Chapter 14 Application of ASCII File Output Function

The FBs-PLC's ASCII file output function allows the PLC to directly drive ASCII output devices such as printers and terminals, and let them print or display English document data or display screens such as production reports, materials details and warning messages. For application of the ASCII file output function, it is necessary to edit, the ASCII file data to be output must be edited to fit the required format of the FBs-PLC FUN 94 (ASCWR) instruction. Then using this instruction, it will be sent out via port 1 to the ASCII output device connected with port 1.

14.1 Format of ASCII File Data

ASCII file data may be divided into fixed, unchanging background file data and dynamically changing variable data. The background file data may be in English characters, numerals, symbols, graphs, etc, and the variable data can only be printed out as binary, decimal, or hexadecimal numeric value data.

ASCII code is a byte length code, which has a total of 256 combinations. Of these, the first 128 (0-127) are fairly clearly defined and are used by most of the ASCII peripherals. For codes greater than 128 each manufacturer has different definitions and graphics and there are no uniform specifications. FBs-PLC designed the FUN 94 (ASCWR) instruction to be solely responsible for transmission, and not for editing. This work is done by the ASCII editor of the WinProladder software package. Below is the editing command format adopted by the WinProladder software package editor.

1. Basic command Symbols

- ／ Linefeed

A line slanting down from right to left means that no matter where the printing is up to, if this symbol is encountered, then the printing head or the terminal display will move to the beginning (the very left) of the next line and go on printing or displaying from that point. A series of "/" will create a succession of linefeeds (one "/" will cause one linefeed) .

- ＼ Pagefeed

A line slanting down from left to right means that when this symbol is encountered the printing head or the terminal display will move to the beginning (top left hand corner) of the next page, and continue printing or displaying from that point. A series of "\" will create a succession of pagefeeds. (One "\" will cause one pagefeed).

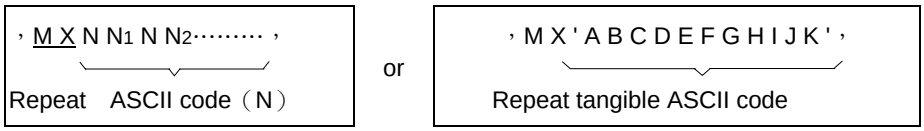
- , Comma

Used to separate statements in the file data. All the data included between two commas is a complete and executable statement (must not be used for beginning and end of file). Note that although the shape of a comma is the same as the shape of a single quotation mark, their positions are different (the comma is in a position near the center of the letter, while the single quotation mark is near the top right corner). The function meaning that they represent is completely different. Please refer to Item 2, background data format - statements.

- END File end

At the end of the ASCII file END is added to show that the ASCII file is finished.

2. Background Data Format



- MX:
  Represents the number of repeats. M can be 1 to 999. The ASCWR instruction can send out M times successively all the hexadecimal ASCII code or tangible ASCII code data contained between X and the first comma (,). If there is no data after X (ie, the comma comes directly after X), then the ASCWR instruction will send out M successive space codes. If you only have to send out the ASCII code or the tangible ASCII code once, then MX can be discarded.

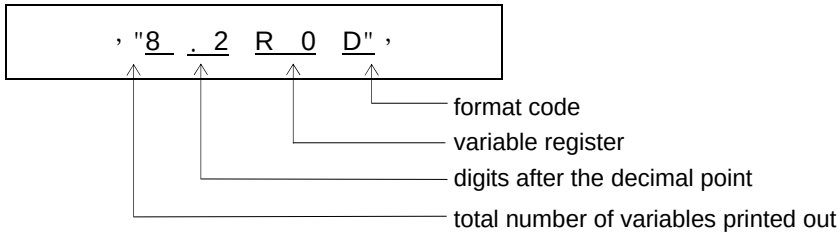
- ASCII code data format: This data format has an N two-digit hexadecimal value. Every two adjoining hexadecimal numerals starting from the right hand side of X will be regarded as an ASCII code. NN can be any ASCII code, including tangible or intangible ASCII code such as English characters, numeric symbols or control codes. However, its main use is as a special tangible code for control codes which cannot be represented by tangible character fonts or cannot find a font or symbol on the WinProladder ASCII editor. For tangible characters or symbols that can be directly represented on the ASCII editor by tangible keys, it should be more convenient to use the original printing out format. For example, if you want to print out the character "A", with the original printing method you can type A via the keyboard. But if you want to use ASCII code, you must check the table on which "A" is represented by 41 H, and then enter 41. It is obviously a lot less convenient.

- Original printing out tangible ASCII code data format: What is enclosed within two single quotation marks' ', can only be tangible ASCII code such as English characters, numerals, symbols, and graphics (characters that can be found on or input via the ASCII editor keyboard). The ASCWR instruction will faithfully print out all the characters that are contained in ' ', so if you need to print out a single quotation mark itself, you must have two successive single quotation marks. For example:

'I''M A BOY' will be printed out as I' M A BOY

If the graphics or symbols of the ASCII output device cannot be found on the ASCII editor keyboard, then you naturally are unable to do input using this format. In such a case you can check the ASCII code for that symbol or graphic, and use ASCII code to input and print out.

3. Variable Data Format



A data statement within two double quotation marks " ", is used to specify the register address of variable data, and what format or format code it will be printed out .

- Total number of variables printed out: In this example, "8" are used to print out the reserved 8 digit columns of the variable (R0) numeric value (including negative signs). If the variable value is larger than the total number of printed out digits then the high digit will be cut out. If the number of digits is insufficient, the remaining positions will be occupied by spaces.

- Digits after the decimal point: The number of digits after the decimal point within the total number of digits of the variable. In this example, in a total number of 8 digits, there are 2 places after the decimal point. The decimal point symbol "." itself occupies one position so the integer will remain 5 digits.

- Variable register: Can be R, D,WX, WY, etc, of a 16-bit register, or DR,DD,DWX, DWY, etc, of a 32-bit register. The contents of these registers can be retrieved and printed out using the format and format code specified by the contents of " " .

- Format code: Can use hexadecimal H, decimal D or binary B format for printing out (when format code is not specified, it will be decimal - therefore D can be omitted).

This example assumes that the content value of R0 is -32768. In the 8.2 format, the print out result is

	−	3	2	7	.	6	8

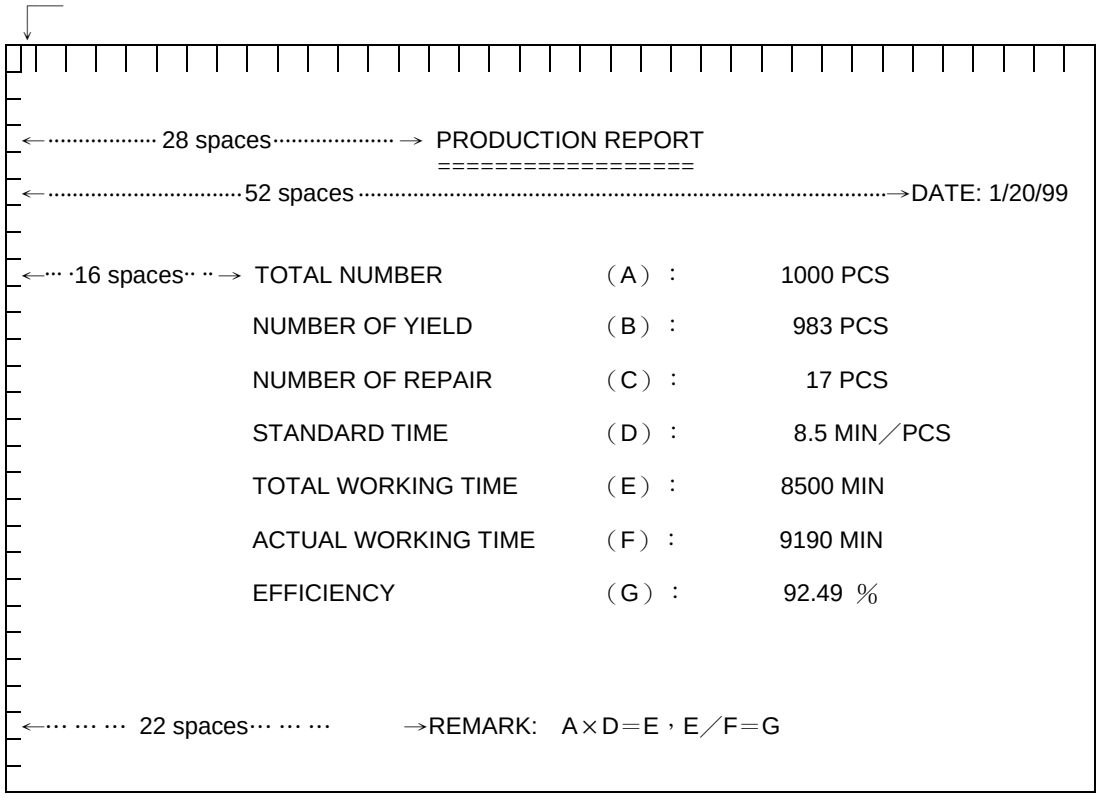
If the format changes from 8.2 to 5.1 then the print out result changes to

2	7	6	.	8

14.2 Application Examples of ASCII File Output

The file data print out will start from the top left hand corner of each page. It will print from left to right with lines going from top to bottom (please refer to the format in the diagram below). When the final character in a line is reached (this varies according to the output device - a printer can have 80 characters or 132 characters), the printer will automatically jump to the start (left-hand side) of the next line. If it has not yet printed to the final character, but encounters the linefeed command (/) or the page feed command (\), then it will jump to the start of the next line or the next page, and start printing from that point.

Suppose that the production statistics table for the manufacturing division of a certain company has the following format. This can be used as an example to explain the editing and printing out of its ASCII file data.



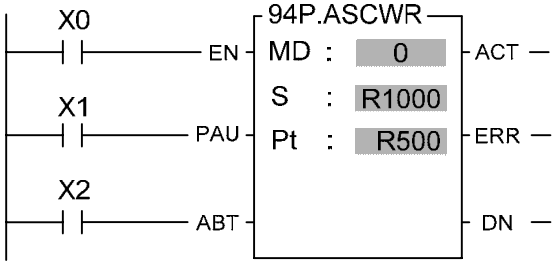
Before editing this file, you must first tell the file editor starting from which register within PLC the file to be edited shall be stored. When editing the file data, you must differentiate whether the file data to be edited (printed out) is fixed background data or variable data. The background data can be input using ASCII characters or symbol graphic of the original print out format (using what is contained inside ' '), or it can directly use the ASCII code of its character or symbol graphics. As for the variable data section, because it is stored in registers (so as long as the variable value changes, the print out numerical value will change with it), the print out message must contain the register number and print out format, such as number of characters, digits after the decimal point etc, as well as the format code that is used for the print out (contained inside " "). In the example in the table above, the year, month, day data and the total number (A) to efficiency (G) figures are all variable data. It assumes that the year, month, day data accesses the year, month, day registers (R4133 to R4131) within the real time clock register RTCR. R0 stores the total number (A), R1 stores the number of yield (B), etc, and R6 stores the efficiency (G) value. Below is the ASCII file data for this statistical table example:

```
///,28X,'PRODUCTION REPORT',/,28X,'==================', /,
52X,'Date:',"2R4132",'/',"2R4131",'/',"2R4133",//,16X,'TOTAL    NUMBER
(A) :',"10R0",' PCS',//,16X, 'NUMBER OF YIELD       (B) :' , "10R1",'
PCS',//,16X,'NUMBER TO REPAIR     (C) :',"10R2",' PCS',//,16X,'STANDARD TIME
(D) :',"10.1R3",' MIN/PCS',//,16X,'TOTAL WORKING    TIME (E) :',"10R4",'
MIN',//,16X,'ACTUAL WORKING TIME(F) :',"10R5",' MIN',//,16X,'EFFICIENCY
(G) :' ," 10.2R6",' %',/////,22X,'REMARK: AXD=E, E/F=G',END
```

* : In the above example ' ================== ' can be replaced by 18X'=' or 18X3D.

During the process of file output, when the output reaches variable data, the CPU will retrieve and do output with the numerical values at that time of the register whose address are contained within the " ". Therefore, if a variable is printed out both at the beginning and end of a file, a different numerical value may be obtained (when it has printed to halfway the register value changes).

After the file editing has been completed, the FUN94 instruction can be used to print out its background and dynamic data. If this file is edited (stored) starting from R1000, then when it is outputting, S must be specified as R1000 before there can be an accurate output, as seen in the program example in the diagram below left. Supposing that the numerical value of the variable register is as shown in the diagram below right, then when X1 and X2 are 0, and X0 goes from 0 to 1, this instruction will print out the statistical table from the previous page, from Port 1 of PLC.



R4133	=82	R2	=17
R4132	=12	R3	=85
R4131	=00	R4	=8500
R0	=1000	R5	=9190
R1	=983	R6	=9249