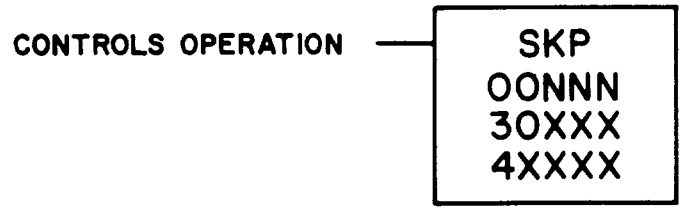
SECTION 5
ADDITIONAL FUNCTIONS

This section describes the additional functions that can be used in programming the 584L Programmable Controller.

5.1 SKIP (SKP)

FUNCTION
The SKIP function allows logic in a group of networks to be skipped, and thus not solved in order to reduce scan time. The SKIP function can be used to bypass seldom used program sequences or to create subroutines. See Section 5.5.1 on Subroutines.



FUNCTION BLOCK
This function block only occupies one node in a network. It contains the symbol SKP and either a constant or a register reference. It can be a constant, up to 999 in a Level 1 or 584L PC or up to 9999 in a Level 2 584L PC, a 30XXX input register reference, or a 4XXXX holding register reference. If a 4XXXX reference is used it should be unique to this function block to avoid mishaps (i.e., using the same register to hold a counter value, etc.). The value specifies the number of networks to be skipped. To skip the remainder of the networks in the current segment, a value of zero is entered into the function block.

NOTES

If a 30XXX input register is used and the input is coming from a thumbwheel, the data read by the controller can be incorrect (i.e., if it was read while the number was still being entered). In order to use a thumbwheel and ensure a correct number, use a 4XXXX reference in the SKIP block and use a Subtract block to subtract zero from the input register to load it into the holding register.

The SKIP function cannot pass the boundary of a segment. Regardless of how many networks were programmed to be skipped, the function stops when it reaches the end of a segment. Also, skips within skips cause the controller to shut down; this should be avoided.

INPUT
The top and only input, when it receives power, causes the remainder of the current network and the specified number of networks to be skipped over by the controller's scan.

OUTPUT
There is no output with this function.

NOTE

Power flow shown on the P190 screen for skipped networks is invalid. To prevent misinterpretation of the data, the message "POWER FLOW INVALID" appears on the P190 screen.

EXAMPLE
The following paragraph provides a detailed explanation of the logic used in Figure 5-1.

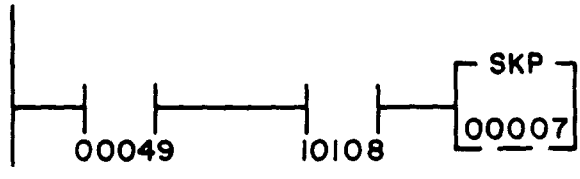


Figure 5-1. Skip

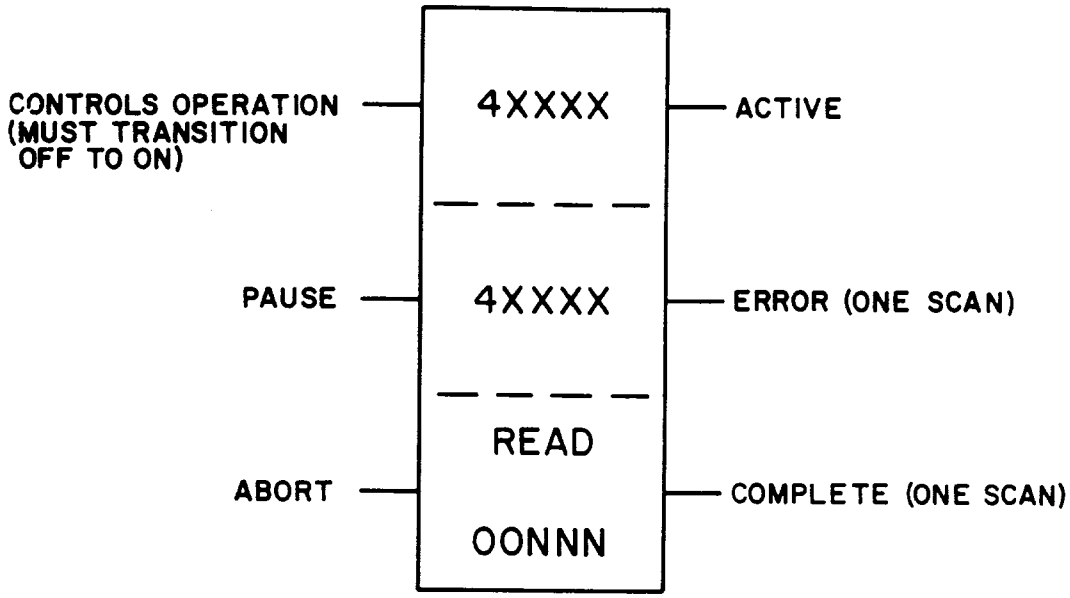
When the SKIP function block's input receives power, the remainder of the network containing the block (if any) is skipped and the next six networks are skipped. If network 17 contains the SKIP function, the remainder of 17 and all of 18-24 are skipped. If there are only five networks left in the segment, the operation stops after the fifth network.

NOTE

A skip node in any location of the network, except the bottom rung, can cause the 584L to stop.

5.2 READ (READ)

FUNCTION
The Read function allows the 584L PC to read data from an ASCII device through a P453 and an RS-232-C port on the 584L PC. The data is stored in a table of registers.



FUNCTION BLOCK

- The top node is the source node. It is a 4XXXX holding register reference. It refers to a table of seven registers. The table starts with the reference in the top node; the next six registers are implied. These registers must be unique to this function block.

  The following are register assignments:

  4XXXX = port number and error codes
  4XXXX + 1 = message number
  4XXXX + 2 = status word
  4XXXX + 3 = number of registers required
  4XXXX + 4 = number of registers received
  4XXXX + 5 = number of registers needed
  4XXXX + 6 = checksum

- The middle node is the destination node. It is a 4XXXX holding register reference. It is the reference to the first register in a table of registers. The information read by the controller is stored in these registers.

- The bottom node contains the symbol READ and the numerical value that specifies the destination table length. This constant can range from 1 to 255.

INPUTS

- The top input controls the operation. When this input transitions from OFF to ON the READ function is performed.

- The middle input, when receiving power, stops the READ function. When this input loses power, the READ function resumes where it left off.

- The bottom input, when it receives power, aborts/stops the operation. The top input must be cycled for the function to begin again; it does not resume where it left off.

OUTPUTS

- The top output, when passing power, indicates that the READ function is communicating with the specified port (FUNCTION ACTIVE).

- The middle output passes power for one scan when an error is detected. The error code is placed in either the 4 most significant bits of the source's first register or the next six bits (first four — 584L error; next six — P453 Remote I/O Interface error). For a list of the error codes, see the ASCII Programming Guide.

- The bottom output passes power for one scan when the operation is complete.

EXAMPLE

The following paragraph provides a detailed explanation of the logic used in Figure 5-2.

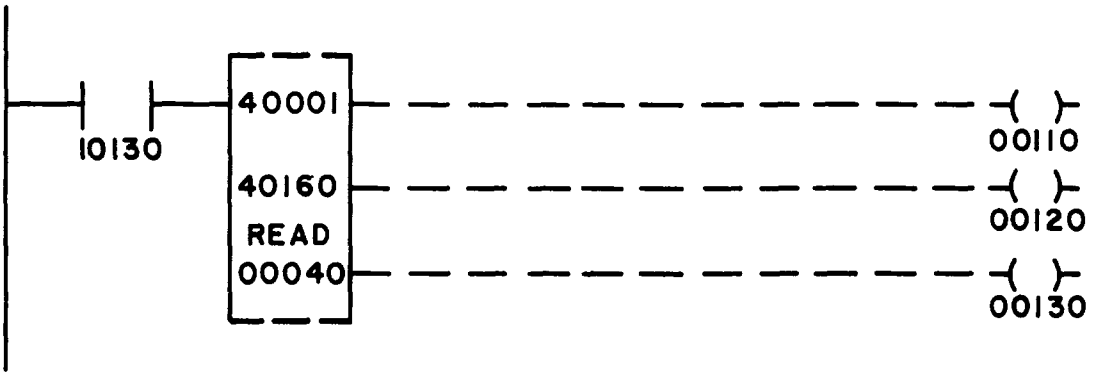


Figure 5-2. Read

When input 10130 transitions from OFF to ON, the top input receives power and starts to read information from an ASCII device. The information is stored in registers 40160 to 40199. The top output passes power and energizes coil 00110 while the read is taking place. If an error is found, the middle output passes power for one scan, energizing coil 00120 for one scan. When the read is complete, the bottom output passes power and energizes coil 00130.

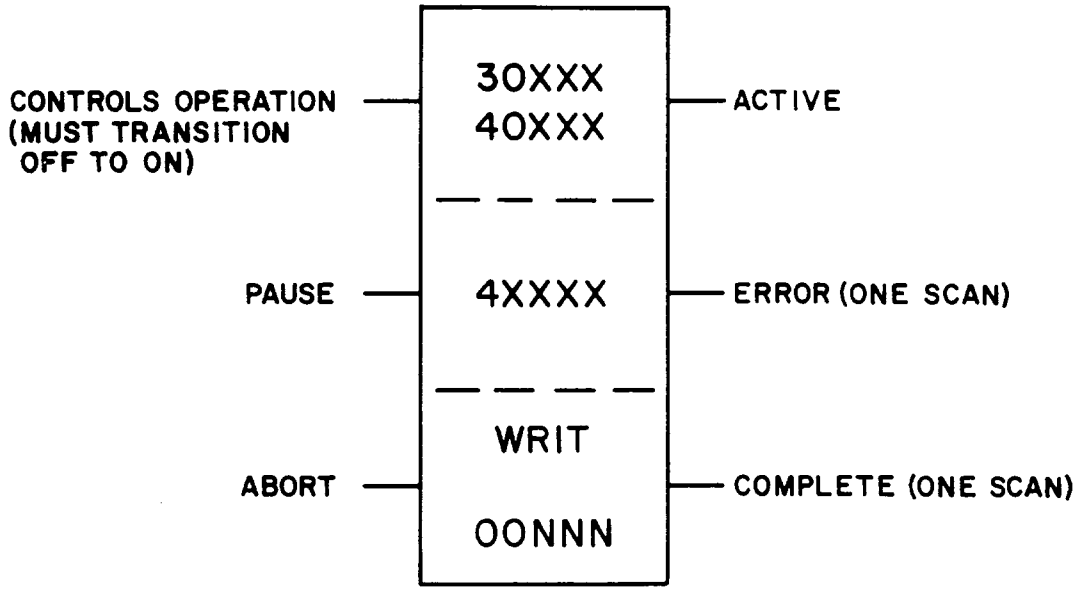
NOTE

An error code goes into the status word for a carriage return on the read block.

5.3 WRITE (WRIT)

FUNCTION

The Write function transmits data from the 584L PC through an RS-232-C port and a P453 to an ASCII device.



FUNCTION BLOCK

- The top node is the source node. It can be either a 30XXX input register or a 4XXXX holding register reference. It is the reference to the first register in a table of registers.

- The middle node is the destination node. It is a 4XXXX holding register reference. It refers to a table of seven registers. The table starts with the reference in the top node; the next six registers are implied. These registers must be unique to this function block.

  The following are register assignments:

  4XXXX = port number and error codes
  4XXXX + 1 = message number
  4XXXX + 2 = status word
  4XXXX + 3 = number of registers required
  4XXXX + 4 = number of registers received
  4XXXX + 5 = number of registers needed
  4XXXX + 6 = checksum

- The bottom node contains the symbol WRIT and the numerical value that specifies the source table length. This constant can range from 1 to 255.

INPUTS

- The top input controls the operation. When this input transitions from OFF to ON the WRITE function is performed.

- The middle input, when receiving power, stops the WRITE function. When this input loses power, the WRITE function resumes where it left off.

- The bottom input, when it receives power, aborts/stops the operation. The top input must be cycled for the function to begin again; it does not resume where it left off.

OUTPUTS

- The top output, when passing power indicates that the WRITE function is communicating with the specified port (function active).

- The middle output passes power for one scan when an error is detected. The error code is placed in either the four most significant bits of the source's first register or the next six bits (first four — 584L error; next six — P453 Remote I/O Interface error). For a list of the error codes, see the ASCII User's Guide.

- The bottom output passes power for one scan when the transfer of data from the 584L PC to the P453 is complete.

EXAMPLE

The following paragraph provides a detailed explanation of the logic used in Figure 5-3.

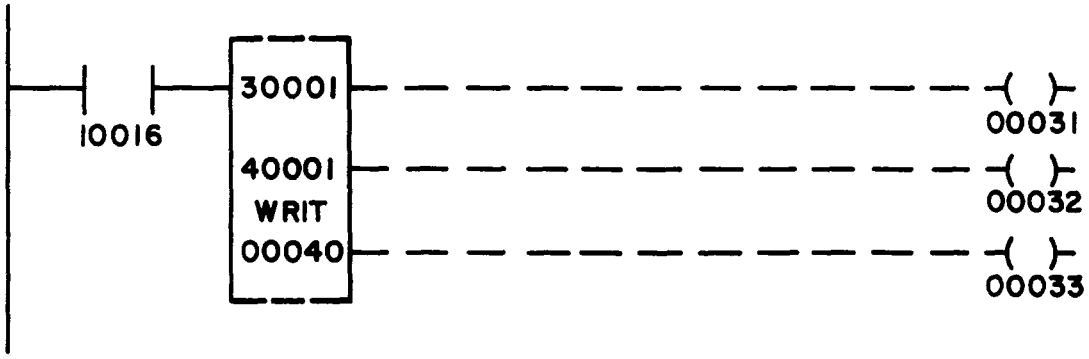


Figure 5-3. Write

When input 10016 transitions from OFF to ON, the top input receives power and starts to write information to an ASCII device. The information is taken from registers 30001 - 30040. The top output passes power and energizes coil 00031 while the write is taking place. If an error is found, the middle output passes power for one scan, energizing coil 00032. When the write is complete, the bottom output passes power and energizes coil 00033.

NOTES

1. A write to any "reserved" ASCII holding register will not set the error output.

2. A write to a busy ASCII port does not set the busy flag.

3. Write Block issues a done flag when the ASCII message is "sent" to the P453, instead of when it is actually "sent".

5.4 SWEEP FUNCTIONS

The SWEEP FUNCTIONS in the 584L PC allow the user's program to be scanned at a fixed interval. This interval can be a fixed time between scans or a fixed number of scans. These functions do not allow the controller to solve logic faster or to end a scan prematurely.

The CONSTANT SWEEP function allows the user to set target scan times from 10 to 200 milliseconds in multiples of 10. Target scan time is the time between the start of one scan and the start of the next scan, not the time between the end of one scan and the start of the next scan. This function is useful in applications in which data must be sampled at constant time intervals. The constant sweep value is recorded when a dump tape is made and can be transferred to another controller in this way.

The SINGLE SWEEP function allows the controller to execute a fixed number of scans, ranging from I to I5, and then to STOP solving logic but continue to service I/O. This function is useful to debug programs in which logic is complicated, or multiple scan logic is used.

When the SWEEP FUNCTIONS software label key is pressed, the following software labels appear on the screen:

INVOKE CONSTANT	CANCEL CONSTANT	INVOKE SINGLE	TRIGGER SINGLE	CANCEL SINGLE

5.4.1 Constant Sweep

The CONSTANT SWEEP function allows target scan times of 10 to 200 milliseconds. Target scan time is the time between the start of one scan and the start of the next scan, not the time between the end of one scan and the start of the next scan. The target scan time can be in multiples of 10 milliseconds and is not restricted to any two scans.

If a constant sweep is invoked with a time lapse smaller than the actual scan time, the time lapse is ignored and the scan is completed before the next scan starts. The constant sweep specifies the time between starts, not the time between the end of one scan and the start of another scan. (For example, if a sweep specifies 200 milliseconds as a desired scan time and the actual scan time is 90 milliseconds, the logic is solved in 90 milliseconds then the controller waits 110 milliseconds before starting the next scan.) To invoke a constant sweep:

1. Press the INVOKE CONSTANT software label key.

Four software labels appear on the screen: COMMENCE, TIME 10-200, HLDG REG 4XXXX, and PREVIOUS MENU.

2. Enter the desired scan time in milliseconds (10-200 in multiples of 10) into the AR.

3. Press the TIME 10-200 software label key to enter the value.

4. Select a holding register (4XXXX reference) and enter the reference number into the AR.

5. Press the HLDG REG 4XXXX software label key.

6. Press the COMMENCE software label key to start the CONSTANT SWEEP.

The constant scan time desired is placed in the previously specified holding register. The actual scan time is placed in the next consecutive holding register. To stop the constant sweep, press the PREVIOUS MENU software label key then the CANCEL CONSTANT software label key. A message appears on the P190 screen indicating that the sweep is cancelled.

NOTES

1. The constant sweep target scan time includes logic solving, I/O servicing, and system diagnostics. It does not include Modbus service time. If a target scan time of 40 milliseconds is set, and logic solving, I/O servicing, and systems diagnostics take only 30 milliseconds, the 584L PC waits 10 milliseconds. The 584L PC checks to see if either or both Modbus ports require service. If the ports require service, an additional 3 milliseconds is automatically added by the controller, to the 40 millisecond target scan time.

2. The constant sweep function should not be used if redundancy (J2II/584L) is active.

5.4.2 Single Sweep

The SINGLE SWEEP function allows the controller to complete between 1 and 15 scans and then to stop scanning. This is very useful for diagnostic work; it allows solved logic, moved data, or performed calculations to be examined for errors.

To invoke a single sweep:

1. Press the INVOKE SINGLE software label key.

Three software labels appear on the screen: COMMENCE, TIME 10-200, and PREVIOUS MENU.

2. Enter the scan time in milliseconds (10-200 in multiples of 10) into the AR.

3. Press the TIME 10-200 software label key to enter the value.

4. Press the COMMENCE software label key.

5. Press the PREVIOUS MENU software label key.

6. Press the TRIGGER SINGLE software label key.

Three software labels appear on the screen: COMMENCE, #SWEEPS 1-15, and PREVIOUS MENU.

7. Enter the number of scans desired (1-15) into the AR.

8. Press the #SWEEPS 1-15 software label key.

9. Press the COMMENCE software label key.

The controller will scan the specified number of times then stop.

To continue single sweeps:

1. Press the TRIGGER SINGLE software label key.
2. Enter the number of sweeps into the AR.
3. Press the #SWEEPS 1-15 software label key.
4. Press the COMMENCE software label key.

To return to normal scanning, press the PREVIOUS MENU software label key then the CANCEL SINGLE software label key.

CAUTION

The SINGLE SWEEP function should not be used to debug controls on machine tools, processes, or material handling systems when they are active. Once the specified number of scans has been solved, all outputs are frozen in their last state. Since no logic solving is taking place, all input information is ignored. This can result in unsafe, hazardous, and destructive operation of the machine or process connected to the 584L PC.

5.5 SUBROUTINE EXAMPLE

The SKIP function can be used to create subroutines within a program. This is useful when a routine needs to be performed every third, or fourth scan, etc. It is also useful to perform a certain routine several times within a function. The SKIP functions allow the routine's logic to be written once and performed as many times as necessary.

The following example uses the SKIP function to allow a different routine to take place each scan for a limited sequence.

On three consecutive scans, three different functions can be performed, one function each scan.

Network 3 contains a skip block with a holding register, not a constant,

Network 5 contains a function, the first of three,

Network 6 contains a function, the second of three,

and Network 7 contains a function, the third of three.

The first network (i.e., 10) in the next segment contains an ADD block which adds a one to the holding register (i.e., 40001) used in network 3's skip block.

On the first scan, at network 3 the controller skips to the beginning of the next segment because 40001 contains zero. In network 10, a one is placed in 40001. On the next scan at network 3, the controller skips to network 5. The first function is

performed. At the end of network 5 is a skip block containing zero. The controller skips to the beginning of the next segment, and adds a one to 40001, increasing it to two.

On the next scan at network 3, the controller skips to network 6. The second function is performed. At the end of network 6 is a skip block containing zero. The controller skips to the beginning of the next segment and adds a one to register 40001, increasing it to three.

On the next scan at network 3, the controller skips to network 7. The third function is performed. At the end of network 7 is an ADD block which places a zero in register 40001. The controller solves the rest of the logic in the network. At network 10 a one is placed in register 40001.

On the next scan, the process starts all over again. The only difference is that the controller Skips to network 3 on this scan. It does not have to Skip to the beginning of the next segment first because there is already a one in register 40001.