Performance Effect of I/O Peripheral Dynamic Reconfiguration on FPGA

Kazuki Hashimoto and Akira Yamawaki

Kyushu Institute of Technology, 1-1 Sensui, Tobata, Kitakyushu 804-8550, Japan

*Corresponding Author: yama@ecs.kyutech.ac.jp

Abstract

Embedded devices are equipped with specialized microcontrollers for each device. If an ultra-general purpose microcontroller with all peripherals existing, any kind of embedded device can be developed by such microcontroller. However, such microcontrollers cannot be built due to the scale and price.

A dynamic partial reconfigurable microcontroller (DPR microcontroller), which has a function to switch circuits for each application, can realize an ultra-general purpose microcontroller that can be used for any application.

For the realization and practical use of DPR microcontrollers, it is necessary to develop a hardware library-like structure in which optimized interface circuits for each peripheral device are designed in advance. In addition, the impact of dynamic switching of peripheral device interface circuits on performance has not been studied, so it is necessary to evaluate these effects.

Using dynamic partial reconfiguration, we measure the execution time, circuit size of the entire system, and power performance improvement ratio of a single system that displays a binarized image on an LCD module after gray scaling and binarizing the image.

The results show that the dynamic partial reconfiguration can be used to improve the performance of the system. As a result, we confirmed that the system works correctly even when dynamic partial reconfiguration is used. In addition, we were able to improve the processing speed by about 2.13 times compared with software processing, and reduce the circuit size by about 24%. We also achieved a 13.8-fold improvement in power consumption.

Keywords: specialized microcontrollers, dynamic partial reconfiguration, very general-purpose microcontroller.

1. Introduction

Currently, microcontrollers (System-on-chip, SoC) are installed in various embedded devices. SoCs have been developed specifically for each application. In order to increase the functionality and performance of embedded devices, it is necessary to develop SoCs with high functionality and high performance. This will inevitably increase the development time and cost. In addition, the lifecycle of products is getting shorter every year, so it is essential to reduce the development time and cost of SoCs in order to earn profits.

If there were an ultra-general-purpose microcontroller that could be used in a wide variety of embedded devices and incorporate all peripheral devices, the burden of SoC development could be eliminated, but such a SoC is impossible to realize in terms of circuit size and price.

One possible solution to these problems is the use of Dynamic Partial Reconfiguration (DPR) using a reconfigurable device. By using DPR, the chip size can be reduced, which leads to lower unit cost of SoC and lower price of applied products. In recent years, research on DPR has included switching data processing circuits to realize large-scale image processing [1-3], but no research has been done to dynamically switch peripheral device interface circuits. However, since the data necessary for data processing is input and output through the peripheral device interface circuit, it is extremely important to evaluate the effect of dynamic switching of the peripheral device interface circuit on performance. In addition, for the practical application of the DPR microcontrollers mentioned above, it is necessary to realize a concept such as a hardware library in which a controller optimized for each product is designed in advance.

In this paper, we develop an optimized controller for a 2.2-inch TFT liquid crystal display (LCD) module

(MSP2202)[4] equipped with ILI9341 manufactured by SWITCHSCIENCE to realize this concept. Then, we perform dynamic partial reconfiguration of both the data processing circuit and the peripheral device interface circuit in FPGA, which is one of the reconfigurable devices, and verify the operation by experiments, and evaluate and study the performance of the dynamic partial reconfiguration on the whole processing. The rest of the paper is organized as follows. Section 2 describes the outline of the dynamic partial reconfigurable microcontroller. In Section 3, we describe the optimal controller for the LCD module. In Section 4, we verify the operation and evaluate the performance of a series of systems using dynamic reconfiguration on actual devices. Finally, Section 5 concludes the paper and presents future work.

2. Dynamic Partial Reconfigurable Microcontroller

2.1 Realization of Ultra General Purpose Microcomputer by DPR

The relationship between various embedded devices and custom microcontrollers is shown in Figure 2.1.

Fig.2.1 Embedded Devices and Custom MCU

As shown in Figure 2.1, various automobiles, home appliances, and other embedded devices are equipped with custom microcontrollers specialized for each application, requiring the development of a dedicated SoC from scratch for each applied product.

Figure 2.2 shows an image of a very general-purpose microcontroller.

Fig.2.2 Image of a Super General-Purpose MCU

Although the burden of SoC development could be eliminated if there existed an ultra-general-purpose microcontroller with all the peripherals that could be used in various embedded devices, such a SoC is impossible to realize due to the scale and price of the circuitry.

One possible solution to these problems is the use of Dynamic Partial Reconfiguration (DPR) using a reconfigurable device. DPR is the ability to reconfigure only specific circuit data on a device without stopping all processing on the device during system operation. DPR is a function that reconfigures only the specific circuit data of a device without stopping all processing on the device during system operation.

Figure 2.3 shows an image of a DPR microcontroller that supports DPR.

Fig.2.3 Image of DPR MCU

2.2 Overview of DPR Microcontroller Configuration

The DPR microcontroller has the framework shown in Figure 2.4, and can switch the necessary data processing and peripheral device interface circuit data statically or dynamically during circuit operation without stopping the system operation. As a result, a single DPR microcontroller can be used to realize dedicated microcontrollers for different embedded systems.

Fig.2.4 Framework of DPR MCU

3. LCD module interface circuit

3.1 LCD Module Overview

In this paper, we develop an optimized controller for a 2.2-inch TFT liquid crystal display (LCD) module (MSP2202)[4] equipped with ILI9341 manufactured by SWITCHSCIENCE. The program to draw an image on the LCD module is shown in Fig. 3.1.

```
void Disp_img(uint16_t *src, uint16_t src_w, uint16_t src_h)
{
    uint32_t w, h;
    w = src_w - 0;
    h = src_h - 0;

    (1) writecommand(CMD_COLUMN_ADDRESS_SET);
    (2) writedata(0 >> 8);
    writedata(0);

    (3) writedata((src_w - 1) >> 8);
    writedata((src_w - 1));

    (4) writecommand(CMD_PAGE_ADDRESS_SET);
    (5) writedata(0 >> 8);
    writedata(0);

    (6) writedata((src_h - 1) >> 8);
    writedata((src_h - 1));

    (7) writecommand(CMD_MEMORY_WRITE);
    for(uint32_t i = 0; i < w*h; i++){
        (8) writedata(src[i] >> 8);
        writedata(src[i]);
    }
}
```

Fig.3.1 Image Rendering Program

3.2 Development of an optimized controller

Normally, SPI controllers support only 1-byte transmission, so when the program shown in Figure 3.1 is used to draw an image, gaps between byte transmissions, bit shifts, and other unnecessary steps are created, and the performance of the LCD module cannot be maximized. Therefore, we developed not only an SPI controller for 1-byte transmission, but also an SPI controller suitable for 2-byte data transmission, and an SPI controller with DMA suitable for transferring large amounts of data such as image data. Figure 3.2 shows the program for image drawing when using the optimized controller.

```
void Disp_img(uint32_t *src, uint16_t src_w, uint16_t src_h)
{
    uint32_t w, h;
    w = src_w - 0;
    h = src_h - 0;

    (1) writecommand(CMD_COLUMN_ADDRESS_SET);
    (2) write_COLUMN_data(0);
    write_COLUMN_data((src_w - 1));

    (3) writecommand(CMD_PAGE_ADDRESS_SET);
    (4) write_PAGE_data(0);
    write_PAGE_data((src_h - 1));

    (5) writecommand(CMD_MEMORY_WRITE);
    (6) img_disp(src, w*h);
}
```

Fig.3.2 Optimized Image Rendering Program

(1) Using the SPI controller for 1-byte transmission, send the

command (8-bit data) that specifies the width of the image to the LCD module.

(2) Using the SPI controller for 2-byte transmission, send 16-bit data that specifies the start and end points of the image width.

(3) Using the SPI controller for 1-byte transmission, send the command (8-bit data) that specifies the image height to the LCD module.

(4) Transmits 16-bit data specifying the start and end points of the image height using the SPI controller for 2-byte transmission.

(5) Transmits a command (8-bit data) to transfer image data from the microcontroller to the frame memory to the LCD module using the SPI controller for 1-byte transmission.

(6) Using the SPI controller for DMA transfer, write the image pixel value (16-bit data) into the frame memory of the LCD module for the number of pixels.

By switching the SPI controller used according to the program shown in Figure 3.2, the performance of the LCD module can be maximized.

4. Experiment and Discussion

4.1 Experimental environment

The DPR HW was designed using Vivado201 8 3 from Xilinx, and Vivado HLS 2018.3 from Xilinx was used as the high-level synthesis tool to create the image processing HW and SPI controller for DMA transfer. The operating frequency of the HW is 100 MHz. The experiments also show the effectiveness of the DPR microcontroller through comparison with software execution. For software execution, a PC (3.5GHz) and an embedded processor (650MHz) are used.

4.2 Unit system operation verification

Using DPR, we fabricated a system that displays a binarized image on an LCD module after grayscale and binarizing the image. Fig. 4.1 shows the results of displaying the binarized image on the LCD module using DPR.

4.3 Execution time measurement results

Fig.4.1 Binarization Image Display Result

Table 4.1 shows the measurement results of the execution time for the SW and HW processes.

Table.4.1 SW and HW Processing Time Measurement Results

Item	SW	HW
Image Processing HW: Total Reconstruction Time(ms)	-	3.29
Image processing :Total execution time (ms)	15.60	3.28
LCD drawing HW: Total reconfiguration time (ms)	-	9.86
LCD drawing: Total execution time (ms)	247.99	107.51
Total execution time (ms)	263.60	123.94

4.4 Circuit size measurement results

Measure the circuit size with and without DPR. Table 4.2 shows the measurement results of the number of truth tables (LUT) and flip-flops (FF) used in the fabricated hardware.

Table.4.2 SW and HW Processing Time Measurement Results

	LUT	FF
DPR not used	5729	8737
DPR used	4467	6556

4.5 Result and Discussion

The measurement results in Table 4.1 show that even when the circuit reconfiguration time is included, the processing speed is improved by a factor of approximately 2.13 compared to the SW processing. The circuit reconfiguration time accounts for about 9.6% of the total HW processing time, but since the execution time is very short 123.94 (ms), the reconfiguration time is not considered to have much effect on the actual operation.

From the measurement results of the circuit scale in Table 4.2, the LUT can be reduced by about 22% and the FF by about 24% by using DPR.

Finally, we obtain the power performance improvement ratio of HW processing to the embedded processor from the operating frequency in a series of systems. The performance improvement ratio to the embedded processor and the performance power improvement ratio are obtained and shown in Table 4.3.

Table.4.3 Electrical Performance Upward Ratio

	Performance upward ratio	Power Up Ratio	Electrical performance upward ratio
HW	2.12	6.50	13.80

Table 4.3 shows that the performance improvement ratio is 2.13, which indicates that the performance is improved by converting the SW processing to HW. In addition, even when the reconfiguration time of DPR is included in the performance power improvement ratio, the HW processing can be expected to improve the performance of the

embedded processor by about 13.8 times. These results show that DPR can improve the performance against the embedded processor even when it is used for processing.

5. Conclusions

In order to realize the DPR microcontroller, we developed an optimized controller for the LCD module as an interface for peripheral devices, verified the operation of a series of systems using DPR on actual devices, and evaluated the effects of DPR.

The experimental results show that the binarized image can be correctly displayed on the LCD module. Compared with the SW processing, the processing speed was improved by about 2.13 times, including the reconfiguration time, and the circuit size was reduced by about 24% by using DPR. And the power performance improvement ratio can be improved by 13.8 times.

Since the reconfiguration time depends on the number of reconfigurations, we believe that the effect of reconfiguration time cannot be neglected in a system where the number of circuit reconfigurations is large. Therefore, we will study how to reduce the reconfiguration time of HW as a future task. In addition, to realize the HW library, we will design and develop interface circuits for various peripheral devices.

References

- (1) Ryosuke Watanabe and Koki Abe, "Evaluation of a dynamically reconfigurable processor using sequential execution of image processing as an implementation example," IEICE Technical Report.CAS, Circuits and Systems, 108(104), pp.57-62, (2008).
- (2) Yohei Hori, Hiroshi Sakane, Toshihiro Katashita, and Kenji Toda, "Reduction of Circuit Size and Power Consumption of Multi-Cryptographic Module Using Dynamic Reconfiguration of FPGA," Transactions of Information Processing Society of Japan, Computing Systems (ACS), 1(2), pp.47-58, (2008)
- (3) Possessing Hagiri, Nozomu Togawa, and Masao Yanagisawa, "Dynamically Reconfigurable System Using FPGA and Its Application to Cryptographic Algorithm," IEICE, 99(No.658), pp7-14, (2000)
- (4) ILI Technology Corp, ILI9341 a-Si TFT LCD Single Chip Driver 240RGBx320 Resolution and 262K color1, (2011)