第５章　訓練課題集

＜　訓練課題集　＞

※　訓練課題の一部のページで、ページ番号が見づらくなっております。予めご了承ください。

実技課題

管理番号：Ｅ－３９

「アナログ回路設計・製作」

■課題概要■

電流帰還バイアス回路の設計を通して、シミュレーション解析、回路製作、特性評価など、アナログ回路設計技術について確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領	○	E-39-00_実施要領
訓練課題	○	E-39-01_訓練課題
解答	○	E-39-02_解答及び解説
作業工程手順書	○	E-39-03_作業工程計画書
訓練課題確認シート	○	E-39-04_訓練課題確認シート及び評価要領
評価要領	○	E-39-04_訓練課題確認シート及び評価要領

実技課題　実施要領 訓練課題名　「アナログ回路設計･製作」

作業準備及び使用機器の準備時間は別途確保した上で事前に行うこととし、作業時間に含まない。
作業時間は、休憩時間を除いた時間とする。
実施形態は、１名で製作することが望ましい。
作業工程計画書は、ポイント（留意事項）のみ記述させる。
以下の手順で確認シートを基に評価を行うこと。
①回路製作・波形計測が行われたかどうかを確認する。
②作業工程計画書、回路図、シミュレーション結果、データシート（負荷線・動作点を記入したもの）、計算過程を記入した用紙を提出させる。

時間	実施内容
9:00～9:10	出欠確認
9:10～9:30	課題内容説明および質問
課題製作開始	
9:30～10:00	使用機器の準備、作業工程の作成
10:00～11:00	回路設計
11:00～12:00	シミュレーション解析
12:00～12:45	昼食
12:45～14:30	回路製作および動作確認
課題製作終了	
14:30～14:40	製作課題提出
14:40～14:55	片付け・整理整頓

実技課題

「アナログ回路設計・製作」

１　作業時間

　　２１０分（作業工程作成時間と休憩時間は除く）

２　配付資料

　　問題用紙，データシート、作業工程計画書

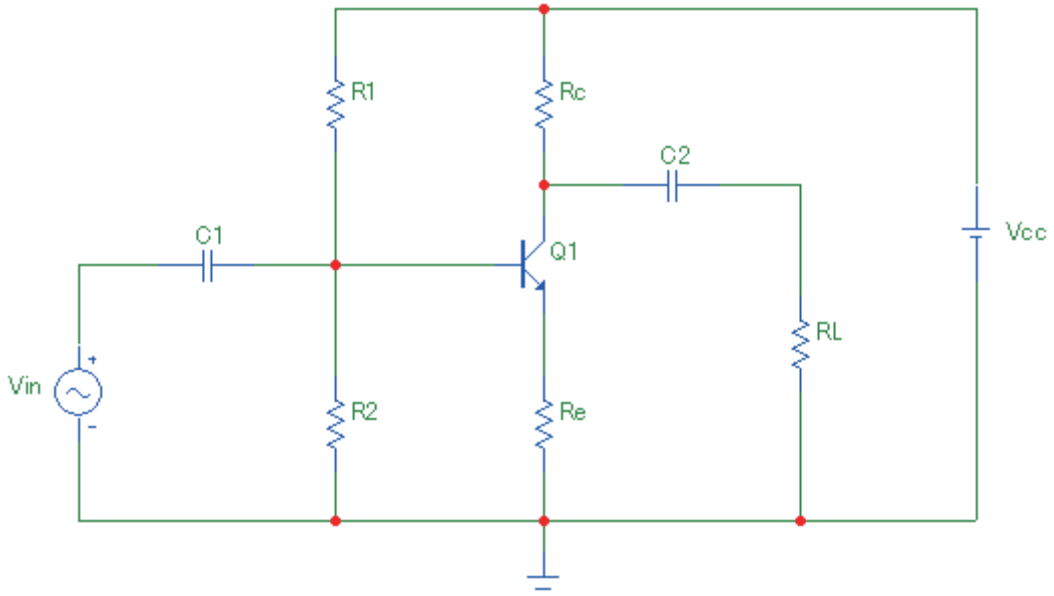
３　課題作成、提出方法

　　回路製作・波形計測を行い、講師が確認した後、作業工程計画書とプリントアウトした回路図とシミュレーション結果、データシート（負荷線と動作点を記入したもの）、計算過程を記入した用紙を回収する。

課題１

電圧増幅度５倍のエミッタ接地電流帰還バイアス回路を設計する。動作点をトランジスタのデータシートの特性表から決定する。

下記の回路において、電源電圧 Vcc、抵抗 R1、R2、Rc、Re の値と、コンデンサ C1 と C2 の値を決定する。
入力信号は正弦波とし、振幅 0.5V（V_{P-P}＝1.0V)、周波数を 1kHzとする。



課題２

設計した回路をシミュレーションソフトで解析する。

入力波形と出力波形を表示させる。
設計した回路の周波数特性を表示させる。

課題３

ブレッドボード上に回路を作成し、計測器を用いて動作確認を行う。

実技課題　解答及び解説

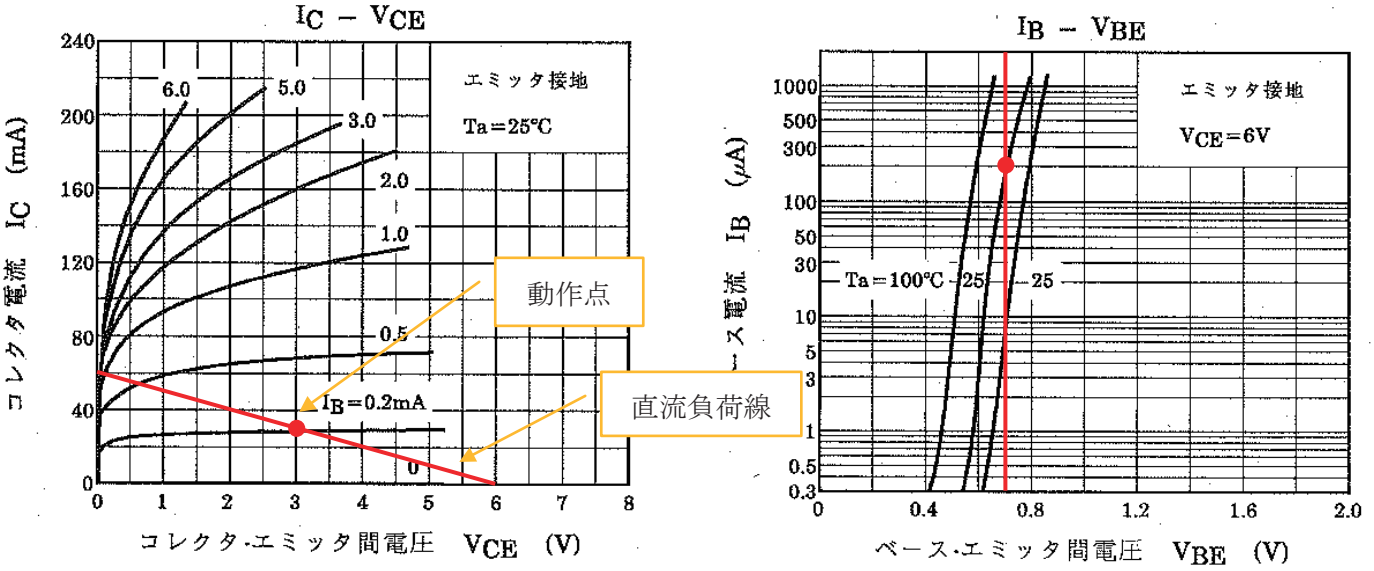
「アナログ回路設計・製作」

課題１　電圧増幅度５倍のエミッタ接地電流帰還バイアス回路の設計

2SC1815 を用いた場合の設計例

2SC1815 のI_C-V_{CE}特性から直流負荷線と動作点を決定する。

I_B-V_{BE}特性からV_{BE}を求める



$I_C = 30 \quad [mA]$　　$V_{CC} = 6 \quad [V]$

$I_B = 0.2 \quad [mA]$　　$V_{CE} = 3 \quad [V]$

$V_{BE} = 0.7 \quad [V]$

電流帰還バイアス回路の等価回路より、電圧増幅度は近似式$\frac{R_C}{R_E}$によって決まるので、$I_E \cong I_C$として、$V_{CC} = 6 \quad [V] \quad V_{CE} = 3 \quad [V]$とすると、抵抗$R_E$の両端には 0.5V の電圧がかかり、抵抗$R_C$の両端には 2.5V の電圧がかかることになる。よって、

$$R_E = \frac{0.5}{I_C} = \frac{0.5}{30m} = 16.66 \quad [\Omega]$$

$$R_C = \frac{2.5}{I_C} = \frac{2.5}{30m} = 83.33 \quad [\Omega]$$

と求められる。

E24 系列の抵抗から近い値を選定すると、

$$R_E = 15 \quad [\Omega]$$
$$R_C = 82 \quad [\Omega]$$

となる。

このとき、抵抗で消費される電力を考慮して選定する。

また、トランジスタのベースのバイアス電圧は

$$V_B = V_{BE} + 0.5 = 0.7 + 0.5 = 1.2 \quad [V]$$

となる。

このとき、熱などの外乱の影響を少なくするために、抵抗 R_2 に流れる電流が I_B に対して 10 倍以上流れるように抵抗 R_2 を設定する。

$$R_2 = \frac{V_B}{10I_B} = \frac{1.2}{2m} = 600 \quad [\Omega]$$

抵抗 R_1 は

$$R_1 = \frac{V_{CC} - V_B}{11I_B} = \frac{6 - 1.2}{2.2m} = 2181.8 \quad [\Omega]$$

と求められる。

同様に、E24 系列の抵抗から近い値を選定すると、

$$R_2 = 620 \quad [\Omega]$$
$$R_1 = 2.2 \quad [k\Omega]$$

となる。

このとき、抵抗で消費される電力を考慮して選定する。

カップリングコンデンサ C_1 は入力インピーダンス Z_i とハイパスフィルタ(HPF)を形成するので、カットオフ周波数と入力信号の周波数を考慮して値を決定する。

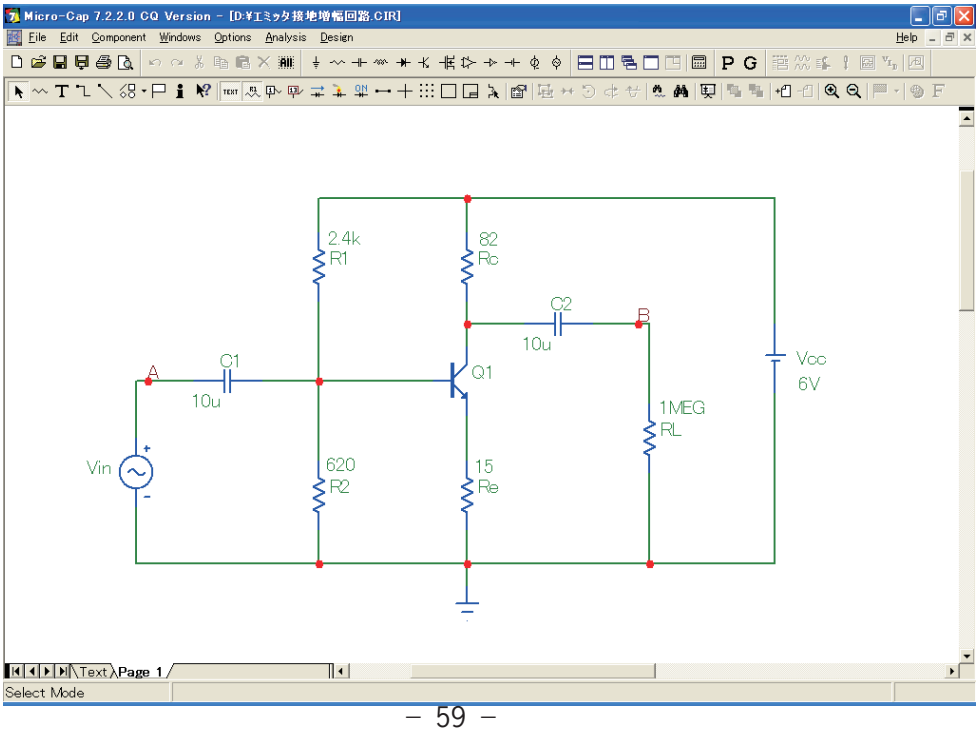
$C_1 = 10 \quad [\mu F]$ とすると、HPF のカットオフ周波数は

$$f_C = \frac{1}{2\pi Z_i C_1} = \frac{1}{2\pi \times 484 \times 10\mu} = 32.8 \quad [Hz]$$

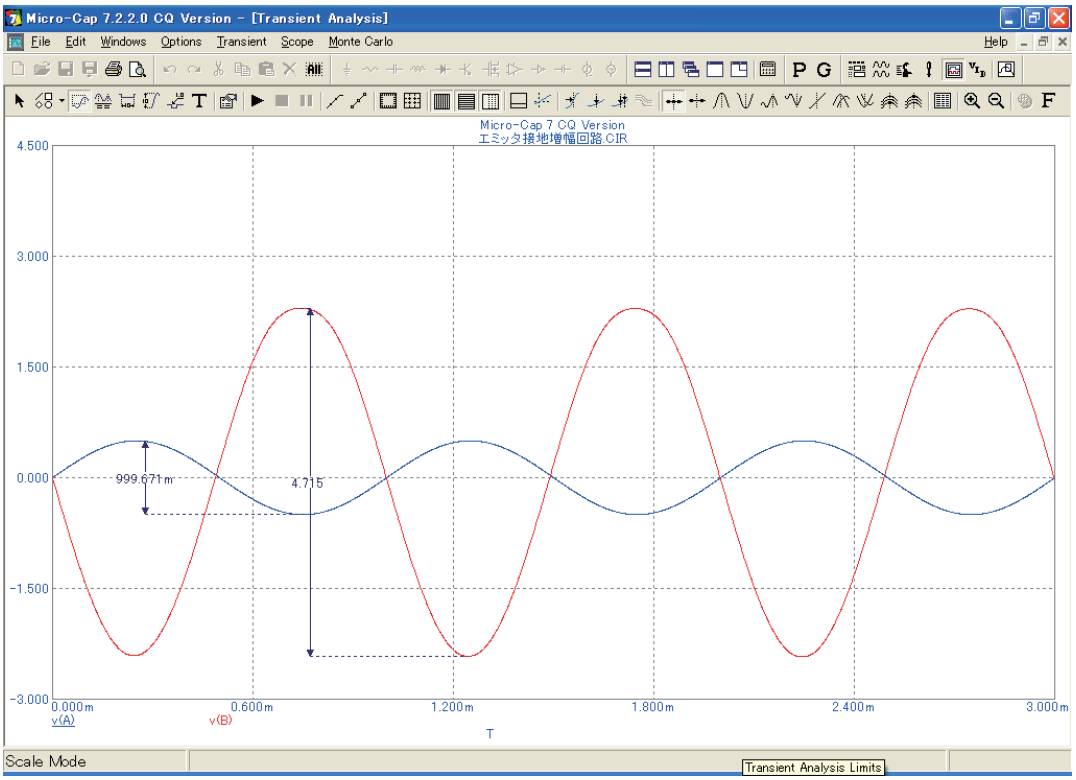
カップリングコンデンサ C_2 も同様に考慮して、$C_2 = 10 \quad [\mu F]$ とする。

課題２　シミュレーションソフトによる解析

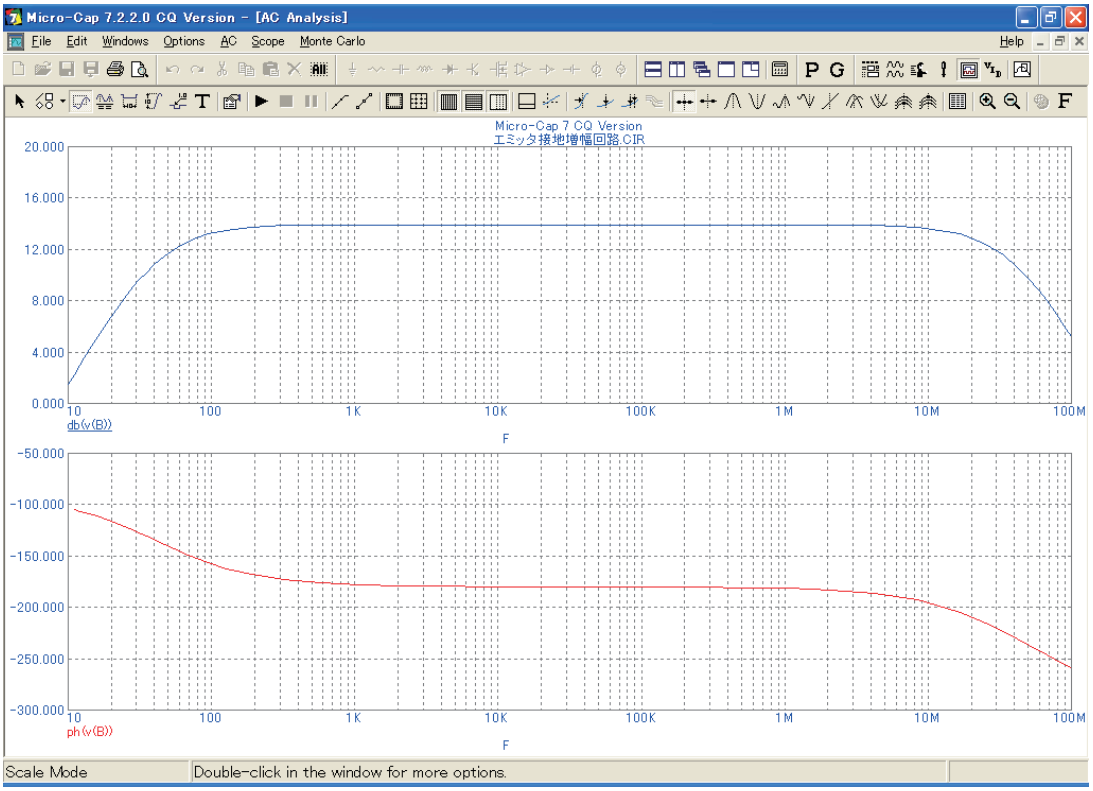
回路図入力例



トランジェント解析（過渡解析）による入出力波形表示例

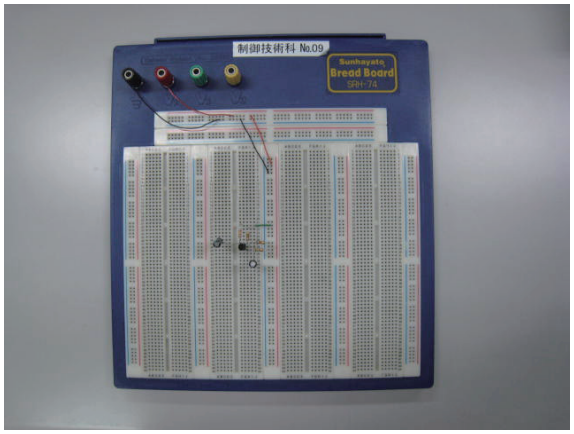


ＡＣ解析による周波数特性と位相特性の表示例

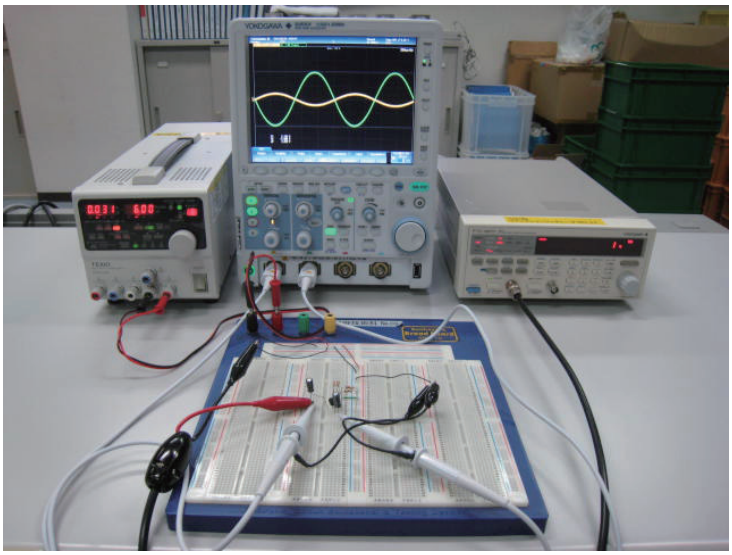


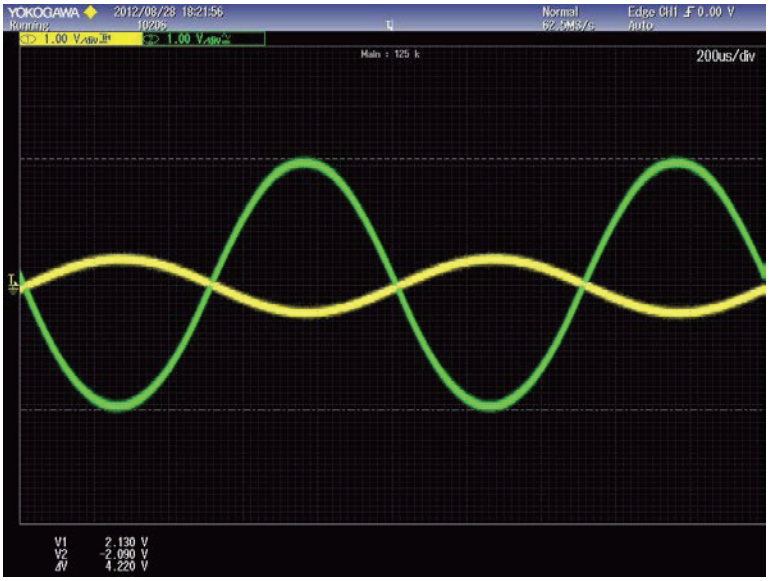
課題３　電子回路製作と動作確認

電子回路製作



測定





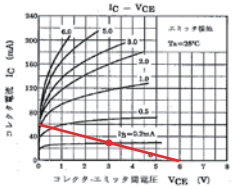
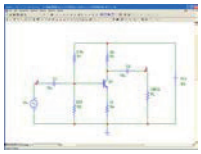
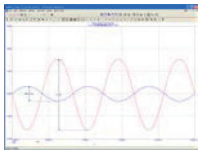
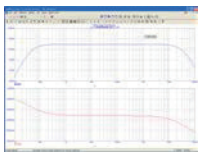
作業工程計画書

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）

作業工程計画書（受講者配布用例）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）

作業工程計画書（模範解答例）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備	使用機器の確認・準備 使用部品の確認・準備	
1.データシートの確認	トランジスタの最大定格と特性表を確認する。	
2.動作点の決定	特性表に直流負荷線を引いて、動作点を決定する。	
3.抵抗値の計算	それぞれの抵抗の値を計算し、定格電力を超えないことを確認してE系列から近い値を選定する。	
4.コンデンサの計算	ハイパスフィルタのカットオフ周波数から、コンデンサの値を計算する。	
5. シミュレーションソフトによる回路図入力	設計した回路図を入力して、値の設定を行なう。	
6.トランジェント（過渡）解析	入力波形と出力波形を表示させる。	
7.AC 解析	周波数特性を表示させて、カットオフ周波数と電圧利得をグラフから読み取る。	
8.回路製作	ブレッドボード上に回路を作成する。	
9.測定	オシロスコープを用いて、入力波形と出力波形を確認し、テスターでバイアス電圧・電流を測定する。	
10. 評価・考察	計算値通りになったかどうか評価し、考察する。	

訓 練 課 題 確 認 シ ー ト

訓練科名 ： 制御技術科

仕上がり像 ： 電気･電子回路の設計及び同回路を用いた制御ができる。

システム名 ： アナログ回路設計技術　　入所期 ：

訓練課題名 ： アナログ回路設計･製作　　氏 名 ：

評価区分	評価項目	細 目	評価（数値）					評価判定	評価基準
作業時間	工程計画作成時間	作業手順	1	2	3	4	5		5点：15分以内、4点：20分以内、3点25分以内、2点：30分以内、1点：30分超え
	作業準備時間	機器･部品の準備	1	2	3	4	5		5点：15分以内、4点：20分以内、3点25分以内、2点：30分以内、1点：30分超え
	作業時間	①回路設計時間 ②シミュレーション時間 ③回路製作時間	1	2	3	4	5		①、②、③全ての作業時間として 5点：210分以内、4点：230分以内、3点250分以内、2点：270分以内、1点：270分超え
作業工程	作業工程における留意事項等	作業工程手順	1	2	3	4	5		作業工程が不適切な場合は、1箇所につき1点減点し、最低点を1点とする。
		作業工程における留意事項等	1	2	3	4	5		作業工程における工夫･改善･留意点が不適切な場合は、1箇所につき1点減点し、最低点を1点とする。
回路設計	動作点の決定	動作点の決定	1		3		5		直流負荷線が引いた上で動作点が適正な位置にあれば5点、直流負荷線が無く動作点のみ決定していれば3点、それ以外を1点とする。
	抵抗値の計算	抵抗値の計算	1	2	3	4	5		不適切な値や計算間違いがあれば1点ずつ減点し、最低点を1点とする。
	コンデンサの計算	コンデンサの値計算	1		3		5		カットオフ周波数の式から適正な値を導き出したら5点、計算間違いがあれば3点、それ以外を1点とする。
シミュレーション	回路図作成	シミュレーションソフトを使用した回路図入力	1	2	3	4	5		回路図入力において、不適切な部品の配置や設定があるごとに1点ずつ減点し、最低点を1点とする。
	トランジェント（過度）解析	入出力波形の表示	1		3		5		適正な波形を表示すれば5点、課題と異なる結果の波形を表示すれば3点、それ以外を1点とする。
	AC解析	周波数特性の表示	1		3		5		適正な結果を表示すれば5点、課題と異なる結果を表示すれば3点、それ以外を1点とする。
回路製作	回路製作	ブレッドボード上での回路製作	1	3	5	7	10		適切な部品を使用し、配置位置も考慮しており、配線が短くきれいに行われているか5段階で評価する。
	測定器の取り扱い	測定器を用いた計測	1		5		10		適正な波形を表示すれば10点、課題と異なる結果の波形を表示すれば5点、それ以外を1点とする。
	回路評価･考察	評価と考察	1	3	5	7	10		設計値とシミュレーション結果、回路作成時の出力波形を比較して、設計仕様を満たしているか、満たしていない場合でもその原因等の考察を行っているかを5段階で評価する。
安全作業	安全作業	部品等の破損	1		3		5		部品の破損が無ければ5点、部品を破損し交換すれば1点とする。 また部品の破損が無くても、電源を入れた状態で配線作業などを行っていれば3点とする。
	片付け、整理･整頓	片付けと整理･整頓	1	2	3	4	5		片付けと整理整頓を5段階で評価する。
工夫・改善	工夫･改善	作業工程の工夫、回路設計･製作上の工夫、作業改善	0	1	2	3	4	5	工夫･改善がなければ0点とし、工夫･改善点1件につき1点ずつ加算し、最高点を5点とする。
	工夫･改善点記入欄		総点					100	＜判定表＞ A ： 80点以上 ：到達水準を十分に上回った B ： 60点以上80点未満 ：到達水準に達した C ： 60点未満 ：到達水準に達しなかった
			合計点						
			換算点						
			総合評価判定						

訓練課題のねらい	コメント
	担当指導員氏名：

評価要領

訓練科名　：　制御技術科
仕上がり像　：　電気･電子回路の設計及び同回路を用いた制御ができる。
システム名　：　アナログ回路設計技術
訓練課題名　：　アナログ回路設計･製作

評価区分	評価項目	細　目	評価要領（採点要領）	備考
作業時間	工程計画作成時間	作業手順	指導員の合図により作業を開始する。作業工程計画書ができたら挙手する。	
	作業準備時間	機器･部品の準備	指導員の合図により作業を開始する。	
	作業時間	①回路設計時間 ②シミュレーション解析時間 ③回路製作時間	指導員の合図で作業を開始、休憩は一斉とし、作業完了は指導員が確認する。	
作業工程	作業工程における留意事項等	作業工程手順	作業工程手順が適切であるか確認する。	
		作業工程における留意事項等	作業工程における工夫･改善･留意点が記載されているか確認する。	
回路設計	動作点の決定	動作点の決定	データシートから最大定格を考慮し、特性表を元に回路に加える電源電圧、回路に流れる電流を決定しているか確認する。	
	抵抗値の計算	抵抗値の計算	動作点を元に抵抗値の計算を行い、定格電力や精度を考慮して抵抗を選定しているか確認する。	
	コンデンサの計算	コンデンサの値計算	ローパスフィルタのカットオフ周波数を考慮して、コンデンサの値を計算し、定格電圧を考慮して選定しているか確認する。	
シミュレーション	回路図作成	シミュレーションソフトを使用した回路図入力	正しく回路図が入力されているか確認する。	
	トランジェント（過度）解析	入出力波形の表示	解析方法が正しく行われているか確認する。	
	AC解析	周波数特性の表示	解析方法が正しく行われているか確認する。	
回路製作	回路製作	ブレッドボード上での回路製作	部品の配置場所を考慮して、配線が短く且つきれいに出来ているか確認する。	
	測定器の取り扱い	測定器を用いた計測	オシロスコープ、テスターの使い方が理解できているか確認する。	
	回路評価･考察	評価と考察	設計値とシミュレーション結果、回路作成時の出力波形を比較して、設計した回路の評価と考察ができているか確認する。	
安全作業	安全作業	部品等の破損や作業行為の確認	配線中や動作確認中に部品を破損していないかどうか確認する。 また、作業中における危険な行為が無いかどうか確認する。	
	片付け、整理･整頓	片付けと整理･整頓	使用機器や使用部品の片付け、机の上の整理整頓が出来ているか確認する。	
工夫・改善	工夫･改善	作業工程の工夫、回路設計･製作上の工夫、作業改善		

筆記課題

管理番号：Ｅ－40
「アナログ回路に関する基礎知識」

■課題概要■

ダイオード、トランジスタ、オペアンプに関する一般的な特性と、回路設計に必要な知識を筆記試験により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領		
訓練課題	○	E-40-01_訓練課題
解答	○	E-40-02_解答及び解説
作業工程手順書		
訓練課題確認シート		
評価要領		

筆記課題

「アナログ回路に関する基礎知識」

１　作業時間
　　90分

２　配付資料
　　問題用紙，解答用紙

３　課題作成、提出方法
　　解答用紙のみを回収する

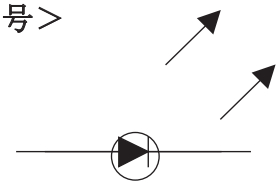
問題１

下記のダイオード(1)～(6)について、それぞれ適切な図記号と特性を（ア）～（シ）の中から選択しなさい。

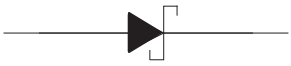
(1)　整流用ダイオード
(2)　LED
(3)　定電圧ダイオード
(4)　ショットキ・バリア・ダイオード
(5)　可変容量ダイオード
(6)　フォトダイオード

＜図記号＞

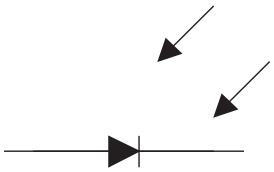
（ア）



（イ）



（ウ）



（エ）

（オ）

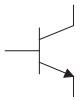
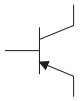
（カ）

＜特性＞

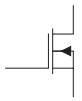
（キ）バリキャップ、バラクタとも呼ばれ、逆方向に加える電圧によって静電容量が変化するダイオード。
（ク）入射した光を電気に変換するダイオード。応答特性に優れている。
（ケ）金属と半導体を接合してできたダイオード。順方向電圧が低いので、高速のスイッチングが可能。
（コ）発光ダイオードのこと。電流を流すと光を放つダイオード。
（サ）交流電源から直流を取り出すために用いられる。逆方向電圧・順方向電流が比較的高い。
（シ）ツェナー現象を利用したダイオード。逆方向に電流を流すことで電圧を安定化させる。

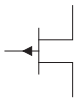
問題２

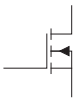
下記の表は、トランジスタの種類、図記号、極性でまとめたものである。（　）の中に入る適切な図記号を（ア）～（カ）の中から選択しなさい。

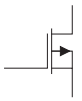
名称	バイポーラトランジスタ		接合型 FET（JFET）		MOS 型 FET			
					デプレッション型		エンハンスメント型	
図記号			(①)	(②)	(③)	(④)	(⑤)	(⑥)
極性	NPN 型	PNP 型	N チャネル	P チャネル	N チャネル	P チャネル	N チャネル	P チャネル

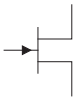
＜図記号＞

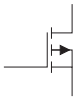
（ア）　 

（イ）　 

（ウ）　 

（エ）　 

（オ）　 

（カ）　 

問題３

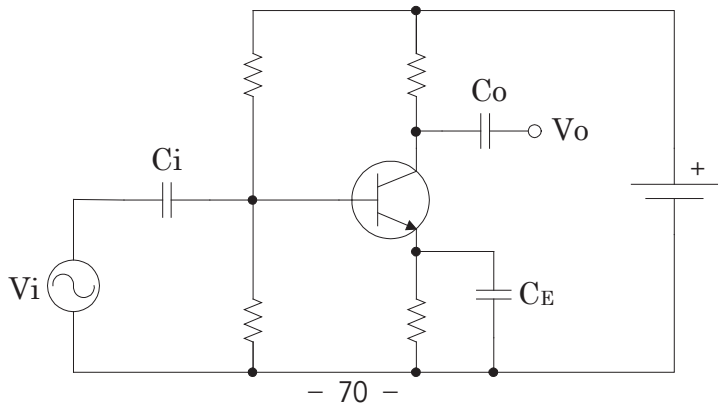
次の記述はトランジスタ回路の動作に関して述べたものである。（　）の中に入る適切な語句を（ア）～（セ）から選択しなさい。

エミッタ接地回路におけるコレクタ電流 I_C とベース電流 I_B の比は h_{FE} で表わされる。この h_{FE} を（　①　）といい、一般的な値は（　②　）である。

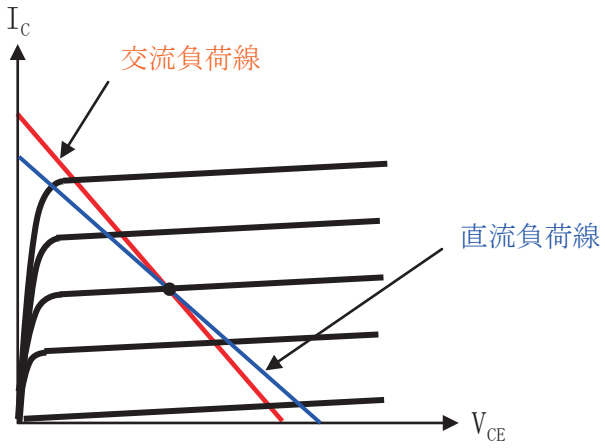
回路図の Ci と Co のコンデンサは（　③　）といい、直流分を通過させないためのものである。

C_Eのコンデンサを（　④　）といい、交流信号に対するインピーダンスを（　⑤　）ことで、増幅度を上げることができる。

Vo から取り出される信号は、入力信号 Vi に対して（　⑥　）出力である。



図において、直流に対する電圧と電流の関係を表した式を直流負荷線といい、交流信号に対する電圧と電流の関係を表した式を交流負荷線という。直流負荷線と交流負荷線の交差する点を（　⑦　）という。



＜語句＞

（ア）上げる　（イ）下げる　（ウ）0.1～1　（エ）10～1000　（オ）1000～100000
（カ）バイパスコンデンサ　（キ）カップリングコンデンサ　（ク）動作点
（ケ）追従点　（サ）電流帰還率　（シ）電流増幅率　（ス）同相　（セ）反転

問題４

次の記述は各接地回路の特徴に関して述べたものである。文章に該当する回路名と回路図を（ア）～（カ）の中から選択しなさい。

（１）電圧増幅度はほぼ１倍であるが、入力インピーダンスが大きく、出力インピーダンスが小さいので、インピーダンス変換として用いられる。この回路はエミッタ・フォロワとも呼ばれている。

（２）電流増幅度がほぼ１倍で、他の二つの回路と比較すると入力インピーダンスが小さく、出力インピーダンスが大きい回路ではあるが、高周波用の増幅回路として使用されている。

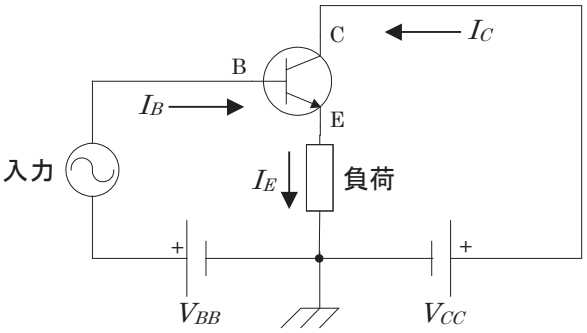
（３）電流増幅度・電圧増幅度が共に大きく、電力利得は最も大きい。低周波用の増幅回路として多く使用されている。

＜回路名＞

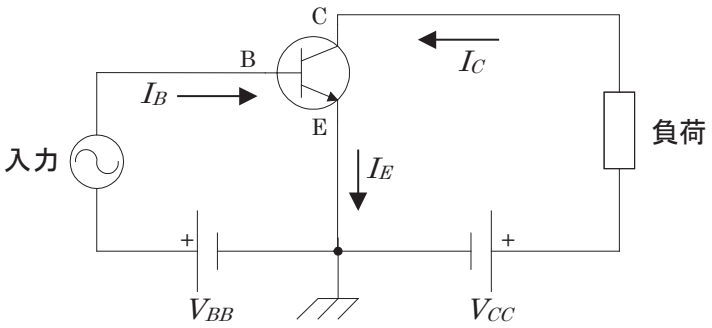
（ア）エミッタ接地回路　（イ）コレクタ接地回路　（ウ）ベース接地回路

＜回路図＞

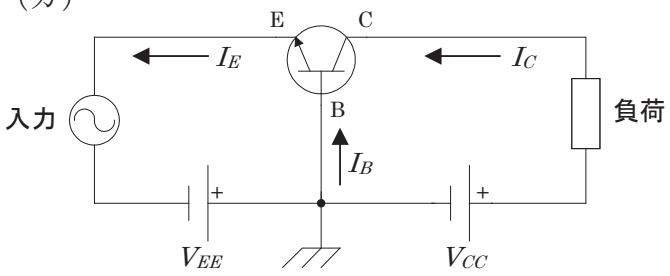
（エ）



（オ）



（カ）

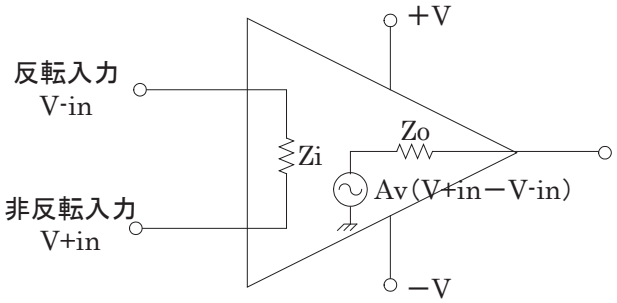


問題５

理想オペアンプの特性について、下記の項目に関して適切な語句（ア)、(イ）のいずれか選択しなさい。

(1)　理想オペアンプの電圧増幅度 Av は（　　）
(2)　理想オペアンプの入力インピーダンス Zi は（　　）
(3)　理想オペアンプの出力インピーダンス Zo は（　　）
(4)　理想オペアンプの入力オフセット電圧は（　　）
(5)　理想オペアンプの入力オフセット電流は（　　）
(6)　理想オペアンプのスルーレートは（　　）

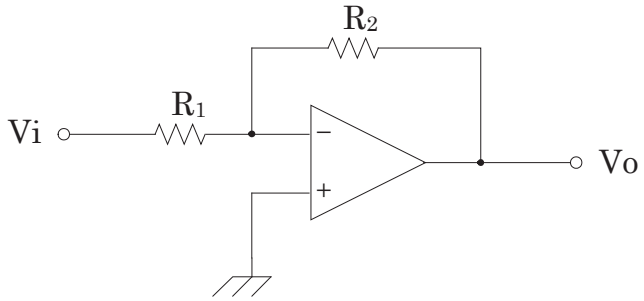
＜語句＞
（ア）0（ゼロ）である
（イ）無限大（∞）である



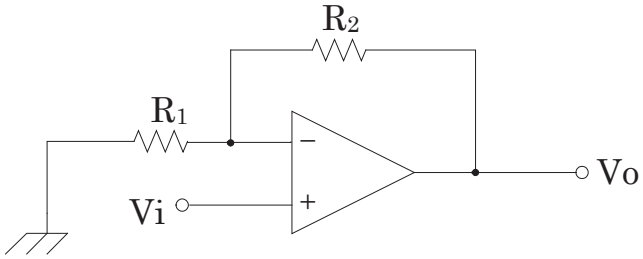
問題６

（１）、（２）それぞれの図に示す回路の名称と電圧増幅度の組み合わせで適切なものを選択肢（ア）～（カ）の中から選択しなさい。

（１）



（２）



＜選択肢＞

（ア）非反転増幅回路　$\dfrac{R_1}{R_1+R_2}$

（イ）反転増幅回路　$1-\dfrac{R_2}{R_1}$

（ウ）反転増幅回路　$-\dfrac{R_2}{R_1}$

（エ）非反転増幅回路　$\dfrac{R_2}{R_1}$

（オ）非反転増幅回路　$1+\dfrac{R_2}{R_1}$

（カ）反転増幅回路　$-\dfrac{R_1}{R_1+R_2}$

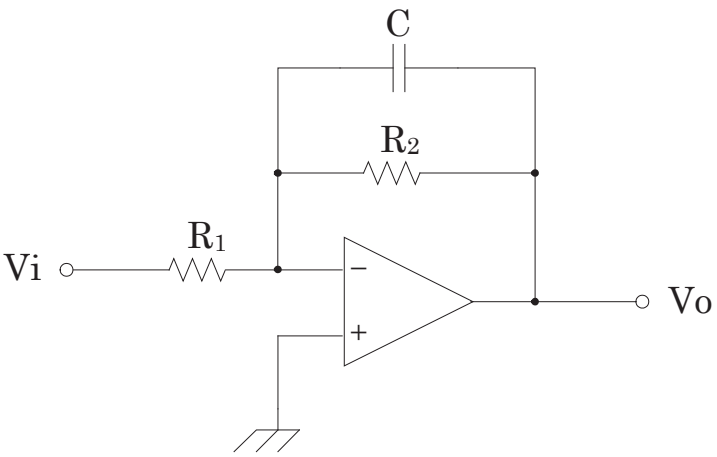
問題 7

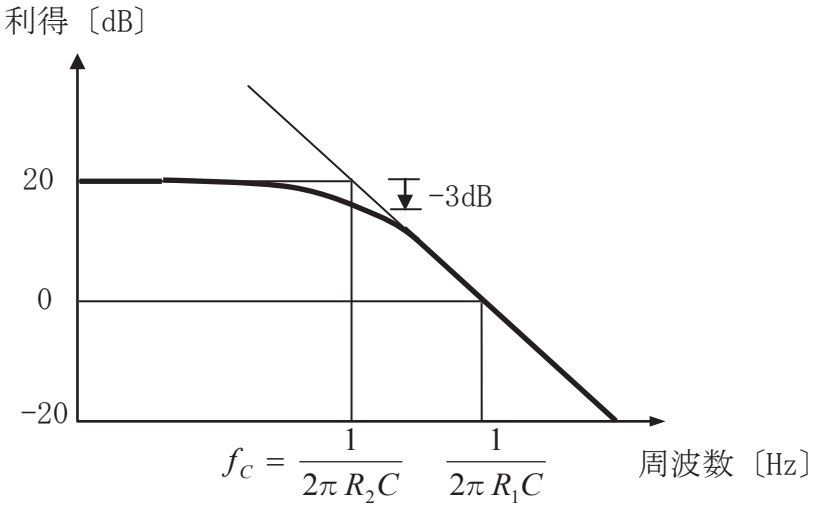
図に示す回路とその周波数特性に関して述べたものである。(　) の中に入る適切な語句を（ア）～（ソ）の中から選択しなさい。

周波数がf_Cの値より小さいときは、(　①　) の動作をし、周波数がf_Cの値より大きいときは (　②　) の動作を行なう。

これらの特性から、この回路は (　③　) フィルタとも呼ばれており、-3dB降下したf_Cを (　④　) 周波数と呼ぶ。

また、周波数がf_Cより大きいときの直線の傾きは-20〔dB/dec〕または-6〔dB/oct〕と表され、周波数が 10 倍になると増幅度が (　⑤　) 倍になり、周波数が (　⑥　) 倍になると増幅度がほぼ 1/2 倍になることを意味する。





＜語句＞

（ア）ハイパス　（イ）ローパス　（ウ）バンドパス　（エ）1/10　（オ）1/4
（カ）2　（キ）6　（ク）通過　（ケ）遮断　（コ）商用　（サ）非反転増幅
（シ）反転増幅　（ス）差動増幅　（セ）微分　（ソ）積分

解答用紙

筆記課題「アナログ回路に関する基礎知識」

入所年月	番号	氏名	合計点	評価判定
平成　　年　　月入所				

問題１（各２点）

	図記号	特性		図記号	特性
（１）			（２）		
（３）			（４）		
（５）			（６）		

問題２（各２点）

（１）		（２）		（３）	
（４）		（５）		（６）	

問題３（各２点）

（１）		（２）		（３）	
（４）		（５）		（６）	
（７）					

問題４（各３点）

	回路名	回路図		回路名	回路図
（１）			（２）		
（３）					

問題５（各２点）

（１）		（２）		（３）	
（４）		（５）		（６）	

問題６（各４点）

（１）		（２）	

問題７（各２点）

（１）		（２）		（３）	
（４）		（５）		（６）	

筆記課題　解答及び解説

「アナログ回路に関する基礎知識」

解答用紙

筆記課題「アナログ回路に関する基礎知識」

入所年月	番号	氏名	合計点	評価判定
平成　　年　　月入所				A：80 点以上 B：60 点以上 80 点未満 C：60 点未満

問題１（各２点）

	図記号	特性		図記号	特性
（1）	オ	サ	（2）	ア	コ
（3）	カ	シ	（4）	イ	ケ
（5）	エ	キ	（6）	ウ	ク

問題２（各２点）

（1）	オ	（2）	イ	（3）	ア
（4）	エ	（5）	ウ	（6）	カ

問題３（各２点）

（1）	シ	（2）	エ	（3）	キ
（4）	カ	（5）	イ	（6）	セ
（7）	ク				

問題４（各３点）

	回路名	回路図		回路名	回路図
（1）	イ	エ	（2）	ウ	カ
（3）	ア	オ			

問題５（各２点）

（1）	イ	（2）	イ	（3）	ア
（4）	ア	（5）	ア	（6）	イ

問題６（各４点）

（1）	ウ	（2）	オ

問題７（各２点）

（1）	シ	（2）	ソ	（3）	イ
（4）	ケ	（5）	エ	（6）	カ

実技課題

管理番号: E－41

「ＨＤＬによるディジタル回路設計･製作」

■課題概要■

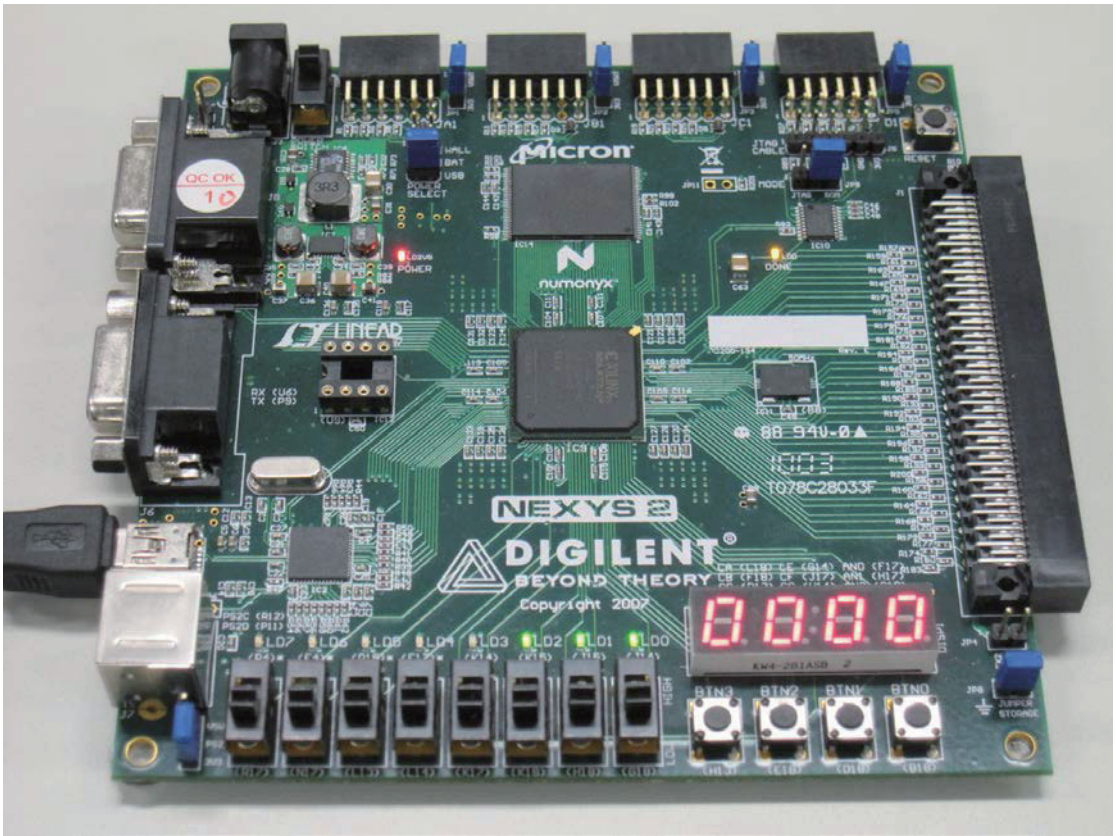
実習用ボードを用いて、HDLによるディジタル回路設計を行います。提示された仕様通りの動作をするディジタル回路をHDLで記述し、論理合成からデバイスへの書き込み、動作確認までを行います。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領	○	E-41-00_実施要領.doc
訓練課題	○	E-41-01_訓練課題.doc
解答	○	E-41-02_解答及び解説.doc
作業工程手順書	○	E-41-03_作業工程計画書.doc
訓練課題確認シート	○	E-41-04_訓練課題確認シート及び評価要領.xls
評価要領	○	E-41-04_訓練課題確認シート及び評価要領.xls

実技課題　実施要領 訓練課題名　「HDL によるディジタル回路設計･製作」

- 作業準備及び使用機器の準備時間は別途確保した上で事前に行うこととし、作業時間に含まない。
- 作業時間は、休憩時間を除いた時間とする。
- 実施形態は、各施設で使用する機器に応じて１～２名で行う。ただし、２名で行う場合は、試験としての公平性、正当性が担保できるような対策を講じること。
- 作業工程計画書は、ポイント（留意事項）のみ記述させる。
- 使用する実習装置のポートマップやクロック周波数については、必要に応じて試験問題と同時に提示すること。
- 各課題ごとに回路動作を確認する。動作の確認は、課題が完成した受講生に挙手させ、その場で動作を確認して記録する。
- 試験終了時にソースコードをプリントアウトして提出させる。また、ソースファイル及び書き込みファイルも電子データで提出させる。
- この課題は、最低限７セグメント LED と押しボタンスイッチを有する PLD（FPGA）の教育用ボードを想定している。また、解答は VHDL にて記載しているので、各施設でハードウェアや使用言語が異なる場合は、それらにあわせて課題を修正したうえで実施することが望ましい。
- VDT 作業を考慮し、1時間を目安に10分程度の休憩時間を設ける。



使用を想定した教育用ボード

実技課題

「ＨＤＬによるディジタル回路設計･製作」

1　作業時間

180 分（休憩を除く）　※延長 30 分

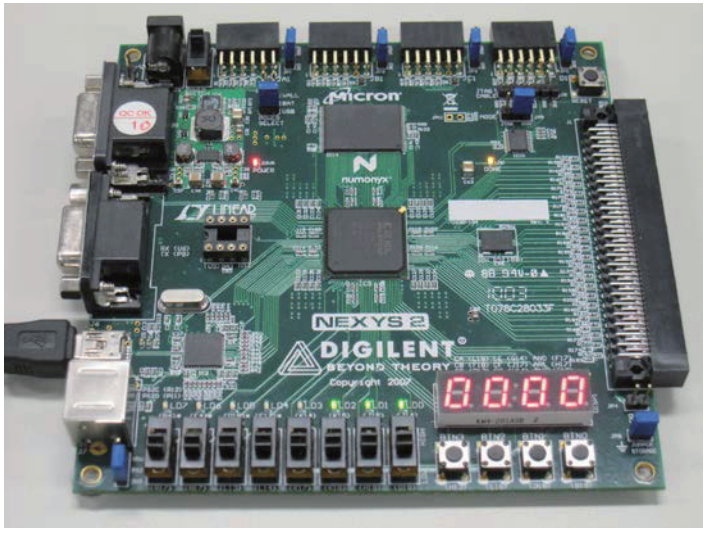
2　配付資料

問題用紙

3　課題作成、提出方法

・各個人で作業すること。

・PLD（FPGA）に設計データを書き込み、教官による動作確認をした上で、ソースコードを印刷して、ソースファイルと書込みファイルの電子データも提出する。

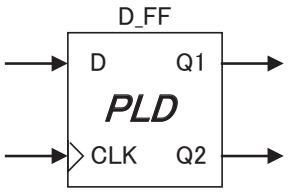


次の回路を設計せよ。なお、入力用のスイッチ及び出力用のＬＥＤについては、特に指定がない限りどれを使ってもかまわない。

１．Ｄフリップフロップを設計せよ。

入力　　：D、CLK

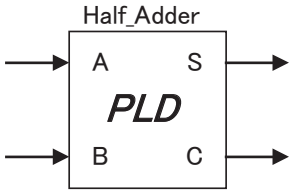
出力　　：Q1、Q2（Q1 の反転出力）



２．半加算器を設計せよ。

入力　　：A、B

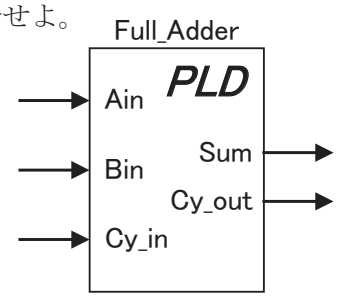
加算出力：S　／　桁上出力：C



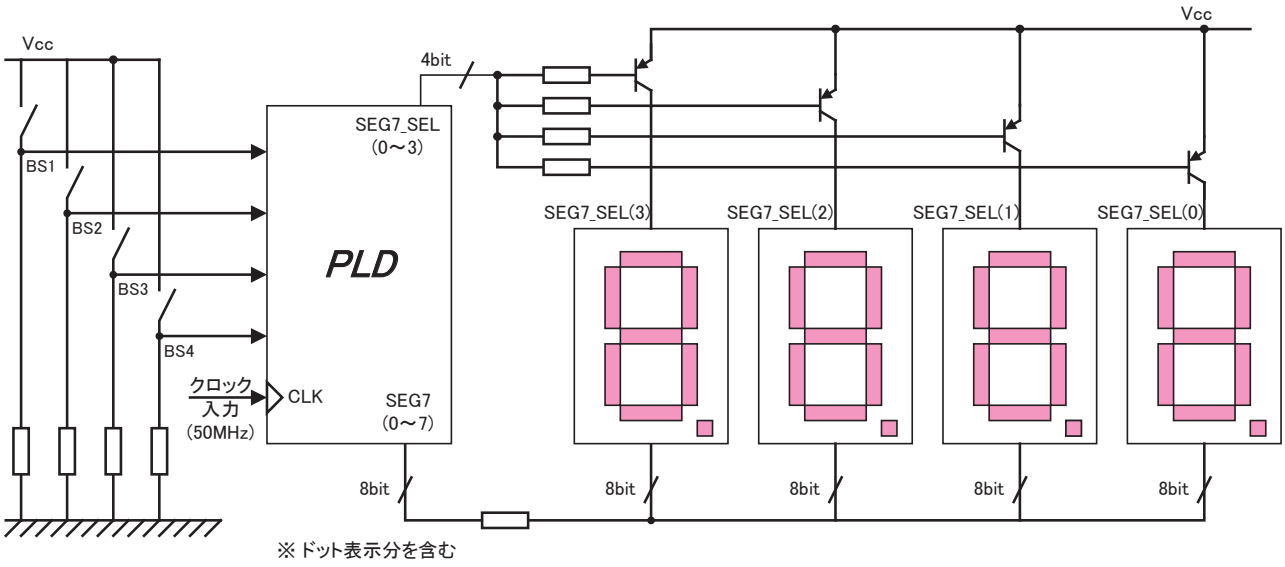
３．“２”で設計した“Half_Adder”を用いて、全加算器を設計せよ。

入力　　：Ain、Bin
桁上入力：Cy_in

加算出力：Sum
桁上出力：Cy_out



４．下図の回路を用いて、次のような回路を設計せよ。



①　４ビットのスイッチ入力に対して、０～９までの数値を１桁の７セグメントＬＥＤで表示するデコーダ回路を設計せよ。なお、10（1010）～15（1111）の入力に対しては、７セグメントＬＥＤを全消灯させよ。

②　10 秒カウンタを設計せよ。非同期リセット入力および同期アップダウン入力を持ち、出力は１桁の７セグメントＬＥＤで表示することとする。

③　60 秒カウンタを設計せよ。非同期リセット入力および同期アップダウン入力を持ち、出力は２桁の７セグメントＬＥＤで表示することとする。

実技課題　解答及び解説

「ＨＤＬによるディジタル回路設計･製作」

1　作業時間

180 分（休憩を除く）　※延長 30 分

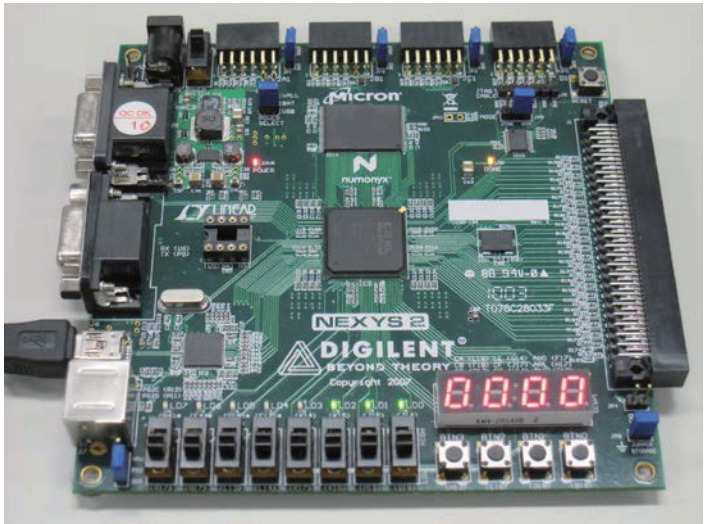
2　配付資料

問題用紙

3　課題作成、提出方法

・各個人で作業すること。

・PLD（FPGA）に設計データを書き込み、教官による動作確認をした上で、ソースコードを印刷して、ソースファイルと書込みファイルの電子データも提出する。

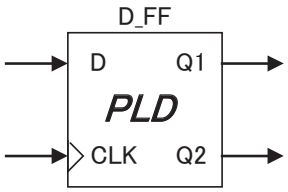


次の回路を設計せよ。なお、入力用のスイッチ及び出力用のＬＥＤについては、特に指定がない限りどれを使ってもかまわない。

１．Ｄフリップフロップを設計せよ。

入力　　：D、CLK

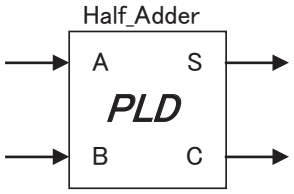
出力　　：Q1、Q2（Q1 の反転出力）



２．半加算器を設計せよ。

入力　　：A、B

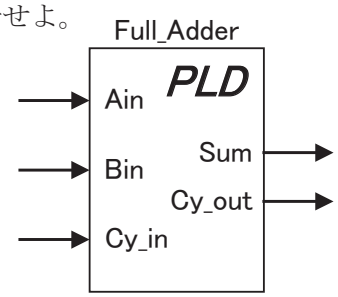
加算出力：S　／　桁上出力：C



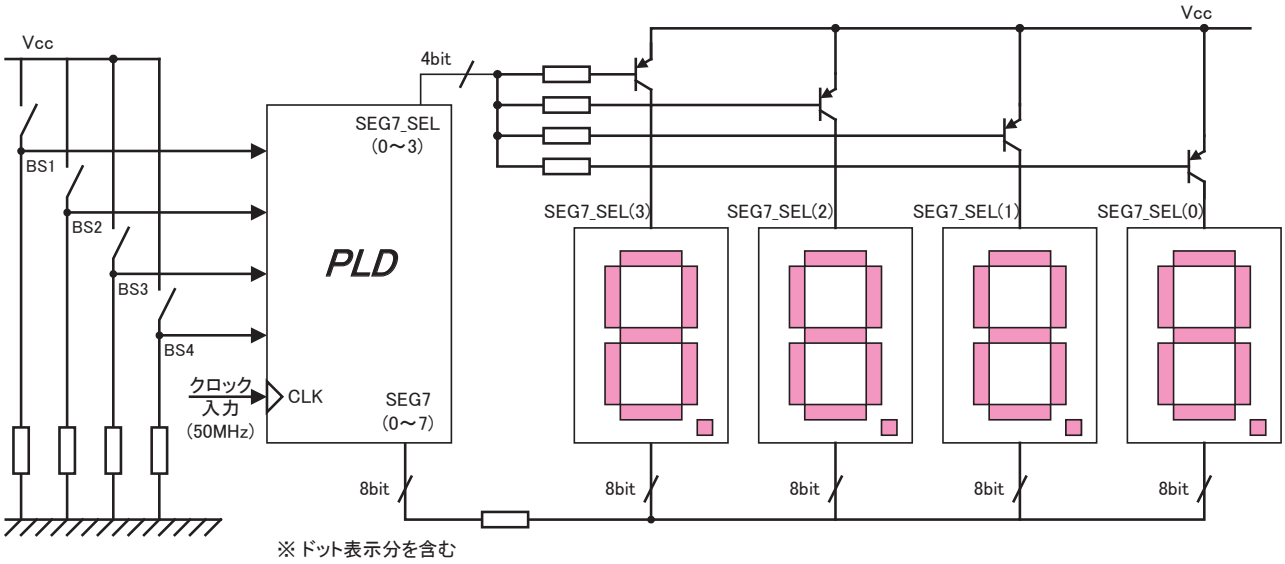
３．“２”で設計した“Half_Adder”を用いて、全加算器を設計せよ。

入力　　：Ain、Bin
桁上入力：Cy_in

加算出力：Sum
桁上出力：Cy_out



４．下図の回路を用いて、次のような回路を設計せよ。



①　４ビットのスイッチ入力に対して、０～９までの数値を１桁の７セグメントＬＥＤで表示するデコーダ回路を設計せよ。なお、10（1010）～15（1111）の入力に対しては、７セグメントＬＥＤを全消灯させよ。

②　10 秒カウンタを設計せよ。非同期リセット入力および同期アップダウン入力を持ち、出力は１桁の７セグメントＬＥＤで表示することとする。

③　①、②で作成した資源を活用して、60 秒カウンタを設計せよ。非同期リセット入力および同期アップダウン入力を持ち、出力は２桁の７セグメントＬＥＤで表示することとする。また、ダイナミック表示の周期は 10ms とする。

1．解答（例）

```
--        VHDL課題          Dフリップフロップ

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity DFF is
          port(     D,CLK     : in std_logic;
                    Q1,Q2     : out std_logic);
end DFF;

architecture RTL of DFF is
          begin
                    process(CLK)
                    begin
                              if(CLK'event and CLK = '1')then
                                        Q1 <= D;
                                        Q2 <= not D;
                              end if;
                    end process;
end RTL;
```

2．解答（例）

```
--        VHDL課題          半加算器
--        構造記述

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity Half_Adder is
   Port ( A,B          : in  std_logic;
          S,C          : out std_logic);
end Half_Adder;


architecture RTL of Half_Adder is
begin
          S <= A xor B;
          C <= A and B;
end RTL;
```

３．解答（例）

```
--          VHDL課題          全加算器
--          階層設計 <Harf_Adder をコンポーネントする >

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity Full_Adder is
    Port ( Ain,Bin,Cy_in          : in  std_logic;
           Sum,Cy_out             : out std_logic);
end Full_Adder;

architecture RTL of Full_Adder is

          component Harf_Adder                    -- 使用するコンポーネントの宣言。
             Port ( A,B           : in  std_logic;
                    S,C           : out std_logic);
          end component;

          signal HA1_S,HA1_C,HA2_C     :std_logic;

begin
          U0 : Harf_Adder                         -- ｲﾝｽﾀﾝｽしたものと同じ順序で記述すれば、
               port map(A,B,HA1_S,HA1_C);         -- 接続先の記述だけでも問題ない。

          U1 : Harf_Adder                         -- 一端子ずつ対応させてもわかりやすい。
               port map(A => HA1_S , B => Ci , S => S, C => HA2_C);
                                                  -- どちらの書き方でもOK。
          Co <= HA1_C or HA2_C;

end RTL;
```

４．①　７セグメントＬＥＤデコーダ　解答（例）

```
--          VHDL課題          7セグメントLEDデコーダ
--          押しボタンスイッチ4個でBCDコードを入力し、右端の7セグメントLEDを点灯させる
--          7セグメントLEDはアノードコモン（点灯は負論理）、7セグのセレクタは負論理

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity DEC7_1 is
    Port (  BS                  : in  std_logic_vector(3 downto 0);
            SEG7_SEL            : out std_logic_vector(3 downto 0);
            SEG7                : out std_logic_vector(7 downto 0));
end DEC7_1;

architecture RTL of DEC7_1 is
begin

            SEG7_SEL <= "1110";

            process(BS)
            begin
                        case BS is
                                    when"0000"          => SEG7 <= "00000011";
                                    when"0001"          => SEG7 <= "10011111";
                                    when"0010"          => SEG7 <= "00100101";
                                    when"0011"          => SEG7 <= "00001101";
                                    when"0100"          => SEG7 <= "10011001";
                                    when"0101"          => SEG7 <= "01001001";
                                    when"0110"          => SEG7 <= "01000001";
                                    when"0111"          => SEG7 <= "00011011";
                                    when"1000"          => SEG7 <= "00000001";
                                    when"1001"          => SEG7 <= "00001001";
                                    when others         => SEG7 <= "XXXXXXXX";
                        end case;
            end process;
end RTL;
```

４．②　１０秒カウンタ　解答（例）

```
--   VHDL課題      10秒カウンタ（ｸﾛｯｸｼﾞｪﾈﾚｰﾀ＋10進ｶｳﾝﾀ＋7ｾｸﾞﾃﾞｺｰﾀﾞ）
--   非同期リセット入力,同期アップダウン入力付

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity COUNT10 is
   port (   CLK,R,UD :  in std_logic;
            AN       :  out std_logic_vector(3 downto 0);
            SEG7     :  out std_logic_vector(7 downto 0));
end COUNT10;

architecture RTL of COUNT10 is

component CLK_GENERATOR
   port (   CLK                  :  in std_logic;
            OUT_1S,OUT_10MS      :  out std_logic);
end component;

component COUNT10_RUD
   port (   CLK,E,R,UD     :  in std_logic;
            COUNT_OUT      :  out std_logic_vector(3 downto 0));
end component;

component DEC7_1
   Port ( BTN   :  in  std_logic_vector(3 downto 0);
          SEG7  :  out std_logic_vector(7 downto 0));
end component;

signal      CNT_CK1S      :  std_logic;
signal      C10_O         :  std_logic_vector(3 downto 0);

begin
      U0   :  CLK_GENERATOR
               port map (CLK => CLK,OUT_1S => CNT_CK1S);

      U1   :  COUNT10_RUD
               port map (CLK => CLK,E => CNT_CK1S,R => R,UD => UD,COUNT_OUT => C10_O);

      U2   :  DEC7_1
               port map (BTN => C10_O,SEG7 => SEG7);

AN <= "1110" ;

end RTL;
```

４．②　１０秒カウンタ　コンポーネント用　クロックジェネレータ

```
--      VHDL課題        クロックジェネレータ                                    <コンポーネント用>
--      50MHzのシステムクロック入力(CLK)に対して、1s間隔と10ms間隔でﾊﾟﾙｽ出力する。(OUT_1S,OUT_10MS)

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity CLK_GENERATOR is                                 --      エンティティの定義
   port (  CLK             : in std_logic;
           OUT_1S,OUT_10MS : out std_logic);
end CLK_GENERATOR;

architecture RTL of CLK_GENERATOR is                    --      アーキテクチャの定義

signal CNT_1S           : integer range 0 to 49999999;
signal CNT_10MS         : integer range 0 to 499999;
signal SIG_1S,SIG_10MS  : std_logic;

begin
        OUT_1S   <= SIG_1S;                             --      1sパルス出力。1sごとに50MHz幅の1パルスを出力。
        OUT_10MS <= SIG_10MS;                           --      10msパルス出力。10msごとに50MHz幅の1パルスを出力。

        process(CLK)                                    --      1sパルス生成部(カウント用)
        begin
                if(CLK'event and CLK = '1')then
                        if(CNT_1S = 49999999)then
                                CNT_1S <= 0;
                                SIG_1S <= '1';          --      パルス生成
                        else
                                CNT_1S <= CNT_1S + 1;
                                SIG_1S <= '0';
                        end if;
                end if;
        end process;

        process(CLK)                                    --      10msパルス生成部(7セグメントLEDダイナミック表示用)
        begin
                if(CLK'event and CLK = '1')then
                        if(CNT_10MS = 499999)then
                                CNT_10MS <= 0;
                                SIG_10MS <= '1';        --      パルス生成
                        else
                                CNT_10MS <= CNT_10MS + 1;
                                SIG_10MS <= '0';
                        end if;
                end if;
        end process;
end RTL;
```

４．②　１０秒カウンタ　コンポーネント用　１０進カウンタ

```
--        VHDL課題          10進カウンタ                                          <コンポーネント用>
--        (非同期リセット入力,同期アップダウン入力付)
--        50MHzのクロックで動作する基本カウンタ。他エンティティへのコンポーネント用。

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity COUNT10_RUD is                                    -- エンティティの定義
   port (  CLK,E,R,UD    : in std_logic;                 -- CLK:ｼｽﾃﾑｸﾛｯｸ／E:1s間隔ﾊﾟﾙｽ入力／UD:UP/DOWN入力
           COUNT_OUT  : out std_logic_vector(3 downto 0)); -- COUNT_OUT:10進カウンタ出力
end COUNT10_RUD;

architecture RTL of COUNT10_RUD is                       -- アーキテクチャの定義

signal CNT10                  : std_logic_vector(3 downto 0);
begin
          process(CLK,R)                                 -- 同期回路はシステムクロックで動作させる。Rは非同期Reset入力
          begin
                    if(R = '1')then
                              CNT10 <= "0000";
                    elsif(CLK'event and CLK = '1')then
                              if(E = '1')then
                                        if(UD = '1')then
                                                  if(CNT10 = "1001")then
                                                            CNT10 <= "0000";
                                                  else
                                                            CNT10 <= CNT10 + 1;
                                                  end if;
                                        else
                                                  if(CNT10 = "0000")then
                                                            CNT10 <= "1001";
                                                  else
                                                            CNT10 <= CNT10 - 1;
                                                  end if;
                                        end if;
                              end if;
                    end if;
          end process;

          COUNT_OUT <= CNT10;
end RTL;
```

４．②　１０秒カウンタ　コンポーネント用　７セグメントＬＥＤデコーダ

```
--        VHDL課題          7セグメントLEDデコーダ                    ＜コンポーネント用＞
--        7セグメントLEDはアノードコモン（点灯は負論理）

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity DEC7_1 is
    Port ( BS          : in  std_logic_vector(3 downto 0);
           SEG7        : out std_logic_vector(7 downto 0));
end DEC7_1;

architecture RTL of DEC7_1 is
begin
            process(BS)
            begin
                        case BS is
                                    when"0000" => SEG7 <= "00000011";
                                    when"0001" => SEG7 <= "10011111";
                                    when"0010" => SEG7 <= "00100101";
                                    when"0011" => SEG7 <= "00001101";
                                    when"0100" => SEG7 <= "10011001";
                                    when"0101" => SEG7 <= "01001001";
                                    when"0110" => SEG7 <= "01000001";
                                    when"0111" => SEG7 <= "00011011";
                                    when"1000" => SEG7 <= "00000001";
                                    when"1001" => SEG7 <= "00001001";
                                    when others => SEG7 <= "XXXXXXXX";
                        end case;
            end process;
end RTL;
```

４．③　６０秒カウンタ　解答（例）

```
--          VHDL課題  60秒カウンタ
--          (ｸﾛｯｸｼﾞｪﾈﾚｰﾀ、10進ｶｳﾝﾀ、6進ｶｳﾝﾀ、7ｾｸﾞﾃﾞｺｰﾀﾞをコンポーネントする)

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity COUNT60 is
    port (  CLK,R,UD            : in std_logic;
            AN                  : out std_logic_vector(3 downto 0);
            SEG7                : out std_logic_vector(7 downto 0));
end COUNT60;

architecture RTL of COUNT60 is

component CLK_GENERATOR
    port (  CLK                 : in std_logic;
            OUT_1S,OUT_10MS     : out std_logic);
end component;

component COUNT10_RUD
    port (  CLK,E,R,UD          : in std_logic;
            COUNT_OUT           : out std_logic_vector(3 downto 0));
end component;

component COUNT6_RUD
    port (  CLK,E,R,UCY,DCY     : in std_logic;
            COUNT_OUT           : out std_logic_vector(3 downto 0));
end component;

component DEC7_1
    Port ( BTN                  : in  std_logic_vector(3 downto 0);
           SEG7                 : out std_logic_vector(7 downto 0));
end component;


signal      CNT_CK1S,CNT_CK10MS,UCY,DCY     :           std_logic;
signal      AN_S                :           std_logic_vector(3 downto 0) := "1110";
signal      C10_O,C6_O          :           std_logic_vector(3 downto 0);
signal      DEC7_10,DEC7_6      :           std_logic_vector(7 downto 0);
```

つづき

```
begin
	U0 : CLK_GENERATOR
	port map (CLK => CLK,OUT_1S => CNT_CK1S,OUT_10MS => CNT_CK10MS);

	U1 : COUNT10_RUD
	port map (CLK => CLK,E => CNT_CK1S,R => R,UD => UD,COUNT_OUT => C10_O);

	U2 : COUNT6_RUD
	port map (CLK => CLK,E => CNT_CK1S,R => R,UCY => UCY,DCY => DCY,COUNT_OUT => C6_O);

	U3 : DEC7_1
	port map (BTN => C10_O,SEG7 => DEC7_10);

	U4 : DEC7_1
	port map (BTN => C6_O, SEG7 => DEC7_6);

	process(CLK)			--10進カウンタの出力を評価してｷｬﾘｰを生成する。
	begin
		if(CLK'event and CLK = '1')then
			if(UD = '1' and C10_O = "1001")then
				UCY <= '1';
				DCY <= '0';
			elsif(UD = '0' and C10_O = "0000")then
				UCY <= '0';
				DCY <= '1';
			else
				UCY <= '0';
				DCY <= '0';
			end if;
		end if;
	end process;

	process(CLK)
	begin
		if(CLK'event and CLK = '1')then
			if(CNT_CK10MS = '1')then
				if(AN_S = "1110")then
					AN_S <= "1101";
					SEG7 <=  DEC7_6;
				elsif(AN_S = "1101")then
					AN_S <= "1110";
					SEG7 <=  DEC7_10;
				else
					AN_S <= "XXXX";
					SEG7 <= "XXXXXXXX";
				end if;
			end if;
		end if;
	end process;

AN <= AN_S;

end RTL;
```

４．③　６０秒カウンタ　コンポーネント用　６進カウンタ

```
--        VHDL課題          6進カウンタ                                                    ＜コンポーネント用＞
--        （非同期リセット入力,同期アップダウンキャリー入力付）
--        6進ｶｳﾝﾀなので、出力は3ﾋﾞｯﾄで足りるが、ﾃﾞｺｰﾀﾞの入力が4ﾋﾞｯﾄなので、4ﾋﾞｯﾄ出力とする。

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity COUNT6_RUD is                                --    エンティティの定義
    port (  CLK,E,R,UCY,DCY   : in std_logic;       --    CLK：ｼｽﾃﾑｸﾛｯｸ／E：1s間隔ﾊﾟﾙｽ入力／UCY,DCY：UPｷｬﾘとDOWNｷｬﾘ
            COUNT_OUT         : out std_logic_vector(3 downto 0));
end COUNT6_RUD;                                     --    COUNT_OUT：6進カウンタ出力

architecture RTL of COUNT6_RUD is                   --    アーキテクチャの定義

signal CNT6                   : std_logic_vector(3 downto 0);

begin
            process(CLK,R)                          --    同期回路はシステムクロックで動作させる。Rは非同期Reset入力
            begin
                        if(R = '1')then
                                    CNT6 <= "0000";
                        elsif(CLK'event and CLK = '1')then
                                    if(E ='1')then
                                                if(UCY = '1')then                   --    UCYとDCYは同時にアクティブになることはない。
                                                            if(CNT6 = "0101")then
                                                                        CNT6 <= "0000";
                                                            else
                                                                        CNT6 <= CNT6 + 1;
                                                            end if;
                                                elsif(DCY = '1')then
                                                            if(CNT6 = "0000")then
                                                                        CNT6 <= "0101";
                                                            else
                                                                        CNT6 <= CNT6 - 1;
                                                            end if;
                                                end if;
                                    end if;
                        end if;
            end process;

            COUNT_OUT <= CNT6;                      --    6進カウンタの出力（デコーダに合わせて4ビット出力））
end RTL;
```

その他に、４．②で作成した　クロックジェネレータ　、　１０進カウンタ　、　７セグメントＬＥＤデコーダ　をコンポーネントする。

作業工程計画書

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備	作業場所の確認・整理 実習装置と開発環境の動作確認	
1.　回路設計（基本回路） 2.　回路設計（アップダウンカウンタ）	Dフリップフロップ ・入力ボタンの回路確認（正論理/負論理） ・表示用LEDの回路確認（正論理/負論理） ・Dフリップフロップの機能の確認 ・入出力と内部信号の確認 ・回路設計 ・入出力の割付 半加算器 ・半加算器の機能の確認 ・入出力と内部信号の確認 ・回路設計 全加算器 ・全加算器の機能の確認 ・入出力と内部信号の確認 ・半加算器を用いた階層設計 デコーダ回路 ・7セグメントLEDの回路確認（正論理/負論理） ・7セグメントLEDの点灯データを作成 ・指定外の7セグメントLEDが点灯しないようにする ・以降の課題で活用できるように入出力を整理する 10 秒カウンタ回路 ・分周回路設計（1s、10ms）の設計 （10ms 出力は次の課題用） ・非同期リセット入力、同期アップダウン入力 ・デコーダ回路の組込み ・指定外の7セグメントLEDが点灯しないようにする ・以降の課題で活用できるように入出力を整理する 60 秒カウンタ回路 ・6 進カウンタ回路の設計 ・デコーダ回路、分周回路、10 進カウンタの組込み ・6 進カウンタ回路へのキャリー信号生成 ・ダイナミック表示 ・単相同期回路としてまとめる	

訓 練 課 題 確 認 シ ー ト

訓練科名　：　制御技術科

仕上がり像　：　電気・電子回路の設計及び同回路を用いた制御ができる。

システム名　：　ディジタル回路設計技術　　入所期　：

訓練課題名　：　HDLによるディジタル回路設計・製作　　氏　名　：

評価区分	評価項目	細　目	評価（数値）					評価判定	評価基準
作業時間	作業準備時間	作業環境の確認	1				5		配布したサンプルコードを用いて作業環境の確認を行う。突然の故障を除いて、開始から10分以内に確認ができれば5点、それ以外は1点と
	作業時間	プログラミング	2	4	6	8	10		指定した作業時間以内に、全作業の動作確認まで終われば10点、5分遅れるごとに2点減点する。（30分で打切り）
作業工程	作業工程における留意事項等	作業工程手順	1	2	3	4	5		エディタを用いたソースコードの作成、PLDの開発環境を用いた論理合成、デバイスへの書き込み、動作確認まで、一連の作業を滞りなく完了させることができれば5点。
デジタル回路設計	基本回路設計	Dフリップフロップの設計	1	2	3	4	5		動作に問題がなく、仕様通りに設計できていれば5点、ソースコードも含めて確認した上で不備があれば、1個所につき1点減点し、最低点を1点とする。
		半加算器の設計	1	2	3	4	5		
		全加算器の設計（階層設計）	1	2	3	4	5		
	デコーダ回路設計	デコーダ回路の設計	1	2	3	4	5		
	10秒カウンタ回路設計	分周回路の設計	1	2	3	4	5		
		同期・非同期入力	1	2	3	4	5		
		階層設計	1	2	3	4	5		
	60秒カウンタ回路設計	6進カウンタ	1	2	3	4	5		
		単相同期回路	1	2	3	4	5		
		同期・非同期入力	1	2	3	4	5		
		階層設計	1	2	3	4	5		
		分周回路の設計	1	2	3	4	5		
安全作業	安全作業	他の作業者への妨げ行為	1	2	3	4	5		持ち点を5点とし、他の作業者への不適切な作業又は行為があるごとに1点ずつ減点し、最低点を1点とする。
		VDT作業	1				5		不適切な姿勢で作業をしている場合や、指示通りに休憩を取らない場合は注意する。注意に従わない場合は1点、注意する必要がない場合や、注意に従う場合は5点
工夫・改善	工夫・改善	装置の動作や機能の工夫・改善	2	4	6	8	10		装置の動作や機能、ソースコードの記述（可読性）に何かしらの工夫・改善がされていなければ2点とし、工夫・改善点1件につき2点ずつ加算し、最高点を10点とする。
	工夫・改善点記入欄		総点						＜判定表＞ A　：　80点以上　：到達水準を十分に上回った B　：　60点以上80点未満　：到達水準に達した C　：　60点未満　：到達水準に達しなかった
			合計点						
			換算点						
			総合評価判定						

訓練課題のねらい	コメント
HDLによるデジタル回路設計技術の基礎を修得しているかどうかを実技により確認します。	担当指導員氏名：

評価要領

訓練科名　：　制御技術科
仕上がり像　：　電気・電子回路の設計及び同回路を用いた制御ができる。
システム名　：　ディジタル回路設計技術
訓練課題名　：　HDLによるディジタル回路設計・製作

評価区分	評価項目	細　目	評価要領（採点要領）	備考
作業時間	作業準備時間	開発環境の準備	時間を計測しながら、訓練生の作業をよく観察すること。	
	作業時間	プログラミング	時間を計測しながら、訓練生の作業をよく観察すること。	
作業工程	作業工程における留意事項等	作業工程手順	訓練生の作業をよく観察し、必要があれば助言をしてもよい。	
ディジタル回路設計	基本回路設計	Dフリップフロップの設計	正常動作とソースコードの確認。	
		半加算器の設計	正常動作とソースコードの確認。	
		全加算器の設計（階層設計）	正常動作とソースコードの確認。 前の課題で設計した半加算器を利用できているか。	
	デコーダ回路設計	デコーダ回路の設計	正常動作とソースコードの確認。 組合せ回路の設計ができているか。 表示させるLEDが特定できているか。	
	10秒カウンタ回路設計	分周回路の設計	正常動作とソースコードの確認。 単相同期回路の設計ができているか。 周期が1sのパルスが生成できているか。	
		同期・非同期入力	正常動作とソースコードの確認。 非同期リセット、同期アップダウンとなっているか。	
		階層設計	正常動作とソースコードの確認。 前の課題を活用した階層設計ができているか。	
	60秒カウンタ回路設計	6進カウンタ	正常動作とソースコードの確認。 アップ入力、ダウン入力が準備されており、正常に機能するか。	
		単相同期回路	正常動作とソースコードの確認。 単相同期回路として設計できているか。	
		同期・非同期入力	正常動作とソースコードの確認。 非同期リセット、同期アップダウンとなっているか。	
		階層設計	正常動作とソースコードの確認。 前の課題を活用した階層設計ができているか。	
		分周回路の設計	正常動作とソースコードの確認。 10ms周期でダイナミック表示ができているか。 1秒間隔でカウントアップしているか。	可能であればオシロスコープで確認する。ソースコードより確認しても可。
安全作業	安全作業	他の作業者への妨げ行為	訓練生の作業態度をよく観察し、必要があれば注意をすること。	
		VDT作業	訓練生の作業姿勢をよく観察し、必要があれば注意をすること。	
工夫・改善	工夫・改善	装置の動作や機能の工夫・改善	何かしらの、装置のユーザーにとって有用な機能が追加されていれば加点する。ソースコードの可読性についても評価対象とする。 例：入力ボタンの工夫、データロード機能の追加、表示LEDの追加、同期リセットの追加など	

筆記課題

管理番号：Ｅ－42

「ＨＤＬによるディジタル回路設計」

■課題概要■

論理回路の基本事項と、HDLを用いたディジタル回路設計のために必要な知識（数値データの表現方法、組合せ回路と順序回路、各種のフリップフロップ回路、回路の同期動作）およびHDLによるディジタル回路の記述について確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領		
訓練課題	○	E-42-01_訓練課題.doc
解答	○	E-42-02_解答及び解説.doc
作業工程手順書		
訓練課題確認シート		
評価要領		

筆記課題

「ＨＤＬによるディジタル回路設計」

1. 作業時間
   90 分（休憩を除く）

2. 配付資料
   問題用紙、解答用紙

3. 課題作成、提出方法
   解答用紙のみを回収する

１．ＡＮＤ素子の図記号（MIL 記号）、真理値表、論理式を書いて下さい。

２．次の論理式をブール代数を用いて簡単化しなさい。

$Y = A \cdot B \cdot C + B(B+C) + \overline{A} \cdot C$

３．次の論理式をそのまま論理回路図で描きなさい。

$Y = \overline{A} \cdot B + B \cdot C + \overline{D}$

４．次の仕様のようなオーディションの合否回路について答えなさい。

仕様

・審査員はA，B，C，Dの４名とし、それぞれが１個ずつの投票ボタンを持つ。
・Ａを委員長とする。
・多数決で合否を決めるが、２－２の場合は委員長の判断が優先されることとする。
・合格の場合は出力Ｙ（ＬＥＤ）がＯＮとなる。
・入出力ともに正論理で表現する。
・保持機能やリセット機能は不要とする。よって、４入力１出力の回路とする。

① この回路の真理値表を作りなさい。
② 真理値表を元にカルノー図を作りなさい。
③ この回路を論理式で表しなさい。
④ 論理回路図を描きなさい。（多入力論理素子の使用可）
⑤ 空欄を埋めてこの回路のＨＤＬ記述を完成させなさい。（ライブラリの宣言は省略してあります）

```
entity CONTEST is
    port ( BTN      : in   std_logic_vector(3 downto 0);          -- 投票ボタン
           LED      : out std_logic);                             -- 出力LED
end CONTEST;

architecture RTL of CONTEST is
begin
        process(      ア      )
        begin
                  イ      BTN is
                        when"0000"      => LED <= '0';
                        when"0001"      => LED <= '0';
                        when"0010"      => LED <= '0';
                        when"0011"      => LED <= '0';
                        when"0100"      => LED <= '0';
                        when"0101"      => LED <= '0';
                        when"0110"      => LED <= '0';
                        when"0111"      => LED <= '1';
                        when"1000"      => LED <=  ウ  ;
                        when"1001"      => LED <= '1';
                        when"1010"      => LED <= '1';
                        when"1011"      => LED <= '1';
                        when"1100"      => LED <= '1';
                        when"1101"      => LED <= '1';
                        when"1110"      => LED <= '1';
                        when"1111"      => LED <= '1';
                              エ                    ;
                end  イ  ;
        end process;
end RTL;
```

５．下表の（ア）～(エ)の空欄に入る語句をａ～ｇのうちから選び、表を完成させなさい。
　また、（オ）～（ク）に入る数値を記入しなさい。

10進数	（ア）	（イ）	（ウ）	（エ）
0	00000000	0	00000000	0
1	00000001	1	00000001	1
ﾚ	ﾚ	ﾚ	ﾚ	ﾚ
6	00000110	6	00000110	6
7	00000111	7	00000111	7
8	00001000	10	00001000	8
9	00001001	11	00001001	9
10	（オ）	12	（キ）	A
11	00010001	13	00001011	B
ﾚ	ﾚ	ﾚ	ﾚ	ﾚ
15	00010101	17	00001111	F
16	00010110	（カ）	00010000	（ク）

＜語句＞

ａ：２進数　ｂ：５進数　ｃ：６進数　ｄ：８進数　ｅ：16進数　ｆ：ＢＣＤコード　ｇ：グレイコード

６．下記文中の括弧に入る語句をａ～ｎより選び、記号で答えなさい。（使用しない語句もあります）

論理回路は、（①）と（②）に大別することができる。（①）は、その時の入力の状態のみで出力が決まる回路で、以前の回路状態に依存しない。（②）はその時の入力の状態とそれ以前の状態で出力が決まる回路で、内部に記憶回路を有している。例えばカウンタ回路は（②）、（③）は（①）に分類される。

（②）が有する記憶回路のことをフリップフロップ回路という。フリップフロップ回路にはいくつかの種類があり、それぞれの動作によって名称が異なる。下図の真理値表のような動作をするフリップフロップ回路を、それぞれ表１（④）、表２（⑤）、表３（⑥）と呼ぶ。

表1

入力		出力
A	B	Q
0	0	Qn
0	1	0
1	0	1
1	1	禁止

表2

入力		出力
A	CLK	Q
0	×	0
1	↑	$\overline{Qn}$

×：don't care
↑：立上りエッジ

表3

入力		出力
A	CLK	Q
0	↑	0
1	↑	1
×	H/L	Qn

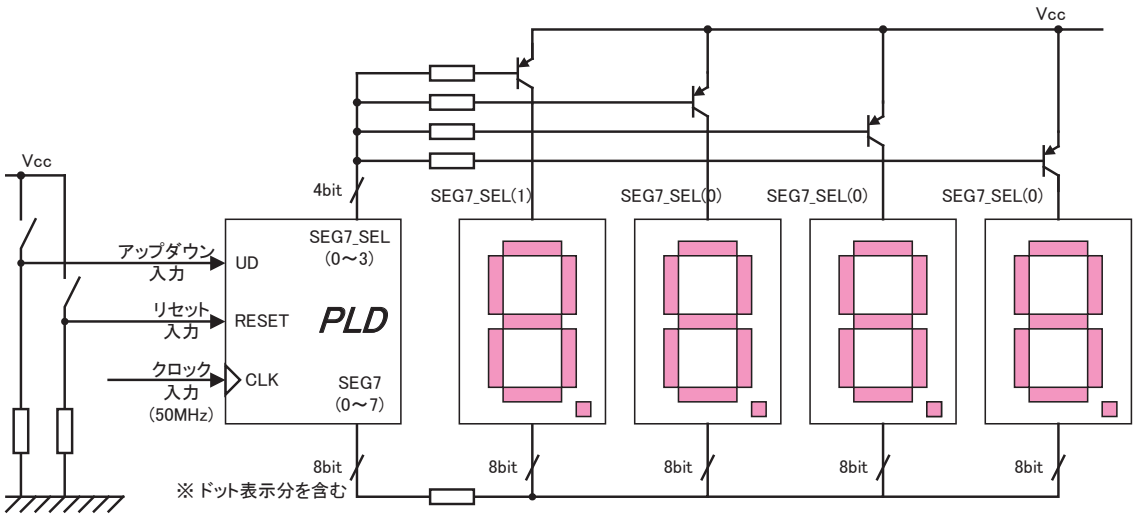
フリップフロップ回路は、その動作によって（⑦）と（⑧）に分類される。（⑦）のフリップフロップ回路は、入力される基準パルス（一般的にクロックと呼ぶ）が変化するタイミング（エッジ）で出力が変化し、（⑧）のフリップフロップ回路は、基準パルスのタイミングとは無関係に、入力信号が変化するタイミングで出力が変化する。表１は（⑧）のフリップフロップ回路、表２と表３は（⑦）のフリップフロップ回路である。

ＨＤＬを用いたデジタル回路設計においては、単一のクロックのタイミングで全ての回路が動作する（⑨）として設計する。ＰＬＤの内部で複数のクロックが用いられると、回路の動作タイミングの検証が困難となり、後段の回路において（⑩）が発生する原因になる。

＜語句＞

ａ：単相同期回路　ｂ：非同期型　ｃ：クロストーク
ｄ：Ｄフリップフロップ　ｅ：組み合わせ回路　ｆ：シフトレジスタ
ｇ：シーケンサ　ｈ：Ｔフリップフロップ　ｉ：スキュー
ｊ：同期型　ｋ：ジッタ　ｌ：ＲＳフリップフロップ
ｍ：デコーダ　ｎ：順序回路　ｏ：ラッチ

7．下図は PLD と 7 セグメント LED の接続図です。このハードウェアを用いて、HDL（VHDL）による 1 桁のカウンタ回路を設計しました。以下の問いに答えなさい。
（HDL による記述では、ライブラリの宣言は所略してあります）



このカウンタ回路は、電源が入ると 1 秒間隔でカウントアップし、その数値を 7 セグメントＬＥＤに表示します。 9 までカウントしたら 0 に戻り、そのままカウントを継続します。
リセットボタンが押されると、カウント値が 0 になります。なお、システムクロックは 50MHz とします。

① A の部分は何をしているのでしょうか。

② B の部分は何をしているのでしょうか。

③ C の部分は何をしているのでしょうか。

④ このカウンタ回路のリセット入力は同期リセットでしょうか。非同期リセットでしょうか。また、リセットボタンを押してから、どのタイミングでリセット機能が有効になるかを答えなさい。

```
entity COUNT is
port(CLK,RESET,UD   : in std_logic;          --  UD は③の課題用。それ以外では使用しない。
     SEG7_LED       : out std_logic_vector(7 downto 0);
     SEG7_SELECT    : out std_logic_vector (3 downto 0));
end COUNT;

architecture RTL of COUNT is
signal COUNT: integer range 0 to 49999999;
signal DO  : std_logic;
signal WORK : std_logic_vector(3 downto 0);

begin
     SEG7_SELECT <= "1110";        ・・・・・・・・・・・・・・・・・・・・・・・・A

     process(CLK)                                          ┐
     begin                                                 │
          if(CLK'event   and   CLK   =   '1') then         │
               COUNT <= COUNT + 1;                         │
          end if;                                          │
     end process;                                          │
                                                           │
     process(COUNT)                                        ├ ・・・・・・・・・・・・・・・B
     begin                                                 │
          if (COUNT = 49999999 ) then                      │
               DO   <=   '1';                              │
          else                                             │
               DO   <=   '0';                              │
          end   if;                                        │
     end process;                                          ┘

     process(CLK)
     begin
          if(CLK'event and CLK = '1')then
               if(DO = '1') then
                    if ( RESET = '1' ) then
                         WORK <=   "0000";
                    else
                         WORK <=   WORK + '1';
                    end if;
               end if;
          end   if;
     end process;

     process(WORK)                                                    ┐
     begin                                                            │
          case WORK   is                                              │
               when "0000" => SEG7_LED <= "00000011";   -- 0          │
               when "0001" => SEG7_LED <= "10011111";   -- 1          │
               when "0010" => SEG7_LED <= "00100101";   -- 2          │
               when "0011" => SEG7_LED <= "00001101";   -- 3          │
               when "0100" => SEG7_LED <= "10011001";   -- 4          │
               when "0101" => SEG7_LED <= "01001001";   -- 5          ├ ・・・C
               when "0110" => SEG7_LED <= "01000001";   -- 6          │
               when "0111" => SEG7_LED <= "00011011";   -- 7          │
               when "1000" => SEG7_LED <= "00000001";   -- 8          │
               when "1001" => SEG7_LED <= "00001001";   -- 9          │
               when others => SEG7_LED <= "XXXXXXXX";                 │
          end case;                                                   │
     end process;                                                     ┘
end RTL;
```

ＨＤＬによるディジタル回路設計　解答用紙　１／２

入所年月	番号	氏　名	合計点	評価判定
平成　　年　　月入所			点	

1

MIL 記号　（２点）

論理式　（２点）

真理値表　（２点）

A	B	Y
0	0	
0	1	
1	0	
1	1	

2　（５点）

3　（５点）

4

① 真理値表　（３点）

A	B	C	D	Y
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

② カルノー図　（３点）

③ 論理式　（３点）

④ 回路図　（３点）

⑤ HDL 記述　（各２点）

ア		イ	
ウ		エ	

入所年月	番号	氏　名
平成　　年　　月入所		

（各２点）

5	ア		イ		ウ		エ	
	オ		カ		キ		ク	

（各２点）

6	①		②		③		④		⑤	
	⑥		⑦		⑧		⑨		⑩	

7	①	（６点）
	②	（６点）
	③	（６点）
	④	（１０点）

筆記課題　解答及び解説

「ＨＤＬによるディジタル回路設計」

１　作業時間

90 分（休憩を除く）

２　配付資料

問題用紙、解答用紙

３　課題作成、提出方法

解答用紙のみを回収する

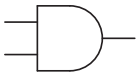
ＨＤＬによるデジタル回路設計　解答用紙　１／２

入所年月	番号	氏名	合計点	評価判定
平成　年　月入所			点	

評価基準　　A:80点以上　　B:60点～79点　　C:60点未満

1

MIL記号　（2点）



論理式　（2点）

Y = A･B

真理値表　（2点）

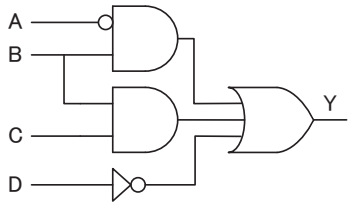
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

2

$Y = B + \overline{A}\cdot C$　（5点）

3

（5点）



4

① 真理値表　（3点）

A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1
1	0	0	0	1
1	0	0	1	1

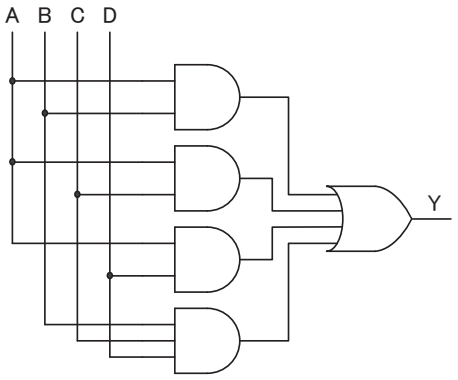
② カルノー図　（3点）

AB ＼ CD	00	01	11	10
00				
01		1		
11	1	1	1	1
10		1	1	1

③ 論理式

$Y = A\cdot B + A\cdot C + A\cdot D + B\cdot C\cdot D$　（3点）

④ 回路図　（3点）



⑤ HDL記述　（各2点）

ア	BTN	イ	case
ウ	'0'	エ	when others =>LED<= 'X' ※ X は、0,1,Zなどでも可

ＨＤＬによるデジタル回路設計　解答用紙　２／２

入所年月	番号	氏名
平成　年　月入所		

（各2点）

5	ア	f	イ	d	ウ	a	エ	e
	オ	00010000	カ	20	キ	00001010	ク	10

（各2点）

6	①	e	②	m	③	l	④	k	⑤	h
	⑥	d	⑦	j	⑧	b	⑨	a	⑩	i

7	①	（6点） 点灯させる7セグメントLEDを選択している。
	②	（6点） 1秒間隔のパルス信号を生成している。
	③	（6点） バイナリのカウント値を、7セグメントLEDに数値表示するための信号に変換している。（デコーダ回路）
	④	（10点） プロセス文のセンシティビティリストにRESETがないので、クロック入力（CLK）に同期して機能する同期リセット入力である。 50MHzのシステムクロックに同期した上で、B部で生成した1秒間隔のパルスに合わせてリセットが有効になる。

１．ＡＮＤ素子の図記号（MIL記号）、真理値表、論理式を書いて下さい。　（各２点：計６点）

２．次の論理式をブール代数を用いて簡単化しなさい。　（5 点）

$Y = A \cdot B \cdot C + B(B+C) + \overline{A} \cdot C$

３．次の論理式をそのまま論理回路図で描きなさい。　（5 点）

$Y = \overline{A} \cdot B + B \cdot C + \overline{D}$

４．次の仕様のようなオーディションの合否回路について答えなさい。　（計 20 点）

仕様

・審査員はA，B，C，Dの４名とし、それぞれが１個ずつの投票ボタンを持つ。
・Ａを委員長とする。
・多数決で合否を決めるが、２－２の場合は委員長の判断が優先されることとする。
・合格の場合は出力Ｙ（ＬＥＤ）がＯＮとなる。
・入出力ともに正論理で表現する。
・保持機能やリセット機能は不要とする。よって、４入力１出力の回路とする。

① この回路の真理値表を作りなさい。　（3 点）
② 真理値表を元にカルノー図を作りなさい。　（3 点）
③ この回路を論理式で表しなさい。　（3 点）
④ 論理回路図を描きなさい。（多入力論理素子の使用可）　（3 点）
⑤ 空欄を埋めてこの回路のＨＤＬ記述を完成させなさい。（ライブラリの宣言は省略してあります）
（各２点：計８点）

```
entity CONTEST is
    port ( BTN      : in   std_logic_vector(3 downto 0);        -- 投票ボタン
           LED      : out std_logic);                           -- 出力LED
end CONTEST;

architecture RTL of CONTEST is
begin
        process( [ア]
        begin
                [イ] BTN is
                        when"0000"      => LED <= '0';
                        when"0001"      => LED <= '0';
                        when"0010"      => LED <= '0';
                        when"0011"      => LED <= '0';
                        when"0100"      => LED <= '0';
                        when"0101"      => LED <= '0';
                        when"0110"      => LED <= '0';
                        when"0111"      => LED <= '1';
                        when"1000"      => LED <= [ウ];
                        when"1001"      => LED <= '1';
                        when"1010"      => LED <= '1';
                        when"1011"      => LED <= '1';
                        when"1100"      => LED <= '1';
                        when"1101"      => LED <= '1';
                        when"1110"      => LED <= '1';
                        when"1111"      => LED <= '1';
                        [エ];
                end [イ];
        end process;
end RTL;
```

５．下表の（ア）～(エ)の空欄に入る語句をａ～ｇのうちから選び、表を完成させなさい。
また、（オ）～（ク）に入る数値を記入しなさい。　（各２点：計 16 点）

10進数	（ア）	（イ）	（ウ）	（エ）
0	00000000	0	00000000	0
1	00000001	1	00000001	1
~	~	~	~	~
6	00000110	6	00000110	6
7	00000111	7	00000111	7
8	00001000	10	00001000	8
9	00001001	11	00001001	9
10	（オ）	12	（キ）	A
11	00010001	13	00001011	B
~	~	~	~	~
15	00010101	17	00001111	F
16	00010110	（カ）	00010000	（ク）

ａ：２進数　ｂ：５進数　ｃ：６進数　ｄ：８進数　ｅ：16 進数　ｆ：ＢＣＤコード　ｇ：グレイコード

６．下記文中の括弧に入る語句をａ～より選び、記号で答えなさい。　（各２点：計 20 点）

論理回路は、（①）と（②）に大別することができる。（①）は、その時の入力の状態のみで出力が決まる回路で、以前の回路状態に依存しない。（②）はその時の入力の状態とそれ以前の状態で出力が決まる回路で、内部に記憶回路を有している。例えばカウンタ回路は（②）、（③）は（①）に分類される。

（②）が有する記憶回路のことをフリップフロップ回路という。フリップフロップ回路にはいくつかの種類があり、それぞれの動作によって名称が異なる。下図の真理値表のような動作をするフリップフロップ回路を、それぞれ表１（④）、表２（⑤）、表３（⑥）と呼ぶ。

表1

入力		出力
A	B	Q
0	0	Qn
0	1	0
1	0	1
1	1	禁止

表2

入力		出力
A	CLK	Q
0	×	0
1	↑	$\overline{Qn}$

×：don't care
↑：立上りエッジ

表3

入力		出力
A	CLK	Q
0	↑	0
1	↑	1
×	H/L	Qn

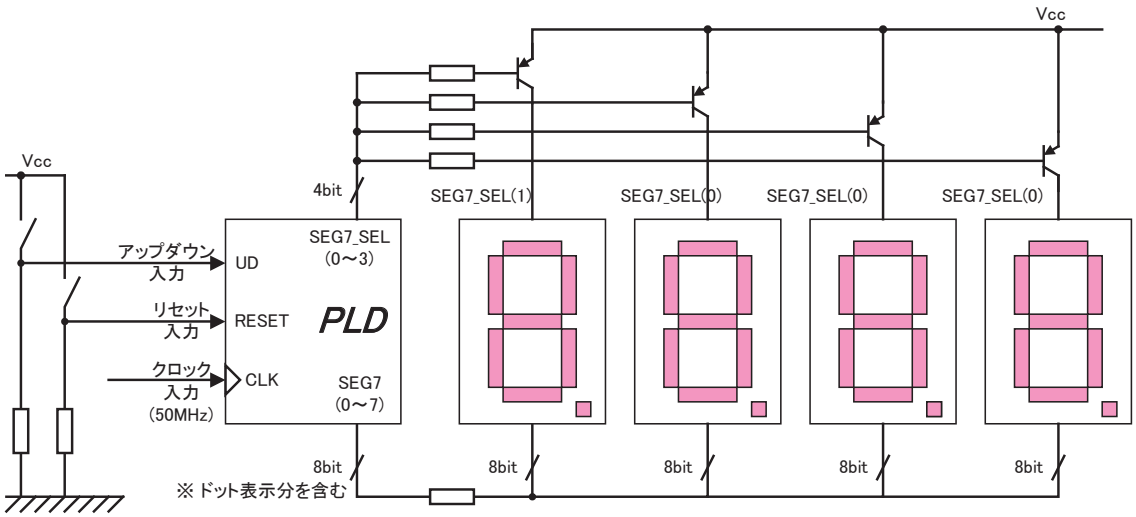
フリップフロップ回路は、その動作によって（⑦）と（⑧）に分類される。（⑦）のフリップフロップ回路は、入力される基準パルス（一般的にクロックと呼ぶ）が変化するタイミング（エッジ）で出力が変化し、（⑧）のフリップフロップ回路は、基準パルスのタイミングとは無関係に、入力信号が変化するタイミングで出力が変化する。表１は（⑧）のフリップフロップ回路、表２と表３は（⑦）のフリップフロップ回路である。

ＨＤＬを用いたデジタル回路設計においては、単一のクロックのタイミングで全ての回路が動作する（⑨）として設計する。ＰＬＤの内部で複数のクロックが用いられると、回路の動作タイミングの検証が困難となり、後段の回路において（⑩）が発生する原因になる。

ａ：単相同期回路　ｂ：非同期型　ｃ：クロストーク
ｄ：D フリップフロップ　ｅ：組み合わせ回路　ｆ：シフトレジスタ
ｇ：シーケンサ　ｈ：T フリップフロップ　ｉ：スキュー
ｊ：同期型　ｊ：ジッタ　ｋ：RS フリップフロップ
ｌ：デコーダ　ｍ：順序回路　ｎ：ラッチ

※　使用しないものもあります

7．下図は PLD と 7 セグメント LED の接続図です。このハードウェアを用いて、HDL（VHDL）による 1 桁のカウンタ回路を設計しました。以下の問いに答えなさい。
（HDLによる記述では、ライブラリの宣言は所略してあります）　（計 20 点）



このカウンタ回路は、電源が入ると 1 秒間隔でカウントアップし、その数値を 7 セグメントＬＥＤに表示します。 9 までカウントしたら 0 に戻り、そのままカウントを継続します。
リセットボタンが押されると、カウント値が 0 になります。なお、システムクロックは 50MHz とします。

① Aの部分は何をしているのでしょうか。

点灯させる 7 セグメントＬＥＤを選択している。（6 点）

② Bの部分は何をしているのでしょうか。

1 秒間隔のパルス信号を生成している。（6 点）

③ Cの部分は何をしているのでしょうか。

バイナリのカウント値を、 7 セグメントLEDに数値を表示するための信号に変換している。
（デコーダ回路）　（6 点）

④ このカウンタ回路のリセット入力は同期リセットでしょうか。非同期リセットでしょうか。また、リセットボタンを押してから、どのタイミングでリセット機能が有効になるかを答えなさい。

プロセス文のセンシティビティリストにRESETがないので、クロック入力（CLK）に同期して機能する同期リセット入力である。50MHzのシステムクロックに同期した上で、B部で生成した 1 秒間隔のパルスに合わせてリセットが有効になる。　（10 点）

```
entity COUNT is
port(CLK,RESET,UD   : in std_logic;          --  UD は③の課題用。それ以外では使用しない。
     SEG7_LED       : out std_logic_vector(7 downto 0);
     SEG7_SELECT    : out std_logic_vector (3 downto 0));
end COUNT;

architecture RTL of COUNT is
signal COUNT: integer range 0 to 49999999;
signal DO  : std_logic;
signal WORK : std_logic_vector(3 downto 0);

begin
     SEG7_SELECT <= "1110";        ・・・・・・・・・・・・・・・・・・・・・・・・A

     process(CLK)                                          ┐
     begin                                                 │
          if(CLK'event   and   CLK   =   '1') then         │
               COUNT <= COUNT + 1;                         │
          end if;                                          │
     end process;                                          │
                                                           │
     process(COUNT)                                        ├ ・・・・・・・・・・・・・・・B
     begin                                                 │
          if (COUNT = 49999999 ) then                      │
               DO   <=   '1';                              │
          else                                             │
               DO   <=   '0';                              │
          end   if;                                        │
     end process;                                          ┘

     process(CLK)
     begin
          if(CLK'event and CLK = '1')then
               if(DO = '1') then
                    if ( RESET = '1' ) then
                         WORK <=   "0000";
                    else
                         WORK <=   WORK + '1';
                    end if;
               end if;
          end   if;
     end process;

     process(WORK)                                                    ┐
     begin                                                            │
          case WORK   is                                              │
               when "0000" => SEG7_LED <= "00000011";   -- 0          │
               when "0001" => SEG7_LED <= "10011111";   -- 1          │
               when "0010" => SEG7_LED <= "00100101";   -- 2          │
               when "0011" => SEG7_LED <= "00001101";   -- 3          │
               when "0100" => SEG7_LED <= "10011001";   -- 4          │
               when "0101" => SEG7_LED <= "01001001";   -- 5          ├ ・・・C
               when "0110" => SEG7_LED <= "01000001";   -- 6          │
               when "0111" => SEG7_LED <= "00011011";   -- 7          │
               when "1000" => SEG7_LED <= "00000001";   -- 8          │
               when "1001" => SEG7_LED <= "00001001";   -- 9          │
               when others => SEG7_LED <= "XXXXXXXX";                 │
          end case;                                                   │
     end process;                                                     ┘
end RTL;
```

実技課題

管理番号:　Ｅ－４３
「マイコンによるアセンブリ言語を用いたモータ制御」

■課題概要■

マイコンでモータを制御する課題を通して、アセンブリ言語によるプログラミングを習得しているかを実技により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領	○	E-43-00_実施要領
訓練課題	○	E-43-01_訓練課題
解答	○	E-43-02_解答及び解説
作業工程手順書	○	E-43-03_作業工程手順書
訓練課題確認シート	○	E-43-04_訓練課題確認シート及び評価要領
評価要領	○	E-43-04_訓練課題確認シート及び評価要領

※フォルダ内に、マイコンのプログラムのデータがあります。

実技課題　実施要領

訓練課題名　「マイコンによるアセンブリ言語を用いたモータ制御」

・訓練課題は個人で行うことが望ましい。
・実施時間は、概要説明を含め、140 分を予定する。(休憩時間は含まない)
・機材は前日までに動作確認して、用意しておく。
・VDT 作業を考慮し、50 分を目安に 10 分程度の休憩時間を設ける。
・プログラムリストを提出する。
・この課題の回答では、H8/3052F マイコンボードを使用した例を示しているため、各施設で使用しているハードウェア（マイコンボードやモータドライバ IC など）に合わせて、課題を修正して実施することが望ましい。
・マイコンボードに接続するスイッチ回路や LED 回路、モータ駆動回路は、本実技課題の実施に先立って、動作確認が完了しているものを使用すること。

※施設の実習環境によっては、時間数や問題を修正して対応すること。

（実施例）

９：２０～９：３０　課題概要説明
９：３０～１０：００　パソコン、エディタ立ち上げ、プログラミング、動作確認
１０：００～１０：１０　（休憩）

１０：１０～１１：００　プログラミング、動作確認
１１：００～１１：１０　（休憩）

１１：１０～１２：００　プログラミング、動作確認
（１２：１０　打ち切り）

講師が動作確認し、OK を示された者で、時間が余るようであれば、別課題などを行う。

実技課題

「マイコンによるアセンブリ言語を用いたモータ制御」

１　作業時間

　　１４０分　（準備時間１０分を含む。休憩時間は除く。）

２　配付資料

　　問題用紙，解答用紙

３　課題作成、提出方法

　　プログラムリストを回収します。

　　課題が終了した時点で、指導員の動作確認を受けること。

１．課題内容

リモコンを用いて、離れた所にあるモータ制御装置の遠隔操作を行うという想定で、リモコンの制御プログラムを作成すること。下記に、リモコンとモータ制御装置の動作概要を示す。

・リモコンの回路は、事前に受講生が作成してあること。

・モータ制御装置は完成済みのものとし、リモコンの動作確認時に用いる。

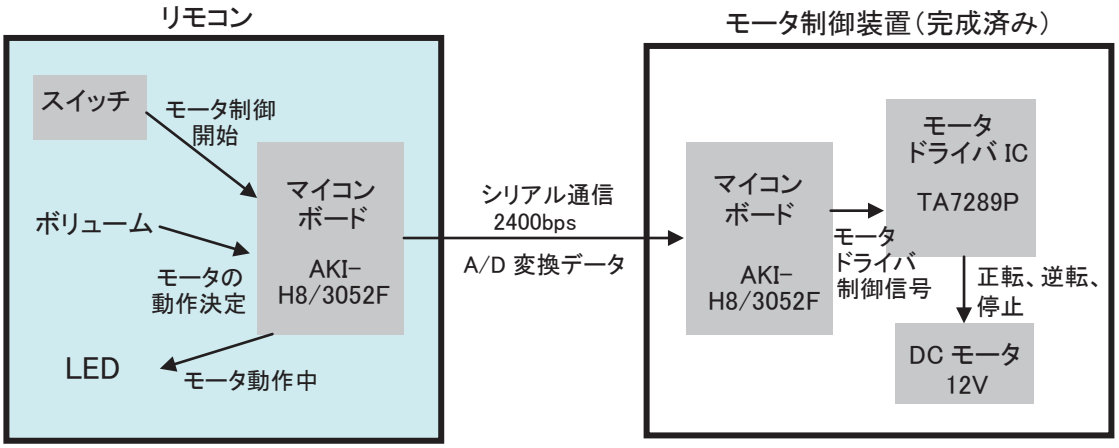


図 1　リモコン回路とモータ制御装置の構成

①リモコン概要

図 1 に示すように、リモコンはマイコンボードにスイッチ、ボリューム（半固定抵抗）、LED のそれぞれ一つずつが接続されている。下記に示す各部品の役割、機能に従って、マイコンのプログラムを作成し、動作を確認すること。

・スイッチ　　：電源投入後、このスイッチが押されるとモータ制御開始とする。

・ボリューム　：図 2 に示すように、つまみを回転させることで、ボリュームの出力電圧（0V～5V）を変化させる。この変化をモータの動作決定に利用する。

・LED　　：タイマの機能（ITU1）を用いて、モータ制御中は、点灯（0.5 秒）と消灯（0.5 秒）を繰り返すようにする。

・割り込みの機能（インターバルタイマ：TU0）を用いて、0.1 秒ごとに次の動作を実行する。

：ボリュームの出力電圧を、A/D 変換機能（AN0）を用いて取り込む。

：A/D 変換して得た値（A/D 変換データ）を、シリアル通信の機能（TXD1）を用いて、モータ制御装置側マイコンに送信する。

通信設定　　2400bps、調歩同期式モード、8 ビットデータ、パリティ付加、チェック禁止、1STOP ﾋﾞｯﾄ

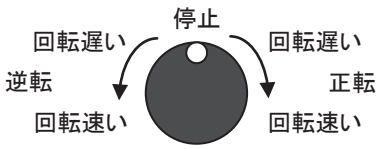


図 2　つまみの回転とモータ制御のイメージ

②モータ制御装置概要

図 1 に示すように、モータ制御装置は、マイコンボードにモータドライバ IC、DC モータが接続されている（詳細は図 4 に示す）。下記に示す各部品の役割、機能に従って動作する。

・電源投入後、自動的に、リモコンから送られる A/D 変換データを待つ状態になり、A/D 変換データを受信後、モータドライバ制御信号に変換（詳細は 2.補足資料⑤に示す）し、モータドライバへ出力する。その後、次の A/D 変換データを待つ状態になる。
・モータの動作は、次のモータドライバ制御信号が出力されるまで、現在の状態を保持する。
・電源を OFF にするまで、モータ制御の動作を続ける。

２．補足資料

③回路図および入出力ピン割り付け表

リモコンおよびモータ制御装置の回路図（図 3、図 4）と、使用するマイコンのピン名称（ポート名とビット番号）の割り付けを表 1 に示す。

表 1　入出力ピン割り付け表

リモコン側マイコン	
機能	ピン名称
スイッチ	PB-1
ボリューム（A/D 変換）	P7-0
LED	PB-0
送信（シリアル通信）	P9-1

モータ制御装置側マイコン	
機能	ピン名称
受信（シリアル通信）	P9-3
DA0（モータドライバ）	PB-0
DA1（モータドライバ）	PB-1
DA2（モータドライバ）	PB-2
DA3（モータドライバ）	PB-3
InA（モータドライバ）	PB-4
InB（モータドライバ）	PB-5

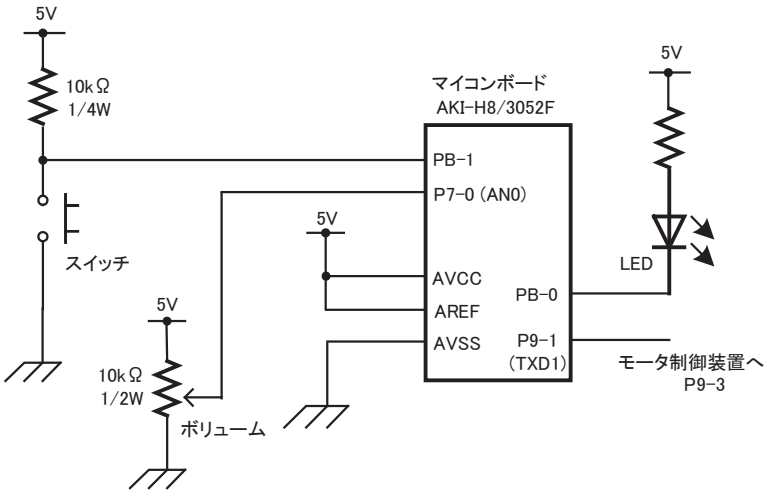


図 3　リモコン回路

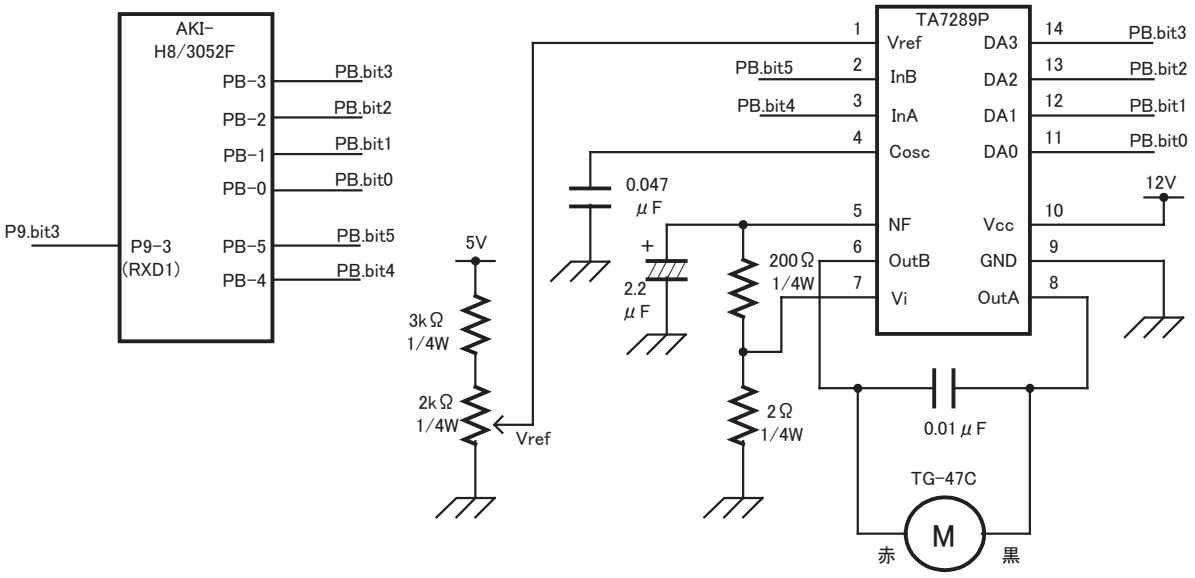


図 4 モータ制御装置回路図

④モータドライバ IC(TA7289P)の動作(図 4 について)

・電源電圧はV_{cc}を 12V、V_{dd}を 5Vにする。

・可変抵抗を調整して制御電源電圧V_{ref}を約 0.6Vに設定する。

・ファンクション入力端子は InA を 5V、InB を 0V として正回転(CW)に設定する。InA を 0V、InB を 5V として逆回転(CCW)、InA を 0V、InB を 0V として停止とする (表 2 参照)

・速度設定(D/A 入力端子 DA3～DA0)は、数値を 0,6・・・・,15 (最大) に変化させて、DC モータの速度を決定する。(表 3 参照)

表 2 ファンクション入力端子と回転方向の関係

回転状況	InB PB. B5	InA PB. B4	回転方向データ
正転	0	1	1 (正転)
逆転	1	0	2 (逆転)
停止	0	0	0 (停止)

表 2、表 3 の‘1’は 5V、‘0’は 0V を示す。
今回、速度設定は 0,6～15 を使用する。

表 3 D/A 入力端子と速度設定の関係

速度設定データ	DA3 PB. B3	DA2 PB. B2	DA1 PB. B1	DA0 PB. B0
0	0	0	0	0
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

⑤ボリューム回転とモータ動作の関係

ボリュームの回転のみで、モータの回転方向、速度、停止などを決定する。以下に、ボリュームの出力電圧から、モータドライバ IC に送る制御信号に変換するまでの流れを示す。また、マイコンのプログラムのフローチャート例を、図 5（リモコン側）、図 6（モータ制御装置側）にそれぞれ示す。

表 4 ボリュームの出力電圧（0V～約 2.5V）からモータドライバ IC への制御信号へのデータ変換

ボリュームの出力電圧	A/D 変換データ（0～127）※1	逆転側データ（127～0）※2	速度設定データ（0、6～15）※3	モータドライバ制御信号※4	モータ動作
0V ～約 2.5V	0 ～ 79	127 ～ 48	15 ～ 6	逆転信号	逆回転
	80 ～ 127	47 ～ 0	0	停止信号	停止

表 5 ボリュームの出力電圧（約 2.5V～5V）からモータドライバ IC への制御信号へのデータ変換

ボリュームの出力電圧	A/D 変換データ（128～255）※1	正転側データ（0～127）※2	速度設定データ（0、6～15）※3	モータドライバ制御信号※4	モータ動作
約 2.5V ～5V	128 ～ 175	0 ～ 47	0	停止信号	停止
	176 ～ 255	48 ～ 127	6 ～ 15	正転信号	正回転

※1　A/D 変換後の 10bit データのうち、上位 8bit を 0～255 の数値に変換したもの

※2　※1 の A/D 変換後のデータについて、

0～127 については　→　127 から引くことで、逆転側データ 127～0 に変換する

128～255 については　→　128 を引くことで、正転側データ 0～127 に変換する

※3　※2 の正転側データおよび逆転側データについて、

0～47 については　→　速度設定データ 0　とする

48～127 については　→　8 で割り、その商の値を　速度設定データ 6～15　とする

※4　モータドライバ制御信号は、モータの回転方向と速度、停止を決定する 6 ビットデータ（下記に示す）で、速度設定データに回転方向データを付加した信号である。

上位 2 ビット【回転方向データ（0～2）】＋下位 4 ビット【速度設定データ（0、6～15）】

↓

1：正転　2：逆転　0：停止

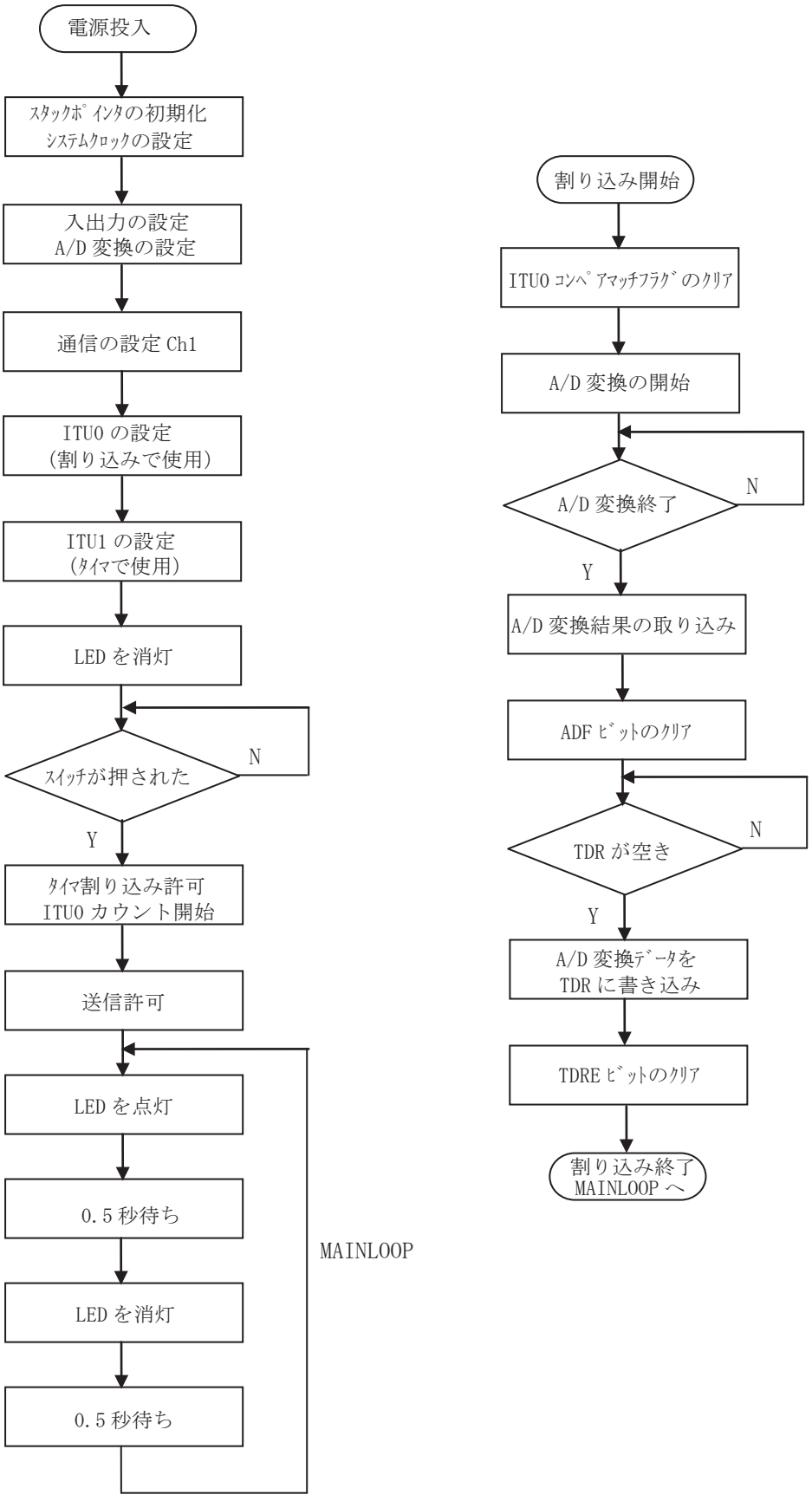


図 5　リモコン側プログラムのフローチャート例

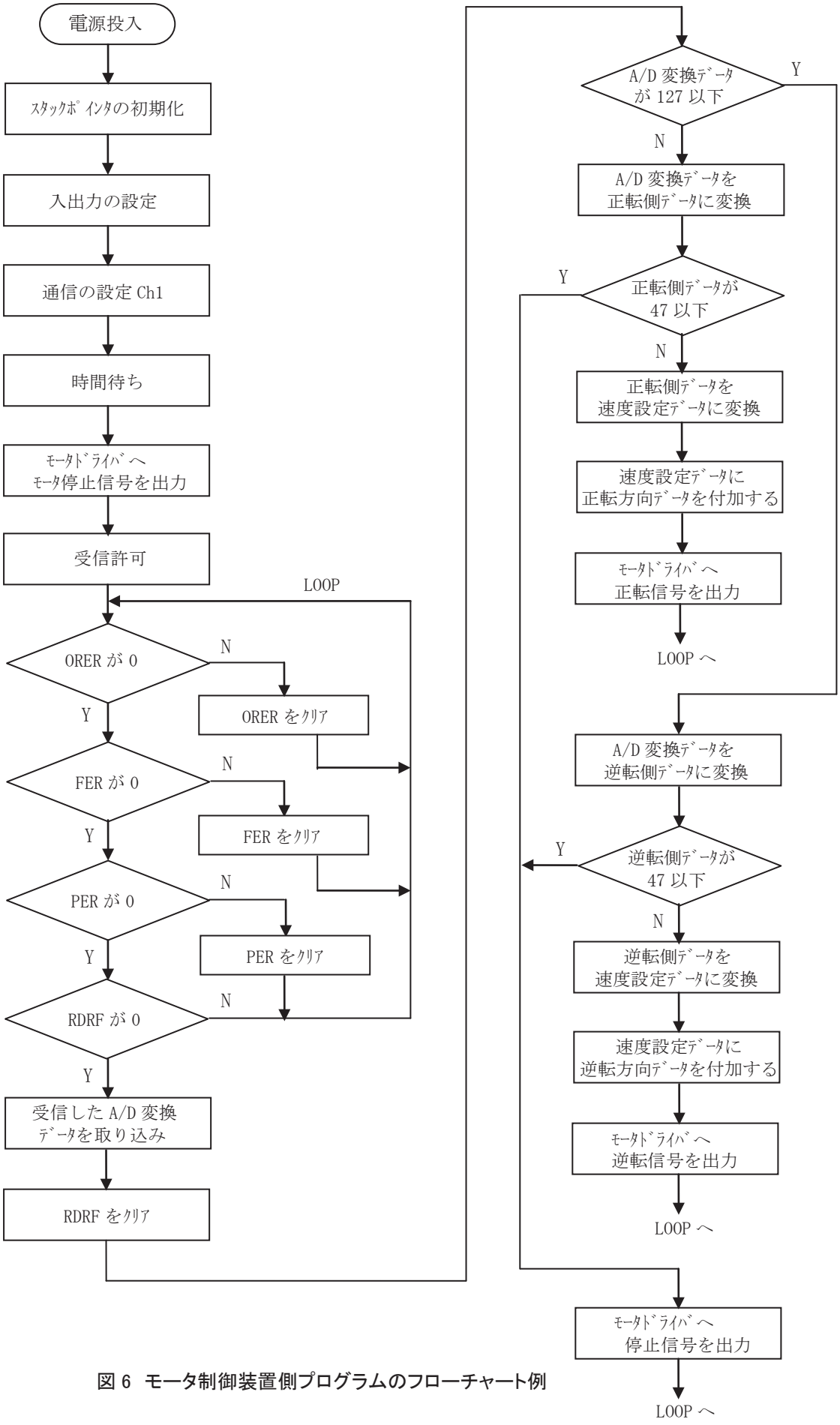


図 6　モータ制御装置側プログラムのフローチャート例

実技課題　解答及び解説

「マイコンによるアセンブリ言語を用いたモータ制御」

【プログラム例】

・リモコン側

```
;*******************************************************************************
;*                    訓練課題：リモコンプログラム　sousin.mar                 *
;*******************************************************************************

        .CPU    300HA
P7DR    .EQU    H'FFFFCE        ; ﾎﾟｰﾄ7ﾃﾞｰﾀﾚｼﾞｽﾀ
PBDDR   .EQU    H'FFFFD4        ; ﾎﾟｰﾄBﾃﾞｰﾀﾃﾞｨﾚｸｼｮﾝﾚｼﾞｽﾀ
PBDR    .EQU    H'FFFFD6        ; ﾎﾟｰﾄBﾃﾞｰﾀﾚｼﾞｽﾀ
DIVCR   .EQU    H'FFFF5D        ; 分周比ｺﾝﾄﾛｰﾙﾚｼﾞｽﾀ
TSTR    .EQU    H'FFFF60        ; ﾀｲﾏｽﾀｰﾄﾚｼﾞｽﾀ
STR0    .BEQU   0,TSTR          ; ﾀｲﾏｽﾀｰﾄﾚｼﾞｽﾀの STR0 ﾋﾞｯﾄ
TCR0    .EQU    H'FFFF64        ; ﾀｲﾏｺﾝﾄﾛｰﾙﾚｼﾞｽﾀ 0
TCNT0   .EQU    H'FFFF68        ; ﾀｲﾏｶｳﾝﾀ 0(W ﾃﾞｰﾀ)
TSR0    .EQU    H'FFFF67        ; ﾀｲﾏｽﾃｰﾀｽﾚｼﾞｽﾀ 0
IMFA0   .BEQU   0,TSR0          ; ﾀｲﾏｽﾃｰﾀｽﾚｼﾞｽﾀ 0 の IMFA ﾋﾞｯﾄ
GRA0    .EQU    H'FFFF6A        ; ｼﾞｪﾈﾗﾙﾚｼﾞｽﾀ A0(W ﾃﾞｰﾀ)
TIER0   .EQU    H'FFFF66        ; ﾀｲﾏｲﾝﾀﾗﾌﾟﾄｲﾈｰﾌﾞﾙﾚｼﾞｽﾀ 0
IMIEA0  .BEQU   0,TIER0         ; ﾀｲﾏｲﾝﾀﾗﾌﾟﾄｲﾈｰﾌﾞﾙﾚｼﾞｽﾀ 0 の IMIEA ﾋﾞｯﾄ
STR1    .BEQU   1,TSTR          ; ﾀｲﾏｽﾀｰﾄﾚｼﾞｽﾀの STR1 ﾋﾞｯﾄ
TCR1    .EQU    H'FFFF6E        ; ﾀｲﾏｺﾝﾄﾛｰﾙﾚｼﾞｽﾀ 1
TCNT1   .EQU    H'FFFF72        ; ﾀｲﾏｶｳﾝﾀ 1(W ﾃﾞｰﾀ)
TSR1    .EQU    H'FFFF71        ; ﾀｲﾏｽﾃｰﾀｽﾚｼﾞｽﾀ 1
IMFA1   .BEQU   0,TSR1          ; ﾀｲﾏｽﾃｰﾀｽﾚｼﾞｽﾀ 1 の IMFA ﾋﾞｯﾄ
GRA1    .EQU    H'FFFF74        ; ｼﾞｪﾈﾗﾙﾚｼﾞｽﾀ A1(W ﾃﾞｰﾀ)
SCR     .EQU    H'FFFFBA        ; ｼﾘｱﾙｺﾝﾄﾛｰﾙﾚｼﾞｽﾀ
SMR     .EQU    H'FFFFB8        ; ｼﾘｱﾙﾓｰﾄﾞﾚｼﾞｽﾀ
BRR     .EQU    H'FFFFB9        ; ﾋﾞｯﾄﾚｰﾄﾚｼﾞｽﾀ
TDR     .EQU    H'FFFFBB        ; ﾄﾗﾝｽﾐｯﾄﾃﾞｰﾀﾚｼﾞｽﾀ
SSR     .EQU    H'FFFFBC        ; ｼﾘｱﾙｽﾃｰﾀｽﾚｼﾞｽﾀ
TDRE    .BEQU   7,SSR           ; ｼﾘｱﾙｽﾃｰﾀｽﾚｼﾞｽﾀの TDRE ﾋﾞｯﾄ
TE      .BEQU   5,SCR           ; ｼﾘｱﾙｺﾝﾄﾛｰﾙﾚｼﾞｽﾀの TE ﾋﾞｯﾄ
ADCSR   .EQU    H'FFFFE8        ; A/D ｺﾝﾄﾛｰﾙ/ｽﾃｰﾀｽﾚｼﾞｽﾀ
ADDRAH  .EQU    H'FFFFE0        ; A/D ﾃﾞｰﾀﾚｼﾞｽﾀ AH
ADST    .BEQU   5,ADCSR         ; A/D ｺﾝﾄﾛｰﾙ/ｽﾃｰﾀｽﾚｼﾞｽﾀの ADST ﾋﾞｯﾄ
ADF     .BEQU   7,ADCSR         ; A/D ｺﾝﾄﾛｰﾙ/ｽﾃｰﾀｽﾚｼﾞｽﾀの ADF ﾋﾞｯﾄ
```

```
;************************        vector        ***************************
        .SECTION        C,DATA,LOCATE=0
        .DATA.L         MAIN
        .ORG            H'60
        .DATA.L         TIM_IN

;************************      main program     ***************************
        .SECTION        P,CODE,LOCATE=H'200
MAIN:
        mov.l   #H'ffff00, er7          ; ｽﾀｯｸﾎﾟｲﾝﾀの初期化

        ;********************    ｼｽﾃﾑｸﾛｯｸの設定     ************************
        mov.b   #H'ff, r0l
        mov.b   r0l, @DIVCR             ; ｼｽﾃﾑｸﾛｯｸ φ を 3.125MHz に設定

        ;********************     入出力の設定     ************************
        mov.b   #H'05, r0l
        mov.b   r0l, @PBDDR             ; PB-0 を出力、PB-1 を入力

        ;********************     AD 変換の設定     ************************
        mov.b   #H'08, r0l
        mov.b   r0l, @ADCSR             ; 単一モード  AN0 使用

        ;********************      通信の設定 ch1     ************************
        ;*** 調歩同期式ﾓｰﾄﾞ、8ﾋﾞｯﾄﾃﾞｰﾀ、ﾊﾟﾘﾃｨ付加、ﾁｪｯｸ禁止、1STOP ﾋﾞｯﾄ ***
        mov.b   #0, r0l
        mov.b   r0l, @SCR               ; ｼﾘｱﾙｺﾝﾄﾛｰﾙﾚｼﾞｽﾀを 0 に設定
        mov.b   r0l, @SMR               ; 通信ﾌｫｰﾏｯﾄの設定
        mov.b   #40, r0l
        mov.b   r0l, @BRR               ; BRR を 40 に設定(2400bps)

        ;**************  ｲﾝﾀｰﾊﾞﾙﾀｲﾏ(割り込み)ITU0 の設定  ****************
        mov.b   #H'a3, r0l
        mov.b   r0l, @TCR0              ; 390625Hz でｶｳﾝﾄ
        mov.w   #39063, r0
        mov.w   r0, @GRA0               ; ｺﾝﾍﾟｱ値 39063
        ldc     #0, CCR

        ;**********************  ﾀｲﾏ ITU1 の設定  ************************
        mov.b   #H'a0, r0l
```

```
        mov.b   r0l, @TCR1              ; 3.125MHz でｶｳﾝﾄ
        mov.w   #3125, r0
        mov.w   r0, @GRA1               ; ｺﾝﾍﾟｱ値 3125

        ;**********************    MAIN LOOP    ***************************
        bset    #0, @PBDR               ; LED を消灯
LOOP1:  btst    #1, @PBDR
        bne     LOOP1                   ; ｽｲｯﾁを押していない間、LOOP1 へ
        bset    IMIEA0                  ; ﾀｲﾏ割り込み許可
        bset    STR0                    ; ｶｳﾝﾄ開始
        bset    TE                      ; 送信許可

        ;**********************    LED は点灯    ***************************
LOOP2:  bclr    #0, @PBDR               ; LED を点灯

        ;*********************    0.5 秒待ち LOOP    ************************
        mov.w   #500, r1                ; 繰り返し回数 500 回
        bset    STR1                    ; ｶｳﾝﾄ開始
MS_1:   btst    IMFA1
        beq     MS_1                    ; 1ms 経つまで MS_1 へ
        bclr    IMFA1                   ; IMFA のｸﾘｱ
        sub.w   #1, r1
        bne     MS_1                    ; r1 が 0 になるまで MS_1 へ
        bclr    STR1                    ; ｶｳﾝﾄ終了
        mov.w   #0, r1
        mov.w   r1, @TCNT1              ; ｶｳﾝﾀｸﾘｱ

        ;**********************    LED は消灯    ***************************
        bset    #0, @PBDR               ; LED を消灯

        ;*********************    0.5 秒待ち LOOP    ************************
        mov.w   #500, r1                ; 繰り返し回数 500 回
        bset    STR1                    ; ｶｳﾝﾄ開始
MS_2:   btst    IMFA1
        beq     MS_2                    ; 1ms 経つまで MS_2 へ
        bclr    IMFA1                   ; IMFA のｸﾘｱ
        sub.w   #1, r1
        bne     MS_2                    ; r1 が 0 になるまで MS_2 へ
        bclr    STR1                    ; ｶｳﾝﾄ終了
        mov.w   #0, r1
```

```
        mov.w   r1, @TCNT1              ; ｶｳﾝﾀｸﾘｱ
        bra     LOOP2                   ; LOOP2 へ

;************************  timer interrupt  ****************************
TIM_IN:         ; ｲﾝﾀｰﾊﾞﾙﾀｲﾏで、割り込み周期 0.1sec ごとに実行される命令文

        bclr    IMFA0                   ; ｺﾝﾍﾟｱﾏｯﾁﾌﾗｸﾞのｸﾘｱ

        ;*********************       A/D 変換        ************************
        bset    ADST                    ; AD 変換の開始
AD:     btst    ADF
        beq     AD                      ; AD 変換が終了するまで、AD へ
        mov.b   @ADDRAH, r0l            ; AD 変換結果を r0l ﾚｼﾞｽﾀにｺﾋﾟｰ
        bclr    ADF                     ; ADF ﾋﾞｯﾄをｸﾘｱ

        ;*********************     ﾃﾞｰﾀの送信     ************************
TX:     btst    TDRE
        beq     TX                      ; TDR が空でないとき TX へ
        mov.b   r0l, @TDR               ; AD 変換結果を TDR へ、送信実行
        bclr    TDRE                    ; TDR に有効な送信ﾃﾞｰﾀがある
        rte                             ; 割り込み終了 MAIN LOOP へ戻る
        .end
```

＜参考＞

・モータ駆動側

```
;*************************************************************************
;*                    訓練課題：モータ駆動プログラム　jusin.mar              *
;*************************************************************************

        .CPU    300HA
PBDDR   .EQU    H'FFFFD4        ; ﾎﾟｰﾄBﾃﾞｰﾀﾃﾞｨﾚｸｼｮﾝﾚｼﾞｽﾀ
PBDR    .EQU    H'FFFFD6        ; ﾎﾟｰﾄBﾃﾞｰﾀﾚｼﾞｽﾀ

SCR     .EQU    H'FFFFBA        ; ｼﾘｱﾙｺﾝﾄﾛｰﾙﾚｼﾞｽﾀ
SMR     .EQU    H'FFFFB8        ; ｼﾘｱﾙﾓｰﾄﾞﾚｼﾞｽﾀ
BRR     .EQU    H'FFFFB9        ; ﾋﾞｯﾄﾚｰﾄﾚｼﾞｽﾀ
RDR     .EQU    H'FFFFBD        ; ﾚｼｰﾌﾞﾃﾞｰﾀﾚｼﾞｽﾀ
SSR     .EQU    H'FFFFBC        ; ｼﾘｱﾙｽﾃｰﾀｽﾚｼﾞｽﾀ
ORER    .BEQU   5,SSR           ; ｼﾘｱﾙｽﾃｰﾀｽﾚｼﾞｽﾀの ORER ﾋﾞｯﾄ
FER     .BEQU   4,SSR           ; ｼﾘｱﾙｽﾃｰﾀｽﾚｼﾞｽﾀの FER ﾋﾞｯﾄ
PER     .BEQU   3,SSR           ; ｼﾘｱﾙｽﾃｰﾀｽﾚｼﾞｽﾀの PER ﾋﾞｯﾄ
RDRF    .BEQU   6,SSR           ; ｼﾘｱﾙｽﾃｰﾀｽﾚｼﾞｽﾀの RDRF ﾋﾞｯﾄ
RE      .BEQU   4,SCR           ; ｼﾘｱﾙｺﾝﾄﾛｰﾙﾚｼﾞｽﾀの RE ﾋﾞｯﾄ

;************************        vector        ****************************
        .SECTION        C,DATA,LOCATE=0
        .DATA.L         MAIN

;************************      main program     ****************************
        .SECTION        P, CODE, LOCATE=H'200
MAIN:
        mov.l   #H'ffff00, er7          ; ｽﾀｯｸﾎﾟｲﾝﾀの初期化

        ;**********************     入出力の設定     ************************
        mov.b   #H'3F, r0l
        mov.b   r0l, @PBDDR             ; PB-0～PB-5 を出力

        ;********************      通信の設定 ch1     ************************
        ;*** 調歩同期式ﾓｰﾄﾞ、8ﾋﾞｯﾄﾃﾞｰﾀ、ﾊﾟﾘﾃｨ付加、ﾁｪｯｸ禁止、1STOP ﾋﾞｯﾄ ***
        mov.b   #0, r0l
        mov.b   r0l, @SCR               ; ｼﾘｱﾙｺﾝﾄﾛｰﾙﾚｼﾞｽﾀを 0 に設定
        mov.b   #H'1, r0l
        mov.b   r0l, @SMR               ; 通信ﾌｫｰﾏｯﾄの設定(φ/4)
```

```
        mov.b   #80, r0l
        mov.b   r0l, @BRR               ; BRR を 80 に設定(2400bps)

        ;************************    時間待ち    ****************************
        mov.w   #500, r1                ; 繰り返し回数 500 回
WAIT:   nop
        nop
        sub.w   #1, r1
        bne     WAIT                    ; r1 が 0 になるまで WAIT へ

        ;**********************    MAIN LOOP    ****************************
        mov.b   #0, r0l
        mov.b   r0l, @PBDR              ; INA と INB は 0 でﾓｰﾀ停止
        bset    RE                      ; 受信許可

LOOP:   btst    ORER
        beq     FER_F                   ; ORER が 0 なら FER_F へ
        bclr    ORER
        bra     LOOP
FER_F:  btst    FER
        beq     PER_F                   ; FER が 0 なら PER_F へ
        bclr    FER
        bra     LOOP
PER_F:  btst    PER
        beq     RDR_F                   ; PER が 0 なら RDR_F へ
        bclr    PER
        bra     LOOP
RDR_F:  btst    RDRF
        beq     LOOP                    ; RDRF が 0 なら LOOP へ
        mov.b   @RDR, r0l               ; 受信ﾃﾞｰﾀを r0l ﾚｼﾞｽﾀにｺﾋﾟｰ
        bclr    RDRF

        cmp.b   #127, r0l
        bcs     REV                     ; 受信ﾃﾞｰﾀが 127 以下のとき REV へ

        ;************    受信ﾃﾞｰﾀが 128 以上のとき(ﾓｰﾀ正転)    *************
        mov.b   #128, r0h
        sub.b   r0h, r0l                ; 受信ﾃﾞｰﾀを 0～127 のﾃﾞｰﾀに変換
        cmp.b   #48, r0l
        bls     STOP                    ; 47 以下のとき STOP へ
```

```
        mov.b   #8, r1l
        mov.b   #0, r0h
        divxu.b r1l, r0                 ; ﾃﾞｰﾀを 8 で割り、6～15 の速度設定に変換
        mov.b   #H'10, r0h
        add.b   r0h, r0l                ; 正転方向ﾃﾞｰﾀ（INA は 1、INB は 0）を付加
        mov.b   r0l, @PBDR              ; 正転方向ﾃﾞｰﾀ＋速度設定ﾃﾞｰﾀ を出力
        bra     LOOP                    ; LOOP へ

        ;*************     受信ﾃﾞｰﾀが 127 以下のとき(ﾓｰﾀ逆転)     ************
REV:    mov.b   #127, r2l
        sub.b   r0l, r2l                ; 受信ﾃﾞｰﾀを 127～0 のﾃﾞｰﾀに変換
        cmp.b   #48, r2l
        bls     STOP                    ; 47 以下のとき STOP へ

        mov.b   #8, r1l
        mov.b   #0, r2h
        divxu.b r1l, r2                 ; ﾃﾞｰﾀを 8 で割り、6～15 の速度設定に変換
        mov.b   #H'20, r2h
        add.b   r2h, r2l                ; 逆転方向ﾃﾞｰﾀ（INA は 0、INB は 1）を付加
        mov.b   r2l, @PBDR              ; 逆転方向ﾃﾞｰﾀ＋速度設定ﾃﾞｰﾀ を出力
        bra     LOOP                    ; LOOP1 へ

STOP:   mov.b   #0, r0l
        mov.b   r0l, @PBDR              ; INA と INB は 0、速度設定も 0 でﾓｰﾀ停止
        bra     LOOP                    ; LOOP1 へ
        .end
```

作業工程計画書

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備（前日までに行う）	・リモコン回路とモータ制御装置の動作確認をする。（回答例のプログラムで、回路が正常に動作することを確認する。） ・パソコンおよびエディタ、ライタソフトが正常に動作するかを確認する。 ・テスタ、オシロスコープを用意して、プログラミングした後の動作確認方法を確認しておく。	
1．課題概要説明	リモコン回路とモータ制御装置の信号の流れ、マイコンの入出力ピン番号、つまみの回転とモータ制御の関係、マイコンの使用する機能の提示。作成したプログラムを印刷して提出することを提示。	
2．パソコン立ち上げ、エディタの起動	エディタ、ライタソフトを使用して、プログラムデータをマイコンに書き込むまでの手順を確認。	
3．プログラミング（初期設定および LED の点灯と消灯）	フローチャートに従って、プログラミングを記述する ・スタックポインタを設定する。 ・割り込み（0.1 秒間隔）でインターバルタイマ（ITU0）を利用するため、設定を考える。その結果、システムクロックの設定を変更する必要があれば行う。 ・シリアル通信の設定を行う。 ・入出力とA/D 変換の設定を行う。 ・LED の点滅で利用するタイマ（ITU1）の設定を行う。 ・スイッチが押される前は LED は消灯するようにする。 ・スイッチが押されたら、LED が点灯（0.5 秒）、消灯（0.5 秒）の繰り返しを行うようにする。 ・ここで一旦、リモコンのマイコンに書き込みし、LED が正常に動作するかを確認する。	
4．プログラミング（割り込み）	フローチャートに従って、割り込み部分のプログラミングを記述する ・タイマ割り込みの許可および ITU0 のカウント開始、送信許可を追加する。 ・ITU0コンペアマッチフラグのクリアをするようにする。 ・A/D 変換の開始、終了を待って A/D 変換したボリュームの出力電圧のデータを取り込むようにする。ADF ビットのクリアも追加する。 ・TDR の空きを確認して、A/D 変換したデータを TDR に書き込み、データを転送するようにする。TDRE ビットのクリアも追加する。 ・ここで一旦、リモコンのマイコンに書き込みし、0.1 秒間隔で割り込みが発生し、データが送信されているか、オシロスコープを利用して動作確認する。 ・正常にモータの制御が行われているか、モータ制御装置と接続して、動作確認する。	

訓 練 課 題 確 認 シ ー ト

訓練科名　：　制御技術科

仕上がり像　：　組込みマイコン制御システムの製作ができる。

システム名　：　組込み型マイクロコンピュータ制御技術　　入所期　：

訓練課題名　：　マイコンによるアセンブリ言語を用いたモータ制御　　氏　名　：

評価区分	評価項目	細　目	評価（数値）					評価判定	評価基準
作業時間	作業準備時間	パソコン、エディタの立ち上げ	1				5		突然の故障を除いて、開始から10分以内にパソコンを立ち上げ、エディタの起動の作業ができる場合は5点、それ以外は1点とする。
	作業時間	プログラミング	1	2	3	4	5		指定した作業時間以内に、全作業の動作確認まで終われば5点、10分遅れるごとに1点減点する。
作業工程	作業工程における留意事項等	作業工程手順	2	4	6	8	10		プログラムをマイコンに書き込みし、（測定器を使う場合も含めて）動作確認する作業手順において、全く不適切な箇所がなければ10点、不適切な箇所があれば、1箇所につき2点減点し、最低点を2点とする。
プログラミング	プログラミング	スタックポインタ、使用レジスタのアドレス設定	2	4	6	8	10		数値に間違いがなければ10点、不適切な箇所があれば、1箇所につき2点減点し、最低点を2点とする。
		スイッチの動作	2	4	6	8	10		動作に間違いがなければ10点、不適切な箇所があり、正常に動作しない場合は2点とする。
		A/D変換の実行	2	4	6	8	10		
		割り込みの実行	2	4	6	8	10		
		LEDの点灯、点滅	2	4	6	8	10		
		通信（送信）の実行	2	4	6	8	10		
安全作業	安全作業	他の作業者への妨げ行為	1	2	3	4	5		持ち点を5点とし、他の作業者への不適切な作業又は行為があるごとに1点ずつ減点し、最低点を1点とする。
		VDT作業	1				5		不適切な姿勢で作業をしている場合や、指示通りに休憩を取らない場合は注意する。注意に従わない場合は1点、注意する必要がない場合や、注意に従う場合は5点
工夫・改善	工夫・改善	装置の動作や機能の工夫・改善	2	4	6	8	10		装置の動作や機能に、何かしらの工夫・改善がされていなければ0点とし、工夫・改善点1件につき2点ずつ加算し、最高点を10点とする。
	工夫・改善点記入欄		総点				100		＜判定表＞ A　：　80点以上　：到達水準を十分に上回った B　：　60点以上80点未満　：到達水準に達した C　：　60点未満　：到達水準に達しなかった
			合計点						
			換算点						
			総合評価判定						

訓練課題のねらい	コメント
マイコンでモータを制御する課題を通して、アセンブリ言語によるプログラミングを習得しているかを実技により確認します。	担当指導員氏名：

評価要領

訓練科名　：　制御技術科
仕上がり像　：　組込みマイコン制御システムの製作ができる。
システム名　：　組込み型マイクロコンピュータ制御技術
訓練課題名　：　マイコンによるアセンブリ言語を用いたモータ制御

評価区分	評価項目	細　目	評価要領（採点要領）	備考
作業時間	作業準備時間	パソコン、エディタの立ち上げ	時間を計測しながら、訓練生の作業をよく観察すること。	
作業時間	作業時間	プログラミング	時間を計測しながら、訓練生の作業をよく観察すること。	
作業工程	作業工程における留意事項等	作業工程手順	訓練生の作業をよく観察し、必要があれば助言をしてもよい。	
プログラミング	プログラミング	スタックポインタ、使用レジスタのアドレス設定	プログラムの数値を見て確認すること。	
		スイッチの動作	スイッチを押すまでは、（ボリュームを回すなどしても）モータの回転は行われず、LEDの点灯、点滅もは行われないこと。	
		A/D変換の実行	通信プログラムが完成している状態で、リモコンとモータ制御装置を接続して、リモコンを操作し、モータの動作を見ることで、正常にA/D変換が実行されることを確認すること。リモコンのマイコンボードに8つのLEDを接続し、A/D変換結果を表示するプログラムを追加しても確認できる。	
		割り込みの実行	オシロスコープを利用し、通信の実行と合わせて、0.1秒間隔でシリアル信号が出力されることを確認すること。	オシロスコープ
		LEDの点灯、点滅	LEDの点灯、点滅状態を見て確認すること。	
		通信（送信）の実行	オシロスコープを利用し、割り込みの実行と合わせて、0.1秒間隔でシリアル信号が出力されることを確認すること。リモコンとモータ制御装置を接続して、正常なデータが送られていることを確認すること。	オシロスコープ
安全作業	安全作業	他の作業者への妨げ行為	訓練生の作業態度をよく観察し、必要があれば注意をすること。	
		VDT作業	訓練生の作業姿勢をよく観察し、必要があれば注意をすること。	
工夫・改善	工夫・改善	装置の動作や機能の工夫・改善	何かしらの、装置のユーザーにとって有用な機能が追加されていれば加点する。	

実技課題

管理番号：Ｅ－44

「Ｃ言語を用いたマイコンによる計測制御」

■課題概要■

C 言語のプログラミング、マイコンにおける割込み、マイコンの周辺機能であるタイマや A/D 変換等の機能を、計測制御システムの構築を想定した実技試験により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領	○	E-44-00_実施要領.doc
訓練課題	○	E-44-01_訓練課題.doc
解答	○	E-44-02_解答及び解説.doc
作業工程手順書	○	E-44-03_作業工程計画書【作成例】.doc
訓練課題確認シート	○	E-44-04_訓練課題確認シート及び評価要領.xls
評価要領	○	E-44-04_訓練課題確認シート及び評価要領.xls

※フォルダ「プログラム」内に、マイコンのソースファイルのデータがあります。

実技課題　実施要領

訓練課題名　「C言語を用いたマイコンによる計測制御」

- 作業準備（ハードウェア製作含む）及び使用機器の準備時間は別途確保した上で事前に行うこととし、作業時間に含まない。
- 作業時間は、休憩時間を除いた時間とする。
- 実施形態は、1 名で行なうことが望ましい。
- 作業工程計画書は、ポイント（留意事項）のみ記述させる。
- この課題は、表示機能（LCD や 7 セグメント LED)、温度センサ等のアナログ信号を扱う A/D 変換機能、及び割込みで使用可能なタイマ機能等を有するマイコン（サンプルは PIC16F88 相当)を使用することを想定している。各施設で使用するハードウェア(マイコンボード、周辺機器、負荷）が異なる場合、それらに合わせて課題を修正した上で実施することが望ましい。
- VDT 作業を考慮し、1 時間を目安に 10 分程度の休憩時間を設ける。
- 回路動作を確認する。動作の確認は、課題が完成した受講生に挙手させ、その場で動作を確認して記録する。
- 試験終了時にソースファイルを電子データで提出させる。

実技課題

「Ｃ言語を用いたマイコンによる計測制御」

１　作業時間

240 分（休憩を除く）

２　配付資料

問題用紙

３　課題作成、提出方法

・各個人で作業すること

・プログラムをマイコンに実装し、ソースファイルを提出すること

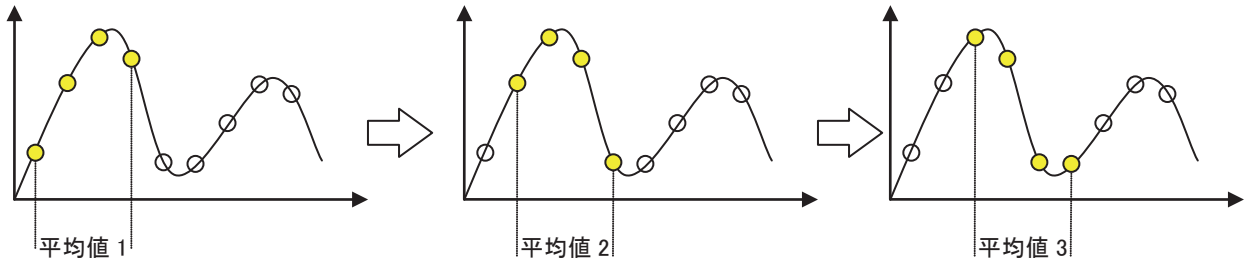
・課題が終了した時点で、指導員の動作確認を受けること。

１．課題名：C 言語を用いたマイコンによる計測制御

２．課題内容

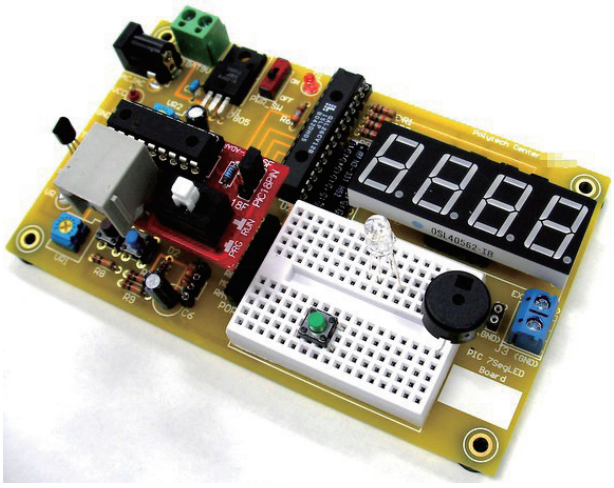
以下の機器の仕様において、下記の条件を満たす計測システムを、C 言語を用いてマイコンに実装しなさい。

- リセット（電源投入）後、押しボタンスイッチ 1 を ON にすると計測を開始する。
- 計測中はセンサ入力より取得した電圧を計測値に換算し 7 セグメント LED へ表示する。
- センサの入力は一定時間毎の移動平均（単純移動平均）で算出すること。ただしセンサの種類等により移動平均による算出が困難である場合は省略して良いものとする。
- 計測中に計測値が一定値に達した場合は、ブザー（LED 等も可による出力を行なう。ブザー出力後押しボタンスイッチ 2 を押すとブザーの出力、及び計測を停止する。

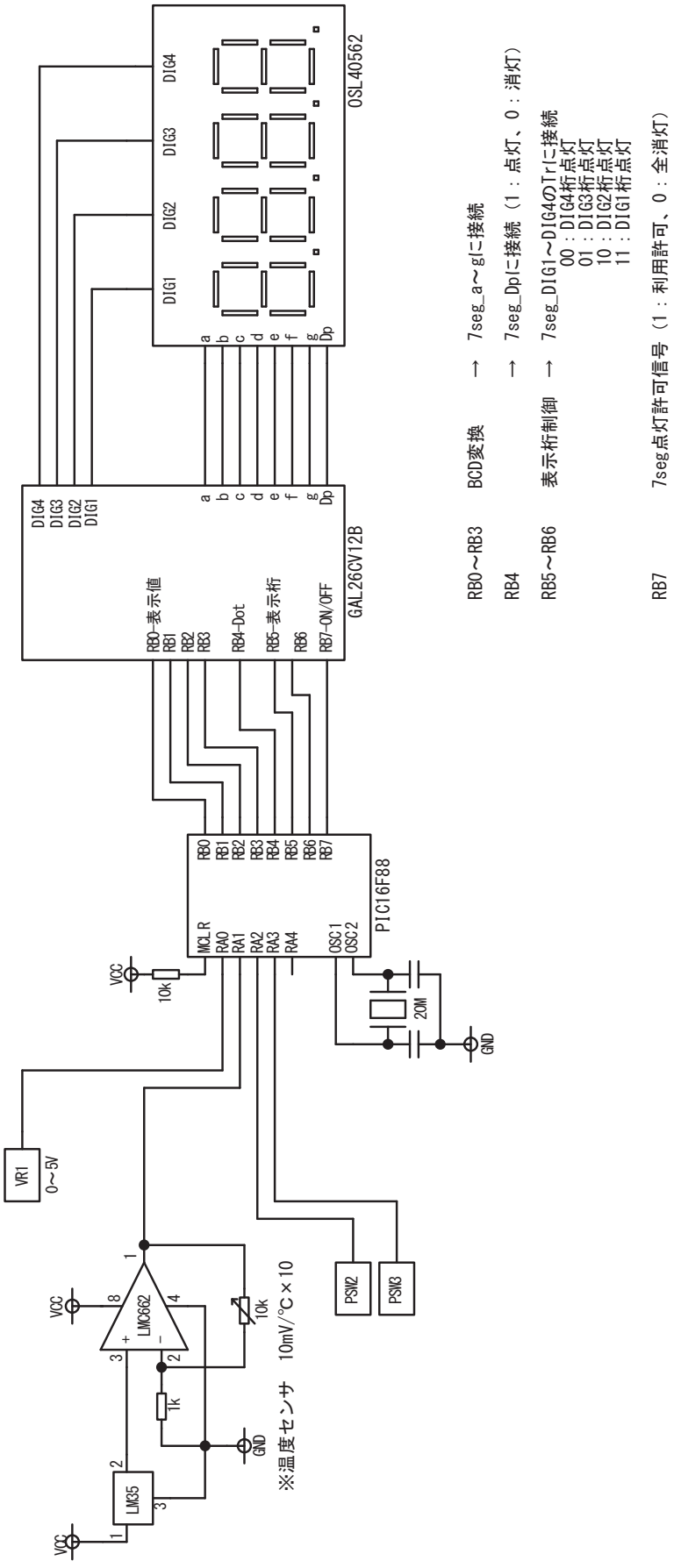


仕様

入出力機器	機能	設定例(PIC)
押しボタンスイッチ 1	計測開始	RA2（負論理）
押しボタンスイッチ 2	ブザー出力停止	RA3（負論理）
センサ入力	計測信号入力	RA1(A/D 変換) 温度センサ、100mV/℃
表示器	計測値出力	PORTB(RB0～RB7) 7 セグメント LED4 桁
ブザー	出力	RA4（矩形波による駆動可）



【参考】実習ボードブロック図



３．課題仕様

（１）課題に必要な機器を用意する。

機器	条件	参考例
マイコンボード	前述の 2 の仕様を満たすもの	PIC16F88
直流安定化電源	DC7V、500mA 以上	AC アダプタ 9V、1A
パソコン	開発環境が動作するもの	Windows XP/Vista/7
開発環境	C 言語をサポートする	MPLAB X / XC8 コンパイラ
マイコン用ツール	ライタ、デバッガ等	PICkit3

（２）動作仕様

① リセット直後は待機状態とする。

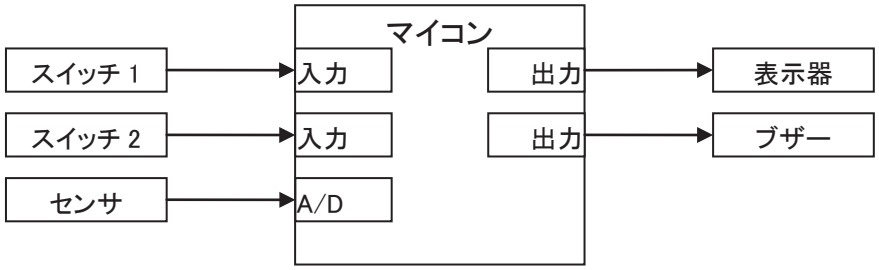
② 待機状態中に押しボタンスイッチ 1 を押すと計測を開始する。

③ 計測中は一定時間毎（タイマ割込み）にセンサより取得した信号の移動平均値（平均数は 4, 8, 16…のような 2^n）を表示器に出力する。

④ 計測中に平均値が既定値を超えた場合、ブザーを出力する。

⑤ ブザー出力後、押しボタンスイッチ 2 を押すと待機状態に移行する。

※訓練習得内容の評価のため A/D 変換、割込み等の機能を含めたプログラムで作成すること。



（３）作業内容

1) 動作仕様を満たすプログラムを main.c に作成する。

ファイル名	機能	備考
main.c	main 関数、割込み関数、各機能の初期化等	このファイルのみ編集
common.h	ソースファイル間の共通設定	
display.c	表示器の関数の実装	
display.h	表示器の関数の定義	
ad.c	A/D 変換の関数の実装	
ad.h	A/D 変換の関数の定義	

2) プログラムの動作確認を行ない、動作仕様の①～⑤のとおり動作するかを確認する。

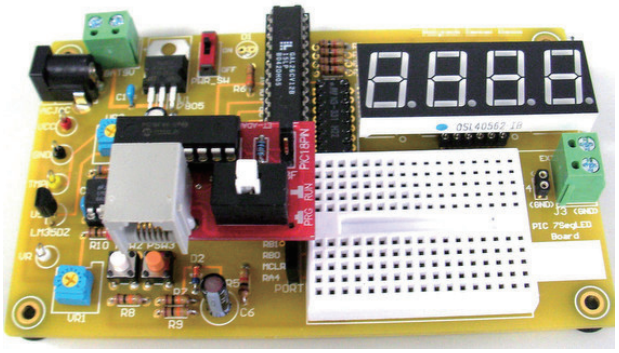
実技課題／　解答及び解説

「Ｃ言語を用いたマイコンによる計測制御」

１．機器の仕様

仕様

入出力機器	機能	設定例(PIC)
押しボタンスイッチ 1	計測開始	RA2（負論理）
押しボタンスイッチ 2	ブザー出力停止	RA3（負論理）
センサ入力	計測信号入力	RA1(A/D 変換) 温度センサ、100mV/℃
表示器	計測値出力	PORTB(RB0～RB7) 7 セグメント LED4 桁
ブザー	出力	RA4（矩形波による駆動可）



２．ソースファイル

課題の解答は各処理機能を複数のファイルに分割している。基本は main 関数を実装している main.c のファイルのみを編集して完成するものとする。

ファイル名	機能	備考
main.c	main 関数、割込み関数、各機能の初期化等	このファイルのみ編集
common.h	ソースファイル間の共通設定	
display.c	表示器の関数の実装	
display.h	表示器の関数の定義	
ad.c	A/D 変換の関数の実装	
ad.h	A/D 変換の関数の定義	

```
/************************************************************
    ファイル名:main.c
    概要:温度センサの値取得と表示
    ターゲット:PIC16F88(20MHz)
    ポートA:アナログ入力(RA0, RA1)、入力(RA2, RA3)
    ポートB:LED(RB0-RB7)
    作成日:2012/07/20
************************************************************/

//各種ファイルのインクルード

#include "common.h"
#include "ad.h"
#include "display.h"

//コンフィグレーションビットの設定
__CONFIG(FOSC_HS & WDTE_OFF & PWRTE_OFF & MCLRE_ON & BOREN_OFF &
    LVP_OFF & CPD_OFF & WRT_OFF & CCPMX_RB3 & CP_OFF & DEBUG_OFF);
__CONFIG(FCMEN_OFF & IESO_OFF);

#define BEEP_CYCLE  0xB5    //ブザー音の半周期(7オクターブのドの音階)
#define DISP_CYCLE  0xCF2C
#define DISP_LOW    (unsigned char)(DISP_CYCLE & 0xFF)
#define DISP_HIGH   (unsigned char)(DISP_CYCLE >> 8)

//スイッチの置換(負論理で接続)
#define PSW1    RA2     //押しボタンスイッチ1はRA2に接続
#define PSW2    RA3     //押しボタンスイッチ1はRA3に接続
#define SW_ON   0       //押しボタンスイッチON
#define SW_OFF  1       //押しボタンスイッチOFF

//プロトタイプ宣言
void interrupt isr(void);
void io_init(void);
void tmr0_init(void);
void tmr1_init(void);
unsigned short move_average(unsigned short add);
```

```
//グローバル変数・定数
unsigned char seg[4];

#define AVERAGE_COUNT   3   //移動平均の数 3 = 2 の 3 乗 = 8 個の平均を取る



//main 関数
void main(void)
{
    unsigned short temp;    //温度の値(25.6℃ => 2560)
    unsigned short ave;     //移動平均値
    unsigned char i;

    io_init();          //I/O の初期化
    tmr0_init();        //タイマ 0 の初期化
    tmr1_init();        //タイマ 1 の初期化
    ad_init();          //A/D の初期化

    //割込みの設定
    TMR0IE = 0;         //TMR0 割込み禁止(後の処理で許可)
    PEIE = 1;           //周辺機能割込み許可
    GIE = 1;            //全割込み許可

    while (1) {             //無限ループ
        dec2temp(seg, 0);
        i = 50;
        //スイッチ 1 が OFF のとき以下のループ
        while (PSW2 == SW_OFF) {    //入力待ち専用表示(0000 を点滅)
            while (i--) seg_disp4(seg);
            SEG_OFF();
            __delay_ms(400);
            continue;
        }

        TMR1IF = 0;             //フラグ初期化
        TMR1IE = 1;             //タイマ 0 割込み許可

        while (1) {
            //温度センサを A/D 変換して 100mV/℃として代入
```

```
            temp = (unsigned short long)ad_get(1) * 5000 >> 10;
            ave = move_average(temp);   //移動平均値の算出
            dec2temp(seg, ave);         //"25.6c"のように表示
            //温度が既定値(30℃)以上のとき
            if (ave >= (ad_get(0) << 2)) break;
            __delay_ms(20);     //A/D 変換のサンプリング周期(適当)
        }
        //ブザー出力
        TMR0IF = 0;             //フラグ初期化
        TMR0IE = 1;             //タイマ 0 割込み許可 T
        //スイッチ 2 が ON になるまでループ
        while(PSW2 == SW_OFF)   ;
        TMR0IE = 0;             //タイマ 0 割込み禁止
        TMR1IF = 0;             //フラグ初期化
        TMR1IE = 0;             //タイマ 0 割込み許可
    }
}

//I/O 初期化関数
void io_init(void)
{
    ANSEL = 0x03;               //RA0 と RA1 をアナログ入力、他をディジタル I/O
    TRISA = 0x0F;               //RA0-RA3 を入力、他を出力
    TRISB = 0x00;               //PORTB を全出力
    PORTB = 0x00;               //7 セグは全て OFF(消灯)
}

//タイマ 0 初期化関数
void tmr0_init(void)
{
    T0CS = 0;               //システムクロックをカウント
    PSA = 0;                //プリスケーラを使用する
    PS2 = 0;                //プリスケーラ 1/16
    PS1 = 1;
    PS0 = 1;
    TMR0 = BEEP_CYCLE;      //タイマの初期値
}

//タイマ 1 初期化関数
```

```
void tmr1_init(void)
{
    T1CKPS0 = 0;            //プリスケーラ 1:1
    T1CKPS1 = 0;
    T1OSCEN = 0;            //外部オシレータパワーオフ
    TMR1CS = 0;             //Fosc/4(0.2us)を使用
    TMR1ON = 1;             //タイマ有効
    TMR1L = DISP_LOW;           //65536 - 12500 = 53036 = 0xCF2C
    TMR1H = DISP_HIGH;          //0.2us x 12500 = 2.5ms
}

//割込み関数
void interrupt isr(void)
{
    static unsigned char i = 0;         //桁の切替用静的変数

    //タイマ 0 割込み(ブザー音)
    if (TMR0IF & TMR0IE) {                  //TMR0 フラグ検出
        RA4 = !RA4;                 //ブザーの信号を反転
        TMR0 = BEEP_CYCLE;
        TMR0IF = 0;                 //TMR0 フラグ初期化
    }

    //タイマ 1 割込み(2ms 毎に発生)
    if (TMR1IF & TMR1IE) {                  //TMR1 フラグ検出
        if (i++ >= 3) {         //3 桁表示済かを検出
            i = 0;              //初期化
        }
        SEG_OFF();              //消灯
        if (i == 2) {   //ドットポイントの有無
            SEG_ON_DP(i, seg[i]);   //ドットポイント含めて表示
        }
        else {
            SEG_ON(i, seg[i]);  //ドットポイントなしで表示
        }
        TMR1L = DISP_LOW;           //65536 - 12500 = 53036 = 0xCF2C
        TMR1H = DISP_HIGH;          //0.2us x 12500 = 2.5ms
        TMR1IF = 0;                 //TMR1 フラグ初期化
    }
```

```
}

//移動平均用の関数(初回呼び出し時は配列内の各要素は 0)
//この関数は動作中にテストできないのでパソコン上で動作する環境での確認となる。
unsigned short move_average(unsigned short add)
{
    static unsigned char index;     //入替対象の要素番号
    static unsigned short sum, array[1 <<  AVERAGE_COUNT];

    sum -= array[index];    //最古のデータを合計から削除
    array[index] = add;     //最新のデータを最古の場所に追加
    sum += array[index];    //最新のデータを合計値に追加

    index++;            //入れ替える要素番号を+1
    //平均数が 8 のとき index = index & (0x07)と等価
    index &= (1 << AVERAGE_COUNT) - 1;  //要素番号の範囲を超えないようにマスク

    return sum >> AVERAGE_COUNT;        //合計値から平均個数を除算して平均値
}
```

作業工程計画書

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）

作業工程計画書（受講者配付用例）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備		
1. 2. 3. 4. 5. 6. 7. 8.		

（　　）内に当てはまる適切な語句を選択肢から選んで記入しなさい。

選択肢（　　）

作業工程計画書（模範解答例）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備		
1. 2. 3. 4. 5. 6. 7. 8.		

（　　）内に当てはまる適切な語句を選択肢から選んで記入しなさい。

選択肢 （　　）

作業工程計画書

<table>
<tr><th>作業工程</th><th>ポイント(留意事項等)</th><th>参考資料(写真、図面等)</th></tr>
<tr><td>準備</td><td>作業場所の確認・整理
使用機器・部品の確認
課題用ソースファイルの配布</td><td></td></tr>
<tr><td>1. 実習機・開発環境等の仕様を確認</td><td>マイコンの仕様
・電源電圧:DC5V
・動作周波数 Fosc:20MHz
・リセット:パワーオンリセットのみ

対象マイコンの入出力の割り付け設定
PORTA
<table>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>OSC</td><td>OSC</td><td>MCLR</td><td>OUT</td><td>IN</td><td>IN</td><td>IN</td><td>IN</td></tr>
</table>
PORTB
<table>
<tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>OUT</td><td>OUT</td><td>OUT</td><td>OUT</td><td>OUT</td><td>OUT</td><td>OUT</td><td>OUT</td></tr>
</table>
開発環境
MPLAB X

開発言語
XC8 コンパイラ</td><td></td></tr>
<tr><td>2. プログラミング(初期化)</td><td>1. 入出力の初期化

2. 周辺機能の初期化
(1)タイマ 0 の初期化
・ブザー駆動用の周波数に設定
・プリスケーラの比率
・カウントの初期値
(2)タイマ 1 の初期化
・2.5ms 程度の割込み周期に設定(表示サイクル)
・プリスケーラの比率: 1:32
・カウントの初期値
(3)A/D 変換の初期化
・10bit 分解能
・基準電圧は 0～5V
・変換速度:Fosc/64

3. 割込み許可
・タイマ 0 割込み許可
・全割込み許可</td><td></td></tr>
</table>

3. プログラミング （main 関数）	1. リセット直後に、各種初期化を実行する。 2. 押しボタンスイッチ1が押されるまでは待機状態とし、待機状態を示す表示を(″….″)を点滅表示する。 3. 押しボタンスイッチ 1 が押されると計測を開始する。計測中は以下の処理を行なうものとする。 ・ 一定時間毎（時間間隔はセンサの仕様等によるがここでは 5ms 程度に設定）にセンサの値を A/D 変換する。 ・ 変換された値の移動平均をとり表示器に出力する。 4. 平均値が既定値を超えるとブザーを出力する。押しボタンスイッチ 2 を押すとブザーを停止し、待機状態に移行する。	
4. プログラミング （割込み関数）	1. 割込み関数では表示サイクルまたは、ブザー出力の時間毎に目的の処理を行なう。 ・ 2ms 程度毎にダイナミック点灯の表示切替を行なう。 ・ 数 kHz 程度の割込みを発生させブザーの出力を反転させる。	
5. 動作確認	以下の項目について確認すること。 1. リセット直後に待機状態であること。待機状態中は″….″を点滅表示すること。 2. 待機状態中に押しボタンスイッチ 1 を押すと計測を開始する。計測中は表示器に測定対象の数値が表示されること。 3. 計測中に測定値が既定値を超えた場合ブザーを出力すること。 4. ブザーは押しボタンスイッチ 2 で停止して待機状態へ移行する。	

訓 練 課 題 確 認 シ ー ト

訓練科名　：

仕上がり像　：組込みマイコン制御システムの製作ができる。

システム名　：　C言語による組込み型マイクロコンピュータ制御技術　　入所期　：

訓練課題名　：　C言語を用いたマイコンによる計測制御　　氏　名　：

評価区分	評価項目	細　目	評価（数値）					評価判定	評価基準
作業時間	作業時間	作業時間内に作業を完了	2	4	6	8	10		指定時間以内に作業が完了していれば10点、以下15分経過する度に2点ずつ減点する。
	資料提出	ソースファイル等の提出	1	2	3	4	5		必要提出物が既定時間以内に提出されていれば5点、既定時間から10分経過する毎に1減点ずつ減点する。
作業工程	作業工程の確認	作業工程計画	1	2	3	4	5		作業のポイントが不適切な場合、1か所につき1点減点。
プログラミング	プログラム作成	マイコンの初期設定	1	2	3	4	5		マイコンの初期設定（動作に関連するモード設定やスタックメモリ等）が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点す
		I/O設定	1	2	3	4	5		入出力ポートの設定が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。
		タイマ設定	1	2	3	4	5		タイマの設定が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。
		A/D変換設定	1	2	3	4	5		A/D変換の設定が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。
		押しボタンスイッチの制御	1	2	3	4	5		スイッチの検出が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。
		表示器の制御	1	2	3	4	5		待機状態時、動作時の数値表示及び表示状態（チラツキ等なし）が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。
		ブザーの制御	1	2	3	4	5		ブザーが適切に動作していれば5点、以下不適切な箇所（ブザー音の途切れ等）があれば1点ずつ減点する。
		割込みの制御	1	2	3	4	5		割込みが適切に行なわれていれば5点、以下不適切な箇所があるたびに1点ずつ減点する。
アルゴリズム	制御	プログラム全体のフロー	2	4	6	8	10		プログラムの動作が仕様のとおり適切に動作していれば10点、以下不適切な箇所がある毎に2点ずつ減点する。
	プログラム	移動平均を算出する関数の作成とテスト	2	4	6	8	10		移動平均の関数が適切に動作していれば10点、以下不適切な箇所が1箇所ある毎に2点ずつ減点する。
安全作業	機材の取り扱い	機材の取り扱い、確認作業等	1	2	3	4	5		マイコン関連の機材の取り扱い、及び動作確認手順等が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。
	VDT作業	座る姿勢、ディスプレイ、休憩時間	1	2	3	4	5		椅子の高さ調整、モニタの角度を適正に行っていない場合、1箇所につき2点減点1時間ごとに休憩を取らない場合、1回につき1点減点、無理な姿勢で作業をしている場合は最低点の1点とする。
工夫・改善	追加・工夫等	課題で提示した以外の要素	2	4	6	8	10		課題で提示した以外の要素（ソースコードのコメント、計測中を示すブザー音の追加、待機中を示す表示等）が1箇所ある度に2点ずつ加点。
	工夫・改善点記入欄		総点						＜判定表＞ A　：　80点以上　：到達水準を十分に上回った B　：　60点以上80点未満　：到達水準に達した C　：　60点未満　：到達水準に達しなかった
			合計点						
			換算点						
			総合評価判定						

訓練課題のねらい	コメント
	担当指導員氏名：

評価要領

訓練科名　：
仕上がり像　：組込みマイコン制御システムの製作ができる。
システム名　：　C言語による組込み型マイクロコンピュータ制御技術
訓練課題名　：　C言語を用いたマイコンによる計測制御

評価区分	評価項目	細　目	評価要領（採点要領）	備考
作業時間	作業時間	作業時間内に作業を完了	指定時間以内に作業が完了していれば10点、以下15分経過する度に2点ずつ減点する。	試験官が管理する時計等で確認。
	資料提出	ソースファイル等の提出	必要提出物が既定時間以内に提出されていれば5点、既定時間から10分経過する毎に1減点ずつ減点する。	提出物で確認。
作業工程	作業工程の確認	作業工程計画	作業のポイントが不適切な場合、1か所につき1点減点	
プログラミング	プログラム作成	マイコンの初期設定	マイコンの初期設定（動作に関連するモード設定やスタックメモリ等）が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	プログラムでチェックする。I/O数が多いマイコンの場合は、課題に使用する箇所のみのチェックでも可。
		I/O設定	入出力ポートの設定が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	プログラムでチェックする。I/O数が多いマイコンの場合は、課題に使用する箇所のみのチェックでも可。
		タイマ設定	タイマの設定が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	
		A/D変換設定	A/D変換の設定が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	
		押しボタンスイッチの制御	スイッチの検出が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	待機状態から動作状態に移行した場合を検討すること。
		押しボタンスイッチ2の制御	操作時の動作が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	ブザー出力から待機状に移行した場合を検討すること。
		表示器の制御	待機状態時、動作時の数値表示及び表示状態（チラツキ等なし）が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	表示はLCD、7セグメントLED等、課題で指定した表示であること。表示サイクルが不適切でチラツキが見える場合については減点対
		ブザーの制御	ブザーが適切に動作していれば5点、以下不適切な箇所（ブザー音の途切れ等）があれば1点ずつ減点する。	ブザーから音が出力されていること。音が途切れがち、聞こえにくい場合には減点対象とする。
		ブザーの制御	割込みが適切に行なわれていれば5点、以下不適切な箇所があるたびに1点ずつ減点する。	
アルゴリズム	制御	プログラム全体のフロー	プログラムの動作が仕様のとおり適切に動作していれば10点、以下不適切な箇所がある毎に2点ずつ減点する。	
	プログラム	移動平均の算出する関数の作成とテスト	移動平均の関数が適切に動作していれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	
安全作業	機材の取り扱い	機材の取り扱い、確認作業等	機材の取り扱い、及び動作確認手順等が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	
	VDT作業	座る姿勢、ディスプレイ、休憩時間	椅子の高さ調整、モニタの角度を適正に行っていない場合、1箇所につき2点減点1時間ごとに休憩を取らない場合、1回につき1点減点、無理な姿勢で作業をしている場合は最低点の1点とする。	
工夫・改善	追加・工夫等	課題で提示した以外の要素	課題で提示した以外の要素（ソースコードのコメント、計測中を示すブザー音の追加、待機中を示す表示等）が1箇所ある度に2点加点。	工夫要素があれば加点対象とする。

実技課題

管理番号：Ｅ－45A
「パソコンを用いた計測制御システムの製作（Visual Basic）」

■課題概要■

パソコン計測のために必要となるプログラミング言語（Visual Basic）の要素と、制御機器、計測機器に関連するプログラミングについて実技試験により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領	○	E-45A-00_実施要領
訓練課題	○	E-45A-01_訓練課題
解答	○	E-45A-02_解答及び解説
作業工程手順書	○	E-45A-03_作業工程計画書【作成例】
訓練課題確認シート	○	E-45A-04_訓練課題確認シート及び評価要領
評価要領	○	E-45A-04_訓練課題確認シート及び評価要領

※「解答例」、「Help」、「Library」のフォルダ及び関連のデータがあります。

※プログラミング言語が異なるAとBの2タイプの課題があります。

実技課題　実施要領

訓練課題名「パソコンを用いた計測制御システムの製作（Visual Basic）」

- 作業準備（ハードウェア製作含む）及び使用機器の準備時間は別途確保した上で事前に行うこととし、作業時間に含まない。
- 作業時間は、休憩時間を除いた時間とする。
- 実施形態は、各施設で使用する機材に応じて 1 名で行なう。
- VDT 作業を考慮し、1 時間ごとに休憩時間を設ける。
- この課題は、パソコン用の開発言語（Visual Basic）を使用し、計測制御を行なうシステムを想定しているが、機器等の状況によってはパソコンのみで簡潔する課題でも良いものとする。
- 計測・制御対象については、施設内で実施している訓練内容や機器に合わせ課題を修正した上で実施することが望ましい。

タイムスケジュール例（訓練時間が 9:00～15:30 の場合）を以下に示す。

時間	実施内容
9：00～9：10	出欠確認
9：10～9：30	課題内容説明および質問
課題作成開始	
9：30～12：00	プログラム作成
12：00～13：00	昼食
13：00～14：00	課題作成終了（作成課題提出）
14：00～15：30	動作確認及び評価
課題作成終了（作成課題提出）	

実技課題

「パソコンを用いた計測制御システムの製作（Visual Basic)」

１　作業時間

240 分(4 時間:休憩を除く)

２　配付資料

問題用紙, 解答用紙

３　課題作成、提出方法

・各個人で作業すること

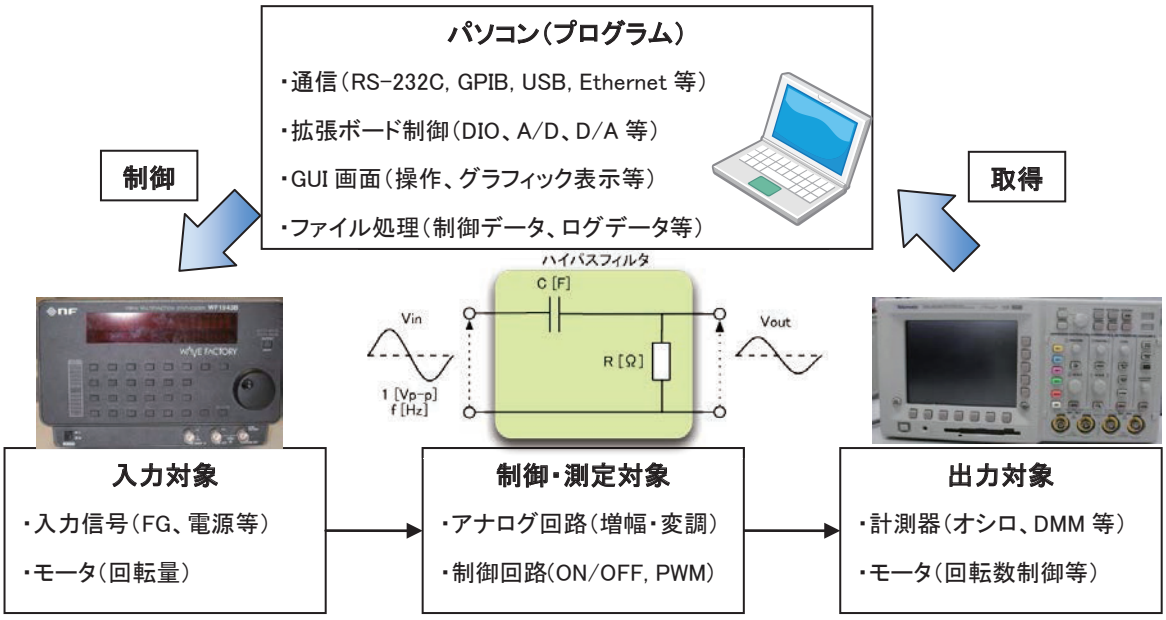
・ソースファイルによる提出

１．課題名：パソコンを用いた計測制御システム製作

２．課題内容（イメージ）

以下の仕様に基づいた計測制御システムのプログラムを作成しなさい。

システムの使用はパソコンから入力対象となる機器へ制御データを入力し、制御対象から得られたデータを出力対象となる機器からパソコンに出力し、その制御内容を GUI で管理する。ただし実習環境によってはパソコンのみで行なって良いものとする。



① 各機器の電源を投入し、プログラムを起動する。

② プログラム起動後、「開始」のボタンをクリックすると入力対象からデータを取得する。

A) 対象がオシロの場合、描画中の波形のデータを RS-232C や GPIB といった計測向けのインタフェース(USB や Ethernet も可)で取得する。

B) 計測対象の状態を ON/OFF 信号（回転数等)、あるいはアナログ信号取得する。

C) 機材の都合上、入力対象機器が用意できない場合は、入力機器の信号を想定したログ等を CSV ファイル形式で用意する。

③ データを取得後、目的の制御に応じた結果を出力対象に反映させる。

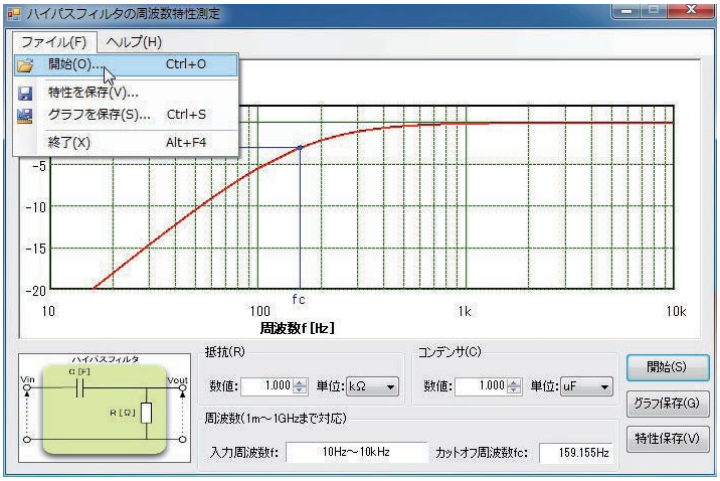
A) 対象が FG の場合、アプリケーションで指定した波形を FG に出力させる信号を送信する。

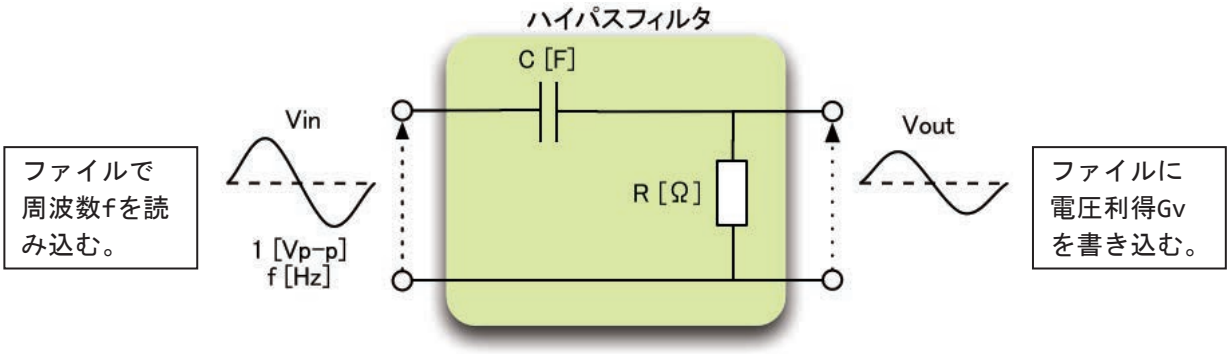
B) 入力対象の信号から演算した結果を出力対象へ出力する。A/D、D/A 変換をしている場合はディジタル値とアナログ信号の換算を行なうこと。

C) 機材の都合上、入力対象機器が用意できない場合は、入力された信号（もしくはファイル）を時系列もしくは特性等（例：周波数等）でまとめたものをグラフで描画し、グラフの保存機能を追加する。

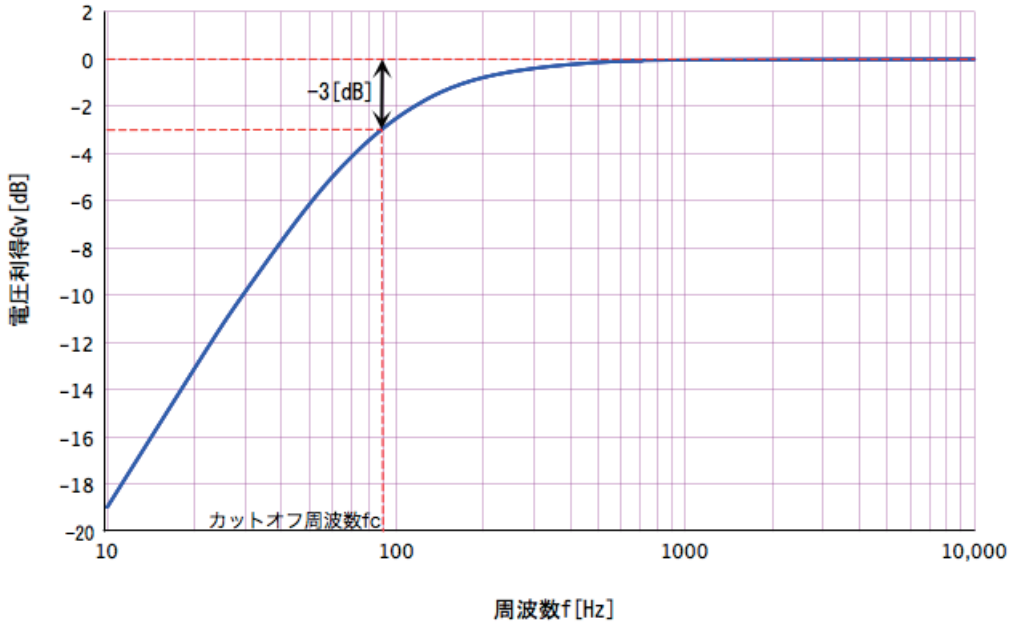
仕様例（フィルタ回路のカットオフ周波数 fc の評価）

下記の回路図は抵抗Rとコンデンサcによって構成されるハイパスフィルタの回路です。この回路の抵抗値 R とコンデンサ C を設定し、フィルタの性能を表す周波数特性の表示と及びカットオフ周波数の算出を行なうプログラムを作成します。





電圧利得： $Gv = 20\log_{10}\left(\frac{Vout}{Vin}\right)$



制御対象回路（ハイパスフィルタ）の電圧利得の周波数（f-Gv）特性

※（X 軸は周波数を対数軸、Y 軸は電圧利得を線形軸で描画）

この回路に 1[Vp-p](固定)の周波数 f[Hz](可変)の電圧 Vin を入力し、出力電圧 Vout から周波数特性とカットオフ周波数 fc（電圧利得 Gv[dB]が-3[dB]低下した周波数）を算出し描画するプログラムを作成しなさい。作成条件は以下のとおりとします。

1. パソコン上で抵抗 R とコンデンサ C の値を指定し、「開始」のボタンをクリックすると周波数特性を測定する。測定する周波数の範囲は右図のようなテキストファイルを選択し読み込むものとする。

   なおカットオフ周波数 fc の理論値は指定された抵抗 R とコンデンサ C を用いて以下の式から算出できる。

$$f_C = \frac{1}{2\pi CR}$$

```
10
20
30
50
70
100
200
300
500
700
1000
2000
…
```

2. フィルタの電圧利得 Gv[dB]は以下の式を用いて算出する。

$$Gv = 20\log_{10}\left(\frac{Vout}{Vin}\right) \qquad \frac{Vout}{Vin} = \sqrt{\frac{(\omega CR)^2}{1+(\omega CR)^2}} \qquad \omega = 2\pi f$$

3. 周波数 f を変化させ、周波数特性を表すグラフを描画する。グラフの X 軸は周波数[Hz]（対数軸）、Y 軸は電圧利得[dB]とする。グラフの描画は、提供する Graph クラスを使用しても良いものとする。

4. 周波数特性からカットオフ周波の測定値 fc を求め表示する。

5. 「保存」のボタンをクリックすると、周波数特性のグラフを「名前を付けて保存」のダイアログを表示して画像ファイル(BMP、JPEG、PNG 等のパソコンで表示可能なファイル形式)を任意の場所に保存できるようにすること。

6. 独自の機能を追加してもよい。以下は例である。

7. アプリケーションの操作方法や動作についてテキストファイル等で第 3 者が理解できるようなドキュメントを作成すること（簡易的なもので可）。独自の機能を追加し合は追加した機能をドキュメントに記載すること。

３．作業時間：240 分（4 時間）

・機材の準備時間等は含まないものとする。

・これまでの訓練で使用したソースファイル等を流用して良いものとする。

４．課題仕様

（１）課題に必要な機器を用意する。実機が用意できない場合はプログラムのみの課題でも良いものとする。

機器	実機による例	プログラムのみによる例 （本資料に記載）
パソコン(PC)	Windows XP/Vista/7/8	Windows 7
開発環境	Visual Studio 2005/2008/2010/2012	Visual Studio 2005 以降
入力・出力 I/F	インタフェース社　PCI-3522A RS-232C/USB/Ethernet/GP-IB	なし
入力機器	ファンクションジェネレータ	CSV ファイルの読み込み
出力機器	オシロスコープ	CSV ファイルの書き込み
制御・負荷回路	フィルタ回路、モータ制御回路	フィルタ回路の計算

（２）動作仕様

① 各種機器の接続（電源、入出力信号、通信ケーブル等）を行なう。

② 機器の電源を投入し、動作可能な状態とする。パソコン側は作成したアプリケーションを起動する。

③ アプリケーションにパラメータ（測定時間、測定用数値）を入力し「開始」のボタンをクリックすると計測を開始する。

④ 計測を開始し制御機器の特性と、その結果を表示する。

グラフ描画には別途配布する DLL を開発環境に組み込んで作成すること。また使用例については後述のソースファイルを参照すること。

（３）作業内容

1) Visual Studio を起動しプロジェクトを Windows フォームアプリケーションで作成する。言語の仕様やバージョン等については講師の指示に従うこと。

2) 提出物

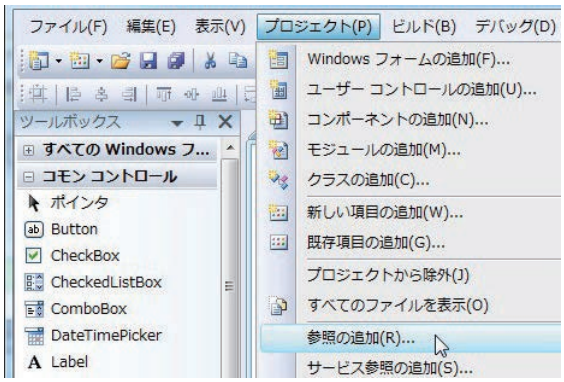
	機能	備考
1	作成したプログラムのファイル一式	
2	プログラムのドキュメント（操作手順、動作、追加機能等を記載したもの）	テキストファイル （形式は講師で指示）

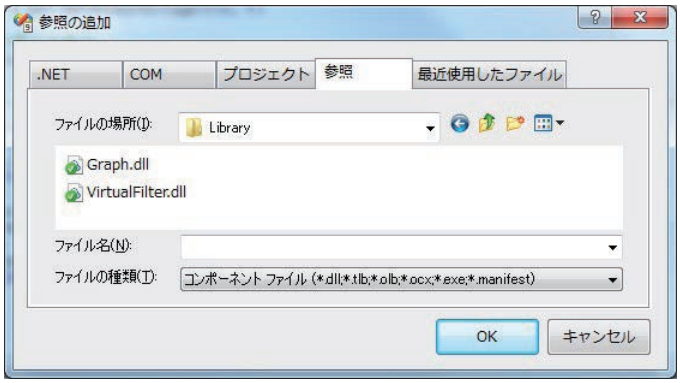
５．参考資料

参考として、グラフを描画する Graph クラスのサンプルを以下に示す。

（１）Graph クラスの使用方法(GraphSample.zip に格納)

1) プロジェクト作成後、グラフ描画用の DLL(Graph.dll)をメニューバーの「プロジェクト」→「参照の追加」から以下のダイアログを表示し、「参照」のタブで、配布した"Graph.dll"を選択する。





2) "Polytech.Windows.Form"名前空間で Graph クラスを定義しているため、Imports ステートメントで"Polytech.Windows.Form"名前空間を追加すること。

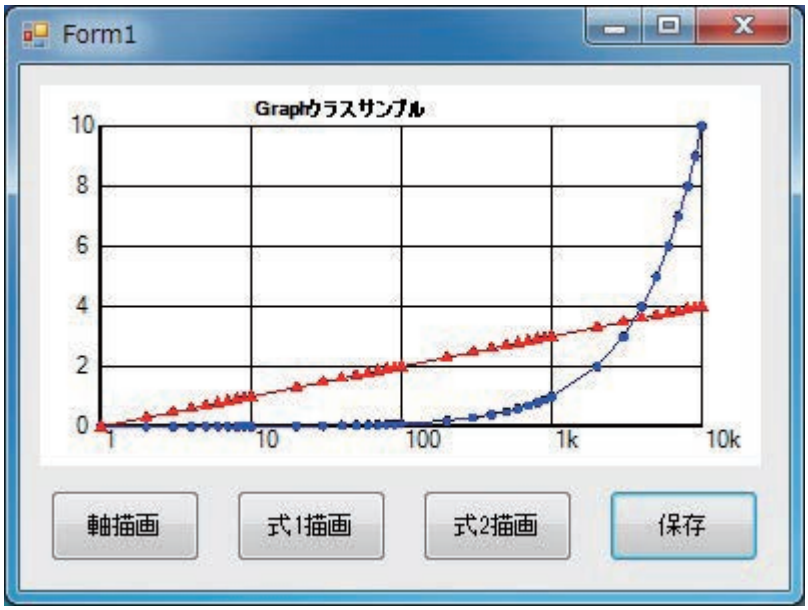
```
'独自の名前空間
Imports Polytech.Windows.Form    'Graphクラス
'既定の名前空間
Imports System.Text
```

3) 例として以下の式を x が対数軸で描画するプログラムのサンプルを示す。x の範囲は 1～10000 とする。

コントロール	オブジェクト名	備考
Form	Form1	Graphクラスの初期化と軸の描画、初期描画の登録
Button	GraphInit	初期描画状態を呼び出す
	DrawLine1	式1のグラフを描画
	DrawLine2	式2のグラフを描画
	SaveGraph	グラフを画像ファイルとして保存
PictureBox	GraphDisplay	グラフの描画対象

$$y = \log_{10} x \quad \cdots\cdots \text{（式 1）}$$

$$y = \frac{x}{1000} \quad \cdots\cdots \text{（式 2）}$$



Graph クラスを用いたグラフの描画サンプル

行	プログラム
1	`Imports Polytech.Windows.Form   'Graphクラス`
2	
3	`Public Class Form1`
4	
5	`    Private myGraph As Graph    'Graphクラスのオブジェクト`
6	
7	`    ''' <summary>プログラム起動時に実行</summary>`
8	`    Private Sub Form1_Load(ByVal sender As System.Object,` `                                ByVal e As System.EventArgs) Handles MyBase.Load`
9	
10	`        Dim x, y As Double  '数式用`
11	`        Dim i As Integer    'ループ変数`
12	
13	`        'PictureBox1に描画座標を左下(1,0)、右上(10000, 10), 左右の余白20pixel,`
14	`        '上下の余白30pixel, X軸を対数軸, Y軸を線形軸でグラフを描画`
15	`        myGraph = New Graph(GraphDisplay, 1, 0, 10000, 10,` `                                            30, 20, Graph.AxisStyle.XlogYlinear)`
16	
17	`        myGraph.BackColor = Color.White '背景色を白`
18	
19	`        'フォントの設定(フォント：ＭＳ ゴシック"、サイズ：8pt、スタイル：太字、色：黒)`
20	`        myGraph.FontName = "ＭＳ ゴシック""`
21	`        myGraph.FontSize = 8`
22	`        myGraph.FontStyle = FontStyle.Bold`
23	`        myGraph.FontColor = Color.Black`
24	
25	`        'グラフのタイトルを(10,11)へ描画`
26	`        myGraph.DrawText(10, 11, "Graphクラスサンプル")`
27	

```
28          'フォントの一括設定(フォント：MSゴシック、サイズ：7pt、スタイル：標準、色：青)
29          myGraph.FontSet("MS ゴシック", 7, FontStyle.Regular, Color.Blue)
30
31          '線のスタイルを実線、太さを2、色を黒
32          myGraph.LineStyle = Drawing2D.DashStyle.Solid
33          myGraph.LineWidth = 2
34          myGraph.LineColor = Color.Black
35
36          'X軸とY軸を描画
37          myGraph.DrawLine(1, 0, 10000, 0)    '(1,0)から(10000,0)へ直線描画
38          myGraph.DrawLine(1, 0, 1, 10)       '(1,0)から(1,10)へ直線描画
39          'グラフのタイトルを(10,11)へ描画
40
41          '線のスタイルを破線、太さを1、色を緑
42          myGraph.LineStyle = Drawing2D.DashStyle.Dash
43          myGraph.LineWidth = 1
44          myGraph.LineColor = Color.Green
45
46          'x軸の目盛線と目盛の数値の描画
47          For i = 0 To 4 Step 1
48              x = 10 ^ i
49              myGraph.DrawLine(x, 0, x, 10)
50              If 3 <= i And i < 6 Then   '10の3乗以上6未満で補助単位k(キロ)
51                  myGraph.DrawText(x, 0, (x / 1000).ToString("###k"))
52              Else    '上記以外のとき補助単位なし
53                  myGraph.DrawText(x, 0, x.ToString())
54              End If
55          Next
56
57          'y軸の目盛線と目盛の数値の描画
58          For i = 0 To 10 Step 2
59              y = i
60              myGraph.DrawLine(1, y, 10000, y)
61              myGraph.DrawText(0.6, y + 0.5, y.ToString.PadLeft(2))
62          Next
63
64          '描画したイメージを記憶する
65          myGraph.SaveInitialGraph()
66
67      End Sub
68
69      ''' <summary>グラフの初期状態描画</summary>
70      Private Sub GraphInit_Click(ByVal sender As System.Object,
                                    ByVal e As System.EventArgs) Handles GraphInit.Click
71
72          myGraph.LoadInitialGraph()  '初期
73
74      End Sub
75
76      ''' <summary>式1のグラフ描画</summary>
```

```
77      Private Sub DrawLine1_Click(ByVal sender As System.Object,
                                    ByVal e As System.EventArgs) Handles DrawLine1.Click
78
79          Dim x, y As Double  '数式用
80          Dim i, j, k As Integer    'ループ変数
81          Dim series(36) As PointF    '座標格納用用配列
82
83          '1～10000まで対数目盛間隔で配列に代入
84          '1,2,3,...,9,10,20,30,...,90,100,200
85          For i = 0 To 4 Step 1       '10のi乗
86              For j = 1 To 9 Step 1   '1～9の値
87                  x = j * (10 ^ i)    'xの値
88                  y = x / 1000        'yの値
89                  series(k) = New PointF(x, y)    '配列に格納
90                  k = k + 1                       '次の配列の要素番号
91                  If x >= 10000 Then Exit For '10000まで格納後このステートメントを抜ける
92              Next
93          Next
94
95          '線のスタイルを実線、太さを1、色を青
96          myGraph.LineStyle = Drawing2D.DashStyle.Solid
97          myGraph.LineWidth = 1
98          myGraph.LineColor = Color.Blue
99
100         'マーカーのスタイルを円、幅を5px、色を青
101         myGraph.MarkerStyle = Graph.MarkerShapeStyle.Circle
102         myGraph.MarkerWidth = 5
103         myGraph.MarkerColor = Color.Blue
104
105         '最初の点のマーカーを描画
106         myGraph.DrawMarker(series(0))
107
108         '配列内の座標の2点間を直線で描画、終点にはマーカー描画
109         For i = 0 To series.Length - 2 Step 1
110             myGraph.DrawMarker(series(i + 1))
111             myGraph.DrawLine(series(i), series(i + 1))
112         Next
113
114     End Sub
115
116     ''' <summary>式2のグラフ描画</summary>
117     Private Sub DrawLine2_Click(ByVal sender As System.Object,
                                    ByVal e As System.EventArgs) Handles DrawLine2.Click
118
119         Dim formula As New PointF  '数式用
120         Dim i As Integer    'ループ変数
121         Dim series As New List(Of PointF)'PointF構造体でListクラスのインスタンス生成
122
123         '1～10000まで対数目盛間隔(1,2,3,...,9,10,20,30,...,90,100,200...)で追加
124         For i = 0 To 4 Step 1       '10のi乗
```

```
125             For j = 1 To 9 Step 1   '1～9の値
126                 formula.X = j * (10 ^ i)    'xの値
127                 formula.Y = Math.Log10(formula.X)       'yの値
128                 series.Add(formula)    '座標を追加
129                 If formula.X >= 10000 Then Exit For
                                            '10000まで格納後このステートメントを抜ける
130             Next
131         Next
132
133         '線のスタイルを実線、太さを1、色を赤
134         myGraph.LineSet(Drawing2D.DashStyle.Solid, 1, Color.Red)
135
136         'マーカーの形状を三角、幅を7px、色を赤
137         myGraph.MarkerSet(Graph.MarkerShapeStyle.Triangle, 7, Color.Red)
138
139         '配列内の座標の2点間を直線で描画、終点にはマーカー描画
140         For i = 0 To series.Count - 1 Step 1
141             If i = 0 Then   '最初の点のマーカーのみ描画
142                 myGraph.DrawMarker(series(0))
143             Else            '以降は前に描画した座標を始点として直線を描画
144                 myGraph.DrawLine(series(i))
145                 myGraph.DrawMarker(series(i))
146             End If
147         Next
148
149     End Sub
150
151     ''' <summary>グラフの保存</summary>
152     Private Sub SaveGraph_Click(ByVal sender As System.Object,
                                    ByVal e As System.EventArgs) Handles SaveGraph.Click
153
154         Dim filePath As String  '画像ファイルの保存先のパス
155
156         filePath = "C:¥Temp¥Graph.jpg"'保存先の代入(形式はBMP, JPEG, GIF. PNG等)
157
158         'Exportメソッドで描画しているPictureBoxのグラフィックを保存
159         If myGraph.Export(filePath) = True Then
160             '保存成功時はTrue
161             MessageBox.Show(filePath & "に保存しました。", "完了",
                                    MessageBoxButtons.OK, MessageBoxIcon.Information)
162         Else
163             '失敗時はFalse
164             MessageBox.Show(filePath & "に保存できません。
                        ", "失敗", MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
165         End If
166
167     End Sub
168
169 End Class
```

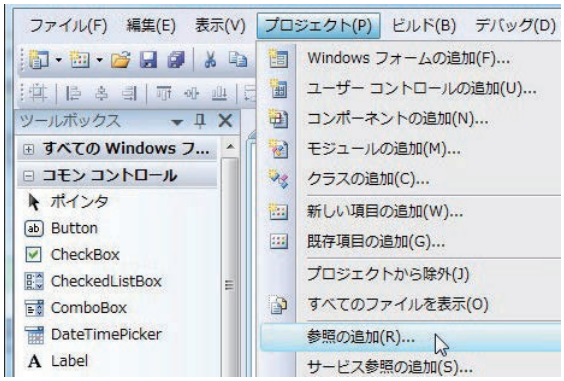
実技課題　解答及び解説

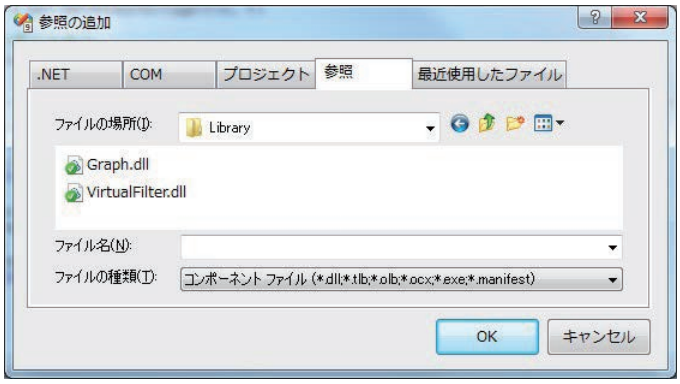
「パソコンを用いた計測制御システムの製作（Visual Basic)」

解答例を以下に示す。

（１）事前の設定(別途ファイルに格納)

1) プロジェクトを開いたあと、グラフ描画用の DLL(Graph.dll)をメニューバーの「プロジェクト」→「参照の追加」から以下のダイアログを表示し、「参照」のタブで、配布した"Graph.dll"を選択する。



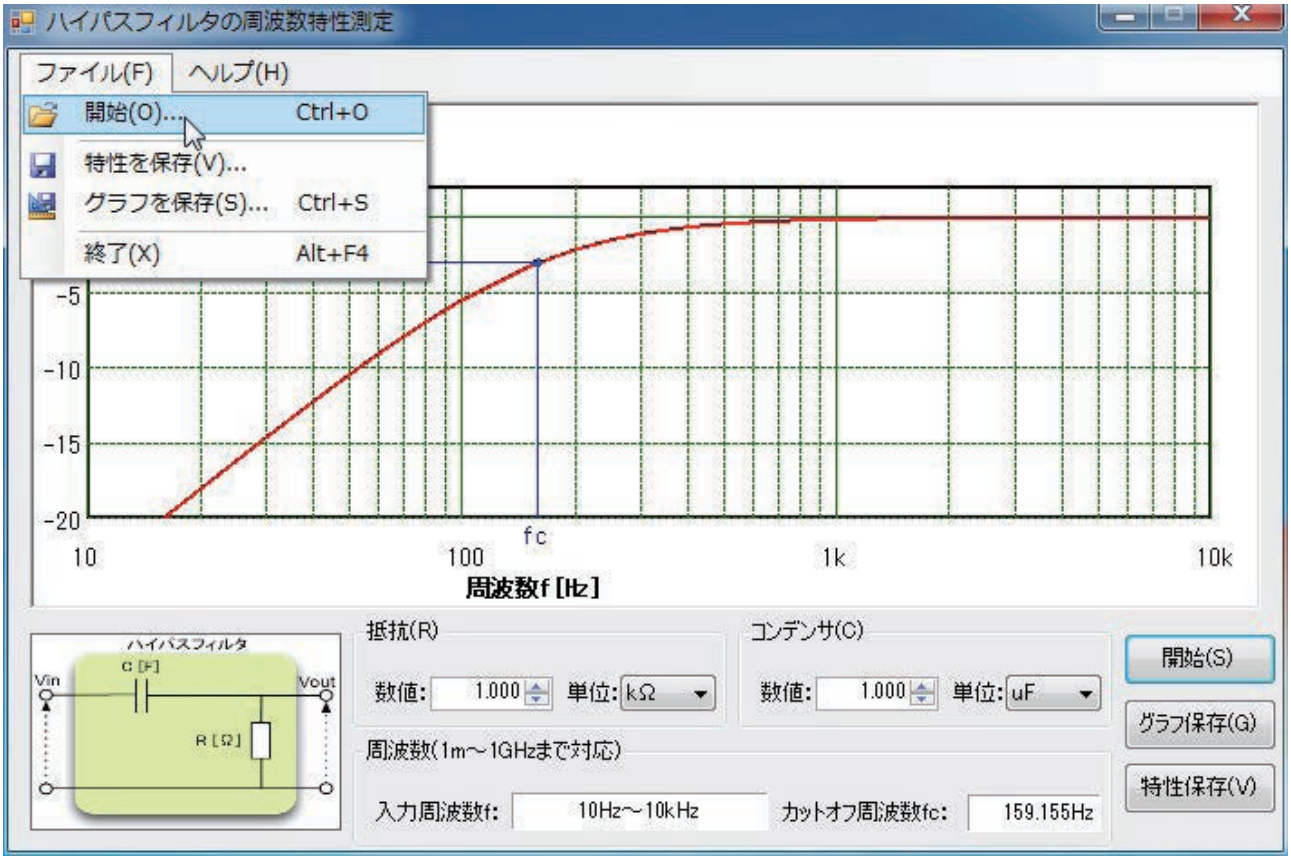


2) "Polytech.Windows.Form"名前空間で Graph クラスを定義しているため、Imports ステートメントで"Polytech.Windows.Form"名前空間を追加すること。

```
'名前空間のインポート
Imports System.Drawing
Imports System.Drawing.Drawing2D
Imports System.Drawing.Imaging
'グラフ用
Imports Polytech.Windows.Form
```

3) コントロールと各機能

コントロール	オブジェクト名	備考
Form	Form1	Graphクラスの初期化と軸の描画、初期描画の登録
Button	GraphInit	初期描画状態を呼び出す
	SaveCSV	周波数特性をCSVファイル等で保存
	SaveGraph	グラフを画像ファイルとして保存
NumericUpdown	InputNumberR	抵抗値の数値を入力
	InputNumberC	静電容量の数値を入力
ListBox	ListPrefixR	抵抗値の単位を入力
	ListPrefixC	静電容量の単位を入力
Label	SimlateShowFreq	カットオフ周波数の理論値を表示
	MeasureShowFreq	カットオフ周波数の測定値を表示
	DifferenceFreqPercemt	理論値と測定値の誤差を%表示
PictureBox	GraphDisplay	グラフの描画対象



解答例のソースコード（フォーム）

行	プログラム
1	'名前空間のインポート
2	Imports System
3	Imports System.Text
4	Imports System.IO
5	Imports System.Windows.Forms
6	Imports System.Drawing
7	Imports System.Drawing.Drawing2D
8	Imports System.Drawing.Imaging
9	Imports System.Math
10	'グラフ用
11	Imports Polytech.Windows.Form
12	
13	Public Class GraphForm
14	
15	'フォーム内で使用するメンバー
16	Private myGraph As Graph 'グラフ描画用オブジェクト
17	Private PartR, PartC As Decimal '抵抗値，静電容量
18	Private CutoffPoint As PointF 'カットオフ周波数の座標
19	Private ReadOnly GvRangeMin As Single = -20 'Y軸最小値
20	Private ReadOnly GvRangeMax As Single = 2 'Y軸最大値
21	Private ReadOnly FreqRangeMin = 10 ^ -3 'X軸最小値
22	Private ReadOnly FreqRangeMax = 10 ^ 9 'X軸最大値
23	Private ReadOnly Vin As Decimal = 1 '入力電圧
24	Private Series As New List(Of PointF) 'グラフの系列

```
25      Private hpf As HighPassFilter   '測定対象のハイパスフィルタ
26
27      ''' <summary>ウィンドウを閉じたときに実行されるイベントです。</summary>
28      Private Sub GraphForm_FormClosing(ByVal sender As Object, ByVal e As
                System.Windows.Forms.FormClosingEventArgs) Handles Me.FormClosing
29
30          Dim result As Windows.Forms.DialogResult    'MessageBoxの選択結果
31
32          result = MessageBox.Show("プログラムを終了しますか?", "確認",
                                     MessageBoxButtons.YesNo, MessageBoxIcon.Question)
33          If result = Windows.Forms.DialogResult.No Then
34              e.Cancel = True     'このイベントをキャンセル
35          End If
36
37      End Sub
38
39      ''' <summary>起動時に実行されるイベントです。</summary>
40      Private Sub GraphForm_Load(ByVal sender As System.Object, ByVal e As
                                               System.EventArgs) Handles MyBase.Load
41
42          'ピクチャーボックスの外形線を立体表示
43          GraphDisplay.BorderStyle = BorderStyle.Fixed3D
44
45          '抵抗の単位のリスト追加
46          ListPrefixR.Items.Add("Ω")
47          ListPrefixR.Items.Add("kΩ")
48          ListPrefixR.Items.Add("MΩ")
49          ListPrefixR.SelectedIndex = 1
50          '抵抗値の入力範囲の設定
51          InputNumberR.Value = 1
52          InputNumberR.Maximum = 10 ^ 5
53          InputNumberR.Minimum = 10 ^ -3
54          InputNumberR.DecimalPlaces = 3
55          InputNumberR.Increment = 0.01
56
57          '静電容量の単位のリスト追加
58          ListPrefixC.Items.Add("pF")
59          ListPrefixC.Items.Add("nF")
60          ListPrefixC.Items.Add("uF")
61          ListPrefixC.SelectedIndex = 2
62          '静電容量の入力範囲の設定
63          InputNumberC.Value = 1
64          InputNumberC.Maximum = 10 ^ 5
65          InputNumberC.Minimum = 10 ^ -3
66          InputNumberC.DecimalPlaces = 3
67          InputNumberC.Increment = 0.01
68
69          'イベントの関連付け(上記の設定変更でイベントが動作しないようにするため)
70          AddHandler InputNumberR.ValueChanged, AddressOf InputParameter
71          AddHandler InputNumberC.ValueChanged, AddressOf InputParameter
```

```
72          AddHandler ListPrefixR.SelectedValueChanged, AddressOf InputParameter
73          AddHandler ListPrefixC.SelectedValueChanged, AddressOf InputParameter
74
75          'パラメータ入力のイベントを呼び出す
76          InputParameter(sender, e)
77
78      End Sub
79
80      ''' <summary>周波数特性の実測値の描画とカットオフ周波数を開始するイベントです。</summary>
81      Private Sub drawGraph_Click(ByVal sender As System.Object, ByVal e As
                    System.EventArgs) Handles DrawGraph.Click, MenuFileOpen.Click
82
83          Dim i As Integer
84          Dim freqMin, freqMax As Single
85          Dim freqList As New List(Of Single) 'ファイルから読み込んだ周波数
86          Dim graphFreqMin, graphFreqMax As Single'グラフに適用する周波数の最小と最大値
87
88          '①ファイル読み込み最小、最大周波数と周波数のListの取得
89          If OpenCSV(freqMin, freqMax, freqList) = False Then Exit Sub
90
91          '②測定対象のハイパスフィルタのインスタンス生成
92          hpf = New HighPassFilter(PartR, PartC)  '抵抗とコンデンサのパラメータ
93          CutoffPoint.X = hpf.GetCutoffFrequency()          'カットオフ周波数の取得
94          CutoffPoint.Y = hpf.GetGain(Vin, CutoffPoint.X)           '利得の計算
95          CutoffFreqLabel.Text = AddPrefixUnit(CutoffPoint.X, "Hz")
96
97          '③周波数の範囲を表示
98          FreqRangeLabel.Text = AddPrefixUnit(freqMin, "Hz") & "～" &
                                                        AddPrefixUnit(freqMax, "Hz")
99          '周波数の範囲を10の乗数になるように変換してからグラフを初期化
100         graphFreqMin = Convert.ToSingle(10 ^ Floor(Log10(freqMin)))
101         graphFreqMax = Convert.ToSingle(10 ^ Ceiling(Log10(freqMax)))
102         DrawSheet(graphFreqMin, graphFreqMax)
103
104         '④描画データの生成
105         Series.Clear()  'グラフの座標のリストをクリア
106         '電圧利得を算出して系列データに代入
107         For Each freq As Single In freqList
108             Series.Add(New PointF(freq, hpf.GetGain(Vin, freq)))
109         Next
110
111         '⑤系列データ間を線分で描画
112         myGraph.LineSet(DashStyle.Solid, 2, Color.Red)
113         For i = 0 To Series.Count - 2
114             If GvRangeMin <= Series(i).Y And Series(i).Y <= GvRangeMax Then
115                 myGraph.DrawLine(Series(i), Series(i + 1))
116             End If
117         Next
118
```

```
119         '⑥カットオフ周波数の描画
120         myGraph.MarkerSet(Graph.MarkerShapeStyle.Circle, 5, Color.Blue)
121         myGraph.DrawMarker(CutoffPoint)
122         myGraph.LineSet(DashStyle.Solid, 1, Color.Blue)
123         myGraph.DrawLine(CutoffPoint.X, GvRangeMin, CutoffPoint.X, CutoffPoint.Y)
124         myGraph.DrawLine(graphFreqMin, CutoffPoint.Y, CutoffPoint.X,
                                                                CutoffPoint.Y)
125         myGraph.FontColor = Color.Blue
126         myGraph.DrawText(CutoffPoint.X * 0.9, GvRangeMin - 0.5, "fc")
127         myGraph.DrawText(graphFreqMin * 0.7, -3 + 0.5, AddPrefixUnit(-3).PadLeft(4))
128
129     End Sub
130
131     ''' <summary>グラフの目盛等を描画するメソッドです。</summary>
132     '''<param name="frequencyMin" >描画する周波数の最小値を指定します。</param>
133     ''' <param name="frequencyMax">描画する周波数の最大値を指定します。</param>
134     Private Sub DrawSheet(ByVal frequencyMin As Single, ByVal frequencyMax As Single)
135
136         Dim digitMin, digitMax As Integer
137
138         digitMin = Convert.ToInt32(Math.Log10(frequencyMin))
139         digitMax = Convert.ToInt32(Math.Log10(frequencyMax))
140         '範囲を左下(最小周波数, 最小利得-20dB)と右上(最大周波数, 最大利得2dB),余白の上
                                下を30pixelと左右20pixel,X軸線形軸,Y軸対数軸でグラフを描画
141         myGraph = New Graph(GraphDisplay, frequencyMin, GvRangeMin - 4,
                    frequencyMax, GvRangeMax + 3, 30, 20, Graph.AxisStyle.XlogYlinear)
142
143         '背景色は白
144         myGraph.BackColor = Color.White
145
146         'フォントの変更
147         myGraph.FontSet("ＭＳ ゴシック", 10, FontStyle.Regular, Color.Black)
148
149         '線を実線、幅2、黒に変更
150         myGraph.LineSet(DashStyle.Solid, 2, Color.Black)
151         myGraph.DrawRectangle(frequencyMin, GvRangeMin, frequencyMax, GvRangeMax)
152
153         'X軸の目盛線を追加
154         For i = digitMin To digitMax Step 1
155             For j = 1 To 9 Step 1
156                 If j = 1 Then
157                     myGraph.LineSet(DashStyle.Solid, 1, Color.Green)
158                     myGraph.DrawText(j * 10 ^ i * 0.9, GvRangeMin - 2,
                                                            AddPrefixUnit(j * 10 ^ i))
159                 Else
160                     myGraph.LineSet(DashStyle.Dash, 1, Color.Green)
161                 End If
162                 myGraph.DrawLine(j * 10 ^ i, GvRangeMin, j * 10 ^ i, GvRangeMax)
163             Next
164         Next
```

```
165         'X軸のタイトル表示
166         myGraph.FontStyle = FontStyle.Bold
167         myGraph.DrawText(10 ^ (digitMax - digitMin - 1), GvRangeMin - 4, "周波数f[Hz]")
168         myGraph.FontStyle = FontStyle.Regular
169
170         'Y軸の目盛線を追加
171         For i = GvRangeMin To GvRangeMax Step 5
172             If i = 0 Then
173                 myGraph.LineSet(DashStyle.Solid, 1, Color.Green)
174             Else
175                 myGraph.LineSet(DashStyle.Dash, 1, Color.Green)
176             End If
177             myGraph.DrawLine(frequencyMin, i, frequencyMax, i)
178             myGraph.DrawText(frequencyMin * 0.7, i + 0.5,
                                                    AddPrefixUnit(i).PadLeft(4))
179         Next
180         'Y軸のタイトル表示
181         myGraph.FontStyle = FontStyle.Bold
182         myGraph.DrawText(frequencyMin * 0.8, GvRangeMax + 2, "電圧利得Gv[dB]")
183         myGraph.FontStyle = FontStyle.Regular
184
185         Series.Clear() ' 系列データのクリア
186
187     End Sub
188
189     ''' <summary>抵抗Rとコンデンサcの入力された数値と単位からパラメータを演算する
                                                    メソッドです。</summary>
190     Private Sub InputParameter(ByVal sender As System.Object,
                                                    ByVal e As System.EventArgs)
191
192         '入力されたRとCが0以下の時は以下を実行しない
193         If (InputNumberR.Value <= 0) OrElse (InputNumberC.Value <= 0) Then Exit Sub
194
195         Select Case ListPrefixR.Text
196             Case "MΩ"
197                 PartR = InputNumberR.Value * 10 ^ 6
198             Case "kΩ"
199                 PartR = InputNumberR.Value * 10 ^ 3
200             Case Else
201                 PartR = InputNumberR.Value
202         End Select
203
204         Select Case ListPrefixC.Text
205             Case "pF"
206                 PartC = InputNumberC.Value * 10 ^ -12
207             Case "nF"
208                 PartC = InputNumberC.Value * 10 ^ -9
209             Case "uF"
210                 PartC = InputNumberC.Value * 10 ^ -6
211             Case Else
```

```
212                 PartC = InputNumberC.Value
213             End Select
214
215         End Sub
216
217         ''' <summary>数値にSI接頭辞と単位を付加して文字列を返すメソッドです。</summary>
218         ''' <param name="number">数値を指定します。</param>
219         ''' <param name="unit">単位を指定します。省略した場合は単位は付加されません。
    </param>
220         ''' <returns>小数点以下第3位までの数値にSI接頭辞を文字列を返します。</returns>
221         Private Function AddPrefixUnit(ByVal number As Single,
                                           Optional ByVal unit As String = "") As String
222
223             Dim addString As String = String.Empty
224
225             Select Case number
226                 Case Is > 10 ^ 12
227                     addString = (number / (10 ^ 12)).ToString("0.###") & "T"
228                 Case Is >= 10 ^ 9
229                     addString = (number / (10 ^ 9)).ToString("0.###") & "G"
230                 Case Is >= 10 ^ 6
231                     addString = (number / (10 ^ 6)).ToString("0.###") & "M"
232                 Case Is >= 10 ^ 3
233                     addString = (number / (10 ^ 3)).ToString("0.###") & "k"
234                 Case Is >= 10 ^ 0
235                     addString = number.ToString("0.###")
236                 Case Is >= 10 ^ -3
237                     addString = (number * (10 ^ 3)).ToString("0.###") & "m"
238                 Case Is >= 10 ^ -6
239                     addString = (number * (10 ^ 6)).ToString("0.###") & "u"
240                 Case Is >= 10 ^ -9
241                     addString = (number * (10 ^ 9)).ToString("0.###") & "n"
242                 Case Is >= 10 ^ -12
243                     addString = (number * (10 ^ 12)).ToString("0.###") & "p"
244                 Case Else
245                     addString = number.ToString("0.###")
246             End Select
247
248             addString = addString & unit
249
250             Return addString
251
252         End Function
253
254         ''' <summary>ダイアログを表示して周波数のファイルを読み込みます。</summary>
255         ''' <param name="fMin">最小周波数を参照渡しで取得します。</param>
256         ''' <param name="fMax">最大周波数を参照渡しで取得します。</param>
257         ''' <param name="fList">周波数をListクラスで取得します。</param>
258         ''' <returns>ファイルを正常に読み込めた場合はTrueを、
                                                読み込めない場合はFalseを返します。</returns>
```

```
259     Private Function OpenCSV(ByRef fMin As Single, ByRef fMax As Single,
                                 ByRef fList As List(Of Single)) As Boolean
260
261         Dim openDialog As New OpenFileDialog    'ファイルから読み込むためのダイアログ
262
263         fList = New List(Of Single)             'ファイルから読み込んだ周波数
264
265         'ダイアログのタイトル
266         openDialog.Title = "入力信号用のファイルを読み込んでください。"
267         'デフォルトのファイル名指定なし
268         openDialog.FileName = ""
269         'デスクトップをデフォルトのフォルダに指定
270         openDialog.InitialDirectory =
                               My.Computer.FileSystem.SpecialDirectories.Desktop
271         'ファイルの種類はCSVとText,すべてのファイル
272         openDialog.Filter =
                       "CSVファイル|*.csv|テキストファイル|*.txt|すべてのファイル|*.*"
273         'ダイアログの表示がOK以外の時は以下の処理を実行しない
274         If openDialog.ShowDialog() <> Windows.Forms.DialogResult.OK Then
                                                                Return False
275
276         'ファイルを読み込む処理
277         Using sr As New StreamReader(openDialog.FileName)
278             Do Until sr.EndOfStream     'ファイルを末尾まで読み込む
279                 Dim f As Single         'ファイルから読み込む数値
280                 If Single.TryParse(sr.ReadLine(), f) Then
                                                    '変換可能な文字列なら変換して格納
281                     fList.Add(f)    '周波数の値を追加
282                 End If
283             Loop
284         End Using
285
286         If fList.Count < 2 Then
287             MessageBox.Show("読み込んだファイルのデータに不足、または異常があります。",
                            "警告", MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
288             Return False
289         End If
290
291         fList.Sort()    '周波数のリストを昇順でソート
292
293         '周波数の最小値と最大値を格納
294         fMin = fList(0)
295         fMax = fList(fList.Count - 1)
296
297         '周波数の測定範囲をチェック
298         If fMin < FreqRangeMin OrElse FreqRangeMax < fMax Then
299             MessageBox.Show("周波数が描画可能が範囲を超えています。", "警告",
                                MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
300             Return False
301         Else
```

```
302             Return True
303         End If
304
305     End Function
306
307     ''' <summary>パラメータと系列データをCSVファイルに保存するイベントです。</summary>
308     Private Sub SaveCSV_Click(ByVal sender As System.Object,
            ByVal e As System.EventArgs) Handles SaveCSV.Click, MenuFileSaveCSV.Click
309
310         Dim saveDialog As New SaveFileDialog
311         'ダイアログのタイトル
312         saveDialog.Title = "周波数特性の保存先を指定してください。"
313         'デフォルトのファイル名は"年月日_時分秒.csv"
314         saveDialog.FileName = Now.ToString("yyyyMMdd_HHmmss") & ".csv"
315         'デスクトップをデフォルトの保存先に指定
316         saveDialog.InitialDirectory =
                                    My.Computer.FileSystem.SpecialDirectories.Desktop
317         'ファイルの種類はCSVとText,すべてのファイル
318         saveDialog.Filter =
                        "CSVファイル|*.csv|テキストファイル|*.txt|すべてのファイル|*.*"
319         'ダイアログの表示がOK以外の時は以下の処理を実行しない
320         If saveDialog.ShowDialog() <> Windows.Forms.DialogResult.OK Then Exit Sub
321
322         Using sw As New StreamWriter(saveDialog.FileName)
323             'ファイルの先頭にパラメータを記述
324             sw.WriteLine("R=" & PartR.ToString())
325             sw.WriteLine("C=" & PartC.ToString())
326             '特性をファイルへ書き込む
327             For Each p As PointF In Series
328                 sw.Write(p.X.ToString())
329                 sw.Write(",")
330                 sw.WriteLine(p.Y.ToString())
331             Next
332         End Using
333
334     End Sub
335
336     ''' <summary>グラフを画像ファイルとして保存するメソッドです。</summary>
337     Private Sub SaveGraph_Click(ByVal sender As System.Object,
        ByVal e As System.EventArgs) Handles SaveGraph.Click, MenuFileSaveGraph.Click
338
339         Dim saveDialog As New SaveFileDialog
340         'ダイアログのタイトル
341         saveDialog.Title = "グラフの保存先を指定してください。"
342         'デフォルトのファイル名は"年月日_時分秒.bmp"
343         saveDialog.FileName = Now.ToString("yyyyMMdd_HHmmss") & ".bmp"
344         'デスクトップをデフォルトの保存先に指定
345         saveDialog.InitialDirectory =
                                    My.Computer.FileSystem.SpecialDirectories.Desktop
346         'ファイルの種類はBMP, JPEG, PNG
```

```
347         saveDialog.Filter = "BMPファイル(*.bmp)|*.bmp|JPEGファイル
                                 (*.jpg;*jpeg)|*.jpg;*jpeg|PNGファイル(*.png)|*.png"
348         'ダイアログの表示がOK以外の時は以下の処理を実行しない
349         If saveDialog.ShowDialog() <> Windows.Forms.DialogResult.OK Then Exit Sub
350
351         If myGraph.Export(saveDialog.FileName) = False Then
352             MessageBox.Show("保存するファイル形式、保存先が正しくありません。")
353         End If
354
355     End Sub
356
357     ''' <summary>メニューバーの「終了」を選択時に実行されるイベントです。</summary>
358     Private Sub MenuExit_Click(ByVal sender As System.Object,
                                   ByVal e As System.EventArgs) Handles MenuExit.Click
359
360         Application.Exit()  '終了
361
362     End Sub
363
364     ''' <summary>メニューバーの「回路図」をクリックしたときに実行される
                                                        イベントです。</summary>
365     Private Sub MenuCircuitView_Click(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles MenuCircuitView.Click
366
367         'ダイアログに回路図が表示
368         AboutBox.ShowDialog()
369
370     End Sub
371
372 End Class
```

解答例のソースコード(ハイパスフィルタのクラス)

行	プログラム
1	'オプション指定
2	Option Explicit On
3	Option Strict On
4	Option Infer Off
5	Option Compare Binary
6	
7	Imports System.Math
8	
9	''' <summary>抵抗とコンデンサの一次フィルタ用のクラスです。</summary>
10	''' <remarks>各種パラメータ及びプロパティはDecimal型を適用しています。</remarks>
11	Public Class HighPassFilter
12	'プロパティ及びメソッド用メンバー
13	Private rValue As Decimal '抵抗値
14	Private cValue As Decimal '静電容量
15	
16	''' <summary>ハイパスフィルタのインスタンスを生成します。</summary>

```
17      ''' <param name="r">抵抗値を指定します。</param>
18      ''' <param name="c">コンデンサの静電容量を指定します。</param>
19      ''' <remarks>インスタンス生成時に抵抗とコンデンサの値でフィルタの特性が設定されます。</remarks>
20      Public Sub New(ByVal r As Decimal, ByVal c As Decimal)
21
22          '抵抗とコンデンサの値とする。
23          rValue = r
24          cValue = c
25
26      End Sub
27
28      ''' <summary>フィルタの入力信号（電圧と周波数を指定）から
                                        電圧利得を取得するメソッドです。</summary>
29      ''' <param name="inputVoltage">入力電圧を指定します。</param>
30      ''' <param name="inputFrequency">入力周波数(Hz)を指定します。</param>
31      ''' <returns>電圧利得を返します。</returns>
32      ''' <remarks>抵抗とコンデンサの値はインスタンス生成時の値が適用されます。</remarks>
33      Public Function GetGain(ByVal inputVoltage As Decimal,
                                    ByVal inputFrequency As Decimal) As Decimal
34
35          Dim vout, gv, w, wcr As Decimal '出力電圧と角周波数ωとωCR算出用の変数
36
37          w = 2 * Convert.ToDecimal(PI) * inputFrequency      '角周波数ω算出
38          wcr = w * cValue * rValue       'ωCR算出
39          vout = Convert.ToDecimal(Sqrt(Pow(wcr, 2) / (1 + Pow(wcr, 2))) * inputVoltage)
40          gv = 20 * Convert.ToDecimal(Log10(vout / inputVoltage)) '電圧利得の算出
41          Return gv       '電圧利得を返す
42
43      End Function
44
45      ''' <summary>フィルタのカットオフ周波数を取得するメソッドです。</summary>
46      ''' <returns>カットオフ周波数を返します。</returns>
47      ''' <remarks>抵抗とコンデンサの値はインスタンス生成時の値が適用されます。</remarks>
48      Public Function GetCutoffFrequency() As Decimal
49
50          Dim fc As Decimal
51
52          fc = Convert.ToDecimal(1 / (2 * PI * rValue * cValue))
53
54          Return fc
55
56      End Function
57
58      ''' <summary>このフィルタの抵抗値を取得します。</summary>
59      ''' <returns>抵抗値を単位[Ω]で返します。</returns>
60      ''' <remarks>このプロパティは確認用です。</remarks>
61      Public ReadOnly Property Rohm() As Decimal
62          Get
63              Return rValue
```

```
64          End Get
65      End Property
66
67      ''' <summary>このフィルタの静電容量を取得します。</summary>
68      ''' <returns>静電容量を単位[F]で返します。</returns>
69      ''' <remarks>このプロパティは確認用です。</remarks>
70      Public ReadOnly Property Cfarad() As Decimal
71          Get
72              Return cValue
73          End Get
74      End Property
75
76  End Class
```

作業工程計画書（受講者配付用）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備		
1. 仕様確認と作成する機能		
2. GUI 設計		
3. イベントの割り付け		
4. コーディング		
5. 動作確認		

作業工程計画書（受講者配付用）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備		
1. 仕様確認と作成する機能		
2. GUI 設計		
3. イベントの割り付け		
4. コーディング		
5. 動作確認		

作業工程計画書（受講者配付用例）

作業工程計画書（模範解答例）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備	作業場所の確認 実習装置と開発環境の確認	
1. 仕様確認と作成する機能	入力パラメータ ・抵抗値 ・コンデンサ 表示機能 ・周波数特性の描画 ・カットオフ周波数の表示 ・理論値と実測値の比較 ファイル処理 ・画像保存	
2. GUI 設計	入力パラメータ ・NumericUpdown で数値指定 ・ListBox で単位指定 表示機能 ・Button で動作開始とする。 ・PictureBox にグラフを描画（Graph クラスを使用） ・カットオフ周波数、比較結果等は Label を使用 ファイル処理 ・SaveFIleDialog クラスで名前を付けて保存	
3. イベントの割り付け	GUI のコントロールにイベントを割り付ける Form1：Click イベント SaveCSV：Click イベント SaveGraph：Click イベント InputNumberR：ValueChanged イベント InputNumberC：ValueChanged イベント ListPrefixR：SelectedValueChanged イベント ListPrefixC：SelectedValueChanged イベント	
4. コーディング	各イベントに処理を記述する。 プロシージャの実装を行なう。	
5. 動作確認	各機能の実装について仕様を満たしているか確認する。	

訓 練 課 題 確 認 シ ー ト

訓練科名　：
仕上がり像　：
システム名　：
訓練課題名　：　パソコンを用いた計測制御システムの製作

入所期　：
氏　名　：

評価区分	評価項目	細　目	評価（数値）					評価判定	評価基準
作業時間	作業時間	作業時間内に作業を完了	2	4	6	8	10	10	既定時間以内に作業が完了していれば10点、以下指定時間経過する度に2点ずつ減点する。
	資料提出	ソースファイル等の提出	1	2	3	4	5	5	必要提出物が全て提出されていれば5点、提出物が既定時間に提出されていなければ指定時間を経過する度に1減点。
作業工程	作業工程の確認	作業工程計画	1	2	3	4	5	5	作業のポイントが不適切な場合、1か所につき1点減点。
必須機能	パラメータの指定	抵抗値	1		3		5	5	抵抗値の入力が適切にできれば5点、エラー等が発生する要因があれば3点、入力が実装されていなければ1点とする。
		コンデンサ	1		3		5	5	コンデンサの静電容量の入力が適切にできれば5点、エラー等が発生する要因があれば3点、入力が実装されていなければ1点とする。
	演算	電圧利得とカットオフ周波数	1		3		5	5	電圧利得及びカットオフ周波数が正しく計算できていれば5点、不適切な箇所があれば3点、演算ができていなければ1点
	グラフ描画	横軸と縦軸	1		3		5	5	横軸が対数軸で表示されていれば2点加点、縦軸が線形軸で電圧利得（デシベル）表示されていれば2点加点、実装されていなければ1点とする。
		測定結果の表示	2	4	6	8	10	10	計測結果をグラフで正しく表示できていれば10点。表示等に不具合がある場合は1箇所につき2点ずつ減点する。
	ファイル処理	ファイルの読み込み	1		3		5	5	入力値のCSVもしくはテキストファイルの値を正しく読み込んでいれば5点、不具合があれば3点、読み込みができない場合は1点。
		ファイルの保存	1		3		5	5	計測で出力される測定値をCSVもしくはテキストファイルに値を正しく保存ができていれば5点、不具合があれば3点、保存ができない場合は1
ユーザビリティ	GUI	フォームのデザイン、オペレーション	1	2	3	4	5	5	フォームのデザインやオペレーションに関して、不具合や欠陥等があれば1箇所につき1点減点する。
	グラフの可読性	目盛線、数値、配色等	1	2	3	4	5	5	グラフの可読性について、目盛の間隔や線種、数値、配色等を妨げるものでなければ5点、可読性を低下させる要素があれば1箇所につき1点減点する。
ドキュメント	可読性	ソースコード中のコメントやその他のドキュメント	2		6		10	10	ソースコード中に適切なコメントが記述されていれば4点加点、アプリケーションの操作方法等を記載したドキュメントがあれば4点間。なければ2点とする。
安全作業	機材の取り扱い	機材の取り扱い、確認作業等	1	2	3	4	5	5	機材の取り扱い、及び動作確認手順等が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。
	VDT作業	座る姿勢、ディスプレイ、休憩時間	1	2	3	4	5	5	椅子の高さ調整、モニタの角度を適正に行っていない場合、1箇所につき2点減点1時間ごとに休憩を取らない場合、1回につき2点減点、無理な姿勢で作業をしている場合0点。
工夫・改善	追加・工夫等	課題で提示した以外の要素	2	4	6	8	10	10	課題で提示した以外の機能が1箇所ある度に2点加点。ただし機能の内容によっては2点以上の加点をしても良い。
	工夫・改善点記入欄		総点					100	＜判定表＞ A　：　80点以上　：到達水準を十分に上回った B　：　60点以上80点未満　：到達水準に達した C　：　60点未満　：到達水準に達しなかった
			合計点					100	
			換算点					100.0	
			総合評価判定					A	

訓練課題のねらい	コメント
	担当指導員氏名：

評価要領

訓練科名　：
仕上がり像　：
システム名　：
訓練課題名　：　パソコンを用いた計測制御システムの製作

評価区分	評価項目	細　目	評価要領（採点要領）	備考
作業時間	作業時間	作業時間内に作業を完了	既定時間以内に作業が完了していれば10点、以下指定時間経過する度に2点ずつ減点する。	
	資料提出	ソースファイル等の提出	必要提出物が全て提出されていれば5点、提出物が既定時間に提出されていなければ指定時間を経過する度に1減点。	
作業工程	作業工程の確認	作業工程計画	作業のポイントが不適切な場合、1か所につき1点減点。	
必須機能	パラメータの指定	抵抗値	抵抗値の入力が適切にできれば5点、エラー等が発生する要因があれば3点、入力が実装されていなければ1点とする。	
		コンデンサ	コンデンサの静電容量の入力が適切にできれば5点、エラー等が発生する要因があれば3点、入力が実装されていなければ1点とする。	
	演算	電圧利得とカットオフ周波数	電圧利得及びカットオフ周波数が正しく計算できていれば5点、不適切な箇所があれば3点、演算ができていなければ1点	
	グラフ描画	横軸と縦軸	横軸が対数軸で表示されていれば2点加点、縦軸が線形軸で電圧利得（デシベル）表示されていれば2点加点、実装されていなければ1点とする。	
		測定結果の表示	計測結果をグラフで正しく表示できていれば10点。表示等に不具合がある場合は1箇所につき2点ずつ減点する。	
	計測	カットオフ周波数の表示	カットオフ周波数が描画したグラフ上に描画されていれば5点、不具合があれば3点、表示されていなければ1点。	
	ファイル処理	ファイルの読み込み	入力値のCSVもしくはテキストファイルの値を正しく読み込んでいれば5点、不具合があれば3点、読み込みができない場合は1点。	
		ファイルの保存	計測で出力される測定値をCSVもしくはテキストファイルに値を正しく保存ができていれば5点、不具合があれば3点、保存ができない場合は1点。	
ユーザビリティ	GUI	フォームのデザイン、オペレーション	フォームのデザインやオペレーションに関して、不具合や欠陥等があれば1箇所につき1点減点する。	
	グラフの可読性	目盛線、数値、配色等	グラフの可読性について、目盛の間隔や線種、数値、配色等を妨げるものでなければ5点、可読性を低下させる要素があれば1箇所につき1点減点する。	
ドキュメント	可読性	ソースコード中のコメントやその他のドキュメント	ソースコード中に適切なコメントが記述されていれば4点加点、アプリケーションの操作方法等を記載したドキュメントがあれば4点間。なければ2点とする。	
工夫・改善	機材の取り扱い	機材の取り扱い、確認作業等	機材の取り扱い、及び動作確認手順等が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	
	VDT作業	座る姿勢、ディスプレイ、休憩時間	椅子の高さ調整、モニタの角度を適正に行っていない場合、1箇所につき2点減点1時間ごとに休憩を取らない場合、1回につき2点減点、無理な姿勢で作業をしている場合0点。	
工夫・改善	追加・工夫等	課題で提示した以外の要素	課題で提示した以外の機能が1箇所ある度に2点加点。ただし機能の内容によっては2点以上の加点をしても良い。	

実技課題

管理番号：Ｅ－45B
「パソコンを用いた計測制御システムの製作（Visual C#）」

■課題概要■

パソコン計測のために必要となるプログラミング言語（Visual C#）の要素と、制御機器、計測機器に関連するプログラミングについて実技試験により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領	○	E-45B-00_実施要領
訓練課題	○	E-45B-01_訓練課題
解答	○	E-45B-02_解答及び解説
作業工程手順書	○	E-45B-03_作業工程計画書【作成例】
訓練課題確認シート	○	E-45B-04_訓練課題確認シート及び評価要領
評価要領	○	E-45B-04_訓練課題確認シート及び評価要領

※「解答例」、「Help」、「Library」のフォルダ及び関連のデータがあります。

※プログラミング言語が異なるAとBの2タイプの課題があります。

実技課題　実施要領

訓練課題名「パソコンを用いた計測制御システムの製作（Visual C#）」

- 作業準備（ハードウェア製作含む）及び使用機器の準備時間は別途確保した上で事前に行うこととし、作業時間に含まない。
- 作業時間は、休憩時間を除いた時間とする。
- 課題は 1 名で行なう。
- VDT 作業を考慮し、1 時間ごとに休憩時間を設ける。
- この課題は、パソコン用の開発言語（Visual C#）を使用し、計測制御を行なうシステムを想定しているが、機器等の状況によってはパソコンのみで簡潔する課題でも良いものとする。
- 計測・制御対象については、施設内で実施している訓練内容や機器に合わせ課題を修正した上で実施することが望ましい。

タイムスケジュール例（訓練時間が 9:00～15:30 の場合）を以下に示す。

時間	実施内容
9：00～9：10	出欠確認
9：10～9：30	課題内容説明および質問
課題作成開始	
9：30～12：00	プログラム作成
12：00～13：00	昼食
13：00～14：00	課題作成終了（作成課題提出）
14：00～15：30	動作確認及び評価
課題作成終了（作成課題提出）	

実技課題

「パソコンを用いた計測制御システムの製作（Visual C#)」

１　作業時間

240 分(4 時間:休憩を除く)

２　配付資料

問題用紙, 解答用紙

３　課題作成、提出方法

・各個人で作業すること

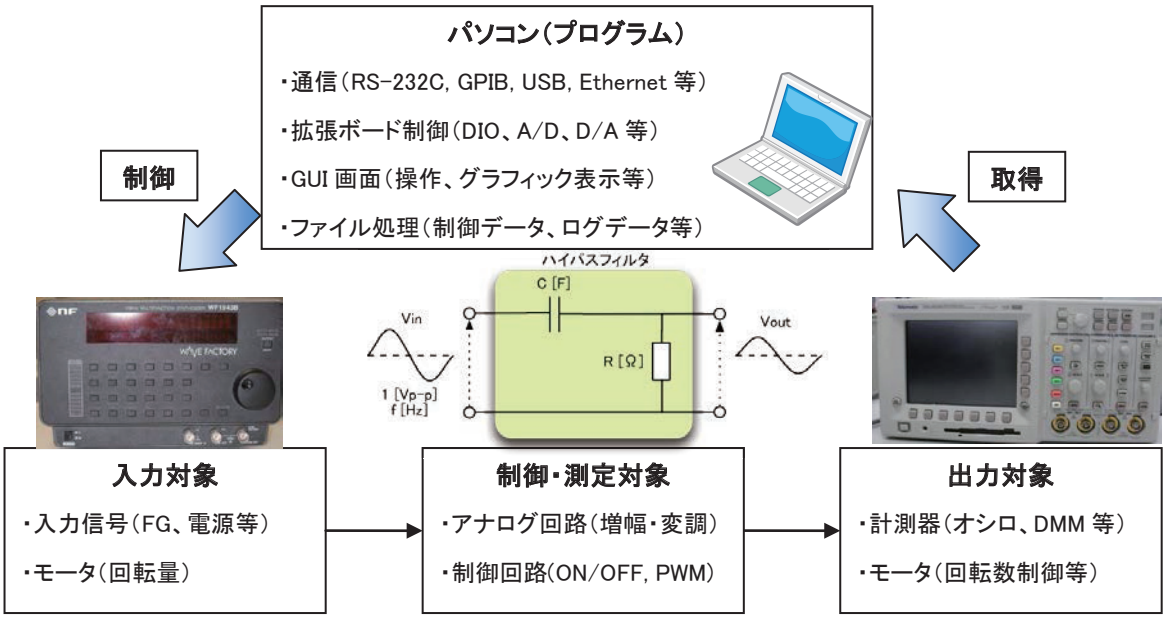
・ソースファイルによる提出

１．課題名：パソコンを用いた計測制御システム製作

２．課題内容（イメージ）

以下の仕様に基づいた計測制御システムのプログラムを作成しなさい。

システムの使用はパソコンから入力対象となる機器へ制御データを入力し、制御対象から得られたデータを出力対象となる機器からパソコンに出力し、その制御内容を GUI で管理する。ただし実習環境によってはパソコンのみで行なって良いものとする。



① 各機器の電源を投入し、プログラムを起動する。

② プログラム起動後、「開始」のボタンをクリックすると入力対象からデータを取得する。

A) 対象がオシロの場合、描画中の波形のデータを RS-232C や GPIB といった計測向けのインタフェース（USB や Ethernet も可）で取得する。

B) 計測対象の状態を ON/OFF 信号（回転数等）、あるいはアナログ信号取得する。

C) 機材の都合上、入力対象機器が用意できない場合は、入力機器の信号を想定したログ等を CSV ファイル形式で用意する。

③ データを取得後、目的の制御に応じた結果を出力対象に反映させる。

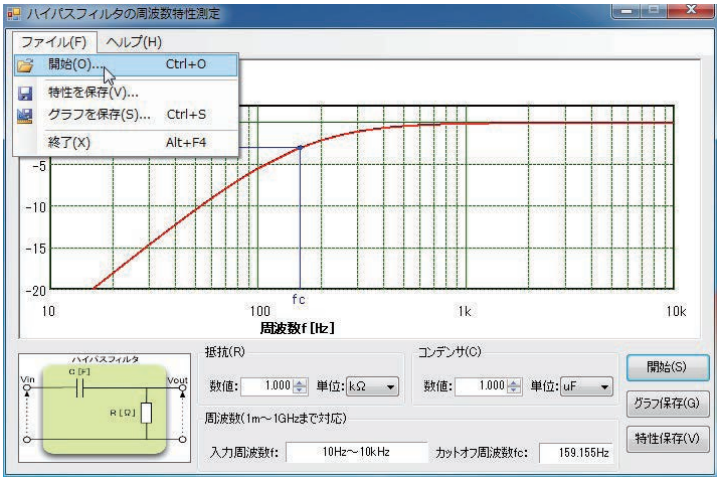
A) 対象が FG の場合、アプリケーションで指定した波形を FG に出力させる信号を送信する。

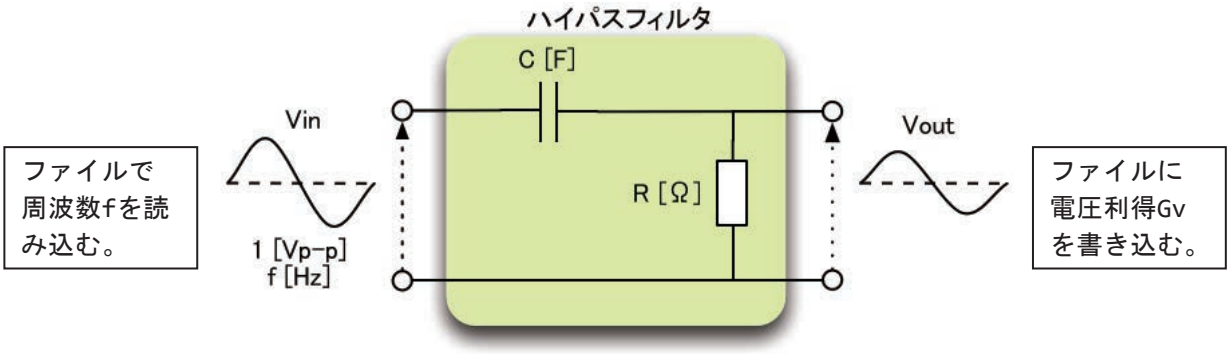
B) 入力対象の信号から演算した結果を出力対象へ出力する。A/D、D/A 変換をしている場合はディジタル値とアナログ信号の換算を行なうこと。

C) 機材の都合上、入力対象機器が用意できない場合は、入力された信号（もしくはファイル）を時系列もしくは特性等（例：周波数等）でまとめたものをグラフで描画し、グラフの保存機能を追加する。

仕様例（フィルタ回路のカットオフ周波数 fc の評価）

下記の回路図は抵抗Rとコンデンサcによって構成されるハイパスフィルタの回路です。この回路の抵抗値 R とコンデンサ C を設定し、フィルタの性能を表す周波数特性の表示と及びカットオフ周波数の算出を行なうプログラムを作成します。

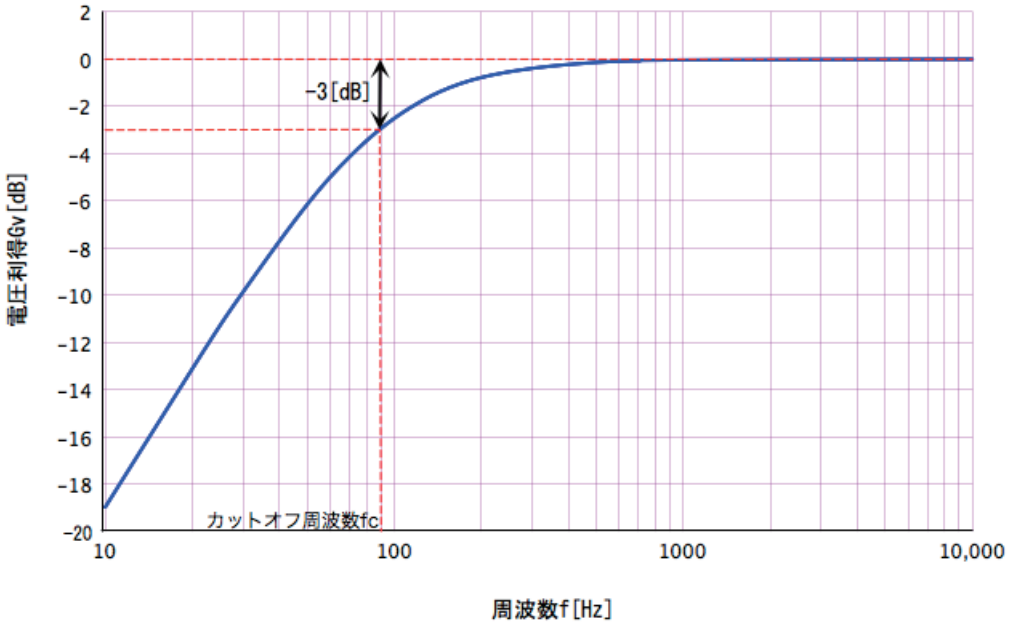




ファイルで周波数fを読み込む。

ファイルに電圧利得Gvを書き込む。

電圧利得： $Gv = 20\log_{10}\left(\frac{Vout}{Vin}\right)$



制御対象回路（ハイパスフィルタ）の電圧利得の周波数（f-Gv）特性

※（X 軸は周波数を対数軸、Y 軸は電圧利得を線形軸で描画）

この回路に 1[Vp-p](固定)の周波数 f[Hz](可変)の電圧 Vin を入力し、出力電圧 Vout から周波数特性とカットオフ周波数 fc（電圧利得 Gv[dB]が-3[dB]低下した周波数）を算出し描画するプログラムを作成しなさい。作成条件は以下のとおりとします。

1. パソコン上で抵抗 R とコンデンサ C の値を指定し、「開始」のボタンをクリックすると周波数特性を測定する。測定する周波数の範囲は右図のようなテキストファイルを選択し読み込むものとする。

   なおカットオフ周波数 fc の理論値は指定された抵抗 R とコンデンサ C を用いて以下の式から算出できる。

$$f_C = \frac{1}{2\pi CR}$$

```
10
20
30
50
70
100
200
300
500
700
1000
2000
…
```

2. フィルタの電圧利得 Gv[dB]は以下の式を用いて算出する。

$$Gv = 20\log_{10}\left(\frac{Vout}{Vin}\right) \qquad \frac{Vout}{Vin} = \sqrt{\frac{(\omega CR)^2}{1+(\omega CR)^2}} \qquad \omega = 2\pi f$$

3. 周波数 f を変化させ、周波数特性を表すグラフを描画する。グラフの X 軸は周波数[Hz]（対数軸）、Y 軸は電圧利得[dB]とする。グラフの描画は、提供する Graph クラスを使用しても良いものとする。

4. 周波数特性からカットオフ周波の測定値 fc を求め表示する。

5. 「保存」のボタンをクリックすると、周波数特性のグラフを「名前を付けて保存」のダイアログを表示して画像ファイル(BMP、JPEG、PNG 等のパソコンで表示可能なファイル形式)を任意の場所に保存できるようにすること。

6. 独自の機能を追加してもよい。以下は例である。

7. アプリケーションの操作方法や動作についてテキストファイル等で第 3 者が理解できるようなドキュメントを作成すること（簡易的なもので可）。独自の機能を追加し合は追加した機能をドキュメントに記載すること。

３．作業時間：240 分（4 時間）

・機材の準備時間等は含まないものとする。

・これまでの訓練で使用したソースファイル等を流用して良いものとする。

４．課題仕様

（１）課題に必要な機器を用意する。実機が用意できない場合はプログラムのみの課題でも良いものとする。

機器	実機による例	プログラムのみによる例 （本資料に記載）
パソコン(PC)	Windows XP/Vista/7/8	Windows 7
開発環境	Visual Studio 2005/2008/2010/2012	Visual Studio 2005 以降
入力・出力 I/F	インタフェース社　PCI-3522A RS-232C/USB/Ethernet/GP-IB	なし
入力機器	ファンクションジェネレータ	CSV ファイルの読み込み
出力機器	オシロスコープ	CSV ファイルの書き込み
制御・負荷回路	フィルタ回路、モータ制御回路	フィルタ回路の計算

（２）動作仕様

① 各種機器の接続（電源、入出力信号、通信ケーブル等）を行なう。

② 機器の電源を投入し、動作可能な状態とする。パソコン側は作成したアプリケーションを起動する。

③ アプリケーションにパラメータ（測定時間、測定用数値）を入力し「開始」のボタンをクリックすると計測を開始する。

④ 計測を開始し制御機器の特性と、その結果を表示する。

グラフ描画には別途配布する DLL を開発環境に組み込んで作成すること。また使用例については後述のソースファイルを参照すること。

（３）作業内容

1) Visual Studio を起動しプロジェクトを Windows フォームアプリケーションで作成する。言語の仕様やバージョン等については講師の指示に従うこと。

2) 提出物

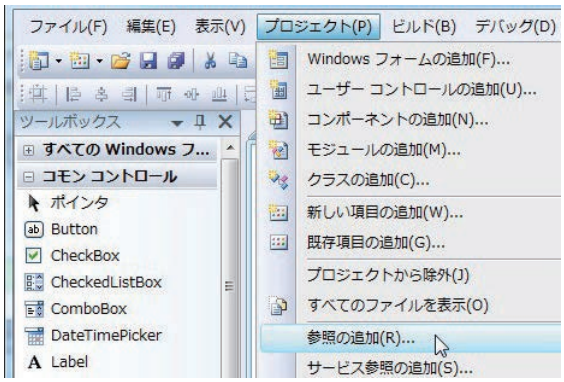
	機能	備考
1	作成したプログラムのファイル一式	
2	プログラムのドキュメント（操作手順、動作、追加機能等を記載したもの）	テキストファイル （形式は講師で指示）

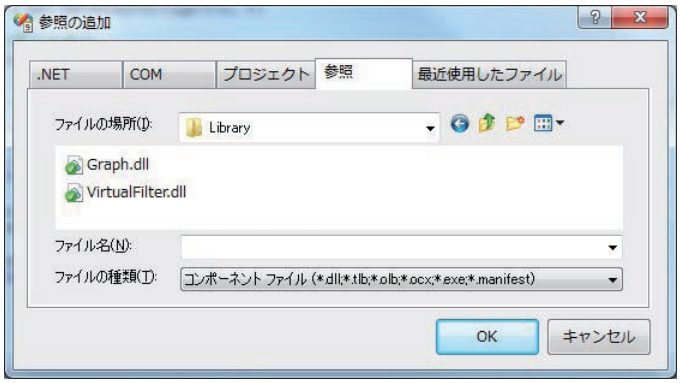
５．参考資料

参考として、グラフを描画する Graph クラスのサンプルを以下に示す。

（１）Graph クラスの使用方法(GraphSample.zip に格納)

1) プロジェクト作成後、グラフ描画用の DLL(Graph.dll)をメニューバーの「プロジェクト」→「参照の追加」から以下のダイアログを表示し、「参照」のタブで、配布した"Graph.dll"を選択する。





2) "Polytech.Windows.Form"名前空間で Graph クラスを定義しているため、Imports ステートメントで"Polytech.Windows.Form"名前空間を追加すること。

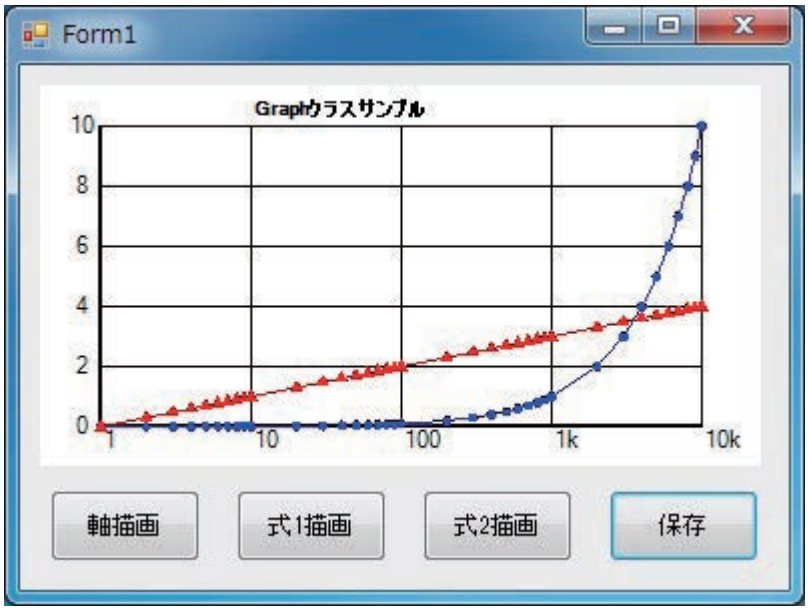
```
'独自の名前空間
Using Polytech.Windows.Form;    'Graphクラス
'既定の名前空間
Using System.Text;
```

3) 例として以下の式を x が対数軸で描画するプログラムのサンプルを示す。x の範囲は 1～10000 とする。

コントロール	オブジェクト名	備考
Form	Form1	Graphクラスの初期化と軸の描画、初期描画の登録
Button	GraphInit	初期描画状態を呼び出す
	DrawLine1	式1のグラフを描画
	DrawLine2	式2のグラフを描画
	SaveGraph	グラフを画像ファイルとして保存
PictureBox	GraphDisplay	グラフの描画対象

$$y = \log_{10} x \quad \cdots\cdots \text{（式 1）}$$

$$y = \frac{x}{1000} \quad \cdots\cdots \text{（式 2）}$$



Graph クラスを用いたグラフの描画サンプル

行	プログラム
1	`using System;`
2	`using System.Collections.Generic;`
3	`using System.ComponentModel;`
4	`using System.Data;`
5	`using System.Drawing;`
6	`using System.Linq;`
7	`using System.Text;`
8	`using System.Windows.Forms;`
9	`using System.Drawing.Drawing2D;`
10	`using Polytech.Windows.Form;`
11	
12	`namespace GraphSample`
13	`{`
14	`    public partial class Form1 : Form`
15	`    {`
16	
17	`        private Graph myGraph;`
18	
19	`        public Form1()`
20	`        {`
21	`            InitializeComponent();`
22	`        }`
23	
24	`        private void Form1_Load(object sender, EventArgs e)`
25	`        {`
26	`            float x, y;//数式用`
27	`            int i;  //ループ変数`
28	
29	`            //PictureBox1に描画座標を左下(1,0)、右上(10000, 10), 左右の余白20pixel,`

```
30              //上下の余白30pixel，X軸を対数軸，Y軸を線形軸でグラフを描画
31              myGraph = new Graph(GraphDisplay, 1, 0, 10000, 10, 30, 20,
                                                        Graph.AxisStyle.XlogYlinear);
32
33              myGraph.BackColor = Color.White;    //背景色を白
34
35              //フォントの設定(フォント:ＭＳ ゴシック",サイズ:8pt,スタイル:太字,色:黒)
36              myGraph.FontName = "ＭＳ ゴシック";
37              myGraph.FontSize = 8;
38              myGraph.FontStyle = FontStyle.Bold;
39              myGraph.FontColor = Color.Black;
40
41              //グラフのタイトルを(10,11)へ描画
42              myGraph.DrawText(10, 11, "Graphクラスサンプル");
43
44              //フォント一括設定(フォント:ＭＳ ゴシック",サイズ:7pt,スタイル:標準,色:青)
45              myGraph.FontSet("MS ゴシック", 7, FontStyle.Regular, Color.Blue);
46
47              //線のスタイルを実線、太さを2、色を黒
48              myGraph.LineStyle = DashStyle.Solid;
49              myGraph.LineWidth = 2;
50              myGraph.LineColor = Color.Black;
51
52              //X軸とY軸を描画
53              myGraph.DrawLine(1, 0, 10000, 0);    //(1,0)から(10000,0)へ直線描画
54              myGraph.DrawLine(1, 0, 1, 10);       //(1,0)から(1,10)へ直線描画
55
56              //線のスタイルを破線、太さを1、色を緑
57              myGraph.LineStyle = DashStyle.Solid;
58              myGraph.LineWidth = 1;
59              myGraph.LineColor = Color.Green;
60
61              //x軸の目盛線と目盛の数値の描画
62              for (i = 0; i < 5; i++) {
63                  x = (float)Math.Pow(10 , i);
64                  myGraph.DrawLine(x, 0, x, 10);
65                  if ((3 <= i) && (i < 6)) {   //10の3乗以上6未満で補助単位k(キロ)
66                      myGraph.DrawText(x, 0, (x / 1000).ToString("###k"));
67                  }
68                  else {    //上記以外のとき補助単位なし
69                      myGraph.DrawText(x, 0, x.ToString());
70                  }
71              }
72
73              //y軸の目盛線と目盛の数値の描画
74              for (i = 0;i <= 10; i += 2) {
75                  y = i;
76                  myGraph.DrawLine(1, y, 10000, y);
77                  myGraph.DrawText(0.6f, y + 0.5f, y.ToString().PadLeft(4));
78              }
```

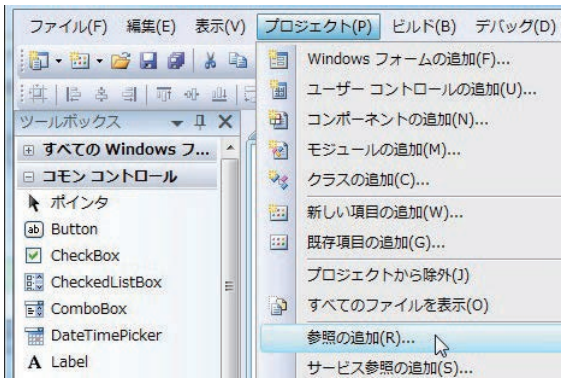
```
79              //描画したイメージを記憶する
80              myGraph.SaveInitialGraph();
81          }
82
83          private void GraphInit_Click(object sender, EventArgs e)
84          {
85              myGraph.LoadInitialGraph();  //初期
86          }
87
88          private void DrawLine1_Click(object sender, EventArgs e)
89          {
90              float x, y; //数式用
91              int i, j, k; //ループ変数
92              PointF[] series = new PointF[37];       //座標格納用用配列
93
94              //1～10000まで対数目盛間隔で配列に代入
95              //1,2,3,...,9,10,20,30,...,90,100,200
96              k = 0;
97              for (i = 0; i < 5; i++) {               //10のi乗
98                  for(j = 1; j < 10; j++) {           //1～9の値
99
100                     y = x / 1000;                   //yの値
101                     series[k] = new PointF(x, y);   //配列に格納
102                     k = k + 1;                      //次の配列の要素番号
103                     if( x >= 10000) break; //10000まで格納後このステートメントを抜ける
104                 }
105
106             //線のスタイルを実線、太さを1、色を青
107             myGraph.LineStyle = DashStyle.Solid;
108             myGraph.LineWidth = 1;
109             myGraph.LineColor = Color.Blue;
110
111             //マーカーのスタイルを円、幅を5px、色を青
112             myGraph.MarkerStyle = Graph.MarkerShapeStyle.Circle;
113             myGraph.MarkerWidth = 5;
114             myGraph.MarkerColor = Color.Blue;
115
116             //最初の点のマーカーを描画
117             myGraph.DrawMarker(series[0]);
118
119             //配列内の座標の2点間を直線で描画、終点にはマーカー描画
120             for (i = 0; i < series.Length - 1; i ++) {
121                 myGraph.DrawMarker(series[i + 1]);
122                 myGraph.DrawLine(series[i], series[i + 1]);
123             }
124         }
125
126         private void DrawLine2_Click(object sender, EventArgs e)
127         {
128             PointF formula = new PointF();     //数式用
```

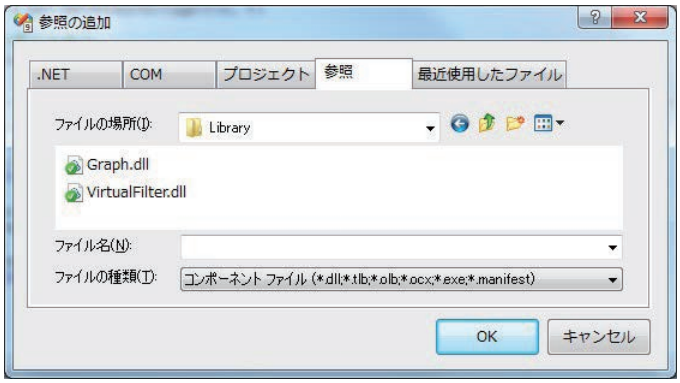
```
129             int i, j;           //ループ変数
130             List<PointF> series = new List<PointF>();
                                        //PointF構造体でListクラスのインスタンス生成
131
132             //1～10000まで対数目盛間隔
                                (1,2,3,...,9,10,20,30,...,90,100,200...)でリストへ追加
133             for (i = 0; i < 5; i++) {         //10のi乗
134                 for (j = 1; j < 10; j++) {    //1～9の値
135                     formula.X = (float)(j * Math.Pow(10, i));       //xの値
136                     formula.Y = (float)(Math.Log10(formula.X));       //yの値
137                     series.Add(formula);    //座標を追加
138                     if (formula.X >= 10000) break;
                                            //10000まで格納後このステートメントを抜ける
139                 }
140             }
141
142             //線のスタイルを実線、太さを1、色を赤
143             myGraph.LineSet(DashStyle.Solid, 1, Color.Red);
144
145             //マーカーの形状を三角、幅を7px、色を赤
146             myGraph.MarkerSet(Graph.MarkerShapeStyle.Triangle, 7, Color.Red);
147
148             //配列内の座標の2点間を直線で描画、終点にはマーカー描画
149             for (i = 0; i < series.Count; i++)
150             {
151                 if (i == 0)
152                 {      //最初の点のマーカーのみ描画
153                     myGraph.DrawMarker(series[0]);
154                 }
155                 else
156                 {   //以降は前に描画した座標を始点として直線を描画
157                     myGraph.DrawLine(series[i]);
158                     myGraph.DrawMarker(series[i]);
159                 }
160             }
161         }
162
163         private void SaveGraph_Click(object sender, EventArgs e)
164         {
165             String filePath;        //画像ファイルの保存先のパス
166
167             filePath = "C:¥¥Temp¥¥Graph.jpg";  //保存先の代入
                                            (画像形式はBMP, JPEG, GIF. PNG, TIFF等)
168
169             //Exportメソッドで描画しているPictureBoxのグラフィックを保存
170             if (myGraph.Export(filePath)) {
171                     //保存成功時はTrue
172                 MessageBox.Show(filePath + "に保存しました。", "完了",
MessageBoxButtons.OK, MessageBoxIcon.Information);
173             }
```

```
174             else {
175                 //失敗時はFalse
176                 MessageBox.Show(filePath + "に保存できません。", "失敗",
                                MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
177             }
178         }
179     }
180 }
```

（１）HighPassFilter クラスの使用方法(FilterSample.zip に格納)

1) プロジェクト作成後、グラフ描画用の DLL(VirtualFilter.dll)をメニューバーの「プロジェクト」→「参照の追加」から以下のダイアログを表示し、「参照」のタブで、配布した"VirtualFilter.dll"を選択する。



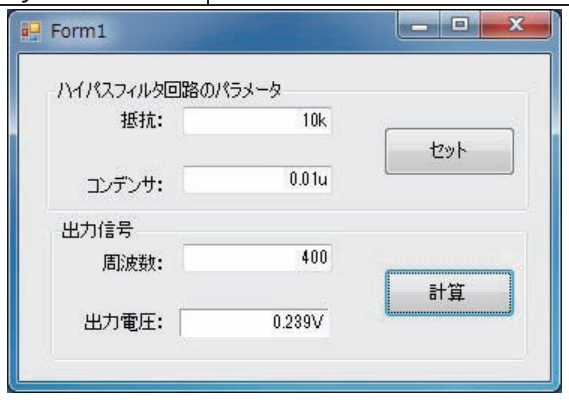


2) 名前空間"Polytech.Circuit"でHighPassFilter クラスを定義しているため、Imports ステートメントで"Polytech.Circuit"名前空間を追加すること。

```
'名前空間のインポート
'独自の名前空間
Imports Polytech.Windows.Form          'Graphクラス
Imports Polytech.Circuit               'HighPassFilterクラス
'既定の名前空間
Imports System
Imports System.Text
```

3) 例として仮想ハイパスフィルタ（HighPassFilter クラス）を用いて抵抗 R、コンデンサ C、周波数 f、入力電圧 vin=1V のときの出力電圧 vout を算出するプログラムを作成する。

コントロール	オブジェクト名	備考
Form	Form1	スタートアップフォーム
TextBox	InputResistance	抵抗値を入力(例:10[kΩ]なら"10k")
	InputCapacitance	静電容量を入力(例:0.1[uF]なら"0.1u")
	InputFrequency	周波数を入力(例:100[Hz]なら"100")
Button	CalculateVout	ハイパスフィルタの出力電圧を算出
Label	DisplayVout	出力電圧Voutを表示



HighPassFilter クラスを用いたグラフの描画サンプル

行	プログラム
1	`Imports Polytech.Circuit    'HighPassFilterクラス`
2	
3	`Public Class Form1`
4	
5	`    'ハイパスフィルタのオブジェクト`
6	`    Private hpf As HighPassFilter`
7	
8	`    ''' <summary>抵抗Rとコンデンサcでフィルタ回路を生成する</summary>`
9	`    Private Sub SetParameter_Click(ByVal sender As System.Object,` `                           ByVal e As System.EventArgs) Handles SetParameter.Click`
10	
11	`        Dim r, c As Decimal  '抵抗、コンデンサ`
12	`        'テキストボックスに入力された文字列を数値に変換して代入`
13	`        r = SIPrefixToDecimal(InputResistance.Text)     '抵抗値[Ω]`
14	`        c = SIPrefixToDecimal(InputCapacitance.Text)    'コンデンサ[F]`
15	
16	`        'ハイパスフィルタのインスタンス生成(R, C)`
17	`        'インスタンスを生成すると内部で許容差を含めた値の回路になる`
18	`        hpf = New HighPassFilter(r, c)`
19	
20	`    End Sub`
21	
22	`    ''' <summary>出力電圧を算出して表示</summary>`
23	`    Private Sub CalculateVout_Click(ByVal sender As System.Object,` `                           ByVal e As System.EventArgs) Handles CalculateVout.Click`
24	
25	`        Dim vin, vout, f As Decimal '入力電圧、出力電圧、周波数`
26	
27	`        'テキストボックスに入力された文字列を数値に変換して代入`
28	`        f = SIPrefixToDecimal(InputFrequency.Text)      '周波数[Hz]`
29	`        vin = 1     '入力電圧を1V`
30	`        'ハイパスフィルタに電圧vinと周波数fを入力して出力電圧を取得`
31	`        vout = hpf.GetOutputVoltage(vin, f)`
32	
33	`        '出力電圧を表示`
34	`        DisplayVout.Text = vout.ToString("0.###V")`
35	
36	`    End Sub`
37	
38	`    ''' <summary>SI接頭辞が含まれる文字列を数値に換算するメソッド</summary>`
39	`    ''' <param name="numberWithSIPrefix">SI接頭辞が末尾にある数値の文字列</param>`
40	`    ''' <returns>SI接頭辞がない数値</returns>`
41	`    Private Function SIPrefixToDecimal(ByVal numberWithSIPrefix As String)` `                                                    As Decimal`
42	
43	`        Dim siPrefix As Char    '末尾の文字`

```
44          Dim number As Decimal   'SI接頭辞がない数値
45          '末尾の1文字を取得
46          siPrefix = numberWithSIPrefix.Chars(numberWithSIPrefix.Length - 1)
47
48          '末尾の1文字を評価
49          If Char.IsDigit(siPrefix) = True Then    '末尾が数値のとき
50              number = Convert.ToDecimal(numberWithSIPrefix)
51          Else
52              number = Convert.ToDecimal(numberWithSIPrefix.TrimEnd(siPrefix))
53              Select Case siPrefix
54                  Case "u"
55                      number = number * (10 ^ -6) '"uを換算"
56                  Case "m"
57                      number = number * (10 ^ -3) '"mを換算"
58                  Case "k"
59                      number = number * (10 ^ 3)  '"kを換算"
60                  Case Else   'それ以外の文字は例外とする
61                      Throw New FormatException("入力された文字が適切ではありません。")
62              End Select
63          End If
64
65          Return number   'SI接頭辞がない数値を渡して復帰
66
67      End Function
68
69  End Class
```

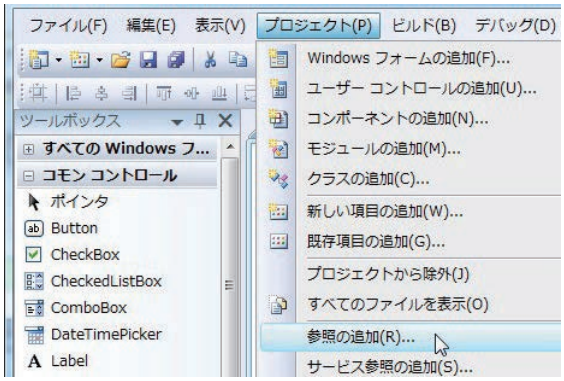
実技課題　解答及び解説

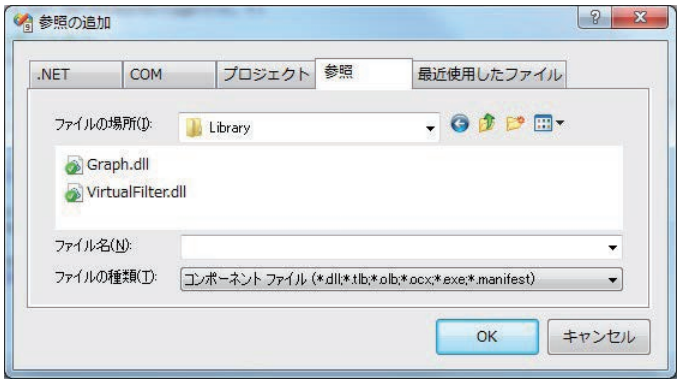
「パソコンを用いた計測制御システムの製作（Visual C#）」

解答例を以下に示す。

（１）事前の設定(別添のファイルに格納)

1) プロジェクトを開いたあと、グラフ描画用の DLL(Graph.dll)をメニューバーの「プロジェクト」→「参照の追加」から以下のダイアログを表示し、「参照」のタブで、配布した"Graph.dll"を選択する。



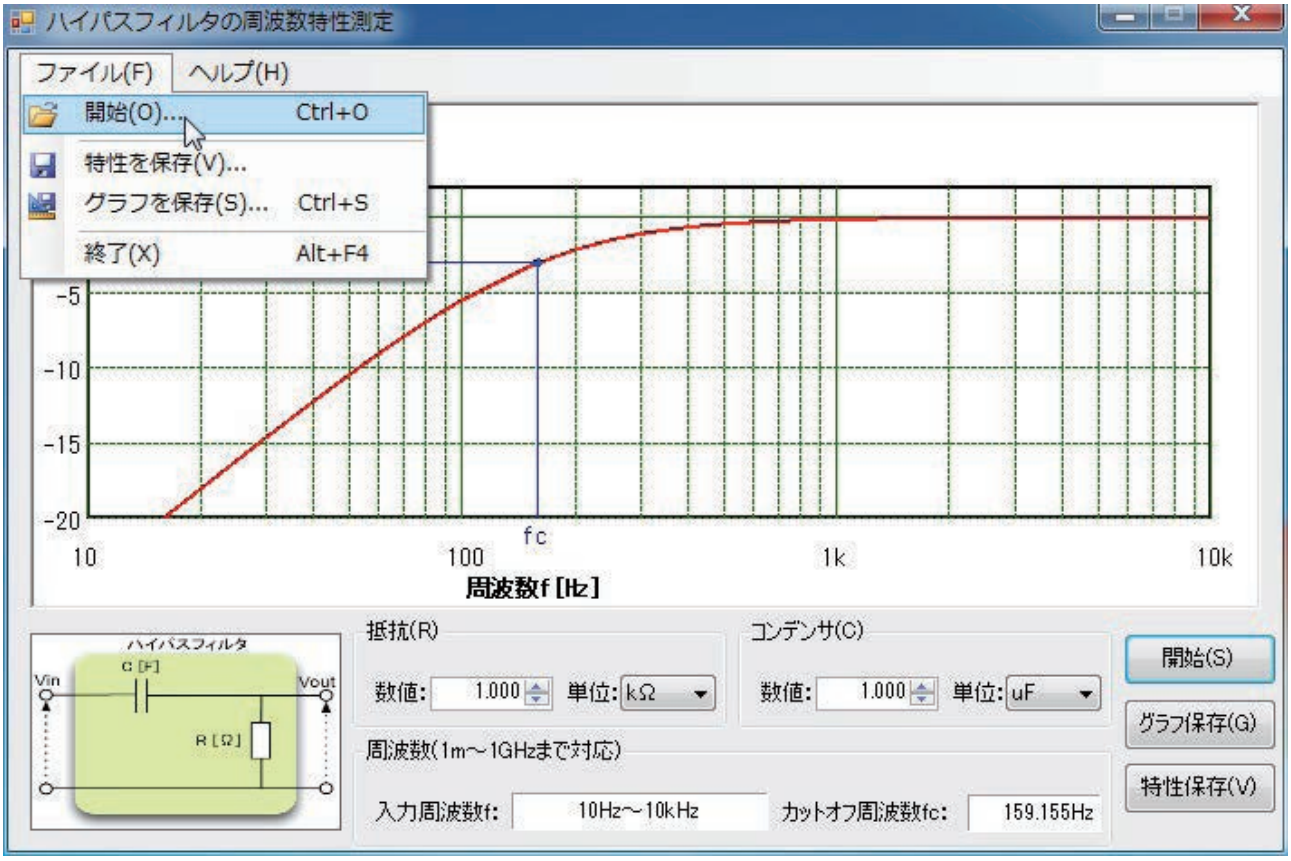


2) "Polytech.Windows.Form"名前空間で Graph クラスを定義しているため、Using ステートメントで"Polytech.Windows.Form"名前空間を追加すること。

```
using System;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Drawing.Imaging;
using Polytech.Windows.Form;    //Graphクラス
using Filter;                   //HighPassFilterクラス
```

3) コントロールと各機能

コントロール	オブジェクト名	備考
Form	Form1	Graphクラスの初期化と軸の描画、初期描画の登録
Button	GraphInit	初期描画状態を呼び出す
	SaveCSV	周波数特性をCSVファイル等で保存
	SaveGraph	グラフを画像ファイルとして保存
NumericUpdown	InputNumberR	抵抗値の数値を入力
	InputNumberC	静電容量の数値を入力
ListBox	ListPrefixR	抵抗値の単位を入力
	ListPrefixC	静電容量の単位を入力
Label	SimlateShowFreq	カットオフ周波数の理論値を表示
	MeasureShowFreq	カットオフ周波数の測定値を表示
	DifferenceFreqPercemt	理論値と測定値の誤差を%表示
PictureBox	GraphDisplay	グラフの描画対象



解答例のソースコード（フォーム）

行	プログラム
1	`using System;`
2	`using System.IO;`
3	`using System.Collections.Generic;`
4	`using System.ComponentModel;`
5	`using System.Data;`
6	`using System.Drawing;`
7	`using System.Linq;`
8	`using System.Text;`
9	`using System.Drawing.Drawing2D;`
10	`using System.Drawing.Imaging;`
11	`using System.Windows.Forms;`
12	`using Polytech.Windows.Form;  //Graphクラス`
13	`using Filter;  //HighPassFilterクラス`
14	
15	`namespace GraphFilterSample`
16	`{`
17	`  public partial class GraphForm : Form`
18	`  {`
19	`    //フォーム内で使用するメンバー`
20	
21	`    static private Graph myGraph;  //グラフ描画用オブジェクト`
22	`    static private decimal PartR, PartC;  //抵抗値，静電容量`
23	
24	`    static private PointF CutoffPoint;  //カットオフ周波数の座標`

```
25
26          static private readonly float GvRangeMin = -20;     //Y軸最小値
27          static private readonly float GvRangeMax = 2;       //Y軸最大値
28          static private readonly float FreqRangeMin = (float)Math.Pow(10, -3);
                                                                //Y軸最小値
29          static private readonly float FreqRangeMax = (float)Math.Pow(10, 9);
                                                                //Y軸最大値
30          static private readonly float Vin = 1;      //入力電圧
31
32          static private List<PointF> Series = new List<PointF>();   //グラフの系列
33          static private HighPassFilter hpf; //測定対象のハイパスフィルタ
34
35          public GraphForm()
36          {
37              InitializeComponent();
38              //イベントハンドラの追加
39              Load += new EventHandler(GraphForm_Load);
40              FormClosing += new FormClosingEventHandler(GraphForm_FormClosing);
41
42              DrawGraph.Click += new EventHandler(DrawGraph_Click);
43              MenuFileOpen.Click += new EventHandler(DrawGraph_Click);
44
45              SaveGraph.Click +=new EventHandler(SaveGraph_Click);
46              MenuFileSaveGraph.Click += new EventHandler(SaveGraph_Click);
47
48              SaveCSV.Click += new EventHandler(SaveCSV_Click);
49              MenuFileSaveCSV.Click += new EventHandler(SaveCSV_Click);
50
51              MenuCircuitView.Click += new EventHandler(MenuCircuitView_Click);
52
53          }
54
55          private void GraphForm_Load(System.Object sender, System.EventArgs e)
56          {
57              //ピクチャーボックスの外形線を立体表示
58              GraphDisplay.BorderStyle = BorderStyle.Fixed3D;
59
60              //抵抗の単位のリスト追加
61              ListPrefixR.Items.Add("Ω");
62              ListPrefixR.Items.Add("kΩ");
63              ListPrefixR.Items.Add("MΩ");
64              ListPrefixR.SelectedIndex = 1;
65              //抵抗値の入力範囲の設定
66              InputNumberR.Value = 1;
67              InputNumberR.Maximum = (decimal)Math.Pow(10, 5);
68              InputNumberR.Minimum = (decimal)Math.Pow(10, -3);
69              InputNumberR.DecimalPlaces = 3;
70              InputNumberR.Increment = 0.01M;
71
72              //静電容量の単位のリスト追加
```

```
73              ListPrefixC.Items.Add("pF");
74              ListPrefixC.Items.Add("nF");
75              ListPrefixC.Items.Add("uF");
76              ListPrefixC.SelectedIndex = 2;
77              //静電容量の入力範囲の設定
78              InputNumberC.Value = 1;
79              InputNumberC.Maximum = (decimal)Math.Pow(10, 5);
80              InputNumberC.Minimum = (decimal)Math.Pow(10, -3);
81              InputNumberC.DecimalPlaces = 3;
82              InputNumberC.Increment = 0.01M;
83
84              //イベントの関連付け(上記の設定変更でイベントが動作しないようにするため)
85              InputNumberR.ValueChanged += new EventHandler(InputParameter);
86              InputNumberC.ValueChanged += new EventHandler(InputParameter);
87              ListPrefixR.SelectedValueChanged += new EventHandler(InputParameter);
88              ListPrefixC.SelectedValueChanged += new EventHandler(InputParameter);
89
90              //パラメータ入力のイベントを呼び出す
91              InputParameter(sender, e);
92
93          }
94
95          /// <summary>起動時に実行されるイベントです。</summary>
96          private void GraphForm_FormClosing(System.Object sender,
                                               System.Windows.Forms.FormClosingEventArgs e)
97          {
98              DialogResult result;
99
100             result = MessageBox.Show("プログラムを終了しますか?", "確認",
MessageBoxButtons.YesNo, MessageBoxIcon.Question);
101             if (result == DialogResult.No)
102             {
103                 e.Cancel = true;     //このイベントをキャンセル
104             }
105         }
106
107         /// <summary>抵抗Rとコンデンサcの入力された数値と単位からパラメータを
                                                   演算するメソッドです。</summary>
108         private void InputParameter(System.Object sender, System.EventArgs e)
109         {
110             //入力されたRとCが0以下の時は以下を実行しない
111             if ((InputNumberR.Value <= 0) || (InputNumberC.Value <= 0))
112                 return;
113
114             switch (ListPrefixR.Text)
115             {
116                 case "MΩ":
117                     PartR = InputNumberR.Value * (decimal)Math.Pow(10, 6);
118                     break;
119                 case "kΩ":
```

```
120                     PartR = InputNumberR.Value * (decimal)Math.Pow(10, 3);
121                     break;
122                 default:
123                     PartR = InputNumberR.Value;
124                     break;
125             }
126 
127             switch (ListPrefixC.Text)
128             {
129                 case "pF":
130                     PartC = InputNumberC.Value * (decimal)Math.Pow(10, -12);
131                     break;
132                 case "nF":
133                     PartC = InputNumberC.Value * (decimal)Math.Pow(10, -9);
134                     break;
135                 case "uF":
136                     PartC = InputNumberC.Value * (decimal)Math.Pow(10, -6);
137                     break;
138                 default:
139                     PartC = InputNumberC.Value;
140                     break;
141             }
142 
143         }
144         /// <summary>周波数特性の実測値の描画とカットオフ周波数を開始する
                                                        イベントです。</summary>
145         private void DrawGraph_Click(Object sender, System.EventArgs e)
146         {
147             int i = 0;
148             float freqMin = 1;  //最小周波数
149             float freqMax = 1;  //最大周波数
150             List<float> freqList = new List<float>(); //ファイルから読み込んだ周波数
151             //グラフに適用する周波数の最小と最大値
152             float graphFreqMin, graphFreqMax;
153 
154             //①ファイル読み込み最小、最大周波数と周波数のListの取得
155             if (!OpenCSV(ref freqMin, ref freqMax, ref freqList)) return;
156 
157             //②測定対象のハイパスフィルタのインスタンス生成
158             hpf = new HighPassFilter(PartR, PartC);
159             //抵抗とコンデンサのパラメータ
160             CutoffPoint.X = (float)hpf.GetCutoffFrequency();
161             //カットオフ周波数の取得
162             CutoffPoint.Y = (float)hpf.GetGain((decimal)Vin,
                                                        (decimal)CutoffPoint.X);
163             //利得の計算
164             CutoffFreqLabel.Text = AddPrefixUnit(CutoffPoint.X, "Hz");
165 
166             //③周波数の範囲を表示
167             FreqRangeLabel.Text = AddPrefixUnit(freqMin, "Hz") + "～" +
```

```
                                                    AddPrefixUnit(freqMax, "Hz");
168             //周波数の範囲を10の乗数になるように変換してからグラフを初期化
169             graphFreqMin = (float)(Math.Pow(10, Math.Floor(Math.Log10(freqMin))));
170             graphFreqMax = (float)(Math.Pow(10,
                                                Math.Ceiling(Math.Log10(freqMax))));
171             DrawSheet(graphFreqMin, graphFreqMax);
172
173             //④描画データの生成
174             Series.Clear();
175             //グラフの座標のリストをクリア
176             //電圧利得を算出して系列データに代入
177             foreach (float freq in freqList)
178             {
179                 Series.Add(new PointF(freq, (float)hpf.GetGain((decimal)Vin,
                                                                (decimal)freq)));
180             }
181
182             //⑤系列データ間を線分で描画
183             myGraph.LineSet(DashStyle.Solid, 2, Color.Red);
184             for (i = 0; i <= Series.Count - 2; i++)
185             {
186                 if (GvRangeMin <= Series[i].Y & Series[i].Y <= GvRangeMax)
187                 {
188                     myGraph.DrawLine(Series[i], Series[i + 1]);
189                 }
190             }
191
192             //⑥カットオフ周波数の描画
193             myGraph.MarkerSet(Graph.MarkerShapeStyle.Circle, 5, Color.Blue);
194             myGraph.DrawMarker(CutoffPoint);
195             myGraph.LineSet(DashStyle.Solid, 1, Color.Blue);
196             myGraph.DrawLine(CutoffPoint.X, GvRangeMin, CutoffPoint.X,
                                                            CutoffPoint.Y);
197             myGraph.DrawLine(graphFreqMin, CutoffPoint.Y, CutoffPoint.X,
                                                            CutoffPoint.Y);
198             myGraph.FontColor = Color.Blue;
199             myGraph.DrawText(CutoffPoint.X * 0.9F, GvRangeMin - 0.5F, "fc");
200             myGraph.DrawText(graphFreqMin * 0.7F, -3F + 0.5F,
                                                    AddPrefixUnit(-3).PadLeft(4));
201         }
202
203         /// <summary>グラフを画像ファイルとして保存するメソッドです。</summary>
204         private void SaveGraph_Click(Object sender, System.EventArgs e)
205         {
206             SaveFileDialog saveDialog = new SaveFileDialog();
207             //ダイアログのタイトル
208             saveDialog.Title = "グラフの保存先を指定してください。";
209             //デフォルトのファイル名は"年月日_時分秒.bmp"
210             saveDialog.FileName = DateTime.Now.ToString("yyyyMMdd_HHmmss") +
                                                                    ".bmp";
```

```
211             //デスクトップをデフォルトの保存先に指定
212             saveDialog.InitialDirectory =
                                System.Environment.SpecialFolder.Desktop.ToString();
213             //ファイルの種類はBMP, JPEG, PNG
214             saveDialog.Filter = "BMPファイル(*.bmp)|*.bmp|JPEGファイル
                                (*.jpg;*jpeg)|*.jpg;*jpeg|PNGファイル(*.png)|*.png";
215             //ダイアログの表示がOK以外の時は以下の処理を実行しない
216             if (saveDialog.ShowDialog() != System.Windows.Forms.DialogResult.OK)
217                 return;
218
219             if (!myGraph.Export(saveDialog.FileName))
220             {
221                 MessageBox.Show("保存するファイル形式、保存先が正しくありません。");
222             }
223         }
224
225         /// <summary>ダイアログを表示して周波数のファイルを読み込みます。</summary>
226         /// <param name="fMin">最小周波数を参照渡しで取得します。</param>
227         /// <param name="fMax">最大周波数を参照渡しで取得します。</param>
228         /// <param name="fList">周波数をListクラスで取得します。</param>
229         /// <returns>ファイルを正常に読み込めた場合はtrueを、
                                        読み込めない場合はfalseを返します。</returns>
230         private bool OpenCSV(ref float fMin, ref float fMax, ref List<float> fList)
231         {
232             //ファイルから読み込むためのダイアログ
233             OpenFileDialog openDialog = new OpenFileDialog();
234
235             fList = new List<float>();                  //ファイルから読み込んだ周波数
236
237             //ダイアログのタイトル
238             openDialog.Title = "入力信号用のファイルを読み込んでください。";
239             //デフォルトのファイル名指定なし
240             openDialog.FileName = "";
241             //デスクトップをデフォルトのフォルダに指定
242             openDialog.InitialDirectory =
                                System.Environment.SpecialFolder.Desktop.ToString();
243             //ファイルの種類はCSVとText,すべてのファイル
244             openDialog.Filter = "CSVファイル|*.csv|テキストファイル|*.txt|
                                                        すべてのファイル|*.*";
245             //ダイアログの表示がOK以外の時は以下の処理を実行しない
246              if (openDialog.ShowDialog() != System.Windows.Forms.DialogResult.OK)
                                                            [return false;
247
248             //ファイルを読み込む処理
249             using (StreamReader sr = new StreamReader(openDialog.FileName))
250             {
251                 //ファイルを末尾まで読み込む
252                 while (!(sr.EndOfStream))
253                 {
254                     float f = 0;        //ファイルから読み込む数値
```

```
255                     //変換可能な文字列なら変換して格納
256                     if (float.TryParse(sr.ReadLine(), out f))
257                     {
258                         fList.Add(f);   //周波数の値を追加
259                     }
260                 }
261             }
262 
263             if (fList.Count < 2)
264             {
265                 MessageBox.Show("読み込んだファイルのデータに不足、または異常があります。", "警告", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
266                 return false;
267             }
268 
269             fList.Sort();   //周波数のリストを昇順でソート
270 
271             //周波数の最小値と最大値を格納
272             fMin = fList[0];
273             fMax = fList[fList.Count - 1];
274 
275             //周波数の測定範囲をチェック
276             if (fMin < FreqRangeMin || FreqRangeMax < fMax)
277             {
278                 MessageBox.Show("周波数が描画可能が範囲を超えています。", "警告",
                                    MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
279                 return false;
280             }
281             return true;
282         }
283 
284         /// <summary>パラメータと系列データをCSVファイルに保存する
                                                        イベントです。</summary>
285         private void SaveCSV_Click(Object sender, System.EventArgs e)
286         {
287             SaveFileDialog saveDialog = new SaveFileDialog();
288             //ダイアログのタイトル
289             saveDialog.Title = "周波数特性の保存先を指定してください。";
290             //デフォルトのファイル名は"年月日_時分秒.csv"
291             saveDialog.FileName = DateTime.Now.ToString("yyyyMMdd_HHmmss") +
                                                                    ".csv";
292             //デスクトップをデフォルトの保存先に指定
293             saveDialog.InitialDirectory =
                                    System.Environment.SpecialFolder.Desktop.ToString();
294             //ファイルの種類はCSVとText,すべてのファイル
295             saveDialog.Filter = "CSVファイル|*.csv|テキストファイル|*.txt|
                                                        すべてのファイル|*.*";
296             //ダイアログの表示がOK以外の時は以下の処理を実行しない
297             if (saveDialog.ShowDialog() != System.Windows.Forms.DialogResult.OK)
                                                                    return;
```

```
298
299             using (StreamWriter sw = new StreamWriter(saveDialog.FileName))
300             {
301                 //ファイルの先頭にパラメータを記述
302                 sw.WriteLine("R=" + PartR.ToString());
303                 sw.WriteLine("C=" + PartC.ToString());
304                 //特性をファイルへ書き込む
305                 foreach (PointF p in Series)
306                 {
307                     sw.Write(p.X.ToString());
308                     sw.Write(",");
309                     sw.WriteLine(p.Y.ToString());
310                 }
311             }
312         }
313
314         private void MenuExit_Click(Object sender, System.EventArgs e)
315         {
316             Application.Exit(); //終了
317         }
318
319         private void MenuCircuitView_Click(Object sender, System.EventArgs e)
320         {
321             AboutBox CircuitView = new AboutBox();
322             CircuitView.ShowDialog();
323         }
324
325         /// <summary>グラフの目盛等を描画するメソッドです。</summary>
326         ///<param name="frequencyMin" >描画する周波数の最小値を指定します。</param>
327         /// <param name="frequencyMax">描画する周波数の最大値を指定します。</param>
328         private void DrawSheet(float frequencyMin, float frequencyMax)
329         {
330
331             int i, j;
332             int digitMin = 0;
333             int digitMax = 0;
334
335             digitMin = (int)(Math.Log10(frequencyMin));
336             digitMax = (int)(Math.Log10(frequencyMax));
337             //範囲を左下(最小周波数, 最小利得-20dB)と右上(最大周波数, 最大利得2dB),
                        余白の上下を20pixelと左右30pixel,X軸線形軸,Y軸対数軸でグラフを描画
338             myGraph = new Graph(GraphDisplay, frequencyMin, GvRangeMin - 4,
                  frequencyMax, GvRangeMax + 3, 30, 20, Graph.AxisStyle.XlogYlinear);
339
340             //背景色は白
341             myGraph.BackColor = Color.White;
342
343             //フォントの変更
344             myGraph.FontSet("ＭＳ ゴシック", 10, FontStyle.Regular, Color.Black);
345
```

```
346             //線を実線、幅2、黒に変更
347             myGraph.LineSet(DashStyle.Solid, 2, Color.Black);
348             myGraph.DrawRectangle(frequencyMin, GvRangeMin, frequencyMax,
                                                                    GvRangeMax);
349
350             //X軸の目盛線を追加
351             for (i = digitMin; i <= digitMax; i += 1)
352             {
353                 for (j = 1; j <= 9; j += 1)
354                 {
355                     if (j == 1)
356                     {
357                         myGraph.LineSet(DashStyle.Solid, 1, Color.Green);
358                         myGraph.DrawText(j * (float)Math.Pow(10, i) * 0.9F,
                                GvRangeMin - 2F, AddPrefixUnit(j * (float)Math.Pow(10, i)));
359                     }
360                     else
361                     {
362                         myGraph.LineSet(DashStyle.Dash, 1, Color.Green);
363                     }
364                     myGraph.DrawLine(j * (float)Math.Pow(10, i), GvRangeMin,
                                              j * (float)Math.Pow(10, i), GvRangeMax);
365                 }
366             }
367             //X軸のタイトル表示
368             myGraph.FontStyle = FontStyle.Bold;
369             myGraph.DrawText((float)Math.Pow(10, (digitMax - digitMin - 1)),
                                                    GvRangeMin - 4, "周波数f[Hz]");
370             myGraph.FontStyle = FontStyle.Regular;
371
372             //Y軸の目盛線を追加
373             for (i = (int)GvRangeMin; i <= (int)GvRangeMax; i += 5)
374             {
375                 if (i == 0)
376                 {
377                     myGraph.LineSet(DashStyle.Solid, 1, Color.Green);
378                 }
379                 else
380                 {
381                     myGraph.LineSet(DashStyle.Dash, 1, Color.Green);
382                 }
383                 myGraph.DrawLine(frequencyMin, i, frequencyMax, i);
384                 myGraph.DrawText(frequencyMin * 0.7F, i + 0.5F,
                                                    AddPrefixUnit(i).PadLeft(4));
385             }
386             //Y軸のタイトル表示
387             myGraph.FontStyle = FontStyle.Bold;
388             myGraph.DrawText(frequencyMin * 0.8F, GvRangeMax + 2F, "電圧利得Gv[dB]");
389             myGraph.FontStyle = FontStyle.Regular;
390         }
```

```
391
392         /// <summary>数値にSI接頭辞を付加して文字列を返すメソッドです。</summary>
393         /// <param name="number">数値を指定します。</param>
394         /// <returns>小数点以下第3位までの数値にSI接頭辞を文字列を返します。</returns>
395         private string AddPrefixUnit(float number)
396         {
397             return AddPrefixUnit(number,"");
398         }
399
400         /// <summary>数値にSI接頭辞と単位を付加して文字列を返すメソッドです。</summary>
401         /// <param name="number">数値を指定します。</param>
402         /// <param name="unit">単位を指定します。省略した場合は単位は付加されません。
                                                                </param>
403         /// <returns>小数点以下第3位までの数値にSI接頭辞を文字列を返します。</returns>
404         private string AddPrefixUnit(float number, string unit)
405         {
406             string addNumber = string.Empty;
407
408             if (number > Math.Pow(10, 12))
409                 addNumber = (number / Math.Pow(10, 12)).ToString("0.###") + "T";
410             else if (number > Math.Pow(10, 9))
411                 addNumber = (number / Math.Pow(10, 9)).ToString("0.###") + "G";
412             else if (number > Math.Pow(10, 6))
413                 addNumber = (number / Math.Pow(10, 6)).ToString("0.###") + "M";
414             else if (number > Math.Pow(10, 3))
415                 addNumber = (number / Math.Pow(10, 3)).ToString("0.###") + "k";
416             else if (number > Math.Pow(10, 0))
417                 addNumber = number.ToString("0.###");
418             else if (number > Math.Pow(10, -3))
419                 addNumber = (number * Math.Pow(10, 3)).ToString("0.###") + "m";
420             else if (number > Math.Pow(10, -6))
421                 addNumber = (number * Math.Pow(10, 6)).ToString("0.###") + "u";
422             else if (number > Math.Pow(10, -9))
423                 addNumber = (number * Math.Pow(10, 9)).ToString("0.###") + "n";
424             else if (number > Math.Pow(10, -12))
425                 addNumber = (number * Math.Pow(10, 12)).ToString("0.###") + "p";
426             else
427                 addNumber = number.ToString("0.###");
428             return addNumber + unit;
429         }
430     }
431 }
```

解答例のソースコード（ハイパスフィルタのクラス）

行	プログラム
1	`using System;`
2	`using System.Collections.Generic;`
3	`using System.Linq;`
4	`using System.Text;`

```
5
6   namespace Filter
7   {
8       using System;
9       using System.Collections;
10      using System.Collections.Generic;
11      using System.Data;
12      using System.Diagnostics;
13
14      /// <summary>抵抗とコンデンサの一次フィルタ用のクラスです。</summary>
15      /// <remarks>各種パラメータ及びプロパティはDecimal型を適用しています。</remarks>
16      public class HighPassFilter
17      {
18          //プロパティ及びメソッド用メンバー
19          private decimal rValue;         //抵抗値
20          private decimal cValue;         //静電容量
21
22          /// <summary>ハイパスフィルタのインスタンスを生成します。</summary>
23          /// <param name="r">抵抗値を指定します。</param>
24          /// <param name="c">コンデンサの静電容量を指定します。</param>
25          /// <remarks>インスタンス生成時に抵抗とコンデンサの値でフィルタの特性が
                                                        設定されます。</remarks>
26          public HighPassFilter(decimal r, decimal c)
27          {
28              //抵抗とコンデンサの値
29              rValue = r;
30              cValue = c;
31          }
32
33          /// <summary>フィルタの入力信号（電圧と周波数を指定）から
                                                電圧利得を取得するメソッドです。</summary>
34          /// <param name="inputVoltage">入力電圧を指定します。</param>
35          /// <param name="inputFrequency">入力周波数(Hz)を指定します。</param>
36          /// <returns>電圧利得を返します。</returns>
37          /// <remarks>抵抗とコンデンサの値はインスタンス生成時の値が
                                                        適用されます。</remarks>
38          public decimal GetGain(decimal inputVoltage, decimal inputFrequency)
39          {
40              //出力電圧と角周波数ωとωCR算出用の変数
41              decimal vout,gv , w , wcr;
42              //角周波数ω算出
43              w = 2 * (decimal)Math.PI * inputFrequency;
44              //ωCR算出
45              wcr = w * cValue * rValue;
46              //電圧利得の算出
47              vout = (decimal)(Math.Sqrt(Math.Pow((double)wcr, 2) /
                                (1 + Math.Pow((double)wcr, 2))) * (double)inputVoltage);
48              gv = 20 * (decimal)(Math.Log10((double)(vout / inputVoltage)));
49              //電圧利得を返す
50              return gv;
```

```
51          }
52
53          /// <summary>フィルタのカットオフ周波数を取得するメソッドです。</summary>
54          /// <returns>カットオフ周波数を返します。</returns>
55          /// <remarks>抵抗とコンデンサの値はインスタンス生成時の値が適用されます。
    </remarks>
56          public decimal GetCutoffFrequency()
57          {
58              decimal fc;
59
60              fc = (1 / (2 * (decimal)Math.PI * rValue * cValue));
61
62              return fc;
63          }
64
65          /// <summary>このフィルタの抵抗値を取得します。</summary>
66          /// <returns>抵抗値を単位[Ω]で返します。</returns>
67          /// <remarks>このプロパティは確認用です。</remarks>
68          public decimal Rohm
69          {
70              get { return rValue; }
71          }
72
73          /// <summary>このフィルタの静電容量を取得します。</summary>
74          /// <returns>静電容量を単位[F]で返します。</returns>
75          /// <remarks>このプロパティは確認用です。</remarks>
76          public decimal Cfarad
77          {
78              get { return cValue; }
79          }
80      }
81  }
```

作業工程計画書（受講者配付用）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備		
1. 仕様確認と作成する機能		
2. GUI 設計		
3. イベントの割り付け		
4. コーディング		
5. 動作確認		

作業工程計画書（受講者配付用）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備		
1. 仕様確認と作成する機能		
2. GUI 設計		
3. イベントの割り付け		
4. コーディング		
5. 動作確認		

作業工程計画書（受講者配付用例）

作業工程計画書（模範解答例）

作業工程	ポイント（留意事項等）	参考資料（写真、図面等）
準備	作業場所の確認 実習装置と開発環境の確認	
1. 仕様確認と作成する機能	入力パラメータ ・抵抗値 ・コンデンサ 表示機能 ・周波数特性の描画 ・カットオフ周波数の表示 ・理論値と実測値の比較 ファイル処理 ・画像保存	
2. GUI 設計	入力パラメータ ・NumericUpdown で数値指定 ・ListBox で単位指定 表示機能 ・Button で動作開始とする。 ・PictureBox にグラフを描画（Graph クラスを使用） ・カットオフ周波数、比較結果等は Label を使用 ファイル処理 ・SaveFIleDialog クラスで名前を付けて保存	
3. イベントの割り付け	GUI のコントロールにイベントを割り付ける Form1：Click イベント SaveCSV：Click イベント SaveGraph：Click イベント InputNumberR：ValueChanged イベント InputNumberC：ValueChanged イベント ListPrefixR：SelectedValueChanged イベント ListPrefixC：SelectedValueChanged イベント	
4. コーディング	各イベントに処理を記述する。 プロシージャの実装を行なう。	
5. 動作確認	各機能の実装について仕様を満たしているか確認する。	

訓 練 課 題 確 認 シ ー ト

訓練科名　：
仕上がり像　：
システム名　：
訓練課題名　：　パソコンを用いた計測制御システムの製作

入所期　：
氏　名　：

評価区分	評価項目	細　目	評価（数値）					評価判定	評価基準
作業時間	作業時間	作業時間内に作業を完了	2	4	6	8	10	10	指定時間以内に作業が完了していれば10点、以下15分経過する度に2点ずつ減点する。
	資料提出	ソースファイル等の提出	1	2	3	4	5	5	必要提出物が全て提出されていれば5点、提出物が既定時間に提出されていなければ指定時間を経過する度に1減点。
作業工程	作業工程の確認	作業工程計画	1	2	3	4	5	5	作業のポイントが不適切な場合、1か所につき1点減点。
必須機能	パラメータの指定	抵抗値	1		3		5	5	抵抗値の入力が適切にできれば5点、エラー等が発生する要因があれば3点、入力が実装されていなければ1点とする。
		コンデンサ	1		3		5	5	コンデンサの静電容量の入力が適切にできれば5点、エラー等が発生する要因があれば3点、入力が実装されていなければ1点とする。
	グラフ描画	横軸と縦軸	1		3		5	5	横軸が対数軸で表示されていれば2点加点、縦軸が線形軸で電圧利得（デシベル）表示されていれば2点加点、いずれも実装されていなければ1点とする。
		測定結果の表示	2	4	6	8	10	10	計測結果をグラフで正しく表示できていれば5点。表示等に不具合がある場合は1箇所につき1点ずつ減点する。
	計測	カットオフの理論値の算出	2		6		10	10	周波数特性からカットオフ周波数の測定値fc_2を求め表示ができれば4点加点、理論値fc_1を求め表示されていれば4点加点、いずれも実装されていなければ10点とする。
	ファイル処理	ファイルの保存	1		3		5	5	測定結果の周波数と利得をCSVやテキストファイルに出力可能であれば2点加点、表示したグラフを画像ファイルと保存可能であれば2点加点する。どちらも実装していなければ1点とする。
ユーザビリティ	GUI	フォームのデザイン、オペレーション	1	2	3	4	5	5	フォームのデザインやオペレーションに関して、不具合や欠陥等があれば1箇所につき1点減点する。
	グラフの可読性	目盛線、数値、配色等	1	2	3	4	5	5	グラフの可読性について、目盛の間隔や線種、数値、配色等を妨げるものでなければ5点、可読性を低下させる要素があれば1箇所につき1点減点する。
ドキュメント	可読性	ソースコード中のコメントやその他のドキュメント	2		6		10	10	ソースコード中に適切なコメントが記述されていれば4点加点、アプリケーションの操作方法等を記載したドキュメントがあれば4点間。なければ2点とする。
安全作業	機材の取り扱い	機材の取り扱い、確認作業等	1	2	3	4	5	5	機材の取り扱い、及び動作確認手順等が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。
	VDT作業	座る姿勢、ディスプレイ、休憩時間	1	2	3	4	5	5	椅子の高さ調整、モニタの角度を適正に行っていない場合、1箇所につき2点減点1時間ごとに休憩を取らない場合、1回につき2点減点、無理な姿勢で作業をしている場合0点。
工夫・改善	追加・工夫等	課題で提示した以外の要素	2	4	6	8	10	10	課題で提示した以外の機能が1箇所ある度に2点加点。ただし機能の内容によっては2点以上の加点をしても良い。
	工夫・改善点記入欄		総点					100	＜判定表＞ A　：　80点以上　：到達水準を十分に上回った B　：　60点以上80点未満　：到達水準に達した C　：　60点未満　：到達水準に達しなかった
			合計点					100	
			換算点					100.0	
			総合評価判定					A	

訓練課題のねらい	コメント
	担当指導員氏名：

評価要領

訓練科名　：
仕上がり像　：
システム名　：
訓練課題名　：

評価区分	評価項目	細　目	評価要領（採点要領）	備考
作業時間	作業時間	作業時間内に作業を完了	指定時間以内に作業が完了していれば10点、以下15分経過する度に2点ずつ減点する。	
	資料提出	ソースファイル等の提出	必要提出物が全て提出されていれば5点、提出物が既定時間に提出されていなければ指定時間を経過する度に1減点。	
作業工程	作業工程の確認	作業工程計画	作業のポイントが不適切な場合、1か所につき1点減点。	
必須機能	パラメータの指定	抵抗値	抵抗値の入力が適切にできれば5点、エラー等が発生する要因があれば3点、入力が実装されていなければ1点とする。	
		コンデンサ	コンデンサの静電容量の入力が適切にできれば5点、エラー等が発生する要因があれば3点、入力が実装されていなければ1点とする。	
	グラフ描画	横軸と縦軸	横軸が対数軸で表示されていれば2点加点、縦軸が線形軸で電圧利得（デシベル）表示されていれば2点加点、いずれも実装されていなければ1点とする。	
		測定結果の表示	計測結果をグラフで正しく表示できていれば5点。表示等に不具合がある場合は1箇所につき1点ずつ減点する。	
	演算	カットオフの理論値の算出	周波数特性からカットオフ周波数の測定値fc2を求め表示ができれば4点加点、理論値fc1を求め表示されていれば4点加点、いずれも実装されていなければ10点とする。	
	ファイル処理	ファイルの保存	測定結果の周波数と利得をCSVやテキストファイルに出力可能であれば2点加点、表示したグラフを画像ファイルと保存可能であれば2点加点する。どちらも実装していなければ1点とする。	
ユーザビリティ	GUI	フォームのデザイン、オペレーション	フォームのデザインやオペレーションに関して、不具合や欠陥等があれば1箇所につき1点減点する。	
	グラフの可読性	目盛線、数値、配色等	グラフの可読性について、目盛の間隔や線種、数値、配色等を妨げるものでなければ5点、可読性を低下させる要素があれば1箇所につき1点減点する。	
ドキュメント	可読性	ソースコード中のコメントやその他のドキュメント	ソースコード中に適切なコメントが記述されていれば4点加点、アプリケーションの操作方法等を記載したドキュメントがあれば4点間。なければ2点とする。	
安全作業	機材の取り扱い	機材の取り扱い、確認作業等	機材の取り扱い、及び動作確認手順等が適切であれば5点、以下不適切な箇所が1箇所ある毎に1点ずつ減点する。	
	VDT作業	座る姿勢、ディスプレイ、休憩時間	椅子の高さ調整、モニタの角度を適正に行っていない場合、1箇所につき2点減点1時間ごとに休憩を取らない場合、1回につき2点減点、無理な姿勢で作業をしている場合0点。	
工夫・改善	追加・工夫等	課題で提示した以外の要素	課題で提示した以外の機能が1箇所ある度に2点加点。ただし機能の内容によっては2点以上の加点をしても良い。	

筆記課題

管理番号：Ｅ－46A
「パソコン計測制御のための基礎知識（Visual Basic）」

■課題概要■

パソコン計測制御のために必要となるプログラミング言語（Visual Basic）の要素と、制御機器、計測機器とパソコンを接続するために必要となるインタフェース技術について筆記試験により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領		
訓練課題	○	E-46A-01_訓練課題
解答	○	E-46A-02_解答及び解説
作業工程手順書		
訓練課題確認シート		
評価要領		

※プログラミング言語が異なるAとBの2タイプの課題があります。

筆記課題

「パソコン計測制御のための基礎知識（Visual Basic)」

1 作業時間

90分

2 配付資料

問題用紙，解答用紙

3 課題作成、提出方法

解答用紙のみを回収します

4 注意事項

この筆記試験はVisual Basic(省略表記VB)は特記がないかぎりVB.NET(バージョン2005）以降に準拠した仕様です。またプログラムのコンパイルオプションはデフォルト（Option Explicit On、Option Strict Off、Option Compare Binary、Option Infer On)とします。

１．以下の文章中の（１）～（５）について、適切な語句を（ア）～（ス）から選択し、解答欄（１）～（５）に記入しなさい。

Visual Basic（以下 VB）は（１）社のプログラミング言語であり、主に Windows で動作するアプリケーション開発で利用される。開発したアプリケーションは VB6 までは VB6 ランタイム上で動作したが、VB.NET(2002)以降は（２）上で動作する。VB はマウス等の操作や状態の変化等に対して動作を行なう（３）型の言語である。

VB.NET で開発可能なアプリケーションは（４）、WPF アプリケーション等がある。Web 関連では ASP.NET や Silverlight 等でも VB.NET の言語が利用されている。中でも（４）は、ボタンやラベルといったパーツを配置して（５）を作成することができる。

(ア)Apple	(イ)Microsoft	(ウ)Oracle
(エ)CUI	(オ)GUI	(カ)タッチ
(キ)WCF アプリケーション		(ク)コンソールアプリケーション
(ケ)Windows フォームアプリケーション		(コ)MFC
(サ).NET Framework	(シ)Java	(ス)イベントドリブン

２．以下の文章中の（６）～（１５）について、適切な語句を（ア）～（ツ）から選択し、解答欄（６）～（１５）に記入しなさい。

VB はオブジェクト指向のプログラミング言語であり、全てのデータ型は System.Object 型を継承している。データ型には値型と参照型があり、（６）型はメモリ上のデータに直接アクセスをすることができ、Visual Basic の組み込みデータ型では符号付き整数型の（７）や符号なしの整数型の（８）、実数型では（９）が該当する。（７）型はメモリ上に生成されたインスタンスのアドレスが格納されており、組込みデータ型では Object や（１０）が該当する。

クラスや構造体では、オブジェクトが持つ状態の設定と参照を（１１）、オブジェクトが行なう手続きを（１２）という。クラスではインスタンス生成時に呼び出す処理を（１３）、オブジェクトを破棄する処理を（１４）、特定の事象が発生したことの通知となる（１５）を実装することができる。

(ア)構造体	(イ)クラス	(ウ)Int と Byte
(エ)UInt と UByte	(オ)Integer と Short	(カ)UInteger と Byte
(キ)UShort と Double	(ク)Single と Double	(ケ)Long と Decimal
(コ)String	(サ)Char	(シ)デストラクタ
(ス)コンストラクタ	(セ)メソッド	(ソ)プロパティ
(タ)イベント	(チ)値型	(ツ)参照型

３．以下のプログラムを実行したとき①～⑧のいずれかの行で例外が発生します。エラーが発生する箇所がどの箇所か解答欄（１６）に記入しなさい。またその理由について最も適切なものを以下の（ア）～（オ）から１つ選択し解答欄（１７）に記入しなさい。

ソースコード

```
Dim n1 As SByte, n2 As Byte, n3 As Short, n4 As UShort
Dim n5 As Integer, n6 As UInteger, n7 As Long, n8 As ULong
n8 = 12345        '①
n7 = n8           '②
n6 = n7           '③
n5 = n6           '④
n4 = n5           '⑤
n3 = n4           '⑥
n2 = n3           '⑦
n1 = n2           '⑧
```

(ア)左辺の型が負の値に対応しないためオーバーフローが発生する。
(イ)右辺の型より左辺の型が大きいためオーバーフローが発生する。
(ウ)右辺の値が左辺の型より大きいためオーバーフローが発生する。
(エ)左辺の型より右辺の型が大きいためオーバーフローが発生する。
(オ)左辺と右辺が共に負の値に対応しないためオーバーフローが発生する

4. 以下のメソッドはある周波数f[Hz]において、抵抗R[Ω]とコンデンサC[F]によるハイパスフィルタの電圧利得 Gv[dB]を求めるメソッドを定義したものです。利得を求める式を以下のものとしたとき、以下System.Mathクラスのメンバーやメソッドを利用し適切な記述を解答欄の(18)～(22)へ記述しなさい。

$$Gv = \sqrt{\frac{(\omega CR)^2}{1+(\omega CR)^2}} \qquad \omega = 2\pi f$$

```
Public Class Form1     'クラス(問題には関係なし)
    Private （１８） HpfGain(ByVal f As Double, ByVal r As Double, ByVal c As Double) _
                                                                    As Double
        Dim gv, w, wcr As Double '利得Gvと角周波数ωとωCR算出用の変数
        w = 2 * （１９） * f        '角周波数ω算出
        wcr = w * c * r       'ωCR算出
        gv = （２０）((（２１） / (1 + （２１）))      '電圧利得算出
        （２２） gv          '利得を戻り値として渡す
    End Function
End Class
```

5. 以下のソースコードはプログラムの実行時に 2 つの数値の比較値 compValue が基準値 stdValue に対してどれだけの誤差があるかをパーセントで算出し、誤差なし、誤差が±3%以内、±5%以内、誤差が±5%を越えている旨をメッセージボックスで表示します。（２３）～（２８）について、適切な記述を解答用紙に記入しなさい。

```
Dim compValue As Short, diffPercent As Short
（２３）  stdValue As Short = 10000                    '定数宣言

compValue = Short.Parse(InputNumber.Text)  'テキストボックスより文字列を取得して変換
diffPercent = (compValue - stdValue) * 100             '基準値と比較値の差分のパーセント

If diffPercent （２４） 0 Then
    MessageBox.Show("誤差なし")
ElseIf diffPercent （２５） 3 And diffPercent （２６） -3 Then
    MessageBox.Show("誤差が±3%以内")
ElseIf diffPercent （２７） 5 Or diffPercent （２８） -5 Then
    MessageBox.Show("誤差が±5%越え")
Else
    MessageBox.Show("誤差が±5%以内")
End If
```

6. 以下の処理を実行した、ラベル ShowSumLabel に表示される数値を以下の（ア）～（オ）から 1 つ選択し解答欄（２９）に記入しなさい。

```
Dim iArray() As Integer = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
Dim x(4) As Integer, y(4) As Integer
Dim i, sum As Integer

For i = 1 To x.Length - 1 Step 1
    x(i) = iArray(i * 2 + 1)
    y(i) = iArray(i / 2)
Next
sum = y(i Mod 2)
For Each j As Integer In x
    sum += j
Next
ShowSumLabel.Text = sum.ToString()  'sum の値をラベルに出力
```

(ア)6	(イ)12	(ウ)24	(エ)36	(オ)45

7. 以下のメソッドはパソコンから外部のディジタル入出力（DIO:Digital Input/Output）とアナログ入出力（AIO:Anarog Input/Output）対応するDevice.IOクラス（独自に用意したクラス）のメソッドです。これらのメソッドとIOの外部に接続された各種機器を制御するプログラムについて以下の問いに答えなさい。

IO クラス　コンストラクタ		
名前空間	Device.IO	
宣言	Public Sub New (ByVal **id** As Byte, ByRef **errNo** As Integer)	
パラメータ 1	ByVal id As Byte	使用する IO の ID（ハードウェア番号 1～127）を指定する。
パラメータ 2	ByRef errNo As Integer	インスタンス生成時に初期化を実行し、成功した場合は、0 を、失敗した場合はエラー番号を errNo に格納される。
動作	使用する IO ポートの ID を指定して初期化する。通常はデバイスの使用開始時に実行する。	

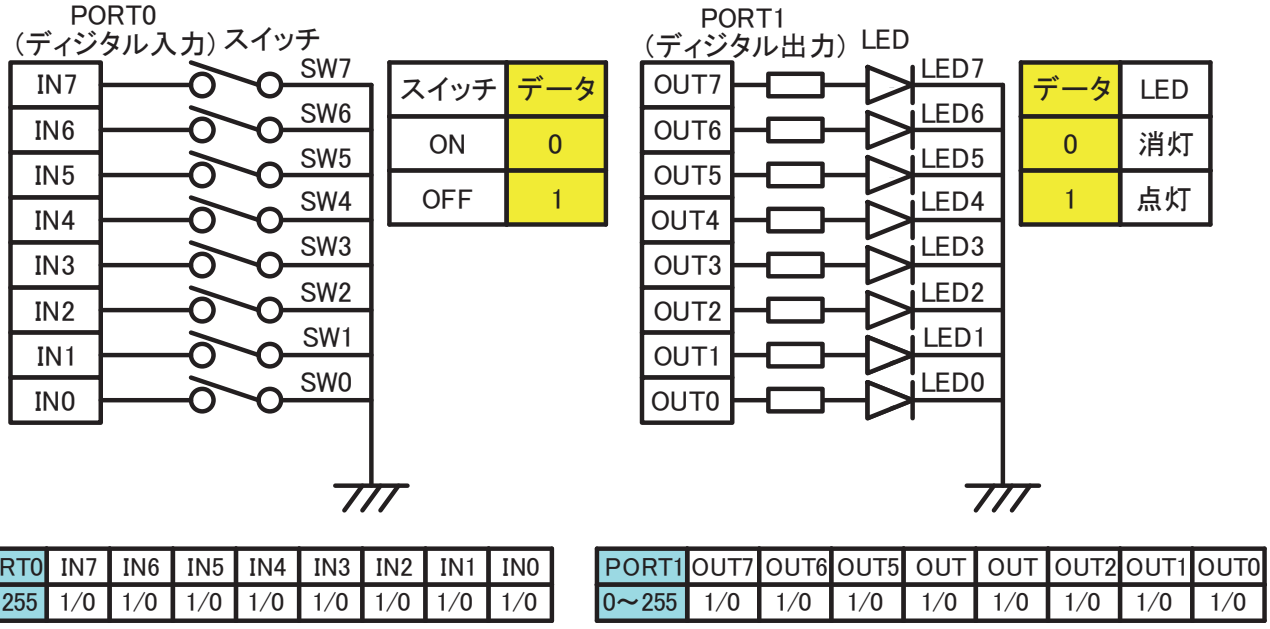
IO.DigitalOut メソッド		
名前空間	Device.IO	
宣言	Public Function **DigitalOut**(ByVal **port** As Byte, ByVal **data** As Byte) As Integer	
パラメータ 1	ByVal port As Byte	使用する IO のポート番号を指定する。
パラメータ2	ByVal data As Byte	出力するデータを 1Byte で指定する。出力成功時は対応するビットが'1'のとき High の電圧が、'0'のとき Low の電圧が出力される。
戻り値	Integer	0 のとき出力成功、0 以外の時はエラー番号を格納。
動作	IO クラスのオブジェクトを使用して指定されたデータを信号として出力する。	

IO.DigitalIn メソッド		
名前空間	Device.IO	
宣言	Public Function **DigitalIn**(ByVal **port** As Byte, ByRef **data** As Byte) As Integer	
パラメータ 1	ByVal port As Byte	使用する IO のポート番号を指定する。
パラメータ2	ByRef data As Byte	入力されたデータを 1Byte で取得する。入力成功時は対応する入力が High のとき'1'を、Low のとき'0'が所定のビットに格納される。
戻り値	Integer	0 のとき入力成功、0 以外の時はエラー番号を格納。
動作	IO クラスのオブジェクトを使用して入力された信号をデータとして取得する。	

IO.AnalogOut メソッド		
名前空間	Device.IO	
宣言	Public Function **AnalogOut** (ByVal **port** As Byte, ByVal **data()** As Short) As Integer	
パラメータ 1	ByVal port As Byte	使用する IO のポート番号を指定する。
パラメータ2	ByVal data() As Short	出力する電圧を Short 型の配列（data(0)～data(3)が DA0～DA3 に対応する 4 要素）で指定する。成功時の電圧はアナログ出力変換特性による。
戻り値	Integer	0 のとき出力成功、0 以外の時はエラー番号を格納。
動作	IO クラスのオブジェクトを使用して指定されたデータをアナログ電圧として出力する。	

IO.AnalogIn メソッド		
名前空間	Device.IO	
宣言	Public Function **AnalogIn**(ByVal **port** As Byte, ByRef **data()** As Short) As Integer	
パラメータ 1	ByVal port As Byte	使用する IO のポート番号を指定する。
パラメータ2	ByRef data As Short	入力された電圧を Short 型の配列（data(0)～data(3)が AD0～AD3 に対応する 4 要素）で取得する。成功時の値はアナログ入力変換電圧による。
戻り値	Integer	0 のとき入力成功、0 以外の時はエラー番号を格納。
動作	IO クラスのオブジェクトを使用してアナログ電圧をデータとして取得する。	

IO.Dispose メソッド	
名前空間	Device.IO
宣言	Public Sub Dispose()
動作	IO オブジェクトが使用するリソースを明示的に開放する。



PORT0	IN7	IN6	IN5	IN4	IN3	IN2	IN1	IN0
0～255	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0

PORT1	OUT7	OUT6	OUT5	OUT	OUT	OUT2	OUT1	OUT0
0～255	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0

回路図 1：PORT0 と PORT1 の機器の接続

回路図１に関するソースコード

```
Imports Device.IO   'DIO, AIO用名前空間
Public Class Form1
    Private myIO As IO    'IOクラスのオブジェクト
    Private errorNo As Integer

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        myIO = New IO(1, errorNo)      'プログラム起動時IOをID1で初期化
        If errorNo <> 0 Then Application.Exit() '初期化失敗時プログラムの終了
        DisplayLED(&HAA)     '初期化後のLEDの出力状態
    End Sub

    Private Sub Form1_FormClosing(sender As Object, e As FormClosingEventArgs) Handles _
                                                                        Me.FormClosing
        [ （３０） ]  'プログラム終了時IOを開放する
    End Sub

    Private Sub ShowSwitchState()
        Dim inData As Byte  'スイッチから取得したデータ
        If myIO.DigitalOut(0, inData) = 0 Then   'スイッチ取得成功時
            SwitchStateLabel.Text = inData.ToString("X2")  'データを16進数2桁大文字で表示
        End If
    End Sub

    Private Sub DisplayLED(ByVal data As Byte)
        '引数に指定されたデータをLEDへ出力する
        Dim outData As Byte
        outData = (&H3F And data) Or &H5

        If myIO.DigitalOut(1, data) <> 0 Then
            MessageBox.Show("出力エラー発生")
        End If

    End Sub

End Class
```

7－1．回路図 1 に関する問題（スイッチと LED）

（1）以下の回路図1に関するソースコードにおいて空白部分[　　]は Device.IO クラスのリソースをプログラム終了時に開放するための処理です。この空白部分に当てはまる適切な記述（オブジェクト名とメソッド名や引数も含め）を解答欄（**30**）に記入しなさい。

（2）初期化成功時 PORT1 に接続されている LED0～LED7 の状態について、点灯しているものに〇、消灯しているものに×を解答欄（**31**）に記入しなさい。なお出力処理中エラーが発生しないものとします。

LED	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
状態								

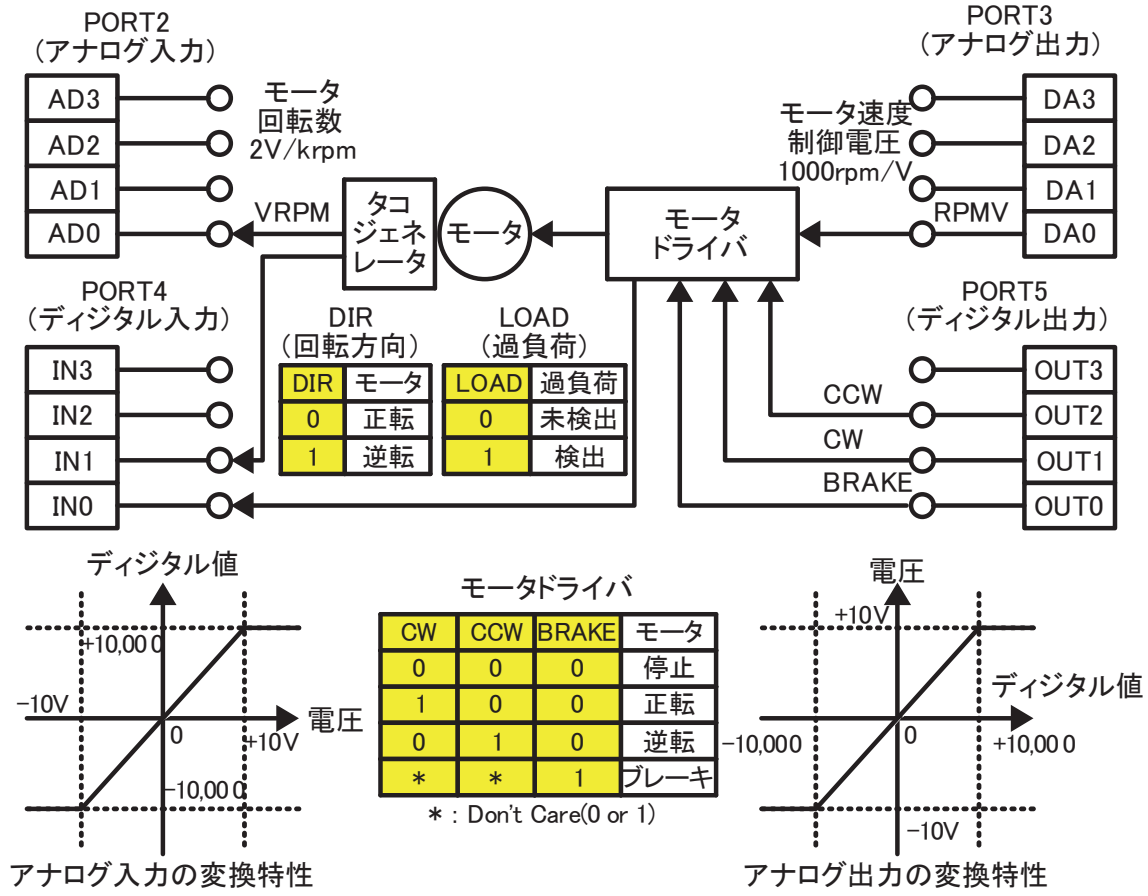
（3）PORT0 に接続されているスイッチ SW0～SW7 が以下の状態で ShowSwitchState メソッドを実行したときにラベルに表示される内容を（**ア**）～（**オ**）から選択し解答欄（**32**）に記入しなさい。

スイッチ	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
状態	OFF	ON	OFF	OFF	ON	ON	OFF	ON

（**ア**）B2　　（**イ**）0xB2　　（**ウ**）B2H　　（**エ**）b2　　（**オ**）&Hb2

7－2. 回路図2に関する問題（モータ制御）

先ほどと同じクラスを使用し、PORT2 から PORT5 に対しモータドライバ、タコジェネレータをそれぞれ接続したものです。これによりモータの回転数を制御するプログラムについて以下の問いに答えなさい。



信号名	機能	備考
VRPM	モータの回転数を 2V/krpm で取得する。 絶対値(正の電圧)で出力する。	1krm(=1000rpm)のとき 2V が入力される。 回転方向は検出できない。
RPMV	モータの回転数を 1000rpm/V で制御する。	1V 出力でモータが 1000rpm で回転する。
DIR	モータの回転方向を取得する。	停止中の状態は保証しない。
VRPM	モータに過負荷がかかっているかを検出する。	過負荷を検出してもモータ制御の変化なし。
CCW	モータを逆転制御する。	モータの制御は CW、CCW、BRAKE の組み合わせで行なわれる。ただし停止状態からの回転は RPMV の電圧が 0.5V 以上必要となる。
CW	モータを正転制御する。	
BRAKE	モータにブレーキをかける。	

※rpm(revolution per minute)で単位分あたりの回転数を表し 1krpm=1000rpm の場合 1 分に 1000 回転

回路図２に関するソースコード（次ページに続きあり）

```
Public Class Form2
    Private errorNo As Integer
    Private myIO As New IO(1, errorNo)
    Private da(3) As Short  'DA0～DA3に出力するための配列

    Private Sub Form2_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        myIO = New IO(1, errorNo)   'プログラム起動時IOをID1で初期化
        If errorNo <> 0 Then Application.Exit() '初期化失敗時プログラムの終了
        myIO.DigitalOut(3, 0)       'モータを停止状態
        da = New Short(3) {0, 0, 0, 0}   '配列の初期化
        myIO.AnalogOut(5, da)       '配列daのアナログ出力
    End Sub

    Private Sub NormalRotate()
        da(0) = （３３）            '3000rpmとなる電圧を出力
        myIO.DigitalOut(（３３）)   '正転となるデータをPORT5へ出力
        myIO.AnalogOut(（３３）)    '配列daをアナログ出力
    End Sub

    Private Sub MotorState(ByRef rpm As Short, ByRef cw As Boolean)
        Dim ad(3) As Short '電圧を格納する配列
        Dim dir As Byte    'DIRが接続されているポートのデータ
        'アナログとディジタル電圧の取得
        myIO.AnalogIn(2, ad)        'ad0～ad3の取得
        myIO.DigitalIn(4, dir)      'dirの取得
        rpm = ad(0) / （３４）      '回転数をrpmで算出()
        If (dir And （３５）) = 0 Then   'DIRが格納されているビットを1でマスク
            cw = True       'モータが正転状態
        Else
            cw = False      'モータが逆転状態
        End If
    End Sub

    Private Sub MotorSequential()
        'このメソッド呼び出し前にMotorNormalRotateを実行している
        For da(0) = 3000 To 6000 Step 60   '①
            myIO.AnalogOut(3, da)
            System.Threading.Thread.Sleep(10)   '10msの待ち時間
```

```
        Next
        System.Threading.Thread.Sleep(1000) '②1000msの待ち時間
        For da(0) = 6000 To 2000 Step 80     '③
            myIO.AnalogOut(3, da)
            System.Threading.Thread.Sleep(10)    '10msの待ち時間
        Next
        For da(0) = [ （３６） ]      '④
            myIO.AnalogOut(3, da)
            System.Threading.Thread.Sleep(10)    '10msの待ち時間
        Next
        While ([ （３７） ])                  '⑤da(0)が0になるまで繰り返す
            da(0) [ （３８） ] 160                    '出力電圧を減算
            myIO.AnalogOut(3, da)
            System.Threading.Thread.Sleep(10)    '10msの待ち時間
        End While
        myIO.DigitalOut(5, &H4)                  'モータをブレーキ
    End Sub
End Class
```

（1）このモータの通常回転状態を正転・3000rpm で回転させるためのメソッド MotorNormalRotate に必要な記述を以下（ア）～（カ）から 1 つ選択し解答欄（３３）に記入しなさい。

（ア）	（イ）	（ウ）
da(0) = 1500 myIO.DigitalOut(5, 1) myIO.AnalogOut(2, da)	da(0) = 1500 myIO.DigitalOut(5, &H2) myIO.AnalogOut(3, da)	da(0) = 1500 myIO.DigitalOut(5, &H5) myIO.AnalogOut(4, da)
（エ）	**（オ）**	**（カ）**
da(0) = 3000 myIO.DigitalOut(5, &H5) myIO.AnalogOut(2, da)	da(0) = 3000 myIO.DigitalOut(5, &H2) myIO.AnalogOut(3, da)	da(0) = 3000 myIO.DigitalOut(5, &H2) myIO.AnalogOut(4, da)

（2）モータに接続されているタコジェネレータから回転数と回転方向を取得するメソッド MotorState について参照渡しされる引数 rpm の演算式と、DIR の状態を判別するifステートメントの条件式を解答欄（３４）と（３５）に記入しなさい。

（3）NomalState 実行中（正転・3000rpm）から MotorSequential メソッド実行し以下のタイムチャートと近似の動作になるような記述を解答欄（３６）～（３８）に記入しなさい。
ただし IO の仕様によりデータ出力の間隔は 10ms を設け実行するものとします。

解答用紙

入所期	氏名	合計点	評価判定
平成　　年　　月生			

1.（2点×5）

（1）	（2）	（3）	（4）	（5）

2.（2点×10）

（6）	（7）	（8）	（9）	（10）
（11）	（12）	（13）	（14）	（15）

3.（2点×2）

（16）	（17）

4.（3点×5）

（18）	（19）	（20）
（21）	（22）	

5.（2点×6）

（23）	（24）	（25）
（26）	（27）	（28）

6.（3点）

（29）

7－1.（4点×3）

（30）					（32）				
（31）	LED	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
	状態								

7－2.（4点×6）

（33）	（34）	（35）
（36）		（37）
（38）		

筆記課題　解答

「パソコン計測制御のための基礎知識（Visual Basic)」

解答用紙

入所期	氏名	合計点	評価判定
平成　　年　　月生		１００	

１．（２点×５）

（１）　イ	（２）　サ	（３）　ス	（４）ケ	（５）オ

２．（２点×１０）

（６）　チ	（７）　オ	（８）　カ	（９）　ク	（１０）　コ
（１１）　ソ	（１２）　セ	（１３）　ス	（１４）　シ	（１５）　タ

３．（２点×２）

（１６）　⑦	（１７）　ウ

４．　（３点×５）

（１８）　Function	（１９）　Math.PI	（２０）Math.Sqrt
（２１）wcr ^ 2 か *Math.Pow(wcr, 2)*	（２２）Return か *HpfGain =*	

５．（２点×６）

（２３）　Const	（２４）　＝	（２５）　＜＝
（２６）　＞＝	（２７）　＞	（２８）　＜

６．（３点）

（２９）　ウ

７－１．（４点×３）

（３０）　myIO.Dispose()	（３２）　ア

（３１）

LED	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
状態	×	×	○	×	○	○	○	○

７－２．（４点×６）

（３３）　オ	（３４）　/ 2	（３５）　&H2
（３６） 2000 To 8000 Step 120		（３７） da(0) ＞ 0
（３８） -= 160　　または = da(0) - 160		

筆記課題

管理番号：E－46B
「パソコン計測制御のための基礎知識（Visual C#）」

■課題概要■

パソコン計測制御のために必要となるプログラミング言語（Visual C#）の要素と、制御機器、計測機器とパソコンを接続するために必要となるインタフェース技術について筆記試験により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領		
訓練課題	○	E-46B-01_訓練課題
解答	○	E-46B-02_解答及び解説
作業工程手順書		
訓練課題確認シート		
評価要領		

※プログラミング言語が異なるAとBの2タイプの課題があります。

筆記課題

「パソコン計測制御のための基礎知識（Visual C#)」

1 作業時間

90分

2 配付資料

問題用紙，解答用紙

3 課題作成、提出方法

解答用紙のみを回収します

4 注意事項

この筆記試験は Visual C#(省略表記 C#)は特記がないかぎり Visual C# 2005(言語バージョン 2.0) 以降に準拠した仕様です。

１．以下の文章中の（１）～（５）について、適切な語句を（ア）～（ス）から選択し、解答欄の（１）～（５）に記入しなさい。

Visual C#（以下 VC#）は（１）社のプログラミング言語であり、主に Windows で動作するアプリケーション開発で利用される。開発したアプリケーションは（２）上で動作する。VC#はマウス等の操作や状態の変化等に対して動作を行なう（３）型の言語である。

VC#で開発可能なアプリケーションは（４）、WPF アプリケーション等がある。Web 関連では ASP.NET や Silverlight 等でも VC#の言語が利用されている。中でも（４）は、ボタンやラベルといったパーツを配置して（５）を作成することができる。

(ア)Apple	(イ)Microsoft	(ウ)Oracle
(エ)CUI	(オ)GUI	(カ)タッチ
(キ)WCF アプリケーション		(ク)コンソールアプリケーション
(ケ)Windows フォームアプリケーション		(コ)MFC
(サ).NET Framework	(シ)DirectX	(ス)イベントドリブン

２．以下の文章中の（６）～（１５）について、適切な語句を（ア）～（ツ）から選択し、解答欄の（６）～（１５）に記入しなさい。

VC#はオブジェクト指向のプログラミング言語であり、全てのデータ型は System.Object 型を継承している。データ型には値型と参照型があり、（６）型はメモリ上のデータに直接アクセスをすることができ、VC#の組み込みデータ型では符号付き整数型の（７）や符号なしの整数型の（８）、実数型では（９）が該当する。（７）型はメモリ上に生成されたインスタンスのアドレスが格納されており、組込みデータ型では Object や（１０）が該当する。

クラスや構造体では、オブジェクトが持つ状態の設定と参照を（１１）、オブジェクトが行なう手続きを（１２）という。クラスではインスタンス生成時に呼び出す処理を（１３）、オブジェクトを破棄する処理を（１４）、特定の事象が発生したことの通知となる（１５）を実装することができる。

(ア)構造体	(イ)クラス	(ウ)int と uint
(エ)ulong と sbyte	(オ)int と short	(カ)uint と byte
(キ)single と ushort	(ク)float と double	(ケ)long と decimal
(コ)string	(サ)char	(シ)デストラクタ
(ス)コンストラクタ	(セ)メソッド	(ソ)プロパティ
(タ)イベント	(チ)値型	(ツ)参照型

３．以下のプログラムを実行したとき①で代入した数値が②～⑦で失われる箇所がどの箇所か回答欄の（１６）に記入しなさい。またその理由について最も適切なものを以下の（ア）～（オ）から１つ選択し回答欄の（１７）に記入しなさい。

```
sbyte n1; byte n2; short n3; ushort n4;
int n5; uint n6; long n7; ulong n8;
n8 = 12345;          //①
n7 = (long)n8;       //②
n6 = (uint)n7;       //③
n5 = (int)n6;        //④
n4 = (ushort)n5;     //⑤
n3 = (short)n4;      //⑥
n2 = (byte)n3;       //⑦
n1 = (sbyte)n2;      //⑧
```

(ア)左辺の型が負の値に対応しない。
(イ)暗黙の型変換によりデータが失われる。
(ウ)明示的な型変換(キャスト)で指定された型が小さいためデータが失われる。
(エ)明示的な型変換(キャスト)で指定された型が大きいためデータが失われる。
(オ)左辺の型と右辺の型が異なるため。

4．以下のメソッドはある周波数f[Hz]において、抵抗R[Ω]とコンデンサC[F]によるハイパスフィルタの電圧利得 Gv[dB]を求めるメソッドを定義したものです。利得を求める式を以下のものとしたとき、以下System.Mathクラスのメンバーやメソッドを利用し適切な記述を回答欄の(18)～(22)へ記述しなさい。

$$Gv = \sqrt{\frac{(\omega CR)^2}{1+(\omega CR)^2}} \qquad \omega = 2\pi f$$

```
Using System;
public partial class Form1 : Form    'クラス(問題には関係なし)
{
    private （１８） HpfGain(double f,double r, double c )
    {
        double gv, w, wcr;          //利得Gvと角周波数ωとωCR算出用の変数
        w = 2 * （１９） * f;       //角周波数ω算出
        wcr = w * c * r;            //ωCR算出
        gv = （２０）((（２１） / (1 + (（２１）)));  //電圧利得算出
        （２２） gv;                  //利得を戻り値として渡す
    }
```

5. 以下のソースコードはプログラムの実行時に 2 つの数値の比較値 compValue が基準値 stdValue に対してどれだけの誤差があるかをパーセントで算出し、誤差なし、誤差が±3%以内、±5%以内、誤差が±5%を越えている旨をメッセージボックスで表示します。（２３）～（２８）について、適切な記述を解答欄の（２３）～（２８）に記入しなさい。

```
compValue = short.Parse(InputNumber.Text);//テキストボックスより文字列を取得して変換
diffPercent = (short)((compValue - stdValue) * 100);//基準値と比較値の差分のパーセント
//以下のifステートメントは"{}"を省略
if (diffPercent （２４） 0)
    MessageBox.Show("誤差なし");
else if (diffPercent （２５） 3 && diffPercent（２６） -3)
    MessageBox.Show("誤差が±3%以内");
else if (diffPercent （２７） 5 || diffPercent （２７） -5)
    MessageBox.Show("誤差が±5%越え");
else
    MessageBox.Show("誤差が±5%以内");
//実行完了
```

6. 以下の処理を実行した、ラベル ShowSumLabel に表示される数値を以下の（ア）～（オ）から 1 つ選択し回答欄（２９）に記入しなさい。

```
int[] iArray = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
int[] x = new int[5]; int[] y = new int[5];
int i = 0, sum = 0;

for (i = 1; i <= x.Length - 1; i += 1)
{
    x[i] = iArray[i * 2 + 1];
    y[i] = iArray[i / 2];
}
sum = y[i % 2];
foreach (int j in x)
{
    sum += j;
}
ShowSumLabel.Text = sum.ToString();              //sum の値をラベルに出力
```

(ア)6	(イ)12	(ウ)24	(エ)36	(オ)45

7. 以下のメソッドはパソコンから外部のディジタル入出力（DIO:Digital Input/Output）とアナログ入出力（AIO:Analog Input/Output）対応する Device.IO クラス（独自に用意したクラス）のメソッドです。これらのメソッドと IO の外部に接続された各種機器を制御するプログラムについて以下の問いに答えなさい。

IO クラス　コンストラクタ		
名前空間	Device.IO	
宣言	public IO(byte **id**, ref int **errNo**)	
パラメータ 1	byte id	使用する IO の ID（ハードウェア番号 1～127）を指定する。
パラメータ 2	ref int errNo	インスタンス生成時に初期化を実行し、成功した場合は、0 を、失敗した場合はエラー番号を errNo に格納される。
動作	使用する IO ポートの ID を指定して初期化する。通常はデバイスの使用開始時に実行する。	

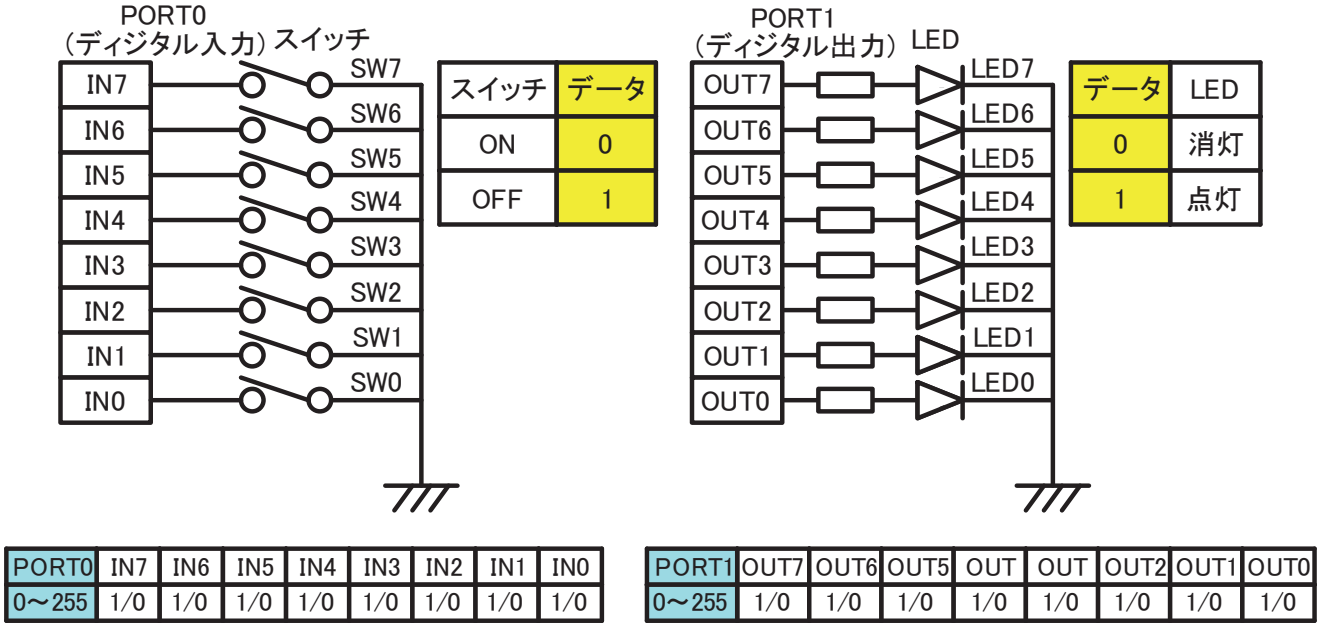
IO.DigitalOut メソッド		
名前空間	Device.IO	
宣言	public int **DigitalOut**(byte **port**, byte **data**)	
パラメータ 1	byte port	使用する IO のポート番号を指定する。
パラメータ2	byte data	出力するデータを 1Byte で指定する。出力成功時は対応するビットが'1'のとき High の電圧が、'0'のとき Low の電圧が出力される。
戻り値	int	0 のとき出力成功、0 以外の時はエラー番号を格納。
動作	IO クラスのオブジェクトを使用して指定されたデータを信号として出力する。	

IO.DigitalIn メソッド		
名前空間	Device.IO	
宣言	public int **DigitalIn**(byte **port**, ref byte **data**)	
パラメータ 1	byte port	使用する IO のポート番号を指定する。
パラメータ2	ref byte data	入力されたデータを 1Byte で取得する。入力成功時は対応する入力が High のとき'1'を、Low のとき'0'が所定のビットに格納される。
戻り値	int	0 のとき入力成功、0 以外の時はエラー番号を格納。
動作	IO クラスのオブジェクトを使用して入力された信号をデータとして取得する。	

IO.AnalogOut メソッド		
名前空間	Device.IO	
宣言	public int **AnalogOut**(byte **port**, short[] **data**)	
パラメータ 1	byte port,	使用する IO のポート番号を指定する。
パラメータ2	short[] data	出力する電圧を short 型の配列（data[0]～data[3]）が DA0～DA3 に対応する 4 要素）で指定する。成功時の電圧はアナログ出力変換特性による。
戻り値	Integer	0 のとき出力成功、0 以外の時はエラー番号を格納。
動作	IO クラスのオブジェクトを使用して指定されたデータをアナログ電圧として出力する。	

IO.AnalogIn メソッド		
名前空間	Device.IO	
宣言	public int **AnalogIn**(byte **port**, ref short[] **data**)	
パラメータ 1	byte port	使用する IO のポート番号を指定する。
パラメータ2	As Short	入力された電圧を Short 型の配列（data(0)～data(3)が AD0～AD3 に対応する 4 要素）で取得する。成功時の値はアナログ入力変換電圧による。
戻り値	int	0 のとき入力成功、0 以外の時はエラー番号を格納。
動作	IO クラスのオブジェクトを使用してアナログ電圧をデータとして取得する。	

IO.Dispose メソッド	
名前空間	Device.IO
宣言	public void Dispose()
動作	IO オブジェクトが使用するリソースを明示的に開放する。



PORT0	IN7	IN6	IN5	IN4	IN3	IN2	IN1	IN0
0～255	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0

PORT1	OUT7	OUT6	OUT5	OUT	OUT	OUT2	OUT1	OUT0
0～255	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0

回路図 1：PORT0 と PORT1 の機器の接続

回路図１に関するソースコード

```
using Device;
public class Form1
{
    //IOクラスのオブジェクト
    private IO myIO;
    private int errorNo;
    private void Form1_Load(object sender, EventArgs e)
    {
         myIO = new IO(1,ref errorNo);          //プログラム起動時IOをID1で初期化
        if (errorNo != 0) System.Windows.Forms.Application.Exit(); //初期化失敗時プログラム終了
        DisplayLED(0xaa);          //初期化後のLEDの出力状態
    }

    private void Form1_FormClosing(object sender, FormClosingEventArgs e)
    {
        （３０）      //プログラム終了時IOを開放する
    }

    private void ShowSwitchState()
    {
        byte inData = 0;        //スイッチから取得したデータ
        if (myIO.DigitalOut(0, inData) == 0)     //スイッチ取得成功時
        {
            SwitchStateLabel.Text = inData.ToString("X2");  //データを16進数2桁大文字で表示
        }
    }

    private void DisplayLED(byte data)
    {
        byte outData = 0;    //引数に指定されたデータをLEDへ出力する
        outData = (byte)((0x3f & data) | 0x05);
        if (myIO.DigitalOut(1, data) != 0)
        {
             MessageBox.Show("出力エラー発生");
        }
    }
}
```

7－1．回路図 1 に関する問題（スイッチと LED）

（1）以下の回路図1に関するソースコードにおいて空白部分 **（30）** は Device.IO クラスのリソースをプログラム終了時に開放するための処理です。この空白部分に当てはまる適切な記述（オブジェクト名とメソッド名や引数も含め）を回答欄（**30**）に記入しなさい。

（2）初期化成功時 PORT1 に接続されている LED0～LED7 の状態について、点灯しているものに〇、消灯しているものに×を回答欄（**31**）に記入しなさい。なお出力処理中エラーが発生しないものとします。

LED	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
状態								

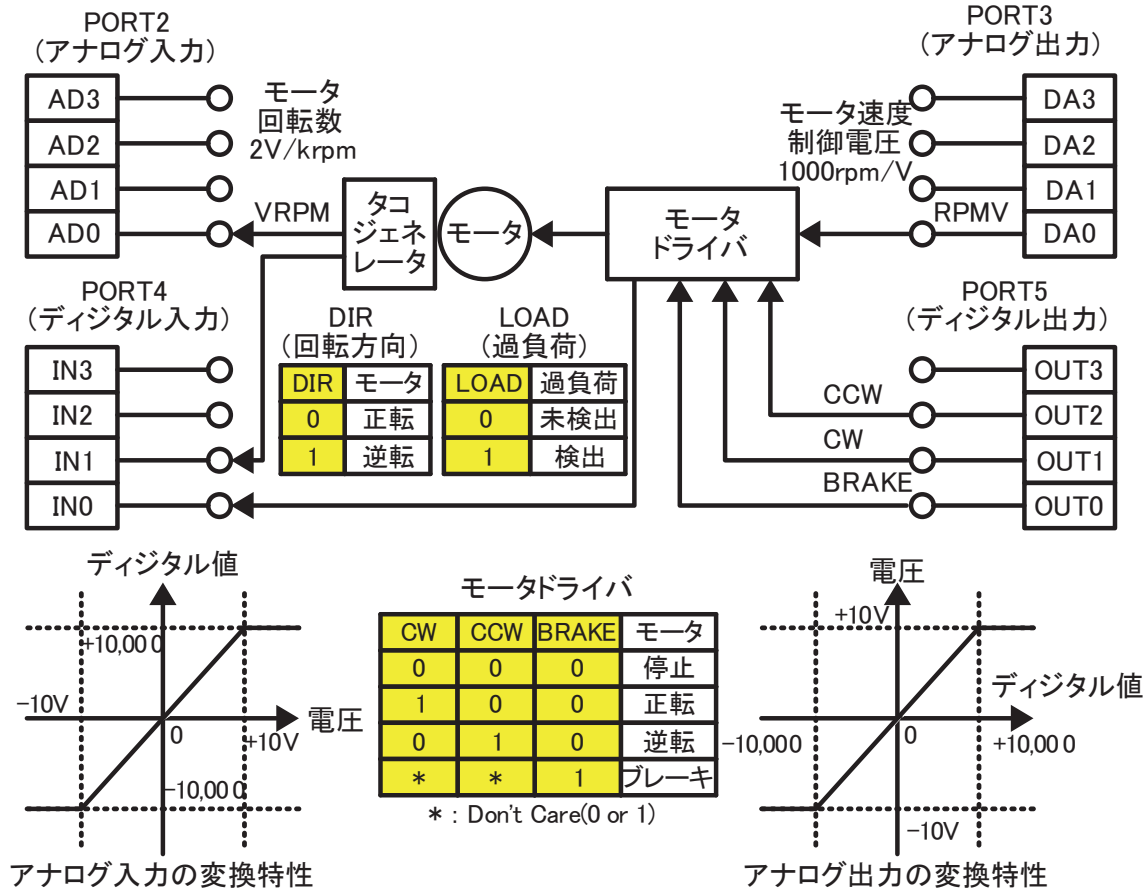
（3）PORT0 に接続されているスイッチ SW0～SW7 が以下の状態で ShowSwitchState メソッドを実行したときにラベルに表示される内容を（**ア**）～（**オ**）から選択し回答欄（**32**）に記入しなさい。

スイッチ	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
状態	OFF	ON	OFF	OFF	ON	ON	OFF	ON

（**ア**）B2　　（**イ**）0xB2　　（**ウ**）B2H　　（**エ**）b2　　（**オ**）&Hb2

7－2. 回路図2に関する問題（モータ制御）

先ほどと同じクラスを使用し、PORT2 から PORT5 に対しモータドライバ、タコジェネレータをそれぞれ接続したものです。これによりモータの回転数を制御するプログラムについて以下の問いに答えなさい。



アナログ入力の変換特性

モータドライバ

CW	CCW	BRAKE	モータ
0	0	0	停止
1	0	0	正転
0	1	0	逆転
*	*	1	ブレーキ

＊：Don't Care(0 or 1)

アナログ出力の変換特性

信号名	機能	備考
VRPM	モータの回転数を 2V/krpm で取得する。 絶対値(正の電圧)で出力する。	1krm(=1000rpm)のとき 2V が入力される。 回転方向は検出できない。
RPMV	モータの回転数を 1000rpm/V で制御する。	1V 出力でモータが 1000rpm で回転する。
DIR	モータの回転方向を取得する。	停止中の状態は保証しない。
VRPM	モータに過負荷がかかっているかを検出する。	過負荷を検出してもモータ制御の変化なし。
CCW	モータを逆転制御する。	モータの制御は CW、CCW、BRAKE の組み合わせで行なわれる。ただし停止状態からの回転は RPMV の電圧が 0.5V 以上必要となる。
CW	モータを正転制御する。	
BRAKE	モータにブレーキをかける。	

※rpm(revolution per minute)で単位分あたりの回転数を表し 1krpm=1000rpm の場合 1 分に 1000 回転

回路図２に関するソースコード（次ページに続きあり）

```
public partial class Form3 : Form
{
    private int errorNo;
    private IO myIO;
    private short[] da = new short[4];        //DA0～DA3に出力するための配列

    private void Form3_Load(object sender, EventArgs e)
    {
        myIO = new IO(1, ref errorNo);   //プログラム起動時IOをID1で初期化
        if (errorNo != 0) System.Windows.Forms.Application.Exit(); //初期化失敗時終了
        myIO.DigitalOut(3, 0);                //モータを停止状態
        da = new short[4] { 0, 0, 0, 0 };   //配列の初期化
        myIO.AnalogOut(5, da);                //配列daのアナログ出力
    }

    private void MotorNormalRotate()
    {
        da[0] = [ （３３） ];                  //3000rpmとなる電圧を出力
        myIO.DigitalOut([ （３３） ]);     //正転となるデータをPORT5へ出力
        myIO.AnalogOut([ （ （３３） ]);       //配列daをアナログ出力
    }

    private void MotorState(ref short rpm, ref bool cw)
    {
        short[] ad = new short[4];   //電圧を格納する配列
        byte dir = 0;                //DIRが接続されているポートのデータ

        //アナログとディジタル電圧の取得
        myIO.AnalogIn(2, ref ad);        //ad0～ad3の取得
        myIO.DigitalIn(4, ref dir);      //dirの取得
        rpm = (short)(ad[0] / [ （３４） ]);        //回転数をrpmで算出
        if ((dir & [ （３５） ]) == 0) {          //DIRが格納されているビットを1でマスク
            cw = true;                   //モータが正転状態
        } else {
            cw = false;                  //モータが逆転状態
        }
    }
```

```
    private void MotorSequential()
    {
        //このメソッド呼び出し前にMotorNormalRotateを実行している
        for (da[0] = 3000; da[0] <= 6000; da[0] += 60)       //①
        {
            myIO.AnalogOut(3, da);
            System.Threading.Thread.Sleep(10);                //10msの待ち時間
        }
        System.Threading.Thread.Sleep(1000);                  //②1000msの待ち時間
        for (da[0] = 6000; da[0] <= 2000; da[0] += 80)       //③
        {
            myIO.AnalogOut(3, da);
            System.Threading.Thread.Sleep(10);                //10msの待ち時間
        }
        for (da[0] = [ （３６） ]) {    //④
            myIO.AnalogOut(3, da);
            System.Threading.Thread.Sleep(10);              //10msの待ち時間
        }
        while ([ （３７） ])                                  //⑤
        {
            da[0] [ （３８） ] 160;                                //出力電圧を減算
            myIO.AnalogOut(3, da);
            System.Threading.Thread.Sleep(10);        //10msの待ち時間
        }
        myIO.DigitalOut(5, 0x04);              //モータをブレーキ
        }
    }
}
```

（1）このモータの通常回転状態を正転・3000rpm で回転させるためのメソッド MotorNormalRotate に必要な記述を以下（**ア**）～（**カ**）から 1 つ選択し回答欄（**３３**）に記入しなさい。

（ア）	（イ）	（ウ）
da[0] = 1500	da[0] = 1500	da[0] = 1500
myIO.DigitalOut(5, 1)	myIO.DigitalOut(5, 0x02)	myIO.DigitalOut(5, 0x05)
myIO.AnalogOut(2, da)	myIO.AnalogOut(3, da)	myIO.AnalogOut(4, da)
（エ）	**（オ）**	**（カ）**
da[0] = 3000	da[0] = 3000	da[0] = 3000
myIO.DigitalOut(5, 0x05)	myIO.DigitalOut(5, 0x02)	myIO.DigitalOut(5, 0x02)
myIO.AnalogOut(2, da)	myIO.AnalogOut(3, da)	myIO.AnalogOut(4, da)

（2）モータに接続されているタコジェネレータから回転数と回転方向を取得するメソッド MotorState について参照渡しされる引数 rpm の演算式と、DIR の状態を判別するifステートメントの条件式を回答欄（**３４**）と（**３５**）に記入しなさい。

（3）NomalState 実行中（正転・3000rpm）から MotorSequential メソッド実行し以下のタイムチャートと近似の動作になるような記述を回答欄（**３６**）～（**３８**）に記入しなさい。
ただし IO の仕様によりデータ出力の間隔は 10ms を設け実行するものとします。

解答用紙

入所期	氏名	合計点	評価判定
平成　　年　　月生			

1.（2点×5）

（1）	（2）	（3）	（4）	（5）

2.（2点×10）

（6）	（7）	（8）	（9）	（10）
（11）	（12）	（13）	（14）	（15）

3.（2点×2）

（16）	（17）

4.（3点×5）

（18）	（19）	（20）
（21）	（22）	

5.（2点×6）

（23）	（24）	（25）
（26）	（27）	（28）

6.（3点）

（29）

7－1.（4点×3）

（30）					（32）				
（31）	LED	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
	状態								

7－2.（4点×6）

（33）	（34）	（35）
（36）		（37）
（38）		

筆記課題　解答

「パソコン計測制御のための基礎知識（Visual C#）」

解答用紙

入所期	氏名	合計点	評価判定
平成　　年　　月生		１００	

１．（２点×５）

（１）　イ	（２）　サ	（３）　ス	（４）ケ	（５）オ

２．（２点×１０）

（６）　チ	（７）　オ	（８）　カ	（９）　ク	（１０）　コ
（１１）　ソ	（１２）　セ	（１３）　ス	（１４）　シ	（１５）　タ

３．（２点×２）

（１６）　⑦	（１７）　ウ

４．　（３点×５）

（１８）　double	（１９）　Math.PI	（２０）Math .Sqrt
（２１）　Math.Pow(wcr, 2)	（２２）　Return	

５．（２点×６）

（２３）　const	（２４）　＝＝	（２５）　＜＝
（２６）　＞＝	（２７）　＞	（２８）　＜

６．（３点）

（２９）　ウ

７－１．（４点×３）

（３０）　myIO.Dispose();	（３２）　ア

（３１）

LED	LED7	LED6	LED5	LED4	LED3	LED2	LED1	LED0
状態	×	×	○	×	○	○	○	○

７－２．（４点×６）

（３３）　オ	（３４）　/ 2	（３５）　0x02
（３６） 2000; da[0] <= 8000; da[0] += 120		（３７）　da[0] > 0
（３８） -= 160　　または = da[0] - 160		

筆記課題

管理番号：Ｅ－47
「フィードバック制御に関する基礎知識」

■課題概要■

自動制御システム設計に必要なフィードバック制御系の伝達関数、ブロック線図、応答特性、安定判別法、PID制御について筆記試験により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領		
訓練課題	○	E-47-01_訓練課題
解答	○	E-47-02_解答及び解説
作業工程手順書		
訓練課題確認シート		
評価要領		

筆記課題

「フィードバック制御に関する基礎知識」

１　作業時間
　　90分

２　配付資料
　　問題用紙，解答用紙

３　課題作成、提出方法
　　解答用紙のみを回収する

問題１

次の記述は自動制御に関して述べたものである。（　）の中に入る適切な語句を（ア）～（シ）から選択しなさい。

自動制御にはシーケンス制御、フィードフォワード制御、フィードバック制御がある。シーケンス制御は「あらかじめ定められた（　①　）に従って制御の各段階を逐次進めていく制御」である。フィードフォワード制御は「（　②　）を予測して、必要な修正を行なう制御」であり、フィードバック制御は「出力側の状況を入力側にフィードバックさせながら、（　③　）に近付ける制御」となる。シーケンス制御は制御量の状態のみを問題にする（　④　）制御に分類されるのに対し、フィードフォワード制御やフィードバック制御は制御量の状態だけでなく量的な値も問題にする（　⑤　）制御に分類される。

制御偏差とは（　③　）から（　⑥　）を引いたもので、定常偏差とは時間が十分経過した後の偏差のことをいい、フィードバック制御系の設計においては定常偏差を０（ゼロ）に近付けることが一つの目標になる。

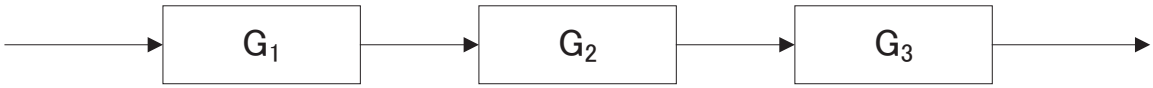
＜語句＞

（ア）信号	（イ）順序	（ウ）外乱	（エ）応答	（オ）特性
（カ）定性	（キ）定量	（ク）目標値	（ケ）制御量	（コ）操作量
（サ）入力量	（シ）変化量			

問題２

各ブロック線図を等価変換した時の伝達関数を次の（ア）～（オ）から選択しなさい。

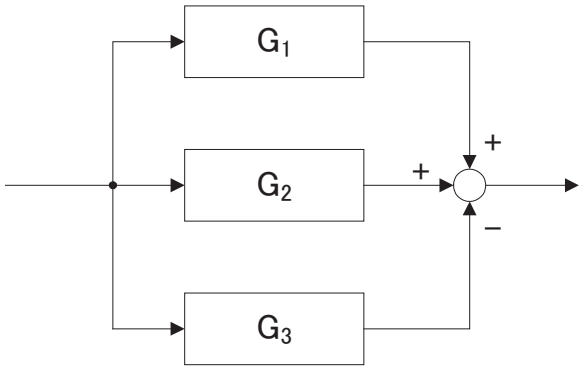
（１）



（ア）$G_1 + G_2 + G_3$　　（イ）$G_1 G_2 G_3$　　（ウ）$\dfrac{1}{G_1 G_2 G_3}$

（エ）$\dfrac{1}{G_1 + G_2 + G_3}$　　（オ）$\dfrac{G_1 G_2 G_3}{G_1 + G_2 + G_3}$

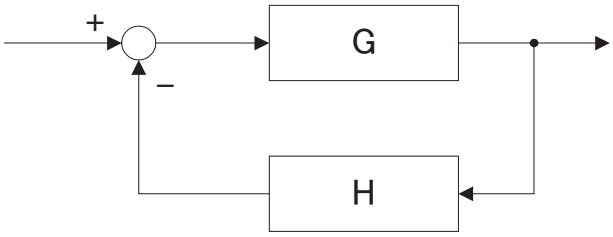
（２）



（ア） $G_1 + G_2 - G_3$　　（イ） $\dfrac{G_1 G_2}{G_3}$　　（ウ） $G_1 G_2 G_3$

（エ） $\dfrac{1}{G_1 + G_2 - G_3}$　　（オ） $\dfrac{G_1 G_2 G_3}{G_1 + G_2 - G_3}$

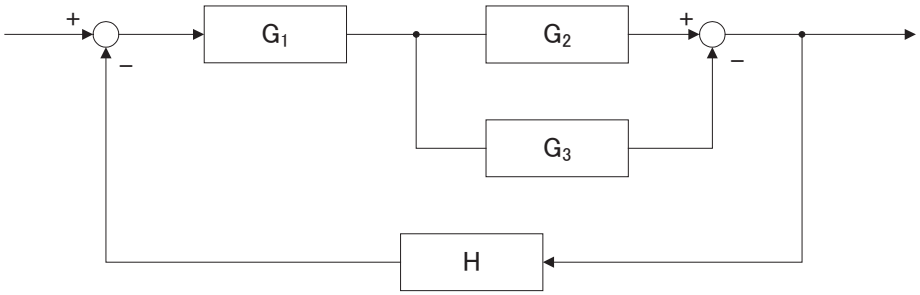
（３）



（ア） $\dfrac{1}{G - H}$　　（イ） $\dfrac{G}{H}$　　（ウ） $\dfrac{1}{1 + GH}$

（エ） $\dfrac{H}{G + GH}$　　（オ） $\dfrac{G}{1 + GH}$

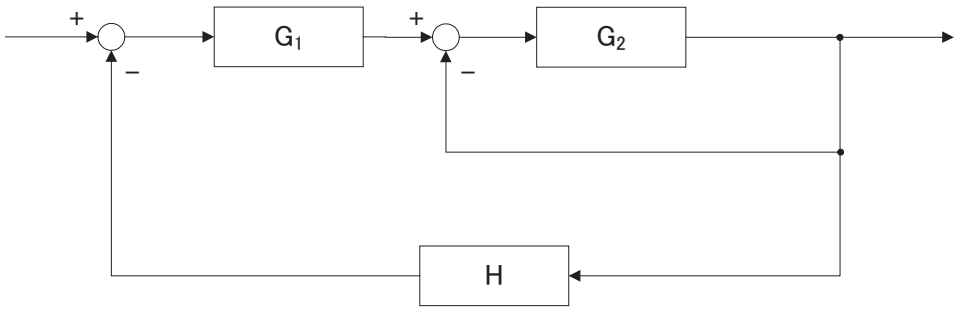
（４）



（ア） $\dfrac{G_1 + G_2 - G_3}{1 + H(G_1 + G_2 - G_3)}$　　（イ） $\dfrac{G_1 G_2 - G_3}{1 + G_1 G_2 G_3 H}$　　（ウ） $\dfrac{G_1 (G_2 - G_3)}{1 + G_1 H (G_2 - G_3)}$

（エ） $\dfrac{G_1 H (G_2 - G_3)}{G_1 + G_2 - G_3}$　　（オ） $\dfrac{H}{1 + G_1 H (G_2 - G_3)}$

（５）



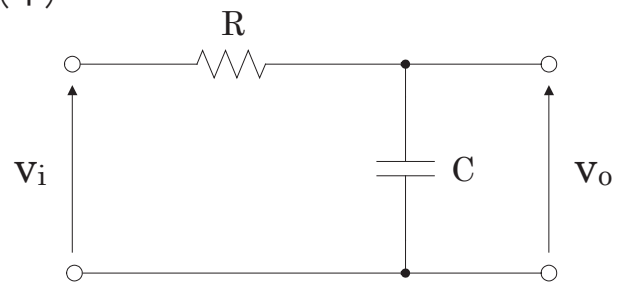
（ア） $\dfrac{G_1+G_2}{1+G_1+G_2H}$　　（イ） $\dfrac{G_1G_2}{1+G_2+G_1G_2H}$　　（ウ） $\dfrac{G_1G_2}{1+G_1G_2H}$

（エ） $\dfrac{G_1+G_2}{G_1+G_2H}$　　（オ） $\dfrac{H}{1+G_2+G_1G_2H}$

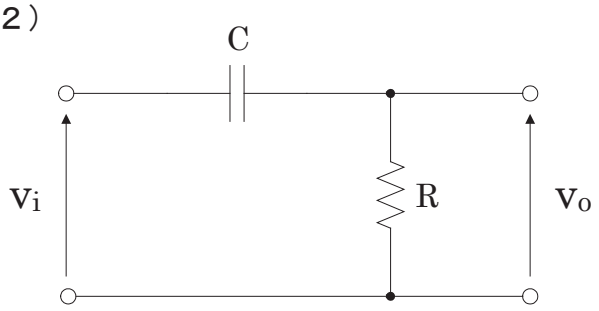
問題３

（１）、（２）の図に示すＲＣ回路において、それぞれの周波数伝達関数として正しいものを（ア）～（カ）から選択しなさい。

（１）



（２）



＜周波数伝達関数＞

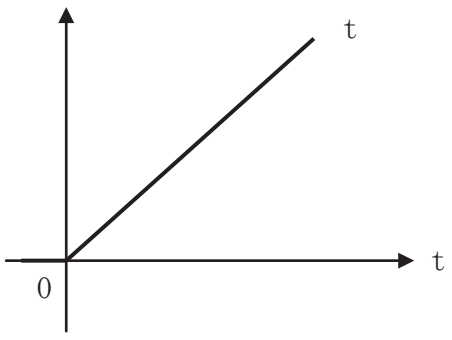
（ア） $\dfrac{j\omega C}{R+j\omega C}$　　（イ） $\dfrac{R}{R+j\omega C}$　　（ウ） $\dfrac{R}{1+j\omega CR}$

（エ） $\dfrac{j\omega CR}{1+j\omega CR}$　　（オ） $\dfrac{j\omega C}{R(1+j\omega C)}$　　（カ） $\dfrac{1}{1+j\omega CR}$

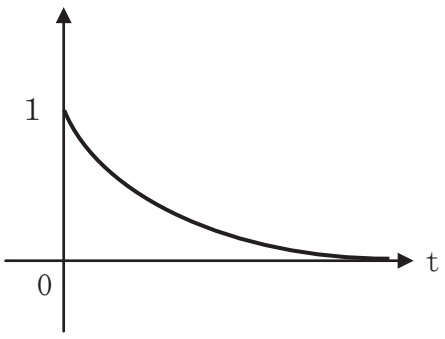
問題４

図に示す各関数の名称で適切なものを（ア）～（エ）から選択しなさい。

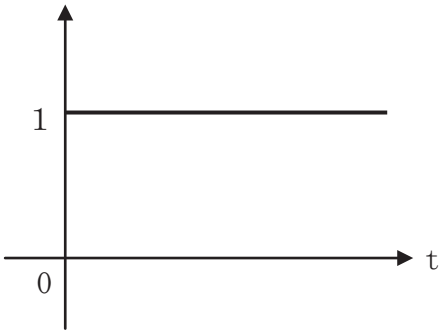
（１）



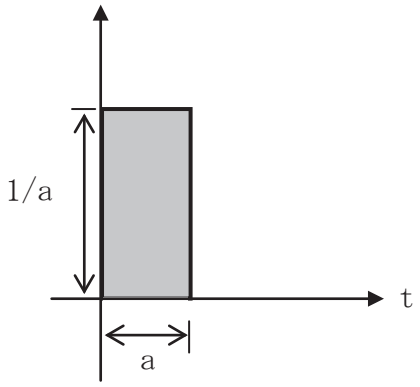
（２）



（３）



（４）



＜名称＞

（ア）デルタ関数　（イ）単位ステップ関数　（ウ）単位ランプ関数　（エ）指数関数

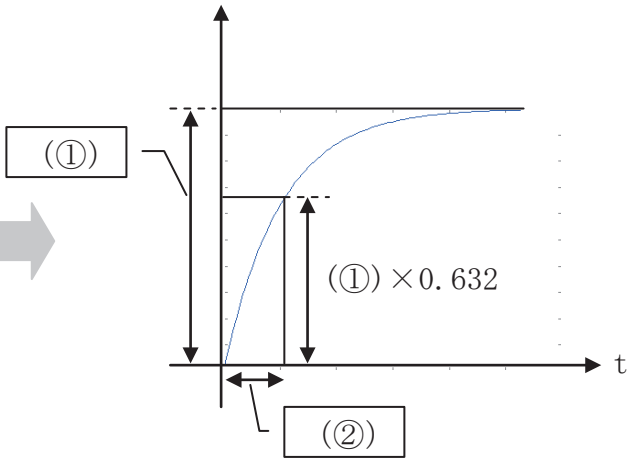
問題５

下記の図は、単位ステップ信号を各伝達関数へ入力したときの過渡応答について示したものである。（ａ）、（ｂ）、（ｃ）の各伝達関数の名称と、（①）～（⑦）に入る適切な値を（ア）～（チ）の中から選択しなさい。

単位ステップ信号

（　a　）
伝達関数

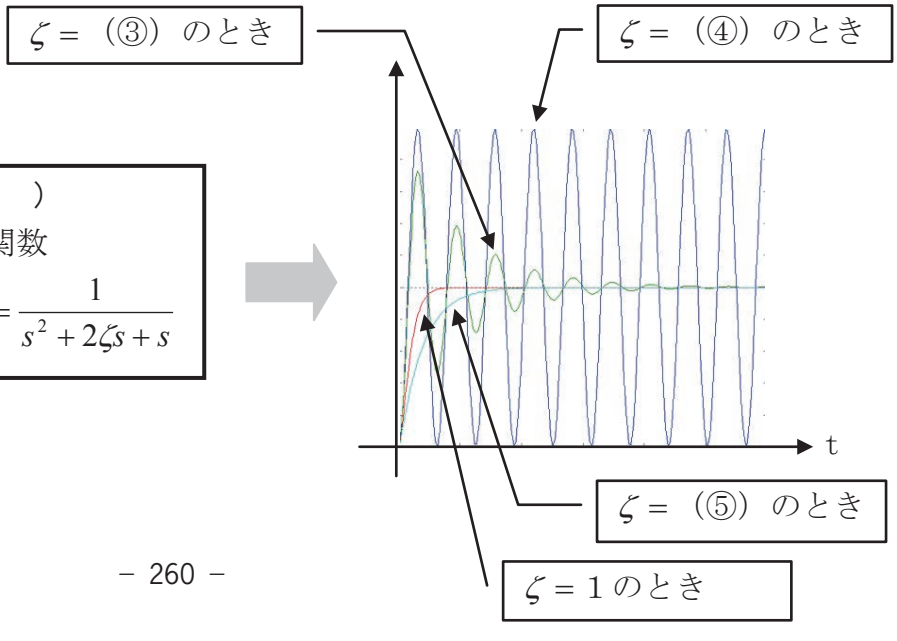
$$G(s)=\frac{K}{1+Ts}$$

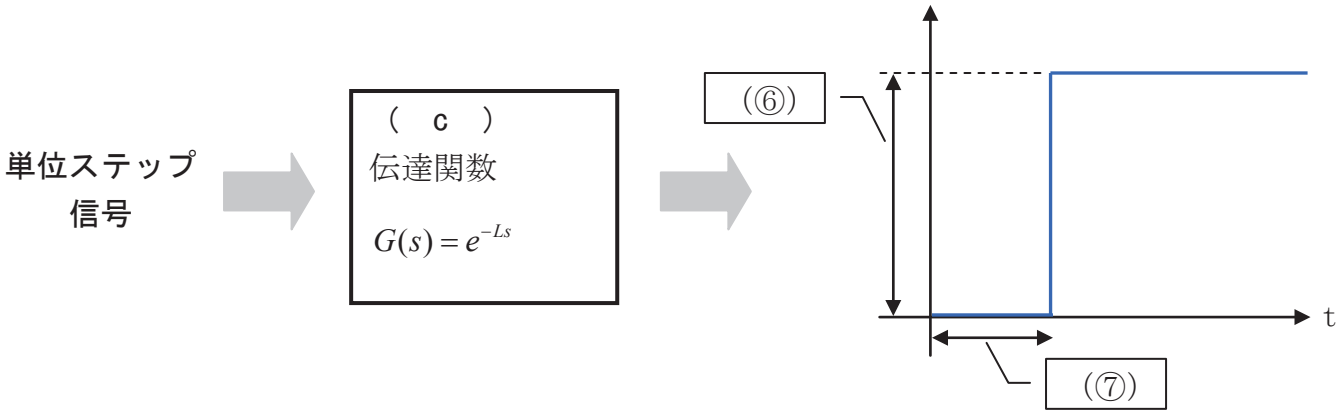


単位ステップ信号

（　b　）
伝達関数

$$G(s)=\frac{1}{s^2+2\zeta s+s}$$





＜名称＞

（ア）比例要素　　（イ）微分要素　　（ウ）積分要素

（エ）むだ時間要素　　（オ）１次遅れ要素　　（カ）２次遅れ要素

＜値＞

（キ）K　　（ク）$\frac{1}{K}$　　（ケ）KT　　（コ）T　　（サ）$\frac{1}{T}$

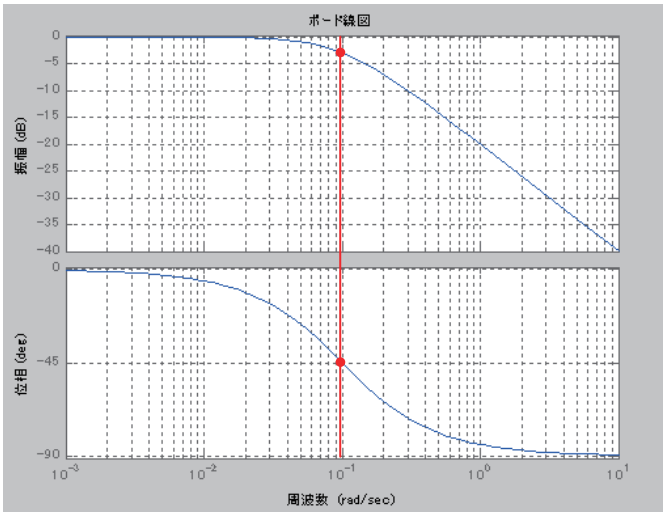
（シ）L　　（ス）$\frac{1}{L}$　　（セ）0　　（ソ）0.1　　（タ）1　　（チ）2

問題６

次の記述はボード線図に関して述べたものである。（　）の中に入る適切な語句を（ア）～（ケ）から選択しなさい。

下記の図は、一次遅れ要素の周波数伝達関数 $G(j\omega)=\frac{K}{1+j\omega T}$ のボード線図である。

このとき $K=1$、$T=10$ とする



ボード線図は各周波数ωに対して、ゲイン特性と（　①　）特性で表わされる。
ゲインは（　②　）[dB]　で求められる。
ωT＝１のときの角周波数を（　③　）と呼ぶ。このとき、ｇ＝－3.01[dB]となる。

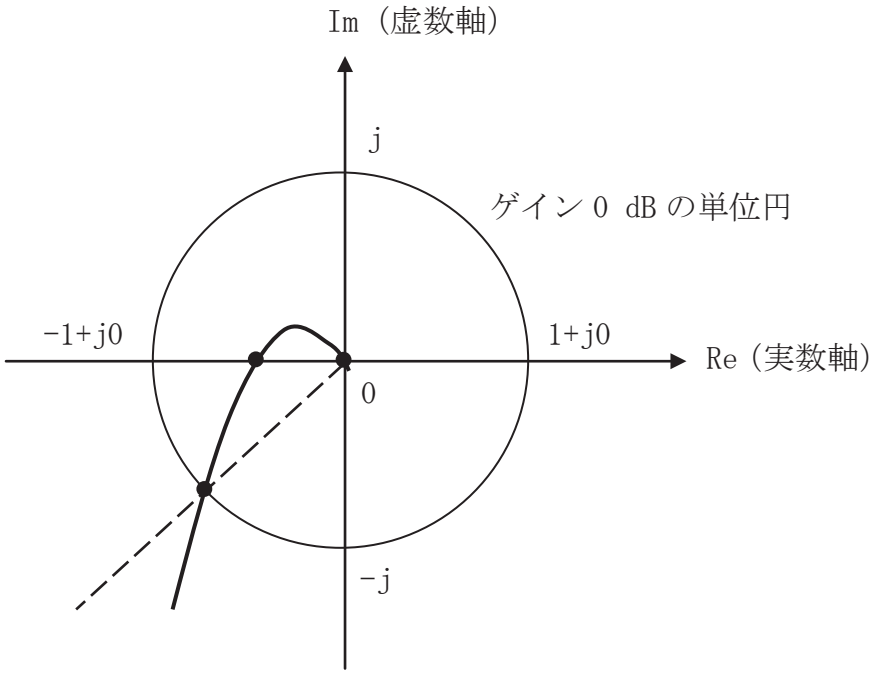
＜語句＞

（ア）過渡　　（イ）定常　　（ウ）位相　　（エ）$g = \log_{10}|G(j\omega)|$　　（オ）$g = 10\log_{10}|G(j\omega)|$

（カ）$g = 20\log_{10}|G(j\omega)|$　（キ）折点角周波数　（ク）振動角周波数　（ケ）固有角周波数

問題７

次の記述はナイキスト線図に関して述べたものである。（　）の中に入る適切な語句を（ア）～（タ）から選択しなさい。



ナイキスト線図とは（　①　）から周波数応答におけるベクトル軌跡を描いたもので、安定かどうか判別する方法である。

描いたナイキスト線図が（　②　）の点よりも（　③　）側を通る場合は不安定、（　④　）側を通る場合は安定であると判別される。

位相余裕とは、ナイキスト線図の（　⑤　）から（　②　）の点に達するまでの角度をいう。安定なシステムの場合、位相差が－180°より大きくなる。

ゲイン余裕とは、ナイキスト線図の（　⑥　）から原点までの距離をいう。安定なシステムの場合、位相差が－180°になる点ではゲインが0 dBより（　⑦　）。

＜語句＞

（ア）閉ループ伝達関数　　（イ）開ループ伝達関数　　（ウ）合成伝達関数
（エ）-j　　（オ）j　　（カ）1+j0　　（キ）-1+j0　　（ク）上　　（ケ）下
（コ）右　　（サ）左　　（シ）実数軸と交差する点　　（ス）虚数軸と交差する点
（セ）単位円と交差する点　　（ソ）大きくなる　　（タ）小さくなる

問題８

次の記述はPID制御に関して述べたものである。（　）の中に入る適切な語句を（ア）～（サ）から選択しなさい。

プロセス制御においてPID制御が多く用いられている。

比例動作（P 動作）のみによる制御では、オフセット（定常偏差）を生じる。比例帯を小さくするとオフセットは（　①　）なり、応答は（　②　）なり、不安定になる。

（　③　）（I 動作）はP 動作と組み合わせることでオフセットをなくすことができる。I 動作時間が長くなると応答は（　④　）なる。

（　⑤　）（D 動作）は出力の急激な変化を抑えることでできるので、D 動作を加えることによりPI 動作より（　⑥　）を上げることができる。

＜語句＞

（ア）大きく	（イ）小さく	（ウ）速く	（エ）遅く
（オ）加算動作	（カ）減算動作	（キ）微分動作	（ク）積分動作
（ケ）出力	（コ）精度	（サ）応答速度	

解答用紙

筆記課題「フィードバック制御に関する基礎知識」

入所年月	番号	氏名	合計点	評価判定
平成　　年　　月入所				

問題１（各２点）

（１）		（２）		（３）	
（４）		（５）		（６）	

問題２（各４点）

（１）		（２）		（３）	
（４）		（５）			

問題３（各４点）

（１）		（２）	

問題４（各２点）

（１）		（２）	
（３）		（４）	

問題５（各２点）

（a）		（b）		（c）	
（1）		（2）		（3）	
（4）		（5）		（6）	
（7）					

問題６（各２点）

（1）		（2）		（3）	

問題７（各２点）

（1）		（2）		（3）	
（4）		（5）		（6）	
（7）					

問題８（各２点）

（1）		（2）		（3）	
（4）		（5）		（6）	

筆記課題　解答及び解説

「フィードバック制御に関する基礎知識」

解答用紙

筆記課題「フィードバック制御に関する基礎知識」

入所年月	番号	氏名	合計点	評価判定
平成　　年　　月入所				A：80 点以上 B：60 点以上 80 点未満 C：60 点未満

問題１（各２点）

（1）	イ	（2）	ウ	（3）	ク
（4）	カ	（5）	キ	（6）	ケ

問題２（各４点）

（1）	イ	（2）	ア	（3）	オ
（4）	ウ	（5）	イ		

問題３（各４点）

（1）	カ	（2）	エ

問題４（各２点）

（1）	ウ	（2）	エ
（3）	イ	（4）	ア

問題５（各２点）

（a）	オ	（b）	カ	（c）	エ
（1）	キ	（2）	コ	（3）	ソ
（4）	セ	（5）	チ	（6）	タ
（7）	シ				

問題６（各２点）

（1）	ウ	（2）	カ	（3）	キ

問題７（各２点）

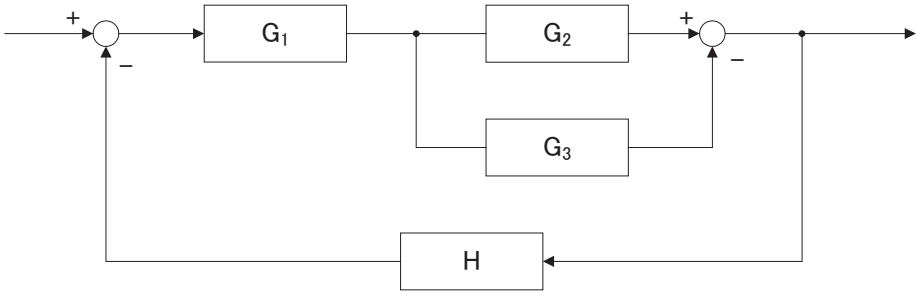
（1）	イ	（2）	キ	（3）	サ
（4）	コ	（5）	セ	（6）	シ
（7）	タ				

問題８（各２点）

（1）	イ	（2）	ウ	（3）	ク
（4）	エ	（5）	キ	（6）	サ

問題２

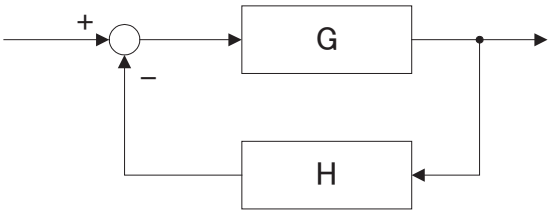
（４）



伝達関数G_1、G_2、G_3は、$G_1(G_2-G_3)$　と等価である。

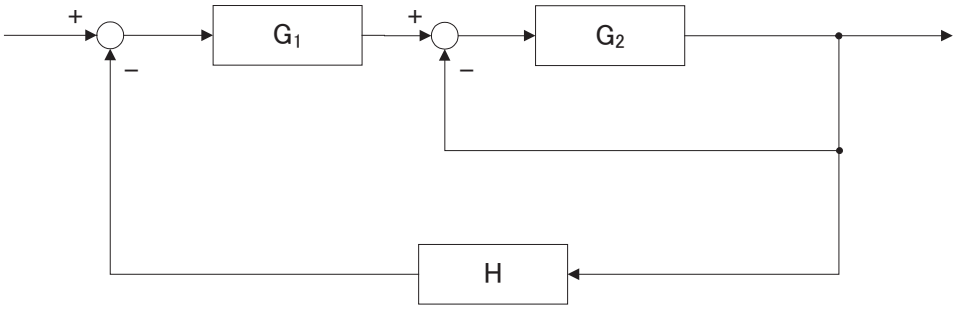
$G_1(G_2-G_3)=G$とおくと、

右の図と等価になるので、



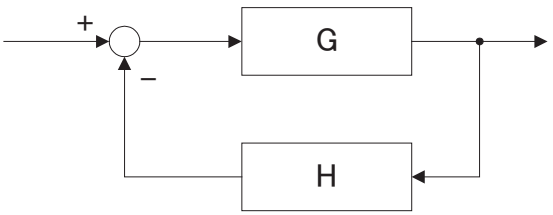
$$\frac{G}{1+GH}=\frac{G_1(G_2-G_3)}{1+G_1H(G_2-G_3)}$$　となる。

（５）



伝達関数G_1、G_2は、$\dfrac{G_1G_2}{1+G_2}$　と等価である。

$\dfrac{G_1G_2}{1+G_2}=G$とおくと、



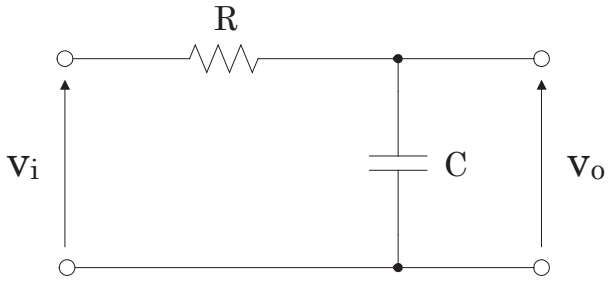
右の図と等価になるので

$$\frac{\dfrac{G_1G_2}{1+G_2}}{1+\dfrac{G_1G_2H}{1+G_2}}=\frac{G_1G_2}{1+G_2+G_1G_2H}$$　となる。

問題３

（１）

図より



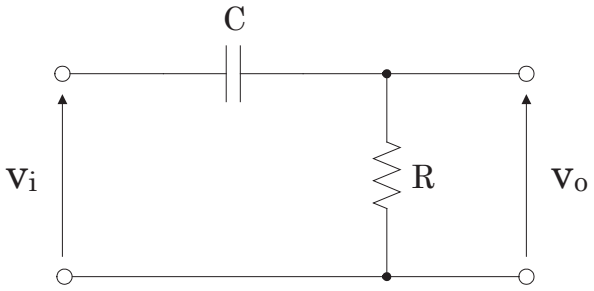
$$V_o = \frac{\dfrac{1}{j\omega C}}{\dfrac{1}{j\omega C} + R} V_i$$

よって、伝達関数は

$$G(j\omega) = \frac{V_o}{V_i} = \frac{\dfrac{1}{j\omega C}}{\dfrac{1}{j\omega C} + R} = \frac{1}{1 + j\omega CR}$$

（２）

図より



$$V_o = \frac{R}{\dfrac{1}{j\omega C} + R} V_i$$

よって、伝達関数は

$$G(j\omega) = \frac{V_o}{V_i} = \frac{R}{\dfrac{1}{j\omega C} + R} = \frac{j\omega CR}{1 + j\omega CR}$$

筆記課題

管理番号：E－48

Esub105　自家用電気設備工事A

「スケルトン読図に関する実技知識」

■課題概要■

高圧受変電設備の保守・点検に必要なスケルトンの読図について筆記試験により確認する。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領		
訓練課題	○	E-48-01_訓練課題 .docx
解答	○	E-48-02_解答及び解説.docx
作業工程手順書		
訓練課題確認シート		
評価要領		

筆記課題

「スケルトン読図に関する実技知識」

年　　月生	氏名	点

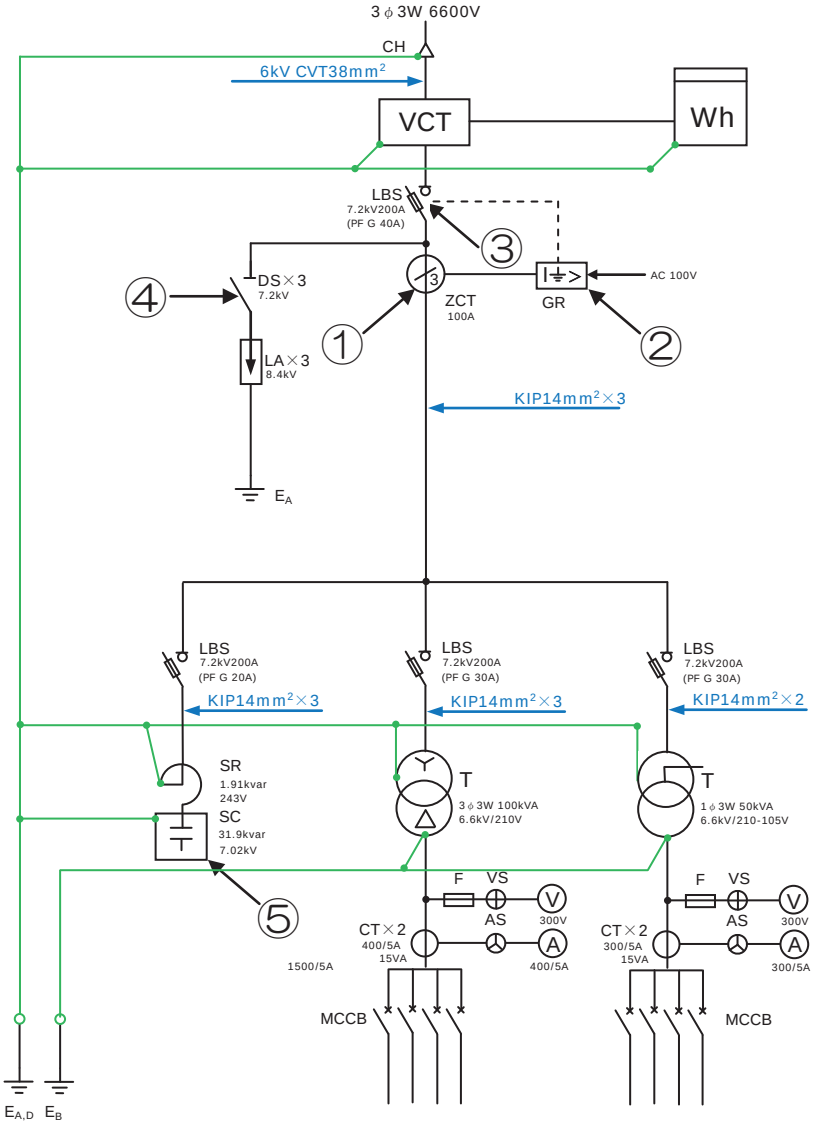
１　試験時間（点数）
　　３０分（１００点満点）

２　配付資料
　　問題用紙

３　注意事項
　（１）指導員の指示があるまで問題は見ないで下さい。
　（２）問題用紙に入所月、氏名を記入して下さい。
　（３）試験中質問があるときは挙手して下さい。

1．図は、キュービクル形高圧受電設備の単線結線図である。下図をもとに、問いに答えなさい。

＜（1）～（10）各3点、（11）①～⑤各4点、計50点＞



（１）受電設備の形態は？

イ．ＣＢ形　ロ．ＰＦ-ＣＢ形　ハ．ＰＦ-Ｓ形　ニ．PF-DS 形

（２）受電設備の容量[ｋＶＡ]は？

イ．150　ロ．200　ハ．２５０　ニ．３００

（３）高圧側で短絡事故が起きた際、短絡電流を遮断する機器は？

イ．ＣＢ　ロ．ＬＢＳ内のＳ　ハ．ＤＳ　ニ．ＬＢＳ内のＰＦ

（４）高圧側で地絡事故が起きた際、地絡電流を遮断する機器は？

イ．ＣＢ　ロ．ＬＢＳ内のＳ　ハ．ＤＳ　ニ．ＬＢＳ内のＰＦ

（５）電力需給用計器用変成器の２次側につながる機器は？

イ．電圧計　ロ．電流計　ハ．電力計　ニ．電力量計

（６）変圧器の外箱の接地工事の種類は？

イ．Ａ種接地工事　ロ．Ｂ種接地工事　ハ．Ｃ種接地工事　ニ．Ｄ種接地工事

（７）零相変流器を用いる目的は？

イ．零相電流を検出する　ロ．零相電圧を検出する　ハ．短絡電流を遮断する　ニ．地絡電流を遮断する

（８）高圧進相コンデンサを用いる目的は？

イ．高圧を低圧へ変圧する　ロ．力率を改善する　ハ．非常用として蓄電する　ニ．誘導雷対策として用いる

（９）電灯用変圧器の２次側に出力される配電方式は？

イ．三相３線式　ロ．三相４線式　ハ．単相２線式　ニ．単相３線式

（１０）キュービクル式受電設備は開放形受電設備と比較した場合、利点として誤っているものはどれか？

イ．現地工事の施工期間の短縮化が図れる　ロ．据付面積が小さく電気室の小型化が図れる　ハ．機器類が金属の箱へ収容されているので、安全性が高い　ニ．機器や配線が直接目視できるので、日常点検が容易である

（１１）図中の矢印で示す５つの機器の写真はどれか？

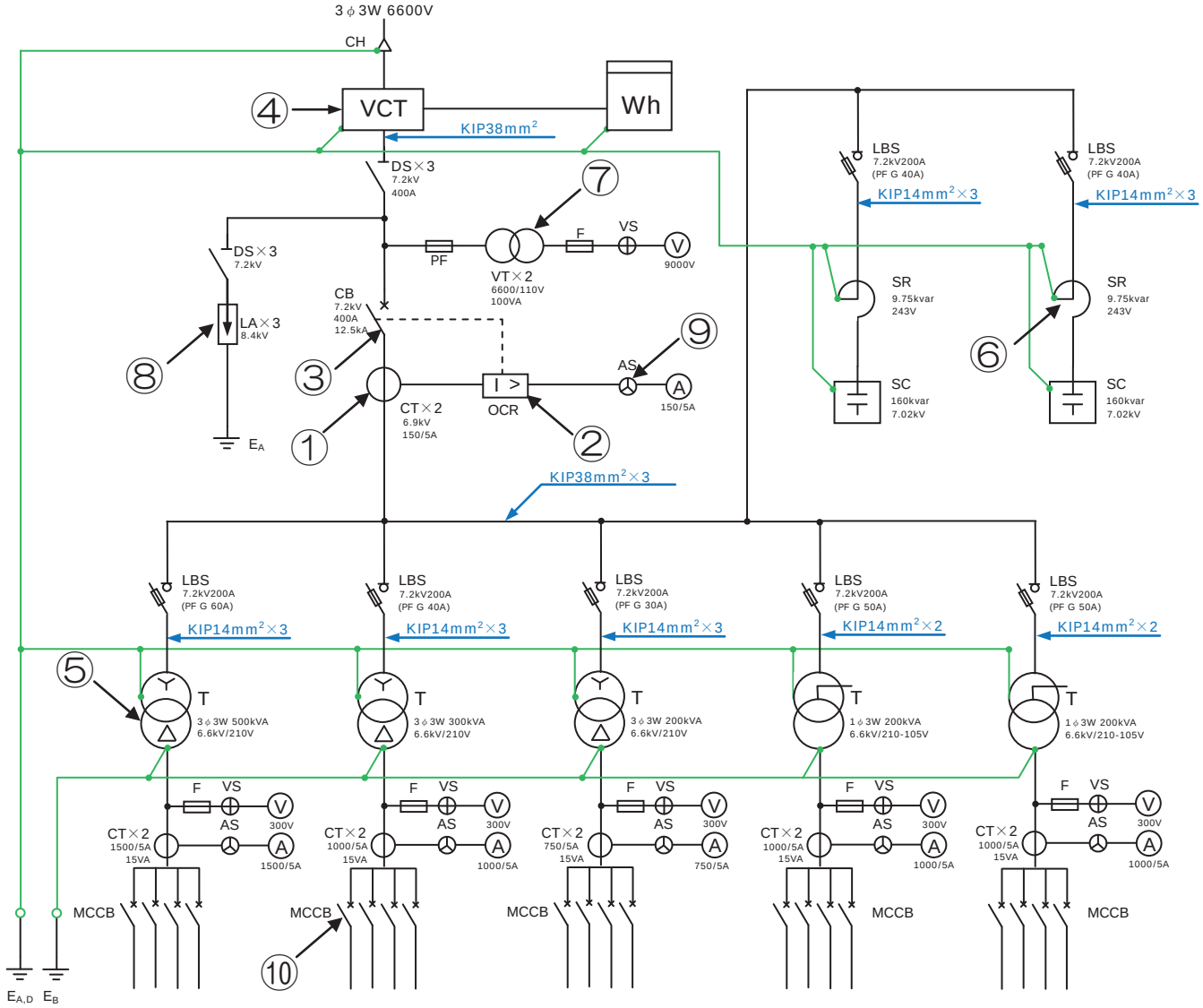
①（　　）②（　　）③（　　）④（　　）⑤（　　）

イ．　ロ．　ハ．　ニ．　ホ．

高圧地絡継電器

２．図は、開放形高圧受電設備の単線結線図である。図中の矢印で示す１０種類の機器の写真はどれか？

＜①～⑩各５点、計５０点＞



①（　　）	②（　　）	③（　　）	④（　　）	⑤（　　）
⑥（　　）	⑦（　　）	⑧（　　）	⑨（　　）	⑩（　　）

イ.

ロ.

ハ.

ニ.

ホ.

ヘ.

ト.

チ.

リ.

ヌ.

筆記課題

「スケルトン読図に関する実技知識」

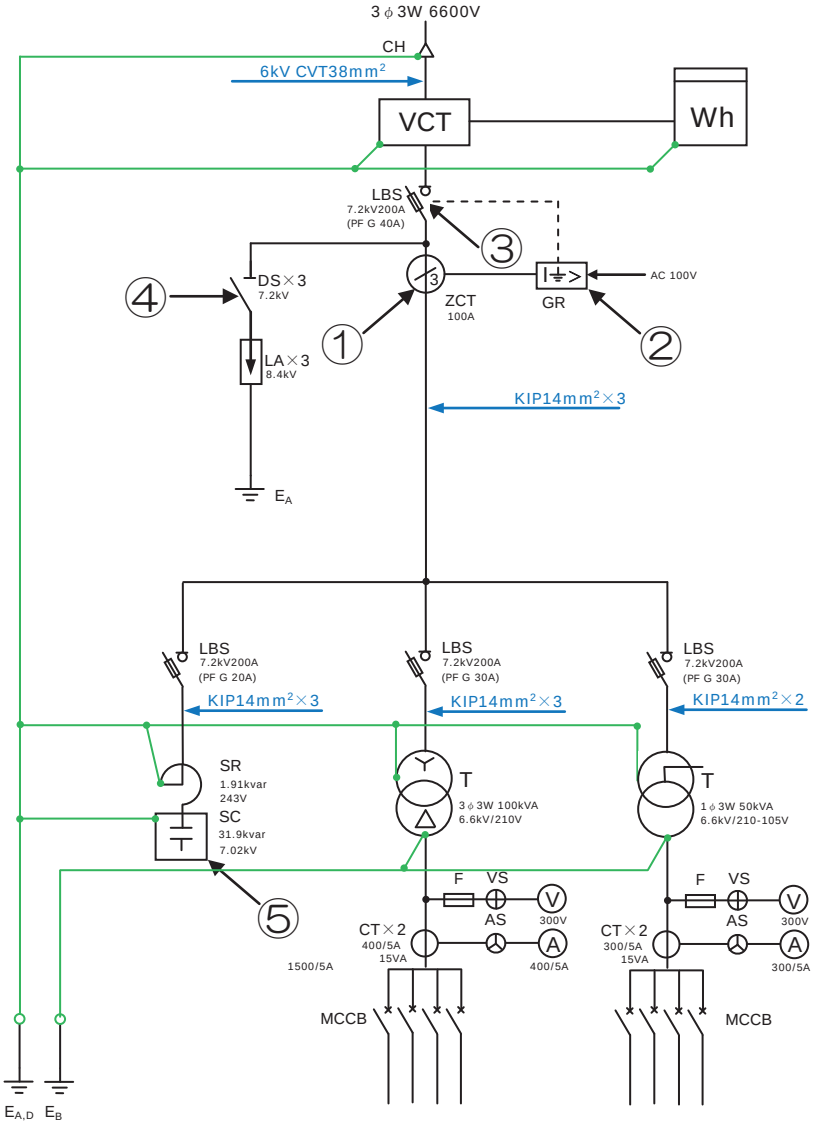
―解答及び解説―

１　試験時間（点数）
　　３０分（１００点満点）

２　配付資料
　　問題用紙

３　注意事項
（１）指導員の指示があるまで問題は見ないで下さい。
（２）問題用紙に入所月、氏名を記入して下さい。
（３）試験中質問があるときは挙手して下さい。

1．図は、キュービクル形高圧受電設備の単線結線図である。下図をもとに、問いに答えなさい。
＜（1）～（10）各3点、（11）①～⑤各4点、計50点＞



（１）受電設備の形態は？

イ．ＣＢ形　ロ．ＰＦ-ＣＢ形　ハ．ＰＦ-Ｓ形　ニ．PF-DS 形

（２）受電設備の容量[ｋＶＡ]は？

イ．150　ロ．200　ハ．２５０　ニ．３００

（３）高圧側で短絡事故が起きた際、短絡電流を遮断する機器は？

イ．ＣＢ　ロ．ＬＢＳ内のＳ　ハ．ＤＳ　ニ．ＬＢＳ内のＰＦ

（４）高圧側で地絡事故が起きた際、地絡電流を遮断する機器は？

イ．ＣＢ　ロ．ＬＢＳ内のＳ　ハ．ＤＳ　ニ．ＬＢＳ内のＰＦ

（５）電力需給用計器用変成器の２次側につながる機器は？※１

イ．電圧計　ロ．電流計　ハ．電力計　ニ．電力量計

（６）変圧器の外箱の接地工事の種類は？

イ．Ａ種接地工事　ロ．Ｂ種接地工事　ハ．Ｃ種接地工事　ニ．Ｄ種接地工事

（７）零相変流器を用いる目的は？

イ．零相電流を検出する　ロ．零相電圧を検出する　ハ．短絡電流を遮断する　ニ．地絡電流を遮断する

（８）高圧進相コンデンサを用いる目的は？

イ．高圧を低圧へ変圧する　ロ．力率を改善する　ハ．非常用として蓄電する　ニ．誘導雷対策として用いる

（９）電灯用変圧器の２次側に出力される配電方式は？※２

イ．三相３線式　ロ．三相４線式　ハ．単相２線式　ニ．単相３線式

（１０）キュービクル式受電設備は開放形受電設備と比較した場合、利点として誤っているものはどれか？

イ．現地工事の施工期間の短縮化が図れる　ロ．据付面積が小さく電気室の小型化が図れる　ハ．機器類が金属の箱へ収容されているので、安全性が高い　ニ．機器や配線が直接目視できるので、日常点検が容易である

（１１）図中の矢印で示す５つの機器の写真はどれか？

①（ロ）　②（ホ）　③（ニ）　④（イ）　⑤（ハ）

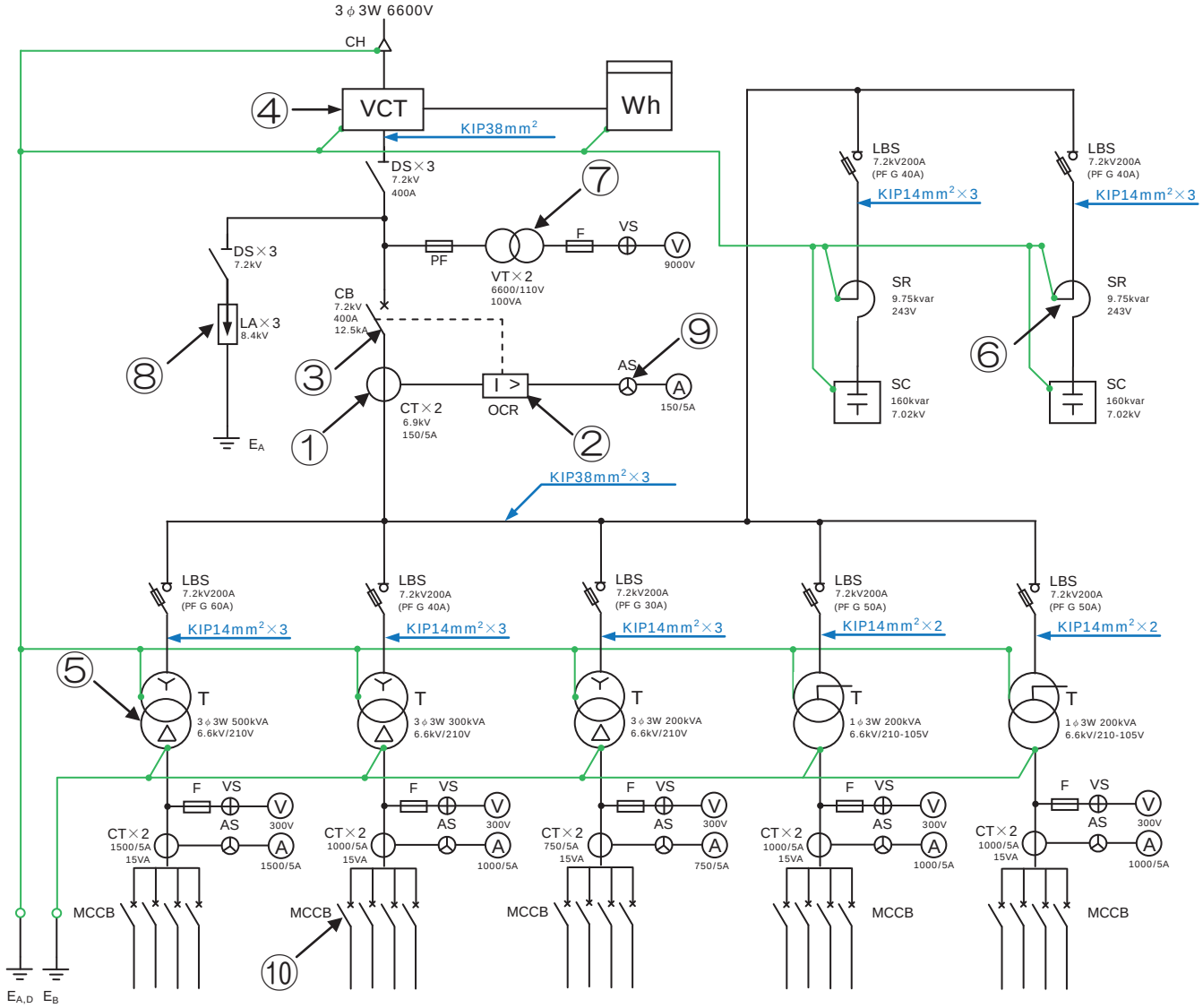
イ．　ロ．　ハ．　ニ．　ホ．

※１　通常、電力需給用複合計器（ＣＤＭ）が繋がる。

※２　図中の電灯用変圧器は、中性点引出し方式である。

２．図は、開放形高圧受電設備の単線結線図である。図中の矢印で示す１０種類の機器の写真はどれか？

＜①～⑩各５点、計５０点＞



①（ト）	②（イ※3）	③（リ）	④（ロ）	⑤（ハ）
⑥（チ）	⑦（ニ）	⑧（ヘ）	⑨（ホ）	⑩（ヌ）

イ.

ロ.

ハ.

ニ.

ホ.

ヘ.

ト.

チ.

リ.

ヌ.

※3　図は、誘導形の OCR である。静止形は下写真の通り。

筆記課題

管理番号：Ｅ－49

Esub105　自家用電気設備工事Ａ

「保護継電器試験に関する実技知識」

■課題概要■

高圧受変電設備の保守・点検に必要な保護継電器試験について筆記試験により確認する。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領		
訓練課題	○	E-49-01_訓練課題 .docx
解答	○	E-49-02_解答及び解説.docx
作業工程手順書		
訓練課題確認シート		
評価要領		

筆記課題

「保護継電器試験に関する実技知識」

年　　月生	氏名	点

１　試験時間（点数）
　　４０分（１００点満点）

２　配付資料
　　問題用紙

３　注意事項
　（１）指導員の指示があるまで問題は見ないで下さい。
　（２）問題用紙に入所月、氏名を記入して下さい。
　（３）試験中質問があるときは挙手して下さい。

1．各問の答えの欄の記号に○を付けなさい。

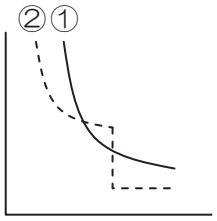
＜（1）～（6）各１０点、計６０点＞

（1） 短絡保護、地絡保護で使用する機器の系統で正しいものはどれか？ ただし順番は、検出機器-保護継電器-スイッチの順番とする。	イ. 短絡保護 ＣＴ-ＯＣＲ-ＣＢ 地絡保護 ＶＴ-ＧＲ-ＰＡＳ ロ. 短絡保護 ＣＴ-ＯＣＲ-ＰＡＳ 地絡保護 ＺＣＴ-ＤＧＲ-ＤＳ ハ. 短絡保護 CT-OCR-CB 地絡保護 ZCT とＺＰＤ-DGR-PAS ニ. 短絡保護 ZCT-GR-PAS 地絡保護 CT-OCR-PAS
（2） 定期点検時の停電操作において、操作する順番で正しいものは？	イ．DS→CB→PAS ロ．DS→PAS→CB ハ．CB→PAS→DS ニ．CB→DS→PAS
（3） 高圧受電設備の点検を行う際、短絡接地器具を用いた。正しい取り付け手順は。	イ．接地側を取付けた後、電路側を取付けた ロ．電路側の１つを取付け、接地側を取付けた後、残りの電路側を取付けた ハ．電路側を取付けた後、接地側を取付けた ニ．電路側のみ取付けた
（4） 受電電圧 6600〔V〕、受電電力 97〔kW〕、力率 80〔%〕の高圧需要家の受電用遮断器に用いる過電流継電器の適切なタップ値〔A〕は？ ただし、変流器の定格は 20/5〔A〕とし、タップ整定値は全負荷電流の 150〔%〕とする。	イ．３ ロ．４ ハ．５ ニ．６
（5） 地絡方向継電器の零相電圧の整定値が５〔%〕に設定されていた。この値を電圧〔V〕に変換した場合、正しいものは？ ただし、完全地絡が起きた場合の零相電圧は 3810〔V〕とする。	イ．95 ロ．190 ハ．285 ニ．381

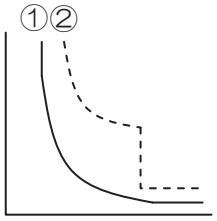
（6）
高圧受電設備（CB 形）と配電用変電所の過電流継電器との保護協調がとれているものは？

ただし、図中①の曲線は配電用変電所の過電流継電器動作特性を示し、②の曲線は高圧受電設備の過電流継電器動作特性+CB の遮断特性を示す。また、横軸は電流、縦軸は時間を表すものとする。

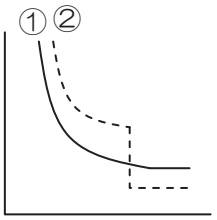
イ.



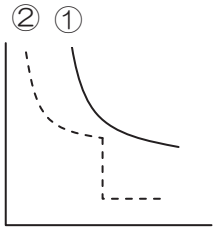
ロ.



ハ.



ニ.



２．図１は、「過電流継電器（電圧引外し方式）の連動試験」を示す。図中の継電器試験器からの結線図を完成させなさい。ただし、「高圧受変電設備結線図（抜粋）」を図２に、継電器試験器の構成を図３「継電器試験器構成図」に示す。

＜計４０点＞

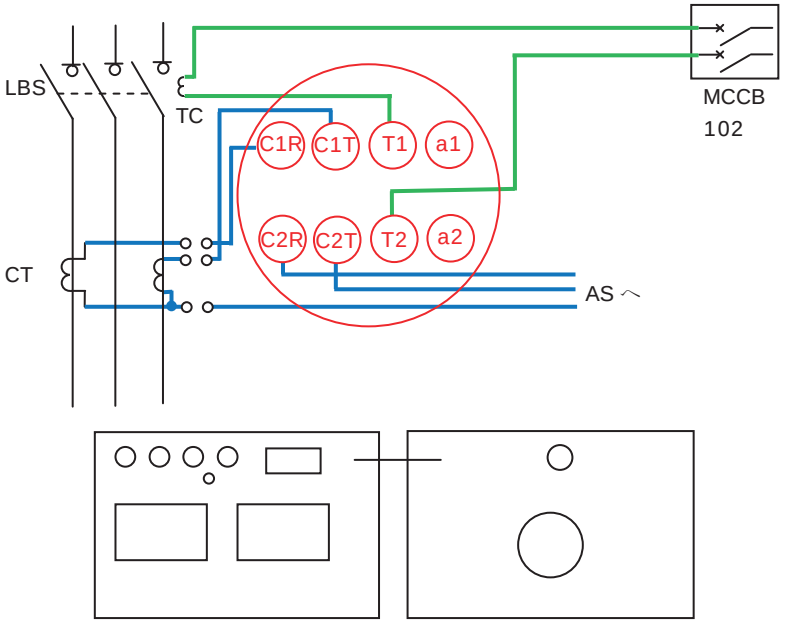


図１．「過電流継電器（電圧引外し方式）の連動試験」

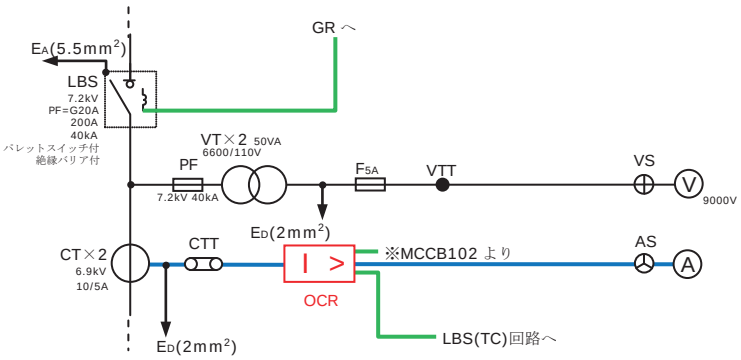


図 2.「高圧受変電設備結線図（抜粋）」

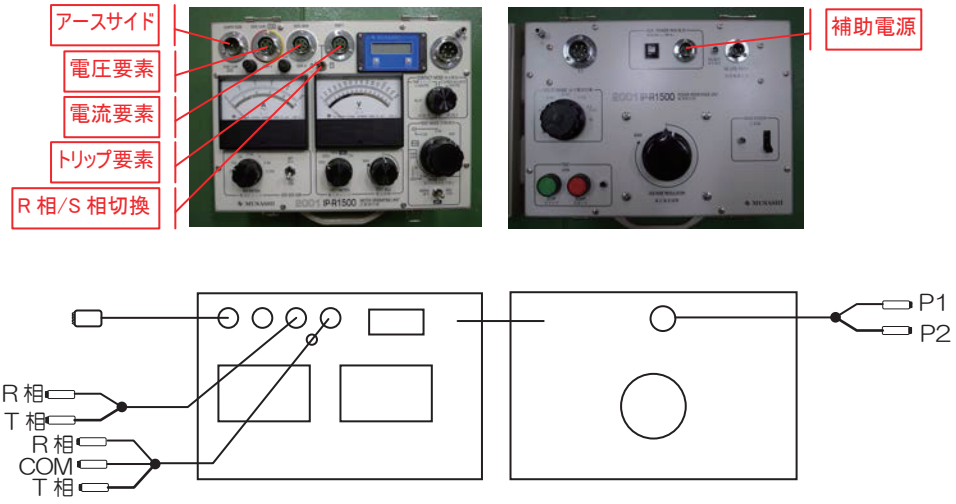


図 3.「継電器試験器構成図」

筆記課題

「保護継電器試験に関する実技知識」

―解答及び解説―

１　試験時間（点数）
　　４０分（１００点満点）

２　配付資料
　　問題用紙

３　注意事項
（１）指導員の指示があるまで問題は見ないで下さい。
（２）問題用紙に入所月、氏名を記入して下さい。
（３）試験中質問があるときは挙手して下さい。

1．各問の答えの欄の記号に○を付けなさい。

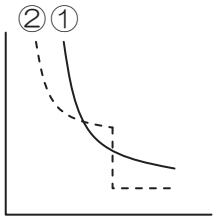
＜（1）～（6）各１０点、計６０点＞

（1） 短絡保護、地絡保護で使用する機器の系統で正しいものはどれか？ ただし順番は、検出機器-保護継電器-スイッチの順番とする。	イ. 短絡保護 ＣＴ-ＯＣＲ-ＣＢ 地絡保護 ＶＴ-ＧＲ-ＰＡＳ ロ. 短絡保護 ＣＴ-ＯＣＲ-ＰＡＳ 地絡保護 ＺＣＴ-ＤＧＲ-ＤＳ ハ. 短絡保護 CT-OCR-CB 地絡保護 ZCT とＺＰＤ-DGR-PAS ニ. 短絡保護 ZCT-GR-PAS 地絡保護 CT-OCR-PAS
（2） 定期点検時の停電操作において、操作する順番で正しいものは？	イ．DS→CB→PAS ロ．DS→PAS→CB ハ．CB→PAS→DS ニ．CB→DS→PAS
（3） 高圧受電設備の点検を行う際、短絡接地器具を用いた。正しい取り付け手順は。	イ．接地側を取付けた後、電路側を取付けた ロ．電路側の１つを取付け、接地側を取付けた後、残りの電路側を取付けた ハ．電路側を取付けた後、接地側を取付けた ニ．電路側のみ取付けた
（4） 受電電圧 6600〔V〕、受電電力 97〔kW〕、力率 80〔%〕の高圧需要家の受電用遮断器に用いる過電流継電器の適切なタップ値〔A〕は？ ただし、変流器の定格は 20/5〔A〕とし、タップ整定値は全負荷電流の 150〔%〕とする。	イ．３ ロ．４ ハ．５ ニ．６
（5） 地絡方向継電器の零相電圧の整定値が５〔%〕に設定されていた。この値を電圧〔V〕に変換した場合、正しいものは？ ただし、完全地絡が起きた場合の零相電圧は 3810〔V〕とする。	イ．95 ロ．190 ハ．285 ニ．381

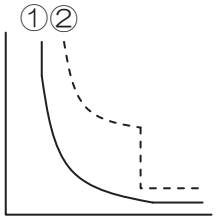
（6）
高圧受電設備（CB 形）と配電用変電所の過電流継電器との保護協調がとれているものは？

ただし、図中①の曲線は配電用変電所の過電流継電器動作特性を示し、②の曲線は高圧受電設備の過電流継電器動作特性+CB の遮断特性を示す。また、横軸は電流、縦軸は時間を表すものとする。

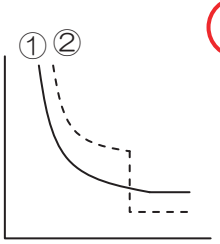
イ.



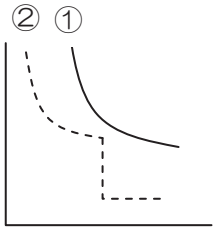
ロ.



ハ.



ニ.



２．図１は、「過電流継電器（電圧引外し方式）の連動試験」を示す。図中の継電器試験器からの結線図を完成させなさい。ただし、「高圧受変電設備結線図（抜粋）」を図２に、継電器試験器の構成を図３「継電器試験器構成図」に示す。

＜計４０点＞

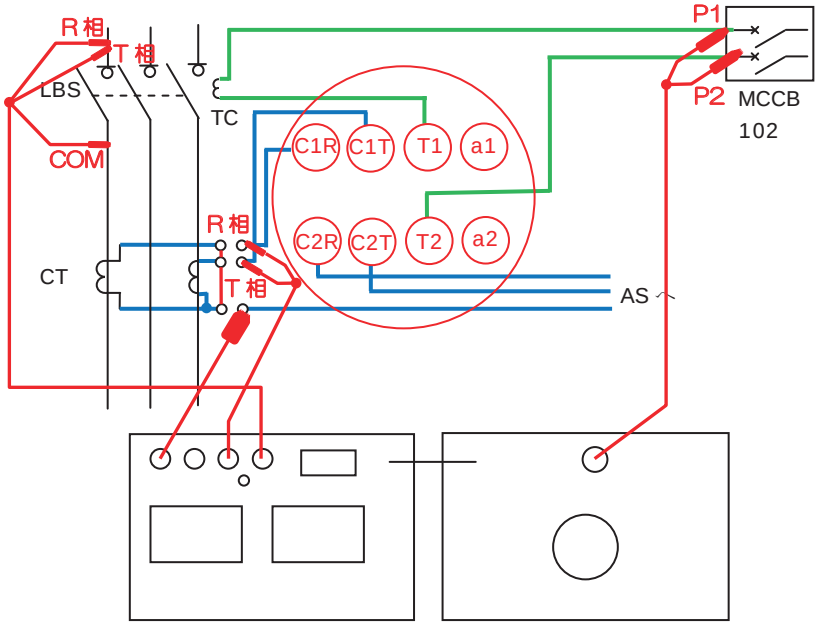


図１．「過電流継電器（電圧引外し方式）の連動試験」

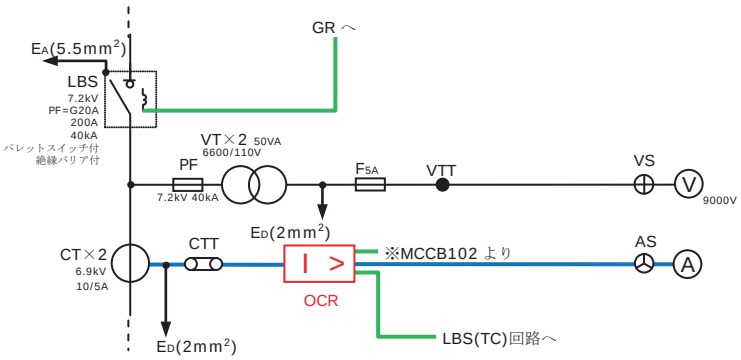


図 2.「高圧受変電設備結線図（抜粋）」

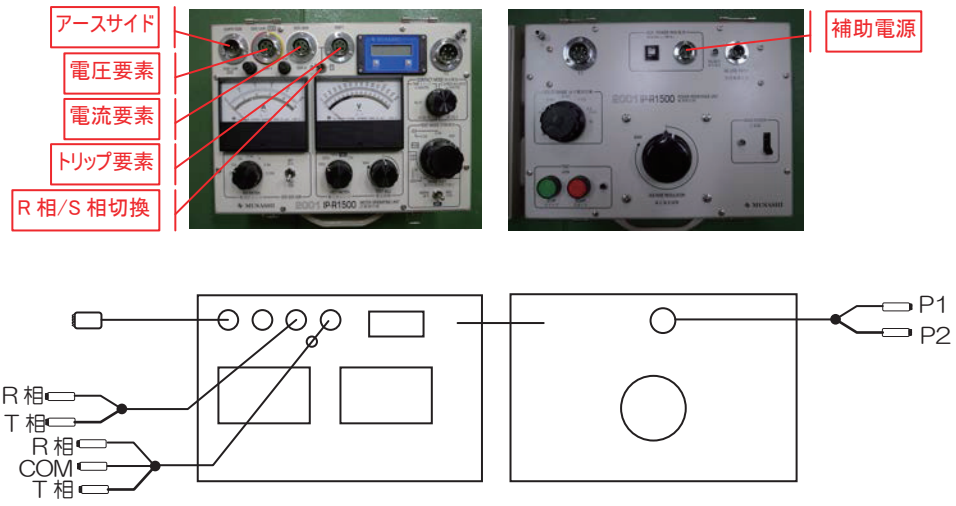


図 3.「継電器試験器構成図」

実技課題

管理番号: E-50

「CADによる屋内配線図の作成」

■課題概要■

CAD製図のために必要な、CAD基本操作、電気設備の技術基準、安全衛生作業等を習得しているか実技により確認します。

■訓練課題資料構成■

資料名		ファイル名
訓練課題実施要領	○	訓練課題(実技)実施要領.doc
訓練課題	○	訓練課題(配布用).JWW
訓練課題(課題図面)	○	訓練課題(完成).JWW
配布図形	○	訓練課題用図形(100)
作業工程手順書	○	訓練課題_作業工程計画書(問題).docx
作業工程手順書解答例	○	訓練課題_作業工程計画書(解答例).docx
訓練課題確認シート	○	訓練課題確認シート及び評価要領.xlsx
評価要領	○	訓練課題確認シート及び評価要領.xlsx

実技課題

「CADによる屋内配線図の作成」

１　作業時間

　　120分

２　配付資料

　ペーパー資料：訓練課題（完成）.jww

　電子ファイル：訓練課題（配布用）.jww、訓練課題用図形（100）

３　注意事項

　（1）指導員の指示があるまで問題は見ないでください。

　（2）課題図面の図面枠の設計者名に科名、入所期、氏名を記入してください。

　（3）携帯電話の使用は不可です。

　（4）試験中、質問等があるときは挙手してください。

実技課題「CAD による屋内配線図の作成」実施要領

実施要領

これは２階建て木造住宅の１階部分の平面図である。次の作図条件に従って、ペーパーで配布された課題図面を、CAD システムを活用して作図し、完成させなさい。

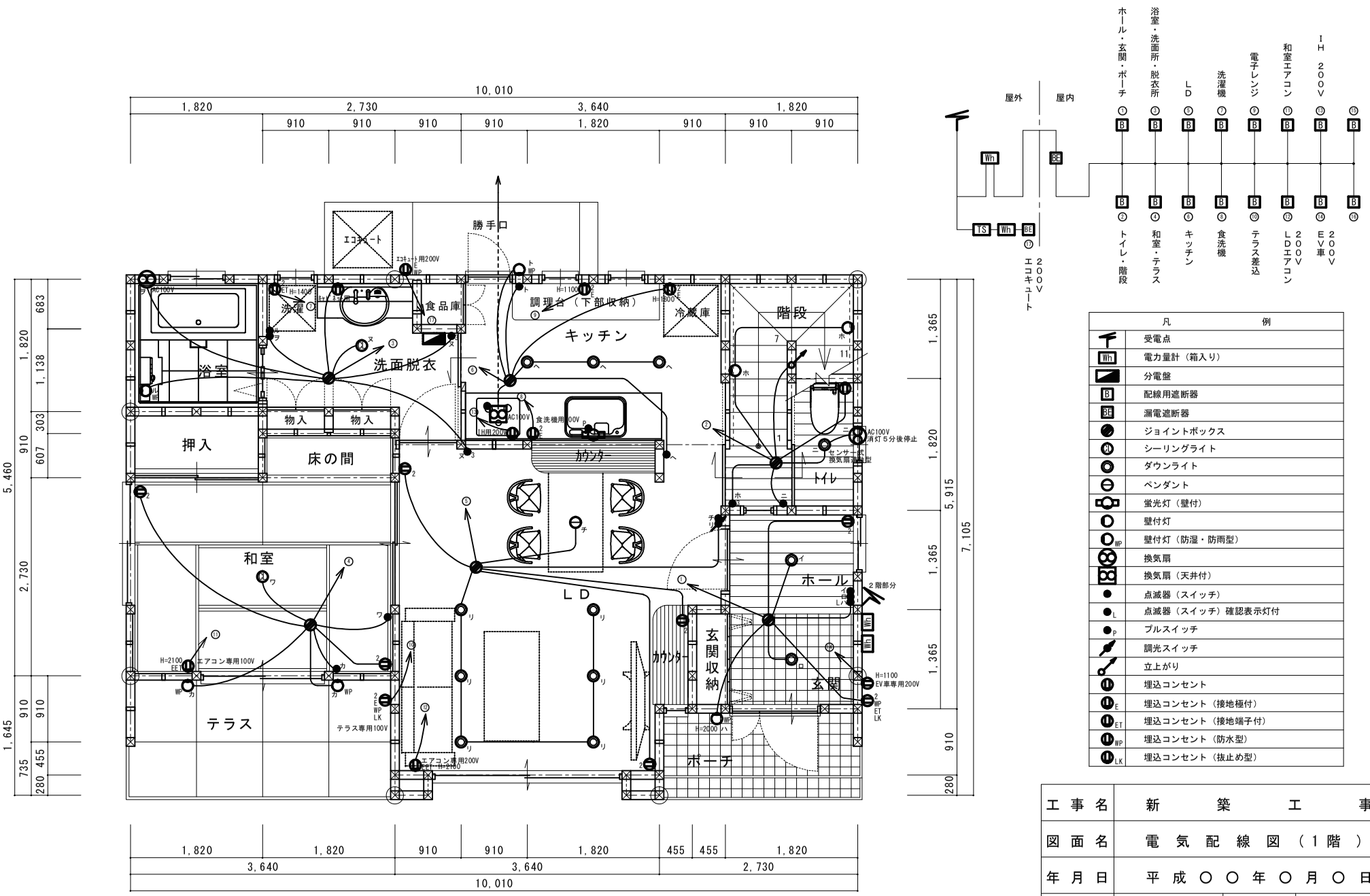
〈作図条件〉

1. 事前に共有フォルダ等で配布する「訓練課題（配布用）.JWW」に不足している部分を追加する。
2. 追加部分は「ＬＤ、和室・テラス、洗面脱衣・浴室」の電気設備配線、分電盤結線図および凡例であり、作図条件 5. で示すレイヤにそれぞれ作図すること。
3. 環境設定ファイル（環境設定（訓練課題）.JWF）から環境設定読込みを行ってから作業を開始する。
4. 電気設備については、はレイヤグループ１に作図することとし、
   用紙サイズは　A３　縮尺は 1：100 とする（注：配布時に設定されている為変更の必要はない）
5. レイヤグループ 0 とレイヤグループ 1 の名称は建築平図面（１階）、電気設備図として設定してあり、レイヤグループ 1 のレイヤ番号は以下の通りとし、レイヤの名称を記入すること
   ⓪ 器具
   ① スイッチ
   ② コンセント
   ③ 文字
   ④ 電気配線
   ⑤ 分電盤結線図
   ⑥ 凡例
   ⑦ 図面枠
   ⑧ 補助線（使用しなくても良い）
6. 図面枠内は以下のように入力すること。ただし、文字種は　４　とすること。
   ① 工事名　　新築工事
   ② 図面名　　電気配線図(1 階)
   ③ 年月日　　平成○○年○月○日
   ④ 尺度　　1：100
   ⑤ 図面番号　1/2
   ⑥ 会社名　　ポリテクセンター○○
   ⑦ 設計者名　科名　入所月　氏名
7. シーリングライトの図記号を円(線種 7・直径　300㎜)と文字(文字種　２)で作成し、基点を中心点とし、図形名称を CL.jws として図形を登録すること。
8. 配線図は、作図条件 7. 以外の図形はあらかじめ配布された「訓練課題用図形（100)」の図形を用いて作図すること。
9. 電気配線は「線色 6」を使用すること。

注意事項

1. テキスト、ノート、補助資料等は参考にしても構わない。
2. あらかじめ記入されている図は変更しないこと。
4. 電線の種類と条数の記入は不要とする。
5. 照明器具と点滅器には符号を付け対応が分かるようにすること。

（参考：イロハニホヘト チリヌルヲ ワカヨタレ ソツネナラム・・・）

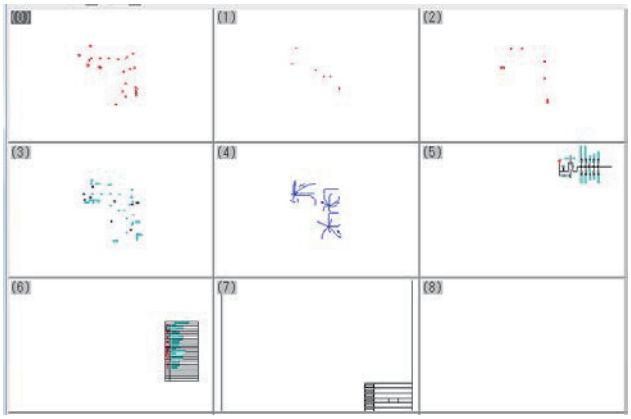
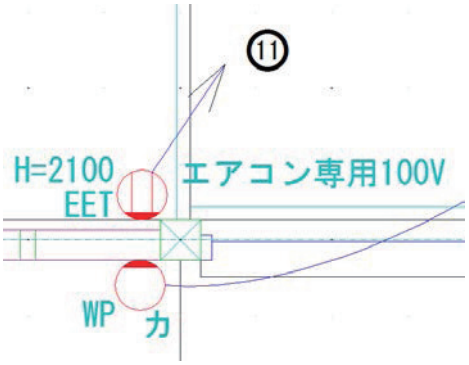


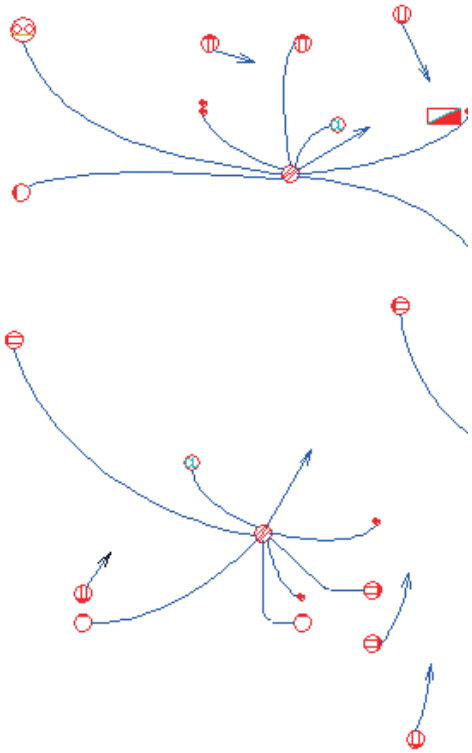
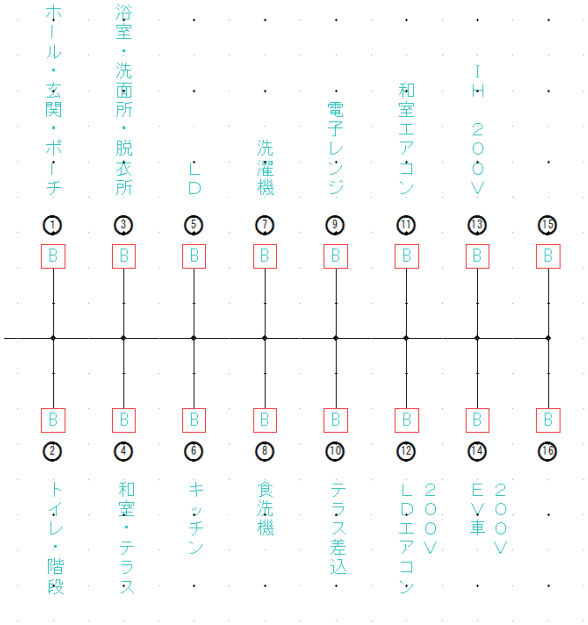
凡例	
	受電点
Wh	電力量計（箱入り）
	分電盤
B	配線用遮断器
BE	漏電遮断器
	ジョイントボックス
	シーリングライト
	ダウンライト
	ペンダント
	蛍光灯（壁付）
	壁付灯
WP	壁付灯（防湿・防雨型）
	換気扇
	換気扇（天井付）
	点滅器（スイッチ）
L	点滅器（スイッチ）確認表示灯付
P	プルスイッチ
	調光スイッチ
	立上がり
	埋込コンセント
E	埋込コンセント（接地極付）
ET	埋込コンセント（接地端子付）
WP	埋込コンセント（防水型）
LK	埋込コンセント（抜止め型）

工事名	新築工事		
図面名	電気配線図（1階）		
年月日	平成○○年○月○日		
尺度	1：100	図面番号	1／2
会社名	ポリテクセンター○○		
設計者名	○○○○科○月入所生　氏名		

作業工程計画書

（　　）内に欄外の語句から選択し工程を完成させよ。

作成手順	ポイント（留意事項等）	参考資料（写真、図面等）
電気設備器具の配置	照明器具、スイッチ、コンセント等の器具を部屋の中心や壁に付けること等を意識して配置する。 その際、（　①　）を分けて配置すること。	
（　②　）の入力 （レイヤは 1-3とする）	器具の種類や用途を入力し、照明と点滅器には符号を付け対応が分かるようにすること。 （参考：イロハニホヘト　チリヌルヲワカヨタレ　ソツネナラム・・・）（文字種は2を使用する）	

作成手順	ポイント（留意事項等）	参考資料（写真、図面等）
（　③　）の作成 （レイヤは 1-4 とする）	曲線や直線、面取り等のコマンドを使用し、照明器具、スイッチ、コンセント等の器具を配線する。 （配線は線種6を使用する）	
（　④　）の作成 （レイヤは 1-5 とする）	平面図に記載された回路番号と対応するように作成すること。 （文字は文字種2、回路番号は文字種3、結線は線色2を使用する）	

作成手順	ポイント（留意事項等）	参考資料（写真、図面等）
（　⑤　）の作成 （レイヤは 1-6 とする）	図記号と名称が対応するように作成すること。 （文字種は2を使用する）	

＜語句選択肢＞

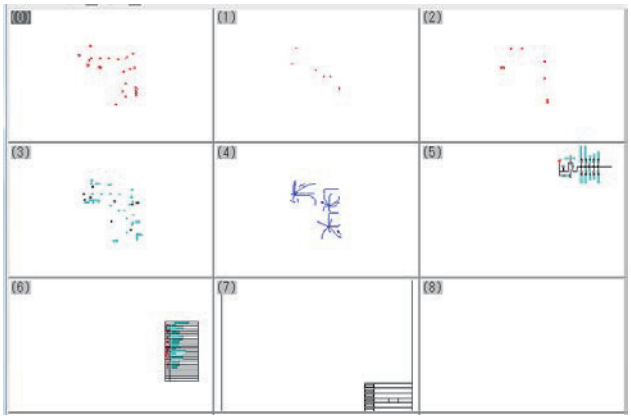
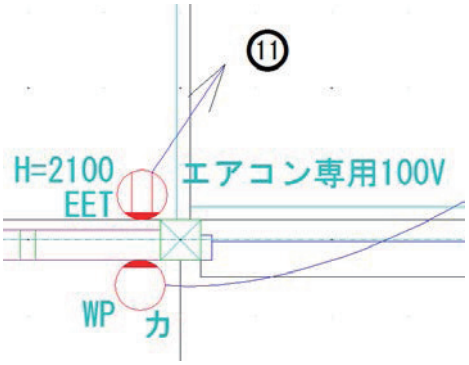
大型機器、凡例表、分電盤結線図、容量、文字、用途、平面図、レイヤ、レイヤグループ、電気配線図

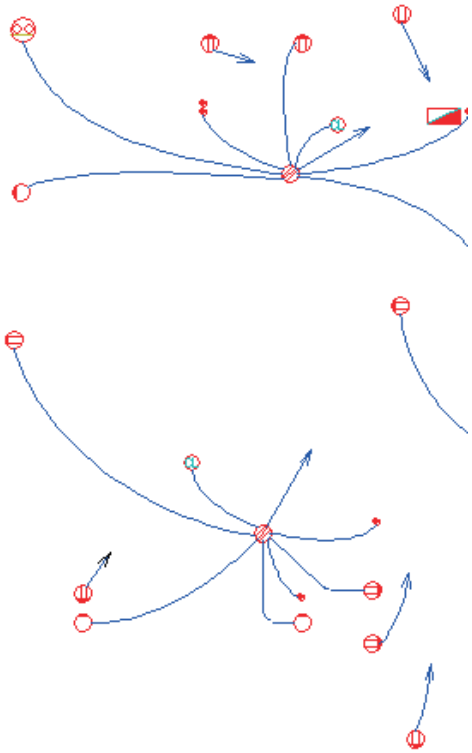
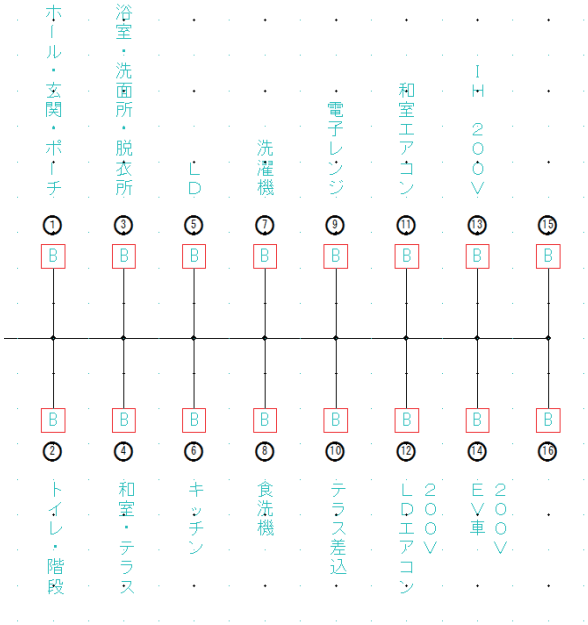
解答欄

①	
②	
③	
④	
⑤	

作業工程計画書（解答例）

（　　）内に欄外の語句から選択し工程を完成させよ。

作成手順	ポイント（留意事項等）	参考資料（写真、図面等）
電気設備器具の配置	照明器具、スイッチ、コンセント等の器具を部屋の中心や壁に付けること等を意識して配置する。 その際、（レイヤ）を分けて配置すること。	
（文字）の入力 （レイヤは 1-3とする）	器具の種類や用途を入力し、照明と点滅器には符号を付け対応が分かるようにすること。 （参考：イロハニホヘト チリヌルヲワカヨタレ ソツネナラム・・・）（文字種は2を使用する）	

作成手順	ポイント（留意事項等）	参考資料（写真、図面等）
（電気配線図）の作成 （レイヤは 1-4 とする）	曲線や直線、面取り等のコマンドを使用し、照明器具、スイッチ、コンセント等の器具を配線する。 （配線は線種6を使用する）	
（分電盤結線図）の作成 （レイヤは 1-5 とする）	平面図に記載された回路番号と対応するように作成すること。 （文字は文字種2、回路番号は文字種3、結線は線色2を使用する）	

作成手順	ポイント（留意事項等）	参考資料（写真、図面等）
（凡例表）の作成 （レイヤは 1-6 とする）	図記号と名称が対応するように作成すること。 （文字種は2を使用する）	

＜語句選択肢＞

大型機器、凡例表、分電盤結線図、容量、文字、用途、平面図、レイヤ、レイヤグループ、電気配線図

訓 練 課 題 確 認 シ ー ト

氏名		訓練課題名	CADによる図面作成（電気設備図面）
入所月		訓練科名	電気設備科
実施日		訓練目標	電気設備設計図書に必要な知識と技術を習得する。
訓練課題のねらい 1. CADにより、図面が作成できること。 2. レイヤの使い分けができること。 3. 電気設備の技術基準に基づく設計・製図ができること。		訓練科目と内容 仕事との関連	

評価する能力等	評価区分	評価項目	細　目	評価（数値）					評価判定	評価基準
1. コマンドが使用できること。 2. 図面作成手順を理解していること。 3. 時間内に作図できること。	作業時間	作業時間	経過時間毎に減点	0	4	6	8	10		標準時間2H　打ち切り時間3H （15分経過毎に2点減点）
1. 工程が理解できること。 2. 作業工程計画書が時間内に記入できること。	工程計画	工程計画書	工程に誤りがないこと※別紙	0	2	∽	8	10		誤り1箇所につき、2点減点する。（全5問）
CADによる図面の作成ができること。		接続線	器具までの未到達や突抜けがないこと	1	2	3	4	5		持ち点を5点とし、不適切な箇所については、スイッチやコンセント等の器具の種類につき1点ずつ減点する。 中心点や線上へ配置していないものや整列していないものがあれば1点減点。
		図記号	大きさ、用途が正しいこと	1	2	3	4	5		
		配線（施工条件）	線種、施工法が正しいこと	1	2	3	4	5		
		配置(壁付きの器具)	壁に接していること	1	2	3	4	5		
		線色	指定された線色であること	1	2	3	4	5		
		不要線又は不足線の有無	不要線又は不足線があるか	1	2	3	4	5		
指定されたレイヤに作図されていること		指定のレイヤであるか	レイヤが正しいこと	1	2	3	4	5		指定されたレイヤ以外に作図されていれば、複数個あっても、1レイヤ毎に1点減点とする。
図面枠の作図ができていること		氏名、図面名称、日付、縮尺	氏名等が記入されていること	1	2	3	4	5		完成していれば5点、未完成であれば3点、作図していない場合は0点とする。 文字列が整列していない場合は2点減点とする。
照明レイアウト図が作成されていること		照明レイアウト図	完成されていること	1	2	3	4	5		完成していれば5点、未完成であれば3点、作図していない場合は0点とする。
電灯配線図が作成されていること		電灯配線図	完成されていること	1	2	3	4	5		
コンセント設備図が作成されていること		コンセント設備図	完成されていること	1	2	3	4	5		
凡例表が作成されていること		凡例表	完成されていること	1	2	3	4	5		
分電盤図が作成されていること		分電盤図	完成されていること	1	2	3	4	5		
印刷ができること。		印刷のレイアウト	ズレによる印刷漏れの無いこと	1	2	3	4	5		著しいズレにより、用紙に表記できない部分がある場合は0点とする。
安全作業ができること。	安全作業	安全作業	他の作業者への妨げ行為等	1	2	3	4	5		持ち点を5点とし、不適切な作業又は行為がある毎に1点ずつ減点する。
		VDT作業	VDTに配慮した作業	1	2	3	4	5		

コメント	実技課題の評価			
		合計得点／満点	0 ／ 100	＜判定表＞ A：80点以上　：到達水準を十分に上回った B：60点以上80点未満　：到達水準に達した C：60点未満　：到達水準に達しなかった
		換算点	0.00 ／ 100	
担当指導員 氏名：		評価		＜算式＞ 換算点　＝（合計得点 ／ 満点（100））× 100
評価担当者 氏名：				

評価要領

訓練課題名	CADによる図面作成（電気設備図面）
科名	電気設備科

評価	評価項目	細　目	評価要領（採点要領）	備考
作業時間	作業時間	経過時間毎に減点	標準時間2H　打ち切り時間3H （15分経過毎に2点減点）	
作業時間	工程計画書	工程に誤りがないこと※別紙	誤り1箇所につき、2点減点する。（全5問）	
	接続線	器具までの未到達や突抜けがないこと	接続していない箇所、はみ出している箇所のないこと。	
	図記号	大きさ、用途が正しいこと	縮尺にあった図記号のサイズであること。器具の用途が適切であること。	
	配線（施工条件）	線種、施工法が正しいこと	施工方法にあった線種であること。施工方法が適切であること。	
	配置(壁付きの器具)	壁に接していること	器具が壁面に接していること。	
	線色	指定された線色であること	課題で指定された線色を使用していること。	
	不要線又は不足線の有無	不要線又は不足線があるか	不要線、不足線がないこと。	
	氏名、図面名称、日付、縮尺	氏名等が記入されていること	A3サイズに納まるように図面枠が作成されていること。（指定寸法あり）	
	#REF!	#REF!	氏名、図面名称、日付、縮尺が記入されていること。	
	照明レイアウト図	完成されていること	完成していれば5点、未完成であれば3点、作図していない場合は0点とする。	
	電灯配線図	完成されていること		
	コンセント設備図	完成されていること		
	凡例表	完成されていること		
	分電盤図	完成されていること		
	印刷のレイアウト	ズレによる印刷漏れの無いこと	著しいズレにより、印刷されない部分のないこと。	
安全作業	安全作業	他の作業者への妨げ行為等	私語は慎むこと。飲食のある場合は1点。	
安全作業	VDT作業	VDTに配慮した作業	1時間あたり5分は休憩すること。（指導員による指示を守ること）	

本報告書等は、基盤整備センターホームページ「職業能力開発ステーションサポートシステム（TETRAS）」の「基盤整備センター刊行物検索」から閲覧、ダウンロードができます。

URL：　http://www.tetras.uitec.jeed.or.jp/

資料シリーズ　No.51-2
「電気・電子系　訓練課題集
－離職者訓練用訓練課題の開発及びメンテナンスに関する調査研究－」

発行　2013年3月
発行者　独立行政法人高齢・障害・求職者雇用支援機構
職業能力開発総合大学校　基盤整備センター
所長　長谷川　健治
〒180-0006　東京都武蔵野市中町１－１９－１８　武蔵野センタービル４F
電話　0422-38-5225（普及促進室）
印刷　株式会社旭クリエイト
〒220-0023　神奈川県横浜市西区平沼１－３－１７　宮方ビル４F
電話　045-620-8890