Journal of Computer Science 4 (7): 545-549, 2008
ISSN 1549-3636

Implementation of Low-Memory Reference FFT on Digital Signal Processor

Yi-Pin Hsu and Shin-Yu Lin
Department of Electrical and Control Engineering,
National Chiao Tung University, P.O. Box 25-92 Hsinchu, Hsinchu City, 30099 Taiwan (R.O.C)

Abstract: Problem statement: In order to improve and implement Fast Fourier Transform (FFT), in general, an efficient parallel form in digital signal processor is necessary. The butterfly structure is an important role in FFT, because its symmetry form is suitable for hardware implementation. Although it can perform a symmetric structure, the performance will be reduced under the data-dependent flow characteristic. Even though recent research which calls as Novel Memory Reference Reduction Methods (NMRRM) for FFT focus on reduce memory reference in twiddle factor, the data-dependent property still exists. **Approach:**In this study, we propose an approach for FFT implementation on DSP from analog device company (ADI) which is based on data-independent property and still hold the property of low-memory reference. We have applied the proposed method of radix-2 FFT algorithm in low-memory reference on ADI BlackFin561 DSP. **Results**: Experimental results show the method can reduce 44.36% clock cycles comparing with the NMRRM FFT implementation and keep the low-memory reference property. **Conclusions/Recommendations:** From our algorithm, the results can be accepted and realized for DSP-based embedded system. In further, we will try to implement on different DSP-based system in order to improve the algorithm values.

Key words: TI DSP, ADI BlackFin561 DSP, FFT

INTRODUCTION

The signal processing plays an important role for audio coding, image compression and video processing in real application. Especially, data domain transformation is an essential step for above application. The discrete Fourier transform (DFT) is main and important procedure in the data analysis, system design and implementation[1]. The DFT formula can be expressed by:

$$X[k] = \sum_{n=0}^{N-1} x[n] W_N^{nK} \quad k = 0,1,.....,N-1$$

where, $W_N^{nk} = e^{-j(2\pi/N)nk}$ is twiddle factor and x[n], X[k] are temporal and frequency signal individually. Obviously, the formula is inefficient in hardware system if programmer directly implement it.

In order to solve the drawback, many fast Fourier transforms (FFT) are proposed and implemented on different platforms. To reduce the complexity computation is central ideal on most of FFT algorithms. These algorithms focus on twiddle factor or radix order to perform a simply and efficient algorithm which includes the higher radix FFT[2], the mixed-radix FFT[3], the prime-factor FFT[4], the recursive FFT[5] and low-memory reference FFT[6]. In general, one is application-specific integrated circuits (ASIC) system such as[7-8]. The ASIC-based system can fit real application for low-power or high performance; however, it is very hard to modify the function. In pattern recognition system, for example, a new symbol is received from input device then the system can not adjust and process directly when a database is set up completely. Thus the flexibility is not enough. Furthermore another is digital signal processing (DSP) system such as[9] which can achieve wide-range design by software. For instance, Texas Instrument (TI), Analogy Device Company (ADI) and Motorola provide a lot of different DSP chip and its relative evaluation board for industry and school usage. Although DSP-based system also keeps a high enough performance, the result is lower than ASIC-based system. However the DSP still achieve a performance of fast enough for real world application. Today, commercial product has a lot of considerations such as cost, performance, flexibility and convenience implementation. Thus the DSP-based system becomes a favorable solution.

Corresponding Author: Yi-Pin Hsu, Department of Electrical and Control Engineering, National Chiao Tung University, P.O. Box 25-92 Hsinchu, Hsinchu City, 30099 Taiwan (R.O.C) Tel: +886-9-39902564

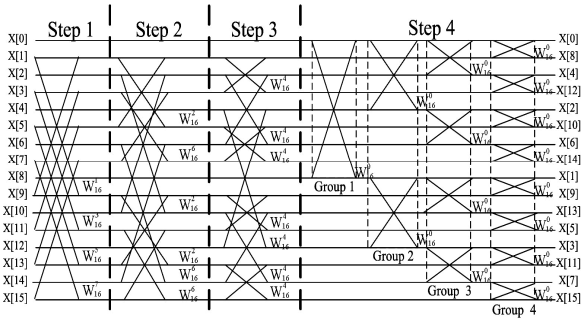

Fig. 1: A structure of NMRRM in 16-pt radix-2 DIF FFT diagram

Nevertheless, an efficient FFT algorithm be implemented on DSP is very difficult. Although TI[10] proposes an efficient FFT algorithm for C64x DSP system, data-dependent condition leads to increase clock cycle. In the research[6], in order to avoid multi time swap in the same data, a reorder FFT structure is proved. In Fig. 1, a 16-pts radix-2 decimation-in-frequency (DIF) FFT structure, each twiddle factor is fetched only once. For example, the twiddle factor of W_{16}^{0} in the step 4 is selected to execute butterfly; but W_{16}^{0} disappear in the other step. Hence, memory reduction method would reduce total memory usage and decrease power consumption as well.

By careful observation, the final step (i.e., step 4) has a serious problem for dual core execution in Fig. 1. First, the step 4 can divides into four groups. Because between each group has data-dependent relation, a series of flow is performed. Even if NMRRM FFT algorithm can improve the data usage, the problem of series process is still existence. The drawback leads to increase the clock cycle and waste hardware resource if the DSP has numerous operation units. Hence, how to hold the low-memory reference property algorithm for dual core system becomes an important issue.

In this study, we propose a novel algorithm to fit dual core processor from NMRRM FFT algorithm. Based on hardware representation, we first build two blocks for dual core operation. Afterward, we modify the data flow in the final two steps to perform a parallel-processing flow. In implementation part, the ADI BlackFin561 DSP is used as verification target. Experimental results demonstrate that the parallel-processing flow can dramatically reduce clock cycles.

The rest of this study is organized as follows. In second section, we will brief introduce NMRRM FFT algorithm and express a method of dual core flow. The experimental results are shown on third section. Finally, some useful conclusions are demonstrated.

MATERIALS AND METHODS

Review of novel memory reduction methods: In this section, we will brief introduce NMRRM FFT structure which is based on TI's library. Afterward we will show that the implementation of radix-2 DIF FFT from NMRRM FFT as a main template for FFT algorithm on DSP.

In embedded system, the memory access is a key point on performance expression. A large number of clock cycles will be increased if memory swap is frequently executed. In Fig. 1, NMRRM FFT algorithm is proposed in order to reduce the memory swap, express a simple and an efficient architecture. We can clear observe that each twiddle factor is used only once on each step; it is not referenced in the other steps. The data-independent property is main ideas to perform this structure. For example, the calculation of twiddle factor W_{16}^{0} in the step 1, 2 and 3 can be moved to the step 4 and no influence in the result. The same skill also is applied to calculate the twiddle factor of W_{16}^{4} which is grouped into the step 3. Although this skill reduces the memory reference, the data-dependent effect is performed in the final step and can not run on pipeline structure. For instance, in Fig. 1, each group from 1 to 4 has mutual dependence property. The part 2 can not be executed if part 1 runs processing. For this reason, NMRRM FFT algorithm has data-dependent status.

A dual core consideration of FFT on DSP: The traditional FFT has a parallel structure which is butterfly for DSP processing, but its drawback is multi-time reference in the same twiddle factor. The revised FFT, NMRRM FFT, can successfully avoid above fault but it performs a data-dependent problem in final step. In order to keep the advantage and discard the disadvantage between both of two different structures, we modify NMRRM FFT to fit dial core approach for hardware implementation. The twiddle factor will be taken only once and parallel-processing in each step. In order to analyze the data flow, in Fig. 2, we redraw NMRRM FFT structure which is based on 16-pt radix-2 FFT algorithm.

Algorithm and hardware analysis: Based on Fig. 2, from group 9 to group 12 in the step 4, each group result will affect the next and later group. For example, we can not skip the group 10 and direct to compute the group 11. Moreover, in the step 3, all groups are independently on computing process. If we change group 6 and group 8, the result is unchangeable. In further, three pairs of the group 6 with group 10, group 7 with group 11 and group 8 with group 12 has relation respectively. For example, the group 8 will be computed in first if we want to execute the group 12.

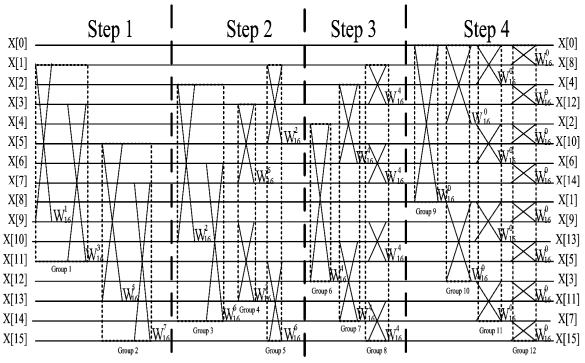

Fig. 2: Partitioning of NMRRM in16-pt radix-2 DIF FFT diagram

Based on this constriction, thus, we can technologically combine step 3 and step 4 into a new step. Due to the step 3 and step 4 only use one twiddle factor respectively, the both twiddle factors are different, the new step needs two twiddle factors. Under the new group, the memory reference and usage equals the steps 3 and step 4 in NMRRM algorithm.

Each group has even element of twiddle factor from the group 1 to group 12 except group 6 and group 9 in Fig. 2. Furthermore, the numbers of twiddle factor are unbalanced in each group. The structure is inefficiently to directly implement on DSP system. Especially, the redundant hardware resource will be exhibited if the DSP system has numerous operation units. For example, we assume that a processor has four operation units which can simultaneously run per clock cycle including two additions and subtractions respectively; however, only one addition and one subtraction are required in the group 6. Thus the remainder operation units will be wasted. Even if dual core processor is used, the hardware resource of processor still locates on idle status when sequence algorithm is running in the group 12.

Notices of ADI DSP implementation: From above analysis, based on computational order, we can arbitrarily move the twiddle factor to perform a new structure which includes different steps and groups. Afterward, only final two steps need to modify. Because the step 1 and step 2 have even number of twiddle factor in group for symmetric parallel design. Thus we propose a parallel processing flow which depends on NMRRM FFT algorithm, take 16-pt FFT as example and it can be separated into two main flows in Fig. 3. The program of Core A is in left part and the variable I, L are its data access index; the program of core B use J and K as its variable in the right part.

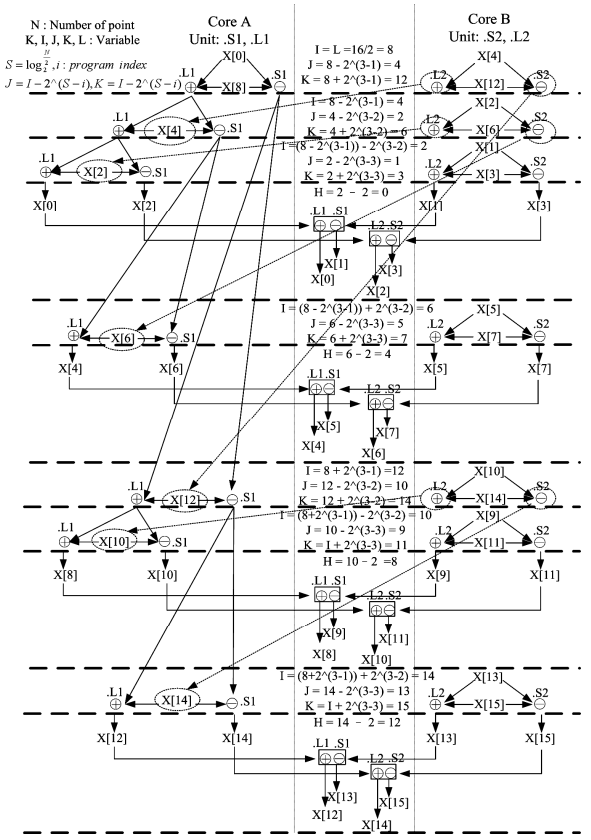

Fig. 3: The system of dual core computing approach after combine the step 3 and 4 in 16-pt radix-2 DIF NMRRM FFT diagram

In vertical direction, the main goal is to explain that the computation core is independent and how many multiplier/ accumulator units are needed. Each stage provides output for next is also independent in the horizontal direction. The starting points are x[0] and x[L], by pre-calculate the index and following the formula set:

$$\begin{cases} I = L = N / 2 \\ S = \log_2^{(N/2)} \\ J = I - 2 \wedge (S - i) \\ K = I + 2 \wedge (S + i) \end{cases}$$

where, N indicates N points in FFT and i presents program index.

It can provide for left and right parts to compute butterflies firstly. This output is prepared for butterflies in the next stage. Following on this order, the left and right part can simultaneously process butterfly when these index are pre-decided.

In order to improve processing speed, most of DSP has numerous operation units. For example, TI TMSC320C64x[11] series includes two register files which has eight parallel calculation units totally. Due to low price consideration, in the ADI BlackFin561, only two multipliers in each core are supported[12]. In Fig. 3, we assume that .L1 and .S1 are labeled as multiplier/subtraction in core A and .L2 and .S2 are labeled as multiplier/subtraction in core B. Beside, each core memory space provides automatically 64K byte in level 1 as cache memory for data operation and 128K byte for programmable cache/RAM in level 2. Although level 2 can provides large enough memory, the 1024-pts FFT based on our new algorithm only use level 1 cache to allocate essential memory. Thus an efficient approach to allocate unit becomes an important work.

Due to the butterfly needs one addition and subtraction in radix-2, it will use two units. Namely, the maximum usage of butterfly is bounded on two units. In Fig. 3, we apply our approach in the DSP system and take 16-pt FFT as example. It can be clear obtained that two parallel-processing flows are executed in two register files and no delay slot occurs. The unit of .L1 and .L2 can operate addition instruction in respective core, as well as the units S1 and .S2 will operate subtraction instruction. In detail, we treat each dotted line in horizontal direction as independent stage. It means that every stage will be finished per cycle. After, these units are uniformly separated to perform a balanced computation.

A parallel computing DSP-based DIF FFT algorithm is evidently depicted in Fig. 3. Besides, a parallel form is integrated in Fig. 4 which is a 16-pt DIF FFT structure. The step 3 and step 4 are combined into one step, namely new step 3 and the stage 3 is split as group from 1 to 11. In further, the step 1 and step 2 has equal properties which also can be separated into the same mode. Each group includes two twiddle factors which are mutual independence. Based on this reason, the parallel-processing can be realized in dual-core system.

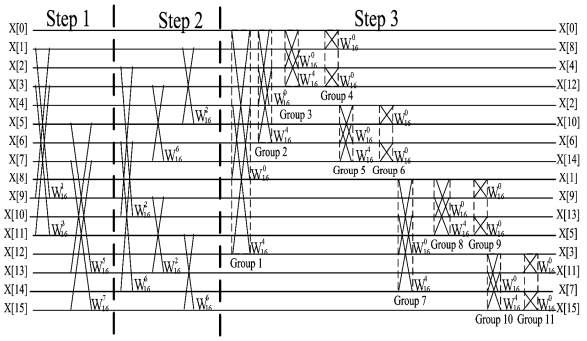

Fig. 4: Integrating NMRRM and our approach in 16-pt radix-2 DIF FFT diagram

RESULTS

The development environment is based on Visual DSP++ 4.0 IDE which is supported by Analogy Device Company. The IDE support a lot of function which includes compiler, assembler and linker. Beside, the IDE also provides two interfaces such as JTAG and USB to connect PC and target. The execution file can be downloaded through two interfaces and directly run on target. In further, the momentary result can be transmitted to PC in immediately. Thus the status of algorithm can be analyzed by user. In the hardware specification, the maximum processing speed of ADSP-BF561 can arrive on 600M Hz for each core. Although the filter is floating type and processor is fixed type, the IDE can transfer float to integer for processor operation.

The comparison item only includes number of clock cycles, in Table 1 with different FFT size, due to the memory reference is equal and NMRRM FFT has better performance than TI's library. The clock cycles are measured by ADI's development environment which also provides a suitable interface and build-in library for programmer debug such as print function.

The experimental results show that our approach has lower clock cycles than NMRRM FFT in radix-2 DIF FFT and average of 44.36% reduction in the number of clock cycles, in addition, the approach also keep low-memory reference property.

Table 1: Comparison of reduction clock cycles in radix-2 DIF FFT

FFT size	16	32	64	128	256	512	1024
NMRRM (C1)	929	1929	3929	7929	15929	31929	63929
Our approach (C2)	476	1024	2117	4366	9042	18952	40338
Reduction ratio ((C1-C2)/C1)×100%	48.76%	46.92%	46.12%	44.94%	43.24%	40.64%	39.90%

DISCUSSION

In this study, a parallel-computing FFT approach on dual core DSP is proposed. The approach is based on low-memory reference property to perform two parallel flows. The performance still equals to NMRRM FFT on radix-2 model which has better performance than ADI's library. ADI BlackFin561 DSP is taken as verification system which has multiple multiply-accumulate units is very suitable for our algorithm. The experimental results demonstrate that our approach can efficiently reduce 44.36% clock cycles and hold the low-memory reference property. Thus no any bottleneck is considered when the algorithm applies on DSP-based embedded system.

CONCLUSION

A specified and efficient DSP-based FFT algorithm have been proved and verified. From the mathematical analysis the algorithm performance is possible to be reached. Through the real emulation the algorithm can reduce 44.36% clock cycles. Moreover in order to fit general application, the ADI BlackFin561 DSP is common and low price DSP is selected as test system. From theory and implementation, our approach is suitable and reliable for DSP system.

REFERENCES

1. Oppenheim, A.V. and C.M. Rader, 1990. Discrete-Time Signal Processing. 2nd Edn., Prentice-Hall, Upper Saddle River, New Jersey, ISBN-10: 0130847127, pp: 24-26.
2. Bergland, G.D., 1969. A radix-eight fast-Fourier transform subroutine for real-valued series. IEEE Trans. Audio Elect., 2: 138-144. URL: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1162043.
3. Singleton, R.C., 1969. An algorithm for computing the mixed radix fast Fourier transform. IEEE Trans. Audio Elect., 2: 93-103.
4. Kolba, D.P. and T.W. Parks, 1977. A prime factor FFT algorithm using high-speed convolution. IEEE Trans. Acoust. Speech Signal Process, 4: 281-294.
5. Varkonyi-Koczy, A.R., 1995. A recursive fast fourier transform algorithm. IEEE Trans. Circuits Syst. II, 42: 614-616. DOI: 10.1109/82.466641.
6. Wang, Y., Y. Tang, Y. Jiang, J.G. Chung, S.S. Song and M.S. Lim, 2007. Novel memory reference reduction methods for FFT implementation on DSP processors. IEEE Trans. Signal Process., 55: 2338-2349. DOI: 10.1109/TSP.2007.892722.
7. Zhou, Y., J.M. Noras and S.J. Shephend, 2007. Novel design of multiplier-less FFT processors. Signal Process., 87: 1402-1407. DOI: 10.1016/j.sigpro.2006.12.004
8. Baas, B.M., 1999. A low-power, high-performance, 1024-point FFT processor. IEEE J. Solid-State Circuits, 34: 380-387. DOI: 10.1109/4.748190
9. He, L. and X. Liao, 2006. Specialising for high performance FFT algorithms based on fixed-point DSP. Proceeding of the IEEE International Conference Communication, Circuits and Systems, June 2006, pp: 563-566, Guilin. DOI: 10.1109/ICCCAS.2006.284699
10. Online available, TMS320C64 DSP Library Programmer's Reference (Rev. G), Texas Instrument, Oct., 2003, SPRU7198G. URL: http://focus.ti.com/general/docs/techdocsabstract.tsp?abstractName=spru565b.
11. Online available, TMS320C64/C64x+ DSP CPU and Instruction Set Reference Guide, Texas Instrument, Aug., 2006, SPRU732C. URL: http://focus.ti.com/general/docs/techdocsabstract.tsp?abstractName=spru732h.
12. Online available, ADSP-BF561 Blackfin Processor Hardware Reference, Revision 1.1, Analog Devices, Feb., 2007, Part Number 82-00561-01.URL: