# Extending STL BASOPs Used in 3GPP Codecs to Leverage Features of Modern DSP Architectures

Ajay Homkar*, Satish Patil*, Lukman Rahumathulla*, Raj Pawate†, Sachin Ghanekar*
Cadence Design Systems Inc.,*Pune, India, † San Jose, USA
{ahomkar, patils, lukmanr, pawateb, ghanekar}@cadence.com

***Abstract*—All 3GPP reference speech, audio and hybrid codecs are implemented using a basic set of operators standardized previously by ITU-T and part of the software tools library (STL). The last major update to the basic operators (BASOPs) was made in 2009. Since then modern DSP architectures with features such as VLIW, SIMD, and 64-bit wide accumulators have become prevalent. Over the years, DSP architectures and processing algorithms have evolved to provide better precision at lower bit-rates and to handle increased coding complexities that have resulted in improved speech and audio quality. To keep in sync with these trends, we contributed to the 3GPP SA4 standards committee with a proposal to update the STL BASOPs. To demonstrate the impact of STL update, reference c-code of EVS codec was implemented using updated STL BASOPs. With the proposed EVS implementation, we are able to achieve 1.25x improvement in cycles, 1.1x improvement in WMOPS along with 100 KBytes reduction in memory footprint.**

***Index Terms*—BASOP, speech codecs, DSP, WMOPS, MCPS.**

## I. Introduction

STL is a standardized mechanism from ITU-T [1] to enable efficient implementation of various speech/audio coding algorithms specified by 3GPP. STL library defines a set of BASOPs along with the complexity weight for each operator. This acts as a set of common, coherent and portable signal processing tools and is utilized in all 3GPP codecs. This common set helps for better structuring of the code and enables the architecture independent complexity comparison and characterization of the codec. Considering the improvements in DSP architectures, there was gap w.r.t. modern DSP architectures and modern coding schemes and hence STL BASOPs needed an update. The extended set of BASOPs is proposed and these new operators are applied to EVS codec. The updated code is referred as modified EVS C-code in this work as it is "algorithmically" identical to reference code and can map efficiently on modern DSP architectures.

The paper is organized as follows. It begins with the motivation for extending STL 2009 library. Section III introduces designing extensions to BASOPs. Section IV emphasizes on applying these operators to example codecs before concluding the paper in section V.

## II. Motivation for extending STL 2009 BASOPs

At the 3GPP SA4 meeting #94 in Sophia Antipolis, a new study item (SI) was started to investigate benefits of updating the ITU-T fixed-point BASOPs [2]. Objectives of the SI were to study state-of-the-art DSP architectures and assess how closely STL 2009 BASOPs reflect them, to identify the gaps and recommend ways of improvement. This paper summarizes our findings and proposal made to ITU-T to update the STL BASOPs.

It appears that the 2009 version of the BASOPs was influenced by older DSP architectures where the accumulator was 40-bits wide. However, a survey of the state-of-the-art processor architectures [3] shows that most of them (recent ones from Intel, ARM, Qualcomm, TI, Tensilica etc.) support 64-bit wide accumulators and registers, single instruction multiple data (SIMD) and very long instruction word (VLIW) features. Wider accumulator enables additional guard bits and eliminates the need for saturation check after every operation. SIMD support enables processing of two 32-bit or four 16-bit data elements in parallel. Several operations to be executed in single cycle with VLIW feature.

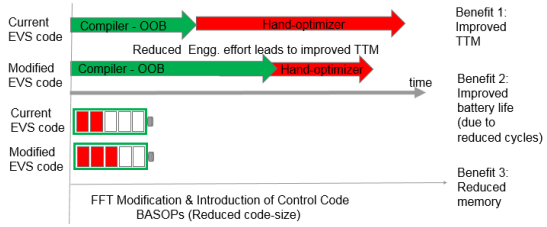


Figure 1. Benefits of proposal

BASOPs that are friendlier to compilers, and enable above features to be leveraged, can significantly reduce implementation time as seen in Figure 1. Improved compiler technology and software development tools interpret data types and associated BASOPs to map them to a processor architecture for better out-of-box (OOB) performance. OOB compilation is a processes to port the code into one architecture without any extra optimization efforts. Without this computer assisted optimization, an engineer would have to hand-optimize the code which would result in increased engineering effort and longer time to market. Due to these reasons, it was concluded that, there is a need for updating STL 2009 library with additional BASOPs and data types.

## III. Designing extensions to BASOPs to leverage features of modern DSP architectures

Based on above analysis [2], to fill the gap between STL 2009 BASOPs and modern DSP architectures, around 100 BASOPs have been added. The new propositions will be referred as STL 2017 in this paper. Wider accumulator, complex

2380-6923/19/$31.00 ©2019 IEEE
DOI 10.1109/VLSID.2019.00120

operators, control code BASOPs and enhancement to 32-bit operators are new additions to STL 2017. 64-bit operators helps to omit the intermediate truncation and saturation since additional guard bits are available for multiply and accumulate (MAC) like operations, which helps to improve the overall precision in FIR like kernels. Advanced DSP algorithms make extensive use of complex FFT, complex FIR and complex arithmetic. Complex BASOPs have devised to address this challenge and are designed to support 16/32-bit complex arithmetic. 32-bit multiplication/MAC operations can be performed in single cycle using enhanced 32-bit operators. Control code BASOPs perform the conditional checks without doing any arithmetic operation.

Following simple examples highlights the issues with STL 2009 BASOPs and how the updated BASOPs enables features of modern DSPs to reduce the cycles.

### A. Dot product illustration to verify usefulness of new BASOPs

Dot product is very commonly used kernel in many DSP algorithms. Following pseudo code shows the implementation of dot product for 4 element of $a$ and $b$ arrays with 32-bit accumulator.

```
/* Regular implementation using STL 2009 */
Int_32 acc;
acc = saturate32(a0*b0);
acc = saturate32(acc + saturate32(a1*b1));
acc = saturate32(acc + saturate32(a2*b2));
acc = saturate32(acc + saturate32(a3*b3));
```

Here, we cannot break the sequence of operations since the saturation and truncation is involved in each MAC operation. This prohibits the use of SIMD and VLIW features. Total 4 cycles are needed for processing four elements of array $a$ and $b$. Same implementation in 2 slots VLIW architecture can be accommodated in 3 cycles as shown below:

```
/* written for 2 slots VLIW architecture */
Int_64 Acc1, Acc2, Acc;
acc1 = a0*b0; /* slot 0, cycle 1 */
acc2 = a1*b1; /* slot 1, cycle 1 */
Acc1 = acc1 + a2*b2; /* slot 0, cycle 2 */
Acc2 = acc2 + a3*b3; /* slot 1, cycle 2 with */
Acc  = acc1 + acc2; /* slot 0, cycle 3 */
```

64-bit accumulator helps to avoid intermediate truncation and saturation. Hence, $acc1$ and $acc2$ can go in parallel in 1 cycle. With additional SIMD feature available, $acc1$ and $acc2$ can be omitted hence all of this can be performed in just 2 cycles. In a nutshell, If $N$ element dot product needs $N$ cycles with STL 2009 operators, then same can be accommodated in $(N/4 + 1)$ cycles with SIMD and VLIW support.

### B. FFT implementation using STL 2017 BASOPs

FFT is the integral part of any signal processing algorithm. Figure 2 show performance comparison of 32x16 precision FFT library in EVS codec. With use of updated BASOPs, improvement of 3.36x and 3.04x is observed in average cycles and codesize respectively. Complex and control code BASOPs are primarily responsible for codesize reduction. This proves that updated BASOPs not only helps in cycle improvement but also in codesize reduction.

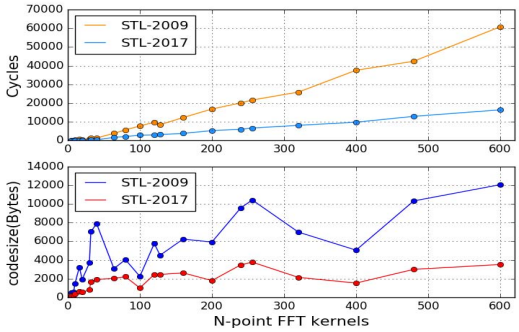


Figure 2. FFT performance comparison

## IV. Applying Extended BASOPs to Example Codecs

We implemented EVS codec [4], [5] from 3GPP with STL 2017 BASOPs. We observed improvement in cycles and memory. Table I shows the gain observed with the modified EVS implementation. Functions to use these BASOPs are shortlisted based on codesize and cycle contribution criterion. More benefit in Table I would have been possible if the quality and interoperability constraints were not there to begin with. But we had to keep interoperability with legacy codecs in mobile applications.

TABLE I
Impact of updated STL operators on EVS codec

| | Reference EVS | Modified EVS |
|---|---|---|
| Codesize GCC[a] (KBytes) | 2367.4 | 2347.9 |
| Codesize XCC[b] (KBytes) | 2107.1 | 2032.8 |
| Average WMOPS | 77.5 | 66.9 |
| Average OOB MCPS | 269 | 162.5 |

[a] GCC version-4.4.7 20120313; [b] Cadence Xtensa XCC compiler

## V. Conclusion

Modern speech and audio codecs such as EVS can benefit from VLIW, and SIMD features of modern DSP architectures. The earlier set of BASOPs, influenced by a previous generation of speech codecs and DSPs, resulted in implementations that consumed more cycles and power. Automation tools and compilers could not easily leverage these earlier BASOPs and implementors were forced to hand optimize the code. The new BASOPs are friendlier to automation tools and compilers resulting in a significant reduction in cycles right out of the box reducing hand optimization efforts and time to market. The extended set of BASOPs is approved by 3GPP and is available at www.3gpp.org/dynareport/26973.htm

## References


[1] Recommendation ITU-T G.191, "Software tools for speech and audio coding standardization" available at: https://www.itu.int/rec/T-REC-G.191-201003-I/en

[2] S4-170915, Proposal for extending STL2009 basic operators for modern DSP architectures.

[3] S4-170670, Study on Update to fixed-point basic operators.

[4] 3GPP TS 26.442, Codec for Enhanced Voice Services (EVS); ANSI C code (fixed-point).

[5] Dietz, Martin, et al. "Overview of the EVS codec architecture." Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on. IEEE, 2015.