短時間スペクトラム解析の計算アーキテクチャの検討

工学部　電気電子工学科
11306032
佐々木秀

OUTLINE

- 研究背景
- 問題提起と提案手法
- 計算アルゴリズムと回路図
    1. 乗算器を使用した構成
    2. 乗算器を減らした構成
    3. 乗算器を使わない構成
- 結果
- まとめと今後の課題

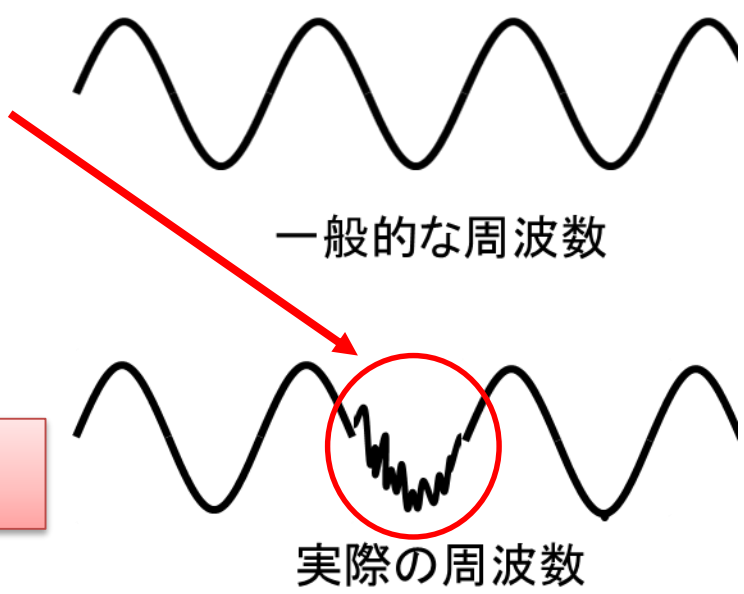
OUTLINE

- 研究背景
- 問題提起と提案手法
- 計算アルゴリズムと回路図
    1. 乗算器を使用した構成
    2. 乗算器を減らした構成
    3. 乗算器を使わない構成
- 結果
- まとめと今後の課題

研究背景

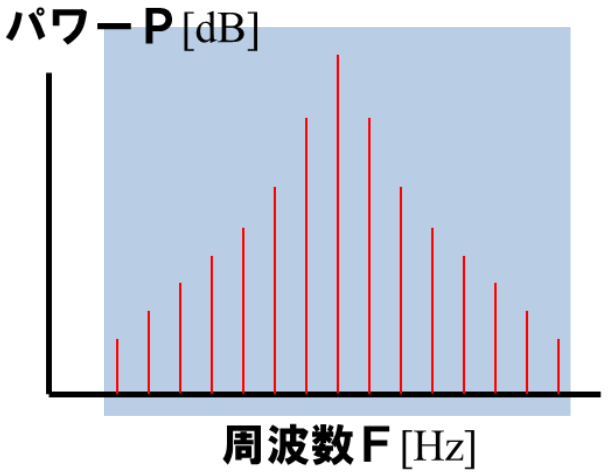
～周波数～
時間によらず統計的性質が一定の波形

ノイズや信号の過度状態による急激な変化

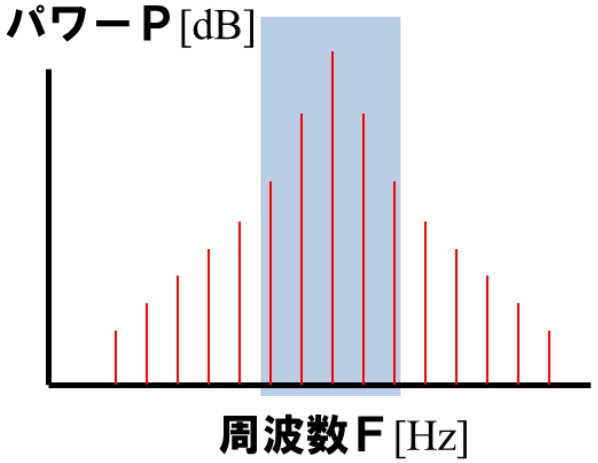


FFT(高速フーリエ変換)で解析

研究背景



FFTによる測定範囲



目的の測定範囲

～FFTによる測定～

- 短い過渡時間で多くのサンプリング点が必要
- 周波数全体を測定

- 高速サンプリングが必要
- 計算時間とコストがかかる

目的の範囲だけを
測定したい！！

研究目標

少ない
サンプリング
点数

必要な周波数
範囲のみ測定

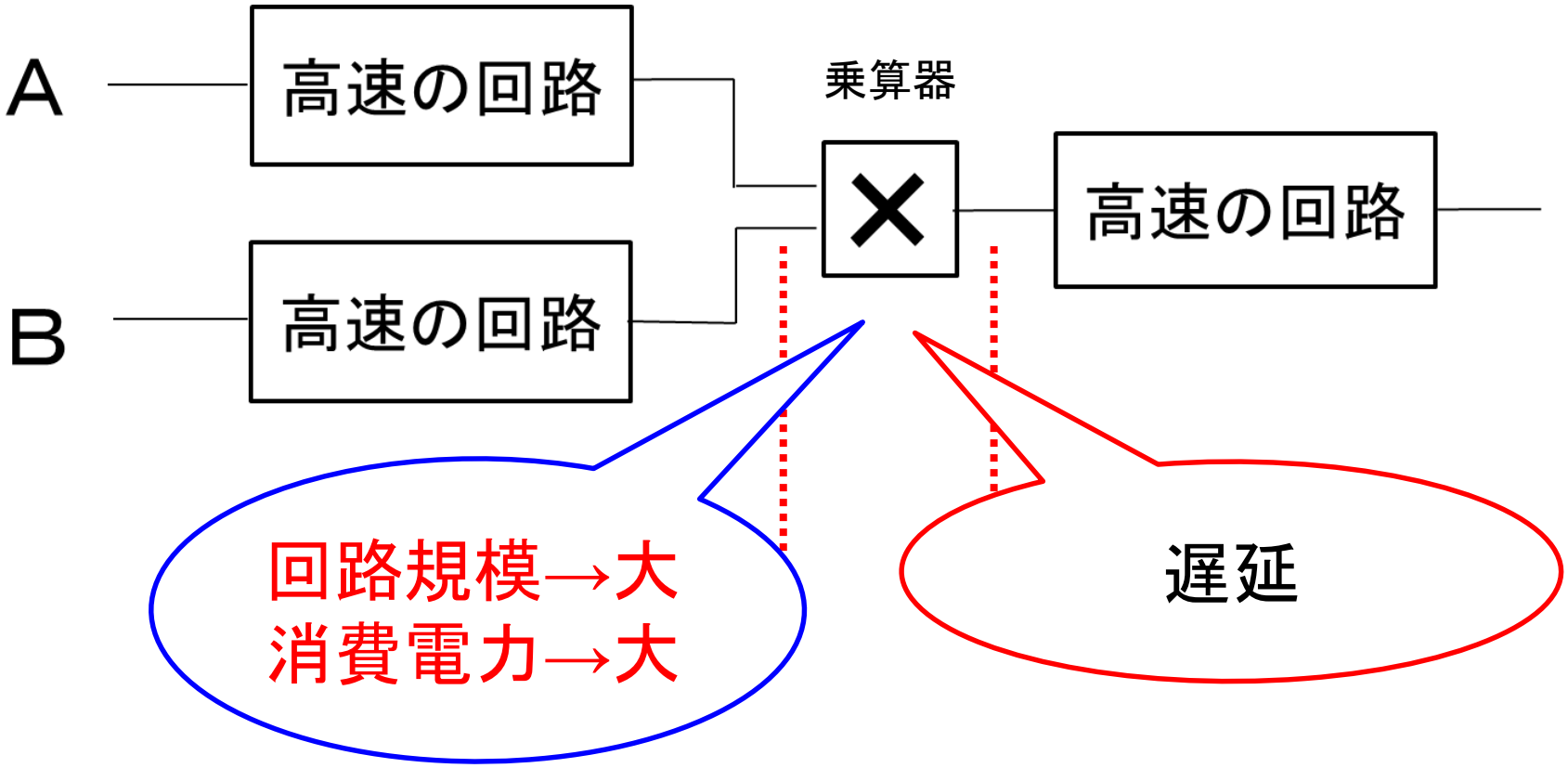
任意の細かい周波数分解能
で計算するアルゴリズム

少量回路・低消費電力でリアルタイム高速計算の
FPGA実現する構成を検討

OUTLINE

- 研究背景
- 問題提起と提案手法
- 計算アルゴリズムと回路図
    1. 乗算器を使用した構成
    2. 乗算器を減らした構成
    3. 乗算器を使わない構成
- 結果
- まとめと今後の課題

乗算器はボトルネック



回路全体の計算時間→大

提案手法

1. 乗算器を使用したハードウェア構成

入力をそのまま計算

2. 対数変換を用いたハードウェア構成

入力を対数変換して計算

3. 乗算器を使用しないハードウェア構成

加算器、減算器、LUTを用いて計算

提案手法

1．乗算器を使用したハードウェア構成

入力をそのまま計算

2．対数変換を用いたハードウェア構成

入力を対数変換して計算

3．乗算器を使用しないハードウェア構成

加算器、減算器、LUTを用いて計算

LUT（Look Up Table）とは

複雑な信号処理 → 計算時間がかかる

LUTを使うと・・・・・

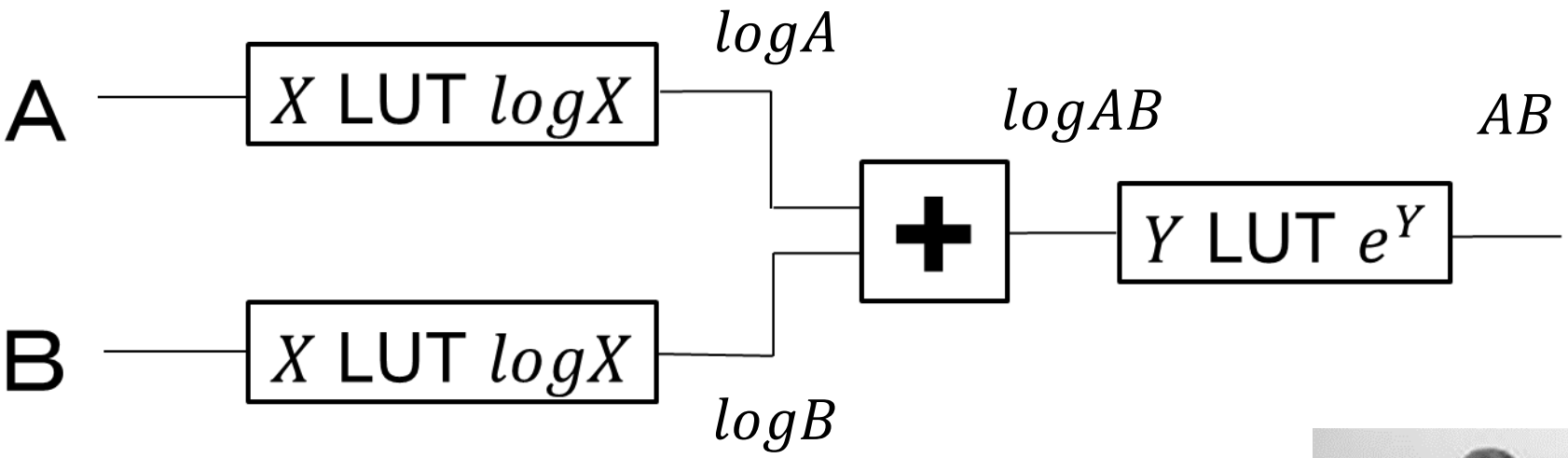
計算せずに出力

単純な配列の参照処理に置き換わり、効率化

※Sin波やCos波、2乗式もLUTによって処理が可能

対数を用いた方式

乗算を、対数を用いて計算する方式



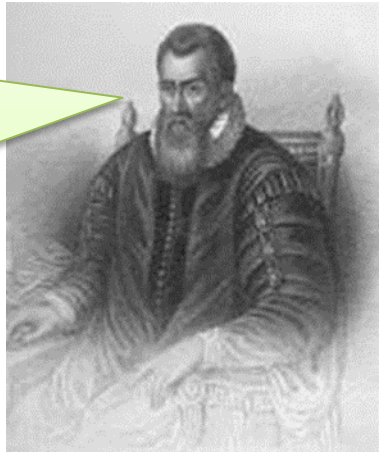
高精度で対数、指数を得るためには
ビット数
LUTサイズ
大

対数の生みの親
ジョン・ネイピア

乗算器を変換する方法

$$AB = \frac{1}{2}\left[(A+B)^2 - A^2 - B^2\right]$$

その発想は
無かった

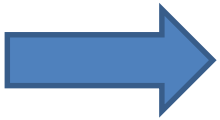


この式を使い、乗算器を加算器と減算器へ変換

2乗式はLUTを使い代用

2倍、1/2倍はシフト演算によって計算

乗算器１つ(24bit)
加算回数：24回



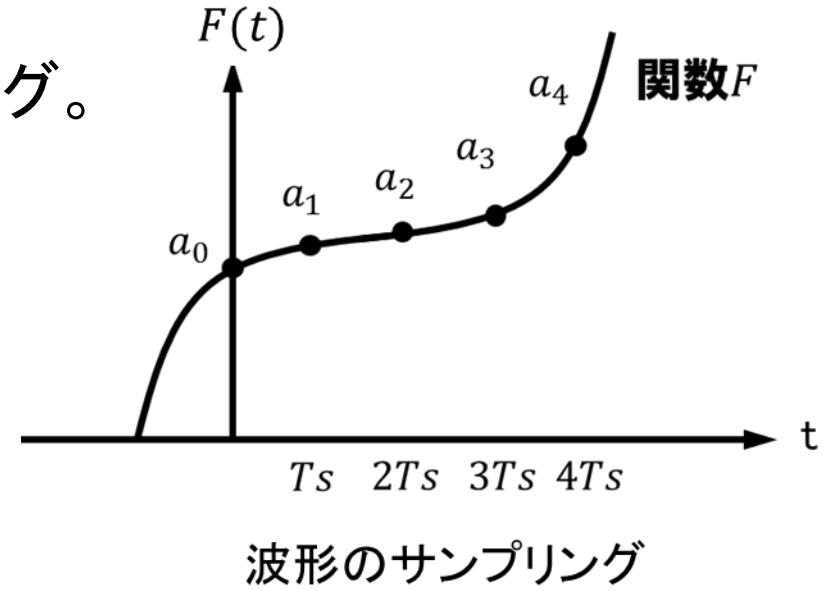
加減算回数：3回
LUT参照：3回

OUTLINE

- 研究背景
- 問題提起と提案手法
- 計算アルゴリズムと回路図
  1. 乗算器を使用した構成
  2. 乗算器を減らした構成
  3. 乗算器を使わない構成
- 結果
- まとめとこれからの課題

計算式の導出

波形を一定時間(Ts)間隔でサンプリング。
それらのサンプリング点を順に
データa_0, a_1, a_2, a_3, a_4とおく。



波形のサンプリング

$$F(j\omega)=a_0 + a_1 e^{-j\omega Ts} + a_2 e^{-j\omega 2Ts} + a_3 e^{-j\omega 3Ts} + a_4 e^{-j\omega 4Ts}$$

特定帯域 $\omega_1 < \omega < \omega_2$で細かな周波数分解能 $\Delta\omega$ で計算

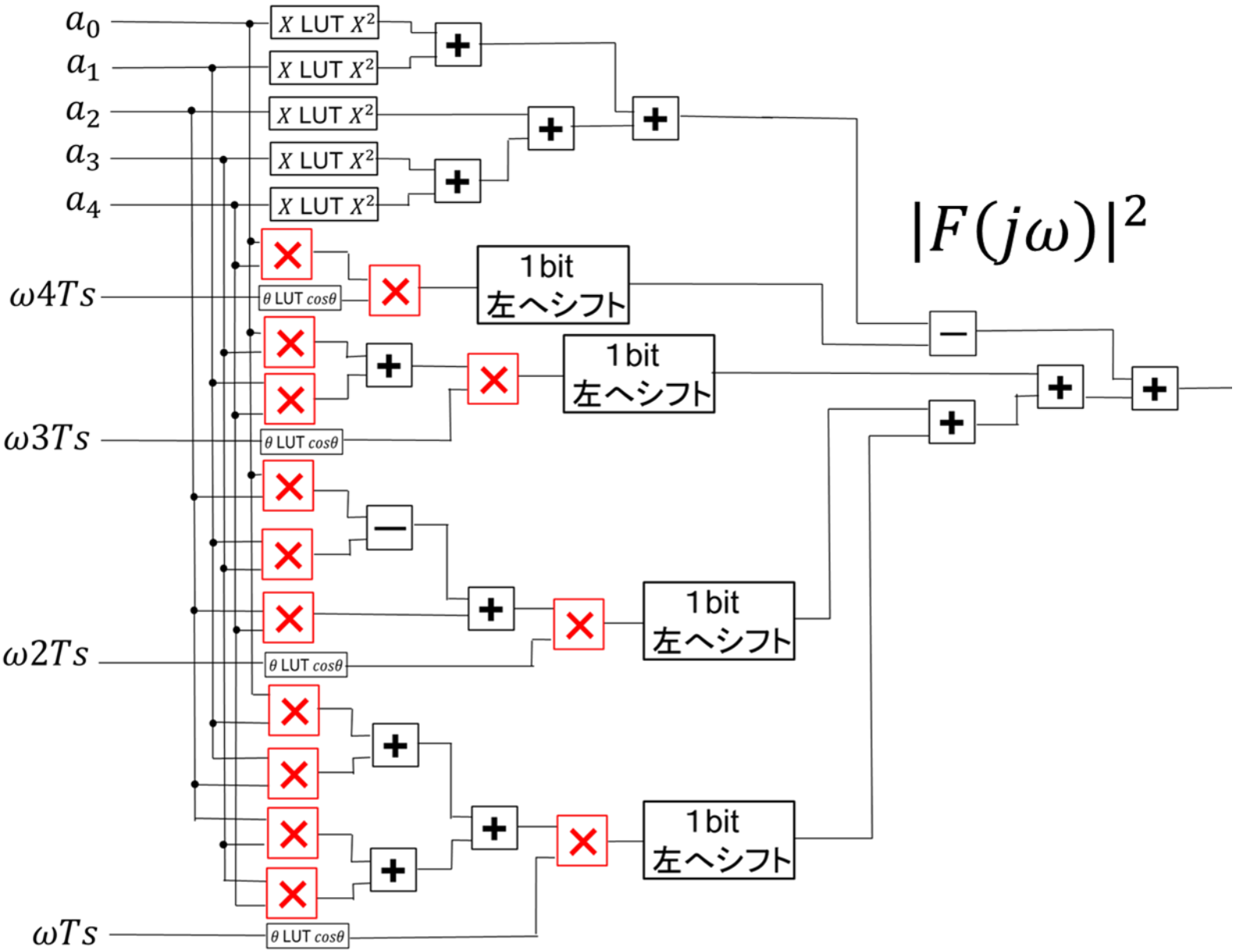
計算式の導出

$$F(j\omega)=a_0 + a_1 e^{-j\omega Ts} + a_2 e^{-j\omega 2Ts} + a_3 e^{-j\omega 3Ts} + a_4 e^{-j\omega 4Ts}$$

この式をオイラーの公式を用いて整理すると

$$\begin{aligned}|F(j\omega)|^2 = a_0^2 + a_1^2 + a_2^2 + a_3^2 + a_4^2 - 2a_0a_4\cos(\omega 4Ts) \\ +2(a_0a_3 + a_4a_1)\cos(\omega 3Ts) \\ +2(a_2a_0 - a_1a_3 + a_2a_4)\cos(\omega 2Ts) \\ +2(a_0a_1 + a_1a_2 + a_2a_3 + a_3a_4)\cos(\omega Ts)\end{aligned}$$

回路図1（乗算器を使用）


a_0
a_1
a_2
a_3
a_4
$\omega 4Ts$
$\omega 3Ts$
$\omega 2Ts$
ωTs
X LUT X^2
θ LUT cosθ
1 bit 左へシフト
$|F(j\omega)|^2$

OUTLINE

- 研究背景
- 問題提起と提案手法
- 計算アルゴリズムと回路図
  1. 乗算器を使用した構成
  2. 乗算器を減らした構成
  3. 乗算器を使わない構成
- 結果
- まとめと今後の課題

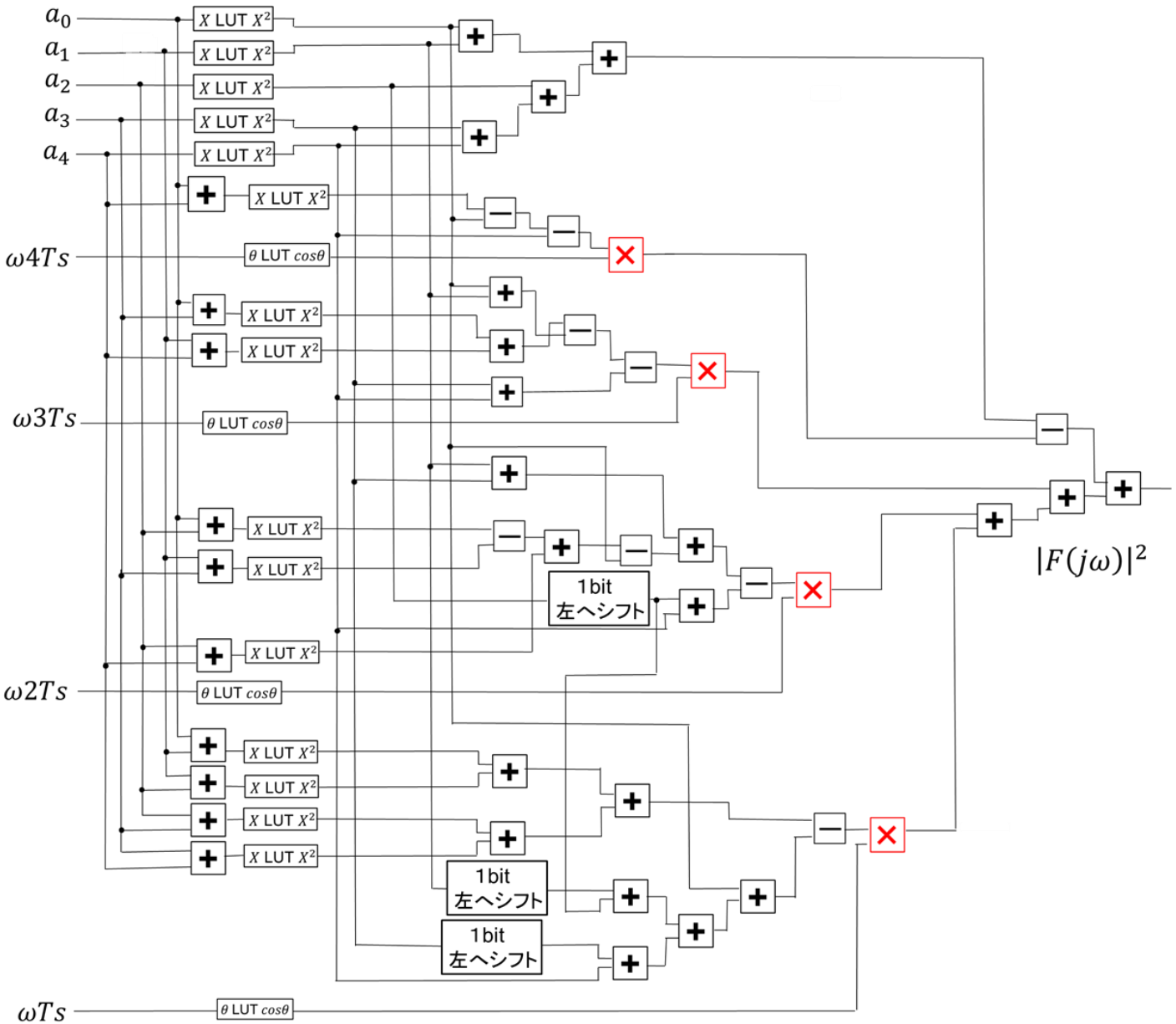
計算式の導出

$$AB = \frac{1}{2}[(A+B)^2 - A^2 - B^2]$$

の式を用いて式を整理すると

$$\begin{aligned}
&|F(j\omega)|^2 \\
&= a_0^2 + a_1^2 + a_2^2 + a_3^2 + a_4^2 - \{(a_0 + a_4)^2 - a_0^2 - a_4^2\}\cos(\omega 4Ts) \\
&+ \{(a_0 + a_3)^2 + (a_1 + a_4)^2 - a_0^2 - a_1^2 - a_3^2 - a_4^2\}\cos(\omega 3Ts) \\
&+ \{(a_0 + a_2)^2 - (a_1 + a_3)^2 + (a_2 + a_4)^2 - a_0^2 + a_1^2 - 2a_2^2 + a_3^2
\end{aligned}$$

回路図2（乗算器が最小限）



OUTLINE

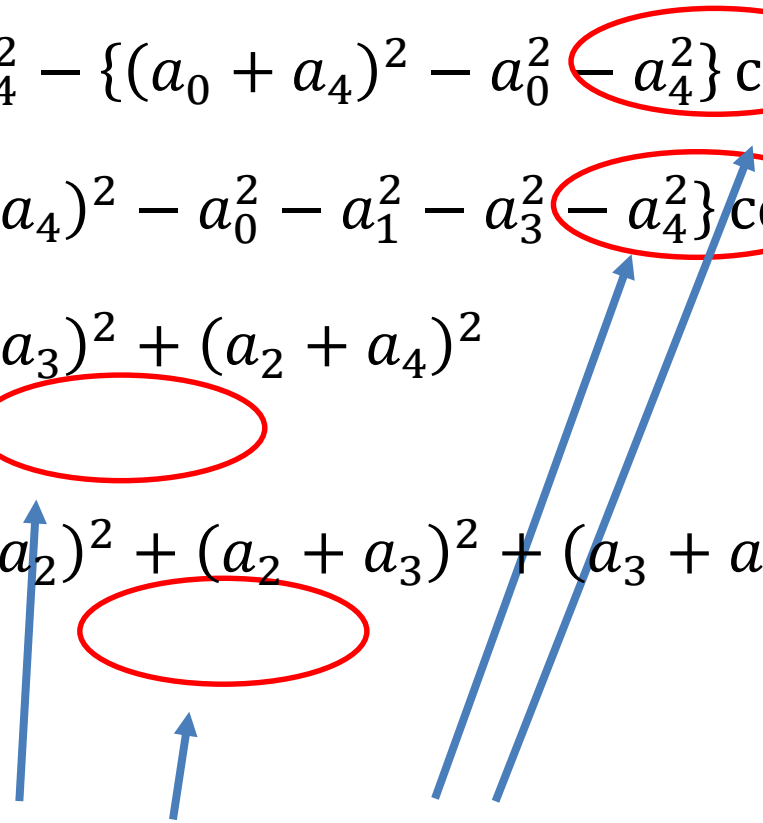
- 研究背景
- 問題提起と提案手法
- 計算アルゴリズムと回路図
  1. 乗算器を使用した構成
  2. 乗算器を減らした構成
  3. 乗算器を使わない構成
- 結果
- まとめと今後の課題

計算式の導出

$$
\begin{aligned}
&|F(j\omega)|^2 \\
&= a_0^2 + a_1^2 + a_2^2 + a_3^2 + a_4^2 - \{(a_0 + a_4)^2 - a_0^2 - a_4^2\}\cos(\omega 4Ts) \\
&+ \{(a_0 + a_3)^2 + (a_1 + a_4)^2 - a_0^2 - a_1^2 - a_3^2 - a_4^2\}\cos(\omega 3Ts) \\
&+ \{(a_0 + a_2)^2 - (a_1 + a_3)^2 + (a_2 + a_4)^2 - a_0^2 + a_1^2 - 2a_2^2 + a_3^2
\end{aligned}
$$

計算式の導出

$$|F(j\omega)|^2$$
$$= a_0^2 + a_1^2 + a_2^2 + a_3^2 + a_4^2 - \{(a_0 + a_4)^2 - a_0^2 - a_4^2\}\cos(\omega 4Ts)$$
$$+\{(a_0 + a_3)^2 + (a_1 + a_4)^2 - a_0^2 - a_1^2 - a_3^2 - a_4^2\}\cos(\omega 3Ts)$$
$$+\{(a_0 + a_2)^2 - (a_1 + a_3)^2 + (a_2 + a_4)^2$$
$$+\{(a_0 + a_1)^2 + (a_1 + a_2)^2 + (a_2 + a_3)^2 + (a_3 + a_4)^2$$



乗算の部分を変換

計算式の導出

$$|F(j\omega)|^2$$
$$= a_0^2 + a_1^2 + a_2^2 + a_3^2 + a_4^2$$
$$-\frac{1}{2}\{(A + \cos(\omega 4Ts))^2 - (B + \cos(\omega 3Ts))^2 - (C + \cos(\omega 2Ts))^2$$
$$-(D + \cos(\omega Ts))^2 - A^2 + B^2 + C^2 + D^2$$

※乗算が無くなっている

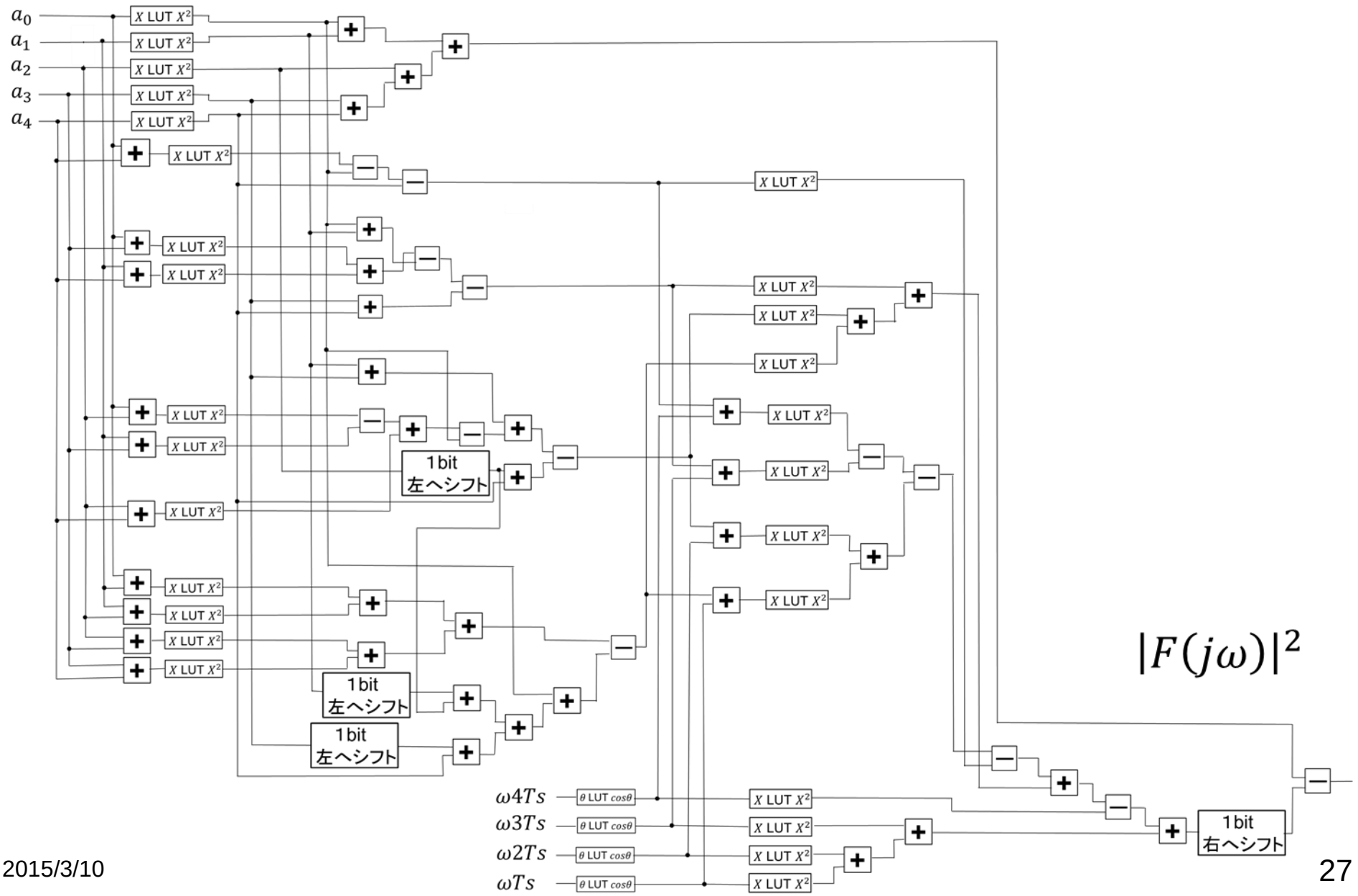
$$A = (a_0 + a_4)^2 - a_0^2 - a_4^2$$

$$B = (a_0 + a_3)^2 + (a_1 + a_4)^2 - a_0^2 - a_1^2 - a_3^2 - a_4^2$$

$$C = (a_0 + a_2)^2 - (a_1 + a_3)^2 + (a_2 + a_4)^2$$
$$- a_0^2 + a_1^2 - 2a_2^2 + a_3^2 - a_4^2$$

$$D = (a_0 + a_1)^2 + (a_1 + a_2)^2 + (a_2 + a_3)^2 + (a_3 + a_4)^2$$
$$- a_0^2 - 2a_1^2 - 2a_2^2 - 2a_3^3 - a_4^2$$

回路図3（乗算器なし）


a_0
a_1
a_2
a_3
a_4
X LUT X^2
1bit 左へシフト
1bit 左へシフト
1bit 左へシフト
$\omega 4Ts$
$\omega 3Ts$
$\omega 2Ts$
ωTs
θ LUT cosθ
1bit 右へシフト
$|F(j\omega)|^2$

OUTLINE

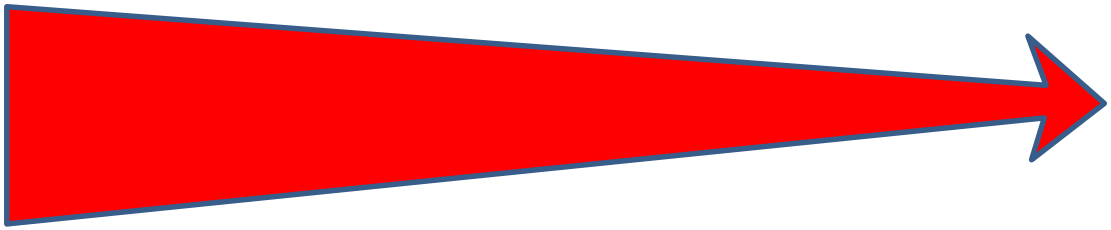
- 研究背景
- 問題提起と提案手法
- 計算アルゴリズムと回路図
    1. 乗算器を使用した構成
    2. 乗算器を減らした構成
    3. 乗算器を使わない構成
- **結果**
- まとめと今後の課題

結果

	回路図１	回路図２	回路図３
乗算器	14	4	0
加算器	12	31	39
減算器	2	9	13
LUT	9	15	27

計算時間と回路規模

大



小

回路図1

回路図2

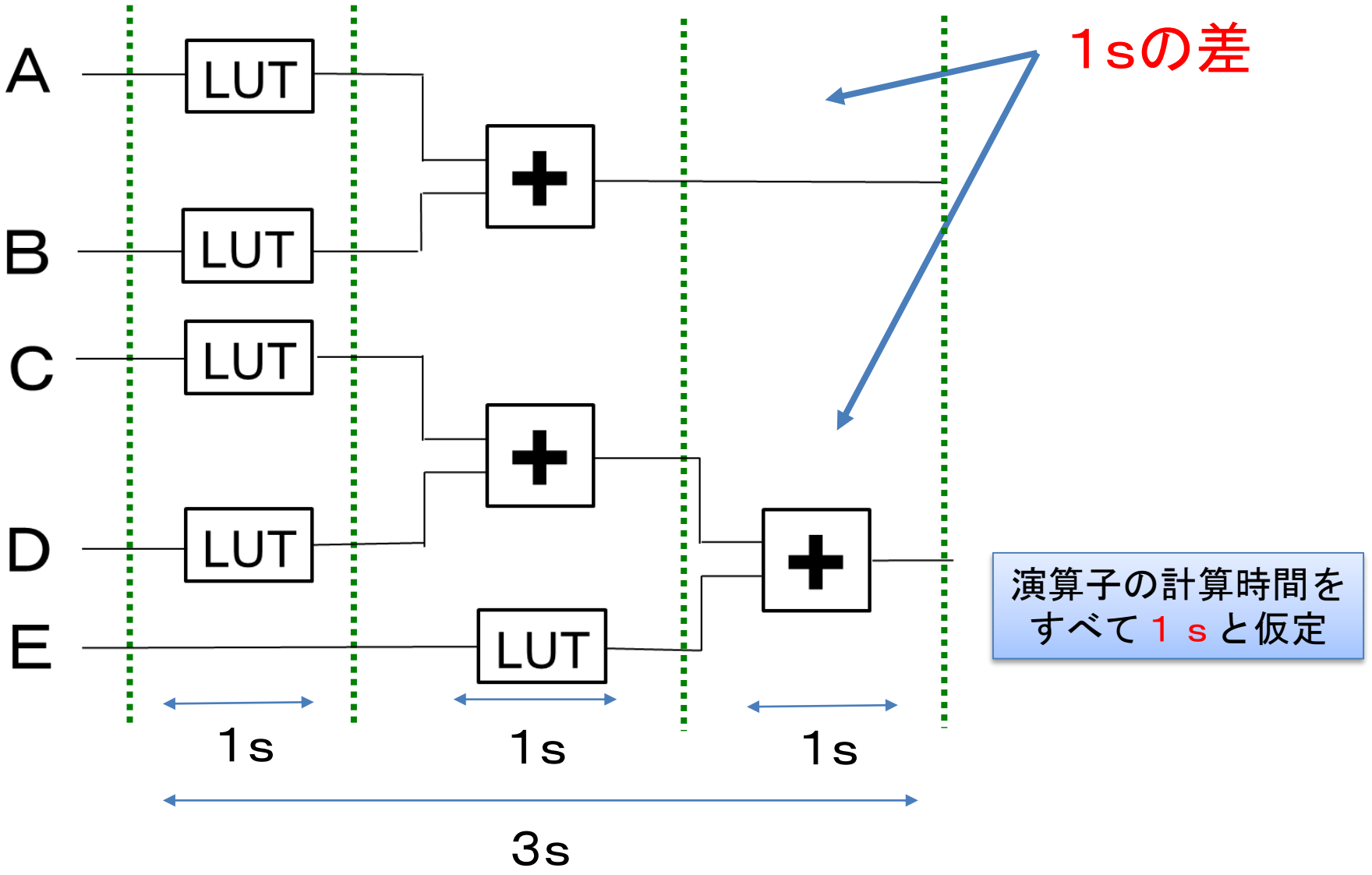
回路図3

結果

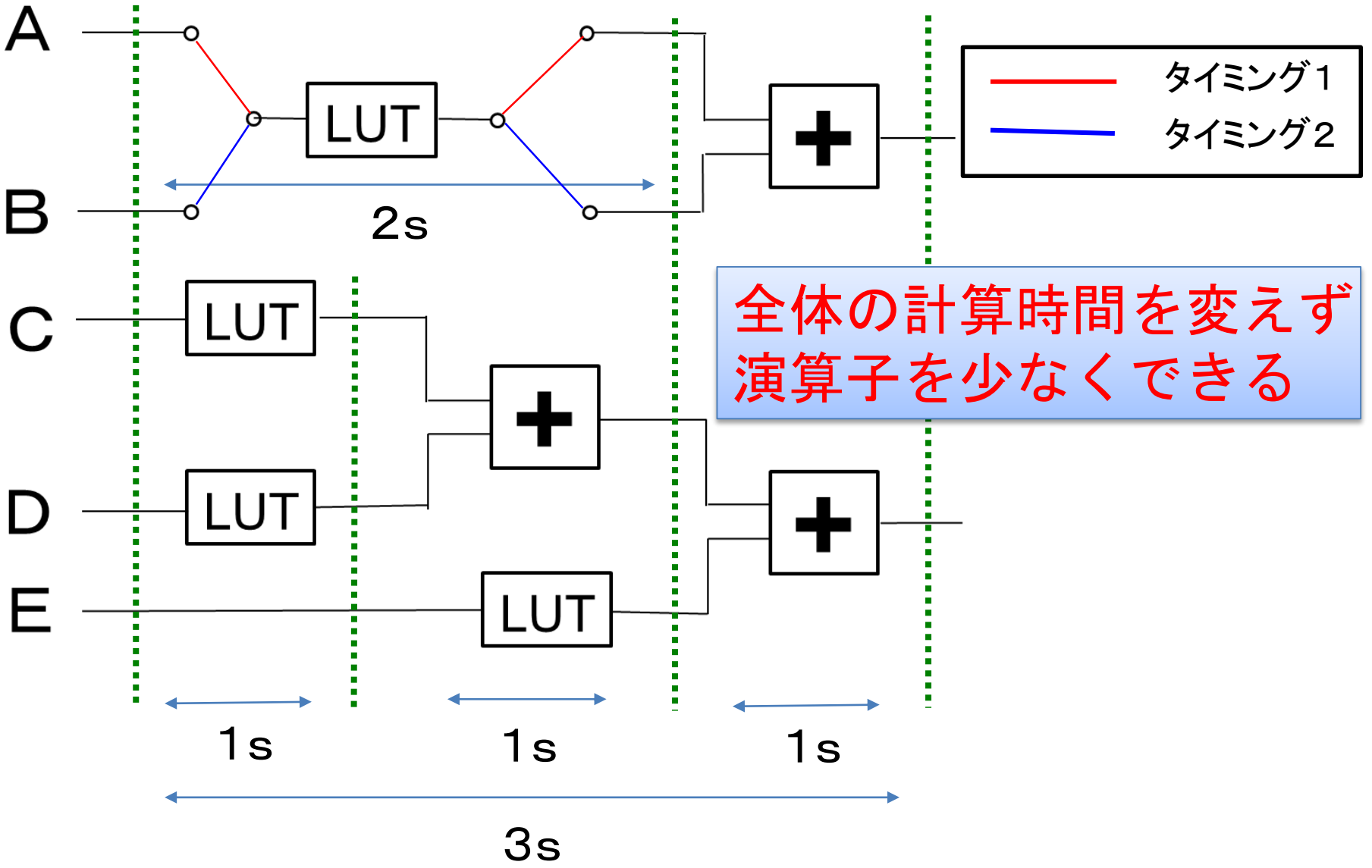
	回路図１	回路図２	回路図３
乗算器	14	4	0
加算器	12	31	39
減算器	2	9	13
LUT	9	15	27

加算器、減算器、LUTの数は
最適化によって減らせる

最適化について



最適化について


A
B
C
D
E
LUT
LUT
LUT
LUT
2s
1s
1s
1s
3s
タイミング１
タイミング２
全体の計算時間を変えず
演算子を少なくできる

OUTLINE

- 研究背景
- 問題提起と提案手法
- 計算アルゴリズムと回路図
    1. 乗算器を使用した構成
    2. 乗算器を減らした構成
    3. 乗算器を使わない構成
- 結果
- まとめと今後の課題

まとめ

・ボトルネックとなる乗算器を使わない計算アルゴリズムとハードウェア構成を検討

ハードウェア構成をFPGA実現

回路規模の縮小
低コスト化
低消費電力化

周波数スペクトル解析がリアルタイムで処理

※回路の正当性

a_0~a_4に適当な値をおいて計算式とブロック図の両方で計算し、結果を比較して検証した。

今後の課題

実際にFPGA実装を行い提案構成の実機検証していく。

・計算精度と演算器・LUTのビット数の明確化

・実装FPGA動作クロック周波数と計算速度の明確化

・データが5点より多くなった場合の計算式と
ハードウェア構成の検討

Q&A

Q：今回の提案手法は他の論文では使われているか

A：自分が調べた限りでは他には使われていない

Q：今回のテーマと乗算器を減らすことが結びつかない

A：従来のFFTで解析するのとは別の新たな手法を検討している。その際に、乗算器を減らすことで計算速度の短縮を図っている。

Q：どの程度の計算時間の短縮が見込めるか

A：まだ検討の最中でお答えすることができません。