第１５章　ＦＰＧＡを用いた
中波帯ＡＭ用ソフトウェアラジオ

本稿掲載のWebページ

http://www.mybook-pub-site.sakura.ne.jp/Radio_note/index.html

目次

第 15 章

FPGA を用いた中波帯 AM 用ソフトウェアラジオ

15.1　組み立て

FPGA を用いたソフトウェアラジオの報告がある（参考文献 [11][12]）．高速 AD 変換器+FPGA+DA 変換器の組み合わせを用いればストレート方式ソフトウェアラジオ，スーパーヘテロダイン方式ソフトウェアラジオなどのソフトウェアラジオをハードウェアの変更を一切施さずに作ることができる．

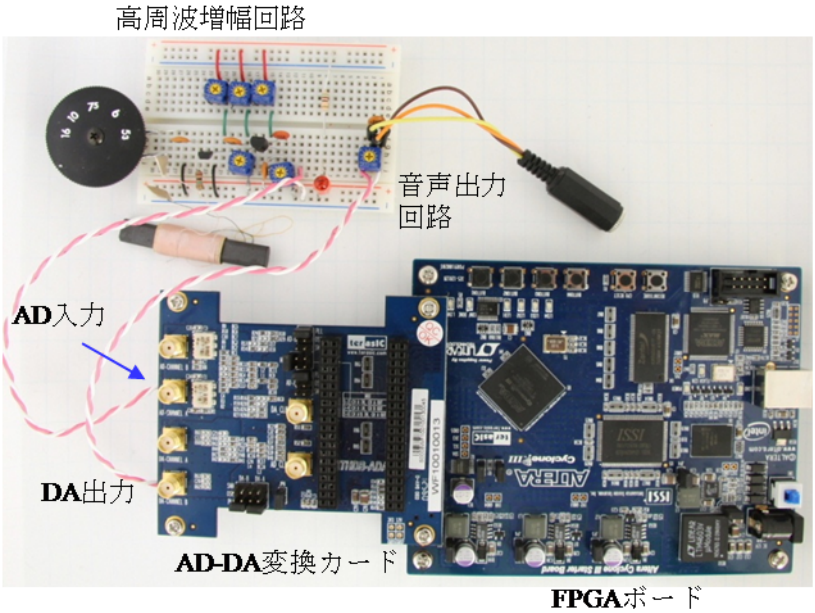


図 15.1: FPGA ラジオ

FPGA のスタータボードを利用すれば，少しお金はかかるが，製作は容易である．その例を図 15.1 に示す．Altera 社の Cyclone III FPGA Starter 開発 Board (199 米ドル，2010 年 8 月時点，アルテラ社国内代理店より入手) と terasic 社の Highspeed AD/DA Card (ADA-HSMC) （219 米ドル，同時点）を用いている．ブレッドボード上の回路は高周波

増幅回路と音声出力用の回路である．図 15.2 に FPGA ラジオの回路図を示す．高周波増幅回路は図 4.3.1 の FET(2SK241) とバイポーラトランジスタ (2SC1815) を用いた 2 段増幅回路である．この増幅回路の出力を直接 AD/DA 変換カードの AD 変換器入力端子につなぎ，DA 変換器の出力端子を可変抵抗器を通してイヤフォンプラグへとつないである．立体配線図を図 15.3 に示す．

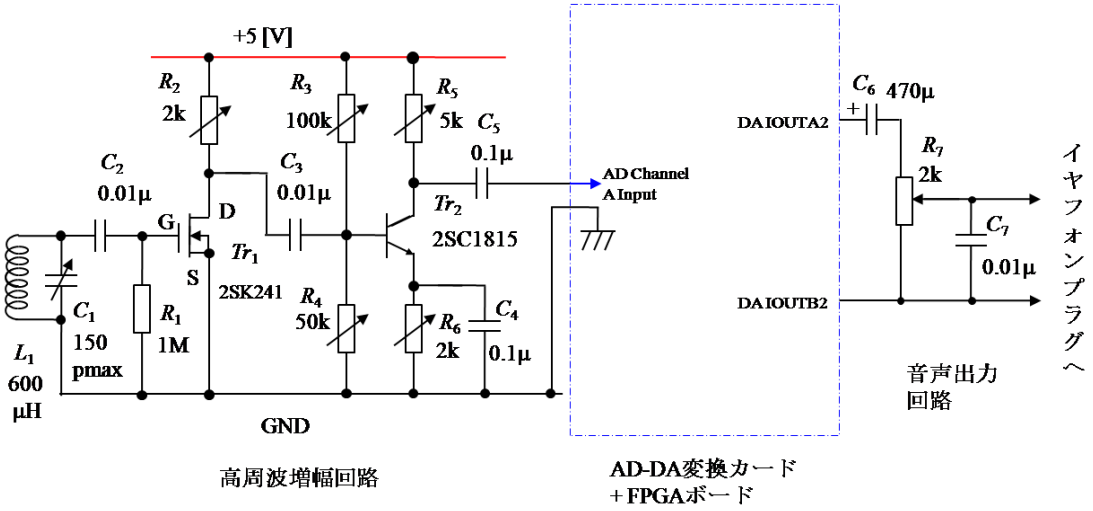


図 15.2: FPGA ラジオの回路図

Altera Cyclone III Starter Board の上面写真を図 15.4 に示す．このボードは Cyclone III EP3C25F324 FPGA を搭載している．クロックは 50MHz である．USB コネクタを持ち，USB ケーブルを介してパソコンから FPGA にコンフィグレーションファイルをダウンロードできる．この操作をする (FPGA に書き込む）だけで，FPGA を動作させることができる．スタータ開発キットを購入し，Altera 社の Web ページから無償の Qurtsu II ソフトウェア ウェブエディションをパソコンにダウンロード／インストールすれば，ただちに FPGA のプログラミングと実験ができる．

terasic の Highspeed AD/DA Card の写真を図 15.5 に示す．このカードは 14 ビット，65MSPS(Mega Sample Per Second) の 2 チャネル AD 変換器と 14 ビット，125MSPS の 2 チャネル DA 変換器を搭載している．65MSPS は中波帯 AM 放送（アジア・オセアニア地域では 531〜1602 [kHz]）の信号処理には十分な（十分過ぎるが，10MSPS 程度の AD-DA 変換器を搭載したカードを見つけることができなかった.）変換速度である．このカードは HSMC(High Speed Mezzanine Connector) コネクタにより FPGA ボードと合体する構造である．

カードの改修を一箇所必要とした．DA 変換器のチャネル B の出力トランス T6 を取り外した．図 15.5 中の楕円で囲った箇所である．理由は，このトランスは数 100 [kHz] より

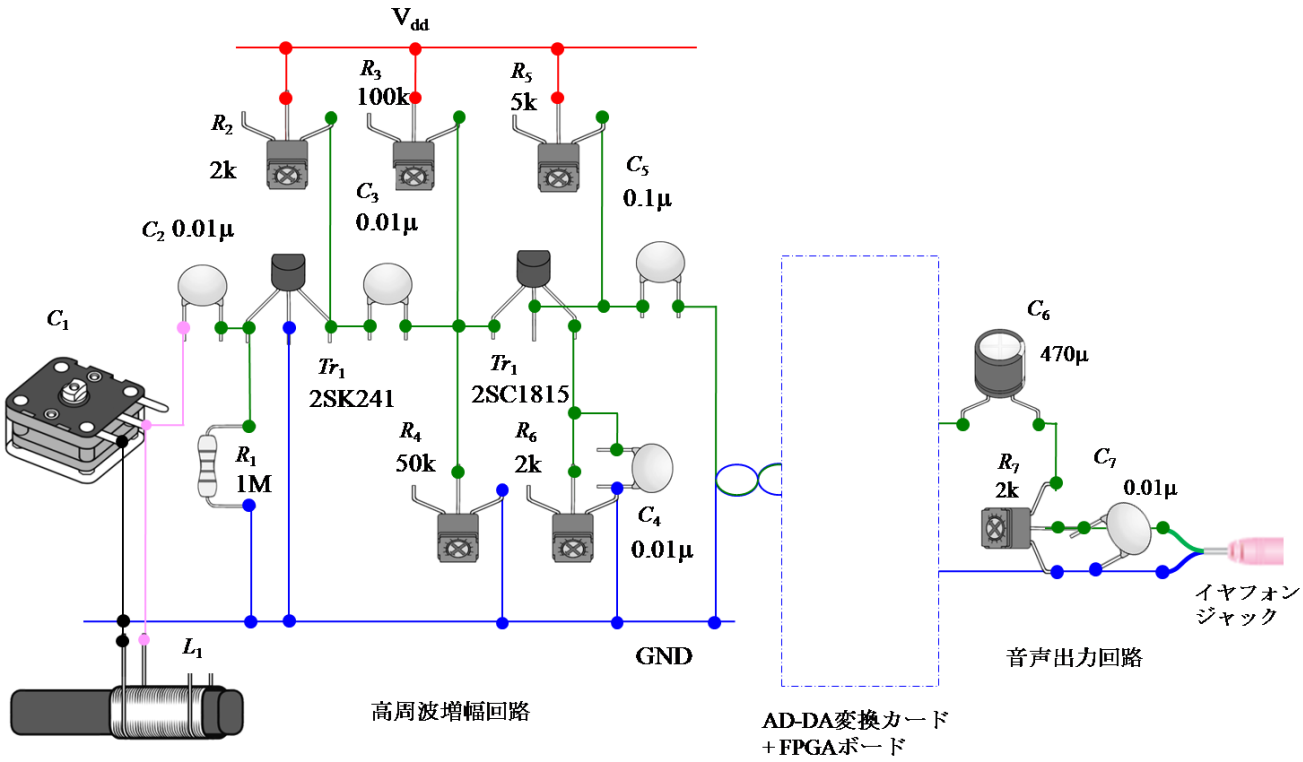


図 15.3: FPGA ラジオの立体配線図

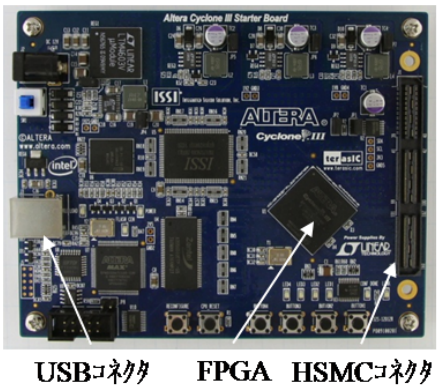


図 15.4: アルテラ Cyclone III スタータボード

低い周波数の信号を通さないからである．改修箇所の拡大写真を図 15.6 に示す．トランス T6 を取り外し，DA 変換器の出力端子 IOUTA2, IOUTB2 にそれぞれ信号線をハンダづけした．（中波帯ラジオ用には高すぎる機能を持つ AD-DA 変換カードのこのような使い方は，ジェット機をプロペラ機に仕様ダウンした感があるが，0〜25 [MHz]（FPGA ボード搭載のクロック 50[MHz] の 1/2 まで（ただし，PLL 機能を使えば DA 変換器をクロック 120[MHz] で駆動できる．すなわち 0〜60 [MHz]）) の広い帯域の信号を出力できるので，以降各種実験を試みる気にさせてくれる．２万円の出費は惜しくない.）また，

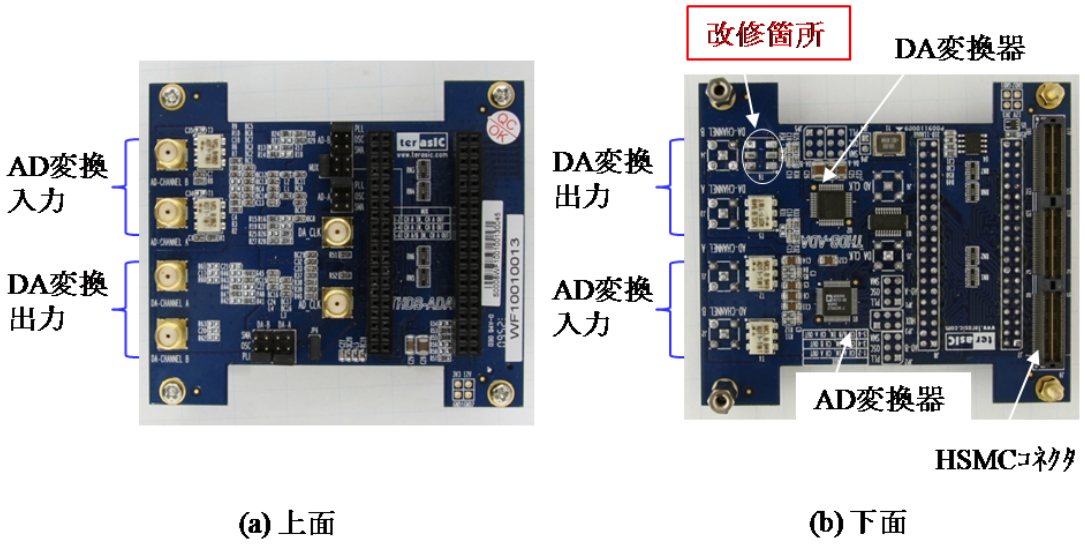


図 15.5: DA-AD 変換カード

AD チャネル A の入力端子にも信号線をハンダづけした．以上でハードウェアに関しては準備完了である．各種方式のラジオはソフトウェアにより記述することで実現できる．

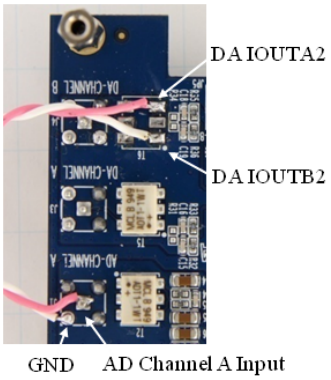
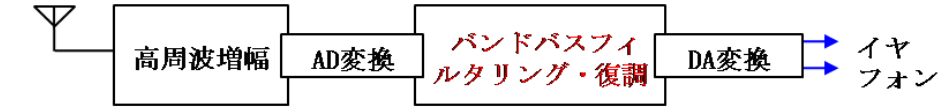


図 15.6: DA-AD 変換カード（改修の様子）

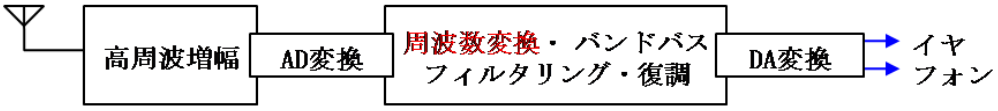
15.2 ストレート方式ソフトウェアラジオ

FPGA のプログラミングに関しては文献 [13][14] に詳しい．文献 [13] で Altera 社の Cyclone FPGA の基本を理解し，文献 [14] で AD, DA 変換器の使い方を知り，Quartus II ウェブエディションを立ち上げるときに最初に現れる Interactive learning を利用して bdf(block design file)，bsf(block symbol file) の作り方を習得すれば，本章の準備には十分である．

図 15.7 は本章でこれから製作するソフトウェアラジオのブロック図を示す．同図 (a) は



(a) ストレート方式ソフトウェアラジオ



(b) スーパーヘテロダイン方式ソフトウェアラジオ

図 15.7: 本章で製作するソフトウェアラジオのブロック図

ストレート方式ソフトウェアラジオであり，(b) はスーパーヘテロダイン方式ソフトウェアラジオである．両者の違いは周波数変換の有無である．ストレート方式では復調前に周波数変換がない．スーパーヘテロダイン方式では放送信号を中間周波数に変換した後に復調が行われている．

本節では図 15.7(a) のストレート方式ソフトウェアラジオを製作する．

15.2.1　AD-DA 変換のテスト

まず，AD 変換器により放送信号を FPGA に取り込み，DA 変換器によりその信号を出力することを試してみる．図 15.8 に製作したテスト装置のブロック図を示す．AD/DA 変換カードの AD Channel A Input にオシレータの信号を入力し，DA IOUTA2 と DA IOUTB2 の各端子と GND 間の波形をオシロスコープにより観測する図 15.9 に AD-DA 変換テストプログラムの bdf（ブロックデザインファイル）を示す．bdf は bsf（ブロックシンボルファイル）とその間の配線情報およびピン配置リストからなる．ただし，ピン配置リストは省略してある．プログラム間の信号の流れを視覚的に把握できるため，プログラムの理解に適している．

図 15.8: AD-DA 変換テスト装置のブロック図

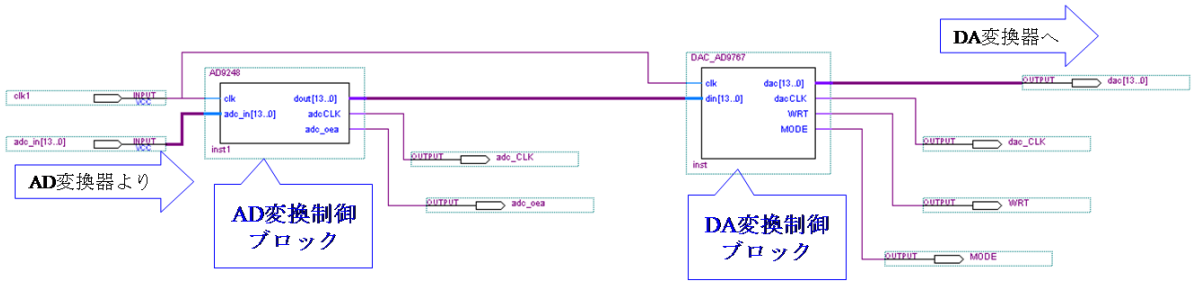


図 15.9: AD-DA 変換テストプログラムの bdf（ブロックデザインファイル）

図 15.10 に AD 変換器の制御プログラムの bsf（ブロックシンボルファイル）を示す．bsf は図中の AD9248 と記された破線のブロックである．bsf のファイル名は AD9248.bsf である．図にはこのブロックの入出力の配線とピン番号も併せて示してある．Quartus II 上でこのブロックを左ダブルクリックすると図 15.11 のプログラムが表示される．VHDL(VHSIC Hardware Description Language) のプログラムリストである．プログラムの詳細は各行のコメントを参照されたい．AD 変換器の出力ディジタル信号は符号無し整数であるため，下から 4 行目の命令により符号付き整数に変換して次ブロックへと出力している．これは FPGA 内の処理を符号付き整数で一貫して行えるようにするためである．

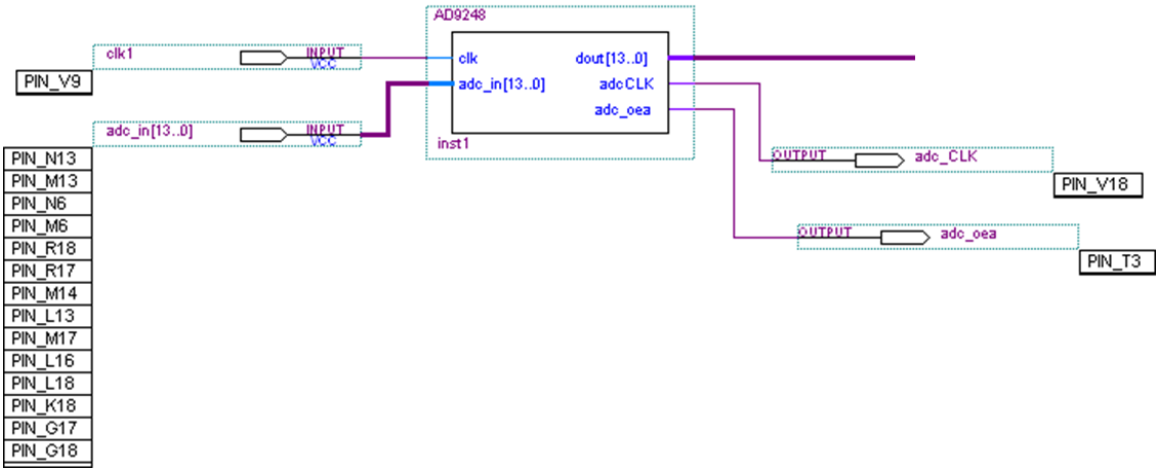


図 15.10: AD 変換器の制御プログラムの bsf(AD9248.bsf)

図 15.10 の bsf へのピン配置は，AD-DA 変換カードと Cyclone III スタータボードのデータシートより，AD 変換器と FPGA の配線を読み取ることで行う．図 15.12 に関係する部分をまとめて示す．例えば，AD 変換器チャネル A の最下位ビット D0_A は FPGA の N13 ピンにつながっている．そこで，Quartus II の Assignments → pin planner → (Location 欄にピン番号入力）により，adc_in[0] に N13 を割り当てる．AD 変換制御ブロックの入出力ポート clk, adc_in, adcCLK, adc_oea にピンを割り当てた結果が図 15.10 のピン番号

```
library ieee;                          --IEEEライブラリはVHDL記述において必須
use ieee.std_logic_1164.all;           --データタイプのパッケージ．これも必須
use ieee.std_logic_arith.all;          --算術演算パッケージ．本プログラムの足し算に必要
use ieee.std_logic_unsigned.all;       --本プログラムのデータは符号無し整数

entity AD9248 is
    port (
        clk         : in  std_logic;                    --このブロックへのクロック入力
        adc_in      : in  std_logic_vector(13 downto 0); --放送信号入力．AD変換器より，14ビット
        dout        : out std_logic_vector(13 downto 0); --放送信号の次ブロックへ出力，14ビット
        adcCLK      : out std_logic;                    --AD変換器用クロック
        adc_oea     : out  std_logic);                  --AD変換器オン指令
end AD9248;

architecture ad9248 of AD9248 is
    signal const        : std_logic_vector (13 downto 0); -- 定数値（符号無し整数→符号付き整数変換用
                                                          --14ビット

begin
    adcCLK <= clk;                     --AD変換器のクロックをブロックの入力クロックと同じにする．
    adc_oea <= '0';                    --AD変換器オン
    const <= "10000000000000";         --14ビットの符号無し整数を符号付き整数に変換する定数

process (clk)
    begin
        if clk'event and clk = '0' then    --クロック立ち下がり時に以下を実行
            dout <= adc_in + const;        --ADコンバータからの信号を符号付き整数に変換して出力
        end if;
end process;

end ad9248;
```

図 15.11: AD 変換器の制御プログラム

リストである．

AD 変換器の動作周波数は 50 [MHz] の設定である．

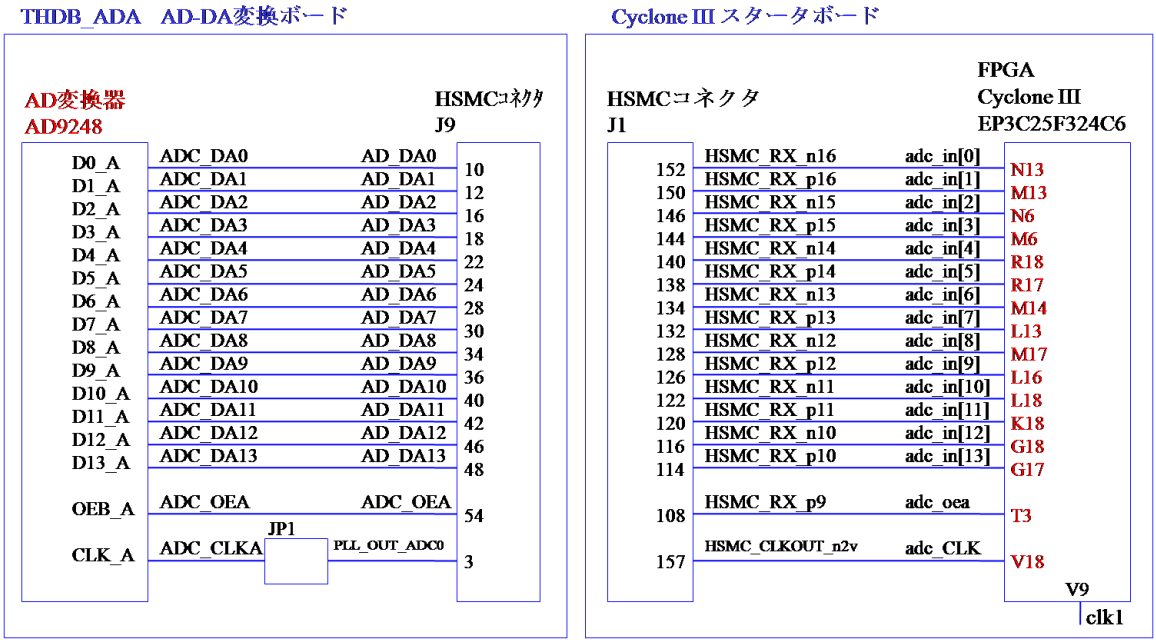


図 15.12: AD 変換器と FPGA 間の配線図

図 15.13 に DA 変換器の制御プログラムの bsf を示す．ファイル名は DAC_AD9767.bsf

である．図 15.14 はプログラムリストである．DA 変換器の制御プログラムへの入力信号 din は符号付き整数であるため，下から 4 行目の命令により符号無し整数に変換して DA 変換器へと出力している．

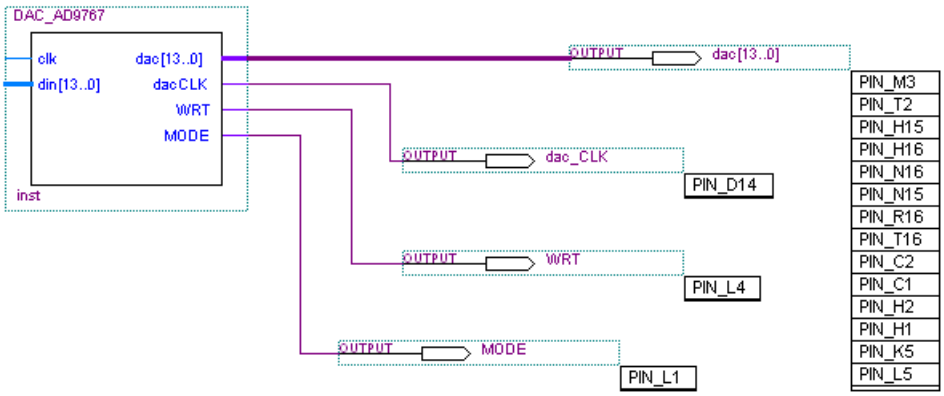


図 15.13: DA 変換器の制御プログラムの bsf(DAC_AD767.bsf)

```
library ieee;                          --IEEEライブラリはVHDL記述において必須
use ieee.std_logic_1164.all;           --データタイプのパッケージ．これも必須
use ieee.std_logic_arith.all;          --算術演算パッケージ．本プログラムの足し算に必要
use ieee.std_logic_unsigned.all;       --本プログラムのデータは符号無し整数

entity DAC_AD9767 is
    port (
        clk         : in  std_logic;                    --このブロックへのクロック入力
        din         : in  std_logic_vector(13 downto 0); --信号入力，前ブロックより，14ビット
        dac         : out std_logic_vector(13 downto 0); --DA変換器への出力信号，14ビット
        dacCLK      : out std_logic;                    --DA変換器用クロック
        WRT         : out  std_logic);                  --DA変換器書き込みオン指令
        MODE        : out  std_logic);                  --DA変換器の出力ポートのモード設定
end DAC_AD9767;

architecture da9767 of DAC_AD9767 is
    signal const    : std_logic_vector (13 downto 0); -- 定数値（符号付き整数→符号無し整数変換用）
                                                      -- 14ビット

begin
    dacCLK <= clk;                     --DA変換器のクロックをブロックの入力クロックと同じにする．
    WRT <= clk;                        --DA変換器の書き込みタイミング，クロックと同じでよい．
    MODE <= '1';                       --2つのポートを独立に使用する設定．'0'：interleaveモード．
    const <= "10000000000000";         -- 14ビットの符号付き整数を符号無し整数に変換する定数

process (clk)
    begin
        if clk'event and clk = '0' then    --クロック立ち下がり時に以下を実行
            dac <= din + const;            --符号無し整数に変換してDAコンバータに出力
        end if;
end process;

end da9767;
```

図 15.14: DA 変換器の制御プログラム

図 15.15 に DA 変換器と FPGA 間の配線を示す．図 15.13 のピン配置はこの配線に基づいている．

図 15.16 は実験波形例を示す．同図 (a) が AD 変換カードの入力信号である．周波数は

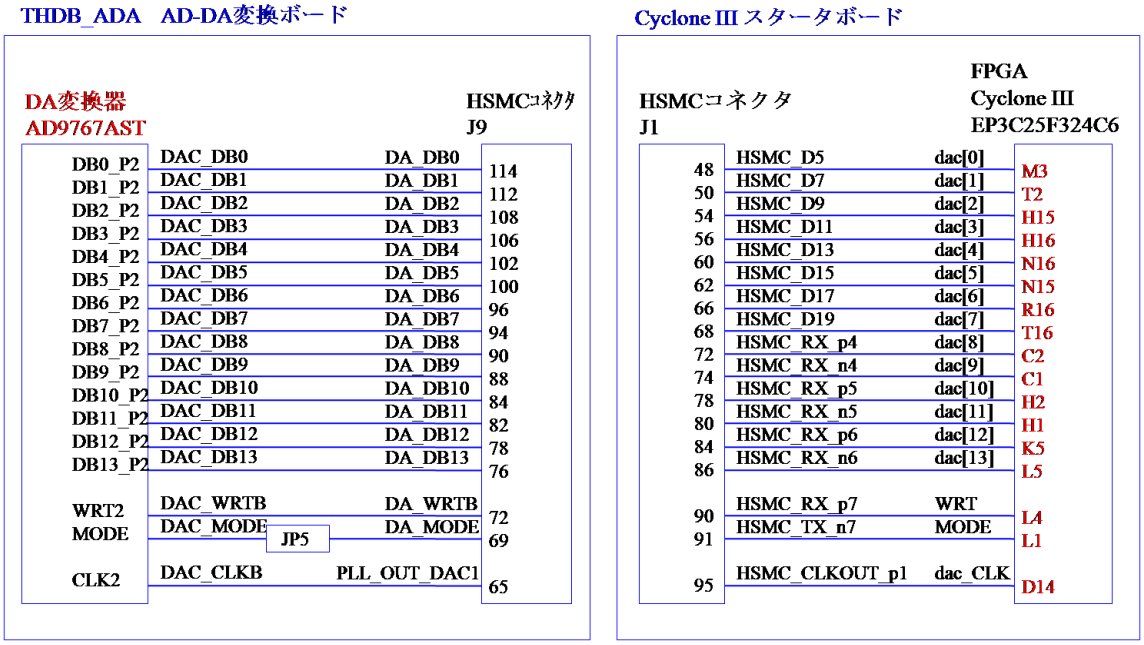


図 15.15: FPGA と DA 変換器間の配線図

700 [kHz] である．同図 (b), (c) が DA 変換器の出力電圧波形である．それぞれ DA 変換器の IOUTA2-GND 間電圧，IOUTB2-GND 間電圧である．両電圧は 0.5 [V] の線を中心として線対称の関係にある．従って，IOUTA2-IOUTB2 間電圧は，IOUTA2-GND 間電圧から直流成分を除き，交流成分を 2 倍とした電圧となる．AD 変換器，DA 変換器ともにサンプリング周波数が 50 [MHz] であることにより，サンプリングによる量子化の影響は見えない．

図 15.16 の出力電圧には入力電圧に対して位相遅れが見られる．時間軸を拡大した図を図 15.17 に示す．入出力電圧のピーク値の間の時間は 0.23 [μs] であった．これは AD 変換器および AD/DA 変換制御ブロックおよび DA 変換器での処理遅れによる．AD 変換器のサンプリング周波数は 50 [MHz]（0.02 [μs]/クロック）である．AD 変換器はサンプルしたアナログ電圧をディジタル値に変換するのに 7 クロック (7 [clock]/50 [MHz] = 0.14[μs]) を要する．AD 変換器ではパイプライン処理がなされているため，毎クロックでサンプルがなされ，毎クロックで 7 クロック前のサンプル値の変換結果が出力される．さらに，AD, DA 変換制御ブロック内の処理にそれぞれ 1 クロックずつを要し，DA 変換器での遅れが 2 クロック弱ある．合計 11 クロック弱の遅れとなる．さらに，AD 変換はサンプリング毎であるので，$-8 \leq t < -7$ クロックの間の時刻 t における変化が変換出力に現れる．そこで，図 15.16 の波形例では約 11.5 クロック分 (=0.23 [μs]) の位相遅れとなって現れている．

15.2.2 ストレート方式ソフトウェアラジオ（バンドパスフィルタ無し）

第 12 章の図 12.2 のストレート方式ソフトウェアラジオと同様のラジオを製作する．図 12.2 のラジオとの違いは AD 変換器のサンプリング周波数が 50 [MHz] と高い点である．図 15.8 のブロックに新たに復調ブロックを導入する． 図 15.18 にブロック図を示す．図 15.19 は製作した bdf である．復調ブロック Detector.bsf が挿入されている．

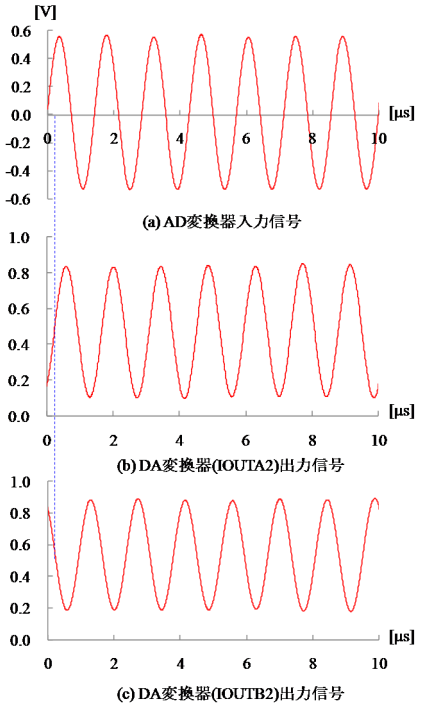


図 15.16: AD-DA 変換実験波形

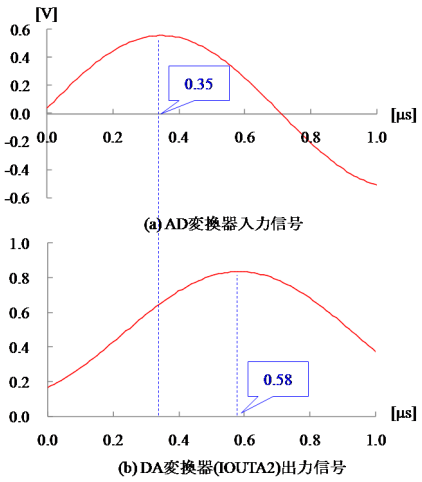


図 15.17: AD-DA 変換実験波形（時間軸拡大）

図 15.18: ストレート方式ソフトウェアラジオ（バンドパスフィルタ無し）のブロック図

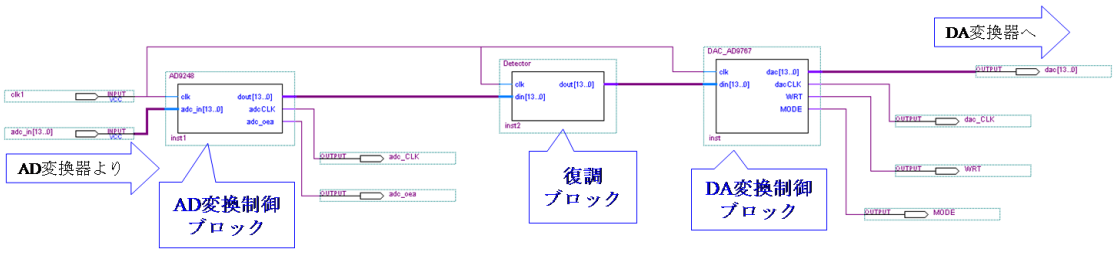


図 15.19: ストレート方式ソフトウェアラジオ（バンドパスフィルタ無し）の bdf

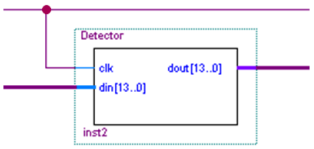


図 15.20: 復調器プログラムの bsf(Detector.bsf)

```
library ieee;                          --IEEEライブラリはVHDL記述において必須
use ieee.std_logic_1164.all;           --データタイプのパッケージ．これも必須
use ieee.std_logic_arith.all;          --算術演算パッケージ．本プログラムの正負反転に必要
use ieee.std_logic_signed.all;         --本プログラムのデータは符号付き整数

ENTITY Detector IS
    PORT
    (
        clk         : IN STD_LOGIC ;        --コンパイラは大文字･小文字を区別しない
        din         : IN STD_LOGIC_VECTOR (13 DOWNTO 0);
        dout        : OUT STD_LOGIC_VECTOR (13 DOWNTO 0)
        );
END Detector;

ARCHITECTURE Det OF Detector IS
    SIGNAL  temp_data    : STD_LOGIC_VECTOR (13 DOWNTO 0);--データの一次格納レジスタの設定

Begin

process (clk)
    begin
        if clk'event and clk = '1' then          --クロック立ち上がり時に以下を実行
            if din > 0 then                      --データの絶対値を求める計算
                temp_data <= din;
            else
                temp_data <= - din;
            end if;
            dout <= temp_data;
        end if;
end process;

end Det;
```

図 15.21: 復調器プログラム

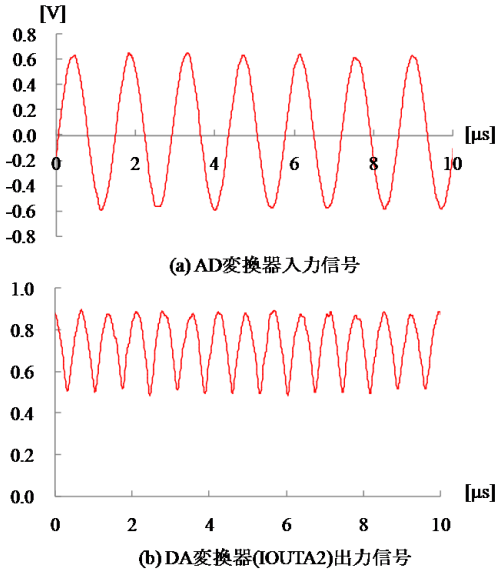


図 15.22: 復調実験波形

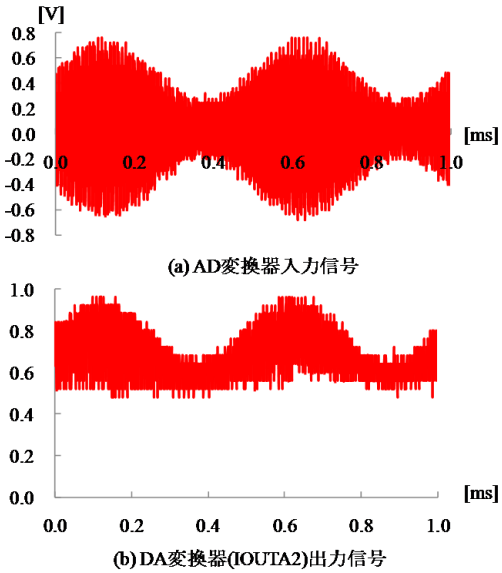


図 15.23: 被変調波の復調実験波形

図 15.20 は復調器プログラムの bsf である．この VHDL ファイルを図 15.21 に示す．復調器は，負の入力信号を符号反転し，絶対値を求める演算を行っている．

AD-DA 変換カードの AD Channel A Input にオシレータの信号を入力し，DA_IOUTA2 と GND 間の波形をオシロスコープにより観測した結果を図 15.22 に示す．DA 変換器は，14 ビットの符号無し整数 00 0000 0000 0000～11 1111 1111 1111 を 0～1 [V] の電圧に変換する．符号付き整数の 0 [V] は符号無し整数の 10 0000 0000 0000 に対応し，この値は DA 変換器により 0.5 [V] に変換される．復調ブロックは符号付き整数の絶対値をとる演

算を行っている．DA 変換器の出力は 0.5 [V] より低い電圧を 0.5 [V] の線を中心に反転した波形となっている．全波整流である．

これだけでラジオを聞くことができる．図 15.23 は被変調波の復調実験波形を示す．キャリアの周波数が 700 [kHz] で変調波の周波数が 2 [kHz] である．AD 変換器の入力電圧は AM 信号であり，DA 変換器の出力電圧が AM 復調の波形である．この出力電圧を図 15.2 の音声出力回路を通してイヤフォンで聞けば 2 [kHz] の音が聞こえてくる．さらには図 15.2 の高周波増幅回路の出力を AD 変換器の入力とすることで，ラジオ放送を聴くことができる．

（ここまでを達成しただけでも，FPGA によりまずはラジオ放送が聴けたという喜びに浸ることができます．… 筆者は浸りました … 少年の頃のラジオと文字通り隔世であることと，また，最新技術を体感できたことの両方でした．）

放送局の選択度は図 4.3.1 のストレートラジオと全く変わらない．非線形要素であるダイオードによる検波がないので，音質はストレートラジオより良い．ノイズがあるが，電源を AC アダプタからではなく，電池からとり，USB ケーブルを外すことで低減できる．

15.2.3　ストレート方式ソフトウェアラジオ（バンドパスフィルタ有り）

FPGA を用いることのメリットの一つに通過帯域外の遮断特性の良いフィルタを実装できることがある．Cyclone III スタータボード搭載の FPGA の場合，IP(Intellectual Property) コアを用いることで， 300 タップを 1 クロックで処理できる FIR バンドパスフィルタを実装できる．8 クロックの処理を要する FIR バンドパスフィルタであれば 1000 タップのものも実装できる (図 15.27 参照)．1000 タップのバンドパスフィルタをストレートラジオに導入すれば，図 11.1.7 の破線の理想特性に近い周波数特性を持つラジオを作ることができる．スーパーヘテロダインラジオのような周波数変換を必要とせずに，スーパーヘテロダインラジオを上回る選択度を持つラジオである．

図 15.24 にバンドパスフィルタを持つストレート方式ソフトウェアラジオのブロック図を示す．図 15.25 は製作した bdf(block design file) である．新たに Bandpass_Filter.bsf を作成し，この bsf(block symbol file) を使えるようにするために，bit_num_down.bsf, const_for_serial_filter.bsf, clk_div.bsf を作成している．Bandpass_Filter.bsf は IP(Intellectual Property) コアを用いて作成している．この IP コアは有償であるが，ライセンスを購入しなくても，CycloneIII スタータボードの USB ケーブルをパソコンにつないでいる限りは，時間無制限で利用できる．USB ケーブルを外すと 1 時間ほどで自動的に停止する．

図 15.26 はバンドパスフィルタプログラムと FIR フィルタ設定プログラムの bsf である．ファイル名は Bandpass_Filter.bsf, const_for_serial_filter.bsf である．バンドパスフィ

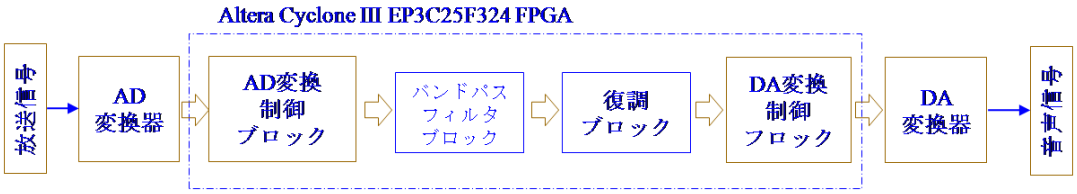


図 15.24: ストレート方式ソフトウェアラジオ（バンドパスフィルタ有り）のブロック図

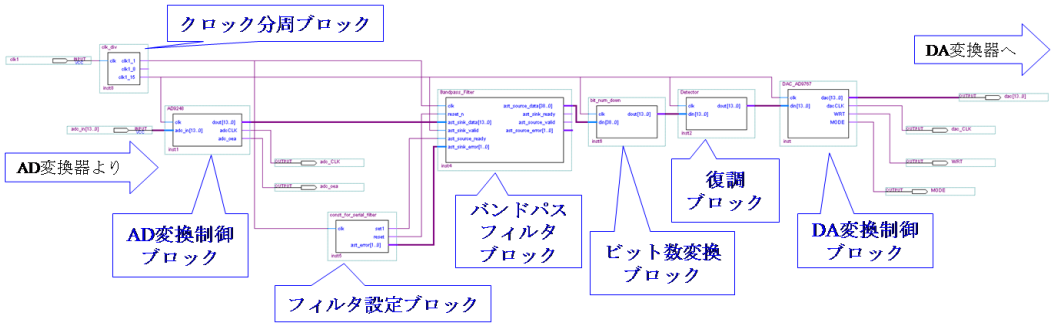


図 15.25: ストレート方式ソフトウェアラジオ（バンドパスフィルタ有り）の bdf

ルタの作成画面を図 15.27 に示す．Quartus II の Tools → MegaWizard Plug-in Manager → Create a new custom magafuntion variation → DSP → Filter → FIR Compiler → (file name 入力）→ Parameterize と進むことでこの作成画面を開くことができる．Fully Serial Filter が最も多くの処理クロックを要し（画面右下に 1 5 クロックごとにデータの入出力を 1 回実行とある），Fully Parallel Filter が最も少ないクロック（ 1 クロック）で処理できる．逆に Fully Serial Filter は使用するロジックセル数が最も少なく，Fully Parallel Filter が最も多くのロジックセルを使用する．Cyclone III スタータボード搭載の FPGA EP3C25F324 の場合，前者は FIR フィルタのタップ数を 1000 程度までとることができ，後者は 300 程度までである．このタップ数の違いはフィルタのしゃ断性能を左右する．スーパーヘテロダインラジオのしゃ断性能を超えることを目指し，Fully Serial Filter を採用し，タップ数 1000 の FIR フィルタを作成する．

入力データのビット数を 14 とし，フィルタ係数のビット数を 18 とすると，画面の設定の場合，出力のビット数は 39 と自動設定される．Edit Coefficient Set ボタンをクリックすると，図 15.28 が現れる．サンプルレートは，15 クロックで 1 サンプルとするために，50 [MSPS]/15 = 3.33 [MSPS] とし，タップ数 (Coefficients)=1000, 低域しゃ断周波数 (Cutoff Freq 1) = 690 [kHz], 高域しゃ断周波数 (Cutoff Freq 2) = 710 [kHz] とした例である．Window Type は Rectangular, Hamming, Hanning, Blackman の 4 種類から選ぶことができる．

IP コアの FIR フィルタの FIR Compiler User Guide(FIR フィルタ作成画面を開く Pa-

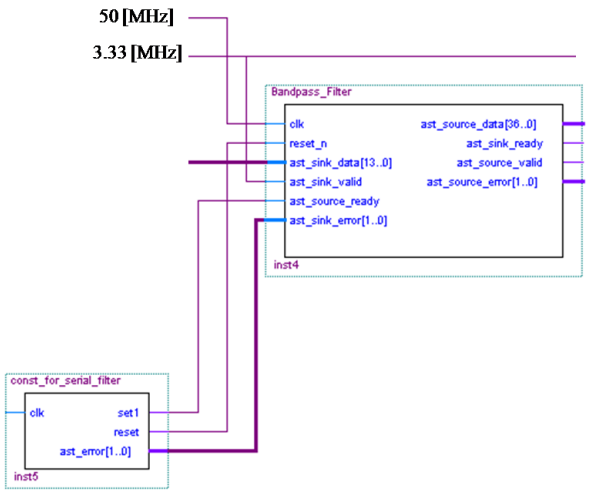


図　15.26:　バンドパスフィルタプログラムと FIR フィルタ設定プログラムの bsf Bandpass_Filter.bsf, const_for_serial_filter.bsf

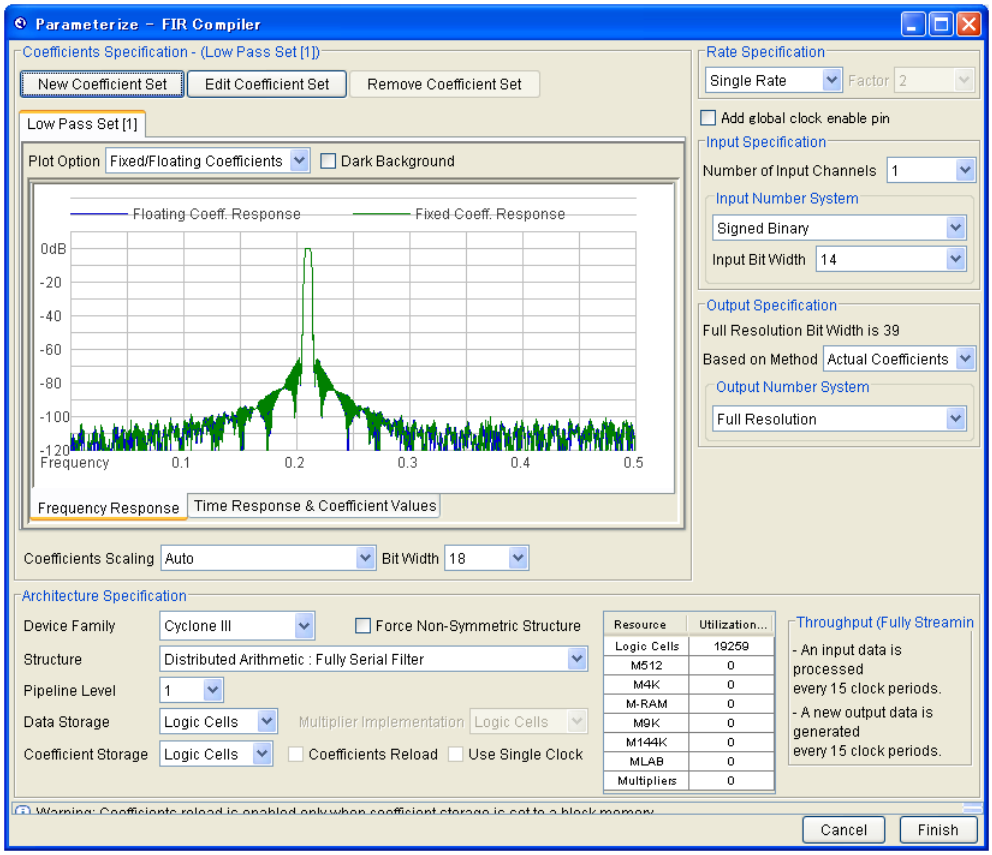


図 15.27: FIR コンパイラ画面

rameterize ボタンの上に Documentation ボタンがある．ここをクリック.）より，バンドパスフィルタの起動時に 1 クロック以上 reset_n に 0 を入力する必要がある．図 15.29 は

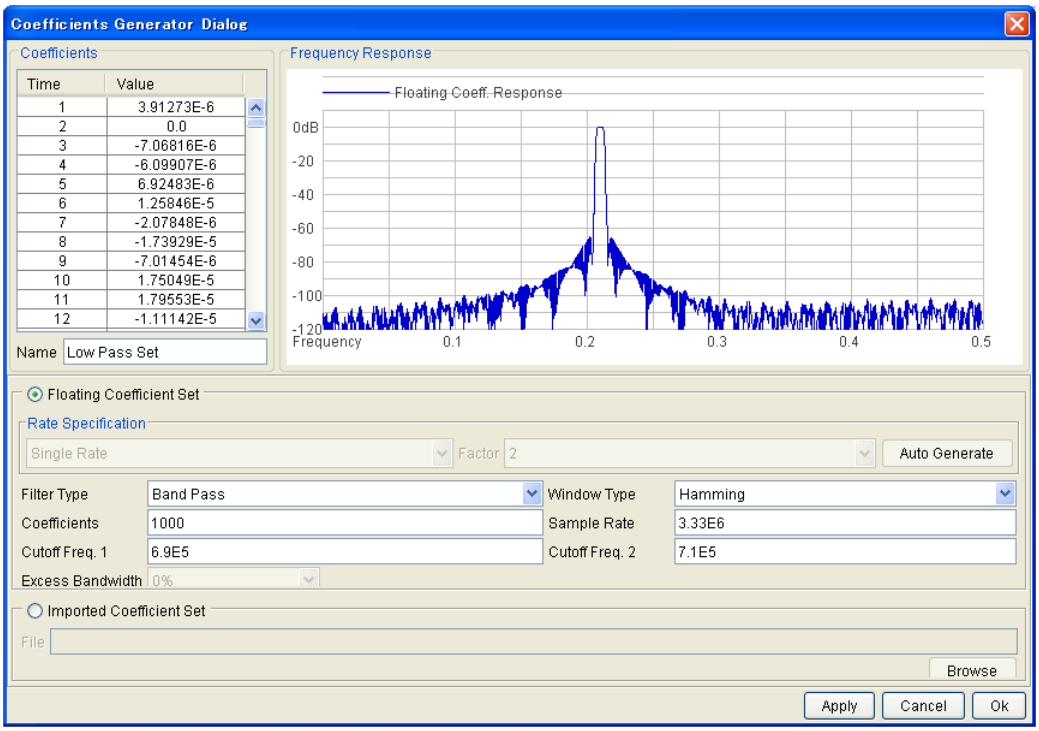


図 15.28: Edit Coefficient Set 画面

const_for_serial_filter の VHDL ファイルである．起動時に 3 クロック間 reset_n に 0 を出力する設定である．

15 クロックの間に 1 サンプリングを実行するために，50 [MHz] のクロックを 1/15 分周して 3.33 [MHz] のクロックを得る必要がある．Bandpass_Filter.bsf の clk を 50 [MHz] クロックに接続し，ast_sink_valid を 3.33 [MHz] クロックに接続することで，バンドパスフィルタを 50 [MHz] で駆動し，データのサンプリングを 3.33 [MHz] とすることができる．図 15.30 は分周クロックを生成する bsf である．図 15.31 にクロック分周プログラムを示す．div_counter $<=$ div_counter $+ 1$; により，50 [MHz] クロックによりカウントアップする．その次の if 文においては div_counter > 8 で 1 を出力し，それ以外で 0 を出力している．また，div_counter $= 14$ で div_counter を 0 にリセットしている．

ここで，FPGA ではカウントアップ文と if 文が同時実行される点に注意すれば，div_counter が 14 であることの判定と，div_counter の 15 へのカウントアップは同じクロックでなされる．したがって，このプログラムでは 15 クロック周期で div_counter が 0 にリセットされ，50 [MHz] のクロックの 1/15 分周がなされる．

クロック分周プログラムには後の利用のために 1/8 分周クロックの生成プログラムも記されている．考え方は 1/15 分周の場合と同じである．

1/15 分周により，AD 変換のサンプリング周波数は 3.33 [MHz] に低減される．サンプル

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity const_for_serial_filter is
    port (
        clk             : in std_logic;
        set1            : out std_logic;
        reset           : out std_logic;
        ast_error       : out std_logic_vector(1 downto 0));
end const_for_serial_filter;

architecture cfsf of const_for_serial_filter is
    signal start_idle : std_logic_vector(3 downto 0) := (others => '0');

begin

    set1 <= '1';                                    -- ast_source_readyに対して 1 を出力
    ast_error <= "00";                              -- ast_sink_errorに対して00を出力

process (clk)
    begin
        if clk'event and clk = '1' then             --クロック立ち上がり時に以下を実行
            if start_idle < 3 then                  --バンドパスフィルタの初期化のため
                reset <= '0';                       --最初の3クロック間はreset_nに対して0を出力
                start_idle <= start_idle + 1;
            else
                reset <= '1';
            end if;
        end if;
end process;

end cfsf;
```

図 15.29: FIR フィルタ設定プログラム

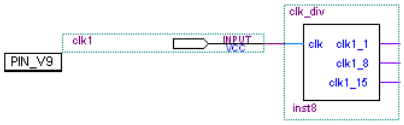


図 15.30: クロック分周プログラムの bsf(clk div.bsf)

波形観測の実験を行った．図 15.32 にクロック分周テスト bdf を示す．AD, DA 変換器のクロックを 3.33 [MHz] とし，AD 変換結果をそのまま DA 変換して，入出力波形を観測した．図 15.33 に実験結果を示す．図は 700 [kHz] の信号を AD 変換器に入力したときの DA 変換器の出力波形である．信号周波数がナイキスト周波数 (3.33 [MHz]/2 = 1.67[MHz]) の半分程度であるのでサンプル波形は正弦波とは大きく異なって見える．図 15.34 は 2 [kHz] の音声信号を含む AM 変調波を AD 変換器に入力したときの DA 変換器の出力波形である，オシロスコープの掃引を遅らせて，2 [KHz] の音声信号が見えるようにしてある．サンプリング周波数を 3.33 [MHz] としても AM 変調波の取り込みには支障がない．

バンドパスフィルタの出力値は 38 ビットの符号付き整数値である．これを 14 ビットの符号付き整数値に変換する必要がある．図 15.35 にビット数低減プログラムの bsf(bit_num_down.bsf) を示す．また，図 15.36 にビット数低減プログラムを示す．38 ビットの上位 2 ビットを捨て，その下の 14 ビットを出力するプログラムである．上位 2 ビットを捨て，その下のビットをとることで，$2^2 = 4$ 倍に増幅する効果がある．

```
library ieee;                          --IEEEライブラリはVHDL記述において必須
use ieee.std_logic_1164.all;           --データタイプのパッケージ．これも必須
use ieee.std_logic_arith.all;          --算術演算パッケージ．本プログラムの足し算に必要
use ieee.std_logic_unsigned.all;       --本プログラムのデータは符号無し整数

entity clk_div is
    port (
        clk          : in  std_logic;                        --このブロックへのクロック入力
        clk1_1       : in  std_logic;                        --1/1分周クロック出力
        clk1_8       : in  std_logic;                        --1/8分周クロック出力
        din1_15      : in  std_logic_vector(13 downto 0);    --1/15分周クロック出力
end clk_div;

architecture clkdiv of clk_div is
    signal div_counter : std_logic_vector(4 downto 0) := (others => '0'); --5ビットのカウンタ用レジスタ設定
    signal div_counter2 : std_logic_vector(4 downto 0) := (others => '0'); --

begin
    clk1_1 <= clk;                     --１／１分周クロック生成

process (clk)
    begin
        if clk'event and clk = '1' then          --クロック立ち上がり時に以下を実行
            div_counter <= div_counter + 1;      --カウントアップ
            if div_counter > "01000" then        --8を超えたら以下を実行
                clk1_15 <= '1';                  --1/15分周クロックに1を出力
                if div_counter = "01110" then    --14となったらカウンタ用レジスタを0にリセット
                    div_counter <= "00000";      --これで1/15分周を実現．なぜなら　同時並列実行なので
                end if;                          --if文が14を判定して，div_counter=0とする直前には．
            else                                 --div_counter=15となっている．
                clk1_15 <= '0';
            end if;

            div_counter2 <= div_counter2 + 1;    --カウントアップ
            if div_counter2 > "00100" then
                    (1／8分周クロック生成・・省略)
            end if;
        end if;
end process;

end clkdiv;
```

図 15.31: クロック分周プログラム

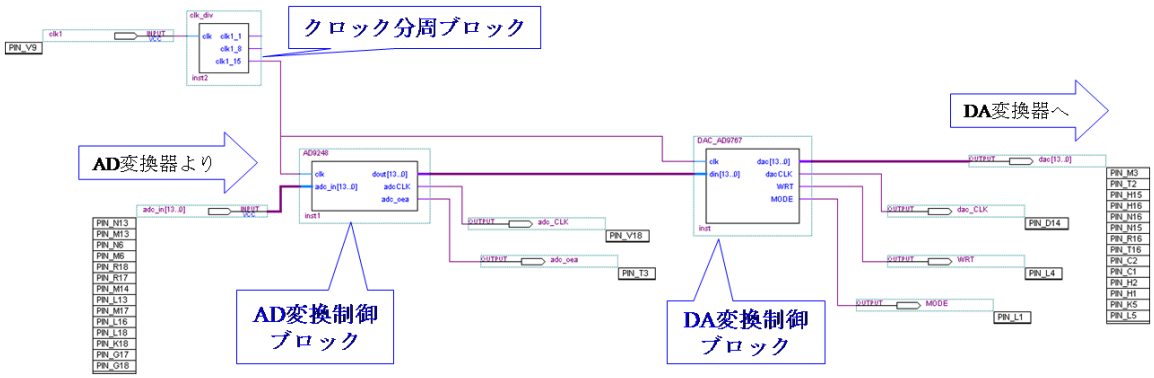


図 15.32: クロック分周テスト bdf

バンドパスフィルタの入出力の電圧比を計測した．バンドパスフィルタの入力値 V_{in} と bit_num_down.bsf の出力値 V_{out} の比 $20log_{10}\left|\frac{V_{out}}{V_{in}}\right|$ の計測結果を図 15.37 に示す．良いしゃ断特性が実現されている．

以上で，バンドパスフィルタ付きストレートラジオの準備が整った．バンドパスフィルタの低域しゃ断周波数 (Cutoff Freq 1) = 719 [kHz], 高域しゃ断周波数 (Cutoff Freq 2)

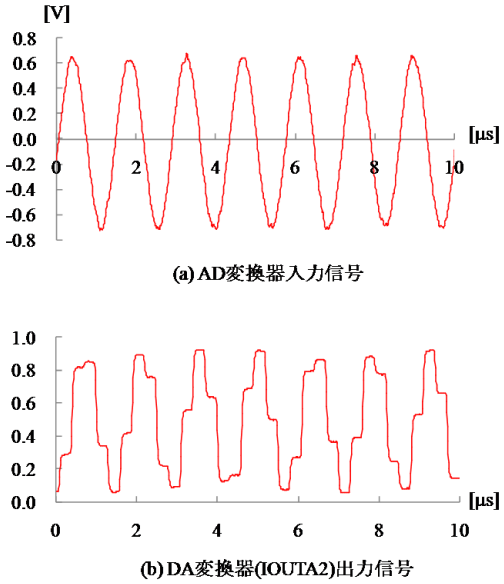


図 15.33: クロック分周実験波形（変調無し）

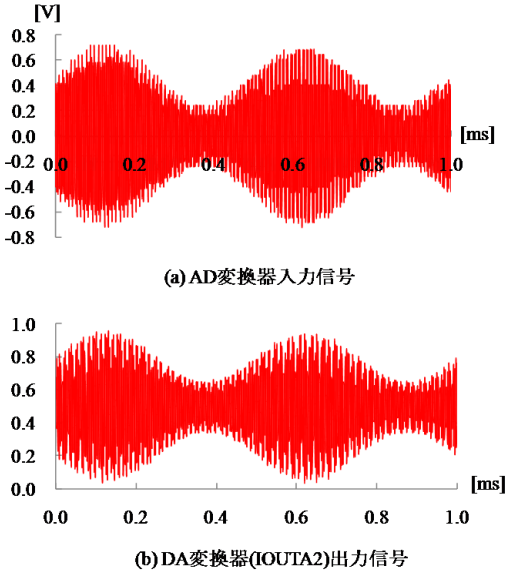


図 15.34: クロック分周実験波形（変調有り）

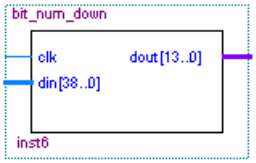


図 15.35: ビット数低減プログラムの bsf（bit num down.bsf

= 739 [kHz] として，図 15.1 の高周波増幅回路の出力を AD 変換器に入力し，イヤフォンをつなぐことで，NHK 名古屋 (729 [kHz]) を聴くことができた．電源を電池に切り替え，

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity bit_num_down is
      port (
            clk                  : in  std_logic;
            din                  : in  std_logic_vector(38 downto 0);
            dout                 : out std_logic_vector(13 downto 0));

end bit_num_down;

architecture rtl of bit_num_down is

begin
      process (clk)
            begin
                  if clk'event and clk ='1' then
                        dout(13 downto 0) <= din(36 downto 23);        --入力データの上位2ビット除いて，
                                                                       --その下位14ビットを出力
                  end if;
end process;

end rtl;
```

図 15.36: ビット数低減プログラム

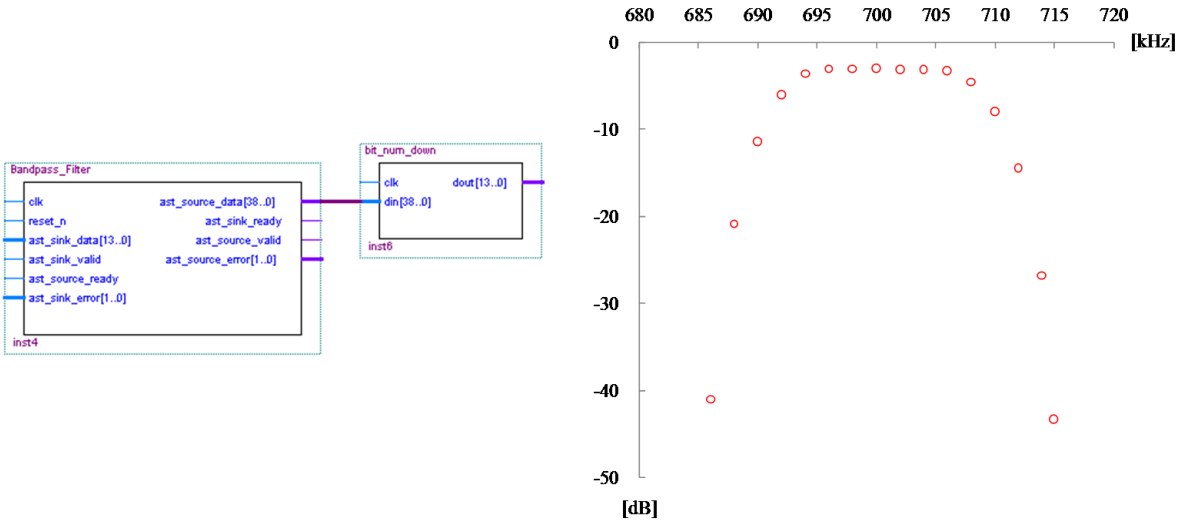


図 15.37: バンドパスフィルタ実験結果

USB ケーブルを外せば，ノイズを小さくでき，しかも，AM ラジオとしてはこれ以上ないと思われる音質で聴くことができた．

15.3 スーパーヘテロダイン方式ソフトウェアラジオ

本節では図 15.7(b) のスーパーヘテロダイン方式ソフトウェアラジオを製作する．

スーパーヘテロダイン方式の特徴は復調前の周波数変換である．第 9 章にも述べたがスーパーヘテロダインは Supersonic heterodyne の略であり，supersonic （超音波の，中

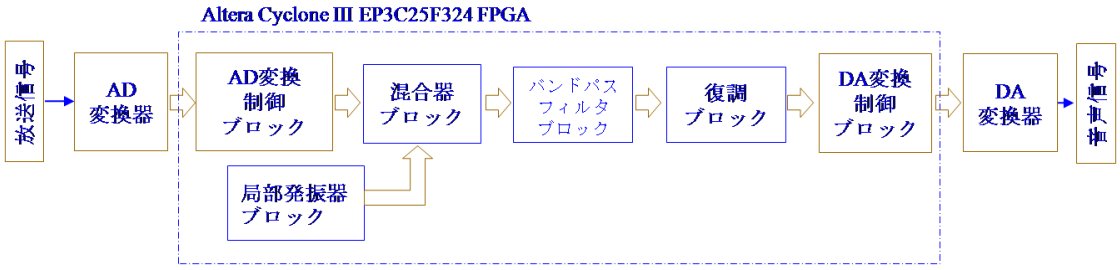


図 15.38: スーパーヘテロダインラジオのブロック図

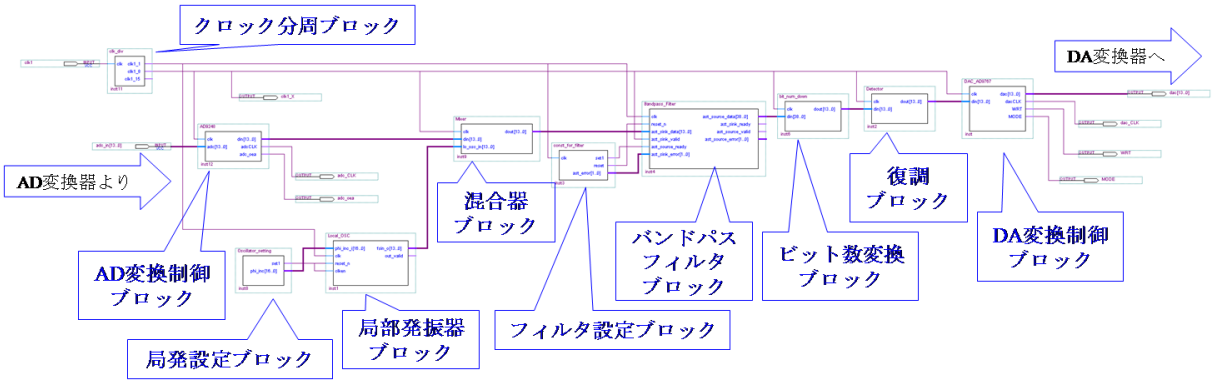


図 15.39: スーパーヘテロダインラジオの bdf

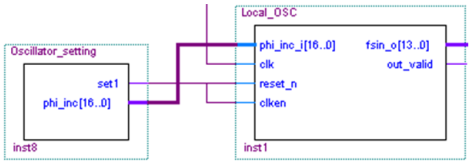


図　15.40:　局部発振プログラムと局発設定プログラムの bsf(Local_OSC.bsf, Oscillator_setting.bsf)

波帯の AM ラジオの場合は 455kHz を指す），hetero（もう一つ別の），dyne（パワー）を意味する．FPGA によりスーパーヘテロダインラジオを製作することは容易である，

図 15.38 はスーパーヘテロダインラジオのブロック図である．新たに混合器ブロックと局部発振器ブロックを挿入してある．図 15.39 はスーパーヘテロダインラジオの bdf(block design file) である．新しい bsf(block symbol file) は Mixer.bsf, Local_OSC.bsf, Oscillator_setting.bsf である．図 15.40 は Local_OSC.bsf と Oscillator_setting.bsf である．Local_OSC.bsf の作成には IP コアを利用する．Quartus II の Tools → MegaWizard Plug-in Manager → Create a new custom megafunction variation → DSP → Signal Generation → NCO → (file name 入力) → Parameterize と進むことで，図 15.41 を開くことができる．クロックレートを 50 [MHz], 希望出力周波数を 1184 [kHz] (=724 [kHz](NHK 名古屋）+ 455 [kHz]（中間周波

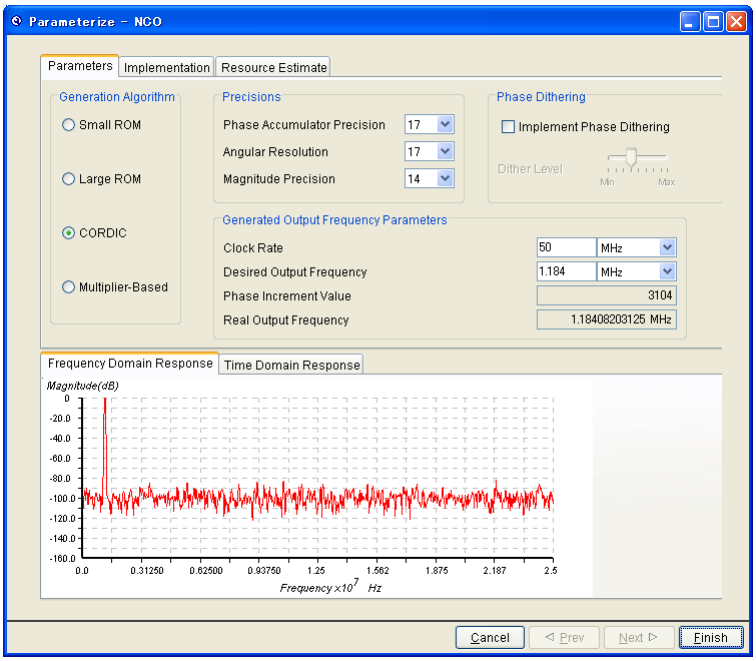


図 15.41: 局部発振の設定画面

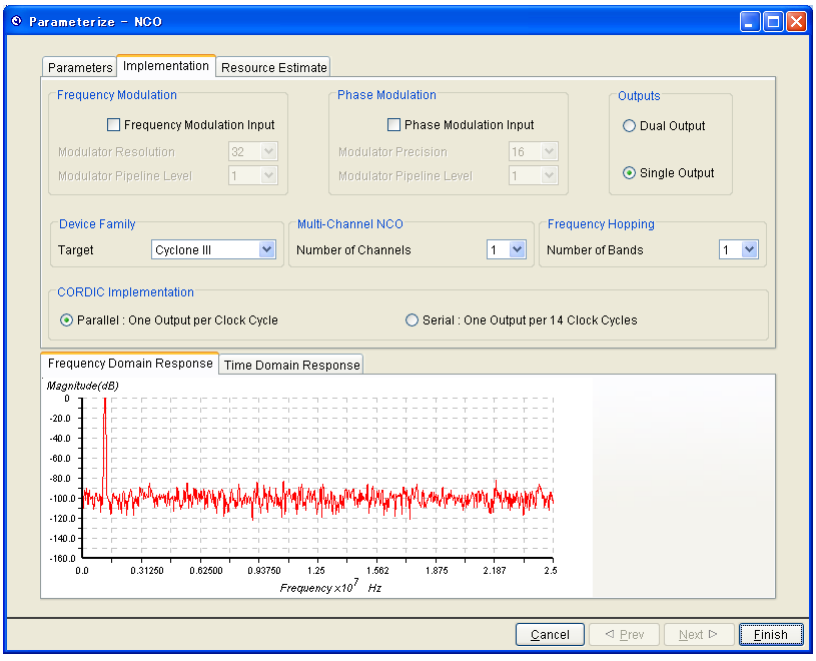


図 15.42: 局部発振の設定画面 2

数）) と設定すると Phase Increment Value = 3104 と出る．この数値は Local_OSC.bsf の入力端子の Phase_inc_i へと与える必要があるので，Oscillator_setting.bsf から与えることとする．出力値の振幅は必要条件ではないが 14 ビットに揃えておく．Implementation のタブをクリックすると図 15.42 が現れる．

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity Oscillator_setting is
      port (
            set1              : out std_logic;
            phi_inc           : out std_logic_vector(16 downto 0));
end Oscillator_setting;

architecture OSCset of Oscillator_setting is

begin
      set1 <= '1';
      phi_inc <= "00000110000100000";    – (10進数) 3104 = (2進数) 110000100000
                                         – Local_OSCブロックをIPコアのNCOを利用して
                                         – 作成する際に,
                                         – クロックを50MHz, 局発周波数を1184kHzに設定すると
                                         – Phase Increment Value は3104であると示される.
                                         – 次図参照

end OSCset;
```

図 15.43: 局部発振の設定プログラム

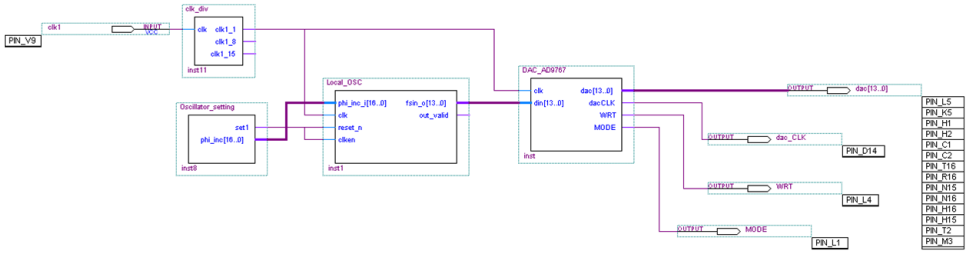


図 15.44: 局部発振テスト bdf

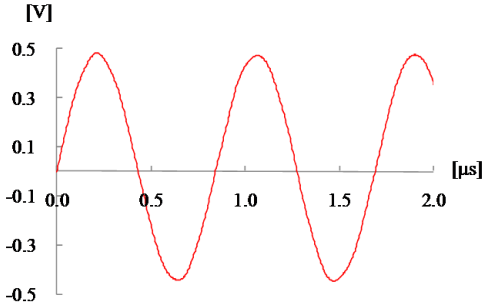


図 15.45: 局部発振実験波形

局部発信器の設定は図 15.43 の局部発振の設定プログラムにより行う．10 進数の 3104 は 17 ビットの 2 進数では 00000110000100000 である．Local_OSC.bsf の reset_n と clken には 1 を与える．

この局部発信ブロックの動きを観測するために製作した bdf を図 15.44 に示す．局部発信ブロックの出力を DA 変換器で観測するテスト用 bdf である．実験結果を図 15.45 に示す．1184 [kHz] の正弦波が生成されていることが分かる．周波数カウンタによる測定結

果は 1184.08[kHz] であった．

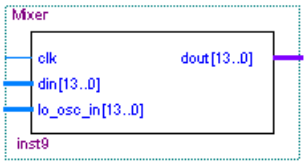


図 15.46: 混合器の bsf(Mixer.bsf)

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
use ieee.std_logic_signed.all;

ENTITY Mixer IS
    PORT(
        clk             : IN STD_LOGIC ;
        din             : IN STD_LOGIC_VECTOR (13 DOWNTO 0);
        lo_osc_in       : IN STD_LOGIC_VECTOR (13 DOWNTO 0);
        dout            : OUT STD_LOGIC_VECTOR (13 DOWNTO 0)
        );
END Mixer;

ARCHITECTURE Mx OF Mixer IS

BEGIN
process(clk)
    begin
        if clk'event and clk = '1' then
            if lo_osc_in > 0 then                   --局発信号が正のとき　放送信号を出力
                dout <= din;
            else                                    --          負のとき　0         を出力
                dout <= "00000000000000";
            end if;
        end if;
end process;

END Mx;
```

図 15.47: 混合回路プログラム

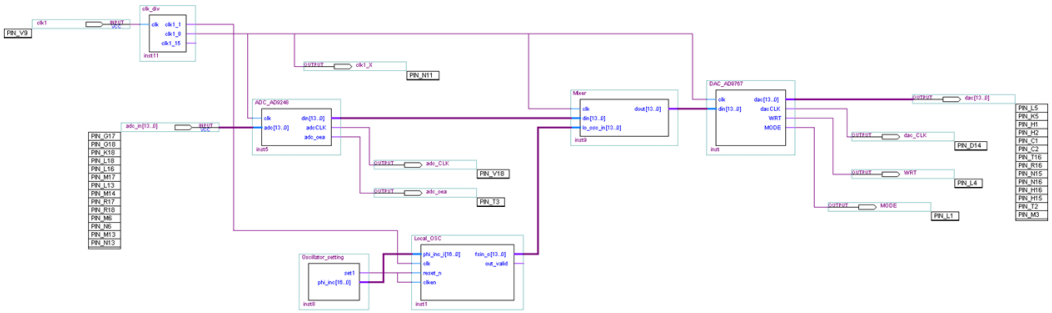


図 15.48: 混合回路テスト bdf

この局部発振信号と放送信号を掛け合わせることで，差分の 455 [kHz] の信号を生成する．掛け合わせを行う回路が混合器である．図 15.46 が混合器の bsf である．ファイル名

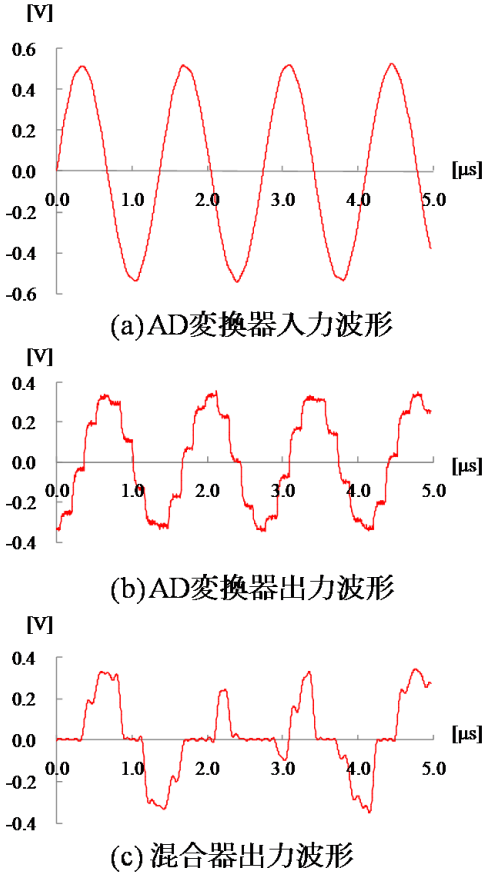


図 15.49: 混合回路実験波形

は Mixer.bsf である．この VHDL ファイルを図 15.47 に示す．混合器のアルゴリズムは簡単である．局部発振信号が正のとき放送信号をそのまま混合器の出力とし，負のとき混合器の出力を 0 とする．この混合器の動作テスト用 bdf を図 15.48 に示す．AD 変換器，混合器, DA 変換器に 1/8 分周クロックを用いている．1/15 分周クロックでも良いのだが，混合器の出力波形が崩れすぎて，混合器の働きを図示するのに適した波形が得られないことによる．Local_OSC のクロックは 50 [MHz] である．実験波形例を図 15.49 に示す．同図 (a) の 729 [kHz] の信号を AD 変換器に入力し，AD 変換器の出力には同図 (b) の 6.25 [MHz] (= 50 [MHz]/8) でサンプルされた波形を得ている．AD 変換器の出力を混合器に通した結果が同図 (c) の混合器出力波形である．放送信号が 1184 [kHz] の局部発振信号により寸断された波形が得られている． 10.1.2 節の周波数変換の理論に述べたように，この波形は 455 [kHz] 成分の他に 729, 1184, 1184+455=1639, ...[kHz] の成分を含む．

この混合器の出力を，455 [kHz] 用のバンドパスフィルタに通せばスーパーヘテロダインラジオが完成する．バンドパスフィルタの低域しゃ断周波数 (Cutoff Freq 1) = 445 [kHz], 高域しゃ断周波数 (Cutoff Freq 2) = 465 [kHz] に設定する．設定画面を図 15.50,

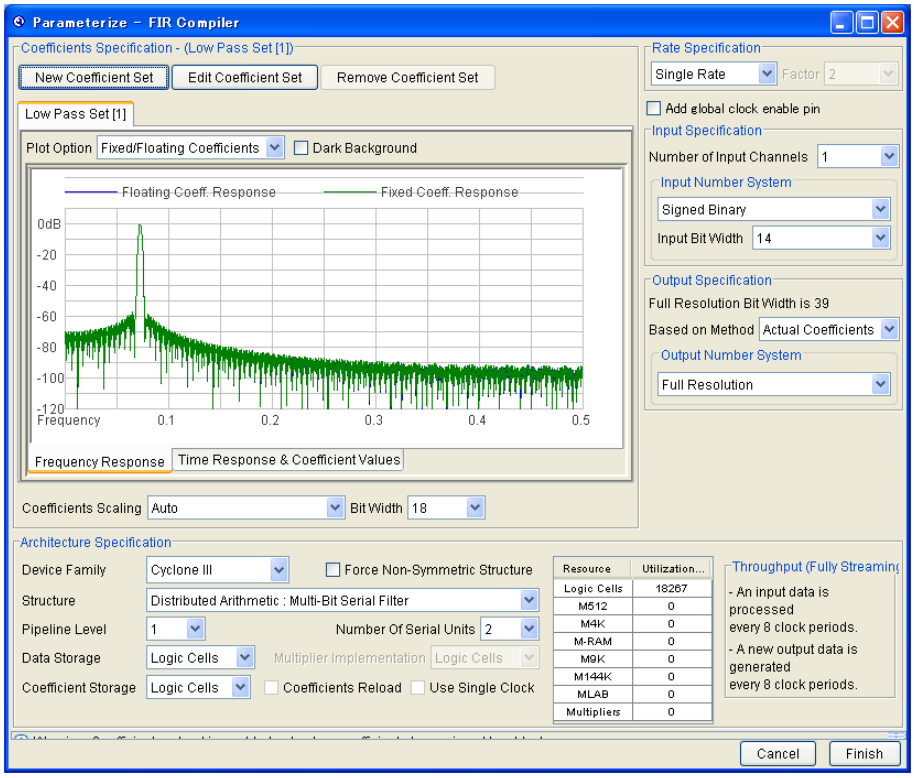


図 15.50: バンドパスフィルタ設定画面 (455kHz)

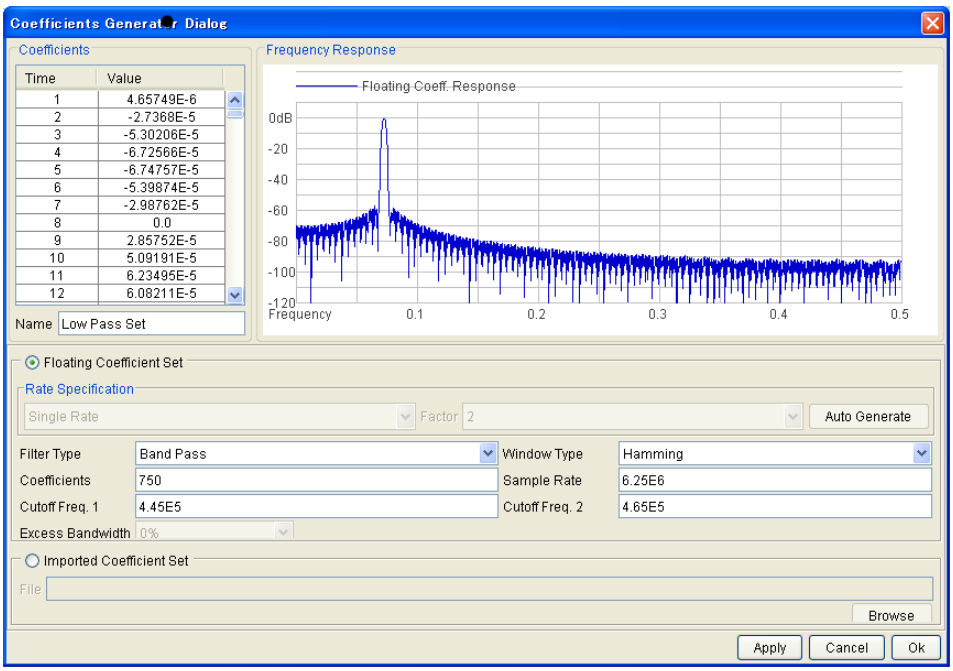


図 15.51: バンドパスフィルタ設定画面 2(455kHz)

図 15.51 に示す．8 クロックに 1 回の入出力を実行できる Multi-Bit Serial Filter を用いている．FIR フィルタのタップ数は約 750 までとることができる．

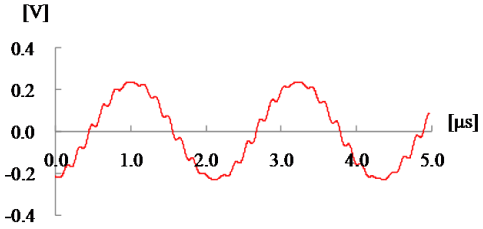


図 15.52: 455kHz バンドパスフィルタ実験波形

図 15.49(c) の波形をバンドパスフィルタに入力し，その出力を DA 変換器を通して観測したところ，図 15.52 の波形が得られた．455 [kHz] 成分が 6.25 [MHz] でサンプルされた波形である．

以上の設定をした後，図 15.39 の bdf をコンパイルして FPGA にダウンロードすれば，NHK 名古屋の放送を聴くことができる．

選択度，音質ともに前節のバンドパスフィルタ付きストレートラジオと同等の性能である．スーパーヘテロダインラジオの利点は，選局が容易なことである．Phase Increment Value の値を変えるだけでよい，例えば FPGA ボード上のブッシュスイッチを利用して，図 15.43 の VHDL ファイルの phi_inc を書き換えるプログラムを書くだけ達成できる．

2011 年 3 月

著者
古橋武
名古屋大学工学研究科情報・通信工学専攻
furuhashi at nuee.nagoya-u.ac.jp