課題研究報告書

ＦＰＧＡを用いたオーディオ電子透かしアルゴリズムの実装に関する研究

北陸先端科学技術大学院大学
情報科学研究科

Zhao Xuyang

平成 29 年 3 月

課題研究報告書

ＦＰＧＡを用いたオーディオ電子透かしアルゴリズムの実装に関する研究

主指導教員　井口 寧

審査委員主査　井口 寧
審査委員　田中 清史
審査委員　金子 峰雄

北陸先端科学技術大学院大学
情報科学研究科

1510033 Zhao Xuyang

提出年月: 平成 29 年 2 月

概 要

デジタルマルチメディア及びコンピュータネットワーク技術の急速な発展に伴い，デジタルマルチメディア情報（テキスト，オーディオ，画像，ビデオなど）のプロセス及び配信するのは，非常に容易になる．しかし，著作権保護の問題がなっている．デジタルマルチメディアの著作権保護のために，電子透かし技術が開発されている．電子透かし技術とは，デジタルマルチメディアのデータを改変し，著作者，税金など情報をデジタルマルチメディアに埋め込み技術である．

ネットワークの不正配信問題を防ぐ，電子透かしを利用されている．ネットワークの電子透かし監視システムはネットワークの装置 (ルート，スイッチなど) に電子透かしの検出回路を置く，ネットワーク経由で送信されるデジタルマルチメディアを監視する．しかし，電子透かしの検出回路を置くに伴う．消費エネルギーを増える．

本研究では，オーディオ電子透かしのアルゴリズムについて調査し，ＦＰＧＡ (Field Programmable Gate Array) に適したアルゴリズムを選択し，ハードウェアに，低消費エネルギー回路でオーディオ電子透かしの検出回路を実現する．

目 次

図 目 次

表 目 次

第1章　はじめに

1.1　研究背景

近年，マルチメディア及びコンピュータネットワーク技術の急速な発展に伴い，マルチメディア（テキスト、オーディオ、画像、ビデオなど）のプロセス及び配布するのは，非常に容易になり，誰でもネットワークで電子情報を収集することできる．しかし，著作権保護の問題がなっている．電子透かし技術とは，デジタルマルチメディアのデータを改変し，著作者，税金など情報をデジタルマルチメディアに埋め込み技術である．

ネットワークの不正配信問題を防ぐ，ネットワークの装置 (ルート，スイッチなど) に電子透かしの検出回路を置く．しかし，消費エネルギーを増える．

現在，オーディオ電子透かし技術には主に周波数領域に埋込みを行う方法と空間領域に埋め込みを行う方法の二つがある．周波数領域アルゴリズムは、オリジナルオーディオを周波数領域に変換し，その値に著作者，税金情報などの電子透かしを埋め込み，次には埋め込んだオーディオを逆変換し，埋め込んだオーディオを生成する．周波数領域アルゴリズムは良好なロバスト性能を持ち，リサンプリング，再量子化，MP3 圧縮などアタックに耐性を持つ．空間領域アルゴリズムは，オリジナルオーディオに直接で電子透かしデータを埋め込むことで行われる，空間領域アルゴリズムは一般的なアタックに耐性を持ちながら音質の劣化も少ない．

1.2　研究目的

本研究の目的は，オーディオ電子透かしのアルゴリズムについて調査し，ＦＰＧＡに適したアルゴリズムを選択し，ハードウェアに，低消費エネルギー回路でオーディオ電子透かしの検出回路を実現する．

1.3　本文の構成

本稿では，以下の構成なる．第 1 章では研究背景と研究目的を述べる．第 2 章ではオーディオ電子透かしの概要を説明し，ハードウェア化にアルゴリズムを検討し，ハードウェアに適するアルゴリズムを選択する．第 3 章では選択するアルゴリズムをソフトウェアに実装する．アルゴリズムアタック耐性を測定する．第 4 章では選択するアルゴリズムを

ハードウェアに実装し，検出回路を構築する．第5章では，ソフトウェアとハードウェア検出回路の消費電力と検出時間を測定する．消費エネルギーを評価する．第6章では全体のまとめを行う．研究中にの問題を分析し，今後の課題を述べる．

第2章　オーディオ電子透かし

2.1　はじめに

電子透かし技術とは，デジタルマルチメディアのデータを改変し，著作者，税金など情報をデジタルマルチメディアに埋め込み技術である．ネットワークの不正配信問題を防ぐ，ネットワークの装置 (ルート，スイッチなど) に電子透かしの検出回路を置く．しかし，消費エネルギーを増える．本研究の目的は，オーディオ電子透かしのアルゴリズムについて調査し，ＦＰＧＡに適したアルゴリズムを選択し，ハードウェアに，低消費エネルギー回路でオーディオ電子透かしの検出回路を実現する．本章では，オーディオ電子透かし技術を紹介する．代表的なオーディオ電子透かしのアルゴリズムを説明し，ハードウェア化にアルゴリズムを検討し，ハードウェアに適するアルゴリズムを選択する．選択したアルゴリズムの埋め込む過程と検出過程を説明する．選択したアルゴリズム中の音質聴取実験を説明し，音質聴取実験をする．

2.2　オーディオ電子透かしの紹介

オーディオ電子透かし技術とは，オーディオのデータを改変し，著作者，税金など情報をオーディオに埋め込み技術である．現在，オーディオに対する，電子透かしのアルゴリズムには，周波数領域へ埋め込みを行うアルゴリズムと空間領域へ埋め込み行うアルゴリズムに大別される．

IFPI(International Federation of the Phonographic Industry)[1] によって，電子透かしのアルゴリズムは以下を満たす必要がある．

- 埋め込んだの透かしはオーディオの品質に影響を与えることがない．
- アルゴリズムは，透かしに 20bps 以上のデータペイロードを提供する必要がある．
- アルゴリズムは，一般的なオーディオアタックの耐性 (リサンプリング，再量子化，MP3 圧縮など) を持つ．

2.2.1 周波数領域のアルゴリズム

周波数領域のアルゴリズムには，オリジナルオーディオを周波数領域に変換し，周波数係数の値を変化させて透かしを埋め込むを行う．離散ウェーブレット変換に基づくオーディオ透かしアルゴリズム [1] はオリジナルオーディオを離散ウェーブレット変換し，離散ウェーブレット係数に電子透かしを埋め込む．検出するとき，周波数領域の変換し，周波数係数から電子透かしを検出する．

離散フーリエ変換に基づくオーディオ透かしアルゴリズム [2] はオリジナルオーディオを離散フーリエ変換し，離散フーリエ係数に電子透かしを埋め込む．検出するとき，周波数領域の変換し，周波数係数から電子透かしを検出する．

2.2.2 空間領域のアルゴリズム

空間領域のアルゴリズムには，さまざまなアルゴリズムがある．エコー拡散に基づく透かしアルゴリズム [3] は，オリジナルオーディオの遅延 (約 100 ミリ秒以内) に微弱なエコーを挿入し，遅延の数値で電子透かしを表示する．埋め込んだオーディオは，人間の耳が聞こえない．これを継時マスキングという．エコー拡散に基づく透かしアルゴリズムは継時マスキングを利用される．検出するとき，埋め込んだオーディオをケプストラム処理し，電子透かしを検出する．

ビット置換法 [4] は，オーディオサンプリングデータの最下位ビット値を電子透かしデータを置換し，透かしを埋め込む．埋め込むと検出過程は簡単である．

2.2.3 オーディオファイルフォーマット

現在、さまざまなオーディオフォーマット (WAV，MP3，WMA など) があり．一般的なオーディオフォーマット中に、WAV オーディオフォーマットは簡単なコーディング，良い音質の特徴を持つ．本研究では WAV オーディオフォーマットを採用される．

- サンプリング周波数とは，単位時間当たりオーディオデータのサンプル数．オーディオのサンプリング周波数が高い，オーディオの品質より良いである．現在では，サンプリング周波数は，一般的に 22,050Hz、44,100Hz と 48,000Hz の 3 つのレベルに分割される．本研究ではオーディオのサンプリング周波数が 44,100Hz を採用される．

- 量子化とは、アナログオーディオを離散値で近似的に表示する．量子化ビット数が多く、オーディオの品質より良いである．現在では，量子化ビット数は，一般的に 8bit、16bit と 32bit の 3 つのレベルに分割される．本研究ではオーディオの量子化ビット数が 16bit を採用される．

- チャンネルとは，主にモノラル、2 チャンネル、ステレオなどいくつかある．本研究ではモノラルオーディオを採用される．

本研究のオーディオフォーマットは表 2.1 に示す.

	説明
オーディオフォーマット	WAV
サンプリング周波数	44,100Hz
量子化ビット数	16bit
チャンネル	1(モノラル)

表 2.1: オーディオファイルフォーマット

2.2.4 電子透かしデータ

オリジナルオーディオに埋め込んだ情報は，著作者名，課金情報などである．ハードウェア化に，電子透かしデータ $W(k)$ が式 2.1 を二進数で示す．$W(k)$ は電子透かしデータの k 番目要素である.

$$W(k) = \begin{cases} 0 \\ 1 \end{cases} \tag{2.1}$$

2.3 ウェーブレット変換アルゴリズム

離散ウェーブレット変換に基づくオーディオ透かしアルゴリズム [1] の処理過程概要は図 2.1 に示す.

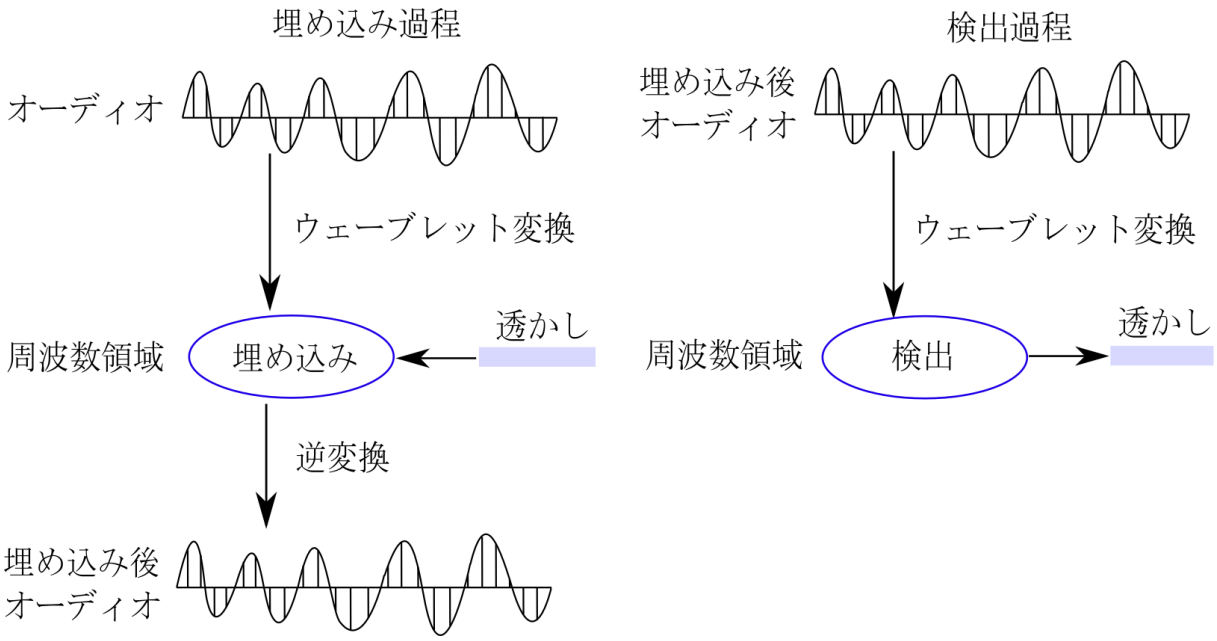

図 2.1: ウェーブレット変換アルゴリズムの埋め込むと検出過程概要

図 2.1 による，電子透かしを埋め込む過程は，オリジナルオーディオを離散ウェーブレット変換し，周波数係数の値を変化させて透かしを埋め込む．次に，逆離散ウェーブレット変換し，埋め込んだオーディオを生成する．図 2.1 による，電子透かしを検出過程は埋め込んだオーディオを離散ウェーブレット変換し，周波数係数から電子透かしを検出する．

2.3.1 ウェーブレット変換アルゴリズムの埋め込む過程

離散ウェーブレットアルゴリズムの埋め込む過程は図 2.2 に示す．

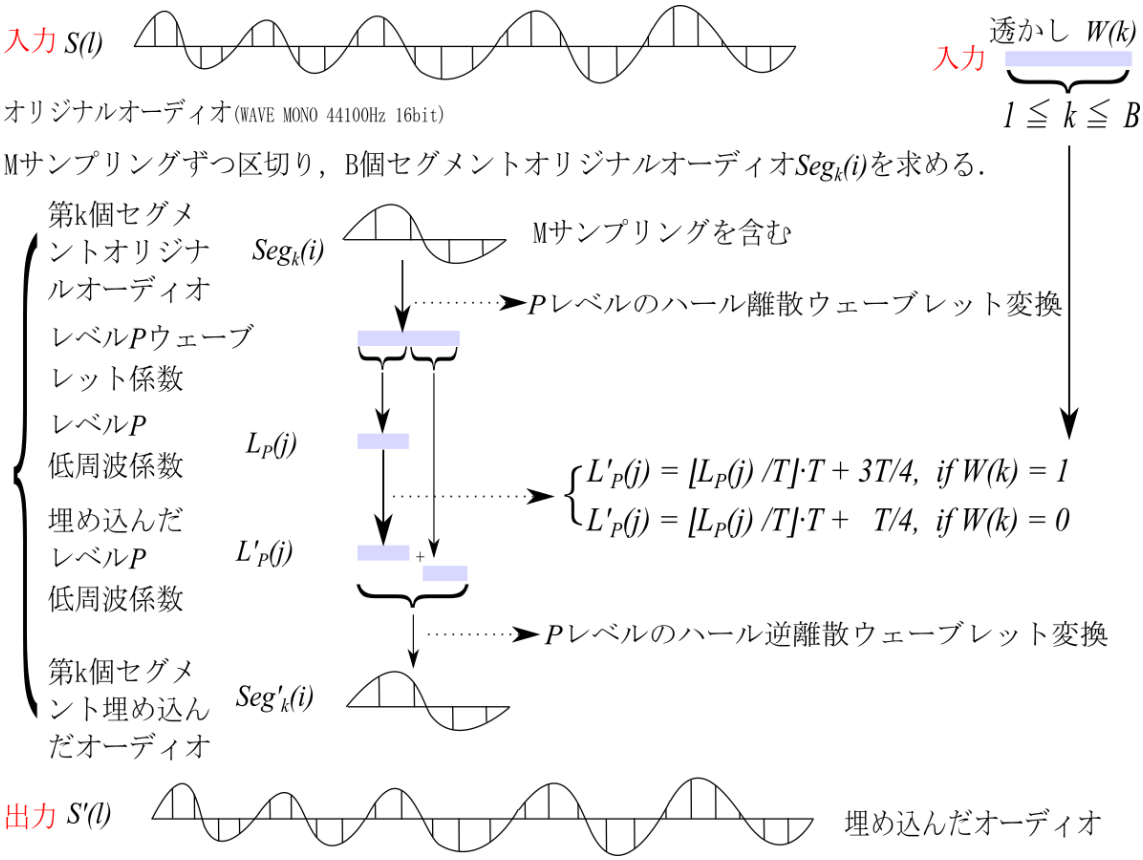

図 2.2: ウェーブレット変換アルゴリズムの埋め込む過程

図 2.2 に示す，埋め込む過程の入力は $S(l)$ と $W(k)$ である．ここでは，$S(l)$ はオリジナルオーディオの l 番目サンプリングであり，$W(k)$ は透かしデータの k 番目要素であり．透かしデータの要素数は B 個である $(1 \leq k \leq B)$．図 2.2 による，オリジナルオーディオ $S(l)$ を M サンプリングずつ区切り，式 2.2 によって，B 個セグメントオリジナルオーディオ $Seg_k(i)$ を求める，$Seg_k(i)$ は第 k 個セグメントオリジナルオーディオの i 番目サンプリングである $(1 \leq k \leq B, 1 \leq i \leq M)$．後ろの部分は端数をなり，端数は電子透かしの埋め込むを行わない．

$$Seg_k(i) = S((k-1) \times M + i) \tag{2.2}$$

図 2.2 による，第 k 個セグメントオリジナルオーディオ $Seg_k(i)$ に透かしを埋め込む過程は，まず，$Seg_k(i)$ を P レベルのハール離散ウェーブレット変換し，毎レベルのウェーブレット係数は低周波係数と高周波係数を構成される．

式 2.3 と式 2.4 によって，レベル n のウェーブレット係数を求める．$L_n(j)$ はレベル n の j 番目低周波係数であり，$H_n(j)$ はレベル n の j 番目高周波係数である $(1 \leq j \leq \frac{M}{2^n}, 1 \leq n \leq P)$.

$$L_n(j) = [L_{n-1}(2j-1) + L_{n-1}(2j)] \times \frac{\sqrt{2}}{2} \tag{2.3}$$

$$H_n(j) = [L_{n-1}(2j-1) - L_{n-1}(2j)] \times \frac{\sqrt{2}}{2} \tag{2.4}$$

式 2.3 と式 2.4 中に，$L_0(j) = Seg_k(j)$ である $(1 \leq j \leq M, 1 \leq k \leq B)$.

次に，レベル $n(n = P)$ の低周波係数 $L_n(j)$ に透かしを埋め込む，式 2.5 によって，レベル n の埋め込んだ低周波係数 $L_n^{'}(j)$ を求める．$L_n^{'}(j)$ はレベル n の j 番目埋め込んだ低周波係数である $(1 \leq j \leq \frac{M}{2^n}, 1 \leq k \leq B)$.

ここでは，T は埋め込む強度であり，音質聴取試験によって，埋め込む強度 T の数値を決める．音質聴取試験とは，実祭に人間に聴いてもらって音質が劣化していないか確認してもらう試験である．2.3.3 に示す，実験による．

$$L_n^{'}(j) = \begin{cases} \lfloor L_n(j)/T \rfloor \times T + 3T/4, & \text{if } W(k) = 1 \\ \lfloor L_n(j)/T \rfloor \times T + T/4, & \text{if } W(k) = 0 \end{cases} \tag{2.5}$$

最後に，P レベルのハール逆離散ウェーブレット変換する．式 2.6 と式 2.7 によって，レベル $n(1 \leq n \leq P)$ の埋め込んだ低周波数係数と高周波数係数から前レベルの埋め込んだ低周波数係数を求める $(1 \leq j \leq \frac{M}{2^n}, 1 \leq n \leq P)$.

$$L_{n-1}^{'}(2j-1) = [L_n^{'}(j) + H_n(j)] \times \frac{\sqrt{2}}{2} \tag{2.6}$$

$$L_{n-1}^{'}(2j) = [L_n^{'}(j) - H_n(j)] \times \frac{\sqrt{2}}{2} \tag{2.7}$$

$L_n^{'}(j)$ はレベル n の j 番目埋め込んだ低周波係数であり，式 2.6 と式 2.7 中に，$Seg_k^{'}(j) = L_0^{'}(j)$ である $(1 \leq j \leq M, 1 \leq k \leq B)$．$Seg_k^{'}(i)$ は第 k 個セグメント埋め込んだオーディオの i 番目サンプリングである

式 2.8 によって．電子透かしを埋め込んだオーディオ $S^{'}(l)$ を求める．以上を B 個セグメントオリジナルオーディオに対して繰り返し，電子透かしを埋め込んだオーディオ $S^{'}(l)$ を生成する $(1 \leq i \leq M, 1 \leq k \leq B)$.

$$S^{'}((k-1) \times M + i) = Seg_k^{'}(i) \tag{2.8}$$

電子透かしを埋め込んだオーディオ $S^{'}(l)$ は埋め込んだオーディオの l 番目サンプリングである．

2.3.2　ウェーブレット変換アルゴリズムの検出過程

アルゴリズムの検出過程は図 2.3 に示す.

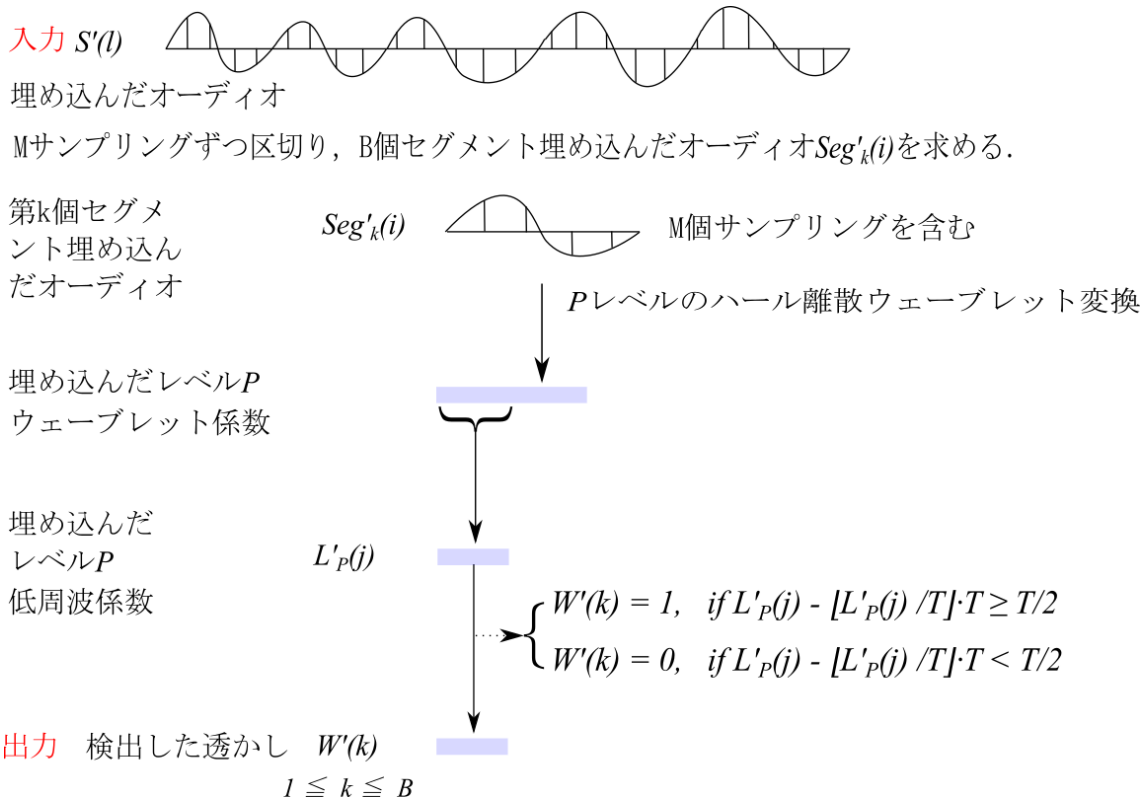

図 2.3: ウェーブレット変換アルゴリズムの検出過程

図 2.3 に示す，検出過程の入力は透かしを埋め込んだオーディオ $S^{'}(l)$ であり．$S^{'}(l)$ は電子透かしを埋め込んだオーディオの l 番目サンプリングである．図 2.3 による，電子透かしを埋め込んだオーディオ $S^{'}(l)$ を M サンプリングずつ区切り，式 2.9 によって，B 個セグメント埋め込んだオーディオ $Seg_k^{'}(i)$ を求める，$Seg_k^{'}(i)$ は第 k 個セグメント埋め込んだオーディオの i 番目のサンプリングである $(1 \leq k \leq B, 1 \leq i \leq M)$．後ろの部分は端数をなり，端数は電子透かしの埋め込むを行わない．

$$Seg_k^{'}(i) = S^{'}((k-1) \times M + i) \tag{2.9}$$

図 2.3 による，第 k 個セグメント埋め込んだオーディオ $Seg_k^{'}(i)$ に透かしを検出過程は，まず，$Seg_k^{'}(i)$ を P レベルのハール離散ウェーブレット変換し，毎レベルのウェーブレット係数は低周波係数と高周波係数を構成される．

式 2.10 と式 2.11 によって，レベル n のウェーブレット係数を求める．$L_n^{'}(j)$ はレベル n の j 番目埋め込んだ低周波係数であり，$H_n(j)$ はレベル n の j 番目高周波係数である $(1 \leq j \leq \frac{M}{2^n}, 1 \leq n \leq P)$.

$$L_n^{'}(j) = [L_{n-1}^{'}(2j-1) + L_{n-1}^{'}(2j)] \times \frac{\sqrt{2}}{2} \tag{2.10}$$

$$H_n(j) = [L_{n-1}^{'}(2j-1) - L_{n-1}^{'}(2j)] \times \frac{\sqrt{2}}{2} \tag{2.11}$$

式 2.10 と式 2.11 中に，$L_0^{'}(j) = Seg_k^{'}(j)$ である $(1 \leq j \leq M, 1 \leq k \leq B)$.

レベル $n(n = P)$ の埋め込んだ低周波係数 $L_n^{'}(j)$ に電子透かしを検出する．式 2.12 によって，1 ビット検出した電子透かしデータ $W^{'}(k)$ を求める．$W^{'}(k)$ は検出した透かしデータの k 番目要素である $((1 \leq j \leq \frac{M}{2^n}), 1 \leq k \leq B)$.

ここでは，T は埋め込む強度であり，音質聴取試験によって，埋め込む強度 T の数値を決める．音質聴取試験とは，実祭に人間に聴いてもらって音質が劣化していないか確認してもらう試験である．2.3.3 に示す，実験による．

$$W^{'}(k) = \begin{cases} 1, & \text{if } (L_n^{'}(j) - \lfloor L_n^{'}(j)/T \rfloor \times T) \geq T/2 \\ 0, & \text{if } (L_n^{'}(j) - \lfloor L_n^{'}(j)/T \rfloor \times T) < T/2 \end{cases} \tag{2.12}$$

第 k 個セグメント埋め込んだオーディオから 1 ビット検出した透かしデータ $W^{'}(k)$ を取り出す．以上を B 個セグメントに対して繰り返し，すべて検出した透かしデータ $W^{'}(k)$ を取り出す $(1 \leq k \leq B)$.

本研究では，M は 256，P は 8，B は 25 とする．

2.3.3 ウェーブレット変換アルゴリズムの音質聴取試験

電子透かしとは，オーディオを知覚できない程度に改変することで著作者の情報をデジタルコンテンツに埋め込み技術である．埋め込んだオーディオの音質は埋め込む強度 T の数値によって決める．

音質聴取試験とは，実祭に人間に聴いてもらって音質が劣化していないか確認してもらう試験である．音質聴取試験によって，埋め込む強度 T の数値を決める．音質聴取試験の方法は，まず，被験者にオリジナルオーディオを聞かせる．次に，1 と 2 オーディオを聞く．1 と 2 オーディオ中に，ひとつがオリジナルオーディオであり，ひとつが電子透かしを埋め込んだオーディオである．最後，1 あるいは 2 がオリジナルオーディオを答えさせて，正答率が 45%から 55%までになると，電子透かしを埋め込んだオーディオの音質が良くになる．大きな埋め込む強度 T の数値より良いアタック耐性がある [1]．試験中に被験者が三名であり，男性 2 名 (28 歳，26 歳) と女性 1 名 (24 歳)．毎被験者に対し，10 回試験をする．試験結果が図 2.4 を示す．

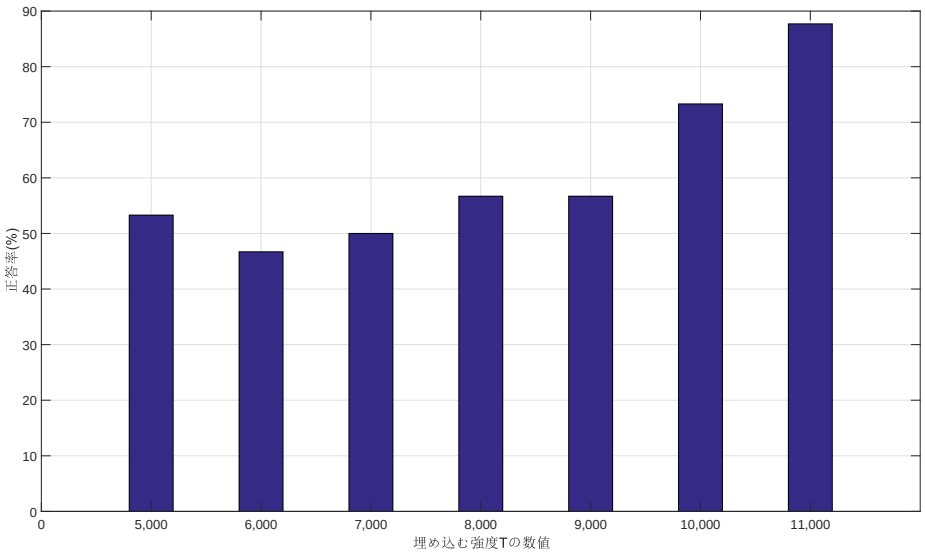

図 2.4: ウェーブレット変換アルゴリズムの音質聴取試験結果

図 2.4 による，参考文献 [1] のよって記述されに従え，正答率が 45%から 55%まで中にの最大数値 9,000 に埋め込む強度 T を決める．

2.4 エコー拡散アルゴリズム

エコー拡散に基づく透かしアルゴリズム [3] はオリジナルオーディオの遅延△ (約 100 ミリ秒以内) 後に微弱なエコーを挿入し，人間の耳が聞こえなくなる．これを継時マスキングという．エコー拡散アルゴリズムは継時マスキングを利用される．

アルゴリズムの処理過程概要は図 2.5 に示す．

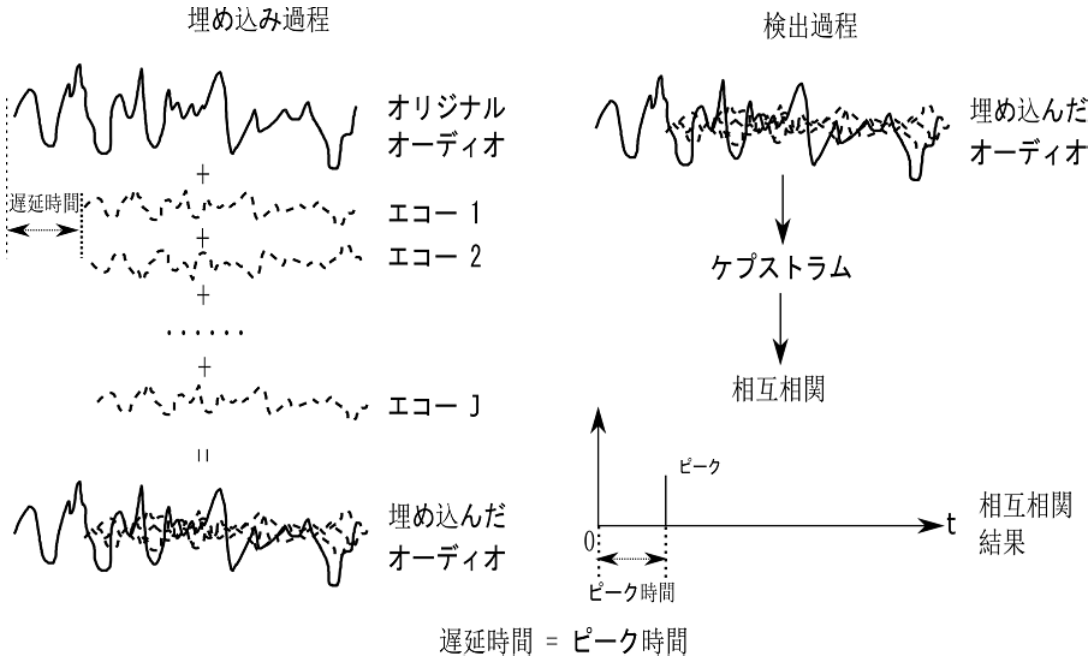

図 2.5: エコー拡散アルゴリズムの埋め込むと検出過程概要

図 2.5 による，埋め込み過程，オリジナルオーディオを H サンプルずつセグメントに区切り，セグメント毎に 1 ビット透かしデータを埋め込む．透かしの数値 (0 または 1) によって，違う遅延時間でセグメントオーディオにエコーを付加する．付加するエコーを単発ではなく，-1 及び 1 の値を持つ M 系列によって時間領域で拡散して生成する．図 2.5 によって，検出過程，セグメント埋め込んだオーディオのケプストラム領域で，埋め込み時に用いた M 系列により逆拡散 (相互相関処理) を行い，ピーク時間を検出する．ピーク時間と遅延時間は同じになる．透かしを検出できる．

2.4.1 エコー拡散アルゴリズムの埋め込む過程

アルゴリズムの埋め込む過程は図 2.6 に示す．

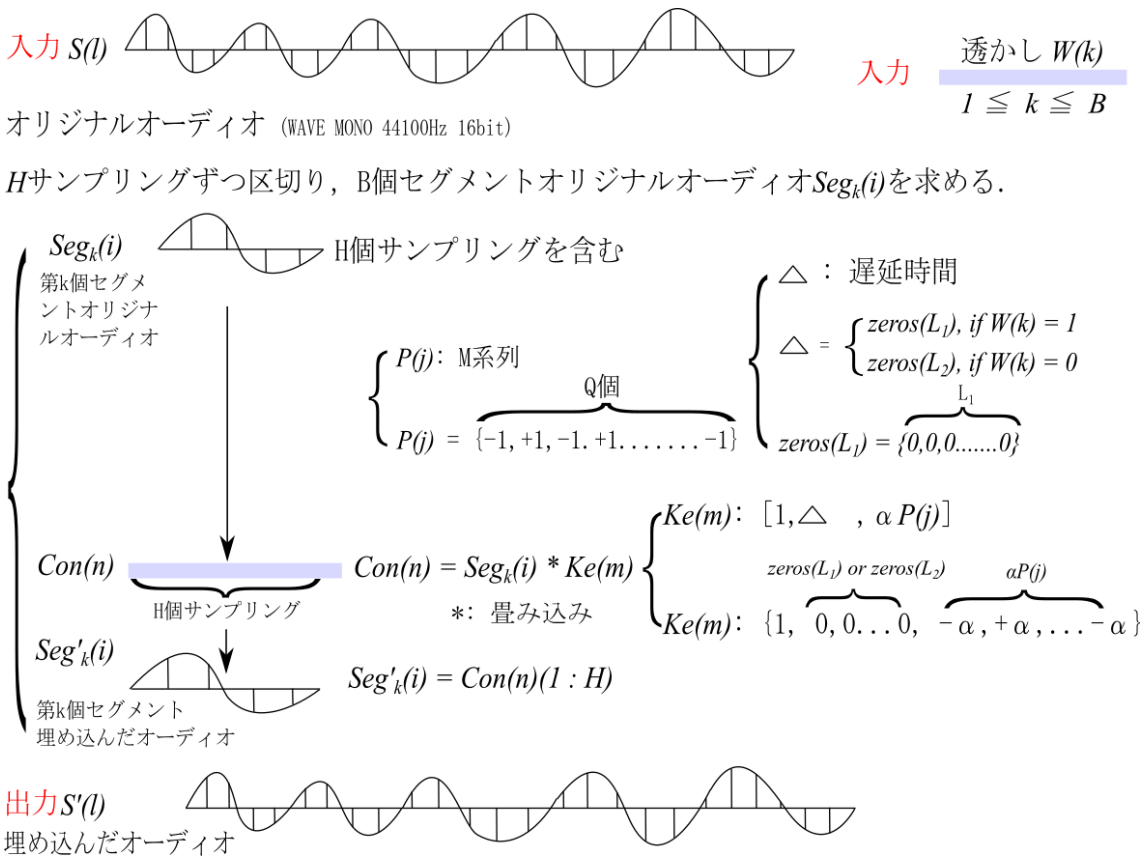

図 2.6: エコー拡散アルゴリズムの埋め込む過程

図 2.6 に示す．埋め込むの入力は $S(l)$ と $W(k)$ である．ここでは，$S(l)$ はオリジナルオーディオの l 番目のサンプリングであり．$W(k)$ は透かしデータの k 番目要素であり，透かしデータの要素は B 個である $(1 \leq k \leq B)$．図 2.6 による，オリジナルオーディオ $S(l)$ を H サンプリングずつ区切り，式 2.13 によって，B 個セグメントオリジナルオーディオ $Seg_k(i)$ を求める，$Seg_k(i)$ は第 k 個セグメントオリジナルオーディオの i 番目のサンプリングである $(1 \leq k \leq B, 1 \leq i \leq H)$．後ろの部分は端数をなり，端数は透かしの埋め込むを行わない．

$$Seg_k(i) = S((k-1) \times H + i) \tag{2.13}$$

図 2.6 によって，第 k 個セグメントオリジナルオーディオ $Seg_k(i)$ に透かしを埋め込む過程は，まず，式 2.14 で透かしデータ $W(k)$ によって，遅延 △ がゼロ系列 $zeros(L_1)$ 又は $zeros(L_2)$ を作る．ここでは $zeros$ は連続した 0 の並び系列，図 2.7 に示す．L_1，L_2 は遅延 △ の数値である．

$$\triangle = \begin{cases} zeros(L_1), & \text{if } W(k) = 1 \\ zeros(L_2), & \text{if } W(k) = 0 \end{cases} \tag{2.14}$$

$$zeros(L_1) = \overbrace{\{0,0,0.....0\}}^{L_1個}$$

$$zeros(L_2) = \overbrace{\{0,0,0.....0\}}^{L_2個}$$

図 2.7: ゼロ系列

式 2.15 によるエコーカーネル $Ke(m)$ を作る．$Ke(m)$ はエコーカーネルの m 番目要素である．ここでは，1 は数字であり，$P(j)$ は M 系列を構成されり (M 系列中の 0 は-1 を変換する)，1 又は-1 を Q 個並べたもの，$P(j)$ は M 系列の j 番目要素であり $(1 \leq j \leq Q)$，α は拡散エコーの振幅であり，音質聴取試験によって，拡散エコーの振幅 α の数値を決める．音質聴取試験とは，実祭に人間に聴いてもらって音質が劣化していないか確認してもらう試験である．2.4.3 に示す，実験による．

$$Ke(m) = Strcat\{1, \triangle, \alpha \times P(j)\} \tag{2.15}$$

図 2.6 による，第 k 個セグメントオリジナルオーディオ $Seg_k(i)$ と $Ke(m)$ を用いて式 2.16 によって，透かしを埋め込む．$*$ は畳み込みである．$Con(n)$ は畳み込み結果の n 番目要素である．

$$Con(n) = Seg_k(i) * Ke(m) \tag{2.16}$$

$Con(n)(1:H)$ は $Con(n)$ の前 H 個サンプリングを格納されるものである，セグメント埋め込んだオーディオ $Seg_k^{'}(i)$ は式 2.17 に求める．$Seg_k^{'}(i)$ はセグメント埋め込んだオーディオの i 番目サンプリングである．第 k 個セグメントオリジナルオーディオに $W(k)$ を埋め込む．

$$Seg_k^{'}(i) = Con(n)(1:H) \tag{2.17}$$

透かしを埋め込んだオーディオ $S^{'}(l)$ は式 2.18 に求める．$S^{'}(l)$ は l 番目埋め込んだオーディオのサンプリングである．以上を B 個セグメントオリジナルオーディオに対して繰り返し，透かしを埋め込んだオーディオ $S^{'}(l)$ とする．

$$S^{'}((k-1) \times H + i) = Seg_k^{'}(i) \tag{2.18}$$

2.4.2 エコー拡散アルゴリズムの検出過程

アルゴリズムの検出過程は図 2.8 に示す．

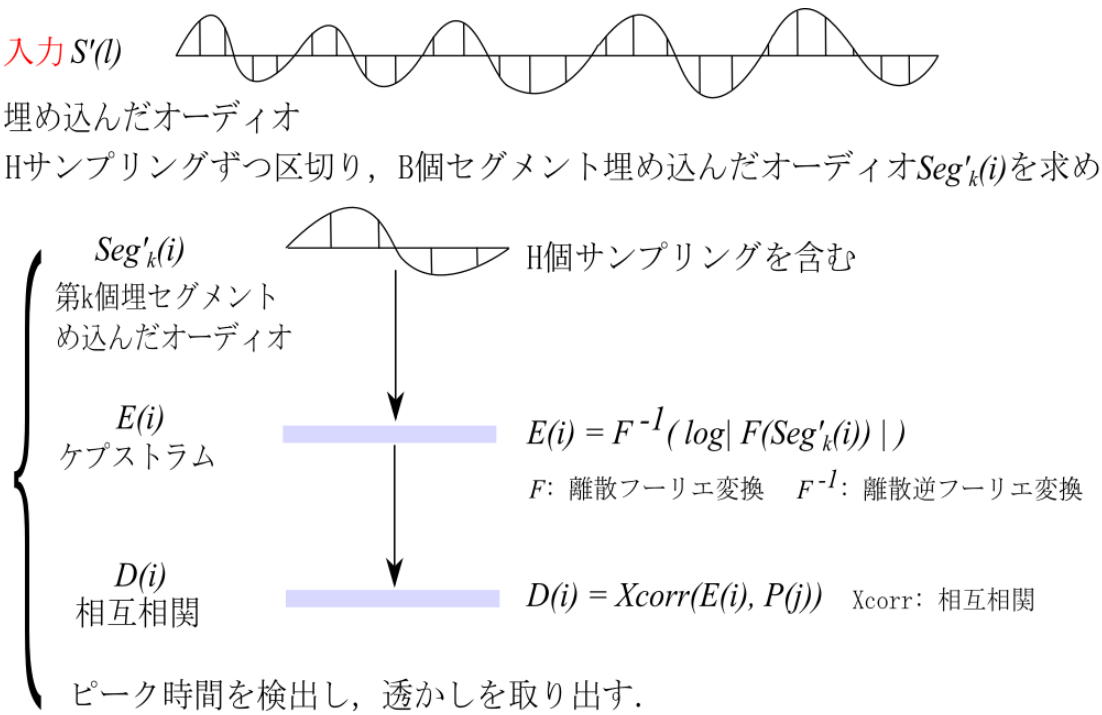

図 2.8: エコー拡散アルゴリズムの検出過程

図 2.8 に示す，検出過程の入力は透かしを埋め込んだオーディオ $S^{'}(l)$ であり．$S^{'}(l)$ は埋め込んだオーディオの l 番目サンプリングである．図 2.8 によって，透かしを埋め込んだオーディオ $S^{'}(l)$ を H サンプルずつ区切り，式 2.19 によって，B 個セグメント埋め込んだオーディオ $Seg_k^{'}(i)$ を求める，$Seg_k^{'}(i)$ は第 k 個セグメント埋め込んだオーディオの i 番目のサンプリングである $(1 \leq k \leq B, 1 \leq i \leq H)$.

$$Seg_k^{'}(i) = S^{'}((k-1) \times H + i) \tag{2.19}$$

$Seg_k^{'}(i)$ のケプストラム結果 $E(i)$ は式 2.20 に求める．F は離散フーリエ変換，F^{-1} は逆離散フーリエ変換．$E(i)$ はケプストラム結果の i 番目要素である．

$$E(i) = F^{-1}(log|F(Seg_k^{'}(i))|) \tag{2.20}$$

ケプストラム演算結果 $E(i)$ と $P(j)$ を相互相関処理し，検出結果 $D(i)$ は式 2.21 に求める．$Xcorr$ は相互相関であり．$D(i)$ は相互相関結果の i 番目要素である．

$$D(i) = Xcorr(E(i), P(j)) \tag{2.21}$$

検出信号 $D(i)$ は，埋め込んだエコーカーネルに対応する遅延△に明らかなピ―クがあり，遅延△の値によって，第 k 個セグメント埋め込んだオーディオ方検出した透かし $W^{'}(k)$

を取り出す．B 個セグメントに対して繰り返し，透かしを検出する．

参考文献 [5]pp128 による．セグメントの長さは 4,096 になるときに，性能が良くなることが分かる．参考文献 [5]pp128 による．$P(j)$ の長さは 1,023 になるときに，性能が良くなることが分かる．本研究では，参考文献 [3] による．H は 4,096，J は 1,023，B は 25，L_1 は 79，L_2 は 39 とする．本研究で使った M 系列の帰還多項式は $x^{10} + x^7 + 1$ である．

2.4.3　エコー拡散アルゴリズムの音質聴取試験

電子透かしとは，オーディオを知覚できない程度に改変することで著作者の情報をデジタルコンテンツに埋め込み技術である．埋め込んだオーディオの音質は拡散エコーの振幅 α の数値によって決める．

音質聴取試験とは，実祭に人間に聴いてもらって音質が劣化していないか確認してもらう試験である．音質聴取試験によって，拡散エコーの振幅 α の数値を決める．音質聴取試験の方法は，まず，被験者にオリジナルオーディオを聞かせる．次に，1 と 2 オーディオを聞き，ひとつがオリジナルオーディオ，ひとつが電子透かしを埋め込んだオーディオである．最後，1 あるいは 2 がオリジナルオーディオを答えさせて．正答率が 45%から 55%までになると，電子透かしを埋め込んだオーディオの音質が良くになる．大きな拡散エコーの振幅 α の数値より良いアタック耐性がある [3]．試験中に被験者が三名であり，男性 2 名 (28 歳，26 歳) と女性 1 名 (24 歳)．毎被験者に対し，10 回試験をする．試験結果が図 2.9 を示す．

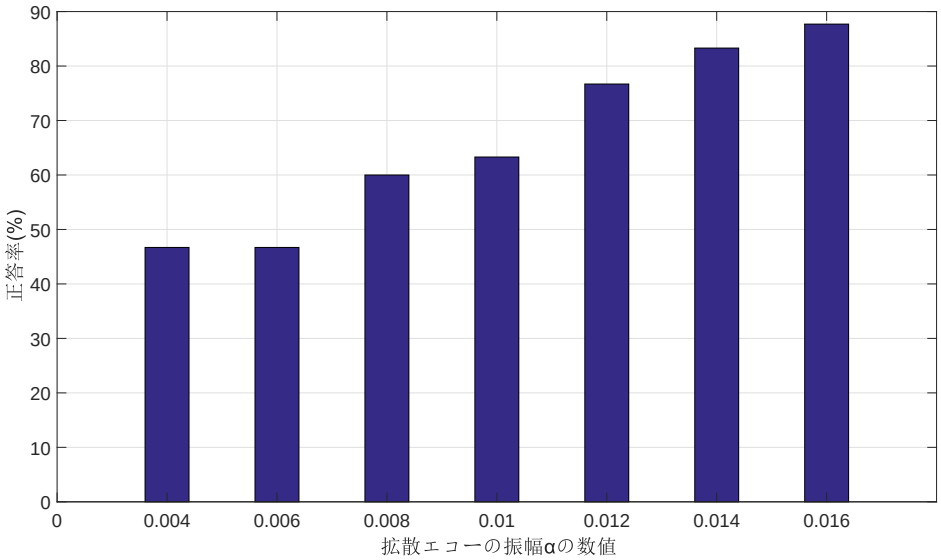

図 2.9: エコー拡散アルゴリズムの音質聴取試験結果

図 2.9 による，参考文献 [3] のよって記述されに従え，正答率が 45%から 55%まで中にの最大数値 0.006 に拡散エコーの振幅 α を決める．

2.5　ハードウェア化にアルゴリズムの検討

本研究の目的は，ＦＰＧＡに適したアルゴリズムをハードウェアに低消費エネルギー回路でオーディオ電子透かしの検出回路を実現する．すべてのオーディオ電子透かしアルゴリズムがハードウェア化に適しているわけではない．ハードウェアの特性を考慮しアルゴリズムを検討する必要がある．

ハードウェアに対する，加算と乗算の比較実験をする．データ D_1(32 ビット，二進数) とデータ D_2(32 ビット，二進数) の加算回路と乗算回路を FPGA に構築し，実装環境は ISE14.7 であり，時間のシミュレーション環境は ISIM であり，シミュレーションのクロックは 10 ナノ秒であり，消費電力のシミュレーション環境は XPower Analyzer である．回路のリソースと演算時間と消費電力は表 2.2 に示す．

	加算	乗算
リソース (Slice LUTs)	32	1181
演算時間 (クロック)	1	2.2
消費電力 (w)	3.422	3.422

表 2.2: 加算と乗算の計算統計

加算と比較すると、乗算は、多くのエネルギーと多くのリソースをかかる．ハードウェアリソースの制限がある．本研究では，Xilinx の ML605 FPGA ボードを利用された．Logic Cells 241,152，Slices 37,680 の制限がある．

アルゴリズム [1][2][3][4] を比較する．

- 離散ウェーブレットアルゴリズムの検出過程では，一回離散ウェーブレット変換し，透かしを検出する．
- 離散フーリエアルゴリズムの検出過程では，一回離散フーリエ変換し，透かしを検出する．
- エコー拡散アルゴリズムの検出過程では，一回ケプストラム演算と一回相互相関演算し，透かしを検出する．ケプストラム処理中に，離散フーリエ変換，逆離散フーリエ変換を一回ずつする．
- ビット置換アルゴリズムの検出過程では，サンプリングデータの最下位ビット値から透かしを検出する．

電子透かしを検出するとき，セグメント埋め込んだオーディオで処理する．毎セグメントオーディオに処理するときの加算と乗算の回数を表 2.3 に示す．

式 2.3 と式 2.4 によって，離散ウェーブレットアルゴリズムセグメントの長さは 256 を決める．参考文献 [2] によって，離散フーリエアルゴリズムセグメントの長さは 4,096 を

決める．参考文献 [5] によって，エコー拡散アルゴリズムセグメントの長さは 4,096 になるときに，性能が良くなることが分かる．参考文献 [4] によって，ビット置換アルゴリズムセグメントの長さは 44,100 を決める．

式 4.1 によって，離散ウェーブレットアルゴリズムの検出過程は 256 回加算をする．長さ R の信号に対し，離散フーリエ変換と逆離散フーリエ変換中に，加算は $R \times (R-1)$ 回であり，乗算は R^2 回である．長さ U の信号に対し，相互相関処理中に，加算は U^2 回であり，乗算は U^2 回である．フーリエ変換とエコー拡散とビット置換アルゴリズムはハードウェアに実装しない，理論的数値である．

	加算	乗算
ウェーブレット変換	256	0
フーリエ変換	$4,096 \times (4,096-1)$	$4,096^2$
エコー拡散	$2 \times 4,096 \times (4,096-1) + 4,096^2$	$3 \times 4,096^2$
ビット置換	0	0

表 2.3: アルゴリズムの計算統計

加算と乗算の消費電力が同じであり，演算回数が少ないアルゴリズムの消費エネルギーも少ないである．低消費エネルギーために，四つアルゴリズムの優先順位はビット置換，ウェーブレット変換，フーリエ変換，エコー拡散である．

埋め込んだオーディオを伝送するとき，悪意のある消費者から，埋め込んだ電子透かしを破壊することがある．悪意のある消費者は埋め込んだオーディオにアタックするとき，アタックされたオーディオから，電子透かし検出を測定する．

アルゴリズム [1][2][3][4] のアタック耐性は表 2.4 に示す．アルゴリズムの耐性を測定ために，リサンプリング，再量子化とＭＰ３圧縮の耐性を測定する．電子透かしを埋め込んだオーディオにアタックされて，電子透かし検出を行う．式 2.22 によって，誤り率 (BER) を求める．誤り率によって，アタック耐性を評価する．

$$BER = \frac{\text{誤り検出ビット数}}{\text{電子透かしデータビット数}} \tag{2.22}$$

	リサンプリング (22,050Hz)	再量子化 (8bit)	MP3 圧縮 (128kbps)
ウェーブレット変換	0.0%	0.0%	0.0%
フーリエ変換	0.0%	0.0%	0.0%
エコー拡散	9.7%	15.2%	4.7%
ビット置換	0.0%	50.8%	49.6%

表 2.4: アルゴリズムのアタック耐性

検出透かしによって，著作権を判定する．検出透かしのエラーがあれば，著作権の判定は問題が発生する．アルゴリズム耐性の閾値は設定できない．四つアルゴリズムの優先順位はウェーブレット変換，フーリエ変換，エコー拡散，ビット置換である．

電子透かしの検出にはオリジナルオーディオ必要がある，管理者以外は検出できない．秘匿性を持つという．電子透かしの検出にはオリジナルオーディオ必要がない，誰でも検出できる．秘匿性を持たないという．アルゴリズム [1][2][3][4] の検出過程中に，オリジナルオーディオ必要がない，秘匿性を持たない．

本研究では，離散ウェーブレット変換に基づくオーディオ透かしアルゴリズム [1] を選択する．

2.6 まとめ

本章では，オーディオ電子透かし技術を紹介した．代表的なオーディオ電子透かしアルゴリズムを説明した．ハードウェア化にアルゴリズムを検討した，ハードウェア化を考慮して，ウェーブレットアルゴリズム [1] を選択した．ウェーブレットアルゴリズムの埋め込む過程と検出過程を説明した．音質聴取実験をした．音質聴取実験の結果によって，埋め込む強度 T の数値を 9,000 に決めた．

第3章　アルゴリズムのソフトウェア実装

3.1　はじめに

本章では，離散ウェーブレットアルゴリズムの消費エネルギーを測定ために，アルゴリズムをソフトウェア (Matlab 2016b) に実装する．アルゴリズムのアタック耐性を確認ために，アルゴリズムのアタックを測定する．アタックはリサンプリング，再量子化，ＭＰ３圧縮である．

3.2　ソフトウェア実装

アルゴリズムをソフトウェア (Matlab 2016b) に実装し．ソフトウェアの開発環境は表 3.1 に示す．

	説明
CPU	Intel Core2 P7570 2.26GHz
メモリ	DDR3 4GB
ハードディスク	ST9320423AS ATA(320GB)
OS	Windows 7
ツール	Matlab 2016b

表 3.1: ソフトウェア実装環境

3.2.1　埋め込むフローチャート

離散ウェーブレットアルゴリズムの埋め込むフローチャートは図 3.1 に示す，

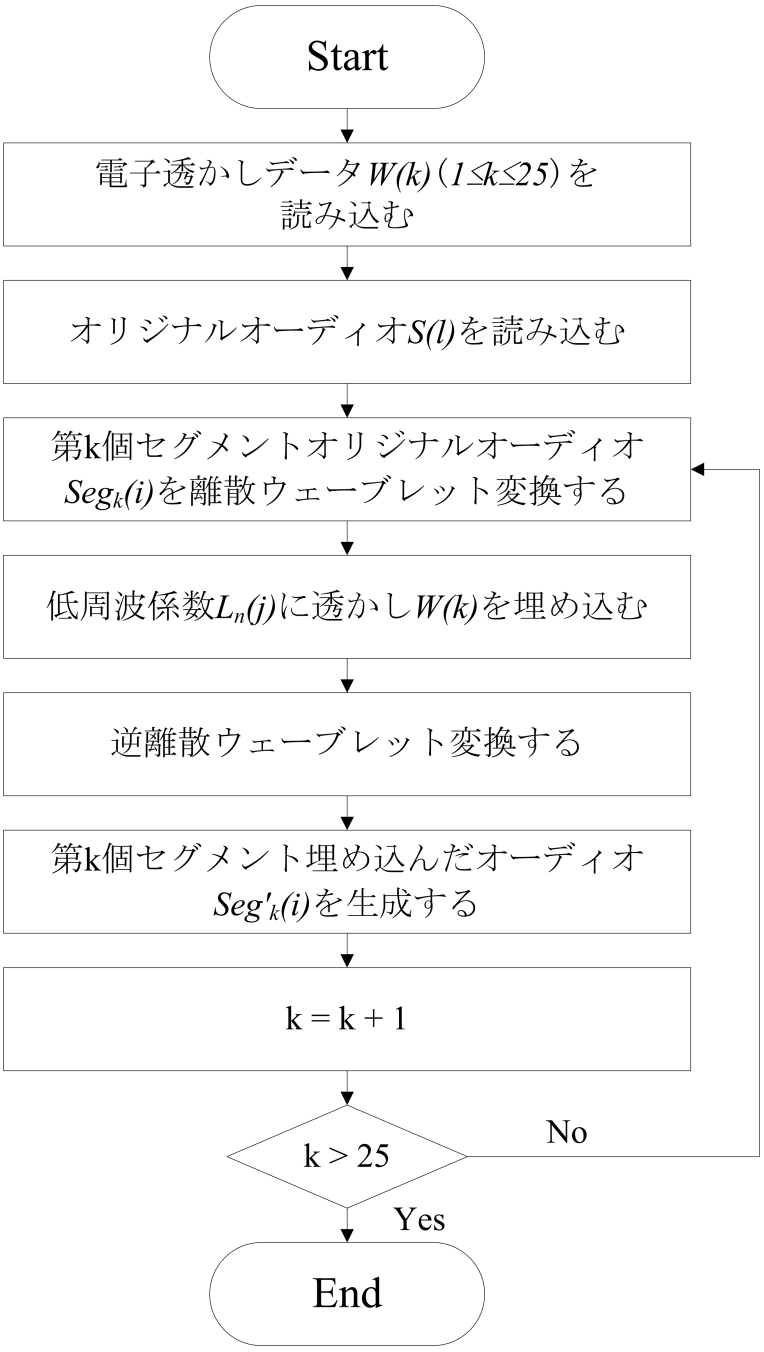

図 3.1: アルゴリズムの埋め込むフローチャート

図 3.1 による，第 k 個セグメントオリジナルオーディオ $Seg_k(i)$ を 8 レベルのハール離散ウェーブレット変換し，レベル 8 の低周波係数 $L_n(j)$ に 1 ビットの電子透かしデータ $W(k)$ を埋め込む．各セグメントは 256 サンプリングであり，ループを 25 回行う．電子透かしデータ $W(k)$(B ビット，乱数で構成される) を利用された．B は 25 ビット，M は 256 サンプルである．埋め込む強度 T は三名の音質聴取実験によって 9,000 を決める．音質聴取

試験とは，実祭に人間に聴いてもらって音質が劣化していないか確認してもらう試験である．

オリジナルオーディオ $S(n)$ の波形は図 3.2 に示す．透かしを埋め込んだオーディオ $S'(n)$ の波形は図 3.3 に示す．

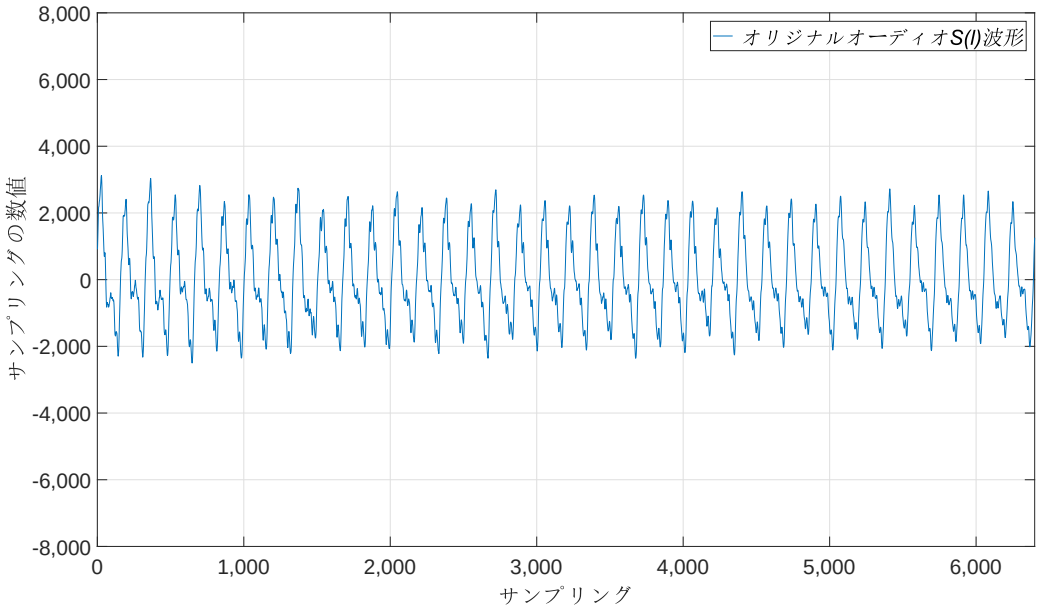

図 3.2: オリジナルオーディオ $S(n)$ の波形

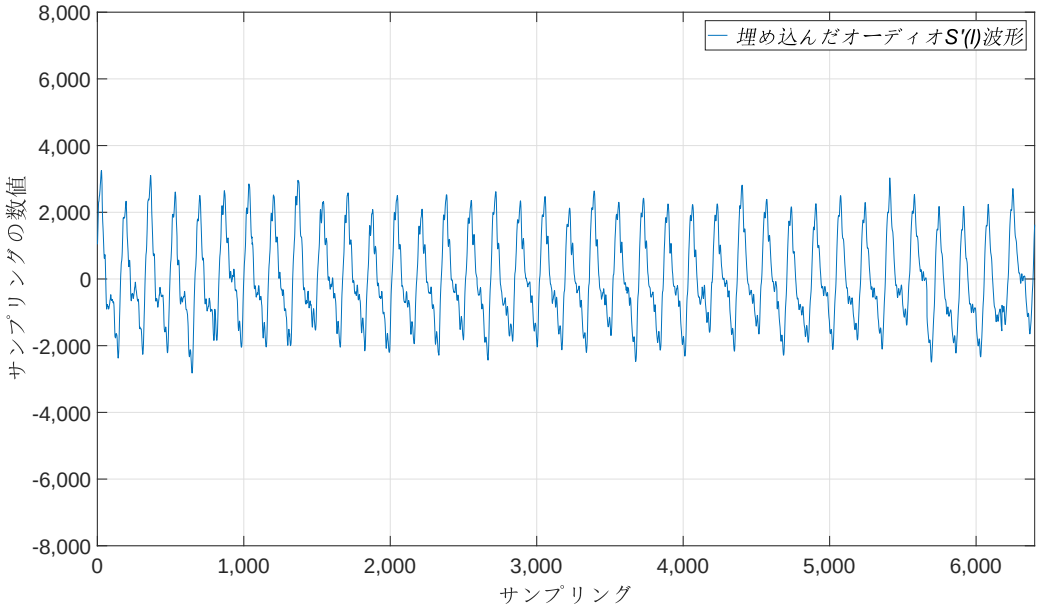

図 3.3: 埋め込んだオーディオ $S^{'}(n)$ の波形

図 3.2 と図 3.3 に示す，式 3.1 によって，電子透かしを埋め込んだオーディオ $S^{'}(l)$ とオリジナルオーディオ $S(l)$ の平均絶対偏差 α は 160.45 を求める．ここで，毎セグメントオリジナルオーディオ $Seg_k(i)(1 \leq i \leq 256)$ に 1 ビット透かしを埋め込み，25 ビット透かしを埋め込んだ $(1 \leq l \leq 6,400, 1 \leq k \leq 25, 6,400 = 256 \times 25)$.

$$\alpha = \frac{1}{6,400} \times \sum_{l=1}^{6,400} |(S^{'}(l) - S(l)| \tag{3.1}$$

結果の評価ために，オリジナルオーディオ $S(l)$ を MP3 圧縮して，圧縮されたオーディオ $S^1(l)$ を求める．$S^1(l)$ を展開して，展開されたオーディオ $S^2(l)$ を求める．

式 3.2 によって，展開されたオーディオ $S^2(l)$ とオリジナルオーディオ $S(l)$ の平均絶対偏差 α はを求める．MP3 圧縮のビットレートは 128kbps とき，$\alpha = 181.26$ であり，MP3 圧縮のビットレートは 256kbps とき，$\alpha = 86.615$ である，透かしを埋め込んだの影響はビットレート 128kbps の MP3 圧縮にような．

$$\alpha = \frac{1}{6,400} \times \sum_{l=1}^{6,400} |(S^{2}(l) - S(l)| \tag{3.2}$$

3.2.2 検出過程フローチャート

離散ウェーブレットアルゴリズムの検出フローチャートは図 3.4 に示す，

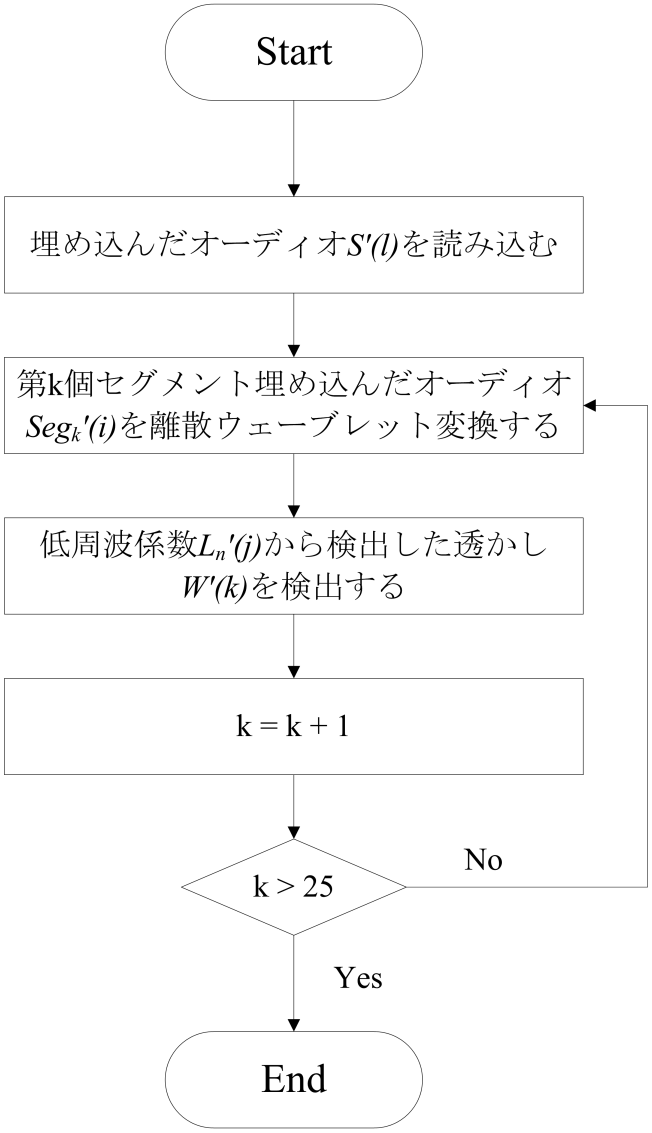

図 3.4: アルゴリズムの検出フローチャート

図 3.4 による，第 k 個セグメント埋め込んだオーディオ $Seg'_k(i)$ をレベル 8 のハール離散ウェーブレット変換し，ウェーブレット係数 $L'_n(j)$ から 1 ビットの検出した透かし $W'(k)$ を取り出す．各セグメントは 256 サンプリングであり，ループも 25 回行う．

3.3 アタック耐性の評価

アタックに対する，アルゴリズムの耐性を測定ために，ガウス，リサンプリング，再量子化とＭＰ３圧縮の耐性を測定する．電子透かしを埋め込んだオーディオ $S'(n)$ にアタックされて，電子透かし検出を行う．

式 3.3 によって，誤り率 (BER) を求める．検出した電子透かし $W'(k)$ の誤り率によって，アタック耐性を評価する．

$$BER = \frac{\text{誤り検出ビット数}}{\text{電子透かしデータビット数}} \tag{3.3}$$

3.3.1 リサンプリング

処理のフローチャートは図 3.5 に示す．

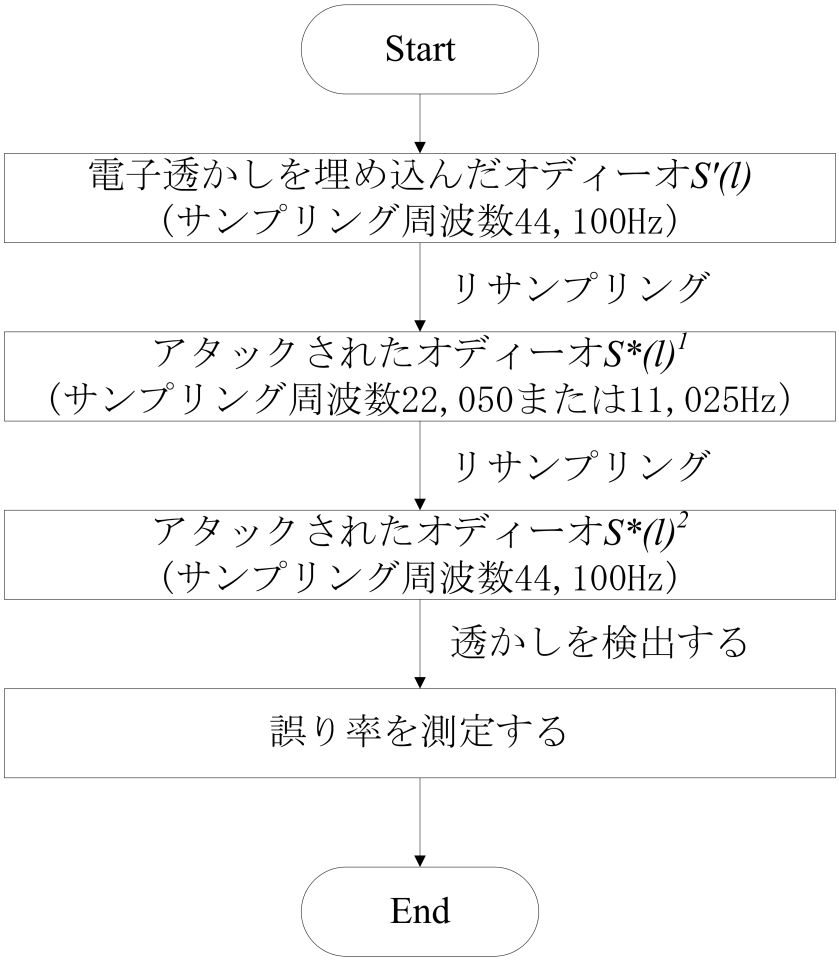

図 3.5: リサンプリングアタックのフローチャート

電子透かしを埋め込んだオーディオ $S'(l)$ に対する，リサンプリングアタックする．ここで，リサンプリングことがソフトウェア (Adobe Audition CS6) で実現する．$S^*(l)^1$ は第 1 回リサンプリングアタックされたオーディオであり，$S^*(l)^2$ は第 2 回リサンプリングアタックされたオーディオである．アタックされたオーディオ $S^*(l)^2$ から電子透かし検出を行う．検出した電子透かしの誤り率を測定する．

図 3.5 による，リサンプリングアタックの誤り率の結果は表 3.2 に示す．

サンプリング周波数	BER(%) (実装結果)	BER(%) (参考文献 [1])
22,050	0	0
11,025	0	0

表 3.2: リサンプリングアタックの誤り率

表 3.2 によって，実験結果は参考文献 [1] 結果が同じであり，同じリサンプリング耐性を持つ．

3.3.2 再量子化

処理のフローチャートは図 3.6 に示す．

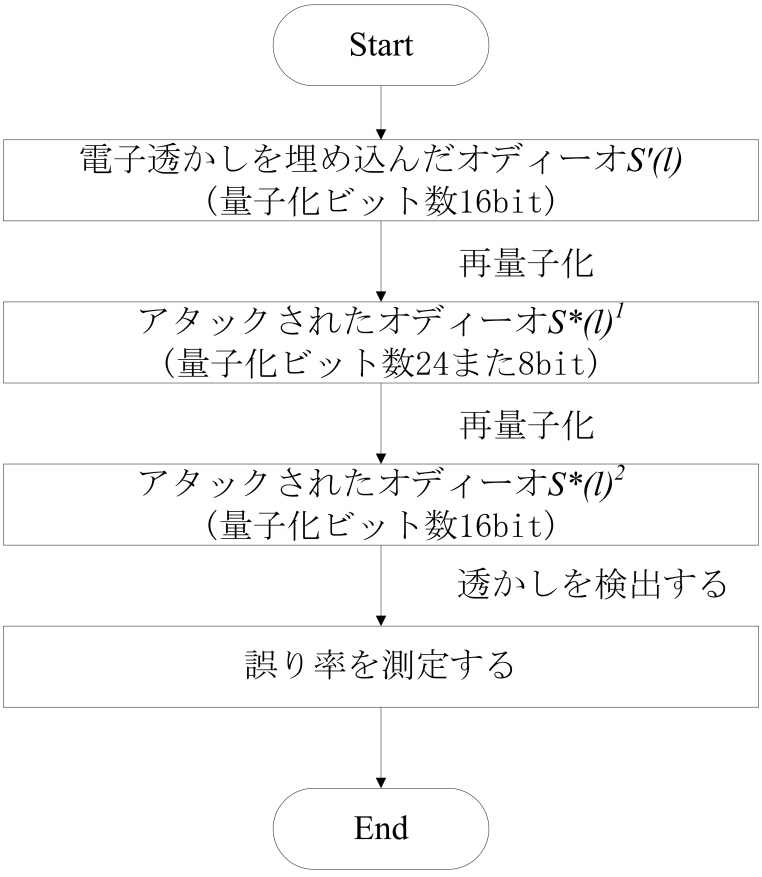

図 3.6: 再量子化アタックのフローチャート

電子透かしを埋め込んだオーディオ $S'(l)$ に対する，再量子化アタックする．ここで，再量子化ことがソフトウェア (Adobe Audition CS6) で実現する．$S^*(l)^1$ は第 1 回再量子化

アタックされたオーディオであり，$S^*(l)^2$ は第 2 回再量子化アタックされたオーディオである．アタックされたオーディオ $S^*(l)^2$ から電子透かし検出を行う．検出した電子透かしの誤り率を測定する．

図 3.6 による，再量子化アタックの誤り率の結果は表 3.3 に示す．

量子化ビット数	BER(%)(実装結果)	BER(%)(参考文献 [1] 結果)
24	0	0
8	0	0

表 3.3: 再量子化アタックの誤り率

表 3.3 による，実験結果は参考文献 [1] 結果が同じであり，同じ再量子化耐性を持つ．

3.3.3 MP3 圧縮

処理のフローチャートは図 3.7 に示す．

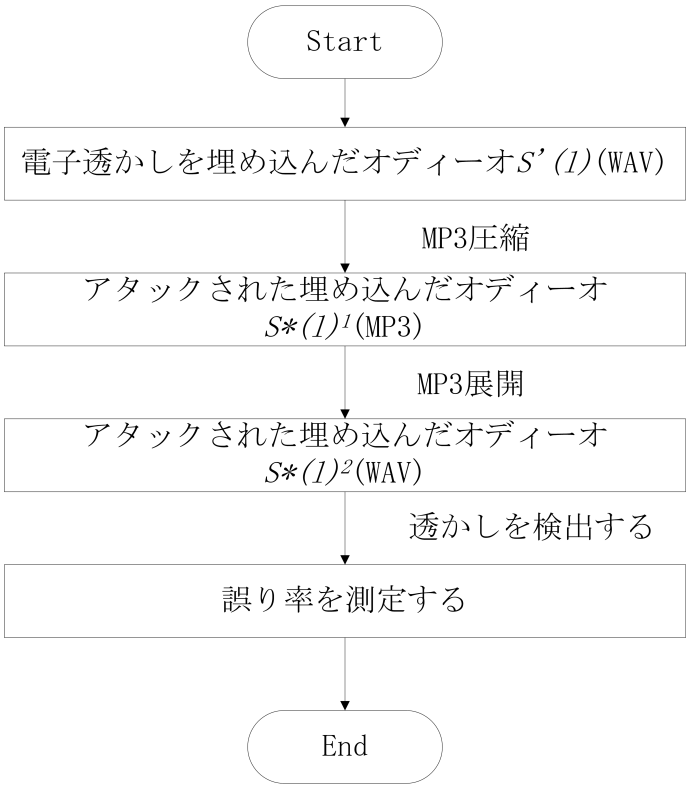

図 3.7: MP3 圧縮のフローチャート

電子透かしを埋め込んだオーディオ $S'(l)$ に対する，MP3 圧縮アタックする．ここで，MP3 圧縮ことがソフトウェア (Format Factory 3.8.0) で実現する．$S^*(l)^1$ は MP3 圧縮ア

タックされたオーディオであり，$S^*(l)^2$ はデコーディングアタックされたオーディオである．アタックされたオーディオ $S^*(l)^2$ から電子透かし検出を行う．検出した電子透かしの誤り率を測定する．

図 3.6 による，MP3 圧縮アタックの誤り率の結果は図 3.8 に示す．

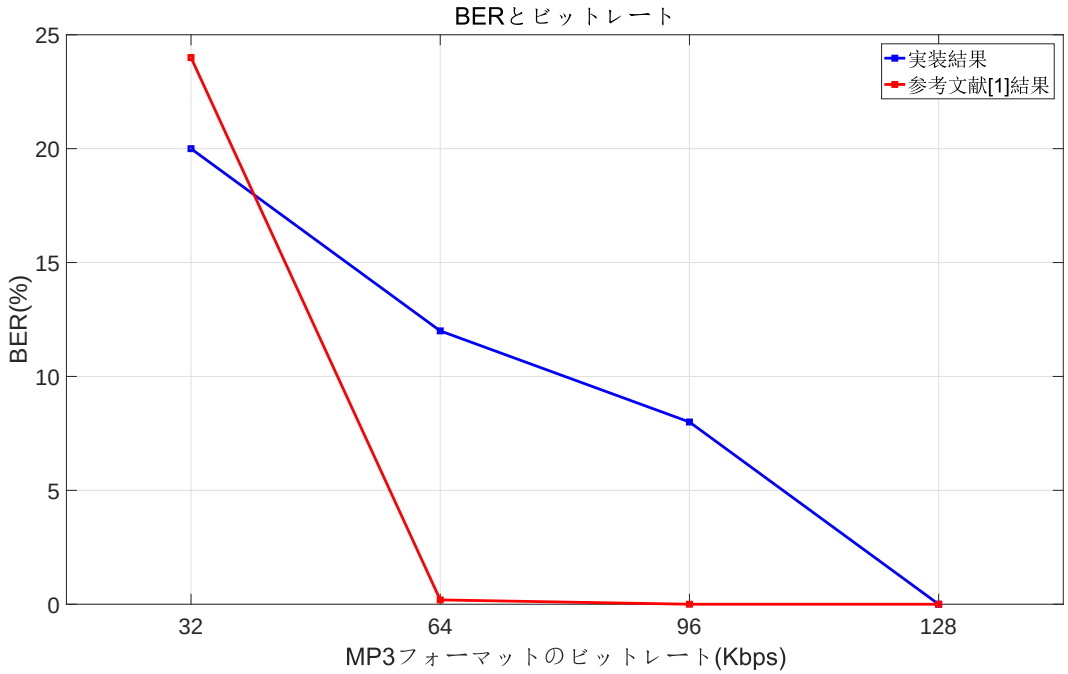

図 3.8: MP3 圧縮アタックの誤り率

図 3.8 によって，実験のオーディオデータと参考文献 [1] のオーディオデータは違うために，実験結果は完全に一致していない．実験結果は参考文献 [1] 結果の傾向が類似である．

3.4 まとめ

選択したアルゴリズムをソフトウェア (Matlab2016b) に実装した．アルゴリズムのアタック耐性 (リサンプリング，再量子化，ＭＰ３圧縮) を測定した．リサンプリングアタックに対し，22,050Hz と 11,025Hz のリサンプリングアタックが良い耐性を持った (検出した電子透かしの誤り率が 0 であった)．再量子化アタックに対し，8 ビットと 24 ビットの再量子化アタックが良い耐性を持った (検出した電子透かしの誤り率が 0 であった)．ＭＰ３圧縮に対し，ＭＰ３のビットレートによって，違う耐性を持った，ビットレートは 96bps から，アルゴリズムが良いＭＰ３圧縮耐性を持った (検出した電子透かしの誤り率が 0 であった)．

実験のオーディオデータと参考文献 [1] のオーディオデータは違うために，アタック実験結果は完全に一致していない．実験結果は参考文献 [1] 結果の傾向が類似であった．

第4章　アルゴリズムのハードウェア実装

4.1　はじめに

本章では，低消費エネルギー回路を構築ために，アルゴリズムをハードウェアに実装し，FPGA に電子透かしの検出回路を構築する．ハードウェアの特性を考慮し，離散ウェーブレット変換アルゴリズムのハードウェア実装方法を検討する．

4.2　ハードウェア化にアルゴリズムの変化

離散ウェブレットアルゴリズム検出する時，8 レベルのハール離散ウェーブレット変換をする．8 レベルのハール離散ウェーブレット変換の過程は図 4.1 に示す．

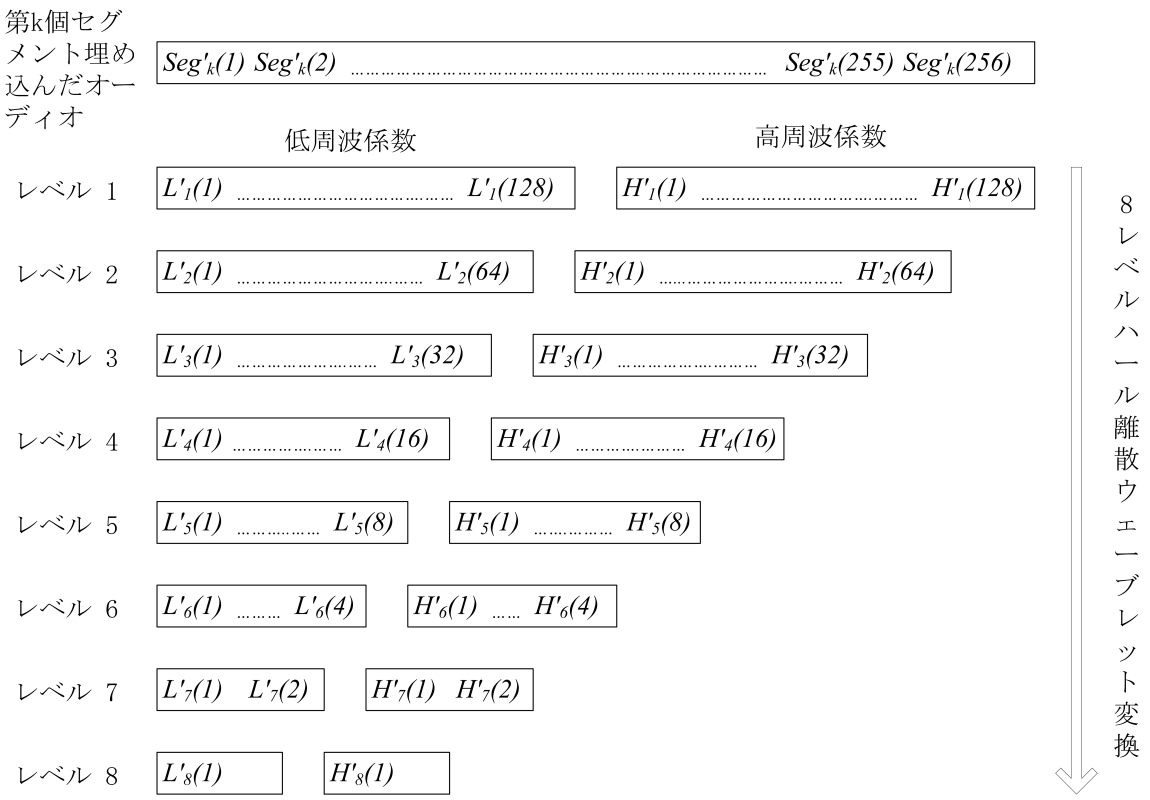

図 4.1: レベル 8 のハール離散ウェーブレット変換

図 4.1 による，毎レベルの演算は前レベルの演算結果を利用される．8 レベルのハール離散ウェーブレット変換する中に，255 回加算と 255 回引き算と 510 回乗算する必要がある．乗算を削減方法を検討する．毎レベルは前レベルの結果から演算する，レベル 8 の埋め込んだ低周波数係数 $L_8^{'}(j)$ とセグメント埋め込んだオーディオ $Seg_k^{'}(i)$ の関係は導出できる．式 4.1 によって，レベル 8 の低周波数係数は透かしを埋め込んだセグメントから直接で計算できる．

$$L_8^{'}(j) = \frac{1}{16} \times \sum_{i=1}^{256} Seg_k^{'}(i) \tag{4.1}$$

ハードウェアに実装アルゴリズムは図 4.2 に示す．

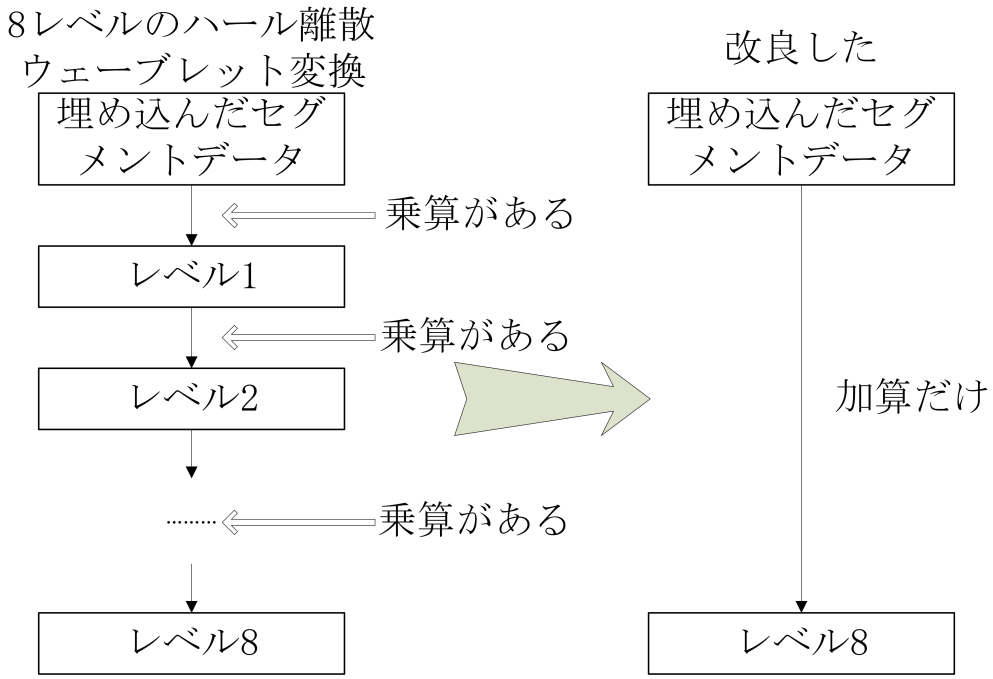

図 4.2: ハードウェアに実装アルゴリズム

図 4.2 による，8 レベルのハール離散ウェーブレット変換は加算だけで計算できる．乗算を回避する．

4.3 回路のアーキテクチャ

4.3.1 開発環境

電子透かしの検出回路を構築するには，デスクトップコンピューターと FPGA ボードから構成される．デスクトップコンピューターの環境は表 4.1 に示す．FPGA の開発環境は表 4.2 に示す．FPGA ボードの特性は表 4.3 に示す．

	説明
CPU	AMD Athlon 64 X2 5000+
メモリ	DDR2 4GB
ハードディスク	00JD-00FYB0 SCSI(250GB)
OS	Windows XP
インターフェイス	PCI Express Gen1 8-lane

表 4.1: デスクトップコンピューターの環境

	説明
CPU	Intel Core2 P7570 2.26GHz
メモリ	DDR3 4GB
ハードディスク	ST9320423AS ATA(320GB)
OS	Windows 7
ツール	ISE Design Suite 14.7

表 4.2: FPGA の開発環境

	説明
ボート	Xilinx ML605 Evaluation Board
FPGA	Virtex-6 XC6VLX240T-1FFG1156
クロック	200MHz
インターフェイス	PCI Express Gen1 8-lane

表 4.3: FPGA ボートの特性

オリジナルオーディオフォーマットと電子透かしデータは表 4.4 に示す.

	説明
オーディオ	WAV
サンプリング周波数	44,100Hz
量子化ビット数	16bit
チャンネル	1(モノラル)
透かしデータ	25bit(乱数)

表 4.4: 実装データのフォーマット

PCIe インターフェイスは Xilinx の IP Core を利用された．IP Core の特性は表 4.5 に示す．

	説明
IP Core	Xilinx Virtex-6 Integrated Block for PCI Express
IP Core Version	1.6
Release Date	September 21, 2010

表 4.5: IP Core

電子透かしの検出回路では，デスクトップコンピューターと FPGA ボードから構成される．システムのアーキテクチャは図 4.3 に示す．実物は図 4.4 に示す．

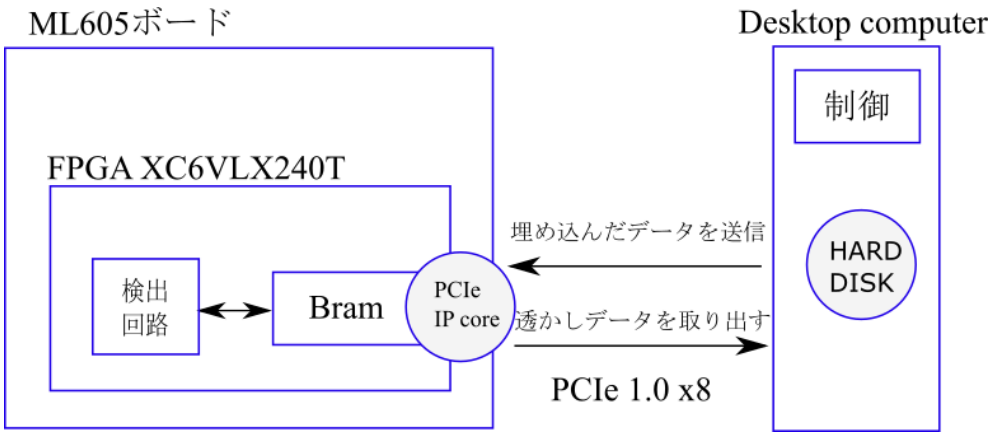

図 4.3: システムのアーキテクチャ

図 4.4: 実物

4.3.2 回路のフローチャート

電子透かしの検出システム中に，デスクトップコンピューターは検出システムの動作を制御する．ひとつ埋め込んだセグメントオーディオしかないの検出回路を構築した ($B = 1$)．複数 (3 個) セグメントの電子透かし検出回路も構築した，電子透かし検出回路はセグメント検出回路とセグメント管理回路から構成する．検出過程中に，各ステップの時間を測定ために，時間測定回路を入る．セグメント管理回路と時間測定回路間に矛盾がある．複数 (3 個) セグメントの電子透かし検出回路を失敗した．

本研究では，ひとつ埋め込んだセグメントオーディオしかないの検出回路を構築した ($B = 1$)．回路のフローチャートは図 4.5 に示す．回路のエネルギー評価を行うには、単一セグメント検出回路によって，推測することができる．単一セグメント検出回路中に，管理回路がない．複数セグメント検出回路のエネルギーが大きくなる可能性がある．

図 4.5: 回路のフローチャート

図 4.5 によって，ひとつ埋め込んだセグメントオーディオしかないの検出回路を構築した ($B = 1$)．

処理の流れを紹介する．検出回路中に，時間測定回路を入り，時間測定回路の検証ために，まず，CPU の時間を送信する．CPU の時間を 32 ビットの二進数で FPGA の BRAM(Offsrt = 400) に送信する．FPGA は Offset = 400 の BRAM から 32 ビット CPU の時間の数値を取り出す．時間測定回路をスタート．

次に，セグメント埋め込んだオーディオを FPGA の BRAM に送信する．ここでは，毎オーディオサンプリングの数値は 32 ビットの二進数で表示する．1 番目から 256 番目のオーディオサンプリングは BRAM の Offset の 1 から 256 までに置く．FPGA は Offset = 1 から 256 までの BRAM からセグメント埋め込んだオーディオを取り出し，透かしを検出する．透かしは Offset = 302 の BRAM に書く．検出したら，Offset = 0 の BRAM に 32 ビット 1 を書く．コンピューターは FPGA の BRAM 中 Offset = 0 の数値を読み込む．数値は 32 ビット 1 のとき，Offset = 302 の BRAM から透かしを取り出す．数値は 32 ビット 1 ではないのとき，FPGA の処理を待つ，

電子透かしの検出回路は PCIe Core ソースの PIO_EP コンポーネット中に入れリ，PIO_EP コンポーネットを改造する．FPGA の回路は図 4.6 に示す．

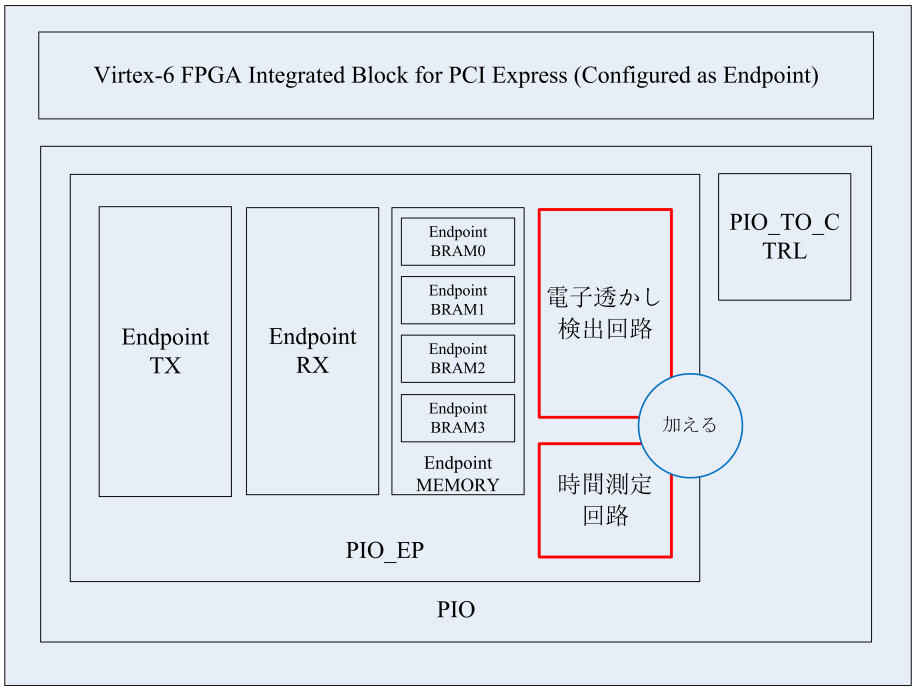

図 4.6: FPGA 回路

FPGA のリソース使用量は表 4.6 に示す.

	使用量	使用可能量	使用率
Slice Registers	1,794	301,440	1%
Slice LUTs	1,864	150,720	1%
RAMB36E1/FIFO36E1s	12	412	2%
BUFG/BUFGCTRLs	6	32	18%

表 4.6: リソース使用量

4.4 まとめ

本章では，実装方法をハードウェアの特性を考慮し，離散ウェーブレット変換の乗算を回避した．アルゴリズムをハードウェアに実装した，デスクトップコンピューターは PCIe インターフェイスで FPGA の BRAM に埋め込んだオーディオデータを送信する．電子透かしの検出回路を構築した．

第5章　評価

5.1　はじめ

本章では，構築した電子透かし検出回路の消費エネルギーを評価ために，オーディオ電子透かしアルゴリズムのソフトウェア処理時間と消費電力を測定する．同様に，アルゴリズムのハードウェア処理時間と消費電力を測定する．消費エネルギーを計算する．構築した電子透かし検出回路の消費エネルギーを評価する．

5.2　評価環境

ソフトウェアとハードウェアの消費エネルギーを比較ために，二つ実装方法の処理時間と検出電力を測定する．実装環境は5.1に示す．ソフトウェアはC言語で実装する．ハードウェアはPCIeインターフェイスでデスクトップコンピューターに接続する．CPUでは，二つCoreがあり，本研究中に，ひとつCoreを利用された．

	説明
CPU	AMD Athlon 64 X2 5000+
メモリ	DDR2 4GB
ハードディスク	00JD-00FYB0 SCSI(250GB)
OS	Windows XP
インターフェイス	PCI Express Gen1 8-lane
ソフトウェア	Visual Studio 2010

表 5.1: 評価の環境

5.3　消費エネルギーの評価

ソフトウェアとハードウェアの消費エネルギーを評価ために，処理時間と消費電力を測定する．

5.3.1 処理時間の評価

ソフトウェアとハードウェアの処理時間を評価する．ソフトウェアとハードウェアが同じ埋め込んだオーディオ $S'(n)$ を利用される．埋め込む強度 T は三名の音質聴取実験によって 9,000 を決める．音質聴取試験とは，実祭に人間に聴いてもらって音質が劣化していないか確認してもらう試験である．

ソフトウェアの検出時間では，10 万回 1 ビット電子透かしを検出するの平均値である．検出の時間は 4.0us である．測定方法はコンピューターのプラグラム中に時間測定関数 (GetTickCount) を挿入し測定する．

ハードウェアの処理時間の測定では，図 5.1 に示す．CPU の時間を 32 ビットの二進数で FPGA の BRAM(Offsrt = 400) に送信する．FPGA は Offset = 400 の BRAM から 32 ビット CPU の時間の数値を取り出す．時間測定回路をスタート．次に，セグメント埋め込んだオーディオを FPGA の BRAM に送信する．ここでは，毎オーディオサンプリングの数値は 32 ビットの二進数で表示する．1 番目から 256 番目のオーディオサンプリングは BRAM の Offset の 1 から 256 までに置く．セグメント埋め込んだオーディオを送信したら，時間測定回路の数値 hard_time1 を Offset = 300 の BRAM に書く．FPGA は Offset = 1 から 256 までの Block RAM からセグメント埋め込んだオーディオを取り出し，透かしを検出する．検出したら，時間測定回路の数値 hard_time2 を Offset = 301 の BRAM に書く．検出透かしは Offset = 302 の BRAM に書く．Offset = 0 の BRAM に 32 ビット 1 を書く．時間測定回路の数値を連続に Offset = 500 の BRAM に書く．コンピューターは FPGA の BRAM 中 Offset = 0 の数値を読み込む．数値は 32 ビット 1 のとき，Offset = 500 の BRAM から時間 hard_time3 を取り出す．

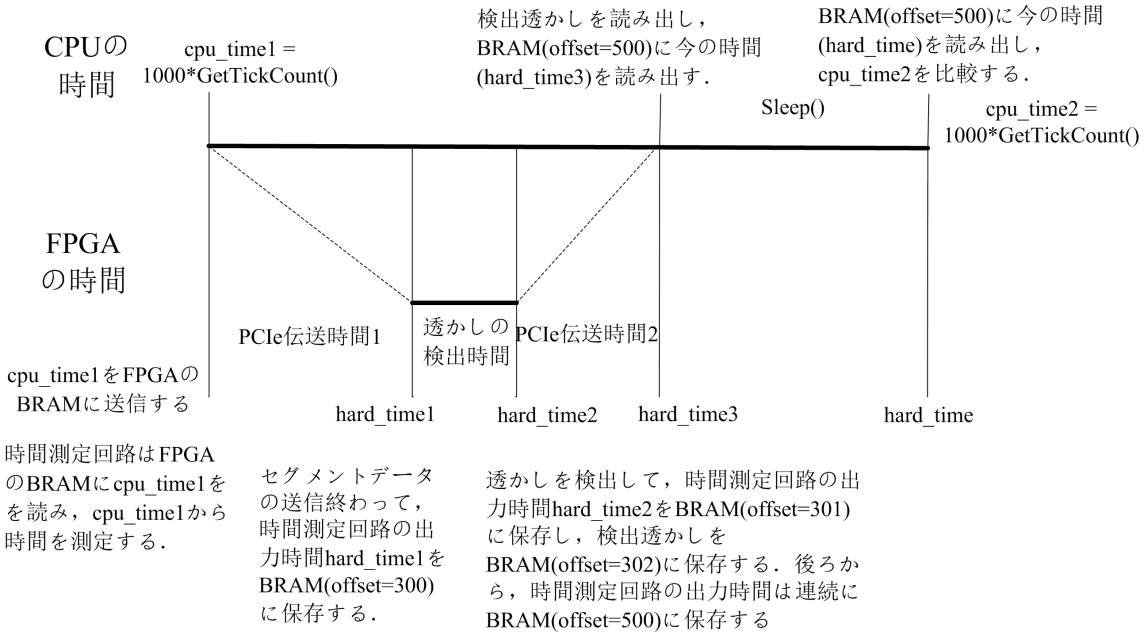

図 5.1: ハードウェア処理時間の測定

実験結果は表 5.2 に示す.

	検出時間 (us)
PCIe 伝送時間 1	6
透かしの検出時間	1
PCIe 伝送時間 2	3

表 5.2: ハードウェア処理時間

5.3.2 消費電力の評価

ソフトウェアとハードウェアの消費電力を評価する.

- ソフトウェア中に，CPU の消費電力を測定ために，消費電力測定ソフトウェア Central Brain Identifier(8.3.6.3) を利用される．ソフトウェアで透かしを検出する時，CPU の消費電力は 51.25w であった．透かしを検出ソフトウェアの CPU 使用率は 81 ± 10%であった.
- ハードウェアの消費電力は ISE ツールの Analyze Power Distribution(XPower Analyze) でシミュレーションする.

検出するのハードウェア消費電力は表 5.3 に示す．PC から FPGA にデータを送信する時，ハードウェア中の検出回路を使わない，データを送信する時のハードウェアの消費電力は表 5.4 に示す.

	消費電力 (W)
Clock	0.066
Logic	0.006
Signals	0.019
BRAMs	0.051
MMCMs	0.083
PCIEs	0.128
IOs	0.019
GTXE1s	1.499
Leakage	2.937
Total	4.807

表 5.3: 検出するのハードウェア消費電力

	消費電力 (W)
Clock	0.060
Logic	0.002
Signals	0.012
BRAMs	0.040
MMCMs	0.083
PCIEs	0.128
IOs	0.000
GTXE1s	1.499
Leakage	2.936
Total	4.759

表 5.4: 送信するのハードウェア消費電力

ハードウェアで透かしを検出する時，CPU の消費電力は 45.35w であった．CPU 使用率は 80 ± 10%であった．

ソフトウェアとハードウェアの消費エネルギーは以下に示す．

ソフトウェアの消費エネルギー (J) : $51.25 \times 0.81 \times 4 \times 10^{-6} = 0.00017$

ハードウェアの消費エネルギー (J) : $(45.35 \times 0.80 + 4.759) \times 9 \times 10^{-6} + (45.35 \times 0.80 + 4.807) \times 1 \times 10^{-6} = 0.00041$

5.4 まとめ

本章では，選択したオーディオ電子透かしアルゴリズムのソフトウェア処理時間と消費電力を測定した．同様に，アルゴリズムのハードウェア処理時間と消費電力を測定した．検出時間と検出電力の細部を分析した．ソフトウェアとハードウェアの消費エネルギーを計算した．ハードウェアの消費エネルギーはソフトウェアの約 2 倍になった．

第6章　結論

6.1　まとめ

デジタルマルチメディア及びコンピュータネットワーク技術の急速な発展に伴い，デジタルマルチメディア情報のプロセス及び配信するのは，非常に容易になる．しかし，著作権保護の問題がなっている．デジタルマルチメディアの著作権保護のために，電子透かし技術が開発されている．オーディオ電子透かしのアルゴリズムについて調査し，ハードウェアの特性を考慮して離散ウェーブレット変換 [1] アルゴリズムを選択した．ＦＰＧＡに電子透かしの検出回路を構築した．ソフトウェアとハードウェアの処理時間と消費電力を測定した．ソフトウェアとハードウェアの消費エネルギーを計算した．処理時間と消費電力の細部を分析する．ハードウェアの消費エネルギー主にはデータを伝送に使用された，ハードウェア検出部分の消費エネルギーはソフトウェアより，小さくになった．

6.2　今後の課題

本研究中に，CPU の消費電力を測定するとき，CPU の使用率でソフトウェア消費電力を計算する．計算方法の精度を改善の余地がある．ハードウェアにデータを送信するとき，大きくエネルギーをかかった．構築した回路を改善の余地がある．コンピュータから送信られたオーディオデータは FPGA の一個 BRAM に保存する，オーディオデータを取り出すとき，ボトルネックがある．データの伝送問題を改善し，検出回路を並列化できる．検出時間を短縮することができる．ハードウェアの消費エネルギー主には PCIe でデータを伝送に使用された．高速な伝送方法は、効果的にエネルギーを減少することができる．

参考文献

[1] Shaoquan Wu, Jiwu Huang, Daren Huang, Y. Q. Shi, Efficiently self-synchronized audio watermarking for assured audio data transmission, IEEE Transactions on Broadcasting, Volume 51, Issue 1, Year 2005.

[2] In-Kwon Yeo, Hyoung Joong Kim, Modified patchwork algorithm: a novel audio watermarking scheme, IEEE Transactions on Speech and Audio Processing, Volume 11, Issue 4, Year 2003.

[3] Byeong-Seob Ko, Ryouichi Nishimura, Yoiti Suzuki, Time-Spread Echo Method for Digital Audio Watermarking, IEEE Transactions on Multimedia, Volume.7, NO.2, Year 2005.

[4] Gerzon Michael A, Craven Peter G, A High-Rate Buried-Data Channel for Audio CD, Journal of the Audio Engineering Society, Volume 43, Issue 1, Year 1995.

[5] 高乗量, 西村竜一, 鈴木陽一, エコー拡散透かし手法における音質と埋め込み容量に関する検討, 情報処理学会研究報告, pp125-130, 2002.