ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.13, No.13, 1–12

An ultra-long FFT architecture implemented in a reconfigurable application specified processor

Feng Han[1], Li Li[1a)], Kun Wang[1], Fan Feng[1], Hongbing Pan[1], Baoning Zhang[1], Guoqiang He[2], and Jun Lin[1]

[1] *School of Electronic Science and Engineering, Nanjing University, Nanjing 210023, China*

[2] *Nanjing Research Institute of Electronic Technology, Nanjing 210093, China*

a) *lili@nju.edu.cn*

Abstract: This paper presents an efficient architecture for performing 128 points to 1M points Fast Fourier Transformation (FFT) based on mixed radix-2/4/8 butterfly unit. The proposed FFT architecture reduces the computation cost by taking the advantage of the radix-8 FFT algorithm while remaining compatible with sequences whose data length is an integral power of 2. Further optimizations for reconfigurable application specified processor are developed. First, we propose a separated radix-2/4/8 butterfly unit which is more flexible than an entire radix-2/4/8 butterfly unit; second, for the sequences longer than 256K points, an efficient 2-epoch FFT solution is realized. This FFT architecture is implemented in a reconfigurable application specified processor. The computation time of our architecture is 676 us and 14.8 ms for 128K and 1M points FFTs respectively.

Keywords: FFT architecture, long length FFT, mixed radix butterfly unit, radix-2/4/8, 2-epoch FFT

Classification: Integrated circuits

References

[1] K. Natroshvili, *et al.*: "Focusing of general bistatic SAR configuration data with 2-D inverse scaled FFT," IEEE Trans. Geosci. Remote Sens. **44** (2006) 2718 (DOI: 10.1109/TGRS.2006.872725).

[2] E. Cetin, *et al.*: "An integrated 256-point complex FFT processor for real-time spectrum analysis and measurement," IEEE Instrumentation and Measurement Technology Conference, 1997. IMTC/97. Proc. Sensing, Processing, Networking (1997) 96 (DOI: 10.1109/IMTC.1997.603923).

[3] M. Sang-Chul, *et al.*: "Area-efficient memory-based architecture for FFT processing," Proc. of the 2003 International Symposium on Circuits and Systems. ISCAS '03 (2003) V (DOI: 10.1109/ISCAS.2003.1206198).

[4] M. Zhen-Guo, *et al.*: "A novel memory-based FFT architecture for real-valued signals based on a radix-2 decimation-in-frequency algorithm," IEEE Trans. Circuits Syst. II, Exp. Briefs **62** (2015) 876 (DOI: 10.1109/TCSII.2015.2435522).

[5] K.-H. Chen: "A low-memory-access length-adaptive architecture for 2^n-point

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

FFT," Circuits Syst. Signal Process. **34** (2015) 459 (DOI: 10.1007/s00034-014-9862-x).
[6] S. Y. Lin, *et al.*: "Low-cost FFT processor for DVB-T2 applications," IEEE Trans. Consum. Electron. **56** (2010) 2072 (DOI: 10.1109/TCE.2010.5681074).
[7] V. I. Kelefouras, *et al.*: "A methodology for speeding up fast Fourier transform focusing on memory architecture utilization," IEEE Trans. Signal Process. **59** (2011) 6217 (DOI: 10.1109/TSP.2011.2168525).
[8] L. Yu-Wei, *et al.*: "A dynamic scaling FFT processor for DVB-T applications," IEEE J. Solid-State Circuits **39** (2004) 2005 (DOI: 10.1109/JSSC.2004.835815).
[9] G. Xuan, *et al.*: "Hierarchical design of an application-specific instruction set processor for high-throughput and scalable FFT processing," IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **20** (2012) 551 (DOI: 10.1109/TVLSI.2011.2105512).
[10] S. Kala, *et al.*: "Energy efficient, scalable, and dynamically reconfigurable FFT architecture for OFDM systems," 2014 Fifth International Symposium on Electronic System Design (ISED) (2014) 20 (DOI: 10.1109/ISED.2014.12).
[11] C. He, *et al.*: "A pipelined memory-efficient architecture for ultra-long variable-size FFT processors," International Conference on Computer Science and Information Technology, 2008. ICCSIT '08 (2008) 357 (DOI: 10.1109/iccsit.2008.160).
[12] O. Jung-yeol, *et al.*: "Area and power efficient pipeline FFT algorithm," IEEE Workshop on Signal Processing Systems Design and Implementation, 2005 (2005) 520 (DOI: 10.1109/SIPS.2005.1579923).
[13] P. P. Boopal, *et al.*: "A reconfigurable FFT architecture for variable-length and multi-streaming OFDM standards," 2013 IEEE International Symposium on Circuits and Systems (ISCAS) (2013) 2066 (DOI: 10.1109/ISCAS.2013.6572279).
[14] B. M. Baas: "A low-power, high-performance, 1024-point FFT processor," IEEE J. Solid-State Circuits **34** (1999) 380 (DOI: 10.1109/4.748190).
[15] S. Singh, *et al.*: "Pipelined FFT architectures: a review," 2015 International Conference on Electrical, Electronics, Signals, Communication and Optimization (EESCO) (2015) 1 (DOI: 10.1109/EESCO.2015.7253865).
[16] H. J. Kang, *et al.*: "Low-complexity twiddle factor generation for FFT processor," Electron. Lett. **49** (2013) 1443 (DOI: 10.1049/el.2013.2461).
[17] Y. Seung-Won, *et al.*: "Constant twiddle factor multiplier sharing in multipath delay feedback parallel pipelined FFT processors," Electron. Lett. **50** (2014) 1050 (DOI: 10.1049/el.2014.1186).

1 Introduction

Pulse compression is a signal processing technique commonly used by radar, sonar and echo-graphy to increase the range resolution as well as the signal to noise ratio. As the demand for ultra-wideband radar signal processing increases, pulse compression of ultra-long series becomes a hot topic. Since fast Fourier transform (FFT) and inverse fast Fourier transform (IFFT) are primary calculations in pulse compression, the modern real-time radar systems raise high performance demands for specific ultra-long series [1, 2].

Reconfigurable computing is rapidly emerging as a third computing paradigm, along with hard-coded designs, often in the form of application specific integrated

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

circuits (ASICs), and programmable systems, such as classic processors, digital signal processors (DSPs), and graphic processing units (GPUs). In hard-coded designs, the hardware datapaths and the computing algorithms are both fixed at production. In programmable systems, the hardware datapaths are fixed before production but implement generic primitives with some regular interconnection; algorithms are implemented post-production by freely scheduling operations on the datapaths through software. In the new paradigm of reconfigurable computing, many algorithms of the same category are customized to the specified application area and dynamically configured once the device is used. With the requirement to handle the high density of data and lots of different algorithms based on multiplication and addition computation, digital signal processing calls for more powerful computation capability and more comprehensive supportability for different algorithm used in application specified digital signal processing.

Common VLSI implementation of FFT architectures can be classified into three categories: memory-based architectures [3, 4, 5, 6], cache-based architectures [7, 8, 9] and pipelined architectures [10, 11, 12, 13]. Memory-based architectures generally consist of processing units and memory blocks. Cache-based architectures adopt a data cache to reduce the memory access. Pipelined architectures own the benefits of high data throughput rate and low controller complexity. However, for ultra-long series, pipelined architectures bear the high area cost, and extra off-chip accessing cost will be the determining factor for both memory-based architectures and cached architectures. For the series not very long, some works adopt larger memory which is enough to store all samples to avoid the off-chip accessing during computation periods. Since the memory capacity is limited by area cost, operating frequency and power consumption, the off-chip accessing is inevitable when comes to the ultra-long cases. Our design adopts the similar 2-epoch algorithm as Guan's and Baas's works that split long length FFT into two smaller FFT loops [9, 14]. However, the data length supported by proposed architecture is much longer and we use off-chip accessing instead of storing all data on chip. Moreover, based on the further consideration of system resources and the off-chip accessing bandwidth, the balanced architecture is proposed.

Compared with Guan's application-specified instruction set processor, our work introduces coarser grain instructions and employs a 2-epoch methodology with off-chip accessing for ultra-long FFT. Chen's work can perform all integral power of 2 lengths, while it has further memory requirement for larger than 32K series. Although our work takes more area cost for 32-bit single-precision data format and large on-chip memory to cope with the ultra-long series, the comparison with some other works for 1K and 32K points series shows that our work achieves some performance improvement. Moreover, taken further consideration of computation and off-chip accessing balance, our design keeps well performance of longer series.

Although lots of efforts of FFT processing have been made in software and hardware, providing both flexibility and high throughput is challenging. In this work, we focus on ultra-long series FFT performing and system balancing. Our contribution lies in three aspects. First, we propose an efficient memory-based FFT architecture adopted in a reconfigurable application specified processor (RASP) for radar digital signal processing. The separated radix-2/4/8 butterfly unit which is

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

more flexible than an entire radix-2/4/8 butterfly unit, reduces the compute level by taking the advantage of the radix-8 FFT algorithm while remaining compatible with sequences whose length is integral power of 2. The twiddle factor generation unit makes a good tradeoff between memory and logic cost and balances the utilization of adder and multiplier. Second, for sequences longer than 256K points an efficient 2-epoch FFT performing solution is realized. Taken the computing performance and off-chip accessing performance into account, the 2-epoch solution tries to make their time consumption approach and uses "tick-tock" methodology to overlap the accessing time consumption; Third, this work is implemented by following the standard cell-based IC design flow and coded in hardware description language (HDL), and the heterogeneous SoC with RASP have been fabricated on 40 nm complementary metal-oxide-semiconductor (CMOS) process.

The proposed FFT architecture applied in a taped out RASP for radar applications. Our RASP costs about 2.6 million gates or a core area of 3.7 mm^2 without SRAM in TSMC 40 nm CMOS technique. The computation time of our architecture is 676 us and 14.8 ms for 128K and 1M points FFTs respectively.

The rest of paper is organized as follows: Section 2 introduces the FFT algorithm and the RASP architecture. Section 3 describes the optimization in the proposed architecture. Section 4 presents our FFT design, and Section 5 shows the experimental results. Finally, we make conclusions in Section 6.

2 Backgrounds

2.1 Radix-2/4/8 algorithm

The N-point discrete Fourier transform (DFT) is presented by

$$X(k) = DFT[x(n)] = \sum_{n=0}^{N-1} x(n) W_N^{nk} \quad (k = 0, 1, 2, \cdots, N-1), \tag{1}$$

where

$$W_N^{nk} = e^{-j*\frac{2\pi}{N}*nk}$$

is called twiddle factor.

The basic concept underlying the radix-2 algorithm is the use of symmetry between the twiddle factors W_N^{nk} and $W_N^{nk+N/2}$ ($W_N^{nk} = -W_N^{nk+N/2}$). Another symmetry of twiddle factors ($W_N^{nk+N/4} = -W_N^{nk+3N/4} = -\mathrm{j}W_N^{nk}$) has been utilized to minimize the number of complex multiplications by $\pm j$ in radix-4 and split radix algorithm. Moreover, the symmetry of the twiddle factors in Eq. (2) has been utilized in radix-8 algorithm.

$$\begin{aligned} W_N^{nk+N/8} &= -W^{nk+5N/8} = \frac{\sqrt{2}}{2} \bullet (1 - j) \\ W_N^{nk+3N/8} &= -W^{nk+7N/8} = \frac{\sqrt{2}}{2} \bullet (1 + j) \end{aligned} \tag{2}$$

In general, a higher radix algorithm should be used to reduce the multiplication times in a butterfly computation and the computation stages.

The mixed-radix (MR) FFT refers to performing FFT with butterfly calculations of different radix. To radix-2/4/8 MR algorithm a situation, an input series

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

consisting of $N = 8^k * 2$ or $N = 8^k * 4$ points, arises naturally. In this case, a radix-2 or radix-4 butterfly calculation performs at beginning of transform, while the rest of transform is calculated by radix-8 algorithm. Compared to radix-2 or radix-4, radix-2/4/8 MR algorithm reduces the computation cycles while remaining compatible with series whose length is integral power of 2.

2.2 2-epoch FFT algorithm for ultra-long series

When performing long length FFT, it is an important issue that solves the data interleaving among different stages. Adopting on-chip memory to store all of the data is an effective way to cope with the interleaving. When comes to the ultra-long sequences, it becomes inefficient and hard to implement that stores all data on-chip. Baas proposed a 2-epoch FFT algorithm to break down a long length transforming to 2 epoch, each of which processes a batch of shorter length FFT [14]. There is no data interleaving among a batch of short transforming and the data interleaving between 2 epochs is easy to handle. Taking this advantage, the butterfly processing and memory accessing of different short length FFT can be performed concurrently.

The N-points 2-epoch FFT algorithm is based on the following decomposition, there $N = L * C$, the N-points sequence is presented as:

$$\begin{bmatrix} p_0 & p_1 & \cdots & p_{C-2} & p_{C-1} \\ p_C & p_{C+1} & \cdots & p_{2C-2} & p_{2C-1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ p_{(L-2)C} & p_{(L-2)C+1} & \cdots & p_{(L-2)C-2} & p_{(L-2)C-1} \\ p_{(L-1)C} & p_{(L-1)C+1} & \cdots & p_{LC-2} & p_{LC-1} \end{bmatrix} \tag{3}$$

From the Eq. (1), can easily derive:

$$\begin{aligned} X(k_1,k_0) &= \sum_{n=0}^{N-1} x(n_1,n_0) W_N^{(Cn_1+n_0)(LkL_1+k_0)} \\ &= \sum_{n_0=0}^{C-1}\sum_{n_1=0}^{L-1} x(n_1,n_0) W_N^{Cn_1Lk_1} W_N^{Cn_1k_0} W_N^{n_0Lk_1} W_N^{n_0k_0} \\ &= \sum_{n_0=0}^{C-1}\left[\sum_{n_1=0}^{L-1} x(n_1,n_0) W_L^{n_1k_0}\right] W_C^{n_0k_1} W_N^{n_0k_0} \\ &= \sum_{n_0=0}^{C-1}[X_1(n_1,n_0) W_N^{n_0k_0}] W_C^{n_0k_1} \\ &= \sum_{n_0=0}^{C-1}[X_1'(n_1,n_0)] W_C^{n_0k_1} \\ &= X_2(k_0,k_1) \end{aligned} \tag{4}$$

$$X(k_1,n_0) = \sum_{n_1=0}^{L} x(n_1,n_0) W_L^{k_1n_1} \tag{5}$$

$$X(k_1,n_0) = \sum_{n_1=0}^{L} x(n_1,n_0) W_L^{k_1n_1} \tag{6}$$

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

In the above decomposition, Eq. (5) is an L-points FFT of the column in the matrix Eq. (3) and Eq. (6) is a C-points FFT of the row in the matrix Eq. (3). Then the N-points ultra-long sequence FFT is decomposed into C L-points FFT, L C-points FFT and multiplications of the medial twiddle factors. The procedure of 2-epoch FFT is as follows:

1) Present the N-points sequence into a matrix have L rows and C columns;
2) Perform L-points FFT of C column sequences in the matrix;
3) Multiply the result by the middle twiddle factors.
4) Perform C-points FFT of L row sequences in the matrix;
5) Transpose and output the results.

3 A reconfigurable processor for radar digital signal processing

From the view of reconfigurable computation hardware, we realize an RASP for radar applications. As a large number of applications based on multiplications and additions widely uses in radar digital signal processing, our RASP integrates 17 applications for vector and matrix computation. This section presents the architecture of the RASP.

FFT is one of the key applications integrated in RASP. The FFT instruction operands include source data address, source data length, destination data address, destination data length, twiddle factors data address, twiddle factor data length, series length, input/output order, inverse FFT flag and the number of off-chip access times for 2-epoch FFT. When instructions received from high performance bus, the RASP selects the reconfigurable controller of FFT and dynamically configures the data paths between memory, calculation logics and the reconfigurable controller to achieve the designed function.

The RASP works as a high performance application-specified processor in a heterogeneous system on a chip (SoC). In the SoC two DSPs, an RASP and a memory controller are connected by a 256-bit high performance bus which bases the advanced microcontroller bus architecture (AMBA) 3.0. The RASP employs two kinds of bus interfaces. As a master device, RASP can access the main memory through the directional memory access (DMA) engine, and as a slave device, RASP can be configured, suspended and interrupted by DSPs. The system employs a DDR3, whose max operating frequency is 1.067 GHz, as a main memory.

Fig. 1 shows the overall architecture of the proposed RASP. This architecture consists of a main controller (MC), a reconfigurable controller (RC), a reconfigurable computing array (RCA), a direct memory access (DMA) unit, bus interfaces and memory.

The MC manages the overall controlling of RASP that includes instruction decoding, DMA configuration and RC configuration. The RC manages the procedures of computation and connects the data paths between memory and RCA. The RCA has six reconfigurable processing elements (RPEs), each of which consists of computation logic and reconfigurable data path. RPE1 to RPE4 are isomorphic processing elements, each of which consists of one complex multiplier, one real multiplier and 4 complex adders. RPE5 is used for the conversation between fixed-point and float-point data type, and RPE6 which includes multipliers, adders and a

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

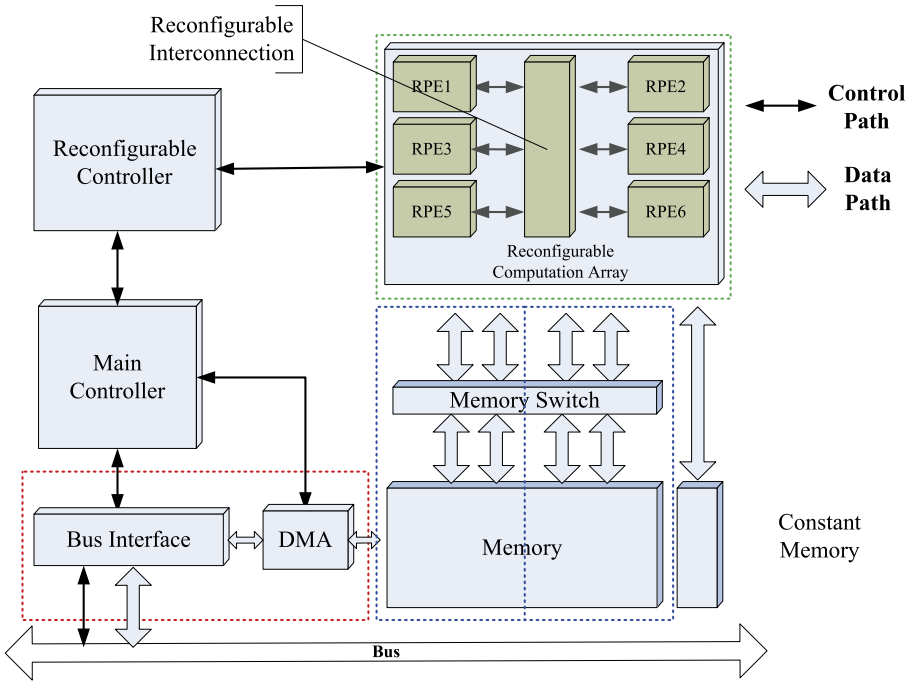

Fig. 1. The overall architecture of RASP

divider, is mainly used for matrix inversion. Employing large on-chip memory is a key feature of the RASP. The proposed RASP uses 2 MB single port Static Random Access Memory (SRAM) as on-chip memory. The memory has 32 banks, each of them consists of eight 64-bit width and 1024 depth SRAM elements.

The correspondence in RASP includes control flow and data flow. Through the control path, shown in Fig. 1, the MC receives the instructions from bus interface and sends the reconfiguration information to the RC and off-chip accessing information to the DMA engine. After the reconfiguration information received, the RC selects one algorithm control module and the connection among the RPEs. The data paths in Fig. 1 are responsible for the data flow in RASP. The DMA is used to read and write data between off-chip memory and on-chip memory. Adapted the memory switch, different banks provide data to appropriate computation logics.

Solved the data interleaving of 2-epoch FFT algorithm, a “tick-tock” processing strategy is used in our architecture. Instead of storing all data on chip, the proposed architecture simultaneously processes butterfly computation and accessing the data to be used in next stage. As Fig. 1 shows, the green dashed frame is responsible for butterfly processing and the red dashed frame is used to off-chip accessing. The on-chip memory is box in blue dashed box. The on-chip memory is divided into two parts and used by butterfly processing and off-chip accessing alternately. Moreover, employed a DMA engine, which has similar throughput with the butterfly units, the off-chip accessing and butterfly processing effectively overlap in the proposed architecture.

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

4 Optimization of the FFT architecture

4.1 Flexible radix-2/4/8 mixed butterfly unit

While the higher radix algorithm reduces the computing stages, it also requires more adders and multipliers compared to radix-2 or radix-4 butterfly units [4, 15]. The increasing utilization of float point units (FPUs) in a complete higher radix butterfly unit (BU) significantly limits the practicability of higher radix algorithm in VLSI design. Moreover, since the sequence length should be an integral power of the radix, traditional higher radix algorithms are not flexible enough. Addressing at these limitations, we propose a flexible pipelined radix-2/4/8 BU. As shown in Fig. 2, it takes a quarter of resources of an entire radix-8 BU to output the radix-8 computation results, 8 samples, in 4 clock cycles. By choosing different datapaths, the BU can perform radix-2, radix-4 or radix-8 butterfly calculations.

The BU consists of registers, multiplexers and FPUs. The registers are used between each pipeline stages in radix-8 algorithm and the multiplexers are used to control the data path between registers and FPUs.

The proposed BU uses in-place accessing method which means the outputs of the BU will be stored in the same memory location as the inputs. Equipping dual-port SRAMs is a straightforward way to avoid the bank conflictions. However, the power consumption and area cost of the dual-port SRAM block is too large to be accepted. In this work we use a two-part memory. For each part memory, 16 single port SRAM banks with a conflict-free algorithm are used to meet the throughput requirement of the BUs.

A twiddle factor generation unit in a VLSI FFT architecture design is usually implemented by look-up-tables or real-time calculation units [16, 17]. By taking the advantage of the circular symmetry in twiddle factors, only the one eighth of the longest supported sequence length coefficients need to be stored, while it still takes considerable memory cost for ultra-long series.

Generating twiddle factors by trigonometric calculating requires high performance pipelined trigonometric units which are barely used in the other applications of RASP except FFT. Using the products of interpolations and the increment is another way to calculate twiddle factors in real time, but the iteration of multiplication may cause a precision reduction. To handle this problem, we adopt a twiddle factor generation unit using the products of interpolation.

$$\begin{aligned} W_{1M}^{k} &= \exp(-2pi * k/1M) \\ &= \exp(-2pi * (m * 1024 + n)/1M) \\ &= \exp(-2pi * m/1024) * \exp(-2pi * n/1M) \\ &= W_{1K}^{m} * W_{1M}^{n} \end{aligned} \tag{7}$$

$$(k = 1, 2, \ldots, 1M; \mathrm{m} = 1, 2, \ldots, 1024; n = 1, 2, \ldots, 1024;)$$

As the Eq. (7) illustrates, the twiddle factors for 1M FFT can be calculated by multiplying two groups of interpolations. Compared with storing all coefficients on chip, it saves near 2 MB storage with a multiplier. Compared with iterated multiplication twiddle factor generation, this method gets higher precision and lower storage cost.

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

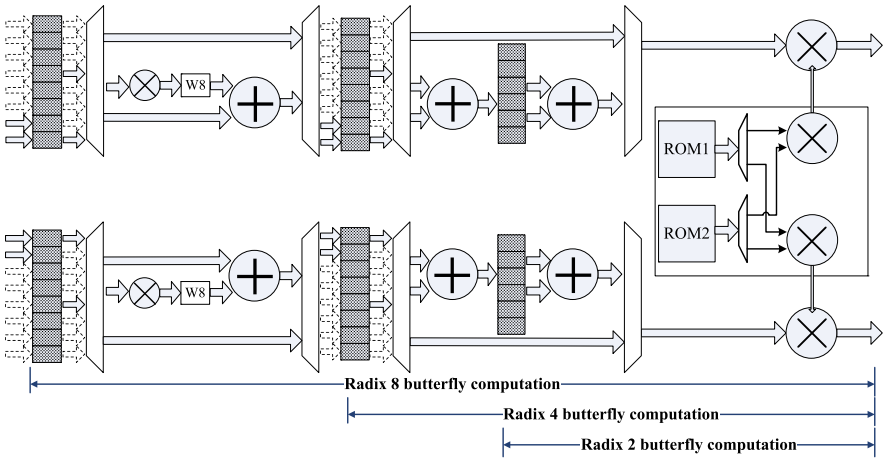

Fig. 2. A radix-2/4/8 butterfly include twiddle factors

4.2 The 2-epoch calculation for ultra-long sequence

The storage resources, one of the most important key points of FFT architectures, include on-chip memory and off-chip memory. On-chip storages are generally implemented by high performance devices, such as SRAMs. While satisfied the performance requirement SRAM devices take higher area cost and power consumption. With high storage density off-chip devices have much lower performance because of the long access latency and conflictions. In order to satisfy the application requirement the proposed architecture employs 2 MB SRAM to guarantee the performance of sequence shorter than or equal to 256K. Nevertheless, the architecture also uses a tick-tock memory management to schedule the memory occupied by the computation stage and the off-chip accessing stage alternately based on the 2-epoch FFT algorithm.

As the Fig. 3 shows, a 512K points FFT is broken down into 4 column computing stages and 4 row computing stages base on the 2-epoch FFT algorithm, and the storages are split into two parts. Most of the time, the two parts of storages are occupied by FPUs and off-chip accessing respectively.

2-epoch method is an efficient way to solve the interdependency problem in FFT, but it also leads extra off-chip accessing for ultra-long series. Off-chip accessing speed, influenced by DDR efficiency, is a key factor to the performance. For 2-epoch processing, the architecture is carefully designed to make the computation and off-chip accessing capability approach. To measure the off-chip accessing bandwidth, a simulation about the interactions of DMA and DDR is carried out. It shows that the accessing reaches 9.7 GB/s for continuous data and 5.2 GB/s for rectangle data accessing. However, it may be lower in a real SoC since the DDR competitions from other devices. The throughput of BUs to calculate 128K data for three stages is 2.67 GB/s. Without off-chip accessing hazards, this architecture will take 392 us to finish the computation of one stage and 309.7 us to finish the off-chip accessing.

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

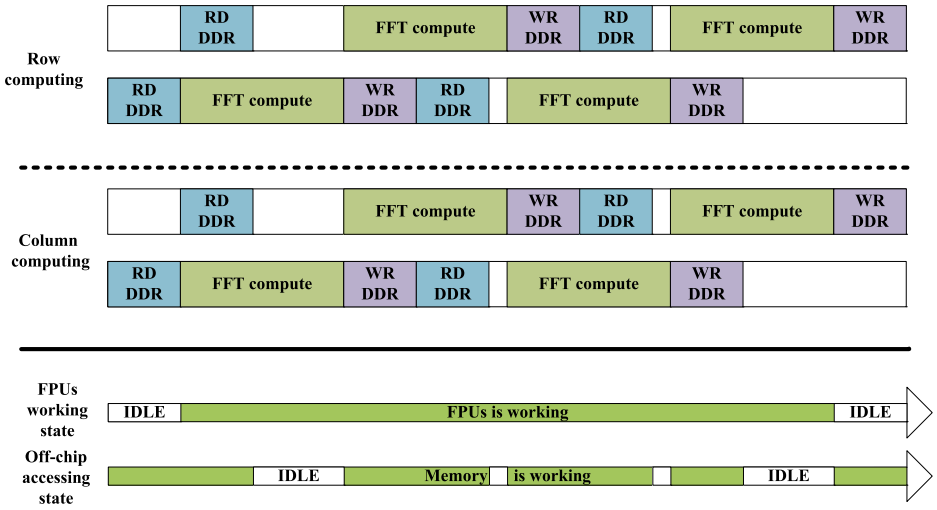

Fig. 3. Example of 512K points FFT processing flow

5 Experimental results

The proposed FFT design has been implemented in a reconfigurable application specified processor by following the standard cell-based IC design flow and coded in HDL. It has been verified through C behavioral simulation by UVM environment, SpyGlass HDL coding rule check, verilog RTL simulation, logic synthesis, verilog gate-level simulation, placement and routing, DRC, LVS, and post-layout simulation. The proposed design has been synthesized using a 40-nm CMOS technology.

The performances from 128 to 1M points are evaluated. Fig. 4 presents the computation, off-chip accessing and configuration cycles, and the cycles saved by tick tock memory accessing in 2-epoch algorithm. Before starting an application, the RASP will take about four thousands cycles to receive all the instructions from bus and decode them. As configuration overhead is influenced by bus efficiency,

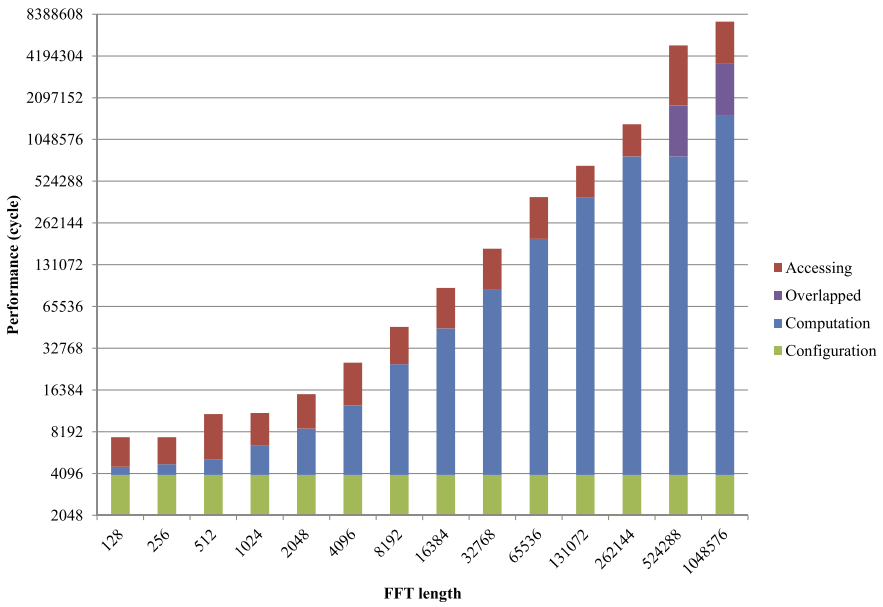

Fig. 4. Example of 512K points FFT processing flow

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

there is some uncertainty in configuration cycles. To the series longer than 256K points the overlaps of accessing and computation are also presented.

With an ideal off-chip memory device, the computation takes more cycles than accessing ones. However, it may get more balanced performance in the SoC, because the off-chip accessing performance will be influenced by DDR hazards.

The proposed RASP processes FFT from 128-points to 256K-points without off-chip accessing on the computing phase, and uses tick-tock off-chip access strategy for the sequences from 512K-points to 1M-points. It completes a 128K-points FFT in 676.5 µs and a 1M points FFT in 14.8 ms.

Table I. Comparison among different FFT designs

Design		Chen	Lin	proposed
Radix		2/4/8/16	2	2/4/8
Max support length without extension		32K	32K	1M
Word length		12 bit ∗ 2	12 bit ∗ 2	32 bit ∗ 2
Technology		180 nm	90 nm	40 nm
BU	# of CM	2	4	2
	# of RM	6	-	2
	Storage	32K ∗ 24 bit	32K ∗ 24 bit	256K ∗ 64 bit
Twiddle	# of CM	-	-	2
	Storage	4K	4K	2K
Area cost		4.49 mm^2	2.51 mm^2	19.2 mm^2
Normalized area		4.9	12.6	5.7
Operating frequency		102 MHz	25 MHz	500 MHz
Computation time (us)	1K	14.98 us	-	5.14 us
	32K	518.04 us	3604.48 us	164.48 us
	1M	-	-	14.8 ms

Table I presents the hardware comparison of this work with 2 different FFT realizations: Chen's low memory access length adaptive FFT architecture for all integral powers of 2 [5]; And a design for long length, Lin's FFT processor, which supports 32K points series [6].

As the proposed architecture, which is designed for ultra-long series and single precision float data, employs float points computing logic and memory, the area cost of our design is larger the others. Based on an effective performance measure proposed in [14], we introduce the factor of word length to normalize area efficiency as Eq. (8).

$$Normalized\ Area = \frac{Area}{(Technology/40\,\mathrm{nm})^2 * FFT\ Sizes * Word\ length} * 10^7 \quad (8)$$

Nevertheless, to make the comparison more reasonable, we compared the computation logic cost in number of multipliers. Our work uses two complex multipliers (CMs) and two real multipliers (RMs) in the BU. For N-points trans-

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016

formation, with two CMs, the proposed architecture reduces the storage of twiddle factors decreased from N/8 words to $N^{1/2}/8$. The supported data length of this work is 128 to 1M points, and can be easily extend with a small mount twiddle factor ROM extension. Compared with the other designs which requiring more complex extensions to support ultra-long FFT, our work is more suitable in radar DSP. Comparing with Chen's work the proposed performance improvement is 2.91× for 1K points FFT. For computation of 32K points, the proposed performance improvement is 3.14× and 21.91× comparing with Chen and Lin's work respectively. Moreover, to the best of our knowledge, there are few of hardware work can perform FFT larger than 128K points without extensions. Taking the advantage of 2-epoch algorithm and tick-tock strategy, our work can keep the good performance for 128K to 1M points computations.

6 Conclusions

In this work, we present an FFT architecture implemented in an application specified reconfigurable processor for ultra-long series. This architecture has been fabricated on 40 nm CMOS process. A radix-2/4/8 BU and a tick-tock memory accessing method are applied to supporting integral power of 2 sequence lengths from 128 to 1M. The twiddle factor generation unit used in this design saves lots of storage resource with multiplier while keeping high precision. The overall performance is greatly improved by parallel computation and tick-tock memory accessing arrangement. By further, using the 2-epoch algorithm and balancing computation and off-chip access the FFT architecture gets a well support for ultra-long series.

Acknowledgments

Thanks for the insightful and detailed comment from Koji Kotani, the reviewer. This work was supported by National Nature Science Foundation of China under Grant No. 61176024, 61370040, 61376075; Research Fund for the Doctoral Program of Higher Education of China under Grant No. 20120091110029; The project on the Integration of Industry, Education and Research of Jiangsu Province BY2015069-05, BY2015069-08; The key Research and Development Program of Jiangsu Province under Grant No: BE2015153; Supported by the Fundamental Research Funds for the Central Universities under Grant No. 021014380030; The Project Funded by the PAPD of Jiangsu Higher Education Institutions (PAPD).

DOI: 10.1587/elex.13.20160504
Received May 18, 2016
Accepted May 25, 2016
Publicized June 14, 2016
Copyedited July 10, 2016