Rapid Design Methodology for Accelerator-based Wireless Sensor Nodes in Structure Health Monitoring

Daming Zhang [1]
NICS, Department of Electronic Engineering
Tsinghua University
Beijing, China
zdm06@mails.tsinghua.edu.cn

Yongpan Liu [2]
NICS, Department of Electronic Engineering
Tsinghua University
Beijing, China
ypliu26@gmail.com

Shuangchen Li [3]
NICS, Department of Electronic Engineering
Tsinghua University
Beijing, China
lschen.26@gmail.com

Huazhong Yang [4]
NICS, Department of Electronic Engineering
Tsinghua University
Beijing, China
yanghz@tsinghua.edu.cn

***Abstract*—Node-level signal processing becomes popular to reduce the data amount and power consumption of wireless sensor nodes. It puts more productivity challenges on the design flow, because various applications lead to different accelerators. In this paper, we proposed a rapid design methodology for accelerator-based sensor nodes (RAS design methodology). We evaluated RAS design methodology in a structure health monitoring (SHM) system and implemented a sensor node with a radix-2 8-point FFT accelerator using ANSI C. Several C-level optimizing techniques are used to further improve the performance. Measured results show that compared with the software solution, RAS design achieves up to 40x times speedup while the design time is shorten from several weeks to days.**

Keywords- Wireless Sensor Nodes, Design Methodology, C2RTL

I. INTRODUCTION

With the rapid costs and volumes shrinking of the sensor nodes, wireless sensor networks (WSN) have been widely used to collect data from the environment. Node-level processing is quite efficient to reduce the transmission data and power of WSNs. For example, plenty of work had explored different approaches to apply node-level FFT in SHM. In [1], a FFT module is implemented on a TMS320 DSP to analyze the vibration of bridges. Later, a PowerPC-based MPC555 Platform is used to monitor the Alamosa Canyon Bridge [2]. Researchers also realized FFT functions on the microcontrollers, such as Atmega8 [3]. However, they either suffer from its high power consumption or low performance.

Hardware-assisted accelerators seem to be a promising solution for an energy-efficient sensor node. In [4], the μAMPS-3 micro sensor DSP chip combines a custom 16-bit CPU with hardware accelerator cores for FIR filtering and FFT modules. In [5], an event-driven architecture is proposed to adopt an ultra-low power event handler to manage different accelerators. However, it is a prohibitive task to design various accelerators for different applications in WSNs.

This paper proposed a rapid design methodology for accelerator-based sensor nodes (RAS design methodology) of various applications. We demonstrated the approach in a SHM system and realized a sensor node with a radix-2 8-point FFT accelerator using ANSI C. What's more, several C-level optimizing techniques are presented to boost the performance. According to the measured results, RAS design methodology achieves significant speedups both in the performance and design productivity.

This paper is organized as follows: Section II proposes RAS design methodology for accelerator-based WSN applications. Section III demonstrates RAS design methodology in a SHM sensor node. Section IV reports the measured results and analysis; Section V concludes the paper.

II. RAS DESIGN METHODOLOGY

Nowadays, WSNs are widely deployed in many domains, such as environmental monitoring, multimedia application, structure health monitoring, remote medical services, locating and so on. The node-level signal processing in different applications needs various hardware accelerators. As Table I has shown, more than 20 types of accelerators should be designed. What's more, the parameter of each type may also vary according to different scenarios. Even worse, the short design-to-market time and small design group make the accelerator design a huge challenge.

To solve the problem, the left side of Figure 1 gives RAS design methodology for hardware accelerator. Instead of the manual C2RTL transformation, the design and optimization of accelerators are finished on C level and the C2RTL tool will handle the following steps automatically. Compared with the traditional flow, the design time can be shortened greatly due to the higher abstract level on C code instead of RTL. However, the designers should focus on the

architecture and code styles of C specification. They will have a great impact on the performance and area of accelerators. For example, our results show that different C specifications lead to accelerators with over 2X performance variations. In order to shorten the design time as well as improving the performance, we present several optimizing techniques in Section III to design a real SHM system.

TABLE I. HARDWARE ACCELERATORS IN DIFFERENT WSN APPLICATIONS

WSNs Applications	Hardware Accelerators in Different WSN Application Fields （examples）
Video/Image Processing	MPEG-2, JPEG, Video/Image codec, SVM, Gabor Wavelet transform
Audio Processing	Audio codec, MPEG-1, DTW, VQ, EM
Data Processing	RLE, FIR filter, FFT, Wavelet transform
Wireless Communication	PSK, FSK, ASK, MSK, FIR filter, FFT
Security	AES, DES, RSA, ECC

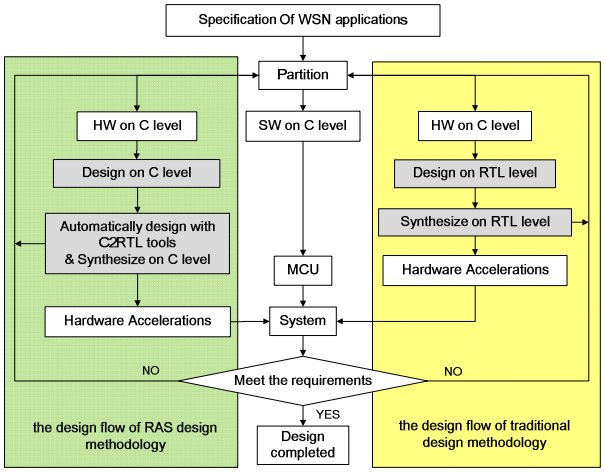

Figure 1. the design flow of traditional and RAS design methodology

III. DESIGNING A SHM SYSTEM USING C SPEC

According to RAS design methodology, we designed and optimized a SHM system using C specification.

A. System Architecture

The typical flow of a SHM system is illustrated as followings: multiple sensors are used to collect the infrastructure vibrating curves. A/D converters transform those analog signals into digital data, which will be processed by MCUs or accelerators. After that, the structure health information is transmitted via a wireless transceiver. In this paper, we partitioned the above flow into software and hardware part. The software controls the data from the sensors to the transceiver, while the hardware is in charge of processing the vibrating data.

Figure 2 shows the architecture of the proposed SHM system. It consists of several peripheral modules (vibration sensor, A/D converter, CC2420) and two controlling modules (the processor-MCU 8051 and the FFT accelerator). The transceiver – CC2420 is connected to MCU 8051 with an 8-bit SPI interface. Between the MCU and the accelerator, we use handshaking (HS) and 16-bit memory (MEM) interface to transmit control signals and data. The MCU sends the start signal to the accelerator and begins the operation. The FFT accelerator send back the done signal to indicate the completion. According to the new flow, both software and hardware part are expressed in C code. The following section will propose several techniques to optimize the C specification of the FFT accelerator.

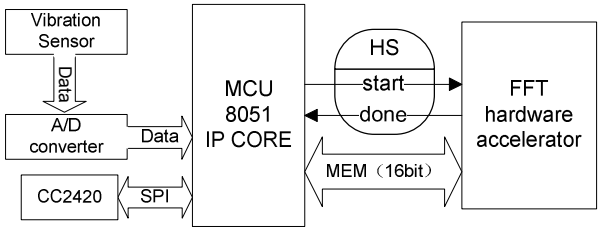

Figure 2. the architecture of the SHM system on the basis of MCU 8051 IP CORE

B. FFT Accelerator Optimizations

According to the bridge monitoring requirement, we choose a radix-2 8-points FFT algorithm to process the vibrating data. Formula (1) and (2) illustrate the output of even numbers and odd numbers by applying FFT on the input vector x. X is the output data in the frequency domain. Formula (3) gives the method to compute the twiddle factors W. The overall computing flow of radix-2 8-points FFT algorithm is described in Figure 3.

$$X[2r]=\sum_{k=0}^{3}\left(x[k]+x[k+4]\right)W_4^{rk}, r=0,1,2,3 \quad (1)$$

$$X[2r+1]=\sum_{k=0}^{2}\left(x[k]-x[k+4]\right)W_8^{k}W_4^{rk}, \quad r=0,1,2,3 \quad (2)$$

$$W_8^{kn}=e^{-j\frac{\pi}{4}kn}, \quad n=0,1,\cdots 7, \quad k=0,1,\cdots 7 \quad (3)$$

The In the C implementation, the FFT algorithm consists of three parts: input-address change unit, twiddle factor generator and butterfly computing unit. We proposed several optimizing techniques as below:

As float operations will lead to inefficient hardware implementation in C2RTL tools, we provide a flow to transform float or double operations into integer ones. In FFT, we enlarge the floating number by shifting left 10bits. For example, the integer values of twiddle factors are shown in Table 2.

The calculation of twiddle factors is another operation to slow down the performance. As the twiddle factors are fixed,

it can be preset to avoid unnecessary computation. In this implementation, the storage of twiddle factors is reduced by 50%, shown in Table II.

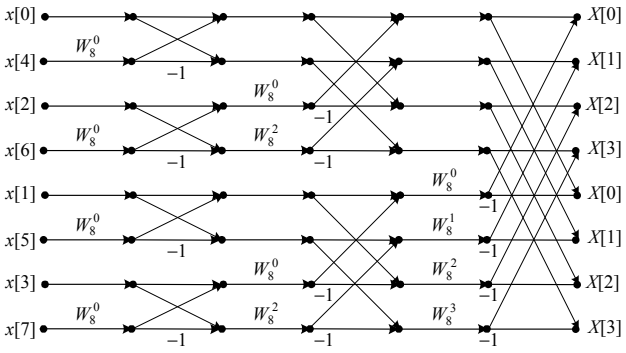

Figure 3. the computation flow of radix-2 8tps DIT FFT algorithm

Twiddle factors (Multiplied by 2^{10})	Value	In power-of-two form
W[0]/W[13]	1024	2^{10}
W[5]/W[8]	-1024	-2^{10}
W[2]/W[11]/W[14]/W[15]	724	$2^9+2^7+2^6+2^4+2^2$
W[3]/W[6] /W[7] /W[10]	-724	$-2^9-2^7-2^6-2^4-2^2$
W[1]/W[4] /W[9] /W[12]	0	0

TABLE II. TWIDDLE FACTORS OF RADIX-2 8TPS FFT ALGORITHM

Finally, the multiplying operations occupy lots of computing time in FFT. We use shifters and adders to replace the multiplier. We transform the twiddle factors in power-of-two form. Thus, a twiddle factor multiplication can be expressed as several shifting and adding operations. In this way, the computing time can be shortened with reasonable area overheads.

C codes with and without these optimizations are provided to a commercial C2RTL tool-eXCite and we achieve and evaluate those HDL modules, separately. Furthermore, we can synthesize the C codes under different goals, such as the best performance, the minimal area and so on. All these cases are explored and reported in the experimental results. Figure 4 shows the simulation results of FFT accelerator under the best performance metric after optimizations.

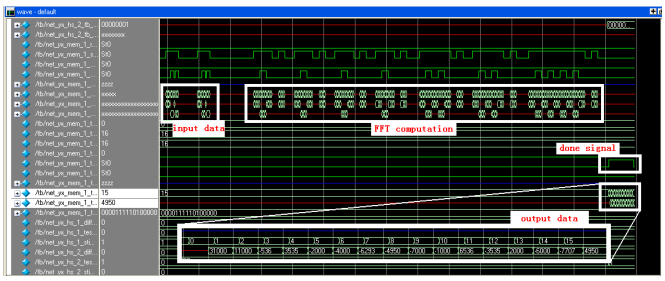

Figure 4. the computation flow of radix-2 8tps DIT FFT algorithm

IV. EXPERIMENTAL RESULTS

A. Implementation Setup

Figure 5 shows the hardware implementation of the SHM system: the sof file, containing the MCU 8051 and the FFT accelerator, is written to the FPGA (EP2C70F896C6, Cyclone II, Altrea) on the DE2-70 board through Quartus II 7.2. The C code of the SHM system is compiled into a hex file by Keil C51 and it is written to the FLASH through the DE2-70 Control Panel. The input data is collected and send to the MCU 8051. And the output data is transmitted to CC2420 through the SPI Bus. The photo of the realized system is shown in Figure 6.

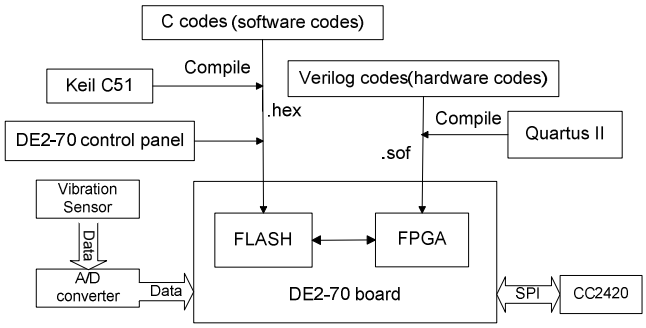

Figure 5. hardware realization flow of the SHM system

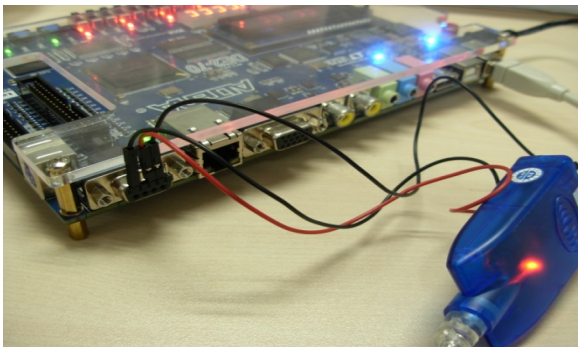

Figure 6. actual hardware system for test

In our experiments, the executing time is measured by an oscilloscope and the area is reported by Quartus II. The clock frequency is set as 50 Mhz. We use three different data sets to evaluate the performance, covering both random (Set1) and real vibrating data (Set2 and Set3). Furthermore, in order to compare the design period and the system performance of the RAS design methodology, a SHM system with a manual designed FFT hardware accelerator is realized. The software-based FFT solution is also included for comparison

B. Design Flow Comparison

The design time and the system performance of the traditional and RAS design methodology are separately

shown in Table III. The design time can be reduced by 5 times with similar same performance and area. What's more, the savings of design time can be even larger when more complex accelerators are used. It should be noted that a manual design with careful tunings may lead to a smaller accelerator, however the redundancy of elements in a FPGA board make the design time more important.

TABLE III. DESIGN PERIOD AND THE SYSTEM PERFORMANCE OF FFT ALGORITHM IN DIFFERENT DESIGN METHODOLOGIES

	Design time	DP area	Execution time (us)
traditional design in RTL level	14 days	1472	126
RAS design in C and RTL level	3 days	1393	138

TABLE IV. EXECUTION TIME OF DIFFERENT DESIGN METHODOLOGIES

Execution time (us)	MCU 8051 only	MCU 8051 with FFT hardware accelerator in Post-Best	ratio
Set1	6881	138	49.9
Set2	7031	143	49.2
Set3	7088	144	49.2

C. *Comparison of C Optimizations on FFT Accelerators*

We show the C optimizations techniques in Table V and Table VI. Table V shows the hardware area of FFT algorithm before and after optimizations in the set of best performance and minimum area. The area after optimizations is larger than before by 32%-39%, as the adders and shifters in parallel computation need more area than multipliers in the hardware accelerator. On the other hand, Table VI presents that the execution time of the system before optimizations is 2.2-2.4 times compared with the system after optimizations.

TABLE V. THE HARDWARE AREA OF FFT ALGORITHM IN OUR DESIGN METHODOLOGY

Combinational DP area of FFT algorithm	Set in eXCite	
	Best Performance	Minimum Area
Post-optimizations	1393	1153
Pre-optimizations	1056	832
ratio	1.32	1.39

V. CONCLUSIONS

In this paper, we proposed RAS design methodology for fast design of hardware accelerators on WSN applications to realize the SHM system. And the optimization of C codes of FFT algorithm can distinctly improve the efficiency of the automatic transformation from software codes into hardware accelerators with a C2RTL tool - eXCite. Based on the experimental results, with our design methodology, the effect of acceleration of the SHM hardware accelerator is very apparent and the design period is reduced quite a lot. The hardware tests indicate that our design methodology is suitable for fast design of hardware accelerators on WSN applications.

TABLE VI. EXECUTION TIME OF FFT ALGORITHM IN RAS DESIGN METHODOLOGY

Execution time(us)	Pre-Min	Post-Min	Ratio
Set1	341	148	2.30
Set2	351	152	2.31
Set3	355	155	2.29
Execution time(us)	**Pre-Best**	**Post-Best**	**Ratio**
Set1	336	138	2.43
Set2	345	143	2.41
Set3	351	144	2.44

Post-Best: post-optimizations in best performance
Pre-Best: pre-optimizations in best performance
Post-Min: post-optimizations in Minimum Area
Pre-Min: pre-optimizations in Minimum Area

REFERENCES

This work was supported in part by the NSFC under grant 60976032,National Science and Technology Major Project under contract 2010ZX03006-003-01, eXCite software donated from Y Explorations Inc. and High-Tech Research and Development (863) Program under contract 2009AA01Z130.

[1] Giovanni Betta, ,etc;A DSP-Based FFT-Analyzer for the Fault Diagnosis of Rotating Machine Based on Vibration Analysis, IEEE TRANSACTIONS, VOL. 51, NO. 6, DECEMBER 2002;

[2] Jerome P.Lynch,etc; "Field validation of a wireless structural monitoring system on the Alamosa Canyon Bridge"; SPIE's 10th Annual International Symposium on Smart Structures and Materials, San Diego, CA, USA,March 2-6, 2003;

[3] Yan Yu, Jinping Ou; Design of wireless intelligent sensor for structural health monitoring, International Conference on Intelligent Sensors, Sensor Networks and Information Processing(ISSNIP) , Melbourne , 2004;

[4] Chu Chad,etc; Design of a high performance FFT processor based on FPGA, Asia and South Pacific Design Automation Conference(ASP-DAC),2005, Shanghai, China, January 18-21;

[5] Hongjiang He; Hui Guo; the Realization of FFT Algorithm Based on FPGA Co-Processor; Intelligent Information Technology Application (IITA), 2008;

[6] Steven Hsu, etc; A 2GHz 13.6mW 12x9b Multiplier for Energy Efficient FFT Accelerators, Solid-State Circuits Conference, 2005. Proceedings of ESSCIRC, Grenoble, France, 2005;

[7] Straser, E. G. and A. S. Kiremidjian. 1998. A Modular Wireless Damage Monitoring System for Structures. Report No. 128, John A. Blume Earthquake Engineering Center, Stanford, CA.