情報科学の歴史（2）
―初期コンピュータのソフトウェア[1]

廣野　喜幸

情報科学の礎は、ウィーナー・フォン・ノイマン・チューリング・シャノンたちによって、1936－48 年頃に築かれた（廣野 2019）[1]。その後、情報科学は、緊密な一つのディシプリンを形成するというよりも、関連するいくつかの領域の連合体として展開していった。本稿ではそうした領域の一つであるソフトウェア関係の歴史を、その創成期に焦点を合わせて振り返っておく。ソフトウェア関係の研究が本格的に開始される前の状況が確認されることになるであろう。

1　序

ソフトウェアという言葉自体の起源についてまず一瞥しておく[2]。日本語のソフトウェアは英語の software に由来する。そして、今日と同じ意味内容をもつ software は 1950 年代、情報科学の創生期から 10 年ほどの後に、それ以前から存在していた hardware から生み出された。語素 ware は、ガラス製品（glassware）・鉄製品（ironware）・銀製品（silverware）といったように、素材を表す語素と結びついたとき、当該素材からできた製品を表す。また、食器類（tableware）・台所用品（kitchenware）・デルフト焼き（Delftware）・陶磁器（Chinaware）のように、場を示す語素と結びつき、そうした場に由来する製品を意味する言葉でもある。

英英辞典をひもとけば、hardware なる言葉は 15 世紀にすでに存在していたことが分かる。この場合、hard は素材の性質であり、具体的には固い金属類を指す。金属類は長持ちするので、software なる言葉は、柔らかい製品や長持ちしない製品を想起させるはずである。実際、software（正確には soft-ware）の初出が確認されるのは以下の 1850 年の文章なのだが、そうした意味で使用されている。

ヴィクトリア時代の娯楽雑誌ブームにのって、ディケンズ（Charles John Huffman Dickens 1812-70）編集の『家庭の言葉（Household Words）』が発刊され、広く読まれた。同誌創刊号に詩人ホーン（Richard Henry Horne 1802-84）が「ゴミ」なる作品を発表しており、ここに現れる soft-ware が、確認できる最古の例になる[3]。

> [ごみ収集の区分としてはさらに、]ソフトウェア（soft-ware）とハードウェア（hard-ware）という二つの部門がとても重要である。前者には、あらゆる野菜と動物類―腐るものすべて―が含まれる。Two other departments [among rubbish-tip pickers], called the“soft-ware” and the“hard-ware,”are very important. The former includes all vegetable and animal matters-everything that will decompose.（Horne 1850, 380）

現在私たちが使う意味がこうした用法に付け加えられる、あるいは置き換わるのは 1950 年代後半あたりであろう。もっとも古い例は、すぐれた統計学者テューキー（John Wilder Tukey 1915-2000）の 1958 年の文章中に見つけることができる。

> 今日、現代の電子計算機にとって、注意深く計画された解釈ルーチン、コンパイラ、そして、自動プログラミングの他の側面から成る「ソフトウェア」は、真空管・トランジスタ・回線・テープ類から成る「ハードウェア」と、少なくとも同じくらい重要である。Today the "software" comprising the carefully planned interpretive routines, compilers, and other aspects of automative programming are at least as important to the modern electronic calculator as its "hardware" of tubes, transistors, wires, tapes and the like.　（Tukey 1958, 2）

ソフトウェア研究もおそらくこの頃、つまり 1950 年代に一つの領域として形成されていったのであろう。そうした進展がある程度見られたあと、言葉が広く社会に浸透していったように思われる。『ニューヨーク・タイムズ』紙では、1961 年 12 月 7 日号にこうした用例が見られる。朝日新聞のデータベースを検索すると、1969 年に「ソフトウェア（利用技術）」という形で初めて現れる。但し書きがあることからすると、この時点では、まだ人口に膾炙する用語にはなっていないと見てよいであろう。利用技術という注釈は 1991 年まで続く。日本におけるソフトウェア研究の自立化あるいは市民権の獲得は、英米より 10 年ほど遅かったのではないだろうか。

2　初期のコンピュータたち

情報科学の基礎が築かれた 1936－48 年はまた、表 1 に示すように、初期のコンピュータ群が開発された時期である 1938－50 年前後に重なる。主としてアメリカ合衆国において開発がなされるが、イギリスやソビエト・スウェーデン・ドイツでも似たような試みが同時多発的になされていた。なお、表 1 には、暗号解読機コロッサスや連立方程式を解くための専用機であるアタナソフ－ベリー計算機 ABC(Atanasov-Berry Computer) など、今日言うコンピュータとはいささか異なる機械も、参考のため*印を付して記して

ある[4]。商用機には＃印を付した。また、表 2 には、情報科学の創始者たち（＊印）の生没年もあげた。初期コンピュータの開発者たちと同時期であることを見て取れよう。

これは、情報科学の基礎固めが終わってからおもむろにコンピュータの製作が進められたわけではないことを意味する。「機械的作業は機械にやらせるべき」という動機は潜在的に常に存在し、ことある毎に噴出し、各人が各様に努力し、実現を図ってきたとみるべきであろう。

表 1　初期のコンピュータ

名称	製作時期	主な関係者	備考
19 世紀			
* ジャカード織機	1801-	ジャカール	
* 階差機関	1830 頃	バベッジ	未完成
解析機関	1870 頃	バベッジ	未製作
* 第二階差機関	1847-9	バベッジ	
20 世紀			
Z1	1938	ツーゼ	
Z2	1939/1940	ツーゼ	
Z3	1941	ツーゼ	
Z4	1942	ツーゼ	
* ABC	1942	アタナソフ、ベリー	未完成
* コロッサス	1943.12	ニューマン、フラワーズ	
Harvard Mark I	1944.8	エイケン	
ENIAC	1946.2	ゴールドスタイン、モークリー、ジョン・エッカート、フォン・ノイマン	
Harvard Mark II	1947/48	エイケン	
SSEC	1948.1	ウォーレス・エッカート	
baby(SSEM)	1948.6	ウィリアムズ、キルバーン	
Manchester Mark I	1949.4	ウィリアム、キルバーン	
Harvard Mark III	1949.9	エイケン	
EDSAC	1949.5	ウィルクス、ホイーラー	
CSIRAC	1949.11	ピアシー	
MADDIDA	1949	スティール	
BARK	1950.2	パーム	
SWAC	1950	ハスキー	

SEAC	1950	アレクサンダー
Ferranti Mark 1	1951.2	ウィリアム、キルバーン
ORDVAC	1951	
＃UNIVAC I	1951	モークリー、ジョン・エッカート
Whirlwind	1951.4	フォレスター
MESM	1951.12	レベデフ
EDVAC	1951.12	ゴールドスタイン、モークリー、エッカート、フォン・ノイマン
ILLIAC I	1952	
Harwell	1952.5	
Harvard Mark VI	1952	エイケン
BESK	1953	パーム、コメット
RAYDAC	1953	
＃IBM 650	1954	

大駒（2005）等をもとに1955年までに完成したコンピュータをあげてある。網羅しているわけではない。なお、フォン・ノイマンによってプログラム内蔵方式が明記された、EDVACに関する文章の日付は、1945年6月30日である。プログラム内蔵方式が最初に実現されたのはbabyであった。babyは実験機であったが、プログラム内蔵方式の実用機で最初のものはEDSACになる。

表2　初期のコンピュータの開発者たち

名前	生没年	国
＊ウィーナー（Norbert Wiener）	1894-1964	米
ニューマン（Maxell Herman Alexander Newman）	1897-1984	英
エイケン（Howard Hathway Aiken）	1900-73	米
ウォーレス・エッカート（Wallace John Eckert）	1902-71	米
レベデフ（Сергей Алексеевич Лебедев）	1902-74	ソ
＊フォン・ノイマン（John von Neumann）	1903-57	ハンガリー→米
アタナソフ（John Vincent Atanasoff）	1903-95	米
フラワーズ（Tommy Flowers）	1905-98	英
パーク（Conrad "Conny" Palm）	1907-51	瑞
モークリー（John　William Mauchly）	1907-80	米
アレクサンダー（Samuel Nathan Alexander）	1910-67	米
ツーゼ（Konrad Zuse）	1910-95	独
ウィリアムズ（Frederick Callonal Williams）	1911-72	米

＊チューリング（Alan Mathison Turing）	1912-54	英
ゴールドスタイン（Hermann He Goldstein）	1913-2004	米
ウィルクス（Maurice Vincent Wilkes）	1913-2010	英
＊シャノン（Claude Elwood Shannon）	1916-2001	米
ハスキー（Harry Douglas Huskey）	1916-2017	米
ベリー（Clifford Edward Berry）	1918-63	米
スティール（Floyd George Steele）	1918-95	米
フォレスター（Jay Wright Forrester）	1918-2016	米
ジョン・エッカート（John Presper Eckert Jr.）	1919-95	米
ピアシー（Trevor Pearcey）	1919-98	英→豪
キルバーン（Tom Kilburn）	1921-2001	米

たとえば、19 世紀は、航海時に使う数式の計算を簡便ならしめるためなどに、数表が多く作られ、使われていた。これはまさに computer=計算手という職種を占める人々が、こつこつ計算をした結果をまとめたものである。しかし、ヒューマンエラーに起因する計算ミス・転記ミス等々により、そうした数表の信頼性は低かった。「数表の危機」と言われるこうした事態に対して、コンピュータ開発の思想的先駆者バベッジ（Charles Babbage 1791-1871）は階差機関や解析機関を構想することによって、乗り越えようとした。また、近代的コンピュータ作製の先陣を切ったのはドイツ人のツーゼだが、航空機会社に就職した彼は、設計のため 30 元連立方程式を解かなければならなくなった。これは熟練した計算手数名で 1 週間かかる計算であり、ツーゼがコンピュータ開発に取り組んだのは、なんとか機械化し、もっと早く結果を出せないかと考えたためである。アタナソフは博士論文でヘリウムの電子軌道構造を解明するとき、エイケンは博士論文で非線形微分方程式を解く際、モークリーは分子エネルギーや気象学における計算を試みる中で、ゴールドスタインは弾道計算表を一刻も早く得ようとして、皆膨大な計算に悩まされたことが、計算機に手を出す動機の理由の一班をなした。

ちなみに、初期コンピュータの開発者のうち他の人々はコンピュータに対する関心そのものが動機となっていたようだ。ウィリアムとキルバーンは、ウィリアムーキルバーン管（ある種のブラウン管）がメモリとして有効かどうかを試すために baby を作製している。ベリーはアタナソフの助手である。エッカートは、モークリーの計画に共鳴して参加した。ウィルクスはケンブリッジ大学数学研究所副所長、その後進である計算機研究所長を歴任し、コンピュータ開発が「任務」であった。

初期の情報科学においては、このようにとにもかくにも「コンピュータ」を作製してみて、他の文脈で明らかになった知識を取り込む、あるいはその知識そのものを生み出すという方式に依っていたように思われる。情報科学はその後も、そうしたパターンを踏襲し

てきたと言えるであろう。これは、必要な知見をまず得ようとする、自然科学の伝統的な様式とはいささか異なるであろう。自然科学は――神がつくったと見るにせよ、自然の理によって生じてきたと見るにせよ――、自然を対象に、その規則性を明らかにする試みである。しかし、情報科学はそうした意味での自然科学ではないのではあるまいか。試行錯誤の上で作製してみせた人工物について、それを統べる規則性を探究してきたのが情報科学であろう。情報科学は、「人工物科学」なのである。

研究途上のものはさておき、今日私たちが普段使っているコンピュータはすべて同一タイプであり、フォン・ノイマン型と言われる[5]。1945 年にフォン・ノイマンがとりまとめた文章（Neumann 1945）に、現在のコンピュータの基本設計が記されており、それに基づくからである。プログラム内蔵方式と命令を一つ処理しては次の命令を実行する逐次処理が、フォン・ノイマン型のポイントだとされることが多いが、具体的な特徴付けは論者によって分かれる。

すでに廣野（2019）でも言及しておいたが、煩瑣を厭わず、黒川利明による特徴付けをここでも掲げておくことにしよう。

> （1）人間は入力として、一組のデータを与える。（2）機械がその動力を利用して、人間の与えた操作の種類とその順序（プログラム）にしたがって、一連のデータ処理操作を定められた方式で実行する。（3）機械がデータ処理結果を表示する。（4）プログラムは機械の内部に、他の記憶と同じ形で蓄えられる。（黒川 1992: 30-31）

特徴（4）は、「プログラム内蔵方式」と呼ばれる[6]。この特徴（4）を除けば、データを入力し、それを処理して、所期のデータを得ることが基本構造になる。どんなに複雑巨大となっても、コンピュータとは畢竟、データをインプットし、それプロセッシングし、新たなデータを得るシステムにすぎない。

この処理＝プロセッシングの過程において、人の関与を最初の処理手順の指令のみに限るのが現代型コンピュータの基本的特性である。しかし、発展途上の初期コンピュータにおいては、処理中に人が関与しなければならないものが多かった。たとえば、ABC マシンでは、計算結果を 2 進数でカードにパンチしたものを出力させ、それを手で取りだし、また 2 進数カード読み取り装置に読み込ませて次の計算をさせていた（大駒 2005, 76-77）。また、ENIAC は、プログラミング・ボードという回路盤を外付けし、ハードウェアを変えることによって、異なった処理を実行させていたのである（黒川 1992, 22）。

私たちは現在、エディタによってキーボードから高級プログラミング言語でソース・コードを作成し、コンパイラにかけて実行ファイルを生成させ、その実行ファイルを実行させることで、目的とするアウトプット・データを得ている。計算機科学領域のノーベル賞に相当するチューリング賞受賞には、各種のプログラミング言語の開発者が多く含まれ

ることにも示されているように[7]、ソフトウェア研究における重要な貢献の一つは、プログラミング言語の作製である。

ソフトウェア研究がプログラミング言語に集約されえるのは、プログラム内蔵方式のゆえである。ソフトウェア研究という領域が花開いたのは、プログラム内蔵方式のお蔭なのだ。たとえば、ENIAC のようにプログラミング・ボード方式だと、ボード自体の大きさが制約となり、長いプログラムに相当する配線など行えない。プログラム可能だとしても、高々ある程度の長さのプログラムを走らせることができるだけで、汎用機のありがたみは薄れてしまう。「この方式の導入こそが、電子計算機というものを単なる数値計算の道具から、真に無限の可能性を内蔵した汎用の情報処理機械に発展させた鍵だったのである。」（高橋 1983, 117-18）「ソフトウェアが……独立の（あるいは大きな）話題となったのも、プログラム内蔵方式による」（黒川 1992, 27）。

かくして、ソフトウェア研究史は、プログラム内蔵方式に収束していく過程を追う作業がその開始点となる。ソフトウェア研究の本格的発展が 1950 年代に始まるとしたら、初期コンピュータにおけるソフトウェアがその出発点になるであろう。以下、プログラム内蔵方式が確立される前後において、ソフトウェアはどのような制約のもとにあったのかを、いくつかのコンピュータに焦点をあわせ、隈取っていくことにしよう。

3　初期コンピュータにおけるプログラムのメディア

ツーゼは会社に計算機の製作を提案したが、却下された。そこで、退社し、自費で開発することにした。試作機 Z1、実験機 Z2 を経て、Z3 でやっと本格的な計算機が完成した。開発費用を浮かすためであろうか、データの入出力もプログラムも、使用済みの映画フィルムに穿孔したものが使われた。異なるプログラムを読み取らせれば、違う計算ができるプログラム可能な多能機であったことは特筆されるべきであろう。ただし、現在とは異なり、プログラムは外部に存在する。ソフトを交換し、また別のゲームを楽しむゲーム機に近い形態をとっていたと言えよう。Z3 は、内蔵方式ではなく、したがって非フォン・ノイマン型であった。また、電気機械式であり、その点でも電子式である今日のコンピュータとは異質である。

ABC マシンは電子式である点は現代のコンピュータと同等であり、それゆえ、コンピュータの祖型とみなされることもある。同機は、Z2 同様、実験機であり、連立方程式を解くプログラムのみの単機能機であった。入力にはカードシステムが、出力には自動車の距離計のようなものが採用されたらしい（大駒 2005）。

29 元の連立方程式があったとしよう。変数にかかる 29 個の係数と定数項で都合 30 個の数値をそれぞれの方程式はもつので、総計 30 x 29 = 870 個の数値が存在する。これをカードにパンチした上で入力すると、内部の機構によって 2 進数に変換され、今日のレジスタに相当する記憶装置に格納される。そして、ある変数に着目し、たとえば、方程式

（1）から方程式（2）を引くことを繰り返していくとついにその変数の係数はゼロになる。これで変数が 28 に減った方程式が一つ得られる。次に方程式（1）と方程式（3）で同様な作業を行うと、変数が 28 のまた別の方程式が得られる。これを繰り返すと、結局、28 元の連立方程式に帰着される。かくして、変数を一つずつ減らしていけば、ついには一元の方程式となり、簡単にその変数を求められる。そこから、他の変数も芋づる式に明らかになっていく。ABC マシンが行うのは、元の数を一つ減らすことであった。29 元の連立方程式における総計 870 個の数値を入力すると、それから得られるはずの 28 元連立方程式の 29 X 28 = 812 個の数値が出力として吐き出されるのである。そして、次にこの数値を今度は入力に用いて、28 x 27 = 756 個の数値を得るという過程を繰り返していき、連立方程式を解いていく。

出力をまた入力するのは人手を使う。今日では、870 個の数値と解法の手続き、すなわちプログラムを入力したら、その後は全自動化されており、解の 29 個の数値が一挙に得られる。ABC マシンの場合、そうではなく、人と協働しながら解が生み出されていく。今日の感覚からすると、プログラムされているとは到底言えない。これは、やはり非フォン・ノイマン型に分類される。なお、ABC マシンは入出力装置が安定して動作せず、真の意味で実験段階にあった実験機であった。

エイケン・グループの Harvard Mark I では、入力はカード、出力はカードと電動タイプライタ、プログラムは紙テープであった。入力はスイッチでも直接設定できるようになっていた。電子式数値積分機・計算機 ENIAC（Electronic Numerical Integrator and Computer）では、入力はカード読取装置、出力はカード穿孔装置、プログラムは先にも述べたように、プログラミング・ボートで配線やスイッチでなすような仕組みになっていた。1 週間かけて配線を変更するなど「プログラミング」を行い、1 時間程度で結果を得るといった案配であったという。

これ以降に述べるのは、すべてプログラム内蔵方式機である。プログラム内蔵方式では、データとプログラムの区別がなくなるので、入力は同じ形式でなされうる。小規模実験機 SSEM（Small-Scale Experimantal Maschine）、愛称 baby においては、番地とメモリの 2 進数パターンをスイッチで設定することで入力がなされる。1 語 1 語手作業で設定することによってなされた。真の実験機であった baby は、ブラウン管というメモリ上にプログラムを書き込むことができたが、スイッチで直接書き込むという、実用機からはたいそうかけ離れた方式であった。ブラウン管がメモリなのだから、計算結果はブラウン管に写しだされるメモリの内容を読み取ればよい。

ウィルクスたちの電子式遅延記録装置自動計算機 EDSAC（Electronic Delay Storage Automatic Calculator）では、磁気テープ読取装置で入力し、磁気テープ書込装置で出力する。

ENIAC は、早く結果を出せという圧力のもと、新技術の開発ではなく、既存技術で遮二無二まとめあげられた「ブリコラージュ」機器であった。それゆえ、「ソフトウェア」

は、あまりスマートとは言えない、プログラミング・ボートというハードウェア方式であったのだろう。ついでに述べておけば、次々と湧いてくるアイディアが「ブリコラージュ」機器という制限では実施できないため構想されたのが電子式離散変数計算機 EDVAC（Electronic Discrete Variable Automatic Computer）であり、その構想の過程でプログラム内蔵方式がもたらされ、その後のソフトウェアの発展が用意されることなる。その入力は紙テープ（5 孔）であり、出力はテレタイプ端末であった。EDVAC の設計思想は広く配布されたので、EDSAC グループに先を越されることになった。

表 3　初期のコンピュータにおける「ソフトウェア」のメディア

名称	データ入力	プログラム	データ出力
Z3	フィルム	同左	同左
* ABC	カード	——	カード
Harvard Mark I	カード	紙テープ	カード・電動タイプライタ
ENIAC	カード	プログラミング・ボート	カード
baby(SSEM)	スイッチ	スイッチ	ブラウン管
EDSAC	磁気テープ		磁気テープ
EDVAC	紙テープ（5 孔）		テレタイプ端末

まとめると表 3 のようになる。表 3 の「プログラム」の項については、注意が必要だ。表 3 では並べて記したが、プログラム内蔵方式誕生以前と以後では、その位置づけが異なる。Z3 や ABC マシン、Harvard Mark I において、カードなるメディアに具現化されたソフトウェアは、ソフトウェアそのものであり、それが直接演算の指示を与える。この指示に時間がかかる。つまり、遅い。一方、EDSAC・EDVAC の磁気テープや紙テープは、そこに具現化されたプログラムを入力することで、それに対応するソフトウェアがそれぞれに機器の記憶装置に具体化され、後者が指令を与えるのである。この方式だと速い。

裁判にまでなったから広く知られているだろうが、ENIAC グループは、けっして一枚岩の仲良しではなかった。モークリー＆エッカート対ゴールドスタイン＆フォン・ノイマンの反目の様相を呈していた。その背景には、モークリー＆エッカートは商業化を目論む工学徒であり、ゴールドスタイン＆フォン・ノイマンは軍事関連の数学者であったという相違があったであろう。

既に述べたように、現在の実用機はすべてフォン・ノイマン型だが、その名称は、EDVAC の構想を認めた文章（Neuman 1945）の執筆者がフォン・ノイマンであったことによる。モークリー＆エッカートが自分たちの名前を冠して呼ばれるのが妥当だと考えていたのに対し、フォン・ノイマンは ENIAC プロジェクトに遅れた参加しただけあって、自らフォン・ノイマン型と称したことはないが、EDVAC のアイディアは ENIAC チーム全体に

帰されるべきだと思っていたようである。

モークリー＆エッカートからすれば、自分たちの工学的アイディアを数学的に整備しただけと映った。EDVAC 構想の要は、それがチューリング完全な（＝万能チューリングマシンと同等な計算能力をもつ）コンピュータの作製にあることをフォン・ノイマンは十分認識していた、つまり、計算科学の基礎と常に関連づけながらコンピュータを考察していたが、モークリー＆エッカートはそれをきちんと理解していないとフォン・ノイマンは感じていたのかもしれない。そして、その肝心要を明晰にしたのがフォン・ノイマンの 1945 年の文章であり、それは大きな貢献だとみなしていたのだろう。1945 年の文章は単にさまざまのアイディアをとりまとめたというよりは、首尾一貫した一つの構想にしあげたのであって、それは大きな貢献だと思っていたようである。

今となっては、「フォン・ノイマン型コンピュータ」という構想全体に対して、誰がどれだけ寄与したかを明らかにすることは困難であろう[8]。しかし、ソフトウェアの発達史という、私たちの今の興味関心から重要なのはプログラム内蔵方式であり、その発端はエッカートによるらしいことが分かっている。プログラム内蔵方式こそ、その後のソフトウェア発展を大きく規定する前提条件となったのだが[9]、プログラム内蔵方式の動機は、その後のソフトウェア発展の可能性にあったのではなく、内蔵すれば演算の速度があがる点にあったと言えよう。

いささか記述が横道にそれた感があるが、まとめると、内蔵記憶方式が確立する前までは、紙テープ等に穿孔したものがソフトウェアの本体であり、確立後は記録装置内の電子的状態がソフトウェアの本体になる[10]。それゆえ、内蔵記憶容量がプログラムの長さを、そしてあり方を規定するようになる。最初の内蔵方式コンピュータである baby のプログラムを対象に――最初の本格的コンピュータである Z3 を参照しつつ――、そうした状況を具体的に見ていくことにしよう。

４　初期コンピュータのプログラム――Z3 と baby

1941 年に完成した Z3 は、使用済み 35mm 映画フィルムに穿孔したものをプログラムに用いたが、一列に 8 カ所穿孔しうる仕様、つまり 1 語（＝1 命令）8 ビットであった。Z3 の命令セットは 9 個からなり（表 4）、命令は 3 種（入出力・貯蔵・演算）に分類される。2 ビットを演算種類の識別、6 ビットを入出力の識別・貯蔵部のアドレス指定・演算の種類（加減乗除および平方根）に用いた。入力型の命令が読み込まれた場合には、Z3 は一時停止し、人がキーボードから引数を入力するのを待つ。出力のときも一時停止した。そうでないと、人はディスプレイランプを読みきれない（Rojas 2002, 243）。本機は、ドイツ航空機研究所で翼のフラッター現象を統計的に解析するために使用された。（Z3 のプラグラムの詳しい検討は次稿に委ねる）

表４ Z3 の命令セット

命令タイプ	命令コード	内容	オペコード
[01] I/O	Lu	キーボード読み込み	01 110000
[02] I/O	Ld	ディスプレイ出力	01 111000
[03] 貯蔵	Pr z	アドレス z をロード	10 $z_6z_5z_4z_3z_2z_1$
[04] 貯蔵	Ps z	アドレス z にストア	10 $z_6z_5z_4z_3z_2z_1$
[05] 演算	Lm	乗算	01 001000
[06] 演算	Li	除算	01 010000
[07] 演算	Lw	平方根	01 011000
[08] 演算	Ls1	加算	01 100000
[09] 演算	Ls2	減算	01 101000

最初の内蔵方式は、Z3 のように長いプログラムは書けなかった。初めての内蔵方式である baby の記憶容量は 32 語で、1 語 32 ビット、総計 1024 ビットである。つまり、32 行のプログラムが書けるにすぎなかった。機器自体はけっこう大きい SSEM が baby なる愛称をもつ所以である。命令は 7 個からなり、命令数は Z3 と大差はない（表 5）。命令部は 3 ビット=2^3=8 で表せる。メモリは 32 語だから、アドレスを指定するためには 5 ビット=2^5=32 で済む。1 語 32 ビット用意されていたが、実際は 8 ビットで記述を済ませたということになる。

表 5　baby の命令セット

オペコード	命令コード	内容
[01] 000	JMP S	絶対無条件：ジャンプ指定されたメモリアドレスの命令にジャンプする
[02] 100	JRP S	相対無条件ジャンプ：指定されたメモリアドレスにアキュムレータの値を加算したアドレスの命令にジャンプする
[03] 010	LDN S	指定されたメモリアドレスにある数を取り出し、符号を反転させ、アキュムレータに格納する。
[04] 110	STO S	アキュムレータにある数を指定されたメモリアドレスに格納する。
[05] 001 または 101[t 1]	SUB S	指定されたメモリアドレスにある数をアキュムレータにある数から引き、減算結果をアキュムレータに格納する。
[06] 011	CMP	アキュムレータにある値が負であれば

		次の命令をスキップする（実行しない）
[07] 111	STP	停止する。

最初にプログラムを走らせたのは 1948 年 6 月 21 日であり、2^{18} ＝262,144 の最大の真の約数を求めさせた。2^{17}＝131,072 が解答であることはすぐに分かるであろうが、このプログラムはキルバーンが作成した。27 語からなり、19 語に命令が、8 語にデータが置かれている（表６）。約数であるかどうかは、減算を繰り返して、0 になるか否かで判定する。当該の数から、1 少ない数、この場合は 262,143 を引けば、2 回で負になるので異なる。さらに 1 少ない数 262,142 を引いても同様である。一つずつ減らした数で試み、ついに 131,072 になったとき、2 回の減算で 0 になるので、これが最大の約数と判定される次第である。

表 6　プログラム内蔵方式で最初に実行されたプログラム

行	コード	命令	トゥーティルによる注釈
1	00011 010	-24 to C	load -b1
2	01011 110	c to 26	store -b1
3	01011 010	-26 to C	load +b1
4	11011 110	c to 27	store +b1
5	11101 010	-23 to C	load a
6	11011 001	Sub 27	Trial subtraction
7	----- 011	Test	is difference negative?
8	00101 100	Add 20, Cl	still positive, Jump back two lines
9	01011 001	Sub 26	overshot, so add back bn
10	10011 110	c to 25	store +r n
11	10011 010	-25 to C	load -rn
12	----- 011	Test	is remainder zero?
13	111	Stop	yes
14	01011 010	-26 to C	no; load +bn
15	10101 001	Sub　21	form b(n+1) = bn - 1
16	11011 110	c to 27	store b(n+1)
17	11011 010	-27 to C	load -b(n+1)
18	01011 110	c to 26	store it
19	011011 000	22 to Cl	Jump to line 5
20	10111 etc [or 10100]	-3	
21	10000	1	

22	00100			4
23	-a			
24	b1			
		init	final	
25	-	rN(=0)	rN(=0)	
26	-	-bN	a-bN	
27	-	bN	bN	

コードに記された数値を順次メモリに置いていく。コードの最初の 5 文字は第 0〜4 ビットを使い、アドレスを指定する。次の 3 文字は第 13〜15 ビットを使用し、命令を表す。baby においては右から数値を読むので、第 1 行の「00011」は 2 進数「11000」であり、これは 10 進法だと 16+8=24 だから、24 番アドレスを意味する。命令表によれば、これは「指定されたメモリアドレスにある数を取り出し、符号を反転させ、アキュムレータに格納する」のだから、b1 の符号を反転し、アキュムレータに格納する動作が行われる。こうした命令が 350 万回ほど実施され、52 分で正答を得たという。

次に実行されたのは 1948 年 7 月 18 日で、やはり最大の約数を求めるプログラムだったが、トゥーティル（Geoff Tootill 1922-2017）によって改良されたものである。同じく 7 月に、チューリングが長除法を使う 3 つめのプログラムを送ってきた。

次稿では、内蔵方式によるこれら 3 つの最初のプログラムの位置づけ、すなわち、Z3 のそれとどう違うのか、また、baby に続く諸コンピュータのプログラムにどう影響を与えたのかなどについて探っていくことにしよう。

文献

小山俊士（2005）「フォン・ノイマン『EDVAC 草稿』について」『科学史・科学哲学』19: 106-122。

黒川利明（1992）『ソフトウェアの話』岩波新書

廣野喜幸（2019）「情報科学の歴史（1）—情報科学の成立　1936–48」『国際哲学研究』別冊 12: 7-30。

Horne, R. H. (1850) Dust; or Ugliness Redeemed, *Household Words* 1(16)[13 July]: 379-384.

Neumann, John von (1945) *First Draft of a Report on the EDVAC by Contract No. W-670-ORD-4926 Between the United States Army Ordnance Department and the University of Pennsylvania*

小野厚夫（2016）『情報ということば－その来歴と意味内容』冨山房インターナショナル。

大駒誠一（2005）『コンピュータ開発史－歴史の誤りをただす「最初の計算機」をたずねる旅―』共立出版

Rojas, Rául. 2002. The architecture of Konrad Zuse's early computing machines. In Rául Rojas and Ulf Hashagen (eds.) *The first computers: history and architectures*, eds. pp.237-261. Cambridge, Massachusetts: MIT press.

Shapiro, Fred R. (2000) Origin of the Term of Software: Evidence from the JSTOR Electric Archive, *IEEE Annals of the History of Computing* 22(2): 69-71

高橋秀俊（1983）『岩波情報科学 1　情報科学の歩み』岩波書店

玉井哲雄（2012）『ソフトウェア社会のゆくえ』岩波書店

田中英彦（1989）『非フォン・ノイマン型コンピュータ』電子情報通信学会

Tukey, J. W. (1958) The Teaching of Concrete Mathematics, *American Mathematical Monthly* 65: 1-9.

註

1　廣野（2019）には、以下の一段落が含まれていた。「今聞けば至極当たり前のこのモデルは、しかし、100 年前は当然ではなかったことを私たちは思い返す必要があるだろう。たとえば、ここであっさり情報なる言葉と概念を用いているが、こうした情報概念の現代的理解が広く共有されるは、せいぜい 1960～70 年あたりであって、それを遡ることはない。それ以前は、情報といえば、軍事機密等を典型例とする、今日言う極秘情報のことであった（小野 2016)。ある知識を知らせたくない側があり、知りたい側があった場合にのみ、情報という言葉・概念が使用された。現在では、知らせたくないアクターという契機を欠いたとしても、情報という言葉・概念は使われる。」この引用の仕方に対し、小野氏から抗議と当該箇所の削除の要請があった。確かに、この引用の仕方だと、（1）「情報」なる言葉の意味内実が、軍事機密から今日の一般的な用法に変化したこと、（2）その転換が 1960～70 年代に起こったことの論拠がともに小野（2016）に記されているように受け止められても仕方がない。しかし、小野（2016）は、第二次大戦中にはすでに一般的用法がなされていたことを証明しており、上記（2）とは明確に異なる主張がなされている。私としては（1）の典拠としてのみ小野（2016）を引いたつもりであったが、誤解を招きやすい引用の仕方であった。当該箇

所を削除するとともに、小野氏および読者のみなさまにお詫び申し上げる。

2 この辺りの記述は全面的に玉井（2012）に依拠している。

3 玉井（2012）は、「ゴミ」をディケンズが記したと述べるが（p.6)、どこか誤解があるのだろう。

4 こうした初期コンピュータのいくつかについては、廣野（2019）で触れておいた。

5 主たる非フォン・ノイマン型コンピュータとしては、量子コンピュータや神経回路を模したパーセプトロンがあげられるであろう。なお、フォン・ノイマン型の内実には揺らぎがある。「プログラム内蔵方式」と等値する用法もあれば、プログラム内蔵方式であっても、演算部と記憶部を物理的に分離するハーバード・アーキテクチャー（Harvard Mark I 等）とは異なり、同じ記憶装置に両者を混在させる様式をフォン・ノイマン型と呼ぶこともある。

6 これらの特徴の他に、逐次実行方式を指摘する者も多い。フォン・ノイマン型では、記憶部に蓄えられた実行命令をいちいち読み出しにいかねばならず、これが実行速度を短縮する際の阻害要因となっている。いわゆる、フォン・ノイマン・ボトルネックである。フォン・ノイマン・ボトルネックにどう対処するかを探求する流れが存在し、逐次実行方式とは異なるアプローチも非フォン・ノイマン型と呼称される。いくつかの例は田中（1989）を参照。

7 チューリング賞は 1966 年に創設された、計算機科学のノーベル賞とも言われる斯界の最高栄誉だが、コンピュータ言語の開発者が多く受賞している。Fortran（1954 年）のバッカス（John Warner Backus 1924-2007）は 1977 年に、algol（1958 年）のパリス（Alab Jay Perlis 1922-90）は 1966 年に、LISP（1958 年）のマッカーシー（John Mccarthy 1927-2011）は 1971 年に、Pascal（1970 年）のヴィルト（Niklaus Emil Wirth 1934 生）は 1984 年に、c 言語（1972 年）のリッチー（Dennis MacAlistair Ritchie 1941-2011）は 1983 年に、栄誉に輝いた。

8 歴史的には、Neumann（1945）によってフォン・ノイマン型なる名称ができたのだが、Neumann（1945）とフォン・ノイマン型コンピュータのあいだには懸隔があることが小山（2005）によって明らかにされている。

9 プログラムとデータが同じ地位をもつがゆえに、プログラムを書き換えるコンピュータ・ウィルスが内蔵方式でははびこることになった。ENIAC のように配線であったら、外部から遠隔操作で配線をつなぎ替えることなど、できなかったであろう。

10 とはいえ、その状況を実現するためには、相変わらず、紙テープ等が使われることになる。現在のように、キーボードで入力し、画面に出力されるシステムの嚆矢は、Whirlwind あたりであろうと言われている。