電波望遠鏡用 AWF 型分光器に関して

中原 啓貴[1]　中西 裕之[1]　直井 隆浩[1]　岩井 一正[2]　笹尾 勤[3]

概要：電波望遠鏡は天体から放射される電波を受信し, 分光器を用いて解析を行う装置である. 既存の分光器は受信した電波に対し, まず, 窓関数 (Window function) をかけ, 高速フーリエ変換 (FFT) を行う. そして, ノイズを除去するため, 周波数スペクトルの積算処理 (Accumulation) を行う. この順序で処理を行う分光器を WFA 型分光器という. WFA 型分光器は国際開発プロジェクト CASPER にて FPGA ボードを用いて実現されており, 様々な電波望遠鏡で利用されている. しかし AD 変換器の高速化が進んでおり, 数百 MHz で動作する FPGA と数十 GHz で動作する AD 変換器間でギャップが生じている. 性能を引き出すためには FFT を並列化するしかなく, FFT のハードウェア量がボトルネックとなっていた. 本論文では, 計算順序を入れ替えた AWF 型分光器を提案する. FFT と比較して窓関数をかける回路は単純であり, 並列化に向く. CASPER で公開されている ROACH ボード上に AWF 型分光器を実装して従来の WFA 型分光器を比較を行い, 提案手法が優れていることを明らかにする.

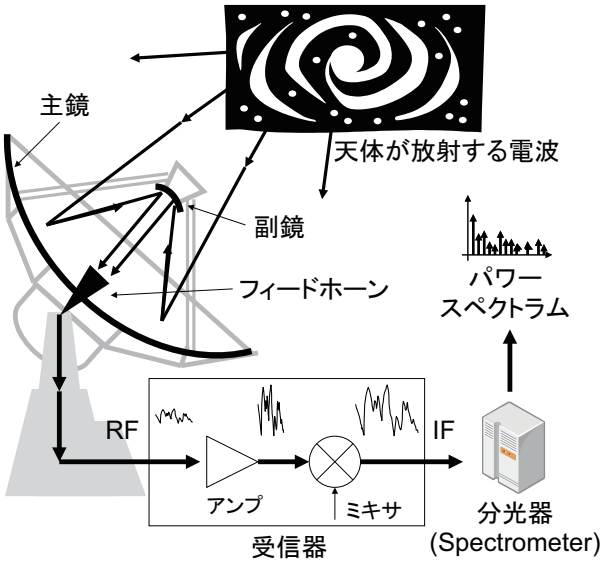

図 1　電波望遠鏡.

1. はじめに

1.1 電波望遠鏡における分光器

電波望遠鏡とは天体が放射する電波を受信し, 観測する装置である. 図 1 に電波望遠鏡の構成を示す. 天体からの電波 (RF: Radio Frequency) を受信アンテナ上の主鏡と副鏡によって反射する. 次に, フィードホーンによって電磁波を空間モードから導波管モードに変換し, 受信機へと転送する. 受信機は, 増幅器とミキサを用いて扱いやすい信号 (IF: Intermediate Frequency) に変換する. 最後に, **分光器 (Spectrometer)** を用いて時間領域の信号を周波数領域の信号に変換し, 観測を行う.

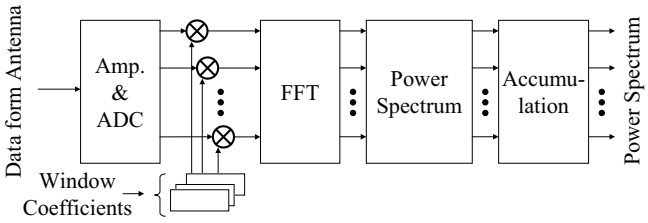

図 2　デジタル分光器.

図 2 に太陽観測用電波望遠鏡である AMATERAS [1], [7] で用いられているデジタル分光器 (AC240 Digitizer, Xilinx 社 Virtex-II Pro 搭載) [3] を示す. アンテナで受信したアナログ信号は 2GS/s の AD 変換器でデジタル信号に変換される. 次に, エイリアジングの影響を押さえるため, 窓関数をかける. そして, 16.384 usec 毎に 2^{14} 点 FFT を行い時間領域から周波数領域の信号 (18 ビット固定小数点) に変換する. FFT の出力は複素信号なので, パワースペクトラムに変換する. 電波望遠鏡では高速信号を扱うため, AD 変換器の精度を上げることができない. 従って, 複数のパワースペクトラムに対して 610 回積算処理を行って精度を上げたスペクトラムを 10 msec 毎に出力する. この分光器は窓関数 (Window), FFT, 積算処理 (Accumulation) を

[1] 鹿児島大学
Korimoto 1–21–40, Kagoshima, 890–0065, Japan
[2] 国立天文台 野辺山太陽電波観測所
Minamisaku, Nagano, 384–1305, Japan
[3] 明治大学
Kawasaki, Kanagawa 214–8571, Japan

表 1 電波望遠鏡開発国際コミュニティ CASPER で用いられている AD 変換器.

ADC ボード名	搭載 ADC	サンプル速度 [GS/s]	分解能 [bit]
ADC2x1000-8	AT84AD001B	2.0	8
ADC1x3000-8	ADC083000	3.0	8
KatADC	ADC08D1520	3.0	8
ADC1x5000-8	EV8AQ160	5.0	8
ADC1x10000-4	ASNT7120	10.0	4
ADC1x2200-10	AT84AS008	2.2	10

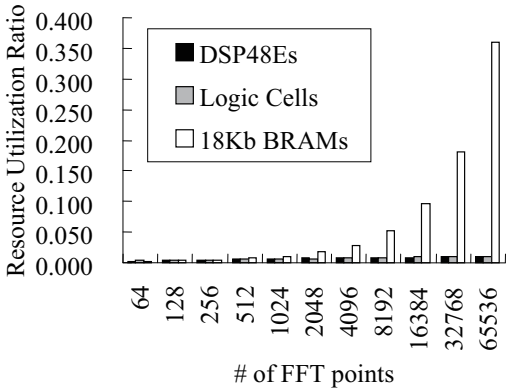

図 3 Xilinx 社 FFT ライブラリ (version 7.1) のリソース利用率 (FPGA: Virtex 7 XC7VX485T).

行うため, 各処理の頭文字を取って WFA 型分光器と呼ぶ. WFA 型分光器は様々な電波望遠鏡で用いられている [4], [8], [10], [11].

1.2 WFA 型分光器の問題点

表 1 にデジタル分光器国際共同開発コミュニティである CASPER [4] で使われている AD 変換器のスペックを示す. 表 1 が示すように, AD 変換器の変換速度はギガサンプル毎秒 (Gsps) であるのに対して, FPGA の動作周波数は数百 MHz と AD 変換器の速度に追いつかない. 従って, 複数の分光器を並列動作させるか, デッドタイムを設けて FPGA の処理速度に AD 変換器を合わせるかのどちらかである. 前者は図 3 が示すように, 通常の FFT ライブラリ [12] を率直に実装しただけではメモリ数が過多となり, 多数の FPGA を必要とするため高コストである. 後者は測定データが著しく劣化してしまう.

1.3 本論文の貢献点

本論文では, 精度を落とさずにコンパクトな分光器を提案する. 処理順序を積算処理, 窓関数, FFT とする分光器を AWF 型分光器とよぶ. また, FFT をコンパクトに実装するため, Six-Step アルゴリズムを用いた FFT [9] を実装する. 本論文の貢献点を以下に示す.

AWF 型分光器の提案: 著者らの知る限りでは, 電波望遠鏡用に AWF 型分光器を提案したのは初めてである.

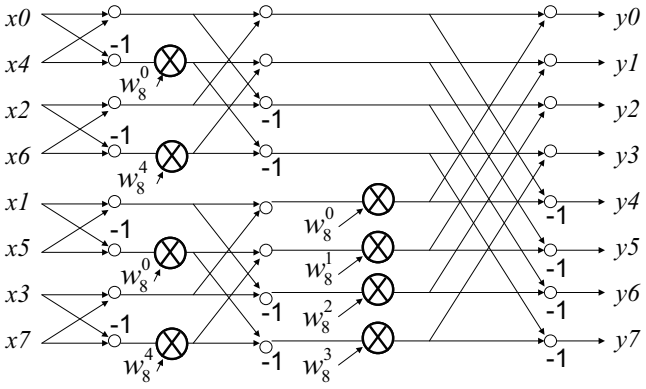

図 4 シグナル・フロー・グラフ.

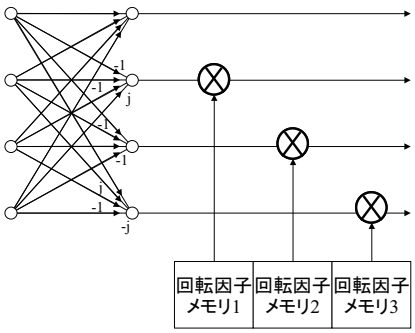

図 5 基数 4 のバタフライ演算器.

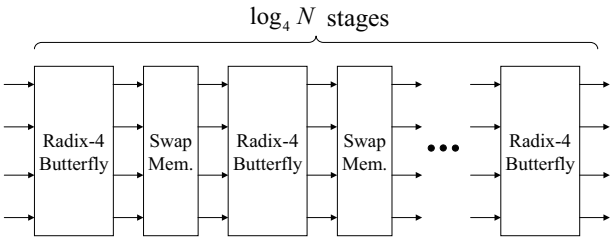

図 6 Radix-4 FFT.

AWF 型分光器の解析: FFT 点数 N, ADC の動作周波数, 及び FPGA の動作周波数に対するハードウェア量を明らかにした.

AWF 型分光器を実装し, 既存の分光器との比較: 国際共同プロジェクト CASPER で開発されている ROACH2 ボード上の FPGA に AWF 型分光器を実装した. また, 既存の分光器と比較を行い, 提案手法の有効性を明らかにした.

1.4 本論文の構成

本論文の構成は以下の通りである. 第 2 章で Radix-4 FFT について述べる. 第 3 章で Six-Step アルゴリズムに基づく並列 FFT について述べる. 第 4 章で AWF 型分光器について述べ, ハードウェア量の解析を行う. 第 5 章で実験結果を示し, 第 5 章で本論文のまとめを行う.

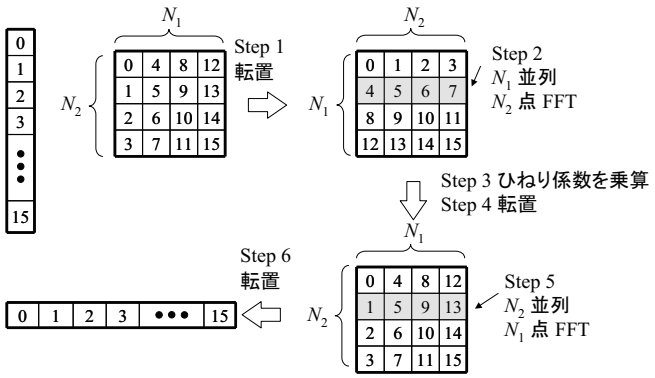

図 7 Six-Step FFT アルゴリズム ($N = 16$).

2. Radix-4 FFT

N 個の複素数データ $(a_0, a_1, \ldots, a_{N-1})$ に対し, 次の式で定義される $(c_0, c_1, \ldots, c_{N-1})$ をその**離散フーリエ変換 (Discrite Fourier Transform)** という.

$$c_k = \sum_{j=0}^{N-1} a_j w_N^{jk}, \tag{1}$$

ここで, $w_N^{jk} = exp(-2\pi i \frac{jk}{N})$ を**回転因子 (Twiddle factor)** という. 定義式を率直に計算した場合, DFT の計算量は $O(N^2)$ である.

図 4 に式 (1) から得られる**シグナル・フロー・グラフ**を示す [5]. 出力を次のステージのバタフライ演算器に入力するために, インデックスをスワップする回路が必要である. 多くの FFT ライブラリはこれをメモリで実現しており, 本論文でも**スワップ・メモリ**で実現する. $s = log_r N$ とする. s をステージ数といい, r を FFT の基数という. 従って, スワップ・メモリの個数は s 個であり, 各スワップ・メモリのサイズは Nw である. ここで, w は FFT 内部のビット精度である. r が 2 よりも大きい場合, 複数のバタフライ演算器を用いて実現する. 従って, r を大きくするとスワップ・メモリを削減できるが, バタフライ演算器の個数は増える. 図 5 に $r = 4$ のときの**基数 4 バタフライ演算器**を示す. 多くの FFT ライブラリは図 6 に示す基数 4 バタフライ演算器を用いた **Radix-4 FFT** を用いている [6]. このとき, 基数 4 バタフライ演算器は複素乗算を行える DSP ブロックを 3 個用いており, 回転因子を保持する回転因子メモリを 3 個用いている. 従って, s ステージの基数 4 バタフライ演算器に必要な DSP48E 数は $3s$ 個であるから, N 点 FFT では $3log_4 N$ 個である. 一方, 回転因子メモリの大きさは Nw ビットであるから, 基数 4 バタフライ演算器では $3Nw$ ビットである. 従って, s ステージでは回転因子メモリの大きさは $3Nwlog_4 N$ ビットである. スワップ・メモリは $Nwlog_4 N$ であるから, N 点 FFT では BRAM のメモリ量は $4Nwlog_4 N$ ビットである.

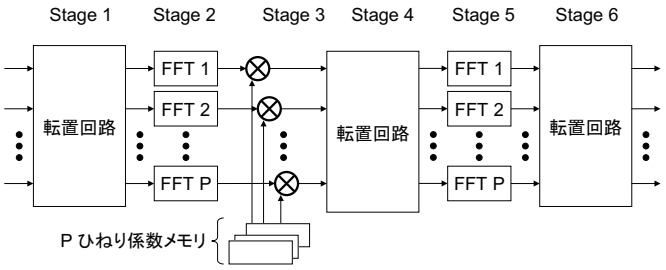

図 8 6StepFFT 回路.

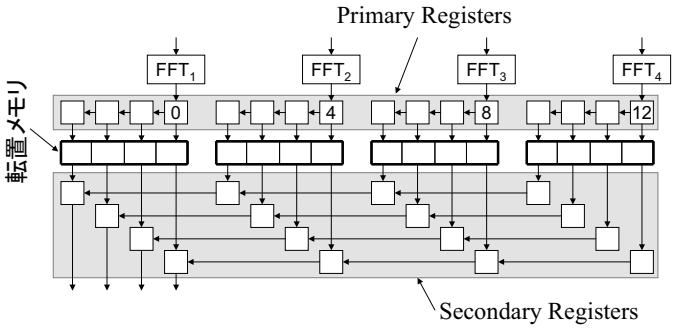

図 9 転置回路.

3. Six-Step FFT アルゴリズムに基づく FFT

3.1 Six-Step FFT アルゴリズム

N 点 DFT を $N = N_1 \times N_2$ と分解できるとき, 式 (1) における j, k は

$$j = j_1 + j_2 N_1$$
$$k = k_2 + k_1 N_2$$

と表現できる. ここで式 (1) の a と c は次に示す二次元配列 (columnwise) で表現できる.

$$a_j = a(j_1, j_2), 0 \le j_1 \le N_1 - 1, 0 \le j_2 \le N_2 - 1$$
$$c_j = c(k_2, k_1), 0 \le k_1 \le N_1 - 1, 0 \le k_2 \le N_2 - 1$$

従って式 (1) は次のように変形できる.

$$c(k_2, k_1) = \sum_{j_1=0}^{N_1-1} \sum_{j_2=0}^{N_2-1} x(j_1, j_2) w_{N_2}^{j_2 k_2} w_N^{j_1 k_2} w_{N_1}^{j_1 k_1} \tag{2}$$

ここで, $w_N^{j_1 k_2}$ をひねり係数という. 式 (2) から Six-Step FFT アルゴリズム [2] が導出できる.

アルゴリズム 3.1 (Six-Step FFT アルゴリズム)

1. 転置: $a_1(j_2, j_1) = a(j_1, j_2)$.
2. N_1 組の N_2 点 multicolumn FFT: $a_2(k_2, j_1) = \sum_{j_2=0}^{N_2-1} a_1(j_2, j_1) w_{N_2}^{j_2 k_2}$.
3. ひねり係数の乗算: $a_3(k_2, j_1) = a_2(k_2, j_1) w_N^{j_2 k_2}$.
4. 転置: $a_4(j_1, k_2) = a_3(k_2, j_1)$.
5. N_2 組の N_1 点 multicolumn FFT: $a_5(k_1, k_2) = \sum_{j_1=0}^{N_1-1} a_4(j_1, k_2) w_{N_1}^{j_1 k_1}$.
6. 転置: $c(k_2, k_1) = a_5(k_1, k_2)$.
7. 停止.

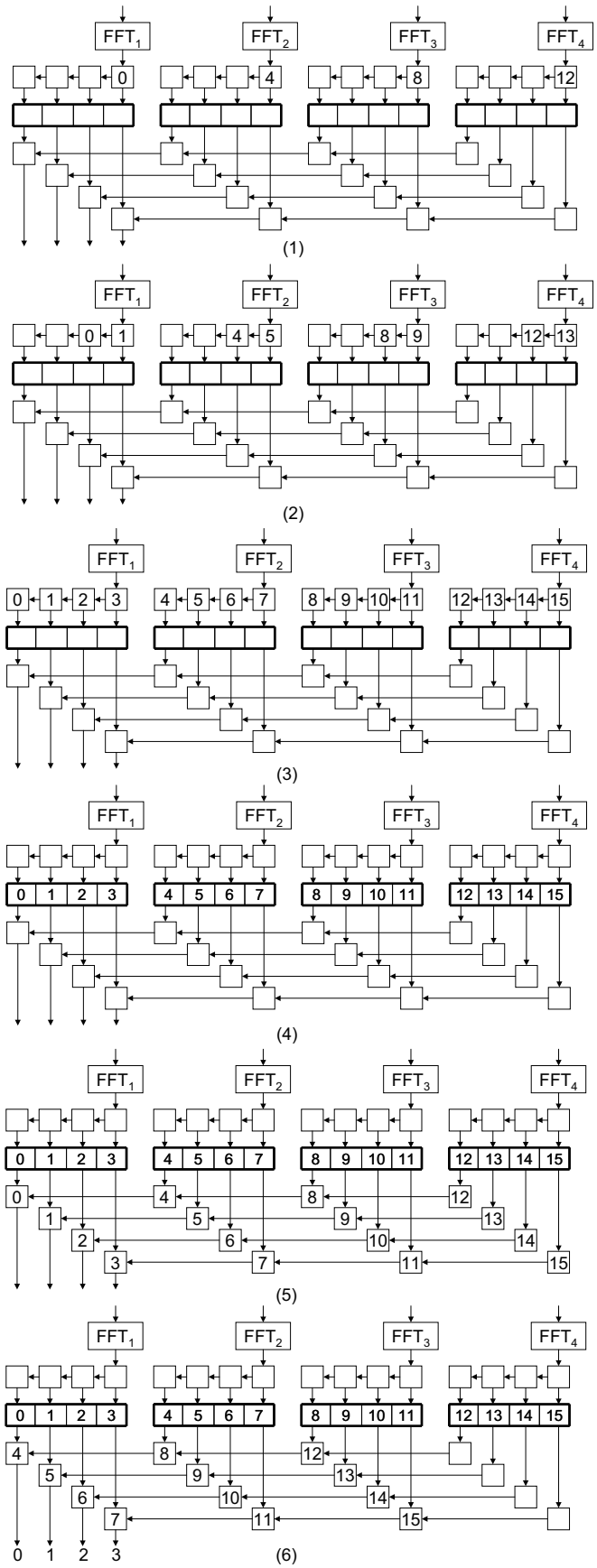

図 10 4 並列 FFT 用転置回路の動作例.

本論文では FFT の面積効率が最も良い $N_1 = N_2 = \sqrt{N} = N'$ とする. Six-Step FFT アルゴリズムでは N 点

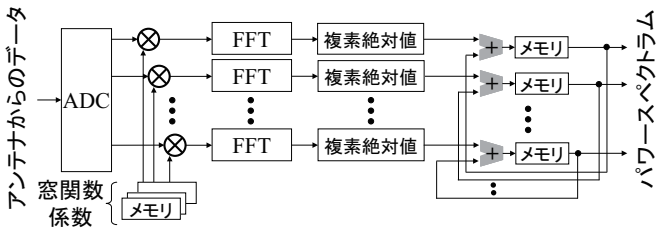

図 11 WFA 型分光器.

FFT を 2 つの $\sqrt{N}$ 点 FFT に分解できるため, FFT の回路を大幅に削減できる. 従って, 空いたリソースを並列化に使うことができる. しかし, Six-Step FFT アルゴリズムでは行列の転置が 3 回必要になる. 図 7 に Six-Step FFT アルゴリズムの概要を示す. Step 2. と Step 5. における N' 点 FFT を並列実行することで, N 点 FFT を高速実行できる.

3.2 Six-Step FFT 回路

図 8 に P **並列 Six-Step FFT 回路**を示す. P を FFT の並列度とし, w を FFT 内部のビット精度とする. P 並列 FFT から P 個のデータが出力されるので, 同時に転置を行わなければならない. 図 9 に**転置回路**を示す. P 個のレジスタで FFT からの出力を受け取る. そして P 個のシフトレジスタを用いてデータをシフトしていき, P^2 個のデータを P 個の**転置メモリ**に同時書き込みする. パイプライン動作のため, 同時に P^2 個のデータを出力する. ただし, 出力側のシフトレジスタは転置を実現するため接続が異なる.

例 3.1 図 10 に図 7 に示した Six-Step FFT アルゴリズムの Step 2 から Step 4 を実現する 4 並列 FFT ($P = 4$) の転置回路の例を示す. なお, 図 10 はひねり係数回路を省略している.

まず, $\sqrt{N}$ 点 FFT を行う. 出力はストリーミングで出力されるので, 転置回路の入力側のシフトレジスタで保持する (図 10 (1)〜(3)). 次に, 並列 FFT の出力を転置メモリに保持する (図 10 (4)). そして, 転置メモリの出力を出力側のシフトレジスタに読み出す (図 10 (5)). 出力側のシフトレジスタをシフトし, 出力を順次ストリーミング出力する (図 10 (6)). (例終)

従って, 転置メモリの大きさは $\frac{\sqrt{N}Pw}{P^2}$ となる. この転置メモリが P 個必要なので, 転置メモリ量は $P\frac{\sqrt{N}Pw}{P^2} = \sqrt{N}w$ である. しかし, FPGA のメモリの大きさは下限があるため, P が大きいとき, メモリ数が増えてしまう. 一方, シフトレジスタは入出力に Pw ビットのシフトレジスタを P 並列用いるのでその大きさは P^2w ビットである. 実際には転置メモリを 2 重化し, 前段の FFT の出力の保持と次段の FFT への出力を同時に行う.

4. AWF 型分光器

既存手法の WFA 型分光器と提案手法の AWF 分光器のハードウェア量の見積もりを行う. 通常, AD 変換器は GHz

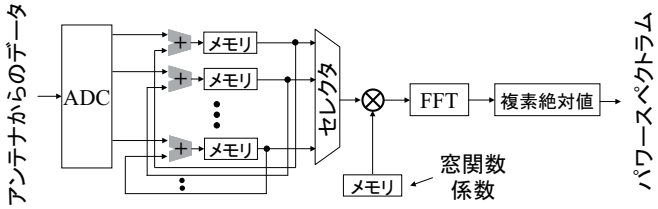

図 12 AWF 分光器.

オーダで駆動するのに対し, FPGA は MHz オーダで駆動するため, 速度ギャップを埋めるために FPGA 上に並列回路を実現する. ここで, 並列度を P, AD 変換器の動作周波数を f_{ADC}, FPGA の動作周波数を f_{FPGA} とすると

$$P = \frac{f_{ADC}}{f_{FPGA}}$$

となる. 以降, N 点 FFT を行う分光器のハードウェア量を解析する.

4.1 WFA 型分光器

図 11 に既存手法である WFA 型分光器を示す. まず, ADC からの信号に対して並列に窓関数をかける. 次に, 並列に FFT を行う. そして, 各出力の複素絶対値を求め, 積算処理を行う. 従って, 窓関数をかけるには係数を保持するメモリが必要なので,

$$\frac{N}{P} w_{win} \times P = N w_{win}$$

ビット必要である. ここで w_{win} は窓関数部のビット精度である. また, 乗算器は $2P$ 個必要である. 次に, Six-Step FFT に必要なハードウェア量を見積る. Six-Step FFT では, N 点 FFT を行うのに $\sqrt{N}$ 点 FFT を 2 個使い, 転置回路を 3 個用いるので, P 並列にした場合, 必要なメモリ量は

$$3 \times 2\sqrt{N} w_{FFT} + 2P(3N w_{FFT} log_4 \sqrt{N} + 4N w_{FFT} log_4 \sqrt{N})$$

ビットである. ここで, w_{FFT} は FFT 内部のビット精度 [*1]である. 一方, 乗算器は FFT 内部のバタフライ演算と Six-Step FFT のひねり回路用に必要なので

$$2P \times 3 log_4 \sqrt{N} + P$$

個必要である. 次に, 複素絶対値回路で必要な乗算器は $2P$ 個である. また積算回路に必要なメモリ量は

$$\frac{N}{p} w_{acc}^{WFA} \times P = N w_{acc}$$

ビットである. また, P 個の加算器が必要である. ここで, w_{acc}^{WFA} は WFA 分光器の積算回路のビット精度である. 通常, 電波望遠鏡は T_{observ} 時間毎にパワースペクトルを出力するので, 積算回数 Add は

$$Add = \frac{T_{observ}}{N \times f_{ADC}^{-1}}$$

[*1] 本論文では, FFT 内部で w_{FFT} ビットに丸めると仮定する.

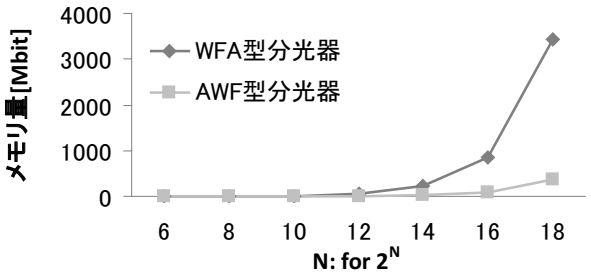

図 13 メモリ量の比較 ($T_{observ} = 1$ $[sec]$).

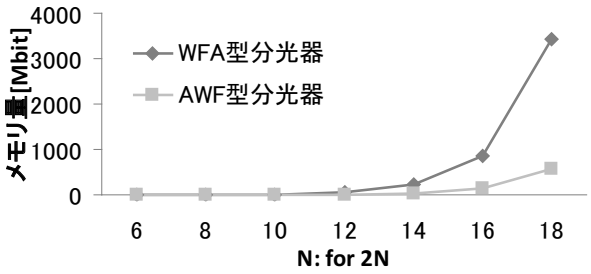

図 14 メモリ量の比較 ($T_{observ} = 1$ $[hour]$).

表 2 CASPER 分光器との比較.

分光器	FFT 点数	実効帯域 (GHz)	Slice 数	BRAM 数	DSP48E 数
CASPER (WFA 型)	2^{12}	0.4	8,518	34	76
提案手法 (AWF 型)	2^{12}	2.5	5,872	288	28

となる. 従って, w_{acc}^{WFA} は

$$w_{acc}^{WFA} = w_{FFT} + \lceil log_2 Add \rceil$$

となる.

4.2 AWF 型分光器

図 12 に提案手法である AWF 型分光器を示す. まず, ADC からの信号に対して並列に積算処理を行う. そして, セレクタを用いて信号を逐次読み出し, 窓関数をかける. 次に, FFT を行い複素絶対値を求める. AWF 型分光器では, 積算以降の処理は逐次処理で十分間に合う, つまり, 十分な時間 T_{observ} で観測すると仮定すると, 並列度 $P = 1$ で十分である. 従って, FFT の面積を大幅に削減できる可能性がある. このとき, 積算回路のビット精度 w_{acc}^{AWF} は

$$w_{acc}^{AWF} = w_{ADC} + \lceil log_2 Add \rceil$$

となる. ここで, w_{ADC} は AD 変換器のビット精度である. T_{observ} が長い時は w_{acc}^{WFA} が大きくなるため, 一概に FFT が小さくなるとは限らない. 本論文では実験的に解析を行い, AWF 型分光器の面積を明らかにする.

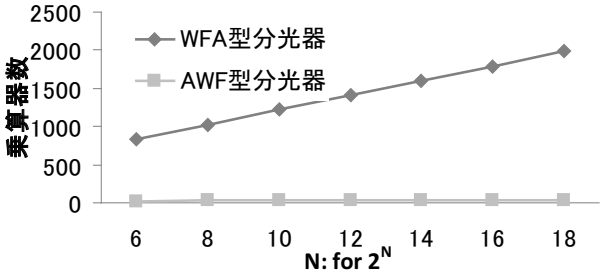

図 15 乗算器数の比較 ($T_{observ} = 1$ [sec]).

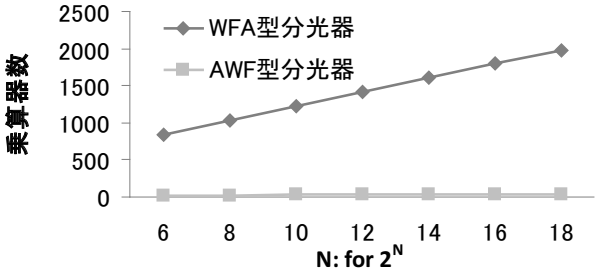

図 16 乗算器数の比較 ($T_{observ} = 1$ [$hour$]).

5. 実験結果

5.1 WFA 型分光器と AWF 型分光器の面積比較

N 点 FFT に対する WFA 型分光器と AWF 型分光器のメモリ量と乗算器数の比較を行った．ここで, AD 変換器は CASPER で使われている ADC1x5000-8 (サンプリング速度: 5.0 [GHz], 分解能: 8 ビット) を使うことを仮定した．従って, $f_{ADC} = 5.0$ [GHz] であり, $P = 64$ 並列の場合, $f_{FPGA} = 156.25$ [MHz] であるから, 過去の分光器の設計事例から FPGA 上に十分設計可能な動作周波数である．図 13, 14 にメモリ量の比較を, 図 15, 16 に乗算器数の比較を示す．これらの比較から, 提案 AWF 型分光器は FFT の並列度 P を大幅に削減できるため, 必要ハードウェア量を大幅に削減することができた．

5.2 他の分光器との比較

提案 AWF 型分光器 (T_{observ} =1 [hour], f_{ADC} = 5.0 [GHz], $N = 2^{12}$) を電波天文学国際共同開発 CASPER で用いている ROACH2 ボード上に実装した．ROACH2 ボードには Xilinx 社の FPGA (Virtex 6 SX475T, Slice LUT 数: 74,400 DSP48E 数: 1,064, BRAM 数: 2,128) が搭載されている．実装結果より, 最大動作周波数は目標の $f_{FPGA} = 156.25$ [MHz] となり, 動作できることが確認できた．このとき, 提案 AWF 型分光器は Slice 数を 5,872 個使用し, BRAM を 407 個使用し, 乗算器ブロック (DSP48E) を 28 個使用する．

国際開発共同プロジェクト CASPER で公開されている分光器を同一 FPGA (Virtex 6 SX475T) 上に実装して比較を行った．比較結果を表 2 に示す．CASPER 分光器と提案 AWF 型分光器は共に観測間隔を T_{observ} =1 [sec] として設計している．表 2 より, CASPER 分光器と比較して, 実効帯域を 6.25 倍拡張できた．

6. まとめと今後の方針

電波望遠鏡は天体から放射される電波を受信し, 分光器を用いて解析を行う装置である．高速化する AD 変換器に追いつくため, AWF 型分光器を提案した．AWF 型分光器は, まず, 積算処理 (Accumulation) を行い, 次に, 窓関数 (Window function) をかけ, Six-Step FFT アルゴリズムに基づく高速フーリエ変換 (FFT) を行う．AWF 型分光器の精度とハードウェア量を明らかにした．AWF 型分光器を国際開発プロジェクトで用いられている ROACH2 ボード上の FPGA に実装し, 既存の分光器よりも高速でかつコンパクトであることを明らかにした．

今後の課題は, 提案 AWF 型分光器を実在する電波望遠鏡に接続し, 天体観測を行うことである．

7. 謝辞

本研究は, 一部, 日本学術振興会・科学研究費補助金 (若手 (B): 課題番号 24700050), 及び日本学術振興会・頭脳循環を加速する若手研究者戦略的海外派遣プログラムによる．

参考文献

[1] AMATERAS, http://pparc.gp.tohoku.ac.jp/data/iprt/index.html

[2] D. H. Bailey, “FFTs in external of hierarchical memory,” *Proc. of ACM/IEEE Conf. on Supercomputing’89 (SC89)*, 1989, pp.234-242.

[3] A. O. Benz, P. C. Grigis, V. Hungerbhler, H. Meyer, C. Monstein, B. Stuber, and D. Zardet, “A broadband FFT spectrometer for radio and millimeter astronomy,” *Astronomy and Astrophysics*, No. 3568, 2008,

[4] CASPER: Collaboration for Astronomy Signal Processing and Electronics Research, https://casper.berkeley.edu/

[5] J. W. Cooley and J. W. Tukey, “An algorithm for the machine computation of complex fourier series,” *Mathematics of Computation*, Vol. 19, pp. 297-301, 1965.

[6] S. He and M. Torkelson, “A new approach to pipeline FFT processor,” *Proc. of the 10th Int’l Parallel Processing Symposium (IPPS1996)*, pp. 766-770, 1996.

[7] K. Iwai, F. Tsuchiya, A. Morioka, and H. Misawa, “IPRT/AMATERAS: A new metric spectrum observation system for solar radio bursts,” *Solar Phys*, Vol. 277, 2012, pp. 447-457.

[8] B. Klein, S. Hochgrtel, I. Krmer, A. Bell, K. Meyer1, and R. Gsten, “High-resolution wide-band Fast Fourier Transform spectrometers,” *Astronomy and Astrophysics*, 2012.

[9] H. Nakahara, K. Iwai, and H. Nakanishi, “A high-speed FFT based on a six-step algorithm: Applied to a radio

telescope for a solar radio burst," *The Int'l Conf. on Field-Programmable Technology (FPT 2013)*, pp. 430-443, 2013.

[10] H. Nakahara, H. Nakanishi, and T. Sasao, "On a wide-band fast Fourier transform for a radio telescope," *3rd Int'l Workshop on Highly-Efficient Accelerators and Reconfigurable Technologies (HEART 2012)*, May 31-June 1, pp.109-114, 2012.

[11] A. Parsons et.al, "PetaOp/Second FPGA signal processing for SETI and radio astronomy," *Proc. of the Asilomar Conference on Signals, Systems, and Computers*, 2006.

[12] Xilinx Inc., "LogiCORE IP fast fourier transform v7.1," 2011.