AN INTRODUCTION TO THE MASTER CONTROL INSTRUCTIONS (MC, MCR)

Introduction

The programming of Mitsubishi PCs will always result in the use of the basic instruction set. It does not matter how many applied instructions are used because the final result is that the majority of the program is written using basic, fundamental, instructions. Once a full understanding of these building blocks has been achieved, advanced and manipulative programming will no longer be a chore.

The Master Control instructions are part of the basic instruction set.

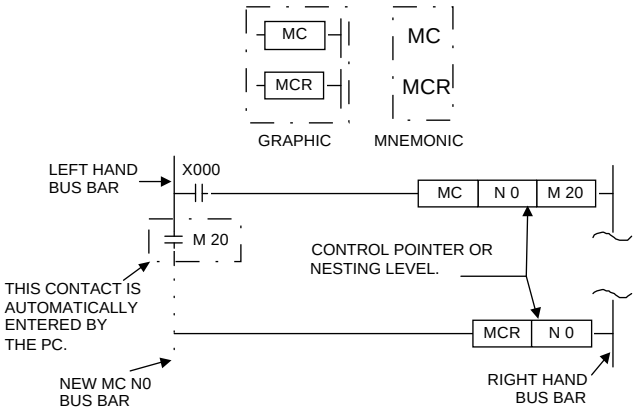
The Master Control Instructions

There are two separate instructions required to make a Master Control program work. Quite simply one instruction marks the beginning of the Master Control (MC), the other instruction marks the end of the Master Control (MCR). Some points worth remembering about MC and MCR now follow;

- Both MC and MCR are special output conditions, which use outputs, Y or internal flags, M. Hence they always appear last on a line of program. On the ladder diagram the MC and MCR symbols are the first instructions from the right hand side bus bar.
- Master control constructions are not jumped by the program scan; even if they are not activated, i.e. all inputs and outputs are continually monitored and refreshed.
- All MC instructions contain two pieces of information, the nesting level which is identified by a nest pointer and a control bit which activates the sub-program under that master control.
- A master control construction can have up to eight control or nesting pointers. These pointers are represented by the N numbers in the MC and MCR instructions.
- Master control constructions can be written, if required, by using only one control pointer. However each master control must still have a different control bit for the program to work correctly.
- When a master control is reset by using the MCR if any further master control constructions are active below the reset nesting level, then those following master control constructions will also be reset.

What do MC and MCR contacts look like?

In ladder format the MC and MCR instructions appear as shown in the following diagram. The diagram also shows the relative location where either an MC or an MCR can be found within a ladder rung.

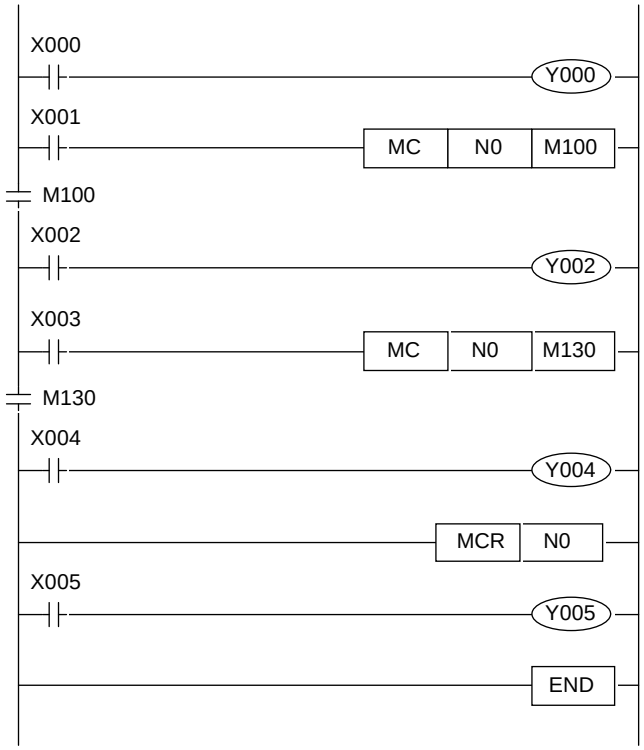


How do the MC , MCR Instructions Work?

There are actually two different methods of use for the MC and MCR commands. The first method described is very similar to the Mitsubishi F series use of the master control construction.

Example program 1

Ladder format;



AN INTRODUCTION TO THE MASTER CONTROL INSTRUCTIONS (MC, MCR)

Instruction format;

PROGRAM STEP No.	INSTRUCTION PROGRAM		
0	LD	X	000
1	OUT	Y	000
2	LD	X	001
3	MC	N	0
		M	100
6	LD	X	002
7	OUT	Y	002
8	LD	X	003
9	MC	N	0
		M	130
12	LD	X	004
13	OUT	Y	004
14	MCR	N	0
16	LD	X	005
17	OUT	Y	005
18	END		

Before analyzing exactly how example program number 1 works, the program structure should be observed. The situation is, that there are two Master control constructions.

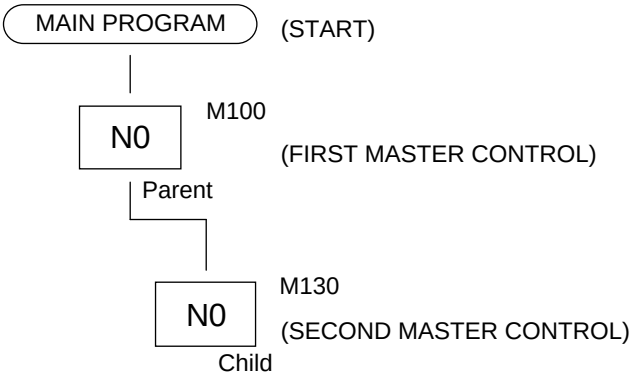
The next question is how are these master controls related?

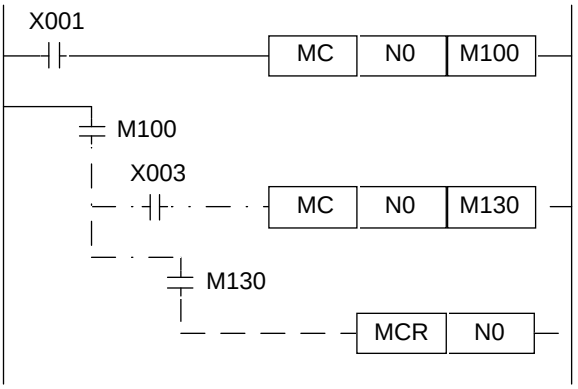
By following the program it can be seen that the two master controls are nested together (this is shown graphically in the following diagram).

What does the term “nesting” mean?

Nesting is when a portion of program (often known as the ’child’) is dependent on the activation of a larger area of program (again often known as a ’parent’). The ’child’ program is actually a sub-program of the ’parent’. The master control instructions offer the programming tools to construct this nested, ’parent’-’child’ relationship.

This relationship is used in example program number 1. The following diagrams display this relationship in a pictorial manor. The first master control N0 M100 is the ’parent’ and the second master control N0 M130 is the “child”.





What does a nest look like?

The first diagram shows the nesting in a flow diagram format. This is very basic but it does show the idea that the flow of the diagram is through each master control.

The second diagram shows the nesting in a programming format. This is showing exactly the same information as the flow diagram but in a more familiar format. When a graphical programmer is used to monitor example program 1, there will be contacts mounted in the vertical bus bar. These contacts are automatically put in by the program when ever a master control is encountered. If the vertical contacts are considered as acting in a similar way to normal contacts, it can be assumed that the program following the contact (up until the MCR instruction) will only be activated when that vertical contact is on. This use of master control has lead to the common analogy that the master control is similar to an electrical circuits master contact-breaker. This is a good way of describing this use of a master control, but it is only one use!

There is no limit to how many master controls may be linked in the above format, but as always it is prudent to find other ways of achieving the same result, especially if a more efficient method is available.

AN INTRODUCTION TO THE MASTER CONTROL INSTRUCTIONS (MC, MCR)

What Actually Happens in Example program 1?

When the program is run the following conditions, when applied, give the noted results;

a) When X000 is turned on output Y000 turns on. If inputs X002 or X004 are turned on, no response is noted (actually if the inputs are monitored they do come on but the associated outputs do not). If X005 is turned on then output Y005 will also turn on.

b) When input X001 is turned on the first master control is activated. This now allows input X002 to drive output Y002. Input X004 will still not drive output Y004 until master control N0 M130 has been activated. Both inputs X000 and X005 will still drive their respective outputs.

c) When inputs X001 and X003 are turned on both master control constructions are active. This means that inputs X000, X002, X004 and X005 will drive their respective outputs.

d) If input X001 is now turned off it can be seen that, not only does output Y002 no longer operate but also the second master control N0 M130; hence its control program involving X004 and Y004 also becomes non-operational.

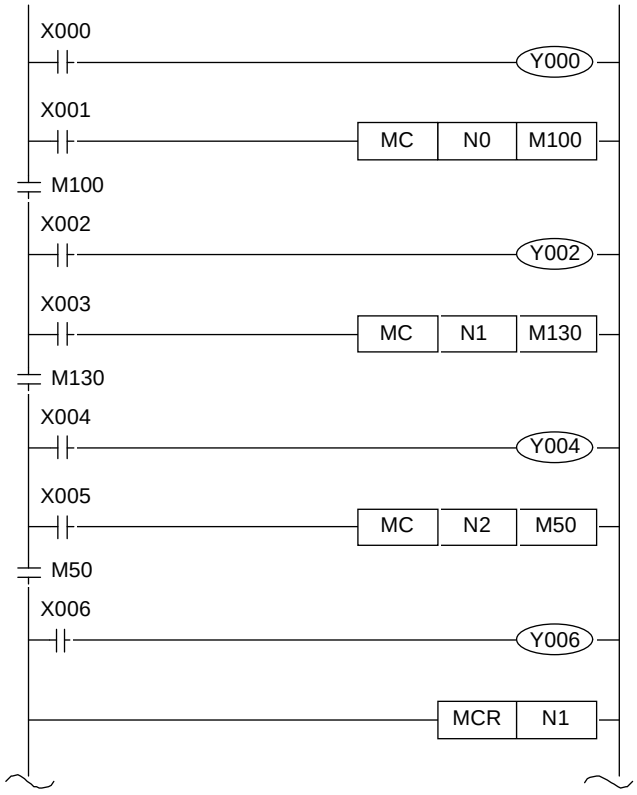
Particular note should be paid to how the inputs are still registered yet the outputs do not operate. This highlights how the master control is preventing outputs coming on if the master control is not active. In some respects the master control can be considered as a switch for selecting a completely separate sub-program.

When programming with master controls, all standard programming techniques must still be followed. It is a common mistake for users to think that the master control construction will allow the successful programming of dual coils. This is not the case. It is easily proved that the inputs are continually scanned by the program. The above tests will verify this. When the inputs are scanned so will the outputs be scanned, therefore dual coiling will cause an error in the operation of the user program.

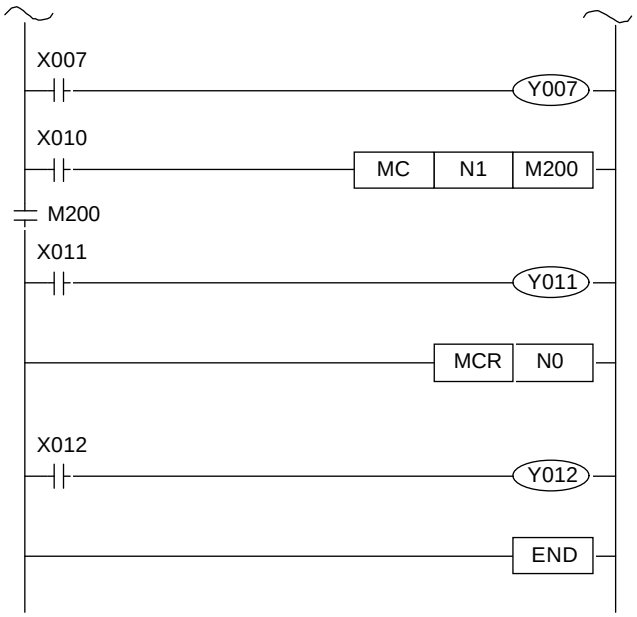
A second use of master control utilizes the nesting level pointers. The following example shows how this method is used.

Example Program 2 - Nesting Pointers

Ladder format; (part 1)



Ladder format; (part 2)



AN INTRODUCTION TO THE MASTER CONTROL INSTRUCTIONS (MC, MCR)

The second example program number 2 (on the previous page) is quite complicated, but it does show two very important points about nesting levels. Explanations of these points will follow but first the instruction program and an explanation of the basic structure of the example program.

Instruction format (Part 1);

PROGRAM STEP No.	INSTRUCTION PROGRAM		
0	LD	X	000
1	OUT	Y	000
2	LD	X	001
3	MC	N	0
		M	100
6	LD	X	002
7	OUT	Y	002
8	LD	X	003
9	MC	N	1
		M	130
12	LD	X	004
13	OUT	Y	004
14	LD	X	005
15	MC	N	2
		M	50
18	LD	X	006
19	OUT	Y	006
20	MCR	N	1
Part 2;			
n	LD	X	007
n+1	OUT	Y	007
n+2	LD	X	010
n+3	MC	N	1
		M	200
n+6	LD	X	011
n+7	OUT	Y	011
n+8	MCR	N	0
n+10	LD	X	012
n+11	OUT	Y	012
n+12	END		

Example Program 2 - The Structure

In example program 1 the definition of a nested program was given in terms of a 'parent'-'child' relationship. In this second example the nest levels are used in a slightly different way. This is best explained in terms of a 'Tree'. This idea can be followed to a logical conclusion with 'Trunks', 'Branches', 'Twigs' and 'Leaves' making up parts of the tree.

How does this relate to the program in example 2?

If the example program is broken down and analyzed it can be seen that there are seven major components;

I) The main program,

II) Master control N0 M100,

III) Master control N1 M130,

IV) Master control N2 M50,

V) Master control reset N1,

VI) Master control N1 M200,

VII) Master control reset N0.

Each of the components appears in the order listed. It should be noted that there are three different nest level numbers. These are N0, N1 and N2. One further point to recognize is that there are two N1 nests but that there is a master control reset of nest level N1 after the first occurrence of master control N1.

Now, to help visualize the program structure, the 'Branch' labels will be applied;

- The main program is the source from where all of the other sub-programs originate from. For this explanation the main program will be known as the 'Trunk'.
- Following down the list of seven components, the next major item is master control N0. This is the first sub-program off of the main program. Master control N0 will now be known as a 'Branch', because the sub-program branches away from the main program.

AN INTRODUCTION TO THE MASTER CONTROL INSTRUCTIONS (MC, MCR)

- The next item encountered is the master control N1 M130. If the same logic is followed that was applied to MC N0, then MC N1 will be another branch. MC N1 cannot be called a 'Branch' because it is a smaller part of master control N0, hence it will be called a 'Twig'.
- Once more looking at the list of major conditions, it can be seen that another master control N2 appears. This is the least important of the conditions encountered so far. Master control N2 is a branch from MC N1 which is a branch from MC N0 which is itself a branch from the main program. The next condition to follow master control N2 is a master control reset N1. The MCR indicates there are no further branches from master control N2.
  So what should nest level N2 be called? Logically if nest level N1 is a 'Twig' and nest level N2 is the last nest on this 'Twig'; nest level N2 must be a 'Leaf'.

Quickly reviewing what has been found so far :

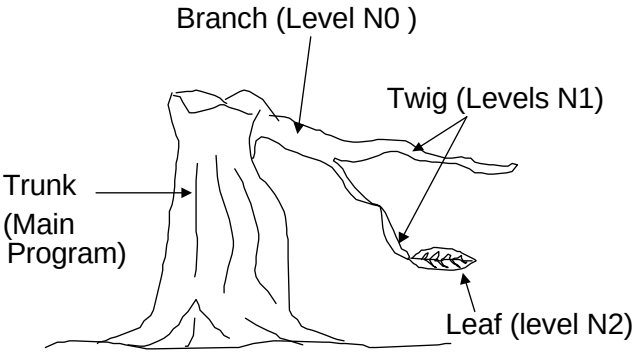
The main program is a 'Trunk',

Master control N0 is a 'Branch',

Master control N1 is a 'Twig'

and Master control N2 is a 'Leaf'.

This sounds rather like a tree! Drawing this out in a graphical format we can see;

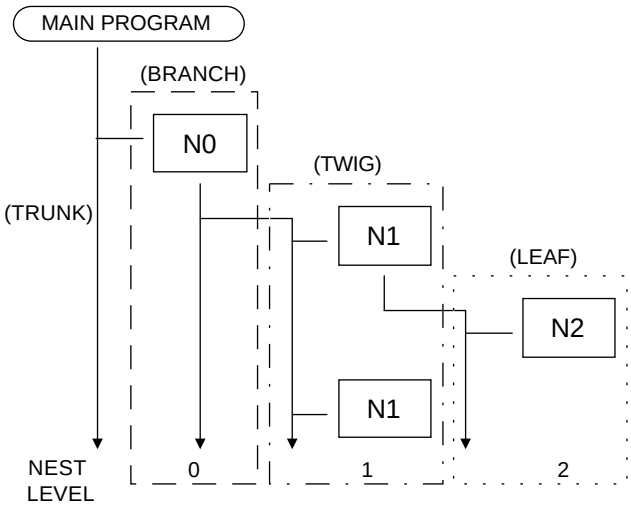


As with any tree for the smaller parts to work , the larger part must be functioning. This is exactly the same idea being used here, with the master control. A further similarity between the 'Tree' and programming with master controls is that when the nest level numbers are used ,branching can be traced back to any previous point.

This is shown through the use of the MCR N1 to return the program from MC N2 (the 'Leaf") back to MC N0 (the 'Branch'). It should be noted that the MCR instruction resets nest level N1, this is correct. Once nest level N1 M130 is reset the next active nest is the 'Branch', nest level N0. One further point of interest associated with the MCR instruction resetting nest level N1 is that the 'Leaf' element nest N2 is also reset. Why? Using the first tree analogy, if a branch is cut the twigs and leaves die. Consequently if the 'Trunk' is cut the branches die.

- Continuing the analysis of example program 2's structure it can be seen that there is now a second master control with a nest level of N1. Logically, the reasoning used to position the first occurrence of nest level N1 will apply here. Hence this new nest level N1 M200 is also a 'Twig'. It may be suggested that this new nest level N1 should also be reset when the MCR N1 is performed. This is not the case because the reset takes place before the new N1 M200 branch was created. In fact it was because of the MCR N1 that the new N1 branch was created at the same nest level.
- Finally the 'Branch' is cut when master control reset resets nest level N0. Hence all 'Branches', 'Twigs' and 'Leaves' that stem from nest level N0 are also reset.

The explanation of the program structure of example 2 has been made in the context of a tree. This can be treated as harmless fun but actually the terms and concepts mentioned here are correct and valid. It is admitted that the diagram would not look like a tree, but probably would be as shown below;



AN INTRODUCTION TO THE MASTER CONTROL INSTRUCTIONS (MC, MCR)

What Actually Happens in Example Program 2?

This example is quite complex. The following notes however, will give a guide to the operation of the program;

- Inputs X000 and X012 will always result in outputs Y000 and Y012 coming on.
- Input X001 will operate master control N0. This then allows inputs X002 to drive output Y002 and X007 to drive Y007. All previous conditions will be current but no further new outputs will be active (even if the inputs states' change).
- If inputs X003 and/or X010 are now made, each respective master control (MC N1 M130 or MC N1 M200) will be activated when its corresponding input is made. This would then allow inputs X004 and X011 to drive their respective outputs dependent on whether the original master control (MC N1 M130 or MC N1 M200) was active or not.
- Once inputs X001, X003 have been made, then input X005 can be made. This then allows the master control MC N2 M50 to be activated. Thus when input X006 is made the output Y006 will now be active.

It is important to remember that the working of this example program is subject to the operating conditions previously pointed out, i.e. if a master control is reset, any master controls that stem from the reset MC will also be reset. Standard programming techniques also apply, i.e. no dual coiling etc.

When using the MCR instruction, resetting to any nest level is available but as previously mentioned care should be taken not to reset an MC construction unwittingly. This flexibility allows the MC program constructions to be completely nested or split into two, or more, parallel program branches. When using the nesting levels be very careful to allocate the nest level number in the correct numerical order, i.e. 0, 1, 2, 3 etc. Care should be taken to ensure that the last MCR should always reset nest level N0. Finally please remember that there are only eight nest level numbers, N0 - 7, but these can be used multiple times if original bit flags are used on each occasion.

Further References

For a summary of basic programming rules please see sheet FX-SBI020.

Note 1:

The LD, OUT and END instructions mentioned in this document can be referenced on datasheets;

- FX-SBI001-FX-SBI003 which refer to the LD, LDI and OUT instructions,
- FX-SBI015 which is about the END instruction.

Note 2:

Internal flags are also known as;

- M coils
- Auxiliary relays (contacts or flags)
- Internal bits