

—座学実験一体型教科書シリーズ第 x 巻—

PICマイコンによるデジタルフィルタ

古橋 武

令和7年10月

まえがき

本書『PIC マイコンによるデジタルフィルタ』は、「座学・実験一体型教科書シリーズ」の一巻として、デジタル信号処理 (Digital Signal Processing: DSP) の基本概念を、マイコン実習を通して体得できるよう構成したものである。近年、アナログ回路とデジタル処理の境界は急速に曖昧となり、信号処理の知識はあらゆる工学分野で必須の素養となっている。とくに制御、通信、音響、センシングなどの分野では、信号をデジタル的に処理する能力が技術者の基礎体力を左右する。本書は、その第一歩として、PIC マイコンに実装したデジタルフィルタの設計・製作・評価を一貫して学ぶことを目的としている。

本書で扱う dsPIC33EV256GM102 は、Microchip 社が提供する高速演算機能付きの 16 ビット・マイクロコントローラである。DSP 命令セットを備え、乗算・加算をハードウェアレベルで実行できるため、IIR フィルタや FIR フィルタのような多重積和演算をリアルタイムに処理できる。第 1 章ではブレッドボード上に構築する回路例を示し、ADALM2000 による信号観測や周波数特性の測定を通して、アナログ入力からデジタル処理、アナログ出力までの信号の流れを具体的にしめす。第 2 章では IIR フィルタ、第 3 章では FIR フィルタを取り上げ、次数・タップ数による特性の違いや、サンプリングおよび処理時間による位相遅れの影響を丁寧に追う。さらに、第 4 章では実際のプログラム例を示し、固定小数点演算 (Q15 フォーマット) の扱い、SPI 通信を用いた D-A コンバータ制御、DSP 関数 (VectorDotProduct, VectorCopy) による高速化など、実践的な開発技法を解説する。

また付録には、部品表と部品の取り扱い方、Scopy の設定手順、MPLAB® X IDE と XC16 コンパイラの導入・起動方法、MCC による周辺モジュール設定など、授業や自習で即利用できる情報を収録した。これらを順に追うことで、読者はハードウェアの接続からソフトウェア開発、動作検証に至るまでを自ら完結できるようになるだろう。

本書の特徴は、単なる理論書でも工作マニュアルでもなく、座学と実験を往復しながら理解を深める「学びの流れ」を重視している点にある。フィルタの伝達関数を式として学ぶだけでなく、実際にその応答を観測し、理論値と照合して「なぜ一致しないのか」を考察する。そうした体験の積み重ねこそが、信号処理の本質的理解と創造的応用力を育てると信じている。デジタル信号処理の世界への第一歩として、本書が多くの学習者にとって確かな導きとなることを願う。

令和 7 年 10 月

古橋武

名古屋大学名誉教授

目次

第1章	製作回路	5
第2章	IIR フィルタ	9
2.1	1次バタワース・ロー・パス・フィルタ	9
2.2	サンプリングと処理時間による位相遅れ	12
2.3	2次バタワース・ロー・パス・フィルタ	14
2.4	3次バタワース・ロー・パス・フィルタ	16
2.5	8次バタワース・ロー・パス・フィルタ	18
第3章	FIR フィルタ	21
3.1	21 タップ・ロー・パス・フィルタ	21
3.2	サンプリングと処理時間による位相遅れ	23
3.3	FIR フィルタの特徴	24
3.4	91 タップ・ロー・パス・フィルタ	25
3.5	DSP 関数 (VectorCopy, VectorDotProduct) によるフィルタ演算の高速化	25
第4章	プログラム	27
4.1	ブロック図	27
4.2	1次バタワース・ロー・パス・フィルタ	28
4.2.1	フィルタ係数	28
4.2.2	1次バタワース・ロー・パス・フィルタ・プログラム	30
4.2.3	SPI モジュールによるデータ転送プログラム	31
4.3	2次バタワース・ロー・パス・フィルタ	32
4.3.1	フィルタ係数	32
4.3.2	2次バタワース・ロー・パス・フィルタ・プログラム	32
4.4	3次バタワース・ロー・パス・フィルタ	33
4.4.1	フィルタ係数	33
4.4.2	3次バタワース・ロー・パス・フィルタ・プログラム	34
4.5	21 タップ FIR ロー・パス・フィルタ	34
4.5.1	フィルタ係数	35
4.5.2	21 タップ FIR ロー・パス・フィルタ・プログラム	35
4.6	91 タップ FIR ロー・パス・フィルタ	36
4.6.1	フィルタ係数	36
4.6.2	91 タップ FIR ロー・パス・フィルタ・プログラム	37

付 録 A 部品	41
A.1 部品表	41
A.2 部品	41
A.2.1 D-A コンバータ	41
A.2.2 アクティブ・ラーニングモジュール (ADALM2000)	43
A.2.3 インサーキット・デバッガ／プログラマ	45
A.2.4 コンデンサ	46
A.2.5 ジャンパー・ワイヤ	46
A.2.6 抵抗	47
A.2.7 電線	47
A.2.8 ピン・ヘッダ, ピン・ソケット	48
A.2.9 ブレッド・ボード	49
A.2.10 マイコン	50
付 録 B Scopy の設定例	51
B.1 Scopy のインストールと立ち上げ	51
B.2 ネットワーク・アナライザの設定	53
B.2.1 基準信号源設定・信号のフィルタリング設定	53
B.2.2 掃引設定	54
B.2.3 グラフ表示範囲設定	55
付 録 C MPLAB[®] X IDE, XC16 コンパイラ, MCC	57
C.1 MPLAB [®] X IDE, XC16 コンパイラ	57
C.1.1 ダウンロード, インストール方法	57
C.1.2 MPLAB [®] X IDE の起動	57
C.1.3 既存プロジェクトの開き方	58
C.2 New Project の作成方法	59
C.3 MPLAB [®] Code Configurator (MCC) による周辺モジュール設定	61
C.3.1 MCC の起動	61
C.3.2 System Module 設定	61
C.3.3 A-D コンバータ・モジュール設定	64
C.3.4 タイマ3モジュール設定	67
C.3.5 SPI モジュール設定	67
C.4 周辺モジュール設定関数のコード自動生成	69
C.5 ビルド, マイコンへの書き込み	70
C.6 デバッガ	71
索 引	73
参考文献	73

第1章 製作回路

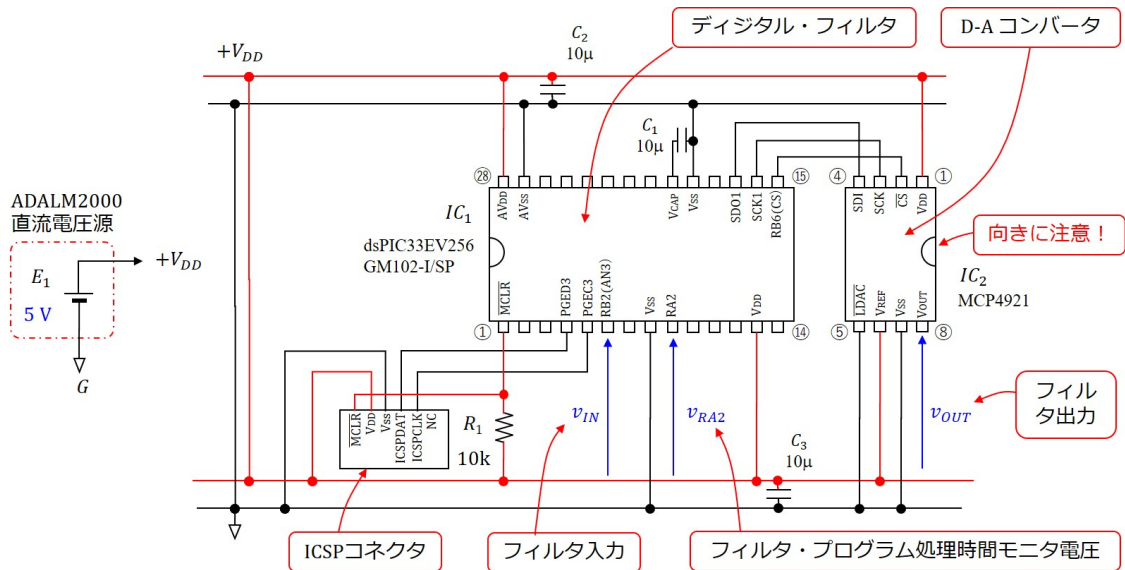


図 1.1: PIC マイコンによるデジタル・フィルタ

図 1.1 は **dsPIC マイコン** による **デジタル・フィルタ** です。デジタル・フィルタは多くのかけ算処理を必要とします。そこで、**DSP**(Digital Signal Processing: 高速演算処理) 機能を持つ PIC マイコン (dsPIC マイコン) を使用することにしました。**dsPIC33EV256GM102-I/SP** は 28 ピンの DIP タイプであり、ブレッドボードに挿入できます。さらに、70MIPS の演算速度を持ち、フィルタの製作・実験において、DSP 機能の効果を実感しやすいです。ネット通販（例えば、秋月電子通商、チップワンストップ、RS コンポーネンツ、Digikey など）で購入できます。

デジタル・フィルタはソフトウェアにより構成します。フィルタの入力電圧 v_{IN} は 6 番ピン (AN3) から 12 ビット A-D コンバータによりマイコンに取り込みます。フィルタ出力電圧 v_{OUT} は 12 ビット **D-A コンバータ** (**MCP4921**) の 8 番ピン (V_{OUT}) から出力します。A-D コンバータ、D-A コンバータのサンプリング周波数のモニタ信号 v_{SAMP} をマイコンの 9 番ピン (RA2) から出力します。

D-A コンバータ MCP4921 は、8 ピンの DIP タイプであり、マイコンと同じ Microchip 社製です。**SPI**(Serial Peripheral Interface) 通信によりマイコンからデータを受信します。**SDI**(Serial Data In), **SCK**(Serial Clock), **CS**(Chip Select) は SPI 通信用ピンの機能を表します。SDI に対応するマイコン側の機能は **SDO**(Serial Data Out) です。ネット通販 (例

えば、上記販売店など）で購入できます。配線の都合で D-A コンバータの向きをマイコンとは逆にしています。

5V 電源には ADALM2000 の直流電圧源を使います。コンデンサ C_1 はマイコン内部の電圧レギュレータの電圧安定化のために必要です。静電容量の要求仕様は $4.7\text{ }\mu\text{F}$ 以上です。コンデンサ C_2 , C_3 はノイズ抑制用として念のため入れてあります。

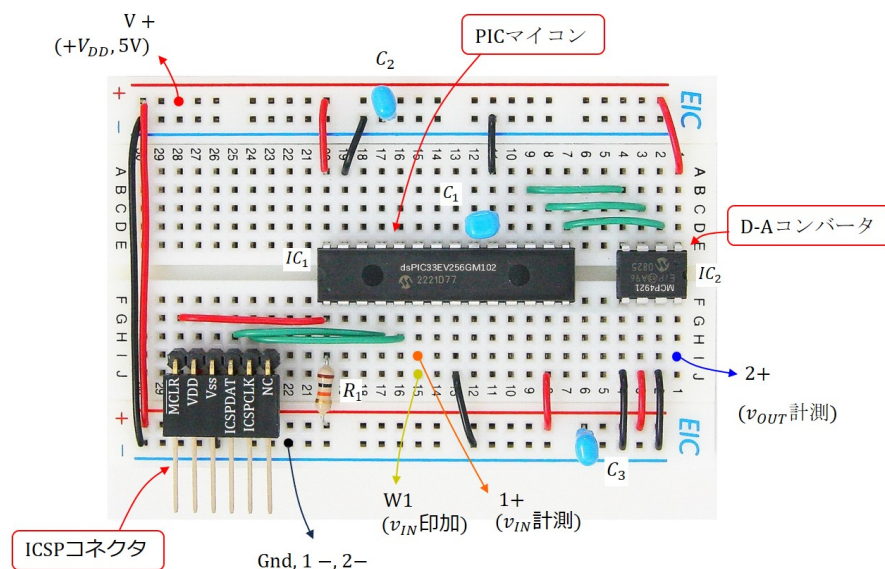


図 1.2: PIC マイコンによるデジタル・フィルタの配線例

図 1.2 は dsPIC マイコンによるデジタル・フィルタの配線例です。ブレッドボード (EIC-801) 上に構築しています。部品については付録 A を参照してください。

図 1.3 は、PIC マイコンによるデジタル・フィルタ回路と USB 計測器 (ADALM2000) との接続例を示しています。マイコン IC_1 (dsPIC33EV256GM102-I/SP) と D-A コンバータ IC_2 (MCP4921) の動作電源 (5 V) は、ADALM2000 の $V+$ ピンから供給します。

1+ ピンと 1- ピンは、オシロスコープおよびネットワーク・アナライザのチャンネル 1 (CH1) に接続され、入力電圧波形や周波数特性の観測に用いられます。同様に、2+ ピンと 2- ピンはチャンネル 2 (CH2) に接続され、出力波形および伝達特性の観測に利用されます。

ADALM2000 の外観およびピン配置は、付録 A.2.2 を参照してください。また、dsPIC マイコン用のデジタル・フィルタ・プログラムについては、第 4 章に示されています。

第2章 IIR フィルタ

2.1 1次バタワース・ロー・パス・フィルタ

フィルタ入力電圧 v_{IN} は,

$$v_{IN} = v_{in} + 2.5 \text{ [V]} \quad (2.1)$$

と, 信号成分 v_{in} を 2.5 V のオフセット電圧に載せて与えます.

フィルタ出力 v_{OUT1} は,

$$v_{OUT1} = v_{out1} + 2.5 \text{ [V]} \quad (2.2)$$

と, 信号成分 v_{out} がオフセット電圧 2.5 [V] に載せて出力されます.

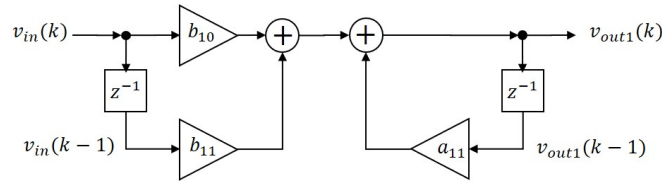


図 2.1: 1次バタワース・ロー・パス・フィルタの構成

図 2.1 は 1 次バタワース・ロー・パス・フィルタの構成を示します. この構成が表す出力電圧 v_{out1} と入力電圧 v_{in} の関係を, 式で表現すると

$$v_{out1}(k) = b_{10}v_{in}(k) + b_{11}v_{in}(k-1) + a_{11}v_{out1}(k-1) \quad (2.3)$$

です. b_{10} , b_{11} , a_{11} は **フィルタ係数** です. k は現在の **サンプリング時刻** を表します. $k-1$ は 1 サンプル前のサンプリング時刻です. $v_{out1}(k)$ は, 入力電圧 $v_{in}(k)$ に対して時間遅れの無い出力電圧です. 実際のマイコンでは, $v_{out1}(k)$ は処理時間分だけ遅れて出力されます.

図 2.2 は 1 次バタワース・ロー・パス・フィルタの入出力波形例です. マイコンの A-D コンバータおよび D-A コンバータのサンプリング周波数 $f_{samp} = 15 \text{ [kHz]}$ とし, フィルタのカットオフ周波数 $f_c = 1 \text{ [kHz]}$ となるようにフィルタ定数を設定して実験を行いました. (a) は信号成分の周波数 $f_s = 200 \text{ [Hz]}$ の場合, (b), (c), (d) はそれぞれ $f_s = 500 \text{ [Hz]}$, 1 [kHz] , 2 [kHz] の場合です. 入力信号成分 v_{in} が正弦波であるのに対して, 出力信号成分 v_{out1} はサンプリングの影響を受けて階段状波形となっています. 階段の踏面幅は **サンプリング周期** $T_{samp} = 1/f_{samp} = 66.7 \text{ [}\mu\text{s]}$ です.

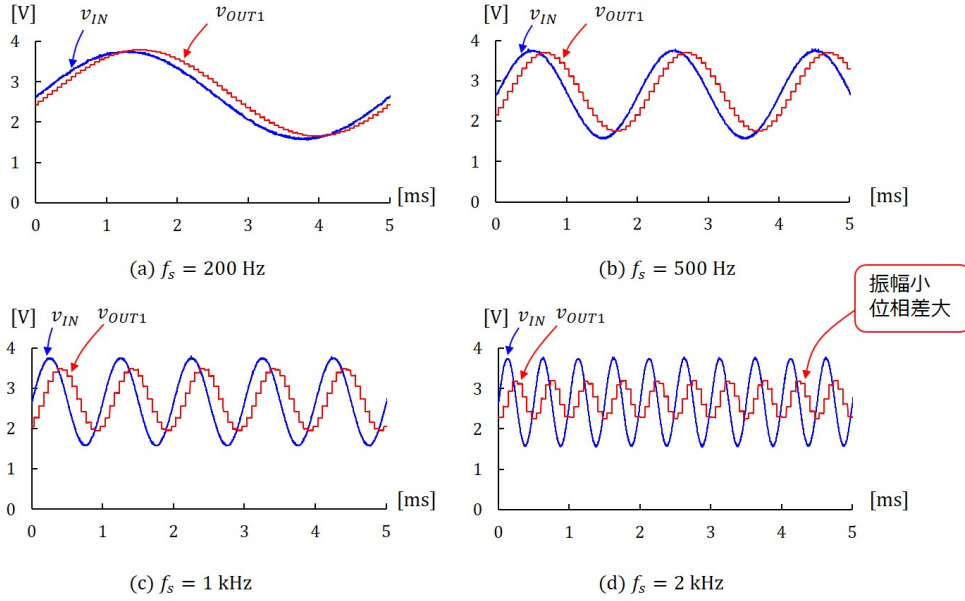


図 2.2: 1 次バタワース・ロー・パス・フィルタの実験波形例

v_{in} の振幅はいずれの場合も 1 V です。信号成分の周波数 f_s が高くなるにつれて、 v_{out1} の振幅は小さくなっています。そして、 v_{in} と v_{out1} の位相差は f_s が高いほど大きくなっています。

図 2.3 は 1 次バタワース・ロー・パス・フィルタの周波数特性です。ネットワーク・アナライザの設定は付録 B.2 節を参照してください。(a) が電圧増幅度特性、(b) が位相特性です。入力信号成分 v_{in} の実効値を V_{in} 、出力信号成分 v_{out} の実効値を V_{out1} として、電圧増幅度 G_v を

$$G_v = 20 \log_{10} \frac{V_{out1}}{V_{in}} \quad [\text{dB}] \quad (2.4)$$

と定義し、位相差 $\psi_{v_{out1}-v_{in}}$ を v_{out1} の v_{in} に対する位相差と定義します。実線が実験値、破線が計算値です。 f_{nyq} は**ナイキスト周波数**です。サンプリング周波数 $f_{samp} = 15 \text{ [kHz]}$ のとき、 $f_{nyq} = f_{samp}/2 = 7.5 \text{ [kHz]}$ です。

カットオフ周波数 f_c より低い周波数域を**通過域**、高い周波数域を**しゃ断域**と呼びます。1 次バタワース・ロー・パス・フィルタでは、 f_c にて、電圧増幅度 G_v が通過域より 3 [dB] 低くなります。そして、位相差 $\psi_{v_{out1}-v_{in}}$ が、計算値では -45° になります。また、しゃ断域で、 G_v は -20 [dB/dec] (周波数が 10 倍になると増幅度が 20 dB 低下する) の傾向を示します。

G_v の計算値は実験値とほぼ一致しました。一方、 $\psi_{v_{out1}-v_{in}}$ の実験値は計算値よりも、しゃ断域で大きく遅れました。計算値は (2.7) 式、(2.8) 式より得ました。

1 次バタワース・ロー・パス・フィルタの**伝達関数** $H_1(z)$ は

$$H_1(z) = \frac{b_{10} + b_{11}z^{-1}}{1 - a_{11}z^{-1}} \quad (2.5)$$

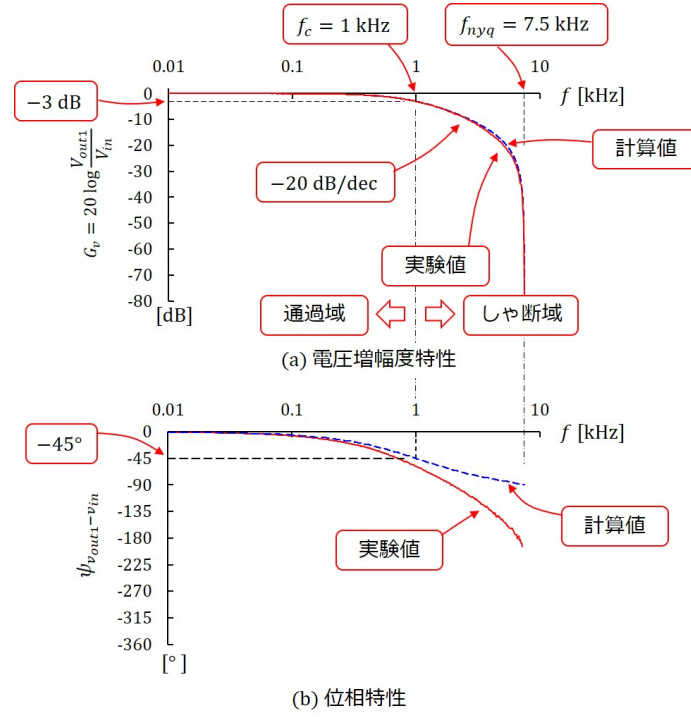


図 2.3: 1次バタワース・ロー・パス・フィルタの周波数特性

です。ただし,

$$\begin{aligned} b_{10} = b_{11} &= \frac{\omega_A T_s}{2 + \omega_A T_s} \\ a_{11} &= \frac{2 - \omega_A T_s}{2 + \omega_A T_s} \end{aligned} \quad (2.6)$$

です。また, $\omega_A T_s = 2 \tan(\omega_c T_s / 2)$, $\omega_c = 2\pi f_c$ で, f_c はカットオフ周波数, T_s はサンプリング周期です。

周波数特性は, (2.5) 式にて $z = e^{j\omega T_s}$ と置き換えて, 実部と虚部をそれぞれまとめることで,

$$G_v(f) = 20 \log_{10} \frac{\sqrt{(b_{10} + b_{11} \cos \omega T_s)^2 + (b_{11} \sin \omega T_s)^2}}{\sqrt{(1 - a_{11} \cos \omega T_s)^2 + (a_{11} \sin \omega T_s)^2}} \quad (2.7)$$

$$\psi_{v_{out1}-v_{in}}(f) = \tan^{-1} \left(\frac{b_{11} \sin \omega T_s}{b_{10} + b_{11} \cos \omega T_s} \right) - \tan^{-1} \left(\frac{a_{11} \sin \omega T_s}{1 - a_{11} \cos \omega T_s} \right) \quad (2.8)$$

と求められます。

2.2 サンプリングと処理時間による位相遅れ

図 2.3(b) の位相特性の計算値は実験値と大きくずれました．これは，サンプリングによる位相遅れと，マイコンの処理時間による位相遅れが (2.8) 式に反映されていないことが原因です．

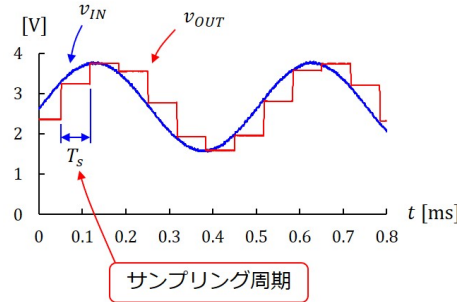


図 2.4: フィルタを通さない場合のマイコン入出力波形例

図 2.4 は図 1.1 の回路において，入力電圧 v_{IN} を A-D コンバータによりマイコンに取り込み，フィルタ演算を通さずに，D-A コンバータより出力した結果です．信号成分の周波数 $f_s = 2$ [kHz] の場合です．出力電圧 v_{OUT} は，サンプリング周期 T_s 毎に更新され， T_s の期間一定値を保つ階段状波形となっています．

v_{OUT} の信号成分 v_{out} の位相は，平均してサンプリング周期 T_s の半周期分 ($T_s/2$) 遅れます．この遅れによる位相差 ψ_{T_s} は

$$\psi_{T_s}(f) = -\frac{T_s}{2} \times f \times 360 \text{ [}^\circ \text{]} \quad (2.9)$$

となります．この位相差 $\psi_{T_s}(f)$ が (2.8) 式の $\psi_{v_{out1}-v_{in}}(f)$ に加わります．

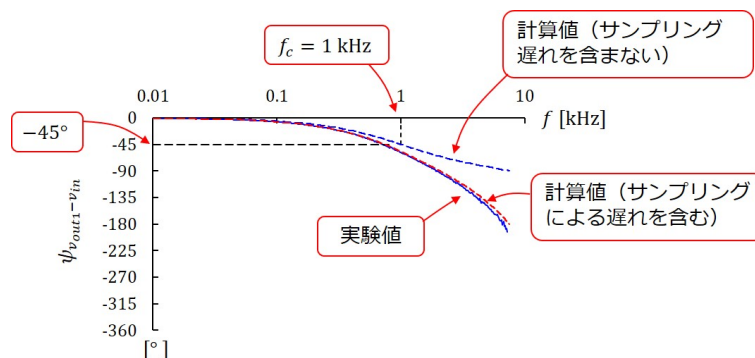
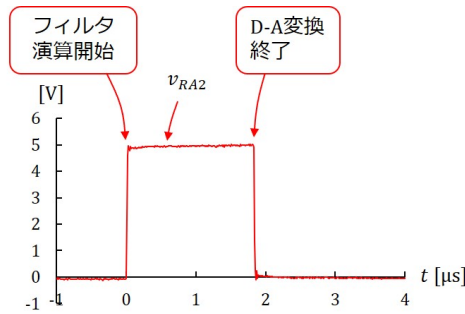


図 2.5: 1 次バタワース・ロー・パス・フィルタの位相特性
(サンプリングによる位相遅れを取り入れた計算値)

図 2.5 に，サンプリングによる遅れを含む場合の位相特性の計算値を，図 2.3 の位相特性に追加して示します． $T_s/2 = 33.3$ [μ s] です．計算値は実験値に近づきました．

マイコン内では A-D コンバータによる入力電圧 v_{IN} のデジタル値への変換，フィルタ演算，D-A コンバータによる出力電圧 v_{OUT1} へのアナログ値への変換が行われます．これらの処理時間は，わずかですが， v_{OUT1} の位相を遅らせます．(2.9) 式にはこれらの処理時間は含まれていません．



図CL13-3 フィルタ・プログラム処理時間モニタ電圧の波形例
(1次バタワース・ロー・パス・フィルタ)

図 2.6: フィルタ・プログラム処理時間モニタ電圧の波形例
(1次バタワース・ロー・パス・フィルタ)

図 2.6 はフィルタ・プログラム処理時間モニタ電圧 v_{RA2} の波形例です．図 1.1 の 9 番ピンの出力電圧です．マイコンのプログラムは，A-D 変換終了時点で $v_{RA2} = “H”$ とし，フィルタ演算および D-A 変換終了時点で $v_{RA2} = “L”$ としています．波形より，フィルタ演算と D-A 変換に要する時間 $T_{fil+DA} = 1.8 \text{ } [\mu\text{s}]$ です．また，データシートの ADC CONVERSION TIMING REQUIREMENTS によると，このマイコンの場合，12 ビット A-D 変換に要する時間 $T_{ADC} = 2.2 \text{ } [\mu\text{s}]$ です．以上の処理時間による位相差 ψ_{proc} は

$$\psi_{proc} = -(T_{fil+DA} + T_{ADC}) \times f \times 360 \text{ } [^\circ] \quad (2.10)$$

と求められます．この位相遅れも (2.8) 式に加わります．

図 2.7 はサンプリングによる位相遅れに加えて，A-D 変換・フィルタ演算・D-A 変換処理による位相遅れを加えた場合の計算値を示します．遅れ時間の総和は

$$\frac{T_s}{2} + T_{ADC} + T_{fil+DA} = 37.3 \text{ } \mu\text{s} \quad (2.11)$$

でした．計算値は実験値にほぼ一致しました．

ただし，以降の 2, 3, 8 次のロー・パス・フィルタの計算値にはこの位相遅れ補償は含まれていません．

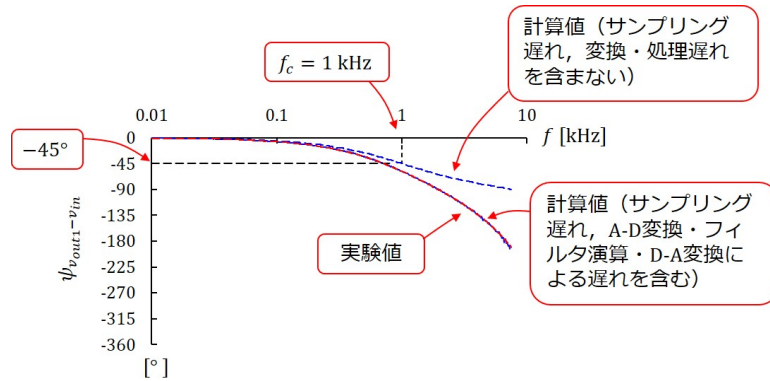


図 2.7: 1 次バタワース・ロー・パス・フィルタの位相特性
(サンプリングによる位相遅れ, A-D 変換・フィルタ演算・D-A 変換
による位相遅れを取り入れた計算値)

2.3 2 次バタワース・ロー・パス・フィルタ

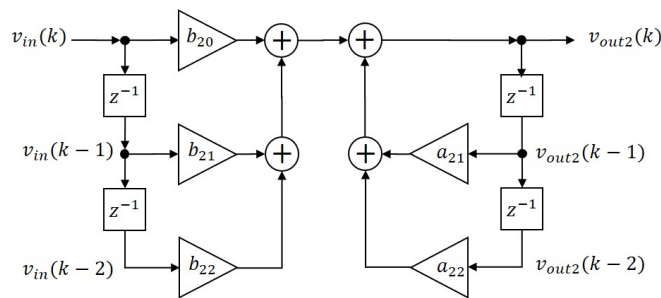


図 2.8: 2 次バタワース・ロー・パス・フィルタの構成

図 2.8 は 2 次バタワース・ロー・パス・フィルタの構成です。この構成が表す入出力関係は

$$v_{out2}(k) = b_{20}v_{in}(k) + b_{21}v_{in}(k-1) + b_{22}v_{in}(k-2) + a_{21}v_{out2}(k-1) + a_{22}v_{out2}(k-2) \quad (2.12)$$

です。 b_{20} , b_{21} , b_{22} , a_{21} , a_{22} はフィルタ係数です。

図 2.9 は 2 次バタワース・ロー・パス・フィルタの周波数特性です。実線が実測値、破線が計算値です。図 13-7 の 1 次バタワース・ロー・パス・フィルタの周波数特性との大きな違いは、しゃ断域で、電圧増幅度 G_v が -40 dB/dec の傾向を示すことです。また、バタワース・ロー・パス・フィルタの特徴は、 G_v が f_c にて通過域より 3 dB 低下することです。1 次でも 2 次でも 3 dB 低下します。

位相差 $\psi_{v_{out2}-v_{in}}$ は、計算値では、 f_c にて -90° です。実験値は計算値よりも遅れました。これは、サンプリングによる位相遅れ、マイコン内の処理による位相遅れが原因です。

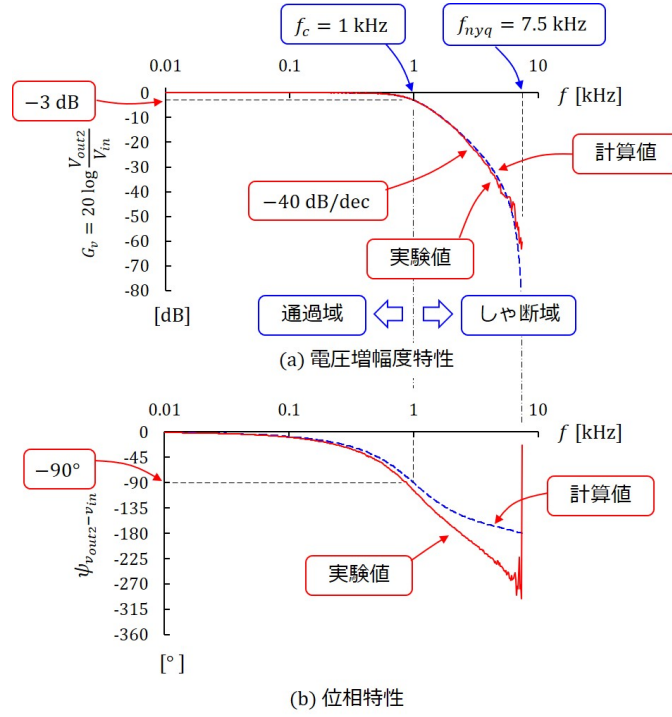


図 2.9: 2次バタワース・ロー・パス・フィルタの周波数特性

また、ナイキスト周波数 f_{nyq} に近いところで、 $\psi_{v_{out2}-v_{in}}$ の実験値は大きく振動しています。これは、出力電圧の変化分 v_{out2} の振幅が小さくなりすぎているためです。この周波数域では、ネットワーク・アナライザの画面に “Response channel voltage < 50 mV!” のアラームが表示されます。

計算値は次式により得ました。 m 次バタワース・ロー・パス・フィルタの伝達関数 $HE_m(z)$ は

$$HE_m(z) = \frac{b_{m0} + b_{m1}z^{-1} + b_{m2}z^{-2}}{1 - a_{m1}z^{-1} - a_{m2}z^{-2}} \quad (2.13)$$

です。ただし、

$$\begin{aligned} 2b_{m0} = b_{m1} = 2b_{m2} &= \frac{2}{\Delta} (\omega_A T_s)^2 \\ a_{m1} &= \frac{1}{\Delta} (8 - 2(\omega_A T_s)^2) \\ a_{m2} &= -\frac{1}{\Delta} (4 + 4\omega_A T_s \cos \theta_m + (\omega_A T_s)^2) \\ \Delta_m &= 4 - 4\omega_A T_s \cos \theta_m + (\omega_A T_s)^2 \end{aligned} \quad (2.14)$$

です。また、 $\theta_m = (m + N - 1)/(2N)$ 、 $\omega_A T_s = 2 \tan(\omega_c T_s/2)$ 、 $\omega_c = 2\pi f_c$ で、 f_c はカットオフ周波数、 T_s はサンプリング周期です。2次バタワース・ロー・パス・フィルタの場合、 $m = 2$ 、 $N = 2$ です。

2.4 3次バタワース・ロー・パス・フィルタ

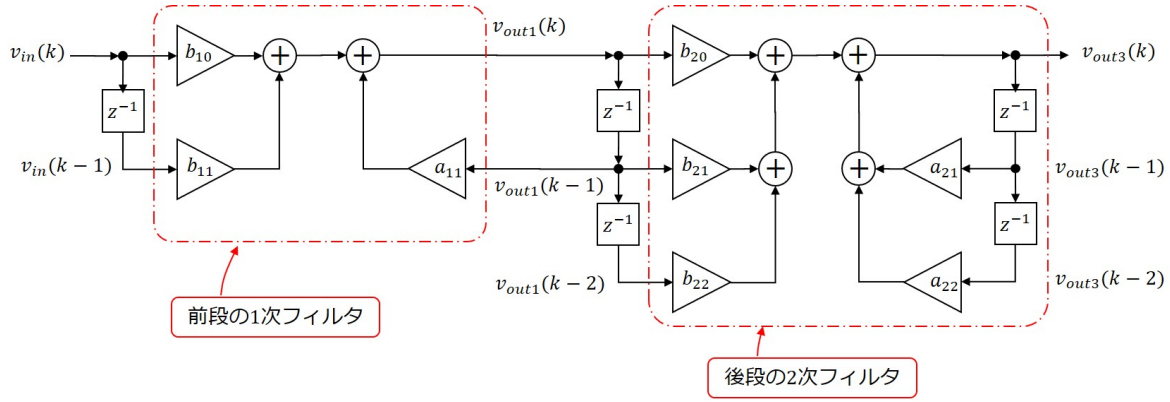


図 2.10: 3次バタワース・ロー・パス・フィルタの構成

図 2.10 は 3 次バタワース・ロー・パス・フィルタの構成です。この構成の表す入出力関係は

$$\begin{aligned}
 v_{out1}(k) &= b_{10}v_{in}(k) + b_{11}v_{in}(k-1) + a_{11}v_{out1}(k-1) \\
 v_{out3}(k) &= b_{20}v_{out1}(k) + b_{21}v_{out1}(k-1) + b_{22}v_{out1}(k-2) + a_{21}v_{out3}(k-1) \\
 &\quad + a_{22}v_{out3}(k-2)
 \end{aligned}
 \tag{2.15}$$

です。 v_{out1} は前段の 1 次フィルタの出力電圧であり、そして、後段の 2 次フィルタの入力電圧です。 b_{10} , b_{11} , $\dots$, a_{21} , a_{22} は **フィルタ係数** です。 3 次バタワース・ロー・パス・フィルタの場合、 b_{10} , b_{11} , a_{11} は (2.6) 式から得られます。 $b_{20} \sim a_{22}$ は、 (2.14) 式において、 $m = 2$, $N = 3$ として求められます。

図 2.11 は 3 次バタワース・ロー・パス・フィルタの周波数特性です。 実線が実測値、破線が計算値です。 電圧増幅度 G_v は、シャ断域で -60 [dB/dec] です。 また、 f_c にて通過域より 3 dB 低下しています。 次数にかかわらず、この 3 dB 低下がバタワース・ロー・パス・フィルタの特徴です。

位相差 $\psi_{v_{out3}-v_{in}}$ は、計算値では、 f_c にて -90° です。 ナイキスト周波数 f_{nyq} に近いところで、 $\psi_{v_{out3}-v_{in}}$ の振動が 2 次フィルタの場合よりも大きくなっています。 3 次フィルタでは、 f_{nyq} 付近での v_{out3} の減衰がより大きくなっているためです。

計算値は次式によります。 3 次バタワース・ロー・パス・フィルタの **伝達関数** $H_3(z)$ は、 (2.5) 式の伝達関数 $H_1(z)$ と (2.13) 式の伝達関数 $HE_m(z)$ を用いて、

$$H_3(z) = H_1(z)HE_2(z) \tag{2.16}$$

と表されます。 $HE_2(z)$ において、 $N = 3$ です。 1 次のフィルタと 2 次のフィルタの縦続接続を表しています。

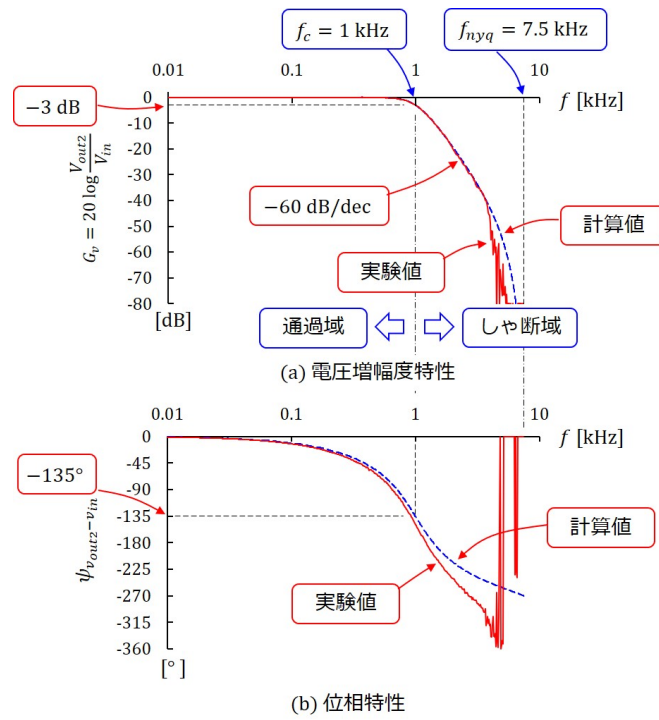


図 2.11: 3次バタワース・ロー・パス・フィルタの周波数特性

2.5 8次バタワース・ロー・パス・フィルタ

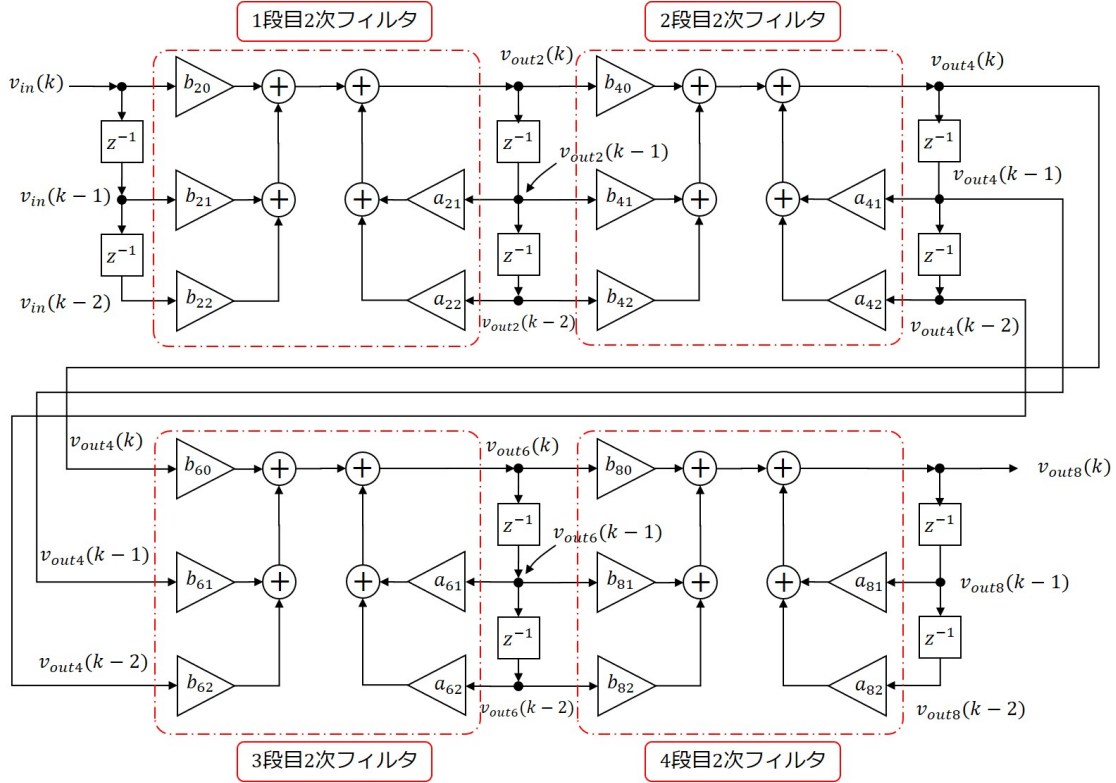


図 2.12: 8 次バタワース・ロー・パス・フィルタの構成

図 2.12 は 8 次バタワース・ロー・パス・フィルタの構成です。この構成の入出力関係は

$$\begin{aligned}
 v_{out2}(k) &= b_{20}v_{in}(k) + b_{21}v_{in}(k-1) + b_{22}v_{in}(k-2) + a_{21}v_{out2}(k-1) + a_{22}v_{out2}(k-2) \\
 v_{4out}(k) &= b_{40}v_{out2}(k) + b_{41}v_{out2}(k-1) + b_{42}v_{out2}(k-2) + a_{41}v_{out4}(k-1) + a_{42}v_{out4}(k-2) \\
 v_{6out}(k) &= b_{60}v_{out4}(k) + b_{61}v_{out4}(k-1) + b_{62}v_{out4}(k-2) + a_{61}v_{out6}(k-1) + a_{62}v_{out6}(k-2) \\
 v_{out8}(k) &= b_{80}v_{out6}(k) + b_{81}v_{out6}(k-1) + b_{82}v_{out6}(k-2) + a_{81}v_{out8}(k-1) + a_{82}v_{out8}(k-2)
 \end{aligned}
 \tag{2.17}$$

です。bm0～am2 ($m = 2, 4, 6, 8$) は、**フィルタ係数**です。8 次バタワース・ロー・パス・フィルタの場合、(2.14) 式より、 $N = 8$ として求められます。

図 2.13 は 8 次バタワース・ロー・パス・フィルタの周波数特性です。実線が実測値、破線が計算値です。電圧増幅度 G_v は、しゃ断域で、 -160 [dB/dec] です。また、 f_c にて通過域より 3 dB 低下しています。位相差 $\psi_{v_{out8}-v_{in}}$ は、計算値では、 f_c にて -360° です。ナイキスト周波数 f_{nyq} に近いところで、 $\psi_{v_{out8}-v_{in}}$ の振動がさらに大きくなっています。8 次フィルタでは、しゃ断域での v_{out8} の減衰がとても大きいからです。

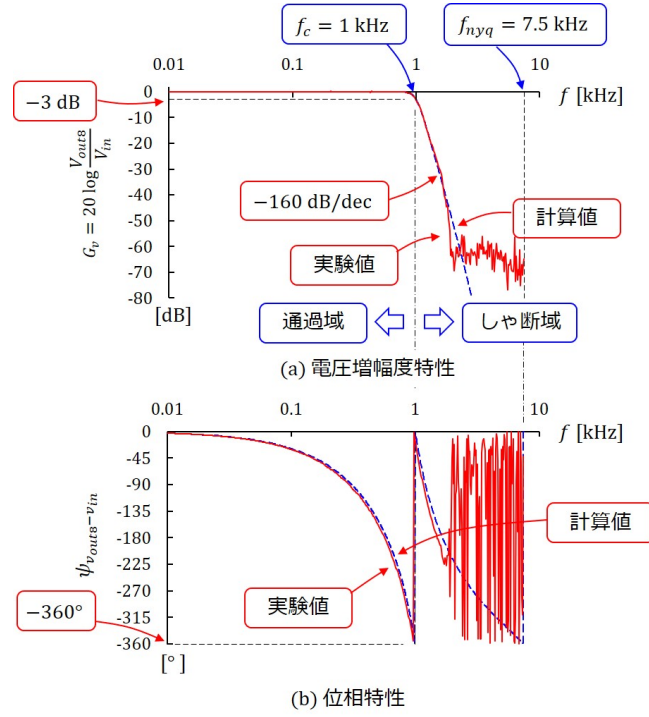


図 2.13: 8次バタワース・ロー・パス・フィルタの周波数特性

計算値は次式によります. 8次バタワース・ロー・パス・フィルタの伝達関数 $H_8(z)$ は, (2.13) 式の伝達関数 $HE_k(z)$ を用いて,

$$H_8(z) = \prod_{m=2, 4, \dots}^8 HE_m(z) \quad (2.18)$$

と表されます. 2次フィルタの4段縦続接続により得られます. $HE_m(z)$ において, $N = 8$ です.

一般に, N 次バタワース・ロー・パス・フィルタの伝達関数 $H_N(z)$ は,

(1) N が奇数の時

$$H_N(z) = H_1(z) \prod_{m=2, 4, \dots, N-1} HE_m(z) \quad (2.19)$$

(2) N が偶数の時

$$H_N(z) = \prod_{m=2, 4, \dots, N} HE_m(z) \quad (2.20)$$

となります.

第3章 FIR フィルタ

3.1 21 タップ・ロー・パス・フィルタ

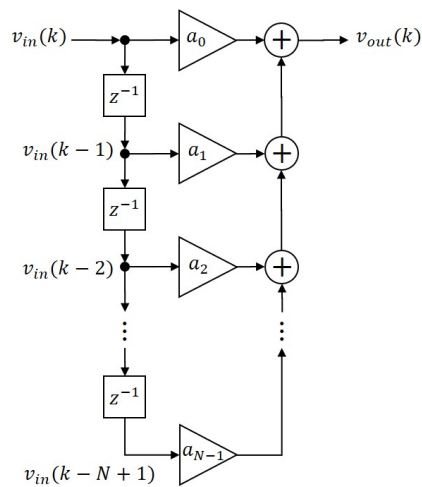


図 3.1: N タップ FIR フィルタの構成

図 3.1 は N タップ FIR フィルタの構成です。この図の表す入出力関係は

$$v_{out}(k) = \sum_{m=0}^{N-1} a_m v_{in}(k-m) \quad (3.1)$$

です。 a_m ($m = 0, \dots, N-1$) は **フィルタ係数** です

図 3.2 は **21 タップ FIR ロー・パス・フィルタ** の周波数特性です。 $N = 21$ で **ハニング窓** を適用しました。実線が実験値、破線が計算値です。電圧増幅度 G_v の計算値は実験値とよく合いました。一方、位相差 $\psi_{v_{out}-v_{in}}$ の計算値は実験値と少しずれました。

計算値は、次の **伝達関数** $H_N(z)$ を元に得ました。

$$H_N(z) = \sum_{m=0}^{N-1} a_j z^{-m} \quad (3.2)$$

上式にて、 $z = e^{j\omega T_s}$ と置き換えて、実部と虚部をそれぞれまとめることで、

$$\begin{aligned} G_v(f) &= 20 \log_{10} \sqrt{(RP)^2 + (IP)^2} \\ \psi_{v_{out}-v_{in}}(f) &= \tan^{-1} \left(\frac{IP}{RP} \right). \end{aligned} \quad (3.3)$$

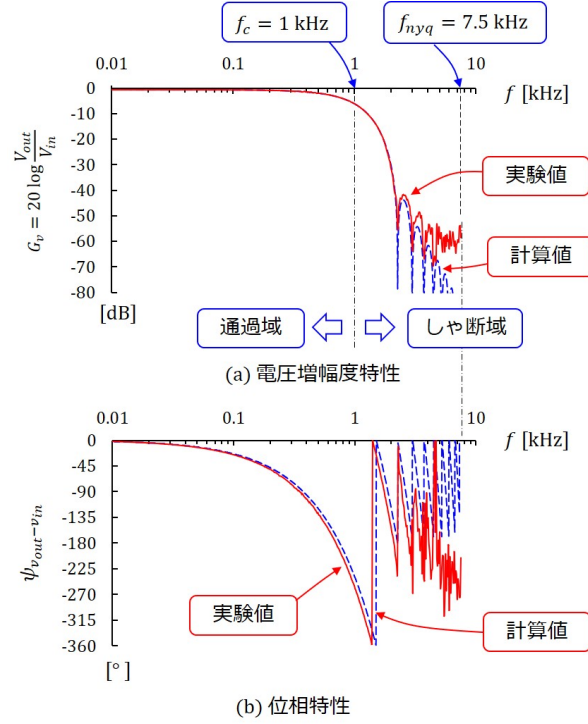


図 3.2: 21 タップ FIR ロー・パス・フィルタの周波数特性

ただし,

$$\begin{aligned}
 RP &= \sum_{m=0}^{N-1} a_m \cos m\omega T_s \\
 IP &= \sum_{m=0}^{N-1} a_m \sin m\omega T_s
 \end{aligned} \tag{3.4}$$

と求められます. T_s はサンプリング周期です.

タップ数 N を奇数とします. フィルタ係数 a_m ($m = 0, 1, \dots, N-1$) は**理想フィルタ** (電圧増幅度 $A_v (= V_{out}/V_{in})$ が, 通過域で 1 倍, 阻止域で 0 倍であり, カットオフ周波数 f_c を境にして A_v がステップ的に変化する特性を持つフィルタ) の**インパルス応答** $H_{IR}(m)$ ($m = 0, 1, \dots, N-1$) は

$$H_{IR}(m) = \frac{\Omega_c}{\pi} \times \frac{\sin \left(m - \frac{N-1}{2} \right) \Omega_c}{\left(m - \frac{N-1}{2} \right) \Omega_c} \tag{3.5}$$

です. ただし, $\Omega_c = 2\pi f_c T_s$ に, ハニング窓 $W_{Hann}(m)$ ($m = 0, 1, \dots, N-1$)

$$W_{Hann}(m) = \frac{1}{2} \left\{ 1 - \cos \left(\frac{2\pi m}{N-1} \right) \right\} \tag{3.6}$$

をかけて

$$a_m = H_{IR}(m) W_{Hann}(m) \tag{3.7}$$

により得ました.

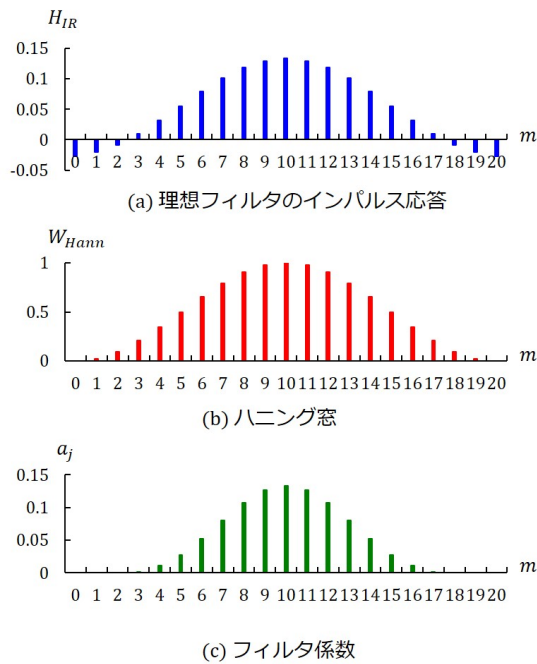


図 3.3: 21 タップ FIR フィルタ係数の計算過程

図 3.3 は 21 タップ ($N = 21$) の場合の FIR フィルタ計数の計算過程を示します. 上から (a) 理想フィルタのインパルス応答, (b) ハニング窓, (c) フィルタ係数です. $f_c = 1$ [kHz], $T_s = 66.7$ [μ s] です.

3.2 サンプリングと処理時間による位相遅れ

図 3.2(b) の位相特性における計算値と実験値のずれは, (3.3) 式がサンプリングによる位相遅れ, マイコンの処理時間による位相遅れを含まないことが原因です.

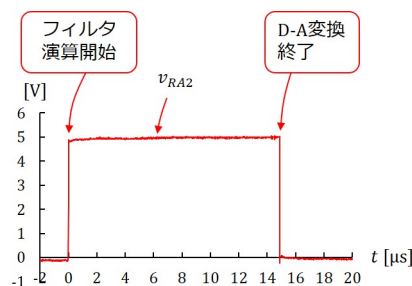


図 3.4: フィルタ・プログラム処理時間モニタ電圧の波形例
(21 タップ FIR ロー・パス・フィルタ)

図 3.4 はフィルタ・プログラム処理時間モニタ電圧 v_{RA2} の波形例です。これより、21 タップ FIR フィルタの場合、フィルタ演算と D-A 変換に $14.9 \mu s$ を要しました。サンプリングによる遅れ $33.3 [\mu s]$ と、A-D 変換に要する時間 $2.2 [\mu s]$ と合わせると、総計 $50.4 [\mu s]$ の遅れがあります。

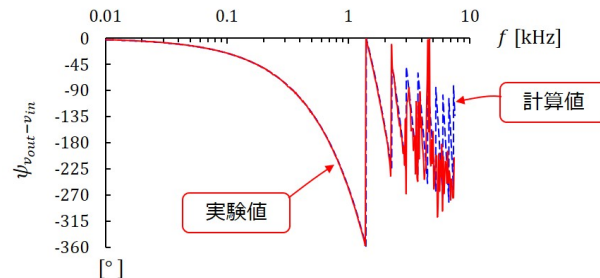


図 3.5: 21 タップ FIR ロー・パス・フィルタの位相特性
(計算値はサンプリング、マイコン内の処理による位相遅れを含む)

図 3.5 はサンプリングによる位相遅れ、マイコン内の処理による位相遅れを含む場合の位相特性です。図 3.2(b) の計算値の実験値に対するずれが解消されました。

ただし、後述の 91 タップ FIR フィルタにこの位相遅れ補償は含まれていません。

3.3 FIR フィルタの特徴

FIR フィルタの特徴は**直線位相特性**です。位相差 $\psi_{v_{out}-v_{in}}$ が周波数 f の一次関数となることを直線位相特性と呼びます。

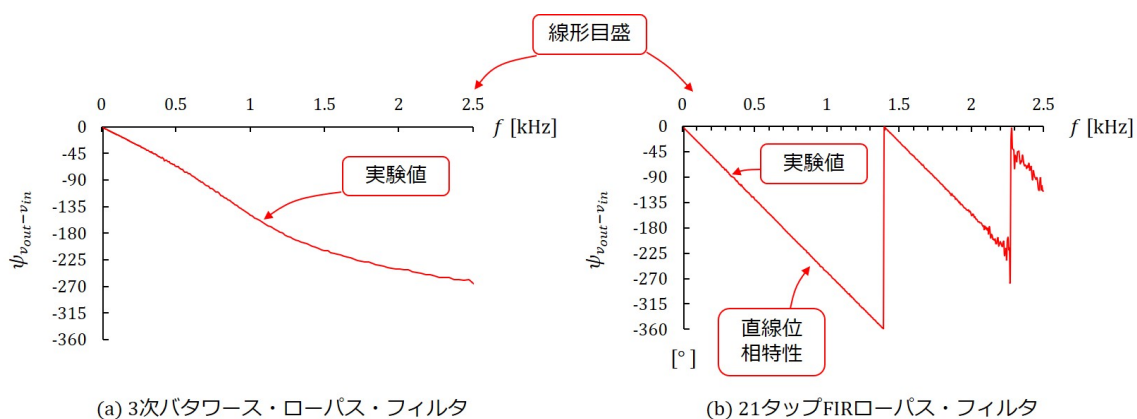


図 3.6: IIR フィルタと FIR フィルタの位相特性比較 (線形目盛)

図 3.6 は IIR フィルタと FIR フィルタの位相特性を比較して示します。(a) は図 2.11(b) の 3 次バタワース・ロー・パス・フィルタの位相特性です。(b) は図 3.2(b) の 21 タップ FIR

ロー・パス・フィルタの位相特性です. 横軸は, 対数目盛ではなく, 線形目盛です. いずれも実験値です. FIR フィルタでは $\psi_{v_{out}-v_{in}}$ が直線になっています.

3.4 91 タップ・ロー・パス・フィルタ

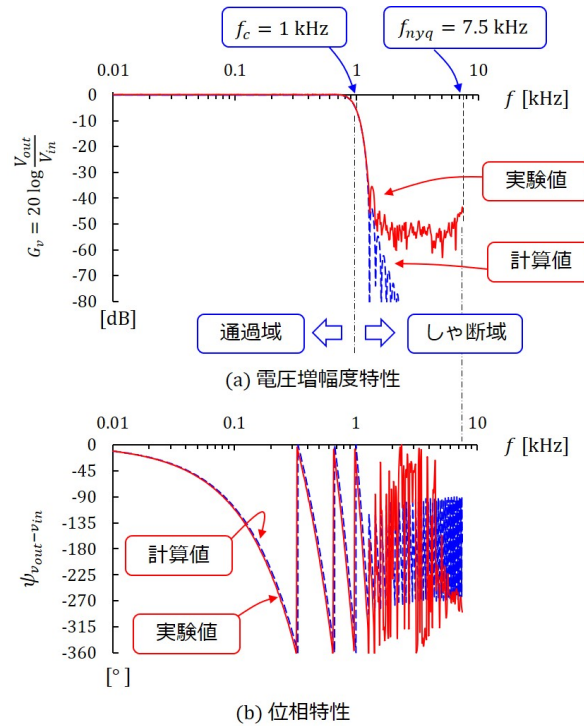


図 3.7: 91 タップ FIR ロー・パス・フィルタの周波数特性

図 3.7 は 91 タップ FIR ロー・パス・フィルタの周波数特性です. タップ数を増やすことで, カットオフ周波数 f_c を超えたところで電圧増幅度 G_v を急峻に低下させられます.

3.5 DSP 関数 (VectorCopy, VectorDotProduct) によるフィルタ演算の高速化

図 3.8 はフィルタ・プログラム処理時間モニタ電圧 v_{RA2} の波形例です. (a) は (3.1) 式のかかけ算に C 言語の $*$ 演算子を用いた場合です. (b) は dsPIC マイコン用の DSP 関数 (VectorDotProduct, VectorCopy) を使用した場合です.

図 3.8(a) の $*$ 演算子を用いた場合では, フィルタ演算と D-A 変換に $61.9 [\mu s]$ を要しました. サンプル周波数 $f_s = 15 [\text{kHz}]$ の場合, サンプル周期 $T_s = 66.7 [\mu s]$ です. この T_s の大部分をフィルタ演算が占めていました.

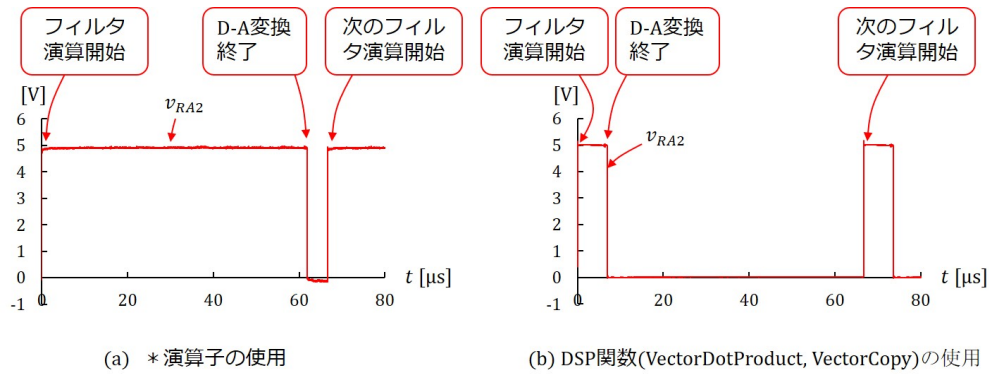


図 3.8: フィルタ・プログラム処理時間モニタ電圧の波形例
(91 タップ FIR ロー・パス・フィルタ)

91 タップ FIR フィルタでは、(3.1) 式の演算でかけ算を 91 回実行します。このかけ算に多くの処理時間を要します。

dsPIC マイコンでは、かけ算を高速に実行する関数（DSP 関数と呼ばれます）が用意されています。ベクトルの内積を求める VectorDotProduct 関数、ベクトルをコピーする VectorCopy 関数などです。図 3.1 において、係数ベクトル $\mathbf{A} = (a_0, a_1, \dots, a_{N-1})^T$ ，入力電圧ベクトル $\mathbf{V}_{in} = (v_{in}(k), v_{in}(k-1), \dots, v_{in}(k-N+1))^T$ と定義すると、(3.1) 式は、次のように、ベクトルの内積で表せます。

$$v_{out}(k) = \mathbf{A}^T \mathbf{V}_{in} \quad (3.8)$$

VectorDotProduct 関数はこの内積演算を高速実行します。

また、FIR フィルタでは、1 サンプル毎に入力電圧ベクトル $\mathbf{V}_{in}$ の各要素を 1 サンプルずつシフトさせます。この処理は、2 つのベクトルを用意して、VectorCopy 関数を適用することで、高速化を図れます。電圧値のシフトは、一方のベクトルの開始番地を 1 番地ずらして指定することで、実現できます。詳しくは図 4.14 を参照してください。図 3.8(b) の DSP 関数を用いた場合では、フィルタ演算と D-A 変換を 6.9 [μs] で実行できました。

第4章 プログラム

本書で解説するソースコードは、

<https://mybook-pub-site.sakura.ne.jp/dsPIC33EV256GM102/>

の圧縮フォルダ「Files_for_BBLabo」に入れてあります。この圧縮フォルダをダウンロード・解凍し、同フォルダ内の「digital_filter」フォルダをユーザー・アカウントフォルダ内のMPLABXProjects フォルダに置いてください。

4.1 ブロック図

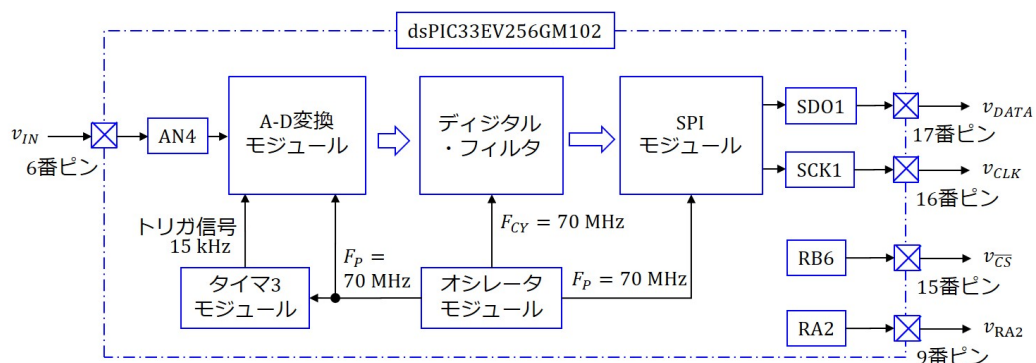


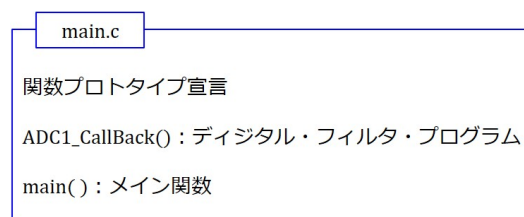
図 4.1: デジタル・フィルタ・プログラムのブロック図

図 4.1 はデジタル・フィルタ・プログラムのブロック図です。入力電圧 v_{IN} を 6 番ピン (AN4) から取り込み、A-D 変換モジュールにて A-D 変換を行います。そして、デジタル・フィルタにてフィルタリング処理を行います。D-A コンバータへのデータ転送は、SPI モジュールによります。17 番ピン (SDO1) がデータ出力で、16 番ピン (SCK1) がクロック出力です。そして、15 番ピン (RB6) は、D-A コンバータ側の SPI モジュールを選択するためのデジタル信号 $\overline{CS}$ の出力です。

9 番ピン (RA2) はフィルタ・プログラムの処理時間モニタ用電圧の出力設定です。

A-D 変換モジュールには、タイマ 3 より、15 kHz ごとにトリガ信号が送られます。A-D 変換のサンプリング周波数は 15 kHz です。

オシレータ・モジュールは 70MHz のクロックを生成します。データ・シートでは CPU に供給するクロックは F_{CY} 、周辺モジュールに供給するクロックは F_P と表記されています。



図E2-2 デジタル・フィルタ (main.c内のプログラムの内訳)

図 4.2: デジタル・フィルタ (main.c 内のプログラムの内訳)

図 4.2 は main.c ファイル内のプログラムです。同プログラムは、大きく分けて (1) 関数プロトタイプ宣言、(2) デジタル・フィルタ・プログラム (ADC1_CallBack() 関数: A-D 変換モジュールからの割り込みの処理関数)、(3) メイン関数 (main() 関数) からなります。

4.2 1 次バタワース・ロー・パス・フィルタ

digital_filter フォルダ内の 1st_order.X プロジェクトを開いてください。MPLABX の立ち上げとプロジェクトの開き方は付録 C.1 節を参照してください。

4.2.1 フィルタ係数

```

//フィルタ係数宣言
#define b10 Q15(0.175296)
#define b11 Q15(0.175296)
#define a11 Q15(0.649408)
  
```

図 4.3: 1 次バタワース・ロー・パス・フィルタ係数のヘッダ・ファイル (fil_coeff.h)

図 4.3 は、1 次バタワース・ロー・パス・フィルタのフィルタ係数を宣言しているヘッダ・ファイルです。fil_coeff.h ファイル内のスクリプトです。フィルタ係数を (2.6) 式により求めました。同式にて、カットオフ周波数 $f_c = 1$ [kHz], サンプルング周期 $T_s = 66.7$ [μ s] です。

Q15() 関数は、小数を 16 ビットの符号付き**固定小数点数**に変換する関数です。16 ビットの整数値の最上位ビットを符号ビットとし、残りの 15 ビットの整数で小数の代替表現を行います。

Q15() 関数は dsp.h ファイル内で定義されています。そこで、main.c ファイルで

```
#include <dsp.h>
```

により, dsp.h ファイルを読み込みます.

● Q15 フォーマット

Q15() 関数の実際の演算は, 小数に 2^{15} をかけています. 例えば, 0.1 は

$$Q15(0.1) = \text{round}(0.1 \times 2^{15}) = 3277 \quad (4.1)$$

です. **round()** は小数点以下を**四捨五入**する関数です. 2^{15} 倍した値なので, 後で $1/2^{15}$ 倍すれば, 元の値にもどります. 小数と整数のかけ算は, 例えば 0.1 と 10000 のかけ算の場合,

$$\frac{Q15(0.1) \times 10000}{2^{15}} = \frac{3277 \times 10000}{2^{15}} = 1000 \quad (4.2)$$

と得られます. 2^{15} で割ることは, 分子のかけ算結果を 15 ビット右シフトすることでできます. 小数と整数のかけ算が, 整数同士のかけ算とほとんど変わらない処理時間で実現できます.

なお, 15 ビット右シフトは, 16 ビット以上の値を残して, 15 ビット以下の値を切り捨てる演算です. 切り捨てではなくて, 15 ビット以下の値を「**0 捨 1 入**」するには, かけ算結果に $0x4000 = 0b0100\ 0000\ 0000\ 0000$ を足してから, 15 ビット右シフトします.

4.2.2 1次バタワース・ロー・パス・フィルタ・プログラム

```

#include "mcc_generated_files/system.h"
#include <dsp.h>
#include "fil_coeff.h"

//関数プロトタイプ宣言
void SPI1_Transmit( uint16_t);

void __attribute__((weak)) ADC1_CallBack(void)
{
    static int32_t vout[2], vin[2];

    _RA2 = 1;                                     // RA2(9番ピン)をHigh
                                                    // フィルタ・プログラム処理時間モニタ用

    // 1次フィルタ
    vout[1] = vout[0];
    vin[1] = vin[0];
    vin[0] = (int32_t)ADC1BUF0;                    // A-D変換結果読み出し
    vout[0] = (b10*vin[0] + b11*vin[1] + a11*vout[1])>>15;
                                                    // 固定小数点数と
                                                    // 整数のかけ算

    SPI1_Transmit(vout[0] | 0x3000);                // SPIにデータ転送
                                                    // 制御データ

    _RA2 = 0;                                     // RA2(9番ピン)をLow
}

```

図 4.4 のコードには、以下の注釈が追加されています：

- Adc1.cファイルから移動**：`void __attribute__((weak)) ADC1_CallBack(void)` の行に注釈が付けられています。
- フィルタ演算**：`vout[1] = vout[0];`、`vin[1] = vin[0];`、`vin[0] = (int32_t)ADC1BUF0;`、`vout[0] = (b10*vin[0] + b11*vin[1] + a11*vout[1])>>15;` の4行に括弧で囲まれた注釈が付けられています。
- 固定小数点数と整数のかけ算**：`vout[0] = (b10*vin[0] + b11*vin[1] + a11*vout[1])>>15;` の行に注釈が付けられています。
- 制御データ**：`SPI1_Transmit(vout[0] | 0x3000);` の行に注釈が付けられています。

図 4.4: 1次バタワース・ロー・パス・フィルタのプログラム
(ADC1 割り込み処理関数 (main.c))

図 4.4 は main.c ファイル内に記述した 1 次バタワース・ロー・パス・フィルタのプログラムです。C.4 節の周辺モジュール用コードの自動生成により、adc1.c ファイル内に **ADC1_CallBack() 関数** が作られています。この関数が、A-D 変換終了時の割り込みにより起動されます。そこで、この **ADC1 割り込み処理関数** を main.c ファイル内に移動させて、同関数内にフィルタ・プログラムを記述しました。

先頭の

```
_RA2 = 1;
```

により、RA2(9 番ピン)に“High”を出力し、最後の

```
_RA2 = 0;
```

により、“Low”を出力します。この出力により、フィルタ・プログラムの処理時間をモニタできます。

このモニタ信号出力スクリプトの内側に、図 2.1 の演算を実行するプログラムを記述します。

```
vout[1] = vout[0];
```

```
vin[1] = vin[0];
```

により、 z^{-1} の **1 サンプル・シフト演算**を行います。

```
vin[0] = (int32_t)ADC1BUF0;
```

は、A-D 変換結果を vin[0] に格納します。ADC1BUF0 は 16 ビットレジスタなので、次のかけ算に備えて、**int32_t** により**符号付き 32 ビット形**に変換します。

```
vout[0] = (b10*vin[0] + b11*vin[1] + a11*vout[1]) >>15;
```

は、フィルタ係数 b10, b11, a11 と各電圧値 vin, vout との積和演算を行い、15 ビット右シフト演算を行います。これは、固定小数点数のフィルタ係数と整数の電圧値との積和演算を実行します。マイコン内での小数と整数のかけ算の詳細はコラム 4.2.1 項の Q15 フォーマットを参照してください。

```
SPI1_Transmit(vout[0] | 0x3000);
```

は、フィルタ出力電圧 vout[0] を D-A 変換 IC に送る関数です。引数の上位 4 ビット (0x3) は SPI モジュールの制御データ、下位 12 ビットが vout[0] の値です。制御データの詳細は付録 A.1 節を参照してください。

4.2.3 SPI モジュールによるデータ転送プログラム

```
// SPI ポート設定
#define SPI_CS _RB6 // RB6をD/Aコンバータ選択信号
// の出力ポートに指定

void SPI1_Transmit( uint16_t TransmitData)
{
    uint8_t delay; // 8ビット符号無し整数

    SPI_CS = 0; // D-Aコンバータを選択
    SPI1BUF = TransmitData; // 送信用バッファにデータを格納
    // → D-Aコンバータへ転送開始

    delay =1;
    while(delay--); // 送信終了まで一定時間待機
    SPI_CS = 1; // D-Aコンバータ選択解除
}
```

図 4.5: SPI モジュールによるデータ転送プログラム (spi1.c)

図 4.5 は **SPI_Transmit()** 関数です。SPI モジュールによるデータ転送プログラムです。spi1.c ファイル内に

```
#define SPI_CS _RB6
```

により、RB6(15 番ピン) が $\overline{CS}$ 信号出力ピンであると宣言します。

```
SPI_CS = 0;
```

は、CS 信号として 0 を出力し、15 番ピンにつながる D-A コンバータ IC (図 1.1, 図 4.1, A.2.1 参照) を選択します。

```
SPI1BUF = TransmitData;
```

は、**SPI1BUF レジスタ** にデータを格納します。これにより、SPI モジュールは D-A コンバータ IC に格納されたデータ転送を開始します。

```
delay =1;
```

```
while(delay--);
```

```
SPI_CS = 1;
```

により、送信終了まで一定時間待機した後に、CS 信号を 1 にして、D-A 変換 IC の選択を解除します。

4.3 2次バタワース・ロー・パス・フィルタ

digital_filter フォルダ内の 2nd_order.X プロジェクトを開いてください。

4.3.1 フィルタ係数



```
//フィルタ係数宣言
#define b20 Q15(0.033572/2) // 1より大きな値があるので、
                             //全部1/2倍して、最後に結果を2倍する。
#define b21 Q15(0.067144/2)
#define b22 Q15(0.033572/2)
#define a21 Q15(1.418983/2)
#define a22 Q15(-0.553270/2)
```

図 4.6: 2次バタワース・ロー・パス・フィルタ係数のヘッダ・ファイル (fil_coeff.h)

図 4.6 は、2次バタワース・ロー・パス・フィルタのフィルタ係数を宣言しているヘッダ・ファイル (fil_coeff.h) です。フィルタ係数を (2.14) 式により求めました。カットオフ周波数 $f_c = 1$ [kHz], サンプリング周期 $T_s = 66.7$ [μ s] です。a21 が 1 より大きくなります。Q15() 関数は 1 以下の数値しか扱えないので、全係数の 1/2 倍を固定小数点変換しています。フィルタ演算の最終結果を 2 倍することで対処します。

4.3.2 2次バタワース・ロー・パス・フィルタ・プログラム



```
void __attribute__((weak)) ADC1_Callback(void)
{
    static int32_t vout2[3], vin[3];

    _RA2 = 1; // RA2(9番ピン)をHigh
              //フィルタ・プログラム処理時間モナタ用

    // 2次フィルタ
    vout2[2] = vout2[1];
    vout2[1] = vout2[0];
    vin[2] = vin[1];
    vin[1] = vin[0];
    vin[0] = (int32_t)ADC1BUF0; // A-D変換結果読み出し
    vout2[0] = (b20*vin[0] + b21*vin[1] + b22*vin[2] + a21*vout2[1] + a22*vout2[2]) >> 14;

    SPI1_Transmit(vout2[0] | 0x3000); // SPIにデータ転送

    _RA2 = 0; // RA2(9番ピン)をLow
}

// 係数を1/2倍したので、結果を2倍する。そのため、15ビットシフトではなく14ビットシフトとする。
```

図 4.7: 2次バタワース・ロー・パス・フィルタのプログラム (ADC1 割り込み処理関数 (main.c))

図 4.7 は main.c ファイル内に記述した 2 次バタワース・ロー・パス・フィルタのプログラムです。図 2.8 の演算を実行するプログラムです。フィルタ出力の `vout2[0]` の計算では、最後に 14 ビット右シフトを行います。図 4.6 のフィルタ係数の定義において全係数を $1/2$ 倍したので、フィルタ演算の結果を 2 倍します。そのため、15 ビット右シフトではなく、14 ビット右シフトを採用しています。

4.4 3次バタワース・ロー・パス・フィルタ

digital_filter フォルダ内の 3rd_order.X プロジェクトを開いてください。

4.4.1 フィルタ係数

```
//フィルタ係数宣言
#define b10 Q15(0.175296)
#define b11 Q15(0.175296)
#define a11 Q15(0.649408)
#define b20 Q15(0.035922/2)
#define b21 Q15(0.071844/2)
#define b22 Q15(0.035922/2)
#define a21 Q15(1.518314/2)
#define a22 Q15(-0.662002/2)
```

図 4.8: 3 次バタワース・ロー・パス・フィルタ係数のヘッダ・ファイル (fil_coeff.h)

図 4.8 は、3 次バタワース・ロー・パス・フィルタのフィルタ係数を宣言しているヘッダ・ファイル (fil_coeff.h) です。b10, b11, a11 は (2.6) 式により求めました。b20～a22 は、(2.14) 式において、 $m = 2$, $N = 3$ として求めました。カットオフ周波数 $f_c = 1$ [kHz], サンプリング周期 $T_s = 66.7$ [μ s] です。b20～a22 は $1/2$ 倍しています。

4.4.2 3次バターワース・ロー・パス・フィルタ・プログラム

```

void __attribute__((weak)) ADC1_CallBack(void)
{
    static int32_t vout3[3], vout1[3], vin[2];

    _RA2 = 1;                                     // RA2(9番ピン)をHigh
                                                // フィルタ・プログラム処理時間モニタ用

    // 3次フィルタ
    vout1[2] = vout1[1];
    vout1[1] = vout1[0];
    vin[1] = vin[0];
    vin[0] = (int32_t)ADC1BUF0;                  // A-D変換結果読み出し
    vout1[0] = (b10*vin[0] + b11*vin[1] + a11*vout1[1]) >> 15;  ← 1倍

    // 後段2次フィルタ
    vout3[2] = vout3[1];
    vout3[1] = vout3[0];
    vout3[0] = (b20*vout1[0] + b21*vout1[1] + b22*vout1[2] + a21*vout3[1] + a22*vout3[2]) >> 14;  ← 1/2倍
                                                // 係数を1/2倍したので、結果を2倍する。そのため
                                                // 15ビットシフトではなく14ビットシフトとする。

    SPI1_Transmit(vout3[0] | 0x3000);            // SPIにデータ転送

    _RA2 = 0;                                     // RA2(9番ピン)をLow
}

```

図 4.9: 3次バターワース・ロー・パス・フィルタのプログラム
(ADC1 割り込み処理関数 (main.c))

図 4.9 は main.c ファイル内に記述した 3 次バターワース・ロー・パス・フィルタのプログラムです。図 2.10 の演算を実行するプログラムです。b20～a22 は 1/2 倍してあるので、vout3[0] の計算の最後で 2 倍しています。

4.5 21 タップ FIR ロー・パス・フィルタ

digital_filter フォルダ内の fir_21taps.X プロジェクトを開いてください。

4.5.1 フィルタ係数

```
// FIR タップ数
#define Ntap 21

//フィルタ係数宣言
int32_t a[Ntap]=
{
    Q15(0.000000),
    Q15(-0.000509),
    Q15(-0.000790),
    Q15(0.001949),
    Q15(0.010773),
    ...
    Q15(0.000000)
};
```

21個の係数

図 4.10: 21 タップ FIR ロー・パス・フィルタ係数のヘッダ・ファイル (fil_coeff.h)

図 4.10 は、21 タップ FIR ロー・パス・フィルタのフィルタ係数 $a[j]$ を宣言しているヘッダ・ファイル (fil_coeff.h) です。 $a[j]$ は (3.7) 式により求めました。カットオフ周波数 $f_c = 1$ [kHz], サンプルング周期 $T_s = 66.7$ [μ s] です。

4.5.2 21 タップ FIR ロー・パス・フィルタ・プログラム

```
void __attribute__((weak)) ADC1_CallBack(void)
{
    int8_t i;
    static int32_t x[Ntap], y;

    _RA2 = 1; // RA2(9番ピン)をHigh
              //フィルタ・プログラム処理時間モニタ用

    //FIRフィルタ
    y = 0;
    for(i = Ntap-1; i >= 1; i--)
    {
        y = y + a[i]*x[i-1];
        x[i] = x[i-1];
    }
    x[0] = (signed long)ADC1BUF0;
    y = (y + a[0] * x[0]) >> 15;

    SPI1_Transmit(y | 0x3000); // SPIにデータ転送

    _RA2 = 0; // RA2(9番ピン)をLow
}
```

フィルタ演算

図 4.11: 21 タップ FIR ロー・パス・フィルタのプログラム (ADC1 割り込み処理関数 (main.c))

図 4.11 は main.c ファイル内に記述した 21 タップ FIR ロー・パス・フィルタのプログラムです。図 13-14 の演算を実行するプログラムです。

4.6 91 タップFIR ロー・パス・フィルタ

digital_filter フォルダ内の fir_91taps_VectorDotProduct.X プロジェクトを開いてください.

4.6.1 フィルタ係数

```
// FIR タップ数
#define Ntap 91

//フィルタ係数宣言
int32_t a[Ntap]=
{
    Q15( 0.000000 ),
    Q15( -0.000004 ),
    Q15( -0.000027 ),
    Q15( -0.000079 ),
    ...,
    Q15( -0.000004 ),
    Q15( 0.000000 )
};
```



図 4.12: 91 タップFIR ロー・パス・フィルタ係数のヘッダ・ファイル (fil_coeff.h)

図 4.12 は, 91 タップFIR ロー・パス・フィルタのフィルタ係数 $a[j]$ を宣言しているヘッダ・ファイル (fil_coeff.h) です. $a[j]$ は (3.7) 式において, $N = 91$ として求めました. カットオフ周波数 $f_c = 1$ [kHz], サンプルング周期 $T_s = 66.7$ [μ s] です.

4.6.2 91 タップ FIR ロー・パス・フィルタ・プログラム

```

void __attribute__((weak)) ADC1_CallBack(void)
{
    static uint8_t i_case;
    static int16_t x[Ntap], y[Ntap];
    static int16_t z;

    _RA2 = 1; // RA2(9番ピン)をHigh

    //FIRフィルタ
    switch(i_case & 0b1)
    {
        case 0:
            VectorCopy(Ntap - 1, y+1, x); // データの1サンプル・シフト
            y[0] = ADC1BUF0; // 入力電圧読み込み
            z = VectorDotProduct(Ntap, a, y); // 係数と入力値のベクトル内積
            SPI1_Transmit(z | 0x3000); // SPIにデータ転送
            i_case++; // インデックス更新
            break;
        case 1:
            VectorCopy(Ntap - 1, x+1, y);
            x[0] = ADC1BUF0;
            z = VectorDotProduct(Ntap, a, x); // SPIにデータ転送
            SPI1_Transmit(z | 0x3000);
            i_case++;
    }

    _RA2 = 0; // RA2(9番ピン)をLow
}

```

電圧値の1サンプル・シフト演算

係数と電圧値の内積演算

図 4.13: 91 タップ FIR ロー・パス・フィルタのプログラム
(ADC1 割り込み処理関数 (main.c))

図 4.13 は main.c ファイル内に記述した 91 タップ FIR ロー・パス・フィルタのプログラムです。DSP 関数の `VectorDotProduct` と `VectorCopy` を使用して、演算時間の短縮を実現するプログラムです。

`VectorCopy(Ntap - 1, y+1, x);`

は、`x` ベクトルを `y` ベクトルへコピーする関数です。 `Ntap - 1` はコピーする要素数、 `y+1`, `x` はポインタで、それぞれコピー先のベクトル要素の先頭アドレス、コピー元のベクトル要素の先頭アドレスです。

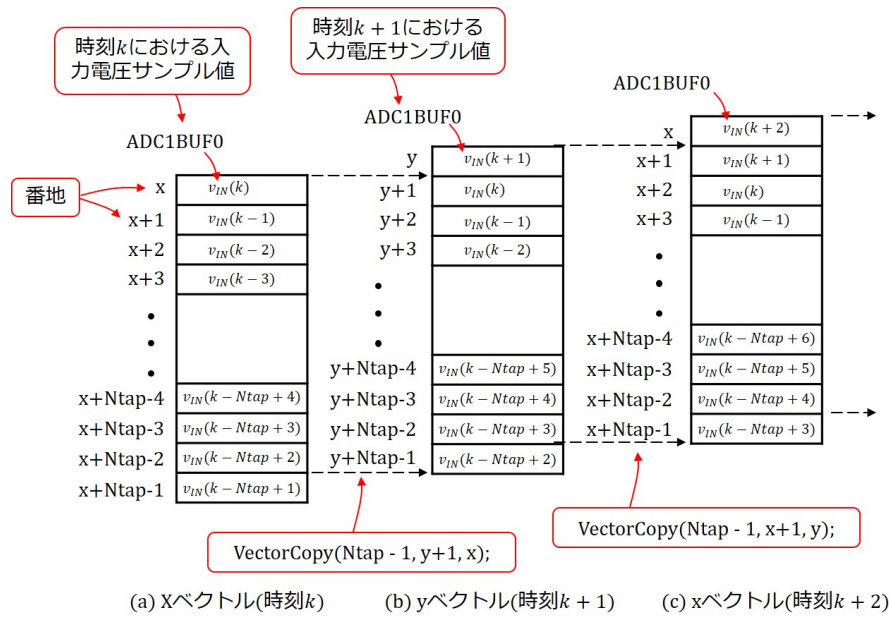


図 4.14: VectorCopy 関数による入力電圧データの1サンプル・シフト演算

図 4.14 に、VectorCopy() 関数による、1 サンプル・シフト演算の様子を示します。(a) はサンプル時刻 k における x ベクトルの状態とします。このベクトルは N_{tap} 個の要素を持ちます。時刻 k における入力電圧サンプル値 (A-D 変換結果) $v_{IN}(k)$ が ADC1BUF0 にあり、この値が x 番地のレジスタに格納されているとします。また、 $x+1$ 番地には1サンプル前の入力電圧値 $v_{IN}(k-1)$ が格納され、以下、 $x+2$ 番地に $v_{IN}(k-2)$ 、 $\dots$ 、 $x+N_{\text{tap}}-1$ 番地に $v_{IN}(k-N_{\text{tap}}+1)$ が格納されているとします。

(b) は時刻 $k+1$ における、VectorCopy() 関数が y ベクトルに x ベクトルの内容をコピーした結果を示します。図のように x 番値 $\rightarrow y+1$ 番地、 $x+1$ 番地 $\rightarrow y+2$ 番地、 $\dots$ 、 $x+N_{\text{tap}}-2$ 番地 $\rightarrow y+N_{\text{tap}}-1$ 番地と、それぞれ1番地シフトして各レジスタの内容をコピーしています。 y ベクトルの先頭レジスタには、時刻 $k+1$ の入力電圧サンプル値 $v_{IN}(k+1)$ を格納します。

(c) は、時刻 $k+2$ において、VectorCopy() 関数により、 x ベクトルに y ベクトルの内容を1サンプル・シフトしてコピーした結果を示します。

以上の x ベクトル $\rightarrow y$ ベクトル、 y ベクトル $\rightarrow x$ ベクトルのコピーを繰り返すことで、図 3.1 の入力電圧 v_{IN} の毎サンプルにおけるシフト演算を実現します。

$z = \text{VectorDotProduct}(N_{\text{tap}}, a, x);$

は、 a ベクトルと x ベクトルの内積を計算する関数です。 N_{tap} は内積に使用する要素数、 a, x はポインタで、各ベクトル要素の先頭アドレスです。

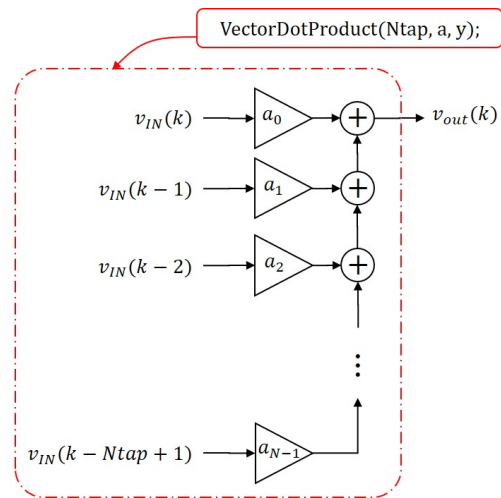


図 4.15: VectorDotProduct 関数による電圧ベクトルと係数ベクトルの内積演算

図 4.15 は VectorDotProduct() 関数が行っている電圧ベクトル $\mathbf{v} = (v_{IN}(k), v_{IN}(k-1), \dots, v_{IN}(k - Ntap + 1))^T$ と係数ベクトル $\mathbf{a} = (a_0, a_1, \dots, a_{N-1})^T$ の内積演算を示します。

付 録 A 部品

A.1 部品表

表 A.1: 部品表

(令和7年10月時点)

部品名		型式	個数	購入先
D/Aコンバータ		MCP4921-E/P	1	秋月電子通商
USB計測器(アクティブラーニングモジュール)		ADALM2000	1	Digikey, チップワ ンストップなど
インサーキットデバッガ/プログラマ		MPLAB SNAP	1	秋月電子通商
コンデンサ	積層セラミックコンデンサー	10 μ F 50V(10個入)	1	"
ジャンパーワイヤ		オス-オス セット 60本以上	1	"
抵抗	カーボン抵抗	10k Ω 1/4W(100本入り)	1	"
電線 (ジャンパ線用)	すずめっき軟銅単線	ETFE電線バック 6色 11m 導体径 0.51mm	1	"
ピンヘッダ		L型 1 $\times$ 6 (6P)	1	"
ピンソケット		メス 1 $\times$ 6 (6P)	1	"
ブレッドボード		EIC-801	1	"
マイコン		dsPIC33EV256GM102-I/SP	1	"

表 A.1 は本稿で使用する部品のリストです。上からアルファベット順、次に五十音順に並べてあります。各部品の外観、記号などの詳細は次節を参照してください。令和7年10月時点の情報です。

A.2 部品

A.2.1 D-A コンバータ

図 A.1 は、**D-A コンバータ IC**(MCP4921)の外観とピン配置です。12ビットのD-Aコンバータを内蔵しています。8ピンのDIP形です。

1番ピンがプラス電源VDD, 7番ピンがマイナス電源AVSS(通常はグラウンド)です。定格電圧は2.7~5.5[V]です。ADALM2000のV+ピンから、直流電圧源5Vが使えます。

- **SCK, SDI**: SPI通信用のピンです。

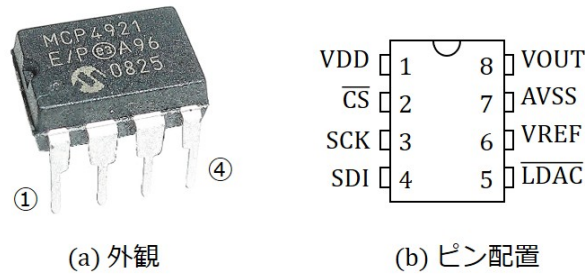


図 A.1: D-A コンバータ (MCP4921)

- **CS** : D-A コンバータ選定用ピンです。データ転送開始前に Low にして、終了後に High にします。
- **LDAC** : D-A 変換実行用のピンです。SPI 通信によるデータ転送終了後に、Low 信号を入力することで、D-A 変換を実行させます。 $\overline{LDAC}$ に常に Low を入力しておくと、 $\overline{CS}$ 信号の（データ転送終了時の）立ち上がりエッジで D-A 変換を実行させられます。
- **VREF** : 基準電圧入力用のピンです。このピンに与える基準電圧を V_{REF} とすると、D-A コンバータの出力電圧 V_{OUT} は、

$$V_{OUT} = V_{REF} \times GA \times \frac{D}{2^n} \quad (A.1)$$

です。 D は SPI 通信により送られてくる電圧値、 n はビット数（8 or 12 ビット）、 GA はゲイン（1 or 2）です。本稿の使用例では、 $n = 12$ 、 $GA = 1$ としています。

- D-A 変換結果：8 番ピンの VOUT から出力されます。

SPI 通信によって送られてくるデータは 16 ビットからなります。上位 4 ビットが **SPI モジュールの制御データ**、下位 n （8 or 12）ビットが電圧値です。第 4 章の各種フィルタにおいて、上位 4 ビットの具体的数値は $0x3 = 0b0011$ です。データ・シートによると、これら上位 4 ビットの機能は、最上位ビットから順に以下の通りです。

A/B 0 : A チャンネル選択。1 : B チャンネル選択

MCP4921 には A チャンネルしかない。 $\overline{A/B} = 0$ です。

BUF 0 : V_{REF} 用入力バッファ不使用。1 : 使用

バッファ不使用で、 V_{REF} の設定可能上限値が V_{DD} V となる。バッファ使用では、 $V_{DD} - 0.4$ V となる。 V_{REF} ピンの入力インピーダンスが、不使用では $165k\Omega$ （標準値）と低くなる。使用では高くなるが、具体的数値の記載なし。 $BUF = 0$ です。

GA 0 : ゲイン 2 倍、1 : ゲイン 1 倍。 $\overline{GA} = 1$ です。

SHDN 0 : 出力バッファ停止。1 : 出力バッファ起動

停止により、出力ピンが高インピーダンスとなる。 $\overline{SHDN} = 1$ です。

A.2.2 アクティブ・ラーニングモジュール (ADALM2000)

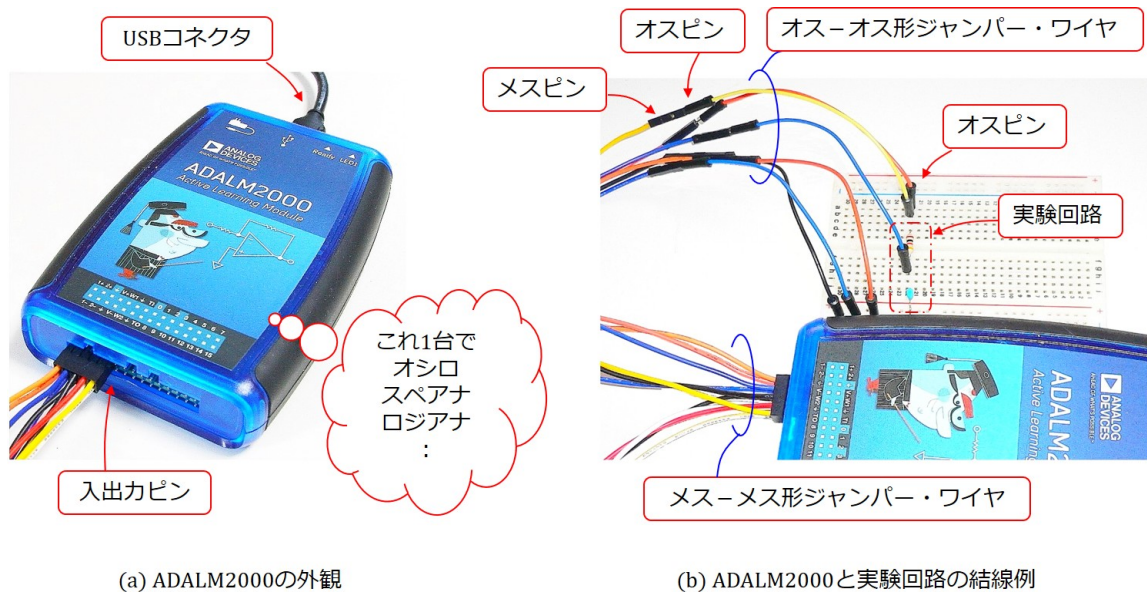


図 A.2: アクティブ・ラーニングモジュール (ADALM2000)

図 A.2 は **アクティブ・ラーニングモジュール (ADALM2000)** の外観と結線例です。

- ADALM2000 は Analog Devices 社の製品で、USB でパソコンに接続し、パソコンから操作します。
- (a) の外観写真のように、USB 給電式で、前面に 30 本の入出力ピンを備えています。
- (b) の結線例のように、ジャンパー・ワイヤを使ってブレッドボード上の実験回路と接続します。製品には 10 芯および 20 芯の専用ジャンパー・ワイヤ（メス - メス形）が付属しています。実験回路にはその先にオス - オス形ジャンパー・ワイヤを接続して使用します。

ソフトウェア「Scopy」（Analog Devices 社のサイトから無料ダウンロード可能）の設定については、付録 B にまとめてあります。Scopy はオシロスコープ、シグナルジェネレータ、直流電圧源など 9 種類の機能を備えています。

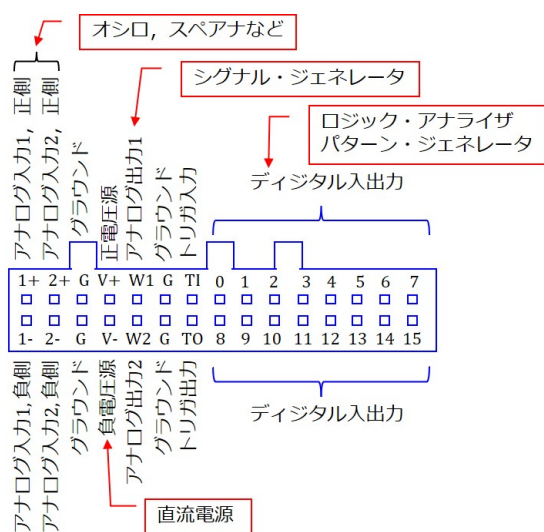


図 A.3: ADALM2000 の入出力ピン配置

図 A.3 は、ADALM2000 の入出力ピン配置を示しています。各ピンの主な用途は以下のとおりです。

- アナログ入力
オシロスコープ、スペクトルアナライザ、ネットワークアナライザ、電圧計用
- アナログ出力
シグナルジェネレータ用
- デジタル入出力
ロジックアナライザ、パターンジェネレータ用
- 正・負電圧源
直流電源用

実験回路との具体的な接続については、本製品を実際に使用する各章で説明します。

A.2.3 インサーキット・デバッガ／プログラマ

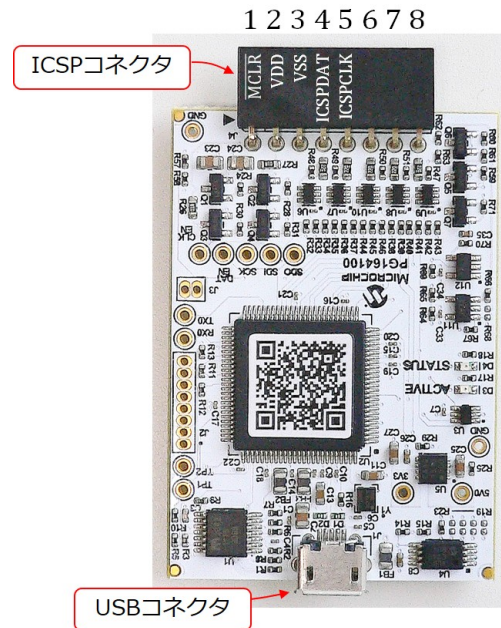


図 A.4: インサーキット・デバッガ／プログラマ (MPLAB® SNAP)

図 A.4 はインサーキット・デバッガ／プログラマ (MPLAB® SNAP) の外観です。MICROCHIP 社製です。パソコンから PIC マイコンへのプログラムの書き込みとデバッグができます。写真の上側の 8 ピンのピン・ソケットが **ICSP コネクタ** です。ソケット上面にピン配置を記してあります。

MICROCHIP 社製のインサーキット・デバッガ／プログラマには PICKit™4, PICKit™5 などがあり、SNAP と同じピン配置を持ちます。ブレッド・ボード側のピン配置を変えることなく、どのデバッガ／プログラマでも使えます。最も低価格の MPLAB® SNAP(2,200 円, 令和 7 年 10 月時点) がおすすめです。

写真の下側の **USB コネクタ** はパソコンとの通信用です。USB ケーブルでパソコンとつなぐことで、パソコン上でプログラムの作成、マイコンへのプログラムの書き込みとデバッグができます。

A.2.4 コンデンサ

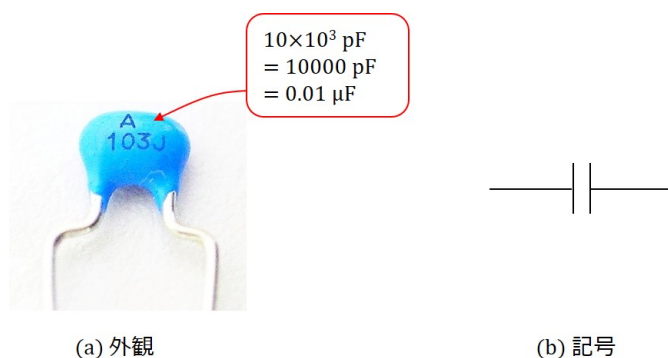


図 A.5: コンデンサ（積層セラミックコンデンサ）

図 A.5 は積層セラミック・コンデンサの外観と記号です。コンデンサの表面には静電容量が印字されています。単位は pF です。写真の例の数値 103 は

$$10 \times 10^3 [\text{pF}] = 0.01 [\mu\text{F}] \quad (\text{A.2})$$

です。

コンデンサの記号は2枚の極板を向かい合わせた形です。これは実際のコンデンサが2枚の金属板が誘電体を挟んで向かい合わせにして作られていることによります。

A.2.5 ジャンパー・ワイヤ



図 A.6: ジャンパー・ワイヤ

図 A.6 はジャンパー・ワイヤの外観です。オス－オス形ジャンパー・ワイヤです。ビニール被覆線は、より線なので、柔軟に曲げられます。ブレッド・ボード上の結線に便利です。特に、応急の回路改編に向いています。

A.2.6 抵抗

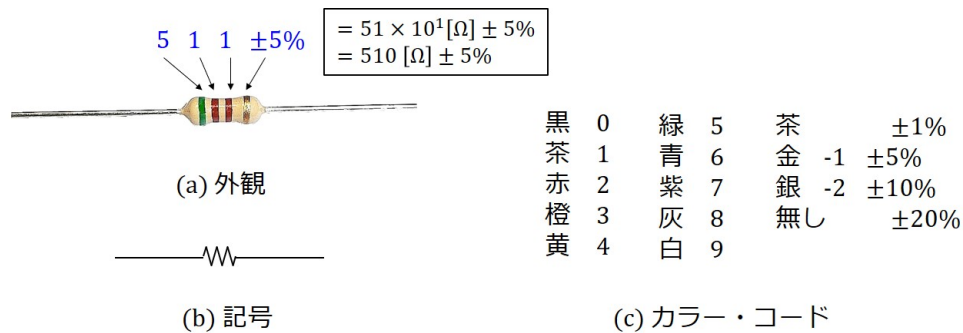


図 A.7: 抵抗（カーボン抵抗）

図 A.7 は**カーボン抵抗**の外観，記号とカラー・コード表です．写真の抵抗の定格値は 510Ω ， $1/4W$ です．抵抗値はカラー・コードで表示されています．単位は Ω です．写真のカーボン抵抗の例では，左から緑 (5)，茶 (1)，茶 (1)，金 ($\pm 5\%$) なので

$$51 \times 10^1 \pm 5\% \Omega = 510 \pm 5\% \Omega \quad (\text{A.3})$$

です．抵抗値の有効数字は 2 桁です．

A.2.7 電線

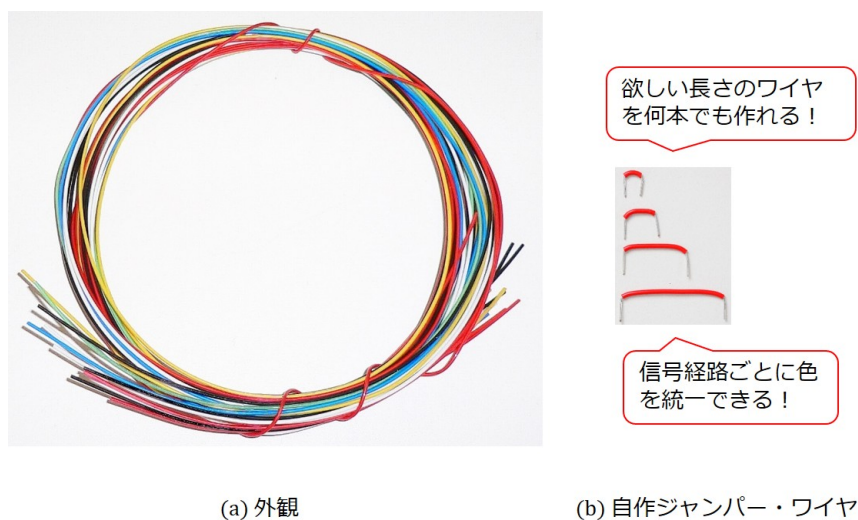


図 A.8: 電線（ジャンパー・ワイヤ用）

図 A.8 は**電線**と自作ジャンパー・ワイヤです．電線は，**すずめっき軟銅単線**の名称で販売されています．導体径 $0.51 [\text{mm}]$ なので，ブレッド・ボードへの挿入に適しています．

また、単芯タイプなので、芯がばらけません。より線タイプの電線では、細い線を束ねた芯がすぐにばらけてしまいます。

任意の長さのジャンパー・ワイヤが必要なときは、単芯電線から作ります。同図 (b) はこの電線を切って作ったジャンパー・ワイヤです。工具にニッパーとワイヤー・ストリッパーが必要です。欲しい長さのワイヤを何本でも作れる点がメリットです。さらに、長さの異なるジャンパー・ワイヤを同じ色で統一できるため、信号経路毎に色分けできます。例えば電源の+ラインは赤、グラウンド・ラインは黒等です。

A.2.8 ピン・ヘッダ, ピン・ソケット

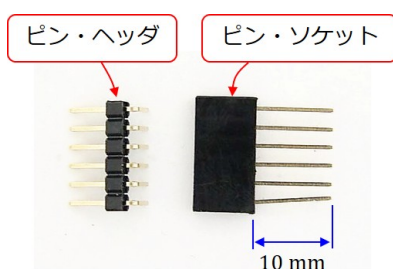


図 A.9: ピン・ヘッダとピン・ソケット

図 A.9 はピン・ヘッダとピン・ソケットの外観です。ピン・ヘッダは6ピンのL形です。ピン・ソケットは同じく6ピンでリード長が10 [mm] のものです。これらを組み合わせて、図 1.2 の配線例における ICSP コネクタを作成します。

A.2.9 ブレッド・ボード

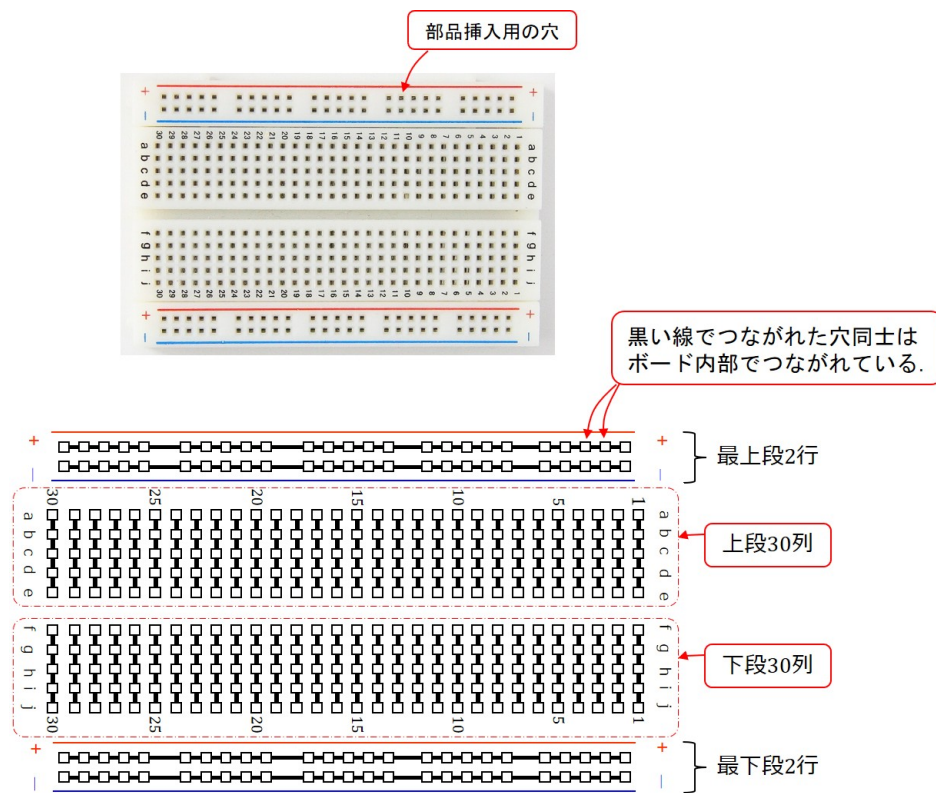
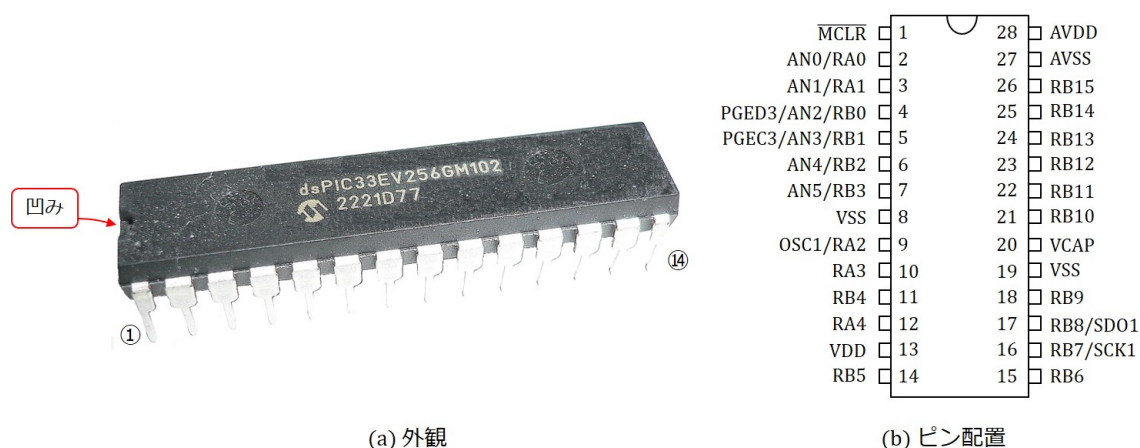


図 A.10: ブレッドボード

図 A.10 は**ブレッド・ボード** (EIC-801) の外観と内部配線を示します。ボード上に穴が 400 個あるタイプです。この穴に電子部品を差し込むことができます。

下の図はブレッド・ボード内部の配線を示します。□印が穴です。黒い線でつながれた穴同士はボード内部でつながれています。最上段の 2 行と最下段の 2 行にはそれぞれ 25 個ずつの穴があり、同じ行内の穴は全てつながっています。これらの行に挟まれて、上下 2 段 30 列の穴が並んでいます。同じ列内の 5 個ずつがつながっています。

A.2.10 マイコン



図A2-28 PICマイコン(dsPIC33EV256GM102)

図 A.11: PIC マイコン (dsPIC33EV256GM102-I/SP)

図 A.11 はマイコン (dsPIC33EV256GM102-I/SP, マイクロチップ製) の外観とピン配置です。ピン番号は、凹みを左に見て、手前左側から反時計回りに 1 → 2 → ... → 28 と付けられています。

同図 (b) はピン配置の抜粋です。各ピンには複数の機能が割り振られているので、図中には本書に関連した機能に絞って記しています。

- 13 番ピンがプラス電源 VDD, 8, 19 番ピンがマイナス電源 VSS (通常はグラウンド) です。VDD-VSS 間の定格電圧は 4.5~5.5 V です。ADALM2000 の V+ピンから直流電源 5V が使えます。
- 27, 28 番ピンは、それぞれアナログ・モジュール用のマイナス電源 AVSS (通常はグラウンド), プラス電源 AVDD です。
- 2, 3, 9, 10, 12 番ピンがポート A のピン, 4~7, 11, 14~18, 21~26 番ピンがポート B のピンです。RA0~RA4, RC0~RC15 は、デジタル値であれば、入力, 出力のどちらかを選べます。AN0~AN5 はアナログ入力の機能です。本稿では、6 番ピンの RB2 を入力設定として、対応する AN4 を選定することで、アナログ値をマイコンに取り込みます。
- 1 番ピンの MCLR と、4, 5 番ピンの PGED3, PGEC3 を ICSP 通信に使用します。
- 15 番ピンの RB6 をデジタル出力設定として、この出力ポートと、16, 17 番ピンの SCK1, SDO1 を SPI 通信に使用します。

付 録B Scopy の設定例

B.1 Scopy のインストールと立ち上げ

Scopy（ADALM2000 の無料専用ソフトウェア）の使い方を手順に沿って説明します。Scopy は以下のサイトからダウンロードできます。

<https://wiki.analog.com/university/tools/m2k/scopy>

読者のパソコンに installer をダウンロードしてください。本稿では Windows 用の scopy-v1.5.0-Winrを使用します。インストールに成功したら、ADALM2000 を付属の USB ケーブルでパソコンに接続します。



図 B.1: Scopy の立ち上げ画面

Scopy のアイコンをクリックすると図 B.1 のウィンドウが立ち上がります。ウィンドウ内に ADALM2000 のアイコンが表示されれば、Scopy が ADALM2000 を検知したことがわかります。



図 B.2: Scopy のインフォメーション・ウィンドウ

このアイコンをクリックすると図 B.2 に示すインフォメーション・ウィンドウが開きます。ウィンドウ内の Connect ボタンをクリックすることで Scopy と ADALM2000 を接続できます。



図 B.3: Scopy と ADALM2000 の接続ランプ

接続が成立すると、図 B.3 のように ADALM2000 のアイコンの下に緑色のランプが点灯します。

B.2 ネットワーク・アナライザの設定

B.2.1 基準信号源設定・信号のフィルタリング設定



図 B.4: ネットワーク・アナライザの設定画面 (その 1)

図 B.4 はネットワーク・アナライザ (Network Analyzer) の設定画面です。以下の手順で設定します。

1. Network Analyzer をクリック
2. 設定パネルボタンをクリック
3. Channel 1 を選択

Channel 1(1+, 1- 間) の電圧を v_1 , Channel 2(2+, 2- 間) の電圧を v_2 とし, それぞれの実効値を V_{1e}, V_{2e} とすると, 電圧増幅度 G_v は

$$G_v = 20 \log_{10} \frac{V_{2e}}{V_{1e}} \quad (\text{B.1})$$

と設定されます。また, 位相差は $\psi_{v_2-v_1}$ は, v_2 の v_1 に対する進み角 $[\circ]$ として計算される。

4. Amplitude を 2 Volts に設定

W1 - G ピン間出力電圧のピーク-ピーク間電圧が 2 [V] に設定されます。Scopy のネットワーク・アナライザの Amplitude はピーク-ピーク間電圧なので注意してください。

5. Offset を 2.5 Volts に設定

W1 - G ピン間の直流電圧が 2.5 [V] に設定されます。

6. DC Filtering を On に設定

v_1, v_2 から直流成分を除去して, $G_v, \psi_{v_2-v_1}$ の計算がされます.

7. Gain Mode に Low を選択

B.2.2 掃引設定

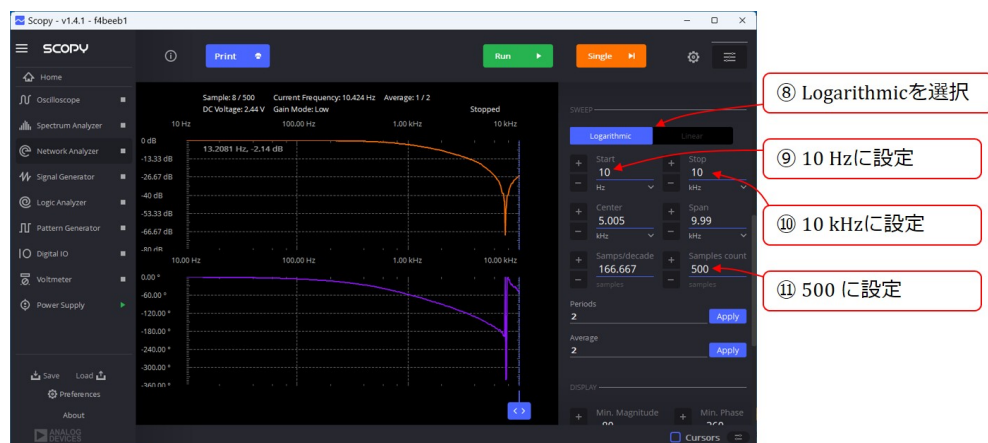


図 B.5: ネットワーク・アナライザの設定画面 (その 2)

図 B.5 はネットワーク・アナライザ (Network Analyzer) の設定画面のつづきです. 設定パネル上にカーソルを置き, パネルをスクロールした画面です. 掃引範囲を設定します.

8. Logarithmic を選択

電圧増幅度および位相差のグラフの横軸を対数表示とします.

9. Start を 10 Hz に設定

掃引開始周波数 (グラフの横軸の最小値) が 10 Hz に設定されます.

10. Stop を 10 kHz に設定

掃引停止周波数 (グラフの横軸の最大値) が 10 kHz に設定されます.

11. Samples count を 500 に設定

対数目盛の区間 ($\log_{10} 25\text{MHz} - \log_{10} 10\text{Hz}$) を 499 等分して, 10 Hz と 25 MHz の点を含む 500 個のデータが計測されます.

B.2.3 グラフ表示範囲設定

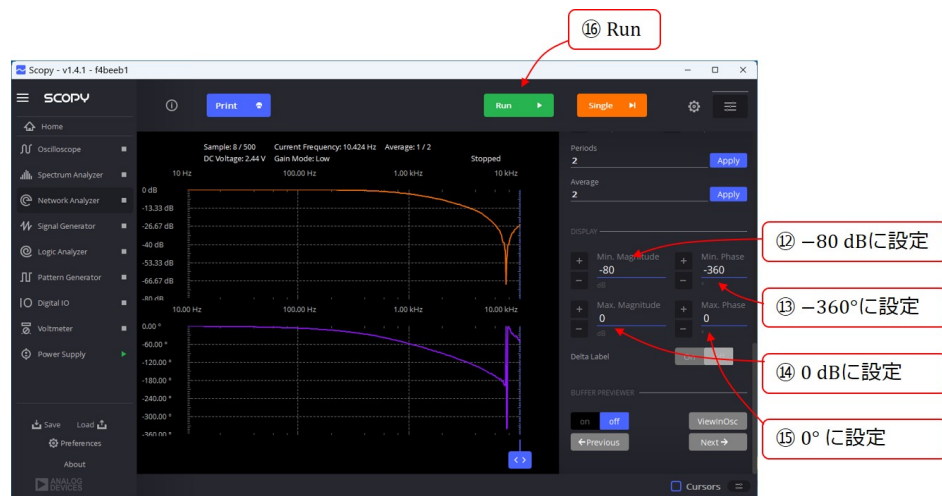


図 B.6: ネットワーク・アナライザの設定画面（その3）

図 B.6 はネットワーク・アナライザ設定画面（その3）です。

12～15. 電圧増幅度と位相差のグラフの上下限值を設定します。それぞれのグラフが画面内に適切に収まるように調整します。

16. Run をクリック

ネットワーク・アナライザが掃引を開始します。

付 録C MPLAB[®] X IDE, XC16 コンパイラ, MCC

C.1 MPLAB[®] X IDE, XC16 コンパイラ

C.1.1 ダウンロード, インストール方法

本節では Microchip 社が無償提供している統合開発環境 MPLAB[®] X IDE (Integrated Development Environment), および, MPLAB[®] XC16 コンパイラのダウンロード, インストール方法と使用方法の概要を紹介します.

MPLAB[®] X IDE のインストーラは以下の URL からダウンロードできます (令和 7 年 10 月時点).

<https://www.microchip.com/en-us/tools-resources/develop/mplab-x-ide>

インストーラ (筆者の場合は windows 版の MPLABX-v6.25-windows-installer.exe) を立ち上げ, インストーラの推奨通りに Next ボタンを押していくことで, MPLAB[®] X IDE をインストールできます. 無事インストールに成功すれば, C:\Program Files のフォルダ内に Microchip という名前のフォルダが, また, ユーザー・アカウントのフォルダ内に MPLABXProjects という名前のフォルダが作られます.

MPLAB[®] XC16 コンパイラ は以下の URL からダウンロードできます (令和 7 年 10 月時点).

<https://www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers>

インストーラ (筆者の場合は windows 版の xc16-v2.10-full-install-windows64-installer) を立ち上げ, 推奨通りに Next ボタンを押していくことで, XC16 コンパイラをインストールできます. 無事インストールに成功すると, Microchip フォルダ内に xc16 v2.10 という名前のフォルダが作られます.

C.1.2 MPLAB[®] X IDE の起動

本書で解説するソースコードは,

<https://mybook-pub-site.sakura.ne.jp/dsPIC33EV256GM102/>

の圧縮フォルダ「Files_for_BBLabo」に入れてあります。この圧縮フォルダをダウンロード・解凍し、同フォルダ内の「digital_filter」フォルダをユーザー・アカウントフォルダ内の MPLABXProjects フォルダに置いてください。

デスクトップ上に置かれた MPLAB® X IDE のアイコンを左ダブルクリックすることで、この統合開発環境を立ち上げることができます。

C.1.3 既存プロジェクトの開き方

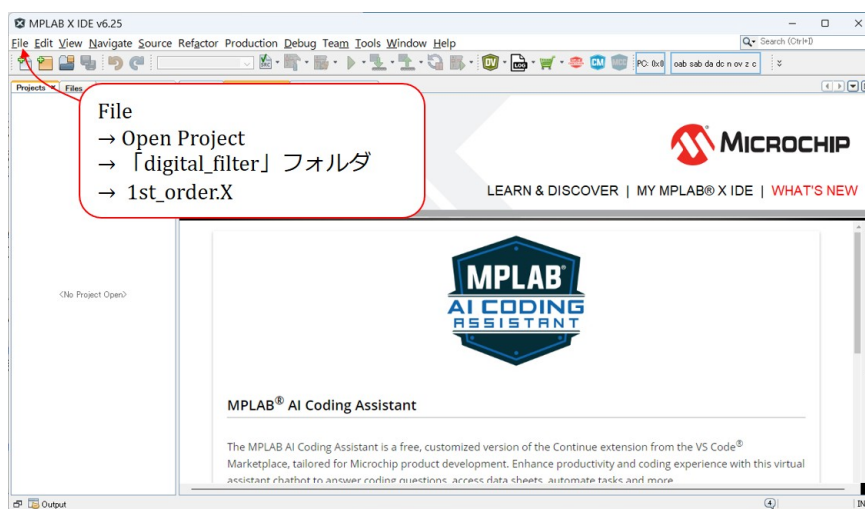


図 C.1: ダウンロードしたプロジェクトの開き方

図 C.1 の画面が立ち上がったら、

1. File → Open Project を選択します。
2. digital_filter → 1st_order.X と選択すると 1 次バタワース・ロー・パス・フィルタのプロジェクト・ツリーを開くことができます。

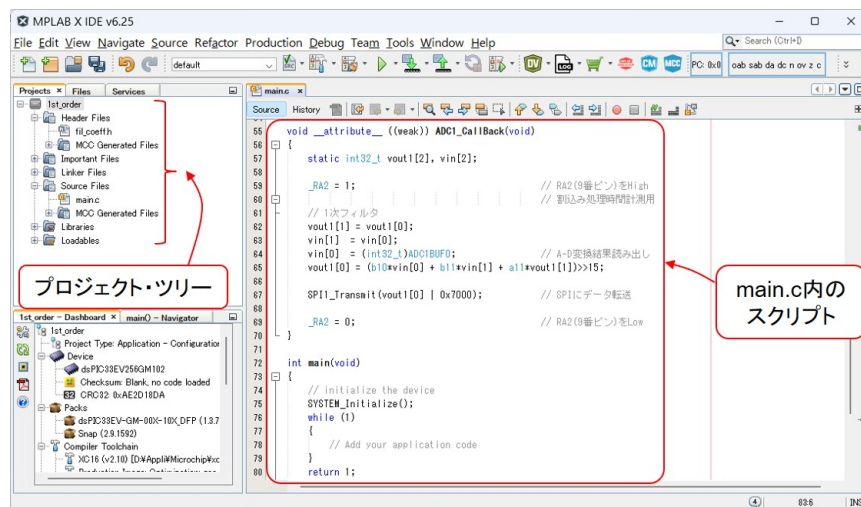


図 C.2: 開かれたプロジェクト・ツリー

このダウンロードした既成のプログラムを利用する場合は、次節～C.4 節を飛ばして、C.5 節へと進んでください。

C.2 New Project の作成方法



図 C.3: 新プロジェクトの作成（設定開始）

ゼロからプロジェクトを作成する場合は、次の手順を進めます。

1. File → **New Project** を選択

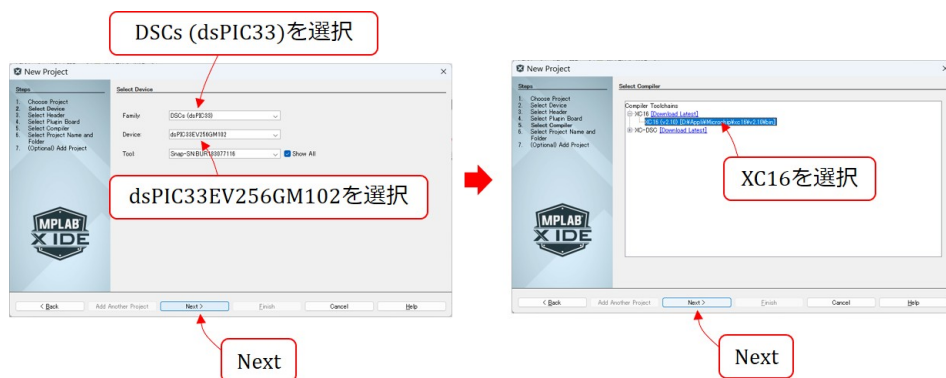


図 C.4: マイコン、コンパイラを選択

2. 図 C.5 のように進み、マイコンに dsPIC33EV256GM102 を選択し、コンパイラに XC16 を選択

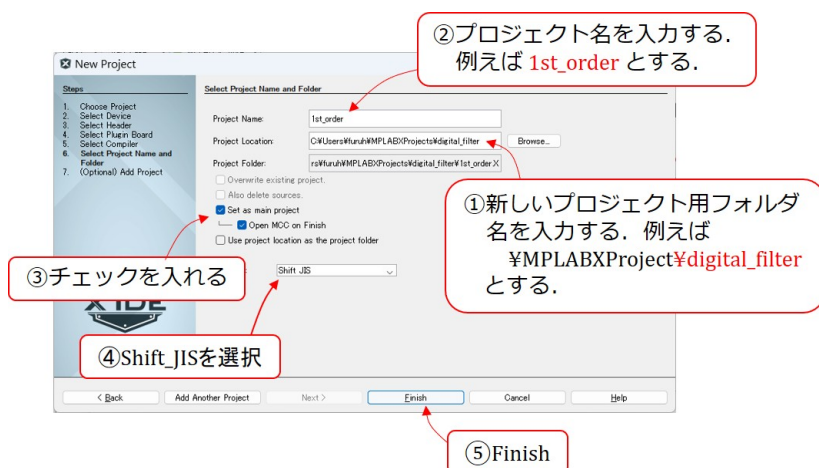


図 C.5: マイコン、コンパイラを選択

3. 新しいプロジェクト用フォルダ名を入力 (例: %MPLABProjects%digital_filter)。
4. プロジェクト名として「1st_order」を入力
5. 「Set as main project」にチェック
6. 言語に **Shift_JIS** を選択
7. Finish ボタンをクリック

これで、dsPIC33EV256GM102 を用いたプロジェクト 1st_order が作成されます。

C.3 MPLAB® Code Configurator (MCC) による周辺モジュール設定

C.3.1 MCC の起動

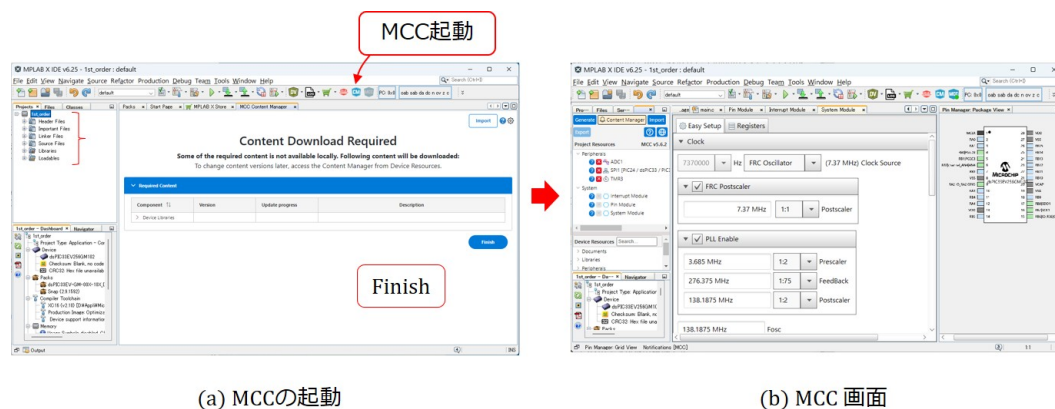


図 C.6: MCC の起動

MPLAB® Code Configurator (MCC) は、図 C.6 のようにツールバー内の MCC ボタンにより起動できます。

以下の周辺モジュール設定を行ってください。この周辺モジュールの設定は、本稿の全てのフィルタ・プログラムに共通です。

<https://mybook-pub-site.sakura.ne.jp/dsPIC33EV256GM102/index.html>

よりダウンロードした「digital_filter」フォルダ内の「***.X」プロジェクトを使用する場合は、New Project の作成、本節の周辺モジュール設定、および、以降のフィルタ・プログラムの記述は必要ありません。既に設定・記述済みです。これらのプロジェクトを開けば、直ちにプログラムのビルド、マイコンへの書き込みへと進めます。

C.3.2 System Module 設定

図 C.7 は System Module の設定画面です。Project Resources エリア内の System Module の文字を左クリックします。すると、Composer エリアに Clock 等の設定画面が開かれます。次の手順で設定を進めます。

2. FRC oscillator を選択

マイコン内蔵のオシレータ (FRC oscillator) を使用します。本稿の製作ではクロックに高い精度を必要としないので、マイコン内蔵の精度の低い ($\pm 1\%$) クロックを使います。

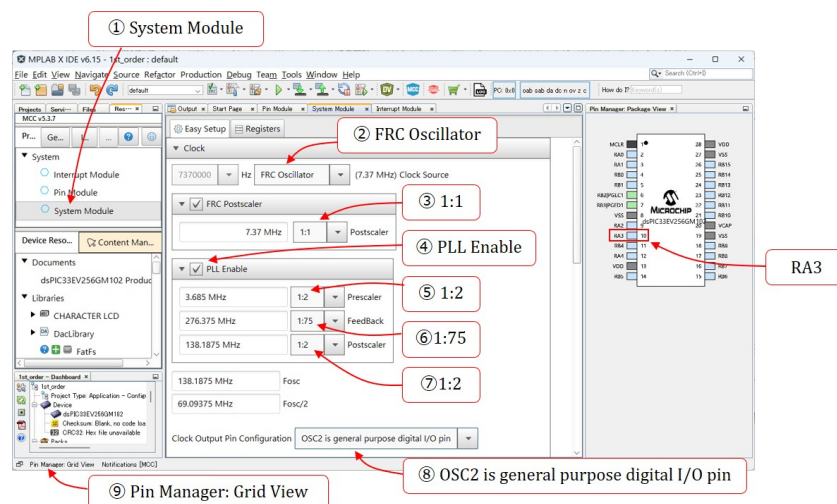


図 C.7: System Module 設定

3. 1:1 を選択

FRC Postscaler を 1:1 にします。

4. PLL Enable

PLL を起動します。FRC Postscaler の出力が PLL の入力となります。

5. 1:2

PLL の Prescaler を 1/2 に選定

6. 1:75

PLL の FeedBack を 1/75 に選定

7. 1:2

PLL の Postscaler を 1/2 に選定

次式により、PLL 出力のクロック周波数 **FOSC** が決まります。

$$\begin{aligned}
 F_{OSC} &= \frac{FRC \times Prescaler \times Postscaler}{Feedback} \\
 &= \frac{7.37[\text{MHz}] \times (1/2) \times (1/2)}{(1/75)} \approx 138 [\text{MHz}]
 \end{aligned}
 \tag{C.1}$$

CPU に供給されるクロックの周波数が **FCY**，周辺モジュールに供給されるクロックの周波数が **FP** と表記されます。

$$F_{CY} = F_P = \frac{F_{OSC}}{2} \approx 69 [\text{MHz}]
 \tag{C.2}$$

です。

8. OSC2 is general purpose digital I/O pin を選択

10 番ピンを GPIO ピンにします。Pin Manager エリアのパッケージ表示において、10 番ピンが CLKO から RA3 に変わります。また、テーブル表示において、CLKO が解錠となります。

9. Pin Manager: Grid View

Window の左下隅にある **Pin Manager: Grid View** を左クリックして、ピン・テーブルを表示します。

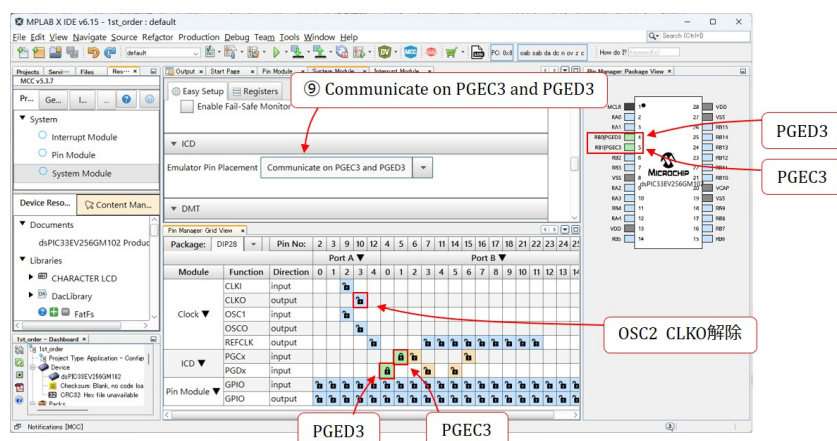


図 C.8: System Module 設定 (つづき)

図 C.8 は System Module 設定のつづきの画面です。Composer エリアを下にスクロールします。

10. Communicate on PGEC3 and PGED3 を選択

ICSP 用のピンに **PGEC3** と **PGED3** を割り当てます。パッケージ表示において、4 番ピンに PGED3, 5 番ピンに PGEC3 の文字が現れ、テーブル表示において、Pin No.4, 5 がそれぞれ PGDx, PGCx に施錠されます。

C.3.3 A-D コンバータ・モジュール設定

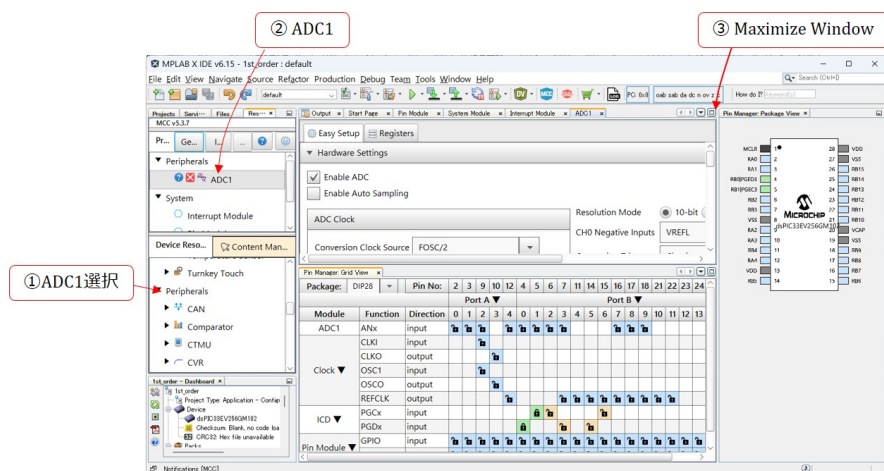


図 C.9: A-D Converter Module 設定

図 C.9 は A-D コンバータ・モジュールの設定画面です。Device Resources エリアから ADC1 を選択します。Composer エリアに ADC1 の設定項目が現れます。

3. 枠が狭いので、同エリア右上の **Maximize Window** ボタンを左クリックします。

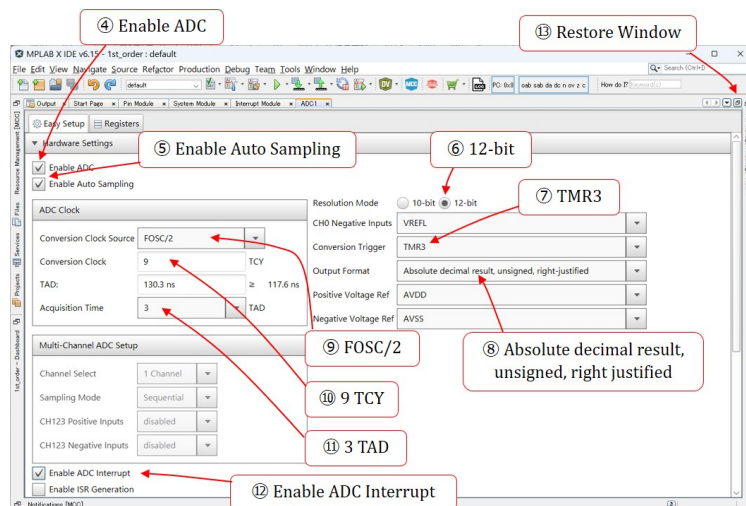


図 C.10: A-D コンバータ・モジュール設定 (つづき)

図 C.10 は A-D コンバータ・モジュールの設定画面のつづきです。

4. Enable ADC

5. Enable Auto Sampling

A-D 変換が終了すると、自動的に電圧入力を開始し、次の A-D 変換に備えます。

6. 12-bit

A-D 変換を 12 ビット・モードにします。

7. TMR3

A-D 変換のトリガをタイマ 3 より得ます。タイマ 3 がカウント・アップすると、A-D 変換モジュールでは、電圧入力を停止し、そのときの入力電圧を保持して、A-D 変換を開始します。

8. Absolute decimal result, unsigned, right justified

符号無し 12 ビットの A-D 変換結果を 16 ビットレジスタの **ADC1BUF0** に右寄せで格納する設定です。

9. FOSC/2

A-D コンバータ・モジュールのクロック源を $F_{OSC}/2 = F_{CY} \approx 69 \text{ MHz}$ とします。

10. 9 TCY

A-D コンバータ・モジュールの動作周期 **TAD** を 9TCY とします。動作クロックは $F_{CY}/9$ です。データ・シートによると、12 ビットモードでは $TAD \geq 117.6 \text{ ns}$ でなければなりません。

11. 3 TAD

入力電圧が安定するまでの時間余裕です。A-D 変換トリガがかかってから、3 TAD 後に A-D 変換を開始します。

12. Enable ADC Interrupt

A-D 変換結果を ADC1BUF0 に格納後に、A-D コンバータ・モジュールは CPU に割り込みをかけます。

13. Restore Window ボタンをクリック

Composer Window を Maximize の状態から元に戻します。

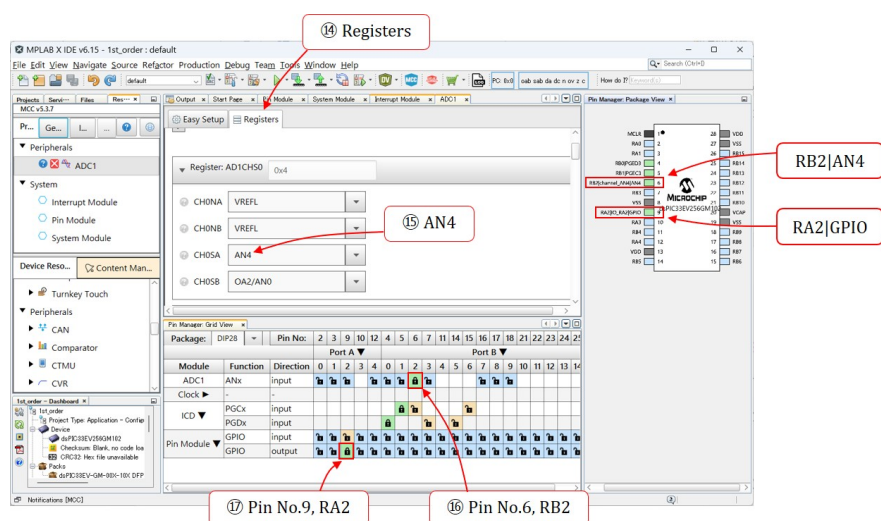


図 C.11: A-D コンバータ・モジュール設定 (つづき 2)

図 C.11 は A-D コンバータ・モジュールの設定画面のつづき 2 です。

14. Register タブをクリック

15. AN4 を選択

CH0SA チャンネルを AN4 に設定します。図 E2-6 の Easy Setup, および, 次の Pin Manager の施錠では, この設定をしてくれません。なぜ設定してくれないのかは不明です。

16. Pin No. 6, RB2 を ADC1 のアナログ入力用に施錠

パッケージ表示において, 6 番ピンに AN4 の文字が表示されます。

17. Pin No. 9, RA2 をデジタル出力用に施錠

パッケージ表示において, 9 番ピンに GPIO の文字が現れます。これは, フィルタ・プログラムの処理時間モニタ用です。

C.3.4 タイマ3 モジュール設定

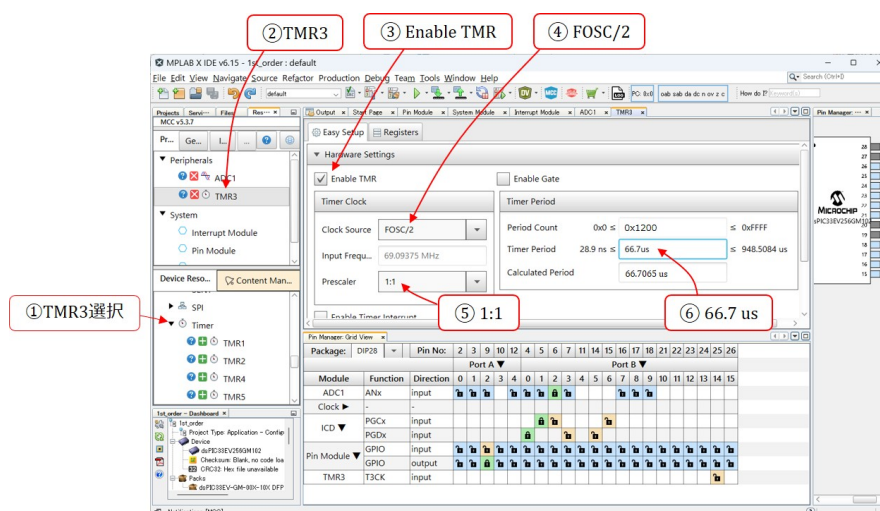


図 C.12: タイマ3・モジュール設定

図 C.12 はタイマ3 モジュールの設定画面です。A-D コンバータ・モジュール設定の (7) にて、A-D 変換開始トリガ源をタイマ3 にしました。そこで、タイマ3 モジュールを設定します。

1, 2. TMR3

3. Enable TMR

4. FOSC/2

タイマ3 のクロック源を $F_{OSC}/2 = F_P = 69 \text{ MHz}$ とします。

5. 1:1

Prescaler を 1/1 にします。

6. 66.7 us

タイマ周期を 66.7 [μs] にします。Period Count の 0x1200 は自動的に計算されます。これは、タイマ3 による A-D 変換開始トリガの周期です。A-D コンバータによる入力電圧のサンプリング周期 f_{SAMP} を 15 kHz とする設定です。

C.3.5 SPI モジュール設定

フィルタ出力は SPI モジュールにより D-A コンバータ IC の MCP4921 に送ります。そこで、SPI モジュールを設定します。

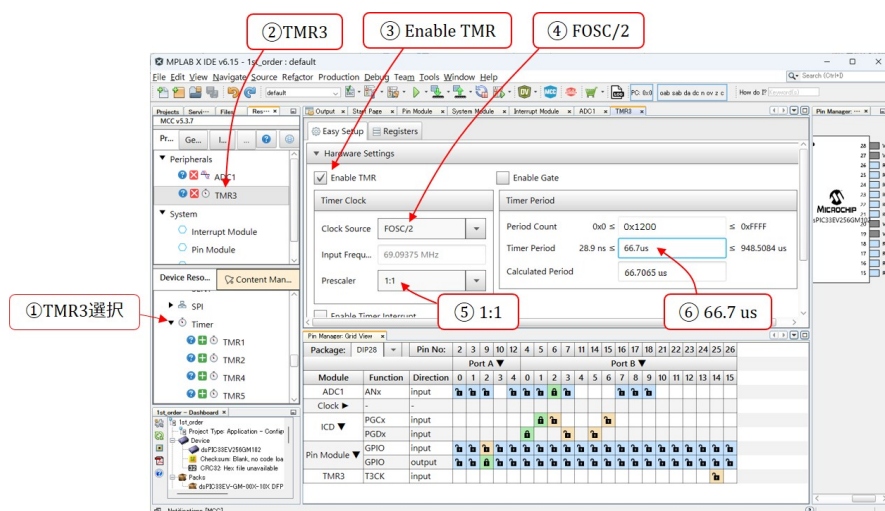


図 C.13: SPI モジュール設定

図 C.13 は SPI モジュールの設定画面です。

1, 2. SPI1 [PIC24/dsPIC33/...] を選択します。

3. Master

マイコン内の SPI モジュールをマスタ・モードとします。MCP4921 内の SPI モジュールがスレーブ・モードで動作します。

4. Enable SPI

5. 16 bit

16 ビット・モードに設定します。12 ビットのデータを送るモードです。

6. 1:1

Primary Prescaler を 1/1 に設定します。

7. 1:1

Secondary Prescaler を 1/1 とします。これにより、SPI モジュールの動作クロックを $F_{OSC}/2 = F_P = 69 \text{ MHz}$ とします。

8. Idle: Low, Active: High

9. Active to Idle

このモードで MCP4921 と通信ができます。

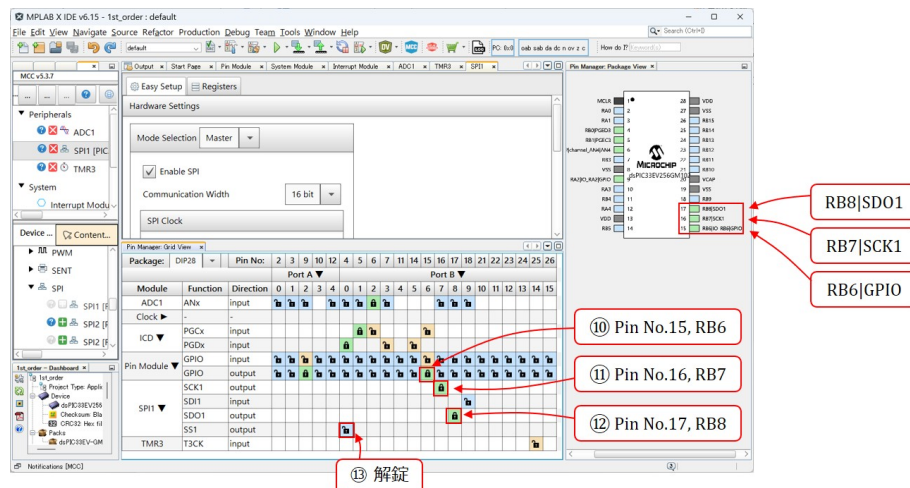


図 C.14: SPI モジュール設定 (つづき)

図 C.14 は SPI モジュール設定画面のつづきです。Pin Manager のテーブル表示にて、

10. Pin No.15, RB6 をデジタル出力に施錠

パッケージ表示において、15 番ピンに GPIO の文字が表示されます。このピンは、MCP4921 に $\overline{CS}$ 信号 (図 E2-1 参照) を出力します。

11. Pin No. 16, RB7 を SCK1 に施錠

12. Pin No. 17, RB8 を SDO1 に施錠

パッケージ表示において、16, 17 番ピンにそれぞれ SCK1, SDO1 の文字が表示されます。それぞれ CLK, DATA 信号 (図 4.1 参照) を出力します。

C.4 周辺モジュール設定関数のコード自動生成

以上で周辺モジュール設定が全て済んだので、画面左上 Project Resources エリアの Generate ボタンを左クリックして、コードを自動生成します。

C.5 ビルド、マイコンへの書き込み

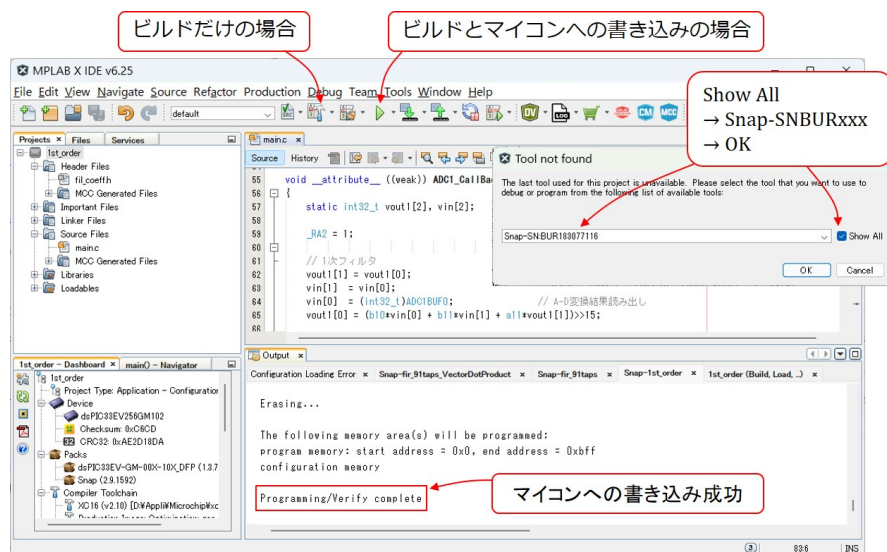


図 C.15: ビルドとマイコンへの書き込み

プログラムのビルドとマイコンへの書き込みの段階です。図 C.15 に示すように、ビルドだけの実行、ビルドとマイコンへの書き込みの実行を選択できます。インサーキット・デバッガ/プログラマ (A.2.3 項参照) が見つからないとのメッセージが出た場合は、Snap-SNBURxxx を選択します。

全てが正常であれば、Programming/Verify complete のメッセージが出ます。マイコンは自動的にプログラム実行を開始します。

C.6 デバッガ

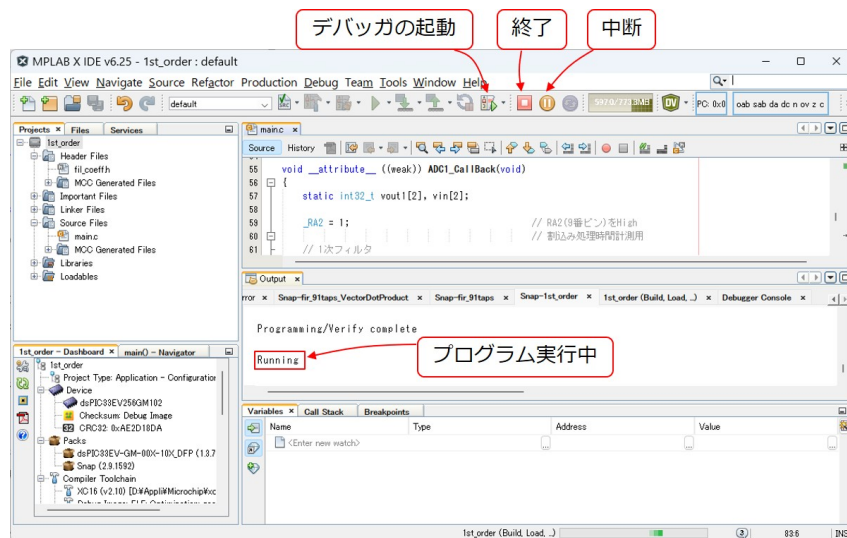


図 C.16: デバッガの起動

図 C.16 はデバッガ起動の様子を示します。ツールバーのデバッガ起動ボタンをクリックすることで、プログラムのマイコンへの書き込みとマイコンの実行を開始します。Running がマイコンが実行中であることのメッセージです。

中断ボタンをクリックすることで、マイコンを実行を中断できます。

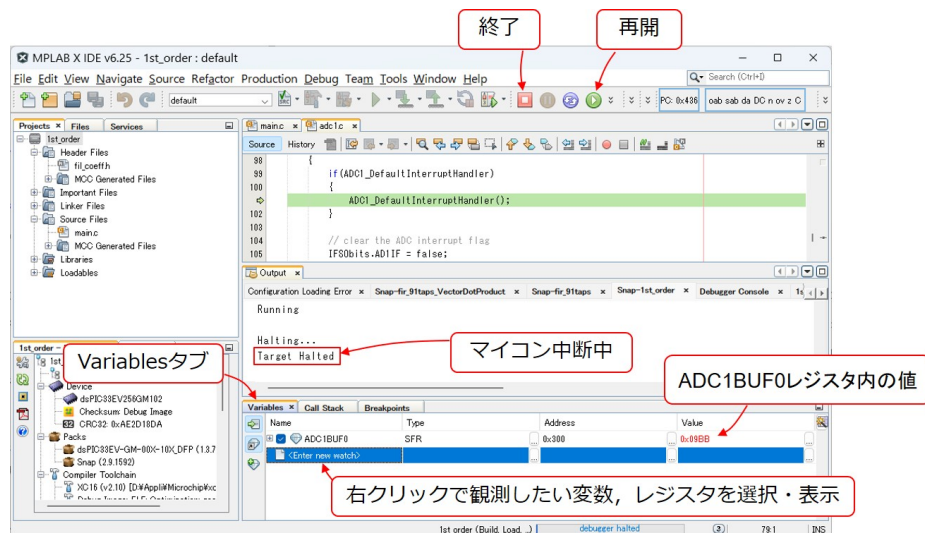


図 C.17: マイコン中断時の変数・レジスタ値の表示

図 C.17 は、マイコン中断時の画面です。Variables タブを左クリックし、Enter new watch を右クリックすることで、観測したい変数、レジスタを選択できます。図では、ADC1BUF0

レジスタを選択しています。中断時、このレジスタには 0x09BB の値が格納されていることがわかります。

なお、変数はグローバル変数のみを表示できます。プログラムにて、観測したい変数をグローバル変数に宣言し直して、デバッガの再実行、中断により変数値を見ることができます。

ツールバーの再開ボタンをクリックすることで、プログラムを中断時点から再開できます。

索引

- 0 捨 1 入, 29
- 1 サンプル・シフト演算, 30
- 1 次バタワース・ロー・パス・フィルタ, 9, 28
- 21 タップ FIR ロー・パス・フィルタ, 21, 35
- 2 次バタワース・ロー・パス・フィルタ, 14, 32
- 3 次バタワース・ロー・パス・フィルタ, 16, 33
- 8 次バタワース・ロー・パス・フィルタ, 18
- 91 タップ FIR ロー・パス・フィルタ, 25, 36
- $\overline{A}/B$, 42
- ADALM2000, 43
- ADC Interrupt, 65
- ADC1, 64
- ADC1BUF0, 65
- ADC1_CallBack() 関数, 30
- ADC1 割り込み処理関数, 30
- A-D コンバータ・モジュール設定, 64
- Auto Sampling, 65
- AVDD, 50
- AVSS, 50
- BUF, 42
- CH0SA チャネル, 66
- Composer エリア, 64
- CS, 5
- $\overline{CS}$, 42
- D-A コンバータ, 5
- D-A コンバータ IC, 41
- dB/dec, 10
- DSP, 5
- dsPIC33EV256GM102-I/SP, 5, 50
- dsPIC マイコン, 5
- DSP 関数, 25
- FCY, 62
- FOSC, 62
- FP, 62
- FRC, 61
- $\overline{GA}$, 42
- ICSP コネクタ, 45
- ICSP 通信, 50
- int32_t, 30
- $\overline{LDAC}$, 42
- Master, 68
- Maximize Window, 64
- MCC, 61
- MCP4921, 5
- MPLAB® SNAP, 45
- MPLAB® Code Configurator, 61
- MPLAB® XC16 コンパイラ, 57
- MPLAB® X IDE, 57
- Network Analyze, 53
- New Project, 59
- N タップ FIR フィルタ, 21
- Period Count, 67
- PGEC3, 63
- PGED3, 63
- Pin Manager: Grid View, 63
- PLL, 62
- Q15 関数, 28

- Q15 フォーマット, 29
- Restore Window, 65
- right justified, 65
- round(), 29
- SCK, 5, 41
- Scopy, 51
- SDI, 5, 41
- SDO, 5
- $\overline{\text{SHDN}}$, 42
- Shift_JIS, 60
- SPI, 5
- SPI Transmit() 関数, 31
- SPI1BUF レジスタ, 31
- SPI 通信, 50
- SPI モジュールの制御データ, 42
- System Module, 61
- TAD, 65
- USB コネクタ, 45
- VectorCopy, 25
- VectroDotProduct, 25, 38
- VectorCopy, 37
- VREF, 42
- アクティブ・ラーニングモジュール, 43
- アナログ入力, 50
- 位相遅れ, 12
- インサーキット・デバッガ／プログラマ, 45
- インパルス応答, 22
- インフォメーション・ウィンドウ, 52
- カーボン抵抗, 47
- 固定小数点数, 28
- コンデンサ, 46
- サンプリング時刻, 9
- サンプリング周期, 9
- 四捨五入, 29
- しゃ断域, 10
- すずめっき軟銅単線, 47
- 積層セラミックコンデンサ, 46
- 直線位相特性, 24
- 通過域, 10
- 抵抗, 47
- デジタル・フィルタ, 5
- 電線, 47
- 伝達関数, 10, 15, 16, 19, 21
- ナイキスト周波数, 10
- 内積, 38
- 入出力ピン配置, 44
- ネットワーク・アナライザ, 53
- ハニング窓, 21
- ピン・ソケット, 48
- ピン・ヘッダ, 48
- フィルタ係数, 9, 14, 16, 18, 21, 23, 28
- 符号付き 32 ビット形, 30
- ブレッドボード, 6, 49
- ポート A, 50
- ポート B, 50
- マイコン, 50
- 理想フィルタ, 22, 23

参考文献

- [1] 古橋武「**モータドライブノート**」.
- [2] 後閑哲也「C 言語& MCC による PIC プログラミング大全」技術評論社, 2023.
- [3] 大類重範「デジタル信号処理」日本理工出版会, 2001.
- [4] 萩原将文「デジタル信号処理 (第 2 版・新装版)」森北出版, 2020.
- [5] 岩田彰「デジタル信号処理 新インターユニバーシティ」Ohmsha, 2013.