High-speed Controller Technology for Printers
Motonori KIRIHARA*, Shoichi Li**

Abstract
We adopted a 2-way interleave for accessing external code memory (ROM) of a microprocessor and data memory (DRAM) to increase the speed of the printer controller. In this controller, data is transferred from data memory to the electrophotographic engine section by direct memory access (DMA).

1. Introduction
As network environments spread, the demand for high-speed and high density electrophotographic printers, for connecting to networks, is increasing. To implement higher speed and higher density for electrophotographic printers, the improvement of the data processing capability of the controller section is becoming a critical issue, as well as increasing the speed and density of the electrophotographic printer engine section.

In a built-in microprocessor (hereafter CPU), on the other hand, the difference between internal processing speed and external memory access speed is widening. For high-speed printing image processing, it is necessary to quickly supply instructions from external memory to the CPU which is in charge of most image processing, so as to prevent downtime in the CPU pipeline, and to increase the speed of data input / output. In such a situation, we developed a controller to generate output images at high-speed.

2. Overview of controller

2.1 Features of controller
For the controller of an electrophotographic printer, various means are taken to increase the speed of output image generation, such as using dedicated hardware for character generation. Before developing this controller, we considered the following issues.

1. Increasing the speed of interrupt processing
   In the controller of an electrophotographic printer, most processing is executed by software using the CPU. Since these processings must be executed in real-time, interrupts are generated from various peripheral input / output devices, and are used as triggers to advance processing. Since these interrupts occur frequently, the high-speed processings of interrupts is indispensable. A factor which delays interrupt processing is saving and restoring internal registers in the CPU, which occurs when tasks of the control program are switched. The RISC processor AM 29040, made by AMD, which we built in to this controller, has functions called “multiple load and multiple store” to continuously write and read the data memory area at saving and restoring of internal registers. The control program uses this function at saving and restoring of internal registers when an interrupt occurs. This makes access to continuous addresses of the data memory area faster, increasing the speed of saving / restoring processing of internal registers.

2. Increasing speed of cache fill operation at mishit of instruction cache
   The output image of an electrophotographic printer is generated in page units, and program processing for image generation is divided into two primary stages. The first half of image generation is the editing process and the latter half is development processing to develop edited image data (vector data) to bit map data for output to the printer engine section. Because of the nature of the processing program, the cache hit rate for instruction code becomes low in the first half edit processing, and this rate becomes high in the latter half development processing. As a consequence, the cache hit rate in the first half of processing must be improved to increase the speed of processing. To compensate for the low cache hit rate, we adopted a method of accessing cache with 2-way interleave using ROM with page mode as code ROM. With this method, a 4-word unit of continuous access to the continuous address of code ROM, which occurs in the cache refill operation at cache mishit, becomes faster.

3. Increasing speed of print data transfer to engine section
   When bit mapped data is transferred to the engine section, the required data transfer speed between data memory and engine interface circuit is 1.5 M word / second. Processing this data transfer by the program increases overhead greatly, so we adopted direct memory access (DMA) for data transfer between data memory and engine interface circuit to decrease the overhead of program processing. Table 1 shows the features of the controller.

2.2 Configuration of controller system
Figure 1 shows a block diagram of this controller. To execute the processing program processing, the AM 29040 50 MHz CPU is used. This processor is a 32-bit RISC type CPU that has a 32-bit data bus and address bus. The following are connected to these buses.

---
* Manager, Second Engineering Department, Oki Data Corporation
** Second Engineering Department, Oki Data Corporation
1. Code ROM to store processing data
   An OTP ROM with page mode (one time PROM) is used. Code ROM is accessed by 2-way interleave.
2. Data memory to temporarily store data for program processing
   DRAM is used as the data memory. Data memory is accessed by 2-way interleave.
3. LSI: The following functions are included in the LSI.
   a. DRAM control circuit
   b. 2-way interleave control circuit for code ROM and DRAM
   c. Interface circuit with engine section
   d. DMA circuit for sending data to the engine interface circuit
   e. IEEE 1284 interface circuit
   f. Interface circuit with control section
   g. Bus conversion circuit
   The bus from the CPU is converted to a 16-bit wide local bus by the bus conversion circuit. Various host interface circuits are connected to this local bus.

3. Features of architecture

3.1 Data memory access method
   We adopted a 2-way interleave method as the data memory access method for this controller. The comparison between the 2-way interleave method and the ordinarily used non-interleave method is as follows.

   In the non-interleave method, if page mode of DRAM with page mode is used and when continuous access occurs to an address in the same column of the DRAM, the first access requires 3 memory clocks and the second and later accesses require 2 memory clocks.

   If the 2-way interleave method is used for accessing under the same conditions, the first access requires 3 memory clocks and the second and later accesses require only 1 memory clock. Therefore, in the case of 4-word continuous access, it takes 2.25 memory clocks per word for non-interleave access, while it takes 1.5 memory clocks for 2-way interleave access. Since the refill operation to data cache is normally a 4-word continuous access, 2-way interleave decreases the average access time of the refill operation to 2/3. When many registers are saved and restored to a continuous address in data memory for such operations as multiple load and store, there are 3 or more words for the second or later access. In this case, the effect of 2-way interleave is more obvious.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>AM29040, 50MHz</td>
</tr>
<tr>
<td>Address bus width</td>
<td>32bit</td>
</tr>
<tr>
<td>Data bus width</td>
<td>32bit</td>
</tr>
<tr>
<td>Bus speed</td>
<td>25MHz (memory clock)</td>
</tr>
<tr>
<td>Local bus specification</td>
<td></td>
</tr>
<tr>
<td>Address bus width</td>
<td>16bit</td>
</tr>
<tr>
<td>Data bus width</td>
<td>16bit</td>
</tr>
<tr>
<td>Bus speed</td>
<td>25MHz (memory clock)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
</table>
| Instruction font      | • ROM with page mode (OTP or mask), access time: 100 ns
|                       | • Access mode: 2-way Interleave                   |
|                       | • First access: 4 memory clocks; Second or later access: 1 memory clock |
| Memory                | • Mask ROM, access time: 100 ns                   |
|                       | • Access mode: simple access only, 4 memory clock access |
| Data                  | • DRAM with page mode, access time: 60 ns         |
|                       | • Access mode: 2-way Interleave                   |
|                       | • First access: 3 memory clocks; second or later access: 1 memory clock |
| Host Interface        | IEEE1284, SCSI, RS232C, Ethernet                  |
| Internal Interface    | Engine interface                                 |
|                       | Control section interface                        |

Table 1: Features of controller
3.2 Code memory access method

Just as in the case of data memory, we adopted 2-way interleave access for the code memory for the storing processing program. Memory used here is OTP ROM with page mode. The comparison with non-interleave access using OTP ROM without page mode, which is normally used, is as follows.

AM 29040 has an internal 8K byte instruction cache, which AM 29040 first accesses to execute the processing program. When a required instruction code does not exist in the instruction cache, 1 cache line (4 words) of code data is transferred from the externally connected code memory to the instruction cache. For the code memory side, this operation reads 4 words of continuous addresses. When 1 cache line (4 words) is refilled to the instruction cache from memory without page mode by non-interleave access, 4 memory clocks are necessary for each one 4-word access, that is, a total of 16 memory clocks.

When 1 cache line is refilled from memory with page mode by 2-way interleave, the first access takes 4 memory clocks, and 3 words for the second or later accesses take only 1 memory clock, a total of 7 memory clocks. This means that the refill speed for 1 cache line is approximately 2.3 times faster.

3.3 Data transfer method to engine interface circuit

Print data developed as bit map data in data memory is transferred to the FIFO (First In First Out) buffer, which is built in to the engine interface circuit, by DMA. DMA performs burst transfer at 4 words per one block while holding the bus right of the CPU. When the transfer of one block ends, DMA returns the bus right to the CPU once to enable memory access from the CPU. Then the DMA control circuit acquires the bus right again after a specified time passes, and transfers the next one block.

In this way, a bus corresponding to one cache line of the CPU is alternately used for a CPU and DMA block transfer, so as to minimize the wait time for CPU memory bus access during DMA operation. Data to the FIFO buffer of DMA is transferred at the same timing as continuous access of the code memory side, this operation reads 4 words of continuous addresses. When 1 cache line (4 words) is refilled to the instruction cache from memory without page mode by non-interleave access, 4 memory clocks are necessary for each one 4-word access, that is, a total of 16 memory clocks.

When 1 cache line is refilled from memory with page mode by 2-way interleave, the first access takes 4 memory clocks, and 3 words for the second or later accesses take only 1 memory clock, a total of 7 memory clocks. This means that the refill speed for 1 cache line is approximately 2.3 times faster.

4. Evaluation of memory access methods

4.1 Increasing interrupt processing speed

Data received from the RS232C interface is processed by interrupts. To compare the response time to interrupts between non-interleave access and 2-way interleave access, we measured time from when an interrupt signal from the RS232C interface circuit turns on to when the CPU reads the received data inside the RS232C interface circuit. The average of 10 times of response time measurement results was 10 m seconds for non-interleave access, but for 2-way interleave access, it was 6.5 m seconds, that is, the interrupt response time becomes 2/3.

4.2 Effect on printing processing in general

Table 2 shows a comparison of the effect of printing processing in general depending on data memory and code memory accessing methods. We compared one page of print data generation time for five types of typical print data between non-interleave access and 2-way interleave access. In the case of data memory access, print data generation by the 2-way interleave method was approximately 20% faster than the non-interleave method. In the case of code memory access, print data generation by the 2-way interleave method was approximately 46% faster than the non-interleave method.

5. Conclusion

This paper described high-speed technology for a high-speed printer controller.

The internal operation frequency of current built-in microprocessors is remarkable, showing a more increased difference in speed between the internal processing operation of a microprocessor and the speed of external memory access. A means to fill this gap is critical.