第１部
「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部
「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

スライディングパズル

2019年7月10日　あすた会資料　　吉宗一夫

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

スライディングパズル

2019年7月10日　あすた会資料　　吉宗一夫

世界初のプログラマ

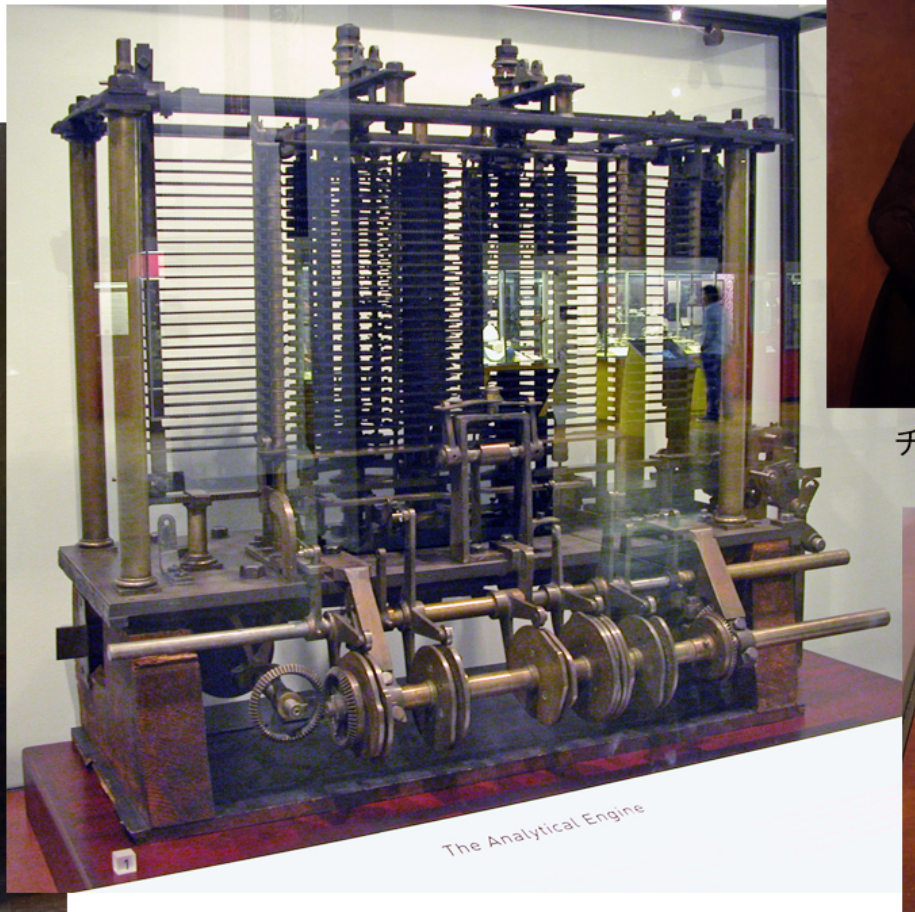
NHKのテレビドラマ「女王ヴィクトリア2」にバベッジの解析機関と女性数学者ラブレスが登場する。

チャールズ・バベッジは英国の数学者で、1830年代に蒸気機関を用いて自動計算を行う「解析機関」を考えついた。これは当時フランスで開発されたジャガード織機の孔あきカードにヒントを得て、純機械的にプログラム情報を読取って計算を行うものだった。完成すれば長さ30m、幅10mになる非常に大掛かりな装置で、完成に至らなかったが、**世界初のプログラミング可能な自動計算機**といわれる。

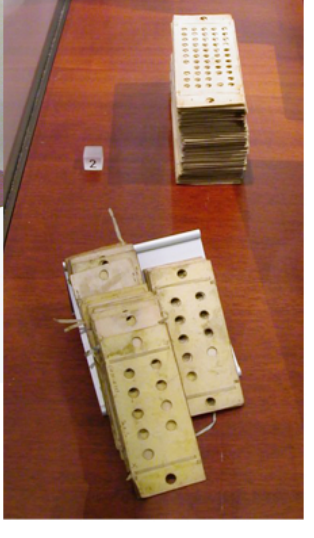
エイダ・ラブレスは英国の貴族で詩人バイロンの一人娘であり、数学を愛好した。エイダはバベッジの解析機関に強い興味を示し、バベッジの解析機関に関する仏語の講演記録を英語に翻訳した。この著書の訳注の中にある解析機関用のプログラムは、世界初のコンピュータプログラムと言われ、エイダは**世界初のプログラマ**とされた。1980年、米国国防総省は国際競争入札で仕様を策定した組み込みシステム向け言語をAdaと名づけた。

バベッジの解析機関

エイダ・ラブレス



チャールズ・バベッジ



バベッジ自身が組み立てた解析機関の一部の試作品（上）
とプログラム用の２種類のパンチカード（右）
サイエンス・ミュージアム（ロンドン）

メインフレームの時代

	＜プログラミング＞	＜米国＞	＜日本＞	<NEC>	備考
1946		ENIAC			
1947					
1948					
1949					
1950					
1951	**マイクロプログラミング方式の提案**	UNIVAC Ⅰ			
1952		IBM701			
1953		IBM702 ／ IBM650			
1954					
1955					
1956		IBM Stretch プロジェクト開始	ETL-MARKⅢ		
1957	FORTRAN		MUSASINO-1		
1958	ALGOL			NEAC-1101	
1959	COBOL	**IBM7090 ／ IBM1401**		NEAC-2201（パリ展示会に出品）	
1960					
1961			JECC 設立		
1962				**Honeywell と技術提携**	
1963		**Honeywell 200 シリーズ**			
1964		**IBM システム 360**	FACOM230 発表／ HITAC5020 完成	NEAC-2200 発表	
1965			FACOM270 ／ HITAC8000 発表	**NEAC シリーズ 2200 発表**	
1966	FORTRAN 66			シリーズ 2200M500 完成	
1967	LOGO				電気進学
1968				シリーズ 2200M700 発表	
1969					
1970		**IBM システム 370 ／ GE 撤退、HIS 社設立**			
1971		**RCA 撤退**／ Intel 4004	DIPS- 1	HIS 社と新機種共同開発で合意	**NEC 入社**
1972	C			東芝と技術研究組合を設立	
1973					
1974			**富士通・日立 M シリーズ発表**	**ACOS シリーズ 77 中小型機出荷**	
1975			コンピュータ輸入自由化		
1976				TK-80	
1977					
1978	「プログラミング言語 C (K&R)」	Intel 8086	東芝撤退		
1979				PC-8001	
1980	Smalltalk 公開	Motorola MC68000			
1981	MS-DOS	IBM PC	**IBM 産業スパイ事件**		

NECのメインフレーム事業

NECは1958年にパラメトロンを用いて初めてデジタル計算機NEAC-1101を完成。これに続いてトランジスタ式計算機NEAC-2201を完成したが、1960年代に入って「ソフトウェア」の重要性に気づき、1962年に米国Honeywell社と技術提携した。

Honeywell社はIBMのビジネス用途コンピュータ1401に類似したアーキテクチャのH200を開発し、1401のプログラムをリコンパイルなしに実行できて価格性能比に勝ることで売り上げを伸ばしていた。NECはこれをノックダウン生産してNEAC-2200として販売した。

IBMは1964年にシステム360を発表。これは従来からの科学技術計算用701シリーズやビジネス用1401シリーズなどを統合するバイトマシンで、小型機から大型機まで統一されたアーキテクチャを有するファミリーシリーズとして画期的なものであった。

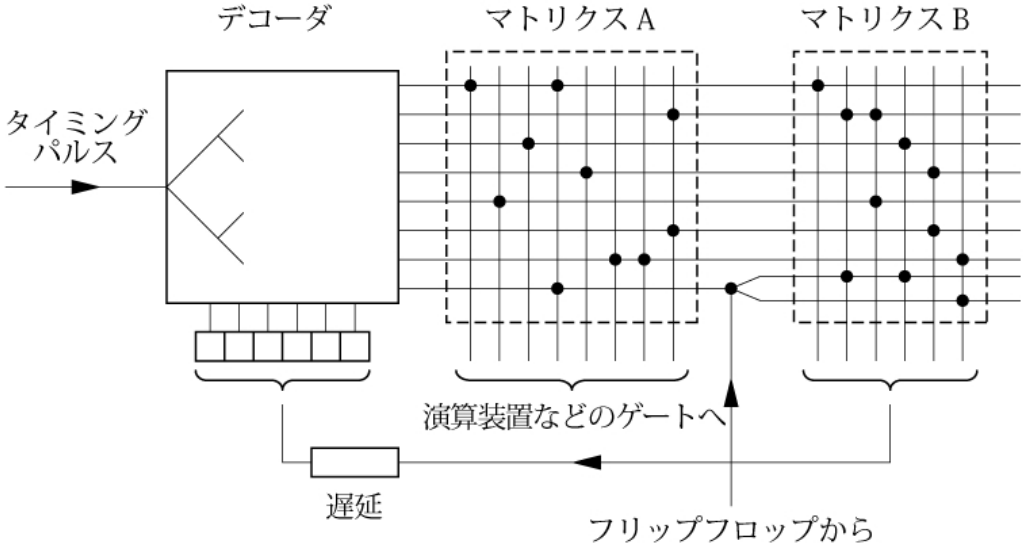
NECはシステム360に対抗して1965年に国産初のワンマシンコンセプトのシリーズとしてNEACシリーズ2200を発表。これはHoneywellのH200シリーズにNEC独自開発の大型機モデル500を加えたものであった。

NECはその後も2200シリーズに最上位機種モデル700を追加するなどシリーズの強化を続け、またDIPS開発にも注力したが、真にシステム360に対抗する新機種の開発はHoneywellに依存して遅れていた。

システム360で大成功を収めたIBMはさらに1970年に仮想記憶機能等を追加したシステム370を発表。IBMの優位が決定的になる中で、GE、RCAが相次いでコンピュータ事業から撤退した。Honeywell はGEのコンピュータ部門を傘下に入れてHIS (Honeywell Information Systems) 社を設立。NECは、GEと提携していた東芝も含めて、IBMシステム370対抗の新機種を共同開発することになった。

マイクロプログラムとは

1951年、モーリス・ウィルクスは、Whirlwind（1947年にMITが開発）で用いられたコントロールストアによる制御に「条件付実行」の概念を導入し、「マイクロプログラム方式」と名付けた。（これに対し従来の論理回路による方式を「wired logic」という）

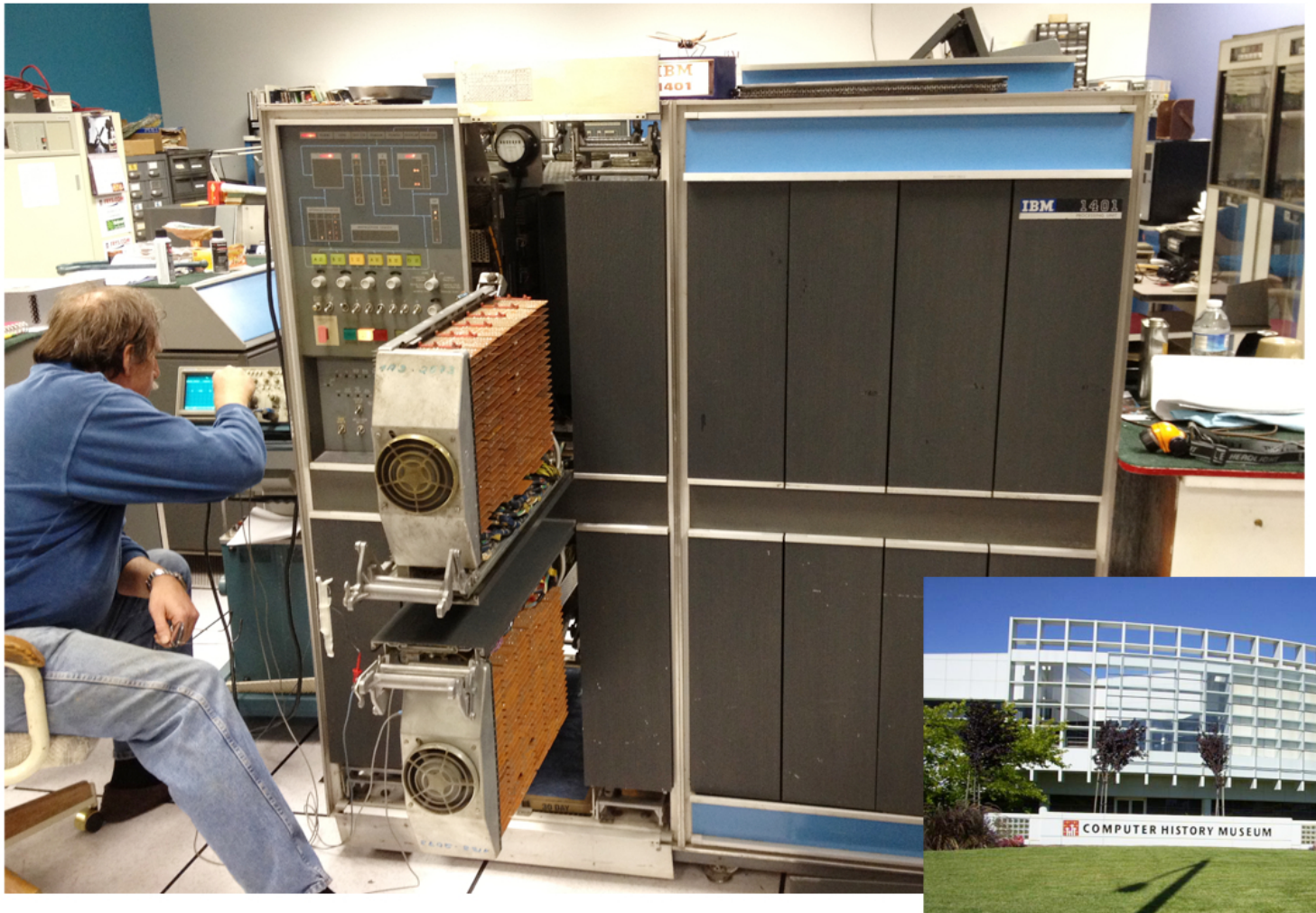


Wilkesのマイクロプログラム制御のモデル

IBMシステム360の制御記憶

モデル	容量	ビット数	サイクルタイム
30	4 k	50＋5	750 ns
40	4 k	52＋2	625 ns
50	2.75 k	85＋3	500 ns
65	2.75 k	87＋4	200 ns
75	なし		

IBMシステム360は小型から大型モデルまで統一された命令体系を実現するためにマイクロプログラムを利用し、他社もこぞってこの手法を取り入れた。



コンピュータ歴史博物館（右）の IBM1401（上）

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

スライディングパズル

2019年7月10日　あすた会資料　　吉宗一夫

2200エミュレータ

NEC入社後に配属された部署は新機種のアーキテクチャを担当し、HIS社から送られてくる新機種関連資料を読み解くのが仕事だった。新機種のアーキテクチャは、データ形式や演算命令に関してはIBMシステム360に類似したものだったが、プロシジャ呼び出し時のスタック生成や、セマフォを用いたプロセス間同期とプロセス切り替えなどのOS機能をマイクロプログラムで実行するという点に特徴があった。

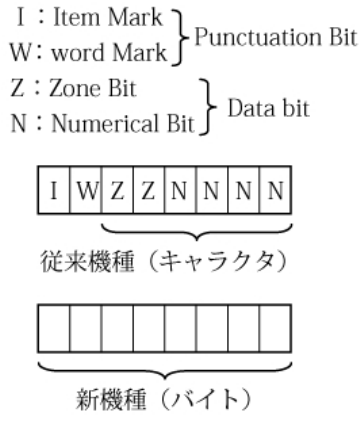
> マイクロプログラムを利用すれば、複数の機械語命令をメモリから読み出す時間を削減でき、さらにマイクロプログラム・レベルだけで利用できるレジスタ群を有効に利用して処理時間を短縮できると考えられた。

担当した新機種関連資料の１つに従来機種2200シリーズのプログラムを新機種上で実行するエミュータの資料があり、従来機種においてNECが独自に追加した機能についてマイクロプログラムの変更を要することに気づいて、その改造設計を担当することになった。

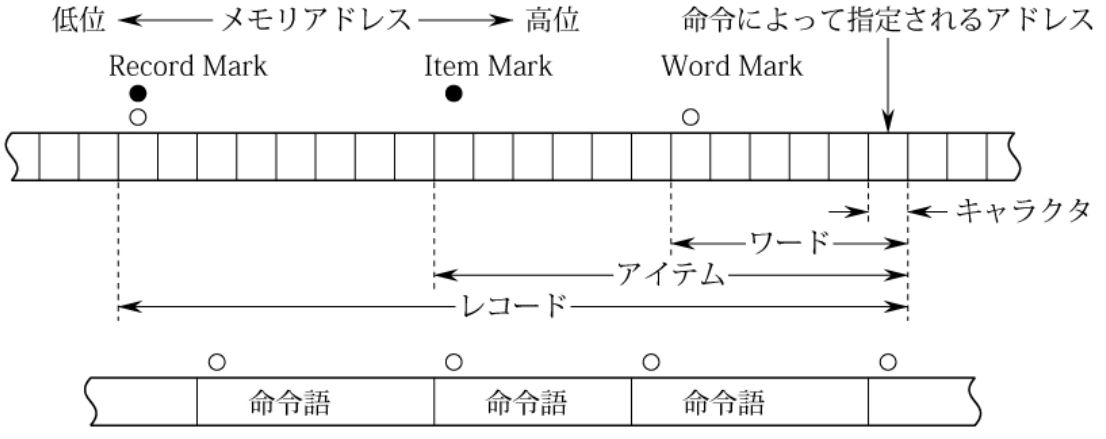
新機種（ホストマシン）の情報単位は8ビットのバイトであるのに対し、エミュレーションの対象となる従来機（ターゲットマシン）の情報単位は6ビットのキャラクタなので、これを効率的に処理するためにエミュレータは下記で構成されていた。

- キャラクタ単位のデータや可変長命令を操作するための付加的な演算器
- ターゲットマシンの命令実行を制御するマイクロプログラム
- ターゲットマシンの入出力要求をホストマシンの入出力動作でシミュレーションするソフトウェア

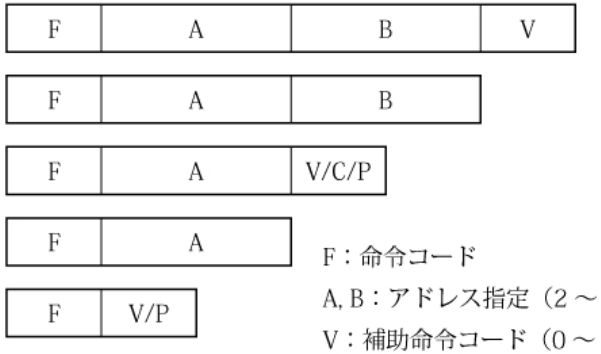
従来機種と新機種のアーキテクチャ比較



＜メモリのアドレス単位＞

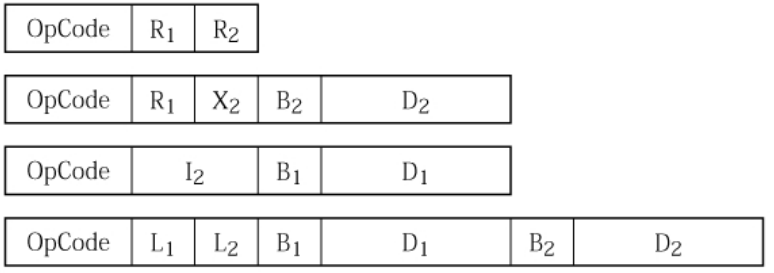


＜従来機種における Punctuation Bit の役割＞



F：命令コード
A, B：アドレス指定（2 ～ 4 キャラクタ）
V：補助命令コード（0 ～ 3 キャラクタ）
C：制御情報（1 ～ 11 キャラクタ）
P：パラメータ（1 ～ 2 キャラクタ）

＜従来機種の命令語（可変長）＞



OpCode：命令コード
R：汎用レジスタ指定　　X：インデックスレジスタ指定
B：ベースレジスタ指定　D：アドレス変位指定
I：イミディエート　　L：オペランド長指定

＜新機種の命令語（固定長）＞

独自アーキテクチャ vs. IBM互換

新機種は1974年に「ACOSシリーズ77」の名称で発表されたが、最初に出荷された中型機はOSの開発が間に合わず、しばらくはエミュレータ機能だけで運用された。

当時、富士通と日立はIBMシステム370対抗機としてIBM機と互換性を持つMシリーズを開発し、いわゆる「コンパチ路線」を選択した。

NEC内でもIBM互換の必要性が議論され、IBM機エミュレータのマイクロプログラム試作を担当した。NECの新機種のデータ形式や命令フォーマットはIBM機に類似したものだったので、マイクロプログラムの変更でエミュレータを実現することに大きな困難はなかった。入出力機能のエミュレーションを行うソフトウエアも2200のエミュレータのソフトウェア担当者らが開発し、OSの立ち上げまで行った。

IBMは1981年に論理アドレスの拡張や入出力機能の大幅な強化を含むsystem/370-XAアーキテクチャを発表。この技術は特許と著作権で守られていたため互換機対策とも言われ、この技術をめぐりIBM産業スパイ事件も発生した。このような状況から公開されている資料だけでは互換機ビジネスは困難との判断に至り、その後、IBM互換機が議論されることはなかった。

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

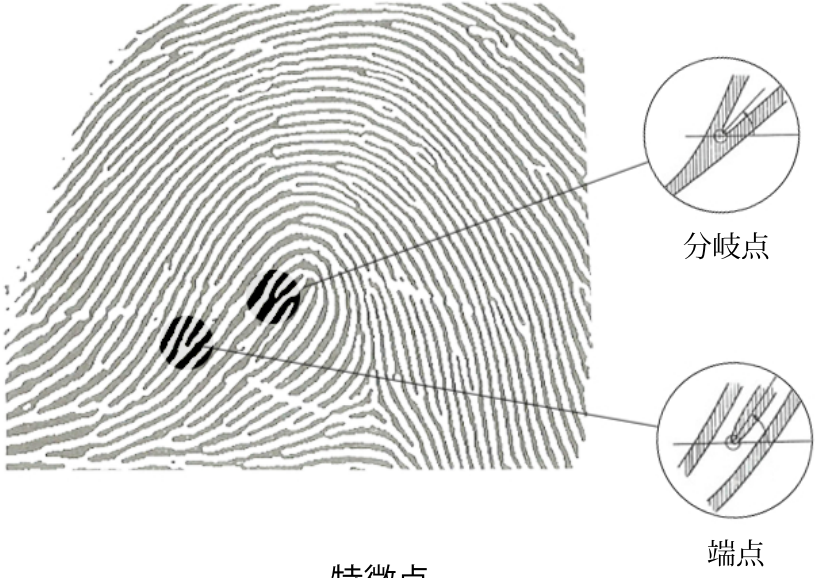
ペントミノ

スライディングパズル

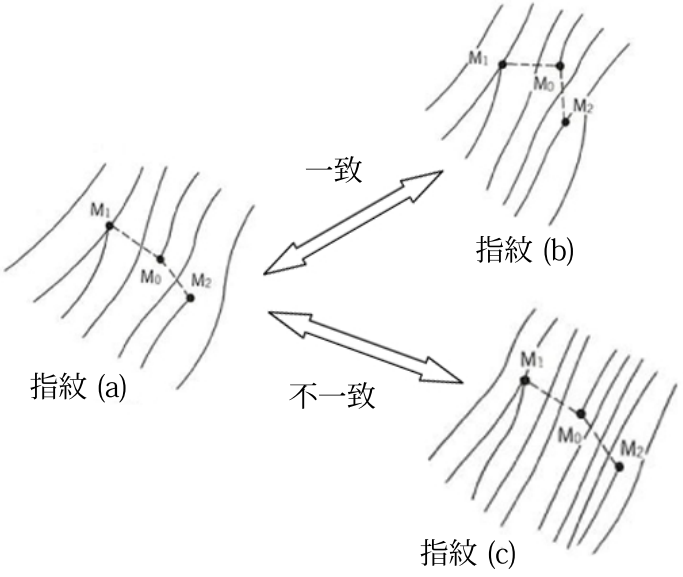
2019年7月10日　あすた会資料　　吉宗一夫

指紋の識別方法

指紋の隆線（指先にある汗腺の開口部が隆起した線）の端点と分岐点を特徴点（マニューシャ：minutiae）といい、指紋照合は特徴点の位置関係や特徴点間の隆線の数（リレーション）に着目して行う。



特徴点

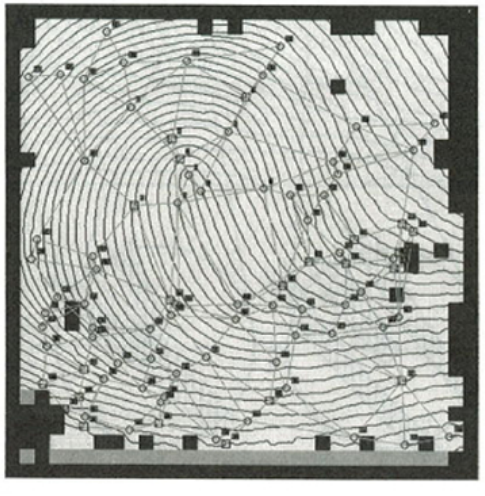


特徴点とリレーションによる照合

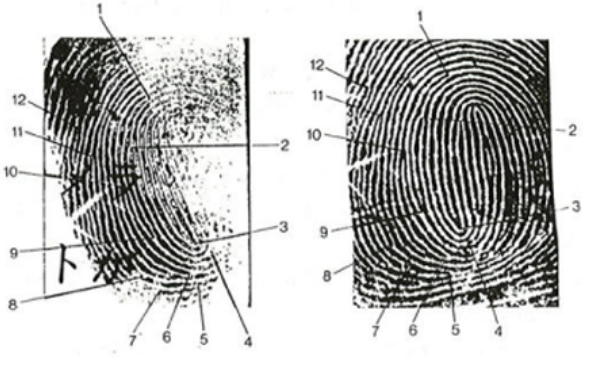
特徴点の数は日本人の場合、50～160点（平均で約100点）あり、12個以上の特徴点が一致すれば、裁判での証拠能力があるとされる（ほとんどの国で同じ基準が採用されている）。

AFIS（指紋自動識別システム）

NECは1971年から警察庁の指導を得て指紋自動識別システム（AFIS：Automated Fingerprint Identification System）の研究を始めた。これは従来から手作業で行っていた指紋照合業務を自動化するもので、特徴点を自動的に抽出し、照会指紋（現場から採取された遺留指紋）とファイル指紋（蓄積されている押捺指紋）の類似度を計算する。1982年に警察庁に納入後、1984年にはサンフランシスコ市警察へも納入し、国内外に多くの納入実績がある。

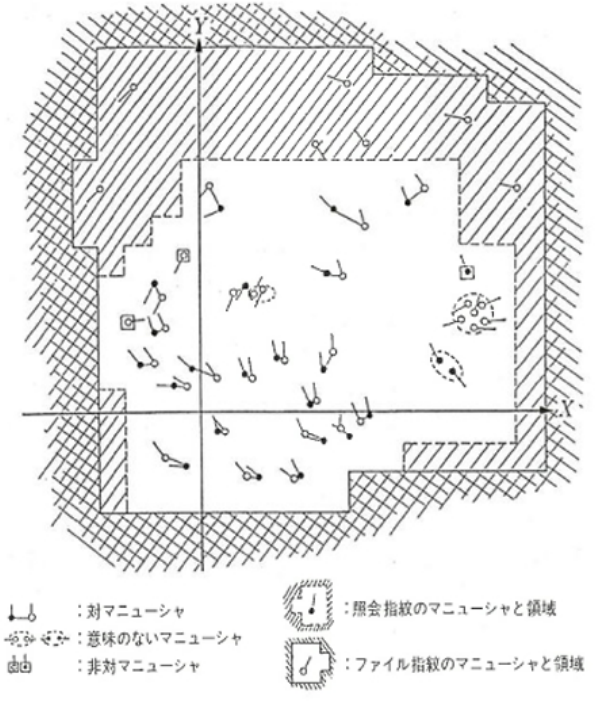


特徴点の抽出



(a) 遺留指紋　(b) 押捺指紋

照会指紋とファイル指紋の照合

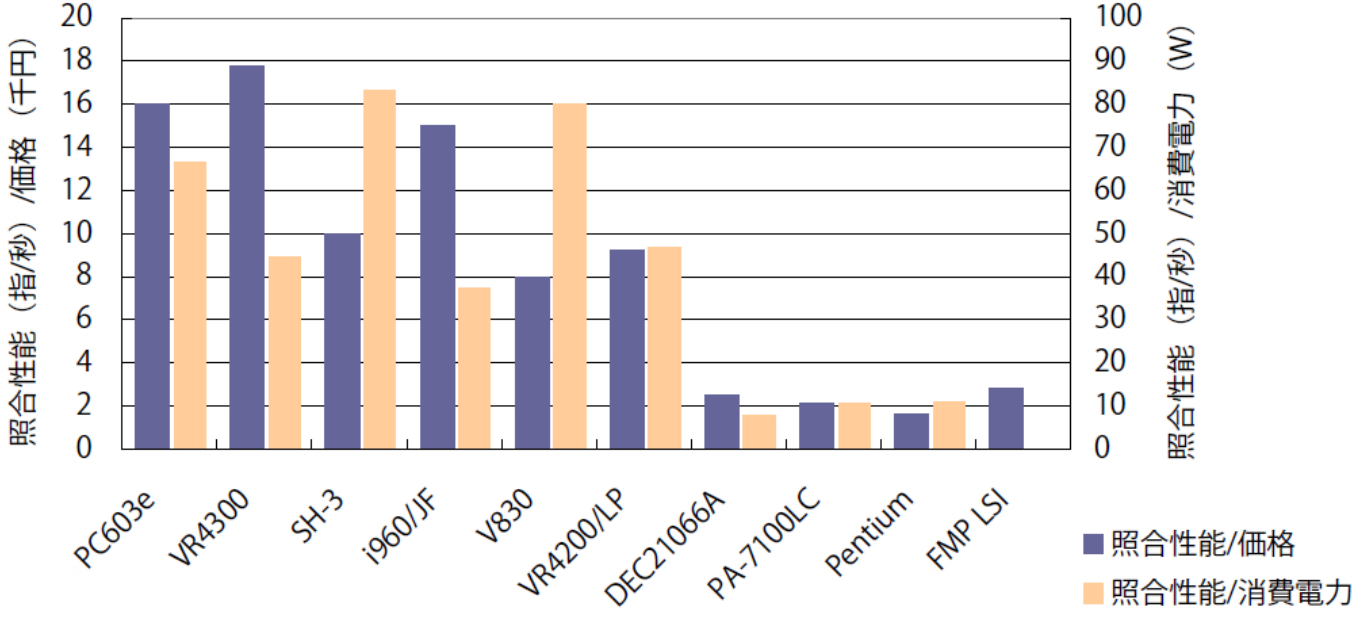


類似度の計算

汎用プロセッサの採用

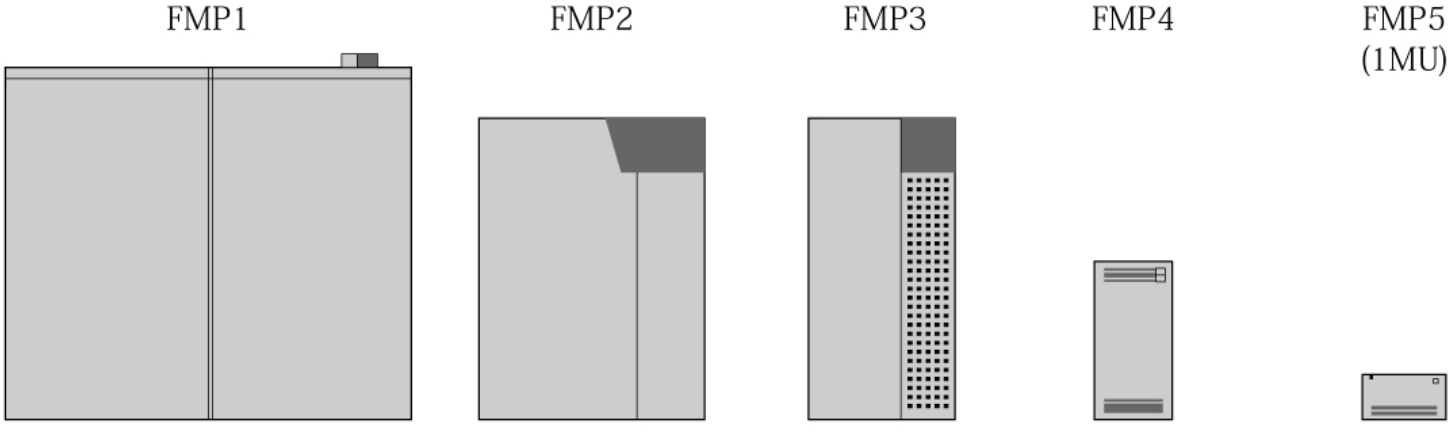
AFISは入力／照合／確認の各サブシステムや遠隔地からの照会を可能とする通信ネットワークからなり、コンピュータ部門が担当する指紋照合装置（FMP：Fingerprint Matching Processor）は、特徴点探索のための座標変換などを高速化するため、専用に設計されたLSIを搭載していた。

1990年代中頃になってマイクロプロセッサの性能向上が著しく、汎用プロセッサを利用してFMPを作れる可能性が出てきた。そこで、照合アルゴリズム開発に用いられたFORTRANプログラムをC言語に書き直し、FMPの精度／性能確認用指紋データを用いてプロセッサの照合性能を比較した結果、PowerPCを採用することになった。



プロセッサ照合性能比較

指紋照合装置の変遷



	FMP1	FMP2	FMP3	FMP4	FMP5
照合制御装置	ACOS S250（内蔵）	ACOS S410（内蔵）	ACOS S3300（内蔵）	UP4800（分離）	UNIX サーバ（分離）
MU (照合ユニット) 数	4	4	4	4	1
照合性能（指 /sec）	800	800	1600	1600	3200
消費電力（kVA）	16	5.6	3	1.2	0.1
テクノロジー	CML	CMOS-4	CMOS-6	CMOS-6	汎用プロセッサ
外形寸法 (W×D×H mm)	1800×760×1560	1000×760×1335	650×760×1335	345×760×700	370×300×200
コスト性能比	1	2	9	20	120
出荷時期	1982/6	1987/6	1991/10	1994/6	1996/6

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

スライディングパズル

2019年7月10日　あすた会資料　　吉宗一夫

FORTRAN CODING SHEET

C FOR COMMENT

STATEMENT NUMBER | CONT | FORTRAN STATEMENT

```
C  SIEVE OF ERATOSTHENES
      DIMENSION IPRIME(1000)
      N=1000
      DO 10 I=1,N
      IPRIME(I)=1
   10 CONTINUE
      DO 20 I=2,N
      IF (IPRIME(I).NE.1) GO TO 20
      WRITE(6,100) I
      IF (I*I.GT.N) GO TO 20
      DO 30 J=I,N,I
      IPRIME(J)=0
   30 CONTINUE
   20 CONTINUE
  100 FORMAT(I3)
      STOP
      END
```

D:¥My Desk¥Program¥FORTRAN&C¥prime.c - 秀丸

ファイル(F) 編集(E) 表示(V) 検索(S) ウィンドウ(W) マクロ(M) その他(O) 21:1

```c
#include <stdio.h>

/* sieve of Eratosthenes */
void main(void)
{
    int iprime[1000], n = 1000;

    for (int i = 0; i < n; i++)
        iprime[i] = 1;
    for (int i = 2; i < n; i++) {
        if (iprime[i] != 1)
            continue;
        printf("%3d¥n", i);
        if (i * i > n)
            continue;
        for (int j = i; j < n; j += i) {
            iprime[j] = 0;
        }
    }
}
[EOF]
```

日本語(Shift-JIS) CR+LF 挿入モード

C言語

C言語は、1972年にAT&Tベル研究所のデニス・リッチーが主体となって、ケン・トンプソンが開発したB言語を改良して誕生した。後にUNIXは大部分をC言語によって書き換えられ、UNIX上のプログラムはC言語を広く利用するようになった。

システム記述言語として開発されたため、高級言語であるが、ポインタ演算、ビットごとの論理演算、シフト演算などのアセンブラ的な低水準の操作ができる。また、仕様の一部にあえて「処理系に依存する」とする自由度を残して、プラットフォームやプロセッサアーキテクチャとの相性による有利不利が生じないようになっている。

> たとえば、3種の整数型変数は、そのサイズが (short) <= (int) <= (long) であるという大小関係だけが規定され、実際のビット数は規定されていない。また、文字型変数 char の符号の有無やシフト演算における符号ビットの扱いは処理系依存である。

言語仕様が小さいためコンパイラの開発が容易だったこと、UNIX環境での実績があり、K&R といった解説文書が存在していたことなど、さまざまな要因からC言語は利用者を増やし、1990年代中盤以降は最初に学ぶプログラミング言語として主流となった。

近年になると、オブジェクト指向の普及、GUI環境の普及などにより、C++，Java，JavaScript，C#など、「C系」と呼ばれる多くの派生言語が生まれた。

K&R

リッチーとカーニハンの共著である「The C Programming Language」は1978年に初版が出版され、1989年にANSIでC言語が標準化されるまでは実質的な標準として参照された。著者名の頭文字から「K&R」と呼ばれてC言語の普及に大きな役割を果たし、第 1 章の最初に例題として掲載されている "hello, world" プログラムは、「プログラミングの最初の例題」として定番となった。

また、ALGOLの思想を受け継いでブロック構造およびスコープをサポートするが、「K&R」に記述されたブロックを表す括弧の位置や字下げのスタイルはK&Rスタイルとして広く知られるようになった。

ポインタ

ポインタは他の変数のアドレスを内容とする変数であり、Cでは頻繁に使用される。たとえば、C言語では文字列を扱う特別な型は存在せず、ヌル文字（ゼロ）を終端とするchar（文字）型の配列を利用するが、そのコピーはポインタとC独特のインクレメント演算を使って下記のように簡潔に書ける。

```
void strcpy(char *s, char *t)     /* t を s にコピーする */
{
    while (*s++ = *t++)           /* C では非ゼロは true、ゼロは false */
        ;
}
```

アセンブラ経験者から見れば自然で便利なポインタであるが、一部のプログラミング初心者には難解で評判が悪く、「C言語ポインタ完全制覇」などという解説本が数多くある。

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

スライディングパズル

2019年7月10日　あすた会資料　　吉宗一夫

マイクロアーキテクチャの発展

回路素子の速度向上によるクロック周波数向上に加え、集積度向上で利用できるようになったハードウェア量を活かして性能向上を図るためにパイプライン処理が利用され、命令実行の並列度増加やパイプラインバブル減少のための技法が発展してきた。

- スーパーパイプライン
- スーパースカラー
- アウトオブオーダ実行とレジスタリネイミング
- 遅延分岐
- 分岐予測と投機実行

＊1 ソフトウェアから見える命令アーキテクチャに対し、これを実現するハードウェアの内部構造をマイクロアーキテクチャという。

＊2 パイプライン処理は IBM Stretch (1961) で分岐予測とともに採用され、以後のメインフレーム大型機でも広く使用された。

＊3 パイプラインに連続的に命令を送り込めずパイプラインの動作に生じる空白をパイプラインバブルといい、①ハードウェア資源の競合に起因する構造ハザード、②データの依存関係に起因するデータハザード、③分岐命令、割り込みなどに起因する制御ハザードの３つがある。

しかし、近年になって発熱量の問題などからクロック周波数の限界が意識され、また、命令実行の並列度はプログラム中の命令間に存在する本質的な依存関係に制限されるので、ハードウェア・マルチスレディングやマルチコアなど、ひとつのプロセッサチップで複数の命令スレッドを実行する方向になってきた。

CISCとRISC

初期のアセンブラ中心のプログラミング環境では、高機能で複雑な命令や、個々の演算命令に任意のアドレッシングモードを組み合わせられる（直交性がある）ことが好まれ、また、主記憶は高価であったから、高機能な命令や可変長命令の利用でプログラム容量を縮小できることが利点であった。さらに複雑な機能をマイクロプログラムで実現すれば、多くのソフトウェア命令を組み合わせるより高速に実行できるとの期待もあった。

高級言語が普及してコンパイラが生成するコードを解析すると、実際に使用される命令は限られたものであり、高機能な命令や直交性の利点がないことがわかった。また、処理に多くのステップを要する複雑な命令はパイプライン処理に馴染まず、性能向上の妨げと考えられるようになった。

そこで、複雑な機能の命令とアドレッシングモードを削減して回路を単純化し、その分を演算器やキャッシュなどにまわして性能向上を図る手法が提唱された。このようなアーキテクチャをRISC（Reduced Instruction Set Computer）といい、従来のものはCISC（Complex Instruction Set Computer）と呼ばれるようになった。

RISCプロセッサの開発は1970年代後半から1980年代初頭にかけて始まり、1990年代になって多くのワークステーションがRISCプロセッサを採用するようになった。

1991 MIPS R4000（NECもサーバに採用）　　1992 SUN SPARC 1
1992 DEC Alpha 21064　　1994 IBM PowerPC601

8086から80286、386と従来プロセッサとの互換性を重視して機能を拡張してきたIntelのx86アーキテクチャはCISCの典型であるが、1995年に出荷を開始したPentium Pro 以降の Intel プロセッサはCISC命令をプロセッサ内部でRISC風命令に変換して実行する技術を採用している。

RISCアーキテクチャの特徴

① ロード／ストア・アーキテクチャ

　演算はレジスタ・レジスタ間の演算に限り、メモリをアクセスする命令は単純なロード／ストア命令に限る。アドレッシングモードの多様化による命令数の増加とメモリアクセスのレイテンシがパイプライン動作に与える悪影響を避ける。

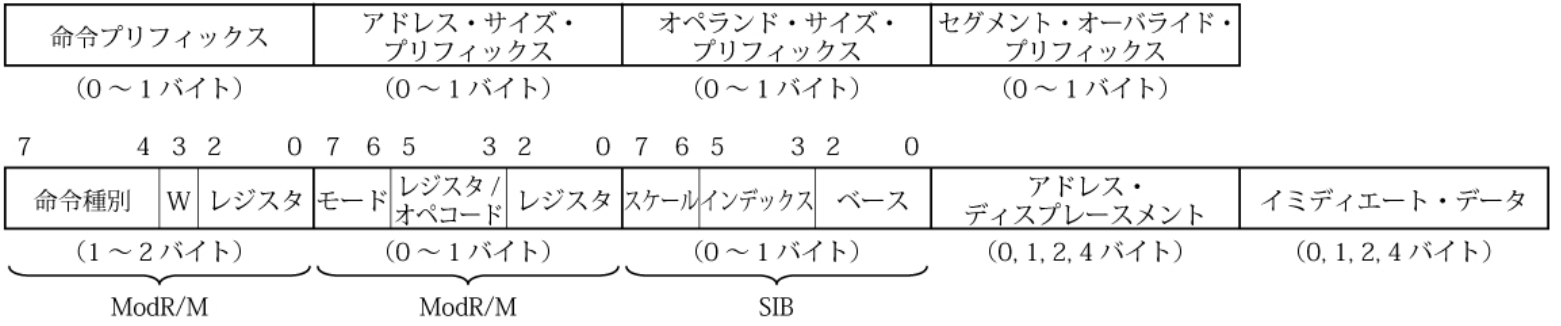
② 豊富な汎用レジスタと３オペランド命令

　汎用レジスタの数を増やして演算の途中結果をCPU内に蓄え、メモリへのアクセスを減らす。演算命令中で不要になったメモリアドレスを指定するビットを、汎用レジスタの指定に充ててプログラミングしやすい３オペランド命令を実現し、また、即値（イミディエート）の指定に充ててメモリアクセスを削減する。

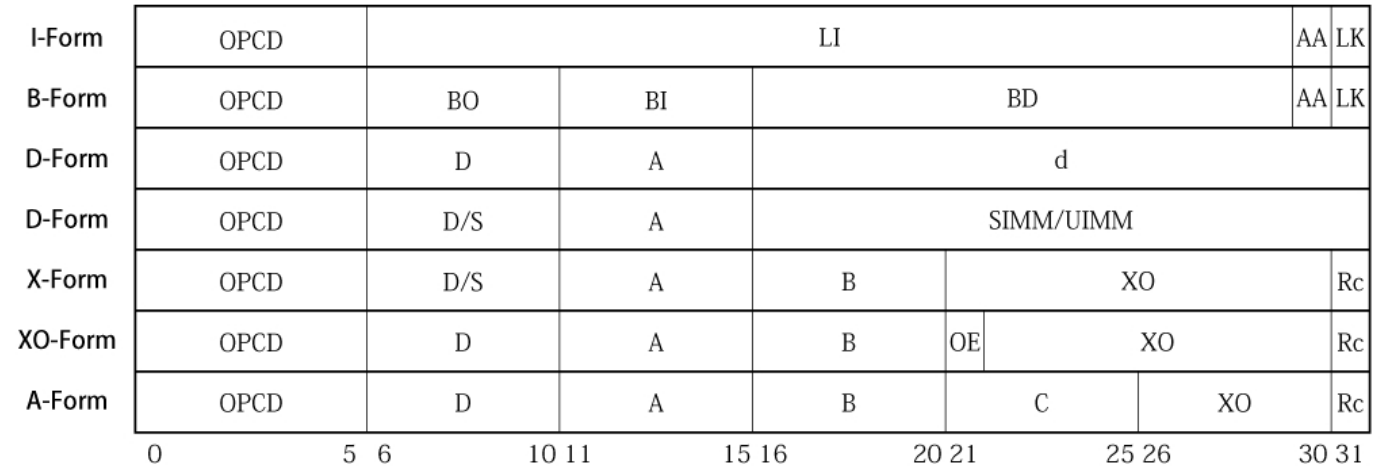
③ 固定長命令

　複雑なアドレッシングモードを使わないので、命令長を固定に（あるいはごく少数に限定）して、可変長命令で命令長の判別と解読にかかっていた時間を排してパイプラインの効率を上げる。

命令フォーマットから見たCISCとRISC



x86 アーキテクチャの命令フォーマット



OPCD：命令コード　XO：補助命令コード　LI, BD, d：アドレス・ディスプレースメント
SIMM/UIMM：イミディエート・データ　D, S, A, B, C：レジスタ指定

PowerPC アーキテクチャの命令フォーマット（抜粋）

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

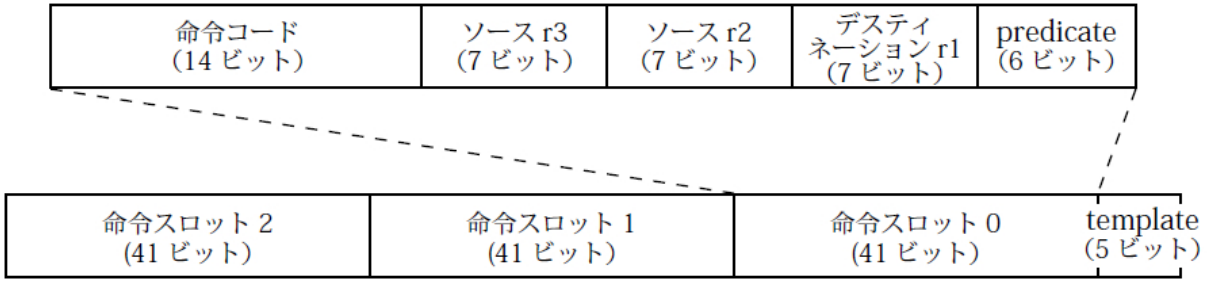
ペントミノ

スライディングパズル

2019年7月10日　あすた会資料　　吉宗一夫

IA-64プロセッサ

1990年代中頃、Intelはパソコン用32ビットプロセッサでは主流であったが、64ビットのサーバ市場はRISC陣営に占められていた。1994年になって、IntelはIA-64という64ビットRISCアーキテクチャのプロセッサをHP社と共同で開発することを発表した。IA-64はHP社のVLIW（Very Long Instruction Word）技術を採用したもので、これは並列に実行可能な命令の組合わせの判断をコンパイラに任せて、ハードウェアの制御回路を複雑にしないで命令実行の並列度を高めようとする技術である。



IA-64 の命令フォーマット（VLIW）

- IA-64のVLIWは3つの命令スロットをひとつのバンドル（128ビット）に格納し、プロセッサは一度に2個のバンドルを読出す。
- 「template」は各命令スロットを実行する演算器と並列に実行可能な命令スロットの区切りを指定する。
- 「predicate」は先行する命令の実行結果（たとえば比較命令の大小関係）の1つを指定し、その真偽で命令の実行／非実行を決める。

Intelプロセッサ搭載のACOS機

IA-64の仕様の詳細が明らかになり始めると、これを用いたエミュレーションでACOS小型機を開発する構想が浮上し、基本的な演算命令のエミュレーションルーチンを部分的に設計して性能の見積もり作業を開始した。

IA-64は汎用レジスタの数が多く、命令の並列実行を細かく制御できることなど、エミュレータの設計に好都合なアーキテクチャと考えられた。

この頃のNECメインフレーム機はNOAHと名づけたCMOS 1 チッププロセッサを搭載していたが、高速で集積度の高いプロセッサを設計するには莫大な手間が必要であった。汎用のプロセッサを利用してメインフレーム機を実現できるなら少ない労力で最先端の性能のマシンを開発できる。さらにNECはIntelプロセッサを搭載したWindowsサーバーも販売していたので、これとの「オープン連携機能」の実現にも好都合と考えられた。

ところが、当初は1999年に出荷とされていた最初のIA-64プロセッサ（開発コード：Merced）の出荷が2年以上遅れることがわかった。そこで、急遽 IA-32プロセッサによるエミュレーションの可能性を検討したところ、IA-32プロセッサでも何とか目標の性能を実現できそうな感触を得た。

IA-32プロセッサによるエミュレーションでは、エミュレーション実行ルーチンをIA-32のマイクロアーキテクチャ（パイプラインの構成やRISC風マイクロ命令への変換方法など）に適したものすることが重要であった。そこで、性能に影響が大きい基本命令の実行ルーチンは、Intel発行の「最適化マニュアル」等の情報を参考に、プロセッサのパイプラインの動作を想定しながらアセンブラで設計した。一方、ACOS独特のプロセス制御等の複雑な機能を実現する部分は品質や保守性を考慮してC言語で設計した。

IA-32プロセッサを用いたエミュレータは、結果的には当初の見積もり以上の性能を得られ、2000年にACOS i-PX7300（PentiumⅢ Zeon プロセッサ搭載）として出荷された。

i-PX7300の高速化技法

シャドーページテーブルの利用

論理アドレスから物理アドレスへの変換はOSが管理するページテーブル等の変換テーブルで指定され、ハードウェアはこの情報をTLBにキャッシュする。エミュレータプログラムは対象マシンの変換テーブルを解釈してホストマシンの形式の変換テーブル（シャドーページテーブル）を作成し、ホストマシンのハードウェア機能を用いて対象マシンの仮想記憶機能を実現する。

制約事項の設定

従来のマシン上で正常に動作していたプログラムは何らの変更なしに動作しなければならないが、不正な動作によって発生する例外については、例外の忠実な検出が性能に及ぼす影響と、例外を正しく検出できないことによる不利益を勘案して、許容可能な制約を設けた。

ソフトウェアパイプライン

エミュレータは対象マシンの機械語命令の「取出し→解読→演算→結果格納」をホストマシンの機械語命令で逐次実行するが、これを IA-32ハードウェアの並列処理に頼るだけでなく、エミュレータプログラムが対象マシン機械語の「演算→結果格納」中に次の命令の「取出し→解読」を同時に行って、これらの時間をオーバラップさせた。

分岐命令の排除

対象マシンの演算命令をエミュレーションするルーチン中では、通常の分岐予測が十分に機能しない場合が想定され、見かけのステップ数は増えても可能な限り条件分岐を使用しない方法を採った。

キャッシュヒット率の向上

エミュレータのメモリアクセスはエミュレータプログラム自体の命令読み出しが圧倒的に多い。そこで、対象マシンで使用頻度が高い命令をエミュレーションするルーチンの容量を、プロセッサ内の1次キャッシュに格納される程度までコンパクトにしてキャッシュミスを削減した。

IA-64プロセッサのその後

IA-64アーキテクチャを採用した最初のプロセッサは、予定より2年遅れの2001年に Itanium としてリリースされたが、ワークステーション用途では価格性能比で既存のRISCプロセッサに及ばず、x86互換モードの性能が低かったためにIA-32プロセッサを置き換えるものにもならなかった。

2002年には性能を改善した Itanium 2がリリースされ、NECはこれを搭載した i-PX9000 を自社OSだけでなくWindows、Linuxも利用できるメインフレームサーバとして2004年に販売を開始した。しかし、その後のIA-64プロセッサの性能向上が進まず、NECは2012年の i-PX9800から独自設計のプロセッサNOAH-6に切り替えた。

2019年1月、Intelは2021年にItaniumシリーズの製造を終了すると発表した。

x64アーキテクチャ

データのビット幅拡張による性能向上策として、複数のデータを64/128ビットにパックしたデータ形式を扱うSIMD命令が、マルチメディア用機能として90年代後期に相次いで登場した。

1997：IntelのMMX（64ビット長データ）
1998：AMD 3DNow!（浮動小数点SIMD)
1999：Intel SSE（128ビット長データ）

アドレス空間の拡張としては、IA-32と互換性を保ちつつ64ビットに拡張した仕様をAMDが2000年に x86-64（後にAMD64）として発表、2003年からAthlon64として出荷した。2004年にはIntelもAMD64互換のプロセッサを製品化し、このアーキテクチャはx64と呼ばれるようになった。

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

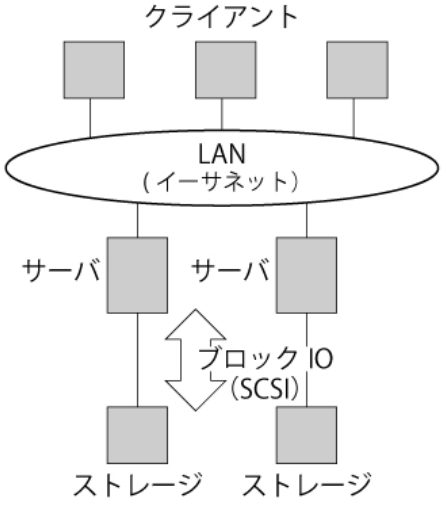
スライディングパズル

2019年7月10日　あすた会資料　　吉宗一夫

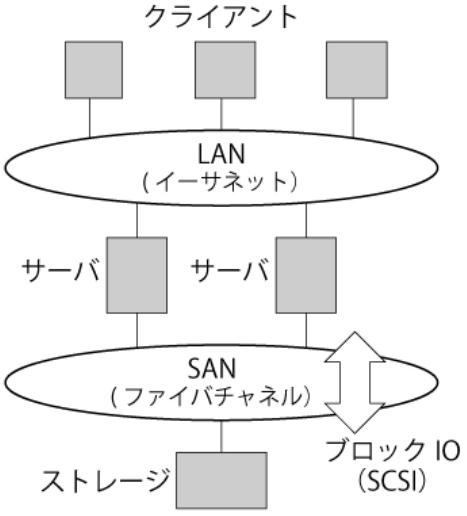
ネットワークストレッジ

1980年代末から1990年代にかけて多くの企業でオープン系クライアントサーバシステムの導入が進んだ。しかし、企業のデータ量が増大すると、ディスク装置をサーバに 1 対 1 に接続する方法（DAS：Direct Attached Storage）では、サーバOSごとに固有の管理手法が必要で、ストレージリソースの柔軟な配分も難しいという問題があり、ネットワークに接続されたストレージ装置をサーバ間で共用して集中的に管理する形態が生まれた。

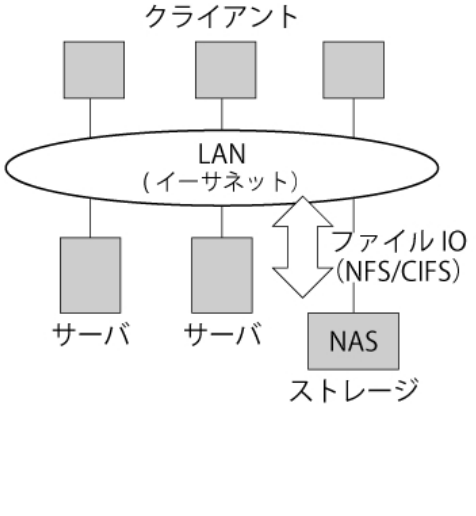
このような接続形態のひとつであるSAN（Storage Area Network）は、ファイバーチャネルを用いた専用のネットワークで複数のサーバとストレージ装置を接続し、サーバはブロックレベルIOでストレージにアクセスする。もう1つの接続形態であるNAS（Network Attached Storage）は、LANに接続されたストレージ装置がファイルレベルでのアクセスを提供するもので、ストレージ機能に特化したファイルサーバと言える。



DAS システム



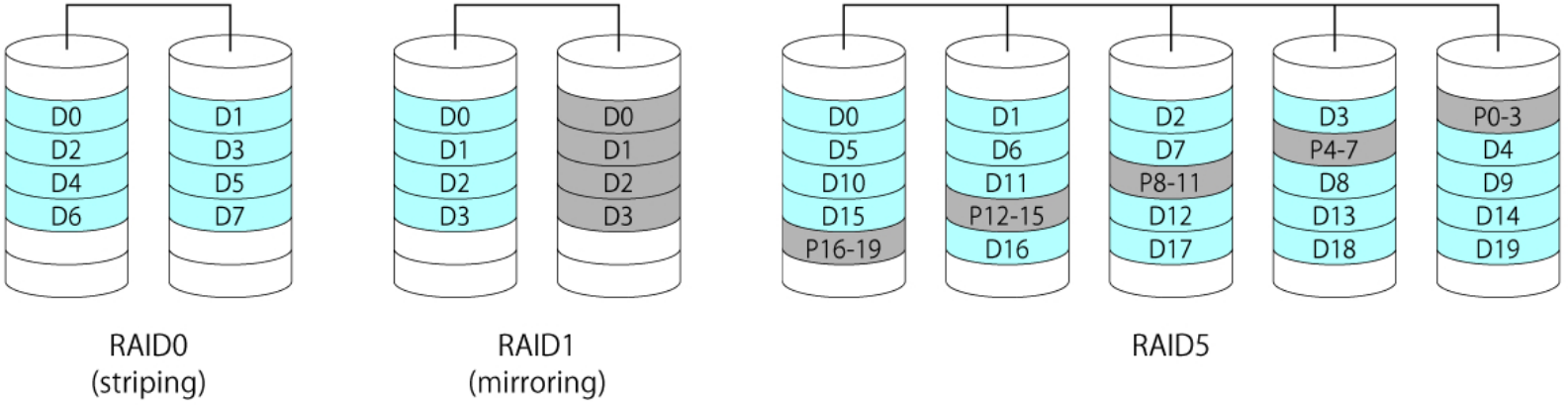
SAN システム



NAS システム

RAID技術

情報システムの中核となるネットワークストレージ装置では、ディスク故障によるデータ損失を防ぎ、あわせて処理性能を向上するために、RAID技術を採用するのが一般的である。これは「Redundant Arrays of Inexpensive Disks」 の頭文字で、カリフォルニア大バークレイのパターソン教授らの1887年の論文で、安価な市販のハードディスクを組み合わせて大容量で信頼性の高いストレージを構成する技術として提案された。この論文では、RAIDの方式をレベル 1 からレベル5に分類したが、現在は下記のようなものが実用にされている。



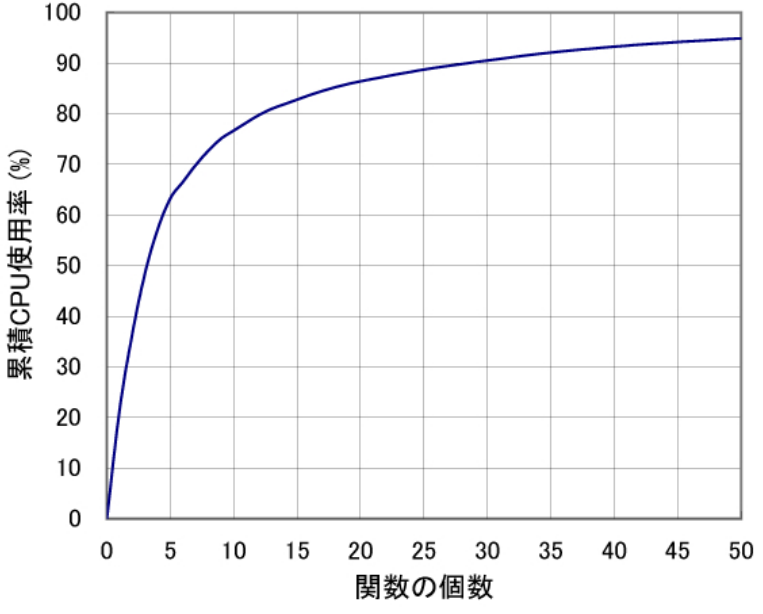
RAIDのレベル

- RAID0は速度向上や容量拡大が目的で冗長度はないが、RAIDの1種とされ、RAID 1 と組み合わせて利用されることも多い。
- RAID5の構成に、もう一つのドライブを追加して２種類のパリティを記録し、任意の２つのドライブの故障に耐えられるようにしたRAID6もある。

RAIDドライバの性能分析

NECでは従来からのSAN装置に加えてNAS装置を製品化することになった。これはLinuxのファイルシステムを利用し、従来のSAN装置のRAID制御ファームウェアをLinux上でRAIDドライバとして動作させるものであった。

しかし、試作段階の性能測定では目標に届かず、プロファイラを用いてRAIDドライバの動作を分析し、CPU使用量が大きくて改善を要する関数を抽出した。

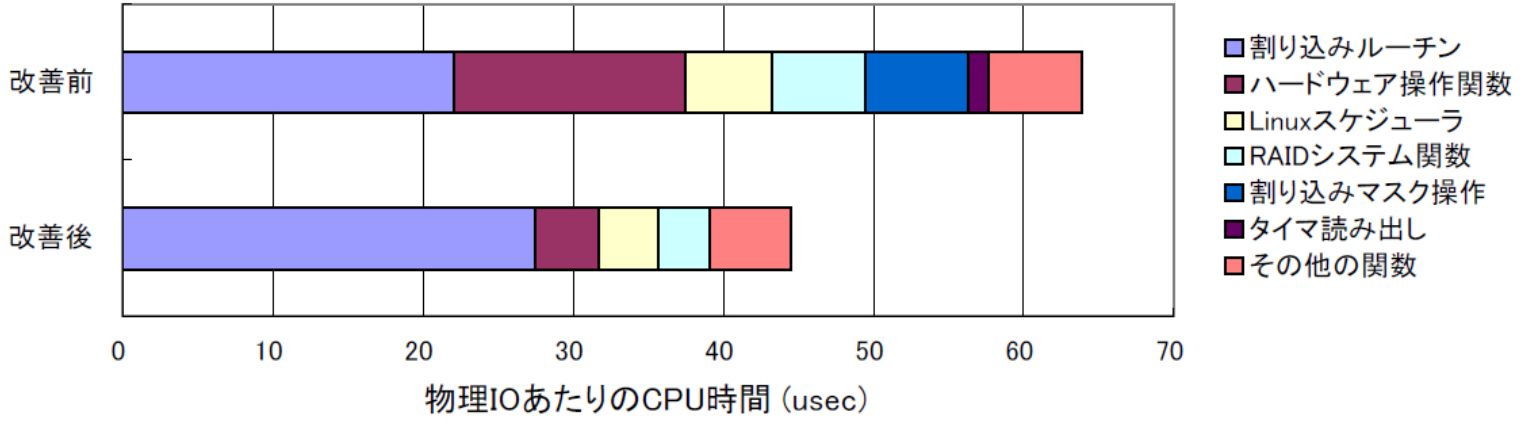


関数名	CPU使用率	機能
Ass_Forward	21.4%	ARY回路操作
_global_sti	15.0%	割込みマスク操作
schedule	12.0%	Linuxスケジューラ
Ppd_GetIocbResidualCount	8.7%	ISP操作
wai_flg	6.2%	RAIDシステム関数
raid_dma_xfer	3.3%	DMA回路操作
do_gettimeofday	3.3%	実時間タイマ読出し
Ppd_ScFunc	2.8%	ISP操作
wai_sem	2.4%	RAIDシステム関数
_snd_msg	1.6%	RAIDシステム関数

プロファイラは一定時間（たとえば10ms）ごとにタイマの割込みを起こし、割込み発生時に動作していたプログラムアドレスの記録から、プログラム各部分のCPU使用時間を統計的に測定する。この測定からは、少数の関数がCPU時間の大部分を消費していることがわかる。（CPU使用率の高い関数上位10個で77%、30個で90%、50個で95%のCPU時間を消費）

改善策と効果

- RAID機能をサポートするハードウェア（たとえばパリティ計算回路、DMA回路）へのアクセスに大きな時間を要していることがわかり、このようなアクセスを見直して回数を削減した。
- スレッド切り替えに関連する関数のCPU使用時間が事前の見積もりより大きいのはキャッシュミスの影響と予想し、一部のキューをFIFOからLIFOに変更した。また、スレッドの動作の一部を割り込みルーチンに移動してスレッド切り替え回数を削減した。
- 割り込み禁止モードを操作する関数のCPU使用率が高いことに注目して割込み禁止操作を見直し、不要なものを削除したり、割込み禁止の範囲をシステムからプロセッサに変更した。
- ディスク統計情報採取のために読み出していたハードウェアタイマをメモリ上の精度の低いタイマで代替した。（多数回の測定の平均値を使用するなら高い精度は不要）



第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

スライディングパズル

2023年6月2日　あすた会資料　　吉宗一夫

Java ＜Write Once, Run Anywhere＞

Javaは、1995年にサン・マイクロシステムズによって公開された。Java言語の構文はC++に類似したものであるが、堅牢性を損ねると考えられるポインタ、演算子オーバーロード、goto文などは破棄する一方、プログラマがメモリを管理する負担を軽減するためにガベージコレクション機能（注）を備えている。

中間言語（バイトコード）にコンパイルしてJava仮想マシンで実行する方式を採ることにより、様々なハードウエアやOS上でリコンパイルなしに動作し、”Write Once, Run Anywhere”がサン・マイクロシステムズ社の売り文句であった。

仮想マシンの実行を高速化するために、ジャストインタイムコンパイル方式（JITコンパイル方式）を採用し、実行中のプログラムを分析して実行頻度の高い部分は自動的にネイティブコードに変換される。

Java が公開された当初には、アプレットという形態でネットワーク経由で読み込んでWebブラウザ上で実行することで、Webブラウザのインタラクティブ性を高める技術として注目された。

HTML5やJavaScriptの登場によりJavaアプレットの有用性は薄れ、サン・マイクロシステムズ社からJavaを引き継いだOracle社は近い将来にJavaアプレットを廃止する予定であるが、Java言語そのものはWebサーバ上のプログラム開発や携帯機器／家電製品の組み込みソフトウェア開発などに広く利用されている。

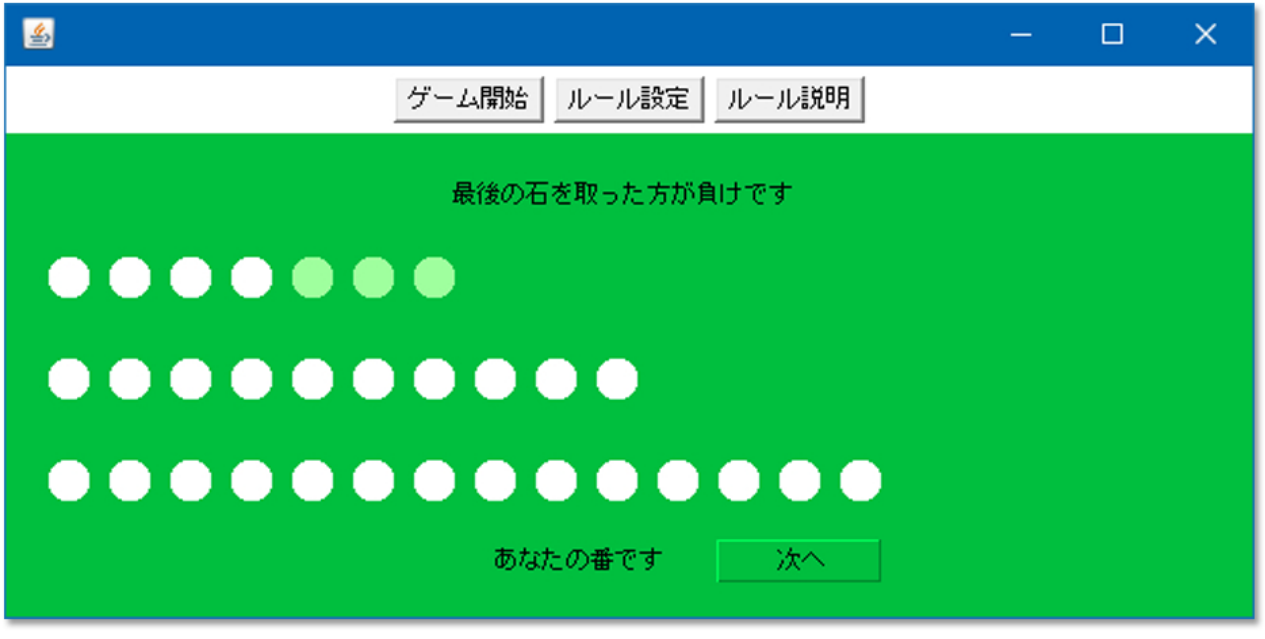
（注）プログラムが動的に確保したメモリ領域のうち、不要になった領域を自動的に解放する機能。メモリーリークなどのメモリ管理に関連するバグを回避することができる。

三山くずし（nim）

複数の山に積まれた石やコインなどを二人で取り合うゲームで、ルーツは古代中国にあるとされる。西欧ではニム（nim） という名前で知られる。

ルール

- 複数の山に積まれた石を交互に取り、どちらが最後の石を取るかで勝敗を決める。
- 一度には一つの山からしか取れないが、石の個数に制限はない。
- 最後の石を取った方を負けとするルールが一般的であるが、最後の石を取った方を勝ちとするルールもある。
- さらにゲーム開始後に一度だけこのルールを変更できるという遊び方もある。（ルールを変更できるのは、先にルール変更を宣言したどちらかのプレイヤーひとりであり、自分の順番のときに石を取らずにルール変更を宣言する。）


ゲーム開始
ルール設定
ルール説明
最後の石を取った方が負けです
あなたの番です
次へ

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

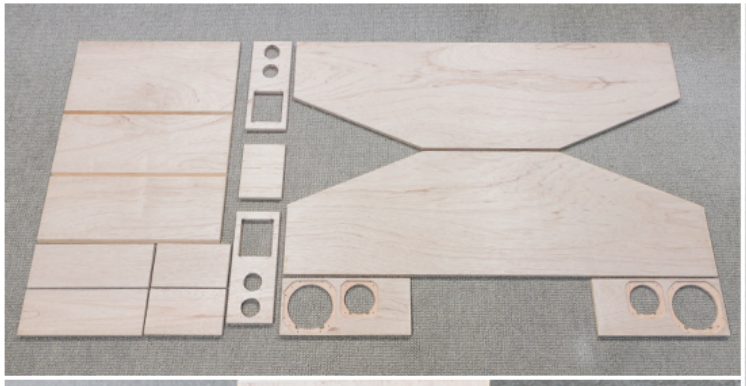
Scratch & Python

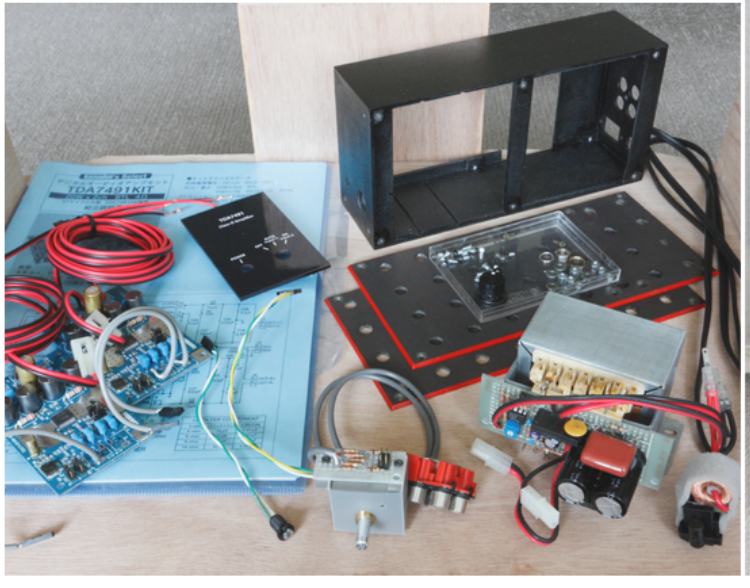
ペントミノ

スライディングパズル

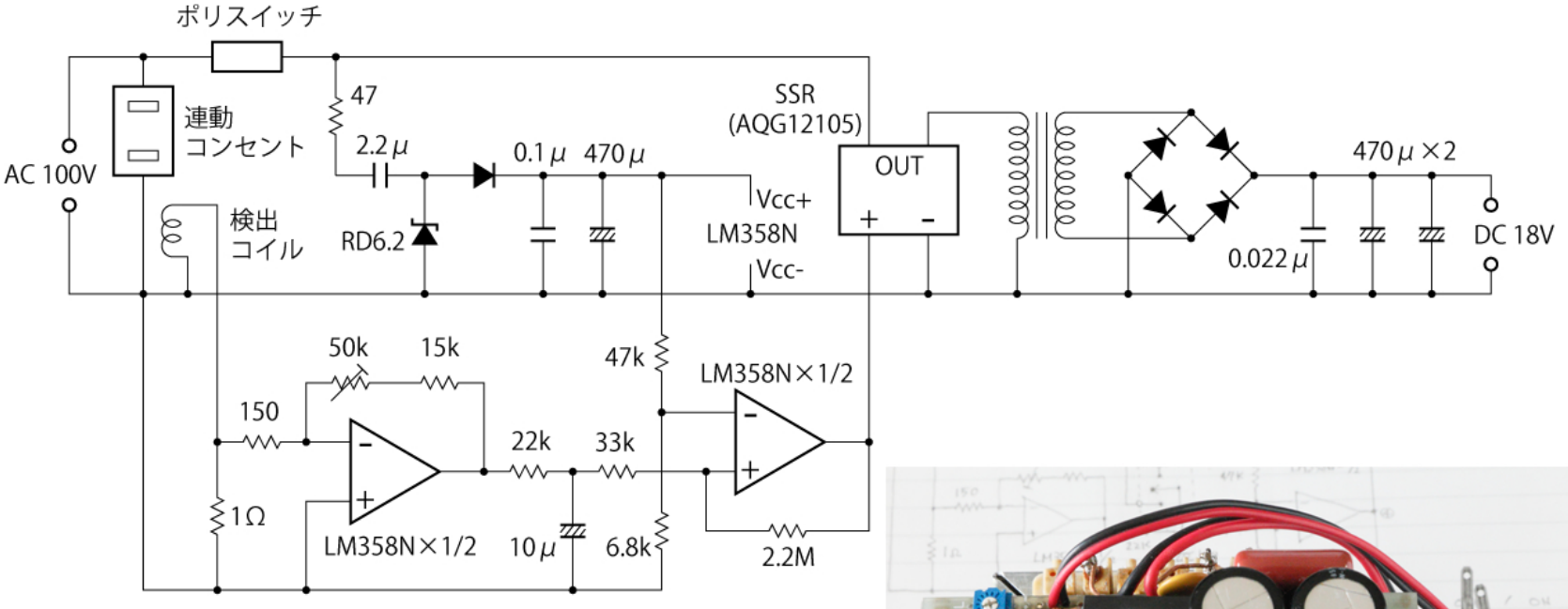
2023年6月2日　あすた会資料　　吉宗一夫

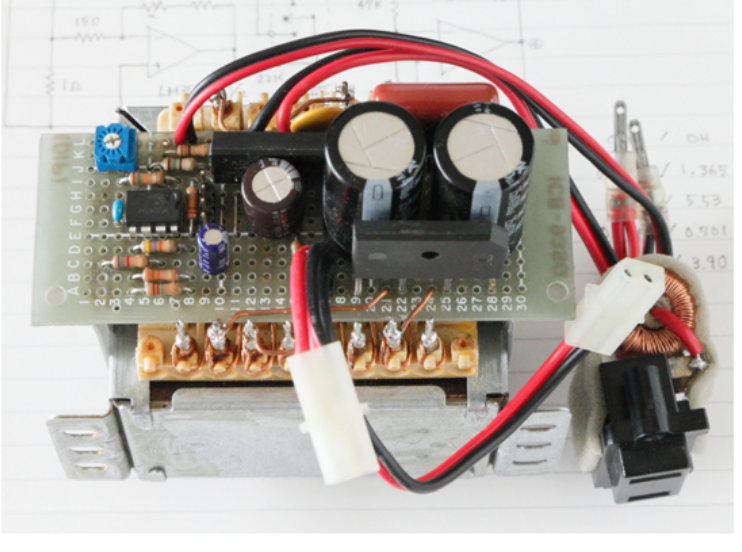
スピーカ内蔵テレビ台



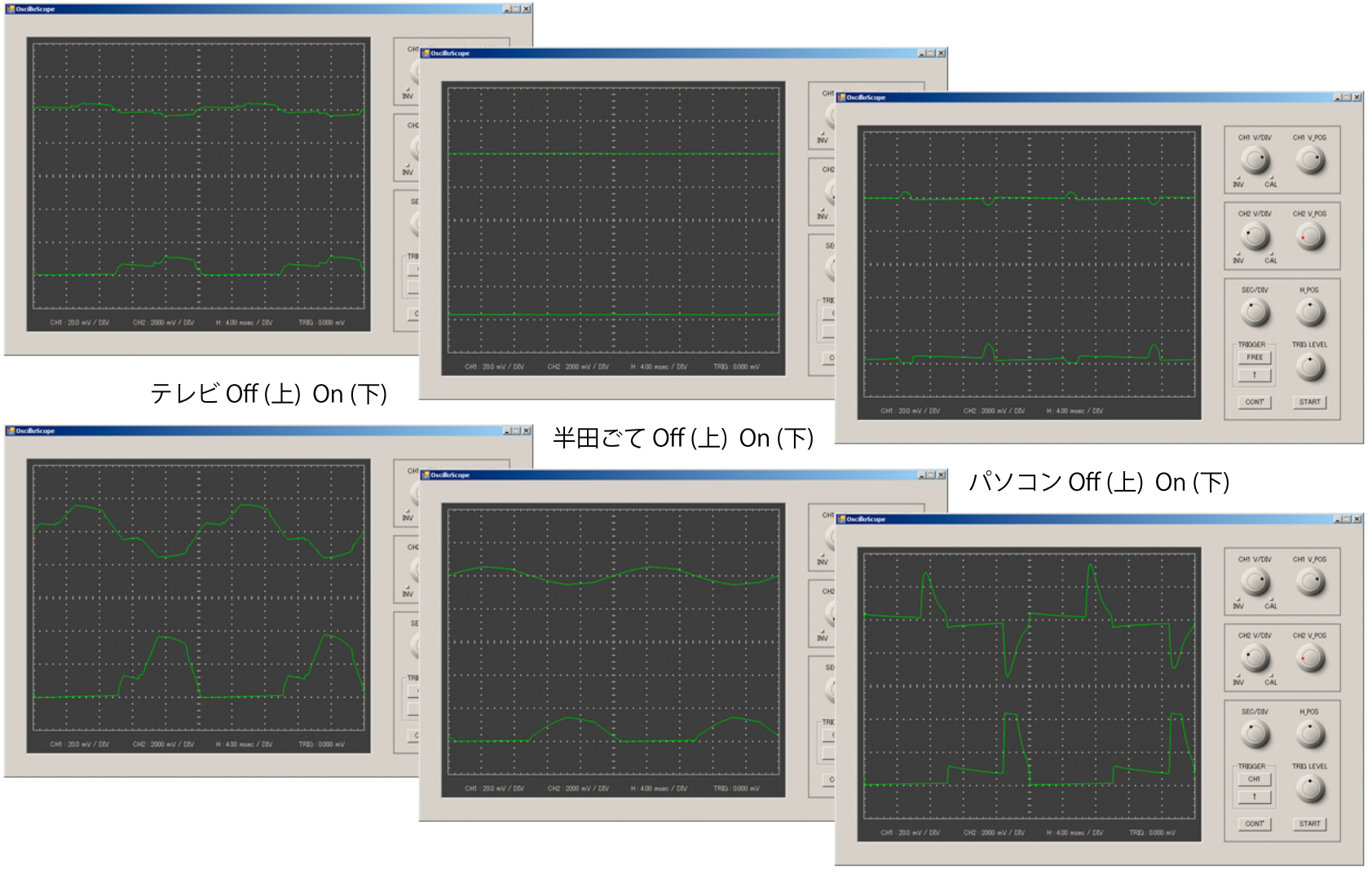


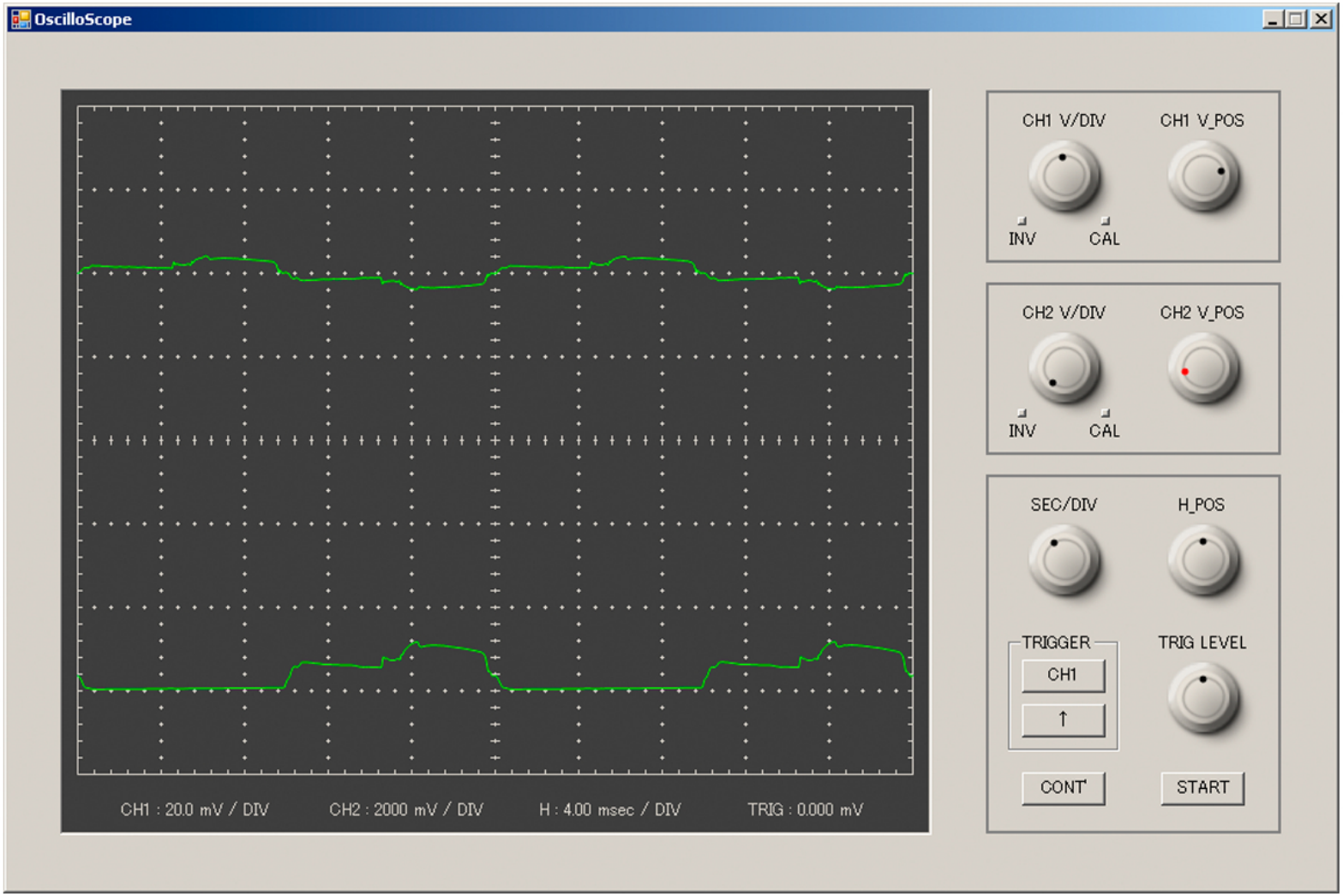
連動電源


ポリスイッチ
AC 100V
連動
コンセント
検出
コイル
47
2.2μ
RD6.2
0.1μ
470μ
SSR
(AQG12105)
OUT
+
−
Vcc+
LM358N
Vcc-
470μ×2
0.022μ
DC 18V
50k
15k
150
1Ω
LM358N×1/2
22k
33k
10μ
47k
LM358N×1/2
6.8k
2.2M

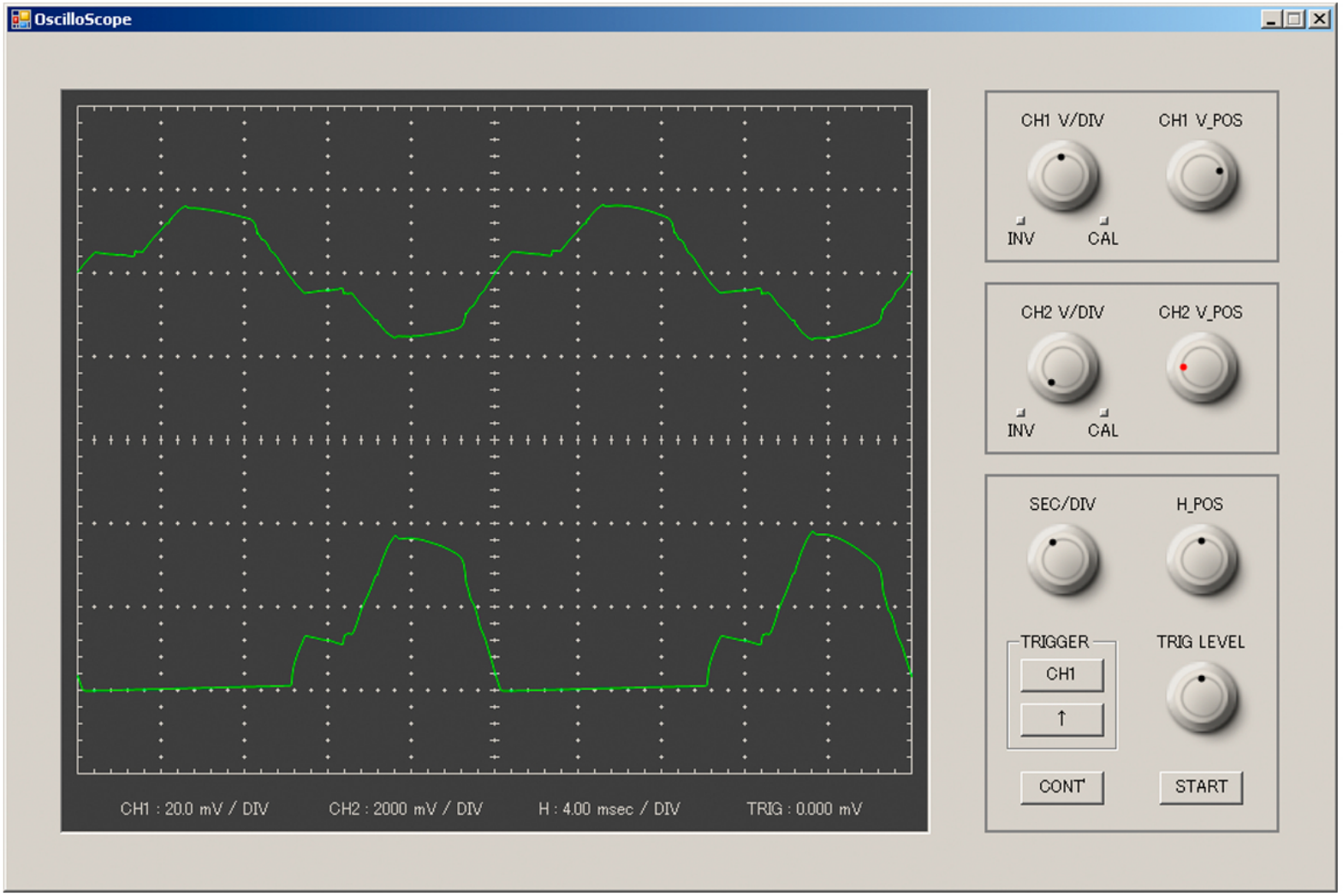


電源波形（パソコンでオシロ）

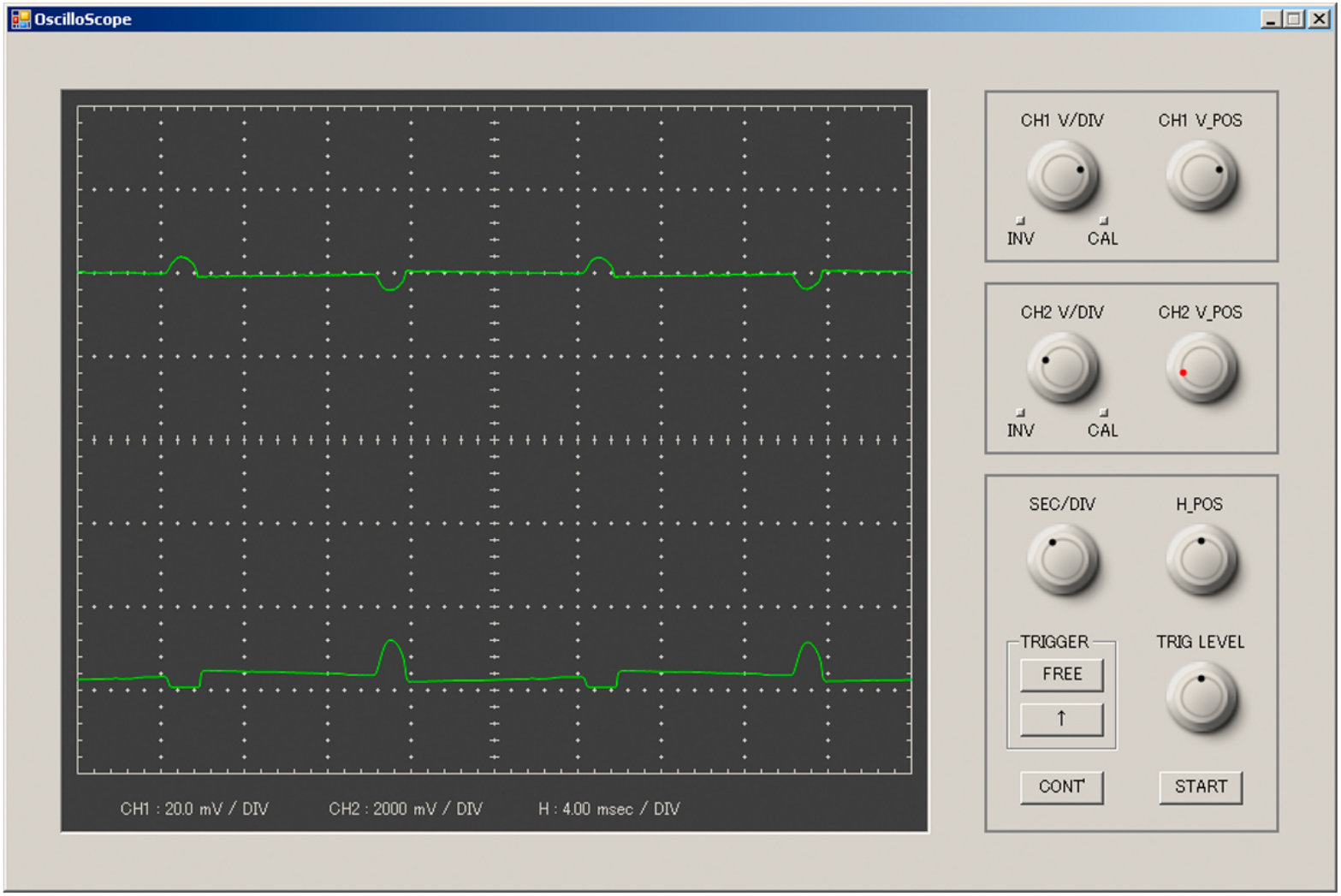



OscilloScope
CH1 V/DIV
CH1 V_POS
INV
CAL
CH2 V/DIV
CH2 V_POS
INV
CAL
SEC/DIV
H_POS
TRIGGER
CH1
↑
TRIG LEVEL
CONT
START
CH1 : 20.0 mV / DIV
CH2 : 2000 mV / DIV
H : 4.00 msec / DIV
TRIG : 0.000 mV

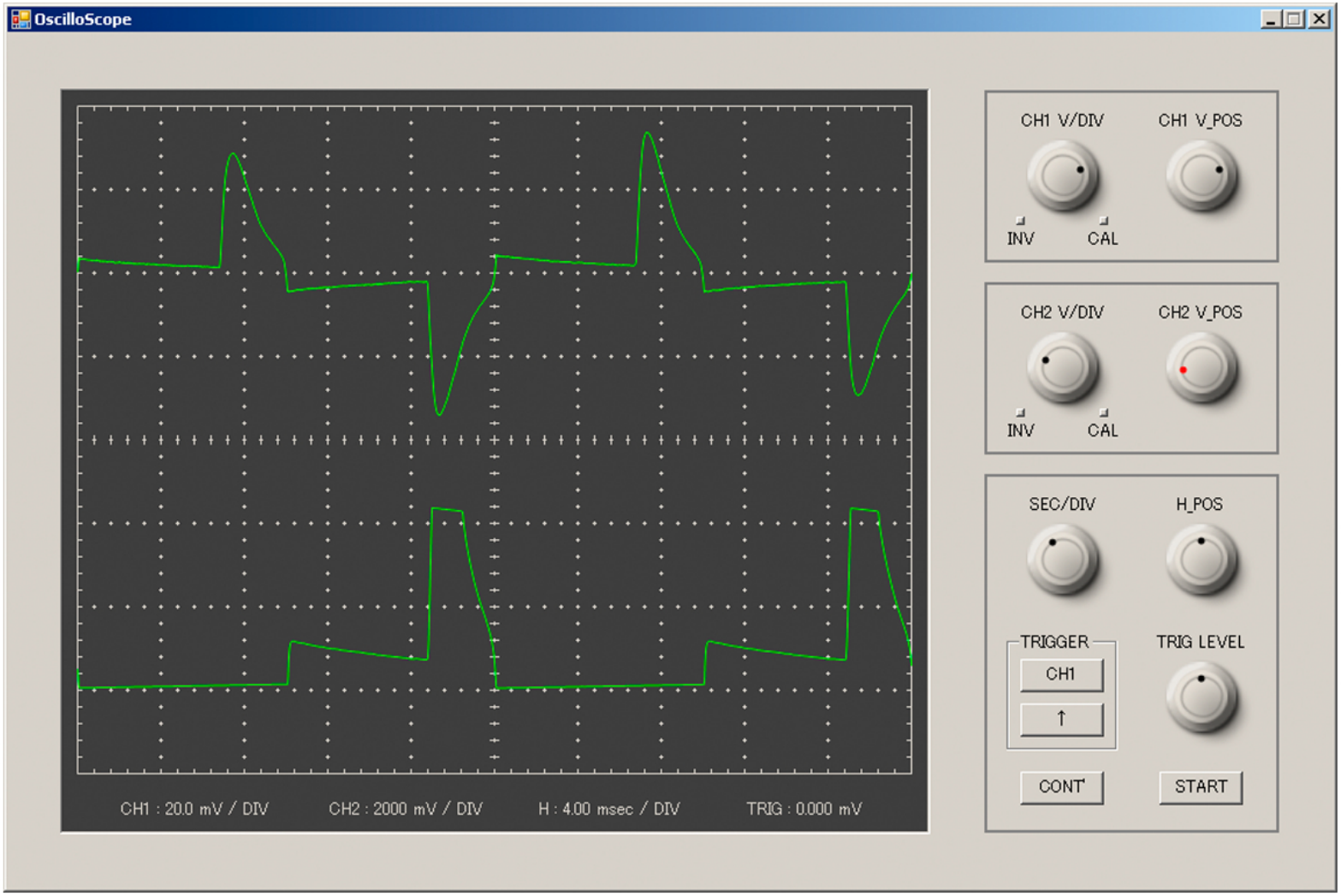
テレビ Off


OscilloScope
CH1 V/DIV
CH1 V_POS
INV
CAL
CH2 V/DIV
CH2 V_POS
INV
CAL
SEC/DIV
H_POS
TRIGGER
CH1
↑
TRIG LEVEL
CONT
START
CH1 : 20.0 mV / DIV
CH2 : 2000 mV / DIV
H : 4.00 msec / DIV
TRIG : 0.000 mV

テレビ On


OscilloScope
CH1 V/DIV
CH1 V_POS
INV
CAL
CH2 V/DIV
CH2 V_POS
INV
CAL
SEC/DIV
H_POS
TRIGGER
FREE
↑
TRIG LEVEL
CONT
START
CH1 : 20.0 mV / DIV
CH2 : 2000 mV / DIV
H : 4.00 msec / DIV

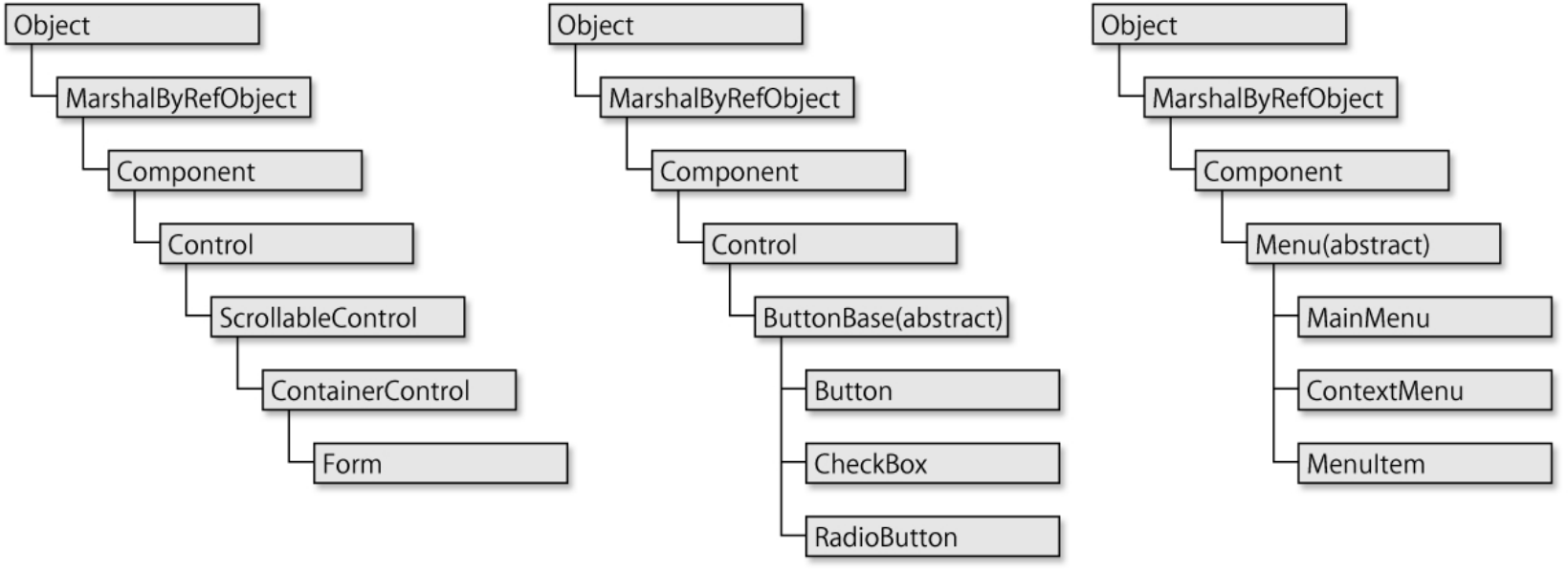
パソコン Off


OscilloScope
CH1 V/DIV
CH1 V_POS
INV
CAL
CH2 V/DIV
CH2 V_POS
INV
CAL
SEC/DIV
H_POS
TRIGGER
CH1
↑
TRIG LEVEL
CONT
START
CH1 : 20.0 mV / DIV
CH2 : 2000 mV / DIV
H : 4.00 msec / DIV
TRIG : 0.000 mV

パソコン On

C#と .NET Framework

C#はマイクロソフトが開発したC系言語で、構文はC++やJavaに類似している。マイクロソフトが策定したCLI（Common Language Interface：共通言語基盤）に基づくCIL（Common Intermediate Language：共通中間言語）にコンパイルされて、Microsoft .NET Frameworkとして提供されるCLR（共通言語ランタイム）上で実行される。CLRはマイクロソフトが提供するVisual Basic.NETなどの他の言語でも利用され、Java仮想マシンと同じように、プロセッサに依存しない実行形態で、ガベージコレクションやJITコンパイルによる実行の高速化などを実現している。



GUI部品の派生階層

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

スライディングパズル

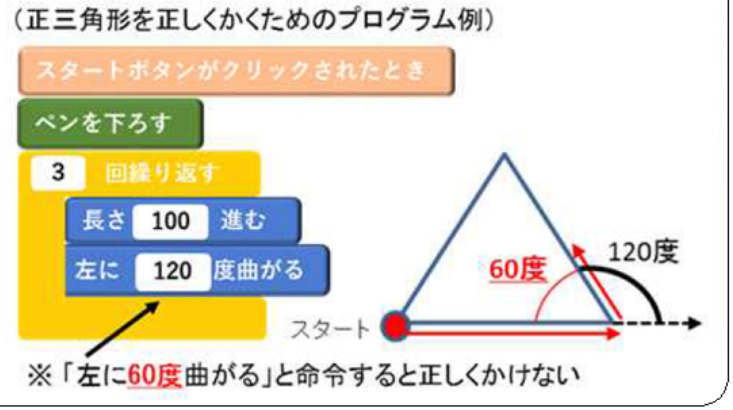
2023年6月2日　あすた会資料　　吉宗一夫

プログラミング教育

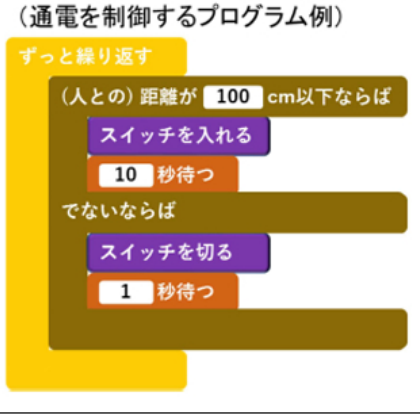
「コンピュータを受け身ではなく積極的に活用する力」や「プログラミング的思考（論理的思考力）」を養うため、2020年度から小学校でプログラミング教育が必修化されることになった。新たな教科を設けてプログラミング言語の使い方を教えるのはなく、すでにある教科の中で、“プログラミング的思考”（物事には手順があり、手順を踏むと物事をうまく解決できるといった、論理的に考えていく力）を養うことが目的とされている。

文部科学省が作成した「小学校プログラミング教育の手引き」に示された指導例

プログラミングを通して、正多角形の意味を基に正多角形をかく（算数第５学年）

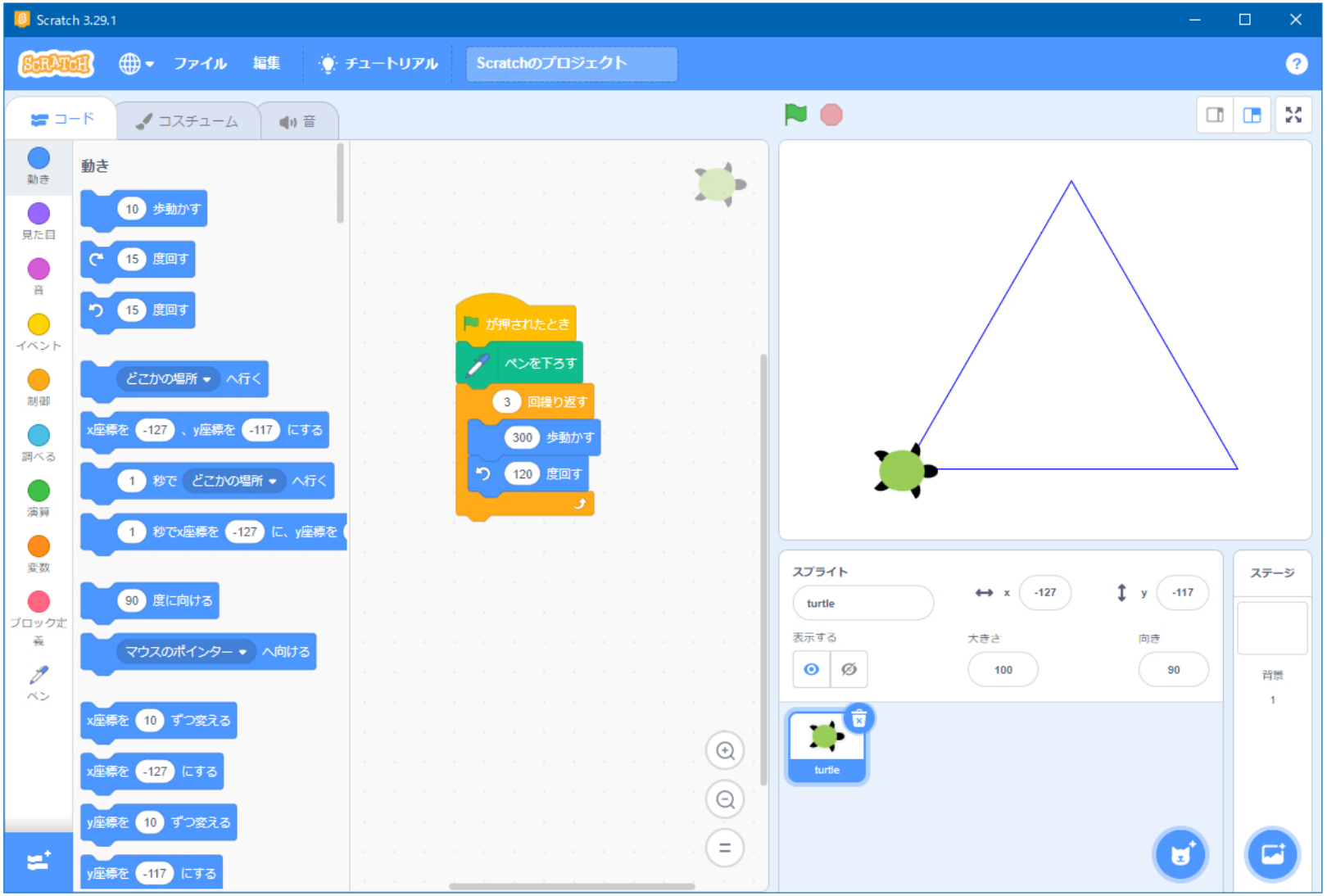


身の回りには電気の性質や働きを利用した道具があること等をプログラミングを通して学習する（理科第６学年）



自動炊飯器に組み込まれているプログラムを考える活動を通して、炊飯について学習する場面（家庭第６学年）

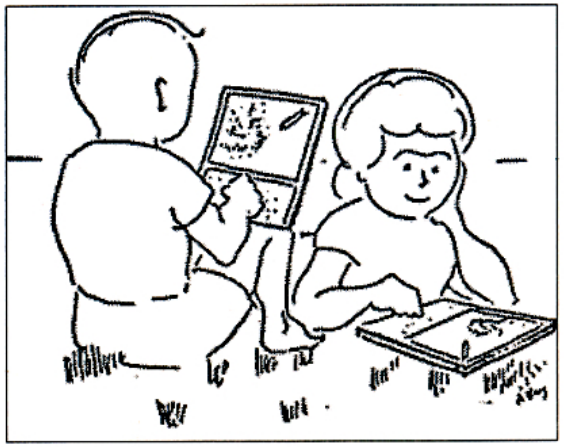
水加減や浸水時間、加熱の仕方、蒸らしなどの炊飯に関する一連の手順について、コンピュータ上で並べ替えと条件設定（プログラミング）を行います。その際、水加減や加熱の仕方（火加減）等の条件を変えて、２回程度行い、ご飯が柔らかくなったり硬くなったりする原因について考えます。


Scratch 3.29.1
ファイル
編集
チュートリアル
Scratchのプロジェクト
コード
コスチューム
音
動き
見た目
音
イベント
制御
調べる
演算
変数
ブロック定義
ペン
動き
10 歩動かす
15 度回す
15 度回す
どこかの場所 へ行く
x座標を -127 、y座標を -117 にする
1 秒で どこかの場所 へ行く
1 秒でx座標を -127 に、y座標を
90 度に向ける
マウスのポインター へ向ける
x座標を 10 ずつ変える
x座標を -127 にする
y座標を 10 ずつ変える
y座標を -117 にする
が押されたとき
ペンを下ろす
3 回繰り返す
300 歩動かす
120 度回す
スプライト
turtle
x -127
y -117
表示する
大きさ
100
向き
90
turtle
ステージ
背景
1

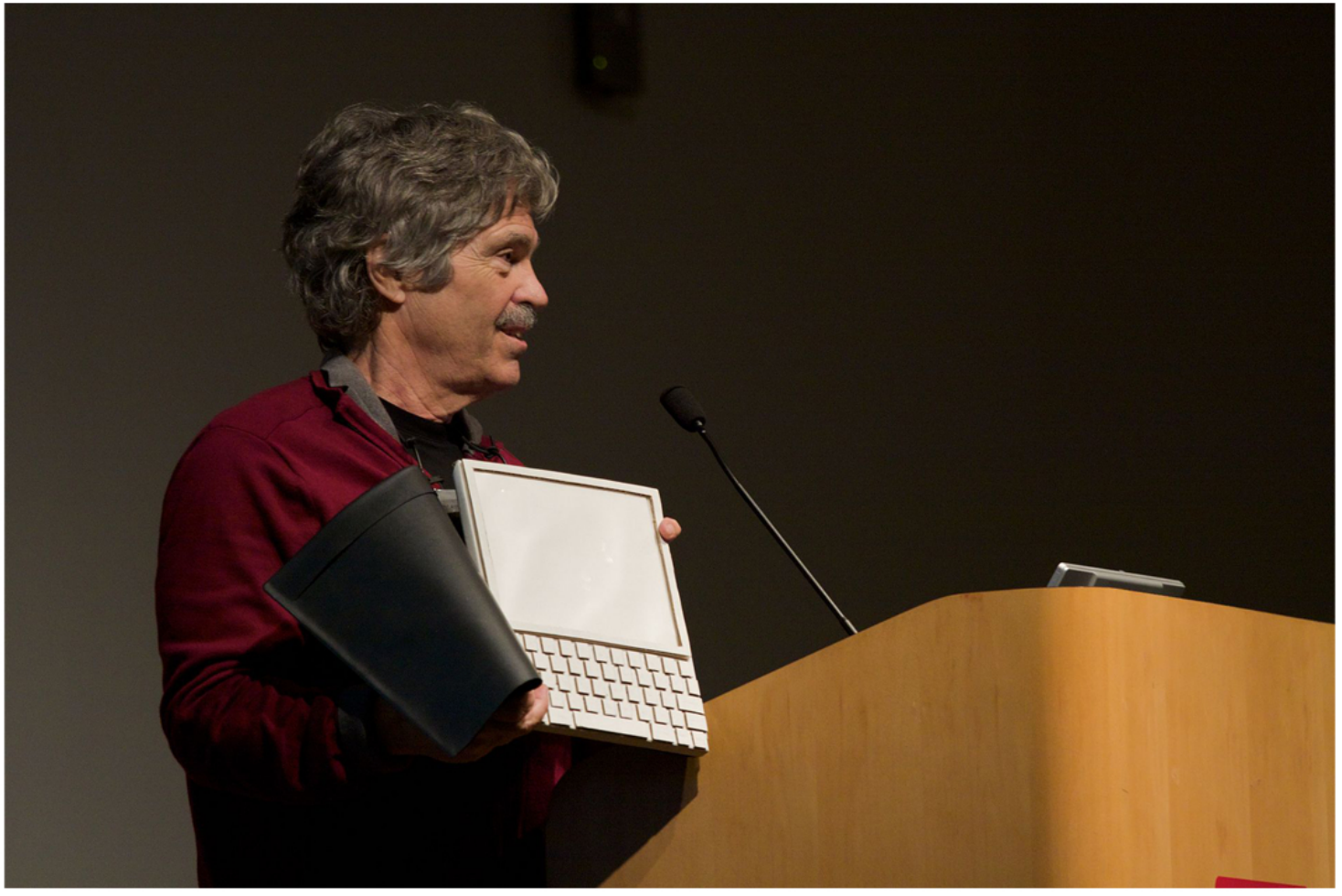
教育用プログラミング言語

1967年にシーモア・パパートらは児童の思考能力の訓練を目的に教育用言語 LOGO を開発した。LOGO は下記の smalltalk にも大きな影響を与え、また、LOGOの特徴的な機能であるタートルグラフィックスは Scratch にも取り入れられている。

Smalltalk は1970年代にゼロックスのパロアルト研究所でパーソナルコンピュータの父といわれるアラン・ケイが中心となって開発された。さらにこれから発展した Squeak をベースに1996年には Etoys という子供向けプログラミング学習環境が開発され、その後、MIT メディアラボのミッチェル・レズニックは Etoys 開発チームのジョン・マロニーを招いて Scratch を開発した（2006）。



出典：『A Personal Computer for Children of All Ages』（Alan Kay, 1972）より



ダイナブックの模型を持つアラン・ケイ

Xerox Alto

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

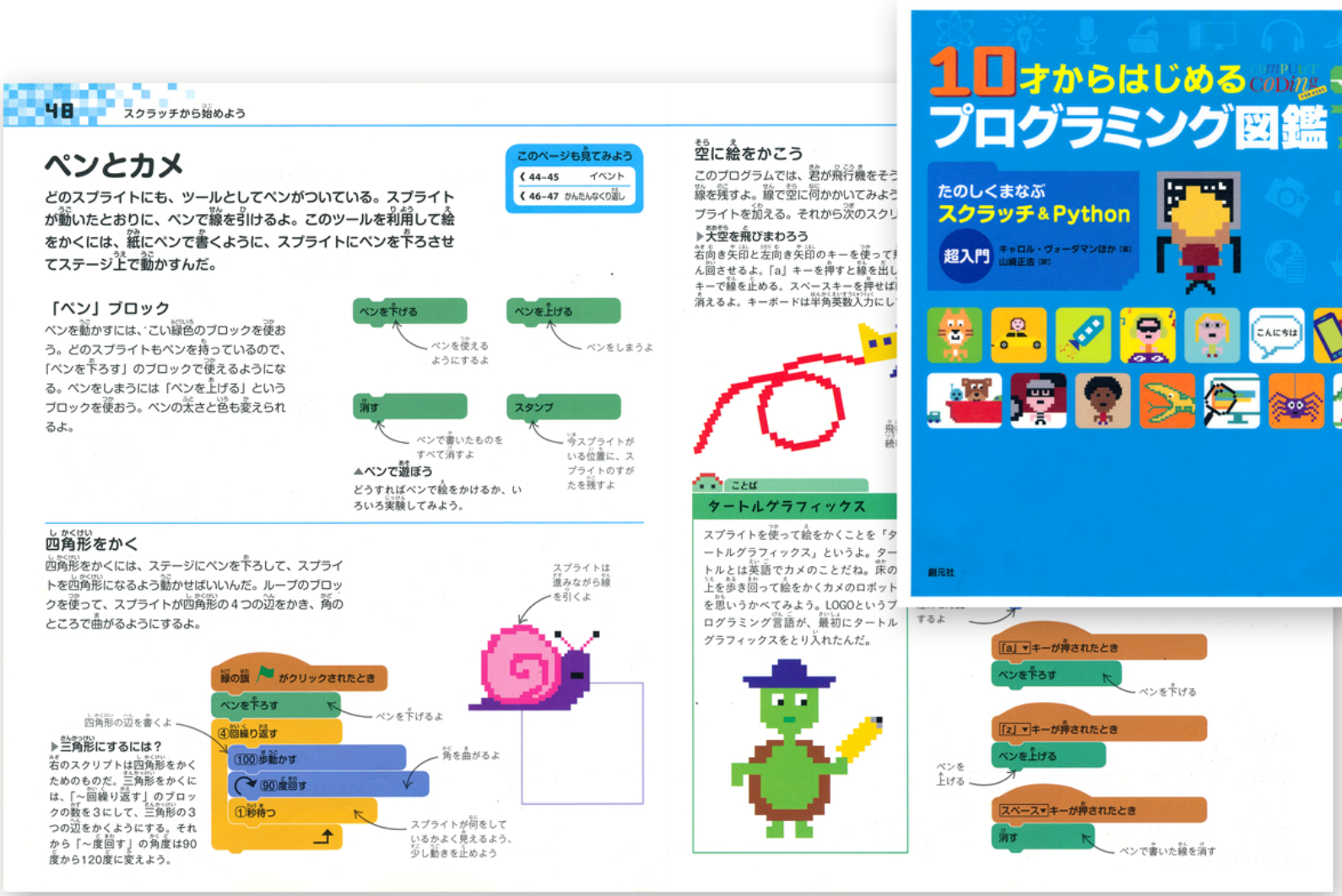
Scratch & Python

ペントミノ

スライディングパズル

2023年6月2日　あすた会資料　　吉宗一夫

「10才からはじめるプログラミング図鑑」

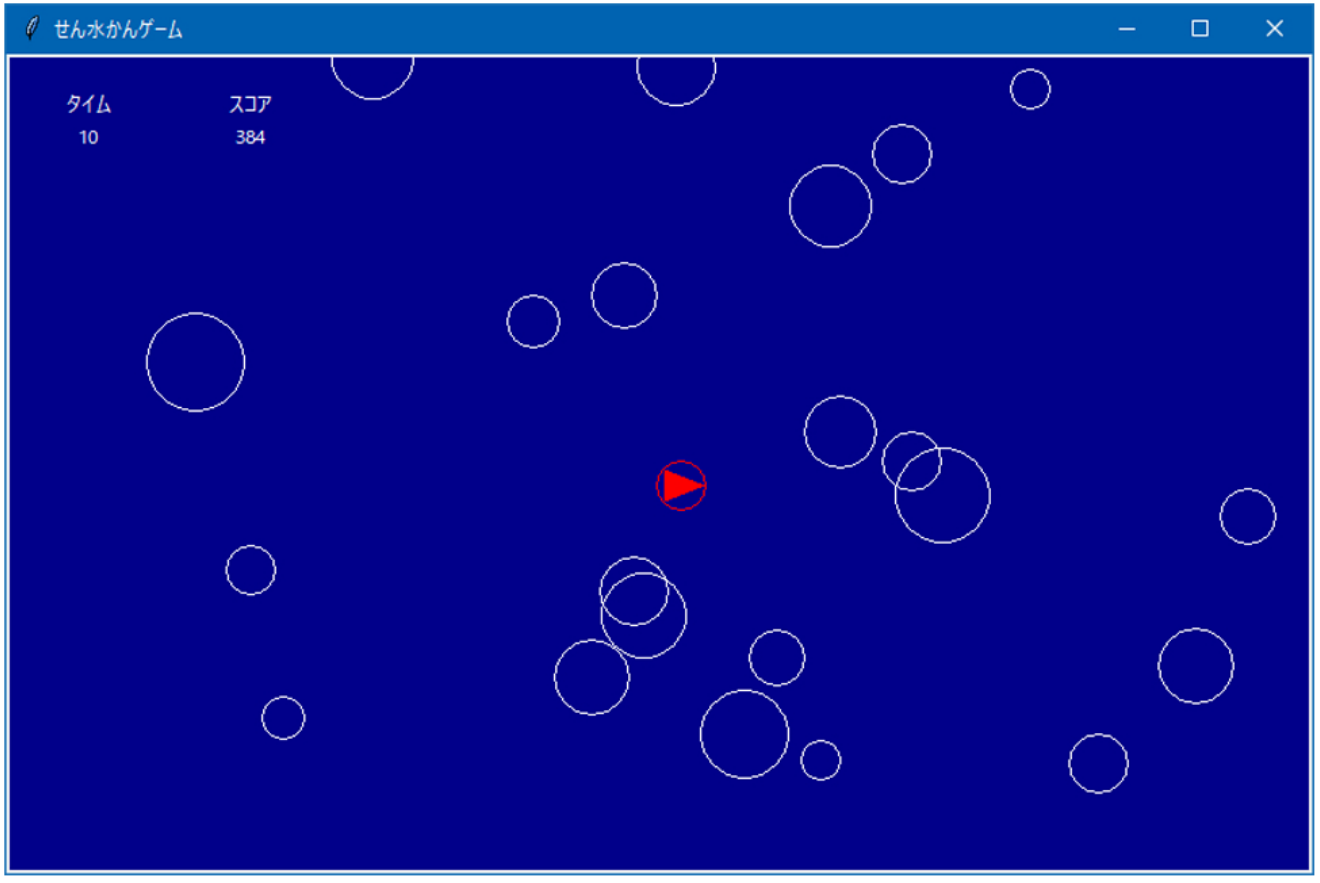


目次

「10才からはじめるプログラミング図鑑」
たのしくまなぶスクラッチ＆Python

4 コンピューターのしくみ

5 現実の世界でのプログラミング


せん水かんゲーム
タイム
10
スコア
384

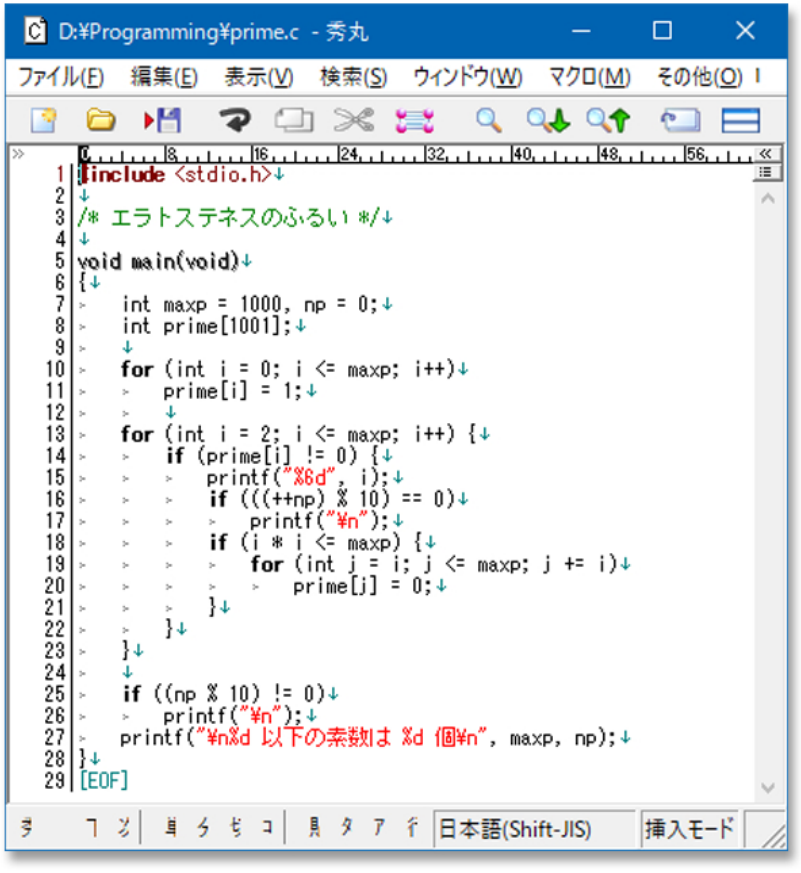
Python　< Battery Included >

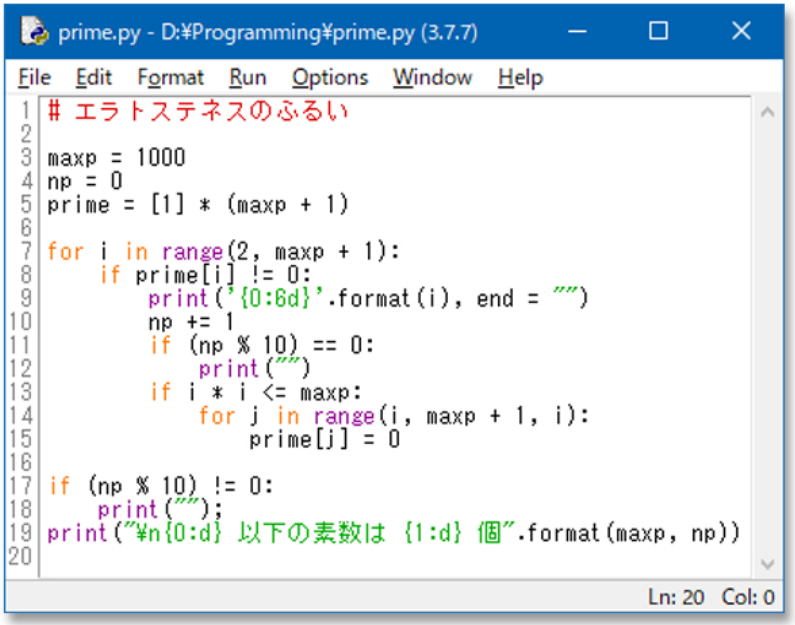
Pythonはオランダ人のグイド・ヴァンロッサムが開発した言語で、文法を極力単純化してコードの可読性を高め、読みやすく、また書きやすくしてプログラマの作業性とコードの信頼性を高めることを重視している。

名前の由来は英テレビ局 BBC が製作したコメディ番組『空飛ぶモンティ・パイソン』であるが、Pythonが意味するニシキヘビがPython言語のマスコットやアイコンとして使われている。

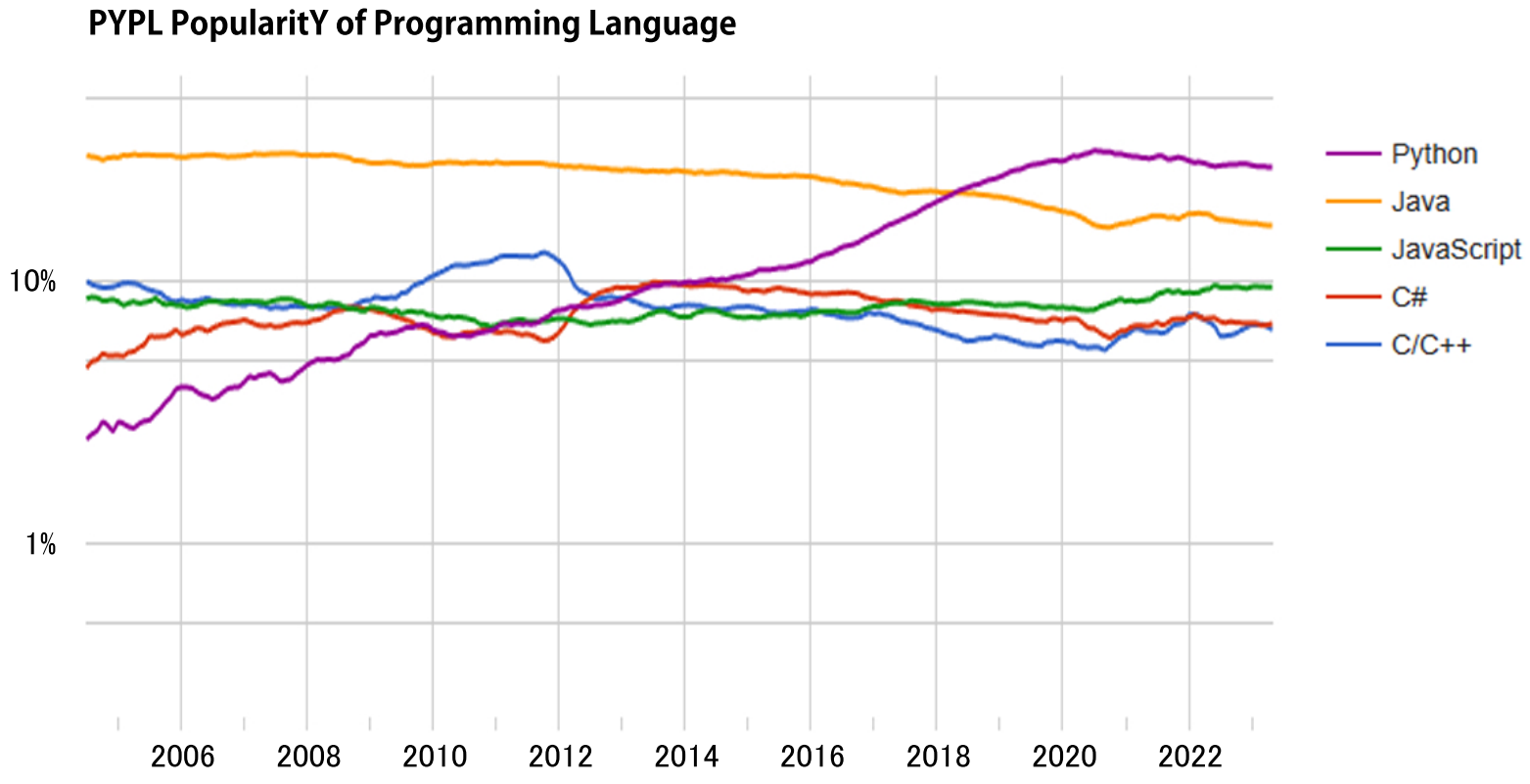
- 基本構文はC言語類似であるが、インデントによりブロックを表す記法が特徴的で可読性に優れる。
- 動的型付け言語であり、変数の型は代入された値によって決まる。
- 参照カウントベースの自動メモリ管理（ガベージコレクション）を持つ。
- インタプリタ上で実行することを前提に設計され、コンパイル不要で対話型シェルで動作確認可能であり、多くのハードウェアとOS (プラットフォーム) に対応している。
- 標準ライブラリやサードパーティ製のライブラリ、関数など、さまざまな領域に特化した豊富なツール群が用意され、「Battery Included」と称されている。

C と Python のソースコード





Pythonは人気急上昇



PYPLはGoogle検索エンジンにおいてプログラミング言語のチュートリアルが
検索された回数を人気度と位置づけてランキングしたもの

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

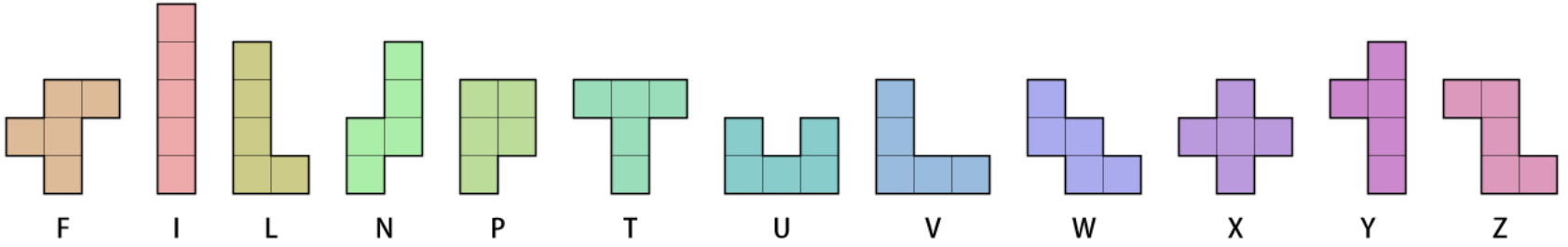
スライディングパズル

2023年6月2日　あすた会資料　　吉宗一夫

ペントミノ

複数の正方形を辺でつなげた多角形、また、それを長方形など指定の形に隙間なく並べるパズルをポリオミノ(polyomino)と呼び、米国の数学／工学者のソロモン・ゴロムが1953年に考案した。

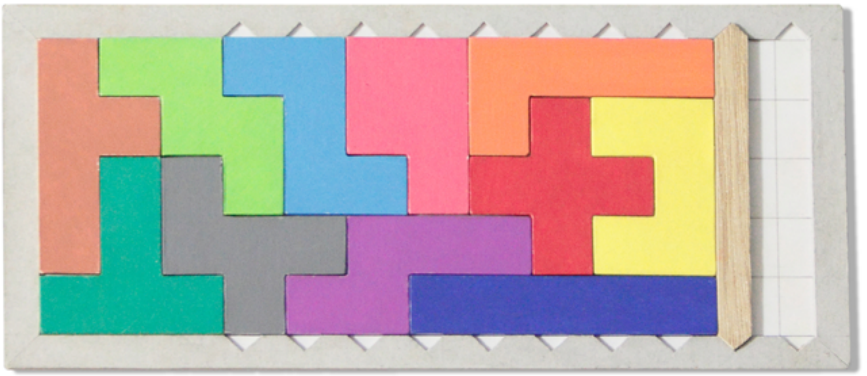
ペントミノは5つの正方形を辺に沿ってつなげたポリオミノの１種で、回転・鏡像によって同じになるものを同一と考えると12種類ある。

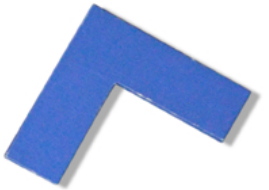


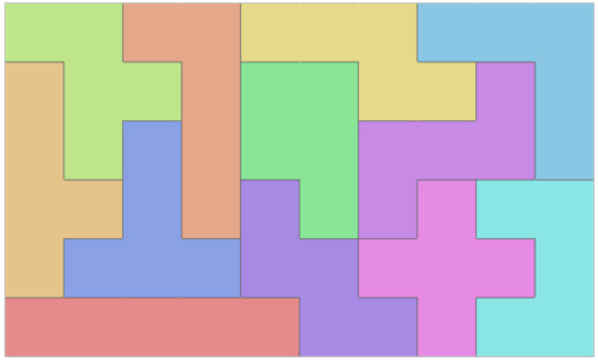
ペントミノは多くのメーカーからパズル・知育玩具として発売されていて、代表的な製品にテンヨーのプラパズルがある。最も一般的なペントミノの問題は、12片すべて使用して長方形を作ることであり、

6×10 には 2,339通り
5×12 には 1,010通り
4×15 には 368通り
3×20 には 2通り

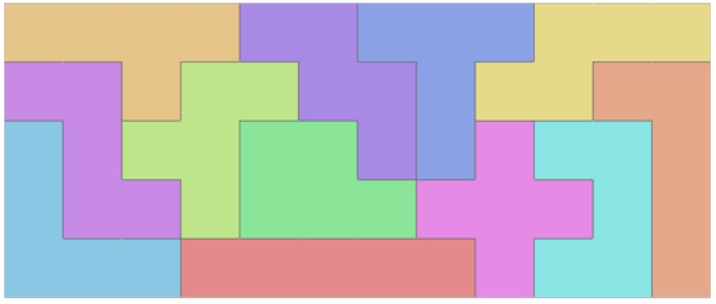
の解がある



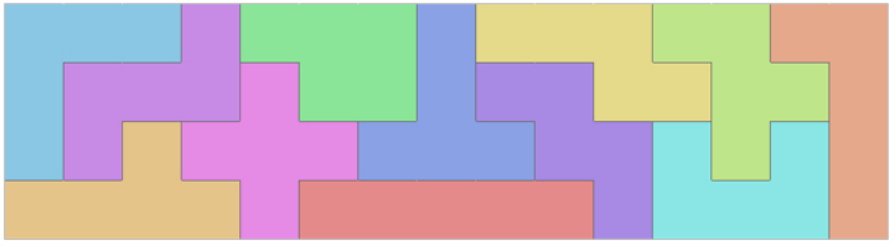




6×10(2339通り)



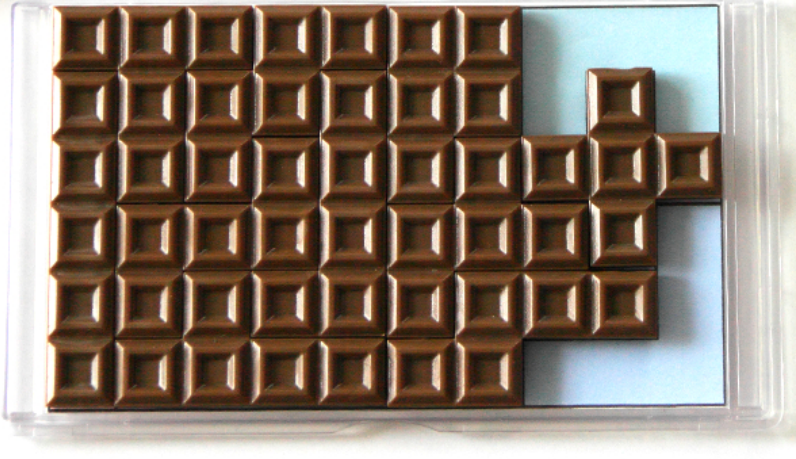
5×12(1010通り)

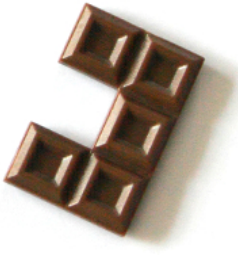


4×15(368通り)

3×20(2通り)


明治ミルクチョコレートパズル
meiji
milkchocolate
ご注意!
食べられません。
難易度：ピュア(甘め)





立体ペントミノ

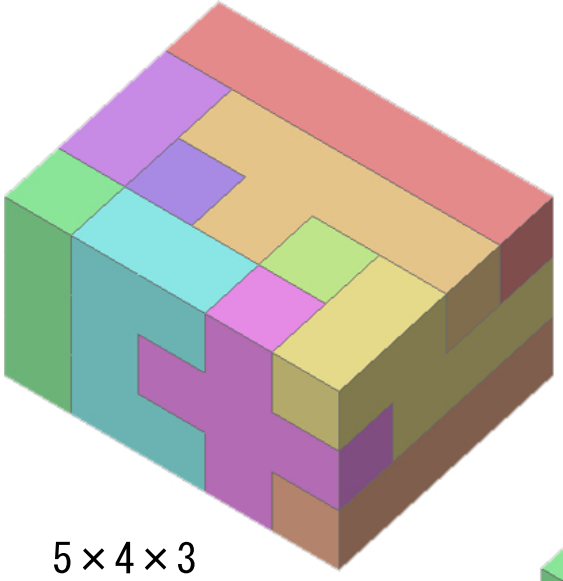
ペントミノの各ピースの厚さを1辺の長さと同じにすると12片の立体ペントミノが得られ、これを 5×4×3 の直方体に組むパズルが、「ソリッドパズルFACOM」の名称でテンヨーから発売されていた。

この名称はその解の数を計算した富士通のFACOM 270 シリーズに由来し、富士通で国産コンピュータの開発に尽力した池田敏雄は、玩具メーカーの依頼で始めたペントミノに凝るあまり、テレビに出演してペントミノの紹介をするに至ったというエピソードもある。

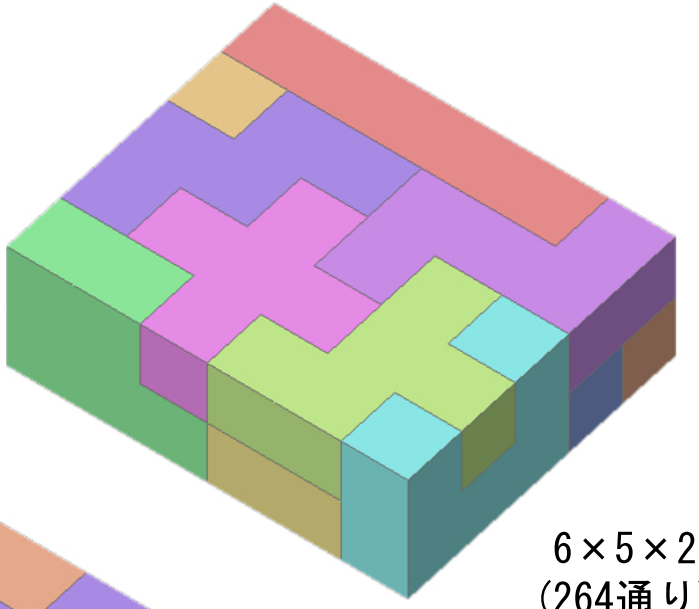
立体ペントミノは何種類かの直方体に組むことができて

5×4×3 で 3940 通り
6×5×2 で 264 通り
10×3×2 で 12 通り

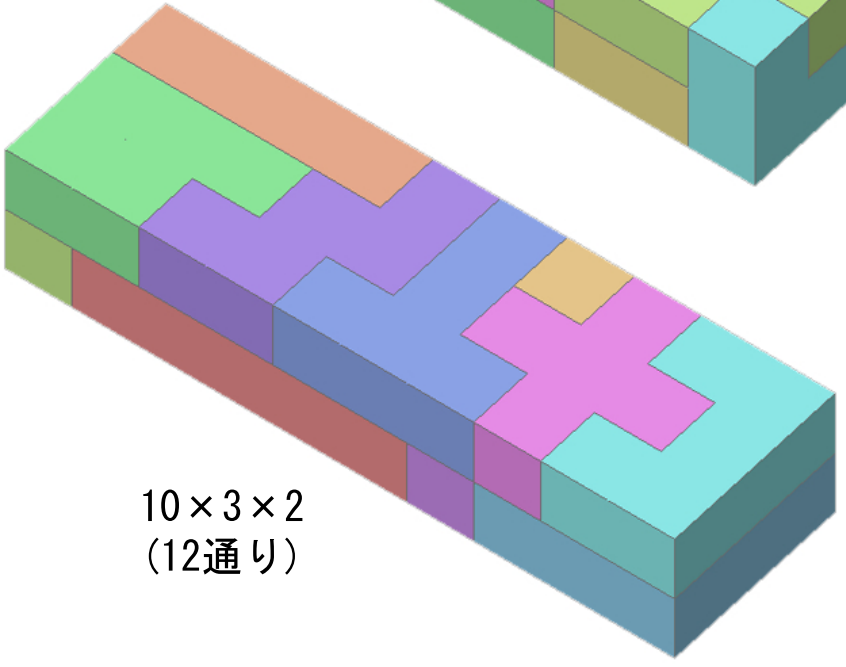
の解がある。



5×4×3
(3940通り)

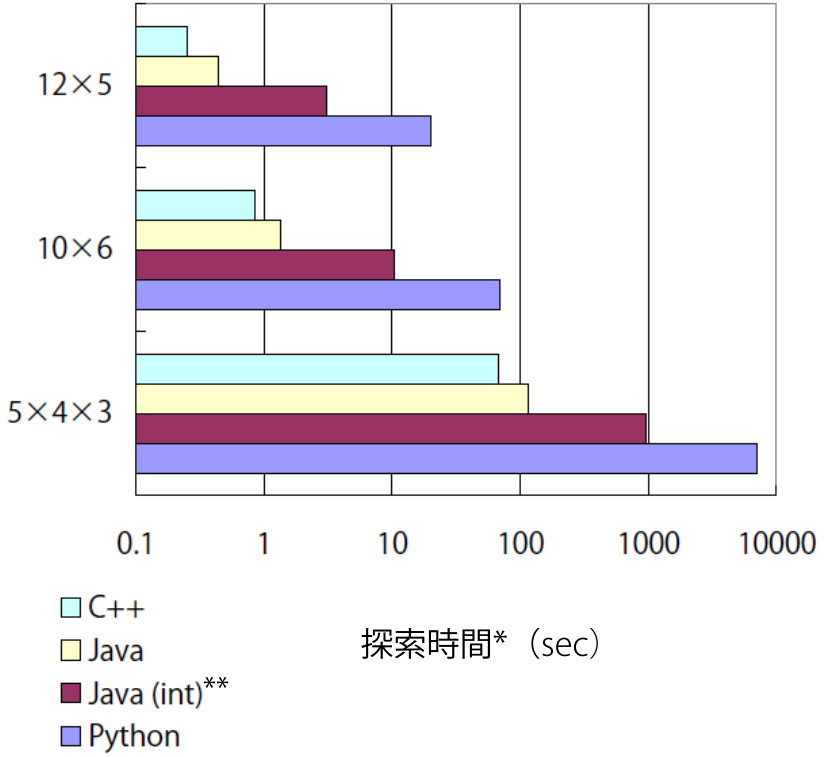


6×5×2
(264通り)

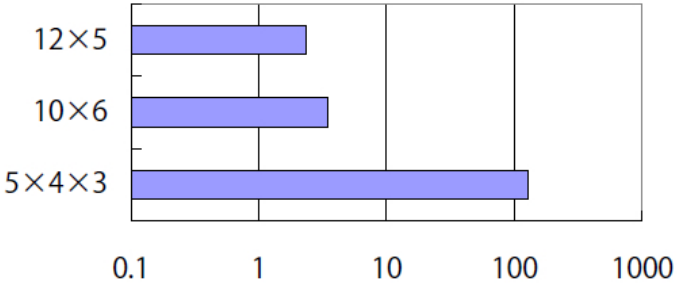


10×3×2
(12通り)

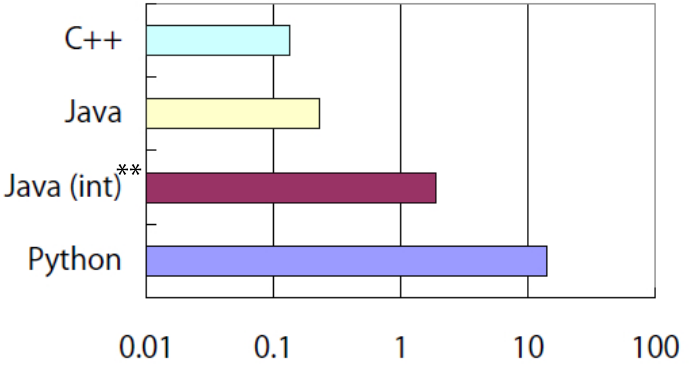
ペントミノの解探索



探索時間*（sec）



試行回数（$\times 10^3$）/ 解の総数



探索時間*（μsec）/ 解の総数

* Core i5-2500K (3.30GHz) 使用

** JITコンパイラを動作させずにインタプリタで実行

第１部

「プログラミングというお仕事」

マイクロプログラム

2200エミュレータ

指紋照合装置

C言語

マイクロプロセッサ

Intelプロセッサ搭載のACOS機

NAS装置の性能チューニング

第２部

「パソコンで遊ぼう」

はじめてのWindowsプログラム

オシロがほしい

プログラミング教育

Scratch & Python

ペントミノ

スライディングパズル

2023年6月2日　あすた会資料　　吉宗一夫

15パズル

15パズルはスライディングパズル（Sliding Puzzle）のひとつで、4×4のボード上の15個の駒を1駒分の空きを利用してスライドさせて目的の配置にする。

1878年、米国のパズル作家サム・ロイドが「15パズルで14と15 を入れ替えた状態を元に戻す」という解のない問題に、1,000ドルの賞金をかけて出題したことにより大きく普及したといわれている。

初期配置

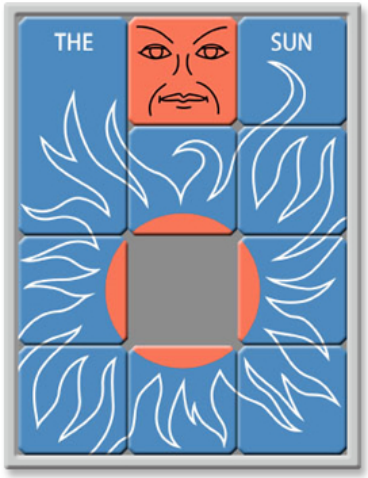
目的の配置

不可能な配置

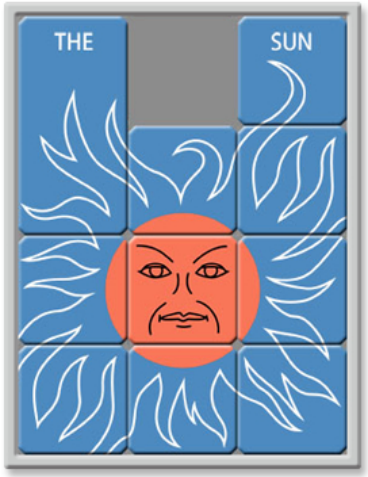
不可能な配置を数学的に分析すると

- 初期配置から目的の配置への並べ替えは15個の駒と空きの16の要素の置換である。
- 空きを利用した駒のスライドは、移動する駒と空きの互換である。
- すべての置換は互換の積として実現できるが、互換の数の偶奇は決まっている。
- 初期配置と目的の配置における空きの位置が同じになる置換は偶置換である。
- 14と15 の入れ替えは 1 つの置換、すなわち奇置換であるから、駒のスライドだけでは実現できない。

できるかな？



?



?

偶置換