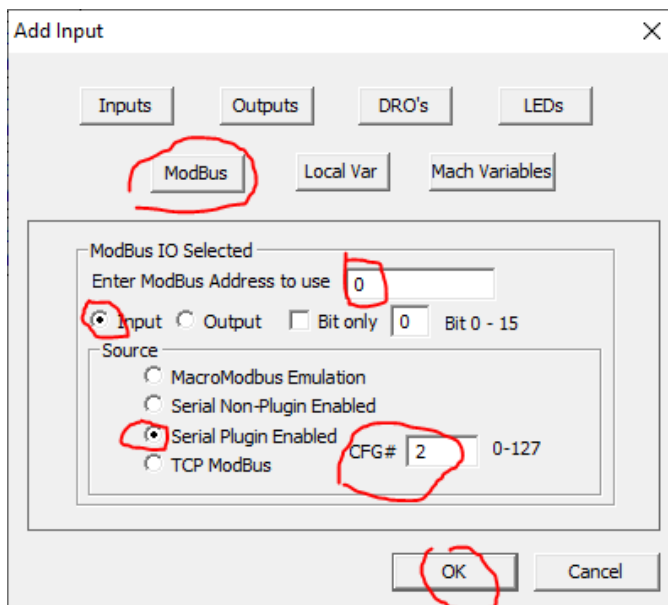
MOD:0-D6-P:2 bấm **No-Operation Pass Signal** cho hộp sáng chuyển sang màu xanh -> Terminate Lobe -> Button Press -> 193- Select Jog Incr 3 -> OK



Về cơ bản như vậy là đã **hoàn thành việc tạo một brain cho MPG có thể hoạt động bình thường**, tuy nhiên chúng ta cần làm thêm hai tác vụ nhỏ nữa để tăng tính tiện dụng như sau:

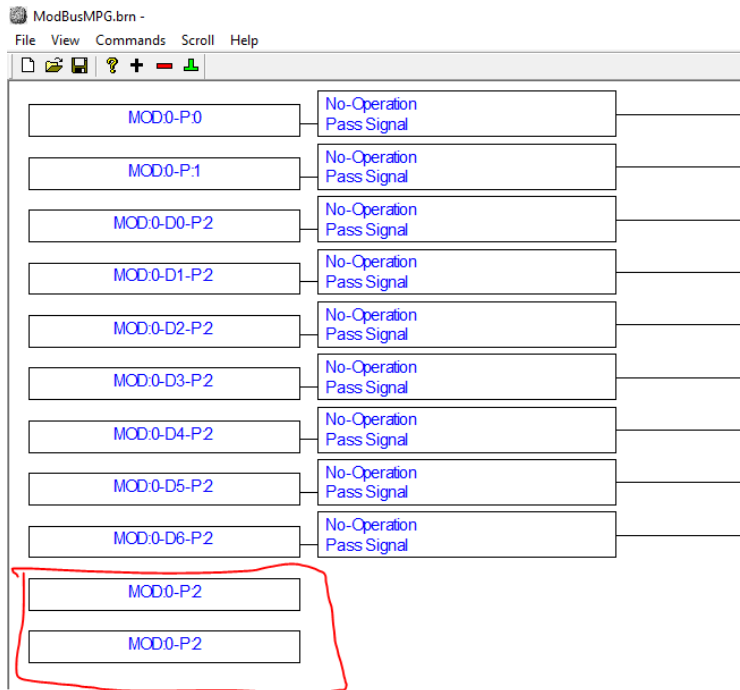
	- Khi vặn núm chọn trục về vị trí OFF thì phần mềm Mach3 cần chuyển Jog Mode (chế độ di chuyển trực thủ công) về chế độ Cont để sử dụng bàn phím máy tính di chuyển trục. - Khi chọn khác vị trí OFF (tức là một trong các trục X, Y, Z hoặc A được chọn bởi núm vặn) thì Jog Mode phải tự động chuyển về chế độ MPG để sử dụng tay cầm MPG.
	
Jog Mode tự về Cont khi nút chọn trục OFF	Jog Mode tự về MPG khi nút chọn X, Y, Z hoặc A

Để làm được điều này cần tạo tiếp 2 đầu vào đọc toàn bộ thanh ghi MPG_Switch trong Cfg #2 bằng cách bấm Add -> ModBus -> chọn như hình dưới



Bấm OK

Lắp lại bước trên lần nữa để tạo thêm một dòng mới giống dòng vừa tạo, như vậy có hai dòng đều là đọc toàn bộ thanh ghi MPG_Switch , kết quả như ảnh dưới đây



Giá trị của thanh ghi chỉ có 4 bit từ bit 0 đến bit 3 của Byte 0 là giá trị cần quan tâm vì nó liên quan đến núm vặn chọn trục, các bit khác sẽ bỏ qua. Do vậy ta cần một xử lý dữ liệu để loại bỏ các bit khác. Bằng cách AND giá trị này với 0x000F (Hex) = 15 (DEC) = 1111 (hệ nhị phân). Tức là xóa hết các bit từ bit4 đến bit15 thành 0.

Thanh ghi MPG_Switch															
Byte 1 (Byte cao)								Byte 0 (Byte thấp)							
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	-	-	-	-	-	-	-	X100	X10	X1	A	Z	Y	X

Trong ngôn ngữ C để lấy 4 bit thấp nhất trong thanh ghi 16 bits thì viết ngắn gọn là: $f(out) = A \& 15$. Tuy nhiên trong Brain của Mach3 thì các cách viết $A \& 15$ hoặc cách viết $A \text{ AND } 15$ hoặc $\text{AND}(A, 15)$ đều không hoạt động, do vậy chúng ta phải sử dụng cách tính toán tương đương ở dạng toán học cơ bản:

$$f(out) = A - \text{INT}(A / 2^{n-1}) * 2^{n-1}.$$

Trong đó: n là số lượng bit thấp cần lấy $n = 4 \Rightarrow 2^{4-1} = 2^3 = 16$

$$\text{Vậy } f(output) = A - \text{INT}(A / 16) * 16$$

Để xử lý dữ liệu bấm vào MOD:0-P:2 cho hộp sáng đổi sang màu xanh -> bấm Add -> Formula

Set Lobe Formula

f(output) = A - INT(A / 16) * 16

The above formula will specify the output condition of this Lobe. You may use the Variables A,B,C,etc.. as parameters in the formula to represent the inputs to this lobe. Also, normal math expressions such as %, sin, cos, etc..
 2 input Example: f(Output) = A * sin(B) / Tan(A) % 1024

Cancel OK

Nhập vào công thức $f(\text{output}) = A - \text{INT}(A / 16) * 16$ sau đó bấm OK

Lập lại tương tự cho dòng số 2. Từ lúc này, giá trị của hàm $f(\text{out})$ trả về kết quả tùy theo vị trí nút xoay chọn trực như sau

Vị trí nút vặn chọn trực	Giá trị các bit tương ứng của từng trực				Giá trị trả về của $f(\text{output})$
	A	Z	Y	X	
OFF	0	0	0	0	0
X	0	0	0	1	1
Y	0	0	1	0	2
Z	0	1	0	0	4
A	1	0	0	0	8

ModBusMPG.brm -

File View Commands Scroll Help

MOD0-P0 No-Operation Pass Signal

MOD0-P1 No-Operation Pass Signal

MOD0-D0-P2 No-Operation Pass Signal

MOD0-D1-P2 No-Operation Pass Signal

MOD0-D2-P2 No-Operation Pass Signal

MOD0-D3-P2 No-Operation Pass Signal

MOD0-D4-P2 No-Operation Pass Signal

MOD0-D5-P2 No-Operation Pass Signal

MOD0-D6-P2 No-Operation Pass Signal

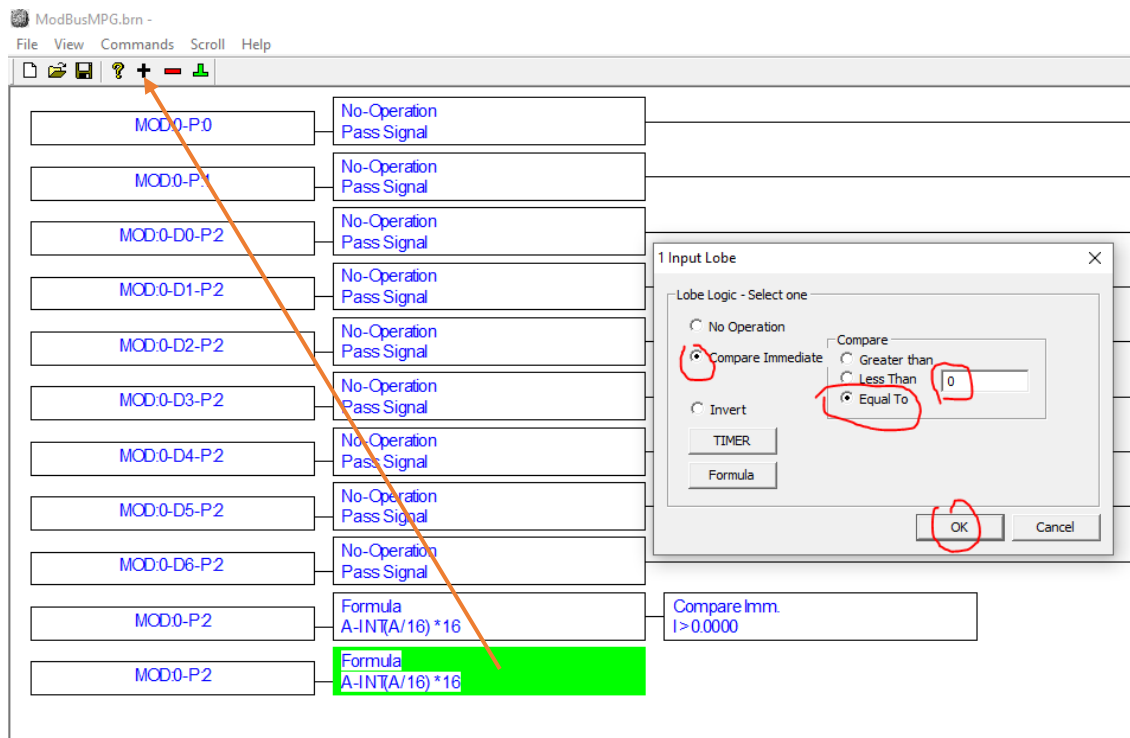
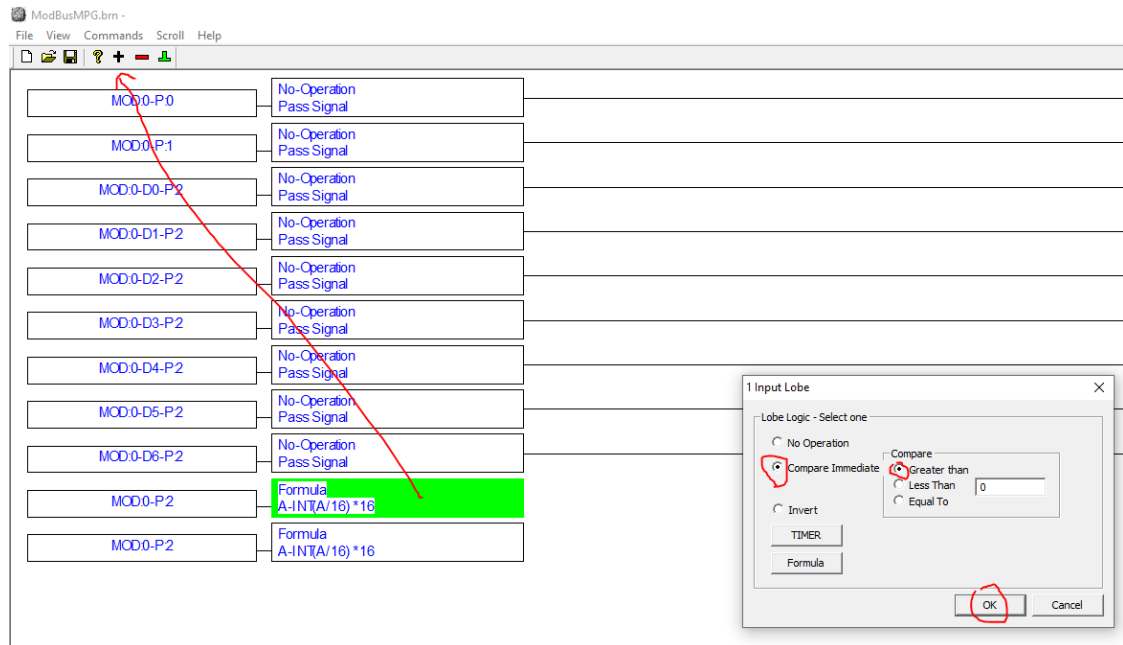
MOD0-P2 Formula $A - \text{INT}(A/16) * 16$

MOD0-P2 Formula $A - \text{INT}(A/16) * 16$

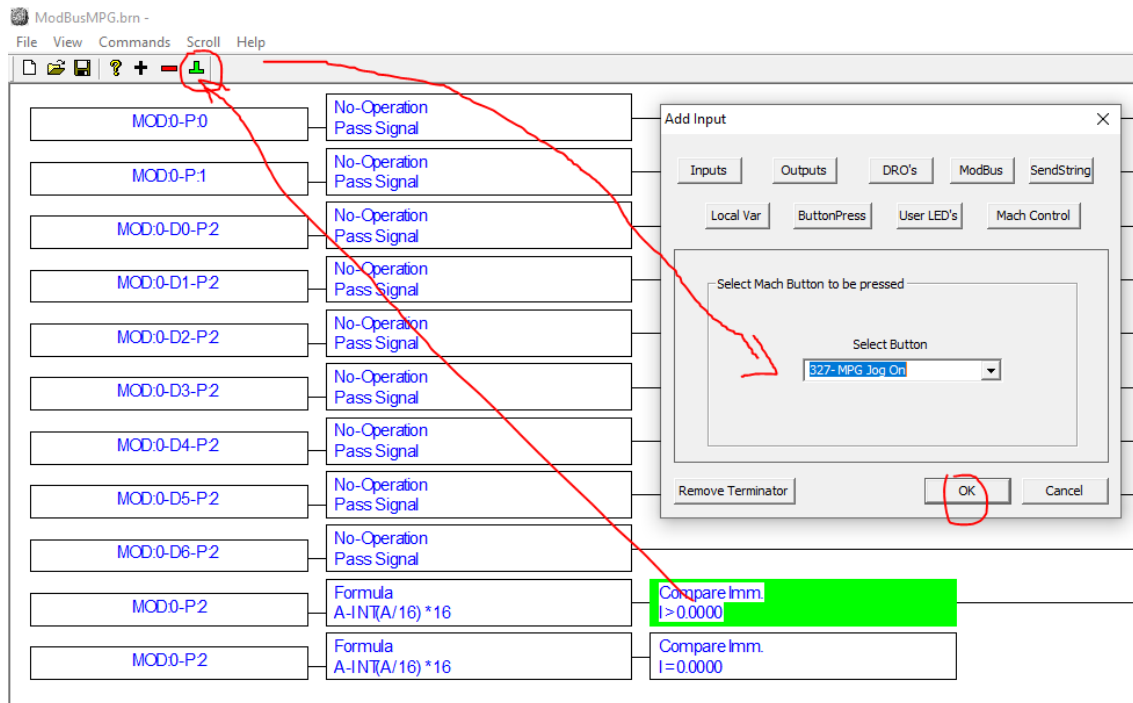
Tiếp theo:

- Một dòng ta thêm điều kiện so sánh nếu $f(out) > 0$ thì chuyển chế độ Jog Mode thành MPG để dừng MPG;

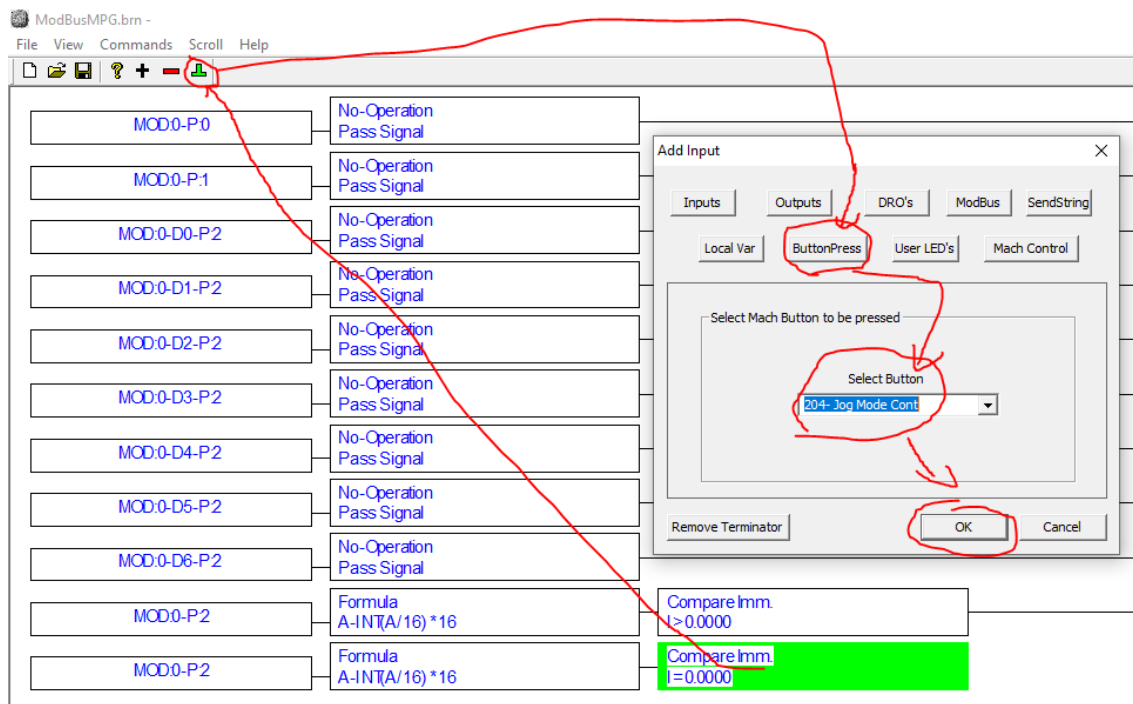
- Một dòng thêm điều kiện so sánh nếu $f(out) = 0$ thì chuyển chế độ Jog Mode thành Cont để dùng bàn phím;

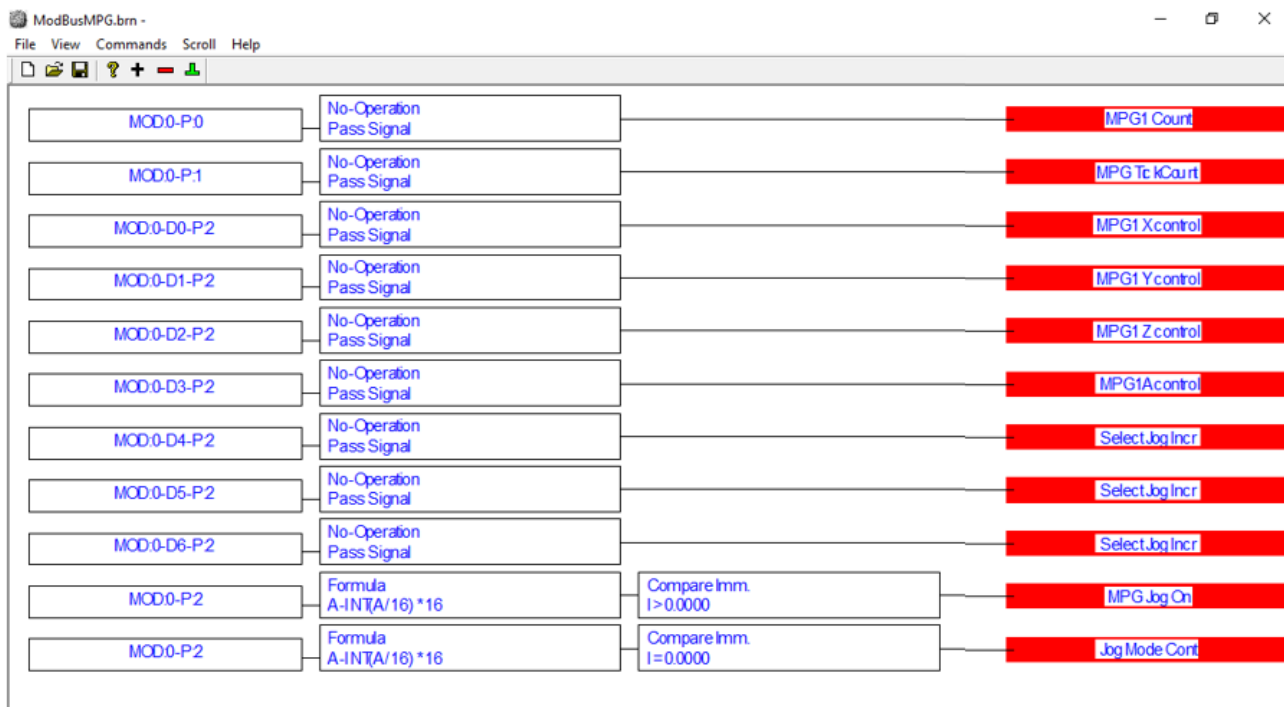


Bấm vào **Compare Lmm I > 0.0000** cho hộp sáng chuyển sang màu xanh -> Chọn **Terminate Lobe -> Button Press -> chọn 327- MPG Jog On -> Bấm OK**



Bấm vào **Compare Lmm I = 0.0000** cho hộp sáng chuyển sang màu xanh -> Chọn **Terminate Lobe -> Button Press -> chọn 204- Jog Mode Cont -> Bấm OK**

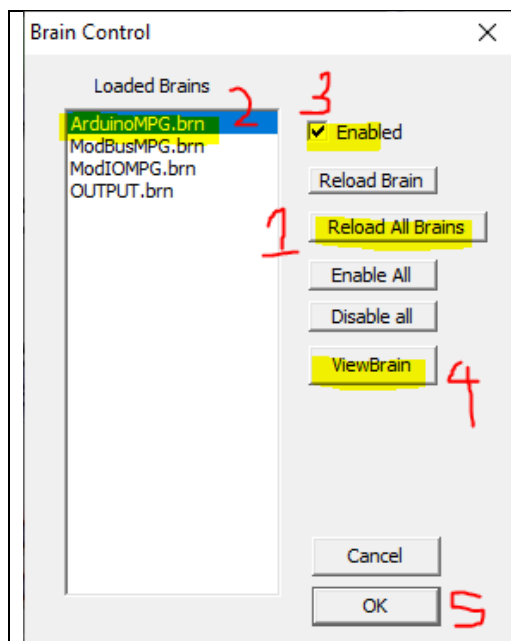




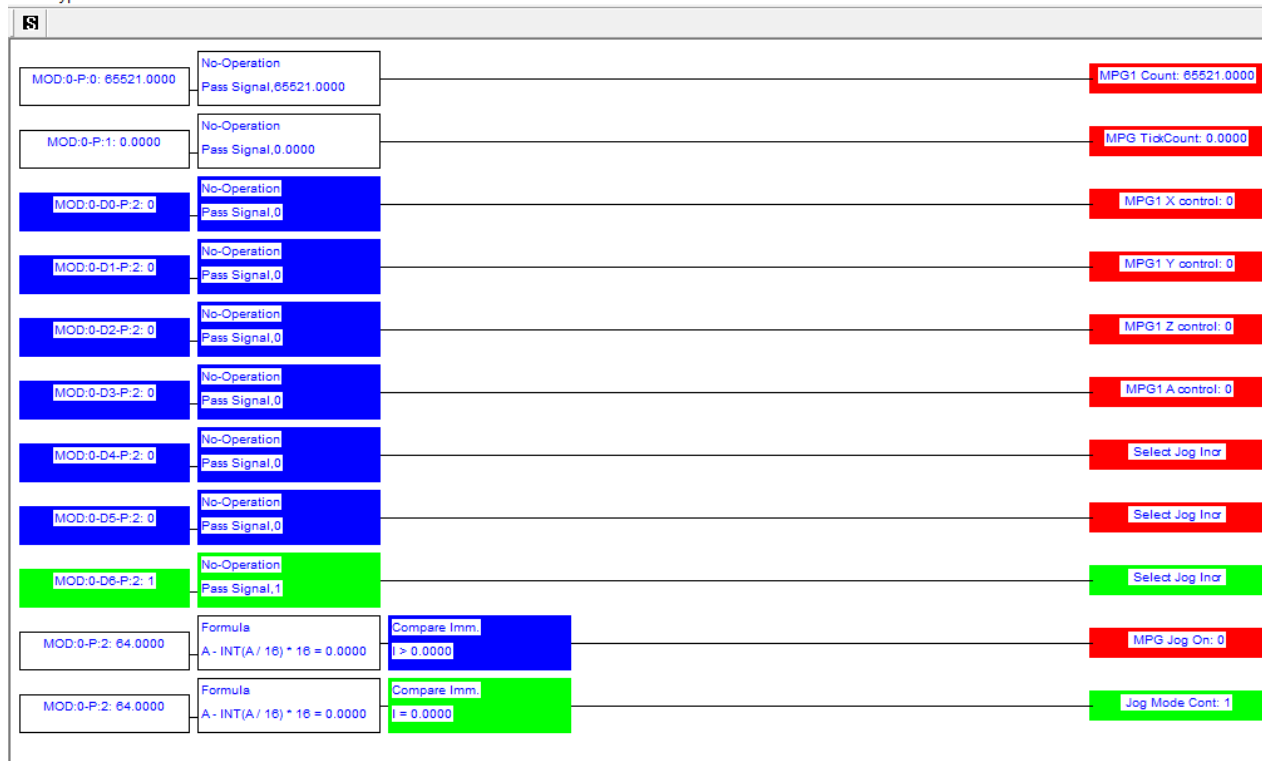
Ảnh chụp màn sau khi hoàn tất các Lobe cho Brain MPG sử dụng Modbus

Từ màn hình Brain bấm **File** -> **Save** -> đặt tên cho brain vừa tạo, ví dụ **ArduinoMPG** sau đó bấm **Save** để lưu lại. Như vậy đã hoàn thành việc tạo ra một Brain mang tên **ArduinoMPG**. Bước tiếp theo chúng ta cần Load Brain này lên mach3 để chạy liên tục.

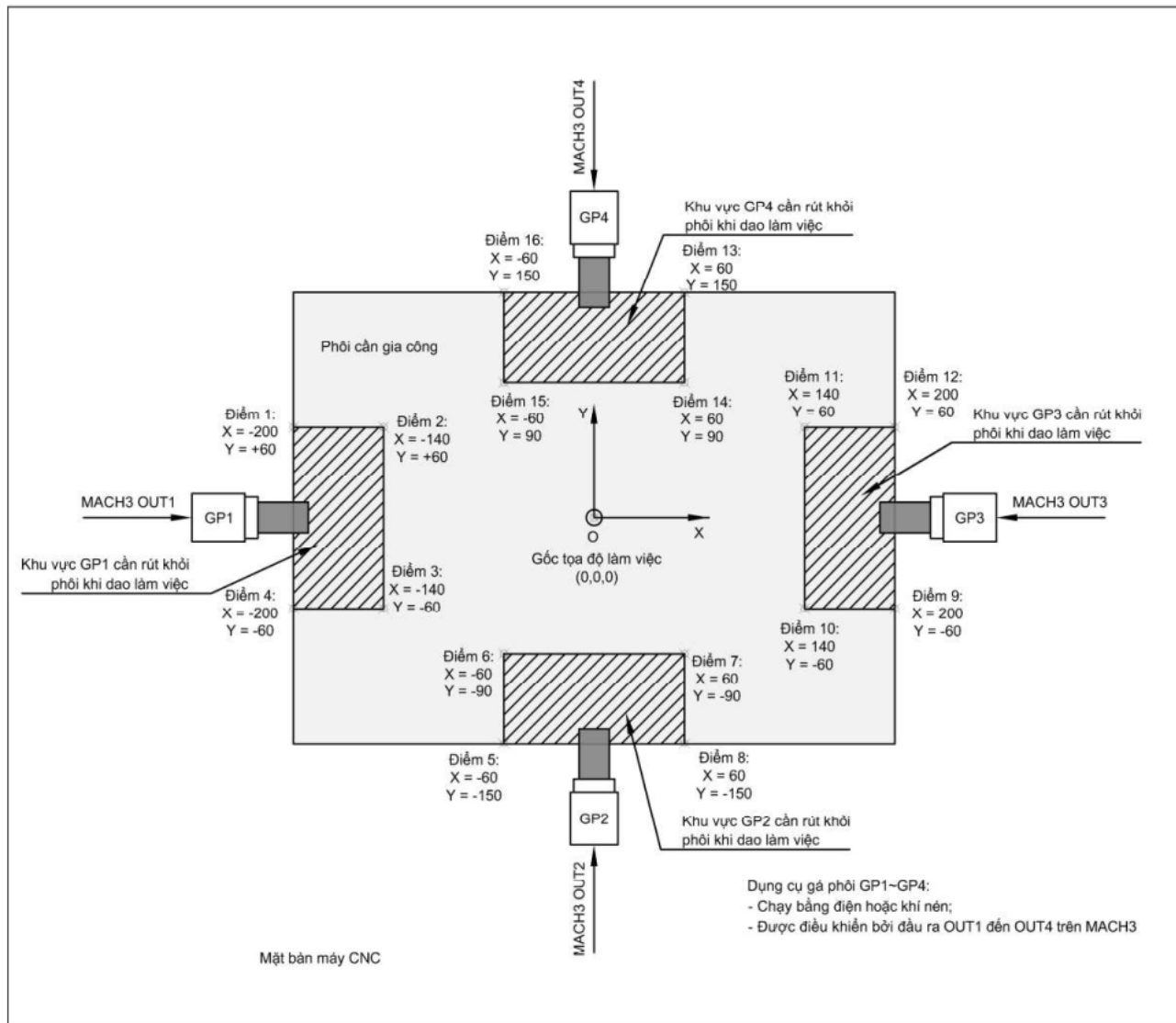
III.1.4. Kích hoạt cho Brain tạo ở bước 3 chạy trong Mach3



Từ màn hình chính của Mach3 chọn **Operator** -> **Brain Control** -> **Reload All Brains** -> chọn **ArduinoMPG** -> Tích vào hộp kiểm **Enable** -> Bấm **ViewBrain** để xem màn hình làm việc của Brain, bấm **OK** để đóng cửa sổ này lại. Như vậy là hoàn tất quá trình tạo một MPG sử dụng Brain kết hợp ModBus. Lúc này tay cầm MPG đã có thể hoạt động bình thường trên Mach3



III.2. Dự án điều khiển các van kẹp phôi tự động



II.2.1. Ý tưởng của dự án

Ý tưởng của dự án này là sử dụng 4 bộ gá phôi chạy bằng điện hoặc khí nén (GP1 đến GP4), điều khiển đóng mở bởi MACH3. Trong thực tế có thể cấu hình đóng mở bằng nhiều cổng khác nhau và nhiều cách khác nhau, thậm chí có thể sử dụng ModBus để đóng mở bằng mạch ngoài, tuy nhiên để đơn giản hóa, dự án này sử dụng các cổng có sẵn từ OUT1 đến OUT4 của bo mạch Mach3 làm ví dụ.

Yêu cầu đặt ra là khi dao chạy vào các vùng được tô gạch chéo trên hình thì bộ gá phôi tại vùng đó phải được mở (tức là rút khỏi phôi) để tránh va chạm giữa dao với bộ gá. Tọa độ các điểm không chế của từng vùng là các điểm từ Điểm 1 đến điểm 16 có tọa độ cụ thể như hình vẽ.

Thuật ngữ dùng cho các bộ gá (GP1 đến GP4) quy ước như sau:

- + **Đóng** - là bộ gá thực hiện gá phôi để giữ cho phôi cố định trên bàn máy CNC
- + **Mở** - là bộ gá rút khỏi phôi để tránh va chạm với dao khi gia công trong vùng được tô gạch chéo trên hình

III.2.2. Lô gic xử lý vấn đề của dự án:

- Nếu tọa độ $X < -140$ đồng thời $-60 < Y < 60$ thì mở GP1. Ngoài các điều kiện này thì Đóng GP1.

- Nếu tọa độ $-60 < X < 60$ đồng thời $Y < -90$ thì mở GP2. Ngoài các điều kiện này thì Đóng GP2.

- Nếu tọa độ $X > 140$ đồng thời $-60 < Y < 60$ thì mở GP3. Ngoài các điều kiện này thì Đóng GP3.

- Nếu tọa độ $-60 < X < 60$ đồng thời $Y > 90$ thì mở GP4. Ngoài các điều kiện này thì Đóng GP4.

III.2.3. Tiến hành tạo Brain

Tạo một Brain mới: Từ màn hình Mach3 -> Operator -> Brain Editor -> đặt tên là Gia-phoi-tu-dong bấm OK để vào màn hình soạn Brain như thường lệ.

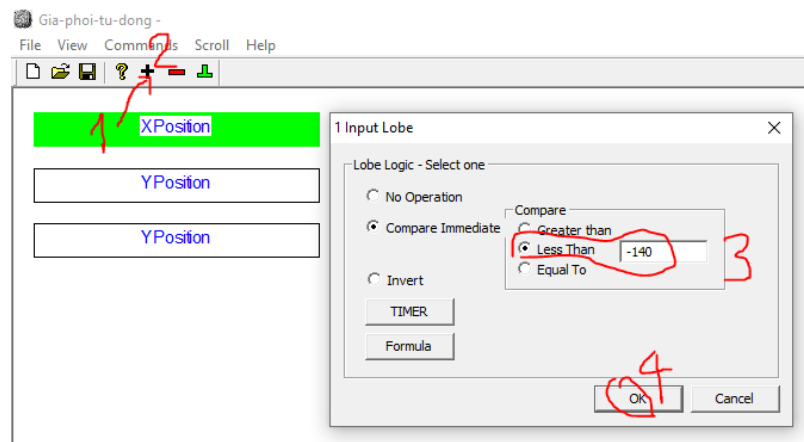
+ Tạo Lobe điều khiển OUT1 (GP1):

Add -> DROs -> 842-X Position

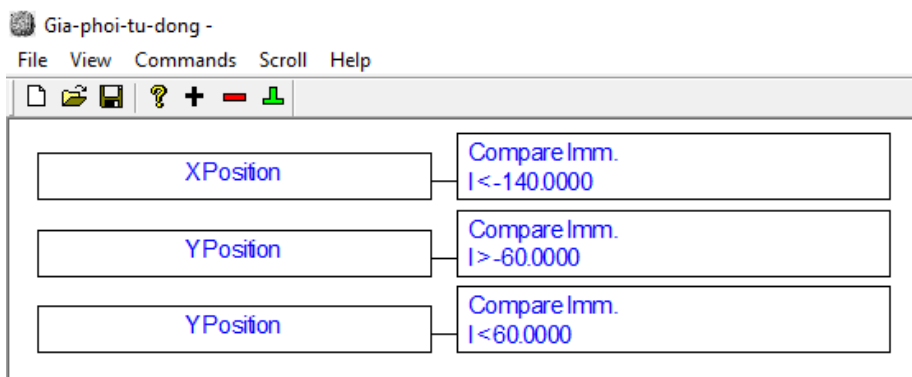
Add -> DROs -> 843-Y Position

Add -> DROs -> 843-Y Position

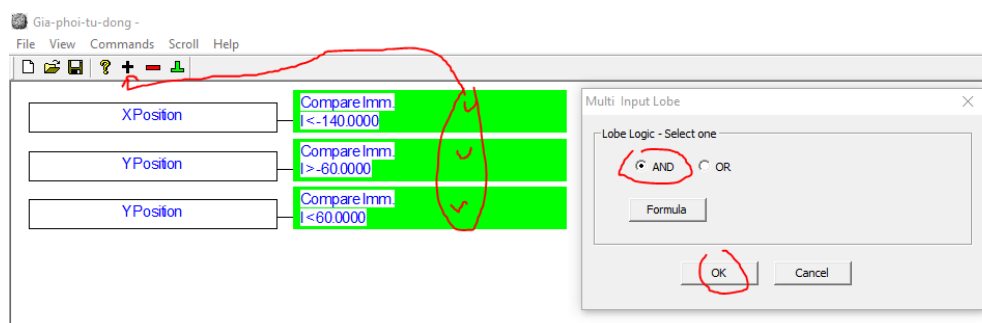
Bấm vào X Position để hộp sáng chuyển màu xanh -> Add -> chọn so sánh < -140 -> bấm OK



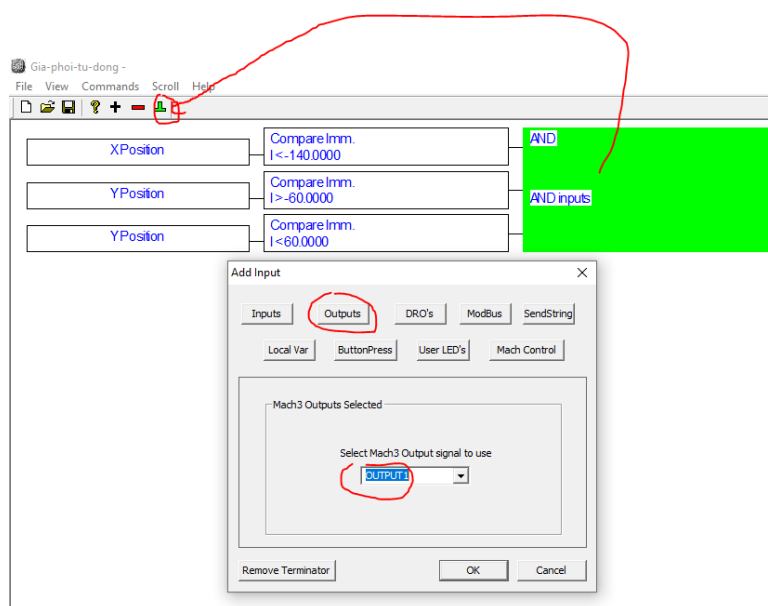
Tương tự cho $Y > -60$ và $Y < 60$



- Chọn cả ba xử lý so sánh cho 3 hộp sáng chuyển màu xanh -> Add -> AND -> OK

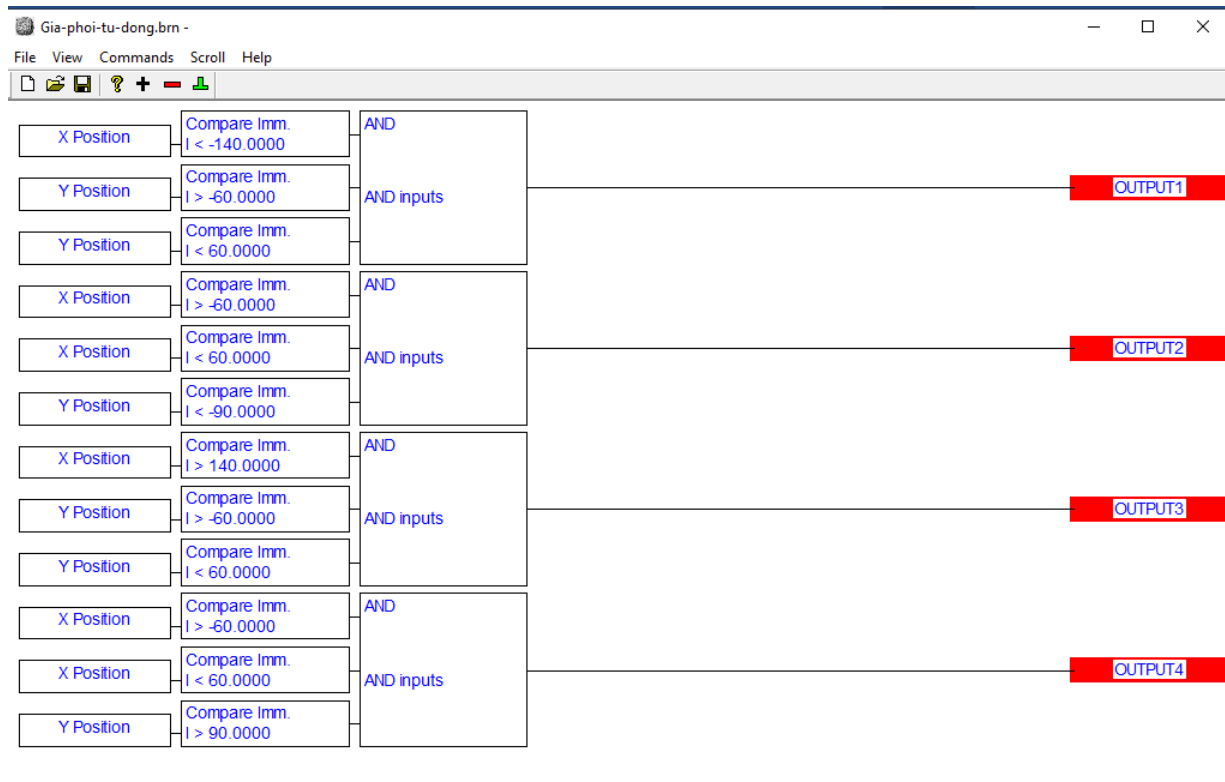


Bấm chọn vào Logic AND vừa tạo cho hộp sáng chuyển xanh -> Terminate Lobe -> Outputs -> OUTPUT1



Hoàn thành xử lý cho OUT1 (tức là GP1)

Thực hiện tương tự cho GP2, GP3 và GP4, cuối cùng ta có Brain như sau



Bấm **File** -> **Save** -> đặt tên file **Gia-phi-tu-dong.brn** -> **Save** để lưu lại brain.

III.2.4. Kích hoạt cho brain chạy

Từ màn hình chính của Mach3 chọn **Operator** -> **Brain Control** -> **Reload All Brains** -> chọn **Gia-phi-tu-dong.brn** -> Tích vào hộp kiểm **Enable** -> Bấm **ViewBrain** để xem màn hình làm việc của Brain, bấm **OK** để đóng cửa sổ này lại. Như vậy là hoàn tất quá trình tạo một Brain gá phôi.

