A Lookup Table of Mixed-radix FFT Implementation and Optimization on Sunway Embedded Platform

Youcai Luo[1,a], Hongsheng Wang[1,b] and Lin Han[1,c]

[1]School of information Engineering Zhengzhou University Zhengzhou, China
a. 136922441@qq.com, b. whs1814@foxmail.com, c. strollerlin@163.com

Keywords: Sunway processor, fast fourier transform, lookup table, loop unrolling, vectorization, data prefetching.

Abstract: The Fast Fourier Transform is one of the most fundamental algorithms used in digital signal processing. It is of great significance to implement an efficient Fast Fourier Transform algorithm on Sunway embedded platform. In this paper, based on the radix 2-radix 4 mixed-radix algorithm implemented in DSPLIB Library of Texas Instruments, it takes the lookup table to replace the reversal computation function, which improves the performance by 1.69 times. In addition, the algorithm is manually optimized for the Sunway embedded processor by using loop unrolling optimization, SIMD vectorization, and data prefetch optimization with the studying of Sunway processor chip architecture. Comparing the initial state, the optimal average performance of the algorithm has been improved by about three times on Sunway.

1. Introduction

In recent years, with the development of domestic Sunway processors, more and more of them are used in high-performance computing, personal computers and other fields, while embedded processors occupy nearly half of the processor market, Sunway's application in the embedded field is not rich enough, and its research is still in the exploration stage. The TMS320 series chips produced by Texas Instruments (TI) are widely used in various fields and become currently the most influential and successful digital signal processing (DSP) chip processors. The TMS320C6000 digital signal processor library (DSPLIB) is a platform-optimized DSP function library[1]. It includes general-purpose signal processing routines called by C programs, which are usually used in computation-intensive real-time applications. These routines can achieve higher performance. By providing ready-made DSP functions for source code, DSPLIB can greatly shorten the application development time.

The Fast Fourier Transform (FFT) algorithm plays an important role in the field of digital signal processing. It solves the problem that the Discrete Fourier Transform (DFT) algorithm is not practical due to the huge amount of calculation[2]. FFT is also a key step affecting the performance of digital signal processing. Implementing an efficient FFT algorithm is very significant to improve the real-time performance of digital signal processing. It is of great research significance and practical value to study the implementation and optimization of FFT on Sunway processor.

Sunway 221 processor is a domestic high-performance dual-core processor based on the third-generation "Sunway 64" core, which is mainly targeted at the demand of high-density computing embedded applications[3]. The third-generation Sunway core microstructure is shown in Figure 1.

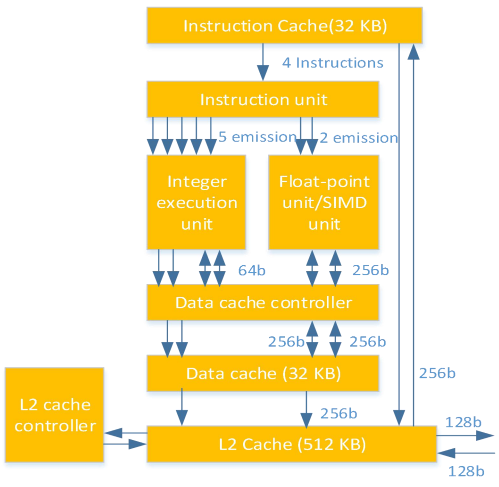

Figure 1: Third-generation Sunway core microstructure.

In this paper, it considers the implementation of the radix 2-radix 4 mixed-radix FFT algorithm, which is one of the most important signal processing algorithms in DSPLIB. and the lookup table is used to improve the original method of the output order rearrangement using the reversal computation functions in the program. In order to adapt to the Sunway embedded platform, it uses loop unrolling optimization, SIMD vectorization, and data prefetching optimization to improve the performance of the mixed-radix FFT on the platform of Sunway by studying the vectorization component and cache component of the Sunway hardware. The efficient mixed-radix FFT algorithm is implemented.

2. Radix 2- Radix 4 Mixed-radix FFT Algorithm

The radix 2-radix 4 mixed-radix FFT algorithm can be used to deal with the FFT with the integer power of the number of sequence points N being 2. If the number of sequence points does not meet the requirements, zero will be filled later. Compared with the most commonly used radix-2 and radix-4 algorithms in FFT processor[4,5], the performance of radix 2-radix 4 mixed-radix FFT algorithm is better than that of radix-2 algorithm, and it can make up for the disadvantage that radix-4 algorithm can not deal with all the integer power sequence points of 2[6]. The idea of radix 2-radix 4 mixed-radix FFT algorithm is as follows: firstly, radix-4 FFT algorithm is used to calculate the FFT process of sequence points except the last butterfly calculation. Then, by judging whether the number of input sequence points N is an integer power of 4, the algorithm of radix-2 or radix-4 is used in the last butterfly calculation[7]. The program implementation is mainly divided into the following steps:

First, according to the number of input points N, all the required butterfly factors are calculated, sorted, and a butterfly factor table is created. For each radix 4 butterfly operation, the first butterfly factor is 1, so there is no need to access and calculate;

Second, according to the radix 4 FFT algorithm, the sequences x(n) are divided into four sequence columns, which has N / 4 points in each columns, namely x (n), x (n + N / 4), x (n + 2N / 4), x (n + 3N / 4), n = 0,1,2,... N / 4;

Third, according to the periodicity of the butterfly factor, the sequences multiplied by the same butterfly factor are combined together to calculate the product between the recombined sequence and the corresponding butterfly factor except for the last stage butterfly calculation, and the calculation results are stored in the output buffer;

Fourth, according to the input sequence number N, it determine whether to use the radix 2 operation or the radix 4 operation in the final stage of butterfly calculation, and the sequence is recombined according to the radix 2 or radix 4 FFT operation rules. It puts the calculation results into the output buffer;

Finally, It uses the reversal computation function to rearrange the output, and gets the FFT calculation results.

3. Program Performance Optimization for Sunway Processor

First of all, on the basis of the above-mentioned radix 2-radix 4 mixed-radix FFT algorithm, it combines with the lookup table method to optimize the mixed-radix FFT program by using the lookup table to replace the calculation process of the reversal computation function in the program to reduce the memory access overhead. Then, according to the characteristics of the mixed-radix FFT program and the hardware structure of Sunway embedded processor, the manual optimization of Sunway processor is carried out from the following three aspects. Firstly, considering that the butterfly operation of each loop in the inner layer of the program is independent of each other, the method of loop unrolling is adopted to optimize, so as to obtain more opportunities of instruction level parallelism. And then it cooperates with Sunway processor's support for the vector components to achieve parallel processing of data. Finally, by inserting prefetch instructions, the cache miss in the loop can be reduced.

3.1. Lookup Table Optimization

Reading data directly from memory is usually much faster than computing with complex data, so using the lookup table method can often get a high performance improvement[8,9].

This method mainly aims at rearranging the output order in the source program, and uses the lookup table to replace the existing calculation process. The storage location information of the final calculation result in the source program is completed by a reversal computation function and a shift operation. After analyzing the program, it is found that each time the FFT is calculated, when the number of input sequences is fixed, the position information of the reordering in each cycle is also fixed. Then through the above functions and shift operations, the position information after each iteration is obtained, and a lookup table is created. All the position information is stored in the table according to the order of iteration. When the subsequent program needs to get the information, the corresponding position information can be obtained by directly looking up the table without recalculation. Compared with the information obtained from each recalculation in the loop, the lookup table method can provide better performance.

3.2. Loop Unrolling Optimization

Loop unrolling is a commonly used method to optimize program performance. It can reduce loop overhead by copying loop body code many times and reducing cycle number of loop branch instructions[10]. Loop unrolling is beneficial to the current processors to improve instruction level parallelism, register locality and hierarchical storage locality[11]. At the same time, loop unrolling is also a necessary means to efficiently develop the hardware characteristics of the processor, such as exploring the opportunity to generate dual instructions, or offsetting the cost of a single prefetch instruction by multiple load / store instructions.

In radix 2-radix 4 mixed-radix FFT algorithm, the input sequence points that need to be loaded for butterfly operation in each cycle are discontinuous in the storage space, which will reduce the access efficiency and affect the running speed of the program. By analyzing the butterfly operation rule of FFT between two cycles, it is found that the butterfly operation of each loop is independent of each other, and the input sequence points required for butterfly operation between two loop are adjacent, and the input sequence is complex. When a single precision complex number is stored, it is stored as two single precision floating-point numbers. After loop expansion, the two adjacent groups of data are processed in one loop for butterfly operations, and the operations of these two groups of data are independent of each other, which can be processed in parallel, that is, four consecutive stored floating-point numbers are operated at one time, which is conducive to making full use of its vectorization components for single instruction multi data optimization on the Sunway platform, reducing the cost of access and storage of reading data from discontinuous storage space, and improving the efficiency of access.

The following is the pseudo-instruction representation before and after the loop expanded:

```
N: input data scale
x [n]: input sequence
Source program:
For i from 0 to N
get x[0],x[N/4],x[2N/4],x[3N/4];
Butterfly operation;
i = i+4;
End for
Loop unrolling program:
For i from 0 to N
get x[0],x[N/4],x[2N/4],x[3N/4];
get x[1],x[N/4+1],x[2N/4+1],x[3N/4+1];
Butterfly operation;
i = i+8;
End for
```

3.3. Manual Vectorization Optimization

Vectorization based on single instruction multiple data (SIMD) is one of the important methods to achieve program parallelism[12]. The SIMD extended instruction can load the continuous address data in memory into vector register at one time, and process all data elements in vector register in parallel through a vector instruction[13]. Sunway processor supports SIMD extension technology[14]. Its extended instruction system is set with 32 256-bit vector registers, which are shared with floating-point registers. Each vector register can store 4 32-bit single precision floating-point numbers. Therefore, during the instruction execution, an instruction can process four floating-point data, which can effectively improve the code parallel execution ability. At the same time, according to the two characteristics of FFT computing: 1) memory access is discontinuous butterfly computing, which leads to low efficiency of memory access; 2) independent butterfly computing, which can be processed in parallel. Using manual vector optimization can effectively develop programs in parallel, complete multiple butterfly computing at one time, greatly improve performance, and also can sufficiently promote the utilization rate of cache lines, reduce cache misses.

The butterfly operation in the mixed radix FFT algorithm is rewritten by SIMD vector instruction of Sunway. The access operation of four consecutive single precision floating-point numbers is mainly realized by simd_load() and simd_store() instructions. The four consecutive single precision floating-point numbers are stored in vector register for inter-vector operation. Because of the complexity of butterfly operation of the mixed-radix FFT, it is necessary to recombine the vector elements in the operation between vectors, but the Sunway processor does not support shuffle instruction, so this paper chooses simd_vsll*(), simd_vsrl*(), simd_vseleq() and other instructions to shift and select the vector elements, so as to achieve shuffle and rearrange the elements between vectors.

3.4. Data Prefetch Optimization

Data prefetching refers to opportunely loading data from memory into the cache before it is used. It is a mechanism to prepare data for the processor in advance to avoid cache misses[15,16]. The software prefetch of Sunway processor is implemented by inserting prefetch instructions. The compiler or optimizer loads the data in memory into the cache in advance, so that the process of fetching data and using data can be executed in parallel to the greatest extent, and the processor pipeline will not be stagnated due to waiting for processing data, and it is sufficiently to hide the access delay. By analyzing the instruction pipeline of the mixed-radix FFT program and delay period of Sunway instructions, it inserts the __builtin_ prefetch() function into the appropriate position in the program to ensure that the next data to be used is loaded into the cache, and the prefetched cache line will not be replaced before use.

4. Results

After the optimization of the FFT algorithm, the performance test and analysis of the mixed-radix FFT algorithm on the Sunway platform are performed. In this paper, it takes Sunway 221 development board as the experimental platform. It compiles the mixed-radix FFT program with sw5cc compiler to the object file on Sunway Taihu light platform[17]. And it writes the test programs calling the FFT algorithm by linking the object file on Sunway 221 development board.

It writes the test program to test 64-point, 128-point, 256-point, 512-point, 1024-point, and 2048-point FFT calculations, the performance of the optimized program on input sequences with different size is accelerated. Because the single FFT operation process is in the microsecond (us) level, it is not easy to observe and compare, and it is easy to have accidental errors. The time of 100000 FFT calculation programs are called as the test reference time. The calculation formula of the speed-up ratio is as follows: time before optimization / time after optimization. The results are shown in Table 1. And the unit time is millisecond(ms).

Table 1: Optimization result of different Points.

Input points	Time before optimization(ms)	Time after optimizaion(ms)	Speed-up ratio
64	220	67	3.28
128	471	137	3.44
256	970	340	2.85
512	2111	699	3.02
1024	4052	1351	3.00
2048	9846	3477	2.83
	Average acceleration ratio		3.07

Experimental results show that the performance of the FFT optimization program can improve by 3.07 times on average compared with the source program when processing 2 integer power input sequence points.

It takes the FFT calculation performance of 1024 complex sequence points as the test result to analyze the performance. The performance results of the four optimization methods are shown in Figure 2. The calculation formula of the speed-up ratio is as follows: running time of the basic program / running time after optimization.

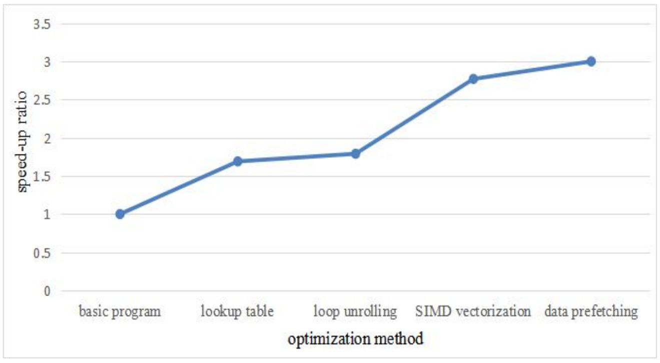

Figure 2: Performance optimization speed-up ratio of mixed-radix program.

As can be seen from the above Figure 2, the test program running on the Sunway 221 development board to call the radix 2-radix 4 mixed-radix FFT algorithm, and compute the FFT of 1024 complex points 100000 times has been improved by three times. Among them, the performance of lookup table optimization is improved by 1.69 times by using the lookup table method to optimize the reversal computation function, and the performance of loop unrolling optimization is improved by 1.79 times, the performance of SIMD vectorization optimization is improved by 2.77 times, Finally, by implementing the data prefetch optimization, its performance is three times higher than the basic algorithm on Sunway.

5. Conclusions

This paper focuses on the implementation and optimization of the lookup table of radix 2-radix 4 mixed-radix algorithm on the Sunway platform for the localization of embedded devices. The method is beneficial to the real-time data processing of FFT processor. The improved algorithm is optimized for Sunway platform through the use of lookup table to optimize the algorithm calculation strategy, besides, it uses loop unrolling and SIMD vectorization to explore and utilize the parallelism at all levels in the processor core, and uses data prefetching and other memory optimizations. The characteristics of Sunway processor hardware are explored. Finally, the performance of the implemented mixed-radix FFT algorithm is improved by about three times compared with the original algorithm.

References

[1] Li Zhou,Yuxing Wei,Yun Xu.Performance Analysis of Code Optimization Based on TMS320C6678 Multi-core DSP[J].2015,4(2):35-39.

[2] B.N.Mohapatra and R.K.Mohapatra, FFT and sparse FFT techniques and applications,2017 Fourteenth International Conference on Wireless and Optical Communications Networks (WOCN),Mumbai,2017,pp. 1-5.

[3] Information on http://www.swcpu.cn.

[4] S. K. Mali and M. C. Lakkannavar, "Parallel pipelined FFT architecture for real valued signals using radix-2," 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, 2016, pp. 1277-1281.

[5] S. Niharika, A. S. Sali, V. Nithin, S. Sivanantham and K. Sivasankaran, "Implementation of radix-4 butterfly structure to prevent arithmetic overflow," 2015 Online International Conference on Green Engineering and Technologies (IC-GET), Coimbatore, 2015, pp. 1-5.

[6] R.H.Neuenfeld,M.B.Fonseca,E.A.C.da Costa and J.P. Oses,Exploiting addition schemes for the improvement of optimized radix-2 and radix-4 fft butterflies,2017 IEEE 8th Latin American Symposium on Circuits & Systems (LASCAS),Bariloche,2017,pp. 1-4.

[7] R. Neuenfeld, M. Fonseca and E. Costa, "Design of optimized radix-2 and radix-4 butterflies from FFT with decimation in time," 2016 IEEE 7th Latin American Symposium on Circuits & Systems (LASCAS), Florianopolis, 2016, pp. 171-174.

[8] S. K. Sahoo, P. K. Meher and G. G. Ganjikunta, "Multichannel Filters for Wireless Networks: Lookup-Table-Based Efficient Implementation," in IEEE Consumer Electronics Magazine, vol. 8, no. 3, pp. 44-49, May 2019.

[9] S. Hsiao, K. Chen and Y. Chen, "Optimization of Lookup Table Size in Table-Bound Design of Function Computation," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, 2018, pp. 1-4.

[10] P. R. Panda, N. Sharma, S. Kurra, K. A. Bhartia and N. K. Singh, "Exploration of Loop Unroll Factors in High Level Synthesis," 2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID), Pune, 2018, pp. 465-466.

[11] G. Velkoski, M. Gusev and S. Ristov, "The performance impact analysis of loop unrolling," 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, 2014, pp. 307-312.

[12] H. Inoue, "How SIMD width affects energy efficiency: A case study on sorting," 2016 IEEE Symposium in Low-Power and High-Speed Chips (COOL CHIPS XIX), Yokohama, 2016, pp. 1-3.

[13] L. Huang, Z. Wang, N. Xiao and Q. Dou, "Efficient Utilization of SIMD Engines for General-Purpose Processors," in The Computer Journal, vol. 57, no. 8, pp. 1141-1154, Aug. 2014.

[14] M. Zhao, R. Liu, Y. Liu, K. Song and D. Qian, "Parallel Image Processing on the Sunway Many-Core Processor," 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Sydney, NSW, 2016, pp. 679-686.

[15] D. Jo, S. Lee, K. Min and Y. H. Song, "Enhanced rolling cache architecture with prefetch," 2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Seoul, 2016, pp. 1-3.

[16] N. Charmchi, C. Collange and A. Seznec, "Compressed Cache Layout Aware Prefetching," 2019 31st International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), Campo Grande, Brazil, 2019, pp. 25-28.

[17] R. Liu, Y. Liu, M. Zhao, K. Song and D. Qian, "SunwayImg: A Parallel Image Processing Library for the Sunway Many-Core Processor," in IEEE Access, vol. 7, pp. 128555-128569, 2019.