# A SUB-WORD PARALLEL DIGITAL SIGNAL PROCESSOR FOR WIRELESS COMMUNICATION SYSTEMS

*Yuan-Hao Huang and Tzi-Dar Chiueh*

Department of Electrical Engineering and
Graduate Institute of Electronics Engineering
National Taiwan University, Taipei, Taiwan 10617
chiueh@cc.ee.ntu.edu.tw

**ABSTRACT**

In this paper, we propose a programmable fixed-point digital signal processor for wireless communications. The architecture of the processor is designed according to the computation requirements of modern communication systems. A decimation-in-frequency (DIF) butterfly unit is built in the processor to enhance the processing capability of FFT operations needed in orthogonal-frequency-division-multiplexing (OFDM) systems. In addition, the butterfly unit can be reconfigured to accelerate squared-difference and add-compare-select calculation in the Viterbi algorithms. A new sub-word parallel complex-valued multiply-and-accumulate (MAC) architecture is proposed to execute complex/real and single/ double precision operations, making it suitable for different requirements of signal formats in signal processing for communication transceivers.

KEY WORDS
Digital Signal Processor, Subword Parallel MAC, FFT Butterfly

## 1. INTRODUCTION

Programmable digital signal processors (DSP) have been widely used to reduce the development costs and time-to-market in many communication applications. In recent years, some DSP [1, 2, 3] have been designed to perform communication functions in order to reduce required MIPS and processing power. Function-specific hardware and parallelism are both popular and important techniques used to improve the processing capabilities of the digital signal processors [4].

In the proposed DSP, we build a specific decimation-in-frequency butterfly unit that can perform different kinds of butterfly operations in the fast Fourier transform (FFT), which is essential in OFDM systems. Besides, the proposed butterfly unit can be reconfigured to perform squared-difference and add-compare-select operations required in the Viterbi algorithm. As more communication baseband processing tasks are realized on digital signal processors, an MAC architecture should preferably be able to accommodate real/complex convolution as well as single/double precision signal format. Therefore, we have also proposed a flexible sub-word parallel complex MAC architecture to meet different precision requirements in the communication systems.

The remainder of this paper describes the proposed chip in detail. Section 2 discusses the overall architecture of the digital signal processor including the butterfly and the sub-word-parallel complex MAC processor. Section 3 demonstrates the performance evaluation of the processor. Chip design and measured results are shown in Section 4. Finally, Section 5 concludes this paper.

## 2. DIGITAL SIGNAL PROCESSOR ARCHITECTURE

Fig. 1 shows the architecture of the proposed digital signal processor. One program memory and two data memory modules have separate addressing space. The data address generator can generate two data addresses in one cycle to access the two embedded SRAM banks via dual data and dual address buses. This structure can reduce the memory access overhead for the I/Q channel data processing often found in communication applications. Besides, the program sequencer generates program counter address in a way that handles subroutine, interrupt, and conditional branches. The main datapath units are ALU (Arithmetic Logic Unit), shifter, and butterfly/complex MAC processor. Butterfly/complex MAC processor is the main feature of this DSP and its detail structure is introduced in the following.

### 2.1. Butterfly Operation

A decimation-in-frequency butterfly unit, shown in Fig. 2, is proposed in the DSP for efficient execution of the FFT algorithm. Additionally, the proposed unit can also execute other operations such as real-valued and complex-valued addition/subtraction/multiplication. Internal ROM that stores complex-valued twiddle factors can also work as part of a numerically-controlled oscillator (NCO) needed in many frequency/phase recovery algorithms in communication receivers.

### 2.2. Viterbi Algorithm Operation

Convolutional coding is widely used in wireless communication applications. The received signal is decoded by the Viterbi algorithm on the trellis diagram. In this algorithm, the most computation-intensive operation is the add-compare-select (ACS) operation, shown as in Fig. 3(a). The ACS operation can be executed using the complex adder/subtractor in the butterfly unit as shown in Fig. 3(b). The sign bit of output of the subtraction is stored in a 32-bit queue for later backward path traceback.

### 2.3. Sub-Word Parallel Complex MAC Operation

The four real $16 \times 16$ multipliers in the butterfly unit can be flexibly configured to execute in one cycle: (1) one $16 \times 16$ complex-valued

0-7803-7363-4/02/$17.00 ©2002 IEEE.

MAC, or (2) four 16 × 16 real-valued MAC, or (3) four 8 × 8 complex-valued MAC, or (4) sixteen 8 × 8 real-valued MAC.

#### 2.3.1. Double-Precision Complex and Real MAC Modes

The 16×16 complex-valued MAC is composed of four real 16×16 multipliers and two accumulators. The two 40-bit accumulators for real part ($ACCR$) and imaginary part ($ACCI$) together hold the accumulated complex-valued products. To realize real-valued 16 × 16 MAC, we can first examine the following example. In a typical order-$K$ FIR filter, $K$ pairs of 16-bit inputs $X(n)$ and 16-bit coefficients $C(n)$ are multiplied and accumulated as output $Y(n)$. In order to achieve maximum hardware utilization, two consecutive filter coefficients are loaded into the A register, one in the real part and the other in the imaginary part. Similarly, two consecutive input samples are also loaded into the $B$ register (see Fig. 4(b)). In $(K/2)$ cycles, $Y(n)$ can be computed in $ACCR$. In addition, half of the terms in $Y(n-1)$ and $Y(n+1)$ will be accumulated. At the end of each FIR iteration (in this example 3 cycles), $Y(n)$ is in $ACCR$ and $Y(n-1)$ is in $ACC-AUX$. Before the next iteration, content in $ACCI$, which contains partially-accumulated sum of the next output $Y(n+1)$, must be moved to $ACC-AUX$. In this way, four 16 × 16 multipliers operate all the time and a speed-up factor of four can be achieved.

#### 2.3.2. Single-Precision Complex MAC Mode

Each 16 × 16 multiplier in the 16 × 16 complex-valued MAC can be divided into four 8 × 8 multipliers that can produce sub-word products, which can be shifted and summed to obtain a 32-bit product. With two additional accumulators, the four sub-word multipliers can be reconfigured as an 8 × 8 complex-valued MAC. In the operation of an order-$K$ complex-valued FIR filter, the processing schedule is the same as that of the 16 × 16 real-valued MAC except that $C(n)$ and $X(n)$ are now combinations of 8-bit real parts, $CR(n)$ and $XR(n)$, and 8-bit imaginary parts, $CI(n)$ and $XI(n)$. Therefore, four 8 × 8 complex-valued MAC can be configured as shown in Fig. 5. At the end of each iteration, complex accumulators $acc_0$ and $acc_1$ are summed as output $Y(n)$ and complex accumulators $acc_2$ contains output $Y(n-1)$. Before the next iteration, content of $acc_3$, which contains sum of half of the terms in $Y(n+1)$, is moved to $acc_2$. Consequently, on average four 8×8 complex-valued MAC operations are executed per cycle.

#### 2.3.3. Single-Precision Real MAC Mode

To configure 8 × 8 real MAC, let's examine the operation of an order-$K$ filter in Fig. 6, where $C(n)$ and $X(n)$ are 8-bit values, and four consecutive coefficients and four consecutive samples can be loaded into $A$ and $B$ registers respectively. At time step $n$, one can compute $Y(n)$ from present products (rectangle). Moreover, terms in $Y(n-1)$ (rhombus), $Y(n-2)$ (ellipsoid), and $Y(n-3)$ (trapezoid) can also computed from products of the present ($t=n$) and previous ($t=n-4$) iterations. Thus, the MAC can be reconfigured as shown in Fig. 7. Four output samples can be generated in one MAC iteration ($K/4$ cycles). Before the next iteration, three accumulated contents are moved to the corresponding accumulators (bold arrows). Therefore, this MAC configuration can effectively execute 16 8 × 8 real-valued MAC calculations in one cycle.

## 3. PERFORMANCE EVALUATION

Table 1 summarizes performance of some DSP algorithms by the proposed DSP and a conventional DSP. The performance improvement for FIR, IIR, and others results from that the sub-word-parallel complex MAC is designed to execute the convolution efficiently, especially in single-precision format. Other algorithms benefits from the function-specific reconfigurable DIF butterfly processor. In summary, the digital signal processor achieves 3 to 16 times of speed up in these algorithms. Furthermore, the number of instruction cycles needed for some single-precision complex-valued FIR examples and some single-precision real-valued FIR example are listed in Table 2. When the proposed processor is compared with conventional DSP, the speed-up factor increases with the increasing tap-length and is between 3 to 14. The above analysis shows that more gain can be made when the proposed DSP is applied to long-tap-length convolution such as equalizers, match filters, and Nyquist-rate filters.

## 4. CHIP IMPLEMENTATION

This proposed processor is designed and the chip is fabricated using a $0.35\mu m$ CMOS process. Summary of the chip is shown in Table 3 and its die photo is shown in Fig. 8. Two embedded 1k SRAM are included for I/Q channel data operation, and two 1k quadrature cos/sin ROM are designed to store phasors for FFT twiddle factors and loop-up table of NCO. Due to the high efficiency of its Butterfly/complex MAC processor, the DSP chip can achieve a maximum of 68M butterfly operations or 1.1G real-valued, single-precision MAC operations per second.

## 5. CONCLUSION

In this paper, we proposed a digital signal processor that supports butterfly, squared-difference, and add-compare-select, operations. With a special reconfigurable MAC architecture, the proposed DSP can achieve up to 16 time speed up when compared to traditional DSP. Therefore, we believe this processor can greatly improve the computation performance in a wide array of communication applications and can serve as a key components in future software radio communication equipment.


## Acknowledgements

The authors would like to thank the Chip Implementation Center (CIC) in Taiwan, the support from National Science Council, Taiwan, ROC under Grant NSC90-2218-E-002-039, and from Ministry of Education, Taiwan, ROC under Grant 89-E-FA06-2-4.


## 6. REFERENCES


[1] M. Alidina *et al.*, "DSP16000: a high Performance, low power dual-MAC DSP for communication applications," in *Proc. IEEE Custom Integrated Circuits Conference*, 1998, pp. 119–122.

[2] B. W. Kim *et al.*, "MDSP-II: a 16-bit DSP with Mobile Communication Accelerator," *IEEE Journal of Solid-State Circuits*, 34(3), 1999, pp. 397–404.

[3] P. C. Tseng *et al.*, "CDSP: A 16-bit digital signal processor for 3G wireless communications," in *Proc. of 11th VLSI Design/CAD Symposium*, Taiwan, 2000, pp. 311–314.

[4] Lapsley *et al.*, *DSP Processor Fundamentals: Architectures and Features*. IEEE Press, New York, 1996.

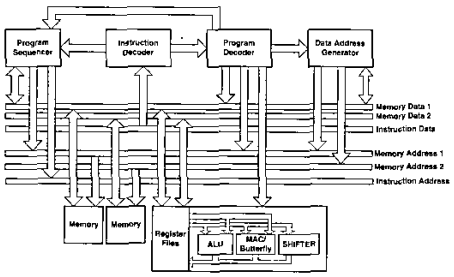



Figure 1: Processor Architecture.

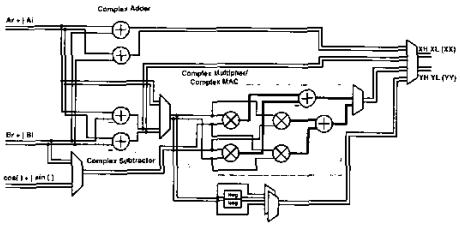
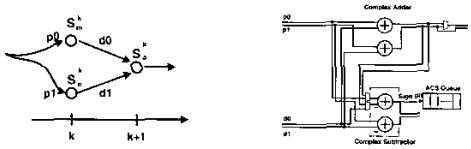
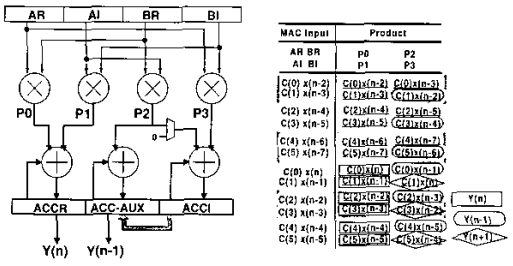



Figure 2: Block diagram of the Proposed Butterfly Unit.




(a) (b)

Figure 3: (a) Path metric selection in trellis diagram and (b) add-compare-select operation.




(a) (b)

Figure 4: (a) $16\times16$ MAC architecture for $16\times16$ complex-valued MAC and $16 \times 16$ real-valued MAC. (b) Processing schedule of a double-precision complex-valued $K$-tap FIR ($K = 6$).

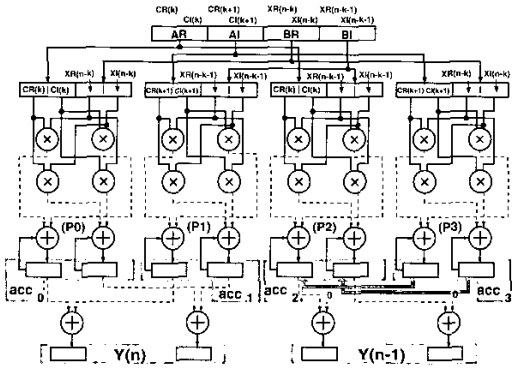



Figure 5: $8 \times 8$ complex-valued MAC architecture.

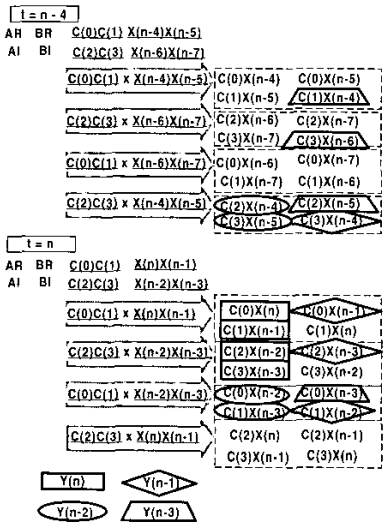
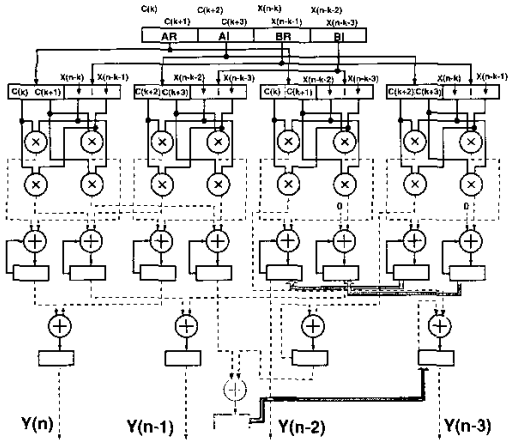



Figure 6: Processing schedule of an $K$-tap single-precision real-valued FIR ($K = 4$).



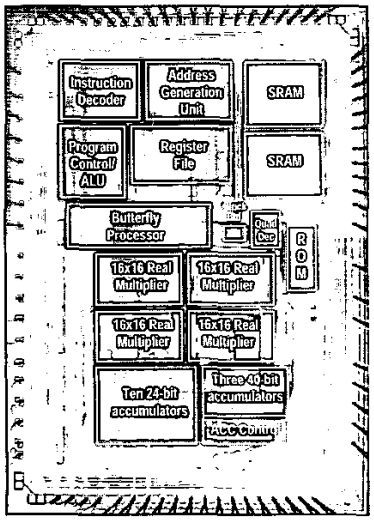




Figure 8: Microphotograph of the DSP chip.

Table 1: PERFORMANCE OF TYPICAL DSP ALGORITHMS.

| DSP algorithm ($N$=16) | proposed DSP | Traditional $N$-bit DSP |
|---|---|---|
| $K$-Tap $N \times N$ Complex FIR | $K$ | $4K$ |
| $K$-Tap $N \times N$ Real FIR | $K/4$ | $K$ |
| $K$-Tap $\frac{N}{2} \times \frac{N}{2}$ Complex FIR | $K/4$ | $4K$ |
| $K$-Tap $\frac{N}{2} \times \frac{N}{2}$ Real FIR | $K/16$ | $K$ |
| $K$-Tap $N \times N$ Complex IIR | $2K+1$ | $8K+2$ |
| $K$-Tap $N \times N$ Real IIR | $K/2+1$ | $2K+1$ |
| $K$-Tap $\frac{N}{2} \times \frac{N}{2}$ Complex IIR | $K/2+1$ | $8k+2$ |
| $K$-Tap $\frac{N}{2} \times \frac{N}{2}$ Real IIR | $K/8+1$ | $2K+1$ |
| $K$-Tap LMS $N \times N$ Adaptive Complex Filter | $4K+1$ | $12K+4$ |
| $K$-Tap LMS $(N/2) \times (N/2)$ Adaptive Complex Filter | $5K/2+1$ | $12K+4$ |
| Length-$K$ Radix-2 FFT | $K/2 \cdot \log(K)$ | $4K \cdot \log(K)$ |
| $K$-Dimension Square Distance | $K/2$ | $2K$ |
| $K$-State Viterbi State Transition | $K$ | $3K$ |

Table 2: NUMBER OF CYCLES FOR FIR OPERATIONS WITH DIFFERENT NUMBER OF TAPS AND SAMPLES.

| FIR (tap) × Data (length) | Proposed MAC | single-bus MAC (speed-up) |
|---|---|---|
| | **Single-Precision Complex-Valued** | |
| 4 × 8 | 55 cycles | 189 cycles (3.4) |
| 256 × 2048 | 205824 cycles | 801576 cycles (3.8) |
| | **Single-Precision Real-Valued** | |
| 8 × 16 | 44 cycles | 157 cycles (3.6) |
| 256 × 2048 | 70656 cycles | 1054720 cycles (14.0) |

Table 3: CHIP SUMMARY.

| | |
|---|---|
| **Technology** | TSMC 1p4m 0.35$\mu m$ |
| **Clock Speed** | 68MHz(@3.3V) |
| **Power Dissipation** | 644mW(@68MHz) |
| **Performance** | 1.1G MAC's/second<br>68M Butterfly/second |
| **Die Size** | 4.2×6.0$mm^2$ |
| **On-Chip Memory** | 1K words SRAM×2<br>1K words ROM×2 |