資　料

システム開発技術者のためのCAI
取扱い説明書

昭和59年3月

JIPDEC

財団法人　日本情報処理開発協会

この資料は、日本自転車振興会から競輪収益の一部である機械工業振興資金の補助を受けて、昭和５８年度に実施した「マイクロコンピュータの利用に関する共通的な技術開発」の一環としてとりまとめたものであります。

目　　次

1. 概　　要

1.1　開発の目的、特徴

マイクロコンピュータ内蔵システムは、ますます増大することが考えられ、これらにたずさわるシステム開発技術者の育成は急務である。

そこで、その育成の一助となるためのＣＡＩによるシステム開発技術者教育について、本年度は入門コースに限定して開発した。

本年度とりあげた特徴は第一にマイクロコンピュータ用の素子として最もポピュラーなＺ－８０を採用し、これの制御用言語としてアセンブラを教えること、第二にシステム開発に必要とする基礎知識について広く網羅したことである。

1.2　ハードウエア

ＣＡＩシステムで用いるハードウェア及びＣＡＩシステムの稼動に必要なハードウェア上の各種条件等について述べる。

(1)　使用機種

ＳＥＩＫＯ　９５００－Ｋ

メモリー	５１２ＫＢ
ディスプレイ	１４インチカラー
ミニフロッピー	６４０ＫＢ×２台

(2)　使用言語

ＦＯＲＴＲＡＮ

ＳＥＩＫＯ９５００スーパーＢＡＳＩＣ

(3)　タッチセンサー

当ＣＡＩシステムの特徴の一つとしてタッチセンサー（透視型指タッチ入力装置：米国エログラフィック社製）を採用した。タッチセンサー

は、ＣＲＴ画面上を指で押すことによりＸ－Ｙ軸の座標を読み取る装置で、ＣＡＩシステムの進行、解答等のマンマシンインターフェイスとして機能する。

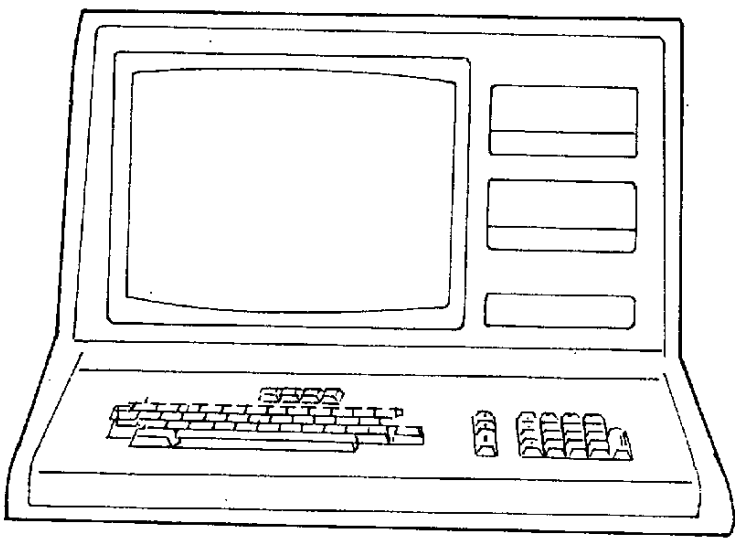


図１－１　外　観　図

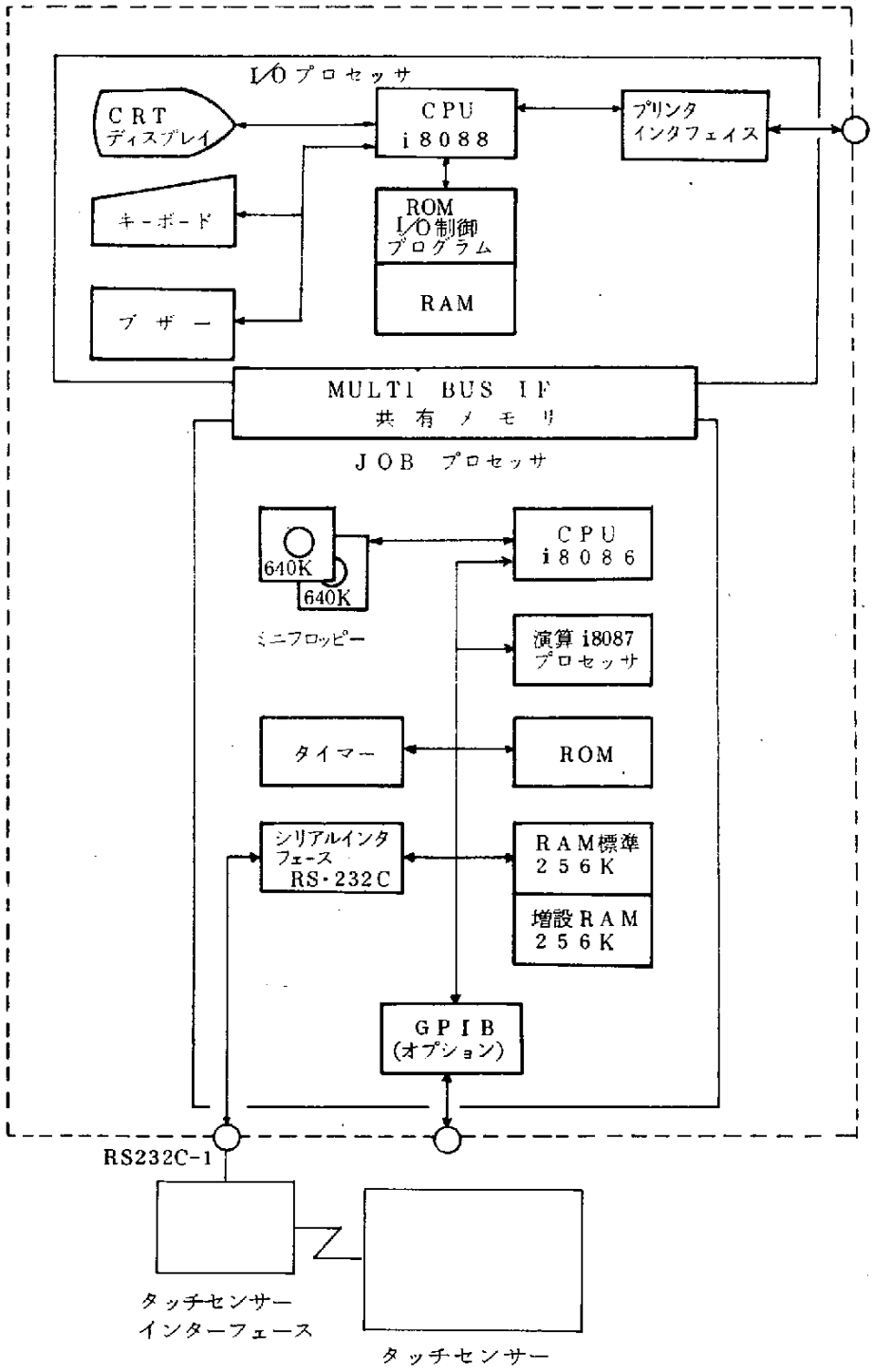

I/O プロセッサ
CRT
ディスプレイ
CPU
i8088
プリンタ
インタフェイス
キーボード
ROM
I/O制御
プログラム
RAM
ブザー
MULTI BUS IF
共有メモリ
JOB プロセッサ
640K
640K
ミニフロッピー
CPU
i8086
演算 i8087
プロセッサ
タイマー
ROM
シリアルインタ
フェース
RS-232C
RAM標準
256K
増設RAM
256K
GPIB
(オプション)
RS232C-1
タッチセンサー
インターフェース
タッチセンサー

図1－2　ブロック図

1.3　教程作成用ソフトウエア

(1)　ソフトウェアの構成

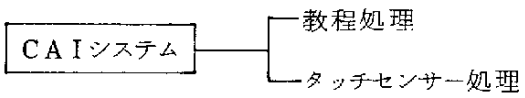


図1－3　教程管理

(2)　機　能

(A)　ディスプレイの活用

①　簡単なアニメを利用する

動きを見せることによって、理解度、興味を高める。

②　漢字入りの和文にし、簡単な説明文にする。

③　タッチセンサーによりディスプレイに直接タッチできるので、親しみやすくなっている。

(B)　シミュレーション方式

①　具体的な内容にする

②　コンピュータ内に具体的なモデル（簡易に図解できるもの）をプログラム化する。

(3)　本ＣＡＩシステムの教科内容

①　システム設計の基礎

ⅰ　ハードウェアの基礎知識

ⅱ　ソフトウェアの基礎知識

②　システムの設計と製作

ⅰ　システム設計

ⅱ　ハードウェアの設計と製作

ⅲ　ソフトウェアの設計と製作

1.4　学習開始の操作手順

学習開始は次の手順で行う

(1)　プログラムの呼出し

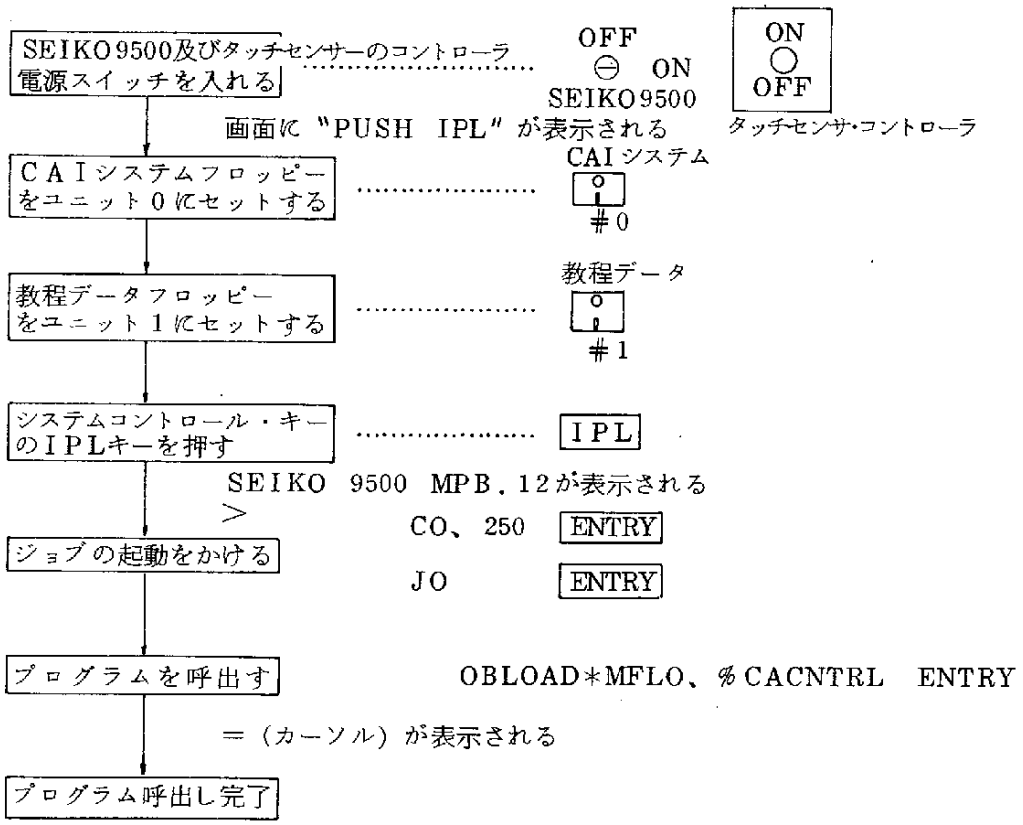


図1－4

(2)　ＣＡＩシステムの開始

前記の手順でプログラムの呼出しが完了したあとの手順である。

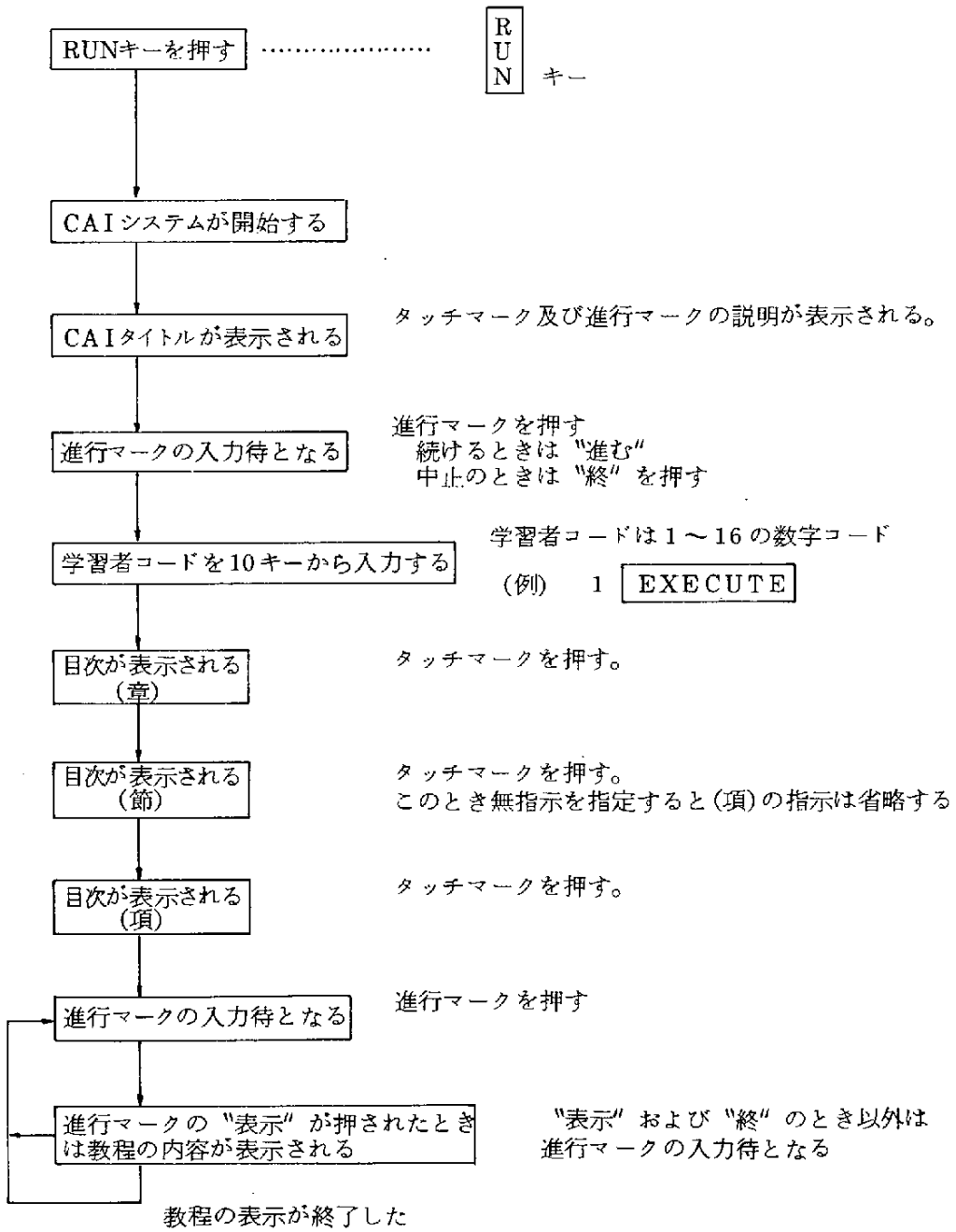


図１－５

1.5 「システム開発技術者のためのＣＡＩ」の教科内容と学習方法

(1) 教科内容

学習用教程は、システム設計の基礎編とシステムの設計と製作編から成る。

以下に学習用教程の内容を記す。

第１部　システム設計の基礎

Ⅰ．ハードウェアの基礎知識

① 基本的なＣＰＵシステムの機能

② ＣＰＵ部の機能

③ ＣＰＵ部の基本構成

④ いろいろなロジック回路

⑤ ＴＴＬの電気的特性

⑥ ＣＰＵの基本的動作について

⑦ ＣＰＵ部の回路

⑧ I/Oインターフェースの基本型態（ディジタルとアナログ）

⑨ I/Oとして実際に使われる回路（入出力回路）

⑩ データ転送

⑪ 長距離データ伝送方式の選択

⑫ 転 送 路

⑬ 関係知識

⑭ センサー

Ⅱ．ソフトウェアの基礎知識

① Ｚ－８０ＣＰＵの構成

② Ｚ－８０の命令

③ プログラムの構成

④ プログラムの開発とデバッグ

第２部　システムの設計と製作

Ⅰ．システム設計

①　システム設計

②　要求内容の分析

③　設計方針の決定

④　ハードウェアとソフトウェアの分担

⑤　ソフトウェアの作成

⑥　ハードウェアの作成

⑦　システムテスト

⑧　試　　用

Ⅱ．ハードウェアの設計と製作

①　システムの分析

②　エレメントの決定

③　コントロール出力系の詳細設計

④　データ入力系の詳細設計

⑤　表示系の詳細設計

⑥　ポートアドレスデコーダーの詳細設計

Ⅲ．ソフトウェアの設計と製作

①　動作手順の決定

②　メインフローチャートの作成

③　プログラムエレメントの構成

④　メモリーの割り当て

⑤　I／Oアドレスの割り当て

⑥　プログラムの作成

⑦　CP／Mによるプログラミング

⑧　デバッグ

(2) 学習方法

前述の学習開始の操作手順に従って電源投入後、ＣＡＩシステムディスクをフロッピーディスク装置にセットし、続いて [ＩＰＬ] キーを押す。しばらくすると画面の左上にカーソルが点滅するので [ＲＵＮ] キーを押す。

これで学習開始となる。あとはタッチマークと進行マークに従って操作する。

以下にタッチマークと進行マークの機能について解説する。

① タッチマークの機能（触れる事により機能する）

タッチマーク

：各教科の画面上に表示される。

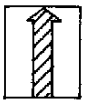
学習したい箇所のタッチマーク（）に触れるとその部分の解説が始まる。

② 進行マークの機能（触れる事により機能する。）

進行マーク

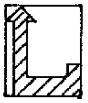

：

進む


：

戻る

} 他のテーマに移る時、あるいは戻る時に押す。


：

始めへ

各レベルから一つ前のレベルに戻る時に使用する。

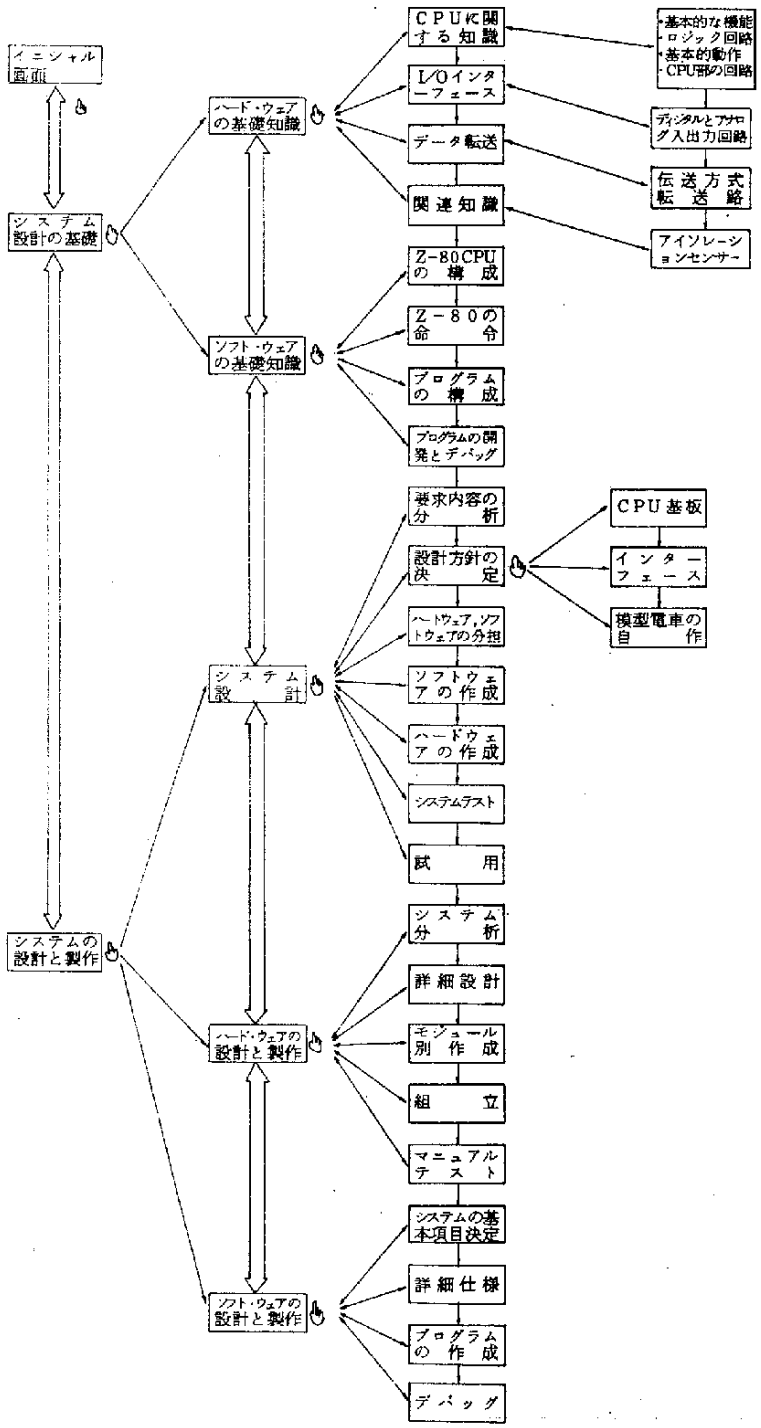

：

終了

学習を途中で終了したい時に使用する。

以上５つのマークにより学習者が自由にテーマを選択し、学習する事が出来る。

2. 学習用教程一覧


イニシャル画面
システム設計の基礎
システムの設計と製作
ハード・ウェアの基礎知識
ソフト・ウェアの基礎知識
システム設計
ハード・ウェアの設計と製作
ソフト・ウェアの設計と製作
CPUに関する知識
I/Oインターフェース
データ転送
関連知識
Z-80CPUの構成
Z-80の命令
プログラムの構成
プログラムの開発とデバッグ
要求内容の分析
設計方針の決定
ハードウェア,ソフトウェアの分担
ソフトウェアの作成
ハードウェアの作成
システムテスト
試用
システム分析
詳細設計
モジュール別作成
組立
マニュアルテスト
システムの基本項目決定
詳細仕様
プログラムの作成
デバッグ
・基本的な機能
・ロジック回路
・基本的動作
・CPU部の回路
ディジタルとアナログ入出力回路
伝送方式転送路
アイソレーションセンサー
CPU基板
インターフェース
模型電車の自作

0110

システム開発技術者のためのCAI（第1部）

1 ハードウェア

2 ソフトウェア

0120

ハードウェアの基礎知識

1 CPUに関する知識

2 I/Oインターフェース

3 データ転送

4 関連知識

0130

CPUに関する知識

① 基本的な機能

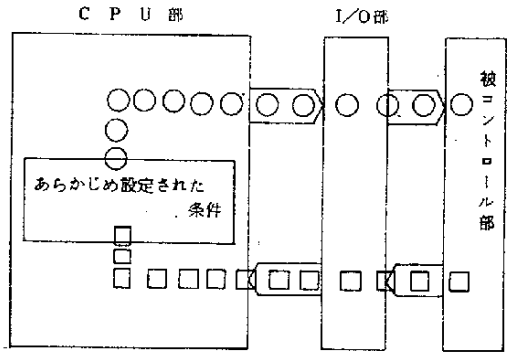
② いろいろなロジック回路

③ 代表的なCPU

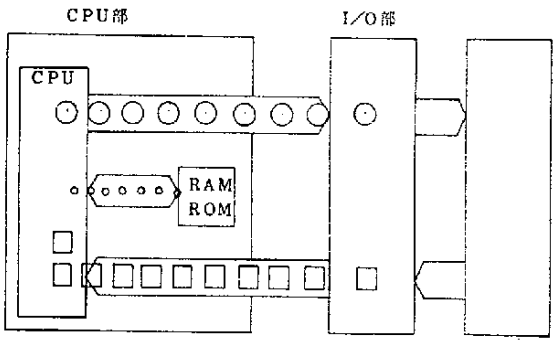
④ 汎用CPU基板

0150

マイクロコンピュータの基本的な機能



0170

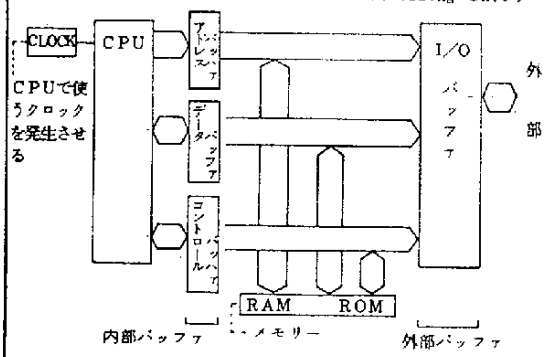


メモリとCPUの間は頻繁にデータが行き交うのでCPU部の信頼性を高く設計する

0180

マイクロコンピュータの基本構成

--- 中央処理装置（Central Processing Unit）



0181

内部バッファ

ＣＰＵはＬＳＩでできており、外部回路に対するドライブ能力は小さいのでRAM，ROMを接続するため内部バッファを設ける。

0182

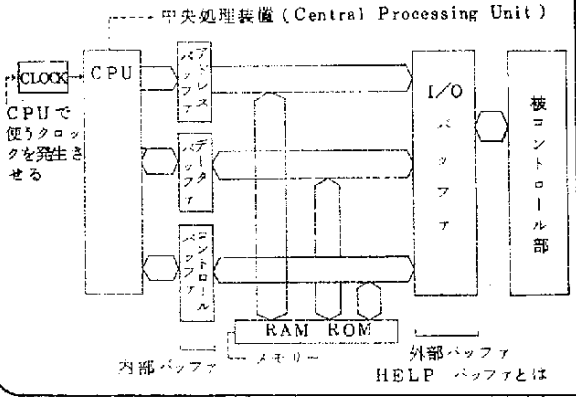
外部バッファ

外部とのインターフェースのため更にバッファを設ける。

この設計次第で信頼性は高まる。

0183

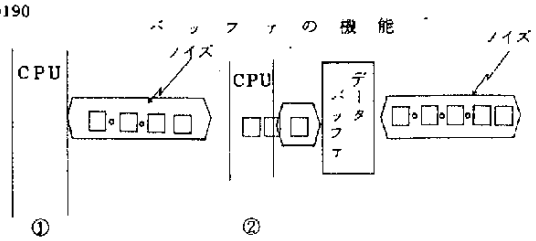
マイクロコンピュータの基本構成



HELP バッファとは

0190

バッファの機能



ノイズによるＣＰＵの誤動作を抑えるためバッファを置く、ドライブ機能の増大、データの一時記憶などの機能も持つ。

0210 いろいろなロジック回路

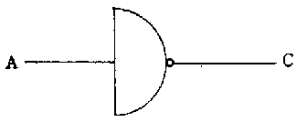
インターフェース用として最も広く利用されるものがＴＴＬ（Transistor Transistor Logic）回路がある。

代表的なロジック回路

名称	機能	表示	計算式	ＩＣ例
バッファ	ドライブ機能アップ	A—▷—B	B＝A	LS08など
インバータ	信号の反転	A—▷○—B	$B=\overline{A}$	LS04, LS06 LS16
AND	入力信号のAND	A, B—D—C	C＝A・B	LS08
OR	入力信号のOR	A, B—)—C	C＝A＋B	LS32
NAND	入力信号のNAND	A, B—D○—	$C=\overline{A\cdot B}$	LS00
フリップフロップ	J－K D			LS73, LS74 LS107
3ステート	アクティブでないとき出力をHiZにする	A—▷—C, B	C＝"H"なら B＝A C＝"L"ならHiZ	LS126

HELP

0211 インバータ

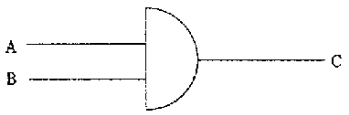


A　1　0

↓　↓

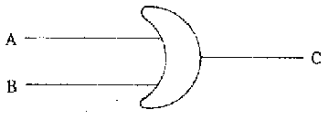
C　0　1

0212　ＡＮＤ回路



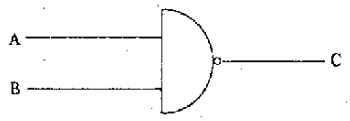
A	1	0	1	0
B	1	1	0	0
	↓	↓	↓	↓
C	1	0	0	0

0213　ＯＲ回路



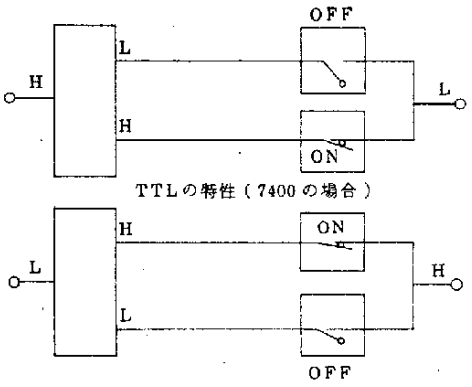
A	1	0	1	0
B	1	1	0	0
	↓	↓	↓	↓
C	1	1	1	0

0214　ＮＡＮＤ回路

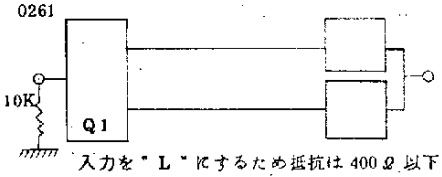


A	1	0	1	0
B	1	1	0	0
	↓	↓	↓	↓
C	0	1	1	1

0260



0261



入力を＂Ｌ＂にするため抵抗は 400Ω 以下

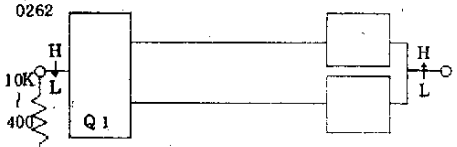
＂Ｈ＂＂Ｌ＂の意味

0.8Ｖ以下の電圧状態を＂Ｌ＂ローというＯＦＦを表す

２Ｖ以上の電圧状態を＂Ｈ＂ハイという　ＯＮを表す

信頼性を高めるため商品としては
＂Ｌ＂入力レベルは 0.4Ｖ
以下、＂Ｈ＂の入力レベ
ルは 2.4Ｖ以上として設計
されている。

0262



10Ｋから 400 へ数字が下って行くと Ｈ→Ｌ、Ｌ→Ｈ へ変わる。

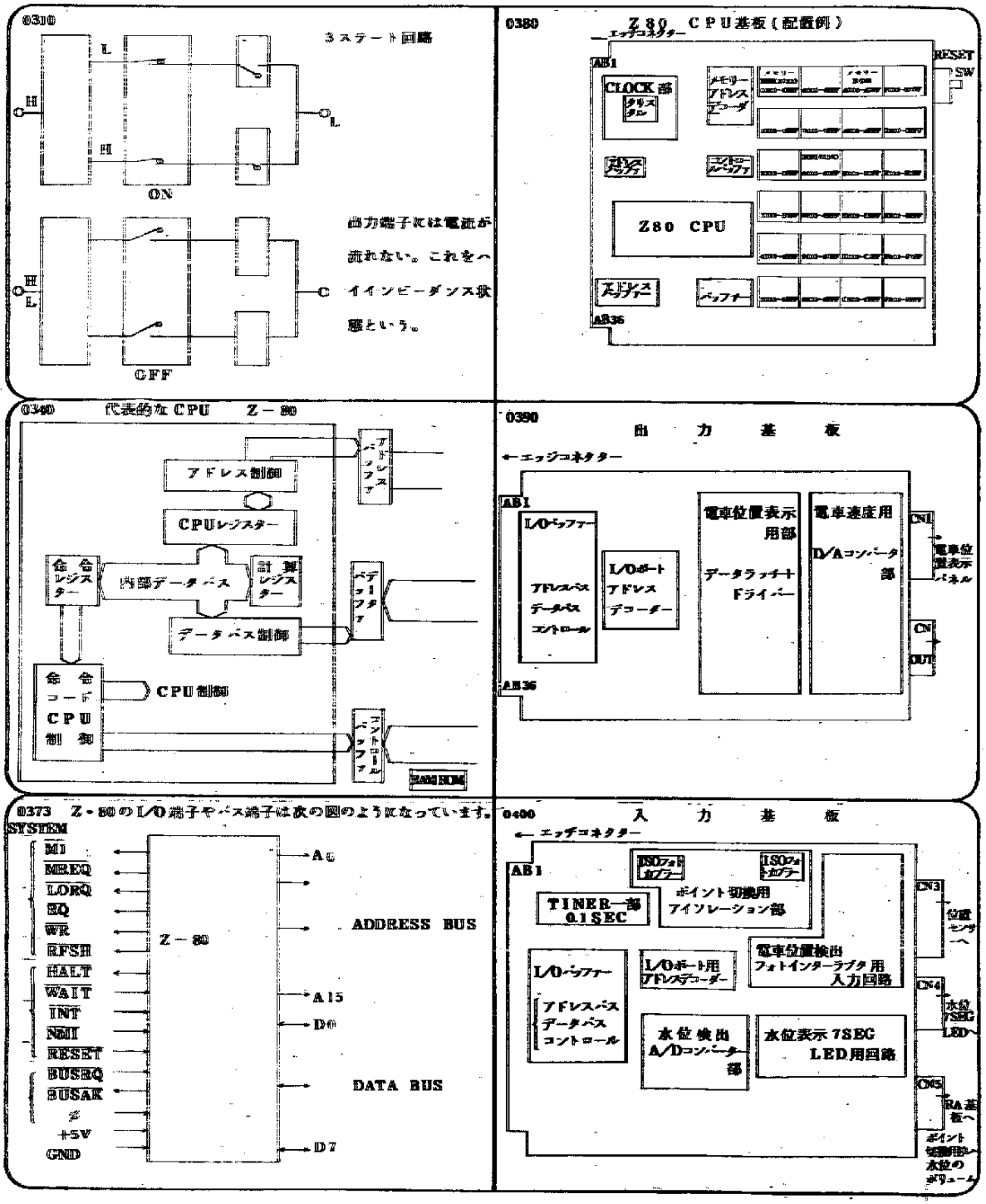
入力を＂Ｌ＂にするため抵抗は 400Ω 以下

＂Ｈ＂＂Ｌ＂の意味

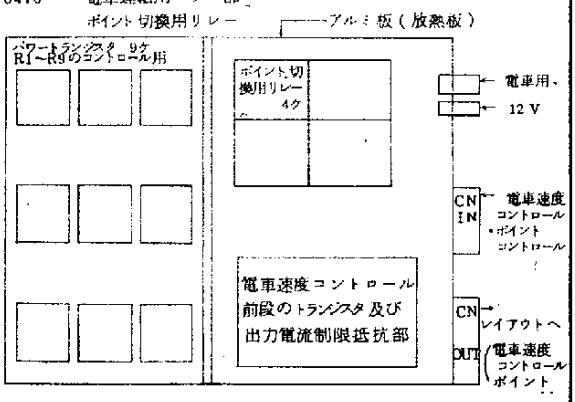
0.8Ｖ以下の電圧状態を＂Ｌ＂ローというＯＦＦを表わす。

２Ｖ以上の電圧状態を＂Ｈ＂ハイというＯＮを表わす。

信頼性を高めるため商品としては
＂Ｌ＂の入力レベルは 0.4Ｖ以下、＂Ｈ＂の入力レベルは 2.4Ｖ以上として設計されている。

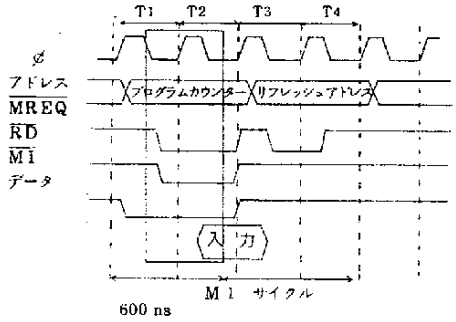

0310
3ステート回路
L
H
H
L
ON
出力端子には電流が流れない。これをハイインピーダンス状態という。
H
L
OFF
0380
Z80 CPU基板（配置例）
エッジコネクター
AB1
RESET SW
CLOCK部
Z80 CPU
AB36
0340
代表的なCPU Z－80
アドレス制御
CPUレジスター
命令レジスター
内部データバス
計算レジスター
データバス制御
命令コード CPU制御
CPU制御
0390
出力基板
エッジコネクター
AB1
I/Oバッファー
アドレスバス データバス コントロール
I/Oポート アドレスデコーダー
電車位置表示用部
データラッチ＋Fライバー
電車速度用
D/Aコンバータ部
CN1
電車位置表示パネル
CN
OUT
AB36
0373 Z－80のI/O端子やバス端子は次の図のようになっています。
SYSTEM
M1
MREQ
IORQ
RD
WR
RFSH
HALT
WAIT
INT
NMI
RESET
BUSRQ
BUSAK
φ
+5V
GND
Z－80
A0
A15
D0
D7
ADDRESS BUS
DATA BUS
0400
入力基板
エッジコネクター
AB1
TINER一部 0.1SEC
ポイント切換用アイソレーション部
I/Oバッファー
アドレスバス データバス コントロール
I/Oポート用アドレスデコーダー
電車位置検出フォトインターラプタ用入力回路
水位検出A/Dコンバーター部
水位表示7SEC LED用回路
CN3
CN4
水位7SEG LED
CN5
AB36

0410　電車運転用パワー部



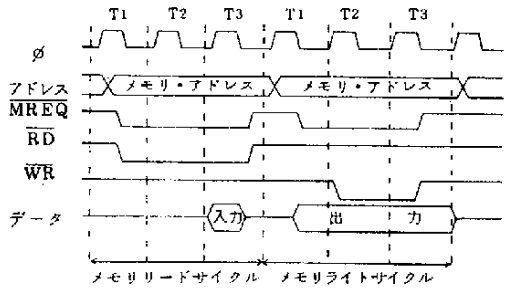
0420　基本動作モード

プログラム　OPコード　読み込み　サイクル



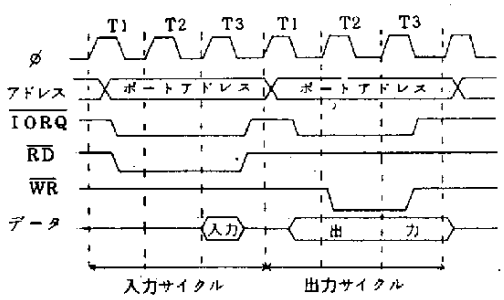
このサイクルは、メモリに記憶されているプログラムを読み込んで次の実行命令を解釈する。メモリへのアクセスは、標準型で600ns、高速タイプだと375nsとなる。

0430　メモリ　アクセス　サイクル



メモリのアクセススピードはフェッチサイクルに比べて若干遅い。

0440　IOアクセス　サイクル



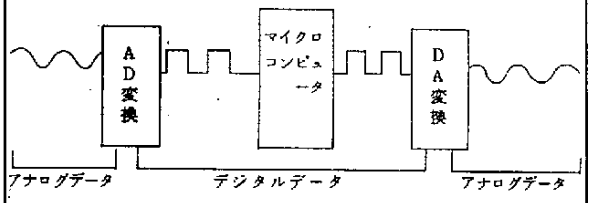
アクセススピードはメモリアクセス時よりきびしい。

0450　I/O　インターフェース

① デジタルとアナログについて説明します。

② 入出力回路について説明します。

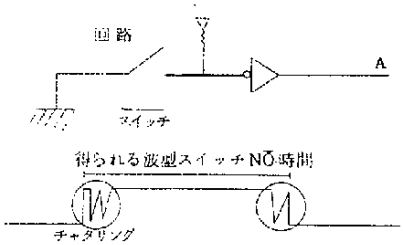
0460



コンピュータはデジタルデータだけを扱うので

入出力ではそれぞれAD、DAの変換を行う。

0470　一般的なデジタル入力回路



チャタリングの問題

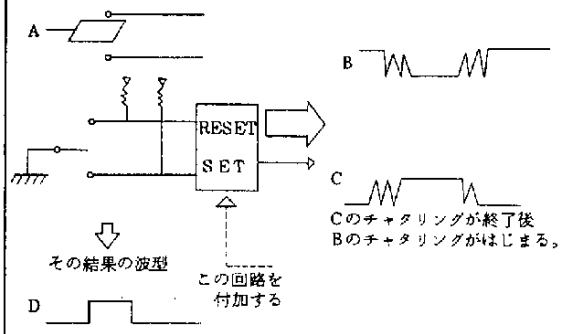
高速でアクセスするCPUはこれを数回のスイッチ

ONがあったと受け取る。

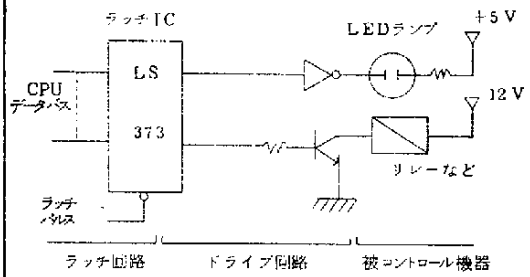
HELP　Ⅰ　Ⅱ

0472

HELPⅡ　ハードウエアによるチャタリング解決法



0480　一般的なデジタル出力回路



100V　ACモーターなどをコントロールする場合

ノイズをCPU側に送らないようリレーなどを用いる。

HELP

0510

データ転送

① データ転送

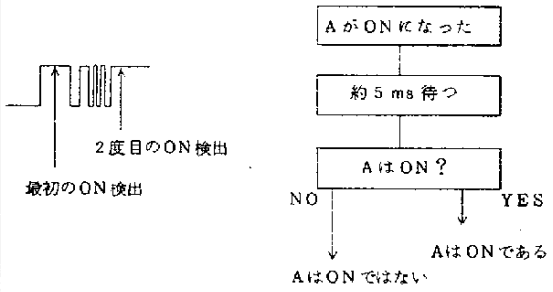
② 転送方法

③ 転送路

0471

HELPⅠ

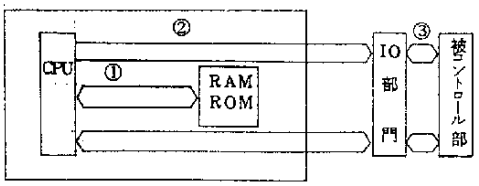
ソフトウエアによるチャタリング解決法



そのほかハードウエアによる方法がある。

0520

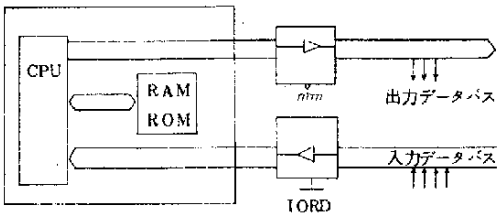
データ転送には3つの場合がある



① RAM ROMとのデータ転送は汎用基板を使用する場合問題ない。

HELP

0530



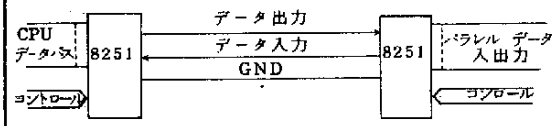
IO基板が多くなると入出力バスを分離し

ドライブ能力、ノイズマージンの増大を計る。

0560

シ リ ア ル 転 送

少ない回線数で長距離転送する。



パラレル ― シリアル ― パラレル

（非同期式の場合）

0540

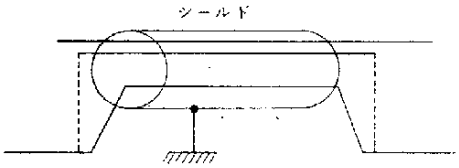
転 送 方 式

1. パ ラ レ ル

2. シ リ ア ル

0570

〔 伝 送 路 の い ろ い ろ 〕



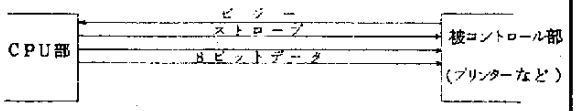
伝送路の長さが50㎝程度以上になるとシールド線を用いる。

0550

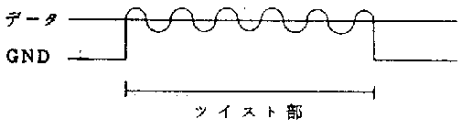
パ ラ レ ル 転 送

基本的データ転送タイミング

回線の接続

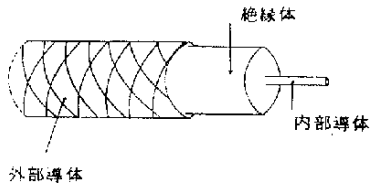


0580



数10ｍ程度のものにはツイストペアをさらに多芯化して用いる。

0590



１００ｍ以上の伝送には同軸伝送路を用いる。

0620

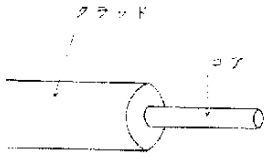
関　　連　　知　　識

① アイソレーション

② センサー

③ ノイズ

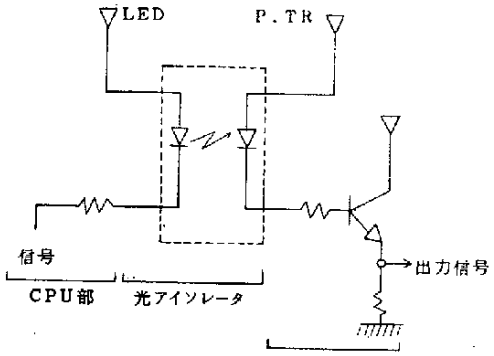
0600



１kmを越える場合の伝送には光伝送路を用いる。

0630

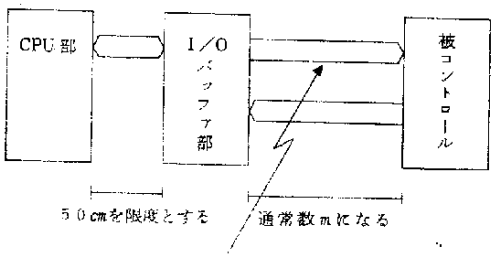
最も用いられているアイソレーション



回路の中で電気的に独立させることをアイソレーションという。

0610

③

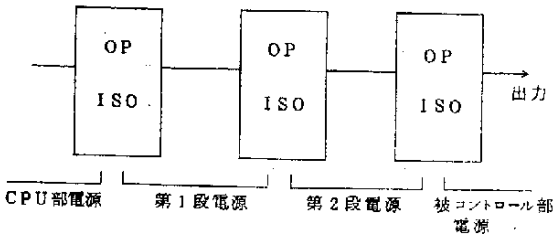


ノイズはかならず入るのでそれを避ける方法を講じる。

0640

さらにアイソレーションの向上をねらうには

多重化を行う。



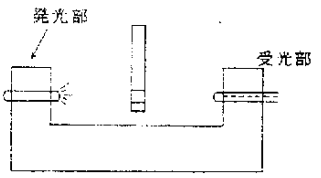
全ての電源を独立させる必要がある。

0650

位置を計るデジタル系

○マイクロスイッチ

○フォトインタラプター



アナログ系

○イメージセンサー

CCDを並べ光のさえぎりをセンスする。

○エンコーダー

光または接触で位置をセンスする。

0660

厚さを計る

厚さ1cm程度以上のものは位置センターを利用する。

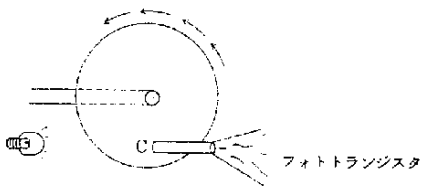
薄いものなら赤外線、X線などを利用すると良い。

0670

速度を計る

○アナログ速度計の出力をデジタル化する。

○簡単な速度計。



一定時間内のトランジスタのONの数をカウントする。

0680

ノイズを避ける

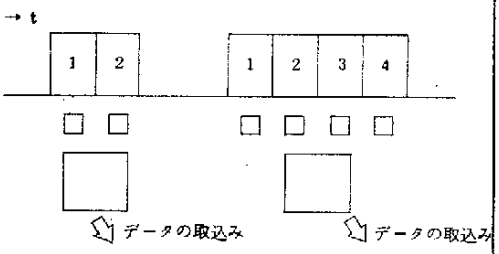
① ノイズ発生源を調査しノイズの発生を押える。

② シールドなどによってノイズが乗らないようにする。

③ ノイズが乗っても安全な設計をするノイズの確認を行う。

0690

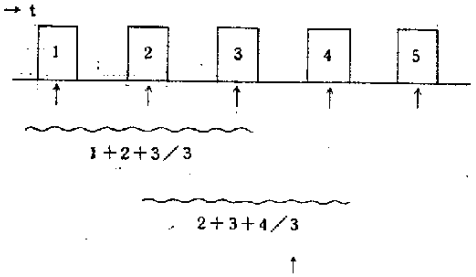
多数決による入力データミスのチェック



このような方法で正しいデータを取り入れる。

0700

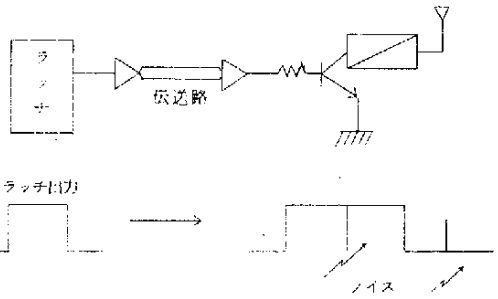
平均値法による入力データミスのチェック



これを多くすればミスは軽減される。

0710

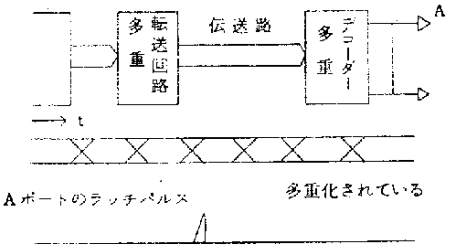
出力が少ない場合



この程度のノイズではリレーが
ON　OFFを間違うことはない。

0720

出力が多い場合（多重化方式をとる）



Aポートのラッチパルス　　多重化されている

Aポートのラッチパルスが来たときデータミスが生じることにそなえて平均値法をとるため、繰り返してAポートのデータを送り続ける。

1110　Z－80 レジスター構成

Z－80 には次のレジスターがある。

A	F
B	C
D	E
H	L

A′	F′
B′	C′
D′	E′
H′	L′

I	R
IX	
IY	
SP	
PC	

1120　Z－80 の命令群

158 種の命令は群に分かれる

Ⓐ　8ビットロード命令
Ⓑ　16ビットロード命令
Ⓒ　エクスチェンジ命令
Ⓓ　ブロック命令
Ⓔ　8ビット算術・論理演算命令
Ⓕ　16ビット算術・論理演算命令
Ⓖ　ローテート、シフト命令
Ⓗ　ビット操作命令
Ⓘ　ジャンプ、コール、リターン命令
Ⓙ　リスタート命令
Ⓚ　入出力命令
Ⓛ　CPU制御命令

1130　8 ビットロード命令群(1)

ソース（もと）

デスティネーション（行き先）	I	R	A	B	C	D	E	H	L
A	○	○	○	○	○	○	○	○	○
B			○	○	○	○	○	○	○
C			○	○	○	○	○	○	○
D			○	○	○	○	○	○	○
E			○	○	○	○	○	○	○
H			○	○	○	○	○	○	○
L			○	○	○	○	○	○	○
(HL)			○	○	○	○	○	○	○
(BC)			○						
(DE)			○						
(IX+d)			○	○	○	○	○	○	○
(IY+d)			○	○	○	○	○	○	○
(nn)			○						
I			○						
R			○						

1140　8 ビットロード命令群(2)

ソース（もと）

デスティネーション（行き先）	(HL)	(BC)	(DE)	(IX+d)	(IY+d)	(nn)	n
	○	○	○	○	○	○	○
	○			○	○		○
	○			○	○		○
	○			○	○		○
	○			○	○		○
	○			○	○		○
	○			○	○		○
							○
							○
							○

1150　16 ビットロード命令群

ソース

デスティネーション	AF	BC	DE	HL	SP	IX	IY	nn	(nn)	(SP)
AF										○
BC								○	○	○
DE								○	○	○
HL								○	○	○
SP				○		○	○	○	○	
IX								○	○	○
IY								○	○	○
(nn)		○	○	○	○	○	○			
(SP)	○	○	○	○		○	○			

←POSH 命令

↑ POP 命令

1160　エクスチェンジ（交換）命令群

	AF′	BC′, DE′, HL′	HL	IX	IY
AF	○				
BC DE HL		○			
DE			○		
(SP)			○	○	○

命令語は BC，DE，HL ↔ BC′，DE′，HL′ は EXX

他は全て EX で書く

1170　ブロック命令群(1)

ブロック転送

	(HL)	HL：ソースポインタ DE：ディスティネーションポインタ BC：バイトカウンタ
(DE)	○	'LDI'で(DE)←(HL) HL,DEを＋1してBCを－1する
	○	'LDIR'で(DE)←(HL) HL,DEを＋1し、BCを－1する BC＝0までくり返す
	○	'LDD'で(DE)←(HL) HL,DEを－1してBCを－1する。
	○	'LDDR'で(DE)←(HL) HL,DEを－1してBCを－1する。 BC＝0までくり返す

1180　ブロック命令群(2)

ブロックサーチ

	(HL)	HL：Aと照合するメモリアドレス BC：バイトカウント
(DE)	○	'CPI'で HLを＋1し，BCを－1する
	○	'CPIR'で HLを＋1し，BCを－1する (HL)＝AかBC＝0までくり返す
	○	'CPD'で HLを－1し，BCを－1する
		'CPDR'で HLを－1し，BCを－1する (HL)＝AかBC=0までくり返す

1190　8ビット算術、論理演算命令群(1)

ソース

命令語	A	B	C	D	E	H	L	(H,L)	(IX+d)
'ADD'	○	○	○	○	○	○	○	○	○
'APC'	○	○	○	○	○	○	○	○	○
'SUB'	○	○	○	○	○	○	○	○	○
'SBC'	○	○	○	○	○	○	○	○	○
'AND'	○	○	○	○	○	○	○	○	○
'XOR'	○	○	○	○	○	○	○	○	○
'OR'	○	○	○	○	○	○	○	○	○
'CP'	○	○	○	○	○	○	○	○	○
'INC'	○	○	○	○	○	○	○	○	○
'DEC'	○	○	○	○	○	○	○	○	○

1200　8ビット算術、論理演算命令群(2)

ソース

命令語	(IY+d)	n	演算内容
'ADD'	○	○	A＋ソース→A
'APC'	○	○	A＋ソース＋キャリー→A
'SUB'	○	○	A－ソース→A
'SBC'	○	○	A－ソース－キャリー→A
'AND'	○	○	Aとソースのアンド→A
'XOR'	○	○	AとソースのXOR→A
'OR'	○	○	AとソースのOR→A
'CP'	○	○	Aとソースを比較
'INC'	○		ソースを＋1→ソース
'DEC'	○		ソースを－1→ソース

1210　特殊演算命令

命令語	演算内容
'DAA'	BCD演算のためAをアジャストする
'CPL'	Aの補数→A
'NEG'	$\bar{A}$→A
'CCF'	キャリーフラグを反転する
'SCF'	キャリーフラグをセットする

1220　16ビット算術演算命令群(1)

ソース

ディスティネーション	BC	DE	HL	SP	IX	IY	命令語
HL	○	○	○	○			'ADD'
IX	○	○		○	○		
IY	○	○		○		○	
HL	○	○	○	○			'ADC'
HL	○	○	○	○			'SBC'
－	○	○	○	○	○	○	'INC'
－	○	○	○	○	○	○	'DEC'

1230　16 ビット算術演算命令群(2)

		演　算　内　容
ディスティネーション	HL IX IY	ソースデスティネーション →ソース
	HL	ソース＋ディスティネーション＋キャリー →ディスティネーション
	HL	ディスティネーション－ソース－キャリー →ディスティネーション
	—	ソース＋1　→　ソース
	—	ソース－1　→　ソース

1240–1250　ローテイト、シフト命令群

命令語	A	B	C	D	E	H	L	(HL)	(IX+d)	(IY+d)	内　容
'RLC'	○	○	○	○	○	○	○	○	○	○	左ローテートサーキュラー
'RRC'	○	○	○	○	○	○	○	○	○	○	右ローテートサーキュラー
'RL'	○	○	○	○	○	○	○	○	○	○	左ローテート
'RR'	○	○	○	○	○	○	○	○	○	○	右ローテート
'SLA'	○	○	○	○	○	○	○	○	○	○	算術的左シフト
'SRA'	○	○	○	○	○	○	○	○	○	○	算術的右シフト
'SRL'	○	○	○	○	○	○	○	○	○	○	論理的右シフト
'RLD'								○			左ローテートディジット
'RRD'								○			右ローテートディジット

1260　ビット操作命令群

命令語	内　容	レジスタ及びメモリ
'BIT'	ビットテスト	A, B, C, D, E, H, L (HL), (IX+D) (IY+D)
'RES'	ビットリセット	
'SET'	ビットセット	

a)　ビットテスト命令
　指定のビットをテストし、Zフラグをセット、リセットする。
b)　ビットリセット命令
　指定のビットをリセットする。
c)　ビットセット命令
　指定のビットをセットする。

1270　ジャンプ、コール、リターン命令群(1)

命令語	PCの内容	条件 無条件	キャリ	ノンキャリ	ゼロ	ノンゼロ
'JP'	NN	○	○	○	○	○
'JR'	PC＋E	○	○	○	○	○
'JP'	(HL)	○				
'JP'	(IX)	○				
'JP'	(IY)	○				
'CALL'	NN	○	○	○	○	○
'DJNZ'	PC＋E					
'RET'	スタックの内容	○				
'RETI'	スタックの内容	○				
'RETN'	スタックの内容	○				

1280　ジャンプ、コール、リターン命令群(2)

命令語	条件 パリティ 偶数	パリティ 奇数	負	正	カウント	内　容
'JP'	○	○	○	○		
'JR'						相対ジャンプ
'JP'						
'JP'						
'JP'						
'CALL'	○	○	○	○		
'DJNZ'						Bを－1してゼロでなければジャンプする
'RET'	○	○	○	○		
'RETI'						インタープラトからのリターン
'RETN'						ノンマスカブルインタープラトからのリターン

1290　リスタート命令

命令語	'RST0'	'RST8'	'RST16'	'RST24'	'RST32'	'RST40'	'RST48'	'RST56'
コールアドレス	0000H	0008H	0010H	0018H	0020H	0028H	0030H	0038H

1300　入出力命令－1 直接及びレジスタ間接入出命令

ＩＮ時のデスティネーション
ＯＵＴ時のソースレジスタ

命令語	A	B	C	D	E	H	L	内　容
ＩＮ (n)	○							直接入力
ＩＮ (c)	○	○	○	○	○	○	○	間接入力
ＯＵＴ(n)	○							直接出力
ＯＵＴ(c)	○	○	○	○	○	○	○	間接出力

→ ポート、アドレス

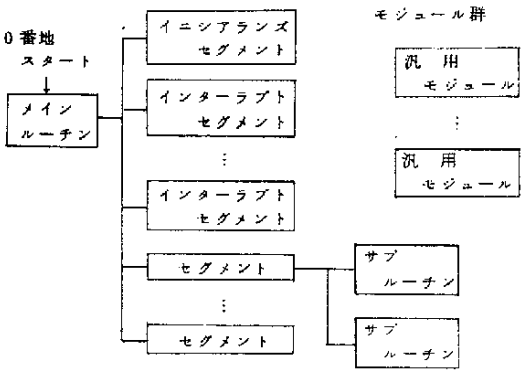
1310　入出力命令－2　ブロック入出力命令

命令語	（Ｈし）	内　容
ＩＮＩ	○	入力後ＨＬを＋１してＢを－１する
ＩＮＩＲ	○	入力後ＨＬを＋１してＢを－１する Ｂ＝０までくり返す
ＩＮＤ	○	入力後ＨＬを－１、Ｂを－１する
ＩＮＤＲ	○	入力後ＨＬを－１、Ｂを－１する Ｂ＝０までくり返す
ＯＵＴＩ	○	出力後ＨＬを＋１し、Ｂを－１する
ＯＴＩＲ	○	出力後ＨＬを＋１、Ｂを－１する Ｂ＝０までくり返す
ＯＵＴＤ	○	出力後ＨＬを－１、Ｂを－１する
ＯＴＤＲ	○	出力後ＨＬを－１しＢを－１する Ｂ＝０までくり返す

1320　ＣＰＵ制御命令群

命令語	内　容
'ＮＯＰ'	ノーオペレーション、ＣＰＵはＰＣを進めるだけで何もしない。
'ＨＡＬＴ'	ＣＰＵのストップ
'ＤＩ'	割り込み禁止（ディスイネーブル）
'ＥＩ'	割り込み許可（イネーブル）
'ＩＭ０'	割り込みモードをモード０にセット
'ＩＭ１'	割り込みモードをモード１にセット
'ＩＭ２'	割り込みモードをモード２にセット

1330　プログラムの構成……一般的に下図の構成をなす



2110　システム開発技術者のためのＣＡＩ（第２部）

Ａ　システム設計

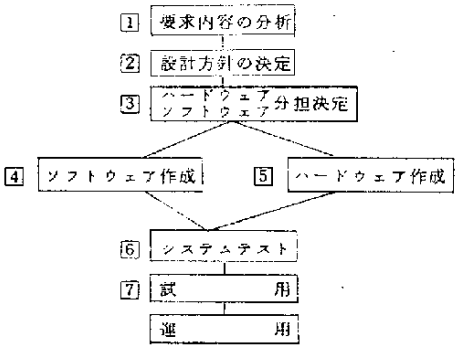
Ｂ　ハードウェア設計と製作

Ｃ　ソフトウェアの設計と製作

数字のところにさわると画面が変わります。

2120　システム設計手順

システム開発は次の手順によって進められる



2130　要求内容の分析

要求

ショーに出品するためにコンピュータ制御により模型電車を動かす。

聞取による要求の整理

○マイクロコンピュータは専用とする。
○電車は２両とする。
○衝突させないようにコントロールする。
○適当な障害物を設け変化を持たせる。

問題点の抽出

○ノイズ対策を充分に考える。
○会場への移動のためコンパクトにする。
○移動や過激な操作に耐えるようにする。

2140

設　計　方　針　の　決　定

1　ＣＰＵ基板は市販・汎用品を購入する。

2　インターフェース部分は自作する。

3　制御対象物は模型電車として自作する。

2150

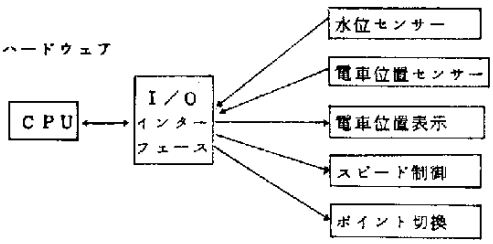
1.　ＣＰＵ基板　　資料１へ

2.　インターフェース　　資料２へ

3.　模型電車の自作　　資料３へ

それぞれをタッチするとＨＥＬＰがあらわれる。

2160　ハードウェアとソフトウェアの分担



ソフトウェア
電車位置データ、水位データを受けあらかじめ与えられた条件にしたがいスピード制御のためレール電圧をコントロールすると同時にポイントを切換、電車位置を表示する。

2170　ソフトウェアの作成

―― 考 え 方 ――

- 速く処理するには？
- 小さなメモリで済むには？
- 正確に処理するには？
- ソフトウェアの修正は楽か？
- 拡張性はあるか？

などを考えて作成する。

2180　ハードウェアの作成

―― 考 え 方 ――

- できるだけ市販品を利用する。
- チェック済みの信頼性の高いものを利用する。

2190　システムテスト

―― 考 え 方 ――

① ハードウェアが確実に動くことを確認する。

② 部分的にソフトウェアで動かしてみる。

③ しだいに、ソフトウェア制御の領域を広げていく。

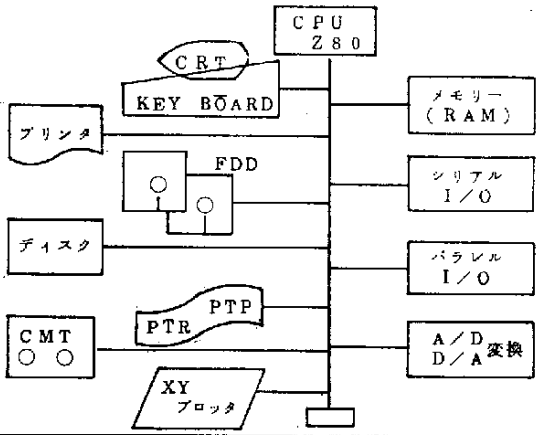
④ ハードウェア作成者、ソフトウェア作成者が必ず立合う。

2200　試　　用

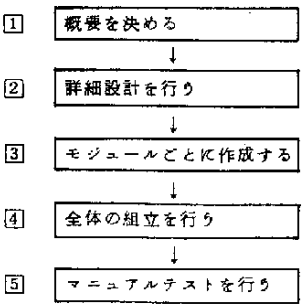
実際に動くことが確認されてもまだ実際の工場や、現場には設置することはできない。

試用期間をもうけ、時間的経過やオペレータが変わることによる障害などを克服しなければならない。

2210　市販・汎用品リスト



2220　ハードウェア作成手順



2230　システムの概要を決める

[1] 制御対象物のレイアウト

[2] 対象物の種類と特性調査

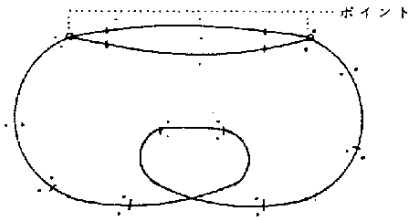
[3] コントロールの概要

[4] データ入力装置

[5] 表示装置

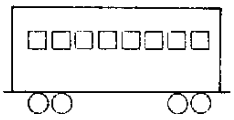
これらを調査、計測し、概要を定める。

2240　レールのレイアウト

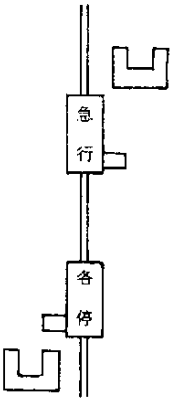


電車のコントロールのためレールを9区画に分ける。
電車位置を検出するためセンサーを置く。
急行用センサー　○
各停用センサー　・

2250　電車の種類と特性



急行1台、各停1台



スピードは電圧でコントロールする
電圧の範囲を4.5 V～100 Vと決める。

2260　コントロールの概要

1. 急行が2周してから各停がスタートする。
2. 各停は毎回停車する。
3. 急行は障害時以外停車しない。
4. 運行状況を表示する。
5. 鉄橋を設け水位によっては、渡橋しない。

2270　データ入力

① 電車位置検出センサー

② 水位検出

③ コントロール用タイマ

2280

表　示　装　置

○電車位置表示

○水位表示

2290

12 詳細設計を行う

① データ出力系の詳細設計

② データ入力系の詳細設計

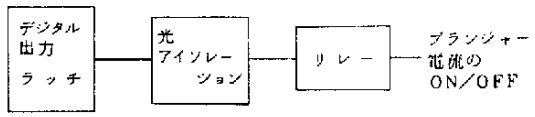
③ 表示系の詳細設計

④ ポートアドレスデータの詳細

詳細設計には以上の工程がある。

2320

ポイントの切替え

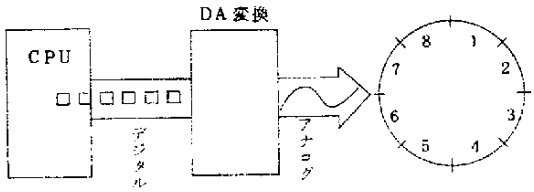


1ポート8BITの5ち4BITを使用

1つのポイントに2BIT×2ポイント

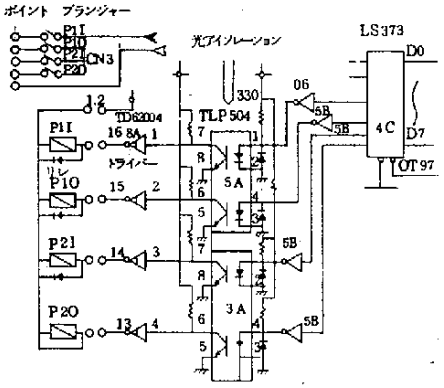
2300

レールへの電圧コントロール



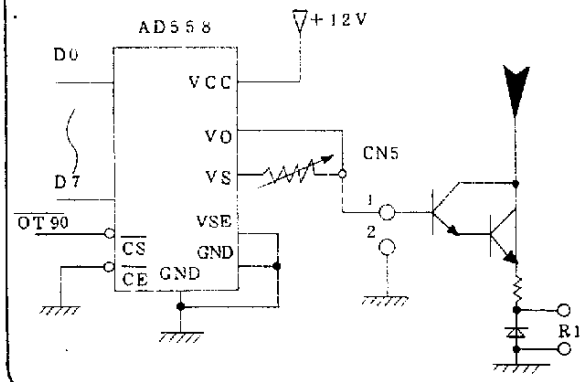
8ビットで電圧を256段階に制御できる。

2330



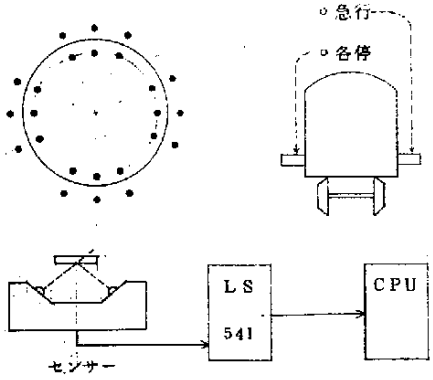
2310

DA コンバータ

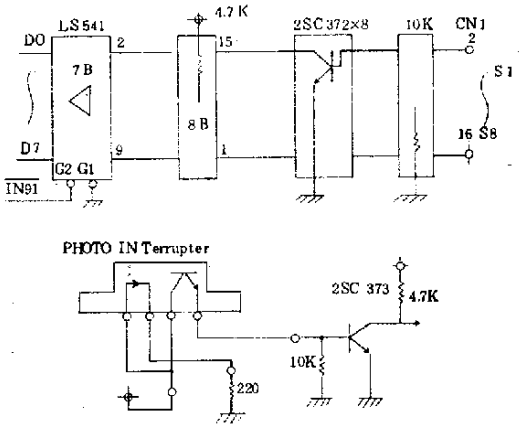


2340

電車位置検出

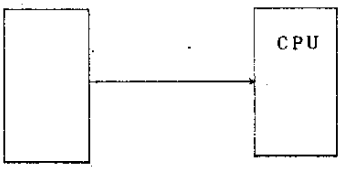


2350



2380

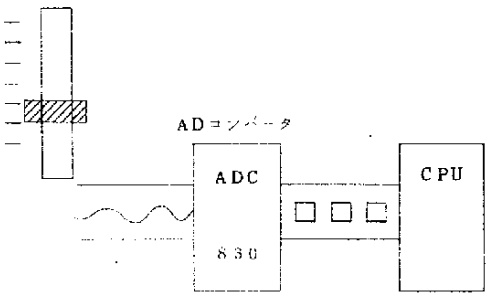
コントロール用タイマー



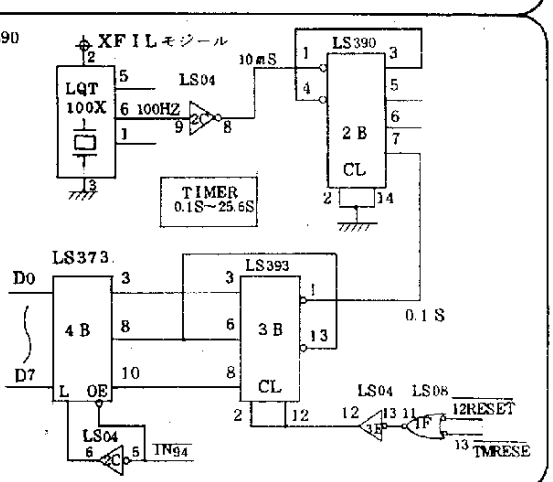
0.1 ～ 2 5.6 秒のクロックデータを取る。

2360

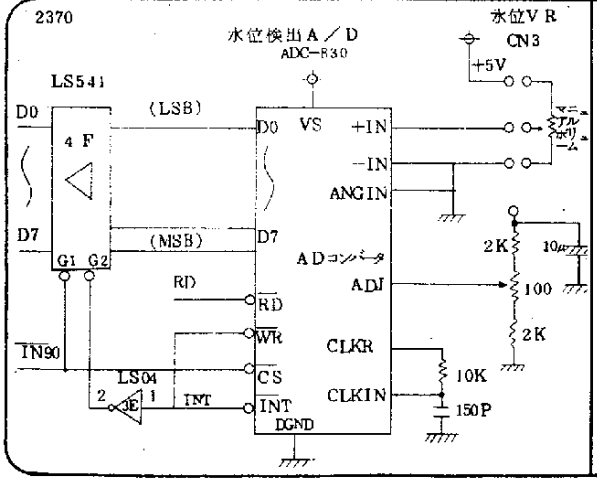
水位アナログ入力



2390

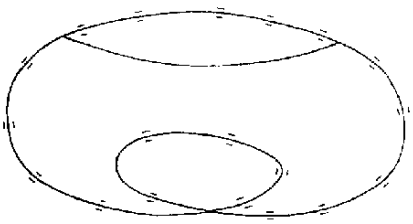


2370



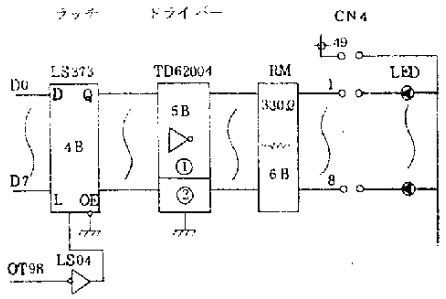
2400

電車位置表示パネル



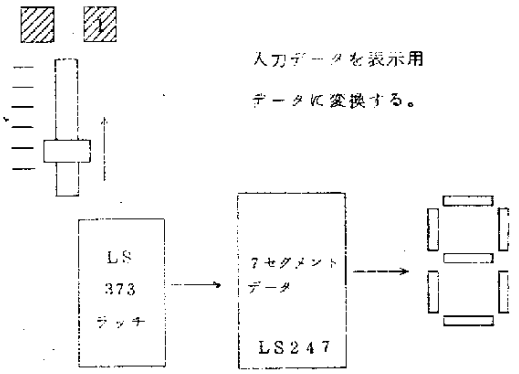
電車の通過点をＬＥＤで表示する。

2410



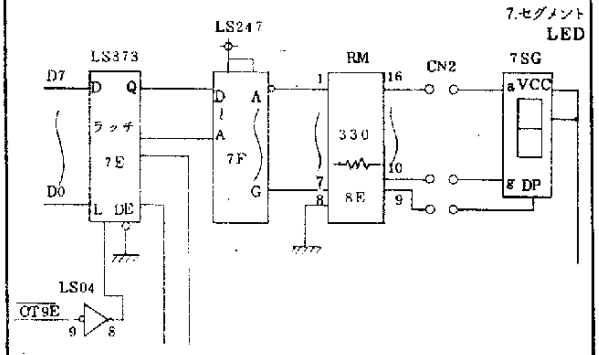
2420

水　位　表　示



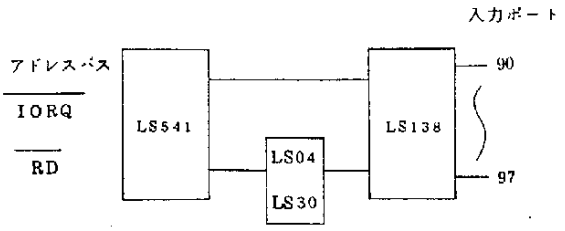
2430

7 セグメント　ルーチン

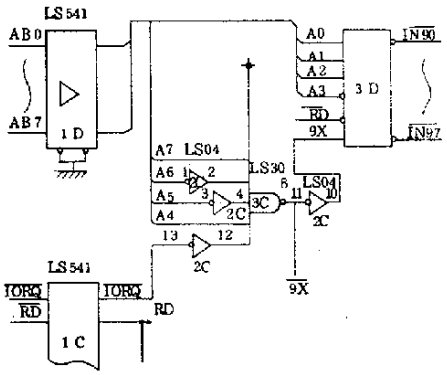


2440

入力ポートアドレスデコーダ

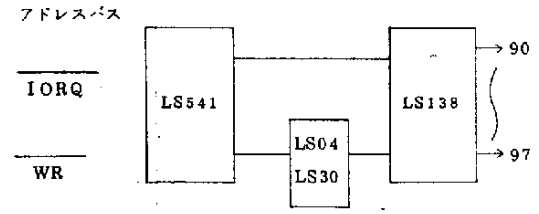


2450

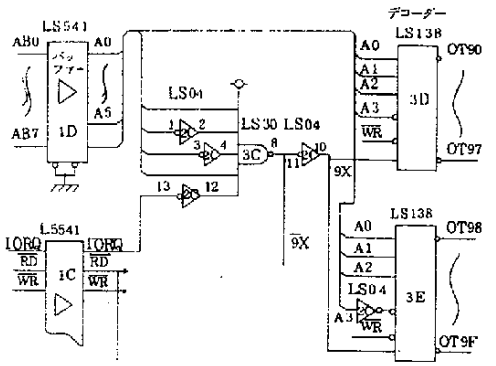


2460

出力ポートアドレスデコーダー



2470


LS541
AB0
A0
AB7
1D
A5
LS04
LS30
LS04
3C
13
12
LS541
IORQ
RD
WR
1C
IORQ
RD
WR
デコーダー
LS138
A0
A1
A2
A3
WR
3D
OT90
OT97
9X
9X
LS138
A0
A1
A2
LS04
A3
WR
3E
OT98
OT9F

3110　ソフトウェアの設計、製作の実例

① 動作手順の決定

② メインフローチャートの作成

③ プログラムエレメントの構成

④ メモリーの割り当て

⑤ I／Oアドレスの割り当て

⑥ プログラムの作成

⑦ CP／Mによるプログラミング

⑧ デバック

3120　①動作手順の決定

プログラムの目的・条件を明確にすることで仕様の決定となる。

○電車のスタート・ストップ条件

	スタート	ストップ
急行	・水位＞0 AND（位置センサNO3がON）	1.水位＞200 2.前の区間に電車がある。
各停	・位置センサNO7が2回センス	1.水位＞200 2.前の区間に電車がある。 3.位置センサNO1をセンスした時

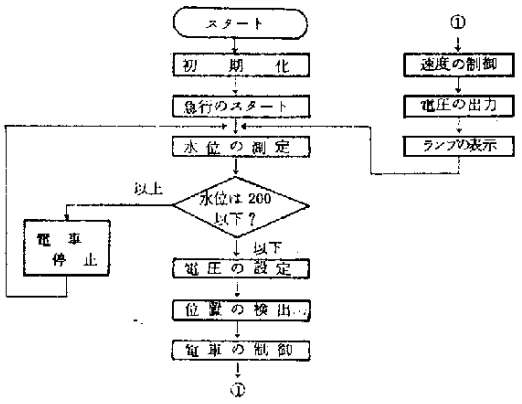
○速度と水位の関係

速度	高速	中速	低速
水位	0～100	101～150	151～200

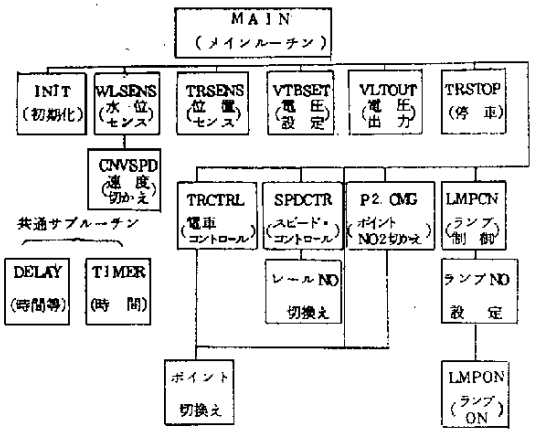
○ランプの表示タイミング

位置センサに対応するランプは電車の通過時にセンサーに対応していないランプは電車のスピードを計算してランプを点灯する。

3130　メインフローチャートの作成

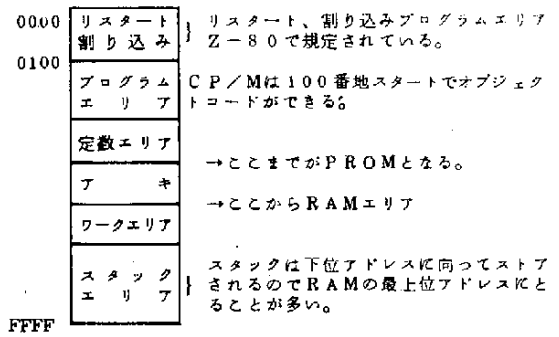


3140　プログラムエレメントの構成



3150　メモリーの割り当て

PROM（又はROM）とRAM部よりなるメモリーを割り当てる。



3160　I／Oアドレス（I／Oポート）の割り当て

Ⓘ 入力ポート

90H	：水位（0～FFH）	正論理
91～93H	：電車位置（ビット単位）	負論理
94H	：タイマー（0～25.5秒）	正論理

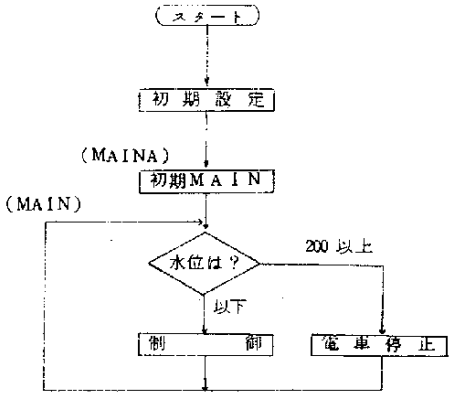
Ⓞ 出力ポート

90～96H	：区間電圧出力	正論理
97H	：ポイント切換ビット	正論理
98～9BH	：電車位置表示（ビット単位）	正論理
9EH	：水位表示	正論理
9FH	：タイマーリセット	正論理

3170　プログラムの作成

① メインルーチン
② 初期化ルーチン
③ 水位センスルーチン
④ 電圧の出力ルーチン
⑤ ポイントの切換えルーチン
⑥ ディレールーチン
⑦ スピードコントロールルーチン
⑧ ワークエリアの設定
⑨ 定数テーブルの作成
⑩ 初期選択画面

3180　メインルーチンのフローチャート



3190　メインルーチンのプログラム

```
        ORG   100H ← 100番地からスタートさせる。
START : LD    SP, 8FFFH ← RAMの最終番地にスタックポインターをセットする。
        CALL  INIT      ← 初期化ルーチンをCALLする。
MAINA :                 ┐
                        ┘ 初期スタート
MAIN  : CALL  WLSENS    ← 水位測定
                        ┐ 水位レベルからM100、M200にJUMPさせる。
M100  :                 ┐
                        ┘ 水位200以上で電車停止のコントロールを行い、再びMAINへ
        JP    MAIN      ┘
M200                    ┐
        JP    MAIN      ┘ 水位200以下で制御を行い再びMAINへ
```

3200　初期化ルーチンのプログラム

```
INIT  : XOR   A       ← Aレジスターをゼロにする。
        OUT   (97H), A ┐ 出力を全てゼロにする。
              :        ┘ I／Oの初期化である。
INIT10: LD    (HL), 00H ┐ ・HL番地からCバイト分をゼロにする。よく使われるパターンである。
        INC   HL        ┘ ・RAMワークエリアの初期化を行っている。
        DEC   C
        JP    NZ, INIT10
              :
        RET     ← 初期化を終了してメインルーチンにもどる。
```

3210　水位センスルーチンのプログラム

```
WLSENS: IN    A, (90H)  ← 90ポートからの入力
WLSO  : CP    10        ┐
        JP    C, WLS1   │
        SUB   10        │
        LD    C, A      │ ・バイナリーをデシマルに変換する。
        LD    A, B      │
        ADD   A, 16     │ ・よく使われるサブプログラムである。
        LD    B, A      │
        LD    A, C      │
        JP    WLSO      ┘
WLSI  : ADD   A, B
        OUT   (9EH), A  ← 水位の表示のポートへの出力
        CALL  CNVSPD    ← 速度切換ルーチンのCALL
        RET             ← メインルーチンへのリターン
```

3220　電圧の出力ルーチンのプログラム

このルーチンへの入力では既に出力データがワークエリアにストアされている。

```
VLTOUT : LD   HL, VOUT  ← VOUTのラベルの番地をHLに設定
         LD   A, (HL)   ← その番地のデータをAレジに入れる。
         OUT  (90H), A  ← 90ポート，つまり区間1に出力
         INC  HL        ← HLを＋1して次のデータ（区間2のデータ）を91ポートに出力する。
         LD   A, (HL)
         OUT  (91H), A
           :
         OUT  (96H), A
           :
         RET            ← 9区間まで出力したらリターンする。
```

3230　ポイントの切換えルーチンのプログラム

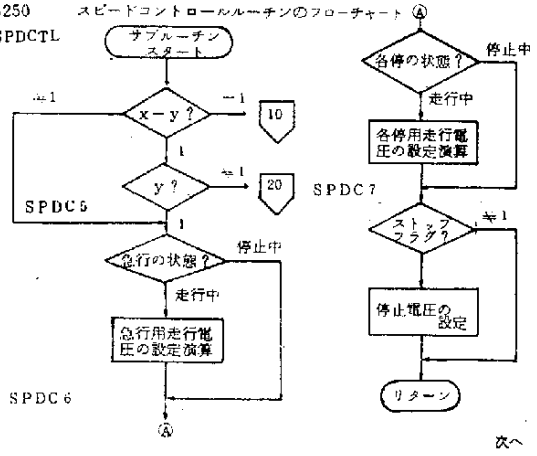
```
P2OUT : LD   A,40H      ←指定のビットを1にする。
        OUT  (97H),A    ←97ポートに出力する。
        LD   D20H       ┐
        CALL DELAY      ┘ 指定時間出力を変えない。つまり1を出したままにする。ディレールーチンを使う例。
        XOR  A          ┐
        OUT  (97H),A    ┘ ポート出力をリセットする。
        RET
```

3240　ディレールーチンのプログラム

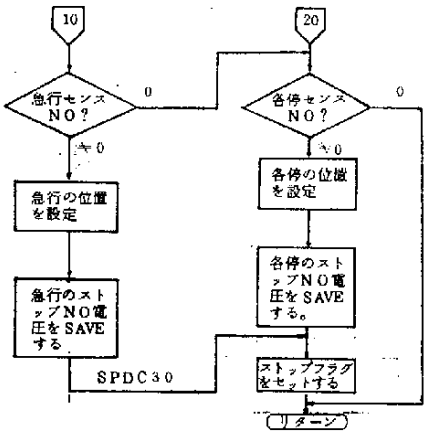
これはサブプログラムとしてよく使われる例である。
Dレジスターにディレー量を設定してCALL L (D)×5 ms 待ってリターンする。

```
DELAY : LD   E,0A6H
DLAY10: DEC  E          ┐
        JP   NZ,DLAY10  ┘ これで5 msディレーする。
        DEC  D
        JP   NZ,DELAY   ← (D)×5 msのディレーとなる。
        RET
```

3250　スピードコントロールルーチンのフローチャート Ⓐ



3260　SPDC10　SPDC20



3270　スピードコントロール、ルーチンのプログラム

```
SPDCTL : LD   A,(T1POSC)  ┐
         LD   HL,T2POSC   │ (T1POSC)−(T2POSC)の計算
         SUB  (HL)        ┘
         CP   −1         ←  −1と比較
         JP   Z,SPDC10    ←−1ならばSPDC10へ
         CP   1           ┐
         JP   NZ,SPDC5    ┘ 1でなければSPDC5へ
         LD   A,(T2POSC)  ┐
         CP   1           │ +2 POSCが1でなければSPDC20へ
         JP   NZ,SPDC20   ┘
SPDC5  : LD   A,(T1STS)   ┐
         CP   0           │ 急行が走行中でなければSPDC6へ
         JP   NZ,SPDC6    ┘
         ⋮                ┐ 急行用走行電圧の設定演算
```

3280

```
SPDC6  : ⋮                急行と同様の演算を各停について行う。
SPDC7  : LD   A,(STPFG)
         CP   1
         RET  NZ          ←  ストップフラグが1でなければリターン
         ⋮                ┐
                          ┘ 停止電圧の設定
         XOR  A           ←  Aレジスターをゼロにする。よく使われる命令
         LD   (STPFG),A   ←ストップフラグの設定
         RET              ←  リターン
```

3290

```
SPDC10 : LD   A,(T1SNS)   ┐急行センスNOが0ならば
         CP   0           │  SPDC20へ
         JP   E,SPDC20    ┘
           :              ┐急行の位置を設定
           :              │急行のストップNO
           :              │電圧をSAVEする
         JP   SPDC30      ┘
SPDC20 : LD   A,(T2SNS)   ┐各停センスNOが0ならば
         CP   0           │  リターン
         RET  Z           ┘
           :              ┐各停の位置を設定
           :              │各停のストップNO
           :              │電圧をSAVEする
SPDC30 :                  ┘ストップフラグのセット
         RET              リターン
```

3300　ワークエリアの設定

プログラム上でRAMのワークエリアを割り当てる。
RAMは8000番地からに割り当てられている。

```
        ORG   8000H ← 8000番地からがRAMエリア

VOUT :  DS    9     ← VOUTというワークエリアを
                      9バイト分割り当てる。
        DS    9       以下同様
          :
          :
```

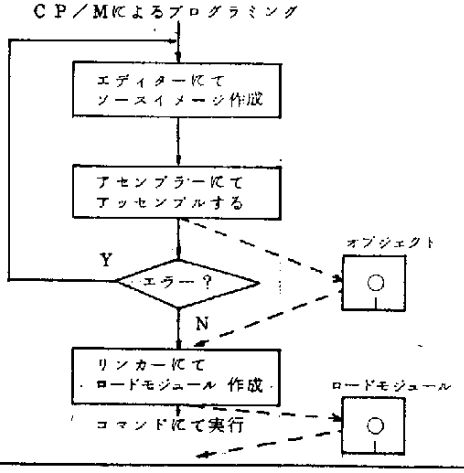
3310　定数テーブルの作成

ROM上に定数テーブルを作成する。
定数テーブルの先頭番地はプログラムの最終番地のすぐ後で625Hとなっている。

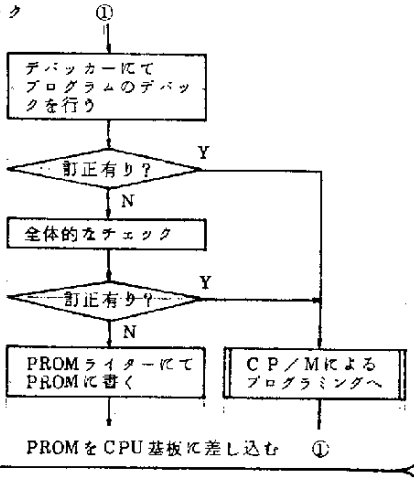
```
PLMPT1 : DB  001H  ← 001Hの1バイトをデータとして設定する。

         DB  010H
           :
           :
                     以下同様
```

3320　CP／Mによるプログラミング



3330　デバック　①



3. 構成および機能

ＣＡＩシステムの構成は次のとおりである。

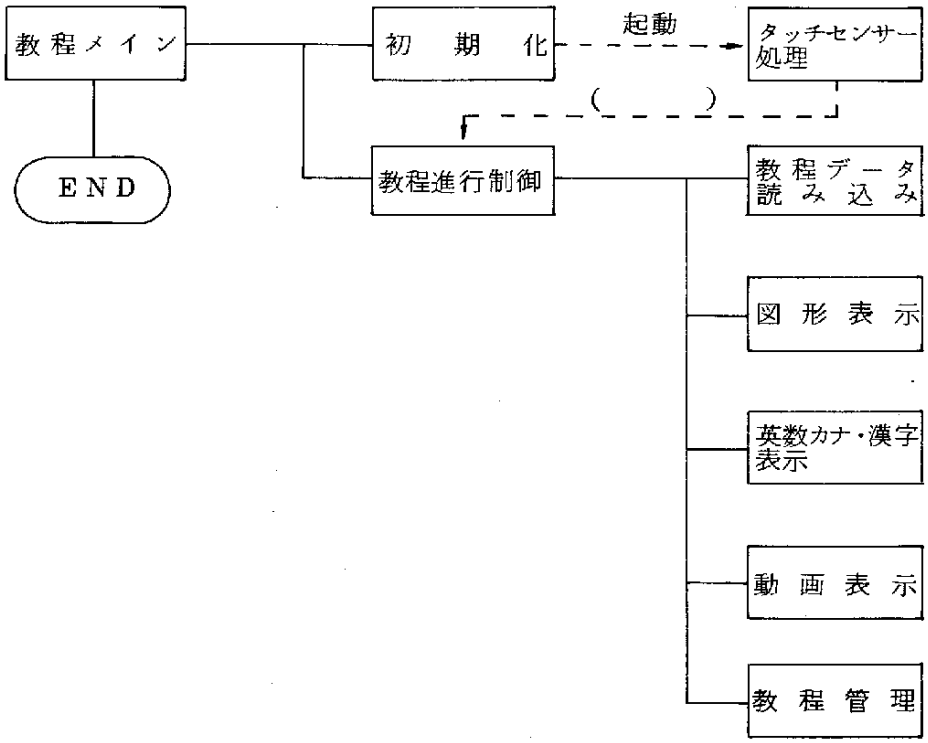


図３－１　ＣＡＩシステム構成図

各ソフトウェアのシステム関連図は次のとおりである。

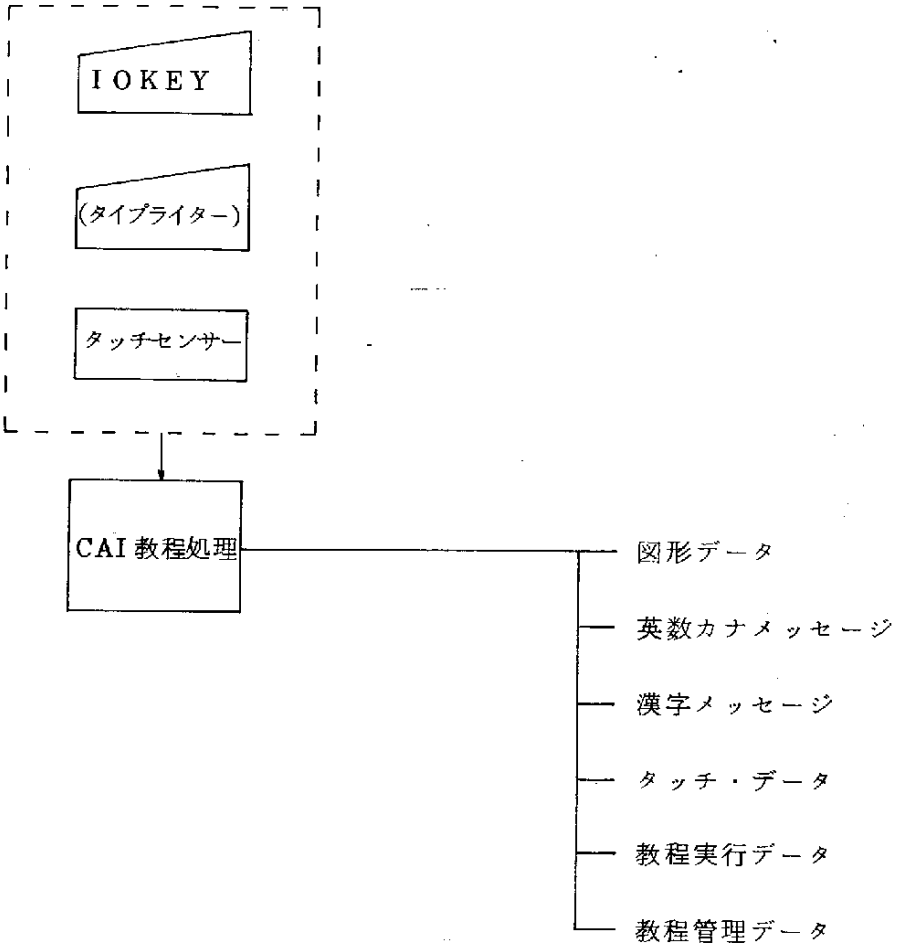


図３－２　システム関連図

3.1　データの種類

ＣＡＩシステムは以下にのべる各種データを利用している。

①　図形データ

各図形を座標ポイントで記憶する。各図形には番号を付けて登録し、呼出しは、その番号を指定することにより行われる。

番号は１～４００まで割当て可能である。

②　英数カナメッセージデータ

教程中に表示される英数カナタイプのメッセージを番号を付けて登録する。

呼出しは、その番号を指定することにより行われる。

番号は１～４００まで割当て可能である。

③　漢字メッセージデータ

教程中に表示される漢字タイプのメッセージを番号を付けて登録する。

呼出しは、その番号を指定することにより行われる。

番号は１～４００まで割当て可能である。

④　タッチデータ

タッチセンサー入力用のデータである。タッチセンサーは座標（Ｘ、Ｙ）で入力されるので、各教程ごとに座標範囲を登録しておき、タッチセンサー入力時これとチェックすることによりタッチセンサーの誤入力を検出できる。タッチデータは各教程に付けられた教程番号と一致している。

⑤　教程実行データ

各教程で他の教程へ移行するとき移行条件及び移行先の教程が記録されている。

初期値データ

各教程実行直前にあるデータを指定された値に置替えるためのデータである。

図形表示データ

画面上に指定された図形を表示するためのデータである。

⑥ 教程管理データ

学習者が教程のどこまで進んだかを記録しておくためのファイルである。

システム管理データ

基本データ、教程実行データ、漢字マスターの更新を行ったときに各データの格納最終ポイントを記録してある。これにより各データのメンテナンス時におけるファイル処理のスピード化を図る。

タッチ番号データ

教程実行中のタッチマーク番号を、入力された順に記録したものである。これはオウンコーディングプログラムで使用される。

昭和５９年 ３ 月　発行

発行所　財団法人日本情報処理開発協会
東京都港区芝公園３－５－８
機械振興会館内
ＴＥＬ（４３４）８２１１（代表）

印刷所　株式会社　昌　文　社
東京都港区芝５－２６－３０
ＴＥＬ（４５２）４９３１（代表）