筆答専門試験科目（午前）　　２０２３ 大修

情報工学系　　時間　９：３０～１２：３０

注　意　事　項

1. 各答案用紙の受験番号欄に受験番号を記入し，試験科目名欄に「情報工学系」と記入せよ．
2. 次の５題のうち，１番～３番の３題は必ず解答し，４番と５番からは１題を選び解答すること．
3. 解答は１題ごとに別々の答案用紙に，問題番号を明記した上で記入せよ．必要であれば，答案用紙の裏面に記入して良いが，答案用紙の表面にその旨を明記すること．
4. １枚の答案用紙に２題以上の解答を記入した場合はそれらの解答を無効とすることがある．
5. １題の解答を２枚以上の答案用紙に記入した場合はその解答を無効とすることがある．
6. 電子式卓上計算機等の使用は認めない．

1

1) 実数列 $a_0, a_1, \ldots$ において，a_{k+2} は a_{k+1} と a_k の算術平均であるとする $(k \geq 0)$.
いま，$a_0 = 0, a_1 = 1$ とおく．以下の問いに答えよ．

   a) 以下の関係を満たす行列 B の固有値を求めよ．また，各々の固有値に対し，固有ベクトルを一つあげよ．

$$\begin{pmatrix} a_{k+2} \\ a_{k+1} \end{pmatrix} = B \begin{pmatrix} a_{k+1} \\ a_k \end{pmatrix}$$

   b) a) の結果を用いて，$n \to \infty$ としたときの B^n の極限値を求めよ．

   c) b) の結果を用いて，$k \to \infty$ としたときの a_k の極限値を求めよ．

2) 実行列 A が $A^{\mathsf{T}} = -A$ という性質を満たすとする．ここで T は転置を表す．以下の問いに答えよ．

   a) A が固有値 λ をもつとき，$-\lambda$ も A の固有値か．理由とともに答えよ．

   b) すべての要素が実数のベクトル $\boldsymbol{x}$ について，$\boldsymbol{x}^{\mathsf{T}} A \boldsymbol{x} = 0$ であることを示せ．

   c) A の固有値は，ゼロでないときは，純虚数であることを示せ．

   d) A の行列式の値は負ではないことを示せ．

2. 次の問いに答えよ.

1) 命題論理について考える. p, q, r を命題記号とする論理式 $\neg(p \to (q \wedge r))$ を φ とする. φ について以下の問いに答えよ.

a) 次に示す φ の真理値表の空欄 ア ～ ク を埋めよ. ただし, T は真, F は偽を表す.

p	q	r	φ
F	F	F	ア
F	F	T	イ
F	T	F	ウ
F	T	T	エ
T	F	F	オ
T	F	T	カ
T	T	F	キ
T	T	T	ク

b) φ の選言標準形のうち, リテラル数が最少となるものを求めよ.

(次ページにつづく)

（前ページのつづき）

2) 一階述語論理について考える．w, x, y, z を変数記号とする．アリティ 1 の関数記号 s と定数記号 0 を用いると，任意の正の整数 n は，s の n 回の適用，すなわち

$$\underbrace{s(s(\cdots s}_{n\text{ 回}}(0)\cdots))$$

で表現できる．

アリティ 3 の述語記号 add に関する論理式 P_1, P_2 を次のように定義する．

$$(P_1) \quad \forall w\ \mathtt{add}(0, w, w)$$
$$(P_2) \quad \forall x \forall y \forall z\ \mathtt{add}(x, y, z) \to \mathtt{add}(s(x), y, s(z))$$

論理式 P_1, P_2 により，$\mathtt{add}(x, y, z)$ は x と y の加算結果が z であることを表している．

図 2.1 に，P_1, P_2 を仮定として，1 と 1 の加算結果が 2 であることを示す論理式

$$\mathtt{add}(s(0), s(0), s(s(0)))$$

を結論とする自然演繹の導出木（証明図ともいう）を示している．

図中の空欄 [ケ] ～ [ツ] に当てはまる最も適切な語句を選択肢群からそれぞれ選び，番号で答えよ．同じ選択肢を複数回選んでもよい．なお，図中の $\forall E$ は全称の除去規則，$\to E$ は含意の除去規則を表す．

選択肢群：

① 0	② w	③ x	④ y	⑤ z
⑥ $s(0)$	⑦ $s(w)$	⑧ $s(x)$	⑨ $s(y)$	⑩ $s(z)$
⑪ $s(s(0))$	⑫ $s(s(w))$	⑬ $s(s(x))$	⑭ $s(s(y))$	⑮ $s(s(z))$

$$\dfrac{\dfrac{\forall w\ \mathtt{add}(0, w, w)}{\mathtt{add}(0, \boxed{\text{ケ}}, \boxed{\text{ケ}})}\ \forall E \qquad \dfrac{\dfrac{\dfrac{\forall x \forall y \forall z\ \mathtt{add}(x, y, z) \to \mathtt{add}(s(x), y, s(z))}{\forall y \forall z\ \mathtt{add}(\boxed{\text{コ}}, y, z) \to \mathtt{add}(\boxed{\text{ス}}, y, s(z))}\ \forall E}{\forall z\ \mathtt{add}(\boxed{\text{コ}}, \boxed{\text{サ}}, z) \to \mathtt{add}(\boxed{\text{ス}}, \boxed{\text{セ}}, s(z))}\ \forall E}{\mathtt{add}(\boxed{\text{コ}}, \boxed{\text{サ}}, \boxed{\text{シ}}) \to \mathtt{add}(\boxed{\text{ス}}, \boxed{\text{セ}}, \boxed{\text{ソ}})}\ \forall E}{\mathtt{add}(\boxed{\text{タ}}, \boxed{\text{チ}}, \boxed{\text{ツ}})}\ \to E$$

図 2.1　導出木

（次ページにつづく）

（前ページのつづき）

3) 文脈自由文法 $G = (N, T, P, S)$ について考える．ここで，$N = \{ S \}$ は非終端記号の集合，$T = \{ a, b, c \}$ は終端記号の集合，P は以下に示す生成規則の集合，S は開始記号（出発記号ともいう）となる非終端記号を表す．

$$
\begin{aligned}
P = \{\ & S \to SaS, \\
& S \to Sb, \\
& S \to c\ \}
\end{aligned}
$$

a) 文法 G によって生成される言語における長さ n の語の数を $\Gamma(n)$ と定義する．$\Gamma(3), \Gamma(4), \Gamma(5)$ の値を求めよ．

b) 文法 G にはあいまい性がある．語 $cbacb$ に対する構文木（導出木ともいう）を 2 種類図示せよ．

c) 文法 G と同じ言語を生成する，あいまい性をもたない文脈自由文法 $G' = (N, T, P', S)$ をひとつ示せ．ただし，$|P'| \leq 3$ とする．

3.

1) スタックは後入れ先出し方式のデータ構造であり，プッシュ操作 PUSH(X) によりデータ X がスタックに追加され，ポップ操作 POP() によりデータがスタックから取り出される．ここでは，データとして整数のみを考えることとし，スタックの状態を以下のようなリスト形式の記法で記す．スタックが空の状態を[ ]とし，ここに PUSH(2)，PUSH(1)，PUSH(3)をこの順に実行すると状態は[2, 1, 3]となり，最後に追加されたデータが右端にある．さらに続いて POP()を実行すると状態は[2, 1]となる．次の問に答えよ．

a) 空のスタック[ ]に対して次の順序で操作を行った結果のスタックの状態はどれか，最も適切なものを次の選択肢群から選び番号で答えよ．

PUSH(1) → PUSH(3) → POP() → PUSH(5) → PUSH(6) → PUSH(4) → POP() → POP() → PUSH(2)

【選択肢群】 ① [2, 4, 3] ② [3, 5, 2] ③ [1, 3, 2]
④ [1, 5, 2] ⑤ [2, 5, 1] ⑥ [2, 3, 1]

b) ある状態のスタックに対して，あるデータ X_1 から X_6 を用いて，下記の順番でプッシュ操作とポップ操作を行ったところ，スタックの状態は[9, 6, 4, 3, 1]となった．下記の操作が行われる前のスタックの状態を上記の記法で答えよ．

PUSH(X_1) → PUSH(X_2) → PUSH(X_3) → POP() → PUSH(X_4) → PUSH(X_5) → POP() → POP() → PUSH(X_6)

c) 後入れ先出し方式のスタックに対して，先入れ先出し方式のデータ構造を何と呼ぶか記せ．

2) 最長共通部分文字列(longest-common substring)は与えられた 2 つの文字列間の連続した文字からなる共通部分文字列のうち最長のものである．例えば文字列 ABCD と文字列 ADBC の最長共通部分文字列は BC となり，その長さは 2 となる．また，1 文字も一致しない場合，その長さは 0 とする．2 つの与えられた長さ 5 以下の文字列 s1 と s2，それらの文字列の長さ s1_len，s2_len からその最長共通部分文字列の長さを返す関数 lcs を C 言語によりプログラム 3.1 のように実装した．この関数では，2 次元配列 mat を用意し，文字 s1[i]と文字 s2[j]が一致した場合，それが含まれる共通部分文字列でその一致箇所が何文字目に当たるかを mat[i+1][j+1]に格納することで，効率的に計算をしている．次の問に答えよ．

a) プログラム 3.1 の空欄 ア，イ，ウ に当てはまる最も適切なものを図 3.1 の①～㉜の選択肢群からそれぞれ選び番号で答えプログラムを完成させよ．同じ選択肢を複数回選んでもよい．

(次ページにつづく)

(前ページのつづき)

① 0	② 1	③ 2	④ 3				
⑤ i	⑥ i*2	⑦ i*3	⑧ j	⑨ j*2	⑩ j*3	⑪ k	⑫ r
⑬ s1[i]		⑭ s1[i+1]		⑮ s2[j]		⑯ s2[j+1]	
⑰ mat[i][j]		⑱ mat[i][j]+1		⑲ mat[i][j]+2		⑳ mat[i][j]+r	
㉑ mat[i][j+1]		㉒ mat[i][j+1]+1		㉓ mat[i][j+1]+2		㉔ mat[i][j+1]+r	
㉕ mat[i+1][j]		㉖ mat[i+1][j]+1		㉗ mat[i+1][j]+2		㉘ mat[i+1][j]+r	
㉙ mat[i+1][j+1]		㉚ mat[i+1][j+1]+1		㉛ mat[i+1][j+1]+2		㉜ mat[i+1][j+1]+r	

図 3.1 選択肢群

```
 1:  int lcs(char s1[], char s2[], int s1_len, int s2_len)
 2:  {
 3:    int i, j, k, mat[6][6];
 4:    for (i = 0; i < s1_len+1; i++) {
 5:      mat[i][0] = 0;
 6:    }
 7:    for (i = 0; i < s2_len+1; i++) {
 8:      mat[0][i] = 0;
 9:    }
10:    k = [   ア   ];
11:    for (i = 0; i < s1_len; i++) {
12:      for (j = 0; j < s2_len; j++) {
13:        if (s1[i] == s2[j]) {
14:          mat[i+1][j+1] = [   イ   ];
15:          if (mat[i+1][j+1] > k) {
16:            k = mat[i+1][j+1];
17:          }
18:        } else {
19:          mat[i+1][j+1] = [   ウ   ];
20:        }
21:      }
22:    }
23:    return k;
24:  }
```

プログラム 3.1

b) 関数 lcs の引数として文字列 s1 に ABDCA, 文字列 s2 に ACBDC, s1_len, s2_len にそれぞれ 5 を与えたとき, プログラム 3.1 の 23 行目の実行直前の時点での 2 次元配列 mat の状態を計算し, 図 3.2 のマス(i)から(v)に当てはまる数を答えよ. ただし, 各マスの上に記された[ ]の数字は mat の添字を表し, 下の数字はその添字がさす mat の要素の値である.

(次ページにつづく)

(前ページのつづき)

		A	C	B	D	C
	[0][0] 0	[0][1] 0	[0][2] 0	[0][3] 0	[0][4] 0	[0][5] 0
A	[1][0] 0	[1][1] 1	[1][2] 0	[1][3] 0	[1][4] 0	[1][5] 0
B	[2][0] 0	[2][1] 0	[2][2] 0	[2][3] 1	[2][4] 0	[2][5] 0
D	[3][0] 0	[3][1] 0	[3][2] 0	[3][3] 0	[3][4] **(i)**	[3][5] 0
C	[4][0] 0	[4][1] 0	[4][2] 1	[4][3] 0	[4][4] **(ii)**	[4][5] **(iii)**
A	[5][0] 0	[5][1] **(iv)**	[5][2] 0	[5][3] 0	[5][4] 0	[5][5] **(v)**

図 3.2 2 次元配列 mat の状態

c) この関数 lcs のように，対象となる問題を複数の部分問題に分割し，部分問題の計算結果を記録して再利用しながら解いていくアルゴリズムを総称して何と呼ぶか，最も適切なものを次の選択肢群から選び番号で答えよ．

【選択肢群】 ① 共役勾配法　② 動的計画法　③ 焼きなまし法
④ 貪欲法　⑤ モンテカルロ法　⑥ ニュートン・ラフソン法

d) 文字列において部分列(subsequence)とは，出現順序を変えずに取り出した文字の列であり，部分文字列とは異なり，元の文字列で連続した文字からなる必要はない．最長共通部分列(longest-common subsequence)は与えられた 2 つの文字列で共通する部分列のうち最長のものである．例えば文字列 ABCD と文字列 ADBC の最長共通部分列は ABC となり，その長さは 3 となる．プログラム 3.1 の 10 行目から 23 行目をプログラム 3.2 に置き換えることで，関数 lcs を 2 つの与えられた長さ 5 以下の文字列の最長共通部分列の長さを返す関数に変更した．mat[i+1][j+1]には，それぞれの先頭の文字から s1[i]と s2[j]までの部分文字列の間の最長共通部分列の長さを格納する．プログラム 3.2 の空欄 エ, オ, カ, キ に当てはまる最も適切なものを図 3.1 の①～㉜の選択肢群からそれぞれ選び番号で答えプログラムを完成させよ．同じ選択肢を複数回選んでもよい．

```
for (i = 0; i < s1_len; i++) {
  for (j = 0; j < s2_len; j++) {
    if (s1[i] == s2[j]) {
      mat[i+1][j+1] = [ エ ];
    } else if (mat[i][j+1] > [ オ ]) {
      mat[i+1][j+1] = mat[i][j+1];
    } else {
      mat[i+1][j+1] = [ カ ];
    }
  }
}
return [ キ ];
```

プログラム 3.2

(次ページにつづく)

（前ページのつづき）

3) 2つの文字列がどの程度異なっているかを示す距離を考える．1文字の挿入・削除・置換の操作を用いて文字列 ABCDE を文字列 ACDFB に変形するには，例えば ABCDE から B を削除し，E を F に置換し，最後尾に B を挿入すればよい（ABCDE→ACDE→ACDF→ACDFB）．このとき，挿入，削除のコストをそれぞれ2，置換のコストを3とすると，その変形のコストは変形に要した操作のコストの総和となり，2+3+2=7 となる．重み付きレーベンシュタイン距離は，1つの文字列から，もう1つの文字列への変形のコストのうち最小のものである．挿入，削除のコストをそれぞれ2，置換のコストを3としたとき，与えられた2つの文字列の重み付きレーベンシュタイン距離をLとする．2つの与えられた長さ5以下の文字列とそれらの文字列の長さからこのLを返す関数 `Levenshtein` を C 言語を用いてプログラム 3.3 のように実装した．この関数では，それぞれの先頭の文字から `s1[i]` と `s2[j]` までの部分文字列の間のLを計算して `mat[i+1][j+1]` に格納し，それを再利用することで効率的に計算をしている．次の問に答えよ．

a) プログラム 3.3 の空欄 [ク], [ケ], [コ], [サ], [シ], [ス], [セ] に当てはまる最も適切なものを図 3.1 の①～㉜の選択肢群からそれぞれ選び番号で答えプログラムを完成させよ．同じ選択肢を複数回選んでもよい．

b) 文字列 ABCDE と文字列 CEADB のLの値を答えよ．

```
int Levenshtein(char s1[], char s2[], int s1_len, int s2_len)
{
  int i, j, r, mat[6][6];
  for (i = 0; i < s1_len+1; i++) {
    mat[i][0] = [ク];
  }
  for (i = 0; i < s2_len+1; i++) {
    mat[0][i] = [ケ];
  }
  for (i = 0; i < s1_len; i++) {
    for (j = 0; j < s2_len; j++) {
      if (s1[i] == s2[j]) {
        r = 0;
      } else {
        r = [コ];
      }
      if (mat[i][j]+r < mat[i][j+1]+2 && mat[i][j]+r < mat[i+1][j]+2) {
        mat[i+1][j+1] = [サ];
      } else if (mat[i][j+1]+2 < mat[i+1][j]+2) {
        mat[i+1][j+1] = [シ];
      } else {
        mat[i+1][j+1] = [ス];
      }
    }
  }
  return [セ];
}
```

プログラム 3.3

4.

1) 実数 t を変数とする実関数 $f(t)$ のラプラス変換を $F(s) = \int_0^\infty \mathrm{e}^{-st} f(t)dt$ とする．ただし，s は複素変数とし，実部が正であるとする．次の関数 $h(t)$ のラプラス変換 $H(s)$ を求めよ．

$$h(t) = f(at - b)$$

ただし，a, b は正の定数，$t \leq 0$ のとき $f(t) = 0$ とする．

2) 図 4.1 に示すシステムを考える．ここで伝達関数 $P(s)$ は

$$P(s) = \frac{1}{\left(\frac{s}{2} + 1\right)\left(\frac{s}{10} + 1\right)} \tag{4.1}$$

であり，$u(t)$ は入力，$y(t)$ は出力，t は時間を表すとする．

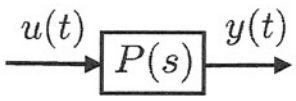

図 4.1

以下の問いに答えよ．

a) $P(s)$ の入力 $u(t)$ を単位ステップ信号

$$u(t) = \begin{cases} 1 & t \geq 0 \\ 0 & t < 0 \end{cases}$$

としたとき，出力 $y(t)$ を求めよ．

b) $P(s)$ の周波数特性をボード線図 (Bode plot) で示せ．横軸は角周波数 ω の対数とし，縦軸はゲインをデシベル (dB)，位相を度 (°) で表すこと．また，各軸上の適切な箇所に適切な数値を記入すること．図は概形でよく，折れ線近似を用いてよい．どのように図を描いたのか，考え方を簡潔に述べること．

(次ページにつづく)

(前ページのつづき)

3) 図 4.2 に示すフィードバック制御系を考える．制御対象の伝達関数 $P(s)$ は式 (4.1) のものとし，制御器の伝達関数 $K(s)$ は次式で与える．

$$K(s) = \frac{K_0}{s+5} \tag{4.2}$$

ただし，$K_0 \geq 0$ は定数ゲインである．また，$r(t)$ は目標値，$y(t)$ は制御量，$e(t)$ は偏差，$u(t)$ は制御入力，t は時間を表すとする．

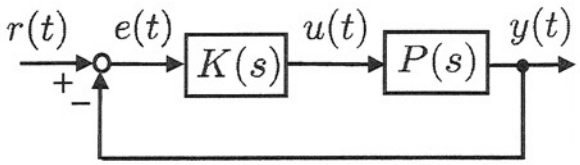

図 4.2

以下の問いに答えよ．

a) 目標値 $r(t)$ から偏差 $e(t)$ への伝達関数 $G(s)$ を求めよ．ただし，伝達関数は分母・分子ともに展開した上で s の降べきの順に整理し，係数は整数にして書くこと．

b) フィードバック制御系が安定となるような K_0 の範囲を求めよ．

c) 目標値 $r(t)$ として単位ステップ信号を用いた場合の定常偏差が 0.1 以下となる K_0 の範囲を求めよ．

d) 目標値 $r(t)$ として単位ステップ信号を用いた場合の定常偏差を 0 とするには，制御器として，どのような性質を持つものに変更すればよいか簡潔に説明せよ．

e) $K(s)$ として式 (4.2) を用いた場合のフィードバック制御系の根軌跡の概形を描け．つまり，K_0 を 0 から ∞ へと変化させたときにフィードバック制御系の極が変化する様子を複素平面に図示せよ．$K_0 = 0$ のときの極の座標を明記し，軌跡が分岐する点とその漸近線を描くこと．

f) 式 (4.2) 中の K_0 を 0 から ∞ へと変化させたときにフィードバック制御系の応答がどう変化するか，理由と共に簡潔に説明せよ．e) で得た根軌跡を用いてもよい．

5． 以下の問 1)～問 3)に答えよ．

1) 論理回路理論とコンピュータアーキテクチャについて，空欄 (A) ～ (I) に入る最も適切な語句を図 5.2 の選択肢群からそれぞれ選び，番号で答えよ．同じ選択肢を複数回選んでもよい．

図 5.1 の論理回路を考える．状態を持たず，入力に対して一意に出力の決まる回路を (A) と呼ぶ．そうではなく，入力と回路自身の状態によって出力の決まる回路を (B) と呼ぶ．ポジティブエッジトリガ型 D フリップフロップ（R1～R4）と NOT ゲートで構成される図 5.1 は (C) である．あるクロックサイクルでクロック ck のポジティブエッジから十分な時間が経過して回路が安定した時刻を t1 とする．続く 4 つのクロックサイクルで，同様に回路が安定した時刻を t1 に近い順からそれぞれ t2， t3， t4， t5 とする．時刻 t1 での w1， w2， w3， w4 の全ての値を 0 とする．NOT ゲートで w1 を反転した値が R1 の入力であることから，時刻 t2 での w1 の値は (D) となる．w1 が R2 の入力であることから，時刻 t3 での w2 の値は (E) となる．時刻 t5 での w1 と w3 の値は (F)，w2 と w4 の値は (G) となる．

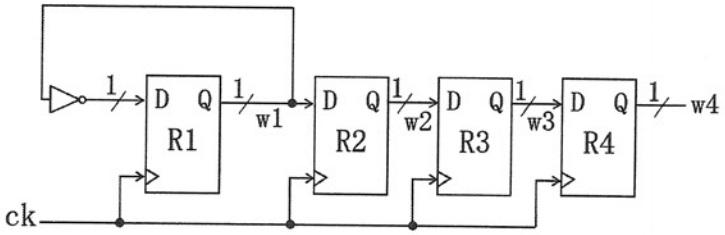

図 5.1 論理回路

プログラム実行時におけるデータアクセスには，最近アクセスしたデータ項目を再利用するという (H) 局所性がある．この局所性を用いて，記憶階層では最近アクセスされたデータ項目ほどプロセッサに近いレベル（上位レベル）に置くようにする．また，最近アクセスされたデータ項目の近くにある別のデータ項目が参照されるという (I) 局所性がある．

(1)DRAM, (2)メインメモリ, (3)キャッシュ, (4)揮発性メモリ,
(5)パイプライン方式, (6)カウンタ回路, (7)組合せ回路, (8)順序回路, (9)NOT,
(10)OR, (11)AND, (12)NOR, (13)NAND, (14)XOR, (15)0, (16)1, (17)2, (18)3,
(19)加算器, (20)デコーダ, (21)時間的, (22)空間的, (23)真理値表

図 5.2 選択肢群

(次ページにつづく)

（前ページのつづき）

2) 図 5.3 の命令形式の set 命令と add 命令から成る命令セットを考える．op フィールドには set 命令の場合に 0 を，add 命令の場合に 1 を格納する．汎用レジスタとして 8 ビット幅のレジスタ x0，x1 を利用する．

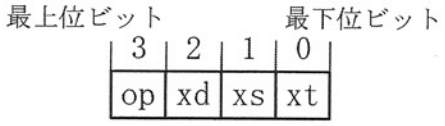

図 5.3 命令形式

2進数の命令アドレス	2進数の命令	アセンブリ言語の命令
000	0100	set x1, 0
001	0010	set x0, 2
010	0111	set x1, 3
011	1101	add x1, x0, x1

表 5.1 命令列

set 命令は xd で指定するレジスタに即値を格納する．即値は，xs を上位ビット，xt を下位ビットとして連結した 2 ビットの符号なし整数を同じ値を持つ 8 ビットの符号なし整数に変換することで得る．例えば，set x1，2 という命令は，レジスタ x1 に 2 という値を格納する．

add 命令は xs，xt で指定する 2 つのレジスタの値の符号なし加算の結果を xd で指定するレジスタに格納する．アセンブリ言語の記述ではレジスタを xd，xs，xt の順番で指定する．例えば，add x1，x0，x1 はレジスタ x0 と x1 の値の加算の結果をレジスタ x1 に格納する．

a) 表 5.1 の命令列を上から順番に実行した後の x0 の値を 10 進数で答えよ．

b) 表 5.1 の命令列を上から順番に実行した後の x1 の値を 10 進数で答えよ．

（次ページにつづく）

（前ページのつづき）

3) プログラムカウンタ（PC），命令メモリ（IM），デコードユニット（DU），レジスタファイル（RF），加算器（A1,A2），レジスタ（r1～r8），マルチプレクサ（M），ゼロ拡張ユニット（ZU）を持つ図 5.4 のパイプライン方式のプロセッサを考える．DU，A1，A2，M，ZU は組合せ回路である．M は，その真上から入力される制御信号が 0 の時に左上からの入力を，制御信号が 1 の時に左下からの入力をそれぞれ右側に出力する．DU は 4 ビット幅の入力 ir の最下位ビットから順に配線 xt，xs，xd，op に出力する．ZU は 2 ビットの符号なし整数と同じ値を持つ 8 ビットの符号なし整数を出力する．u ビット幅のレジスタは u 個のポジティブエッジトリガ型 D フリップフロップを並べることで実現する．全てのレジスタは同一クロックで駆動する．

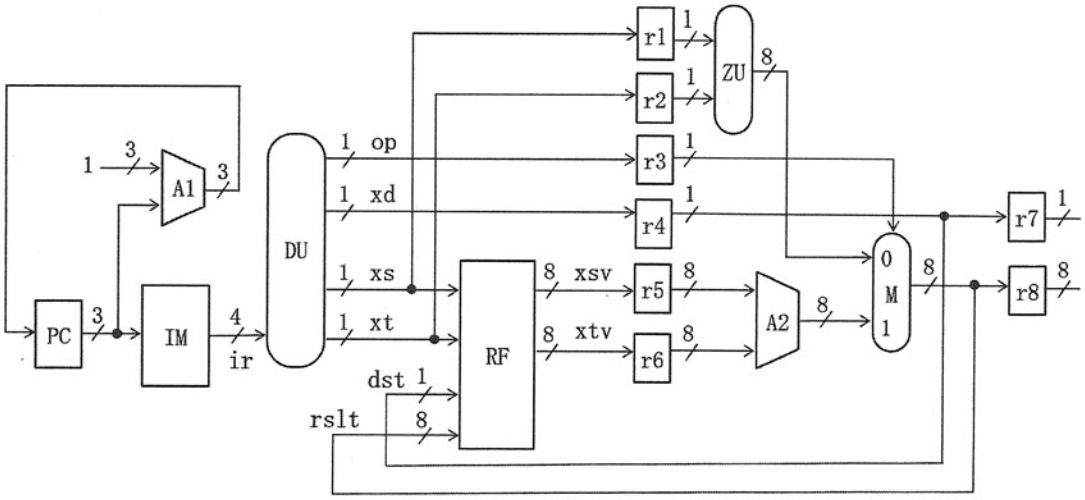

図 5.4 プロセッサ

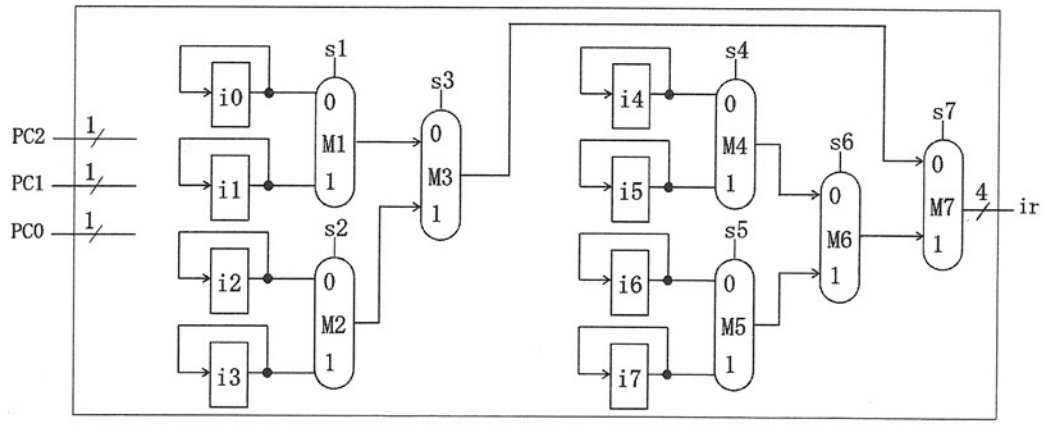

図 5.5 命令メモリ（IM）

（次ページにつづく）

（前ページのつづき）

a) 図 5.5 に s1～s7 の配線が未接続である IM の構成を示す．IM には最大で 8 個の命令を格納できる．i0～i7 は命令を格納する 4 ビット幅のレジスタ，M1～M7 はマルチプレクサである．IM は 3 ビット幅の PC の値をアドレスとして入力し，この値が d $(0 \leq d \leq 7)$ の時にレジスタ id の値を配線 ir に出力する．PC の各ビットを最下位ビットから PC0，PC1，PC2 と表記する．

（ア）i5 の値を配線 ir に出力するための s4，s6，s7 のそれぞれの値を 0，1 から選択せよ．

（イ）s1～s7 のそれぞれに接続すべき配線を PC0，PC1，PC2 から選択せよ．

b) 図 5.6 に s8～s11 の配線が未接続である 2 つのレジスタを持つ RF の構成を示す．x0，x1 は 8 ビット幅のレジスタ，M8～M11 はマルチプレクサである．RF は dst の値が p $(p = 0, 1)$ の時に rslt の値をレジスタ xp に書き込む．RF は xs の値が n $(n = 0, 1)$ の時にレジスタ xn の値を xsv に出力し，xt の値が m $(m = 0, 1)$ の時にレジスタ xm の値を xtv に出力する．

（ア）x0 の値を x0 に書き込み，rslt の値を x1 に書き込むための s8 と s9 のそれぞれの値を 0，1 から選択せよ．

（イ）s8～s11 のそれぞれに接続すべき配線を xs，xt，dst，rslt から選択せよ．

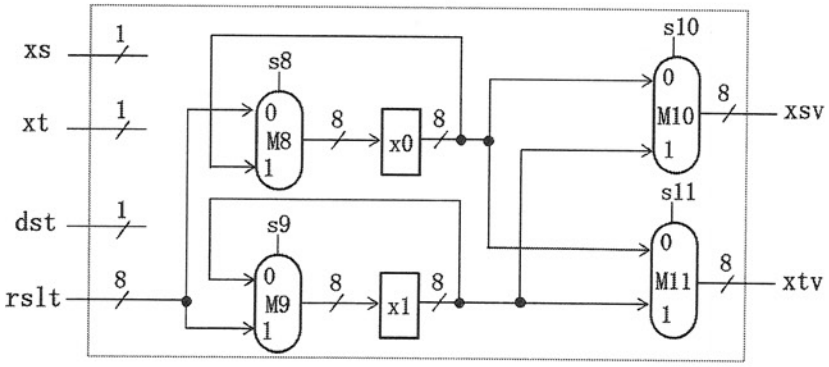

図 5.6 レジスタファイル（RF）

（次ページにつづく）

（前ページのつづき）

c) 図 5.4 のプロセッサの IM には**問 2)**の表 5.1 の命令列が格納されており，それ以外のすべてのレジスタは 0 で初期化される．PC の値が 0 で，最初の set 命令が IM からフェッチされるクロックサイクルを考える．そのクロックサイクルで十分な時間が経過して回路が安定した時刻を T1 とする．続く 3 つのクロックサイクルで，同様に回路が安定した時刻を T1 に近い順からそれぞれ T2，T3，T4 とする．

図 5.4 のパイプライン方式のプロセッサは，T2 では，2 番目の set 命令を IM からフェッチする．また，この時に，最初の set 命令の即値を ZU で 8 ビットに変換した値が M から出力され，dst と rslt の値はそれぞれ 10 進数で 1 と 0 となる．T3 では，dst と rslt の値はそれぞれ 10 進数で 0 と 2 となる．

（ア）T4 における配線 dst と rslt の値を 10 進数で答えよ．

（イ）T4 における配線 xsv と xtv の値を 10 進数で答えよ．

d) 図 5.4 のプロセッサは，表 5.1 の 4 番目の add 命令が処理するデータを，RF 内のレジスタから読み出した後に r5，r6 を経由して A2 に供給する．このため，3 番目の set 命令が生成したデータを直後の add 命令が利用できず，**問 2)**の命令の仕様の通りに表 5.1 の命令列を実行できない．これを解決するには，3 番目の set 命令と 4 番目の add 命令とのデータ依存関係を判断して，RF を経由しないで，3 番目の set 命令が生成したデータを A2 に供給すればよい．この手法をフォワーディングと呼ぶ．

図 5.4 のプロセッサを修正して，フォワーディングのためのマルチプレクサ M12 とそのための配線を追加したプロセッサを図 5.7 に示す．M12 の制御信号は s12 である．

（ア）フォワーディングのために，s12 を生成する組合せ回路の入力として必要となる最も適切な 2 つのレジスタの名前を答えよ．

（イ）それらのレジスタを必要とする理由を 100 文字程度で述べよ．

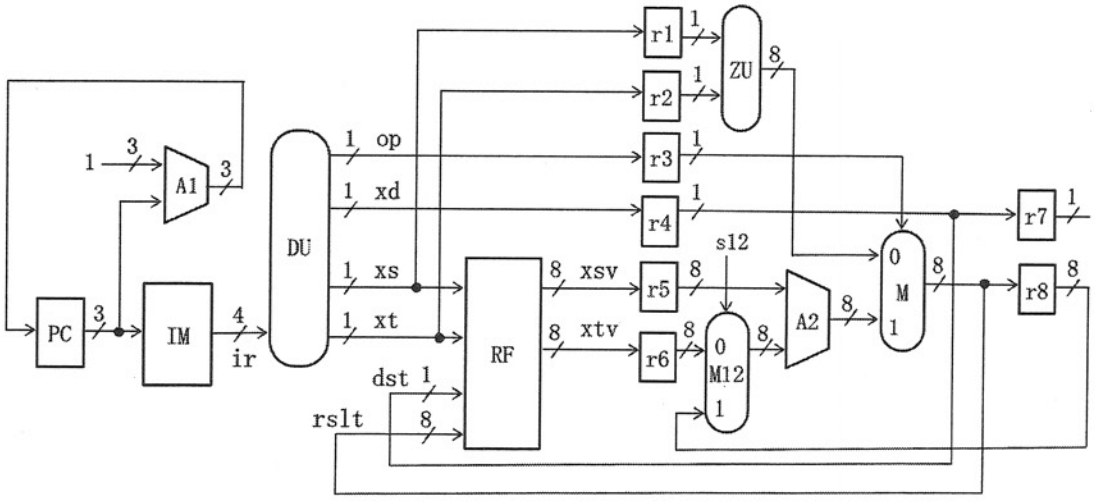

図 5.7 修正したプロセッサ