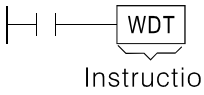
5 General Rules for Applied Instructions

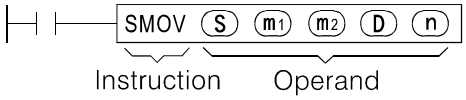
5-1 Formats of Applied Instructions

Instruction and Operand

- Each applied instruction has its unique instruction mnemonic, e.g. ADD, CMP…, Etc.
- Some applied instructions are made up by themselves:



- Most of the applied instructions are constituted by themselves and some "Operands":



As shown above Ⓢ, ⓜ1, ⓜ2, Ⓓ, ⓝ are Operands. There are many types of Operand in applied instructions, their symbolic meanings are:

Ⓢ :Source Operand (device). It usually refers to the Operand with unchanged contents after executed. Ⓢ1, Ⓢ2, … represent multiple source Operands for an instruction.

Ⓓ :Destination Operand (device). It usually refers to the Operand in which instruction execution outcomes are stored. Ⓓ1, Ⓓ2, … represent multiple destination Operands for an instruction.

ⓜ, ⓝ : Those Operands used to specify operational constants. But some ⓜ, ⓝ of instruction can use Register D to execute indirect specification. ⓜ1, ⓜ2, ⓝ1, ⓝ2… Represent multiple ⓜ, ⓝ.

Devices for Operand

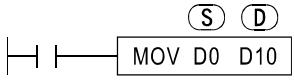
- Based on the needs, each applied instruction owns different number of Operands. And each applied instruction has different device ranges. The ranges of each Operand device are shown as in the following table:

Operand	Devices															
	X	Y	M	S	K*n*X	K*n*Y	K*n*M	K*n*S	T	C	D	SD	P	V,Z	K,H	VZ index
S1					○	○	○	○	○	○	○	○		○	○	○
S2					○	○	○	○	○	○	○	○		○	○	○
D						○	○	○	○	○	○	○		○		○

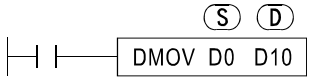
- The "M" in the table above does not include Special Coil M9000 ~ M9255.
- The "D" in the table above does not include Special Register D9000 ~ D9255, and "SD" is specially pointed to D9000 ~ D9255.
- The "VZ index" in the table above indicates whether the Operand can be modified by Index Register V, Z.
- Under the applied instructions, if V, Z or SD is specified as the Operand Device, using V or Z for modification is prohibited.
- After organized, bit devices are displayed as K*n*X, K*n*Y, K*n*M, K*n*S to store data.
- T, C in the table above refer the current value registers of Timer (T) and Counter (C).
- All of T0 ~ T255, C0 ~ C199 and D are 16-bit registers. When the instruction specifies the process of 32-bit data, continuous two 16-bit registers will be occupied. For example, if a 32-bit Operand instruction specifies to D100, then a 32-bit register (composed of D101 and D100) will be used. while D101 will assigned for higher 16 bits and D100 for lower 16 bits. The same rules are also plied to T and C.
- 32-bit Counters (C200 ~ C255) only can be used as Operands of 32-bit instructions.

16-bit and 32-bit Instructions

- Because of different Operand value sizes, some of the applied instructions can be classified into 16-bit instructions and 32-bit instructions.



A 16-bit instruction, the content of D0 is transferred to D10.

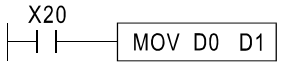


A 32-bit instruction, the contents of (D1, D0) are transferred to (D11, D10).

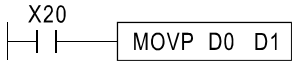
- A 32-bit instruction is displayed with a "D" (to be added directly BEFORE the instruction mnemonic), e.g. MOV represents a 16-bit instruction, while DMOV represents a 32-bit instruction.
- The device ID. number specified by an Operant of a 32-bit instruction can be an even or odd number. In order not to get confusion, it is recommended to use an even number, if it is possible.
- 32-bit Counters, C200 ~ C255, only can be used as Operands of 32-bit instructions.

Pulse Execution Instructions

- Based on requires, some applied instruction can be classified into sequential execution instructions and pulse execution instructions.



Sequential execution instruction: When X20= "ON", the instruction will be executed once in each scan cycle.



Pulse execution instruction: The instruction is only executed once when X20="OFF" → "ON".

- A pulse instruction displayed with a "P" (to be added directly AFTER the instruction mnemonic), e.g. MOV represents a sequential execution instruction, while MOVP represents a pulse execution instruction.
- Suitable using pulse execution instructions to replace sequential execution instructions in a program, can cut down unnecessary execution time.
- When X20= "OFF", both MOV and MOVP instuctions are not executed.

5-2 Data Process of Applied Instructions

- The X, Y, M and S are called bit devices, because they have only two different status ("ON"or "OFF"). But the T, C and D are called word devices because they are specially used to store data. Some bit devices can be a group together as a word device pattern, shown in the form of K*n*X, K*n*Y, K*n*M and K*n*S. This organized bits become a word device, that can be used in applied instructions for storage of data.

- When bit devices are organized as a word device, each digit of a hexadecimal word is composed by 4 bit devices. The K*n* portion of the statement identifies the range of devices included. The "n" can be a number from the range 1 to 8 and it actual represents 4**n* bit devices (*n* digits hexadecimal word). Hence all groups of bit devices are divisible by 4.

  K1M0 refers to a one-digit of hexadecimal word device, that is composed of M0 ~ M3.

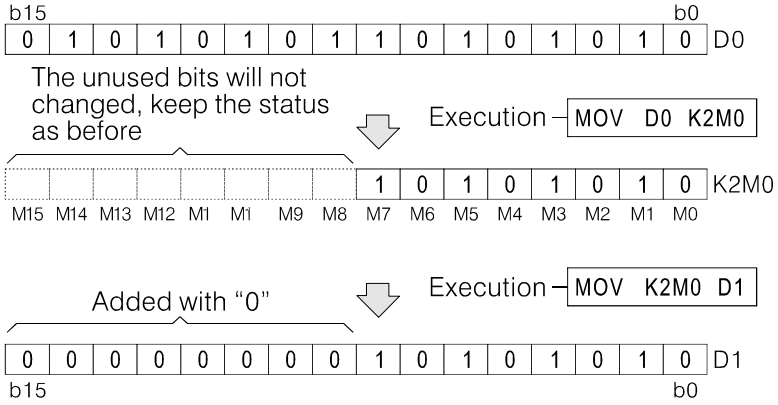
  K2M0 refers to a two-digit of hexadecimal word device, that is composed of M0 ~ M7.

  K4M0 refers to a four-digit of hexadecimal word device, that is composed of M0 ~ M15.

  K5M0 refers to a five-digit of hexadecimal word device, that is composed of M0 ~ M19.

  K8M0 refers to an eight-digit of hexadecimal word device, that is composed of M0 ~ M31.

- Data transference between registers and word devices which are composed of bit devices, the change should study up by the example below.



- When bit devices are organized as a word device, the header ID number of bit device can be specified as any legally device. But itis recommended that K*n*X and K*n*Y specify the ID number started with "0" such as X0, X20, Y20, Y30..., while K*n*M and K*n*S specify the ID number which is multiple of "8", such as M0, M8, M16.... The recommendations can improve system efficiency.

- When the Operand of an applied instruction is transformed to few sequential devices, the sequential ID number at different types are referred as below:

  ① Word Device (16 bits)

  D0, D1, D2, D3·····
  T0, T1, T2 ·····
  C0, C1, C2 ·····

  ② Double-word Device (32 bits)

  D0 (D1, D0), D2 (D3, D2), D4 (D5, D4)·····
  T0 (T1, T0), T2 (T3, T2), T4 (T5, T4)·····
  C200, C201, C202 ·····

  ③ Word Device Composed of Bit Devices

  K1X20, K1X24, K1X30, K1X34 ·····
  K2Y20, K2Y30, K2Y40, K2Y50 ·····
  K3M0, K3M12, K3M24, K3M36 ·····
  K4S0, K4S16, K4S32, K4S48 ·····

5-3 Using Index Register V and Z to Modify Operands

Index Register

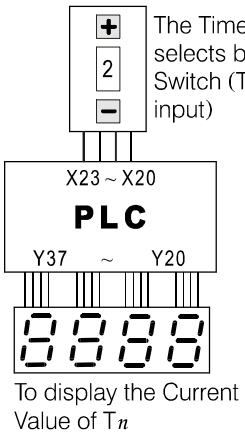
16-bit	16-bit
V0	Z0
V1	Z1
V2	Z2
V3	Z3
V4	Z4
V5	Z5
V6	Z6
V7	Z7
Higher 16-bit	Lower 16-bit
32-bit	

- There are 16 of 16-bit Index Registers, V0 ~ V7 and Z0 ~ Z7, in M, VB and VH Series PLC.
- When using Index Registers V and Z in a 32-bit applied instruction, it must specifies an Index Register Z and the relative Index Register V will be taken over. For example, specifying Z0 will use two Index Registers(V0, Z0), the V0 is for higher16 bits and Z0 for lower 16 bits.
- The devices which can be modified by Index Register V, Z are shown below:

  X, Y, M, S, P, T, C, D, K, H, K*n*X, K*n*Y, K*n*M, K*n*S
- The following cannot be modified by V, Z:
  ① V, Z (themselves)
  ② SD (D9000 ~ D9255)
  ③ The *n* of K*n*
  ④ Used for Jump Destination or Subprogram Pointer P

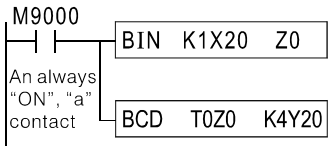
- The followings are examples for operand modified by V and Z.
  ① For a 16-bit instruction, when Z0=3
  Y20Z0=Y23
  T5Z0=T8
  D0Z0=D3
  K1M10Z0=K1M13
  ② For a 32-bit instruction (Index Registers V and Z will be taken over), when (V1, Z1)=8
  Y20Z1=Y30
  D0Z1=D8
  K8M40Z1=K8M48

- Example of use of Index Register
  Under the program and external inputs below, using Z0 to modify T0, and easily display the current value of T0 ~ T9 on the external 7-Segment Displayer.



The Timer ID (*n*) selects by a Digital Switch (Thumbwheel input)

To display the Current Value of T*n*



To convert the Digital Switch (Thumbwheel input) into BIN value thruth X20 ~ X23 and transferred to Z0 as the selected Timer ID; Z0=0 ~ 9.

To convert the current value of T0Z0 (one of T0 ~ T9) into BCD format value and transfer the value into Y20 ~ Y37, which will displayed on the 7-Segment Displayer.

5-4 The Special Notes for Using Applied Instructions

Flags

- The execution results are relate to Applied Instructions and causes some changes to corresponding flags:

  M9020: Zero Flag
  M9021: Borrow Flag
  M9022: Carry Flag
  M9029: Instruction Execution Completed Flag

- The execution results are relate to Applied Instructions and causes some changes to corresponding flags:

Limits on Using Applied Instructions

- Some of the Applied Instructions can only appear once in the program, the list showing below are those instructions.

  MTR (FNC52) PWM (FNC58)
  SORT (FNC69) HKY (FNC71) DSW (FNC72) SEGL (FNC74)
  PR (FNC77) LINK (FNC89) MBUS(FNC149)

  Using Index Registers to modify the instructions in operands, which will perform a better effect for the above-mentioned instructions.

- Some of the applied instructions can be used as many times as required, but the instruction executed at the same moment are limited the number of times.

  ① The instructions DHSCS, DHSRC and DHSZ executed in the program at same time, the number of times at most will be 6 in total.

  ② Only one RS instruction can be executed at the same time in the program.

Floating Point Instructions

- The list of relative Applied Instructions for processing floating point values.

  FLT (FNC49) DECMP (FNC110) DEZCP (FNC111) DEBCD(FNC118)
  DEBIN (FNC119) DEADD (FNC120) DESUB (FNC121) DEMUL(FNC122)
  DEDIV (FNC123) DESQR (FNC127) INT (FNC129) DSIN(FNC130)
  DCOS (FNC131) DTAN (FNC132)

- Every floating point number will occupys two registers.
- The format of floating point number store in registers, please refer to Section 2-12 “Numerical System”.
- If the source operands of floating point operation instructions are assigned to constant numbers K or H, the instructions will let the constant numbers transform to a BIN floating point numbers for the processing.
- When using the floating point operation functions, please pay attention to the format of operands.

MEMO