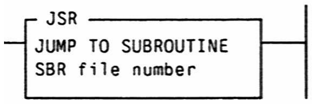
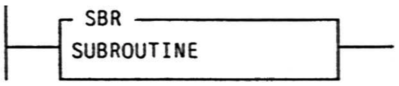
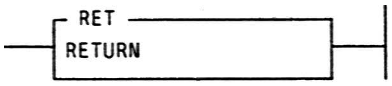
Introduction to Programmable Logic Controllers – Part II

Module 6: Control Instructions

Control instructions are used to enable or to disable a block of the ladder logic program or to move the execution of a ladder logic program from one place to another place. The control instructions discussed in this module include the following:

- Master Control Reset (MCR), instruction format: —(MCR)—
- Jump To Label (JMP), instruction format: —(JMP)—
- Label (LBL), instruction format: —]LBL[—
- Jump to Subroutine (JSR), instruction format: 

- Subroutine (SBR), instruction format: 

- Return from Subroutine (RET), instruction format: 

A Master Control Reset (MCR) instruction is an output instruction. MCR instructions are always used in pair. The paired instructions cause the PLC to enable or inhibit a zone of ladder logic program outputs according the application logic. The zone being controlled is known as the *MCR zone*. It begins with the rung that has the first MCR instruction. The MCR zone ends with the rung that has the second MCR instruction only. Control within the zone depends on the status of the rung with the first MCR instruction. When this rung is true, the MCR instruction is high and the outputs of the rungs within the controlled zone can be energized or de-energized by their rungs. In this case, these rungs operate as if the MCR zone does not exist. When the rung with the first MCR instruction is false, all the outputs in the zone are de-energized, regardless of the statuses of their rungs. The MCR zone's action can be compared to that of a circuit breaker. That is to say, when a circuit breaker is on, the electrical devices that receive power through the breaker can be turned on or off by their own control switches. However, when the circuit breaker is off, all the electrical devices under its control are disabled, irrespective of the individual control switches being on or off.

Figure 6.1 is a ladder logic program that uses MCR instructions to respond to an emergency stop button. The input terminal I:2/1 is connected to a normally-closed emergency button. When this button is released and the normally-open start button I:2/0 is momentarily pushed, the bit instruction B3:1/1 in the first rung is activated and latched. This causes the second rung, which has the first MCR instruction, to be true. The rungs located between the two MCR instructions operate as if the MCR instructions are not there. As soon as the emergency stop button is pushed, its contacts open. That turns the B3:1/1 instruction to low and the rung having the first MCR instruction becomes false. Now, all the outputs in between the two MCR instructions are de-energized, regardless of the statuses of their rungs. The MCR zone stays disabled until the emergency button is released, the start button is pushed again, and the rung with the first MCR instruction returns to true.

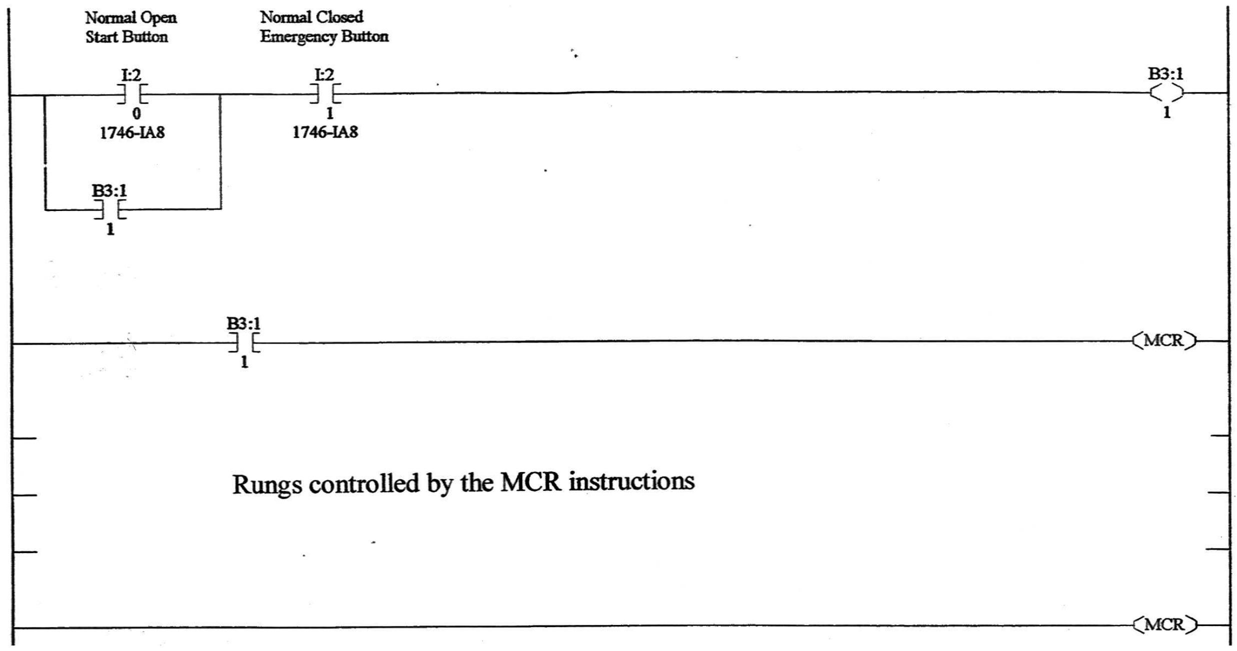


Figure 6.1. A program using Master Control Instructions

Normally, a PLC executes a ladder logic program in a rung by rung sequence. Jump instructions allow a PLC to break this sequence and to move the program execution to another rung or to a subroutine.

The Jump to Label instruction and the Label instruction are used in combination to redirect the execution of a ladder logic program. The Jump to Label instruction is a controlled instruction; when its rung is false, the PLC scans the next rung. When the rung of a Jump to Label instruction is true, the PLC breaks its sequence execution and moves to the rung with a Label instruction that has the same label number as the Jump to Label instruction. When a Label instruction is used, it is always the first (most left) instruction in a rung. It does not cause its rung to be true or false; it only shows the location of the label.

Figure 6.2 is a ladder logic program to count parts and their values. Input I:2/1 is a sensor detecting part A, which is worth $10. Input I:2/2 is a sensor detecting part B, which is worth $20. Counter C5:1 counts the number of part A while memory location B3:10 (a word) keeps the total value associated with part A. Counter C5:2 counts the number of part B while memory location B3:11 keeps the total value associated with part B. When a piece of part A or part B comes down a conveyer, a corresponding sensor sends a high signal to the PLC and the corresponding counter and memory word are updated. When a piece of part A is detected by the sensor, the input I:2/1 in the program of Figure 6.2 causes the Jump to Label instruction in the rung 0000 to be activated. Instead of executing the next rung, the PLC moves to rung 0003 that has a Label instruction with number Q2:1. The PLC executes rung 0003 and 0004. The Jump to Label instruction in rung 0004 causes the PLC to move back to rung 0001. The relations of input statuses and program execution sequences in a scan cycle are listed in Table 6.1. Using jump instructions may eliminate the unnecessary rung execution and reduce the time of a scan cycle.

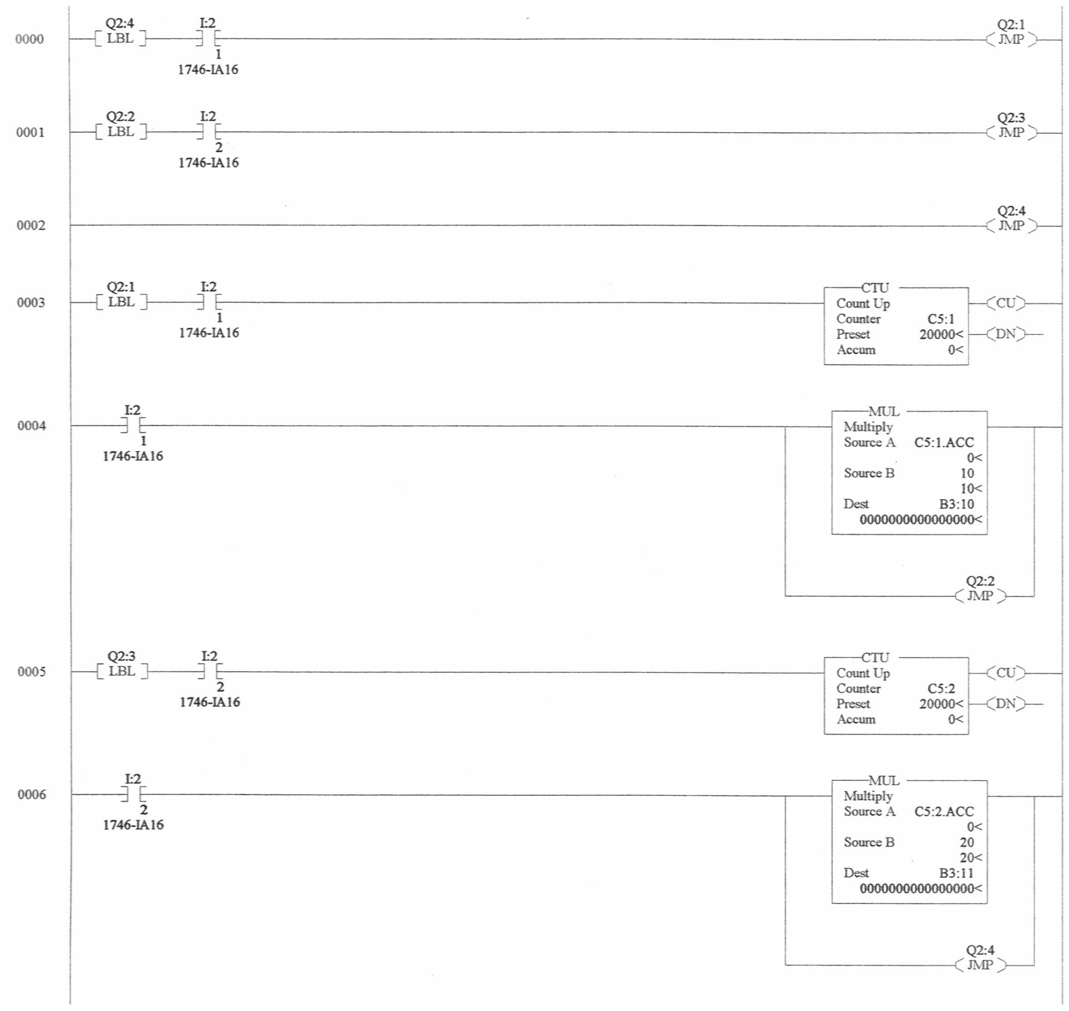

0000
Q2:4 LBL
I:2 1 1746-IA16
Q2:1 JMP
0001
Q2:2 LBL
I:2 2 1746-IA16
Q2:3 JMP
0002
Q2:4 JMP
0003
Q2:1 LBL
I:2 1 1746-IA16
CTU Count Up Counter C5:1 Preset 20000< Accum 0<
CU
DN
0004
I:2 1 1746-IA16
MUL Multiply Source A C5:1.ACC 0< Source B 10 10< Dest B3:10 0000000000000000<
Q2:2 JMP
0005
Q2:3 LBL
I:2 2 1746-IA16
CTU Count Up Counter C5:2 Preset 20000< Accum 0<
CU
DN
0006
I:2 2 1746-IA16
MUL Multiply Source A C5:2.ACC 0< Source B 20 20< Dest B3:11 0000000000000000<
Q2:4 JMP

Figure 6.2 A ladder logic program using Jump to Label and Label instructions

Input Status	Program scan sequence in rung number
I:2/1 = Low and I:2/2 = Low	0000, 0001, 0002, 0000
I:2/1 = Low and I:2/2 = High	0000, 0001, 0005, 0006, 0000
I:2/1 = High and I:2/2 = Low	0000, 0003, 0004, 0001, 0002, 0000
I:2/1 = High and I:2/2 = High	0000, 0003, 0004, 0001, 0005, 0006, 0000

Table 6.1 Ladder logic program execution sequence

A Jump to Subroutine instruction, which is a controlled instruction, must have a corresponding subroutine program file. When the rung of a Jump to Subroutine instruction is false, the PLC does not jump to the subroutine; the next rung scanned will be the one following the Jump to Subroutine instruction. However, when the rung of a Jump to Subroutine instruction is true, the PLC jumps to the Subroutine instruction at the beginning of the target subroutine file and resumes execution at that point. Movement is always to the first instruction of the subroutine file. A Jump to Subroutine instruction can be used in both a main application program and subroutine programs to “call” (execute) another subroutine program.

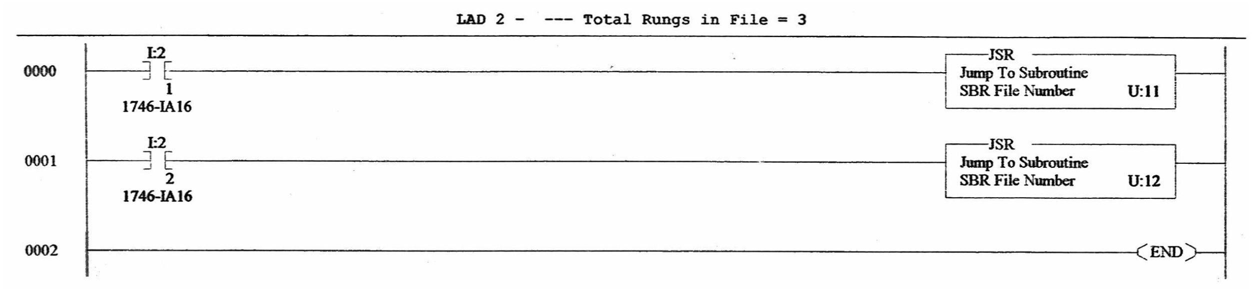
A subroutine is a separate program file. For Allen-Bradley PLCs, File 0 and File 1 are system files. File 2 is the default main application program file. Whenever a new main application program is created, the program is automatically labeled as File 2. However, when a subroutine program file is created, a unique file number from 3 to 255 may be chosen as the subroutine file number. Because each subroutine must have its own file number, an application program can have as many as 253 subroutines.

The first instruction in the first rung of a subroutine file must be a Subroutine instruction. The sole purpose of this instruction is to indicate the subroutine file’s beginning. It does not cause its rung to be true or false.

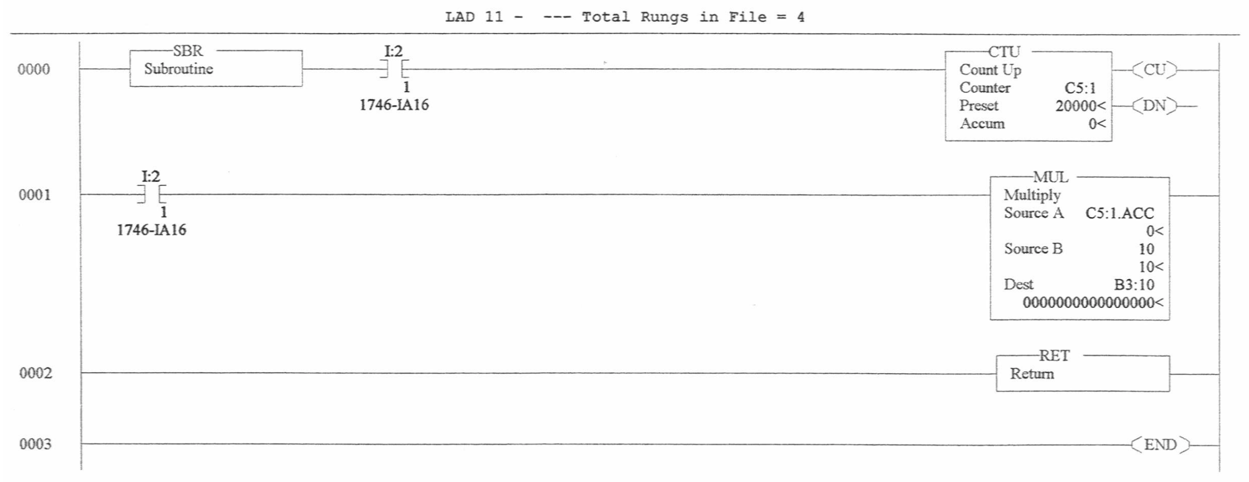
A Return from Subroutine instruction marks the end of subroutine execution or the end of the subroutine file. It causes the PLC to resume execution in the calling program file at the rung following the Jump to Subroutine instruction where the subroutine was called.

A Return from Subroutine instruction can be programmed (placed) before the last rung of a subroutine program. In this case, when the rung of this instruction is true, the PLC ignores the remainder of the rungs in the subroutine and returns to the calling program file. When a Return from Subroutine instruction is not included in a subroutine file, the END statement, which always comes with a program file (main or subroutine) as the last rung in the program, causes the PLC to move the execution back to the rung following the Jump to Subroutine instruction in the calling program file.

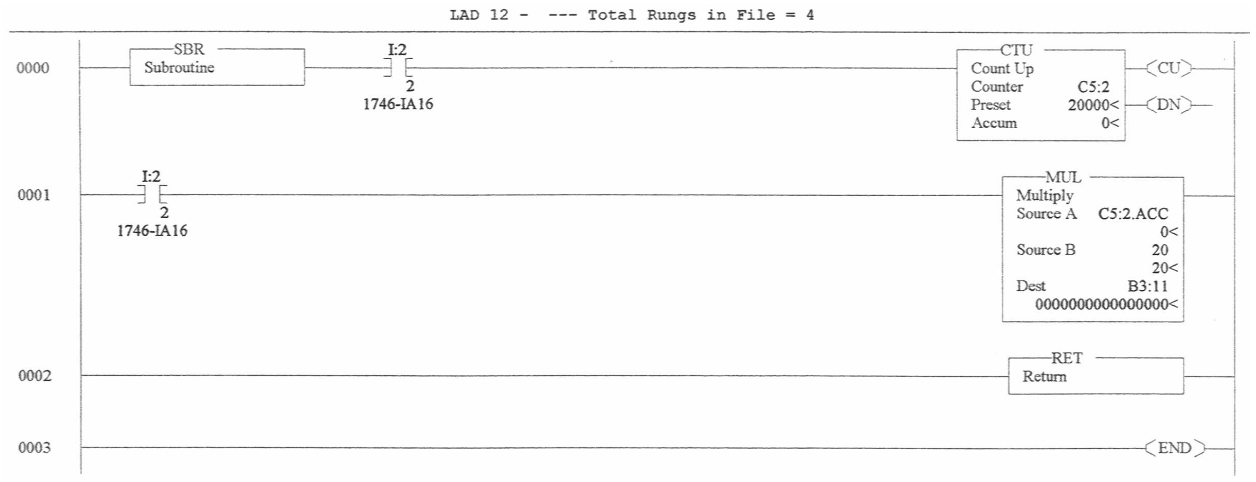
The ladder logic program in Figure 6.2 can be replaced by a program with subroutines. Figure 6.3 shows the new main program and subroutines. Not counting the rung for the END statement, the main program has only two rungs. When the input I:2/1 is high, subroutine 11 is called. The execution of subroutine 11 causes the number and value of part A to be updated. Then the PLC returns to the second rung in the main program. When the input I:2/2 is high, subroutine 12 is called. The execution of subroutine 12 causes the number and value of part B to be updated. Then the PLC returns to the third rung in the main program, which is the end of the program. When the input I:2/1 or I:2/2 is low, or both of them are low, the corresponding subroutine is not called. The PLC executes the next rung in the main program.



(a) Main program



(b) First subroutine



(c) Second subroutine

Figure 6.3 A ladder logic program using subroutines

Related web site:

http://www.plcs.net