HIGH-SPEED SOFTWARE FRAME SYNCHRONIZER USING SSE2 TECHNOLOGY

In-Hoi KOO[1], Sang-Il Ahn[1], Tae-Hoon Kim[2], Young-Bo Sakong[2]

Ground System Development Department, Korea Aerospace Research Institute[1]
freewill@kari.re.kr[1], siahn@kari.re.kr[1],
SOLETOP Inc. Satellite Image Dept. [2]
freekid99@soletop.com[2], ybsakong@soletop.com[2]

ABSTRACT: Frame Synchronization is applied to not only digital data transmission for data synchronization between transmitter and receiver but also data communication with satellite. When satellite image data with high resolution and mass storage is transmitted, hardware frame synchronizer for real-time processing or software frame synchronizer for post-processing is used. In case of hardware, processing with high speed is available but data loss may happen for Search of Frame Synchronization. In case of software, data loss does not happen but speed is relatively slow.
In this paper, Pending Buffer concept was proposed to cope with data loss according to processing status of Frame Synchronization. Algorithm to process Frame synchronization with high speed using bit threshold search algorithm with pattern search technique and SIMD is also proposed.

KEY WORDS: CCSDS, Frame Synchronization, SSE2

1. INTRODUCTION

Nowadays the digital communication is widely used between satellite and ground station for TT&C and payload data downlink application. CCSDS(Consultative Committee for Space Data Systems) provides some recommendations for digital communications. For transmission side in digital communications, Reed-Solomon or Turbo Encoding, Pseudo-Random Sequence Generation, Attachment of Attached Sync Marker, Convolutional Encoding are recommended while Convolutional Decoding, Frame Synchronization, Pseudo-Random Sequence Removal, Reed-Solomon or Turbo Decoding are recommended for receiver side.

To process the massive binary data for meaningful data extraction, the pre-requisite is to perform frame synchronization via frame synchronizer for real-time processing or software frame synchronizer for post-processing

In case of hardware frame synchronizer, small amount of data loss is normally expected. But software frame synchronizer shows no data loss but speed limitation is expected.

This paper proposes software frame synchronizer for high-speed processing. With Intel's SSE2 technology and Pattern Search Method, the processing speed in software approach was dramatically increased and frame data loss was minimized.

2. FRAME SYNCHRONIZATION & SSE2

2.1 Frame Synchronization

Frame Synchronization means to have special bit combination to indicate the start or stop of data frame in continuous bit data stream. With this, the data frame between TX and RX can be synchronized. Figure 1 shows state transition diagram in Frame Synchronization.

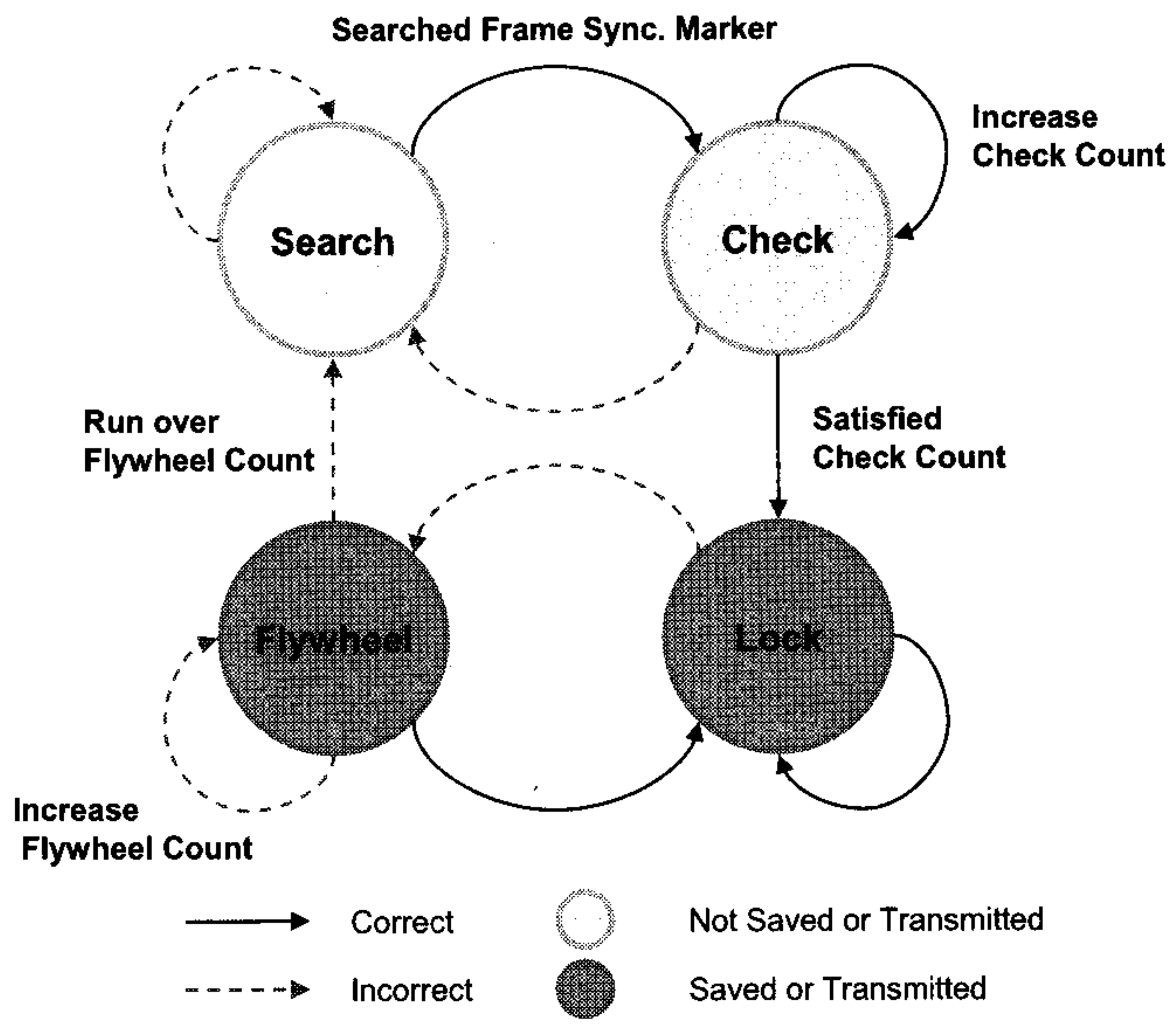

Figure 1 Frame Synchronization Final State Diagram

Frame Synchronization has 4 kind of states; Search, Check, Lock, Flywheel State.

2.1.1 Search State

In Search State, the Frame Synchronize Marker (hereafter, FSM) is searched in input data. Input data is shifted by bit until to find the FSM. This FSM, in fact, varies according to modulation method used. All possible FSM values are compared with input data and detect the FSM not exceeding the bit error threshold in FSM. When FSM is successfully searched, the state transition to Check State is done.

This Search State is most time consuming or highest computational load in 4 States. The rest 3 States like Check, Lock, Flywheel State moves position by frame length and just compare and detect the FSM using pre-defined phase information while both bit level shift and all FSM pattern are considered in Search State.

In this paper the Search State processing time in frame synchronization was reduced by adopting the bit threshold method with Pattern Search, and SSE2.

2.1.2 Check State

Check State detects the FSM by every frame size using pre-detected FSM in Search State. When the FSM detection is continuously successfully by check count configured, state transition to Lock State is done but when fails, state transition to Search State is done.

2.1.3 Lock State

Lock State detects the FSM by every frame size step like Check State. When FSM is detected, detected frame data is saved or transferred for next processing step. When FSM detection is failed, Flywheel state is started.

2.1.4 Flywheel State

Flywheel State detects FSM by moving frame size step in data. When FSM detection fails, the flywheel count value increases by 1. When flywheel count value exceeds the defined value, the state transition to Search State begins. When FSM detection is done, Lock state begins. The frame data during Flywheel state is saved or transferred to next processing step.

2.2 SSE2 (Streaming SIMD Extensions2)

SIMD is the one of three performance enhancement factor in MMX technology. With the help of this, just single instruction can be used for iterative loop with multiple instructions.

Figure 2 shows how SIMS works. .

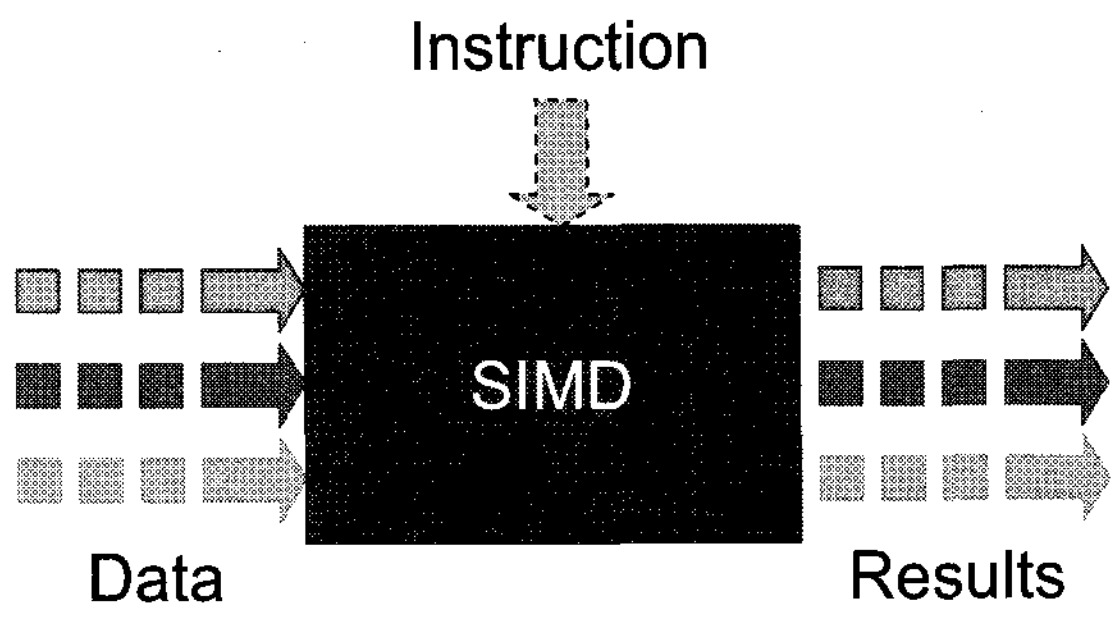

Figure 2 SIMD Processing

3. ASSUMPTIONS IN IMPLEMENTATION

Both Bit Pattern value and modulation/demodulation scheme are thoroughly considered before implementation of frame synchronization software.

3.1 Bit Pattern

Bit Pattern values recommended by CCSDS were considered. Table 1 shows possible 5 pattern values.

Table 1 Frame Synchronization Bit Patterns

Name	Value
For non-turbo coded data	1ACFFC1D
For rate-1/2 turbo coded data	034776C7272895B0
For rate-1/3 turbo coded data	25D5C0CE8990F6C9461BF79C
For rate-1/4 turbo coded data	034776C7272895B0 FCB88938D8D76A4F
For rate-1/6 turbo coded data	25D5C0CE8990F6C9461BF79C DA2A3F31766F0936B9E40863

In this paper, simple 32bit FSM value of "1ACFFC1D" without Turbo coding was used.

Figure 3 shows bit pattern in FSM.

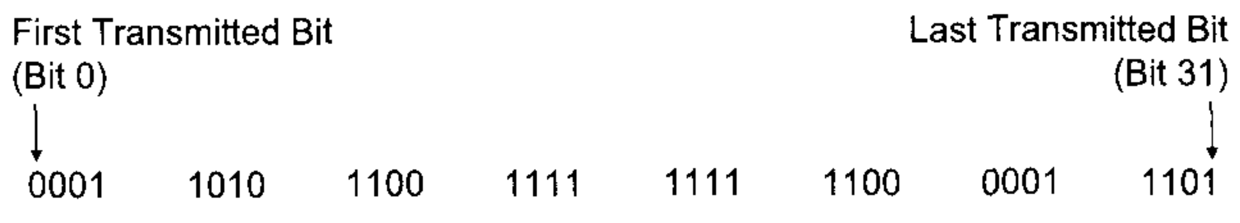

Figure 3 FSM for non-turbo coded data

3.2 Modulation/Demodulation

Widely used modulation scheme in data transmission in satellite like MTSAT-1R, MSG, METOP, and KOMPSAT is QPSK (Quadrature Phase Shift Keying). QPSK is one of PSK (Phase Shift Keying) and uses 4 phase values to express the 2bits information. 4 phase means 0°, 90°, 180°, 270°.

In this paper, FSM detection in QPSK modulation was considered.

4. FRAME SYNCHRONIZATION PERFORMANCE ENHANCEMENT

4.1 Pending Buffer Concept

Pending Buffer Concept is used not to lose the data in Input Data Buffer when new Input Data Buffer data arrives before finishing the FSM Search State.

Figure 4 shows the useful concept of Pending Buffer method.

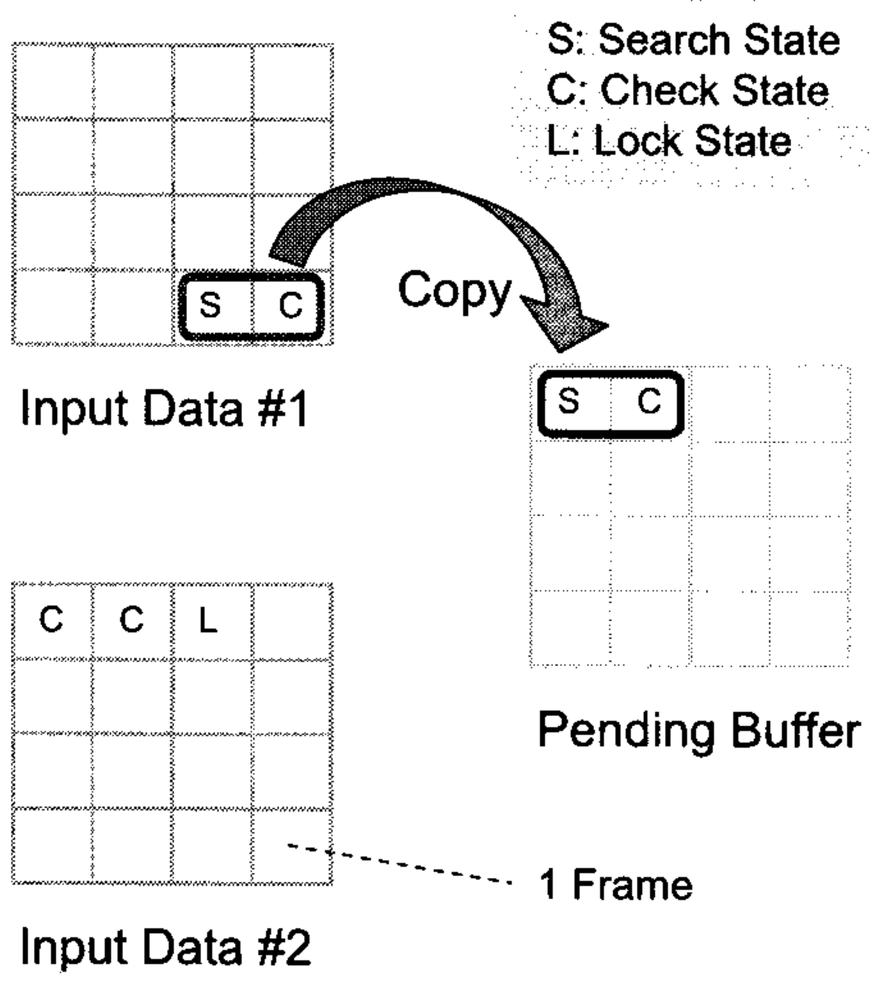

Figure 4 Pending Buffer Method

Input Data#1 is data under processing while Input Data#2 is data for next processing. In case Search State Frame and Check State Frame are in Input Data#1, these two frame data are copied and FSM search continues. To minimize the data loss, this kind of Pending Buffer concept can be used.

4.2 Pattern Search

Pattern Search is to check if input data is made of same data bits. When there is no downlink data transmission, the output data pattern from receiver to serial telemetry card is "00" or "FF."

In this case, it is not necessary to find FSM in Search State.

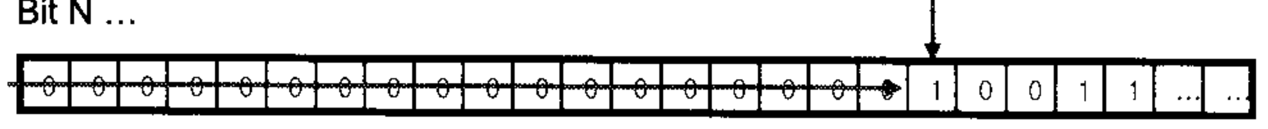

Figure 5 Pattern Search

Pattern Search saves first 4bytes data and compare with 4bytes value after 1bit shift and then if these two values are same, no data from satellite is assumed.

As shown in Figure 5, Search State begins to search FSM value after data bit value 1. This pattern search concept is useful when no data transmission and data loss are expected.

4.3 Bits Threshold with SSE2

In Search State, the nearest Hamming distance value under Bits threshold for 4 possible FSM in QPSK is FSM. In case of no FSM, 1bit shift operation and hamming distance calculation continue until successful FSM search. These two activities of bit shift and hamming distance calculation require huge computational load.

SSE2 was applied to quick comparison between input data and 4 possible FSM values. With SSE2 technology, simple one instruction command lead equivalent effects obtained 4 times individual commands.

Consequently, FSM can be detected by comparing the bit threshold value and FSM Hamming distance after calculating.

To apply the SSE2 in bit threshold comparison sequence, all 4 possible FSM value of "1ACFFC1D" were calculated in advance and saved in XMM1 register and 4 bytes of data in input data buffer was saved in XMM2 register up to 4 times and finally Hamming distance calculation between these two XMM were calculated.

When the Hamming distance is less than the Bits threshold, we can see FSM detection was successful. But when there is no Hamming distance less than threshold, the 1bit shift operation is done and the new Hamming distance calculation and comparison sequence is started until successful FSM detection.

Using 128-bit XMM register, 4 of phase values are compared with just 1 instruction command and simply we can expect 4-times higher processing power from calculation load's point a view.

Figure 6 shows an example of bits threshold calculation using SSE2.

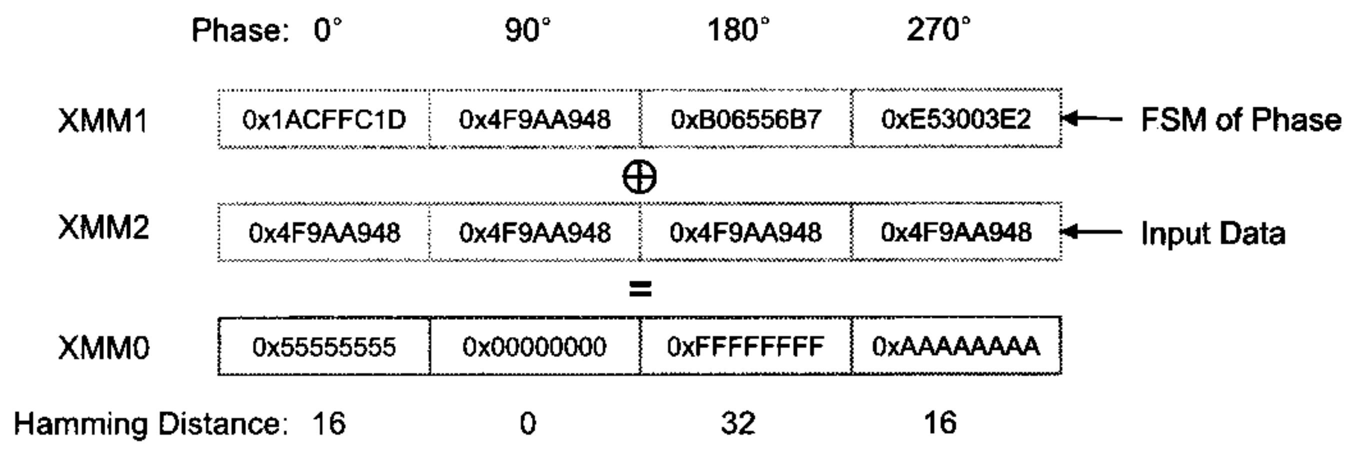

Figure 6 Bits Threshold using SSE2

4-byte value of Input Data Buffer is 0x4F9AA948. XMM1 includes 4 kinds of FSM for different phases. XMM2 includes 4 times 4-byte data of 0x4F9AA948. XMM1 and XMM2 are just exclusive OR-ed and its results are recorded in XMM0 and its hamming distance was calculated. If value of 2 is configured to bits threshold, the 4-byte of FSM showing its Hamming distance under value of 2 is FSM. In Figure 6, phase is 90° and FSM value is 0x4F9AA948.

5. RESULT

The processing speed for (1) Frame Synchronization Software with SSE2 and (2) Frame Synchronization Software without SSE2 were measured. The computer specification for test was shown in Table 2.

Name	Description
CPU	Intel Core2 Duo E6700
Memory	2GB
OS	Windows XP

Table 2 Test Environment

256-byte of Frame data were used for input test data. Input file was made of multiple number of 256-byte frame data.

Input file data was fed into software using configurable speed from 48.8Mbytes to 488.3Mbytes for simulating real operational environment.

Test results are shown in Table 3 and Figure7.

Input Data Size(Kbytes)	Applied SSE2 (Mbps)	Not-applied SSE2 (Mbps)
50,000	301	62
100,000	292	69
150,000	298	66
200,000	279	70
250,000	300	60
300,000	303	59
350,000	317	57
400,000	310	60
450,000	309	59
500,000	314	59

Table 3 Comparison between Applied SSE2 and Not-applied SSE2

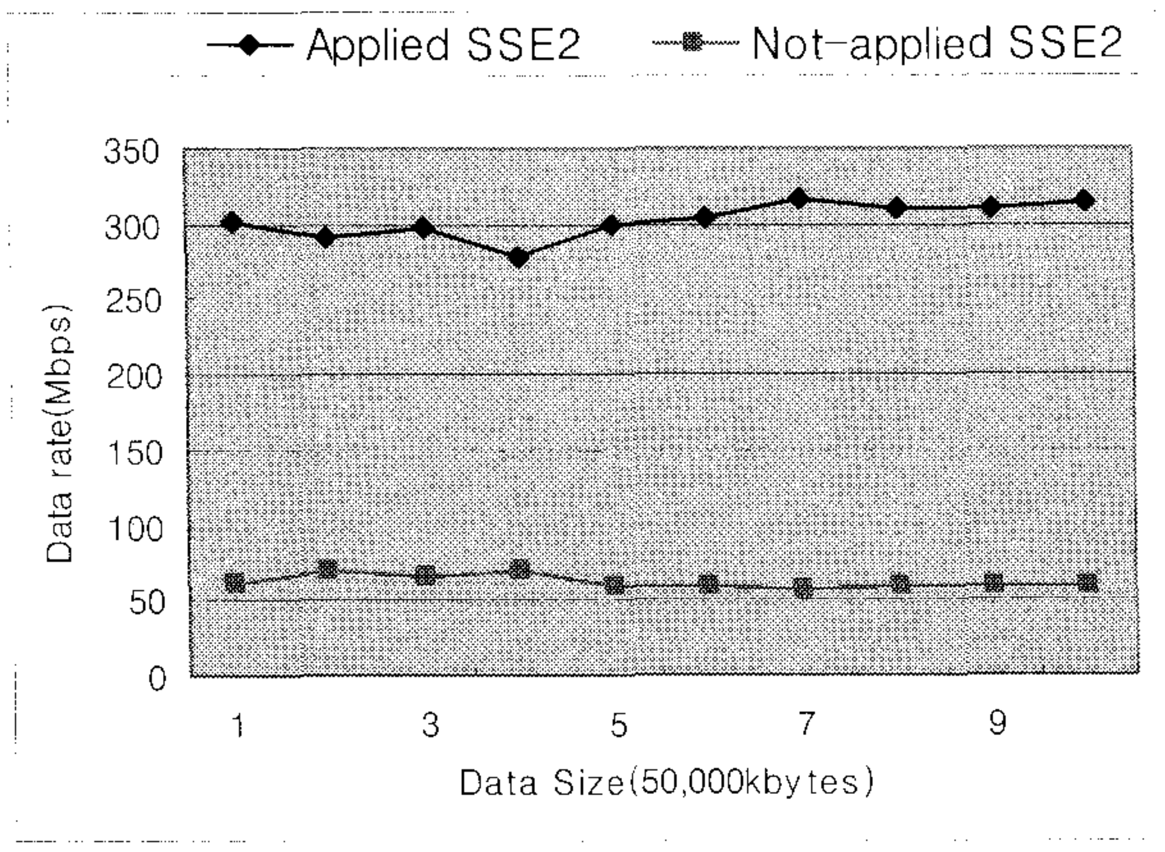

Figure 7 Comparison between Applied SSE2 and Not-applied SSE2

From test results, we found the data size per sec, processing speed, was independant to input data size.

Average speed for SSE2 case reached to about 302.3Mbytes while the speed for non-SSE2 case reached to 62.1Mbps.

This means SSE2 case shows 4.8 times higher performance than non-SSE2 case. But this exceeded the maximum expected performance value of 4. This discrepancy was investigated to be caused by different implementation in software realization.

6. CONCLUTION

This paper shows software frame synchronizer implementation using both SSE2 for high speed processing and pending buffer concept for no data loss, which is one of drawback in hardware frame synchronizer.

The high speed software frame synchronizer with SSE2 can provide several benefits like expandability and update which is, in fact, inherent to software.

The scheme in this paper can be applied to not only satellite communication but also other digital communications.

7. REFERENCES

[1]CCSDS, 'TM Synchronization and Channel Coding', Issue 1, Sep. 2003
[2]The Software Vectorization Handbook, Aart J.C.Bik
[3]IA-32 Intel® Architecture Software Developer's Manual Volume 2A: Instruction Set Reference, N-Z