Soft DSP Design Methodology of Face Recognition System on Nios II Embedded Platform

Hao-Chung Shih and Chian C. Ho

Abstract—Based on the emerging and significant Soft DSP technique, this paper develops Soft DSP of face recognition to boost Face IDentification (FaceID) embedded system without extra coprocessor cost and power consumption. This paper not only illustrates the HW/SW codesign methodology of Soft DSP of face recognition in detail, but also gives a step-by-step implementation tutorial on Nios II embedded platform. The measurement results of Soft DSP of face recognition can verify Soft DSP technique easily achieves faster processing performance than the pure SW technique. Soft DSP technique has been entitled to be the development mainstream of FaceID embedded system field.

Keywords—face recognition; Soft DSP; embedded platform

I. INTRODUCTION

Face IDentification (FaceID) embedded system can assist with identifying suspects and alarming instantly, and it is more efficient than the security personnel’s eye tracking and check. Thus, FaceID embedded system can be extensively applied to various sites, like airport surveillance, roadside inspection, banking security, access control, emergent medicine, hooligan filtering and high-mountain administration. However, FaceID embedded system placing important emphasis on real-time biometric processing and cost-effective chipset architecture is an implementation dilemma. This paper adopts Soft DSP technique to design Soft DSP of face recognition so that FaceID embedded system can perform faster signal processing functions without extra coprocessor cost and power consumption.

In addition, how to select a useful open source library is inevitably the first and fundamental step to construct the embedded system implementation. Among a variety of open source face recognition libraries currently exploited by FaceID-oriented biometric industries, CSU FaceID system is doubtless one of the favorite choices, because it is full of royalty-free and feature-rich open source sample applications [1]. More importantly, all code of CSU FaceID system is written in ANSI C, and is easier to be ported onto typical embedded platforms. Rather than popular Intel Open source Computer Vision (OpenCV) library, Kitware Visualization ToolKit (VTK) system, CVSSP Recognition and Vision Library (RAVL) [2], [3], CSU FaceID system is also easier to be reformed by Soft DSP technique. This is because Intel OpenCV, Kitware VTK, and CVSSP RAVL are written in C++ [4]-[7]. Due to so many advantages, source code of CSU FaceID system has already been integrated into a lot of well-known biometric commercial products and research projects. In the same way, this paper will port CSU FaceID system onto Nios II embedded platform and further improve the performance of CSU FaceID system by Soft DSP technique. In more detail, this paper will focus on exploring the architecture of CSU FaceID system and breaking its bottleneck by Soft DSP technique in the following sections.

II. ARCHITECTURE OF CSU FACEID SYSTEM

CSU FaceID system consists of complete preprocessing and training processes, four fundamental face recognition algorithms, and two statistical analyses of recognition performance. The four fundamental face recognition algorithms involved in CSU FaceID system are Principle Components Analysis (PCA), a.k.a. Eigenfaces, combined Principle Components Analysis and Linear Discriminant Analysis (PCA+LDA), Bayesian Intrapersonal/Extrapersonal Classifier (BIC), and Elastic Bunch Graph Matching (EBGM) [8]-[12].

Figure 1 shows the architecture and flowchart of CSU FaceID system. The functionality of each phase in CSU FaceID system is interpreted in order as below. First, in order to reduce the unwanted variation from the original video frames, five significant normalization techniques at the preprocessing phase, such as integer to float conversion, geometric alignment, elliptical masking, histogram equalization, and contrast/brightness normalization, are carried out. All preprocessing code of CSU FaceID system can replicate the preprocessing operation for evaluation of acknowledged FERET database [1], [13]. Next, in order to extract highly correlated features for the PCA, PCA+LDA and BIC face recognition algorithms, the training phase creates subspaces by projecting and matching original video frames, but EBGM algorithm doesn’t need the global training subspaces. Third, the testing phase generates distance matrices to evaluate the distance from each video frame to all other video frames in the gallery after four face recognition algorithms above are performed. Finally, the analysis phase adopts either of two statistical analyses on the distance matrices to compare the performance of the recognition rate.

Among all blocks of CSU FaceID system in Figure 1, the computational complexity of the histogram equalization at
the preprocessing phase is simpler but heavily repeated. That is rightly where our developed Soft DSP of face recognition will concentrate on and improve on, because Soft DSP technique is better at solving such a computation bottleneck than those pure HW or SW techniques. More explicit reasons will be clarified in Section 3.

III. SOFT DSP AGAINST PURE HW/SW TECHNIQUES

With the on-going progress of semiconductor technologies, it has made System-on-Chip (SoC) come true, as well as the reconfigurable computing platform. Reconfigurable computing platform composed of one or more Reduced Instruction Set Computer (RISC) processor cores (e.g. ARM, PowerPC, or Nios II), embedded memory, and programmable logic is a preferable alternative to Application-Specific Integrated Circuit (ASIC) platform or general-purpose processor system, since it can take full advantages of hardware logic’s throughput and software processor’s flexibility [14]-[16]. Next, this paper will address the performance-to-cost-ratio difference between the pure HW technique, the pure SW technique, and the HW/SW codesign method like Soft DSP technique, and explain why the Soft DSP technique is the compromise way to boost the face recognition processing on the reconfigurable computing platform like Nios II embedded platform.

Pure HW technique represents the computationally intensive segments of code are processed and accelerated by specific hardware fabric, especially by the dedicated coprocessor like Codec chip or the comprehensive coprocessor like DSP. This specific hardware fabric, whether Codec chip or DSP, is working apart from the general-purpose central processor and is able to access the memory, bus, interface and peripherals independently. The pure HW technique with the specific hardware fabric can execute dedicated or comprehensive arithmetic functions much more quickly than any of pure SW and Soft DSP techniques, but it consumes more circuit area and power consumption than other techniques. These are critical issues to embedded systems. In addition, DSP users must take extra efforts to get familiar with another assembly instruction set or specific programming language based on some complex DSP architecture.

There are a large number of pure SW techniques to raise the processing performance, like algorithm-level ideas, code optimization by profiling tools, preliminary lookup table, programming by assembly or low-level language, compiler efficiency evolution, and so on. They are performed only by the general-purpose central processor, and do not need additional coprocessor’s hardware fabric and power consumption. The pure SW techniques are usually the most economic solution to cost-critical issues. But all pure SW techniques above can only contribute a limited extent of improvements. These SW techniques are not good enough often.

Soft DSP technique is also referred to as Custom Instruction or Very-Long-Instruction-Word-like (VLIW-like) RISC. They are actually similar to each other, because all of them raise the embedded signal processing performance by running some custom instructions directly on the dedicated VLIW-like logic blocks [17]-[19]. From the circuit configuration in Figure 2, it is clearly seen that the dedicated logic blocks are compact and adjacent to Arithmetic Logic Unit (ALU) in the processor’s data path configuration. Therefore Soft DSP technique not only can finish some complicated arithmetic as soon as the pure HW technique, but also can make circuit cost and power consumption considerations as simple as the pure SW technique. Soft DSP technique can take both advantages of effective HW design and efficient SW design, but cannot access the memory, bus, interface and peripherals independently and promptly. Therefore, this paper will adopt Soft DSP technique to boost face recognition processing for real-time biometric embedded application.

IV. DESIGN METHODOLOGY AND TUTORIAL

As commented in Section 3, Soft DSP technique is a HW/SW codesign method utilizing both throughput of hardware logic and flexibility of software processor. So reconfigurable Nios II embedded platform featuring
programmable processor core and tailored instruction set is one of highly qualified benchmarks to show the HW/SW codesign methodology of Soft DSP technique. This paper will make use of reconfigurable Nios II embedded platform and Nios II HW/SW seamless toolchain, like SOPC Builder, Quartus II, and Nios II IDE, to implement Soft DSP of face recognition as below.

Besides, as mentioned in Section 2, the code of the histogram equalization preprocessing of CSU FaceID system is where Soft DSP technique is good at and where this paper is interested in, because it is filled with simpler but heavily repeated segments. Figure 3 shows some original code segment of the histogram equalization preprocessing of CSU FaceID system. Inside the triple loop of Figure 3, MIN and MAX are simply to determine which one is the smallest and largest, respectively. In addition, following ASCII standard, the loop variables, width, length, and channel, mean every video frame’s width size, height size, and channel size, respectively. By default, the loop variables, width, length, and channel, are equal to 160 (pixel), 120 (pixel), and 1, respectively, for CSU FaceID system. For this so-called simpler but heavily repeated bottleneck, the tutorial will run it with Soft DSP of face recognition based on Soft DSP technique, specifically by the enhanced custom instructions of MIN[] and MAX[]. The detailed design methodology is illustrated as follows.

First, the dedicated logic for running the custom instructions of MIN[] and MAX[] must be designed through Verilog Hardware Description Language (HDL). The objectives of the dedicated logic, min.v and max.v, are just to quickly find out which one is the smallest and largest, respectively, in hardware way. After the connectivity diagram and pin assignment of Nios II processor is ensured to be valid and appropriate through SOPC Builder tool, the dedicated logic in Verilog HDL can be seamlessly integrated into Nios II processor core through SOPC Builder tool. In the stage of coupling custom instructions with Nios II processor, the imported combinational logic files, max_f.v and min_f.v, are the aforementioned dedicated logic for running the floating-point custom instructions of MIN[] and MAX[], and the imported combinational logic files, max_i.v and min_i.v, are the aforementioned dedicated logic for running the integer custom instructions of MIN[] and MAX[].

In addition, beyond SOPC Builder v6.0, the tool provides Nios II processor core with a built-in module of “Floating-Point Hardware” for accelerating the floating-point computation of Nios II processor [20]. Next, Nios II processor core must be re-synthesized by Quartus II tool and download the updated .sof image file to the embedded platform through Quartus II tool. Afterward, this reformed Nios II hardware platform has the capability to run the enhanced custom instructions.

On the other end, for the software code segment as shown in Figure 3, the tutorial has to replace the original instructions, MIN[] and MAX[], with the enhanced custom instructions, ALT_CI_MIN_F() and ALT_CI_MAX_F(), through Nios II IDE tool as marked in Figure 4. In the marked section of software code of Figure 4, ALT_CI_MIN_F() and ALT_CI_MAX_F() are created and engaged in driving the dedicated logic, min_f.v and max_i.v, to decide which one is the smallest and largest quickly. Next, Nios II IDE tool applies GNU C Compiler (GCC) to rebuild the reformatted software code of the histogram equalization preprocessing of CSU FaceID system into .elf executable file, and downloads the executable file to the embedded platform. In short, the HW/SW codesign methodology of Soft DSP
V. IMPLEMENTATION RESULTS

In order to analyze and compare the implementation results, this paper inserts a built-in module of “Performance Counter Unit” into the code of the histogram equalization preprocessing of CSU FaceID system [21]. The built-in module can support to count the processing time of Nios II processor with/without Soft DSP of face recognition more precisely. In addition, in order to simplify the implementation results and performance analysis, the implementations have to confine the performance comparison to the function of the histogram equalization preprocessing, histEqual(), as shown in Figures 3 and 4. The software code and implementation result of original histogram equalization preprocessing, histEqual(), is run by Nios II IDE tool and shown in Figure 5, and that of Soft histogram equalization preprocessing improved by Soft DSP technique, histEqual_CI(), is also run by Nios II IDE tool and shown in Figure 6. The implementation results show that the processing time of Nios II processor with support of only two custom instructions is about 119.1 ms shorter than original Nios II processor for every video frame’s preprocessing, and its improvement rate achieves 28%. So the boosting effect of Soft DSP of face recognition without extra coprocessor cost and power consumption is dramatic and significant.

VI. CONCLUSIONS

According to the implementation results of the histogram equalization preprocessing in the tutorial, the processing time of Nios II processor with Soft DSP of face recognition can be decreased by about 120 ms for each 160 × 120 video frame’s preprocessing. In other words, Soft DSP of face recognition implemented in the tutorial can decrease the processing time of 120n ms totally when preprocessing a video sequence consisting of n frames. Furthermore, it is worth noting that only two custom instructions are implemented in the tutorial so far and Nios II processor is permitted to extend and couple 256 unique custom instructions at most [19]. Hence, the more bottlenecks that is simpler but heavily repeated are found out to reform, the faster Soft DSP of face recognition can push the recognition processing of FaceID embedded system. Besides, Soft DSP technique features: 1) smaller circuit area, 2) lower power consumption, 3) better design flexibility, and 4) richer feature scalability, against the pure HW technique.

REFERENCES