논문 2009-46SD-8-16

UWB시스템을 위한 고속 저복잡도 2-비트 레벨 파이프라인 비터비 복호기 설계

(High-Speed Low-Complexity Two-Bit Level Pipelined Viterbi Decoder for UWB Systems)

구 용 제*, 이 한 호**

(Yong-Je Goo and Hanho Lee)

요 약

본 논문에서는 MB-OFDM 초광대역 시스템을 위한 높은 속도와 저복잡도를 갖는 2-비트 레벨 파이프라인 비터비 디코더를 소개한다. 가산-비교-선택 유닛(ACSU)은 비터비 복호기의 주요 병목지점으로서, 임계경로를 줄이는 2-step look-ahead 기법에 기반을 둔 2-비트 레벨 파이프라인 MSB-first ACSU 유닛에 대해 제안한다. 제안하는 ACSU 구조는 1.8V의 공급 전압에서 동작하는 0.18-μm CMOS 공정을 이용하여 구현하였다. ACSU유닛은 870MHz의 클록 주파수에서 동작하며, 1.7Gb/s 의 데이터 처리율을 가진다.

Abstract

This paper presents a high-speed low-complexity two-bit level pipelined Viterbi decoder architecture for MB-OFDM UWB systems. As the add-compare-select unit (ACSU) is the main bottleneck of the Viterbi decoder, this paper proposes a novel two-bit level pipelined MSB-first ACSU, which is based on 2-step look-ahead techniques to reduce the critical path. The proposed ACSU architecture requires approximately 12% fewer gate counts and 9% faster speed than the conventional MSB-first ACSU. The proposed Viterbi decoder was implemented with 0.18-um CMOS standard cell technology and a supply voltage of 1.8V. It operates at a clock frequency of 870 MHz and has a throughput of 1.74 Gb/s.

Keywords: Vitrbi decoder, radix-4, multiband orthogonal frequency-division multiplexing (MB-OFDM), ultra wide band (UWB)

I. Introduction

Ultrawideband (UWB) is a wireless communications technology that operates in a newly allocated unlicensed spectrum. Advantages of UWB include low power consumption and very low cost/complexity with high data rates (up to 640 Mb/s raw uncoded, 480 Mb/s coded). UWB specifications target emerging wireless personal area network (WPAN) communications, which enable high-speed, short-range, cable-free connectivity for a wide array of multimedia consumer electronics, PC peripherals, and mobile devices, including wireless USB, wireless 1394, and the emerging wireless UPnP/IP protocols. UWB communication systems enable the delivery of data from a rate of 110 Mb/s at a distance of 10 m to a rate of 480 Mb/s at a distance of 2 m in a realistic multipath environment with existing

* 학생회원, ** 정회원, 인하대학교 정보통신공학부
(School of Information and Communication Engineering, Inha University)

※ 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (IITA-2009-C1090-0902-0019)

접수일자: 2009년6월8일, 수정완료일: 2009년8월8일

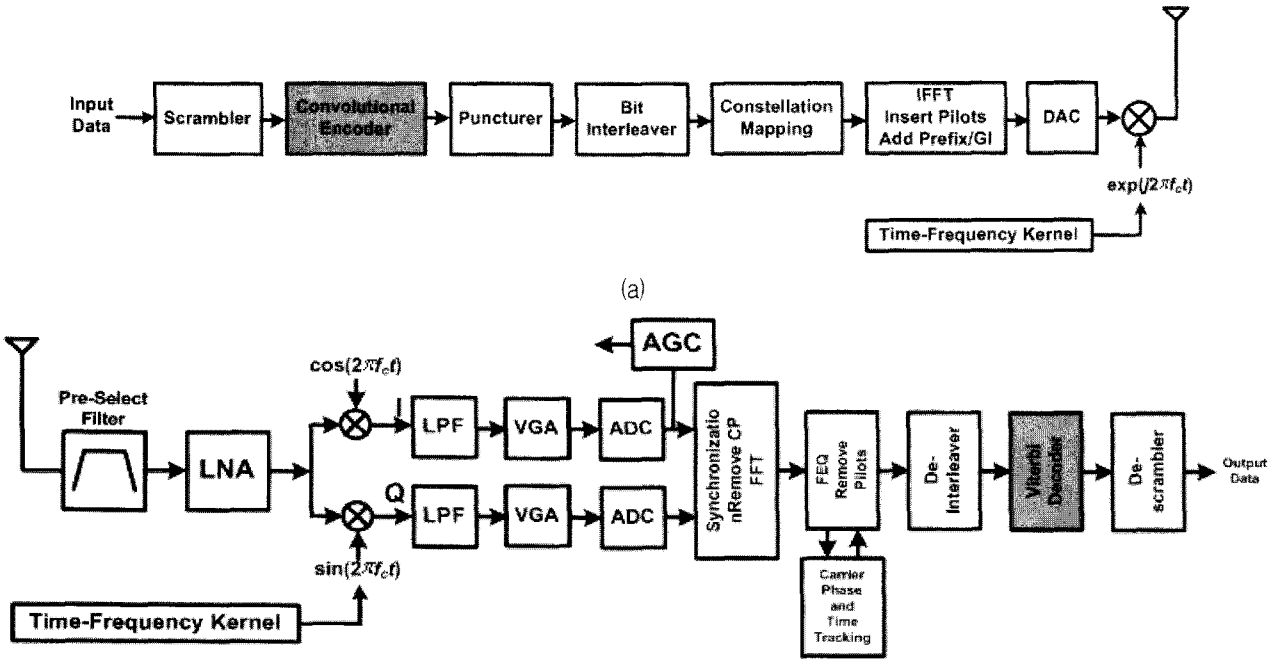

그림 1. MB-OFDM UWB 시스템에서의 블록도 (a) 송신기, (b) 수신기
Fig. 1. Block diagram of MB-OFDM UWB system (a) transmitter, and (b) receiver.

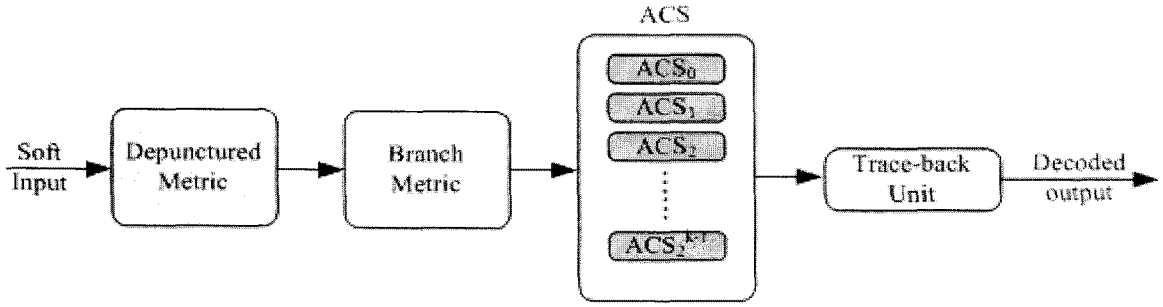

그림 2. 비터비 복호기의 블록도
Fig. 2. Block diagram of the punctured Viterbi decoder.

narrowband systems. They offer a higher data rate than 802.11 or Bluetooth, andare ideally suited to application in short range wireless communications, because they can share a frequency band[1]. One of the communication methods for the IEEE 802.15.3a standard is Multiband Orthogonal Frequency Division Multiplexing (MB-OFDM), which offers 528 MHz bandwidth[2~3]. This system transmits OFDM symbols using a different carrier frequency from symbol to symbol according to Time Frequency (TF) codes[2~3]. MB-OFDM-based UWB not only has reliable high-data-rate transmission in time-dispersive or frequency-selective channels without having complex time-domain channel equalizers but also can provide high-spectral efficiency.

The Viterbi algorithm, which is widely used for channel decoding in digital telecommunications, was first introduced as a method for convolutional decoding in 1967[4]. Generally, a Viterbi decoder consists of three basic computation units: the branch metric unit (BMU), the add-compare-select unit (ACSU), and the TraceBack unit (TBU), as shown in Fig. 2. The BMU calculates the branch metrics by the hamming distance or the Euclidean distance, and the ACSU calculates a summation of the branch metric from the BMU and previous state metrics,

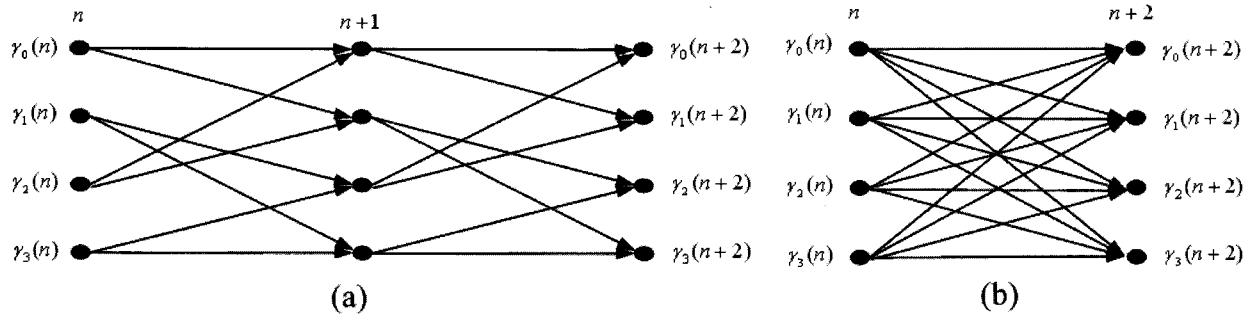

그림 3. (a) Radix-2 와 radix-4 의 격자 블록도
Fig. 3. Trellis diagrams of (a) radix-2, and (b) radix-4.

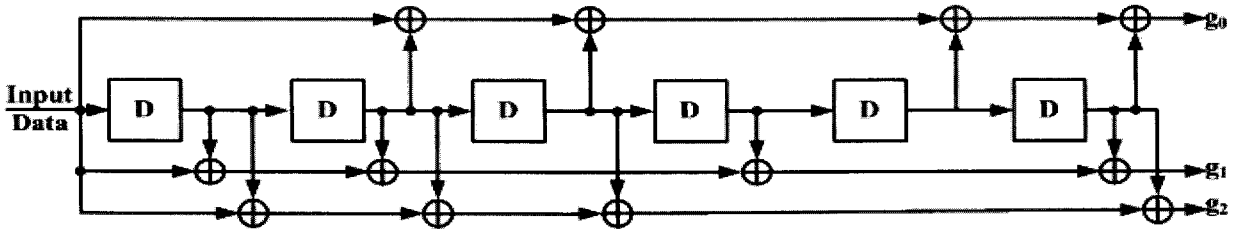

그림 4. 길쌈 부호기 $(k=7,\ g_0=133_8,\ g_1=165_8,\ g_2=171_8)$
Fig. 4. Convolution encoder $(k=7,\ g_0=133_8,\ g_1=165_8,\ g_2=171_8)$.

which are called the path metrics. After this summation, the value of each state is updated and then the survivor path is chosen by comparing path metrics[4]. The TBU processes the decisions made in the BMU and ACSU, and outputs the decoded data. The BMU and the TBU are only composed of feed-forward paths. It is relatively easy to shorten the critical path in these two units by utilizing pipelining and parallel processing techniques. However, the feedback loop of the ACSU is a major critical path for the Viterbi decoder. There are several techniques to implement a high speed Viterbi decoder.

In [5], the M-step look-ahead technique, a well-known high speed algorithm speed algorithm for elimination of feedback operation, is utilized. Generally, as M is increased, the power consumption and the hardware complexity are increased Thus, M is limited to 2. Fig.3 shows trellis diagrams for the case of 4 states. Fig. 3(a) is a general trellis diagram of a Viterbi decoder in which each state has two branches connected to the other states. The trellis diagram of the Viterbi decoder is iterated as same paths in every time, as shown in Fig. 3(a), and thus it can be modified to a 2-step look-ahead structure, as shown in Fig. 3(b)[5]. There are four branches connected to each state, as shown in Fig. 3(b), by the 2-step look-ahead technique.

With M-step look-ahead, the iteration in the ACSU is equivalent to iterations in a non-look-ahead implementation. Thus, the speed requirements of the ACSU for a given decoding data rate are reduced by M-times. The least significant bit (LSB) first computation is available for accumulation similar to the summation, but the most significant bit (MSB) first computation is suitable for comparing and selecting operations in the ACSU. The ACSU structure combining MSB-first compare-select with carry propagation was proposed based on redundant number representation in [6]. The CSAU structure was proposed to reduce the critical path of the ACS operation[8]. However, when the ACSU is constructed for the radix-4 structure, the area of the CSAU is increased exponentially and the critical path of the CSAU is increased by the total bits of the adder.

In this paper, a high-speed low-complexity two-bit

level pipelined MSB-first ACSU is proposed with the aim of reducing the hardware complexity and improving the clock frequency in the Viterbi decoder. This design does not require adder computation and code converter delay, which are required in the critical path of the conventional bit-level pipelined ACSU[6]. In addition, the pre-TraceBack method is adapted to reduce the area of the memory for the TraceBack operation in the Viterbi decoder.

The proposed Viterbi decoder uses code with a rate $R = 1/3$ ($g_0 = 133_8$, $g_1 = 165_8$, and $g_2 = 171_8$) and a constraint length k=7 for a basic coding rate, as shown in Fig. 4. The number of the states is 64 states, as given by 2^{k-1}.

This paper is organized as follows. Section II explains the proposed punctured Viterbi decoder architecture using the proposed ACSU and the pre-TraceBack method. In section III, the hardware complexity and critical path delay of the proposed ACSU are compared with the conventional bit-level pipelined ACSU and the implementation results of the proposed Viterbi decoder are presented. Conclusions are presented in section IV.

II. Two-Bit Level Pipelined Viterbi Decoder

1. Depunctured unit

The punctured Viterbi decoder is used for diverse code rates using one Viterbi decoder. To implement the punctured Viterbi decoder, a depunctured unit is added in the front of the BMU. The depunctured unit is implemented for code rates (R) of 1/2, 1/3, 3/4, and 5/8, and the basis code rate of the Viterbi decoder is 1/3. The depunctured unit takes the input signals and the code rate mode signal, and then the input signal is changed to an appropriate signal by a depunctured metric. The outputs of the depunctured unit, which are appropriately changed to 2 input symbols of the Viterbi decoder by using 2-step look-ahead

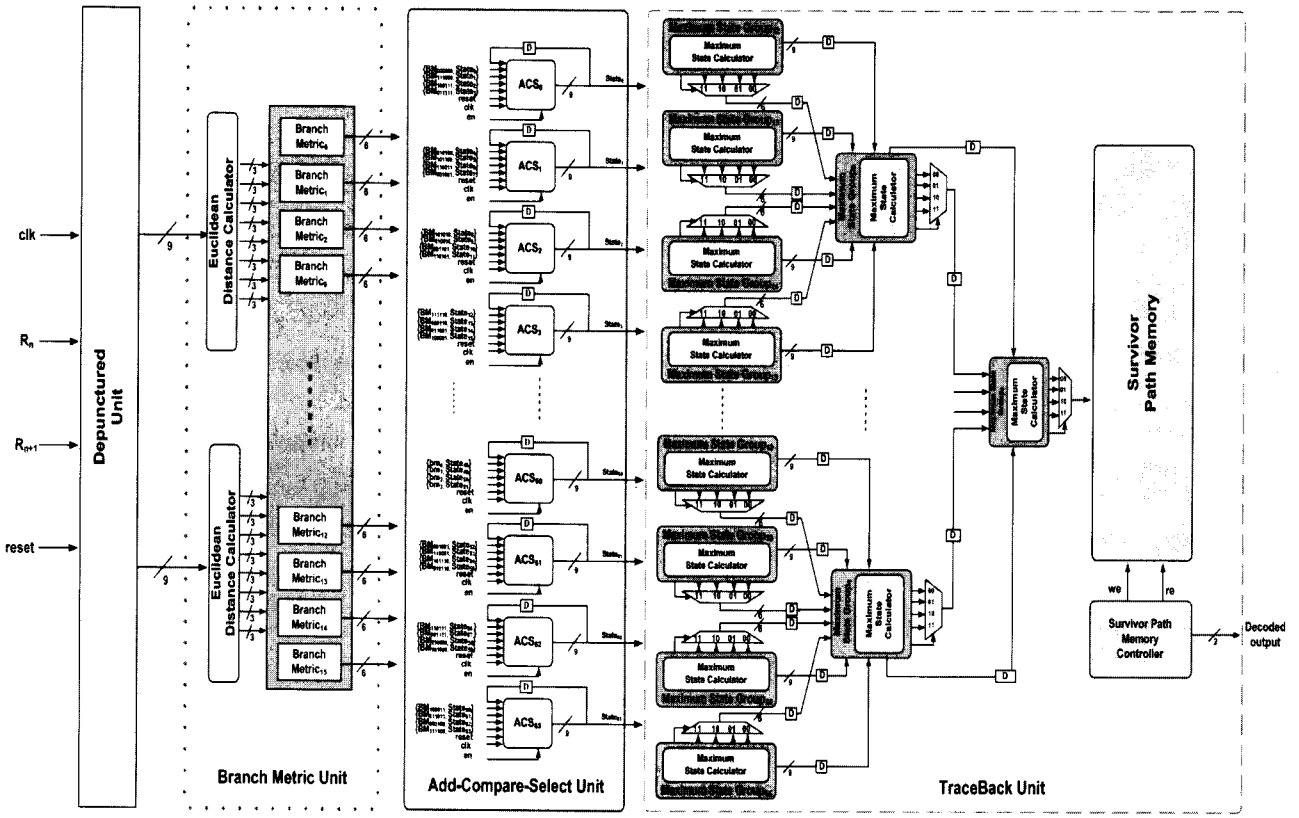

그림 5. 제안하는 2비트 레벨의 파이프라인 비터비 복호기 블록도
Fig. 5. Block diagram of the proposed two-bit level pipelined Viterbi decoder.

techniques, are passed to the BMU.

2. Branch metric

The integers for branch metrics are used instead of log likelihoods. This integer method has a simple structure and offers negligible performance reduction for 8-level quantization[9]. Brach metrics for the 2-step look-ahead trellis are generated by combining branch metrics of two radix-2 trellis structures. For each iteration of the radix 2-trellis from n to n+1, three 3-bits soft-decision inputs can be constructed as G_3, G_2, G_1. As shown in Table 1, each input (G) can be implemented as follows:

$$\{BM_0[2], BM_0[1], BM_0[0]\} = \{G[2], \overline{G[1]}, \overline{G[0]}\} \quad (1)$$

$$\{BM_1[2], BM_1[1], BM_1[0]\} = \{\overline{G[2]}, G[1], G[0]\} \quad (2)$$

G: received data, BM: branch metric

In addition, each of the three 3-bits soft-decision inputs can be organized to logically form eight possible branch metrics, as BM_{000}, $BM_{001}\cdots BM_{111}$. 64 possible 2-step look-ahead branch metrics for iteration from n to n+2 are formed from the radix-2 metrics $(G_3, G_2, G_1)_n \times (G_3, G_2, G_1)_{n-1}$ for

표 1. 입력비트에 대한 브랜치 메트릭
Table 1. Branch metric for input soft bits.

G	000	001	010	011	100	101	110	111
BM0	011 (3)	010 (2)	001 (1)	000 (0)	111 (7)	110 (6)	101 (5)	100 (4)
BM1	100 (4)	101 (5)	110 (6)	111 (7)	000 (0)	001 (1)	010 (2)	011 (3)

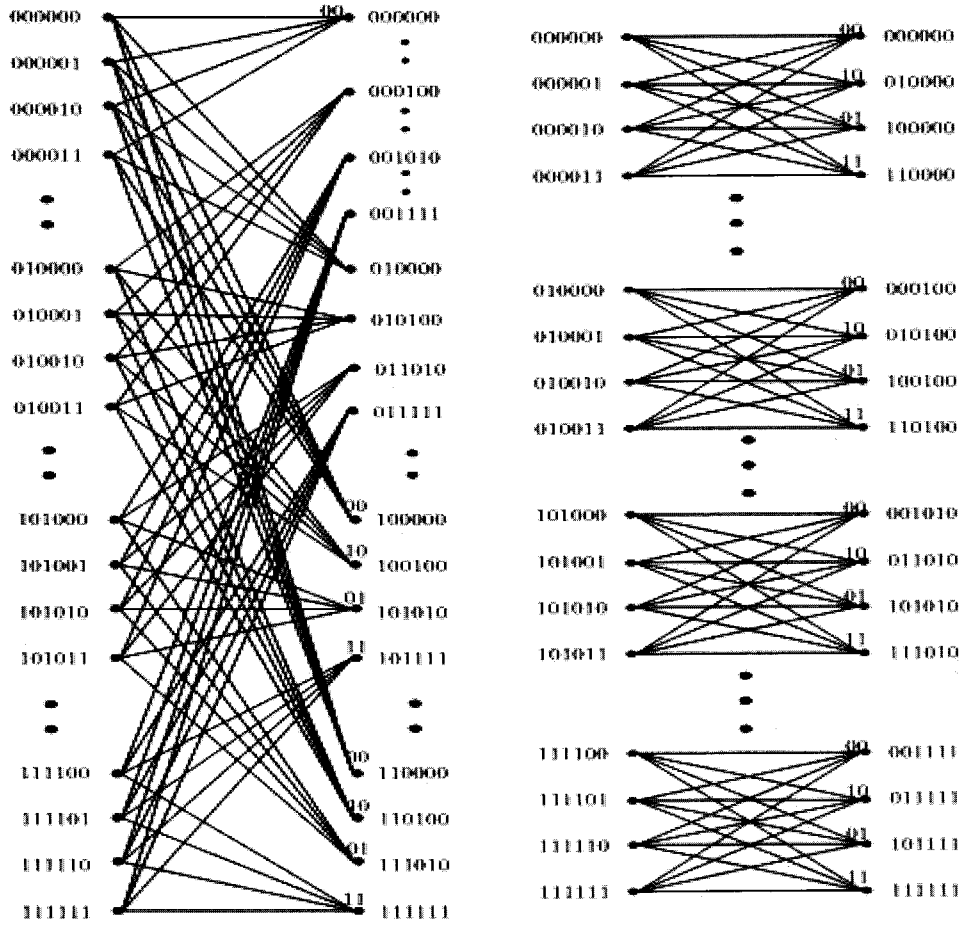

그림 6. 초광대역 시스템을 위한 길쌈 부호기의 트렐리스 가지도
Fig. 6. Trellis diagram of convolution encoder for UWB system.

iterations n and n+1. In this design, the radix-4 metrics are centrally calculated and then globally distributed to the ACS units. As shown in Table 1, the branch metric is constructed as

$$BM_{000.000} = BM_{000}^{n} + BM_{000}^{n+1} \quad (3)$$

$$BM_{000.001} = BM_{000}^{n} + BM_{001}^{n+1} \quad (4)$$

$$\vdots$$

$$BM_{111.110} = BM_{111}^{n} + BM_{110}^{n+1} \quad (5)$$

$$BM_{111.111} = BM_{111}^{n} + BM_{111}^{n+1} \quad (6)$$

Because the constraint length of the convolution encoder is 7, the total number of the states is 64 on the UWB system, as given by 2^{K-1}. The trellis structure is extremely complicated to indicate a trellis diagram of the radix-4 structure, as shown in Fig. 6. The butterfly contains 8 states, $S_{i,t}$, $S_{i+1,t}$, $S_{i+2,t}$, $S_{i+3,t}$, $S_{j,t+2}$, $S_{j+16,t+2}$, $S_{j+32,t+2}$ $S_{j+48,t+2}$, and each state consists of 6 bits. The butterfly for the radix-4 structure is paired depending on the input bits, which are '00_2', '01_2', '10_2' and '11_2'.

For instance, if the current state ($S_{i,t}$) is '010000' at time t interval and receives input bit '00_2 ' '01_2' '10_2' '11_2' at time t and time t+1, the next state ($S_{j,t+2}$) is '000100_2', '010100_2', '100100_2', '110100_2' at time t+2 interval. The 2 inputs $\{C_{in,t}, C_{in,t+1}\}$ of the convolution encoder at time t and time t+1 and $\{S_{i,t}[5], S_{i,t}[4], S_{i,t}[3], S_{i,t}[2]\}$ are used as decision bits to decide the next state of the trellis diagram for the radix-4 architecture. The next state of the trellis diagram has the following characteristic: $S_{i,t+2}=\{C_{in,t+1}, C_{in,t}, S_{i,t}[5], S_{i,t}[4], S_{i,t}[3], S_{i,t}[2]\}$. $S_{i,t+2}[5], S_{i,t+2}[4]$ are used as decision bits for the TraceBack operation.

3. Proposed add-compare-select unit

The recursive state metric update results in unbounded word growth due to the addition of branch metrics, which are always nonnegative.

Generally, the proved modular arithmetic approach is adopted to avoid normalization[9]. For the 64-state 2-step look-ahead decoder implementation, the required state metric precision is 9 bits, as given by

$$S_{bits} = \lfloor \log_2(\Delta_{max} + k \times \lambda_{max}) \rfloor + 1 \quad (7)$$

$$S_{bits} = \lfloor \log_2(126 + 2 \times 21) \rfloor + 1 = 9bits \quad (8)$$

where, k=7, λ_{max}=21, and Δ_{max}=126.

The proposed MSB-first ACSU is pipelined in 2 bit-levels to reduce the critical path of the ACSU, as shown in Fig. 6. The proposed MSB-first ACSU can be divided into 3 basic function blocks, which are the 2-bits adders, the maximum position (MP) blocks, and the maximum selection (MS) blocks. To implement the two-bit level pipelined MSB-first ACSU, it is necessary to consider the influence of the carries in the 2-bits full adders, which can affect the decision of the maximum state value in each state at time n+1, n+2, n+3. The MP block is designed to defend against the influence of the carries, which occur at time n+1, n+2, n+3 as shown in Fig. 7. Fig. 7 and Fig. 8 show the structure of the MP block. The MP block has three sub-blocks: the partial decision(PD) block, the maximum-position-detection (MPD) block, and the maximum-position- selection (MPS) block. The PD block is constructed to find which state has 1 less value than the maximum value ($S_{max}^{n}[1], S_{max}^{n}[0]$) among the four branch values $\{(S_a^n[1], S_a^n[0]), (S_b^n[1], S_b^n[0]), (S_c^n[1], S_c^n[0]), (S_d^n[1], S_d^n[0])\}$ at time n+1. If one of the branch values, which are calculated at time n, is a value 1 less than the maximum value that is calculated at time n, the output of the PD block is 1. Because of the influence of the carries, the maximum state value S_{max}^{n} and the state values $\{S_a^n, S_b^n, S_c^n, S_d^n\}$ can be changed. When a carry occurs in states, which have the same values as the maximum state value at time n, the survivor path can be modified by carries at

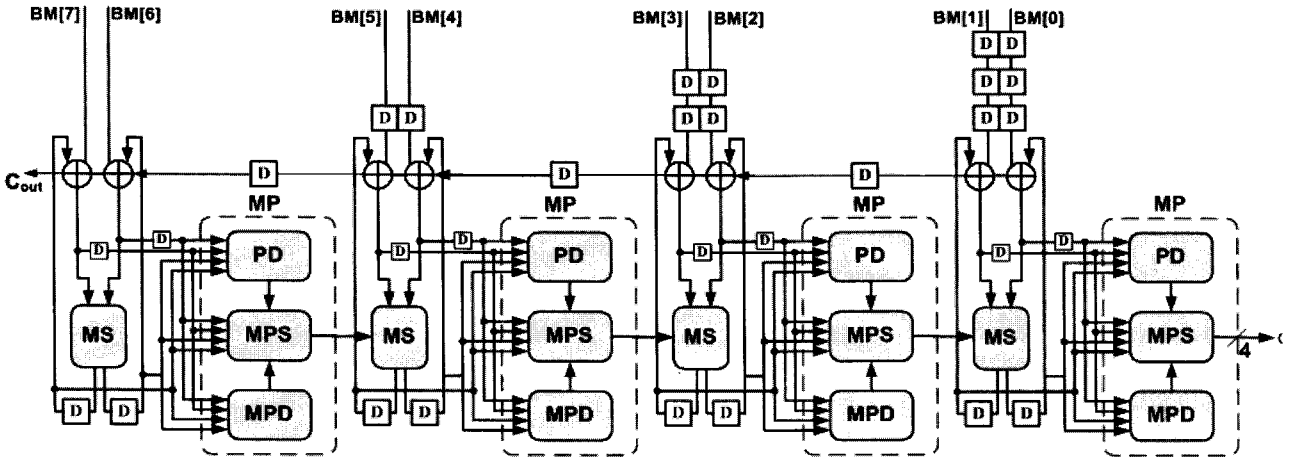

그림 7. 제안하는 2-비트 레벨 파이프라인 MSB-first ACSU 블록도
Fig. 7. Block diagram of the proposed two-bit level pipelined MSB-first ACSU.

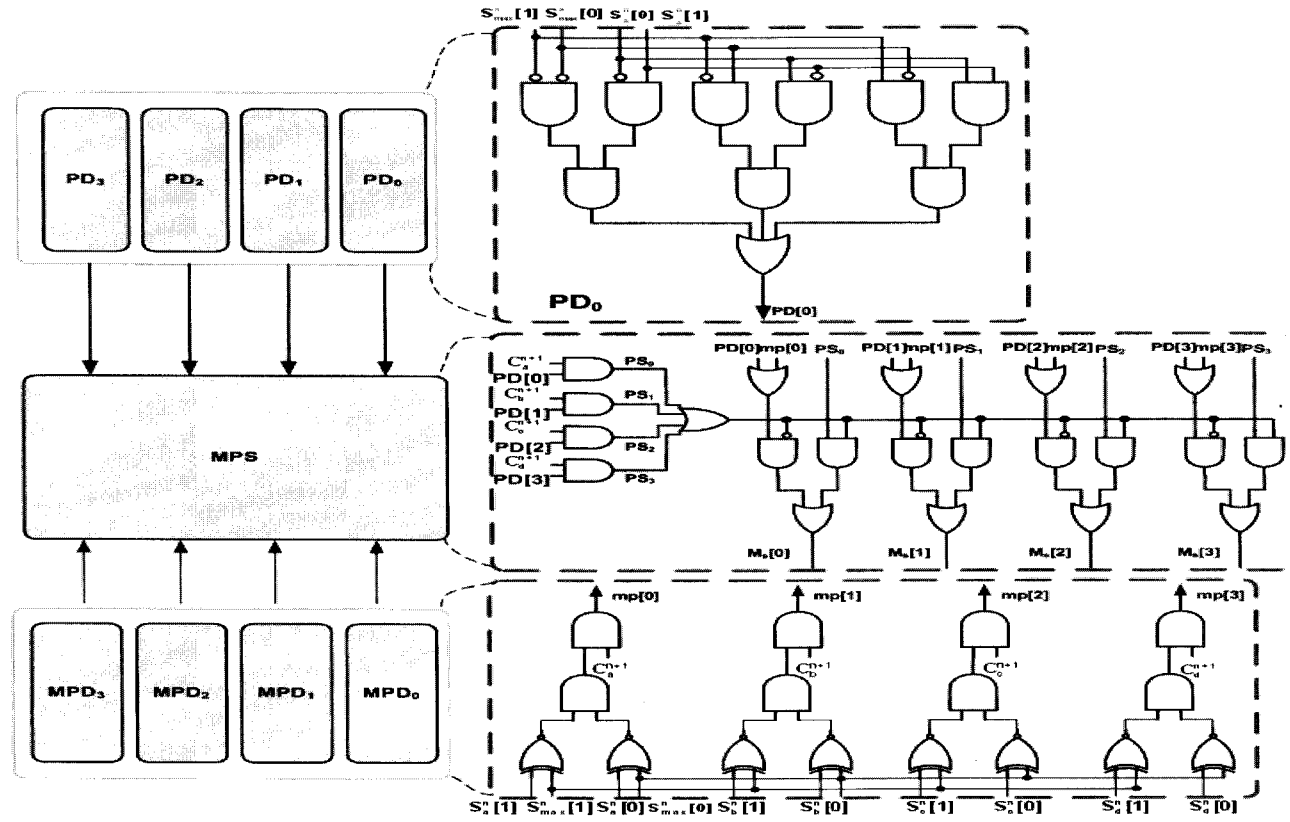

그림 8. PD, MPS, MPD 블록의 회로도
Fig. 8. Circuit of the PD block, the MPS block, and the MPD block.

time n+1.

The MPD block finds which state has the same value as the maximum value ($S^n_{\max}[1], S^n_{\max}[0]$) at time n and checks the carries at time n+1 by considering the influence of the carries ($C_a^{n+1}, C_b^{n+1}, C_c^{n+1}, C_d^{n+1}$). As the MPD block operates to find the state that has the same branch value during operation of the PD block, the critical path of the ACSU is reduced. The MPS block is constructed to decide which branch value is the maximum branch value in each state. The MPS block receives the inputs from the result of the PD block, MPD block, and the carries at time n

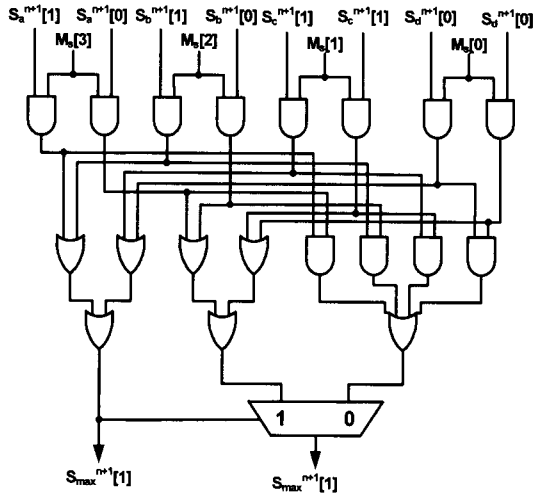

그림 9. MS 블록의 회로도
Fig. 9. Circuit of the MS block.

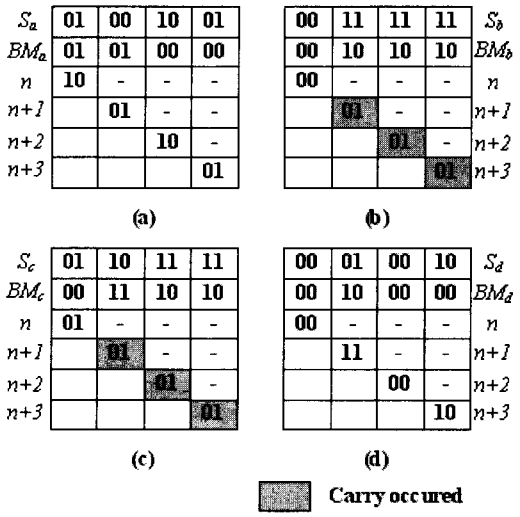

그림 10. 가산 연산의 결과
Fig. 10. Results of the add operation.
(a)S_a, (b)S_b, (c)S_c, (d)S_d

+1. The MPS block makes a decision as to which branch value is the maximum state at time n, and supports the MS block in finding the maximum branch values at time n+1. If the output of the MPS block is enabled, the MS block does not consider the value of the 2-bits adder, which is calculated at time n+1, to find the maximum value, as shown in Fig. 9. After the operation of the ACSU is finished, the output of the MPS in the least significant bits (LSB) is handed over to the TBU to decide the survivor path at each time and store the information of the survivor path in the survivor path memory in the TBU.

For instance, if the values of the four states are S_a=01001001$_2$, S_b=00111111$_2$, S_c=01101111$_2$, and S_d=00010010$_2$, and the values of the branch metric to each state are BM_a=01010000$_2$, BM_b=00101010$_2$, BM_c=00111010$_2$, and BM_d=00100000$_2$, as shown in Fig. 10, then the ACSU runs the add operation first before deciding the maximum state value. As the comparison operation of the proposed ACSU, which decides the maximum state value, is operated from the MSB to the LSB, the result of the comparison operation is influenced by carries.

When a carry occurs, the carry affects the decision of the comparison operation, as shown in Fig. 10. Fig. 10 (c) shows the state with the maximum state value. However, as shown in Fig. 10. Fig. 10 (c) shows the state with the maximum state value. However, as shown in Fig. 10, the comparison operation decides the maximum state to S_a at time n, because the results of the add operation are "10$_2$" in Fig. 10 (a) and "01$_2$" in Fig. 10 (c).

Although $\{S_a[7], S_a[6]\}$ and $\{S_b[7], S_b[6]\}$ are modified to the same values at time n+1, S_a is selected as the maximum state value by the comparison result at time n, because a carry of S_c occurs at n+1. The PD-block calculates whether the value of each state at time n has a value 1 less than the maximum value ($S^n_{\max}$) at time n+1. If the value of the state is 1 less than the maximum value ($S^n_{\max}$) at time n, the result of the PD-block is '1'. The MPD block takes the carries (C_a^{n+1}, C_b^{n+1}, C_c^{n+1}, C_d^{n+1}) from the 2-bit full adder at time n+1 and finds the states that have the same values with $S^n_{\max}$+1. If a carry occurs in the states, which have the same value as $S^n_{\max}$, the comparison operation does not need to consider the influence of the carries of the other states. In the MPD block, if a carry does not occur in the states that have the maximum state value, the result of the MPD block is '0' at n+1.

When the values of the states have values 1 less than $S_{\max}^{n}$, it is not necessary to consider whether a carry (C_{n+1}) has occurred or not. In this case, the result of the MPD block is also '0'. When the states have the same values as $S_{\max}^{n}$ and a carry occurs at n+1, the result of the MPD block is '1'. As shown in Fig. 10 (c), a carry of S_c^{n+1} occurs by the 2-bit add operation at n+1. However, because S_c^{n+1} has a value 1 less than the maximum state value at time n, $\mathrm{mp}[2]_{n+1}$, which is the result of the MPD block for S_c^{n+1} at n+1, is '0'. The results of the MPD block for the other states are '0', as shown in Fig. 10 (c). After the MPD block finishes the operation, the MPS block finds the positions that have the same value as $S_{\max}^{n}$, by using the results from the MPD block, the PD block, and the carries of the 2-bit adder at n+1. As $\{S_c^n[7], S_d^n[6]\}$ is 1 less than $S_{\max}^{n}$ and a carry (C_c^{n+1}) occurs as a result of the operation of the 2-bits full adder at time n+1, as shown in Fig. 10 (c), the result of the PD block (PD $[2]_{n+1}$) of the $\{S_c^n[5], S_d^n[4]\}$ is '1' and the result of the MPD block (mp[2]) is '0' at time n+1.

As the MPS block decides which states have the same values according to the results of the PD block and the MPD block, which consider the carry at time n+1 and the maximum state value at time n, the result of the MPS block is '0101_2' at time n+1. If a carry of S_a occurs at time n+1, the result of the MPS block is '0001_2' as the maximum state is still S_a by a carry of S_a at time n+1. The MS block finds the positions that have the maximum state values among four states and decides the maximum state value by using the result of the MPS block. If the result of the MPS block is '0' at time n+1, then the value of the state is not the maximum state value at time n. If the value of the state is not the maximum state value, it is not necessary to find the maximum state value at time n+1. For this reason, the state values that are not the maximum state values are modified to zeroes. As the states that are not the maximum state values are zeroes, the carry does not influence the operation of the comparison of the ACSU, and the MS block can decide the maximum state value and the survivor path correctly.

4. Traceback unit

There are several methods to obtain the decoded output of the Viterbi decoder. The register exchange algorithm and the TraceBack algorithm are representative methods to decide the decoded output. As the TraceBack algorithm uses less power than the register exchange algorithm, it is more suitable [11]. To obtain the decoded output of the Viterbi decoder, the TBU first has to decide the maximum state among the 64 states, which is the initial state for the TraceBack operation. As the TBU invokes too much delay time to compute the maximum states among the 64 states at one time for the TraceBack operation, 64-states are separated to 16 groups per 4 states. The TBU finds the maximum states in each group at time n. The outputs of the 16 groups are then separated to these 4 groups and are calculated to find the maximum states in each group at time n +1. Finally, the last 4-states that are selected to maximum state at time n+1 are compared to decide the initial state for the TraceBack operation. There are several known TraceBack algorithms[11~13]. As a large portion of the area and the power consumption by the Viterbi decoder is dedicated to the survivor path memory and the access operations associated with it, the number of memory access operations has to be reduced to save area and power consumption. Compared to the operation of the conventional TBU, which is separated into three types of memory access operation (decision bits write (WR), TraceBack read (TB), and decode read (DC)), the pre-TraceBack algorithm needs only two types of memory access operation (decision bits write (WR) and decode read (DC))[13]. As applying the pre-TraceBack algorithm to implement the TBU, the WR operation writes the decision bit vectors into the survivor path memory in

an increased memory address order, which are provided by the ACSU, and at the same time the pointer register updates the initial state in each state according to the survivor path. When the pointer register receives the enable signal, the pointer registers are initiated and the content of the register pointer is then updated by the updating index. The updating index is calculated by the decision bits, which are the results of the ACSU[12]. The DC operation performs two operational steps. First, after selecting an arbitrary state for the TraceBack, DC operation decides target TraceBack state corresponding to the arbitrary state in the pre-TraceBack pointer register. Second, starting with the state identified with the first step, an iterative read operation is performed from the memory for decoding. As adapted in the pre-TraceBack algorithm, the latency for the TB operation is removed in the TBU. The depth of the survivor path memory and the latency for the decoding are decreased from 4 to 3. Finally, the area of the survivor path memory is reduced 25% by reduction of the latency for decoding[13].

III. Implementation and Comparison

The proposed Viterbi decoder was modeled in Verilog HDL and simulated to verify its functionality. After complete verification of the design functionality, it was then synthesized using appropriate time and area constraints. Both simulation and synthesis steps were carried out using SYNOPSYS design tools and 0.18-μm CMOS technology optimized for a 1.8V supply voltage.

Table 2 presents a comparison of the critical path with the conventional ACSU[6, 8]. The critical path of the proposed two-bit level pipelined ACSU requires fewer XNOR, AND gates compared to the conventional bit-level pipelined ACSU[6], and the critical path of the CSAU[8] requires more AND, OR, XOR gates compared with the proposed ACSU.

Table 3 shows the hardware complexity for the proposed two-bit level pipelined ACSU and the conventional bit-level pipelined ACSU. The proposed ACSU in each state requires 7% fewer gates than does the conventional bit-level pipelined ACSU.

표 2. 제안된 ACSU에 대한 critical path 비교
Table 2. Comparison of the critical path for the proposed ACSU.

Architecture	Critical path
Proposed 2 bit-level pipelined ACU	$3T_{or}+4T_{nand}+T_{xor}+T_{nor}$
Bit-level pipelined ACSU [6]	$3T_{xnor}+3T_{nand}+2T_{or}+T_{xor}+T_{and}+T_{nor}$
Transformed CSAU [8]	$8t_{xor}+8t_{and}+8t_{or}+t_{4x1mux}$

표 3. 1개의 상태와 64개의 상태에서의 성능비교
Table 3. Performance comparison for 1 state and 64 states ACSU.

Architecture	State	Gate count	Critical path delay(ns)
Proposed 2 bit-level pipelined ACSU	1	1,915	1.02
	64	108,379	1.15
Bit-level pipelined ACSU [6]	1	2,115	1.12
	64	123,303	1.25

표 4. 코드 rate K가 7일 때 비터비 디코더 구현 결과
Table 4. Implementation results of the Viterbi decoder with code rate K=7.

Architecture		Proposed Two-bit level Pipelined Viterbi Decoder
Branch Metric Unit		8,201
ACS Unit		108,379
TraceBack Unit	Pointer Register	6,935
	Others	31,512
Total # of Gates		148,101
Clock Rate (MHz)		870
Throughput (Gb/s)		1.74
Technology		0.18-m CMOS 1.8V

When the ACSU is implemented for 64 states, the number of gates for the proposed ACSU is reduced by about 12%, as shown in Table 3.

The total number of gates for the proposed Viterbi decoder is 148,101 from the synthesized results excluding the survivor path memories. The proposed Viterbi decoder operates at a clock frequency of 870 MHz and has a throughput of 1.74 Gb/s.

Ⅳ. Conclusion

This paper presents a high-speed low-complexity two-bit level pipelined Viterbi decoder for high-performance UWB applications. A two-bit level pipelined ACSU is proposed to reduce the hardware complexity and the critical path delay of the Viterbi decoder. The proposed ACSU requires approximately 12% fewer gate counts and 9% faster speed than the conventional MSB-first ACSU. As a result, the Viterbi decoder using the proposed ACSU has lower hardware complexity compared to a previously reported Viterbi decoder based on the conventional bit-level pipelined ACSU. The proposed Viterbi decoder operates at a clock frequency of 870MHz and has a throughput of 1.74Gb/s, while requiring small hardware complexity.

References

[1] Time Domain, "UWB Applications, Demonstration & Regulatory Update," Sept 2001 workshop, March 20, 2001.

[2] A. Batra et al., "Multi-band OFDM Physical Layer Proposal for IEEE 802.15 Task Group 3a," IEEE P802.15-03/268r3, March 2004.

[3] A. Batra, J. Balakrishnan, G. R. Aiello, J. R. Foerster, A. Dabak, "Design of Multiband OFDM System for Realistic UWB Channel Environment," IEEE Trans. on Microwave Theory and Techniques 52 (2004) 2123-2138.

[4] A. J. Viterbi, "Error bounds for convolutional coding and an asymptotically optimum decoding algorithm," IEEE Trans. Information Theory IT-13 (1967) 260-269.

[5] P. J. Black and T. H. Meng, "A 140-Mb/s, 32-state, radix-4 Viterbi decoder," IEEE Journal of Solid-State Circuits 27 (1992) 1877-1885.

[6] G. Feygin, and P. G. Gulak, "Architectural Tradeoffs for Survivor Sequence Memory Management in Viterbi decoders," IEEE Trans. Commun 41 (1993) 425-429.

[7] D. Dabiri, and I.F Blake "An Area Efficient Surviving Paths Unit for the Viterbi Algorithm," IEEE Trans. Commun. 1 (1999) 300-304.

[8] Y. Gang, A. T. Erdogan, and T. Arslan, "An efficient pre-tracback approach for Viterbi decoder Targeting Wireless Communication Applications" IEEE Transaction on Circuits and Systems 53 (2007)1918-1927.

[9] J. A. Heller, I. M. Jacobs, "Viterbi decoding for satellite and space communication," IEEE Trans. Commun. Technol 19 (1971) 835-848.

[10] A. P. Hekstra, "An alternative to metric rescaling in Viterbi decoders," IEEE Trans. Commun. 37 (1989) 1220-1222.

[11] G. Feygin, and P. G. Gulak, "Architectural Tradeoffs for Survivor Sequence Memory Management in Viterbi decoders," IEEE Trans. Commun 51 (1993) 425-429.

[12] D. Dabiri, and I.F Blake "An Area Efficient Surviving Paths Unit for the Viterbi Algorithm," IEEE Trans. Commun. 1 (1999) 300-304.

[13] Y. Gang, A. T. Erdogan, and T. Arslan, "An efficient pre-tracback approach for Viterbi decoder Targeting Wireless Communication Applications" IEEE Transaction on Circuits and Systems 53 (2007) 1918-1927.

저 자 소 개

구 용 제(정회원)

2006년 인하대학교 전자공학과 학사 졸업

2008년 인하대학교 정보공학과 석사졸업

2008년~현재 현대자동차그룹 케피코

<주관심분야 : 통신용 VLSI 및 SoC설계>

이 한 호(정회원)

1993년 충북대학교 전자공학과 학사 졸업.

1996년 Univ. of Minnesota 전기컴퓨터공학 석사

2000년 Univ. of Minnesota 전기컴퓨터공학 박사

2002년 Member of Technical Staff, Lucent Technologies, USA.

2004년 Assistant Prof. Dept. of Electrical and Computer Engineering, Univ of Connecticut, USA

2004년~현재 인하대학교 정보통신공학부 부교수

<주관심분야 : 통신용 VLSI설계, SoC설계>